reledmac
Typeset scholarly editions with LaTeX*

Maïeul Rouquette†

based on the original ledmac by

Peter Wilson

Herries Press

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

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

Abstract

The reledmac provides many tools in order to typeset scholarly editions. It is based on the eledmac package, which was based on the ledmac package, which was based on the edmac TeX package.

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

reledmac provides many tools and options. Normally, they are all documented in this file. Also provided is a help folder, “examples”. The folder contains additional examples (although not for every possible case). Examples starting with “1-” are for basic uses, those starting with “2-” are for advanced uses.

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

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

Contents

1 Introduction **11**

1.1 Aim of the package 11

1.2 History 12

1.2.1 edmac 12

1.2.2 ledmac 13

1.2.3 eledmac 14

[*] This file (reledmac.dtx) has version number v2.22.0, last revised 2017/06/08.

[†] maieul at maieul dot net

1.2.4 `reledmac` . . . 14
1.3 Bibliography . . . 14

2 How the package works — the problem of the number of LaTeX runs 14

3 Compatibility warning 15

4 Options 15
4.1 Specific features . . . 15
4.2 Optimizing package performance . . . 16

5 Text lines and paragraphs numbering 16
5.1 Text lines numbering . . . 16
5.2 Paragraphs . . . 17
5.2.1 Basics . . . 17
5.2.2 Automatically producing `\pstart` ... `\pend` . . . 18
5.2.3 Content before specific `\pstart` and after specific `\pend` . . . 18
5.2.4 Content before every `\pstart` and after every `\pend` . . . 19
5.2.5 Numbering paragraphs (`\pstart`) . . . 19
5.2.6 Languages written in Right to Left . . . 20
5.2.7 Memory limits . . . 20
5.3 Lineation commands . . . 20
5.3.1 Disabling lineation . . . 20
5.3.2 Setting lineation start and step . . . 21
5.3.3 Setting lineation reset . . . 21
5.3.4 Setting line number margin . . . 21
5.3.5 Other settings . . . 22
5.4 Changing the line numbers . . . 22
5.4.1 Sublineation . . . 22
5.4.2 Locking lineation . . . 22
5.4.3 Setting and changing line number . . . 22
5.4.4 Line number style . . . 23
5.4.5 Skipping and hidding number . . . 23
5.5 Executing code at each line . . . 24

6 Apparatus commands 24
6.1 Terminology . . . 24
6.2 Critical notes . . . 24
6.2.1 The lemma . . . 24
6.2.2 Footnotes . . . 25
6.2.3 Endnotes . . . 25
6.2.4 Paragraph in critical apparatus . . . 27
6.2.5 Change lemma and line number . . . 27
6.2.6 Changing the names of commands for critical apparatus . . . 28
6.3 Disambiguation of identical words in the apparatus . . . 28
6.3.1 Basic use . . . 28

6.3.2 Notes about input encoding with UTF-8 processor 29
6.3.3 Use with `\lemma` command 29
6.3.4 Sameword for a group of words 31
6.3.5 Customizing 32
6.4 Apparatus of Manuscripts 32
6.4.1 Marking sections of text 33
6.4.2 Layout of the apparatus of manuscripts 33
6.4.3 Settings 34
6.5 Familiar notes 34
6.5.1 Basic use 34
6.5.2 Customizing mark 34
6.5.3 Separator for multiple footnotes 34
6.6 Printing the footnote mark without printing the footnote text 35
6.7 Changing series 35
6.7.1 Create a new series 35
6.7.2 Delete series 35
6.7.3 Series order 35
6.8 Position of critical and familiar footnotes 36

7 Critical apparatus appearance 36
7.1 Notes arrangement in a series 37
7.2 Control line number printing 38
7.2.1 Print line number only at first time 38
7.2.2 Print page number only at first time 38
7.2.3 Arbitrary text before line number 39
7.2.4 Separator for line range 39
7.2.5 Abbreviate line range 39
7.2.6 Disable line number 40
7.2.7 Printing pstart number 40
7.2.8 Printing stanza number 40
7.2.9 Separator between line and subline numbers 40
7.2.10 Separator between page and line numbers 41
7.2.11 Space around number 41
7.2.12 Space around line symbol 41
7.2.13 Space in place of number 41
7.2.14 Boxing line number and line symbol 41
7.3 For endnotes 42
7.4 Arbitrary code around line number 42
7.5 Separator between the lemma and the note 43
7.5.1 For footnotes 43
7.5.2 For endnotes 43
7.6 Font style 44
7.6.1 For line number 44
7.6.2 For the lemma 44
7.6.3 For all notes 44
7.7 Wrapping notes 44

7.7.1 Wrapping lemmas . . . 44
7.7.2 Wrapping contents . . . 45
7.8 Indent of notes content . . . 45
7.9 Arbitrary code at the beginning of notes . . . 45
7.10 Arbitrary code before inserting note . . . 45
7.11 Options for footnotes in columns . . . 46
7.11.1 Alignment . . . 46
7.11.2 Size of the columns . . . 46
7.12 Options for paragraphed footnotes and notes grouped by line . . . 46
7.12.1 Mark separation of notes . . . 46
7.12.2 Ragged text . . . 47
7.13 Options for block of notes . . . 47
7.13.1 Grouping notes by line . . . 47
7.13.2 Text before notes . . . 47
7.13.3 Code before notes . . . 47
7.13.4 Spacing . . . 48
7.13.5 Rule . . . 48
7.13.6 Maximum height . . . 48
7.13.7 Width . . . 49
7.14 Footnotes and the `reledpar` columns . . . 49
7.15 Endnotes in one paragraph . . . 49

8 Fonts 49

9 Verse 50
9.1 Basic . . . 50
9.2 Define stanza indents . . . 50
9.3 Repeating stanza indents . . . 51
9.4 Manual stanza indent . . . 51
9.5 Stanza breaking . . . 52
9.6 Hanging symbol . . . 52
9.7 Long verse and page break . . . 52
9.8 Content before/after verses . . . 52
9.9 Numbering stanza . . . 53
9.10 Various tools . . . 54
9.11 Notes on empty lines . . . 54

10 Grouping 54

11 Cross referencing 55
11.1 Basic use . . . 55
11.2 Cross-referencing to a critical note . . . 56
11.3 Cross-referencing which return a number in any case . . . 56
11.3.1 Cross-referencing in order to define line number of a critical note . 56
11.4 Not automatic cross-referencing . . . 57
11.5 Normal LaTeX cross-referencing . . . 57

11.6 References to start and end lines 57
11.6.1 Reference to main text lines 57
11.6.2 References to lines that are commented on in the apparatus 57
11.6.3 Settings 58
11.7 Compatibility with `xr` package 59

12 Side notes 60
12.1 Basics 60
12.2 Setting 60
12.2.1 Width 60
12.2.2 Vertical position 60
12.2.3 Distance to the main text 60
12.2.4 Font 61
12.2.5 Separator between notes 61

13 Indexing 61
13.1 Basics 61
13.2 Referring to critical notes 61
13.3 Separator between page and line numbers 62
13.4 Using xindy 62
13.5 Advanced setting 63

14 Glossary 63
14.1 Preamble setting 63
14.2 Commands 63

15 Tabular material 64

16 Sectioning commands 67
16.1 Sectioning commands without line numbers or critical notes 67
16.2 Sectioning commands with line numbering and critical notes 67
16.3 Optimization 68

17 Quotation environments 68

18 Page breaks 68
18.1 Control page breaking 68
18.2 Prevent page break in a long verses 69

19 Miscellaneous 69
19.1 Known and suspected limitations 70
19.1.1 Non-standard geometry 70
19.1.2 `floatrow` package compatibility 70
19.1.3 ‘No room for a new’ 70
19.1.4 Marginal notes 70
19.1.5 Paragraph shape 71
19.1.6 Paragraphed footnotes 71

19.1.7 Use with other packages 71
19.2 Parallel typesetting 72

I Implementation overview 73

II Preliminaries 73
II.1 Links with original `edmac` 73
II.2 Package declaration 73
II.3 Package options 74
II.4 Loading packages 76
II.5 Compatibility with LuaTEX 76
II.6 Boolean flags 76
II.7 Messages 77
II.8 Gobbling 84
II.9 Miscellaneous commands 84
II.10 Prepare `reledpar` 84
II.11 Booleans provided by other optional packages which are required in any case 85

III Sectioning commands 85

IV List macros 89

V Line counting 91
V.1 Choosing the system of lineation 91
V.2 Line number margin 92
V.3 Line number initialization and increment 93
V.4 Line number locking 95
V.5 Line number style 96
V.6 Line number printing 96
V.7 Line number counters and lists 97
V.8 Line number locking counter 98
V.9 Line number associated to lemma 98
V.10 Reading the line-list file 102
V.11 Commands within the line-list file 104
V.12 Writing to the line-list file 117

VI Marking text for notes 123
VI.1 `\edtext` itself 124
VI.2 Substitute lemma 131
VI.3 Substitute line numbers 132
VI.4 Lemma disambiguation 133

VII Paragraph decomposition and reassembly 139
VII.1 Boxes, counters, `\pstart` and `\pend` 139
VII.2 Processing one line 145
VII.2.1 General process 145
VII.2.2 Process for “normal” line 146
VII.2.3 Process for line containing `\eledsection` command 148
VII.2.4 Hooks 149
VII.2.5 Sidenotes and marginal line number initialization 149

VIII Line and page number computation 150

IX Line number printing 153

X Pstart number printing in side 157

XI Restoring footnotes and penalties 158
XI.1 Add insertions to the vertical list 158
XI.2 Penalties 160
XI.3 Printing leftover notes 161
XI.4 Text before notes 162

XII Critical footnotes 164
XII.1 Fonts 164
XII.2 Individual note options 164
XII.3 Notes language 165
XII.4 General survey of the way we manage notes 166
XII.5 General setup 166
XII.6 Footnotes arrangement 167
XII.6.1 User level macro 167
XII.6.2 Normal footnote 167
XII.6.3 Paragraphed footnotes 173
XII.6.4 Columnar footnotes 180
XII.7 Critical notes presentation 187
XII.7.1 Font tools 187
XII.7.2 Pstart number in footnote 187
XII.7.3 Lemma printing 188
XII.7.4 Line number printing 189
XII.7.5 Footnote grouped by line 197

XIII Familiar footnotes 199
XIII.1 Adjacent footnotes 199
XIII.2 Regular footnotes for numbered texts 200
XIII.3 Footnote formats 202
XIII.4 Footnote arrangement 203
XIII.4.1 User level macro 203
XIII.4.2 Normal footnotes 203
XIII.4.3 Two columns footnotes 209

XIII.4.4 Three columns footnotes 211
XIII.4.5 Paragraphed footnotes 213
XIII.5 Wrapping footnote marks in hyperlink 217

XIV Code common to both critical and familiar footnote in normal arrangement 218

XV Footnotes' width for two columns 219

XVI Footnotes' order 220

XVII Footnotes' rule 220

XVIII Specific skip for first series of footnotes 221
XVIII.0.1 Overview 221
XVIII.0.2 User level command 222
XVIII.0.3 Internal commands 222

XIX Endnotes 223
XIX.1 Internal commands 223
XIX.2 User level commands 227
XIX.2.1 Inserting contents to endnotes 227
XIX.2.2 Printing endnotes 228

XX Generate series of notes 234
XX.1 Test if series is still existing 235
XX.2 Init specific to `reledpar` 235
XX.3 For critical footnotes 235
XX.3.1 Options 235
XX.3.2 Create inserts, needed to add notes in foot 237
XX.3.3 Create commands for critical apparatus, `\Afootnote`, `\Bfootnote` etc. 237
XX.3.4 Set standard display 240
XX.4 For familiar footnotes 240
XX.4.1 Options 240
XX.4.2 Create tools for familiar footnotes (`\footnoteX`) 241
XX.5 The endnotes 243
XX.5.1 The auxiliary file 243
XX.5.2 The main macro 243
XX.5.3 Tools 244
XX.5.4 Internal commands 244
XX.5.5 The options 245
XX.6 Init standards series (A,B,C,D,E) 246

XXI Setting series display 246
XXI.1 Change series order 246
XXI.2 Test series order 247
XXI.2.1 Get the first series 247
XXI.3 Series setting 247
XXI.3.1 General way of working 247
XXI.3.2 Tools to set options 248
XXI.3.3 Tools to generate options commands 249
XXI.3.4 Options for critical notes 251
XXI.3.5 Options for familiar notes 252
XXI.3.6 Options for endnotes 253
XXI.4 Hooks for a particular footnote 255
XXI.5 Alias 255

XXII Output routine 256
XXII.1 Extra footnotes output 256
XXII.2 Patching standard output's commands 259

XXIII Cross referencing 262
XXIII.1 Compatibility with `xref` 276

XXIV Side notes 277

XXV Minipages and such 285

XXVI Indexing 290
XXVI.1 Looking on package order 290
XXVI.2 Auxiliary macros for `\edindex` 291
XXVI.3 Code specific to `\edindexin` critical footnotes 292
XXVI.4 Analysis of command in indexed text 293
XXVI.5 Code for the formatted index 293
XXVI.6 Main code 294
XXVI.7 Hyperlink 296
XXVI.8 'innote' and 'notenumber' option of `indextols` package 298

XXVII Glossaries 299

XXVIII Verse 301
XXVIII.1 Hanging symbol management 301
XXVIII.2 Using & character 302
XXVIII.3 Code category setting 302
XXVIII.4 Stanza count and indent 302
XXVIII.5 Numbering stanza 304
XXVIII.6 Stanza number in note 305
XXVIII.7 Main work 305
XXVIII.8 Restore catcode and penalties 308

XXIX Apparatus of Manuscripts 309
XXIX.1 User level macro 309
XXIX.2 Setting macro 310
XXIX.3 Counters and lists 311
XXIX.4 Auxiliary file macros 311
XXIX.5 Action macro 312
XXIX.6 Inserting footnote 317
XXIX.7 Other 317

XXX Arrays and tables 318
XXX.1 Preamble: macro as environment 318
XXX.2 Tabular environments 321
XXX.2.1 Disabling and restoring commands 321
XXX.2.2 Counters, boxes and lengths 325
XXX.2.3 Tabular typesetting 328
XXX.2.4 Environments 339

XXXI Quotation's commands 340

XXXII Section's title commands 341
XXXII.1 Commands to disable some feature 341
XXXII.2 General overview 341
XXXII.3 `\beforeeledchapter` command 342
XXXII.4 Auxiliary commands 342
XXXII.5 Patching standard commands 343
XXXII.6 Main code of `\eledxxx` commands 348
XXXII.7 Macros written in the auxiliary file 351

XXXIII Page breaking or no page breaking depending of specific lines 353

XXXIV Long verse: prevents being separated by a page break 355

XXXV Tools for `hyperref` package 355

XXXVI Compatibility with `eledmac` 356

Appendix A Things to do when changing versions 359
Appendix A.1 Migrating from `edmac` to `ledmac` 359
Appendix A.2 Migration from `ledmac` to `eledmac` 360
Appendix A.3 Migration to eledmac 1.5.1 361
Appendix A.4 Migration to eledmac 1.12.0 361
Appendix A.5 Migration to eledmac 17.1 362
Appendix A.6 Migration to eledmac 1.21.0 362
Appendix A.6.1 `\Xledsetnormalparstuff` and `\ledsetnormalparstuffX` 362
Appendix A.6.2 Endnotes 362
Appendix A.7 Migration to eledmac 1.22.0 362
Appendix A.8 Migration to eledmac 1.23.0 362

Appendix A.9 Migration from `eledmac` to `reledmac` 363
Appendix A.9.1 Risk of ‘no room for a new’ 363
Appendix A.9.2 Multiple indices with memoir 363
Appendix A.9.3 Deprecated commands and options 363
Appendix A.9.4 `\renewcommand` replaced by command 364
Appendix A.9.5 Commands the names of which have been changed . . . 364
Appendix A.9.6 Endnotes 366
Appendix A.9.7 Z Series 366
Appendix A.9.8 Internal commands 366
Appendix A.10 Migration to `reledmac` 2.1.0 366
Appendix A.11 Migration to `reledmac` 2.1.3 366
Appendix A.12 Migration to `reledmac` 2.3.0 366
Appendix A.13 Migration to `reledmac` 2.4.0 367
Appendix A.14 Migration to `reledmac` 2.5.0 367
Appendix A.15 Migration to `reledmac` 2.7.0 367
Appendix A.16 Migration to `reledmac` 2.7.2 367
Appendix A.17 Migration to `reledmac` 2.8.0 367
Appendix A.18 Migration to `reledmac` 2.13.1 367
Appendix A.19 Migration to `reledmac` 2.18.0 368
Appendix A.20 Migration to `reledmac` 2.21.0 368

References **369**

Index **369**

Change History **418**

1 Introduction

1.1 Aim of the package

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

- automatic stepped line numbering, by page, section or paragraph;
- sub-lineation within the main series of line numbers;
- variant readings automatically keyed to line numbers;
- caters to both prose and verse;
- multiple series of footnotes and endnotes;
- block or columnar formatting of the footnotes;
- simple tabular material may be line numbered;
- indexing keyed to page and line numbers.

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

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

Any suggestions for new features are welcome.

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

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

1.2 History

1.2.1 edmac

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

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

The present version was developed in the summer of 1990, with the intent of adding necessary features, streamlining and documenting the code, and further generalizing it to make it easily adaptable to the needs of editors in different disciplines. John did most of the general reworking and documentation, with the financial assistance of the Division of the Humanities and Social Sciences, California Institute of Technology. Dominik adapted the code to the conventions of Frank Mittelbach's doc option, and added some documentation, multiple-column footnotes, cross-references, and crop marks.[1] A description by John and Dominik of this version of edmac was published as 'An overview of edmac: a PLAIN TeX format for critical editions', *TUGboat 11* (1990), pp. 623–643.

From 1991 through 1994, the macros continued to evolve, and were tested at a number of sites. We are very grateful to all the members of the (now defunct) edmac@mailbase.ac.uk discussion group who helped us with smoothing out the bugs and infelicities in the macros. Ron Whitney and our anonymous reviewer at the TUG

[1] 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).

were both of great help in ironing out last-minute wrinkles, while Ron made some important suggestions which may help to make future versions of `edmac` even more efficient. Wayne Sullivan, in particular, provided several important fixes and contributions, including adapting the Mittelbach/Schöpf 'New Font Selection Scheme' for use with Plain TeX and `edmac`. Another project Wayne has worked on is a `DVI` post-processor which works with an `edmac` that has been slightly modified to output `\specials`. This combination enables you to recover to some extent the text of each line as ASCII code, facilitating the creation of concordances, an *index verborum*, etc.

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

1.2.2 `ledmac`

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

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

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

[2] 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).

[3] Being prepared at the German Copernicus Research Institute, Munich.

[4] Being prepared by Menso Folkerts *et al.*, at the Institut für Geschichte der Naturwissenschaften in Munich.

[5] Richard Lorch, Gerhard Brey *et al.*, at the same Institute.

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

[7] Menso Folkerts, 'Die *Rithmachia* des Werinher von Tegernsee', ibid.

[8] Geert H. M. Claassens, *De Middelnederlandse Kruisvaartromans*, (Amsterdam: Schiphower en Brinkman, 1993).

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

[10] Being produced, as was the previous book, by Gyula Mayer in Budapest.

[11] 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`)

[12] Being prepared at Poona and Lausanne Universities.

February 2004. Sidenotes and regular footnotes in numbered text were added in April 2004. This port was called `ledmac` (LaTeX `edmac`).

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

1.2.3 `eledmac`

Important changes were put in version 1.0, to make `ledmac` more easily extensible (see 7 p. 36). 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. 360.

1.2.4 `reledmac`

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

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

A new name was selected: `reledmac` (extended renewed `eledmac`). To migrate from `eledmac` to `reledmac`, please read Appendix A.9 p. 363.

1.3 Bibliography

A collaborative list of works edited with (r)(e)(ledmac is available at `https://www.zotero.org/groups/critical_editions_typeset_with_edmac_ledmac_and_eledmac/items`. Please add your own edition made with (r)(e)ledmac.

If you write book or article about (r)(e)ledmac, please add it on the collaborative bibliography on `https://www.zotero.org/groups/articles_and_books_about_reledmac/items`.

2 How the package works — the problem of the number of LaTeX runs

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

However, the best way to be sure that one has made the right number of runs is to use some of LaTeX's run scripts like *latexmk*.

3 Compatibility warning

If you use other classes than `\article` or `\book`, or modify the layout with `geometry`, some settings should be made to have correct height for the blocks of notes.

Please read 7.13.6 p. 48.

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.

4 Options

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

4.1 Specific features

draft underlines lemmas in the main text.

auxdir `reledmac` generates auxiliary files. It could be useful to store them in a specific directory. You can set it using `auxdir=`⟨*folder*⟩ option. Note the two following point:

1. TeX is not able to create folder. You should create it yourself.
2. The option does not change the default LaTeXauxiliary files (.aux, .toc, ...).

eledmac-compat help to migrate from `eledmac` to `reledmac` (see Appendix A.9.5 p. 364).

nopenalties must be called in some cases when using paragraphed endnotes (**??** p. **??**

nopbinverse prevents page break within verse environment.

noquotation by default, the quotation environment is redefined within numbered text. You can disable this redefinition with `noquotation` (see 17 p. 68).

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

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

xindy and `xindy+hyperref` select `xindy` as the index processor (13.4 p. 62).

4.2 Optimizing package performance

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

noeledsec disables tools for `\eledsection` and related commands (16.2 p. 67).

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

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

noledgroup `reledmac` allows use of a series of critical notes and a new series of normal notes inside minipage and ledgroup environments (see 10 p. 54). However, such features use up computer memory, at the expense of other processing needs. So if you do not need this feature, use `noledgroup` option. This should make `reledmac` faster.

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

5 Text lines and paragraphs numbering

5.1 Text lines numbering

`\beginnumbering`
`\endnumbering`

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

```
\beginnumbering
Text
\endnumbering
```

The `\beginnumbering` macro resets the line number to zero, reads an auxiliary file called ⟨*jobname*⟩`.nn` (where ⟨*jobname*⟩ is the name of the main input file for this job, and `nn` is 1 for the first numbered section, 2 for the second section, and so on), and then creates a new version of this auxiliary file to collect information during this run. The first instance of `\beginnumbering` also opens a file called ⟨*jobname*⟩`.<series>end` to receive the text of the endnotes. `\endnumbering` closes the ⟨*jobname*⟩`.nn` file.

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

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

5.2 Paragraphs

5.2.1 Basics

`\pstart`
`\pend`

Within a numbered section, each paragraph of numbered text must be marked using the `\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
```

5.2.2 Automatically producing `\pstart` … `\pend`

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

```
\begingroup
  \beginnumbering
  \autopar

  A paragraph of numbered text.

  Another paragraph of numbered
  text.

  \endnumbering
\endgroup
```

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

5.2.3 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`.

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

Note that a \noindent will be automatically added before this argument, and, consequently, a \parskip will be inserted. You can use a second optional argument, in brackets, to not have this \noindent.

```
\pstart[foo] % A \noindent will be inserted before foo.
\pstart[][foo]% No \noindent before foo.
```

The second optional argument of \pstart / \pend replace the argument of \AtEveryPstart* / \AtEveryPend*.

If you need to start a \pstart with brackets, or to add brackets after a \pend, just add a \relax between \pstart … \pend and the brackets.

This feature is also useful when typesetting verses (see 9 p. 50) or reledpar (see 19.2 p. 72).

A \noindent is automatically added before this argument.

5.2.4 Content before every \pstart and after every \pend

\AtEveryPstart
\AtEveryPend

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

Note that a \noindent will be inserted before the argument, and, consequently, a \parskip. You can use the starred version of \AtEveryPstart and \AtEveryPend to no insert the \noindent.

\AtStartEveryPstart
\AtEndEveryPend

The argument of \AtStartEveryPstart / \AtEndEveryPend will be inserted at the beginning / the end of every \pstart / \pend in the same paragraph. For example, if you want each \pstart to start with a star, you can use:

```
\AtStartEveryPstart{*}
```

Instead of manually doing

```
\pstart * Real pstart content.\pend
```

5.2.5 Numbering paragraphs (\pstart)

\numberpstarttrue
\numberpstartfalse
\thepstart

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

```
\setcounter{pstart}{value}
```

On each \beginnumbering the numbering restarts.

\sidepstartnumtrue

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

\labelpstarttrue

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

5.2.6 Languages written in Right to Left

If you use languages written right to left with LuaLaTeX or XeLaTeX, you must switch text direction *before* the `\pstart` command.

5.2.7 Memory limits

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

`\pausenumbering`
`\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 LaTeX may reach its memory limit. There are two solutions to this.

The first solution is to get a larger LaTeX with increased memory.

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

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

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

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

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

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

5.3 Lineation commands

5.3.1 Disabling lineation

`\numberlinefalse`
`\numberlinetrue`

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

5.3.2 Setting lineation start and step

\firstlinenum
\linenumincrement

By default, reledmac numbers every 5th line. There are two counters that control this behaviour: firstlinenum and linenumincrement. They can be changed using \firstlinenum{⟨*num*⟩} and \linenumincrement{⟨*num*⟩}. \firstlinenum specifies the first line that will have a printed number, and \linenumincrement is the difference between succesive 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

There are similar commands, \firstsublinenum{⟨*num*⟩} and \sublinenumincrement{⟨*num*⟩} for controlling sub-line numbering.

\linenumberlist

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

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

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

```
\gdef\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.

5.3.3 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.

5.3.4 Setting line number margin

\linenummargin

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

```
\linenummargin{left}
```

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

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

5.3.5 Other settings

`\leftlinenum`
`\rightlinenum`
`\linenumsep`

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

5.4 Changing the line numbers

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

5.4.1 Sublineation

`\startsub`
`\endsub`

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

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

You can change the separator between line number and subline number either by using `\Xsublinesep` without any optional argument (7.2.9 p. 40) or by using `\Xsublinesepside`. But in the second case, it will change the separator only for line numbers in the margins, not in the footnotes.

5.4.2 Locking lineation

`\startlock`
`\endlock`

The `\startlock` command, used in running text, locks the line number at its current value, until you insert `\endlock`. It can tell for itself whether you are in a patch of line or sub-line numbering. One use for line-number locking is in printing poetry: there the line numbers should be those of verse lines rather than of printed lines, even when a verse line requires several printed lines. But in this case you may use the `\stanza` mechanism, see 9 p. 50.

`\lockdisp`

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

5.4.3 Setting and changing line number

`\setline`
`\advanceline`

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

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

5.4.4 Line number style

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

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

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

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

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

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

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

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

5.4.5 Skipping and hidding 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.

`\hidenumberingonleftpage` `\hidenumberingonleftpage` is like hidenumbering, but is applied only on left page. `\hidenumberingonrightpage` is applied on right page. They can be useful if the position of the line number is depending of the position of the page, but the position of marginal note is fixed.

5.5 Executing code at each line

\dolinehook
\doinsidelinehook

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

6 Apparatus commands

6.1 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.

6.2 Critical notes

6.2.1 The lemma

\edtext

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

`\edtext{⟨lemma⟩}{⟨commands⟩}`

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

For example:

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

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

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

1 Smith] Jones C, D.

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

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

6.2.2 Footnotes

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

`\Afootnote` `\Bfootnote` `\Cfootnote` `\Dfootnote` `\Efootnote`

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

If you need more series of critical notes, please look at 6.7.1 p. 35.

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. 7.2.5 p. 39).
- `nonum` disables line numbering for this note. A horizontal blank space is added instead. You can use `\Xinplaceoflemmaseparator` to set it (7.5.1 p. 43).
- `nosep` to disable the lemma separator for this note.
- `linerangesep=`⟨*c*⟩ to change to ⟨*c*⟩ the separator between start line and end line for this particular note.

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

6.2.3 Endnotes

`\Aendnote` `\Bendnote` `\Cendnote` `\Dendnote` `\Eendnote`

Inserting endnotes The package also maintains five separate series of endnotes.

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

The mechanism is similar to the one for footnotes: each macro takes one or more optional arguments and one single argument, like:
`\Aendnote[`⟨*option*⟩`]{`⟨*text*⟩`}`.

⟨*option*⟩ can contain a comma-separated list of values. Allowed values are:

- `fulllines` to disable `\Xendtwolines` and `\Xendmorethantwolines` features for this particular note (cf. 7.2.5 p. 39).
- `nonum` to disable line number for this particular note.
- `nosep` to disable the lemma separator for this particular note. A horizontal blank space is added instead. You can use `\Xendinplaceoflemmaseparator` to set it (7.5.2 p. 43).
- `linerangesep=`⟨*c*⟩ to change to ⟨*c*⟩ the separator between start line and end line for this particular note.

`\doendnotes`

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

`\doendnotesbysection`

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

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

Note that by default inside endnotes no separator is used between the lemma and the content. However you can use the `\Xendlemmaseparator` macro to define one (7.5.2 p. 43).

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

`toendnotes`
`Xtoendnotes`

Code between endnotes Sometimes, it is useful to insert content between endnotes of the same series: for example to separate endnotes of different sections of the same text. In this case, you could use *inside numbered text* the command:
`\toendnotes[`⟨*series*⟩`]{`⟨*content*⟩`}` where ⟨*series*⟩ is a comma-separated list of the series of endnotes where ⟨*content*⟩ must be inserted. If ⟨*series*⟩ is empty, then ⟨*content*⟩ is inserted to all the series.

For example:

```
\toendnotes{\section{Section's title}}
```

Alternatively, you can use `\Xtoendnotes{`⟨*content*⟩`}`, where “X” must be replaced by a series letter.

Remember that the endnotes are temporarily stored in an auxiliary file. That means in general you want to write the ⟨*content*⟩ in the auxiliary file *without expanding it*, that is without interpreting TeX content.

However, in some cases, you may want to write a once-expanded[14] version of the ⟨*content*⟩, that is the version where the commands are expanded on the first level. This can be, for example, to get a counter value. Use the starred version in this case. For example:

```
\Atoendnotes*{\string\section{Letter 1 (chap. \thechapter)}}
```

6.2.4 Paragraph in critical apparatus

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

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

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

6.2.5 Change lemma and line number

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

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

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

   ____________
   1  Smith ]  Jones C, D.
   ____________
   1–2  I … Tuesday. ]  The date was July 16, 1954.
```

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

[14] The expansion mechanism of TeX is a quite complex problem, but fundamental. We have no place to explain it fully here. Read introduction to TeX to understand well.

`\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 (11 p. 55) 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.

6.2.6 Changing the names of commands for critical apparatus

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

We recommend that you create a series of such aliases and use them instead of the names chosen here; all you have to do is put commands of this form at the start of your file:[15]

```
\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}}
```

6.3 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 that once in the same line.

6.3.1 Basic use

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

[15] 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

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.

6.3.2 Notes about input encoding with UTF-8 processor

If you use UTF-8 processor, like XƎLaTeX or LuaLaTeX, 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, “ᾳ” 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 XƎLaTeX, add this line in your preamble: `\XeTeXinputnormalization 1`.
- If you use LuaLaTeX, use the `uninormalize` package of Michal Hoftich[16] with the `buffer` option set to true.

With these tools, XƎTeX / LuaTeX will dynamicaly normalize unicode input when reading the file. Consequently, you will have no problems with the `\sameword` command.

6.3.3 Use with `\lemma` command

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

For example in the following example:

```
some thing
   \edtext{\sameword{sw}
```

handbook of `xargs` to know more about `\newcommandx`.

[16] `https://github.com/michal-h21/uninormalize`.

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

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

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

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

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

Note also that the ⟨*X*⟩ does not refer to the level where the \sameword occurs, but to the level of the \lemma that refers to that \sameword. For example:

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

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

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

$$\boxed{\boxed{1_{\text{inlemma}}\ \boxed{2}\ 3_2}^{1\ldots3}\ 4\ 5_1}^{1\ldots5}$$

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 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.

Here, the corresponding code:

```
\edtext{%
  \edtext{%
    \sameword[inlemma]{A} (1)
      \edtext{%
        \sameword{A} (2)
      }%
      {%
      \Afootnote{level~3}%
      }%
    \sameword[2]{A} (3)
  }%
  {%
    \lemma{%
       \sameword{A}%
       \ldots%
       \sameword{A}%
       }%
    \Afootnote{level~2}%
  }%
  \sameword{A} (4)
  \sameword[1]{A} (5)
}%
{%
  \lemma{\sameword{A}\ldots\sameword{A}}%
  \Afootnote{level~1}%
}
```

A (1) A (2) A (3) A (4) A (5)

1 A^1...A^5] level 1

1 A^1...A^3] level 2

1 A^2 (2)] level 3

6.3.4 Sameword for a group of words

Sometime, a group of words, and not only a single word, occurs multiple times. In this case, you have two possibilities.

First, you can consider only the individual words, and not groups of word. For example:

```
\sameword{per} \sameword{causam}
tamen scire
\edtext{\sameword{causam}}{\Bfootnote{fnote}}
est
\edtext{\sameword{per} \sameword{causam}}{\Bfootnote{causam rei B}}
cognoscere
\edtext{\sameword{causam}}{\Bfootnote{fnote}}
```

per causam tamen scire causam est per causam cognoscere causam

1 causam^2] fnote

1 per^{2} causam^{3}] causam rei B

1 causam^{4}] fnote

Here, it is not ambiguous what "per causam" refers to.

However, we may think that as "per causam" is the lemma of the second note, there should be only one number for the whole lemma. In this case we can mark all "per causam" groups. But as "causam" is also called as lemma in note 1 and 3, we need to use nested `\sameword`. Consequently, we need to use `\lemma` for the `\edtext` linked to "per causam", as we don't want to number each individual word.

```
\sameword{per \sameword{causam}} tamen scire
\edtext{\sameword{causam}}{\Bfootnote{fnote}} est
\edtext{\sameword[1]{per \sameword{causam}}}{\lemma{\sameword{per causam}}\Bfootnote{causam
\edtext{\sameword{causam}}{\Bfootnote{fnote}}
```

per causam tamen scire causam est per causam cognoscere causam

1 causam^{2}] fnote

1 per causam^{2}] causam rei B

1 causam^{4}] fnote

6.3.5 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}%
}
```

6.4 Apparatus of Manuscripts

The critical notes mostly refer to textual variants between manuscripts which contain the text to be edited. It may so happen that the manuscripts only contain parts of the text. Depending on one's wishes, `reledmac` can generate lists of relevant manuscripts for any delimited portion of text. Such lists are referred to as "apparatuses of manuscripts".

To produce an apparatus of manuscripts with `reledmac`, you have to insert specific commands that are used to mark the sections for which only part of the manuscripts are relevant. These commands will be processed, and **after the second TeX run**, corresponding apparatuses of manuscripts will be inserted in the first (viz. 'A' series) level of footnotes.

As the insertion of this apparatus can change the page breaks, you may have to run TeX two or more times. We strongly recommend to use tools like *latexmk* to do that.

6.4.1 Marking sections of text

\msdata

\msdata{⟨*text*⟩} must be inserted at the point where a section for which only part of the manuscripts are relevant starts. ⟨*text*⟩ can be any arbitrary text, viz. a list of the manuscripts that are used for the section that starts. The command must be attached right at the point where the section starts, with no space, like so:

```
\msdata{ABC}Lorem ipsum
```

Which means that the section of text starting by “Lorem ipsum” is witnessed by manuscripts A, B and C.

\stopmsdata

\stopmsdata must be inserted at the point where the section of text previously marked by \msdata ends. The command must be attached right to the end of the section, with no space. As \stopmsdata is a LaTeX argumentless macro, it will gobble the following space. To keep that space, you have to either append a backslash followed by a space or {} to \stopmsdata, like so:

```
\msdata{ABC}Lorem ipsum dolor
[…]
amet\stopmsdata{} \msdata{ABCD}sic transit […]
```

Which means that the part of text containing “Lorem ipsum dolor ... amet” is witnessed by manuscripts A, B and C, while the part of text starting by “sic transit” is witnessed by manuscripts A, B, C and D.

\stopmsdata is also automatically inserted by \msdata.

Note that in most cases, any \stopmsdata is followed by \msdata. However, as these two command are usually separated by a space, it may happen that a line break be automatically inserted between them. This is why it is advised to always insert \stopmsdata, even if \msdata inserts it in case it is forgotten.

6.4.2 Layout of the apparatus of manuscripts

On every page, the apparatus of manuscripts marks the corresponding section with starting and ending line numbers. However, the following rules will be applied:

- If the section does not start on the current page, the starting line number will be the line number of the first line on the page.
- If the section does not stop on the current page, the ending line number will be the line number of the last line on the page.
- If the section neither starts nor ends on the current page, no line number will be printed. The same is true in case both \msdata is called at the very beginning of the page and \endmsdata is called at the very end of the page.

6.4.3 Settings

As the apparatus of manuscripts technically consists of first-level critical notes ('A' series), any setting available for critical notes can be applied (7 p. 36). However, the following *additional* commands are available.

`\setmsdataseries` The series used by default for the apparatus of manuscripts is series A. However, you can change it with `\setmsdataseries{`⟨*series*⟩`}`.

`\setmsdatalabel` As the apparatus of manuscripts consists of regular critical footnotes, a lemma is associated to them. By default, it is "Ms.". You can change it using `\setmsdatalabel{`⟨*txt*⟩`}`.

`\setmsdataposition` If you want the manuscript apparatus to be on the same level of critical footnotes as the other apparatuses, for each line, reledmac will first insert the manuscript apparatus, then the other footnotes. You can change it using:
`\msdataposition{regular-msdata}`
And restore the default behaviour using `\msdataposition{msdata-regular}`

6.5 Familiar notes

6.5.1 Basic use

`\footnoteA`
`\footnoteB`
`\footnoteC`
`\footnoteD`
`\footnoteE`

As well as the standard LaTeX footnotes generated via `\footnote`, the package also provides five series of additional footnotes called `\footnoteA` through `\footnoteE`. These have the familiar marker in the text, and the marked text at the foot of the page can be formated using any of the styles described for the critical footnotes. Note that the 'regular' footnotes have the series letter at the end of the macro name whereas the critical footnotes have the series letter at the start of the name.

6.5.2 Customizing mark

`\thefootnoteA`
`\bodyfootmarkA`
`\footfootmarkA`

Each series uses a set of macros for styling the marks. The mark numbering scheme of series A is defined by the `\thefootnoteA` macro; the default is:

```
\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.

6.5.3 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[3,4] 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}{^{\normalfont,}}
and can be changed if necessary.

6.6 Printing the footnote mark without printing the footnote text

\footnoteXmark
\footnoteXtext

In certain cases, you can't directly use \footnoteX; for example, when using\uline command of the ulem package. You need to print the footnote mark first, then call the footnote text to be inserted.

For all ⟨X⟩ command, reledmac provides a \footnote⟨X⟩mark command and a \footnote⟨X⟩text command, equivalent to standard LaTeX's command \footnotemark and \footnotetext. For example, to use with \uline, do:

```
  This is \uline{a test containing\mbox{\footnoteAmark}}\footnoteAtext{A
 simple footnote.}\uline{ a simple footnote.}
```

If you use reledpar, you can't use these two commands to print the footnote mark on one side and the footnote text on the other side.

You must use \footnote⟨X⟩nomk and \footnote⟨X⟩mk, defined in reledpar (?? p. ??)

6.7 Changing series

6.7.1 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}.

6.7.2 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}
```

6.7.3 Series order

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

\seriesatbegin
\seriesatend

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

6.8 Position of critical and familiar footnotes

\fnpos
\mpfnpos

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

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

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

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

You can also decide to alternate familiar and critical footnotes with your own order. In this case, the second argument of \fnpos or \mpfnpos is a comma separated list of values. Each value has the following form:
⟨*series*⟩⟨*type*⟩

⟨*series*⟩ is a series letter (A,B,C etc.), while ⟨*type*⟩ must be either “critical” or “familiar”.

For example, suppose you want to first print the familiar footnotes of the “A” series, then all the series of critical footnotes, and finally all the series of familiar footnotes, except the “A” series. In this case, use the following command:

```
\fnpos{%
  {A}{familiar},
  {A}{critical},%
  {B}{critical},%
  {C}{critical},%
  {D}{critical},%
  {E}{critical},%
  {B}{familiar},%
  {C}{familiar},%
  {D}{familiar},%
  {E}{familiar}%
}
```

Note that you must define the position of all the series of footnotes you use. If you don’t, you will have infinite runs of LaTeX.

7 Critical apparatus appearance

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

When a length, noted $\langle l\rangle$, is used, it can be stretchable: `a plus b minus c`. The final length `m` is calculated by LaTeX to have: $a-c \leq m \leq a+b$. If you use some relative unit[17], it will be relative to font size of the footnote, except for commands concerning the place kept by the notes — including blank space.

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

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

There is also name convention:

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

7.1 Notes arrangement in a series

`\Xarrangement`
`\arrangementX`

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

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

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

- `paragraph` formats all of the footnotes of a series as a single paragraph; if you use this arrangement, you are strongly encouraged to read 19.1.6 p. 71.
- `twocol` formats them as separate paragraphs, but in two columns;
- `threecol`, in three columns.
- `normal`, restore normal arrangement.

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

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

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

[17] Like `em` which is the width of an 'm' in a given font.

`\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.

7.2 Control line number printing

7.2.1 Print line number only at first time

`\Xnumberonlyfirstinline`

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

`\Xnumberonlyfirstintwolines`

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

`\Xsymlinenum`

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

`\Xendnumberonlyfirstinline`
`\Xendnumberonlyfirstintwolines`
`\Xendsymlinenum`

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

7.2.2 Print page number only at first time

`\Xendpagenumberonlyfirst`

For endnotes, `reledmac` provides a mechanism for printing the page number only the first time it is seen. However, when a lemma spans over two pages, the line numbers are normally printed in the following pattern: starting page number - starting line number - ending page number - ending line number. It follows that what corresponds to the actual 'page number' may not be self-evident. So: `\Xendpagenumberonlyfirst[`⟨*s*⟩`]` can be called to ensure that the starting page number of a lemma be not printed if it is the same as the ending page number of the preceding lemma. You can use *additionally* one (and only one) of the following commands:

- `\Xendpagenumberonlyfirstifsingle[`⟨*s*⟩`]`: the first page number of the lemma will not be printed only if the following conditions are true:

  1. The starting page number of the lemma is the same as the ending page number of the preceding lemma.
  2. The ending page number of the lemma is the same as the starting page number of the lemma.

  In this case the ending page number will always be printed if it is different from the starting page number.

- `\Xendpagenumberonlyfirstintwo[`⟨*s*⟩`]`: both the starting page number and the ending page number of a lemma are not printed if they are both the same as the starting page number and the ending page number of the preceding lemma respectively.

In any case, you can use:

\Xendsympagenum

- `\Xendsympagenum[`⟨*series*⟩`]{`⟨*c*⟩`}` to print ⟨*c*⟩ when the page number is not printed.

\Xendinplaceofpagenumber

- `\Xendinplaceofpagenumber[`⟨*series*⟩`]{`⟨*l*⟩`}` to print a ⟨*l*⟩ length horizontal space in case no symbol is printed instead of the page number.

7.2.3 Arbitrary text before line number

\Xbeforenumber

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

7.2.4 Separator for line range

\Xlinerangeseparator
\Xendlinerangeseparator

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

7.2.5 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*⟩`]`.

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

\Xtwolinesonlyinsamepage

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

You can disable \Xtwolines and related for a specific note by using the '[fulllines]' argument in the note macro cf. 6.2.2 p. 25.

\Xendtwolines \Xendmorethantwolines \Xendtwolinesbutnotmore

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

7.2.6 Disable line number

\Xnonumber \Xendnonumber

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

7.2.7 Printing pstart number

\Xpstart

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

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

\Xpstarteverytime

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

\Xonlypstart

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

7.2.8 Printing stanza number

\Xstanza

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

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

\Xstanzaseparator

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

7.2.9 Separator between line and subline numbers

\Xsublinesep

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

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

\Xendsublinesep

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

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

The default value is \textnormal{.}.

7.2.10 Separator between page and line numbers

\Xpagelinesep \Xpagelinesep[⟨*s*⟩]{⟨*txt*⟩} changes the separator between the page and line number in footnotes.

By default, the value defined for \Xsublinesep is used.

7.2.11 Space around number

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

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

\Xendbeforenumber Xendafterenumber \Xendbeforenumber and \Xendafterenumber are the equivalents of \Xbeforenumber and \afterenumber for endnotes.

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

7.2.12 Space around line symbol

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

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

\Xendbeforesymlinenum \Xendaftersymlinenum \Xendbeforesymlinenum and \Xendaftersymlinenum are the equivalents of \Xbeforesymlinenum and \Xaftersymlinenum for the endnotes.

7.2.13 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.

7.2.14 Boxing line number and line symbol

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

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

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

\Xendboxsymlinenum

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

\Xboxlinenumalign

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

L to align left (default value);

R to align right;

C to center.

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

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

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

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

\Xendboxlinenum
\Xendboxlinenumalign
\Xendboxstartlinenumalign
\Xendboxendlinenumalign

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

7.3 For endnotes

\Xendbeforepagenumber

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

\Xendafterpagenumber
\Xendlineprefixsingle
\Xendlineprefixmore

\Xendafterpagenumber[⟨*s*⟩]{⟨*text*⟩} defines the text after the page number in endnotes. Default value is) (open parenthesis followed by a single space). \Xendlineprefixsingle[⟨*s*⟩] defines the text before the line number in endnotes, when there is only one line. Default value is empty. \Xendlineprefixmore[⟨*s*⟩]{⟨*text*⟩} defines the text before the line number in endnotes, when there is more than one line. Default value is empty. If you don’t define it, use the value defined by \Xendlineprefixsingle.

7.4 Arbitrary code around line number

\Xendbhooklinenumber

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

\Xendahooklinenumber

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

\Xendbhookinplaceofnumber

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

\Xendahookinplaceofnumber

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

7.5 Separator between the lemma and the note

7.5.1 For footnotes

\Xlemmaseparator

By default, in a footnote, the separator between the lemma and the note is a right bracket (\rbracket)[18]. 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.

7.5.2 For endnotes

\Xendlemmaseparator

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

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

\Xendbeforelemmaseparator

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

\Xendafterlemmaseparator

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

ndinplaceoflemmaseparator

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

[18] For polyglossia, when the lemma is RTL, the bracket automatically switches to a left bracket.

7.6 Font style

7.6.1 For line number

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

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

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

7.6.2 For the lemma

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

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

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

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

`\Xendlemmafont[`⟨*s*⟩`]{`⟨*cmd*⟩`}` is the same for endnotes.

7.6.3 For all notes

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

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

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

7.7 Wrapping notes

7.7.1 Wrapping lemmas

`\Xwraplemma` `\Xwraplemma[`⟨*s*⟩`]{`⟨*cmd*⟩`}` is used to wrap, in the footnote, the lemma in a LaTeX command. For example, with the bidi package, to ensure having a lemma written right to left, use \Xwraplemma{\RL}.

\Xwrapendlemma \Xendwraplemma[⟨*s*⟩]{⟨*cmd*⟩} is the same for endnotes.

7.7.2 Wrapping contents

\Xwrapcontent \Xwrapcontent[⟨*s*⟩]{⟨*cmd*⟩} is used to wrap the footnote contents — excluding the lemma — in a LaTeXcommand.

For example, if the language of your note is not the same as the language of the lemma, use \Xwrapcontent{\foreignlanguage{⟨*language*⟩}} (with babel) or \Xwrapcontent{\text⟨*language*⟩} (for babel).

\Xendwrapcontent \Xendwrapcontent[⟨*s*⟩]{⟨*cmd*⟩} is the same for endnotes.

\wrapcontentX \wrapcontentX[⟨*s*⟩]{⟨*cmd*⟩} is the same for critical footnotes.

7.8 Indent of notes content

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

\parindentX By default, reledmac does not add indentation before the paragraphs inside familiar footnotes. Use \parindentX[⟨*s*⟩] to enable indentation.

\Xhangindent For critical notes NOT paragraphed you can define an indent with \Xhangindent[⟨*s*⟩]{⟨*l*⟩}, which will be applied in the second line of notes. It can help to make distinction between a new note and a break in a note. The default value is 0 pt.

\hangindentX For familiar notes NOT paragraphed you can define an indentation with \hangindentX[⟨*s*⟩]{⟨*l*⟩}, which will be applied in the second line of notes. It can help to make a distinction between a new note and a break in a note.

\Xendhangindent For critical endnotes NOT paragraphed you can define an indentation with \Xendhangindent[⟨*s*⟩]{⟨*l*⟩}, which will be applied in the second line of notes. It can help to make a distinction between a new note and a break in a note.

7.9 Arbitrary code at the beginning of notes

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

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

\Xbhooknote \Xbhooknote[⟨*s*⟩]{⟨*code*⟩} is to be used at the beginning of the critical footnotes.

\bhooknoteX \bhooknoteX[⟨*s*⟩]{⟨*code*⟩} is to be used at the beginning of the familiar footnotes.

\Xendbhooknote \Xendbhooknote[⟨*s*⟩]{⟨*code*⟩} is to be used at the beginning of the endnotes.

7.10 Arbitrary code before inserting note

\Xbeforeinserting beforeinsertingX \Xbeforeinserting[⟨*s*⟩]{⟨*code*⟩} and \beforeinsertingX[⟨*s*⟩]{⟨*code*⟩} are very technical commands.

They allow one to add any arbitrary code just before the footnotes are added in the list of footnotes. The main use is to insert text direction code. For example, if you edit right-to-left text with `bidi`, but want your critical footnote be left-to-right, use `\Xbeforeinserting\LTR`. You should also use `\Xwraplemma` to ensure your lemmas are right-to-left in a left-to-right paragraph (7.7.1 p. 44)).

Note that the changes are local to the footnote.

7.11 Options for footnotes in columns

7.11.1 Alignment

`\Xcolalign`
`\colalignX`

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

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

`\justifying` to have text justified, as usual with 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`).

7.11.2 Size of the columns

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

`\Xhsizetwocol` `\Xhsizetwocol[`⟨*s*⟩`]{`⟨*l*⟩`}` is used to change width of a column when critical notes are displaying in two columns. Default value is `.45 \hsize`.

`\Xhsizethreecol` `\Xhsizethreecol[`⟨*s*⟩`]{`⟨*l*⟩`}` is used to change width of a column when critical notes are displaying in three columns. Default value is `.3 \hsize`.

`\hsizetwocolX` `\hsizetwocolX[`⟨*s*⟩`]{`⟨*l*⟩`}` is used to change width of a column when familiar notes are displaying in two columns. Default value is `.45 \hsize`.

`\hsizethreecolX` `\hsizethreecolX[`⟨*s*⟩`]{`⟨*l*⟩`}` is used to change width of a column when familiar notes are displaying in three columns. Default value is `.3 \hsize`.

7.12 Options for paragraphed footnotes and notes grouped by line

7.12.1 Mark separation of notes

`\Xafternote`
`\afternoteX`

You can add some horizontal 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`
`\parafootsepX`

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 `\Xparafootsep{$\parallel$}`.

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.

7.12.2 Ragged text

`\Xragged`

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

`\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.

7.13 Options for block of notes

7.13.1 Grouping notes by line

`\Xgroupbyline`

If you do not use `\Xarrangement{paragraph}`, you may want to group all the critical footnotes related to the same line in the same paragraph. In this case, use `\Xgroupbyline[⟨series⟩]`.

In many cases, you might like to use it in combination with `\Xnumberonlyfirstinline` (7.2.1 p. 38).

`groupbylineseparetwolines`

Note that the `\Xafternote` and `\Xparafootsep` settings are used to determine space and content between footnotes (7.12 p. 46). Suppose you have two notes on line 1 which overlap lines 1 and 2. This last note will be printed, if you use `\Xgroupbyline` in the same group as the previous one. In the case you want that note to be distinct, you must use both `\Xgroupbyline` and `\Xgroupbylineseparetwolines[⟨s⟩]`.

In many cases, you might like to use it in combination with `\Xnumberonlyfirstintwolines` (7.2.1 p. 38)

7.13.2 Text before notes

`\Xtxtbeforenotes`
`\txtbeforenotesX`
`\Xtxtbeforenotesonlyonce`
`\txtbeforenotesonlyonceX`

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

By default, such texts are inserted at the beginning of the groups of notes on each pages. You can add `\Xtxtbeforenotesonlyonce` (for critical footnotes) and `\txtbeforenotesonlyonceX` (for familiar footnotes) to insert them only the first time notes are typeset.

7.13.3 Code before notes

`\Xbhookgroup`
`\bhookgroupX`

While `\Xtxtbeforenotes` is for typesetting code before notes, `\Xbhookgroup` and

\bhookgroupX (respectively for critical and familiar) are for executing code before a groups of notes, between the rules and the printing of the notes.

7.13.4 Spacing

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

Be careful, the standard LaTeX footnote rule used by reledmac decreases by 3pt. This 3pt decrease is not changed by this command.

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

Be careful, the standard LaTeX footnote rule, which is used by reledmac, decreases 3pt. These 3pt are not changed by this command.

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

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

7.13.5 Rule

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

Be careful, the standard LaTeX footnote rule, which is used by reledmac, adds 2.6pt. These 2.6pt are not changed by this command.

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

Be careful, the standard LaTeX footnote rule, which is used by reledmac, adds 2.6pt. These 2.6pt are not changed by this command.

7.13.6 Maximum height

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

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

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

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

```
 }
```

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

7.13.7 Width

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

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

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

Note that changes the with of the block of notes. If you want to change the width of each column when typesetting notes in columns, use `\Xhsizetwocol`, `\Xhsizethreecol`, `\hsizetwocolX`, `\hsizethreecolX`, see 7.11.2 p. 46.

7.14 Footnotes and the `reledpar` columns

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

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

7.15 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{$\parallel$}`.

8 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.

`\numlabfont` Line numbers for the main text are usually printed in a smaller font in the margin.

The `\numlabfont` macro is provided as a standard name for that font: it is initially defined as
`\newcommand{\numlabfont}{\normalfont\scriptsize}`
You might wish to use a different font if, for example, you preferred to have these line numbers printed using old-style numerals.

`\select@lemmafont`

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

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

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

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

9 Verse

9.1 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.

If you need to add brackets directly after `\stanza`, `&` or `\&`, add `\norelax`. Otherwise, the brackets will be interpreted as delimitation of an optional argument (cf. 9.8 p. 52)

9.2 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 `\sethanginsymbol`: see p. 9.6 p. 52.

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.

9.3 Repeating stanza indents

Since version 0.13, if the indentation is repeated every *n* verses of the stanza, you can define only the *n* first indentations, and indicate that they are repeated, defining the value of the `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. 361.

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

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

```
 \setcounter{stanzaindentsrepetition}{0}
```

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

9.4 Manual stanza indent

`\stanzaindent`
`\stanzaindent*`

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

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

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

9.5 Stanza breaking

\setstanzapenalties

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

The command
`\setstanzapenalties{1,5000,10100,5000,0}`
results in a penalty of 5000 being placed after the first and third lines of the stanza, and a penalty of -100 after the second.

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

9.6 Hanging symbol

\sethangingsymbol

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{[\,}
```

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

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

9.7 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 18.2 p. 69 for further details.

9.8 Content before/after verses

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

- The `\stanza` command can take an optional argument (in brackets). Its content will be printed before the stanza. A `\noindent` is inserted before the content of first optional argument. If you don't want this \noindent, you can use the second optional argument (also in brackets):

```
\stanza[foo] % \noindent is inserted before foo.
\stanza[][foo] % There is no \noindent inserted before foo.
```

\AtEveryStanza

- Use \AtEveryStanza{⟨*arg*⟩} to automatically add content before the stanza (not in the same paragraph).

  Note that a \noindent will be inserted before the argument, and, consequently, a \parskip. You can use the starred version of \AtEveryStanza to avoid this \noindent.

\AtStartEveryStanza

- Use \AtStartEveryStanza to automatically add content at the beginning of stanza (in the same paragraph).
- & 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.

  A \noindent is automatically inserted before the contents of these optional arguments.

  Use a third and fourth optional argument to not add these \noindents (to add content respectively after the current verse / before the next verse).
- Use \AtEveryPend{⟨*arg*⟩} to automatically add content after verses (including the final one) and \AtEveryPstart{⟨*arg*⟩} to automatically add content before verses (including the first one).
- \& can take an optional argument (in brackets). Its content will be printed after the stanza.

\AtEveryStopStanza

- Use \AtEveryStopStanza to automatically add content after the end of stanzas (not in the same paragraph).

  Note that a \noindent will be inserted before the argument, and, consequently, a \parskip. You can use the starred version of \AtEveryStopStanza to avoid this \noindent.

\AtStartEveryStanza

- Use \AtStartEveryStanza to automatically add content at the end of stanza (in the same paragraph).

9.9 Numbering stanza

\numberstanzatrue
\numberstanzafalse

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

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

You can use this feature in combination with \Xstanza (7.2.8 p. 40).

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 LaTeX.

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

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

9.10 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`.

9.11 Notes on empty lines

Since v2.3.0 of `reledmac`, empty lines when typesetting verses no longer produce new paragraphs, and consequently, do not insert vertical spaces. Use optional argument of `\stanza` or `\newverse` to insert vertical space (9.8 p. 52).

10 Grouping

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

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

You can also put familiar footnotes (see section 6.5) 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.

`ledgroupsized` The `ledgroupsized` environment is similar to `ledroup` except that you must specify a width for the environment, as with a minipage.
`\begin{ledgroupsized}[`⟨*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{ledgroupsized}{\textwidth}` is effectively the same as `\begin{ledgroup}`

11 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.

11.1 Basic use

`\edlabel`

First you place a label in the text using the command `\edlabel{`⟨*lab*⟩`}`. ⟨*lab*⟩ can be almost anything you like, including letters, numbers, punctuation, or a combination—anything but spaces; you might type `\edlabel{toves-3}`, for example.[19]

`\edpageref`
`\edlineref`
`\sublineref`
`\pstartref`

Elsewhere in the text, either before or after the `\edlabel`, you can refer to its location with `\edpageref{`⟨*lab*⟩`}`, `\edlineref{`⟨*lab*⟩`}`, `\edsublineref{`⟨*lab*⟩`}` or `\pstartref{`⟨*lab*⟩`}`, that will produce, respectively, the page, line, sub-line and pstart on which the `\edlabel{`⟨*lab*⟩`}` command occurred.

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

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

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

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

[19] More precisely, you should stick to characters in the TeX categories of “letter” and “other”.

11.2 Cross-referencing to a critical note

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

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

If you want to refer to starting and ending lines, you should use `\appref` and related tools (11.6.2 p. 57).

11.3 Cross-referencing which return a number in any case

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

Where `#1` stands for the reference.

However, there are situations in which you will want `reledmac` to return a number without displaying any warning messages about undefined labels or the like: if you want to use the reference in a context where LaTeX 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 6.2.5 p. 27).

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

- They will not tell you if the label is undefined.
- They must be preceded in the file by at least one of the four other cross-reference commands—e.g., a `\edlabel{foo}` command, even if you never refer to that label—since those commands can all do the necessary processing of the `.aux` file, and the `\x...` ones cannot.
- When `hyperref` is loaded, the hyperref link will not be added. (Indeed, it is not a limitation, but a feature.)
- With `reledpar`, the `\xlineref` does not insert the right side flag, in order to obtain a line number. Use `\xflagref` to obtain the side flag, depending of your flag.

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

`\xxref`

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 automatically calls `\linenum` (q.v., 6.2.5 p. 27 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.

For example, one might use the following:

```
\beginnumbering

\pstart
```

```
\edlabel{Queritur}Queritur utrum metaphysica sit scientia una.
\pend

\pstart
\edtext{Et videtur quod non\edlabel{non}.}{\xxref{Queritur}{non}\lemma{queritur \dots{} non}\Afootnote{
\pend

\endnumbering
```

11.4 Not automatic cross-referencing

\edmakelabel

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

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

11.5 Normal LaTeX cross-referencing

\label
\ref
\pageref

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

11.6 References to start and end lines

11.6.1 Reference to main text lines

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

\edlabelS

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

\edlabelE

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

\edlabelSE

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

\SEref

The main utility is to use them with three other commands. `\SEref{`⟨*label*⟩`}` will make a cross-reference printed as a reference in critical footnotes.

\SErefwithpage

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

\SErefonlypage

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

11.6.2 References to lines that are commented on in the apparatus

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

\applabel

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

If you use \linenum (6.2.5 p. 27) 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.

11.6.3 Settings

\setapprefprefixsingle
\setapprefprefixmore

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

For example, you may use:

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

Note that if you do not use \setapprefprefixmore, the argument of \setapprefprefixsingle will be used in any case.

\setSErefprefixsingle
\setSErefprefixmore
\setSErefonlypageprefixsingle
\setSErefonlypageprefixmore

\setSErefprefixsingle and \setSErefprefixmore are similar for \SEref command.

Use \setSErefonlypageprefixsingle{⟨*prefix*⟩} to set the page prefix for \SErefonlypage when there is only one page. Use \setSErefonlypageprefixmore{⟨*prefix*⟩} to set it when there is more than one page. For example:

```
\setSErefonlypageprefixsingle{p.~}
\setSErefonlypageprefixmore{pp.~}
```

Note that if you do not use \setSErefonlypageprefixmore, the value of \setSErefonlypageprefixsi is used instead.

Also note that \setSErefonlypageprefixsingle is only a shortcut for \Xendbeforepagenumbe (see 11.6.3 p. 58). So if you use \Xendbeforepagenumber without any optional argument, it will override this setting.

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

These commandes are the following:

- \Xlineflag (for reledpar), enabled by default.
- \Xlinerangeseparator
- \Xmorethantwolines

- \Xsublinesep
- \Xtwolines
- \Xtwolinesbutnotmore
- \Xtwolinesonlyinsamepage

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

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

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

For one specific command When calling \appref and \SEref, you can use as a first optional argument, in brackets ([]), any optional argument which can be used for critical footnotes (6.2.2 p. 25).

When calling \apprefwithpage, \SErefwithpage or \SErefonlypage you can use as a first optional argument, in brackets ([]), any optional argument which can be used for critical endnotes (6.2.3 p. 25).

11.7 Compatibility with xr package

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

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

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

12 Side notes

12.1 Basics

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

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

The notes will appear only after the second LaTeX run. If the note positions change in your `.tex` file, you need two runs to get the correction position in the output file. You are strongly encouraged to use tools like *latexmk*, to be sure to get the correct number of runs.

12.2 Setting

12.2.1 Width

`\ledlsnotewidth`
`\ledrsnotewidth`

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

12.2.2 Vertical position

`\rightnoteupfalse`
`\leftnoteupfalse`

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

12.2.3 Distance to the main text

`\ledlsnotesep`
`\ledrsnotesep`

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

12.2.4 Font

\ledlsnotefontsetup
\ledrsnotefontsetup

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

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

These can of course be changed to suit.

12.2.5 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⟩}`.

13 Indexing

13.1 Basics

\edindex

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

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

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

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

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

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

Also note that using `\edtext` in familiar footnotes refers to the line where the footnotes are called

13.2 Referring to critical notes

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

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

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

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

13.3 Separator between page and line numbers

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

`-` is the default separator used by the MakeIndex program.

You can reconfigure it, this example defines a colon as separator:

```
\renewcommand{\pagelinesep}{:}
```

However, you also have to configure your `.ist` index style file. For example if you use `:` as separator[20].

```
page_compositor ":"
```

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

13.4 Using xindy

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

An example of such a file is provided in the "examples" folder. Read the xindy handbook to learn how to use it.[21]

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

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

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

[20] For further detail, you can read `http://tex.stackexchange.com/a/32783/7712`.

[21] Or, for people who read French, read `http://geekographie.maieul.net/174`.

1. Use `xindy+hyperref` option when loading the `reledmac` package. When you run (Xe/Lua)LaTeXwith 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.[22].

13.5 Advanced setting

`\edindexlab` The `\edindex` process uses a `\label` and `\ref` mechanism to get the correct line number. It automatically generates labels of the form `\label{\edindexlab N}`, where N is a number, and the default definition of `\edindexlab` is:
`\newcommand*{\edindexlab}{$&}`
in the hopes that this will not be used by any other labels (`\edindex`'s labels are like `\label{$&27}`). You can change `\edindexlab` to something else if you need to.

14 Glossary

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

14.1 Preamble setting

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

- Or set . `glossaries`:
  `\glsSetCompositor{-}`
- Or set `reledmac`:
  `\renewcommand{\pagelinesep}{-}`
  In this case, the above will have consequences for your use of `\edindex` and you should set your `.ist` file (13.3 p. 62).

14.2 Commands

The `\gls`, `\Gls`, and related commands of `glossaries` packages have a prefixed version with `ed`, which refers to the page line. The argument are the same as for the standard commands. So for example:
`\edgls[`⟨*options*⟩`]{`⟨*label*⟩`}[`⟨*insert*⟩`]`

[22] These are the recommended lines to provide the best possible compatibility between hyperref and xindy, even without using `reledmac`.

15 Tabular material

LaTeX'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
edarrayc
edarrayr
edtabularl
edtabularc
edtabularr

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

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

1	2	3
a	bb	ccc
AAA	BB	C

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

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

For example:

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

produces the following parallel pair of verses.

1	**I**	wish I was a little bug	**I**	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 \spreadtext \spreadmath{⟨*math*⟩} typesets {⟨*math*⟩} but the {⟨*math*⟩} has no effect on the calculation of column widths. \spreadtext{⟨*text*⟩} is the analagous command for use in edtabular environments.

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

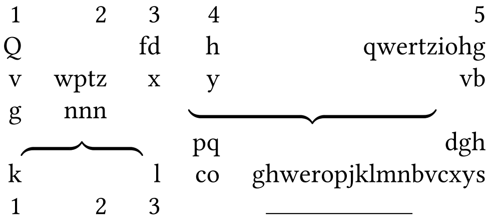
$$\begin{array}{llll} 1 & 2 & 3 & 4 \\ & F+G+C & & \\ a & bb & ccc & dddd \end{array}$$

\edrowfill The macro \edrowfill{⟨*start*⟩}{⟨*end*⟩}{⟨*fill*⟩} fills columns number ⟨*start*⟩ to ⟨*end*⟩ inclusive with ⟨*fill*⟩. The ⟨*fill*⟩ argument can be any horizontal 'fill'. For example \hrulefill or \upbracefill.

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

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

```
\begin{edtabularr}
1                                & 2    & 3  & 4  & 5 \\
Q                                &      & fd & h  & qwertziohg \\
v                                & wptz & x  & y  & vb \\
g                                & nnn  & \edrowfill{3}{5}{\upbracefill} & & \\
\edrowfill{1}{3}{\downbracefill} &      &    & pq & dgh \\
k                                &      & l  & co & ghweropjklmnbvcxys \\
1                                & 2 & 3 & \edrowfill{4}{5}{\hrulefill} &
\end{tabularr}
```



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

You can also define your own 'fill'. For example:

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

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

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

$$\begin{array}{cccc} 1 & 2 & 3 & 4 \\ a & \multicolumn{2}{c}{\text{␣}} & d \\ A & B & C & D \end{array}$$

\edatleft
\edatright

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

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

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

\edbeforetab
\edaftertab

\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}
```

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

\edvertline
\edvertdots

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

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

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

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

16 Sectioning commands

16.1 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` (5.2.3 p. 18):

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

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

16.2 Sectioning commands with line numbering and critical notes

You have to use the following commands:

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

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

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

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

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

16.3 Optimization

\noeledsec

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

17 Quotation environments

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

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

18 Page breaks

18.1 Control page breaking

\ledpb
\lednopb

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

These commands have effect only at the second run.

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

`\ledpbsetting`

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

18.2 Prevent page break in a long verses

`\lednopbinversetrue`

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

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

19 Miscellaneous

`\extensionchars`

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

`\ifledfinal`

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

`\showlemma`

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

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

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

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

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

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

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

19.1 Known and suspected limitations

19.1.1 Non-standard geometry

If you use classes other than `article` or `book`, or if you use the geometry package, you should use maxhnotesX and/or `\Xmaxhnotes` as explained in 7.13.6 p. 48 in order to prevent footnotes from overlapping the bottom margin.

19.1.2 `floatrow` package compatibility

The `floatrow` package must be loaded before the `reledmac`.

19.1.3 'No room for a new'

Sometimes, especially when using `reledmac` with other packages, you could obtain warning messages such 'no room for a new count' or 'no room for a new write'.

In order to prevent such problems, the first thing is to use the options to optimize `reledmac`. For example, if you need only two series of notes, use the `series={A,B}` option. Read 16.3 p. 68 in order to know which are the available options.

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

'no room for a new count' is often caused by `biblatex` being used at the same time. Load `reledmac` (and `reledpar`) *before* `biblatex`.

'no room for a new write' can be caused by 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.

19.1.4 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.

19.1.5 Paragraph shape

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

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

```
\setcounter{ballast}{100}
```

or some such figure, and with any luck the page breaks will settle down. Luckily, this problem does not crop up at all often.

19.1.6 Paragraphed footnotes

The restriction on explicit line-breaking in paragraphed footnotes, mentioned on 7.1 p. 37, and described in more detail on XII.6.3 p. 175, 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.

If you use more than one series of paragraphed notes, it may happen, in some particular cases, that only the footnote rule, with no accompanying footnotes, be printed. In this case use reledmac package option nopenalties which should solve the problem, but also may produce widow or orphan lines. For the time being, we have no solution of this problem.

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

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

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

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

Finally, in many cases you should use \Xmaxhnotes and / or \maxhnotesX (7.13.6 p. 48), in order to define the maximum height relative to \textheight and not to \vsize, because the \vsize value is not the same inside and outside of the preamble.

19.1.7 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[23] 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.

19.2 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 maintened by Maïeul Rouquette. You should use the capabilities of a modern TeX processor, like Xe(La)TeX

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

I Implementation overview

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

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

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

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

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

II Preliminaries

II.1 Links with original edmac

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

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

II.2 Package declaration

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

```
%<*code>
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{reledmac}[2017/06/08 v2.22.0 typesetting critical editions]%
%
```

II.3 Package options

\ifledfinal
\ifnocritical@
\if@noeled@sec
\ifnoend@
\ifnofamiliar@
\ifnoledgroup@
\ifparapparatus@
\ifnoquotation@
\iflednopbinverse
\ifparledgroup
\ifwidthliketwocolumns
\ifxindy@
\ifxindyhyperref@
\ifeledmaccompat@

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.

```
\RequirePackage{xkeyval}
%
```

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

```
\newif\ifparledgroup
%
```

And now, the options of reledmac.

```
\DeclareOptionX{series}[A,B,C,D,E]{\xdef\default@series{#1}}
\ExecuteOptionsX{series}%

\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}
```

```
\ExecuteOptionsX{final}

\newif\ifparapparatus@
\DeclareOptionX{parapparatus}{\parapparatus@true}

\newif\iflednopbinverse
\DeclareOptionX{nopbinverse}{\lednopbinversetrue}

\newif\ifwidthliketwocolumns%
\DeclareOptionX{widthliketwocolumns}{\widthliketwocolumnstrue}%

\newif\ifcontinuousnumberingwithcolumns
\DeclareOptionX{continuousnumberingwithcolumns}{\
continuousnumberingwithcolumnstrue}%

\newif\ifxindy@
\DeclareOptionX{xindy}[eledmac-markup-attr.xdy]{%
  \AtBeginDocument{\immediate\openout\eledmac@xindy@out=#1}%
  \newwrite\eledmac@xindy@out%
  \xindy@true%
  \gdef\eledmacmarkuplocrefdepth{:depth 1}%
  \AtEndDocument{\immediate\closeout\eledmac@xindy@out}%
}%

\newif\ifxindyhyperref@
\DeclareOptionX{xindy+hyperref}{%
  \xindyhyperref@true%
}%

\newif\ifeledmaccompat@%
\DeclareOptionX{eledmac-compat}{%
  \eledmaccompat@true%
}%
\DeclareOptionX{nopenalties}{%
  \AtBeginDocument{\let\add@penalties\relax}%
}
\def\l@auxdir{}%
\DeclareOptionX{auxdir}{%
  \xdef\l@auxdir{#1/}%
}%
%
```

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.

```
\ProcessOptionsX*\relax

%
```

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 XƎTEX is running, and ragged2e to manage ragged justification for paragraphed notes.

```
\RequirePackage{xargs}
\RequirePackage{etoolbox}
\@ifl@t@r\fmtversion{2015/10/01}
 {\ifboolexpr{not test{\@ifl@t@r\fmtversion{2016/03/31}} or (test{\ifdefstring{\fmtversion}{2016/03/31}} and test {\ifnumless{\patch@level}{3}})}%
   {\PackageWarning{reledmac}{You are using a LaTeX version older than 2016/03/31 patch 3.%
   \MessageBreak You are strongly encouraged to use a newer version.}}%
   {}%
 }%
 {\RequirePackage{etex}%
 \csname reserveinserts\endcsname{32}%
 }%
\RequirePackage{suffix}
\RequirePackage{xstring}
\RequirePackage{ifluatex}
\RequirePackage{ragged2e}
\RequirePackage{ifxetex}%
%
```

II.5 Compatibility with LuaTEX

Here, we enable some primitives for LuaTEX.

```
\ifx\directlua\undefined\else%
  \directlua{tex.enableprimitives("",{"textdir","pardir","bodydir"})}
\fi
%
```

II.6 Boolean flags

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

```
\newif\ifl@dmemoir
\@ifclassloaded{memoir}{\l@dmemoirtrue}{\l@dmemoirfalse}

%
```

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

```
\newif\if@ledgroup%
%
```

\ifl@imakeidx

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

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

\ifl@indextools

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

```
\newif\ifl@indextools%
\@ifpackageloaded{indextools}{%
  \l@indextoolstrue%
  \l@imakeidxtrue%
  \let\imki@wrindexentry\indtl@wrindexentry%
  }{}%
%
```

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`.

\ifl@footmisc

Define a flag if the `footmisc` package has been loaded.

```
\newif\ifl@footmisc
\@ifpackageloaded{footmisc}{\l@footmisctrue}{}%False is the default value
%
```

\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.

```
\ifdef{\if@RTL}{}{\newif\if@RTL}
%
```

\if@firstlineofpage

`\if@firstlineofpage` is set to TRUE at the first line of every page. `\if@firstlineofpageR` is for the right side.

```
\newif\if@firstlineofpage%
\newif\if@firstlineofpageR%
%
```

II.7 Messages

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

\reledmac@warning

Write a warning message.

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

\reledmac@error Write an error message.

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

\led@err@NumberingStarted
\led@err@NumberingNotStarted
\led@err@NumberingShouldHaveStarted

```
\newcommand*{\led@err@NumberingStarted}{%
  \reledmac@error{Numbering has already been started}{\@ehc}}
\newcommand*{\led@err@NumberingNotStarted}{%
  \reledmac@error{Numbering was not started}{\@ehc}}
\newcommand*{\led@err@NumberingShouldHaveStarted}{%
  \reledmac@error{Numbering should already have been started}{\@ehc}}
%
```

\led@err@edtextoutsidepstart

```
\newcommand*{\led@err@edtextoutsidepstart}{%
  \reledmac@error{\string\edtext\space outside numbered paragraph (\…pstart
\pend)}{\@ehc}}%
%
```

\led@mess@NotesChanged

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

\led@mess@SectionContinued

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

\led@err@LineationInNumbered

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

\led@warn@BadLineation
\led@warn@BadLinenummargin
\led@warn@BadLockdisp
\led@warn@BadSublockdisp

```
\newcommand*{\led@warn@BadLineation}{%
  \reledmac@warning{Bad \string\lineation\space argument}}
\newcommand*{\led@warn@BadLinenummargin}{%
  \reledmac@warning{Bad \string\linenummargin\space argument}}
\newcommand*{\led@warn@BadLockdisp}{%
  \reledmac@warning{Bad \string\lockdisp\space argument}}
\newcommand*{\led@warn@BadSublockdisp}{%
  \reledmac@warning{Bad \string\sublockdisp\space argument}}
%
```

\led@warn@NoFile

```
\newcommand*{\led@warn@NoFile}[1]{%
  \reledmac@warning{File `#1' not found}}
%
```

led@warn@LineFileObsolete

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

arn@BadAdvancelineSubline
d@warn@BadAdvancelineLine

```
\newcommand*{\led@warn@BadAdvancelineSubline}{%
  \reledmac@warning{\string\advanceline\space produced a sub-line
                number less than zero.}}
\newcommand*{\led@warn@BadAdvancelineLine}{%
  \reledmac@warning{\string\advanceline\space produced a line
                number less than zero.}}
%
```

\led@warn@BadSetline
\led@warn@BadSetlinenum

```
\newcommand*{\led@warn@BadSetline}{%
  \reledmac@warning{Bad \string\setline\space argument}}
\newcommand*{\led@warn@BadSetlinenum}{%
  \reledmac@warning{Bad \string\setlinenum\space argument}}
%
```

led@err@PstartNotNumbered
\led@err@PstartInPstart
\led@err@PendNotNumbered
\led@err@PendNoPstart
ed@err@AutoparNotNumbered
rr@NumberingWithoutPstart

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

\led@warn@BadAction

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

\led@warn@DuplicateLabel
\led@warn@AppLabelOutSecondArgEdtext
\led@warn@RefUndefined
\led@warn@RefUndefined

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

\led@warn@NoMarginpars

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

\led@warn@BadSidenotemargin

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

\led@warn@NoIndexFile

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

\led@warn@SeriesStillExist

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

\led@err@BadAction

```
\newcommand*{\led@err@StanzaIndentNotDefined}{%
  \reledmac@error{You have not defined the indentation for the line \number\stanza@count}{\@ehc}}%
%
```

\led@err@ManySidenotes
\led@err@ManyLeftnotes
\led@err@ManyRightnotes

```
\newcommand{\led@err@ManySidenotes}{%
  \ifledRcol@%
     \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%
}%
%
```

\led@err@TooManyColumns
\led@err@UnequalColumns
\led@err@LowStartColumn
\led@err@HighEndColumn
\led@err@ReverseColumns

```
\newcommand*{\led@err@TooManyColumns}{%
  \reledmac@error{Too many columns}{\@ehc}}
\newcommand*{\led@err@UnequalColumns}{%
  \reledmac@error{Number of columns is not equal to the number
                 in the previous row (or \protect\\ \space forgotten?)}{\@ehc}}
\newcommand*{\led@err@LowStartColumn}{%
  \reledmac@error{Start column is too low}{\@ehc}}
\newcommand*{\led@err@HighEndColumn}{%
  \reledmac@error{End column is too high}{\@ehc}}
\newcommand*{\led@err@ReverseColumns}{%
  \reledmac@error{Start column is greater than end column}{\@ehc}}
%
```

\led@err@toendnotes@outsidenumbering

```
\newcommand{\led@err@toendnotes@outsidenumbering}{%
  \reledmac@error{\string\toendnotes\space and related commands must be called inside a numbered texte (\string\…beginnumbering\string\endnumbering)}{\@ehc}%
}%
%
```

\led@err@EdtextWithoutFootnote

```
\newcommand{\led@err@EdtextWithoutFootnote}{%
  \reledmac@error{edtext without Xfootnote. Check syntaxis}{\@ehc}%
}%
%
```

\led@err@FootnoteNotInSecondArgEdtext

```
\newcommand{\led@err@FootnoteNotInSecondArgEdtext}[1]{%
  \reledmac@error{#1footnote outside of the second argument of an edtext. Check syntax}{\@ehc}%
}%
%
```

\led@error@PackageAfterEledmac

```
\newcommand{\led@error@PackageAfterEledmac}[1]{%
  \reledmac@error{#1 must be loaded before reledmac}{\@ehc}%
}%
%
```

\led@error@fail@patch@@makecol

```
\newcommand{\led@error@fail@patch@@makecol}{%
  \reledmac@error{Fail to patch \string\@makecol\space command}{\@ehc}%
}%
%
```

\led@error@fail@patch@@reinserts

```
\newcommand{\led@error@fail@patch@@reinserts}{%
  \reledmac@error{Fail to patch \string\@reinserts\space command}{\@ehc}%
}%
%
```

\led@error@fail@patch@@doclearpage

```
\newcommand{\led@error@fail@patch@@doclearpage}{%
  \reledmac@error{Fail to patch \string\@doclearpage\space command}{\@ehc}%
}%
%
```

\led@error@fail@patch@@iiiminipage

```
\newcommand{\led@error@fail@patch@@iiiminipage}{%
  \reledmac@error{Fail to patch \string\@iiiminipage\space command}{\@ehc}%
}%
%
```

or@fail@patch@endminipage

```
\newcommand{\led@error@fail@patch@endminipage}{%
  \reledmac@error{Failed to patch the \string\endminipage\space command}{\
@ehc}%
}%
%
```

or@fail@patch@endminipage

```
\newcommand{\led@error@fail@patch@makeindex}{%
  \reledmac@error{Failed to patch the \string\makeindex\space command}{\
@ehc}%
}%
%
```

n@edinde@outsidenumbering

```
\newcommand{\led@warn@edinde@outsidenumbering}{%
  \reledmac@warning{\string\edindex\space called outside of \string\
…beginnumbering\string\endnumbering. \MessageBreak Automatically switched
to \string\index.}%
}%
%
```

warning@hsizeX@deprecated

```
\newcommand{\led@warning@hsizeX@deprecated}{%
  \reledmac@warning{\string\hsizeX\space command deprecated, use \string\
widthX\space instead.}%
}%
%
```

warning@Xhsize@deprecated

```
\newcommand{\led@warning@Xhsize@deprecated}{%
  \reledmac@warning{\string\Xhsize\space command deprecated, use \string\
Xwidth\space instead.}%
}%
%
```

warning@msdatawithoutstop

```
\newcommand{\led@warning@msdatawithoutstop}{%
  \reledmac@warning{\string\msdata\space without corresponding \string\
stopmsdata}%
}%
%
```

ning@preXnotes@deprecated

```
\newcommand{\led@warning@preXnotes@deprecated}{%
  \reledmac@warning@preXnotes@deprecated%
}%
%
```

II.8 Gobbling

Here, we define some commands which gobble their arguments.

\@gobblethree
\@gobblefour
\@gobbleseven

```
\providecommand*{\@gobblethree}[3]{}
\providecommand*{\@gobblefour}[4]{}
\providecommand*{\@gobbleseven}[7]{}
%
```

II.9 Miscellaneous commands

\showlemma

\showlemma{⟨*lemma*⟩} typesets the lemma text in the body. It depends on the option.

```
\ifledfinal
  \newcommand*{\showlemma}[1]{#1}
\else
  \newcommand*{\showlemma}[1]{\underline{#1}}
\fi

%
```

\linenumberlist

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

Initialize it as \empty.

```
\let\linenumberlist=\empty

%
```

\@l@dtempcnta
\@l@dtempcntb

In imitation of LaTeX, we create a couple of scratch counters.

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).

```
\newcount\@l@dtempcnta \newcount\@l@dtempcntb
%
```

II.10 Prepare reledpar

\ifl@dpairing
\ifl@dpaging
\ifl@dprintingpages
\ifl@dprintingcolumns
\ifpst@rtedL
\l@dnumpstartsL

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

```
\newif\ifl@dpairing
\newif\ifl@dpaging%
\newif\ifl@dprintingpages%
\newif\ifl@dprintingcolumns%
\newif\ifpst@rtedL
```

```
\newcount\l@dnumpstartsL
%
```

\ifledRcol
\ifledRcol@

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

```
\newif\ifledRcol
\newif\ifledRcol@
%
```

\ifnumberingR

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

```
\newif\ifnumberingR
%
```

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

```
\newif\ifXnote@%
%
```

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

\ifindtl@innote
\ifindtl@notenumber

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

```
\providebool{indtl@innote}%
\providebool{indtl@notenumber}%
%
```

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. LaTeX will maintain and display a 'section number' as a count named \section@num that counts how many \beginnumbering and \resumenumbering commands have appeared; it need not be related to the logical divisions of your text.

\extensionchars

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

```
\newcount\section@num
\section@num=0
\let\extensionchars=\empty
%
```

\ifnumbering
\numberingtrue
\numberingfalse

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.

```
\newif\ifnumbering
%
```

\beginnumbering
\initnumbering@reg

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

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

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

```
\newcommand*{\beginnumbering}{%
  \ifnumbering
    \led@err@NumberingStarted
    \endnumbering
  \fi
  \global\numberingtrue
  \global\advance\section@num \@ne
  \initnumbering@reg
  \message{Section \the\section@num }%
  \line@list@stuff{\jobname.\extensionchars\the\section@num}%
  \l@dend@stuff
  \setcounter{pstart}{1}
  \ifl@dpairing
    \global\l@dnumpstartsL \z@
    \global\pst@rtedLfalse
%
```

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.

```
  \else
    \begingroup
    \global\@afterindenttrue%In order to retablish normal feature if the \begingroup was not here
    \initnumbering@quote
    \ifwidthliketwocolumns%
       \csuse{setwidthliketwocolumns@\columns@position}%
       \csuse{setpositionliketwocolumns@\columns@position}%
    \fi%
  \fi
  \gdef\eled@sections@@{}%
  \if@noeled@sec\else%
    \makeatletter%
    \InputIfFileExists%
      {\l@auxdir\jobname.eledsec\the\section@num}%
      {}%
      {\led@warn@NoFile{\l@auxdir\jobname.eledsec\the\section@num}}%
    \makeatother%
    \immediate\openout\eled@sectioning@out=\l@auxdir\jobname.eledsec\the\section@num\relax%
  \fi%
}
\newcommand*{\initnumbering@reg}{%
  \global\pst@rtedLfalse
  \global\l@dnumpstartsL \z@
  \global\absline@num \z@
  \gdef\normal@page@break{}
  \gdef\l@prev@pb{}
  \gdef\l@prev@nopb{}
  \global\line@num \z@
  \global\subline@num \z@
  \global\@lock \z@
  \global\sub@lock \z@
  \global\sublines@false
  \global\let\next@page@num=\relax
  \global\let\sub@change=\relax
  \resetprevline@
  \resetprevpage@num
  \global\stopmsdata@inserted@true%
  \global\let\@msdata@list\relax%
  \global\csundef{@msdata@\add@msd@c @data}%
  }

%
```

\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.

```
\def\endnumbering{%
  \ifnumbering
    \global\numberingfalse
    \normal@pars
    \ifnum\l@dnumpstartsL=0%
      \led@err@NumberingWithoutPstart%
    \fi%
    \ifl@dpairing
      \global\pst@rtedLfalse
    \else
      \ifx\insertlines@list\empty\else
        \global\noteschanged@true
      \fi
      \ifx\line@list\empty\else
        \global\noteschanged@true
      \fi
    \fi
    \ifnoteschanged@
      \led@mess@NotesChanged
    \fi
  \else
    \led@err@NumberingNotStarted
  \fi
  \autoparfalse
  \if@noeled@sec\else%
    \immediate\closeout\eled@sectioning@out%
  \fi%
  \ifl@dpairing\else
    \global\l@dnumpstartsL=\z@%
    \endgroup
  \fi
}
%
```

\pausenumbering
\resumenumbering

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.[24]

```
\newcommand{\pausenumbering}{%
  \ifautopar\global\autopar@pausetrue\fi%
  \endnumbering\global\numberingtrue}
%
```

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

[24] Peter Wilson's thanks to Wayne Sullivan, who suggested the idea behind these macros.

that no check is made by \resumenumbering to ensure that \pausenumbering was actually invoked.

```
\newcommand*{\resumenumbering}{%
  \ifnumbering
     \ifautopar@pause\autopar\fi
     \global\pst@rtedLtrue
     \global\advance\section@num \@ne
     \led@mess@SectionContinued{\the\section@num}%
     \line@list@stuff{\jobname.\extensionchars\the\section@num}%
     \l@dend@stuff
     \ifl@dpairing\else%
        \begingroup%
        \initnumbering@quote%
        \ifwidthliketwocolumns%
          \csuse{setwidthliketwocolumns@\columns@position}%
          \csuse{setpositionliketwocolumns@\columns@position}%
        \fi%
     \fi%
     \ifcontinuousnumberingwithcolumns%
       \ifdefined\line@numR%
         \ifnum\line@numR>\line@num%
           \expandafter\setlinenum\expandafter{\the\line@numR}%
         \fi%
         \ifnum\last@page@numR>\last@page@num%
           \global\last@page@num=\last@page@numR%
         \fi%
       \fi%
     \fi%
  \else
    \led@err@NumberingShouldHaveStarted
    \endnumbering
    \beginnumbering
  \fi}


%
```

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

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

It could be nice to unify them to the LaTeX3 list, however such migration would take quite time with some risk of error, for a gain which will be minor.

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

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

\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 .

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

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

```
\newtoks\led@toksa \newtoks\led@toksb
\global\led@toksa={\\}
\long\def\xright@appenditem#1\to#2{%
  \global\led@toksb=\expandafter{#2}%
  \xdef#2{\the\led@toksb\the\led@toksa\expandafter{#1}}%
  \global\led@toksb={}}
%
```

\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.

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

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

```
\def\gl@p#1\to#2{\expandafter\gl@poff#1\gl@poff#1#2}
\long\def\gl@poff\\#1#2\gl@poff#3#4{\gdef#4{#1}\gdef#3{#2}}

%
```

V Line counting

V.1 Choosing the system of lineation

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

\ifbypstart@
\bypstart@true
\bypstart@false
\ifbypage@
\bypage@true
\bypage@false

The \ifbypage@ and \ifbypstart@ flag specifie 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.

```
\newif\ifbypage@
\newif\ifbypstart@
%
```

The \ifbypage@R and \ifbypstart@R flag specifie 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@R
\ifbypstart@R

```
\newif\ifbypage@R
\newif\ifbypstart@R
%
```

\lineation

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

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

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

```
  \ifnumbering
    \led@err@LineationInNumbered
  \else
%
```

If the argument is page.

```
    \def\@tempa{#1}\def\@tempb{page}%
    \ifx\@tempa\@tempb
        \global\bypage@true
        \global\bypstart@false
        \unless\ifnocritical@%
          \Xpstart[][false]%
        \fi%
%
```

If the argument is pstart.

```
    \else
       \def\@tempb{pstart}%
       \ifx\@tempa\@tempb
           \global\bypage@false
           \global\bypstart@true
           \unless\ifnocritical@%
             \Xpstart%
           \fi%
%
```

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

```
       \else
          \def\@tempb{section}
          \ifx\@tempa\@tempb
            \global\bypage@false
            \global\bypstart@false
            \unless\ifnocritical@%
              \Xpstart[][false]%
            \fi%
%
```

In other case, it is an error.

```
          \else
            \led@warn@BadLineation
          \fi
       \fi
    \fi
  \fi}}
%
```

V.2 Line number margin

\linenummargin
\line@margin
\l@dgetline@margin

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

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

```
\newcount\line@margin%
\newcount\line@margin@columns%Only for parallel typesetting
```

```
\line@margin@columns=\m@ne%

\newcommand*{\linenummargin}[1]{{%
  \l@dgetline@margin{#1}%
  \ifnum\@l@dtempcntb>\m@ne
    \ifledRcol
      \global\line@marginR=\@l@dtempcntb
      \led@warn@setting@in@rightside{\linenummargin}%
    \else
      \global\line@margin=\@l@dtempcntb
    \fi
  \fi}}

\newcommand*{\l@dgetline@margin}[1]{%
  \def\@tempa{#1}\def\@tempb{left}%
  \ifx\@tempa\@tempb
    \@l@dtempcntb \z@
  \else
     \def\@tempb{right}%
     \ifx\@tempa\@tempb
       \@l@dtempcntb \@ne
     \else
       \def\@tempb{outer}%
       \ifx\@tempa\@tempb
         \@l@dtempcntb \tw@
       \else
         \def\@tempb{inner}%
         \ifx\@tempa\@tempb
           \@l@dtempcntb \thr@@
         \else
           \led@warn@BadLinenummargin
           \@l@dtempcntb \m@ne
         \fi
       \fi
     \fi
  \fi}

%
```

V.3 Line number initialization and increment

\c@firstlinenum
\c@linenumincrement

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.

```
\newcounter{firstlinenum}
  \setcounter{firstlinenum}{5}
```

```
\newcounter{linenumincrement}
  \setcounter{linenumincrement}{5}
%
```

\c@firstsublinenum
\c@sublinenumincrement

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

```
\newcounter{firstsublinenum}
  \setcounter{firstsublinenum}{5}
\newcounter{sublinenumincrement}
  \setcounter{sublinenumincrement}{5}

%
```

\firstlinenum
\linenumincrement
\firstsublinenum
\sublinenumincrement

These macros can be used to set the corresponding counters.

```

\newcommand*{\firstlinenum}[1]{%
  \ifledRcol%
    \setcounter{firstlinenumR}{#1}%
    \led@warn@setting@in@rightside{\firstlinenum}%
  \else%
    \setcounter{firstlinenum}{#1}%
  \fi%
}
\newcommand*{\linenumincrement}[1]{%
  \ifledRcol%
    \setcounter{linenumincrementR}{#1}%
    \led@warn@setting@in@rightside{\linenumincrement}%
  \else%
    \setcounter{linenumincrement}{#1}%
  \fi%
}
\newcommand*{\firstsublinenum}[1]{%
  \ifledRcol%
    \setcounter{firstsublinenumR}{#1}%
    \led@warn@setting@in@rightside{\firstsublinenum}%
  \else%
    \setcounter{firstsublinenum}{#1}%
  \fi%
}
\newcommand*{\sublinenumincrement}[1]{%
  \ifledRcol%
    \setcounter{sublinenumincrementR}{#1}%
    \led@warn@setting@in@rightside{\sublinenumincrement}%
  \else%
    \setcounter{sublinenumincrement}{#1}%
  \fi%
}

```

```
%
```

V.4 Line number locking

\lockdisp
\lock@disp
\l@dgetlock@disp

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.

```
\newcount\lock@disp
\newcommand{\lockdisp}[1]{{%
  \l@dgetlock@disp{#1}%
  \ifnum\@l@dtempcntb>\m@ne
    \global\lock@disp=\@l@dtempcntb
  \else
    \led@warn@BadLockdisp
  \fi}}
\newcommand*{\l@dgetlock@disp}[1]{
  \def\@tempa{#1}\def\@tempb{first}%
  \ifx\@tempa\@tempb
    \@l@dtempcntb \z@
  \else
     \def\@tempb{last}%
     \ifx\@tempa\@tempb
       \@l@dtempcntb \@ne
     \else
       \def\@tempb{all}%
       \ifx\@tempa\@tempb
         \@l@dtempcntb \tw@
       \else
         \@l@dtempcntb \m@ne
       \fi
     \fi
  \fi}

%
```

\sublockdisp
\sublock@disp

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.

```
\newcount\sublock@disp
\newcommand{\sublockdisp}[1]{{%
  \l@dgetlock@disp{#1}%
  \ifnum\@l@dtempcntb>\m@ne
    \global\sublock@disp=\@l@dtempcntb
  \else
    \led@warn@BadSublockdisp
  \fi}}
```

```

%
```

V.5 Line number style

\linenumberstyle
\linenumrep
\linenumr@p
\sublinenumberstyle
\sublinenumrep
\sublinenumr@p

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

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

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

```
\newcommand*{\linenumberstyle}[1]{%
  \def\linenumrep##1{\@nameuse{@#1}{##1}}}
\newcommand*{\sublinenumberstyle}[1]{%
  \def\sublinenumrep##1{\@nameuse{@#1}{##1}}}
%
```

Initialise the number styles to arabic.

```
\linenumberstyle{arabic}
  \let\linenumr@p\linenumrep
\sublinenumberstyle{arabic}
  \let\sublinenumr@p\sublinenumrep

%
```

V.6 Line number printing

\leftlinenum
\rightlinenum
\linenumsep
\numlabfont
\ledlinenum

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

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

\ledlinenum typesets the line (and subline) number.

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

```
\newlength{\linenumsep}
  \setlength{\linenumsep}{1pc}
\newcommand*{\numlabfont}{\normalfont\scriptsize}
```

```
\newcommand*{\ledlinenum}{%
  \bgroup%
  \ifluatex%
    \textdir TLT%
  \fi%
  \numlabfont\linenumrep{\line@num}%
  \ifsublines@
    \ifnum\subline@num>0\relax
      \unskip%
      \Xsublinesep@side%
      \sublinenumrep{\subline@num}%
    \fi
  \fi%
  \egroup%
}%

\newcommand*{\leftlinenum}{%
  \ledlinenum
  \kern\linenumsep}
\newcommand*{\rightlinenum}{%
  \kern\linenumsep
  \ledlinenum}

%
```

V.7 Line number counters and lists

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

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

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

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

```
\newcount\line@num
%
```

\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.

```
\newcount\subline@num
%
```

\ifsublines@
\sublines@true
\sublines@false

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

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.

```
\newif\ifsublines@
%
```

\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.

```
\newcount\absline@num
%
```

We will call \absline@num numbers "absolute" numbers, and \line@num and \subline@num numbers "visible" numbers.

V.8 Line number locking counter

\@lock
\sub@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.

```
\newcount\@lock
\newcount\sub@lock
%
```

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/cmr/m/n`.

  There is one item in this list for every lemma marked by `\edtext`, even if there are several notes to that lemma, or no notes at all. `\edtext` reads the data in this list, making it available for use in the text of notes.

- `\insertlines@list`: the line numbers of lines that have footnotes or other insertions. These are the absolute numbers where the corresponding lemmas begin. This list contains one entry for every footnote in the section; one lemma may contribute no footnotes or many footnotes. This list is used by `\add@inserts` within `\do@line`, to tell it where to insert notes.

- `\actionlines@list`: a list of absolute line numbers at which we are to perform special actions; these actions are specified by the `\actions@list` list defined below.

- `\actions@list`: action codes corresponding to the line numbers in `\actionlines@list`. These codes tell `reledmac` what action it is supposed to take at each of these lines. One action, the page-start action, is generated behind the scenes by `reledmac` itself; the others, for specifying sub-lineation, line-number locking, and line-number alteration, are generated only by explicit commands in your input file. The page-start and line-number-alteration actions require arguments, to specify the new values for the page or line numbers; instead of storing those arguments in another list, we have chosen the action-code values so that they can encode both the action and the argument in these cases. Action codes greater than -1000 are page-start actions, and the code value is the page number; action codes less than -5000 specify line numbers, and the code value is a transformed version of the line number; action codes between these two values specify other actions which require no argument.

  Here is the full list of action codes and their meanings:

  Any number greater than -1000 is a page-start action: the line number associated with it is the first line on a page, and the action number is the page number. (The cutoff of -1000 is chosen because negative page-number values are used by some macro packages; we assume that page-number values less than -1000 are not common.) Page-start action codes are added to the list by the `\page@action` macro, which is (indirectly) triggered by the workings of the `\page@start` macro; that macro should always be called in the output routine, just before the page contents are assembled. `Eledmac` calls it in `\pagecontents`.

The action code -1001 specifies the start of sub-lineation: meaning that, starting with the next line, we should be advancing `\subline@num` at each start-of-line command, rather than `\line@num`.

The action code -1002 specifies the end of sub-lineation. At the next start-of-line, we should clear the sub-line counter and start advancing the line number. The action codes for starting and ending sub-lineation are added to the list by the `\sub@action` macro, as called to implement the `\startsub` and `\endsub` macros.

The action code -1003 specifies the start of line number locking. After the number for the current line is computed, it will remain at that value through the next line that has an action code to end locking.

The action code -1004 specifies the end of line number locking.

The action code -1005 specifies the start of sub-line number locking. After the number for the current sub-line is computed, it will remain at that value through the next sub-line that has an action code to end locking.

The action code -1006 specifies the end of sub-line number locking.

The four action codes for line and sub-line number locking are added to the list by the `\do@lockon` and `\do@lockoff` macros, as called to implement the `\startlock` and `\endlock` macros.

An action code of -5000 or less sets the current visible line number (either the line number or the sub-line number, whichever is currently being advanced) to a specific positive value. The value of the code is $-(5000 + n)$, where n is the value (always ≥ 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:

```
\list@create{\line@list}
\list@create{\insertlines@list}
\list@create{\actionlines@list}
\list@create{\actions@list}

%
```

\page@num
\endpage@num
\endline@num
\endsubline@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.

```
\newcount\page@num
\newcount\endpage@num
\newcount\endline@num
\newcount\endsubline@num
%
```

\ifnoteschanged@
\noteschanged@true
\noteschanged@false

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.

```
\newif\ifnoteschanged@
%
```

\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@

```
\newcommand*{\resetprevline@}{%
    \def\do##1{\global\csundef{prevline##1}}%
    \dolistloop{\@series}%
}
%
```

\resetprevpage@num

Inside the apparatus, at each note, the page number is stored in a macro called \prevpageX@num, where X is the letter of the current series. This macro is called when using \Xparafootsep or \parafootsepX. This macro must be reset at the beginning of each numbered section The \resetprevpage@ command resets this macro for every series.

\resetprevpage@

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

```
        \global\csname ##1prevpage@num\endcsname=\z@%
        \global\csname ##1prevpage@numR\endcsname=\z@%
        }%
        {}%
      }%
    \dolistloop{\@series}%
}
%
```

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.

```
\newread\@inputcheck
\newcommand*{\read@linelist}[1]{%
  \ifledRcol%
    \list@clearing@regR%
  \else%
    \list@clearing@reg%
  \fi%
%
```

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).

```
  \list@clear{\maxlinesinpar@list}
%
```

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

Our line, page, and line-locking counters were already zeroed by \line@list@stuff if this is being called from within \beginnumbering; sub-lineation will be turned off as well in that case. On the other hand, if this is being called from \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.

```
  \get@linelistfile{#1}%
  \ifcontinuousnumberingwithcolumns
      \global\page@numR=\page@numR\relax
       \global\last@page@numR=\last@page@numR\relax
       \global\page@num=\page@num\relax
       \global\last@page@num=\last@page@num\relax
  \fi
  \@stopmsd%Security if last \endms{} is forgotten
  \unless\ifledRcol%Get the last line of the last page
    \csnumgdef{@lastabsline@forpage@\the\page@num}{\the\absline@num}%
    \csnumgdef{@lastline@forpage@\the\page@num}{\the\line@num}%
  \else%
    \csnumgdef{@lastabsline@forpageR@\the\page@numR}{\the\absline@numR}%
    \csnumgdef{@lastline@forpageR@\the\page@numR}{\the\line@numR}%
  \fi%
  \endgroup
%
```

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.

```
  \ifledRcol
    \global\page@numR=\m@ne
    \ifx\actionlines@listR\empty
      \gdef\next@actionlineR{1000000}%
    \else
      \gl@p\actionlines@listR\to\next@actionlineR
      \gl@p\actions@listR\to\next@actionR
    \fi
  \else
    \global\page@num=\m@ne
    \ifx\actionlines@list\empty
      \gdef\next@actionline{1000000}%
    \else
      \gl@p\actionlines@list\to\next@actionline
      \gl@p\actions@list\to\next@action
    \fi
  \fi
}
%
```

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

```
\newcommand*{\list@clearing@reg}{%
  \list@clear{\line@list}%
  \list@clear{\insertlines@list}%
  \list@clear{\actionlines@list}%
  \list@clear{\actions@list}%
  \list@clear{\linesinpar@listL}%
  \list@clear{\linesonpage@listL}%
  }%
%
```

\get@linelistfile reledmac can take advantage of the LaTeX 'safe file input' macros to get the line-list file.

```
\newcommand*{\get@linelistfile}[1]{%
  \InputIfFileExists{\l@auxdir#1}{%
    \global\noteschanged@false
    \begingroup
      \catcode`\[=1 \catcode`\]=2
      \makeatletter \catcode`\^^M=9}{%
    \led@warn@NoFile{\l@auxdir#1}%
    \global\noteschanged@true
    \begingroup}%
}

%
```

This version of \read@linelist creates list macros containing data for the entire section, so they could get rather large. It would be no more difficult to read the line-list file incrementally rather than all at once: we could read, at the start of each paragraph, only the commands relating to that paragraph. But this would require that we have two line-lists open at once, one for reading, one for writing, and on systems without version numbers we would have to do some file renaming outside of LaTeX for that to work. We have retained this slower approach to avoid that sort of hacking about, but have provided the \pausenumbering and \resumenumbering macros to help you if you run into macro memory limitations (see 5.2.7 p. 20 above).

V.11 Commands within the line-list file

This section defines the commands that can appear within a line-list file. They all have very short names because we are likely to be writing very large numbers of them out. One macro, \@nl, is especially short, since it will be written to the line-list file once for every line of text in a numbered section. (Another of these commands, \@lab, will be introduced in a later section, among the cross-referencing commands it is associated with.)

When these commands modify the various page and line counters, they deliberately do not use \global. This is because we want them to affect only the counter values within the current group when nested calls of \@ref occur. (The code assumes throughout that the value of \globaldefs is zero.)

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

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

```
\newcommand{\line@list@version}[1]{%
  \IfStrEq{#1}{\this@line@list@version}%
    {}%
    {\ifledRcol%
       \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
     \else%
       \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
     \fi%
     \endinput%
    }%
}%
%
```

\@nl
\@nl@reg

\@nl does everything related to the start of a new line of numbered text.

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.

```

\newcommand*{\@nl}[2]{%
  \fix@page{#1}%
  \ifledRcol%
    \@nl@regR%
  \else%
    \@nl@reg%
  \fi%
}
\newcommand*{\@nl@reg}{%
  \ifx\l@dchset@num\relax \else
    \advance\absline@num \@ne
    \set@line@action
    \let\l@dchset@num=\relax
    \advance\absline@num \m@ne
    \advance\line@num \m@ne
  \fi
%
```

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

```
  \advance\absline@num \@ne
         \ifx\next@page@num\relax \else
             \page@action
             \let\next@page@num=\relax
         \fi
         \ifx\sub@change\relax \else
            \ifnum\sub@change>\z@
               \sublines@true
            \else
               \sublines@false
            \fi
            \sub@action
            \let\sub@change=\relax
         \fi
%
```

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

```
         \ifcase\@lock
            \or
               \@lock \tw@
            \or \or
               \@lock \z@
         \fi
         \ifcase\sub@lock
            \or
               \sub@lock \tw@
            \or \or
               \sub@lock \z@
         \fi
%
```

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

```
         \ifsublines@
              \ifnum\sub@lock<\tw@
                \advance\subline@num \@ne
              \fi
         \else
              \ifnum\@lock<\tw@
                \advance\line@num \@ne \subline@num \z@
              \fi
         \fi}

%
```

\last@page@num
\fix@page

\fix@page basically replaces \@page. It determines whether or not a new page has been started, based on the page values held by \@nl.

```
\newcount\last@page@num
  \last@page@num=-10000

\newcommand*{\fix@page}[1]{%
  \ifledRcol
```

```
    \ifnum #1=\last@page@numR
    \else
      \csnumgdef{@lastabsline@forpageR@\the\page@numR}{\the\absline@numR}%
      \csnumgdef{@lastline@forpageR@\the\page@numR}{\the\line@numR}%
      \ifbypage@R
        \line@numR \z@ \subline@numR \z@
      \fi
      \page@numR=#1\relax
      \last@page@numR=#1\relax
      \def\next@page@numR{#1}%
    \fi
  \else
    \ifnum #1=\last@page@num
    \else
      \csnumgdef{@lastabsline@forpage@\the\page@num}{\the\absline@num}%
      \csnumgdef{@lastline@forpage@\the\page@num}{\the\line@num}%
      \ifbypage@
        \line@num \z@ \subline@num \z@
      \fi
      \page@num=#1\relax
      \last@page@num=#1\relax
      \def\next@page@num{#1}%
      \listxadd{\normal@page@break}{\the\absline@num}
    \fi
  \fi}
%
```

\@pend
\@pendR
\@lopL
\@lopR

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.

```
\newcommand*{\@pend}[1]{}
\newcommand*{\@pendR}[1]{}
\newcommand*{\@lopL}[1]{}
\newcommand*{\@lopR}[1]{}

%
```

\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 `\@nl` of the necessary action.

```
\newcommand*{\sub@on}{\ifsublines@
     \let\sub@change=\relax
  \else
     \def\sub@change{1}%
  \fi}
\newcommand*{\sub@off}{\ifsublines@
     \def\sub@change{-1}%
  \else
```

```
    \let\sub@change=\relax
  \fi}

%
```

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

```

\newcommand*{\@adv}[1]{%
  \ifsublines@
    \ifledRcol
      \advance\subline@numR by #1\relax
      \ifnum\subline@numR<\z@
        \led@warn@BadAdvancelineSubline
        \subline@numR \z@
      \fi
    \else
      \advance\subline@num by #1\relax
      \ifnum\subline@num<\z@
        \led@warn@BadAdvancelineSubline
        \subline@num \z@
      \fi
    \fi
  \else
    \ifledRcol
      \advance\line@numR by #1\relax
      \ifnum\line@numR<\z@
        \led@warn@BadAdvancelineLine
        \line@numR \z@
      \fi
    \else
      \advance\line@num by #1\relax
      \ifnum\line@num<\z@
        \led@warn@BadAdvancelineLine
        \line@num \z@
      \fi
    \fi
  \fi
  \set@line@action}

%
```

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

```

\newcommand*{\@set}[1]{%
  \ifledRcol
    \ifsublines@
```

```
      \subline@numR=#1\relax
    \else
      \line@numR=#1\relax
    \fi
    \set@line@action
  \else
    \ifsublines@
      \subline@num=#1\relax
    \else
      \line@num=#1\relax
    \fi
    \set@line@action
  \fi}

%
```

\l@d@set \l@dchset@num 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 is a flag to the \@nl? macro. If it is not \relax then a linenumber change is to be done.

```

\newcommand*{\l@d@set}[1]{%
  \ifledRcol
    \line@numR=#1\relax
    \advance\line@numR \@ne
    \def\l@dchset@num{#1}
  \else
    \line@num=#1\relax
    \advance\line@num \@ne
    \def\l@dchset@num{#1}
  \fi}
\let\l@dchset@num\relax

%
```

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

```

\newcommand*{\page@action}{%
  \ifledRcol
    \xright@appenditem{\the\absline@numR}\to\actionlines@listR
    \xright@appenditem{\next@page@numR}\to\actions@listR
  \else
    \xright@appenditem{\the\absline@num}\to\actionlines@list
    \xright@appenditem{\next@page@num}\to\actions@list
  \fi}
%
```

\set@line@action

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

```

\newcommand*{\set@line@action}{%
  \ifledRcol
    \xright@appenditem{\the\absline@numR}\to\actionlines@listR
    \ifsublines@
       \@l@dtempcnta=-\subline@numR
    \else
       \@l@dtempcnta=-\line@numR
    \fi
    \advance\@l@dtempcnta by -5000\relax
    \xright@appenditem{\the\@l@dtempcnta}\to\actions@listR
  \else
    \xright@appenditem{\the\absline@num}\to\actionlines@list
    \ifsublines@
       \@l@dtempcnta=-\subline@num
    \else
       \@l@dtempcnta=-\line@num
    \fi
    \advance\@l@dtempcnta by -5000\relax
    \xright@appenditem{\the\@l@dtempcnta}\to\actions@list
  \fi}
%
```

\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.

```

\newcommand*{\sub@action}{%
  \ifledRcol
    \xright@appenditem{\the\absline@numR}\to\actionlines@listR
    \ifsublines@
      \xright@appenditem{-1001}\to\actions@listR
    \else
      \xright@appenditem{-1002}\to\actions@listR
    \fi
  \else
    \xright@appenditem{\the\absline@num}\to\actionlines@list
    \ifsublines@
      \xright@appenditem{-1001}\to\actions@list
    \else
      \xright@appenditem{-1002}\to\actions@list
    \fi
  \fi}
%
```

\lock@on
\do@lockon
\do@lockonL

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

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

```
\newcommand*{\lock@on}{\futurelet\next\do@lockon}

\newcommand*{\do@lockon}{%
  \ifx\next\lock@off
    \global\let\lock@off=\skip@lockoff
  \else
    \ifledRcol
      \do@lockonR
    \else
      \do@lockonL
    \fi
  \fi}


\newcommand*{\do@lockonL}{%
  \xright@appenditem{\the\absline@num}\to\actionlines@list
  \ifsublines@
    \xright@appenditem{-1005}\to\actions@list
    \ifnum\sub@lock=\z@
      \sub@lock \@ne
    \else
      \ifnum\sub@lock=\thr@@
        \sub@lock \@ne
      \fi
    \fi
  \else
    \xright@appenditem{-1003}\to\actions@list
    \ifnum\@lock=\z@
      \@lock \@ne
    \else
      \ifnum\@lock=\thr@@
        \@lock \@ne
      \fi
    \fi
  \fi}

%
```

\lock@off
\do@lockoff
\do@lockoffL
\skip@lockoff

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

```
\newcommand*{\do@lockoffL}{%
  \xright@appenditem{\the\absline@num}\to\actionlines@list
  \ifsublines@
    \xright@appenditem{-1006}\to\actions@list
    \ifnum\sub@lock=\tw@
```

```
      \sub@lock \thr@@
    \else
      \sub@lock \z@
    \fi
  \else
    \xright@appenditem{-1004}\to\actions@list
    \ifnum\@lock=\tw@
      \@lock \thr@@
    \else
      \@lock \z@
    \fi
  \fi}

\newcommand*{\do@lockoff}{%
  \ifledRcol
    \do@lockoffR
  \else
    \do@lockoffL
  \fi}
\newcommand*{\skip@lockoff}{\global\let\lock@off=\do@lockoff}
\global\let\lock@off=\do@lockoff

%
```

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

```
\newcommand*{\n@num}{%
  \ifledRcol%
    \xright@appenditem{\the\absline@numR}\to\actionlines@listR
    \xright@appenditem{-1007}\to\actions@listR
  \else%
    \xright@appenditem{\the\absline@num}\to\actionlines@list%
    \xright@appenditem{-1007}\to\actions@list%
  \fi%
}%

%
```

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

```
\newcommand*{\n@num@stanza}{%
  \ifledRcol%
    \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
    \xright@appenditem{-1008}\to\actions@listR%
  \else%
    \xright@appenditem{\the\absline@num}\to\actionlines@list%%
    \xright@appenditem{-1008}\to\actions@list%
  \fi%
}
%
```

\ifl@dhidenumber
\hidenumbering
\hidenumberingonleftpage
\hidenumberingonrightpage
\hide@num

\hidenumbering hides number in margin. It uses action code 1009. \hidenumberingonleftpage and \hidenumberingonrightpage ara variant, using action code only conditionnaly

```
\newif\ifl@dhidenumber
\newcommand*{\hidenumbering}{
  \ifledRcol%
    \write\linenum@outR{\string\hide@num}%
  \else%
    \write\linenum@out{\string\hide@num}%
  \fi%
}%
\newcommand*{\hide@num}{%
  \ifledRcol%
    \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
    \xright@appenditem{-1009}\to\actions@listR%
  \else%
    \xright@appenditem{\the\absline@num}\to\actionlines@list%%
    \xright@appenditem{-1009}\to\actions@list%
  \fi%
}
\newcommand*{\hidenumberingonleftpage}{%
  \ifledRcol%
    \write\linenum@outR{\string\hide@num@left}%
  \else%
    \write\linenum@out{\string\hide@num@left}%
  \fi%
}%

\newcommand*{\hide@num@left}{%
  \ifledRcol%
    \ifodd\page@numR\else%
      \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
      \xright@appenditem{-1009}\to\actions@listR%
    \fi%
  \else%
    \ifodd\page@num\else%
      \xright@appenditem{\the\absline@num}\to\actionlines@list%%
      \xright@appenditem{-1009}\to\actions@list%
    \fi%
  \fi%
}%

\newcommand*{\hidenumberingonrightpage}{%
  \ifledRcol%
    \write\linenum@outR{\string\hide@num@right}%
  \else%
```

```
    \write\linenum@out{\string\hide@num@right}%
  \fi%
}%

\newcommand*{\hide@num@right}{%
  \ifledRcol%
    \ifodd\page@numR%
      \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
      \xright@appenditem{-1009}\to\actions@listR%
    \fi%
  \else%
    \ifodd\page@num%
      \xright@appenditem{\the\absline@num}\to\actionlines@list%%
      \xright@appenditem{-1009}\to\actions@list%
    \fi%
  \fi%
}%

%
```

\@ref \insert@count `\@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`.

```
\newcount\insert@count
%
```

- `#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.

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

\@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.

```
\newcommand*{\@ref}[2]{%
  \ifledRcol%
    \@ref@regR{#1}{#2}%
  \else%
    \@ref@reg{#1}{#2}%
  \fi%
}%
```

```
\newcommand*{\@ref@reg}[2]{%
  \global\insert@count=#1\relax
  \global\advance\@edtext@level by 1%
  \loop\ifnum\insert@count>\z@
    \xright@appenditem{\the\absline@num}\to\insertlines@list
    \global\advance\insert@count \m@ne
  \repeat
%
```

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.

```
  \begingroup
    \let\@ref=\dummy@ref
    \let\@lopL\@gobble
    \let\page@action=\relax
    \let\sub@action=\relax
    \let\set@line@action=\relax
    \let\@lab=\relax
    \let\@lemma=\relax%
    \let\@sw\@gobblethree%
    #2
    \global\endpage@num=\page@num
    \global\endline@num=\line@num
    \global\endsubline@num=\subline@num
  \endgroup
%
```

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

```
    \xright@appenditem%
      {\the\page@num|\the\line@num|%
       \ifsublines@ \the\subline@num \else 0\fi|%
       \the\endpage@num|\the\endline@num|%
       \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@list
%
```

And now, call `\@ref@reg@parsearg`, which can be also called by `\@ref@later`

```
     \@ref@reg@parse{#2}%
%
```

Decrease edtext level counter.

```
  \global\advance\@edtext@level by -1%
}
%
```

`\@ref@reg@parse` The `\@ref@reg@parsearg` command parses the second argument of a `\@ref` or the unique argument of `\@ref@later` written in the auxiliary fill.

First, 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.

```
\newcommand{\@ref@reg@parse}[1]{%
     \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\@edtext@level\endcsname}%
     \providebool{lemmacommand@\the\@edtext@level}%
     \boolfalse{lemmacommand@\the\@edtext@level}%
%
```

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

```
  #1%
%
```

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.

```
  \ifnum\@edtext@level>0%
    \def\create@this@edtext@level{\expandafter\list@create\expandafter{\csname sw@list@edtext@\the\@edtext@level\endcsname}}%
    \ifcsundef{sw@list@edtext@\the\@edtext@level}{\create@this@edtext@level}{}%
    \letcs{\@tmp}{sw@list@edtext@\the\@edtext@level}%
    \letcs{\@tmpp}{sw@list@edtext@tmp@\the\@edtext@level}
    \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
    \global\cslet{sw@list@edtext@\the\@edtext@level}{\@tmp}%
  \fi%
%
```

```
}

%
```

`\ref@reg@later` This macro is stored in the auxiliary file when using `\edtextlater`. It is used only to get the correct value for the `\sameword` tools.

```
\newcommand{\@ref@later}[1]{%
  \global\advance\@edtext@level by \@ne%
  \ifledRcol%
    \@ref@reg@parseR{#1}%
  \else%
    \@ref@reg@parse{#1}%
  \fi%
  \global\advance\@edtext@level by -\@ne%
}%
%
```

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`.

```
\newwrite\linenum@out
%
```

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

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

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

```
\newif\iffirst@linenum@out@
  \first@linenum@out@true
%
```

`\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.

```
\newcommand{\this@line@list@version}{6}%
%
```

`\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.

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

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

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

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.

```
  \iffirst@linenum@out@
     \immediate\closeout\linenum@out%
     \global\first@linenum@out@false%
     \immediate\openout\linenum@out=\l@auxdir#1\relax%
     \immediate\write\linenum@out{\string\line@list@version{\
this@line@list@version}}%
     \ifl@dpaging%
       \immediate\write\linenum@out{\string\@par@sync@option{\
@par@this@sync@option}}%
     \fi%
  \else
%
```

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.

```
     \if@minipage%
       \leavevmode%
     \fi%
     \closeout\linenum@out%
     \openout\linenum@out=\l@auxdir#1\relax%
     \write\linenum@out{\string\line@list@version{\this@line@list@version}}
%
     \ifl@dpaging%
       \write\linenum@out{\string\@par@sync@option{\@par@this@sync@option}}
%
     \fi%
  \fi}

%
```

`\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.

```
\newcommand*{\new@line}{%
  \IfStrEq{\led@pb@setting}{after}%
    {\xifinlist{\the\absline@num}{\l@prev@nopb}%
      {\xifinlist{\the\absline@num}{\normal@page@break}%
        {\numgdef{\@next@page}{\c@page+\@ne}%
        \write\linenum@out{\string\@nl[\@next@page][\@next@page]}%
      }%
      {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
    }%
    {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}}%
  {}%
  \IfStrEq{\led@pb@setting}{before}%
    {\numdef{\next@absline}{\the\absline@num+\@ne}%
    \xifinlist{\next@absline}{\l@prev@nopb}%
      {\xifinlist{\the\absline@num}{\normal@page@break}%
        {\numgdef{\nc@page}{\c@page+\@ne}%
        \write\linenum@out{\string\@nl[\nc@page][\nc@page]}%
        }%
        {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
      }%
      {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
    }%
    {}%
  \IfStrEqCase{\led@pb@setting}{{before}{\relax}{after}{\relax}}[\write\
linenum@out{\string\@nl[\the\c@page][\thepage]}]%
}

%
```

\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
\flag@end

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.

```
\newif\if@noneed@Footnote%

\newcommand*{\flag@start}{%
  \ifledRcol%
    \edef\next{\write\linenum@outR{%
                 \string\@ref[\the\insert@countR][]}}%
    \next%
    \ifnum\insert@countR<1%
      \if@noneed@Footnote\else%
        \led@err@EdtextWithoutFootnote%
      \fi%
    \fi%
  \else%
    \edef\next{\write\linenum@out{%
                 \string\@ref[\the\insert@count][]}}%
    \next%
    \ifnum\insert@count<1%
      \if@noneed@Footnote\else%
        \led@err@EdtextWithoutFootnote%
      \fi%
    \fi%
  \fi}%

\newcommand*{\flag@end}{%
  \ifledRcol%
    \write\linenum@outR{[]}%
  \else%
```

```
    \write\linenum@out{[]}%
  \fi}%


%
```

\flag@start@RTL
\flag@end@RTL

With XƎLaTeX, there is a problem when using RTL: the writing of a command in the numbered auxiliary files (.1, .2 etc) is reversed when the first argument of \edtext is typset in one line, but it is **not** reversed when this first argument is typset in two lines or more.[25]

To solve this problem, we use a crossref mechanism. At the first run, we put a label, but we do not write any \@ref command. When the value of the label can be tested, that is after three runs, we're doing:

- If the first argument of \edtext is typeset on only one line, we first call \flag@end, at the point we normally call \flag@start, at the beginning of the content of the first argument, and we call \flag@end at the point we normally call \flag@start, at the end of the content of the first argument.
- If the first argument of \edtext is typeset on only two lines, we use the normal order.

This system is a workaround for the problem of order when writing in auxiliary files.

The \flag@start@RTL and \flag@end@RTL macro put the label, do the test and call the right commands.

```
\newcommand{\flag@start@RTL}{%
  \edlabel{edtext:start:\csuse{thisedtext@\the\@edtext@level}}%
  \IfStrEq{\xabslineref{edtext:start:\csuse{thisedtext@\the\@edtext@level
}}}%
    {000}%
    {}%
    {%
    \ifnumequal%
      {\xabslineref{edtext:start:\csuse{thisedtext@\the\@edtext@level}}}%
      {\xabslineref{edtext:end:\csuse{thisedtext@\the\@edtext@level}}}%
      {\flag@end}%
      {\flag@start}%
  }%
}%

\newcommand{\flag@end@RTL}{%
  \edlabel{edtext:end:\csuse{thisedtext@\the\@edtext@level}}%
  \IfStrEq{\xabslineref{edtext:start:\csuse{thisedtext@\the\@edtext@level
}}}%
```

[25] This problem is caused by the way XƎTeX manages right-to-left typsetting. David Carlisle explains it on http://tex.stackexchange.com/a/333373/7712 and provides a potential solution, using \vadjust. However in some cases this adds spurious vertical spaces in reledmac. That is why we are using the solution explained below.

```
      {000}%
      {}%
      {%
      \ifnumequal%
        {\xabslineref{edtext:start:\csuse{thisedtext@\the\@edtext@level}}}%
        {\xabslineref{edtext:end:\csuse{thisedtext@\the\@edtext@level}}}%
        {\flag@start}%
        {\flag@end}%
    }%
}%
%
```

\flag@start@later
\flag@end@later

\flag@start@later and \flag@end@later: these send the\@ref@later to the line-list file command to the line-list file

```
\newcommand*{\flag@start@later}{%
  \ifledRcol%
    \write\linenum@outR{\string\@ref@later[}%
  \else%
    \write\linenum@out{\string\@ref@later[}%
  \fi%
}%
\newcommand{\flag@end@later}{%
  \ifledRcol%
    \write\linenum@outR{]}%
  \else%
    \write\linenum@out{]}%
  \fi%
}
%
```

\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.

```


\newcommand*{\startsub}{\dimen0\lastskip
  \ifdim\dimen0>0pt \unskip \fi
  \ifledRcol \write\linenum@outR{\string\sub@on}%
  \else      \write\linenum@out{\string\sub@on}%
  \fi
  \ifdim\dimen0>0pt \hskip\dimen0 \fi}
\def\endsub{\dimen0\lastskip
  \ifdim\dimen0>0pt \unskip \fi
  \ifledRcol \write\linenum@outR{\string\sub@off}%
  \else      \write\linenum@out{\string\sub@off}%
  \fi
  \ifdim\dimen0>0pt \hskip\dimen0 \fi}

%
```

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

```
\newcommand*{\advanceline}[1]{\leavevmode%
  \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
  \else      \write\linenum@out{\string\@adv[#1]}%
  \fi%
}
%
```

\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.

```

\newcommand*{\setline}[1]{%
 \leavevmode%
  \ifnum#1<\z@
    \led@warn@BadSetline
  \else
    \ifledRcol \write\linenum@outR{\string\@set[#1]}%
    \else      \write\linenum@out{\string\@set[#1]}%
    \fi
  \fi}

%
```

\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.

```

\newcommand*{\setlinenum}[1]{%
  \ifnum#1<\z@
    \led@warn@BadSetlinenum
  \else
    \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
    \else      \write\linenum@out{\string\l@d@set[#1]} \fi
  \fi}

%
```

\startlock
\endlock

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

```

\newcommand*{\startlock}{%
  \ifledRcol \write\linenum@outR{\string\lock@on}%
  \else      \write\linenum@out{\string\lock@on}%
  \fi}
\def\endlock{%
  \ifledRcol \write\linenum@outR{\string\lock@off}%
  \else      \write\linenum@out{\string\lock@off}%
  \fi}
%
```

\ifl@dskipnumber
\ifl@dskipversenumber
\l@dskipnumbertrue
\l@dskipnumberfalse
\skipnumbering

In numbered text \skipnumbering will suspend the numbering for that particular line.

```
\newif\ifl@dskipnumber
\newif\ifl@dskipversenumber%
\newcommand*{\skipnumbering}{%
  \leavevmode%
  \ifledRcol%
    \ifinstanza%
       \write\linenum@outR{\string\n@num@stanza}%
    \else%
      \write\linenum@outR{\string\n@num}%
    \fi%
    \advanceline{-1}%
  \else%
    \ifinstanza%
       \write\linenum@out{\string\n@num@stanza}%
    \else%
      \write\linenum@out{\string\n@num}%
    \fi%
    \advanceline{-1}%
  \fi%
}%

%
```

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. 145). 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 ca 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.

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

\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.

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

\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.

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

We are going to need another macro that takes one argument and ignores it entirely. This is supplied by the 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.[26] This is done because we want to pass into our notes the generic commands to change to roman or whatever, and not their expansions that will ask for a particular style at a specified size. The notes may well be in a smaller font, so the command should be expanded later, when the note's environment is in effect.

A second sort to turn off includes a few of the accent macros. Most are not a problem: an accent that is expanded to an \accent command may be harder to read but it works just the same. The ones that cause problems are: those that use alignments—TeX seems to

[26] Since 'control sequences equivalent to characters are not expandable'—*The TeXbook*, answer to Exercise 20.14.

get confused about the difference between alignment parameters and macro parameters; those that use temporary control sequences; and those that look carefully at what the current font is.

(The `\copyright` macro defined in Plain TeX has this sort of problem as well, but is not used enough to bother with. That macro, and any other that causes trouble, will get by all right if you put a `\protect` in front of it in your file.)

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

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

The `\new@series` command also adds `\let\footnote`$\langle X \rangle$`\@gobble` to the end of the `\no@expands` macro for the series $\langle X \rangle$.

(A related problem, not addressed by these two macros, is that of characters whose category code are changed by any of 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 characters, using LuaTeX or XeLaTeX.)

```
\newcommand*{\no@expands}{%
  \let\select@@lemmafont=0%
  \let\startsub=\relax  \let\endsub=\relax
  \let\startlock=\relax \let\endlock=\relax
  \let\edlabel=\@gobble
  \let\setline=\@gobble \let\advanceline=\@gobble
  \let\sameword\sameword@inedtext%
  \let\edtext=\dummy@edtext
  \let\edindex\dummy@edindex%
  \l@dtabnoexpands
  \morenoexpands}
\let\morenoexpands=\relax

%
```

`\@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.

```
\newcommand{\@tag}{}
```

```
%
```

\@edtext@level This counter is increased by 1 at each level of \edtext.

```
\newcount\@edtext@level%
\@edtext@level=0%
%
```

\if@edtext@secondarg@ This boolean is set to TRUE before reading the second argument of a \edtext. It is tested on some macro which must be executed only inside a second argument.

```
\newif\if@edtext@secondarg@%
%
```

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

```
\newcounter{edtext}
\renewcommand{\theedtext}{edtxt@\arabic{edtext}}%
%
```

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

```
\newcommand{\edtext}[2]{\leavevmode%
%
```

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

```
  \ifnumberedpar@%
%
```

we increment the \@edtext@level TeX counter to know in which level of \edtext we are.

```
      \global\advance\@edtext@level by 1%
%
```

We also increase the edtext LaTeX counter to insert a hypertarget if the hyperref package is loaded, and also works with \edtext on right-to-left typesetting with XeLaTeX.

We store the value for the current level in a global macro. So we have one macro by level of \edtext. That is required, because \edtext can contain \edtext.

```
      \stepcounter{edtext}%
      \csxdef{thisedtext@\the\@edtext@level}{\theedtext}%
%
```

By default, we do not use \lemma

```
      \global\@lemmacommand@false%
%
```

```
      \begingroup%
%
```

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.

```
      \ifledRcol%
        \ifcsvoid{sw@list@edtextR@\the\@edtext@level}%
              {\global\let\sw@inthisedtext\empty}%
              {\expandafter\gl@p\csname sw@list@edtextR@\the\@edtext@level\
endcsname\to\sw@inthisedtext}%
      \else%
        \ifcsvoid{sw@list@edtext@\the\@edtext@level}%
              {\global\let\sw@inthisedtext\empty}%
              {\expandafter\gl@p\csname sw@list@edtext@\the\@edtext@level\
endcsname\to\sw@inthisedtext}%
      \fi%
%
```

\@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.

```
        \global\renewcommand{\@tag}{%
          \no@expands #1%
          }%
%
```

\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.

```
        \set@line%
%
```

\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.

```
        \ifledRcol \global\insert@countR \z@%
        \else      \global\insert@count \z@ \fi%
%
```

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.

```
        \@edtext@secondarg@true%
        \ignorespaces #2\relax%
        \@edtext@secondarg@false%
%
```

With XeLaTeX, you must track whether the language reads left to right (English) or right to left (Arabic). reledmac defines an \if@RTL boolean test is not already defined.

```
        \if@RTL%
          \flag@start@RTL%
        \else%
          \flag@start%
        \fi%
%
```

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

```
       \if@lemmacommand@%
         \ifledRcol%
           \write\linenum@outR{\string\@lemma}%
         \else%
           \write\linenum@out{\string\@lemma}%
         \fi%
       \fi%
%
```

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.

```
      \endgroup%
      \ifdef{\hypertarget}%
        {%
        \Hy@raisedlink@left{\hypertarget{\csuse{thisedtext@\the\
@edtext@level}:start}{}}%
        \showlemma{#1}%
        \Hy@raisedlink{\hypertarget{\csuse{thisedtext@\the\@edtext@level}:
end}{}}%
        }%
        {%
        \showlemma{#1}%
        }%
%
```

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.

```
      \ifx\end@lemmas\empty \else%
        \gl@p\end@lemmas\to\x@lemma%
        \x@lemma%
        \global\let\x@lemma=\relax%
      \fi%
      \if@RTL%
        \flag@end@RTL%
      \else%
        \flag@end%
      \fi%
%
```

We switch some flags to false.

- The one that checks having footnotes inside a \edtext.
- The one that says we are inside a \edtext. In fact, it is not a flag, but a counter which is increased to 1 in each leavel of \edtext.
- The one that says we are inside a \@lemma.

```
      \global\@noneed@Footnotefalse%
      \global\advance\@edtext@level by -1%
      \global\@lemmacommand@false%
%
```

We also reset \@beforeinsertofthisedtext

```
      \global\let\@beforeinsertofthisedtext\relax%
%
```

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

```
  \else%
  \showlemma{#1} (\textbf{\textsc{Edtext outside numbered paragraph}})\led@err@edtextoutsidepstart%
  \fi%
}%


%
```

\@beforeinsertofthisedtext \@beforeinsertofthisedtext is an internal macro. reledmac or reledpar can add in this macro any content required to be executed before doing any \insert related to a \edtext. Its content is \let equal to \relax at the end of every \edtext.

```
\let\@beforeinsertofthisedtext\relax
%
```

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

```
\newif\ifnumberline
\numberlinetrue
%
```

\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.

```
\newcommand*{\set@line}{%
  \ifledRcol
    \ifx\line@listR\empty
      \global\noteschanged@true
      \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
    \else
      \gl@p\line@listR\to\@tempb
      \xdef\l@d@nums{\@tempb|\edfont@info}%
      \global\let\@tempb=\undefined
    \fi
  \else
    \ifx\line@list\empty
      \global\noteschanged@true
      \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
    \else
      \gl@p\line@list\to\@tempb
      \xdef\l@d@nums{\@tempb|\edfont@info}%
      \global\let\@tempb=\undefined
    \fi
  \fi}

%
```

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

```
\newcommand*{\edfont@info}{\f@encoding/\f@family/\f@series/\f@shape}

%
```

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. 128).

```
\newcommand*{\lemma}[1]{%
  \global\@lemmacommand@true%
  \global\renewcommand{\@tag}{%
    \no@expands #1%
  }%
  \ignorespaces%
}%
%
```

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

```
\newcommand{\@lemma}{%
  \booltrue{lemmacommand@\the\@edtext@level}%
}%
%
```

`\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.

```
\newif\if@lemmacommand@%
%
```

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. 99): 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.

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

`\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.

```
\def\line@set#1|#2\\#3|#4\\{%
  \gdef\@tempb{#1}%
  \ifx\@tempb\empty
        \l@d@add{#3}%
  \else
        \l@d@add{#1}%
  \fi
  \gdef\@tempb{#4}%
  \ifx\@tempb\empty\else
      \l@d@add{|}\line@set#2\\#4\\%
  \fi}
%
```

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

```
\newcommand{\l@d@add}[1]{\xdef\l@d@nums{\l@d@nums#1}}

%
```

VI.4 Lemma disambiguation

The mechanism which counts the occurrence of a same word in a same line is quite complex, because, when 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 reledmac section (the .1, .2… file).
- **When this auxiliary file is read at the second run**, different operations are achieved:
    1. Get the rank of each \sameword in a line (relative rank) from the rank of each \sameword in all the numbered section (absolute rank):
        - For each paired \sameword argument and absolute line number, a counter is defined. Its value corresponds to the number of times \sameword{⟨*argument*⟩} is called from the beginning of the lineation to the end of the current line. We also store the same data for the preceding absolute line number, if it does not have \sameword{⟨*argument*⟩}.
        - For each \sameword having the same argument, we subtract from its absolute rank the number stored for the paired \sameword argument and previous absolute line number. Consequently, we obtain the relative rank.

  - See the following example which explains how, for same `\sameword`, absolute ranks are transformed to relative ranks.

    ```
    At line 1:
    absolute rank 1 becomes relative rank 1-0 = 1
    1 is stored for this \sameword and line 1
    At line 2:
    absolute rank 2 becomes relative rank 2-1 = 1
    absolute rank 3 becomes relative rank 3-1 = 2
    3 is stored for this \sameword and line 2
    At line 3:
    no \sameword for this line.
    3 is stored for this \sameword and line 3
    At line 4:
    absolute rank 4 becomes relative rank 4-3 = 1
    4 is stored for this \sameword and line 4
    ```

2. Create lists of lists of `\sameword` by depth of `\edtext`. That is: create a list for `\edtexts` of level 1, a list for `\edtexts` of level 2, a list for `\edtexts` of level 3 etc. For each `\edtext` in these lists, we store all of the relative ranks of `\saweword` which are called as lemma information. That is: 1) either called in the first argument of `\sameword`, or, 2) called in the `\lemma` macro of the second argument of `\sameword` AND marked by the optional argument of `\saweword` in first argument of `\edtext`.

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

   $$\text{bar}^1\ \boxed{\boxed{\text{foo}^1\ \text{foo}^2\ \text{bar}^2\ \text{foo}^3}\ \text{(A)(B) foo}^4\ \text{bar}^3}\ \text{(C)}\ \boxed{\boxed{\text{foo}^5}\ \text{(D) bar}^4}\ \text{(E)}$$

   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 levels 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 its `\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 a list of relative ranks 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 a method of knowing the absolute line number associated to a `\sameword` is required because a `\edtext` can overlap many lines, but `\sameword` can't get it.
    * When reading the auxiliary file, we get the value associated to the pair composed by the current marked word and the current absolute line number. To this value, we subtract the value associated to the pair composed by the current marked word and the previous absolute line number. Let the result be n.
- If $n > 1$, that means the current word appears more than once in its line. In this case, we call `\showwordrank` with the word as the first argument and r as the 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 an 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 the dynamic name of macro with `\csname`.[27]

Because there is a bug with `\detokenize` and XeTeX when using non BMP characters[28], we detokenize only for non-XeTeXengines. In any case, in XeTeXa `\csname` construction can contain UTF-8 characters without a problem, as UTF-8 characters are not managed with category codes, but instead read directly as UTF-8 characters.

```
\newcommand{\get@sw@txt}[1]{%
  \begingroup%
    \let\sameword\@firstofone%Allow to have nested \sameword
    \ifxetex%
      \xdef\sw@txt{#1}%
    \else%
      \expandafter\xdef\expandafter\sw@txt\expandafter{\detokenize{#1}}%
    \fi%
  \endgroup%
}%
%
```

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

[27] See `http://tex.stackexchange.com/q/244538/7712`.

[28] `http://sourceforge.net/p/xetex/bugs/108/`

```
\newcommandx{\sameword}[2][1,usedefault]{%
     \leavevmode%
     \get@sw@txt{#2}%
%
```

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

```
  \unless\ifledRcol%
     \csnumgdef{sw@\sw@txt}{\csuse{sw@\sw@txt}+\@ne}%
%
```

Then, write its value to the numbered file.

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

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

```
  \else%
     \csnumgdef{sw@\sw@txt}{\csuse{sw@\sw@txt}+\@ne}%
     \protected@write\linenum@outR{}{\string\@sw{\sw@txt}{\csuse{sw@\sw@txt
}}{#1}}%
  \fi%
%
```

And print the word.

```
  #2%
}%
%
```

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

\if@addsw

```
\newif\if@addsw%
%
```

\@sw The command printed in the auxiliary files.

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

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

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

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

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

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

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

And do the same thing for the right side.

```
  \else%
    \csxdef{sw@\sw@txt @\the\absline@numR @\the\section@numR @R}{#2}%
    \numdef{\prev@line}{\the\absline@numR-1}%
    \ifcsundef{sw@\sw@txt @\prev@line @\the\section@numR @R}{%
      \csnumgdef{sw@\sw@txt @\prev@line @\the\section@numR @R}{#2-1}%
     }{}%
    \numdef{\the@sw}{#2-\csuse{sw@\sw@txt @\prev@line @\the\section@numR @R}}%
  \fi%
%
```

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

```
  \@tempcnta=\@edtext@level
  \@whilenum{\@tempcnta>0}\do{%
    \ifcsdef{sw@list@edtext@tmp@\the\@tempcnta}%
      {%
        \@addswfalse%
        \notbool{lemmacommand@\the\@tempcnta}%
          {\@addswtrue}%
          {\IfStrEq{#3}{inlemma}%
            {\@addswtrue}%
            {%
             \def\do##1{%
                \ifnumequal{##1}{\the\@tempcnta}%
                  {\@addswtrue\listbreak}%
                  {}%
             }%
             \docsvlist{#3}%
            }%
          }%
          \if@addsw%
            \letcs{\@tmp}{sw@list@edtext@tmp@\the\@tempcnta}%
            \ifledRcol%
              \xright@appenditem{{\the@sw}{\the\absline@numR}}\to\@tmp%
            \else%
              \xright@appenditem{{\the@sw}{\the\absline@num}}\to\@tmp%
            \fi%
            \cslet{sw@list@edtext@tmp@\the\@tempcnta}{\@tmp}%
          \fi%
      }%
      {}%
    \advance\@tempcnta by -1%
  }%
}%
%
```

\sameword@inedtext

The command called when \sameword is called in a \edtext.

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

Just a precaution.

```
      \ifx\sw@list@inedtext\empty%
        \def\the@sw{999}%
        \def\this@absline{-99}%
      \else%
%
```

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

```
        \gl@p\sw@list@inedtext\to\@tmp%
        \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
        \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
      \fi%
%
```

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

```
      \ifcsdef{sw@\sw@txt @\this@absline @\the\section@num}{%
        \numdef{\prev@line}{\this@absline-1}%
        \numdef{\sw@atthisline}{\csuse{sw@\sw@txt @\this@absline @\the\section@num}-\csuse{sw@\sw@txt @\prev@line @\the\section@num}}%
        }%
        {\numdef{\sw@atthisline}{0}}%
%
```

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

```
      \ifnumgreater{\sw@atthisline}{1}%
          {\showwordrank{#2}{\the@sw}}%
          {#2}%
%
```

And the same for right side.

```
  \else%
      \ifx\sw@list@inedtext\empty%
```

```
        \def\the@sw{999}%
        \def\this@absline{-99}%
      \else%
        \gl@p\sw@list@inedtext\to\@tmp%
        \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
        \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
      \fi%
      \ifcsdef{sw@\sw@txt @\this@absline @\the\section@numR @R}{%
        \numdef{\prev@line}{\this@absline-1}%
        \numdef{\sw@atthisline}{\csuse{sw@\sw@txt @\this@absline @\the\section@numR @R}-\csuse{sw@\sw@txt @\prev@line @\the\section@numR @R}}%
        }%
        {\numdef{\sw@atthisline}{0}}%
        \ifnumgreater{\sw@atthisline}{1}%
          {\showwordrank{#2}{\the@sw}}%
          {#2}%
  \fi%
}%
%
```

\showwordrank

Finally, the way the rank will be printed.

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

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
\ifnumberedpar@
\numberedpar@true
\numberedpar@false
\num@lines
\one@line
\par@line

Here are numbers and flags that are used internally in the course of the paragraph decomposition.

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

```
\newbox\raw@text
\newif\ifnumberedpar@
\newcount\num@lines
\newbox\one@line
\newcount\par@line
%
```

\pstart
\AtEveryPstart
\AtStartEveryPstart
\numberpstarttrue
\numberpstartfalse
\labelpstarttrue
\labelpstartfalse
\thepstart
\ifat@every@pstart@star@

\pstart starts the paragraph by clearing the \inserts@list list and other relevant variables, and then arranges for the subsequent text to go into the \raw@text box. \pstart needs to appear at the start of every paragraph that is to be numbered; the \autopar command below may be used to insert these commands automatically.

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

```

\newcommand{\AtStartEveryPstart}[1]{%
  \ifstrempty{#1}%
    {\gdef\@at@start@every@pstart{}}%
    {\gdef\@at@start@every@pstart{#1}}%
}%
\def\@at@start@every@pstart{}%

\newif\ifat@every@pstart@star@%
\newcommand{\AtEveryPstart}[1]{%
  \ifstrempty{#1}%
    {\gdef\at@every@pstart{}}%
    {\gdef\at@every@pstart{\noindent#1}}%
 \global\at@every@pstart@star@false%
}%
\WithSuffix\newcommand\AtEveryPstart*[1]{%
  \ifstrempty{#1}%
    {\gdef\at@every@pstart{}}%
    {\gdef\at@every@pstart{#1}}%
  \global\at@every@pstart@star@true%
}%
\def\at@every@pstart{}%

\newcounter{pstart}
\renewcommand{\thepstart}{{\bfseries\@arabic\c@pstart}. }
\newif\ifnumberpstart
\numberpstartfalse
\newif\iflabelpstart
\labelpstartfalse
\newcommandx*{\pstart}[2][1,2,usedefault]{%
  \normal@pars%
  \ifboolexpr{%
    test {\ifstrempty{#1}}%
    and test {\ifstrempty{#2}}%
  }%
    {\at@every@pstart}%
    {%
    \ifstrempty{#1}{}{\noindent#1}%
    \ifstrempty{#2}{}{#2}%
    }%
  \ifautopar%
    \autopar%
  \fi%
  \ifluatex%
    \edef\l@luatextextdir@L{\the\textdir}%
  \fi%
  \@nobreaktrue%
  \ifnumbering \else%
    \led@err@PstartNotNumbered%
    \beginnumbering%
  \fi%
  \ifnumberedpar@%
    \led@err@PstartInPstart%
    \pend%
  \fi%
  \list@clear{\inserts@list}%
  \global\let\next@insert=\empty%
  \begingroup\normal@pars%
  \global\advance \l@dnumpstartsL\@ne
  \global\setbox\raw@text=\vbox\bgroup%
    \if@nobreak%
      \if@afterindent\else%
        \noindent%
        \global\@afterindenttrue%
      \fi%
    \fi%
    \ifautopar\else%
    \ifnumberpstart%
      \ifinstanza\else%
       \ifsidepstartnum\else%
         \thepstart%
        \fi%
       \fi%
      \fi%
     \fi%
  \numberedpar@true%
  \iflabelpstart\protected@edef\@currentlabel%
       {\p@pstart\thepstart}
  \fi%
  \l@dzeropenalties%
  \@at@start@every@pstart%
  \ignorespaces%because not automatically ignored if an optional argument is used (classical TeX behavior for space after commands)
  }
```

```
%
```

\pend

\pend must be used to end a numbered paragraph.

```
\newcommandx*{\pend}[2][1,2,usedefault]{\ifnumbering \else%
    \led@err@PendNotNumbered%
  \fi%
  \global\l@dskipversenumberfalse%
  \ifnumberedpar@ \else%
    \led@err@PendNoPstart%
  \fi%
%
```

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.

```
  \l@dzeropenalties%
  \@at@end@every@pend%
  \endgraf\global\num@lines=\prevgraf\egroup%
  \global\par@line=0%
%
```

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.

```
  \csnumdef{pstartline}{0}%
  \loop\ifvbox\raw@text%
    \csnumdef{pstartline}{\pstartline+\@ne}%
    \do@line%
    \ifbypstart@%
     \ifnumequal{\pstartline}{1}{%
       \bgroup%
       \let\leavevmode\relax%
       \setline{1}%
       \egroup%
     \resetprevline@}{}%
    \fi%
  \repeat%
%
```

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

```
  \flush@notes%
  \endgroup%
  \ignorespaces%
%
```

Increase pstart counter.

```
  \ifnumberpstart%
     \global\pstartnumtrue%
  \fi%
  \addtocounter{pstart}{1}%
%
```

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

```
  \normal@pars%
  \ifboolexpr{%
    test {\ifstrempty{#1}}%
    and test {\ifstrempty{#2}}%
  }%
    {\at@every@pend}%
    {%
    \ifstrempty{#1}{}{\noindent#1}%
    \ifstrempty{#2}{}{#2}%
    }%
%
```

Restore standard "nobreak" and "autopar" settings. Normally, `\if@nobreak` is true only immediately after a sectioning command (see latex.ltx file). As a `\pstart`…`\pend` structure can't contain any sectioning command, we set `\if@nobreak` to false.

```
  \@nobreakfalse%
  \ifautopar%
    \autopar%
  \fi%
}
%
```

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.

`\AtEveryPend`
`\at@every@pend`
`\ifat@every@pend@star@`

```


\newif\ifat@every@pend@star@%
\newcommand{\AtEveryPend}[1]{%
  \ifstrempty{#1}%
    {\gdef\at@every@pend{}}%
    {\gdef\at@every@pend{\noindent#1}}%
  \global\at@every@pend@star@false%
}%
\WithSuffix\newcommand\AtEveryPend*[1]{%
  \ifstrempty{#1}%
    {\gdef\at@every@pend{}}%
    {\gdef\at@every@pend{#1}}%
  \global\at@every@pend@star@true%
}%
```

```
\xdef\at@every@pend{}%

%
```

\AtEndEveryPend Here a macro to insert automatically any content at the end of \pend, in numbered lines.

```
\newcommand{\AtEndEveryPend}[1]{%
  \ifstrempty{#1}%
    {\xdef\@at@end@every@pend{}}%
    {\gdef\@at@end@every@pend{#1}}%
}%
\def\@at@end@every@pend{}%
%
```

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

```
\newcommand*{\l@dzeropenalties}{%
  \brokenpenalty \z@ \clubpenalty \z@
  \displaywidowpenalty \z@ \interlinepenalty \z@ \predisplaypenalty \z@
  \postdisplaypenalty \z@ \widowpenalty \z@}

%
```

\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.

```
\newif\ifautopar
\autoparfalse
\newcommand*{\autopar}{
  \ifledRcol
    \ifnumberingR \else
    \led@err@AutoparNotNumbered
    \beginnumberingR
    \fi
    \else
    \ifnumbering \else
    \led@err@AutoparNotNumbered
    \beginnumbering
    \fi
  \fi
  \autopartrue
  \everypar{\setbox0=\lastbox
    \endgraf \vskip-\parskip
    \pstart \noindent \kern\wd0 \ifnumberpstart\ifinstanza\else\thepstart\fi\fi
    \let\par=\pend}%
  \ignorespaces}
%
```

\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.

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

%
```

\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 neeeded.

```
\newif\ifautopar@pause
%
```

VII.2 Processing one line

VII.2.1 General process

\do@line
\l@dunhbox@line
The \do@line macro is called by \pend to do all the processing for a single line of text.

```
 \newcommand*{\l@dunhbox@line}[1]{\unhbox #1}
 \newcommand*{\do@line}{%
  {\vbadness=10000
   \splittopskip=\z@
   \do@linehook
```

```
\l@demptyd@ta
   \global\setbox\one@line=\vsplit\raw@text to\baselineskip}%
  \unvbox\one@line \global\setbox\one@line=\lastbox
  \getline@num
  \IfStrEq{\led@pb@setting}{before}{\led@check@pb\led@check@nopb}{}
  \ifnum\@lock>\@ne
    \inserthangingsymboltrue
  \else
    \inserthangingsymbolfalse
  \fi
  \check@pb@in@verse
  \ifl@dhidenumber%
    \global\l@dhidenumberfalse%
    \f@x@l@cks%
  \else%
    \affixline@num%
  \fi%
%
```

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

```
  \xifinlist{\the\l@dnumpstartsL}{\eled@sections@@}%
     {\print@eledsection}%
     {\print@line}%
  \IfStrEq{\led@pb@setting}{after}{\led@check@pb\led@check@nopb}{}
  }%
%
```

VII.2.2 Process for "normal" line

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

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

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

```
    \affixpstart@num%
%
```

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

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

User hook.

```
        \do@insidelinehook%
%
```

Left line number

```
        \l@dld@ta%
%
```

Prepare text to be inserted before notes.

```
        \if@firstlineofpage%
          \set@Xtxtbeforenotes%
          \set@txtbeforenotesX%
          \global\@firstlineofpagefalse%
        \fi%
%
```

Insert footnotes made of manuscripts data and critical footnotes.

```
        \ifdefstring{\ms@data@position}{msdata-regular}{%
          \insert@msdata%
          \add@inserts%
          \add@Xgroupbyline%
        }{%
          \add@inserts%
          \add@Xgroupbyline%
          \insert@msdata%
        }%
%
```

Insert marginal notes.

```
        \affixside@note%
%
```

Print left notes.

```
        \l@dlsn@te
%
```

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

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

If we use LuaLaTeX then restore the direction.

```
        \ifluatex%
          \textdir\l@luatextextdir@L%
        \fi%
%
```

Insert, if needed, the hanging symbol.

```
        \inserthangingsymbol%
%
```

And so, print the line.

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

Right line number

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

Print right notes.

```
        \l@drsn@te
      }}%
%
```

And reinsert penalties (for page breaking)...

```
    \add@penalties%
}
%
```

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.

```
\def\print@eledsection{%
    \if@firstlineofpage%
      \set@Xtxtbeforenotes%
      \set@txtbeforenotesX%
      \global\@firstlineofpagefalse%
    \fi%
    \ifdefstring{\ms@data@position}{msdata-regular}{%
      \insert@msdata%
      \add@inserts%
      \add@Xgroupbyline%
    }{%
      \add@inserts%
      \add@Xgroupbyline%
      \insert@msdata%
    }%
    \affixside@note%
    \numdef{\temp@}{\l@dnumpstartsL-1}%
    \xifinlist{\temp@}{\eled@sections@@}{\@nobreaktrue}{\@nobreakfalse}%
    \@eled@sectioningtrue%
    \csuse{eled@sectioning@\the\l@dnumpstartsL}%
    \@eled@sectioningfalse%
    \global\csundef{eled@sectioning@\the\l@dnumpstartsL}%
    \if@RTL%
      \hspace{-3\paperwidth}%
    {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
    \else%
      \hspace{3\paperwidth}%
    {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
    \fi%
    \vskip-\baselineskip%
}
%
```

VII.2.4 Hooks

\do@linehook
\do@insidelinehook

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

```
\newcommand*{\do@linehook}{}
\newcommand*{\do@insidelinehook}{}
%
```

\dolinehook
\doinsidelinehook

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

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

%
```

VII.2.5 Sidenotes and marginal line number initialization

\l@demptyd@ta
\l@dld@ta
\l@drd@ta
\l@dcsnotetext
\l@dcsnotetext@l
\l@dcsnotetext@r

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

```
\newcommand*{\l@demptyd@ta}{%
  \gdef\l@dld@ta{}%
  \gdef\l@drd@ta{}%
  \gdef\l@dcsnotetext@l{}%
  \gdef\l@dcsnotetext@r{}%
  \gdef\l@dcsnotetext{}}

%
```

\l@dlsn@te
\l@drsn@te

Zero width boxes of the left and right side notes, together with their kerns.

```
\newcommand{\l@dlsn@te}{%
  \hb@xt@ \z@{\hss\box\l@dlp@rbox\kern\ledlsnotesep}}
\newcommand{\l@drsn@te}{%
   \hb@xt@ \z@{\kern\ledrsnotesep\box\l@drp@rbox\hss}}

%
```

\ledllfill
\ledrlfill

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

```
\newcommand*{\ledllfill}{\hfil}
\newcommand*{\ledrlfill}{}

%
```

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.

```
\newcommand*{\getline@num}{%
  \global\advance\absline@num \@ne%
  \do@actions
  \do@ballast
  \ifnumberline
      \ifsublines@
         \ifnum\sub@lock<\tw@
           \global\advance\subline@num \@ne
         \fi
      \else
        \ifnum\@lock<\tw@
           \global\advance\line@num \@ne
           \global\subline@num \z@
        \fi
      \fi
  \fi
}
%
```

\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. 160).

\ballast@count
\c@ballast

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

```
\newcount\ballast@count
\newcounter{ballast}
  \setcounter{ballast}{0}
%
```

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.

```
\newcommand*{\do@ballast}{\global\ballast@count \z@
  \begingroup
    \advance\absline@num \@ne
    \ifnum\next@actionline=\absline@num
      \ifnum\next@action>-1001\relax
        \global\advance\ballast@count by -\c@ballast
       \fi
     \fi
  \endgroup}
%
```

\do@actions
\do@actions@next

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

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

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

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.

```
    \ifnum\next@action>-1001
       \global\page@num=\next@action
       \global\@firstlineofpagetrue%
       \ifbypage@
         \global\line@num=\z@ \global\subline@num=\z@
         \resetprevline@
       \fi
       \add@msdata@firstlineofpage%
%
```

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.)

```
    \else
       \ifnum\next@action<-4999
          \@l@dtempcnta=-\next@action
          \advance\@l@dtempcnta by -5001
          \ifsublines@
             \global\subline@num=\@l@dtempcnta
          \else
             \global\line@num=\@l@dtempcnta
          \fi
%
```

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

```
       \else
          \@l@dtempcnta=-\next@action
          \advance\@l@dtempcnta by -1000
          \do@actions@fixedcode
       \fi
    \fi
%
```

Now we get information about the next action off the list, and then set \do@actions@next so that we will call ourself 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.

```
    \ifx\actionlines@list\empty
         \gdef\next@actionline{1000000}%
    \else
         \gl@p\actionlines@list\to\next@actionline
         \gl@p\actions@list\to\next@action
         \global\let\do@actions@next=\do@actions
    \fi
  \fi
%
```

Make the recursive call, if necessary.

```
\do@actions@next}

%
```

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

```
\newcommand*{\do@actions@fixedcode}{%
  \ifcase\@l@dtempcnta
  \or%                     % 1001 = starting sublineation
    \global\sublines@true
  \or%                     % 1002 = ending sublineation
    \global\sublines@false
  \or%                     % 1003 = starting locking number
      \global\@lock=\@ne
  \or%                     % 1004 = ending locking number
    \ifnum\@lock=\tw@
      \global\@lock=\thr@@
    \else
      \global\@lock=\z@
    \fi
  \or%                     % 1005 = starting locking subnumber
     \global\sub@lock=\@ne
  \or%                     % 1006 = ending locking subnumber
    \ifnum\sub@lock=\tw@
      \global\sub@lock=\thr@@
    \else
```

```
      \global\sub@lock=\z@
    \fi
  \or%                      % 1007 = skipping numbering
    \l@dskipnumbertrue
  \or%                      % 1008 = skipping numbering in stanza
    \l@dskipversenumbertrue%
  \or%                     % 1009 = hiding number
    \l@dhidenumbertrue
  \or%                     % 1010 = inserting msdata
    \add@msdata%
  \else
    \led@warn@BadAction
  \fi}


%
```

IX Line number printing

\affixline@num

`\affixline@num` just puts a left line number into `\l@dld@ta` or a right line number into `\l@drd@ta` if required.

To determine whether we need to affix a line number to this line, we compute the following:

$$\begin{aligned} n &= \mathit{int}((\mathit{linenum} - \mathit{firstlinenum})/\mathit{linenumincrement}) \\ m &= \mathit{firstlinenum} + (n \times \mathit{linenumincrement}) \end{aligned}$$

(where *int* truncates a real number to an integer). m will be equal to *linenum* only if we are to paste a number on here. However, the formula breaks down for the first line to number (and any before that), so we check that case separately: if `\line@num` $\leq$ `\firstlinenum`, we compare the two directly instead of making these calculations.

We compute, in the scratch counter `\@l@dtempcnta`, the number of the next line that should be printed with a number (m in the above discussion), and move the current line number into the counter `\@l@dtempcntb` for comparison.

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

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

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

```
  \ifledgroupnotesL@\else
    \ifnumberline
      \ifl@dskipnumber
        \global\l@dskipnumberfalse
      \else
        \ifsublines@
```

```
           \@l@dtempcntb=\subline@num
           \ifnum\subline@num>\c@firstsublinenum
               \@l@dtempcnta=\subline@num
               \advance\@l@dtempcnta by-\c@firstsublinenum
               \divide\@l@dtempcnta by\c@sublinenumincrement
               \multiply\@l@dtempcnta by\c@sublinenumincrement
               \advance\@l@dtempcnta by\c@firstsublinenum
           \else
               \@l@dtempcnta=\c@firstsublinenum
           \fi
%
```

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.

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

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

```
        \else
           \@l@dtempcntb=\line@num
%
```

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

```
           \ifx\linenumberlist\empty
              \ifnum\line@num>\c@firstlinenum
                 \@l@dtempcnta=\line@num
                 \advance\@l@dtempcnta by-\c@firstlinenum
                 \divide\@l@dtempcnta by\c@linenumincrement
                 \multiply\@l@dtempcnta by\c@linenumincrement
                 \advance\@l@dtempcnta by\c@firstlinenum
              \else
                 \@l@dtempcnta=\c@firstlinenum
              \fi
           \else
%
```

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

```
              \@l@dtempcnta=\line@num
              \edef\rem@inder{,\linenumberlist,\number\line@num,}%
              \edef\sc@n@list{\def\noexpand\sc@n@list
                ####1,\number\@l@dtempcnta,####2|{\def\noexpand\rem@inder
{####2}}}%
              \sc@n@list\expandafter\sc@n@list\rem@inder|%
              \ifx\rem@inder\empty%
                \advance\@l@dtempcnta\@ne
              \fi
           \fi
%
```

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

```
           \ch@ck@l@ck
        \fi
%
```

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

```
        \ifnum\@l@dtempcnta=\@l@dtempcntb
           \ifl@dskipversenumber\else
%
```

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`
`\l@drd@ta`

A left line number is stored in `\l@dld@ta` and a right one in `\l@drd@ta`.

```
             \if@twocolumn
                \if@firstcolumn
                   \gdef\l@dld@ta{\llap{{\leftlinenum}}}%
                \else
                   \gdef\l@drd@ta{\rlap{{\rightlinenum}}}%
                \fi
             \else
                \ifboolexpr{bool {l@dprintingcolumns} and test {\
ifnumgreater{\line@margin@columns}{\m@ne}}}%
                  {\@l@dtempcntb=\line@margin@columns}%
                  {\@l@dtempcntb=\line@margin}%
                \ifnum\@l@dtempcntb>\@ne
                   \advance\@l@dtempcntb \page@num
                \fi
                \ifodd\@l@dtempcntb
                   \gdef\l@drd@ta{\rlap{{\rightlinenum}}}%
                 \else
                   \gdef\l@dld@ta{\llap{{\leftlinenum}}}%
                \fi
             \fi
          \fi
       \fi
%
```

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

```
       \f@x@l@cks
      \fi
    \fi
  \fi
}

%
```

\ch@cksub@l@ck
\ch@ck@l@ck
\f@x@l@cks

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.

```
\newcommand*{\ch@cksub@l@ck}{%
    \ifcase\sub@lock
      \or
        \ifnum\sublock@disp=\@ne
           \@l@dtempcntb=\z@ \@l@dtempcnta=\@ne
        \fi
      \or
        \ifnum\sublock@disp=\tw@ \else
           \@l@dtempcntb=\z@ \@l@dtempcnta=\@ne
        \fi
      \or
        \ifnum\sublock@disp=\z@
           \@l@dtempcntb=\z@ \@l@dtempcnta=\@ne
        \fi
    \fi}
%
```

Similarly for line numbers.

```
\newcommand*{\ch@ck@l@ck}{%
    \ifcase\@lock
       \or
         \ifnum\lock@disp=\@ne
            \@l@dtempcntb=\z@ \@l@dtempcnta=\@ne
         \fi
       \or
         \ifnum\lock@disp=\tw@ \else
            \@l@dtempcntb=\z@ \@l@dtempcnta=\@ne
         \fi
       \or
         \ifnum\lock@disp=\z@
            \@l@dtempcntb=\z@ \@l@dtempcnta=\@ne
         \fi
    \fi}
%
```

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

```
\newcommand*{\f@x@l@cks}{%
  \ifcase\@lock
  \or
    \global\@lock=\tw@
  \or \or
    \global\@lock=\z@
  \fi
  \ifcase\sub@lock
  \or
    \global\sub@lock=\tw@
  \or \or
    \global\sub@lock=\z@
  \fi}

%
```

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 \rightstartnum commands. After the try, it is set to FALSE.

\leftpstartnum
\rightstartnum
\ifsidepstartnum

```

\newif\ifsidepstartnum
\newcommand*{\affixpstart@num}{%
    \ifsidepstartnum
        \if@twocolumn
            \if@firstcolumn
                  \gdef\l@dld@ta{\llap{{\leftpstartnum}}}%
            \else
                  \gdef\l@drd@ta{\rlap{{\rightpstartnum}}}%
            \fi
        \else
            \@l@dtempcntb=\line@margin%
            \ifnum\@l@dtempcntb>\@ne
                  \advance\@l@dtempcntb \page@num
            \fi
            \ifodd\@l@dtempcntb
                  \gdef\l@drd@ta{\rlap{{\rightpstartnum}}}%
            \else
```

```
                  \gdef\l@dld@ta{\llap{{\leftpstartnum}}}%
            \fi
        \fi
    \fi

}
%

\newif\ifpstartnum
\pstartnumtrue
\newcommand*{\leftpstartnum}{
    \ifpstartnum\thepstart
    \kern\linenumsep\fi
    \global\pstartnumfalse
}
\newcommand*{\rightpstartnum}{
    \ifpstartnum
    \kern\linenumsep
    \thepstart
    \fi
    \global\pstartnumfalse
}
%
```

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 by 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.

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

`\add@inserts`
`\add@inserts@next`

`\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.

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

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

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

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.

```
  \ifx\next@insert\empty
    \ifx\insertlines@list\empty
      \global\noteschanged@true
      \gdef\next@insert{100000}%
    \else
      \gl@p\insertlines@list\to\next@insert
    \fi
  \fi
%
```

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.

```
  \ifnum\next@insert=\absline@num
    \gl@p\inserts@list\to\@insert
    \@insert
    \global\let\@insert=\undefined
    \global\let\next@insert=\empty
    \global\let\add@inserts@next=\add@inserts
  \fi
\fi
%
```

Make the recursive call, if necessary.

```
\add@inserts@next}

%
```

\add@Xgroupbyline If you use \Xgroupbyline, the insertion of the critical footnotes are not made immediately in \add@inserts, but the content to be inserted is stored, to be inserted in one block. This insertion in one block is made by \add@Xgroupbyline.

```
\newcommand{\add@Xgroupbyline}{%
  \unless\ifnocritical@%
```

```
    \def\do##1{%Looping on the series
      \def\do####1{%Looping on the ##1@forinserting command
         \ifcsdef{##1@forinserting@####1}{%
           \X@beforeinsertion{##1}%
           \if@ledgroup%
             \global\setbox\@nameuse{mp##1footins}=\vbox%
           \else%
             \insert\csname ##1footins\endcsname%
           \fi%
           {%
             \ifcsdef{Xhsize\csuse{series@display##1}@##1}%
               {\hsize \csuse{Xhsize\csuse{series@display##1}@##1}}%
               {}%
             \if@ledgroup%
               \unvbox\@nameuse{mp##1footins}%
             \fi%
             \X@atbegininsertion{##1}%
             \ifcsstring{series@display##1}%
               {%
                  \Xledsetnormalparstuff{##1}%
                  \rule\z@\splittopskip%
               }%
               {}%
             \csuse{##1@forinserting@####1}%
             \strut\par%
           }%
           \global\csundef{##1@forinserting@####1}%
           }%
           {}%
         }%
      \ifcsdef{##1@forinserting}{%
        \dolistcsloop{##1@forinserting}%
      }{}%
      \global\csundef{##1@forinserting}%
      }%
    \dolistloop{\@series}%
  \fi%
}%


%
```

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@dtempcnta 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. 150). Finally, the penalty is checked to see that it does not go below −10000.

```
\newcommand*{\add@penalties}{\@l@dtempcnta=\ballast@count
  \ifnum\num@lines>\@ne
    \global\advance\par@line \@ne
    \ifnum\par@line=\@ne
      \advance\@l@dtempcnta \clubpenalty
    \fi
    \@l@dtempcntb=\par@line \advance\@l@dtempcntb \@ne
    \ifnum\@l@dtempcntb=\num@lines
      \advance\@l@dtempcnta \widowpenalty
    \fi
    \ifnum\par@line<\num@lines
      \advance\@l@dtempcnta \interlinepenalty
    \fi
  \fi
    \ifnum\@l@dtempcnta=\z@
      \relax
    \else
      \ifnum\@l@dtempcnta>-10000
        \penalty\@l@dtempcnta
      \else
        \penalty -10000
      \fi
    \fi}

%
```

XI.3 Printing leftover notes

\flush@notes

The \flush@notes macro is called after the entire paragraph has been sliced up and sent on to the vertical list. If the number of notes to this paragraph has increased since the previous run of TeX, then there can be leftover notes that have not yet been printed. An appropriate error message will be printed elsewhere; but it is best to go ahead and print these notes somewhere, even if it is not in quite the right place. What we do is dump them all out here, so that they should be printed on the same page as the last line of the paragraph. We can hope that is not too far from the proper location, to which they will move on the next run.

```
\newcommand*{\flush@notes}{%
  \@xloop
    \ifx\inserts@list\empty \else
      \gl@p\inserts@list\to\@insert
      \@insert
      \global\let\@insert=\undefined
  \repeat}

%
```

\@xloop

\@xloop is a variant of the Plain TeX \loop macro, useful when it's hard to construct a positive test using the TeX \if commands—as in \flush@notes above. One types \@xloop ... \if ... \else ... \repeat, and the action following \else is repeated as long as the \if test fails. (This macro will work wherever the Plain TeX \loop is used, too, so we could just call it \loop; but it seems preferable not to change the definitions of any of the standard macros.)

This variant of \loop was introduced by Alois Kabelschacht in *TUGboat* **8** (1987), pp. 184–5.

```
\def\@xloop#1\repeat{%
  \def\body{#1\expandafter\body\fi}%
  \body}

%
```

XI.4 Text before notes

\set@Xtxtbeforenotes

The \set@Xtxtbeforenotes macro resets the Xtxtbeforesnotes@⟨*series*⟩@typeset boolean to false. Just before the first note of the ⟨*series*⟩ in a page, the \Xtextbeforenotes will be inserted.

```
\newcommand{\set@Xtxtbeforenotes}{%
  \unless\ifnocritical@%
    \def\do##1{%
      \nottoggle{Xtxtbeforenotesonlyonce@##1}{%
        \global\togglefalse{Xtxtbeforesnotes@##1@typeset}%
      }{}%
    }%
    \dolistloop{\@series}%
  \fi%
}%
%
```

\set@txtbeforenotesX

The \set@txtbeforenotesX does the same for the \textbeforenotesX.

```
\newcommand{\set@txtbeforenotesX}{%
  \unless\ifnofamiliar@%
    \def\do##1{%
      \nottoggle{txtbeforenotesonlyonceX@##1}{%
        \global\togglefalse{txtbeforesnotesX@##1@typeset}%
      }{}%
    }%
    \dolistloop{\@series}%
  \fi%
}%
%
```

\insert@Xtxtbeforenotes
\insert@txtbeforenotesX

\insert@Xtxtbeforenotes{⟨*series*⟩}, called when inserting a familiar footnote, will insert the text before the note if it is not already inserted. For paragraphed footnotes, it will insert it as a component of the first footnote. For other types of footnotes, it will insert it as a regular footnote.

\insert@txtbeforenotesX is the same for familiar footnotes.

```
\newcommand{\insert@Xtxtbeforenotes}[1]{%
  \nottoggle{Xtxtbeforesnotes@#1@typeset}{%
    \global\toggletrue{Xtxtbeforesnotes@#1@typeset}%
    \ifcsvoid{Xtxtbeforenotes@#1}{}{%
      \ifcsstring{series@display#1}{paragraph}%
        {\noindent\csuse{Xtxtbeforenotes@#1}}%
        {\expandafter\insert\csname#1footins\endcsname%
          \bgroup%
            \noindent%
            \ifcsdef{\csuse{series@display#1}@begin@insert}{%
              \csuse{\csuse{series@display#1}@begin@insert}{#1}%
            }{}%
            \strut\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}%
          \egroup%
        }%
    }%
  }%
  {}%
}%


\newcommand{\insert@txtbeforenotesX}[1]{%
  \nottoggle{txtbeforesnotesX@#1@typeset}{%
    \global\toggletrue{txtbeforesnotesX@#1@typeset}%
    \ifcsvoid{txtbeforenotesX@#1}{}{%
      \ifcsstring{series@displayX#1}{paragraph}%
        {\noindent\csuse{txtbeforenotesX@#1}}%
        {\expandafter\insert\csname footins#1\endcsname%
          \bgroup%
            \noindent%
            \ifcsdef{\csuse{series@displayX#1}@begin@insert}{%
              \csuse{\csuse{series@displayX#1}@begin@insert}{#1}%
            }{}%
            \strut\csuse{notefontsizeX@#1}\csuse{txtbeforenotesX@#1}%
          \egroup%
        }%
    }%
  }%
  {}%
}%


%
```

XII Critical footnotes

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

XII.1 Fonts

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

\select@lemmafont
\select@@lemmafont

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

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

%
```

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.

```
\newcommand*{\footnoteoptions@}[3]{%
   \def\do##1{%
      \ifstrequal{#1}{L}{% In Leftside
          \xright@appenditem{\noexpand\setkeys[mac]{#3footnoteoption}{\unexpanded{##1}}}\to\inserts@list%
         \global\advance\insert@count \@ne% Increment the left insert counter.
      }%
      {%
          \xright@appenditem{\noexpand\setkeys[mac]{#3footnoteoption}{\unexpanded{##1}}}\to\inserts@listR%
          \global\advance\insert@countR \@ne% Increment the right insert counter insert.
      }%
   }%
   \notblank{#2}{\docsvlist{#2}}{}% Parsing all options
}
%
```

XII.3 Notes language

\footnotelang@lua \footnotelang@lua is called to remember the information about the direction of a lemma when LuaLaTeX is used.

```
\newcommandx*{\footnotelang@lua}[1][1=L,usedefault]{%
   \ifstrequal{#1}{L}{%
   \xright@appenditem{{\csxdef{footnote@luatextextdir}{\the\textdir}}}\to\
inserts@list%Know the dir of lemma
    \global\advance\insert@count \@ne%
    \xright@appenditem{{\csxdef{footnote@luatexpardir}{\the\pardir}}}\to\
inserts@list%Know the dir of lemma
    \global\advance\insert@count \@ne%
   }%
   {%
   \xright@appenditem{{\csxdef{footnote@luatextextdir}{\the\textdir}}}\to\
inserts@listR%Know the dir of lemma
    \global\advance\insert@countR \@ne%
    \xright@appenditem{{\csxdef{footnote@luatexpardir}{\the\pardir}}}\to\
inserts@listR%Know the dir of lemma
    \global\advance\insert@countR \@ne%
   }%
}
%
```

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

```
\newcommandx*{\footnotelang@poly}[1][1=L,usedefault]{%
   \ifstrequal{#1}{L}{%
    \if@RTL%
        \xright@appenditem{{\csxdef{footnote@dir}{@RTLtrue}}}\to\
inserts@list%Know the language used in the lemma
        \global\advance\insert@count \@ne%
    \else
         \xright@appenditem{{\csxdef{footnote@dir}{@RTLfalse}}}\to\
inserts@list%Know the language of lemma
         \global\advance\insert@count \@ne%
    \fi%
    \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\languagename}}}\
to\inserts@list%Know the language of lemma
    \global\advance\insert@count \@ne%
   }%
   {%
    \if@RTL
        \xright@appenditem{{\csxdef{footnote@dir}{@RTLtrue}}}\to\
inserts@listR%Know the language of lemma
        \global\advance\insert@countR \@ne%
    \else
         \xright@appenditem{{\csxdef{footnote@dir}{@RTLfalse}}}\to\
inserts@listR%Know the language of lemma
```

```
        \global\advance\insert@countR \@ne%
    \fi
    \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\languagename}}}\to\inserts@listR%Know the language of lemma
    \global\advance\insert@countR \@ne%
   }%
}
%
```

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 LaTeX `\floatingpenalty`.

```
\newcommand*{\footsplitskips}{%
  \interlinepenalty=\interfootnotelinepenalty
  \unless\ifl@dprintingpages%
    \floatingpenalty=\@MM%
  \fi%
  \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
  \leftskip=\z@skip \rightskip=\z@skip}

%
```

\normalfootnoterule

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

```
\let\normalfootnoterule=\footnoterule
%
```

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.

```
\newcommandx{\Xarrangement}[2][1,usedefault]{%
  \def\do##1{%
     \csname Xarrangement@#2\endcsname{##1}%
  }%
  \ifstrempty{#1}%
    {%
    \dolistloop{\@series}%
    }%
    {
    \docsvlist{#1}%
    }%
}%
%
```

XII.6.2 Normal footnote

\Xarrangement@normal

We can now define all the parameters for the series of footnotes; initially they use the “normal” footnote formatting.

What we want to do here is to insert something like the following for each footnote series. (This is an example, not part of the actual `reledmac` code.)

```
\skip\Afootins=12pt plus5pt minus5pt
\count\Afootins=1000
\dimen\Afootins=0.8\vsize
\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.

```
\newcommand*{\Xarrangement@normal}[1]{%
  \csgdef{series@display#1}{normal}
  \expandafter\let\csname #1footstart\endcsname=\normalfootstart
  \expandafter\let\csname v#1footnote\endcsname=\normalvfootnote
  \expandafter\let\csname #1footfmt\endcsname=\normalfootfmt
  \expandafter\let\csname #1footgroup\endcsname=\normalfootgroup
  \expandafter\let\csname #1footnoterule\endcsname=%
                                          \normalfootnoterule
  \count\csname #1footins\endcsname=1000
  \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
  \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
  \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
%
```

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.

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

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).

```
  \ifnoledgroup@\else%
    \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
    \expandafter\let\csname mp#1footgroup\endcsname=\mpnormalfootgroup
    \count\csname mp#1footins\endcsname=1000
    \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
    \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
    \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
  \fi
}

%
```

`\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.

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnote}[2]{%
  \iftoggle{Xgroupbyline@#1}{%In the case we use \Xgroupbyline, the insertion is done later, in \add@Xgroupbyline.
    \prepare@Xgroupbyline{#1}{#2}{\normalvfootnote@inserted}%
  }{%In the case we don't use \Xgroupbyline, the insertion is made directly
    \X@beforeinsertion{#1}%
    \insert\csname #1footins\endcsname{%
      \X@atbegininsertion{#1}%
      \normalvfootnote@inserted{#1}{#2}%
    }%
  }%
}%
%
```

\normalvfootnote@inserted

The \normalvfootnote@inserted macro is expanded to the content to be add to a \insert for normal critical footnote.

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnote@inserted}[2]{%
  \nottoggle{Xgroupbyline@#1}{\noindent}{}\csuse{Xbhooknote@#1}%
  \csuse{Xnotefontsize@#1}%
  \footsplitskips
  \ifl@dpairing\ifl@dpaging\else%
    \setXnoteswidthliketwocolumns@{#1}%
  \fi\fi%
  \setXnotespositionliketwocolumns@{#1}%
  \spaceskip=\z@skip \xspaceskip=\z@skip%
  \csname #1footfmt\endcsname #2{#1}%
}%
%
```

\X@beforeinsertion

```
\newcommand{\X@beforeinsertion}[1]{%
  \if@ledgroup\else%
    \insert@Xtxtbeforenotes{#1}%
  \fi%
  \csuse{Xbeforeinserting@#1}%
}%
%
```

\beforeinsertion@X

```
\newcommand{\beforeinsertion@X}[1]{%
  \if@ledgroup\else%
    \insert@txtbeforenotesX{#1}%
  \fi%
  \csuse{beforeinsertingX@#1}%
}%
%
```

\X@atbegininsertion

```
\newcommand{\X@atbegininsertion}[1]{%
  \hsize=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
}%
%
```

And somewhat different versions of `\normalvfootnote` and `\normalvfootnote@inserted` for minipages.

\mpnormalvfootnote

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\mpnormalvfootnote}[2]{%
  \iftoggle{Xgroupbyline@#1}{%
    \prepare@Xgroupbyline{#1}{#2}{\mpnormalvfootnote@inserted}%
  }%
  {%
      \global\setbox\@nameuse{mp#1footins}%
      \vbox{%
        \unvbox\@nameuse{mp#1footins}%
        \mpnormalvfootnote@inserted{#1}{#2}%
    }%
  }%
}%

%
```

\mpnormalvfootnote@inserted

```
\newcommand{\mpnormalvfootnote@inserted}[2]{%
    \noindent\csuse{Xbhooknote@#1}%
    \csuse{Xnotefontsize@#1}%
    \hsize\columnwidth%
    \@parboxrestore%
    \color@begingroup%
    \csname #1footfmt\endcsname #2{#1}\color@endgroup%
}%
%
```

\normalfootfmt

`\normalfootfmt` is a 'normal' macro to take the footnote line and page number information (see V.9 p. 99), 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.

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmt}[4]{%
  \Xstorelineinfo{#1}{#4}%
  \nottoggle{Xgroupbyline@#4}{\Xledsetnormalparstuff{#4}}{}%
  \hangindent=\csuse{Xhangindent@#4}%
  \everypar{\hangindent=\csuse{Xhangindent@#4}}%
  \nottoggle{Xgroupbyline@#4}{\rule\z@\splittopskip}{}%
  {\printlinefootnote{#1}{#4}}%
  \print@lemma{#1}{#2}{#4}%
  \csuse{Xwrapcontent@#4}{#3}%
  \nottoggle{Xgroupbyline@#4}{\strut\par}{}%
}%
%
```

\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 \Xprenotes@ 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.

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

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

```
  \ifdimequal{0pt}{\Xprenotes@}{}%
      {%
      \iftoggle{Xprenotes@}{%
            \togglefalse{Xprenotes@}%
            \skip\csname #1footins\endcsname=%
              \glueexpr\csuse{Xprenotes@}+\csuse{Xafterrule@#1}\relax%
            }%
          {}%
      }%
  \vskip\skip\csname #1footins\endcsname%
%
```

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. 219 for the general description of the problem.

```
  \leftskip0pt \rightskip0pt
  \ifl@dpairing\else%
     \hsize=\old@hsize%
  \fi%
  \setXnoteswidthliketwocolumns@{#1}%
  \setXnotespositionliketwocolumns@{#1}%
%
```

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

```
  \print@Xfootnoterule{#1}%
}%
%
```

\normalfootgroup

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

```
\newcommand*{\normalfootgroup}[1]{%
  \csuse{Xbhookgroup@#1}%
  \unvbox\csname #1footins\endcsname%
  \hsize=\old@hsize%
  }%

%
```

\mpnormalfootgroup

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

```
\unless\ifnoledgroup@
\newcommand*{\mpnormalfootgroup}[1]{{
  \vskip\skip\@nameuse{mp#1footins}
  \ifl@dpairing\ifparledgroup%
    \leavevmode\marks\parledgroup@{begin}%
    \marks\parledgroup@series{#1}%
    \marks\parledgroup@type{Xfootnote}%
  \fi\fi\normalcolor%
  \ifparledgroup%
    \ifl@dpairing%
    \else%
      \setXnoteswidthliketwocolumns@{#1}%
      \setXnotespositionliketwocolumns@{#1}%
      \print@Xfootnoterule{#1}%%
    \fi%
  \else%
    \setXnoteswidthliketwocolumns@{#1}%
    \setXnotespositionliketwocolumns@{#1}%
    \print@Xfootnoterule{#1}%%
  \fi%
  \setlength{\parindent}{0pt}
 \csuse{Xbhookgroup@#1}%
 \unvbox\csname mp#1footins\endcsname}}
\fi
%
```

XII.6.3 Paragraphed footnotes

The paragraphed-footnote option reformats all the footnotes of one series for a page into a single paragraph; this is especially appropriate when the notes are numerous and brief. The code is based on *The TeXbook*, pp. 398–400, with alterations for our environment. This algorithm uses a considerable amount of save-stack space: a TeX of ordinary size may not be able to handle more than about 100 notes of this kind on a page.

`\Xarrangement@paragraph`

The `\Xarrangement@paragraph` macro sets up everything for one series of the footnotes so that they will be paragraphed; it takes the series letter as argument. We include the setting of `\count\footins` to 1000 for the footnote series just in case user is switching to paragraphed footnotes after having columnar ones, since they change this value (see below).

The argument of `\Xarrangement@footparagraph` is the letter denoting the series of notes to be paragraphed.

```
\newcommand*{\Xarrangement@paragraph}[1]{%
  \csgdef{series@display#1}{paragraph}
  \expandafter\let\csname #1footstart\endcsname=\parafootstart
  \expandafter\let\csname v#1footnote\endcsname=\paravfootnote
  \expandafter\let\csname #1footfmt\endcsname=\parafootfmt
  \expandafter\let\csname #1footgroup\endcsname=\parafootgroup
  \count\csname #1footins\endcsname=1000
  \csxdef{default@#1footins}{1000}%Use this to confine the  notes to one side only
  \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
  \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
  \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
  \para@footsetup{#1}
%
```

And the extra setup for minipages.

```
  \ifnoledgroup@\else
    \expandafter\let\csname mpv#1footnote\endcsname=\mpparavfootnote
    \expandafter\let\csname mp#1footgroup\endcsname=\mpparafootgroup
    \count\csname mp#1footins\endcsname=1000
    \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
    \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
    \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
  \fi
}
%
```

`\footfudgefiddle`

For paragraphed footnotes TeX has to estimate the amount of space required. If it underestimates this then the notes may get too long and run off the bottom of the text block. `\footfudgefiddle` can be increased from its default 64 (say, to 70) to increase the estimate.

```
\providecommand{\footfudgefiddle}{64}
%
```

\para@footsetup

\footparagraph calls the \para@footsetup macro to calculate a special fudge factor, which is the ratio of the \baselineskip to the \hsize. We assume that the proper value of \baselineskip for the footnotes (normally 9 pt) has been set already. The argument of the macro is again the note series letter.

Peter Wilson thinks that \columnwidth should be used here for LaTeX not \hsize. Peter Wilson have also included \footfudgefiddle.

```
\newcommand*{\para@footsetup}[1]{{\csuse{Xbhookgroup@#1}\csuse{Xnotefontsize@#1}
  \setXnoteswidthliketwocolumns@{#1}%
  \ifcsempty{Xwidth@#1}%
    {}%
    {\columnwidth=\expandafter\dimexpr\csuse{Xwidth@#1}\relax}%
  \dimen0=\baselineskip
  \multiply\dimen0 by 1024
  \divide \dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\relax
  \csxdef{#1footfudgefactor}{%
    \expandafter\strip@pt\dimen0 }}}

%
```

\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.

```
\newcommand*{\parafootstart}[1]{%
  \rightskip=0pt \leftskip=0pt%
  \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
    \ifdimequal{0pt}{\Xprenotes@}{}%
      {%
      \iftoggle{Xprenotes@}{%
            \togglefalse{Xprenotes@}%
            \skip\csname #1footins\endcsname=%
              \glueexpr\csuse{Xprenotes@}+\csuse{Xafterrule@#1}\relax%
            }%
          {}%
      }%
  \vskip\skip\csname #1footins\endcsname%
  \setXnoteswidthliketwocolumns@{#1}%
  \setXnotespositionliketwocolumns@{#1}%
  \print@Xfootnoterule{#1}%
  \let\bidi@RTL@everypar\@empty%
  \noindent\leavevmode}
%
```

\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 \discretionarys. 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.[29]

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.[30] 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.[31] 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.[32] In other words, be very careful not to use \break, or \penalty-10000, or any equivalent inside your para-footnote. If you do, most of the note will probably disappear. You *are* allowed to make strong suggestions; in fact \penalty-9999 will be quite okay. Just do not make the break mandatory. We have not applied any of Michael's solutions here, since we feel that the problem is exiguous, and reledmac is quite baroque enough already. If you think you are having this problem, look up Michael's solutions.

One more thing; we set \leftskip and \rightskip to zero. This has the effect of neutralizing any such skips which may apply to the main text (cf. XII.6.2 p. 171 above). We need to do this, since \footfudgefactor is calculated on the assumption that the

[29] Michael Downes, 'Line Breaking in \unhboxed Text', *TUGboat* **11** (1990), pp. 605–612.

[30] See *The TeXbook*, p. 455 (editions after January 1990).

[31] 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.

[32] 'Line Breaking', p. 610.

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.

```
\newcommand*{\paravfootnote}[2]{%
  \csuse{Xbeforeinserting@#1}%
  \insert\csname #1footins\endcsname
  \bgroup
    \csuse{Xnotefontsize@#1}
    \footsplitskips
    \setbox0=\vbox{\hsize=\maxdimen%
      \let\bidi@RTL@everypar\@empty%
      \insert@Xtxtbeforenotes{#1}%
      \noindent\csuse{Xbhooknote@#1}%
      \csname #1footfmt\endcsname #2{#1}}%
    \setbox0=\hbox{\Xunvxh{0}{#1}}%
    \dp0=0pt
    \ht0=\csname #1footfudgefactor\endcsname\wd0
%
```

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

```
    \if@RTL\noindent \leavevmode\fi\box0%
    \penalty0
  \egroup}

%
```

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.

```
\newcommand*{\mpparavfootnote}[2]{%
  \global\setbox\@nameuse{mp#1footins}\vbox{%
    \unvbox\@nameuse{mp#1footins}%
    \csuse{Xnotefontsize@#1}
    \footsplitskips
    \setbox0=\vbox{\hsize=\maxdimen%
      \let\bidi@RTL@everypar\@empty%
      \insert@Xtxtbeforenotes{#1}%
      \noindent\color@begingroup%
      \csuse{Xbhooknote@#1}%
      \csname #1footfmt\endcsname #2{#1}\color@endgroup}%
    \setbox0=\hbox{\Xunvxh{0}{#1}}%
    \dp0=\z@
    \ht0=\csname #1footfudgefactor\endcsname\wd0
    \box0
    \penalty0
}}

%
```

\Xunvxh Here is (modified) Michael's definition of \unvxh, used above. Michael's macro also takes care to remove some unwanted penalties and glue that TeX automatically attaches to the end of paragraphs. When TeX finishes a paragraph, it throws away any remaining glue, and then tacks on the following items: a \penalty of 10000, a \parfillskip and a \rightskip (*The TeXbook*, pp. 99–100). \unvxh cancels these unwanted paragraph-final items using \unskip and \unpenalty.

```
\newcommand*{\Xunvxh}[2]{%
  \setbox0=\vbox{\unvbox#1%
    \global\setbox1=\lastbox}%
  \unhbox1
  \unskip            % remove \rightskip,
  \unskip            % remove \parfillskip,
  \unpenalty         % remove \penalty of 10000,
  \hskip\csuse{Xafternote@#2}\relax}% add the glue to go between the notes

%
```

\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).

```
\newcommand*{\parafootfmt}[4]{%
  \Xstorelineinfo{#1}{#4}%
  \Xinsertparafootsep{#4}%
  \ledsetnormalparstuff@common%
  \printlinefootnote{#1}{#4}%
  \print@lemma{#1}{#2}{#4}%
  \csuse{Xwrapcontent@#4}{#3}%
  \penalty-10 }
%
```

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 `\paravfootnote`.

The call to `\Xnotefontsize@`⟨*s*⟩ is to ensure that the correct `\baselineskip` for the footnotes is used. The argument is the note series letter.

```
\newcommand*{\parafootgroup}[1]{%
  \hsize=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
  \unvbox\csname #1footins\endcsname
  \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
  \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
  \makehboxofhboxes
  \setbox0=\hbox{\unhbox0 \removehboxes}%
  \csuse{Xbhookgroup@#1}%
  \csuse{Xnotefontsize@#1}%
  \unhbox0\par%
  \global\hsize=\old@hsize%
  }%

%
```

`\mpparafootgroup` The minipage version.

```
\newcommand*{\mpparafootgroup}[1]{{%
  \setXnoteswidthliketwocolumns@{#1}%
  \vskip\skip\@nameuse{mp#1footins}
  \ifl@dpairing\ifparledgroup%
    \leavevmode\marks\parledgroup@{begin}%
    \marks\parledgroup@series{#1}%
    \marks\parledgroup@type{Xfootnote}%
  \fi\fi\normalcolor
  \ifparledgroup%
    \ifl@dpairing%
    \else%
      \setXnoteswidthliketwocolumns@{#1}%
      \setXnotespositionliketwocolumns@{#1}%
      \print@Xfootnoterule{#1}%%
    \fi%
  \else%
      \setXnoteswidthliketwocolumns@{#1}%
      \setXnotespositionliketwocolumns@{#1}%
      \print@Xfootnoterule{#1}%
  \fi%
  \unvbox\csname mp#1footins\endcsname
  \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
  \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
  \makehboxofhboxes
  \setbox0=\hbox{\unhbox0 \removehboxes}%
  \csuse{Xbhookgroup@#1}%
  \csuse{Xnotefontsize@#1}%
  \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
  \unhbox0\par}}

%
```

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

\makehboxofhboxes
\removehboxes

```
\newcommand*{\makehboxofhboxes}{\setbox0=\hbox{}%
  \loop
    \unpenalty
    \setbox2=\lastbox
  \ifhbox2
    \setbox0=\hbox{\box2\unhbox0}%
  \repeat}

\newcommand*{\removehboxes}{\setbox0=\lastbox
  \ifhbox0{\removehboxes}\unhbox0 \fi}

%
```

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

\prevpage@num
\Xinsertparafootsep

```
\newcommand{\Xinsertparafootsep}[1]{%
   \ifledRcol@%
     \ifnumequal{\csuse{#1prevpage@numR}}{\page@numR}%
        {\ifcsdef{prevline#1}% Be sur \prevline#1 exists.
          {\ifcsequal{prevline#1}{lineinfo@}%
             {\ifcsempty{Xsymlinenum@#1}{\csuse{Xparafootsep@#1}}{}}%
             {\csuse{Xparafootsep@#1}}%
          }%
          {\csuse{Xparafootsep@#1}}%
     }%
     {}%
     \global\csname #1prevpage@numR\endcsname=\page@numR%
   \else%
     \ifnumequal{\csuse{#1prevpage@num}}{\page@num}%
        {\ifcsdef{prevline#1}% Be sur \prevline#1 exists.
          {\ifcsequal{prevline#1}{lineinfo@}%
             {\ifcsempty{Xsymlinenum@#1}{\csuse{Xparafootsep@#1}}{}}%
             {\csuse{Xparafootsep@#1}}%
          }%
        {\csuse{Xparafootsep@#1}}%
     }%
     {}%
     \global\csname #1prevpage@num\endcsname=\page@num%
   \fi%
}
%
```

XII.6.4 Columnar footnotes

Common tools

\rigidbalance \rigidbalanceX \Xrigidbalance \dosplits \splitoff \@h \@k

We will now define macros for three-column notes and two-column notes. Both sets of macros will use `\rigidbalance`, which splits a box (#1) into into a number (#2) of columns, each with a space (#3) between the top baseline and the top of the `\vbox`. The `\rigidbalance` macro is taken from *The TeXbook*, p. 397, with a slight change to the syntax of the arguments so that they do not depend on white space. Note also the extra unboxing in `\splitoff`, which allows the new `\vbox` to have its natural height as it goes into the alignment.

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

We do not call directly `\rigidbalance`, but we call `\Xrigidbalance` for critical notes and `\rigidbalanceX` for familiar notes. Both of them call `\rigidbalance`.

```
\newcount\@k \newdimen\@h
\newcommand*{\Xrigidbalance}[3]{%
    \hsize=\expandafter\dimexpr\csuse{Xwidth@\@currentseries}\relax%
    \rigidbalance{#1}{#2}{#3}%
}%

\newcommand*{\rigidbalanceX}[3]{%
    \hsize=\expandafter\dimexpr\csuse{widthX@\@currentseries}\relax%
    \rigidbalance{#1}{#2}{#3}%
}%

\newcommand*{\rigidbalance}[3]{%
  \setbox0=\box#1 \@k=#2 \@h=#3%
  \@@line{\splittopskip=\@h \vbadness=\@M \hfilneg
  \valign{##\vfil\cr\dosplits}}}

\newcommand*{\dosplits}{\ifnum\@k>0 \noalign{\hfil}\splitoff
  \global\advance\@k-1\cr\dosplits\fi}

\newcommand*{\splitoff}{\dimen0=\ht0
  \divide\dimen0 by\@k \advance\dimen0 by\@h
  \setbox2 \vsplit0 to \dimen0
  \unvbox2 }

%
```

Three columns

\Xarrangement@threecol

```
\newcommand*{\Xarrangement@threecol}[1]{%
  \csgdef{series@display#1}{threecol}
  \expandafter\let\csname v#1footnote\endcsname=\threecolvfootnote
  \expandafter\let\csname #1footfmt\endcsname=\threecolfootfmt
  \expandafter\let\csname #1footgroup\endcsname=\threecolfootgroup
  \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
  \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
  \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
  \threecolfootsetup{#1}
%
```

The additional setup for minipages.

```
  \ifnoledgroup@\else
    \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
    \expandafter\let\csname mp#1footgroup\endcsname=\mpthreecolfootgroup
    \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
    \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
    \mpthreecolfootsetup{#1}
  \fi
}

%
```

The `\footstart` and `\footnoterule` macros for these notes assume the normal values (XII.6.2 p. 171 above).

\threecolfootsetup

The `\threecolfootsetup` macro calculates and sets some numbers for three-column footnotes.

We set the `\count` of the foot insert to 333. Each footnote can be thought of as contributing only one third of its height to the page, since the footnote insertion has been made as a long narrow column, which then gets trisected by the `\rigidbalance` routine (inside `\threecolfootgroup`). These new, shorter columns are saved in a box, and then that box is *put back* into the footnote insert, replacing the original collection of the footnotes. This new box is, therefore, only about a third of the height of the original one.

The `\dimen` value for this note series has to change in the inverse way: it needs to be three times the actual limit on the amount of space these notes are allowed to fill on the page, because when TEX is accumulating material for the page and checking that limit, it does not apply the `\count` scaling.

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

\mpthreecolfootsetup

The setup for minipages.

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

%
```

\threecolvfootnote

\threecolvfootnote This is the \vfootnote command for three-column notes. However, most of the code is deported on \threecolvfootnote@inserted. The call to \Xnotefontsize@$\langle s \rangle$ ensures that the \splittopskip and \splitmaxdepth take their values from the right \strutbox: the one used in a footnotes. Note especially the importance of temporarily reducing the \hsize to 0.3 of its normal value. This determines the widths of the individual columns. So if the normal \hsize is (say) 10 cm, then each column will be $0.3 \times 10 = 3$ cm wide, leaving a gap of 1 cm spread equally between columns (i.e., .5 cm between each).

The arguments are #1 the note series letter and #1 the full text of the note (including numbers, lemma and text).

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolvfootnote}[2]{%
  \iftoggle{Xgroupbyline@#1}{%
    \prepare@Xgroupbyline{#1}{#2}{\threecolvfootnote@inserted}%
  }%
  {%
    \X@beforeinsertion{#1}%
    \insert\csname #1footins\endcsname{%
      \threecolvfootnote@inserted{#1}{#2}%
    }%
  }%
}%
%
```

\threecolvfootnote@inserted

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\
threecolvfootnote@inserted}[2]{%
  \hsize=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
  \noindent\csuse{Xbhooknote@#1}%
  \csuse{Xnotefontsize@#1}%
  \footsplitskips%
  \csname #1footfmt\endcsname #2{#1}%
}%
%
```

\threecolfootfmt

\threecolfootfmt is the command that formats one note. The arguments are #1 the line numbers, #2 the lemma and #4 the text of the -footnote command #4 optional (for backward compatibility): the series.

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmt}[4]{%
  \Xstorelineinfo{#1}{#4}%
  \threecol@begin@insert{#4}%
  \hspace{\parindent}%
  \printlinefootnote{#1}{#4}%
  \print@lemma{#1}{#2}{#4}%
  \csuse{Xwrapcontent@#4}{#3}%
  \nottoggle{Xgroupbyline@#4}%
    {\strut\par\allowbreak}%
    {}%
}%
%
```

\threecol@begin@insert

The \threecol@begin@insert contains code used at the beginning of any \insert for critical footnotes in three columns. It is used both by \threecolfootfmt and by \insert@Xtxtbeforenotes.

```
\newcommand{\threecol@begin@insert}[1]{%
  \normal@pars%
  \nottoggle{Xgroupbyline@#1}%
    {\hsize \csuse{Xhsizethreecol@#1}}%
    {}%
  \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
  \tolerance=5000%
  \hangindent=\csuse{Xhangindent@#1}%
  \everypar{\hangindent=\csuse{Xhangindent@#1}}%
  \@tempdima=\parindent%
  \csuse{Xcolalign@#1}%
  \parindent=\@tempdima%
  \strut%
}%
%
```

\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 ouput 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.

```
\newcommand*{\threecolfootgroup}[1]{%
  \csuse{Xbhookgroup@#1}\par%
  \splittopskip=\ht\strutbox
  \expandafter
  \Xrigidbalance\csname #1footins\endcsname \thr@@ \splittopskip}
%
```

\mpthreecolfootgroup

The setup for minipages.

```
\newcommand*{\mpthreecolfootgroup}[1]{{%
```

```
  \vskip\skip\@nameuse{mp#1footins}
  \ifl@dpairing\ifparledgroup%
    \leavevmode\marks\parledgroup@{begin}%
    \marks\parledgroup@series{#1}%
    \marks\parledgroup@type{Xfootnote}%
  \fi\fi\normalcolor
  \ifparledgroup%
    \ifl@dpairing%
    \else%
      \setXnoteswidthliketwocolumns@{#1}%
      \setXnotespositionliketwocolumns@{#1}%
      \print@Xfootnoterule{#1}%
    \fi%
  \else%
    \setXnoteswidthliketwocolumns@{#1}%
    \setXnotespositionliketwocolumns@{#1}%
    \print@Xfootnoterule{#1}%
  \fi%
  \csuse{Xbhookgroup@#1}\par%
  \splittopskip=\ht\strutbox
  \expandafter
  \Xrigidbalance\csname mp#1footins\endcsname \thr@@ \splittopskip}}

%
```

Two columns

\Xarrangement@twocol

```
\newcommand*{\Xarrangement@twocol}[1]{%
  \csgdef{series@display#1}{twocol}
  \expandafter\let\csname v#1footnote\endcsname=\twocolvfootnote
  \expandafter\let\csname #1footfmt\endcsname=\twocolfootfmt
  \expandafter\let\csname #1footgroup\endcsname=\twocolfootgroup
  \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
  \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
  \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
  \twocolfootsetup{#1}
%
```

The additional setup for minipages.

```
  \ifnoledgroup@\else
    \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
    \expandafter\let\csname mp#1footgroup\endcsname=\mptwocolfootgroup
    \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
    \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
    \mptwocolfootsetup{#1}
  \fi
}

%
```

\twocolfootsetup
\twocolvfootnote
\twocolvfootnote@inserted
\twocolfootfmt
\twocolfootgroup

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 \hsize.

```
\newcommand*{\twocolfootsetup}[1]{%
  \count\csname #1footins\endcsname 500
  \csxdef{default@#1footins}{500}%
  \multiply\dimen\csname #1footins\endcsname \tw@}
%
```

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolvfootnote}[2]{%
  \iftoggle{Xgroupbyline@#1}{%
    \prepare@Xgroupbyline{#1}{#2}{\twocolvfootnote@inserted}%
  }{%
    \X@beforeinsertion{#1}%
    \insert\csname #1footins\endcsname{%
      \twocolvfootnote@inserted{#1}{#2}%
    }%
  }%
}%
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolvfootnote@inserted}[2]{%
  \hsize=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
  \noindent\csuse{Xbhooknote@#1}%
  \csuse{Xnotefontsize@#1}%
  \footsplitskips%
  \csname #1footfmt\endcsname #2{#1}%
}%
%
```

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolfootfmt}[4]{% 4th arg is optional, for backward compatibility
  \Xstorelineinfo{#1}{#4}%
  \twocol@begin@insert{#4}%
  \hspace{\parindent}%
  \printlinefootnote{#1}{#4}%
  \print@lemma{#1}{#2}{#4}%
  \csuse{Xwrapcontent@#4}{#3}%
  \nottoggle{Xgroupbyline@#4}%
    {\strut\par\allowbreak}%
    {}%
}%
%
```

```
\newcommand{\twocol@begin@insert}[1]{%
  \normal@pars%
  \hsize \csuse{Xhsizetwocol@#1}%
  \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
  \tolerance=5000%
  \hangindent=\csuse{Xhangindent@#1}%
  \everypar{\hangindent=\csuse{Xhangindent@#1}}%
  \@tempdima=\parindent%
  \csuse{Xcolalign@#1}%
  \parindent=\@tempdima%
  \strut%
}%

\newcommand*{\twocolfootgroup}[1]{%
  \csuse{Xbhookgroup@#1}\par%
  \splittopskip=\ht\strutbox
  \expandafter
  \Xrigidbalance\csname #1footins\endcsname \tw@ \splittopskip}

%
```

\mptwocolfootsetup
\mptwocolfootgroup

The versions for minipages.

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

```
\newcommand*{\mptwocolfootgroup}[1]{{%
  \vskip\skip\@nameuse{mp#1footins}
  \ifl@dpairing\ifparledgroup%
    \leavevmode\marks\parledgroup@{begin}%
    \marks\parledgroup@series{#1}%
    \marks\parledgroup@type{Xfootnote}%
  \fi\fi\normalcolor
  \ifparledgroup%
    \ifl@dpairing%
    \else%
      \setXnoteswidthliketwocolumns@{#1}%
      \setXnotespositionliketwocolumns@{#1}%
      \print@Xfootnoterule{#1}%
    \fi%
  \else%
    \setXnoteswidthliketwocolumns@{#1}%
    \setXnotespositionliketwocolumns@{#1}%
    \print@Xfootnoterule{#1}%
  \fi%
   \csuse{Xbhookgroup@#1}\par%
  \splittopskip=\ht\strutbox
  \expandafter
  \Xrigidbalance\csname mp#1footins\endcsname \tw@ \splittopskip}}

%
```

XII.7 Critical notes presentation

Here, we define some commons macro which are used in order to print a critical notes, that is a note with 1) line number 2) lemma 3) lemma separator 4) text associated to the lemma.

XII.7.1 Font tools

\endashchar
\fullstop
\rbracket

The fonts that are used for printing notes might not have the character mapping we expect: for example, the Computer Modern font that contains old-style numerals does not contain an en-dash or square brackets, and its period and comma are in odd locations. To allow use of the standard footnote macros with such fonts, we use the following macros for certain characters.

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

Note that these commands are not directly called by reledmac, but are enclosed as default value of specific hooks. Consequently, people should not redefine them, but use instead the \Xlinerangeseparator, \Xendlinerangeseparator, \Xsublinesep, \Xendsublinesep and \Xlemmaseparator macros.

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.

```
\def\endashchar{\textnormal{--}}

\newcommand*{\fullstop}{\textnormal{.}}
\def\Xsublinesep@side{\fullstop}

\newcommand*{\rbracket}{\textnormal{%
   \csuse{text\csuse{footnote@lang}}{%
         \ifluatex%
         \ifdefstring{\footnote@luatextextdir}{TRT}{\thinspace[}{\thinspace]}%
            \else%
         \thinspace]%
         \fi}%
   }%
}

%
```

XII.7.2 Pstart number in footnote

\printpstart

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

```
\newcommand{\printpstart}[0]{%
```

```
    \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
        \ifledRcol%
            \thepstartR%
        \else%
            \thepstartL%
        \fi%
    }{%
        \thepstart%
    }%
}
%
```

XII.7.3 Lemma printing

\print@lemma

\print@lemma is called inside critical footnotes to print the lemma and the lemma separator (#1: line number and font information, #2: lemma, #3: series).

```
%
\newcommand{\print@lemma}[3]{%
  \bgroup%
    \nottoggle{Xlemmadisablefontselection@#3}%
      {\select@lemmafont#1|}%
      {}%
    \bgroup%
      \csuse{Xlemmafont@#3}%Deprecated
      \csuse{Xwraplemma@#3}{#2}%
    \egroup%
  \egroup%
  \iftoggle{nosep@}{%
    \hskip\csuse{Xinplaceoflemmaseparator@#3}%
    \relax%
    }%
    {\ifcsempty{Xlemmaseparator@#3}%
      {%
        \hskip\csuse{Xinplaceoflemmaseparator@#3}%
        \relax%
      }%
      {%
        \nobreak%
        \hskip\csuse{Xbeforelemmaseparator@#3}%
        \csuse{Xlemmaseparator@#3}%
        \hskip\csuse{Xafterlemmaseparator@#3}%
        \relax%
      }%
  }%
}%
%
```

XII.7.4 Line number printing

\Xstorelineinfo

The \Xstorelineinfo macro is used to store some data about line number of the current critical footnote, data which will be reused later for the \Xnumberonlyfirstinline and related setting.

#1 footnote specification for the current footnote ; #2 footnote series.

```
\newcommand{\Xstorelineinfo}[2]{%
   \l@dp@rsefootspec#1|%
   \iftoggle{Xnumberonlyfirstintwolines@#2}{%
       \xdef\lineinfo@{\l@dparsedstartline - \l@dparsedstartsub - \
l@dparsedendline - \l@dparsedendsub}%
       }%
       {%
       \xdef\lineinfo@{\l@dparsedstartline - \l@dparsedstartsub}%
       }%
}%
%
```

\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.

```
\newcommand{\printlinefootnote}[2]{%
   \iftoggle{nonum@}{%Try if the line number must printed for this specific
 not (by default, yes)
   \hspace{\csuse{Xinplaceofnumber@#2}}%
   }%
   {%
       {%
           \iftoggle{Xnonumber@#2}%Try if the line number must printed (by
default, yes)
           {%
           \hspace{\csuse{Xinplaceofnumber@#2}}%
           }%
           {%
              {\iftoggle{Xnumberonlyfirstinline@#2}% If for this series the
 line number must be printed only in the first time.
                 {%
                 \ifcsdef{prevline#2}%
                    {%Be sure the \prevline exists.
                    \ifcsequal{prevline#2}{lineinfo@}%Try it
                       {%
                       \ifcsempty{Xsymlinenum@#2}%
                        {%
                            \hspace{\csuse{Xinplaceofnumber@#2}}%
                        }%
                        {\printsymlinefootnotearea{#2}}%
                       }%
                       {%
                        \printlinefootnotearea{#1}{#2}%
                        }%
                    }%
                    {%
                    \printlinefootnotearea{#1}{#2}%
                    }%
              }%
              {%
              \printlinefootnotearea{#1}{#2}%
              }%
              \csxdef{prevline#2}{\lineinfo@}%
           }%
        }%
     }%
   }%
}
%
```

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

```
\newcommand{\printsymlinefootnotearea}[1]{%
  \hspace{\csuse{Xbeforesymlinenum@#1}}%
  \csuse{Xnotenumfont@#1}%
  \ifdimequal{\csuse{Xboxsymlinenum@#1}}{\z@}%
     {\csuse{Xsymlinenum@#1}}%
     {\hbox to \csuse{Xboxsymlinenum@#1}%
       {\csuse{Xsymlinenum@#1}\hfill}%
     }%
  \hspace{\csuse{Xaftersymlinenum@#1}}%
}%
%
```

\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.)

```
\newcommand{\printlinefootnotearea}[2]{%
  \printXbeforenumber{#2}%
  \csuse{Xnotenumfont@#2}%
  \boxfootnotenumbers{#1}{#2}%
  \printXafternumber{#2}%
}%
%
```

\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.

```
\newcommand{\boxfootnotenumbers}[2]{%
  \ifdimequal{\csuse{Xboxlinenum@#2}}{0pt}{%
       \printlinefootnotenumbers{#1}{#2}%
     }%
     {%
       \hbox to \csuse{Xboxlinenum@#2}%
         {%
         \IfSubStr{RC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
         \printlinefootnotenumbers{#1}{#2}%
         \IfSubStr{LC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
         }%
     }%
}%
%
```

\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.

```
\newcommand{\printlinefootnotenumbers}[2]{%
  \xdef\@currentseries{#2}%
  \ifboolexpr{%
   (togl{Xpstart@#2} and bool{numberpstart})%
   or togl{Xpstarteverytime@#2}}%
  {\printpstart}{}%
  \iftoggle{Xstanza@#2}{%
    \ifnumberstanza%
      \printstanza%
      \csuse{Xstanzaseparator@#2}%
    \fi%
  }{}%
  \iftoggle{Xonlypstart@#2}{}{%
    \csuse{Xtxtbeforenumber@#2}%
    \printlines#1|\ifledRcol@\@Rlineflag\fi|}%
}%
%
```

\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.)

```
\newcommand{\printXbeforenumber}[1]{%
  \hspace{\csuse{Xbeforenumber@#1}}%
}%
%
```

\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

```
\newcommand{\printXafternumber}[1]{%
  \iftoggle{Xnonbreakableafternumber@#1}{\nobreak}{}%
  \hspace{\csuse{Xafternumber@#1}}%
}%
%
```

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. 99: the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma.

`edmac`' creator have defined six boolean in order to know which component of line number description we have to print:

- `\ifl@d@pnum` for page numbers;
- `\ifl@d@ssub` for starting sub-line;
- `\ifl@d@elin` for ending line;
- `\ifl@d@esl` for ending sub-line; and
- `\ifl@d@dash` for the dash between the starting and ending groups.

There is no boolean for the line number because it is always printed.

Maïeul Rouquette has added `\ifl@d@Xtwolines` and `\ifl@d@Xmorethantwolines` to print a symbol which stands for "and subsequent" when there are two, three or more lines.

\ifl@d@pnum
\ifl@d@ssub
\ifl@d@elin
\ifl@d@esl
\ifl@d@dash
\ifl@d@Xtwolines
\ifl@d@Xmorethantwolines

```
\newif\ifl@d@pnum
\newif\ifl@d@ssub
\newif\ifl@d@elin
\newif\ifl@d@esl
\newif\ifl@d@dash
\newif\ifl@d@Xtwolines%
\newif\ifl@d@Xmorethantwolines%
%
```

\l@dparsefootspec
\l@dp@rsefootspec
\l@dparsedstartpage
\l@dparsedstartline
\l@dparsedstartsub
\l@dparsedendpage
\l@dparsedendline
\l@dparsedendsub

`\l@dparsefootspec{`⟨*spec*⟩`}{`⟨*lemma*⟩`}{`⟨*text*⟩`}` parses a footnote specification. ⟨*lemma*⟩ and ⟨*text*⟩ are the lemma and text respectively. ⟨*spec*⟩ is the line and page number and lemma font specifier in `\l@d@nums` style format. The real work is done by `\l@dp@rsefootspec` which defines macros holding the numeric values. In many cases, this last command is called directly. Just a reminder of the arguments:

```
\printlines    #1      |  #2  |   #3    |    #4    |  #5  |  #6     |   #7
\printlines start-page | line | subline | end-page | line | subline | fontflag
```

```
\newcommand*{\l@dparsefootspec}[3]{\l@dp@rsefootspec#1|}
\def\l@dp@rsefootspec#1|#2|#3|#4|#5|#6|#7|{%
  \gdef\l@dparsedstartpage{#1}%
  \gdef\l@dparsedstartline{#2}%
  \gdef\l@dparsedstartsub{#3}%
  \gdef\l@dparsedendpage{#4}%
  \gdef\l@dparsedendline{#5}%
  \gdef\l@dparsedendsub{#6}%
}
%
```

Initialise the several number value macros.

```
\def\l@dparsedstartpage{0}%
\def\l@dparsedstartline{0}%
\def\l@dparsedstartsub{0}%
\def\l@dparsedendpage{0}%
\def\l@dparsedendline{0}%
\def\l@dparsedendsub{0}%

%
```

\setprintlines

The macro \setprintlines does the work of deciding what numbers should be printed. Its arguments are the same as the first 6 of \printlines.

```
\newcommand*{\setprintlines}[6]{%
  \l@d@pnumfalse \l@d@dashfalse
%
```

We print the page numbers only if: 1) we are doing the lineation by page, and 2) the ending page number is different from the starting page number.a

```
  \ifbypage@
     \ifnum#4=#1 \else
        \l@d@pnumtrue
        \l@d@dashtrue
     \fi
  \fi
%
```

We print the ending line number if: (1) we are printing the ending page number, or (2) it is different from the starting line number.

```
  \ifl@d@pnum \l@d@elintrue \else \l@d@elinfalse \fi
  \ifnum#2=#5 \else
      \l@d@elintrue
      \l@d@dashtrue
  \fi
%
```

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

```
  \l@d@ssubfalse
  \ifnum#3=0 \else
      \l@d@ssubtrue
  \fi
%
```

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.

```
  \l@d@eslfalse
  \ifnum#6=0 \else
      \ifnum#6=#3
         \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
      \else
         \l@d@esltrue
         \l@d@dashtrue
      \fi
  \fi%
%
```

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

```
  \ifl@d@dash%
    \ifboolexpr{togl{fulllines@} or test{\ifcsempty{Xtwolines@\
@currentseries}}}%
      {}%
      {%
      \setistwofollowinglines{#1}{#2}{#4}{#5}%
        \ifboolexpr{%
          (%
               togl {Xtwolinesbutnotmore@\@currentseries}%
               and not%
                 (%
                  bool {istwofollowinglines@}%
                 )%
          )%
          or%
          (%
             (not test{\ifnumequal{#1}{#4}})%
               and togl{Xtwolinesonlyinsamepage@\@currentseries}%
             )%
          }%
          {}%
          {%
          \l@d@dashfalse%
          \l@d@Xtwolinestrue%
          \l@d@elinfalse%
          \l@d@eslfalse%
          \ifcsempty{Xmorethantwolines@\@currentseries}%
            {}%
            {\ifistwofollowinglines@\else%
              \l@d@Xmorethantwolinestrue%
             \fi%
            }%
          }%
      }%
  \fi%
%
```

End of \setprintlines.

```
}%
%
```

\setistwofollowinglines

The \ifistwofollowinglines boolean, used by the \Xtwolines and related setting, is set to true by \setistwofollowinglines. This command takes the following arguments:

- #1 First page number.
- #2 First line number.
- #3 Last page number.
- #4 Last line number.

If #3-#2 = 1, then that means the two lines are subsequent, and consequently \ifistwofollowinglines is set to true. However, if we use lineation by page, two given lines can be subsequent if:

- The first line number is equal to the last line number of the first page.
- The last line number is equal to 1.
- #3-#1 is equal to 1.

```
\newif\ifistwofollowinglines@%
\newcommand{\setistwofollowinglines}[4]{%
    \ifcsdef{lastlinenumberon@#1}%
      {\numdef{\tmp}{\csuse{lastlinenumberon@#1}}}%
      {\numdef{\tmp}{0}}%
    \istwofollowinglines@false%
    \ifnumequal{#4-#2}{1}%
    {\istwofollowinglines@true}%
    {\ifbypage@%
      \ifnumequal{#3-#1}{1}%
      {%
      \ifnumequal{#2}{\tmp}%
        {\ifnumequal{#4}{1}{\istwofollowinglines@true}{}}%
        {}%
      }%
      {}%
     \fi%
    }%
}%
%
```

\printlines So, we have decided which part of line number sets will be printed depending of these value. Now we are ready to print them. If the lineation is by pstart, we print the pstart. Arguments are 1) start page number 2) start line number 3) start subline number 4) end page number 5) end line number 6) end subline number 7) font specification 8) side flag

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

If we use LuaTEX, ensure we use good text's direction.

```
  \ifluatex%
    \edef\@tmp{\the\textdir}%
    \ifdefstring{\@tmp}{TLT}{}{\textdir TLT}%Test in order to prevent spurious space (bug #397)
  \fi%
%
```

Decide which part of line number components we will print.

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

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.

```
  \ifdimequal{\csuse{Xboxstartlinenum@\@currentseries}}{0pt}%
    {\bgroup}%
    {\leavevmode\hbox to \csuse{Xboxstartlinenum@\@currentseries}\bgroup\hfill}%
  \ifl@d@pnum%
    \wrap@edcrossref{\@this@crossref@start}{#1}%
    \csuse{Xpagelinesep@\@currentseries}%
  \fi%
  \wrap@edcrossref{\@this@crossref@start}{%
    \linenumrep{#2}%
    \iftoggle{Xlineflag@\@currentseries}{#8}{}%
  }%
  \ifl@d@ssub%
    \csuse{Xsublinesep@\@currentseries}%
    \wrap@edcrossref{\@this@crossref@start}{\sublinenumrep{#3}}%
  \fi
  \egroup%
%
```

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

```
  \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
    {\bgroup}%
    {\hbox to \csuse{Xboxendlinenum@\@currentseries}\bgroup}%
  \ifl@d@Xtwolines%
```

```
    \ifl@d@Xmorethantwolines%
      \csuse{Xmorethantwolines@\@currentseries}%
    \else%
      \csuse{Xtwolines@\@currentseries}%
    \fi%
  \else%
    \ifl@d@dash%
      \ifdefined\linerangesep@%
        \linerangesep@%
      \else%
        \csuse{Xlinerangeseparator@\@currentseries}%
      \fi%
    \fi%
    \ifl@d@pnum%
       \wrap@edcrossref{\@this@crossref@end}{#4}%
       \csuse{Xpagelinesep@\@currentseries}%
    \fi%
    \ifl@d@elin%
      \wrap@edcrossref{\@this@crossref@end}{%
        \linenumrep{#5}%
        \iftoggle{Xlineflag@\@currentseries}{#8}{}%
       }%
    \fi%
    \ifl@d@esl%
       \ifl@d@elin%
         \csuse{Xsublinesep@\@currentseries}%
       \fi%
       \wrap@edcrossref{\@this@crossref@end}{\sublinenumrep{#6}}%
     \fi%
  \fi%
  \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
    {}%
    {\hfill}%Prevent underfull hbox
  \egroup%
  \endgroup%
}%
%
```

XII.7.5 Footnote grouped by line

\prepare@Xgroupbyline

\prepare@Xgroupbyline is a macro called on on the \⟨*XXX*⟩vfootnote if \Xgroupbyline is set to true, instead of calling directly the \insert.

```
\newcommand{\prepare@Xgroupbyline}[3]{%
  \iftoggle{Xgroupbylineseparetwolines@#1}{%
    \l@dparsefootspec#2%
    \ifcsdef{#1@forinserting@\l@dparsedendpage-\l@dparsedendline-\
l@dparsedendsub}%
    {%
      \csgappto%
    {#1@forinserting@\l@dparsedendpage-\l@dparsedendline-\l@dparsedendsub}%
    {%
     \ifcsempty{Xsymlinenum@#1}%
      {\csuse{Xparafootsep@#1}}%
      {}%
     #3{#1}{#2}%
     \hskip\csuse{Xafternote@#1}\relax%
    }%
  }%
  {%
   \csdef%
    {#1@forinserting@\l@dparsedendpage-\l@dparsedendline-\l@dparsedendsub}%
    {%
     #3{#1}{#2}%
     \hskip\csuse{Xafternote@#1}\relax%
    }%
  }%
  \listcsxadd{#1@forinserting}{\l@dparsedendpage-\l@dparsedendline-\l@dparsedendsub}%
 }{%
  \ifcsdef{#1@forinserting@all}{%
   \csgappto%
    {#1@forinserting@all}%
    {%
     \ifcsempty{Xsymlinenum@#1}%
      {\csuse{Xparafootsep@#1}}%
      {}%
     #3{#1}{#2}%
     \hskip\csuse{Xafternote@#1}\relax%
    }%
  }%
  {%
   \csdef%
    {#1@forinserting@all}%
    {%
     #3{#1}{#2}%
     \hskip\csuse{Xafternote@#1}\relax%
    }%
  }%
  \listcsgadd{#1@forinserting}{all}%
 }%
}%
%
```

XIII Familiar footnotes

XIII.1 Adjacent footnotes

The original edmac provided users with five series of critical footnotes (\Afootnote \Bfootnote \Cfootnote \Dfootnote \Efootnote), and LaTeX provides a single numbered footnote. The reledmac package uses the edmac mechanism to provide six series of numbered footnotes.

First, though, the footmisc package has an option whereby two or more consecutive \footnotes have their marks separated by commas. This seemed to Peter Wilson such a useful ability that it was provided automatically by eledmac.

Maïeul Rouquette has maintened this feature in reledmac, despite he thought that is not directly in relationship with the aim of reledmac.

\multiplefootnotemarker
\multfootsep

These macros may have been defined by the memoir class, are provided by the footmisc package and perhaps by other footnote packages. That is why we use \providecommand and not \newcommand.

```
\providecommand*{\multiplefootnotemarker}{3sp}
\providecommand*{\multfootsep}{\textsuperscript{\normalfont,}}

%
```

\m@mmf@prepare

A pair of self-cancelling kerns. This may have been defined in the memoir class.

```
\providecommand*{\m@mmf@prepare}{%
  \kern-\multiplefootnotemarker
  \kern\multiplefootnotemarker\relax}
%
```

\m@mmf@check

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

```
\providecommand*{\m@mmf@check}{%
  \ifdim\lastkern=\multiplefootnotemarker\relax
    \edef\@x@sf{\the\spacefactor}%
    \unkern
    \multfootsep
    \spacefactor\@x@sf\relax
  \fi}

%
```

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

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

\@footnotetext

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

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

\@footnotemark

Modify \@footnotemark to cater for adjacent \footnotes.

```

\patchcmd{\@footnotemark}
  {\nobreak}
  {\m@mmf@check
  \nobreak
  }
  {}{}
\patchcmd{\@footnotemark}
  {\@makefnmark}
  {\@makefnmark
  \m@mmf@prepare
  }
  {}{}
%
```

Finished the modifications for the non-memoir case.

```
}

%
```

XIII.2 Regular footnotes for numbered texts

\l@doldold@footnotetext
\@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.

```
\pretocmd{\@footnotetext}{%
  \ifnumberedpar@
    \edtext{}{\l@dbfnote{#1}}%
  \else
  }{}{}
\apptocmd{\@footnotetext}{\fi}{}{}%
%
```

\l@dbfnote
\vl@dbfnote
\vl@dbfnote
\footnote
\get@fnmark
\get@thisfootnote

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

```

\patchcmd%
  {\footnote}%
  {\stepcounter\@mpfn}%
  {%
  \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
    \global\advance\footnote@reading by \@ne%
    \get@thisfootnote%
    \get@fnmark{\thisc@footnote}%
    \ifcsdef{footnotereading\the\footnote@reading=typeset}%
      {\setcounter{\@mpfn}{\csuse{footnotereading\the\footnote@reading=
typeset}}}%
      {\setcounter{\@mpfn}{\footnote@reading}}%
    }{%
      \stepcounter\@mpfn%
    }%
  }%
  {}
  {}

\newcommand{\get@thisfootnote}{%
    \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
      \protected@xdef\thisc@footnote{\the\footnote@reading}%
    }{%
      \protected@xdef\thisc@footnote{\the\c@footnote}%
    }%
}%

\newcommand{\l@dbfnote}[1]{%
 \get@thisfootnote%
  \gdef\@tag{#1\relax}%
    \ifledRcol%
      \xright@appenditem{%
          \ifdefined\Hy@footnote@currentHref%
            \noexpand\def\noexpand\Hy@footnote@currentHref{\
Hy@footnote@currentHref}%
          \fi%
          \noexpand\vl@dbfnote{{\expandonce\@tag}}{\thisc@footnote}%
        }%
                         \to\inserts@listR
       \global\advance\insert@countR \@ne%
    \else%
      \xright@appenditem{%
          \ifdefined\Hy@footnote@currentHref%
            \noexpand\def\noexpand\Hy@footnote@currentHref{\
Hy@footnote@currentHref}%
          \fi%
          \noexpand\vl@dbfnote{{\expandonce\@tag}}{\thisc@footnote}%
       }%
                         \to\inserts@list
      \global\advance\insert@count \@ne%
    \fi
  \ignorespaces%
}%

\newcommand{\get@fnmark}[1]{%
  \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{l@dprintingcolumns}}%
    {%
      \stepcounter{footnote@typeset}%
      \setcounter{footnote}{\c@footnote@typeset}%
        \immediate\write\@mainaux{%
          \csgdef{footnotereading#1=typeset}{\the\c@footnote@typeset}%
         }%
      \def\@thefnmark{\thefootnote}%
    }%
    {%
      \setcounter{footnote}{#1}%
      \def\@thefnmark{\thefootnote}%
    }%
}%

\newcommand{\vl@dbfnote}[2]{%
  \get@fnmark{#2}%
  \@footnotetext{#1}%
}%
%
```

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
\postbodyfootmark

Two convenience macros for use by `\...@footnotemark...` macros.

```
\newcommand*{\prebodyfootmark}{%
  \leavevmode
  \ifhmode
    \edef\@x@sf{\the\spacefactor}%
    \m@mmf@check
    \nobreak
  \fi}
\newcommand{\postbodyfootmark}{%
  \m@mmf@prepare
  \ifhmode\spacefactor\@x@sf\fi\relax}

%
```

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.

```
\newcommandx{\arrangementX}[2][1,usedefault]{%
  \def\do##1{%
    \csname arrangementX@#2\endcsname{##1}%
  }%
  \ifstrempty{#1}%
    {%
    \dolistloop{\@series}%
    }%
    {
    \docsvlist{#1}%
    }%
}%
%
```

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.

```
\newcommand*{\normal@footnotemarkX}[1]{%
  \prebodyfootmark
  \wrapped@bodyfootmarkX{#1}%
  \postbodyfootmark}

%
```

\normalbodyfootmarkX

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

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

\normalvfootnoteX

\normalvfootnoteX{⟨*series*⟩}{⟨*text*⟩} does the \insert for the ⟨*series*⟩ and calls the series' \footfmt... to format the ⟨*text*⟩.

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnoteX}[2]{%
  \beforeinsertion@X{#1}%
  \insert\@nameuse{footins#1}\bgroup
    \reset@font%
    \hsize=\expandafter\dimexpr\csuse{widthX@#1}\relax%
    \noindent\csuse{bhooknoteX@#1}%
    \csuse{notefontsizeX@#1}%
    \footsplitskips
    \ifl@dpairing\ifl@dpaging\else%
      \setnoteswidthliketwocolumnsX@{#1}%
    \fi\fi%
    \setnotesXpositionliketwocolumns@{#1}%
    \spaceskip=\z@skip \xspaceskip=\z@skip
    \csuse{\csuse{footnote@dir}}\@nameuse{footfmt#1}{#1}{#2}\egroup}

%
```

\mpnormalvfootnoteX The minipage version.

```
\newcommand*{\mpnormalvfootnoteX}[2]{%
  \get@thisfootnoteX{#1}%
  \get@fnmarkX{#1}{\thisc@footnote}%
  \edef\this@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
  \global\setbox\@nameuse{mpfootins#1}\vbox{%
    \unvbox\@nameuse{mpfootins#1}
    \noindent\csuse{bhooknoteX@#1}%
    \csuse{notefontsizeX@#1}%
    \hsize\columnwidth
    \@parboxrestore
    \color@begingroup
    \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}}

%
```

\normalfootfmtX \normalfootfmtX{⟨*series*⟩}{⟨*text*⟩} typesets the footnote text, prepended by the marker.

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmtX}[2]{%
 \ifluatex%
    \textdir\footnote@luatextextdir%
    \pardir\footnote@luatexpardir%
    \par%
 \fi%
   \protected@edef\@currentlabel{%
       \@nameuse{@thefnmark#1}%
   }%
  \ledsetnormalparstuffX{#1}%
  \hangindent=\csuse{hangindentX@#1}%
  \everypar{\hangindent=\csuse{hangindentX@#1}}%
  \rule\z@\splittopskip%
  {{\csuse{notenumfontX@#1}\wrapped@footfootmarkX{#1}}%
    \csuse{wrapcontentX@#1}{#2}%
  \strut\par}}

%
```

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

```
\newcommand*{\normalfootfootmarkX}[1]{%
  \textsuperscript{\@nameuse{@thefnmark#1}}

%
```

\normalfootstartX

\normalfootstartX{⟨*series*⟩} is the ⟨*series*⟩ footnote starting macro used in the output routine.

```
\newcommand*{\normalfootstartX}[1]{%
    \ifdimequal{0pt}{\prenotesX@}{}%
      {%
      \iftoggle{prenotesX@}{%
           \togglefalse{prenotesX@}%
           \skip\csname footins#1\endcsname=%
             \glueexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
           }%
          {}%
      }%
  \vskip\skip\csname footins#1\endcsname%
  \leftskip=\z@
  \rightskip=\z@
  \ifl@dpairing\else%
     \hsize=\old@hsize%
  \fi%
  \setnoteswidthliketwocolumnsX@{#1}%
  \setnotesXpositionliketwocolumns@{#1}%
  \print@footnoteXrule{#1}%
}%

%
```

\normalfootnoteruleX

The rule drawn before the footnote series group.

```
\let\normalfootnoteruleX=\footnoterule

%
```

\normalfootgroupX

\normalfootgroupX{⟨*series*⟩} sends the contents of the ⟨*series*⟩ insert box to the output page without alteration.

```
\newcommand*{\normalfootgroupX}[1]{%
  \csuse{bhookgroupX@#1}%
  \unvbox\@nameuse{footins#1}%
  \hsize=\old@hsize%
  }%

%
```

\mpnormalfootgroupX

The minipage version.

```
\newcommand*{\mpnormalfootgroupX}[1]{%
  \vskip\skip\@nameuse{mpfootins#1}
  \ifl@dpairing\ifparledgroup%
    \leavevmode\marks\parledgroup@{begin}%
    \marks\parledgroup@series{#1}%
    \marks\parledgroup@type{footnoteX}%
  \fi\fi\normalcolor
  \ifparledgroup%
    \ifl@dpairing%
    \else%
      \setnoteswidthliketwocolumnsX@{#1}%
      \setnotesXpositionliketwocolumns@{#1}%
      \print@footnoteXrule{#1}%
    \fi%
  \else%
    \setnoteswidthliketwocolumnsX@{#1}%
    \setnotesXpositionliketwocolumns@{#1}%
    \print@footnoteXrule{#1}%
  \fi%
  \csuse{bhookgroupX@#1}%
  \unvbox\@nameuse{mpfootins#1}}

%
```

\normalbfnoteX

```

\newcommand{\normalbfnoteX}[2]{%
  \get@thisfootnoteX{#1}%
    \ifledRcol%
      \ifluatex
          \footnotelang@lua[R]%
      \fi
      \@ifundefined{xpg@main@language}%if polyglossia
        {}%
        {\footnotelang@poly[R]}%
      \xright@appenditem{%
        \noexpand\led@set@index@fornote{#1}%
        \noexpand\prepare@edindex@fornote{\l@d@nums}%
        \unexpanded{\def\this@footnoteX@reading}{\the\csname footnote#1
@reading\endcsname}%
        \noexpand\vbfnoteX{#1}{#2}{\thisc@footnote}%
        \noexpand\led@reinit@index@fornote%
        \unexpanded{\advance\@edindex@fornote@\m@ne}%
      }%
                         \to\inserts@listR
       \global\advance\insert@countR \@ne%
    \else%
      \ifluatex
          \footnotelang@lua%
      \fi
      \@ifundefined{xpg@main@language}%if polyglossia
        {}%
        {\footnotelang@poly}%
      \xright@appenditem{%
        \noexpand\led@set@index@fornote{#1}%
        \noexpand\prepare@edindex@fornote{\l@d@nums}%
        \unexpanded{\def\this@footnoteX@reading}{\the\csname footnote#1@reading\endcsname}%
        \noexpand\vbfnoteX{#1}{#2}{\thisc@footnote}%
        \noexpand\led@reinit@index@fornote%
        \unexpanded{\advance\@edindex@fornote@\m@ne}%
      }%
                         \to\inserts@list
      \global\advance\insert@count \@ne%
    \fi
  \ignorespaces}

%
```

\get@thisfootnoteX

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

```
\newcommand{\get@thisfootnoteX}[1]{%
    \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{l@dprintingcolumns}}{%
      \protected@xdef\thisc@footnote{\the\csname footnote#1@reading\endcsname}%
    }{%
      \protected@xdef\thisc@footnote{\the\csname c@footnote#1\endcsname}%
    }%
}%
%
```

\vbfnoteX

This command calls the correct footnote-inserting commands.

```
\newcommand{\vbfnoteX}[3]{%
  \get@fnmarkX{#1}{#3}%
  \@nameuse{regvfootnote#1}{#1}{#2}%
}%

%
```

\get@fnmarkX

This command gets the correct footnote number when typesetting parallel texts.

```
\newcommand{\get@fnmarkX}[2]{%
  \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{l@dprintingcolumns}}%
    {%
      \stepcounter{footnote#1@typeset}%
      \setcounter{footnote#1}{\value{footnote#1@typeset}}%
      \@namedef{@thefnmark#1}{\csuse{thefootnote#1}}%
        \immediate\write\@mainaux{%
          \csgdef{footnote#1reading#2=typeset}{\the\csname c@footnote#1
@typeset\endcsname}%
         }%
    }%
    {%
      \setcounter{footnote#1}{#2}%
      \@namedef{@thefnmark#1}{\csuse{thefootnote#1}}%
    }%
}
%
%
```

\vnumfootnoteX

```
\newcommand{\vnumfootnoteX}[2]{%
  \ifnumberedpar@
    \edtext{}{\normalbfnoteX{#1}{#2}}%
  \else
    \def\this@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
    \get@thisfootnoteX{#1}%
    \get@fnmarkX{#1}{\expandonce\thisc@footnote}%
    \@nameuse{regvfootnote#1}{#1}{#2}%
  \fi}

%
```

arrangementX@normal

\arrangementX@normal{⟨*series*⟩} initialises the settings for the ⟨*series*⟩ footnotes. This should always be called for each series.

```
\newcommand*{\arrangementX@normal}[1]{%
  \csgdef{series@displayX#1}{normal}
  \expandafter\let\csname footstart#1\endcsname=\normalfootstartX
  \@namedef{@footnotemark#1}{\normal@footnotemarkX{#1}}
  \@namedef{bodyfootmark#1}{\normalbodyfootmarkX{#1}}
  \expandafter\let\csname regvfootnote#1\endcsname=\normalvfootnoteX
  \expandafter\let\csname vfootnote#1\endcsname=\vnumfootnoteX
  \expandafter\let\csname footfmt#1\endcsname=\normalfootfmtX
  \@namedef{footfootmark#1}{\normalfootfootmarkX{#1}}
  \expandafter\let\csname footgroup#1\endcsname=\normalfootgroupX
  \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
  \count\csname footins#1\endcsname=1000
  \csxdef{default@footins#1}{1000}%Use to have note only for one side
  \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
  \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
  \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
%
```

Additions for minipages.

```
  \ifnoledgroup@\else%
    \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
    \expandafter\let\csname mpfootgroup#1\endcsname=\mpnormalfootgroupX
    \count\csname mpfootins#1\endcsname=1000
    \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
    \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
    \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
  \fi
}

%
```

XIII.4.3 Two columns footnotes

The following macros set footnotes in two columns. It is assumed that the length of each footnote is less than the column width.

\arrangementX@twocol

```
\newcommand*{\arrangementX@twocol}[1]{%
  \csgdef{series@displayX#1}{twocol}
  \expandafter\let\csname regvfootnote#1\endcsname=\twocolvfootnoteX
  \expandafter\let\csname footfmt#1\endcsname=\twocolfootfmtX
  \expandafter\let\csname footgroup#1\endcsname=\twocolfootgroupX
  \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
  \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
  \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
  \twocolfootsetupX{#1}
  \ifnoledgroup@\else%
    \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
    \expandafter\let\csname mpfootgroup#1\endcsname=\mptwocolfootgroupX
    \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
    \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}
    \mptwocolfootsetupX{#1}
  \fi%
}

%
```

\twocolfootsetupX
\mptwocolfootsetupX

\twocolfootsetupX{⟨*series*⟩}

```
\newcommand*{\twocolfootsetupX}[1]{%
  \count\csname footins#1\endcsname 500
  \csxdef{default@footins#1}{500}%Use this to confine the  notes to one side only
  \multiply\dimen\csname footins#1\endcsname by \tw@}
\newcommand*{\mptwocolfootsetupX}[1]{%
  \count\csname mpfootins#1\endcsname 500
  \multiply\dimen\csname mpfootins#1\endcsname by \tw@}

%
```

\twocolvfootnoteX

\twocolvfootnoteX{⟨*series*⟩}

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolvfootnoteX}[2]{%
  \beforeinsertion@X{#1}%
  \insert\csname footins#1\endcsname\bgroup%
    \hsize=\expandafter\dimexpr\csuse{widthX@#1}\relax%
    \noindent\csuse{bhooknoteX@#1}%
    \csuse{notefontsizeX@#1}%
    \footsplitskips%
    \spaceskip=\z@skip \xspaceskip=\z@skip%
    \@nameuse{footfmt#1}{#1}{#2}\egroup}

%
```

\twocolfootfmtX

\twocolfootfmtX{⟨*series*⟩}

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolfootfmtX}[2]{%
   \protected@edef\@currentlabel{%
       \@nameuse{@thefnmark#1}%
   }%
  \normal@pars%
  \hangindent=\csuse{hangindentX@#1}%
  \everypar{\hangindent=\csuse{hangindentX@#1}}%
  \hsize \csuse{hsizetwocolX@#1}%
 \nottoggle{parindentX@#1}{\parindent=\z@}{}%
  \tolerance=5000\relax%
  \par%
 \@tempdima=\parindent%
 \csuse{colalignX@#1}%
 \parindent=\@tempdima%
  {\hspace{\parindent}%
  \csuse{notenumfontX@#1}\wrapped@footfootmarkX{#1}\strut%
    \csuse{wrapcontentX@#1}{#2}%
    \strut\par}%
 \allowbreak%
}%

%
```

\twocolfootgroupX
\mptwocolfootgroupX

\twocolfootgroupX{⟨*series*⟩}

```
\newcommand*{\twocolfootgroupX}[1]{{\csuse{bhookgroupX@#1}\csuse{notefontsizeX@#1}
  \splittopskip=\ht\strutbox
  \expandafter
  \rigidbalanceX\csname footins#1\endcsname \tw@ \splittopskip}}

\newcommand*{\mptwocolfootgroupX}[1]{{%
  \vskip\skip\@nameuse{mpfootins#1}
  \ifl@dpairing\ifparledgroup%
```

```
    \leavevmode\marks\parledgroup@{begin}%
    \marks\parledgroup@series{#1}%
    \marks\parledgroup@type{footnoteX}%
  \fi\fi\normalcolor
  \ifparledgroup%
    \ifl@dpairing%
    \else%
      \setnoteswidthliketwocolumnsX@{#1}%
      \setnotesXpositionliketwocolumns@{#1}%
      \print@footnoteXrule{#1}%
    \fi%
  \else%
    \setnoteswidthliketwocolumnsX@{#1}%
    \setnotesXpositionliketwocolumns@{#1}%
    \print@footnoteXrule{#1}%
  \fi%
  \csuse{bhookgroupX@#1}%
  \splittopskip=\ht\strutbox
  \expandafter
  \rigidbalanceX\csname mpfootins#1\endcsname \tw@ \splittopskip}}

%
```

XIII.4.4 Three columns footnotes

The following macros set footnotes in three columns. It is assumed that the length of each footnote is less than the column width.

\arrangementX@threecol

```
\newcommand*{\arrangementX@threecol}[1]{%
  \csgdef{series@displayX#1}{threecol}
  \expandafter\let\csname regvfootnote#1\endcsname=\threecolvfootnoteX
  \expandafter\let\csname footfmt#1\endcsname=\threecolfootfmtX
  \expandafter\let\csname footgroup#1\endcsname=\threecolfootgroupX
  \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
  \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
  \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
  \threecolfootsetupX{#1}
  \ifnoledgroup@\else%
    \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
    \expandafter\let\csname mpfootgroup#1\endcsname=\mpthreecolfootgroupX
    \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
    \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}
    \mpthreecolfootsetupX{#1}
  \fi%
}

%
```

\threecolfootsetupX
\mpthreecolfootsetupX

\threecolfootsetupX{⟨*series*⟩}

```
\newcommand*{\threecolfootsetupX}[1]{%
  \count\csname footins#1\endcsname 333
  \csxdef{default@footins#1}{333}%Use this to confine the  notes to one side only
  \multiply\dimen\csname footins#1\endcsname by \thr@@}
\newcommand*{\mpthreecolfootsetupX}[1]{%
  \count\csname mpfootins#1\endcsname 333
  \multiply\dimen\csname mpfootins#1\endcsname by \thr@@}

%
```

\threecolvfootnoteX

\threecolvfootnoteX{⟨*series*⟩}{⟨*text*⟩}

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolvfootnoteX}[2]{%
  \beforeinsertion@X{#1}%
  \insert\csname footins#1\endcsname\bgroup%
    \hsize=\expandafter\dimexpr\csuse{widthX@#1}\relax%
    \noindent\csuse{bhooknoteX@#1}%
    \csuse{notefontsizeX@#1}%
    \footsplitskips%
    \@nameuse{footfmt#1}{#1}{#2}\egroup}

%
```

\threecolfootfmtX

\threecolfootfmtX{⟨*series*⟩}

```
\notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmtX}[2]{%
   \protected@edef\@currentlabel{%
       \@nameuse{@thefnmark#1}%
   }%
  \hangindent=\csuse{hangindentX@#1}%
  \everypar{\hangindent=\csuse{hangindentX@#1}}%
  \normal@pars%
  \hsize \csuse{hsizethreecolX@#1}%
  \nottoggle{parindentX@#1}{\parindent=\z@}{}%
  \tolerance=5000\relax%
 \@tempdima=\parindent%
  \csuse{colalignX@#1}%
  \parindent=\@tempdima%
  {\hspace{\parindent}%
  \csuse{notenumfontX@#1}\wrapped@footfootmarkX{#1}\strut%
    \csuse{wrapcontentX@#1}{#2}%
    \strut\par}\allowbreak}

%
```

\threecolfootgroupX
\mpthreecolfootgroupX

\threecolfootgroupX{⟨*series*⟩}

```
\newcommand*{\threecolfootgroupX}[1]{{\csuse{bhookgroupX@#1}\csuse{
notefontsizeX@#1}
  \splittopskip=\ht\strutbox
  \expandafter
  \rigidbalanceX\csname footins#1\endcsname \thr@@ \splittopskip}}

\newcommand*{\mpthreecolfootgroupX}[1]{{%
  \vskip\skip\@nameuse{mpfootins#1}
  \ifl@dpairing\ifparledgroup
    \leavevmode\marks\parledgroup@{begin}%
    \marks\parledgroup@series{#1}%
    \marks\parledgroup@type{footnoteX}%
  \fi\fi\normalcolor
  \ifparledgroup%
    \ifl@dpairing%
    \else%
      \setnoteswidthliketwocolumnsX@{#1}%
      \setnotesXpositionliketwocolumns@{#1}%
      \print@footnoteXrule{#1}%
    \fi%
  \else%
    \setnoteswidthliketwocolumnsX@{#1}%
    \setnotesXpositionliketwocolumns@{#1}%
    \print@footnoteXrule{#1}%
  \fi%
  \csuse{bhookgroupX@#1}%
  \splittopskip=\ht\strutbox
  \expandafter
  \rigidbalanceX\csname mpfootins#1\endcsname \thr@@ \splittopskip}}

%
```

XIII.4.5 Paragraphed footnotes

The following macros set footnotes as one paragraph.

\arrangementX@threecol

\footparagraphX{⟨*series*⟩}

```
\newcommand*{\arrangementX@paragraph}[1]{%
  \csgdef{series@displayX#1}{paragraph}%
  \expandafter\let\csname footstart#1\endcsname=\parafootstartX
  \expandafter\let\csname regvfootnote#1\endcsname=\para@vfootnoteX
  \expandafter\let\csname footfmt#1\endcsname=\parafootfmtX
  \expandafter\let\csname footgroup#1\endcsname=\para@footgroupX
  \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
  \count\csname footins#1\endcsname=1000
  \csxdef{default@footins#1}{1000}%Use this to confine the  notes to one
side only
  \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
  \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
  \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
  \para@footsetupX{#1}
  \ifnoledgroup@\else
    \expandafter\let\csname mpvfootnote#1\endcsname=\mppara@vfootnoteX
    \expandafter\let\csname mpfootgroup#1\endcsname=\mppara@footgroupX
    \count\csname mpfootins#1\endcsname=1000
    \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
    \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
    \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
  \fi
  }

%
```

\para@footsetupX \para@footsetupX{⟨*series*⟩}

```
\newcommand*{\para@footsetupX}[1]{{\csuse{bhookgroupX@#1}\csuse{
notefontsizeX@#1}
  \setnoteswidthliketwocolumnsX@{#1}%
  \ifcsempty{widthX@#1}%
    {}%
    {\columnwidth=\expandafter\dimexpr\csuse{widthX@#1}\relax}%
  \dimen0=\baselineskip
  \multiply\dimen0 by 1024
  \divide\dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\relax
%
  \expandafter
  \xdef\csname footfudgefactor#1\endcsname{%
    \expandafter\strip@pt\dimen0 }}}

%
```

\parafootstartX \parafootstartX{⟨*series*⟩}

```
\newcommand*{\parafootstartX}[1]{%
    \ifdimequal{0pt}{\prenotesX@}{}%
      {%
      \iftoggle{prenotesX@}{%
            \togglefalse{prenotesX@}%
            \skip\csname footins#1\endcsname=%
              \glueexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
            }%
          {}%
      }%
  \leftskip=\z@
  \rightskip=\z@
  \nottoggle{parindentX@#1}{\parindent=\z@}{}%
  \vskip\skip\@nameuse{footins#1}%
  \setnoteswidthliketwocolumnsX@{#1}%
  \setnotesXpositionliketwocolumns@{#1}%
  \print@footnoteXrule{#1}%
  }

%
```

\para@vfootnoteX
\mppara@vfootnoteX

\para@vfootnoteX{⟨*series*⟩}{⟨*text*⟩}

```
\newcommand*{\para@vfootnoteX}[2]{%
 \csuse{beforeinsertingX@#1}%
 \insert\csname footins#1\endcsname%
  \bgroup
    \csuse{notefontsizeX@#1}
    \footsplitskips
    \setbox0=\vbox{\hsize=\maxdimen%
      \let\bidi@RTL@everypar\@empty%
      \insert@txtbeforenotesX{#1}%
      \noindent\csuse{bhooknoteX@#1}%
      \@nameuse{footfmt#1}{#1}{#2}}%
    \setbox0=\hbox{\unvxhX{0}{#1}}%
    \dp0=\z@
    \ht0=\csname footfudgefactor#1\endcsname\wd0
    \box0
    \penalty0
  \egroup}
\newcommand*{\mppara@vfootnoteX}[2]{%
  \get@thisfootnoteX{#1}%
  \get@fnmarkX{#1}{\thisc@footnote}%
  \edef\this@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
  \global\setbox\@nameuse{mpfootins#1}\vbox{%
    \unvbox\@nameuse{mpfootins#1}
    \csuse{notefontsizeX@#1}
    \footsplitskips
    \setbox0=\vbox{\hsize=\maxdimen%
      \let\bidi@RTL@everypar\@empty%
      \noindent\color@begingroup%
      \csuse{bhooknoteX@#1}%
      \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}%
    \setbox0=\hbox{\unvxhX{0}{#1}}%
    \dp0=\z@
    \ht0=\csname footfudgefactor#1\endcsname\wd0
    \box0
    \penalty0}}

%
```

\unvxhX

```
\newcommand*{\unvxhX}[2]{% 2th is optional for retro-compatibility
  \setbox0=\vbox{\unvbox#1%
    \global\setbox1=\lastbox}%
  \unhbox1
  \unskip            % remove \rightskip,
  \unskip            % remove \parfillskip,
  \unpenalty         % remove \penalty of 10000,
  \hskip\csuse{afternoteX@#2}%
  \relax}% but add the glue to go between the notes

%
```

\parafootfmtX

\parafootfmtX{⟨*series*⟩}

```
\newcommand*{\parafootfmtX}[2]{%
   \protected@edef\@currentlabel{%
       \@nameuse{@thefnmark#1}%
   }%
  \insertparafootsepX{#1}%
  \ledsetnormalparstuff@common%
  {\csuse{notenumfontX@#1}%
   \csuse{notenumfontX@#1}%
   \wrapped@footfootmarkX{#1}%
   \strut%
    \csuse{wrapcontentX@#1}{#2}%
    \penalty-10}}

%
```

\para@footgroupX
\mppara@footgroupX

\para@footgroupX{⟨*series*⟩}

```
\newcommand*{\para@footgroupX}[1]{%
  \hsize=\expandafter\dimexpr\csuse{widthX@#1}\relax%
  \unvbox\csname footins#1\endcsname
  \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
  \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
  \makehboxofhboxes
  \setbox0=\hbox{\unhbox0 \removehboxes}%
  \csuse{bhookgroupX@#1}
  \csuse{notefontsizeX@#1}
  \unhbox0\par}

\newcommand*{\mppara@footgroupX}[1]{{%
  \setnoteswidthliketwocolumnsX@{#1}%
  \vskip\skip\@nameuse{mpfootins#1}
  \ifl@dpairing\ifparledgroup
    \leavevmode%
    \leavevmode\marks\parledgroup@{begin}%
    \marks\parledgroup@series{#1}%
    \marks\parledgroup@type{footnoteX}%
  \fi\fi\normalcolor
  \ifparledgroup%
    \ifl@dpairing%
    \else%
```

```
      \setnoteswidthliketwocolumnsX@{#1}%
      \setnotesXpositionliketwocolumns@{#1}%
      \print@footnoteXrule{#1}%
    \fi%
  \else%
      \setnoteswidthliketwocolumnsX@{#1}%
      \setnotesXpositionliketwocolumns@{#1}%
      \print@footnoteXrule{#1}%
  \fi%
  \unvbox\csname mpfootins#1\endcsname
  \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
  \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
  \makehboxofhboxes
  \setbox0=\hbox{\unhbox0 \removehboxes}%
  \csuse{bhookgroupX@#1}%
  \csuse{notefontsizeX@#1}%
  \nottoggle{parindentX@#1}{\parindent=\z@}{}%
  \unhbox0\par}}

%
```

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

\insertparafootsepX

```
\newcommand{\insertparafootsepX}[1]{%
   \ifledRcol@%
     \ifnumequal{\csuse{prevpage#1@numR}}{\page@numR}%
        {\csuse{Xparafootsep@#1}}%
        {}%
     \global\csname prevpage#1@numR\endcsname=\page@numR%
   \else%
     \ifnumequal{\csuse{prevpage#1@num}}{\page@num}%
        {\csuse{Xparafootsep@#1}}%
        {}%
     \global\csname prevpage#1@num\endcsname=\page@num%
   \fi%
}
%
```

XIII.5 Wrapping footnote marks in hyperlink

\wrapped@footfootmarkX

\wrapped@footfootmarkX prints the footnote mark of the footpage, wrapped in hyperref package's commands, if needed.

```
\newcommand{\wrapped@footfootmarkX}[1]{%
  \ifdefined\hypertarget%
    \hyperlink%
        {@bodyfootmark#1@\this@footnoteX@reading}%
        {\@nameuse{footfootmark#1}}%
    \Hy@raisedlink{%
      \hypertarget%
        {@footnotemark#1@\this@footnoteX@reading}%
        {}%
      }%
  \else%
    \@nameuse{footfootmark#1}%
  \fi%
}%
%
```

\wrapped@bodyfootmarkX

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

```
\newcommand{\wrapped@bodyfootmarkX}[1]{%
  \ifdefined\hypertarget%
    \hyperlink%
        {@footnotemark#1@\expandafter\the\csname footnote#1@reading\
endcsname}%
        {\@nameuse{bodyfootmark#1}}%
    \Hy@raisedlink{%
      \hypertarget%
        {@bodyfootmark#1@\expandafter\the\csname footnote#1@reading\
endcsname}%
        {}%
      }%
  \else%
    \@nameuse{bodyfootmark#1}%
  \fi%
}%
%
```

XIV Code common to both critical and familiar footnote in normal arrangement

\par should always be redefined to \endgraf within the format macro (this is what \normal@pars does), to override tricky material in the main text to get the lines numbered automatically (as set up by \autopar, for example).

In the case of footnote arranged in a "normal" way, we also must set some setting for paragraph indent and text direction when using LuaLaTeX.

That why we have defined \ledsetnormalparstuff@common in order to make this setting for both familiar and critical notes. This command is called by command to make specific setting to critical or familiar footnote.

\ledsetnormalparstuff@common
\Xledsetnormalparstuff
\ledsetnormalparstuffX

```
\newcommand*{\ledsetnormalparstuff@common}{%
  \ifluatex%
```

```
   \ifdefstring{\footnote@luatextextdir}{TLT}{}%
     {\textdir\footnote@luatextextdir}%
   \pardir\footnote@luatexpardir%
  \fi%
  \csuse{\csuse{footnote@dir}}%
  \normal@pars%
  \parfillskip \z@ \@plus 1fil}%

\newcommand*{\Xledsetnormalparstuff}[1]{%
  \ledsetnormalparstuff@common%
  \nottoggle{Xparindent@#1}{\parindent=\z@}{\hspace{\parindent}}%
}%

\newcommand*{\ledsetnormalparstuffX}[1]{%
  \ledsetnormalparstuff@common%
  \nottoggle{parindentX@#1}{\parindent=\z@}{\hspace{\parindent}}%
}%
%
```

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
noteswidthliketwocolumns@
oteswidthliketwocolumnsX@

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.

```

\newdimen\old@hsize%
\AtBeginDocument{\old@hsize=\hsize}%

\newcommand{\setXnoteswidthliketwocolumns@}[1]{%
  \global\let\hsize@fornote=\hsize%
  \global\old@hsize=\hsize%
  \let\old@columwidth=\columnwidth%
  \iftoggle{Xnoteswidthliketwocolumns@#1}%
    {%
    \csuse{setwidthliketwocolumns@\columns@position}%
    \global\let\hsize@fornote=\hsize%
    }%
    {}%
  \let\hsize=\hsize@fornote%
  \let\columnwidth=\old@columwidth%
}%

\newcommand{\setnoteswidthliketwocolumnsX@}[1]{%
  \global\let\hsize@fornote=\hsize%
  \global\old@hsize=\hsize%
  \let\old@columwidth=\columnwidth%
  \iftoggle{noteswidthliketwocolumnsX@#1}%
    {%
    \csuse{setwidthliketwocolumns@\columns@position}%
    \global\let\hsize@fornote=\hsize%
    }%
    {}%
  \let\hsize=\hsize@fornote%
  \let\columnwidth=\old@columwidth%
}%

%
```

\setXnotespositionliketwocolumns@
\setnotesXpositionliketwocolumns@

These two commands set the position of the critical / familiar footnotes, depending on the hooks `Xnoteswidthliketwocolumns` and `noteswidthliketwocolumnsX`. They call commands which are defined only in `reledpar`, because this feature has no sens without `reledpar`.

```
\newcommand{\setXnotespositionliketwocolumns@}[1]{%
  \iftoggle{Xnoteswidthliketwocolumns@#1}{%
    \csuse{setnotespositionliketwocolumns@\columns@position}%
  }{}%
}%

\newcommand{\setnotesXpositionliketwocolumns@}[1]{%
  \iftoggle{noteswidthliketwocolumnsX@#1}{%
    \csuse{setnotespositionliketwocolumns@\columns@position}%
  }{}%
}%

%
```

XVI Footnotes' order

\fnpos
\mpfnpos
\@fnpos
\@mpfnpos

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

```
\def\@fnpos{familiar-critical}
\def\@mpfnpos{critical-familiar}
\newcommand{\fnpos}[1]{\xdef\@fnpos{#1}}
\newcommand{\mpfnpos}[1]{\xdef\@mpfnpos{#1}}
%
```

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
\print@footnoteXrule

```
\newcommand{\print@Xfootnoterule}[1]{%
  \vskip-\csuse{Xafterrule@#1}%Because count in \dimen\csuse{#1footins}
  \nointerlineskip%
  \moveleft-\leftskip\vbox{\csuse{#1footnoterule}}%
  \nointerlineskip%
  \vskip\csuse{Xafterrule@#1}%
}%

\newcommand{\print@footnoteXrule}[1]{%
  \vskip-\csuse{afterruleX@#1}%Because count in \dimen\csuse{footins#1}
  \nointerlineskip%
  \moveleft-\leftskip\vbox{\csuse{footnoterule#1}}%
  \nointerlineskip%
  \vskip\csuse{afterruleX@#1}%
}%

%
```

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@Xprenotes 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

\Xprenotes@
\Xprenotes

If user redefines \Xprenotes@, via \Xprenotes to a value greater than 0 pt, this skip will be added before first series notes instead of the notes skip.

```
\newtoggle{Xprenotes@}%
\toggletrue{Xprenotes@}%
\newcommand{\Xprenotes@}{0pt}%
\newcommand*{\Xprenotes}[1]{\renewcommand{\Xprenotes@}{#1}}%
\newcommand{\preXnotes}[1]{\led@warning@preXnotes@deprecated\Xprenotes{#1}}
%For compatibility
%
```

The same, but for familiar footnotes.

\Xprenotes
\Xprenotes@

```
\newtoggle{prenotesX@}
\toggletrue{prenotesX@}
\newcommand{\prenotesX@}{0pt}
\newcommand*{\prenotesX}[1]{\renewcommand{\prenotesX@}{#1}}
%
```

XVIII.0.3 Internal commands

firstXseries@
prepare@Xprenotes

```
\gdef\firstXseries@{}
\newcommand{\prepare@Xprenotes}[1]{%
  \ifdimequal{0pt}{\Xprenotes@}%
  {}%
  {%
    \IfStrEq{\firstXseries@}{}{%
      \global\skip\csuse{#1footins}=\Xprenotes@%
      \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
      \gdef\firstXseries@{#1}%
    }%
    {%
     \ifseriesbefore{#1}{\firstXseries@}%
       {%
       \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@\firstXseries@}%
       \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
       \gdef\firstXseries@{#1}%
       }%
       {}%
    }%
  }%
}
%
```

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

firstseriesX@

prepare@prenotesX

```
\gdef\firstseriesX@{}
\newcommand{\prepare@prenotesX}[1]{%
  \ifdimequal{0pt}{\prenotesX@}%
  {}%
  {%
    \IfStrEq{\firstseriesX@}{}{%
      \global\skip\csuse{footins#1}=\prenotesX@%
      \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
      \gdef\firstseriesX@{#1}%
    }%
    {%
     \ifseriesbefore{#1}{\firstseriesX@}%
       {%
       \global\skip\csuse{footins#1}=\csuse{beforenotesX@\firstseriesX@}%
       \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
       \gdef\firstXseries@{#1}%
       }%
       {}%
    }%
  }%
}
%
```

XIX Endnotes

First, check the noend option.

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

XIX.1 Internal commands

\l@dend@open
\l@dend@close

\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.

```
\newcommand{\l@dend@open}[1]{%
  \global\booltrue{l@dend@#1}%
  \expandafter\immediate%
    \expandafter\openout%
      \csname l@d@#1end\endcsname%
      =\l@auxdir\jobname.#1end\relax%
}%
\newcommand{\l@dend@close}[1]{%
  \global\boolfalse{l@dend@#1}%
  \expandafter\immediate%
    \expandafter\closeout\csname l@d@#1end\endcsname%
}%

%
```

`\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.

```
\newcommand{\l@dend@stuff}{%
 \def\do##1{%
   \ifbool{l@dend@##1}{}%
     {\l@dend@open{##1}}%
   \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname{\string\l@d@section{\the\section@num}\@percentchar}%
   }%
 \dolistloop{\@series}%
}%

%
```

`\endprint` `\l@d@section` The `\endprint` here is nearly identical in its functioning to `\normalfootfmt`.

The endnote file also contains `\l@d@section` commands, which supply the section numbers from the main text; standard `reledmac` does nothing with this information, but it is there if you want to write custom macros to do something with it. Arguments are:

- #1 Line numbers and font selection.
- #2 Lemma.
- #3 Note content.
- #4 Series.
- #5 Optional argument of `\Xendnote`.
- #6 Side (L or R).
- #7 Label for cross-referencing.

```
\global\notbool{parapparatus@}{}{\long}\def\endprint#1#2#3#4#5#6#7{{%
  \csuse{Xendbhooknote@#4}%
  \csuse{Xendnotefontsize@#4}%
  \hangindent=\csuse{Xendhangindent@#4}%
  \ifXendinsertsep@%
    \hskip\csuse{Xendafternote@#4}\relax%
    \csuse{Xendsep@#4}%
  \else%
    \iftoggle{Xendparagraph@#4}%
      {\global\Xendinsertsep@true}%
      {}%
  \fi%
  \xdef\@currentseries{#4}%
  \def\do##1{%
    \setkeys[mac]{truefootnoteoption}{##1}%
  }%
  \notblank{#5}{\docsvlist{#5}}{}%
  \IfStrEq{#6}{R}{\ledRcol@true}{}%
  \def\@this@crossref@start{#7:start}%
  \def\@this@crossref@end{#7:end}%
  \printlineendnote{#1}{#4}%
  \IfStrEq{#6}{R}{\ledRcol@false}{}%
  \undef\@this@crossref@start%
  \undef\@this@crossref@end%
  \nottoggle{Xendlemmadisablefontselection@#4}%
    {\select@lemmafont#1|}%
    {}%
  \bgroup%
    \csuse{Xendlemmafont@#4}%
    \csuse{Xendwraplemma@#4}{#2}%
  \egroup%
  \ifboolexpr{%
    togl {nosep@}%
    or test{\ifcsempty{Xendlemmaseparator@#4}}%
    }%
    {\hskip\csuse{Xendinplaceoflemmaseparator@#4}\relax}%
    {\nobreak%
     \hskip\csuse{Xendbeforelemmaseparator@#4}%
     \csuse{Xendlemmaseparator@#4}%
     \hskip\csuse{Xendafterlemmaseparator@#4}%
     \relax%
    }%
  \csuse{Xendwrapcontent@#4}{#3}%
  \nottoggle{Xendparagraph@#4}{\par}{}%
  \def\do##1{%
    \setkeys[mac]{falsefootnoteoption}{##1}%
  }%
  \notblank{#5}{\docsvlist{#5}}{}%
}}%

```

```
\let\l@d@section=\@gobble

%
```

\printlineendnote This macro controls, in endnote, whether the line number is printed or not, according to the series options. Its first argument is the information about lines; its second is the series of the footnote.

```
\newcommand{\printlineendnote}[2]{%
  \l@dp@rsefootspec#1|%
  \iftoggle{Xendnumberonlyfirstintwolines@#2}{%
       \xdef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
l@dparsedstartsub - \l@dparsedendpage - \l@dparsedendline - \
l@dparsedendsub}%
       }%
       {%
       \xdef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
l@dparsedstartsub}%
       }%
  \ifboolexpr{%
    togl {nonum@}%
    or togl {Xendnonumber@#2}%
    }%
    {\hspace{\csuse{Xendinplaceofnumber@#2}}}%
    {%
      \iftoggle{Xendnumberonlyfirstinline@#2}%
        {\ifcsdef{prevendline#2}%
          {\ifcsequal{prevendline#2}{lineinfo@}%
            {%
            \csuse{Xendbhookinplaceofnumber@#2}%
            \ifcsempty{Xendsymlinenum@#2}%
              {\hspace{\csuse{Xendinplaceofnumber@#2}}}%
              {\printsymlineendnotearea{#2}}%
            \csuse{Xendahookinplaceofnumber@2}%
            }%
            {\printlineendnotearea{#1}{#2}}}%
          {\printlineendnotearea{#1}{#2}}%
        }%
        {\printlineendnotearea{#1}{#2}}%We keep every time line
    \csxdef{prevendline#2}{\lineinfo@}%
    }%
}%
%
```

\printsymlineendnotearea

```
\newcommand{\printsymlineendnotearea}[1]{%
  \hspace{\csuse{Xendbeforesymlinenum@#1}}%
  \csuse{Xendnotenumfont@#1}%
  \ifdimequal{\csuse{Xendboxsymlinenum@#1}}{\z@}%
     {\csuse{Xendsymlinenum@#1}}%
```

```
    {\hbox to \csuse{Xendboxsymlinenum@#1}%
      {\csuse{Xendsymlinenum@#1}\hfill}%
    }%
  \hspace{\csuse{Xendaftersymlinenum@#1}}%
}%
%
```

\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.)

```
\newcommand{\printlineendnotearea}[2]{%
  \csuse{Xendbhooklinenumber@#2}%
  \hspace{\csuse{Xendbeforenumber@#2}}%
  \bgroup%
    \csuse{Xendnotenumfont@#2}%
    \ifdimequal{\csuse{Xendboxlinenum@#2}}{0pt}%
      {\printendlines#1||\ifledRcol@\@Rlineflag\fi}%
      {\leavevmode%
       \hbox to \csuse{Xendboxlinenum@#2}%
       {%
       \IfSubStr{RC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}{}%
       \printendlines#1||\ifledRcol@\@Rlineflag\fi%
       \IfSubStr{LC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}{}%
      }}%
  \egroup%
  \hspace{\csuse{Xendafternumber@#2}}%
  \csuse{Xendahooklinenumber@#2}%
}%
%
```

XIX.2 User level commands

XIX.2.1 Inserting contents to endnotes

The \Xendnotes commands are defined above, when defining apparatus commands by series. Here, we define only \toendnotes command not specific to a series, in order to insert arbitrary code. The regular version writes an unexpanded argument, while the regular version writes a once-expanded argument.

\toendnotes
\toendnotes*

```
\newcommandx{\toendnotes}[2][1,usedefault]{%
  \ifboolexpr{bool{numbering} or bool{numberingR}}{%
    \def\do##1{%
      \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname%
        {\unexpanded{#2}\@percentchar}%
    }%
    \ifstrempty{#1}%
      {\dolistloop{\@series}}%
      {\docsvlist{#1}}%
  }{\led@err@toendnotes@outsidenumbering}%
}%
\WithSuffix\newcommandx\toendnotes*[2][1,usedefault]{%
  \ifboolexpr{bool{numbering} or bool{numberingR}}{%
    \def\do##1{%
      \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname%
        {#2\@percentchar}%
    }%
    \ifstrempty{#1}%
      {\dolistloop{\@series}}%
      {\docsvlist{#1}}%
  }{\led@err@toendnotes@outsidenumbering}%
}%
%
```

XIX.2.2 Printing endnotes

\doendnotes
\ifXendinsertsep@

\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.

```
\newif\ifXendinsertsep@%
\newcommand*{\doendnotes}[1]{%
  \l@dend@close{#1}%
   \begingroup
      \csxdef{prevpagenum@#1}{}%
      \csxdef{prevpagerange@#1}{}%
      \makeatletter
      \expandafter\let\csname #1end\endcsname=\endprint
      \input\l@auxdir\jobname.#1end%
      \global\Xendinsertsep@false%
   \endgroup}
%
```

\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.

```
\newcommand*{\doendnotesbysection}[1]{%
 \l@dend@close{#1}%
 \csxdef{prevpagenum@#1}{}%
 \csxdef{prevpagerange@#1}{}%
 \global\expandafter\advance\csname #1end@bysection\endcsname by 1%
 \begingroup%
    \makeatletter%
    \def\l@d@section##1{%
```

```
   \ifnumequal{##1}{\csname #1end@bysection\endcsname}%
    {\cslet{#1end}{\endprint}}%
    {\cslet{#1end}{\@gobbleseven}}%
  }%
  \input\l@auxdir\jobname.#1end%
  \global\Xendinsertsep@false%
 \endgroup%
}%
%
```

We close now the conditional period, which depends on `\ifnoend@`, because the following commands can be used by other commands than those speicific to endnotes.

```
}%
%
```

`\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.

```
\newcommand*{\setprintendlines}[6]{%
 \l@d@pnumfalse \l@d@dashfalse
 \ifnum#4=#1 \else
  \l@d@pnumtrue
  \l@d@dashtrue
 \fi
%
```

We print the ending line number if: (1) we are printing the ending page number, or (2) it is different from the starting line number.

```
 \ifl@d@pnum \l@d@elintrue \else \l@d@elinfalse \fi
 \ifnum#2=#5 \else
   \l@d@elintrue
   \l@d@dashtrue
 \fi
%
```

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

```
 \l@d@ssubfalse
 \ifnum#3=0 \else
   \l@d@ssubtrue
 \fi
%
```

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.

```
  \l@d@eslfalse
  \ifnum#6=0 \else
      \ifnum#6=#3
         \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
      \else
         \l@d@esltrue
         \l@d@dashtrue
      \fi
  \fi%
%
```

```
  \ifl@d@dash%
    \ifboolexpr{togl{fulllines@} or test{\ifcsempty{Xendtwolines@\
@currentseries}}}%
      {}%
      {%
      \setistwofollowinglines{#1}{#2}{#4}{#5}%
        \ifboolexpr{%
          (%
               togl {Xendtwolinesbutnotmore@\@currentseries}%
               and not%
                 (%
                  bool {istwofollowinglines@}%
                 )%
          )%
          or%
          (%
             (not test{\ifnumequal{#1}{#4}})%
               and togl{Xendtwolinesonlyinsamepage@\@currentseries}%
             )%
          }%
          {}%
          {%
          \l@d@dashfalse%
          \l@d@Xtwolinestrue%
          \l@d@elinfalse%
          \l@d@eslfalse%
          \ifcsempty{Xendmorethantwolines@\@currentseries}%
            {}%
            {\ifistwofollowinglines@\else%
              \l@d@Xmorethantwolinestrue%
             \fi%
            }%
          }%
      }%
  \fi%
%
```

End of \setprintendlines.

```
}%
%
```

\printendlines Now we are ready to print it all.

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

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, start the starting line box, if needed.

```
  \ifdimequal{\csuse{Xendboxstartlinenum@\@currentseries}}{0pt}%
    {\bgroup}%
    {\leavevmode\hbox to \csuse{Xendboxstartlinenum@\@currentseries}\bgroup\hfill}%
%
```

Then, print the starting page number-

```
  \ifboolexpr{%
    (%
      test{\ifcsstring{prevpagenum@\@currentseries}{#1}}%
        and not%
      (togl{Xendpagenumberonlyfirstifsingle@\@currentseries} and bool{l@d@pnum})%
    )%
    or%
    (%
    test {\ifcsstring{prevpagerange@\@currentseries}{#1-#4}}%
    )%
  }%
    {%
    \ifcsempty{Xendsympagenum@\@currentseries}%
      {\hspace{\csuse{Xendinplaceofpagenumber@\@currentseries}}}%
      {\csuse{Xendsympagenum@\@currentseries}}%
    }%
    {%
    \wrap@edcrossref{\@this@crossref@start}{\printnpnum{#1}}%
    }%
%
```

Then, determine what must be printed before the start line.

```
  \ifl@d@dash%
    \ifl@d@pnum%
```

```
      \csuse{Xendlineprefixsingle@\@currentseries}%
    \else%
      \ifcsempty{Xendlineprefixmore@\@currentseries}%
        {\csuse{Xendlineprefixsingle@\@currentseries}}%
        {\csuse{Xendlineprefixmore@\@currentseries}}%
    \fi%
  \else%
    \csuse{Xendlineprefixsingle@\@currentseries}%
  \fi%
%
```

The print the starting line, followed, if needed, by the side flag and the starting sub line number.

```
  \wrap@edcrossref{\@this@crossref@start}{%
    \ifledRcol@%
      \linenumrepR{#2}%
    \else%
      \linenumrep{#2}%
    \fi%
  }%
  \iftoggle{Xendlineflag@\@currentseries}{\ifledRcol@\@Rlineflag\fi}{}%
  \ifl@d@ssub%
    \csuse{Xendsublinesep@\@currentseries}%
    \wrap@edcrossref{\@this@crossref@start}{%
      \ifledRcol@%
        \sublinenumrepR{#3}%
      \else%
        \sublinenumrep{#3}%
      \fi%
    }%
  \fi%
%
```

Close the box.

```
  \egroup%
%
```

Open the box for the ending line number.

```
  \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
    {\bgroup}%
    {\hbox to \csuse{Xendboxendlinenum@\@currentseries}\bgroup}%
%
```

Print the dash + the ending line number, or the line number range symbol.

```
  \ifl@d@Xtwolines%
    \ifl@d@Xmorethantwolines%
      \csuse{Xendmorethantwolines@\@currentseries}%
    \else%
      \csuse{Xendtwolines@\@currentseries}%
    \fi%
  \else%
    \ifl@d@dash%
      \ifdefined\linerangesep@%
        \linerangesep@%
      \else%
        \csuse{Xendlinerangeseparator@\@currentseries}%
      \fi%
    \fi%
%
```

Print the ending page number.

```
    \ifl@d@pnum%
     \ifcsstring{prevpagerange@\@currentseries}{#1-#4}%
       {%
       \ifcsempty{Xendsympagenum@\@currentseries}%
         {\hspace{\csuse{Xendinplaceofpagenumber@\@currentseries}}}%
         {\csuse{Xendsympagenum@\@currentseries}}%
       }%
       {%
       \wrap@edcrossref{\@this@crossref@end}\printnpnum{#4}%
       }%
    \fi%
%
```

Print the ending line number, with, if needed, the line prefix, and followed by the side flag and the subline number.

```
    \ifl@d@elin%
      \ifl@d@pnum\csuse{Xendlineprefixsingle@\@currentseries}\fi%
      \wrap@edcrossref{\@this@crossref@end}{%
        \ifledRcol@%
          \linenumrepR{#5}%
        \else%
          \linenumrep{#5}%
        \fi%
      }%
      \iftoggle{Xendlineflag@\@currentseries}{\ifledRcol@\@Rlineflag\fi}{}%
    \fi%
    \ifl@d@esl%
      \ifl@d@elin%
        \csuse{Xendsublinesep@\@currentseries}%
      \fi%
      \wrap@edcrossref{\@this@crossref@end}{%
        \ifledRcol@%
          \sublinenumrepR{#6}%
        \else%
          \sublinenumrep{#6}%
        \fi%
      }%
    \fi%
```

```
  \fi%
%
```

Close the ending line box.

```
  \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
    {}%
    {\hfill}%Prevent underfull hbox
  \egroup%
%
```

And, finally, save, if needed, the current page number for the Xendpagenumberonlyfirst hooks.

```
  \iftoggle{Xendpagenumberonlyfirst@\@currentseries}%
    {\iftoggle{Xendpagenumberonlyfirstintwo@\@currentseries}%
      {\csxdef{prevpagerange@\@currentseries}{#1-#4}}%
      {\csxdef{prevpagenum@\@currentseries}{#4}}%
    }%
    {}%
%
```

Now, the end of `\printendlines` macro.

```
  \endgroup%
}%

%
```

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

```
\newcommand*{\printnpnum}[1]{\csuse{Xendbeforepagenumber@\@currentseries}#1\csuse{Xendafterpagenumber@\@currentseries}}

%
```

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@`.

```
\newcommand{\newseries}[1]{%
    \def\do##1{\newseries@{##1}}%
    \docsvlist{#1}
}
%
```

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

```
\newcommand{\@series}{}
%
```

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

\newseries@

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

XX.1 Test if series is still existing

```
    \xifinlist{#1}{\@series}{\led@warn@SeriesStillExist{#1}}%
    {%
%
```

XX.2 Init specific to `reledpar`

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

```
      \ifdefined\newseries@par%
        \newseries@par{#1}%
      \fi%
%
```

XX.3 For critical footnotes

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

```
    \unless\ifnocritical@
%
```

XX.3.1 Options

```
      \newtoggle{Xlineflag@#1}
      \newtoggle{Xparindent@#1}
      \newtoggle{Xlemmadisablefontselection@#1}
      \csgdef{Xwrapcontent@#1}{}%
      \csgdef{Xbeforeinserting@#1}{}%
      \csgdef{Xhangindent@#1}{0pt}%
      \csgdef{Xragged@#1}{}%
      \csgdef{Xhsizetwocol@#1}{0.45 \hsize}%
      \csgdef{Xhsizethreecol@#1}{.3 \hsize}%
      \csgdef{Xcolalign@#1}{\raggedright}%
      \csgdef{Xnotenumfont@#1}{\normalfont}%
      \csgdef{Xnotefontsize@#1}{\footnotesize}%
      \csgdef{Xbhooknote@#1}{}%
      \csgdef{Xbhookgroup@#1}{}%

      \csgdef{Xboxlinenum@#1}{0pt}%
      \csgdef{Xboxlinenumalign@#1}{L}%

      \csgdef{Xboxstartlinenum@#1}{0pt}%
      \csgdef{Xboxendlinenum@#1}{0pt}%

      \csgdef{Xboxsymlinenum@#1}{0pt}%
      \newtoggle{Xnumberonlyfirstinline@#1}%
      \newtoggle{Xgroupbyline@#1}%
      \newtoggle{Xgroupbylineseparetwolines@#1}%
      \newtoggle{Xnumberonlyfirstintwolines@#1}%
      \csgdef{Xtwolines@#1}{}%
      \csgdef{Xmorethantwolines@#1}{}%
      \csgdef{Xsublinesep@#1}{\fullstop}%
      \csgdef{Xpagelinesep@#1}{\csname Xsublinesep@#1\endcsname}%for backward compatibility, call Xsublinesep@#1
      \newtoggle{Xtwolinesbutnotmore@#1}%
      \newtoggle{Xtwolinesonlyinsamepage@#1}%
      \newtoggle{Xonlypstart@#1}%
      \newtoggle{Xpstarteverytime@#1}%
      \newtoggle{Xpstart@#1}%
      \newtoggle{Xstanza@#1}%
      \csgdef{Xstanzaseparator@#1}{}%
      \csgdef{Xsymlinenum@#1}{}%
      \newtoggle{Xnonumber@#1}%
      \csgdef{Xbeforenumber@#1}{0pt}%
      \csgdef{Xtxtbeforenumber@#1}{}%
      \csgdef{Xafternumber@#1}{0.5em}%
      \newtoggle{Xnonbreakableafternumber@#1}%
      \csgdef{Xbeforesymlinenum@#1}{\csuse{Xbeforenumber@#1}}%
      \csgdef{Xaftersymlinenum@#1}{\csuse{Xafternumber@#1}}%
      \csgdef{Xinplaceofnumber@#1}{1em}%
      \global\cslet{Xlemmaseparator@#1}{\rbracket}%
      \csgdef{Xbeforelemmaseparator@#1}{0em}%
      \csgdef{Xafterlemmaseparator@#1}{0.5em}%
      \csgdef{Xinplaceoflemmaseparator@#1}{1em}%
      \csgdef{Xbeforenotes@#1}{1.2em \@plus .6em \@minus .6em}
      \csgdef{Xafterrule@#1}{0pt}

      \csgdef{Xtxtbeforenotes@#1}{}
      \newtoggle{Xtxtbeforesnotes@#1@typeset}%Not directly used by user, but internal
      \newtoggle{Xtxtbeforenotesonlyonce@#1}%

      \csgdef{Xmaxhnotes@#1}{0.8\vsize}
      \newtoggle{Xnoteswidthliketwocolumns@#1}%
      \csgdef{Xparafootsep@#1}{}%
      \csgdef{Xafternote@#1}{1em plus.4em minus.4em}
      \csgdef{Xlinerangeseparator@#1}{\endashchar}%

      \csgdef{Xlemmafont@#1}{}%
      \csgdef{Xwraplemma@#1}{}
      \csgdef{Xwidth@#1}{\hsize}%
%
```

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

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

```
      \expandafter\newinsert\csname #1footins\endcsname%
      \unless\ifnoledgroup@%
        \expandafter\newinsert\csname mp#1footins\endcsname%
      \fi%
%
```

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.

```
      \global\notbool{parapparatus@}{\expandafter\newcommand\expandafter
*}{\expandafter\newcommand}\csname #1footnote\endcsname[2][]{%
          \if@edtext@secondarg@%
            \ifledRcol%
              \ifcsstring{Xonlyside@#1}{L}{\
led@error@note@called@onrightside{#1footnote}}{}%
            \else%
              \ifcsstring{Xonlyside@#1}{R}{\
led@error@note@called@onleftside{#1footnote}}{}%
            \fi%
            \begingroup%
            \newcommand{\content}{##2}%
            \ifnumberedpar@%
                \ifledRcol%
                   \ifluatex%
                       \footnotelang@lua[R]%
                   \fi%
                   \@ifundefined{xpg@main@language}%if polyglossia
                         {}%
                         {\footnotelang@poly[R]}%
                    \footnoteoptions@{R}{##1}{true}%
                    \xright@appenditem{%
                      \ifbool{indtl@innote}%
                        {\unexpanded{\let\index\nindex}}%
                        {}%
                      \ifbool{indtl@notenumber}%
                        {\unexpanded{\let\index\nindex}}%There is no note
…number so
                        {}%
                      \noexpand\Xnote@true%
                      \noexpand\prepare@Xprenotes{#1}%
                      \noexpand\prepare@edindex@fornote{\l@d@nums}%
                      \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.
                      \noexpand\setcounter{stanzaR}{\the\c@stanzaR}%Save
stanzaR counter for footnote
                      \unexpanded{\def\@this@crossref@start}{\theedtext:
start}%
                      \unexpanded{\def\@this@crossref@end}{\theedtext:end}%
                      \expandonce{\@beforeinsertofthisedtext}% Internal for
 now, no reason to make it public
                      \noexpand\csuse{v#1footnote}{#1}%
                       {{\l@d@nums}{\expandonce\@tag}{\expandonce\content}}
%
                      \noexpand\Xnote@false%
                      \unexpanded{\advance\@edindex@fornote@\m@ne}%
                      \unexpanded{\undef\@this@crossref@start}%
                      \unexpanded{\undef\@this@crossref@end}%
                      \ifbool{indtl@innote}%
                        {\unexpanded{\let\index\orig@@index}}%
                        {}%
                      \ifbool{indtl@notenumber}%
                        {\unexpanded{\let\index\orig@@index}}%
                        {}%
                    }\to\inserts@listR
                    \footnoteoptions@{R}{##1}{false}%
                    \global\advance\insert@countR \@ne%
                \else%
                     \ifluatex%
                       \footnotelang@lua%
                     \fi%
                   \@ifundefined{xpg@main@language}%if polyglossia
                         {}%
                         {\footnotelang@poly}%
                    \footnoteoptions@{L}{##1}{true}%
                    \xright@appenditem{%
                      \ifbool{indtl@innote}%
                        {\unexpanded{\let\index\nindex}}%
                        {}%
                      \ifbool{indtl@notenumber}%
                        {\unexpanded{\let\index\nindex}}%There is no note
…number so
                        {}%
                      \noexpand\Xnote@true%
                      \noexpand\prepare@Xprenotes{#1}%
                      \noexpand\prepare@edindex@fornote{\l@d@nums}%
                      \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.
                      \ifl@dpairing%
                        \noexpand\setcounter{stanzaL}{\the\c@stanzaL}%Save stanzaR counter for footnote
                      \fi%
                      \unexpanded{\def\@this@crossref@start}{\theedtext:start}%
                      \unexpanded{\def\@this@crossref@end}{\theedtext:end}%
                      \expandonce{\@beforeinsertofthisedtext}%Internal for now, no reason to make it public
                      \noexpand\csuse{v#1footnote}%
                         {#1}%
                         {{\l@d@nums}{\expandonce\@tag}{\expandonce\content}}%
                      \unexpanded{\undef\@this@crossref@start}%
                      \unexpanded{\undef\@this@crossref@end}%
                      \noexpand\Xnote@false%
                      \unexpanded{\advance\@edindex@fornote@\m@ne}%
                      \ifbool{indtl@innote}%
                        {\unexpanded{\let\index\orig@@index}}%
                        {}%
                      \ifbool{indtl@notenumber}%
                        {\unexpanded{\let\index\orig@@index}}%
                        {}%
                     }\to\inserts@list
                    \global\advance\insert@count \@ne%
                    \footnoteoptions@{L}{##1}{false}%
                \fi
            \else
                \csuse{v#1footnote}{#1}{{0|0|0|0|0|0|0}{}{##1}}%
            \fi%
            \endgroup%
          \else%
            \led@err@FootnoteNotInSecondArgEdtext{#1}%
          \fi%
   \ignorespaces%
          }
%
```

Create counter used to determine on which page the previous note was called.

```
      \expandafter\newcount\csname #1prevpage@num\endcsname%
      \expandafter\newcount\csname #1prevpage@numR\endcsname%
%
```

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

```
      \global\csletcs{#1@@footnote}{#1footnote}
%
```

XX.3.4 Set standard display

```
      \Xarrangement@normal{#1}%
%
```

End of for critical footnotes.

```
    \fi
%
```

XX.4 For familiar footnotes

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

```
    \unless\ifnofamiliar@
%
```

XX.4.1 Options

```
      \newtoggle{parindentX@#1}
      \csgdef{wrapcontentX@#1}{}%
      \csgdef{hangindentX@#1}{0pt}%
      \csgdef{beforeinsertingX@#1}{}%
      \csgdef{raggedX@#1}{}%
      \csgdef{hsizetwocolX@#1}{0.45 \hsize}%
      \csgdef{hsizethreecolX@#1}{.3 \hsize}%
      \csgdef{colalignX@#1}{\raggedright}%
      \csgdef{notenumfontX@#1}{\normalfont}%
      \csgdef{notefontsizeX@#1}{\footnotesize}%
      \csgdef{bhooknoteX@#1}{}%
      \csgdef{bhookgroupX@#1}{}%
      \csgdef{afterruleX@#1}{0pt}
      \csgdef{beforenotesX@#1}{1.2em \@plus .6em \@minus .6em}
      \csgdef{maxhnotesX@#1}{0.8\vsize}%
      \newtoggle{noteswidthliketwocolumnsX@#1}%
      \csgdef{parafootsepX@#1}{}%
      \csgdef{afternoteX@#1}{1em plus.4em minus.4em}
      \csgdef{widthX@#1}{\hsize}%
      \csgdef{txtbeforenotesX@#1}{}%
      \newtoggle{txtbeforesnotesX@#1@typeset}%Not directly used by user,
but internal
      \newtoggle{txtbeforenotesonlyonceX@#1}%
% End of for familiar footnotes.
% \subsubsection{Create inserts, needed to add notes in foot}
% As regards inserts, see chapter 15 of the TeXBook by D. Knuth.
%    \begin{macrocode}
      \expandafter\newinsert\csname footins#1\endcsname%
      \unless\ifnoledgroup@%
        \expandafter\newinsert\csname mpfootins#1\endcsname%
      \fi%
```

```
%
```

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.

```

      \global\expandafter\newcommand\csname footnote#1\endcsname[1]{%
          \begingroup%
              \prepare@prenotesX{#1}%
              \newcommand{\content}{##1}%
%
```

If we use the \csquotes package, we reset quotation level.

```
              \ifdefined\csq@qlevel%
                \csq@qlevel=0\relax%
              \fi%
%
```

If we are preparing parallel typesetting, we cannot just increase the footnote counter. Read reledpar's handbook about that (V.1.2 p. 50).

```
               \global\expandafter\advance\csname footnote#1@reading\
endcsname by \@ne%
               \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or
bool{l@dprintingcolumns}}{%
                  \ifcsdef{footnote#1reading\the\csname footnote#1@reading\
endcsname=typeset}%
                    {\setcounter{footnote#1}{\csuse{footnote#1reading\the\
csname footnote#1@reading\endcsname=typeset}}}%
                    {\setcounter{footnote#1}{\the\csname footnote#1@reading
\endcsname}}%
               }{%
                  \stepcounter{footnote#1}%
               }%
%
```

We also have to check consistency with \onlysideX setting.

```
            \ifledRcol%
              \ifcsstring{onlysideX@#1}{L}{\
led@error@note@called@onrightside{footnote#1}}{}%
            \else%
              \ifcsstring{onlysideX@#1}{R}{\
led@error@note@called@onleftside{footnote#1}}{}%
            \fi%
%
```

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

```
              \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
              \nottoggle{nomk@}%Nomk is set to true when using \
footnoteXnomk with \parpackage
```

```
                 {\csuse{@footnotemark#1}}%
                 {}%
              \ifluatex%
                \xdef\footnote@luatextextdir{\the\textdir}%
                \xdef\footnote@luatexpardir{\the\pardir}%
              \fi%
              \if@ledgroup%
                \led@set@index@fornote{#1}%
              \fi%
              \csuse{vfootnote#1}{#1}{\expandonce\content}\m@mmf@prepare%
              \ifbool{indtl@innote}%
                {\let\index\orig@@index}%
                {}%
              \ifbool{indtl@notenumber}%
                {\let\index\orig@@index}%
                {}%
          \endgroup%
      }
%
```

Then define the counters. The LaTeX counter `footnoteX` is the only one manipulated by the user. This is the one which is printed. The TeX counter `\footnoteX@reading` is increased at each footnote. It is used for hyperlinks, for using `hyperlink` package, and for getting the correct footnote number when using parallel typesetting (V.1.2 p. 50).

```
      \newcounter{footnote#1}
        \global\expandafter\renewcommand\csname thefootnote#1\endcsname{\
arabic{footnote#1}}
      \expandafter\newcount\csname footnote#1@reading\endcsname%
%
```

Create counter used to determine on which page the previous note was called.

```
      \expandafter\newcount\csname prevpage#1@num\endcsname%
      \expandafter\newcount\csname prevpage#1@numR\endcsname%
%
```

Add `\let\footnoteX\@gobble` to `\no@expands`.

```
      \expandafter\gappto\expandafter\no@expands\expandafter{\expandafter\
let\csname footnote#1\endcsname\@gobble}%
%
```

And now, define `\footnoteXmark` and `\footnoteXtext`, equivalent to classical`\footnotemark` and `\footnotetext`.

```
      \expandafter\newcommand\csname footnote#1mark\endcsname{%
         \begingroup%
            \prepare@prenotesX{#1}%
            \stepcounter{footnote#1}%
            \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
            \csuse{@footnotemark#1}%
            \m@mmf@prepare%
         \endgroup%
      }%
      \expandafter\newcommand\csname footnote#1text\endcsname[1]{%
         \begingroup%
            \csuse{vfootnote#1}{#1}{\expandonce{##1}}%
         \endgroup%
      }%
%
```

Do not forget to initialize the series.

```
     \arrangementX@normal{#1}%
  \fi
%
```

XX.5 The endnotes

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

```
  \unless\ifnoend@
%
```

XX.5.1 The auxiliary file

\l@d@Xend
\ifl@dend@X
\l@dend@Xtrue
\l@dend@Xfalse

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

```
      \expandafter\newwrite\csname l@d@#1end\endcsname%
      \expandafter\newif\csname ifl@dend@#1\endcsname%
%
```

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.

```

      \global\expandafter\newcommandx\csname #1endnote\endcsname[2][1,
usedefault]{%
         \bgroup%
         \newlinechar='40%
         \global\@noneed@Footnotetrue%
         \newcommand{\content}{##2}%
         \stepcounter{labidx}%
         \expandafter\immediate\expandafter\write\csname l@d@#1end\
endcsname{%
```

```
           \unexpanded{\def\sw@list@inedtext}{\expandafter\unexpanded\expandafter{\sw@inthisedtext}}\@percentchar\space%Explicit space, to add a linebreak in the output file
           \expandafter\string\csname #1end\endcsname%
           {\ifnumberedpar@\l@d@nums\fi}%
           {\ifnumberedpar@\expandonce\@tag\fi}%
           {\expandonce\content}%
           {#1}%
           {\unexpanded{##1}}%
           {\ifledRcol R\else L\fi}%
           {\theedtext}%
           \@percentchar%
         }%
         \egroup%
         \ignorespaces%
     }%
%
```

XX.5.3 Tools

The \Xtoendnotes command inserts any arbitrary content into the endnote file. It is an alias of the more generalist \addtoendnotes

```

     \global\expandafter\newcommand\csname #1toendnotes\endcsname[1]{%
        \toendnotes[#1]{##1}%
     }%

     \expandafter\WithSuffix\expandafter\newcommand\csname #1toendnotes\endcsname*[1]{%
       \toendnotes*[#1]{##1}%
     }%

%
```

XX.5.4 Internal commands

\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 \@gobbleseven, which just skips over its seven arguments.

```

      \global\cslet{#1end}{\@gobbleseven}
%
```

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

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

XX.5.5 The options

```
      \csgdef{Xendwraplemma@#1}{}
      \csgdef{Xendwrapcontent@#1}{}%
      \csgdef{Xendtwolines@#1}{}%
      \csgdef{Xendmorethantwolines@#1}{}%
      \newtoggle{Xendtwolinesbutnotmore@#1}{}%
      \newtoggle{Xendtwolinesonlyinsamepage@#1}{}%
      \newtoggle{Xendlemmadisablefontselection@#1}%
      \csgdef{Xendnotenumfont@#1}{\normalfont}%
      \csgdef{Xendnotefontsize@#1}{\footnotesize}%
      \csgdef{Xendbhooknote@#1}{}%

      \csgdef{Xendsublinesep@#1}{\fullstop}%

      \csgdef{Xendbeforenumber@#1}{0pt}
      \csgdef{Xendafternumber@#1}{0.5em}

      \csgdef{Xendboxlinenum@#1}{0pt}%
      \csgdef{Xendboxlinenumalign@#1}{L}%

      \csgdef{Xendboxstartlinenum@#1}{0pt}%
      \csgdef{Xendboxendlinenum@#1}{0pt}%

      \csgdef{Xendlemmaseparator@#1}{}%
      \csgdef{Xendbeforelemmaseparator@#1}{0em}%
      \csgdef{Xendafterlemmaseparator@#1}{0.5em}%
      \csgdef{Xendinplaceoflemmaseparator@#1}{0.5em}%

      \newtoggle{Xendparagraph@#1}%
      \csgdef{Xendafternote@#1}{1em plus.4em minus.4em}%
      \csgdef{Xendsep@#1}{}%

      \csgdef{Xendinplaceofnumber@#1}{0pt}%
      \newtoggle{Xendnonumber@#1}%

      \csgdef{Xendhangindent@#1}{0pt}%
      \newtoggle{Xendnumberonlyfirstinline@#1}%
      \newtoggle{Xendnumberonlyfirstintwolines@#1}%

      \csgdef{Xendbeforesymlinenum@#1}{\csuse{Xendbeforenumber@#1}}%
      \csgdef{Xendaftersymlinenum@#1}{\csuse{Xendafternumber@#1}}%
      \csgdef{Xendsymlinenum@#1}{}%
      \csgdef{Xendboxsymlinenum@#1}{0pt}%

      \csgdef{Xendbhooklinenumber@#1}{}%
      \csgdef{Xendehooklinenumber@#1}{}%
      \csgdef{Xendbhookinplaceofnumber@#1}{}%
      \csgdef{Xendehookinplaceofnumber@#1}{}%

      \csgdef{Xendlinerangeseparator@#1}{\endashchar}%

      \csgdef{Xendbeforepagenumber@#1}{p.}%
      \csgdef{Xendafterpagenumber@#1}{) }%
      \csgdef{Xendlineprefixsingle@#1}{}%
      \csgdef{Xendlineprefixmore@#1}{}%

      \newtoggle{Xendlineflag@#1}

      \csgdef{Xendlemmafont@#1}{}%

  \newtoggle{Xendpagenumberonlyfirst@#1}%
  \newtoggle{Xendpagenumberonlyfirstifsingle@#1}%
  \newtoggle{Xendpagenumberonlyfirstintwo@#1}%
  \csgdef{Xendsympagenum@#1}{}%
  \csgdef{Xendinplaceofpagenumber@#1}{0pt}%

%
```

End of endnotes declaration

```
  \fi%
%
```

Dump series in \@series

```
    \listxadd{\@series}{#1}
  }
}% End of \newseries
%
```

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

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

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.

```
\newcommand{\seriesatbegin}[1]{%
   \StrDel{\@series}{#1}[\@series]%
   \edef\@new{}%
   \listeadd{\@new}{#1}%
   \listeadd{\@new}{\@series}%
   \xdef\@series{\@new}%
}
%
```

\seriesatend

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

```
\newcommand{\seriesatend}[1]{%
   \StrDel{\@series}{#1}[\@series]%
   \edef\@new{}%
   \listeadd{\@new}{\@series}%
   \listeadd{\@new}{#1}%
   \xdef\@series{\@new}%
}
%
```

XXI.2 Test series order

\ifseriesbefore

\ifseriesbefore{⟨*seriesA*⟩}{⟨*seriesB*⟩}{⟨*true*⟩}{⟨*false*⟩} expands ⟨*true*⟩ if ⟨*seriesA*⟩ is printed before ⟨*seriesB*⟩, expands ⟨*false*⟩ otherwise.

```
\newcommand{\ifseriesbefore}[4]{%
 \StrPosition{\@series}{#1}[\@first]%
 \StrPosition{\@series}{#2}[\@second]%
 \ifnumgreater{\@second}{\@first}{#3}{#4}%
}
%
```

XXI.2.1 Get the first series

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

\@getfirstseries

```
\newcommand{\@getfirstseries}{%
 \ifdefempty{\@series}%
   {\xdef\@firstseries{}}%
   {\StrChar{\@series}{1}[\@firstseries]}%
}%
%
```

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.

```
\newcommandx{\settoggle@series}[5][4,5,usedefault]{%
    \def\do##1{%
      \global\settoggle{#2@##1}{#3}%
      \ifstrequal{#4}{critical}{
          \csuse{Xarrangement@\csuse{series@display##1}}{##1}%
      }{}
      \ifstrequal{#4}{familiar}{
          \csuse{arrangementX@\csuse{series@displayX##1}}{##1}%
      }{}
    }%
    \ifstrempty{#1}{%
              \dolistloop{\@series}%
              \ifstrempty{#5}{}{%
                 \docsvlist{#5}%
               }
           }%
           {%
              \docsvlist{#1}%
           }%
}
%
```

\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`.

```
\newcommandx{\setcommand@series}[5][4,5,usedefault]{%
    \def\do##1{
        \csgdef{#2@##1}{#3}
      \ifstrequal{#4}{critical}{%
            \csuse{Xarrangement@\csuse{series@display##1}}{##1}%
      }{}
      \ifstrequal{#4}{familiar}{%
          \csuse{arrangementX@\csuse{series@displayX##1}}{##1}%
      }{}%
    }%
    \ifstrempty{#1}{%
              \dolistloop{\@series}%
              \ifstrempty{#5}{}{%
                 \docsvlist{#5}
               }
    }%
    {%
              \docsvlist{#1}%
    }%
}%
%
```

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><pseudoseries>`, like `\Xtwolinesappref`.

```
\newcommandx{\newhookcommand@series}[2][2,usedefault]{%
  \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][]{%
    \setcommand@series{##1}{#1}{##2}[][#2]%
  }%
  \ifstrempty{#2}{}{%
    \def\do##1{%
```

```
      \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname[1]{%
        \csuse{#1}[##1]{####1}%
      }%
    }%
    \docsvlist{#2}%
  }%
}
%
```

\newhooktoggle@series

\newhooktoggle@series\command names is a generic command to add new commands for a new toggle hook, like \Xnumberonlyfirstinline. The second argument is also used to create commands named \<hookname><pseudoseries>, like \Xtwolinesbutnotmoreappref.

```
\newcommandx{\newhooktoggle@series}[2][2,usedefault]{%
  \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={true},usedefault]{%
    \settoggle@series{##1}{#1}{##2}[][#2]%
    }%
  \ifstrempty{#2}{}{%
    \def\do##1{%
      \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname{%
        \csuse{#1}[##1]%
      }%
    }%
    \docsvlist{#2}%
  }%
}
%
```

\newhooktoggle@series@reload

\newhookcommand@toggle@reload does the same thing as \newhooktoggle@series but the commands created by this macro also reload the series arrangement, depending of type os notes

```
\newcommand{\newhooktoggle@series@reload}[2]{%
  \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={true},usedefault]{%
    \settoggle@series{##1}{#1}{##2}[#2]%
    }%
}%
%
```

\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.

```
\newcommand{\newhookcommand@series@reload}[2]{%
  \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][]{%
    \setcommand@series{##1}{#1}{##2}[#2]%
  }%
}
%
```

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.

```
\unless\ifnocritical@
  \newhookcommand@series{Xwrapcontent}%
  \newhookcommand@series{Xbeforeinserting}%
  \newhookcommand@series{Xlemmafont}%
  \newhookcommand@series{Xwraplemma}%
  \newhooktoggle@series{Xparindent}
  \newhookcommand@series{Xhangindent}
  \newhookcommand@series{Xragged}
  \newhookcommand@series{Xhsizetwocol}
  \newhookcommand@series{Xhsizethreecol}
  \newhookcommand@series{Xcolalign}%
  \newhookcommand@series{Xnotenumfont}
  \newhookcommand@series{Xbhooknote}
  \newhookcommand@series@reload{Xbhookgroup}{critical}
  \newhookcommand@series{Xboxsymlinenum}%
  \newhookcommand@series{Xsymlinenum}
  \newhookcommand@series{Xbeforenumber}
  \newhookcommand@series{Xtxtbeforenumber}
  \newhookcommand@series{Xafternumber}
  \newhookcommand@series{Xbeforesymlinenum}
  \newhookcommand@series{Xaftersymlinenum}
  \newhookcommand@series{Xinplaceofnumber}
  \newhookcommand@series{Xlemmaseparator}
  \newhookcommand@series{Xbeforelemmaseparator}
  \newhookcommand@series{Xafterlemmaseparator}
  \newhookcommand@series{Xinplaceoflemmaseparator}
  \newhookcommand@series{Xtxtbeforenotes}
  \newhooktoggle@series{Xtxtbeforenotesonlyonce}%
  \newhookcommand@series@reload{Xafterrule}{critical}
  \newhooktoggle@series{Xnumberonlyfirstinline}
  \newhooktoggle@series{Xnumberonlyfirstintwolines}
  \newhooktoggle@series{Xgroupbyline}%
  \newhooktoggle@series{Xgroupbylineseparetwolines}%
  \newhooktoggle@series{Xnonumber}
  \newhooktoggle@series{Xpstart}
  \newhooktoggle@series{Xpsteverytime}%

  \newhooktoggle@series{Xstanza}%
  \newhookcommand@series{Xstanzaseparator}%

  \newhooktoggle@series{Xonlypstart}
  \newhooktoggle@series{Xnonbreakableafternumber}
  \newhooktoggle@series{Xlemmadisablefontselection}
  \newhookcommand@series@reload{Xmaxhnotes}{critical}
  \newhookcommand@series@reload{Xbeforenotes}{critical}
  \newhooktoggle@series@reload{Xnoteswidthliketwocolumns}{critical}%
  \newhookcommand@series@reload{Xnotefontsize}{critical}

  \newhookcommand@series{Xboxlinenum}%
  \newhookcommand@series{Xboxlinenumalign}%

  \newhookcommand@series{Xboxstartlinenum}%
  \newhookcommand@series{Xboxendlinenum}%

  \newhookcommand@series{Xafternote}%
  \newhookcommand@series{Xparafootsep}

  \newhookcommand@series@reload{Xwidth}{critical}%

  \ifundef{\Xhsize}%
    {%
      \newcommandx{\Xhsize}[2][1,usedefault]{%
        \led@warning@Xhsize@deprecated%
        \Xwidth[#1]{#2}%
      }%
    }%
    {}%
\fi
\newhooktoggle@series{Xlineflag}[appref,SEref]
\newhookcommand@series{Xtwolines}[appref,SEref]
\newhookcommand@series{Xmorethantwolines}[appref,SEref]
\newhookcommand@series{Xsublinesep}[appref,SEref,side]%
\newhookcommand@series{Xpagelinesep}[appref,SEref,side]%
\newhooktoggle@series{Xtwolinesbutnotmore}[appref,SEref]
\newhooktoggle@series{Xtwolinesonlyinsamepage}[appref,SEref]
\newhookcommand@series{Xlinerangeseparator}[appref,SEref]
%
```

XXI.3.5 Options for familiar notes

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

```
\unless\ifnofamiliar@
  \newhookcommand@series{wrapcontentX}%
  \newhookcommand@series{beforeinsertingX}%
  \newhooktoggle@series{parindentX}
  \newhookcommand@series{hangindentX}
  \newhookcommand@series{raggedX}
  \newhookcommand@series{hsizetwocolX}
  \newhookcommand@series{hsizethreecolX}
  \newhookcommand@series{colalignX}%
  \newhookcommand@series{notenumfontX}
  \newhookcommand@series{bhooknoteX}
  \newhookcommand@series@reload{bhookgroupX}{familiar}
  \newhookcommand@series@reload{beforenotesX}{familiar}
  \newhookcommand@series@reload{maxhnotesX}{familiar}
  \newhooktoggle@series@reload{noteswidthliketwocolumnsX}{familiar}%
  \newhookcommand@series@reload{afterruleX}{familiar}
  \newhookcommand@series@reload{notefontsizeX}{familiar}
  \newhookcommand@series{afternoteX}
  \newhookcommand@series{parafootsepX}
  \newhookcommand@series{txtbeforenotesX}%
  \newhooktoggle@series{txtbeforenotesonlyonceX}%
  \newhookcommand@series@reload{widthX}{familiar}%
  \ifundef{\hsizeX}%
    {%
      \newcommandx{\hsizeX}[2][1,usedefault]{%
        \led@warning@hsizeX@deprecated%
        \widthX[#1]{#2}%
      }%
    }%
    {}%
\fi
%
```

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.

```
\unless\ifnoend@
  \newhookcommand@series{Xendwraplemma}
  \newhookcommand@series{Xendwrapcontent}
  \newhookcommand@series{Xendnotenumfont}
  \newhookcommand@series{Xendlemmafont}%
  \newhookcommand@series{Xendbhooknote}

  \newhookcommand@series{Xendboxlinenum}%
  \newhookcommand@series{Xendboxlinenumalign}%

  \newhookcommand@series{Xendboxstartlinenum}%
  \newhookcommand@series{Xendboxendlinenum}%

  \newhookcommand@series{Xendnotefontsize}
  \newhooktoggle@series{Xendlemmadisablefontselection}
  \newhookcommand@series{Xendlemmaseparator}
  \newhookcommand@series{Xendbeforelemmaseparator}
  \newhookcommand@series{Xendafterlemmaseparator}
  \newhookcommand@series{Xendinplaceoflemmaseparator}

  \newhookcommand@series{Xendbeforenumber}%
  \newhookcommand@series{Xendafternumber}%

  \newhooktoggle@series{Xendparagraph}
  \newhookcommand@series{Xendafternote}
  \newhookcommand@series{Xendsep}

  \newhookcommand@series{Xendinplaceofnumber}%
  \newhooktoggle@series{Xendnonumber}%

  \newhooktoggle@series{Xendnumberonlyfirstinline}%
  \newhooktoggle@series{Xendnumberonlyfirstintwolines}%

  \newhookcommand@series{Xendsymlinenum}%
  \newhookcommand@series{Xendbeforesymlinenum}%
  \newhookcommand@series{Xendaftersymlinenum}%
  \newhookcommand@series{Xendboxsymlinenum}%

  \newhookcommand@series{Xendbhooklinenumber}%
  \newhookcommand@series{Xendahooklinenumber}%
  \newhookcommand@series{Xendbhookinplaceofnumber}%
  \newhookcommand@series{Xendahookinplaceofnumber}%

  \newhookcommand@series{Xendhangindent}%

 \newhooktoggle@series{Xendpagenumberonlyfirst}%
 \newhooktoggle@series{Xendpagenumberonlyfirstifsingle}%
 \newhooktoggle@series{Xendpagenumberonlyfirstintwo}%
 \newhookcommand@series{Xendsympagenum}%
 \newhookcommand@series{Xendinplaceofpagenumber}%

\fi
\newhooktoggle@series{Xendlineflag}[apprefwithpage,SErefwithpage]
\newhookcommand@series{Xendtwolines}[apprefwithpage,SErefwithpage]
\newhookcommand@series{Xendmorethantwolines}[apprefwithpage,SErefwithpage]
\newhooktoggle@series{Xendtwolinesbutnotmore}[apprefwithpage,SErefwithpage]
\newhooktoggle@series{Xendtwolinesonlyinsamepage}[apprefwithpage,SErefwithpage]
\newhookcommand@series{Xendlinerangeseparator}[apprefwithpage,SErefwithpage]
\newhookcommand@series{Xendbeforepagenumber}[apprefwithpage,SErefwithpage,SErefonlypage]
\newhookcommand@series{Xendafterpagenumber}[apprefwithpage,SErefwithpage]
\newhookcommand@series{Xendlineprefixsingle}[apprefwithpage,SErefwithpage]
\newhookcommand@series{Xendlineprefixmore}[apprefwithpage,SErefwithpage]
\newhookcommand@series{Xendsublinesep}[apprefwithpage,SErefwithpage]

%
```

XXI.4 Hooks for a particular footnote

\newhooktoggle@specific

\newhooktoggle@specific is a generic command to create boolean hook specific to a note.

```
\newcommand{\newhooktoggle@specific}[1]{%
  \newtoggle{#1@}%
  \define@key[mac]{truefootnoteoption}{#1}[]{\global\settoggle{#1@}{true}}%
When enabling footnote option
  \define@key[mac]{falsefootnoteoption}{#1}[]{\global\settoggle{#1@}{false
}}
}
%
```

\newhookarg@specific

\newhookarg@specific is a generic command to create argumen hook specific to a note.

```
\newcommand{\newhookarg@specific}[1]{%
  \define@key[mac]{truefootnoteoption}{#1}{\global\def\linerangesep@{##1}}%
When enabling footnote option
  \define@key[mac]{falsefootnoteoption}{#1}{\global\undef\linerangesep@}%
When
}
%
```

And now, we define some hooks specific to a note.

```
\newhooktoggle@specific{fulllines}%
\newhooktoggle@specific{nonum}
\newhooktoggle@specific{nosep}
\newhookarg@specific{linerangesep}
%
```

linerangesep@

\linerangesep@ is defined by the option linerangesep of critical notes to change temporarily the line range separator for a specific line. As we have to define it before typesetting the line and undefine it after, we use the family of xkeyval package's key.

```
%
```

\nomk@

\nomk@ toggle is used by reledpar to remove the footnote mark in the text when using \footnoteXmk. Read reledpar handbook.

```
\newtoggle{nomk@}%
%
```

XXI.5 Alias

\Xnolemmaseparator

\Xnolemmaseparator[⟨*series*⟩] is just an alias for \Xlemmaseparator[⟨*series*⟩]{}.

```
\newcommandx*{\Xnolemmaseparator}[1][1]{\Xlemmaseparator[#1]{}}
%
```

XXII Output routine

Now we begin the output routine and associated things.

XXII.1 Extra footnotes output

With luck we might only have to change \@makecol and \@reinserts of the LaTeX's kernel. Since reledmac, we use etoolbox's patching commands instead of overriding. It should provides better compatibility with other package which modify these commands

\doxtrafeet

\doxtrafeet is the code extending \@makecol to cater for the extra reledmac feet. We have two categories of extra footnotes. By default, we order the footnote inserts so that the regular footnotes of LaTeX are first, then familiar familiar footnotes and finally the critical footnotes.

```
\newcommand*{\l@ddoxtrafeet}{%
  \IfStrEq{familiar-critical}{\@fnpos}
    {\do@feetX\do@Xfeet}%
    {%
    \IfStrEq{critical-familiar}{\@fnpos}%
        {\do@Xfeet\do@feetX}%
        {%
          \setbox\@outputbox \vbox{%
            \unvbox\@outputbox%
            \do@feet@custom@order{}{\@fnpos}%
          }%
        }%
    }%
}%

%
```

\do@feet@custom@order

\do@feet@custom@order is called when \@fnpos is neither 'familiar-critical', nor 'critical-familiar', that is, when the order is more complex. In this case, people must define the order for all footnote series. If they don't, LaTeX could perform an infinite run.

```
\newcommand{\do@feet@custom@order}[2]{%
  \def\do##1{%
    \edef\@@notesseries{\@firstoftwo##1}%
    \edef\@@notetype{\@secondoftwo##1}%
    \ifdefstring{\@@notetype}{critical}%
      {\csuse{#1append@Xnotes}{\@@notesseries}}%
      {\ifdefstring{\@@notetype}{familiar}%
        {\csuse{#1append@notesX}{\@@notesseries}}%
        {}%
      }%
  }%
  \expandafter\docsvlist\expandafter{#2}%
}%
```

```
%
```

\do@Xfeet

\do@Xfeet is the code extending \@makecol to cater to the extra critical feet.

```
\newcommand*{\do@Xfeet}{%
  \setbox\@outputbox \vbox{%
    \unvbox\@outputbox
    \@opXfeet}}
%
```

\@opXfeet
\append@Xnotes
\print@Xnotes

The extra critical feet to be added to the output. . A macro which appends critical notes to the output's routine, also adding vertical space before notes

```
\newcommand{\append@Xnotes}[1]{%
  \ifvoid\csuse{#1footins}\else%
      \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@#1}%
      \global\advance\skip\csuse{#1footins} by\csuse{Xafterrule@#1}%
      \print@Xnotes{#1}%
   \fi%
}%
%
```

The normal way to add one series, \print@Xnotes, is replaced by reledpar when using \Pages.

```
\newcommand\print@Xnotes[1]{%
  \xdef\@currentseries{#1}%
  \csuse{#1footstart}{#1}%
  \csuse{#1footgroup}{#1}%%
}%
%
```

We print all series of notes by looping on them. We check before printing them that they are not voided.

```
\newcommand*{\@opXfeet}{%
  \unless\ifnocritical@%
    \gdef\firstXseries@{}%
    \def\do##1{%
      \append@Xnotes{##1}%
      }%
    \dolistloop{\@series}%
  \fi%
}%
%
```

\l@ddodoreinxtrafeet

\l@ddodoreinxtrafeet is the code for catering for the extra footnotes within \@reinserts. We use the same category and ordering as in \l@ddoxtrafeet.

```
\newcommand*{\l@ddodoreinxtrafeet}{%
  \IfStrEq{familiar-critical}{\@fnpos}
```

```
    {\@doreinfeetX\X@doreinfeet}%
    {%
    \IfStrEq{critical-familiar}{\@fnpos}%
        {\X@doreinfeet\@doreinfeetX}%
        {\@doreinfeetX\X@doreinfeet}%
    }%
}

%
```

\X@doreinfeet

\X@doreinfeet is the code for catering for the extra critical footnotes within \@reinserts.

```
\newcommand*{\X@doreinfeet}{%
  \unless\ifnocritical@%
    \def\do##1{%
      \ifvoid\csuse{##1footins}\else%
        \insert\csuse{##1footins}{\unvbox\csuse{##1footins}}%
      \fi}%
    \dolistloop{\@series}
  \fi%
}

%
```

\print@notesX
\append@notesX
\do@feetX
\@doreinfeetX

We have to add all the new kinds of familiar footnotes to the output routine. A macro which appends the familiar footnotes of one series onto the output routine, also adding vertical skip before notes.

```
\newcommand{\append@notesX}[1]{%
  \ifvoid\csuse{footins#1}\else%
      \global\skip\csuse{footins#1}=\csuse{beforenotesX@#1}%
      \global\advance\skip\csuse{footins#1} by\csuse{afterruleX@#1}%
      \print@notesX{#1}%
  \fi%
}%
%
```

The normal way to print one series of notes. \print@Xnotes is replaced by reledpar when using \Pages.

```
\newcommand\print@notesX[1]{%
  \xdef\@currentseries{#1}%
  \csuse{footstart#1}{#1}%
  \csuse{footgroup#1}{#1}%
}%
%
```

We print all the series of notes by looping on them. We check before printing them that they are not voided.

```
\newcommand*{\do@feetX}{%
```

```
  \unless\ifnofamiliar@%
    \gdef\firstseriesX@{}%
    \setbox\@outputbox \vbox{%
      \unvbox\@outputbox%
      \def\do##1{%
        \append@notesX{##1}%
      }%
      \dolistloop{\@series}}%
  \fi%
}%

\newcommand{\@doreinfeetX}{%
  \unless\ifnofamiliar@%
    \def\do##1{%
      \ifvoid\csuse{footins##1}\else
        \insert%
          \csuse{footins##1}
          {\unvbox\csuse{footins##1}}%
       \fi%
    }%
    \dolistloop{\@series}%
  \fi%
}%

%
```

XXII.2 Patching standard output's commands

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.)

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

`memoir` is loaded so we use `memoir`'s built in hooks.

```
  \g@addto@macro{\m@mdoextrafeet}{\l@ddoxtrafeet}%
  \g@addto@macro{\m@mdodoreinextrafeet}{\l@ddodoreinxtrafeet}%
  }{%
%
```

`memoir` has not been loaded, so patch `\@makecol` and `\@reinserts`. If the `fancyhdr` package $<$ version 3.8 has been loaded, we patch the `\latex@makecol` command, because this package redefines the standard `\@makecol` in the preamble, to call `\latex@makecol` which have been `\let` to `\@makecol`. If this package is not loaded, we directly patch `\@makecol`. If the `fancyhdr` package $\geq$ version 3.8, we also directly patch `\@makecol`, because `fancyhdr` does its own patch `\AtBeginDocument`.

```
  \ifboolexpr{%
    test{\@ifpackageloaded{fancyhdr}}%
    and test {\ifdef{\latex@makecol}}%
  }{%
    \patchcmd%
      {\latex@makecol}%
      {\xdef\@freelist{\@freelist\@midlist}}%
      {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
      {}%
      {\led@error@fail@patch@@makecol}%
    }{%
    \patchcmd%
      {\@makecol}%
      {\xdef\@freelist{\@freelist\@midlist}}%
      {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
      {}%
      {\led@error@fail@patch@@makecol}%
    }%

  \patchcmd%
    {\@reinserts}%
    {\ifvbox}%
    {\l@ddodoreinxtrafeet\ifvbox}%
    {}%
    {\led@error@fail@patch@@reinserts}%
}

%
```

It turns out that \@doclearpage also needs modifying.

\if@led@nofoot We have to check if there are any leftover feet.

```
\newif\if@led@nofoot

%
```

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

If the memoir class is loaded we hook into its modified \@doclearpage.

\@mem@extranofeet

```
\g@addto@macro{\@mem@extranofeet}{%%
  \def\do#1{%
    \unless\ifnocritical@%
      \ifvoid\csuse{#1footins}\else\@mem@nofootfalse\fi%
    \fi%
    \unless\ifnofamiliar@%
      \ifvoid\csuse{footins#1}\else\@mem@nofootfalse\fi%
    \fi%
    }
  \dolistloop{\@series}%
  }%
}{%
%
```

As memoir is not loaded we have patch \@doclearpage.

\@led@testifnofoot
\@doclearpage

```
\newcommand*{\@led@testifnofoot}{%
  \@led@nofoottrue%
  \ifvoid\footins\else%
    \@led@nofootfalse%
  \fi%
  \def\do##1{%
    \unless\ifnocritical@%
      \ifvoid\csuse{##1footins}\else%
        \@led@nofootfalse%
      \fi%
    \fi%
    \unless\ifnofamiliar@%
      \ifvoid\csuse{footins##1}\else%
        \@led@nofootfalse%
      \fi%
    \fi%
  }%
  \dolistloop{\@series}%
}%

\pretocmd%
 {\@doclearpage}%
 {\@led@testifnofoot}%
 {}%
 {\led@error@fail@patch@@doclearpage}%

\patchcmd%
 {\@doclearpage}%
 {\ifvoid\footins}%
 {\if@led@nofoot}%
 {}%
 {\led@error@fail@patch@@doclearpage}%

}

%
```

XXIII Cross referencing

You can mark a place in the text using a command of the form \edlabel{⟨*foo*⟩}, and later refer to it using the label ⟨*foo*⟩ by typing \edpageref{⟨*foo*⟩}, or \lineref{⟨*foo*⟩} or \sublineref{⟨*foo*⟩} or \pstartref. These reference commands will produce, respectively, the page, line sub-line and pstart on which the \edlabel{⟨*foo*⟩} command occurred.

The reference macros warn you if a reference is made to an undefined label. If {⟨*foo*⟩} has been used as a label before, the \edlabel{⟨*foo*⟩} command will issue a complaint; subsequent \edpageref and \edlineref commands will refer to the latest occurrence of \edlabel{⟨*foo*⟩}.

\labelref@list Set up a new list, \labelref@list, to hold the page, line and sub-line numbers for each label.

```
\list@create{\labelref@list}
%
```

\zz@@@ A convenience macro to zero three labeling counters in one go.

```
\newcommand*{\zz@@@}{000|000|000}% Set three counters to zero in one go

%
```

\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.[33]

This version of the original edmac \label uses \@bsphack and \@esphack to eliminate extra space problems and also use the LaTeX write methods for the .aux file.

Jesse Billett[34] found that the original code could be off by several pages. This version, hopefully cures that, and also allows for non-arabic page numbering.

```
\newcommand*{\edlabel}[1]{%
  \ifl@dpairing\ifautopar%
    \strut%
  \fi\fi%
  \@bsphack%
  \ifboolexpr{bool{ledRcol} or bool{ledRcol@}}{%
    \ifXnote@%
     \protected@write\@auxout{}%
      {\string\l@dmake@labelsR\space\thepage|\l@dparsedstartline|\
l@dparsedstartsub||\the\c@pstartR|{#1}}%
      \ifdef{\hypertarget}%
        {\Hy@raisedlink{\hypertarget{#1}{}}}%
```

[33] The remaining macros in this section were kindly revised by Wayne Sullivan, who substantially improved their efficiency and flexibility.

[34] (jdb43@cam.ac.uk) via the ctt thread ‘ledmac cross referencing’, 25 August 2003.

```
        {}%
    \else%
      \write\linenum@outR{\string\@lab}%
      \ifx\labelref@listR\empty%
        \xdef\label@refs{\zz@@@}%
      \else%
        \gl@p\labelref@listR\to\label@refs%
      \fi%
      \ifvmode%
          \advancelabel@refs%
      \fi%
%
```

Use code from the kernel `\label` command to write the correct page number. Also define an hypertarget if `hyperref` package is loaded.

```
      \protected@write\@auxout{}%
        {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstartR
|{#1}}%
      \ifdef{\hypertarget}%
        {\Hy@raisedlink{\hypertarget{#1}{}}}%
        {}%
    \fi%
  }{%
    \ifXnote@%
       \ifl@dpairing%pstart or pstartL?
         \protected@write\@auxout{}%
          {\string\l@dmake@labels\space\thepage|\l@dparsedstartline|\
l@dparsedstartsub||\the\c@pstartL|{#1}}%
          \ifdef{\hypertarget}%
            {\Hy@raisedlink{\hypertarget{#1}{}}}%
            {}%
        \else%
          \protected@write\@auxout{}%
          {\string\l@dmake@labels\space\thepage|\l@dparsedstartline|\
l@dparsedstartsub||\the\c@pstart|{#1}}%
          \ifdef{\hypertarget}%
            {\Hy@raisedlink{\hypertarget{#1}{}}}%
            {}%
        \fi%
    \else%
      \write\linenum@out{\string\@lab}%
      \ifx\labelref@list\empty%
        \xdef\label@refs{\zz@@@}%
      \else%
        \gl@p\labelref@list\to\label@refs%
      \fi%
      \ifvmode%
          \advancelabel@refs%
      \fi%
      \ifl@dpairing%Pstart or PstartL?
```

```
        \protected@write\@auxout{}%
          {\string\l@dmake@labels\space\thepage|\label@refs|\the\c@pstartL|{#1}}%
          \ifdef{\hypertarget}%
            {\Hy@raisedlink{\hypertarget{#1}{}}}%
            {}%
       \else%
         \protected@write\@auxout{}%
          {\string\l@dmake@labels\space\thepage|\label@refs|\the\c@pstart|{#1}}%
          \ifdef{\hypertarget}%
            {\Hy@raisedlink{\hypertarget{#1}{}}}%
            {}%
       \fi%
    \fi%
  }%
\@esphack}%

%
```

\advancelabel@refs
\labelrefsparseline
\labelrefsparsesubline

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 \advancelabel@refs command.

```
\newcounter{line}%
\newcounter{subline}%
\newcounter{absline}%
\newcommand{\advancelabel@refs}{%
    \setcounter{line}{\expandafter\labelrefsparseline\label@refs}%
    \stepcounter{line}%
    \setcounter{absline}{\expandafter\labelrefsparseabsline\label@refs}%
    \stepcounter{absline}%
    \ifsublines@%
        \setcounter{subline}{\expandafter\labelrefsparsesubline\label@refs}%
        \stepcounter{subline}{1}%
        \def\label@refs{\theline|\thesubline|\theabsline}%
    \else%
        \def\label@refs{\theline|0|\theabsline}%
    \fi%
}
\def\labelrefsparseline#1|#2|#3{#1}%
\def\labelrefsparsesubline#1|#2|#3{#2}%
\def\labelrefsparseabsline#1|#2|#3{#3}%
%
```

\l@dmake@labels

The \l@dmake@labels macro gets executed when the labels file is read. For each label it defines a macro, whose name is made up partly from the label you supplied, that contains the page, line and sub-line numbers. But first it checks to see whether the label has already been used (and complains if it has).

The initial use of \newcommand is to catch if \l@dmake@labels has been previously defined (by a class or package).

#1 page number, #2 line number, #3 sub-line number, #4 absolute line number, #5 pstart number, #6 label.

```
\newcommand*{\l@dmake@labels}{}
\def\l@dmake@labels#1|#2|#3|#4|#5|#6{%
  \expandafter\ifx\csname the@label\csuse{XR@prefix}#6\endcsname%
    \relax%
  \else%
    \led@warn@DuplicateLabel{\csuse{XR@prefix}#6}%
  \fi
  \global\providetoggle{label@#6@ledRcol}%False is the default value of this toggle, which tell us if a label is linked to a right or a left side
  \expandafter\gdef\csname the@label\csuse{XR@prefix}#6\endcsname{#1|#2|#3|#4|#5|\relax}%
  \ignorespaces}

%
```

LaTeX 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.

```
\AtBeginDocument{%
  \def\l@dmake@labels#1|#2|#3|#4|#5|#6{}%
}

%
```

\@lab The \@lab command, which appears in the \linenum@out file, appends the current value of page, line, sub-line, and absolute 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.

LaTeX 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.

```

\newcommand*{\@lab}{%
  \ifledRcol
    \xright@appenditem{\linenumr@p{\line@numR}|%
      \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi|\the\absline@numR}%
      \to\labelref@listR
  \else
    \xright@appenditem{\linenumr@p{\line@num}|%
```

```
      \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi|\the\absline@num}%
      \to\labelref@list
  \fi}
%
```

\applabel

`\applabel`, if called in `\edtext` will insert automatically both a start and an end label for the current edtext lines.

```
\newcommand*{\applabel}[1]{%
  \if@edtext@secondarg@%
%
```

Label should not be already defined.

```
      \ifcsundef{the@label#1}{%
        \csdef{the@label#1}{applabel}%
       }%
       {%
         \led@warn@DuplicateLabel{#1 (applabel)}%
       }%
%
```

Parse the `\edtext` line numbers.

```
     \expandafter\l@dp@rsefootspec\l@d@nums|%
%
```

Use the LaTeX standard hack for label.

```
     \@bsphack%
%
```

And now, write the data in the auxiliary file.

```
      \ifledRcol%
        \protected@write\@auxout{}%
          {\string\l@dmake@labelsR\space\l@dparsedstartpage|\l@dparsedstartline|\l@dparsedstartsub||\the\c@pstartR|{#1:start}}%
        \ifdef{\hypertarget}%
          {\Hy@raisedlink{\hypertarget{#1:start}{}}}%
          {}%
        \protected@write\@auxout{}%
          {\string\l@dmake@labelsR\space\l@dparsedendpage|\l@dparsedendline||\l@dparsedendsub|\the\c@pstartR|{#1:end}}%
      \else%
        \ifl@dpairing%pstart or pstartL?
          \protected@write\@auxout{}%
            {\string\l@dmake@labels\space\l@dparsedstartpage|\l@dparsedstartline|\l@dparsedstartsub||\the\c@pstartL|{#1:start}}%
          \ifdef{\hypertarget}%
            {\Hy@raisedlink{\hypertarget{#1:start}{}}}%
            {}%
          \protected@write\@auxout{}%
            {\string\l@dmake@labels\space\l@dparsedendpage|\l@dparsedendline|\l@dparsedendsub||\the\c@pstartL|{#1:end}}%
         \else%
            \protected@write\@auxout{}%
              {\string\l@dmake@labels\space\l@dparsedstartpage|\l@dparsedstartline|\l@dparsedstartsub||\the\c@pstart|{#1:start}}%
            \ifdef{\hypertarget}%
              {\Hy@raisedlink{\hypertarget{#1:start}{}}}%
              {}%
            \protected@write\@auxout{}%
              {\string\l@dmake@labels\space\l@dparsedendpage|\l@dparsedendline|\l@dparsedendsub||\the\c@pstart|{#1:end}}%
         \fi%
      \fi%
%
```

Use the LaTeX standard hack for label.

```
      \@esphack%
%
```

Warning if \applabel is called outside of \edtext.

```
  \else%
    \led@warn@AppLabelOutSecondArgEdtext{#1}%
  \fi%
%
```

End of \applabel

```
}%
%
```

\edlabelS
\edlabelE
\edlabelSE

\edlabelS and \edlabelE are just used to mark the beginning and the end of a passage.

```
\newcommand{\edlabelS}[1]{%
  \edlabel{#1:start}%
}
\newcommand{\edlabelE}[1]{%
  \edlabel{#1:end}%
}
\newcommand{\edlabelSE}[1]{%
  \edlabelS{#1}%
  \edlabelE{#1}%
}
%
```

\wrap@edcrossref

\wrap@edcrossref is called around all reledmac crossref commands, except those which start with x. It adds the hyperlink.

```
\newrobustcmd{\wrap@edcrossref}[2]{%
  \ifdef{\hyperlink}%
    {\hyperlink{#1}{#2}}%
    {#2}%
}
%
```

\edpageref
\xpageref

If the specified label exists, \edpageref gives its page number.

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.

LaTeX already defines a \pageref, so changing the name to \edpageref.

```
\newcommand*{\edpageref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{1}{#1}}}
\newcommand*{\xpageref}[1]{\l@dgetref@num{1}{#1}}

%
```

\edlineref
\xlineref

If the specified label exists, \lineref gives its line number.

```
\newcommand*{\edlineref}[1]{%
  \l@dref@undefined{#1}%
  \wrap@edcrossref{#1}{%
    \providetoggle{label@#1@ledRcol}%Required for the first run, when the label has not yet been parsed on the .aux file
    \iftoggle{label@#1@ledRcol}%
        {\linenumrepR{\l@dgetref@num{2}{#1}}}%
        {\linenumrep{\l@dgetref@num{2}{#1}}}%
    \xflagref{#1}%
  }%
}%
\newcommand*{\xlineref}[1]{\l@dgetref@num{2}{#1}}%

%
```

\sublineref
\xsublineref

If the specified label exists, \sublineref gives its sub-line number.

```
\newcommand*{\sublineref}[1]{%
  \l@dref@undefined{#1}%
  \wrap@edcrossref{#1}{%
    \providetoggle{label@#1@ledRcol}%Required for the first run, when the label has not yet been parsed on the .aux file
    \iftoggle{label@#1@ledRcol}%
      {\sublinenumrepR{\l@dgetref@num{3}{#1}}}%
      {\sublinenumrep{\l@dgetref@num{3}{#1}}}%
  }%
}%
\newcommand*{\xsublineref}[1]{\l@dgetref@num{3}{#1}}

%
```

\xabslineref

If the specified label exists, \xabslineref gives its absolute line number. That is used usually only by some reledmac internal macros.

```
\newcommand*{\xabslineref}[1]{\l@dgetref@num{4}{#1}}%
%
```

\pstarteref
\xpstartref

If the specified label exists, \pstartref gives its pstart number.

```
\newcommand*{\pstartref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{5}{#1}}}%
\newcommand*{\xpstartref}[1]{\l@dgetref@num{5}{#1}}%

%
```

\xflagref

\xflagref finds the side flag of any ref defined with \edlabel.

```
\newcommand*{\xflagref}[1]{\l@dgetref@num{6}{#1}}%
%
```

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.

```
\newcommand*{\l@dref@undefined}[1]{%
  \expandafter\ifx\csname the@label#1\endcsname\relax
    \led@warn@RefUndefined{#1}%
  \fi}

%
```

\l@dgetref@num

Next, \l@dgetref@num fetches the number we want. It has two arguments: the first is simply a digit, specifying whether to fetch a page (1), line (2), sub-line (3), (4) pstart number or (5) side flag. (This switching is done by calling \l@dlabel@parse.) The second argument is the label-macro, which because of the \@lab macro above is defined to be a string of the type 123|456|789.

```
\newcommand*{\l@dgetref@num}[2]{%
  \expandafter
  \ifx\csname the@label#2\endcsname \relax
```

```
    000%
  \else
    \expandafter\expandafter\expandafter
    \l@dlabel@parse\csname the@label#2\endcsname|#1%
  \fi}

%
```

\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, 4 or 5) which defines which of the earlier six numbers to pick out. (It was earlier given as the first argument of \l@dgetref@num.)

```
\newcommand*{\l@dlabel@parse}{}
\def\l@dlabel@parse#1|#2|#3|#4|#5|#6|#7{%
  \ifcase #7%
  \or #1%
  \or #2%
  \or #3%
  \or #4%
  \or #5%
  \or #6%
  \fi}
%
```

\xxref

The \xxref command takes two arguments, both of which are labels, e.g., \xxref{mouse}{elephant}. It first does some checking to make sure that the labels do exist (if one does not, those numbers are set to zero). Then it calls \linenum and sets the beginning page, line, and sub-line numbers to those of the place where \label{mouse} was placed, and the ending numbers to those at {elephant}. The point of this is to be able to manufacture footnote line references to passages which cannot be specified in the normal way as the first argument to \edtext for one reason or another. Using \xxref in the second argument of \edtext lets you set things up at least semi-automatically.

```
\newcommand*{\xxref}[2]{%
  {%
   \expandafter\ifx\csname the@label#1\endcsname \relax%
     \expandafter\let\csname the@@label#1\endcsname\zz@@@%
    \else%
      \expandafter\def\csname the@@label#1\endcsname{\l@dgetref@num
{1}{#1}|\l@dgetref@num{2}{#1}|\l@dgetref@num{3}{#1}}%
   \fi%
   \expandafter\ifx\csname the@label#2\endcsname \relax%
     \expandafter\let\csname the@@label#2\endcsname\zz@@@%
   \else%
```

```
      \expandafter\def\csname the@@label#2\endcsname{\l@dgetref@num{1}{#2}|\l@dgetref@num{2}{#2}|\l@dgetref@num{3}{#2}}%
   \fi%
   \letcs{\@tempa}{the@@label#1}%
   \letcs{\@tempb}{the@@label#2}%
   \global\appto\@beforeinsertofthisedtext{\def\@this@crossref@start{#1}}%
   \global\appto\@beforeinsertofthisedtext{\def\@this@crossref@end{#2}}%
   \linenum{\@tempa|%
    \@tempb}}}%

%
```

\appref
\apprefwithpage
\SEref
\SErefwithpage
\SErefwithpage

\appref, \SEref, \apprefwithpage, \SErefwithpage and \SEonlypage print crossref to some start / end lines defined by specific commands. It prints the lines as they should be printed in the apparatus (critical notes for not suffixed versions, endnotes for suffixed versions).

Here we define hooks similar to some those related to critical footnotes or endnotes.

So, first declare the default value of the hooks for the pseudo-series. Also declare the internal toggle which are switch by reledmac.

```
\def\Xtwolines@appref{}%
\def\Xtwolines@SEref{}%

\def\Xmorethantwolines@appref{}%
\def\Xmorethantwolines@SEref{}%

\def\Xlinerangeseparator@appref{\endashchar}%
\def\Xlinerangeseparator@SEref{\endashchar}%

\def\Xsublinesep@appref{\fullstop}%
\def\Xsublinesep@SEref{\fullstop}%

\def\Xpagelinesep@appref{\fullstop}%
\def\Xpagelinesep@SEref{\fullstop}%


\newtoggle{Xtwolinesbutnotmore@appref}%
\newtoggle{Xtwolinesbutnotmore@SEref}%

\newtoggle{Xtwolinesonlyinsamepage@appref}%

\newtoggle{Xtwolinesonlyinsamepage@SEref}%

\newtoggle{Xlineflag@appref}%
\toggletrue{Xlineflag@appref}%Here exception
\newtoggle{Xlineflag@SEref}%
\toggletrue{Xlineflag@SEref}%%Here exception

\def\Xendtwolines@apprefwithpage{}%
```

```
\def\Xendtwolines@SErefwithpage{}%

\def\Xendmorethantwolines@apprefwithpage{}%
\def\Xendmorethantwolines@SErefwithpage{}%

\def\Xendlinerangeseparator@apprefwithpage{\endashchar}
\def\Xendlinerangeseparator@SErefwithpage{\endashchar}
\def\Xendlinerangeseparator@SErefonlypage{\endashchar}

\def\Xendbeforepagenumber@apprefwithpage{p.}%
\def\Xendbeforepagenumber@SErefwithpage{p.}%
\def\Xendbeforepagenumber@SEonlypage{p.}%

\def\Xendafterpagenumber@apprefwithpage{) }%
\def\Xendafterpagenumber@SErefwithpage{) }%


\def\Xendlineprefixsingle@apprefwithpage{}%
\def\Xendlineprefixsingle@SErefwithpage{}%

\def\Xendlineprefixmore@apprefwithpage{}%
\def\Xendlineprefixmore@SErefwithpage{}%

\newtoggle{Xendtwolinesbutnotmore@apprefwithpage}%
\newtoggle{Xendtwolinesbutnotmore@SErefwithpage}%

\def\Xendsublinesep@apprefwithpage{\fullstop}%
\def\Xendsublinesep@SErefwithpage{\fullstop}%

\newtoggle{Xendtwolinesonlyinsamepage@apprefwithpage}%
\newtoggle{Xendtwolinesonlyinsamepage@SErefwithpage}%

\newtoggle{Xendlineflag@apprefwithpage}
\toggletrue{Xendlineflag@apprefwithpage}%Here, exception
\newtoggle{Xendlineflag@SErefwithpage}
\toggletrue{Xendlineflag@SErefwithpage}%Here, exception

%
```

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.

```

\gdef\Xboxstartlinenum@appref{0pt}
\gdef\Xboxstartlinenum@SEref{0pt}

\gdef\Xboxendlinenum@appref{0pt}
\gdef\Xboxendlinenum@SEref{0pt}

\gdef\Xendboxstartlinenum@apprefwithpage{0pt}
```

```
\gdef\Xendboxstartlinenum@SErefwithpage{0pt}

\gdef\Xendboxendlinenum@apprefwithpage{0pt}
\gdef\Xendboxendlinenum@SErefwithpage{0pt}

\newtoggle{Xendpagenumberonlyfirst@apprefwithpage}
\newtoggle{Xendpagenumberonlyfirst@SErefwithpage}

\newtoggle{Xendpagenumberonlyfirstifsingle@apprefwithpage}
\newtoggle{Xendpagenumberonlyfirstifsingle@SErefwithpage}

\newtoggle{Xendpagenumberonlyfirstintwo@apprefwithpage}
\newtoggle{Xendpagenumberonlyfirstintwo@SErefwithpage}

\gdef\Xendsympagenum@apprefwithpage{}
\gdef\Xendsympagenum@SErefwithpage{}

\gdef\Xendinplaceofpagenumber@apprefwithpage{}
\gdef\Xendinplaceofpagenumber@SErefwithpage{}

%
```

Now, declare the default values of `\@apprefprefixsingle` and `\@apprefprefixmore`, `\@SErefprefix`, `\@SErefprefixmore` and the commands which defines them.

```
\newcommand\@apprefprefixsingle{}%
\newcommand\@SErefprefixsingle{}%

\newcommand\@apprefprefixmore{}%
\newcommand\@SErefprefixmore{}%

\newcommand{\setapprefprefixsingle}[1]{%
  \gdef\@apprefprefixsingle{#1}%
}
\newcommand{\setSErefprefixsingle}[1]{%
  \gdef\@SErefprefixsingle{#1}%
}

\newcommand{\setapprefprefixmore}[1]{%
  \gdef\@apprefprefixmore{#1}%
}
\newcommand{\setSErefprefixmore}[1]{%
  \gdef\@SErefprefixmore{#1}%
}

%
```

And not `\setSErefonlypageprefixsingle` and `\setSErefonlypageprefixmore`.

```
\newcommand{\setSErefonlypageprefixsingle}[1]{%
  \gdef\SErefonlypage@prefixsingle{#1}%
}%
```

```
\newcommand{\setSErefonlypageprefixmore}[1]{%
  \gdef\SErefonlypage@prefixmore{#1}%
}%
%
```

And now, the main commands: \appref, \apprefwithpage, \SEref and \SErefwithpage. These commands call \refformated@ and \refformatedwithpage, which calls \printlines and \printendlines . That is why we have previously declared all hooks values tested inside these last commands.

```

\newcommandx{\appref}[2][1,usedefault]{\refformated@{#1}{#2}{appref}}
\newcommandx{\SEref}[2][1,usedefault]{\refformated@{#1}{#2}{SEref}}

\newcommandx{\apprefwithpage}[2][1,usedefault]{\refformatedwithpage@
{#1}{#2}{appref}}
\newcommandx{\SErefwithpage}[2][1,usedefault]{\refformatedwithpage@
{#1}{#2}{SEref}}
\newcommandx{\SErefonlypage}[2][1,usedefault]{\refformatedonlypage@
{#1}{#2}{SEref}}


\newcommand{\refformated@}[3]{%
  \def\do##1{%
    \setkeys[mac]{truefootnoteoption}{##1}%
  }%
 \notblank{#1}{\docsvlist{#1}}{}%
 \xdef\@currentseries{#3}%
 \ifcsempty{@#3prefixmore}%
   {\@apprefprefixsingle}%
   {%
     \IfEq{\xlineref{#2:start}}{\xlineref{#2:end}}%
       {\csuse{@#3prefixsingle}}%
       {\csuse{@#3prefixmore}}%
   }%
 \ifboolexpr{%
   test{\ifcsundef{the@label#2:start}}%
   or test{\ifcsundef{the@label#2:end}}%
  }%
   {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}%
   {%
    \def\@this@crossref@start{#2:start}%
    \def\@this@crossref@end{#2:end}%
    \printlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:
start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}|\relax|\
xflagref{#2:start}|%
    \undef\@this@crossref@end%
    \undef\@this@crossref@start%
   }%
  \def\do##1{%
    \setkeys[mac]{falsefootnoteoption}{##1}%
  }%
 \notblank{#1}{\docsvlist{#1}}{}%
}%

\newcommand{\refformatedwithpage@}[3]{%
  \def\do##1{%
    \setkeys[mac]{truefootnoteoption}{##1}%
  }%
 \notblank{#1}{\docsvlist{#1}}{}%
 \xdef\@currentseries{#3withpage}%
 \ifboolexpr{%
   test{\ifcsundef{the@label#2:start}}%
   or test{\ifcsundef{the@label#2:end}}%
  }%
   {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}%
   {%
   \def\@this@crossref@start{#2:start}%
   \def\@this@crossref@end{#2:end}%
   \printendlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}|\relax|\xflagref{#2:start}|%
    \undef\@this@crossref@end%
    \undef\@this@crossref@start%
   }%
  \def\do##1{%
    \setkeys[mac]{falsefootnoteoption}{##1}%
  }%
 \notblank{#1}{\docsvlist{#1}}{}%
}%

\newcommand{\refformatedonlypage@}[3]{%
  \def\do##1{%
    \setkeys[mac]{truefootnoteoption}{##1}%
  }%
 \notblank{#1}{\docsvlist{#1}}{}%
 \xdef\@currentseries{#3onlypage}%
 \ifboolexpr{%
   test{\ifcsundef{the@label#2:start}}%
   or test{\ifcsundef{the@label#2:end}}%
  }%
   {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}%
   {\ifnumequal{\xpageref{#2:end}}{\xpageref{#2:start}}%
       {%
       \ifcsvoid{#3onlypage@prefixsingle}%
         {}%
         {\csletcs{Xendbeforepagenumber@#3onlypage}{#3onlypage@prefixsingle}}%
       \printnpnum{%
         \wrap@edcrossref{#2:start}{\xpageref{#2:start}}%
         }%
       }%
     {%
      \ifcsvoid{#3onlypage@prefixmore}%
        {}%
        {\csletcs{Xendbeforepagenumber@#3onlypage}{#3onlypage@prefixmore}}%
      \ifdefined\linerangesep@%
        \printnpnum{%
          \wrap@edcrossref{#2:start}{\xpageref{#2:start}}%
          \linerangesep@%
          \wrap@edcrossref{#2:end}{\xpageref{#2:end}}%
        }%
      \else%
        \printnpnum{%
          \wrap@edcrossref{#2:start}{\xpageref{#2:start}}%
          \csuse{Xendlinerangeseparator@\@currentseries}%
          \wrap@edcrossref{#2:end}{\xpageref{#2:end}}%
        }%
      \fi%
     }%
   }%
  \def\do##1{%
    \setkeys[mac]{falsefootnoteoption}{##1}%
  }%
 \notblank{#1}{\docsvlist{#1}}{}%
}%
%
```

\edmakelabel

Sometimes the \edlabel command cannot be used to specify exactly the page and line desired; you can use the \edmakelabel macro make your own label. For example, if you insert \edmakelabel{elephant}{10|25|0} you will have created a new label, and a later call to \edpageref{elephant} would print '10' and \lineref{elephant} would print '25'. The sub-line number here is zero. \edmakelabel takes a label, followed by a page and a line number(s) as arguments. LaTeX defines a \makelabel macro which is used in lists. Peter Wilson has changed the name to \edmakelabel.

```
\newcommand*{\edmakelabel}[2]{\expandafter\xdef\csname the@label#1\endcsname{#2}}

%
```

(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. 132 and V.9 p. 99), since \xxref makes a call to \linenum in order to do its work.)

XXIII.1 Compatibility with xref

Here, we provide compatibility with the xref to enable reledmac's cross-referencing to external documents. We assume that the user loads xref *before* reledmac, but uses \externaldocument *after* loading reledmac.

\XR@test

First, we patch the xr macro \XR@test, which is called on every line of the external .aux file, in order to also call macros specific to reledmac.

```
\pretocmd{\XR@test}%
  {\XR@test@mac+++#1#2#3#4+++}%
  {}%
  {}%
%
```

\XR@test@mac

The \XR@test@mac takes the full content of a line of the external .aux files, with the three final dots added by xr.

```
\long\def\XR@test@mac+++#1+++{\XR@test@mac@test#1}
%
```

\XR@test@mac@test

And finally, \XR@test@mac@test does the job. This code is based on the \XR@test macro of the xr package. However, not that the \XR@prefix is not called here, but it is integrated directly in \l@dmake@labels and \l@dmake@labelsR.

```
\long\def\XR@test@mac@test#1#2...{%The triple dots (NOT \ldots) are because
 of the line 22 of xr.sty  v5.02 1994/05/28
  \ifx#1\l@dmake@labels%
   \l@dmake@labels#2%
  \else
    \ifx#1\l@dmake@labelsR%
      \l@dmake@labelsR #2%
    \fi%
  \fi%
}%
%
```

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.

```
\pretocmd{\@xympar}%
  {\ifnumberedpar@
    \led@warn@NoMarginpars
    \@esphack
  \else}%
  {}%
  {}%

\apptocmd{\@xympar}%
  {\fi}%
  {}
  {}

%
```

We provide side notes as replacement for \marginpar in numbered text.

\sidenote@margin
\sidenotemargin
\l@dgetsidenote@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

```
\newcount\sidenote@margin
\newcommand*{\sidenotemargin}[1]{{%
  \l@dgetsidenote@margin{#1}%
  \ifnum\@l@dtempcntb>\m@ne
    \ifledRcol
      \global\sidenote@marginR=\@l@dtempcntb
    \else
      \global\sidenote@margin=\@l@dtempcntb
    \fi
  \fi}}
\newcommand*{\l@dgetsidenote@margin}[1]{%
  \def\@tempa{#1}\def\@tempb{left}%
  \ifx\@tempa\@tempb
    \@l@dtempcntb \z@
  \else
    \def\@tempb{right}%
    \ifx\@tempa\@tempb
      \@l@dtempcntb \@ne
    \else
      \def\@tempb{outer}%
      \ifx\@tempa\@tempb
        \@l@dtempcntb \tw@
      \else
        \def\@tempb{inner}%
        \ifx\@tempa\@tempb
          \@l@dtempcntb \thr@@
        \else
          \led@warn@BadSidenotemargin
          \@l@dtempcntb \m@ne
        \fi
      \fi
    \fi
  \fi}
\sidenotemargin{right}

%
```

\l@dlp@rbox
\l@drp@rbox

We need two boxes to store sidenote texts.

```
\newbox\l@dlp@rbox
\newbox\l@drp@rbox

%
```

\ledlsnotewidth
\ledrsnotewidth
\ledlsnotesep
\ledrsnotesep
\ledlsnotefontsetup
\ledrsnotefontsetup

These specify the width of the left/right boxes (initialised to `\marginparwidth`), their distance from the text (initialised to `\linenumsep`), and the fonts used.

```
\newdimen\ledlsnotewidth \ledlsnotewidth=\marginparwidth
\newdimen\ledrsnotewidth \ledrsnotewidth=\marginparwidth
\newdimen\ledlsnotesep \ledlsnotesep=\linenumsep
\newdimen\ledrsnotesep \ledrsnotesep=\linenumsep
\newcommand*{\ledlsnotefontsetup}{\raggedleft\footnotesize}
\newcommand*{\ledrsnotefontsetup}{\raggedright\footnotesize}

%
```

\ledleftnote
\ledrightnote
\ledinnernote
\ledouternote
\ledsidenote

`\ledleftnote`, `\ledrightnote`, `\ledinnernote`, `\ledouternote` are the user commands for left, right, inner and outer sidenotes. The two last one are just alias for the two first one, depending of the page number. `\ledsidenote{`⟨*text*⟩`}` is the command for a moveable sidenote.

```
\newcommand*{\ledleftnote}[1]{\edtext{}{\l@dlsnote{#1}}}
\newcommand*{\ledrightnote}[1]{\edtext{}{\l@drsnote{#1}}}
\newcommand*{\ledsidenote}[1]{\edtext{}{\l@dcsnote{#1}}}%
\newcommand*{\ledinnernote}[1]{\edtext{}{\l@disnote{#1}}}%
\newcommand*{\ledouternote}[1]{\edtext{}{\l@dosnote{#1}}}%
%
```

\l@dlsnote
\l@drsnote
\l@dcsnote
\l@desnote
\l@disnote

. The 'footnotes' for left, right, and moveable sidenotes. The whole scheme is reminiscent of the critical footnotes code.

```
\newif\ifrightnoteup
  \rightnoteuptrue

\newcommand*{\l@dlsnote}[1]{%
  \begingroup%
  \newcommand{\content}{#1}%
  \ifnumberedpar@
    \ifledRcol%
      \xright@appenditem{\noexpand\vl@dlsnote{\expandonce\content}}%
```

```
                         \to\inserts@listR
      \global\advance\insert@countR \@ne%
    \else%
      \xright@appenditem{\noexpand\vl@dlsnote{\expandonce\content}}%
                         \to\inserts@list
      \global\advance\insert@count \@ne%
    \fi
  \fi%
  \ignorespaces%
  \endgroup%
}%

\newcommand*{\l@drsnote}[1]{%
  \begingroup%
  \newcommand{\content}{#1}%
  \ifnumberedpar@
    \ifledRcol%
      \xright@appenditem{\noexpand\vl@drsnote{\expandonce\content}}%
                         \to\inserts@listR
       \global\advance\insert@countR \@ne%
    \else%
      \xright@appenditem{\noexpand\vl@drsnote{\expandonce\content}}%
                         \to\inserts@list
      \global\advance\insert@count \@ne%
    \fi
  \fi\ignorespaces%
  \endgroup%
}%

\newcommand*{\l@dcsnote}[1]{%
  \begingroup%
  \newcommand{\content}{#1}%
  \ifnumberedpar@
    \ifledRcol%
      \xright@appenditem{\noexpand\vl@dcsnote{\expandonce\content}}%
                         \to\inserts@listR
       \global\advance\insert@countR \@ne%
    \else%
      \xright@appenditem{\noexpand\vl@dcsnote{\expandonce\content}}%
                         \to\inserts@list
      \global\advance\insert@count \@ne%
    \fi
  \fi\ignorespaces%
  \endgroup%
}%

\newcommand*{\l@disnote}[1]{%
  \begingroup%
  \newcommand{\content}{#1}%
  \ifnumberedpar@%
```

```
    \ifledRcol%
      \xright@appenditem{\noexpand\vl@disnote{\expandonce\content}}%
                         \to\inserts@listR%
       \global\advance\insert@countR \@ne%
    \else%
      \xright@appenditem{\noexpand\vl@disnote{\expandonce\content}}%
                         \to\inserts@list%
      \global\advance\insert@count \@ne%
    \fi%
  \fi\ignorespaces%
  \endgroup%
}%

\newcommand*{\l@dosnote}[1]{%
  \begingroup%
  \newcommand{\content}{#1}%
  \ifnumberedpar@%
    \ifledRcol%
      \xright@appenditem{\noexpand\vl@dosnote{\expandonce\content}}%
                         \to\inserts@listR%
       \global\advance\insert@countR \@ne%
    \else%
      \xright@appenditem{\noexpand\vl@dosnote{\expandonce\content}}%
                         \to\inserts@list%
      \global\advance\insert@count \@ne%
    \fi%
  \fi\ignorespaces%
  \endgroup%
}%

%
```

\vl@dlsnote
\vl@drsnote
\vl@dcsnote
\vl@disnote
\vl@dosnote

Put the left/right text into boxes, but just save the moveable text. \l@dcsnotetext, \l@dcsnotetext@l and \l@dcsnotetext@r are etoolbox's lists which will store the content of side notes. We store the content in lists, because we need to loop later on them, in case many sidenote co-exist for the same line. That is there some special test to do, in order to:

- Store the content of \ledsidenote to \l@dcsnotetext in any cases.
- Store the content of \rightsidenote to:
  - \l@dcsnotetext if \ledsidenote is to be put on right.
  - \l@dcsnotetext@r if \ledsidenote is to be put on left.
- Store the content of \leftsidenote to:
  - \l@dcsnotetext if \ledsidenote is to be put on left.
  - \l@dcsnotetext@l if \ledsidenote is to be put on right.

\vl@disnote and \vl@dosnote just call \vl@dlsnote or \vl@drsnote, depending of the page.

```
\newcommand*{\vl@dlsnote}[1]{%
  \ifledRcol@%
    \@l@dtempcntb=\sidenote@marginR%
    \ifnum\@l@dtempcntb>\@ne%
      \advance\@l@dtempcntb by\page@numR%
    \fi%
  \else%
    \@l@dtempcntb=\sidenote@margin%
    \ifnum\@l@dtempcntb>\@ne%
      \advance\@l@dtempcntb by\page@num%
    \fi%
  \fi%
  \ifodd\@l@dtempcntb%
    \listgadd{\l@dcsnotetext@l}{#1}%
  \else%
    \listgadd{\l@dcsnotetext}{#1}%
  \fi
}
\newcommand*{\vl@drsnote}[1]{%
  \ifledRcol@%
    \@l@dtempcntb=\sidenote@marginR%
    \ifnum\@l@dtempcntb>\@ne%
      \advance\@l@dtempcntb by\page@numR%
    \fi%
  \else%
    \@l@dtempcntb=\sidenote@margin%
    \ifnum\@l@dtempcntb>\@ne%
      \advance\@l@dtempcntb by\page@num%
    \fi%
  \fi%
  \ifodd\@l@dtempcntb%
    \listgadd{\l@dcsnotetext}{#1}%
  \else%
    \listgadd{\l@dcsnotetext@r}{#1}%
  \fi%
}
\newcommand*{\vl@dcsnote}[1]{\listgadd{\l@dcsnotetext}{#1}}

\newcommand{\vl@disnote}[1]{%
  \ifledRcol@%
    \@tempcnta=\page@numR%
  \else%
    \@tempcnta=\page@num%
  \fi%
  \ifodd\@tempcnta% ODD => right page => inner side =  left side
    \vl@dlsnote{#1}%
  \else%
```

```
    \vl@drsnote{#1}%
  \fi%
}%

\newcommand{\vl@dosnote}[1]{%
  \ifledRcol@%
    \@tempcnta=\page@numR%
  \else%
    \@tempcnta=\page@num%
  \fi%
  \ifodd\@tempcnta% ODD => right page => outer side = right side
    \vl@drsnote{#1}%
  \else%
    \vl@dlsnote{#1}%
  \fi%
}%

%
```

\setl@dlp@rbox
\setl@drpr@box

\setl@dlprbox{⟨*lednums*⟩}{⟨*tag*⟩}{⟨*text*⟩} puts ⟨*text*⟩ into the \l@dlp@rbox box. And similarly for the right side box. It is these boxes that finally get displayed in the margins.

```
\newcommand*{\setl@dlp@rbox}[1]{%
   \begingroup%
     \parindent\z@\hsize=\ledlsnotewidth%
     \ledlsnotefontsetup%We kept it outside of the vbox, because can affect the ragging
     \global\setbox\l@dlp@rbox%
     \ifleftnoteup%
       =\vbox to\z@{{\ledlsnotefontsetup\vss #1}}%We put \ledlsnotefontsetup inside footnote because required for color command. Note the {} to keep setting local.
     \else%
       =\vbox to 0.70\baselineskip{{\ledlsnotefontsetup\strut#1\vss}}%
     \fi%
   \endgroup%
}

\newcommand*{\setl@drp@rbox}[1]{%
  \begingroup%
     \parindent\z@\hsize=\ledrsnotewidth%
     \ledrsnotefontsetup%We kept it outside of the vbox, because can affect the ragging
     \global\setbox\l@drp@rbox%
     \ifrightnoteup%
       =\vbox to\z@{{\ledrsnotefontsetup\vss#1}}%We put \ledrsnotefontsetup inside footnote because required for color command. Note the {} to keep setting local.
     \else%
```

```
       =\vbox to0.7\baselineskip{{\ledrsnotefontsetup\strut#1\vss}}%
     \fi%
  \endgroup%
}%
\newif\ifleftnoteup
  \leftnoteuptrue
%
```

\@sidenotesep This macro is used to separate sidenotes of the same line.

```
\newcommand{\setsidenotesep}[1]{\gdef\@sidenotesep{#1}}
\newcommand{\@sidenotesep}{, }
%
```

\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.

```
\newcommand*{\affixside@note}{%
    \prepare@edindex@fornote{\the\page@num|\the\line@num|\the\subline@num|\
the\page@num|\the\line@num|\the\subline@num|}%
    \def\sidenotecontent@{}%
    \numgdef{\itemcount@}{0}%
    \def\do##1{%
        \ifnumequal{\itemcount@}{0}%
            {%
            \appto\sidenotecontent@{##1}}% Not print not separator before
the 1st note
            {\appto\sidenotecontent@{\@sidenotesep ##1}%
            }%
            \numgdef{\itemcount@}{\itemcount@+\@ne}%
    }%
    \dolistloop{\l@dcsnotetext}%
    \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
%
```

And we do the same for left and right notes (not movable).

```
  \gdef\@templ@d{}%
  \gdef\@templ@n{\l@dcsnotetext\l@dcsnotetext@l\l@dcsnotetext@r}%
  \ifx\@templ@d\@templ@n \else%
    \if@twocolumn%
      \if@firstcolumn%
        \setl@dlp@rbox{##1}{\sidenotecontent@}%
      \else%
        \setl@drp@rbox{\sidenotecontent@}%
      \fi%
    \else%
```

```
      \@l@dtempcntb=\sidenote@margin%
      \ifnum\@l@dtempcntb>\@ne%
        \advance\@l@dtempcntb by\page@num%
      \fi%
      \ifodd\@l@dtempcntb%
        \setl@drp@rbox{\sidenotecontent@}%
        \gdef\sidenotecontent@{}%
        \numgdef{\itemcount@}{0}%
        \dolistloop{\l@dcsnotetext@l}%
        \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
        \setl@dlp@rbox{\sidenotecontent@}%
      \else%
        \setl@dlp@rbox{\sidenotecontent@}%
        \gdef\sidenotecontent@{}%
        \numgdef{\itemcount@}{0}%
        \dolistloop{\l@dcsnotetext@r}%
        \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
        \setl@drp@rbox{\sidenotecontent@}%
      \fi%
    \fi%
  \fi%
  \advance\@edindex@fornote@\m@ne%
}
%
```

XXV Minipages and such

We can put footnotes into minipages. The preparatory code has been set up earlier, all that remains is to ensure that it is available inside a minipage box. This requires some alteration to the kernel code, specifically the \@iiiminipage and \endminipage macros. We will arrange this so that additional series can be easily added.

\l@dfeetbeginmini
\l@dfeetendmini

These will be the hooks in \@iiiminpage and \endminipage.

They can be extended to handle other things if necessary.

```
\ifnoledgroup@\else%
\newcommand*{\l@dfeetbeginmini}{\@ledgrouptrue\l@dedbeginmini\l@dfambeginmini}
\newcommand*{\l@dfeetendmini}{%
    \IfStrEq{critical-familiar}{\@mpfnpos}%
        {\l@dedendmini\l@dfamendmini}%
        {%
            \IfStrEq{familiar-critical}{\@mpfnpos}%
                {\l@dfamendmini\l@dedendmini}%
                {\do@feet@custom@order{mp@}{\@mpfnpos}}%
        }%
    }%
%
```

\l@dedbeginmini
\l@dedendmini
\mp@append@Xnotes

These handle the initiation and closure of critical footnotes in a minipage environment.

```
\newcommand*{\l@dedbeginmini}{%
  \unless\ifnocritical@%
    \def\do##1{%
      \csletcs{v##1footnote}{mpv##1footnote}%
    }%
    \dolistloop{\@series}%
  \fi%
  }
\newcommand*{\l@dedendmini}{%
  \unless\ifnocritical@%
    \ifl@dpairing%
      \ifledRcol%
        \flush@notesR%
      \else%
        \flush@notes%
      \fi%
    \fi
    \def\do##1{%
      \mp@append@Xnotes{##1}%
    }%
    \dolistloop{\@series}%
  \fi%
}%
\newcommand{\mp@append@Xnotes}[1]{%
\ifvoid\csuse{mp#1footins}\else%
  \ifl@dpairing%
    \ifparledgroup%
      \ifledRcol%
        \dimgdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+\skip\@nameuse{mp#1footins}}%
      \else%
         \dimgdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL+\skip\@nameuse{mp#1footins}}%
      \fi%
     \fi%
   \fi%
  \ifcsstring{series@display#1}{paragraph}{}{%
    \setbox\@nameuse{mp#1footins}=\vbox{%
      \csuse{Xnotefontsize@#1}%
      \ifcsdef{Xhsize\csuse{series@display#1}@#1}{%
        \hsize\csuse{Xhsize\csuse{series@display#1}@#1}%
      }{}%
      \noindent\csuse{Xtxtbeforenotes@#1}%
      \unvbox\@nameuse{mp#1footins}%
      \@parboxrestore%
    }%
  }%
  \csuse{mp#1footgroup}{#1}%
\fi%
}%
%
```

\l@dfambeginmini
\l@dfamendmini
\mp@append@notesX

These handle the initiation and closure of familiar footnotes in a minipage environment.

```
\newcommand*{\l@dfambeginmini}{%
 \unless\ifnofamiliar@%
   \def\do##1{\csletcs{vfootnote##1}{mpvfootnote##1}}%
     \dolistloop{\@series}%
 \fi%
}%

\newcommand*{\l@dfamendmini}{%
 \unless\ifnofamiliar@%
   \def\do##1{%
     \mp@append@notesX{##1}%
   }%
   \dolistloop{\@series}%
 \fi%
}%
\newcommand{\mp@append@notesX}[1]{%
  \ifvoid\csuse{mpfootins#1}\else%
    \ifcsstring{series@displayX#1}{paragraph}{}{%
      \setbox\@nameuse{mpfootins#1}=\vbox{%
        \csuse{notefontsizeX@#1}%
        \ifcsdef{hsize\csuse{series@display#1}X@#1}{%
          \hsize\csuse{hsize\csuse{series@display#1}X@#1}%
        }{}%
        \noindent\csuse{txtbeforenotesX@#1}%
        \unvbox\@nameuse{mpfootins#1}%
        \@parboxrestore%
      }%
    }%
    \csuse{mpfootgroup#1}{#1}%
  \fi%
}%
%
```

\@iiiminipage

This is our extended form of the kernel \@iiiminipage defined in ltboxes.dtx.

```
\patchcmd%
  {\@iiiminipage}%
  {\let\@footnotetext\@mpfootnotetext}%
  {\let\@footnotetext\@mpfootnotetext\l@dfeetbeginmini}%
  {}%
  {\led@error@fail@patch@@iiiminipage}%
%
```

\endminipage

This is our extended form of the kernel \endminipage defined in ltboxes.dtx.

```
\patchcmd%
  {\endminipage}%
  {\footnoterule}%
  {\footnoterule\l@advance@parledgroup@beforenormalnotes}%
  {}%
  {\led@error@fail@patch@endminipage}

\patchcmd%
  {\endminipage}%
  {\@minipagefalse}%
  {\l@dfeetendmini\@minipagefalse}%
  {}%
  {\led@error@fail@patch@endminipage}

%
```

\l@dunboxmpfoot
dvance@parledgroup@beforenormalnotes

\@ldunboxmpfoot insert normal footnotes for ledgroup.

```
\newcommand*{\l@dunboxmpfoot}{%
    \vskip\skip\@mpfootins
    \normalcolor
    \footnoterule
    \l@advance@parledgroup@beforenormalnotes
    \unvbox\@mpfootins%
}
%
```

When using parallel ledgroup, we need to store the vertical space added before footnote, in order to compensate them between left and right pages.

```
\newcommand{\l@advance@parledgroup@beforenormalnotes}{%
  \ifparledgroup
    \ifl@dpairing
      \ifledRcol
          \dimgdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+\
skip\@mpfootins}
      \else
          \dimgdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL+\
skip\@mpfootins}
      \fi
    \fi
  \fi
}
%
```

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.

```
\newenvironment{ledgroup}{%
  \resetprevpage@num%
  \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@%
  \let\@footnotetext\@mpfootnotetext
  \l@dfeetbeginmini%
}{%
  \par
  \unskip
  \ifvoid\@mpfootins\else
    \l@dunboxmpfoot
  \fi
  \l@dfeetendmini%
  \@ledgroupfalse%
}


%
```

ledgroupsized `\begin{ledgroupsized}[`⟨*pos*⟩`]{`⟨*width*⟩`}`

This environment puts footnotes at the end, even if that happens to be in the middle of a page, or crossing a page boundary. It is a sort of unboxed, variable ⟨*width*⟩ minipage. The optional ⟨*pos*⟩ controls the sideways position of numbered text.

```
\newenvironment{ledgroupsized}[2][l]{%
%
```

Set the various text measures.

```
  \hsize #2\relax
%
```

Initialize fills for centering.

```
  \let\ledllfill\hfil
  \let\ledrlfill\hfil
  \def\@tempa{#1}\def\@tempb{l}%
%
```

Left adjusted numbered lines

```
      \ifx\@tempa\@tempb
    \let\ledllfill\relax
  \else
    \def\@tempb{r}%
    \ifx\@tempa\@tempb
%
```

Right adjusted numbered lines

```
      \let\ledrlfill\relax
    \fi
  \fi
%
```

Set up the footnoting.

```
  \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
  \let\@footnotetext\@mpfootnotetext
  \l@dfeetbeginmini%
}{%
  \par
  \unskip
  \ifvoid\@mpfootins\else
    \l@dunboxmpfoot
  \fi
  \l@dfeetendmini%
}

%
```

Close the `\ifnoledgroup@\else`.

```
\fi%
%
```

`\ifledgroupnotesL@`
`\ifledgroupnotesR@`

These boolean tests check if we are in the notes of a ledgroup. If we are, we do not number the lines. It could be useful for parallel ledgroup of `reledpar`.

```
\newif\ifledgroupnotesL@
\newif\ifledgroupnotesR@
%
```

XXVI Indexing

Here is some code for indexing using page and line numbers.

XXVI.1 Looking on package order

First, ensure that `imakeidx` or `indextools` is loaded *before* `eledmac`.

```
\AtBeginDocument{%
  \unless\ifl@imakeidx%
    \@ifpackageloaded{imakeidx}{\led@error@PackageAfterEledmac{imakeidx}}{}
%
  \fi%
  \unless\ifl@indextools%
    \@ifpackageloaded{indextools}{\led@error@PackageAfterEledmac{indextools
}}{}%
  \fi%
  \unless\ifl@footmisc%
    \@ifpackageloaded{footmisc}{\led@error@PackageAfterEledmac{footmisc}}{}
%
  \fi%
}
%
```

XXVI.2 Auxiliary macros for `\edindex`

`\pagelinesep` `\edindexlab` `\c@labidx` In order to get a correct line number we have to use the label/ref mechanism. These macros are for that.

```
\newcommand{\pagelinesep}{-}
\newcommand{\edindexlab}{$&}
\newcounter{labidx}
\setcounter{labidx}{0}

%
```

`\doedindexlabel` This macro sets an `\edlabel`.

```
\newcommand{\doedindexlabel}{%
  \stepcounter{labidx}%
  \edlabel{\edindexlab\thelabidx}%
}

%
```

`\thepageline` This macro makes up the page/line number combo from the label/ref. The associated counter is never directly used, but it is required in order to not have any error message with `\edgls`.

```
\newcounter{pageline}%
\renewcommand{\thepageline}{%
  \thepage%
  \pagelinesep%
  \xlineref{\edindexlab\thelabidx}%
}
%
```

`\thestartpageline` `\theendpageline` These macros make up the page/line start/end number when the `\edindex` command is called in critical notes.

```
\newcommand{\thestartpageline}{%
  \l@dparsedstartpage%
  \pagelinesep%
  \l@dparsedstartline%
}
\newcommand{\theendpageline}{%
  \l@dparsedendpage%
  \pagelinesep%
  \l@dparsedendline%
}
%
```

XXVI.3 Code specific to \edindex in critical footnotes

\@edindex@fornote@

This counter is incremented at the beginning of each note (either a footnote or a sidenote), and decremented at the end of each note. If its value is greater than 0, that means we are inside a note.

```
\newcount\@edindex@fornote@
%
```

\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 |.

```
\newcommand{\prepare@edindex@fornote}[1]{%
    \l@dp@rsefootspec#1|%
    \advance\@edindex@fornote@\@ne%
}
%
```

\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.

```
\newcommand{\get@edindex@ledinnote@command}{%
  \ifxindy@%
    \gdef\@ledinnote@command{%
      ledinnote\thelabidx%
    }%
    \ifxindyhyperref@%
      \immediate\write\eledmac@xindy@out{%
        (define-attributes ("ledinnote\thelabidx"))^^J
        \space\space(markup-locref^^J
          \eledmacmarkuplocrefdepth^^J
          :open "\string\ledinnote[\edindexlab\thelabidx]{\@index@command}{"^^J
          :close "}"^^J
          :attr "ledinnote\thelabidx"^^J
          )
        }%
    \else%
      \immediate\write\eledmac@xindy@out{%
        (define-attributes ("ledinnote\thelabidx"))^^J
        \space\space(markup-locref^^J
          \eledmacmarkuplocrefdepth^^J
          :open "\string\ledinnote{\@index@command}{"^^J
          :close "}"^^J
          :attr "ledinnote\thelabidx"^^J
          )
        }%
    \fi%
%
```

If we do not use xindy option, `\@ledinnote@command` will produce something like `ledinnote{formattingcommand}`.

```
  \else%
    \gdef\@ledinnote@command{%
       ledinnote[\edindexlab\thelabidx]{\@index@command}%
    }%
  \fi%
}
%
```

XXVI.4 Analysis of command in indexed text

\get@index@command This macro is used to analyze if a text to be indexed has a command after a `|`.

```
\def\get@index@command#1|#2+{%
  \gdef\@index@txt{#1}%
  \gdef\@index@command{#2}%
  \xdef\@index@parenthesis{}%
  \IfBeginWith{\@index@command}{(}{%
    \StrGobbleLeft{\@index@command}{1}[\@index@command@]%
    \global\let\@index@command\@index@command@%
    \xdef\@index@parenthesis{(}%
    }{}%
  \IfBeginWith{\@index@command}{)}{%
    \StrGobbleLeft{\@index@command}{1}[\@index@command@]%
    \global\let\@index@command\@index@command@%
    \xdef\@index@parenthesis{)}%
    }{}%
}
%
```

XXVI.5 Code for the formatted index

\ledinnote
\ledinnotehyperpage
\ledinnotemark

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.

```
\newcommandx{\ledinnote}[3][1,usedefault]{%
  \ifboolexpr{%
    test{\ifdefequal{\iftrue}{\ifHy@hyperindex}}%
    or%
    bool {xindyhyperref@}%
    }%
    {%
    \csuse{#2}{\hyperlink{#1}{\ledinnotemark{#3}}}%
    }%
    {%
    \csuse{#2}{\ledinnotemark{#3}}%
    }%
}%
\newcommand{\ledinnotehyperpage}[2]{\csuse{#1}{\ledinnotemark{\hyperpage{#2}}}}%
\newcommand{\ledinnotemark}[1]{#1\emph{n}}%
%
```

XXVI.6 Main code

Eledmac and ledmac were using the specific indexing tools of the memoir in order to allow multiple index. However, eledmac used imakeidx or indextools tools when one these two package was loaded. This system forced to maintained a double code, which was not very useful. Since reledmac, we use only the imakeidx or indextools tools.

The memoir class provides more flexible indexing than the standard classes. We need different code if the memoir class is being used, except if imakeidx or indextools is used.

\edindex Write the index information to the idx file.

\@wredindex
\dummy@edindex

```
\newcommandx{\@wredindex}[2][1=\expandonce\jobname,usedefault]{%#1 = the index name, #2 = the text
  \begingroup%
    \let\emph\@firstofone%
    \let\textbf\@firstofone%
    \let\textit\@firstofone%
    \let\textmd\@firstofone%
    \let\textnormal\@firstofone%
    \let\textrm\@firstofone%
    \let\textsc\@firstofone%
    \let\textsf\@firstofone%
    \let\textsl\@firstofone%
    \let\texttt\@firstofone%
    \let\textup\@firstofone%
    \xdef\@tmp{#2}%To be used in IfSubStr instead of #2 directly. Avoid some expansion bugs (for example with \edindex{textsc{something}})
  \endgroup%
  \ifl@imakeidx%
    \ifnum\@edindex@fornote@>\z@%
      \IfSubStr[1]{\@tmp}{|}{\get@index@command#2+}{\get@index@command#2|+}%
      \get@edindex@ledinnote@command%
      \expandafter\imki@wrindexentry{#1}{\@index@txt|(\@ledinnote@command}{\thestartpageline}%
      \expandafter\imki@wrindexentry{#1}{\@index@txt|)\@ledinnote@command}{\theendpageline}%
    \else%
      \get@edindex@hyperref{#2}%
      \imki@wrindexentry{#1}{\@index@txt\@edindex@hyperref}{\thepageline}%
    \fi%
  \else%
    \ifnum\@edindex@fornote@>\z@%
      \IfSubStr[1]{\@tmp}{|}{\get@index@command#2+}{\get@index@command#2|+}
%
      \get@edindex@ledinnote@command%
      \expandafter\protected@write\@indexfile{}%
  {\string\indexentry{\@index@txt|(\@ledinnote@command}{\thestartpageline}
  }%
      \expandafter\protected@write\@indexfile{}%
  {\string\indexentry{\@index@txt|)\@ledinnote@command}{\theendpageline}
       }%
    \else%
      \protected@write\@indexfile{}%
  {\string\indexentry{#2}{\thepageline}
  }%
    \fi%
   \fi%
   \endgroup
   \@esphack%
}
%
```

Need to add the definition of \edindex to \makeindex, and initialise \edindex to do nothing.

```
\pretocmd{\makeindex}{%
  \def\edindex{%
    \ifboolexpr{bool{numbering} or bool{numberingR} or bool{
l@dprintingpages} or bool{l@dprintingcolumns}}{%
      \@bsphack%
      \doedindexlabel%
      \begingroup%
      \@sanitize%
      \@wredindex%
    }%
    {%
      \led@warn@edinde@outsidenumbering%
      \index%
    }%
  }%
}%
{}%
{\led@error@fail@patch@makeindex}%
\newcommand{\edindex}[1]{\@bsphack\@esphack}
\newcommandx{\dummy@edindex}[2][1=\expandonce\jobname,usedefault]{}%
%
```

XXVI.7 Hyperlink

\hyperlinkformat

\hyperlinkformat command is to be used to have both a internal hyperlink and a format, when indexing.

```
\newcommand{\hyperlinkformat}[3]{%
  \ifstrempty{#1}%
    {\hyperlink{#2}{#3}}%
    {\csuse{#1}{\hyperlink{#2}{#3}}%
  }}
%
```

\hyperlinkR

\hyperlinkR command is to be used to create a internal hyperlink and \ledRflag, when indexing.

```
\newcommand{\hyperlinkR}[2]{%
 \hyperlink{#1}{#2\@Rlineflag}%
}%

%
```

\hyperlinkformatR

\hyperlinkformatR command is to be used to create a internal hyperlink, a format and a \@Rlineflag, when indexing.

```
\newcommand{\hyperlinkformatR}[3]{%
  \hyperlinkformat{#1}{#2}{#3\@Rlineflag}%
}%

%
```

\get@edindex@hyperref
\@edindex@hyperref

\get@edindex@hyperref is to be used to define the \@edindex@hyperref macro, which, in index, links to the point where the index was called (with hyperref.

```
\newcommand{\get@edindex@hyperref}[1]{%
%
```

We have to disable temporary spaces to work through a xstring bug (or feature?)

```
  \edef\temp@{%
  \catcode`\ =9 %space need for catcode
  \detokenize{#1}%For active character in unicode
  \catcode`\ =10 % space need for catcode
  }%
%
```

Now, we define \@edindex@hyperref if the hyperindex of hyperref is enabled.

```
  \ifdefequal{\iftrue}{\ifHy@hyperindex}{%
    \IfSubStr{\temp@}{|}%
      {\get@index@command#1+%
      \ifledRcol%
        \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
    hyperlinkformatR{\@index@command}%
    {\edindexlab\thelabidx}}%
   \else%
    \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
    hyperlinkformat{\@index@command}%
    {\edindexlab\thelabidx}}%
   \fi%
   }%
   {\get@index@command#1|+%
    \ifledRcol%
    \gdef\@edindex@hyperref{|hyperlinkR{\edindexlab\thelabidx}}%
    \else%
    \gdef\@edindex@hyperref{|hyperlink{\edindexlab\thelabidx}}%
    \fi%
   }%
  }%
%
```

```
% If we use both xindy and hyperref, first get the \protect\cs{index@command} command.
% Then define \protect\cs{@edindex@hyperref} in the form \verb+eledmacXXX+
%    \begin{macrocode}
  {\ifxindyhyperref@%
   \IfSubStr{\temp@}{|}%
    {\get@index@command#1+}%
    {\get@index@command#1|+}%
   \gdef\@edindex@hyperref{|eledmac\thelabidx}%
%
```

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`.

```
   \IfStrEq{\@index@parenthesis}{(}%
    {%
    \csxdef{xindyparenthesis@\@index@txt}{\thelabidx}%
    \gdef\@edindex@hyperref{|(eledmac\thelabidx}%
    }%
    {}%
%
```

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`.

```
   \IfStrEq{\@index@parenthesis}{)}%
    {%
    \xdef\@edindex@hyperref{|)eledmac\csuse{xindyparenthesis@\@index@txt}}%
    \global\csundef{xindyparenthesis@\@index@txt}%
    }%
%
```

Write in the .xdy file the attributes of the location.

```
        {%
        \immediate\write\eledmac@xindy@out{%
         (define-attributes ("eledmac\thelabidx"))^^J
          \space\space(markup-locref^^J
            \eledmacmarkuplocrefdepth^^J
            :open "\string\hyperlink%
                   \ifledRcol R\fi%
                   {\edindexlab\thelabidx}%
                   {\ifdefempty{\@index@command}%
                     {}%
                     {\@backslashchar\@index@command}%
                   {"^^J
            :close "}}"^^J
            :attr "eledmac\thelabidx"^^J
            )
          }%
        }%
%
```

And now, in any other case.

```
    \else%
      \gdef\@index@txt{#1}%
      \gdef\@edindex@hyperref{}%
    \fi%
    }%
}
%
```

XXVI.8 'innote' and 'notenumber' option of indextools package

\led@set@index@fornote

The \led@set@index@fornote is called when a familiar footnote is inserted — and not when it is read — and changes the \index command depending of the option of the indextools package. Its only argument is the note series.

```
\newcommand{\led@set@index@fornote}[1]{%
  \ifbool{indtl@innote}%
    {\let\index\nindex}%
    {}%
  \ifbool{indtl@notenumber}%
    {%
    \renewcommand{\index}[2][\indtl@jobname]{%
      \orig@@index[##1]{%
        ##2|innotenumber{\this@footnoteX@reading}%
        }%
      }%
    }%
    {}%
}%
%
```

\led@reinit@index@fornote

The \led@reinit@index@fornote just reset the default value of \index.

```
\newcommand{\led@reinit@index@fornote}{%
  \ifbool{indtl@innote}%
    {\let\index\orig@@index}%
    {}%
 \ifbool{indtl@notenumber}%
  {\let\index\orig@@index}%
  {}%
}%
%
```

XXVII Glossaries

Here, we define the \gls-like commands prefixed by ed, only if the package glossaries is loaded.

```
\AtBeginDocument{%
  \@ifpackageloaded{glossaries}{%
%
```

First those which arguments are [⟨*options*⟩]{⟨*label*⟩}[⟨*insert*⟩].

```
    \renewcommand{\do}[1]{%
      \expandafter\DeclareRobustCommandx\csname ed#1\endcsname[3][1,3,usedefault]{%
        \doedindexlabel%
        \csname#1\endcsname[counter=pageline,##1]{##2}[##3]%
      }%
      \expandafter\WithSuffix\expandafter\DeclareRobustCommandx\csname ed#1\endcsname*[3][1,3,usedefault]{%
        \doedindexlabel%
        \csname#1\endcsname*[counter=pageline,##1]{##2}[##3]%
      }%
    }%
    \docsvlist{%
      gls,%
      Gls,%
      GLS,%
      glspl,%
      Glspl,%
      GLSpl,%
      glstext,%
      Glstext,%
      GLStext,%
      Glsfirst,%
      GLSfirst,%
      glsplural%
      Glsplural,%
      GLSplural,%
      glsfirstplural,%
      Glsfirstplural,%
      GLSfirstplural,%
      glsname,%
      Glsname,%
      GLSname,%
      glssymbol,%
      Glssymbol,%
      GLSsymbol,%
      glsdesc,%
      Glsdesc,%
      GLSdesc,%
      glsuseri,%
      Glsuseri,%
      GLSuseri,%
      glsuserii,%
      Glsuserii,%
      GLSuserii,%
      glsuseriii,%
      Glsuseriii,%
      GLSuseriii,%
      glsuseriv,%
      Glsuseriv,%
      GLSuseriv,%
      glsuserv,%
      Glsuserv,%
      GLSuserv,%
      glsuservi,%
      Glsuservi,%
      GLSuservi%
      }%
%
```

Then those which arguments are [⟨*options*⟩]{⟨*label*⟩}{⟨*link text*⟩}.

```
    \renewcommand{\do}[1]{%
      \expandafter\DeclareRobustCommandx\csname ed#1\endcsname[3][1,usedefault]{%
        \doedindexlabel%
        \csname#1\endcsname[counter=pageline,##1]{##2}{##3}%
      }%
      \expandafter\WithSuffix\expandafter\DeclareRobustCommandx\csname ed#1\endcsname*[3][1,usedefault]{%
        \doedindexlabel%
        \csname#1\endcsname*[counter=pageline,##1]{##2}{##3}%
      }%
    }%
    \docsvlist{glsdisp,glslink}%
```

```
%
```

Then those which arguments are [⟨*options*⟩]{⟨*label*⟩}.

```
    \renewcommand{\do}[1]{%
      \expandafter\DeclareRobustCommandx\csname ed#1\endcsname[2][1,usedefault]{%
        \doedindexlabel%
        \csname#1\endcsname[counter=pageline,##1]{##2}%
      }%
      \expandafter\WithSuffix\expandafter\DeclareRobustCommandx\csname ed#1\endcsname*[2][1,usedefault]{%
        \doedindexlabel%
        \csname#1\endcsname*[counter=pageline,##1]{##2}%
      }%
    }%
    \docsvlist{glsadd}%
  }{}%
}%
%
```

XXVIII Verse

The original code is principally Wayne Sullivan's code from edstanza. However, the code has been many time modified by Maïeul Rouquette in order to obtain new features and improved compatibility with reledpar.

XXVIII.1 Hanging symbol management

\@hangingsymbol
\sethangingsymbol
\ifinstanza

The macro \@hangingsymbol is used to insert a symbol on each hanging of verses. It is set by user level macro \sethangingsymbol.

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.

```
\def\@hangingsymbol{}
\newcommand*{\sethangingsymbol}[1]{%
  \gdef\@hangingsymbol{#1}%
}%
\newif\ifinstanza
%
```

\inserthangingymbol
\ifinserthangingsymbol

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.

```
\newif\ifinserthangingsymbol
\newcommand{\inserthangingsymbol}{%
\ifinserthangingsymbol%
   \ifinstanza%
      \@hangingsymbol%
   \fi%
\fi%
}
%
```

XXVIII.2 Using & character

\ampersand

Within a stanza the \& macro is going to be usurped. We need an alias in case an & needs to be typeset in a stanza. Define it rather than letting it in case some other package has already defined it.

```
\newcommand*{\ampersand}{\char`\&}

%
```

XXVIII.3 Code category setting

\stanza@count
\stanzaindentbase

Before we can define the main macros we need to save and reset some category codes. To save the current values we use \next and \body from the \loop macro.

```
  \chardef\body=\catcode`\@
  \catcode`\@=11
  \chardef\next=\catcode`\&
  \catcode`\&=\active

%
```

XXVIII.4 Stanza count and indent

A count register is allocated for counting lines in a stanza; also allocated is a dimension register which is used to specify the base value for line indentation; all stanza indentations are multiples of this value. The default value of \stanzaindentbase is 20pt.

```
  \newcount\stanza@count
  \newlength{\stanzaindentbase}
  \setlength{\stanzaindentbase}{20pt}

%
```

\strip@szacnt
\setstanzavalues

The indentations of stanza lines are non-negative integer multiples of the unit called \stanzaindentbase. To make it easier for the user to specify these numbers, some list macros are defined. These take numerical values in a list separated by commas and assign the values to special control sequences using \mathchardef. Though this does limit the range from 0 to 32767, it should suffice for most applications, including *penalties*, which will be discussed below.

```
\def\strip@szacnt#1,#2|{\def\@tempb{#1}\def\@tempa{#2|}}
\newcommand*{\setstanzavalues}[2]{\def\@tempa{#2,,|}%
        \stanza@count\z@
     \def\next{\expandafter\strip@szacnt\@tempa
            \ifx\@tempb\empty\let\next\relax\else
            \expandafter\mathchardef\csname #1@\number\stanza@count
            @\endcsname\@tempb\relax
            \advance\stanza@count\@ne\fi\next}%
      \next}

%
```

\setstanzaindents
\setstanzapenalties

In the original edmac, \setstanzavalues{sza}{⟨...⟩} had to be called to set the indents, and similarly \setstanzavalues{szp}{⟨...⟩}. to set the penalties. \setstanzaindents and \setstanzapenalties macros are a convenience to give the user one less thing to worry about (misspelling the first argument).

```
\newcommand*{\setstanzaindents}[1]{\setstanzavalues{sza}{#1}}
\newcommand*{\setstanzapenalties}[1]{\setstanzavalues{szp}{#1}}
%
%
```

\managestanza@modulo

Since version 0.13, the stanzaindentsrepetition counter can be used when the indentation is repeated every n verses. The \managestanza@modulo is a command which modifies the counter stanza@modulo. The command adds 1 to stanza@modulo, but if stanza@modulo is equal to the stanzaindentsrepetition counter, the command restarts it.

```
\newcounter{stanzaindentsrepetition}
\newcount\stanza@modulo

\newcommand*{\managestanza@modulo}[0]{%
    \advance\stanza@modulo\@ne%
    \ifnum\stanza@modulo>\value{stanzaindentsrepetition}%
     \stanza@modulo\@ne%
    \fi%
}
%
```

\stanzaindent
\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 skips the current verse for the repetition of stanza indent.

```
\newcommand{\stanzaindent}[1]{%
  \hspace{\dimexpr#1\stanzaindentbase-\parindent\relax}%
  \ignorespaces%
}%
```

```
\WithSuffix\newcommand\stanzaindent*[1]{%
  \stanzaindent{#1}%
  \global\advance\stanza@modulo-\@ne%
  \ifnum\stanza@modulo=0%
    \global\stanza@modulo=\value{stanzaindentsrepetition}%
  \fi%
  \ignorespaces%
}%
%
```

XXVIII.5 Numbering stanza

Here, macro for numbering stanza. First, the stanza counter.

\thestanza

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

\ifnumberstanza

Then, macro to activate automatically numbering of stanza.

```
\newif\ifnumberstanza%
%
```

\@insertstanzanumber

Now, macro called at the first line of of verse of a stanza.

```
\newcommand{\@insertstanzanumber}[0]{%
  \ifnumberstanza%
    \ifl@dpairing%
      \ifledRcol%
      \stanzanumwrapper{\thestanzaR}%
      \else%
      \stanzanumwrapper{\thestanzaL}%
      \fi%
    \else%
      \stanzanumwrapper{\thestanza}%
    \fi%
    \setline{1}%
  \fi%
}%
%
```

\@advancestanzanumber

Also a command to advance the counter of stanza.

```
\newcommand{\@advancestanzanumber}[0]{%
  \ifnumberstanza%
    \ifl@dpairing%
      \ifledRcol%
```

```
        \addtocounter{stanzaR}{1}%
      \else%
        \addtocounter{stanzaL}{1}%
      \fi%
    \else%
      \addtocounter{stanza}{1}%
    \fi%
  \fi%
}%
%
```

\stanzanumwrapper

And finally, the wrapper for stanza number

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

XXVIII.6 Stanza number in note

Here, the command called when printing stanza number in notes.

```
\newcommand{\printstanza}[0]{%
   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
        \ifledRcol@%
            \thestanzaR%
        \else%
            \thestanzaL%
        \fi%
    }{%
        \thestanza%
    }%
}
%
```

XXVIII.7 Main work

\stanza@line
\stanza@hang
\sza@penalty

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.

```
\newcommandx{\stanza@line}[2][1,2,usedefault]{%
    \ifnum\value{stanzaindentsrepetition}=0
```

```
        \ifcsdef{sza@\number\stanza@count @}%
          {%
          \parindent=\csname sza@\number\stanza@count  @\endcsname\stanzaindentbase%
          }{%
          \led@err@StanzaIndentNotDefined%
          }%
    \else
        \ifcsdef{sza@\number\stanza@modulo @}{%
          \parindent=\csname sza@\number\stanza@modulo @\endcsname\stanzaindentbase%
          \managestanza@modulo%
        }%
        {%
        \led@err@StanzaIndentNotDefined%
        }%
    \fi
    \pstart[#1][#2]\stanza@hang\ignorespaces%
}%
\xdef\stanza@hang{\noexpand\leavevmode\noexpand\startlock
            \hangindent\expandafter
            \noexpand\csname sza@0@\endcsname\stanzaindentbase
            \hangafter\@ne}
\def\sza@penalty{\count@\csname szp@\number\stanza@count @\endcsname
         \ifnum\count@>\@M\advance\count@-\@M\penalty-\else
         \penalty\fi\count@}
%
```

\@startstanza
\stanza
\@stopstanza
\AtEveryStopStanza
\AtEveryStanza
\AtStartEveryStanza
\BeforeEveryStopStanza
\newverse

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 \&.

```
\xdef\@startstanza[#1][#2]{%
   \noexpand\instanzatrue\expandafter
   \begingroup%
   \catcode`\noexpand\&\active%
   \global\stanza@count\@ne\stanza@modulo\@ne
   \noexpand\ifnum\expandafter\noexpand
   \csname sza@0@\endcsname=\z@\let\noexpand\stanza@hang\relax
   \let\noexpand\endlock\relax\noexpand\else\interlinepenalty
   \@M\rightskip\z@ plus 1fil\relax\noexpand\fi\noexpand\ifnum
   \expandafter\noexpand\csname szp@0@\endcsname=\z@
   \let\noexpand\sza@penalty\relax\noexpand\fi%
   \def\noexpand&{%
```

```
         \noexpand\newverse[][]}%
   \def\noexpand\&{\noexpand\@stopstanza}%
   \noexpand\@advancestanzanumber%
   \noexpand\stanza@line[#1][#2]%
   \noexpand\@insertstanzanumber%
   \let\par\relax\ignorespaces%No paragraph in verses
}

\newcommandx{\stanza}[2][1,2,usedefault]{%
  \ifboolexpr{%
    not test{\ifdefvoid{\at@every@stanza}}%
    and test{\ifstrempty{#1}}%
    and test{\ifstrempty{#2}}}%
    {\@startstanza[][\at@every@stanza]\at@start@every@stanza}%
    {\@startstanza[#1][#2]\at@start@every@stanza}%
}%

\newcommandx{\@stopstanza}[2][1,2,usedefault]{%
  \unskip%
  \endlock%
  \ifboolexpr{%
    not test{\ifdefvoid{\at@every@stop@stanza}}%
    and test{\ifstrempty{#1}}%
    and test{\ifstrempty{#2}}}%
      {\before@every@stop@stanza\pend[][\at@every@stop@stanza]}%
      {\before@every@stop@stanza\pend[#1][#2]}%
  \endgroup%
  \instanzafalse%
}

\newcommand{\AtEveryStopStanza}[1]{%
  \ifstrempty{#1}%
    {\gdef\at@every@stop@stanza{}}%
    {\gdef\at@every@stop@stanza{\noindent#1}}%
}%
\WithSuffix\newcommand\AtEveryStopStanza*[1]{%
  \ifstrempty{#1}%
    {\gdef\at@every@stop@stanza{}}%
    {\gdef\at@every@stop@stanza{#1}}%
}%
\def\at@every@stop@stanza{}%

\newcommand{\AtEveryStanza}[1]{%
  \ifstrempty{#1}%
    {\gdef\at@every@stanza{}}%
    {\gdef\at@every@stanza{\noindent#1}}%
}%
\WithSuffix\newcommand\AtEveryStanza*[1]{%
  \ifstrempty{#1}%
    {\gdef\at@every@stanza{}}%
    {\gdef\at@every@stanza{#1}}%
}%



\newcommand{\AtStartEveryStanza}[1]{%
  \ifstrempty{#1}%
    {\gdef\at@start@every@stanza{}}%
    {\gdef\at@start@every@stanza{#1}}%
}%
\def\at@start@every@stanza{}%

\newcommand{\BeforeEveryStopStanza}[1]{%
  \ifstrempty{#1}%
    {\gdef\before@every@stop@stanza{}}%
    {\gdef\before@every@stop@stanza{#1}}%
}%
\def\before@every@stop@stanza{}%

\newcommandx*{\newverse}[4][1,2,3,4,usedefault]{%
  \unskip%
  \endlock\pend[#1][#3]\sza@penalty\global%
  \advance\stanza@count\@ne\stanza@line[#2][#4]%
  }

%
```

\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`.

```
\newcommand*{\flagstanza}[2][\stanzaindentbase]{%
  \hskip -#1\llap{#2}\hskip #1\ignorespaces}

%
```

XXVIII.8 Restore catcode and penalties

The ampersand & is used to mark the end of each stanza line, except the last, which is marked with `\&`. This means that `\halign` may not be used directly within a stanza line. This does not affect macros involving alignments defined outside `\stanza` `\&`. Since these macros usurp the control sequence `\&`, the replacement `\ampersand` is defined to be used if this symbol is needed in a stanza. Also we reset the modified category codes and initialize the penalty default.

```
  \catcode`\&=\next
  \catcode`\@=\body
  \setstanzavalues{szp}{0}

%
```

XXIX Apparatus of Manuscripts

XXIX.1 User level macro

\msdata
\msdata@c
\msdata@cR

The user level \msdata command only writes the manuscripts data in numbered auxiliary file. There are two associated etoolbox counters.

```
\def\msdata@c{}%
\def\msdata@cR{}%
\newcommand{\msdata}[1]{%
  \leavevmode%
  \unless\ifstopmsdata@inserted@%
    \stopmsdata%
    \led@warning@msdatawithoutstop%
  \fi%
  \global\stopmsdata@inserted@false%
  \unless\ifledRcol%
    \numgdef{\msdata@c}{\msdata@c+1}%
    \ifdef{\hypertarget}{%
      \edlabel{\msdata@c:start:msdata}%
    }{}%
    \protected@write\linenum@out{}{%
      \string\@msd{#1}%
    }%
  \else%
    \numgdef{\msdata@cR}{\msdata@cR+1}%
    \ifdef{\hypertarget}{%
      \edlabel{\msdata@cR:start:msdata}%
    }{}%
    \protected@write\linenum@outR{}{%
      \string\@msd{#1}%
    }%
  \fi%
}%
%
```

\stopmsdata

The user level \stopmsdata command only writes information about the end of manuscripts data in numbered auxiliary file.

```
\newcommand{\stopmsdata}[0]{%
  \leavevmode%
  \unless\ifledRcol%
    \protected@write\linenum@out{}{%
      \string\@stopmsd%
    }%
    \ifdef{\hypertarget}{%
      \edlabel{\msdata@c:end:msdata}%
    }{}%
  \else%
    \protected@write\linenum@outR{}{%
```

```
      \string\@stopmsd%
    }%
   \ifdef{\hypertarget}{%
     \edlabel{\msdata@cR:end:msdata}%
   }{}%
  \fi%
  \global\stopmsdata@inserted@true%
}%
%
```

\ifstopmsdata@inserted@ The \ifstopmsdata@inserted@ boolean is set to TRUE at every \stopmsdata and reset to FALSE at all \msdata. It also set to TRUE at every \beginnumbering. It is used to automatically insert \stopmsdata if forgotten before \msdata

```
\newif\ifstopmsdata@inserted@%
%
```

XXIX.2 Setting macro

Setting macros for the manuscripts apparatus tools is very easy: they just save their argument in an internal macro.

\setmsdataseries In which series of notes will be printed the apparatus of manuscripts?

```
\newcommand{\setmsdataseries}[1]{%
  \gdef\@msdata@series{#1}%
}%
\def\@msdata@series{A}%
%
```

\setmsdataposition The label for the manuscripts data.

```
\def\ms@data@position{msdata-regular}%
\newcommand{\setmsdataposition}[1]{%
  \gdef\ms@data@position{#1}%
}%
%
```

\setmsdatalabel The label for the manuscripts data.

```
\def\ms@data@label{Ms.}%
\newcommand{\setmsdatalabel}[1]{%
  \gdef\ms@data@label{#1}%
}%
%
```

XXIX.3 Counters and lists

\@msd@c

\@msd@c is a counter incremented at each \@msd read in auxiliary file.

```
\numdef{\@msd@c}{0}
\numdef{\@msd@cR}{0}
%
```

\add@msd@

\add@msd@ is a counter incremented at each \add@msddata, that is at each time we prepare the insertion of manuscripts data footnote.

```
\numdef{\add@msd@c}{0}%
\numdef{\add@msd@cR}{0}%
%
```

\@msdata@list

The \@msdata@list will contain, for each line, the lists of command to be executed to insert the manuscripts apparatus. It will be filled on \add@msdata and looped on \insert@msdata, then emptied.

```
\def\@msdata@list{}%
%
```

XXIX.4 Auxiliary file macros

\@msd

The \@msd macro is written in the auxiliary file. It just defines three macros by \msdata macro, which allow us to know the manuscripts data, the line number and the absolute line number where it was called

It also stores the action code 1010 in the list of actions by line.

```
\newcommand{\@msd}[1]{%
  \unless\ifledRcol%
    \global\numdef{\@msd@c}{\@msd@c+\@ne}%
    \csgdef{@msdata@\@msd@c @data}{#1}%
    \csxdef{@msdata@\@msd@c @linenumber}{\the\line@num}%
    \csxdef{@msdata@\@msd@c @abslinenumber}{\the\absline@num}%
    \xright@appenditem{\the\absline@num}\to\actionlines@list%
    \xright@appenditem{-1010}\to\actions@list%
  \else%
    \global\numdef{\@msd@cR}{\@msd@cR+\@ne}%
    \csgdef{@msdata@\@msd@cR @dataR}{#1}%
    \csxdef{@msdata@\@msd@cR @linenumberR}{\the\line@numR}%
    \csxdef{@msdata@\@msd@cR @abslinenumberR}{\the\absline@numR}%
    \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
    \xright@appenditem{-1010}\to\actions@listR%
 \fi%
}%
%
```

\@stopmsd

Inserted in the auxiliary file by \stopmsd, the \@stopmsd macro will store in two commands the line number and the absolute line number on which it is called.

```
\newcommand{\@stopmsd}[0]{%
  \unless\ifledRcol%
     \ifcsundef{@msdata@\@msd@c @stoplinenumber}{%
       \csxdef{@msdata@\@msd@c @stopabslinenumber}{\the\absline@num}%
       \csxdef{@msdata@\@msd@c @stoplinenumber}{\the\line@num}%
     }{}%
  \else%
     \ifcsundef{@msdata@\@msd@cR @stoplinenumberR}{%
       \csxdef{@msdata@\@msd@cR @stopabslinenumberR}{\the\absline@numR}%
       \csxdef{@msdata@\@msd@cR @stoplinenumberR}{\the\line@numR}%
     }%
     {}%
 \fi%
}%
%
```

XXIX.5 Action macro

\add@msdata

\add@msdata is executed on each line when action code 1010 is seen. It will not insert immediately the manuscript data footnote, as action code are executed before the line be typeset, and, consequently, could be on the previous page. So it just stores the manuscript data footnote to \@msdata@list.

```
\newcommand{\add@msdata}{%
  \bgroup%
  \normalfont%
  \unless\ifledRcol@%
    \numgdef{\add@msd@c}{\add@msd@c+\@ne}%
      \ifcsdef{@msdata@\add@msd@c @data}{%
        \letcs{\@data}{@msdata@\add@msd@c @data}%
        \edef\l@d@nums{%
          000|% Start page = we don't print it
          \csuse{@msdata@\add@msd@c @linenumber}|% Start line number
          000|% Start subline number, for now, not used
          000|% End page number, we don't print it
          \ifnumless{\csuse{@msdata@\add@msd@c @stopabslinenumber}}{\csuse{@lastabsline@forpage@\the\page@num}}%
            {\csuse{@msdata@\add@msd@c @stoplinenumber}}%End line number if in the same page
            {\csuse{@lastline@forpage@\the\page@num}}%Otherwiser, last number of the page
          |%
          000|% End sub line number, for now, not used
          \edfont@info%Font
          }%
          \@msd@options@fullpagefalse%
          \if@firstlineofpage%Try if the data are for the full page. If yes, will add options to the list.
            \unless\if@msdata@insertedfrompreviouspage%
```

```
       \ifnumless{\csuse{@lastabsline@forpage@\the\page@num}}{\csuse{@msdata@\add@msd@c @stopabslinenumber}+\@ne}%
       {%
       \numdef{\@tmp}{\add@msd@c+\@ne}%
       \ifcsdef{@msdata@\@tmp @abslinenumber}%
        {\ifnumequal{\csuse{@msdata@\@tmp @abslinenumber}}{\csuse{@lastabsline@forpage@\the\page@num}}%
         {}%
         {\@msd@options@fullpagetrue}%
        }%
        {\@msd@options@fullpagetrue}%
       }%
      {}%
      \fi%
     \fi%
     \listxadd{\@msdata@list}{%
      \@msd@options@iffullpage%
      \ifluatex%
       \csxdef{footnote@luatextextdir}{\the\textdir}%
       \csxdef{footnote@luatexpardir}{\the\pardir}%
      \fi%
      \csdef{@this@crossref@start}{\add@msd@c:start:msdata}%
       \csdef{@this@crossref@end}{\add@msd@c:end:msdata}%
      \noexpand\csuse{v\@msdata@series footnote}{\@msdata@series}{{\expandonce\l@d@nums}{\ms@data@label}{\expandonce\@data}}%
      \reset@msd@options@iffullpage%
    }%
   }%
   {}%
 \else%
  \numgdef{\add@msd@cR}{\add@msd@cR+\@ne}%
   \ifcsdef{@msdata@\add@msd@cR @dataR}{%
    \letcs{\@data}{@msdata@\add@msd@cR @dataR}%
    \edef\l@d@nums{%
     000|% Start page = we don't print it
     \csuse{@msdata@\add@msd@cR @linenumberR}|% Start line number
     000|% Start subline number, for now, not used
     000|% End page number, we don't print it
     \ifnumless{\csuse{@msdata@\add@msd@cR @stopabslinenumberR}}{\csuse{@lastline@forpageR@\the\page@numR}}%
      {\csuse{@msdata@\add@msd@cR @stoplinenumberR}}%End line number if in the same page
      {\csuse{@lastline@forpageR@\the\page@numR}}%Otherwiser, last number of the page
     |%
     000|% End sub line number, for now, not used
     \edfont@info%Font
     }%
     \@msd@options@fullpagefalse%
     \if@firstlineofpageR%
```

```
            \unless\if@msdata@insertedfrompreviouspage%
              \ifnumless{\csuse{@lastabsline@forpageR@\the\page@numR}}{\
csuse{@msdata@\add@msd@c @stopabslinenumberR}+\@ne}%
              {%
              \numdef{\@tmp}{\add@msd@cR+\@ne}%
              \ifcsdef{@msdata@\@tmp @abslinenumberR}%
                {\ifnumequal{\csuse{@msdata@\@tmp @abslinenumberR}}{\csuse{
@lastabsline@forpageR@\the\page@numR}}%
                  {}%
                  {\@msd@options@fullpagetrue}%
                }%
                {\@msd@options@fullpagetrue}%
              }%
             {}%
            \fi%
          \fi%
          \listxadd{\@msdata@list}{%
            \@msd@options@iffullpage%
            \ifluatex%
              \csxdef{footnote@luatextextdir}{\the\textdir}%
              \csxdef{footnote@luatexpardir}{\the\pardir}%
            \fi%
            \csdef{@this@crossref@start}{\add@msd@cR:start:msdata}%
             \csdef{@this@crossref@end}{\add@msd@cR:end:msdata}%
            \noexpand\csuse{v\@msdata@series footnote}{\@msdata@series}{{\
expandonce\l@d@nums}{\ms@data@label}{\expandonce\@data}}%
            \reset@msd@options@iffullpage%
         }%
      }%
      {}%
  \fi%
  \egroup%
}%
%
```

\if@msdata@insertedfrompreviouspage

The \if@msdata@insertedfrompreviouspage boolean is set to TRUE if reledmac automatically inserts data from previous page in the first line of a page.

```
\newif\if@msdata@insertedfrompreviouspage%
%
```

\add@msdata@firstlineofpage

\add@msdata@firstlineofpage is called at the first line of every page. It inserts manuscript data which start on one of the previous pages and continue on this page.

```
\newcommand{\add@msdata@firstlineofpage}{%
  \bgroup%
  \normalfont%
  \unless\ifledRcol@%
    \ifcsdef{@msdata@\add@msd@c @data}{%
    \ifnumless{\the\absline@num-\@ne}{\csuse{@msdata@\add@msd@c @stopabslinenumber}}%
      {%
      \global\@msdata@insertedfrompreviouspagetrue%
      \letcs{\@data}{@msdata@\add@msd@c @data}%
      \edef\l@d@nums{%
          000|% Start page = we don't print it
          \numexpr\the\line@num+\@ne\relax|% Start line number = first line of the page. As \add@msdata@firstlineofpage is called before line number has been incremented, we increment it for printing
          000|% Start subline number, for now, not used
          000|% End page number, we don't print it
          \ifnumless{\csuse{@msdata@\add@msd@c @stopabslinenumber}}{\csuse{@lastabsline@forpage@\the\page@num}}%
            {\csuse{@msdata@\add@msd@c @stoplinenumber}}%End line number if in the same page
            {\csuse{@lastline@forpage@\the\page@num}}%Otherwise, last number of the page
          |%
          000|% End sub line number, for now, not used
          \edfont@info%Font
          }%
          \@msd@options@fullpagefalse%
          \ifnumless{\csuse{@lastabsline@forpage@\the\page@num}}{\csuse{@msdata@\add@msd@c @stopabslinenumber}+\@ne}%We will test if the ms data is for the full page
          {%
          \numdef{\@tmp}{\add@msd@c+\@ne}%
          \ifcsdef{@msdata@\@tmp @abslinenumber}%
            {\ifnumequal{\csuse{@msdata@\@tmp @abslinenumber}}{\csuse{@lastabsline@forpage@\the\page@num}}%
              {}%
              {\@msd@options@fullpagetrue}%
            }%
            {\@msd@options@fullpagetrue}%
          }%
         {}%
         \listxadd{\@msdata@list}{%
          \@msd@options@iffullpage%
          \ifluatex%
            \csxdef{footnote@luatextextdir}{\the\textdir}%
            \csxdef{footnote@luatexpardir}{\the\pardir}%
          \fi%
          \csdef{@this@crossref@start}{\add@msd@c:start:msdata}%
           \csdef{@this@crossref@end}{\add@msd@c:end:msdata}%
          \noexpand\csuse{v\@msdata@series footnote}{\@msdata@series}{{\expandonce\l@d@nums}{\ms@data@label}{\expandonce\@data}}%
          \reset@msd@options@iffullpage%
        }%
      }%
      {\global\@msdata@insertedfrompreviouspagefalse}%
  }{}%
  \else%
    \ifcsdef{@msdata@\add@msd@cR @dataR}{%
    \ifnumless{\the\absline@numR-\@ne}{\csuse{@msdata@\add@msd@cR @stopabslinenumberR}}%
          {%
          \global\@msdata@insertedfrompreviouspagetrue%
          \letcs{\@data}{@msdata@\add@msd@cR @dataR}%
          \edef\l@d@nums{%
          000|% Start page = we don't print it
          \numexpr\the\line@numR+\@ne\relax|% Start line number = first line of the page. As \add@msdata@firstlineofpage is called before line number has been incremented, we increment it for printing
          000|% Start subline number, for now, not used
          000|% End page number, we don't print it
          \ifnumless{\csuse{@msdata@\add@msd@cR @stopabslinenumberR}}{\csuse{@lastline@forpageR@\the\page@numR}}%
            {\csuse{@msdata@\add@msd@cR @stoplinenumberR}}%End line number if in the same page
            {\csuse{@lastline@forpageR@\the\page@numR}}%Otherwise, last number of the page
          |%
          000|% End sub line number, for now, not used
          \edfont@info%Font
          }%
          \@msd@options@fullpagefalse%
          \ifnumless{\csuse{@lastabsline@forpageR@\the\page@numR}}{\csuse{@msdata@\add@msd@cR @stopabslinenumberR}+\@ne}%
          {%
          \numdef{\@tmp}{\add@msd@cR+\@ne}%
          \ifcsdef{@msdata@\@tmp @abslinenumberR}%
            {\ifnumequal{\csuse{@msdata@\@tmp @abslinenumberR}}{\csuse{@lastabsline@forpageR@\the\page@numR}}%
              {}%
              {\@msd@options@fullpagetrue}%
            }%
            {\@msd@options@fullpagetrue}%
          }%
         {}%
         \listxadd{\@msdata@list}{%
           \@msd@options@iffullpage%
           \ifluatex%
             \csxdef{footnote@luatextextdir}{\the\textdir}%
             \csxdef{footnote@luatexpardir}{\the\pardir}%
           \fi%
          \csdef{@this@crossref@start}{\add@msd@cR:start:msdata}%
           \csdef{@this@crossref@end}{\add@msd@cR:end:msdata}%
           \noexpand\csuse{v\@msdata@series footnote}{\@msdata@series}{{\expandonce\l@d@nums}{\ms@data@label}{\expandonce\@data}}%
          \reset@msd@options@iffullpage%
       }%
    }%
    {\global\@msdata@insertedfrompreviouspagefalse}%
  }{}%
  \fi%
  \egroup%
}%
%
```

XXIX.6 Inserting footnote

Just before inserting standard insert (familiar and critical footnotes, sidenotes), we call `\insert@msdata` to insert manuscripts data's footnotes.

`\insert@msdata`

```
\newcommand{\insert@msdata}{%
  \def\do##1{##1}%
  \dolistloop{\@msdata@list}%
  \global\let\@msdata@list\relax%
}%
%
```

XXIX.7 Other

`\@msd@options@iffullpage`
`et@msd@options@iffullpage`

`\@msd@options@iffullpage` sets some options if the manuscripts data are for all the page. `\reset@msd@options@iffullpage` resets them after the footnote. `\if@msd@options@fullpage` is switch to true in add@msdata@firstlineofpage if these option must be inserted.

```
\newif\if@msd@options@fullpage%
\newcommand{\@msd@options@iffullpage}[0]{%
  \if@msd@options@fullpage%
    \noexpand\toggletrue{nonum@}%
    \ifdefvoid{\ms@data@label}%
      {\noexpand\toggletrue{nosep@}}%
      {}%
  \fi%
}%
\newcommand{\reset@msd@options@iffullpage}[0]{%
  \noexpand\togglefalse{nonum@}%
  \noexpand\togglefalse{nosep@}%
}%
%
```

XXX Arrays and tables

XXX.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.

```
\newtoks\@emptytoks

%
```

The rest is from `amsmath`.

`\l@denvbody` A token register to contain the body.

```
\newtoks\l@denvbody

%
```

`\addtol@denvbody` `\addtol@denvdody{arg}` adds `arg` to the token register `\l@denvbody`.

```
\newcommand{\addtol@denvbody}[1]{%
  \global\l@denvbody\expandafter{\the\l@denvbody#1}}

%
```

`\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`.

```
\newcommand{\l@dcollect@body}[1]{%
  \l@denvbody{\expandafter#1\expandafter{\the\l@denvbody}}%
  \edef\processl@denvbody{\the\l@denvbody\noexpand\end{\@currenvir}}%
  \l@denvbody\@emptytoks \def\l@dbegin@stack{b}%
  \begingroup
    \expandafter\let\csname\@currenvir\endcsname\l@dcollect@@body
    \edef\processl@denvbody{\expandafter\noexpand\csname\@currenvir\
endcsname}%
    \processl@denvbody%
    }%

%
```

\l@dpush@begins

When adding a piece of the current environment's contents to \l@denvbody, we scan it to check for additional \begin tokens, and add a 'b' to the stack for any that we find.

```
\def\l@dpush@begins#1\begin#2{%
  \ifx\end#2\else b\expandafter\l@dpush@begins\fi}

%
```

\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.

```
\def\l@dcollect@@body#1\end#2{%
  \edef\l@dbegin@stack{\l@dpush@begins#1\begin\end
                       \expandafter\@gobble\l@dbegin@stack}%
  \ifx\@empty\l@dbegin@stack
    \endgroup
    \@checkend{#2}%
    \addtol@denvbody{#1}%
  \else
    \addtol@denvbody{#1\end{#2}}%
  \fi
  \processl@denvbody % A little tricky! Note the grouping
}

%
```

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>
```

XXX.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`.

XXX.2.1 Disabling and restoring commands

`\l@dtabnoexpands` More no expansion for critical and familiar footnotes in tabular environment.

```
\newcommand*{\l@dtabnoexpands}{%
  \let\rtab=0%
  \let\ctab=0%
  \let\ltab=0%
  \let\rtabtext=0%
  \let\ltabtext=0%
  \let\ctabtext=0%
  \let\edbeforetab=0%
  \let\edaftertab=0%
  \let\edatleft=0%
  \let\edatright=0%
  \let\edvertline=0%
  \let\edvertdots=0%
  \let\edrowfill=0%
}

%
```

\disable@familiarnotes
\restore@familiarnotes

Macros to disable and restore familiar notes, to prevent them from printing multiple times in edtabularx and edarrayx environments.

```
\newcommand{\disable@familiarnotes}{%
  \unless\ifnofamiliar@%
    \def\do##1{%
        \csletcs{footnote@@##1}{footnote##1}%
        \expandafter\renewcommand \csname footnote##1\endcsname[1]{%
          \protected@csxdef{@thefnmark##1}{\csuse{thefootnote##1}}%
          \csuse{@footnotemark##1}%
        }%
    }%
    \dolistloop{\@series}%
  \fi%
}%
\newcommand{\restore@familiarnotes}{%
  \unless\ifnofamiliar@%
    \def\do##1{%
        \csletcs{footnote##1}{footnote@@##1}%
    }%
    \dolistloop{\@series}%
  \fi%
}%

%
```

\disable@sidenotes
\restore@sidenotes

The sames, for side notes.

```
\newcommand{\disable@sidenotes}{%
  \let\@@ledrightnote\ledrightnote%
  \let\@@ledleftnote\ledleftnote%
  \let\@@ledsidenote\ledsidenote%
  \let\ledrightnote\@gobble%
  \let\ledleftnote\@gobble%
  \let\ledsidenote\@gobble%
}%
\newcommand{\restore@sidenotes}{%
  \let\ledrightnote\@@ledrightnote%
  \let\ledleftnote\@@ledleftnote%
  \let\ledsidenote\@@ledsidenote%
}%
%
```

\disable@notes
\restore@notes

Disable/restore side and familiar notes.

```
\newcommand{\disable@notes}{%
  \disable@sidenotes%
  \disable@familiarnotes%
}%
\newcommand{\restore@notes}{%
  \restore@sidenotes%
  \restore@familiarnotes%
}%
%
```

\EDTEXT
\xedtext

We need to be able to modify the `\edtext` macros and also restore their original definitions.

```
\let\EDTEXT=\edtext
\newcommand{\xedtext}[2]{\EDTEXT{#1}{#2}}
%
```

\EDLABEL
\xedlabel

We need to be able to modify and restore the `\edlabel` macro.

```
\let\EDLABEL=\edlabel
\newcommand*{\xedlabel}[1]{\EDLABEL{#1}}
%
```

\xedindex
\nulledindex

Macros supporting modification and restoration of `\edindex`.

```
\AtBeginDocument{\let\xedindex\edindex}%
\newcommand{\nulledindex}[2][\jobname]{\@bsphack\@esphack}

%
```

\@line@@num

Macro supporting restoration of `\linenum`.

```
\let\@line@@num=\linenum
%
```

\l@dgobblearg

`\l@dgobbleoptarg[`⟨*arg*⟩`]{`⟨*arg*⟩`}` replaces these two arguments (first is optional) by `\relax`.

```
\newcommand*{\l@dgobbleoptarg}[2][]{\relax}%

%
```

\Relax
\NEXT

```
\let\Relax=\relax
\let\NEXT=\next

%
```

\l@dmodforedtext
\l@drestoreforedtext

Modify and restore various macros for when \edtext is used.

```
\newcommand{\l@dmodforedtext}{%
  \let\edtext\relax
  \def\do##1{\global\csletcs{##1footnote}{l@dgobbleoptarg}}%
  \dolistloop{\@series}%
  \let\edindex\nulledindex
  \let\linenum\@gobble}
\newcommand{\l@drestoreforedtext}{%
  \def\do##1{\global\csletcs{##1footnote}{##1@@footnote}}
  \dolistloop{\@series}%
  \let\edindex\xedindex}
%
```

\l@dnullfills
\l@drestorefills

Nullify and restore some column fillers, etc.

```
\newcommand{\l@dnullfills}{%
  \def\edlabel##1{}%
  \def\edrowfill##1##2##3{}%
}
\newcommand{\l@drestorefills}{%
  \def\edrowfill##1##2##3{\@EDROWFILL@{##1}{##2}{##3}}%
}

%
```

\letsforverteilen

Gathers some lets and other code that is common to the *verteilen* macros.

```
\newcommand{\letsforverteilen}{%
  \let\edtext\xedtext
  \let\edindex\xedindex
  \def\do##1{\global\csletcs{##1footnote}{##1@@footnote}}
  \dolistloop{\@series}%
  \let\linenum\@line@@num
  \hilfsskip=\l@dcolwidth%
  \advance\hilfsskip by -\wd\hilfsbox
  \def\edlabel##1{\xedlabel{##1}}}

%
```

\disablel@dtabfeet
\enablel@dtabfeet

Declarations for using or using \edtext inside tabulars. The default at this point is for \edtext.

```
\newcommand\disablel@dtabfeet{\l@dmodforedtext}%
\newcommand\enablel@dtabfeet{\l@drestoreforedtext}%
%
```

XXX.2.2 Counters, boxes and lengths

\l@dampcount
\l@dcolcount

\l@dampcount is a counter for the & column dividers and \l@dcolcount is a counter for the columns.

```
\newcount\l@dampcount
  \l@dampcount=1\relax
\newcount\l@dcolcount
  \l@dcolcount=0\relax

%
```

\hilfsbox
\hilfsskip
\Hilfsbox
\hilfscount

Some (temporary) helper items.

```
\newbox\hilfsbox
\newskip\hilfsskip
\newbox\Hilfsbox
\newcount\hilfscount

%
```

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).

```
\newdimen\dcoli
\newdimen\dcolii
\newdimen\dcoliii
\newdimen\dcoliv
\newdimen\dcolv
\newdimen\dcolvi
\newdimen\dcolvii
\newdimen\dcolviii
\newdimen\dcolix
\newdimen\dcolx
\newdimen\dcolxi
\newdimen\dcolxii
\newdimen\dcolxiii
\newdimen\dcolxiv
\newdimen\dcolxv
\newdimen\dcolxvi
\newdimen\dcolxvii
\newdimen\dcolxviii
\newdimen\dcolxix
\newdimen\dcolxx
\newdimen\dcolxxi
\newdimen\dcolxxii
\newdimen\dcolxxiii
\newdimen\dcolxxiv
\newdimen\dcolxxv
\newdimen\dcolxxvi
\newdimen\dcolxxvii
```

```
\newdimen\dcolxxviii
\newdimen\dcolxxix
\newdimen\dcolxxx
\newdimen\dcolerr   % added for error handling

%
```

\l@dcolwidth This is a cunning way of storing the columnwidths indexed by the column number \l@dcolcount, like an array. (was \Dimenzuordnung)

```
\newcommand{\l@dcolwidth}{\ifcase \the\l@dcolcount \dcoli %???
  \or \dcoli \or \dcolii \or \dcoliii
  \or \dcoliv \or \dcolv \or \dcolvi
  \or \dcolvii \or \dcolviii \or \dcolix \or \dcolx
  \or \dcolxi \or \dcolxii \or \dcolxiii
  \or \dcolxiv \or \dcolxv \or \dcolxvi
  \or \dcolxvii \or \dcolxviii \or \dcolxix \or \dcolxx
  \or \dcolxxi \or \dcolxxii \or \dcolxxiii
  \or \dcolxxiv \or \dcolxxv \or \dcolxxvi
  \or \dcolxxvii \or \dcolxxviii \or \dcolxxix \or \dcolxxx
  \else \dcolerr \fi}

%
```

\stepl@dcolcount This increments the column counter, and issues an error message if it is too large.

```
\newcommand*{\stepl@dcolcount}{\advance\l@dcolcount\@ne
  \ifnum\l@dcolcount>30\relax
    \led@err@TooManyColumns
  \fi}

%
```

\l@dsetmaxcolwidth Sets the column width to the maximum value seen so far.

```
\newcommand{\l@dsetmaxcolwidth}{%
  \ifdim\l@dcolwidth < \wd\hilfsbox
    \l@dcolwidth = \wd\hilfsbox
  \else \relax \fi}

%
```

\measuremcell Measure (recursively) the width required for a math cell.

```
\def\measuremcell #1&{%
    \ifx #1\\ \ifnum\l@dcolcount=0\let\NEXT\relax%
              \else\l@dcheckcols%
                    \l@dcolcount=0%
                    \let\NEXT\measuremcell%
              \fi%
    \else\setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
      \stepl@dcolcount%
      \l@dsetmaxcolwidth%
      \let\NEXT\measuremcell%
    \fi\NEXT}

%
```

\measuretcell

Measure (recursively) the width required for a text cell.

```
\def\measuretcell #1&{%
    \ifx #1\\ \ifnum\l@dcolcount=0\let\NEXT\relax%
              \else\l@dcheckcols%
                    \l@dcolcount=0%
                    \let\NEXT\measuretcell%
              \fi%
    \else\setbox\hilfsbox=\hbox{#1}%
      \stepl@dcolcount%
      \l@dsetmaxcolwidth%
      \let\NEXT\measuretcell%
    \fi\NEXT}

%
```

\measuremrow

Measure (recursively) the width required for a math row.

```
\def\measuremrow #1\\{%
   \ifx #1&\let\NEXT\relax%
   \else\measuremcell #1&\\&\\&%
      \let\NEXT\measuremrow%
   \fi\NEXT}
%
```

\measuretrow

Measure (recursively) the width required for a text row.

```
\def\measuretrow #1\\{%
   \ifx #1&\let\NEXT\relax%
   \else\measuretcell #1&\\&\\&%
     \let\NEXT\measuretrow%
   \fi\NEXT}

%
```

\edtabcolsep

The length \edtabcolsep controls the distance between columns.

```
\newskip\edtabcolsep
\global\edtabcolsep=10pt

%
```

\variab

```
\newcommand{\variab}{\relax}

%
```

\l@dcheckcols

Check that the number of columns is consistent.

```
\newcommand*{\l@dcheckcols}{%
  \ifnum\l@dcolcount=1\relax
  \else
    \ifnum\l@dampcount=1\relax
    \else
      \ifnum\l@dcolcount=\l@dampcount\relax
      \else
        \l@d@err@UnequalColumns
      \fi
    \fi
    \l@dampcount=\l@dcolcount
  \fi}

%
```

\edfilldimen

A length.

```
\newdimen\edfilldimen
\edfilldimen=0pt

%
```

\c@addcolcount
\theaddcolcount

A counter to hold the number of a column. We use a roman number so that we can grab the column dimension from `\dcol`. We do not use the `\roman` LaTeX command, because some packages, like `babel` can override it in some specific cases (Greek, for example).

```
\newcounter{addcolcount}
  \renewcommand{\theaddcolcount}{\romannumeral \c@addcolcount}
%
```

XXX.2.3 Tabular typesetting

\setmcellright

Typeset (recursively) cells of display math right justified.

```
\def\setmcellright #1&{\def\edlabel##1{}%
       \let\edindex\nulledindex
       \ifx #1\\ \ifnum\l@dcolcount=0%\removelastskip
                     \let\Next\relax%
                \else\l@dcolcount=0%
                      \let\Next=\setmcellright%
                \fi%
       \else%
           \disablel@dtabfeet%
           \stepl@dcolcount%
           \disable@notes%
           \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
           \restore@notes%
           \letsforverteilen%
           \hskip\hilfsskip$\displaystyle{#1}$%
           \hskip\edtabcolsep%
           \let\Next=\setmcellright%
       \fi\Next}

%
```

\settcellright Typeset (recursively) cells of text right justified.

```
\def\settcellright #1&{\def\edlabel##1{}%
       \let\edindex\nulledindex
       \ifx #1\\ \ifnum\l@dcolcount=0%\removelastskip
                    \let\Next\relax%
                \else\l@dcolcount=0%
                      \let\Next=\settcellright%
                \fi%
       \else%
           \disablel@dtabfeet%
           \stepl@dcolcount%
           \disable@notes%
           \setbox\hilfsbox=\hbox{#1}%
           \restore@notes%
           \letsforverteilen%
           \hskip\hilfsskip#1%
           \hskip\edtabcolsep%
           \let\Next=\settcellright%
       \fi\Next}
%
```

\setmcellleft Typeset (recursively) cells of display math left justified.

```
\def\setmcellleft #1&{\def\edlabel##1{}%
    \let\edindex\nulledindex
    \ifx #1\\ \ifnum\l@dcolcount=0 \let\Next\relax%
              \else\l@dcolcount=0%
                    \let\Next=\setmcellleft%
              \fi%
     \else   \disablel@dtabfeet%
              \stepl@dcolcount%
              \disable@notes%
              \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
              \restore@notes%
              \letsforverteilen%
              $\displaystyle{#1}$\hskip\hilfsskip\hskip\edtabcolsep%
              \let\Next=\setmcellleft%
     \fi\Next}

%
```

\settcellleft Typeset (recursively) cells of text left justified.

```
\def\settcellleft #1&{\def\edlabel##1{}%
    \let\edindex\nulledindex
    \ifx #1\\ \ifnum\l@dcolcount=0 \let\Next\relax%
              \else\l@dcolcount=0%
                    \let\Next=\settcellleft%
              \fi%
     \else   \disablel@dtabfeet%
              \stepl@dcolcount%
              \disable@notes%
              \setbox\hilfsbox=\hbox{#1}%
              \restore@notes%
              \letsforverteilen%
              #1\hskip\hilfsskip\hskip\edtabcolsep%
              \let\Next=\settcellleft%
     \fi\Next}
%
```

\setmcellcenter Typeset (recursively) cells of display math centered.

```
\def\setmcellcenter #1&{\def\edlabel##1{}%
    \let\edindex\nulledindex
    \ifx #1\\ \ifnum\l@dcolcount=0\let\Next\relax%
             \else\l@dcolcount=0%
                   \let\Next=\setmcellcenter%
             \fi%
    \else    \disablel@dtabfeet%
             \stepl@dcolcount%
             \disable@notes%
             \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
             \restore@notes%
             \letsforverteilen%
             \hskip 0.5\hilfsskip$\displaystyle{#1}$\hskip0.5\hilfsskip%
             \hskip\edtabcolsep%
             \let\Next=\setmcellcenter%
     \fi\Next}

%
```

\settcellcenter Typeset (recursively) cells of text centered.

```
\def\settcellcenter #1&{\def\edlabel##1{}%
    \let\edindex\nulledindex
    \ifx #1\\ \ifnum\l@dcolcount=0 \let\Next\relax%
              \else\l@dcolcount=0%
```

```
                    \let\Next=\settcellcenter%
              \fi%
     \else   \disablel@dtabfeet%
              \stepl@dcolcount%
              \disable@notes%
              \setbox\hilfsbox=\hbox{#1}%
              \restore@notes%
              \letsforverteilen%
              \hskip 0.5\hilfsskip #1\hskip 0.5\hilfsskip%
              \hskip\edtabcolsep%
              \let\Next=\settcellcenter%
     \fi\Next}

%
```

\NEXT

```
\let\NEXT=\relax

%
```

\setmrowright Typeset (recursively) rows of right justified math.

```
\def\setmrowright #1\\{%
    \ifx #1& \let\NEXT\relax
    \else \centerline{\setmcellright #1&\\&\\&}
          \let\NEXT=\setmrowright
    \fi\NEXT}
%
```

\settrowright Typeset (recursively) rows of right justified text.

```
\def\settrowright #1\\{%
    \ifx #1& \let\NEXT\relax
    \else \centerline{\settcellright #1&\\&\\&}
          \let\NEXT=\settrowright
    \fi\NEXT}

%
```

\setmrowleft Typeset (recursively) rows of left justified math.

```
\def\setmrowleft #1\\{%
    \ifx #1&\let\NEXT\relax
    \else \centerline{\setmcellleft #1&\\&\\&}
          \let\NEXT=\setmrowleft
    \fi\NEXT}
%
```

\settrowleft Typeset (recursively) rows of left justified text.

```
\def\settrowleft #1\\{%
    \ifx #1& \let\NEXT\relax
    \else \centerline{\settcellleft #1&\\&\\&}
          \let\NEXT=\settrowleft
    \fi\NEXT}

%
```

\setmrowcenter

Typeset (recursively) rows of centered math.

```
\def\setmrowcenter #1\\{%
    \ifx #1& \let\NEXT\relax%
    \else \centerline{\setmcellcenter #1&\\&\\&}
          \let\NEXT=\setmrowcenter
     \fi\NEXT}
%
```

\settrowcenter

Typeset (recursively) rows of centered text.

```
\def\settrowcenter #1\\{%
    \ifx #1& \let\NEXT\relax
    \else \centerline{\settcellcenter #1&\\&\\&}
          \let\NEXT=\settrowcenter
    \fi\NEXT}

%
```

\nullsetzen

```
\newcommand{\nullsetzen}{%
    \stepl@dcolcount%
    \l@dcolwidth=0pt%
    \ifnum\l@dcolcount=30\let\NEXT\relax%
          \l@dcolcount=0\relax
    \else\let\NEXT\nullsetzen%
    \fi\NEXT}

%
```

\edatleft

\edatleft[⟨*math*⟩]{⟨*symbol*⟩}{⟨*len*⟩}. Left ⟨*symbol*⟩, 2⟨*len*⟩ high with prepended ⟨*math*⟩ vertically centered.

```
\newcommand{\edatleft}[3][\@empty]{%
  \ifx#1\@empty
    \vbox to 10pt{\vss\hbox{$\left#2\vrule width0pt height #3
                           depth 0pt \right. $\hss}\vfil}
  \else
    \vbox to 4pt{\vss\hbox{$#1\left#2\vrule width0pt height #3
                depth 0pt \right. $}\vfil}
  \fi}
%
```

\edatright

\edatright[⟨*math*⟩]{⟨*symbol*⟩}{⟨*len*⟩}. Right ⟨*symbol*⟩, 2⟨*len*⟩ high with appended ⟨*math*⟩ vertically centered.

```
\newcommand{\edatright}[3][\@empty]{%
  \ifx#1\@empty
    \vbox to 10pt{\vss\hbox{$\left.\vrule width0pt height #3
                depth 0pt \right#2 $\hss}\vfil}
  \else
    \vbox to 4pt{\vss\hbox{$\left.\vrule width0pt height #3
                depth 0pt \right#2 #1 $}\vfil}
  \fi}

%
```

\edvertline

\edvertline{⟨*len*⟩} vertical line ⟨*len*⟩ high.

```
\newcommand{\edvertline}[1]{\vbox to 8pt{\vss\hbox{\vrule height #1}\vfil}}

%
```

\edvertdots

\edvertdots{⟨*len*⟩} vertical dotted line ⟨*len*⟩ high.

```
\newcommand{\edvertdots}[1]{\vbox to 1pt{\vss\vbox to #1%
         {\cleaders\hbox{$\m@th\hbox{.}\vbox to 0.5em{ }$}\vfil}}}

%
```

\l@dtabaddcols

\l@dtabaddcols{⟨*startcol*⟩}{⟨*endcol*⟩} adds the widths of the columns ⟨*startcol*⟩ through ⟨*endcol*⟩ to \edfilldimen. It is a LaTeX style reimplementation of the original \@add@.

```
\newcommand{\l@dtabaddcols}[2]{%
  \l@dcheckstartend{#1}{#2}%
  \ifl@dstartendok
  \setcounter{addcolcount}{#1}%
  \@whilenum \value{addcolcount}<#2\relax \do
  {\advance\edfilldimen by \the \csname dcol\theaddcolcount\endcsname
   \advance\edfilldimen by \edtabcolsep
   \stepcounter{addcolcount}}%
  \advance\edfilldimen by \the \csname dcol\theaddcolcount\endcsname
  \fi
}

%
```

\ifl@dstartendok
\l@dcheckstartend

\l@dcheckstartend{⟨*startcol*⟩}{⟨*endcol*⟩} checks that the values of ⟨*startcol*⟩ and ⟨*endcol*⟩ are sensible. If they are then \ifl@dstartendok is set TRUE, otherwise it is set FALSE.

```
\newif\ifl@dstartendok
\newcommand{\l@dcheckstartend}[2]{%
```

```
  \l@dstartendoktrue
  \ifnum #1<\@ne
    \l@dstartendokfalse
    \led@err@LowStartColumn
  \fi
  \ifnum #2>30\relax
    \l@dstartendokfalse
    \led@err@HighEndColumn
  \fi
  \ifnum #1>#2\relax
    \l@dstartendokfalse
    \led@err@ReverseColumns
  \fi
}

%
```

\edrowfill
\@edrowfill@
\@EDROWFILL@

\edrowfill{⟨*startcol*⟩}{⟨*endcol*⟩}fill fills columns ⟨*startcol*⟩ to ⟨*endcol*⟩ inclusive with ⟨*fill*⟩ (e.g. \hrulefill, \upbracefill). This is a LaTeX style reimplementation and generalization of the original \waklam, \Waklam, \waklamec, \wastricht and \wapunktel macros.

```
\newcommand*{\edrowfill}[3]{%
  \l@dtabaddcols{#1}{#2}%
  \hb@xt@ \the\l@dcolwidth{\hb@xt@ \the\edfilldimen{#3}\hss}}
\let\@edrowfill@=\edrowfill
\def\@EDROWFILL@#1#2#3{\@edrowfill@{#1}{#2}{#3}}

%
```

\edbeforetab
\edaftertab

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.

```
\newcommand{\leftltab}[1]{%
  \hb@xt@\z@{\vbox{\edtabindent%
  \moveleft\Hilfsskip\hbox{\ #1}}\hss}}

%
```

\leftrtab

\leftrtab{⟨*text*⟩}{⟨*math*⟩} for \edbeforetab in \rtab.

```
\newcommand{\leftrtab}[2]{%
  #2\hb@xt@\z@{\vbox{\edtabindent%
    \advance\Hilfsskip by\dcoli%
    \moveleft\Hilfsskip\hbox{\ #1}}\hss}}

%
```

\leftctab \leftctab{⟨*text*⟩}{⟨*math*⟩} for \edbeforetab in \ctab.

```
\newcommand{\leftctab}[2]{%
        \hb@xt@\z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
        \advance\Hilfsskip by 0.5\dcoli%
        \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
        \disablel@dtabfeet$\displaystyle{#2}$}%
        \advance\Hilfsskip by -0.5\wd\hilfsbox%
        \moveleft\Hilfsskip\hbox{\ #1}}\hss}%
        #2}

%
```

\rightctab \rightctab{⟨*math*⟩}{⟨*text*⟩} for \edaftertab in \ctab.

```
\newcommand{\rightctab}[2]{%
        \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
        \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
        #1\hb@xt@\z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
        \advance\Hilfsskip by 0.5\l@dcolwidth%
        \advance\Hilfsskip by -\wd\hilfsbox%
        \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
        \disablel@dtabfeet$\displaystyle{#1}$}%
        \advance\Hilfsskip by -0.5\wd\hilfsbox%
        \advance\Hilfsskip by \edtabcolsep%
        \moveright\Hilfsskip\hbox{ #2}}\hss}%
        }

%
```

\rightltab \rightltab{⟨*math*⟩}{⟨*text*⟩} for \edaftertab in \ltab.

```
\newcommand{\rightltab}[2]{%
        \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
        \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
        #1\hb@xt@\z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
        \advance\Hilfsskip by\l@dcolwidth%
        \advance\Hilfsskip by-\wd\hilfsbox%
        \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
        \disablel@dtabfeet$\displaystyle{#1}$}%
        \advance\Hilfsskip by-\wd\hilfsbox%
        \advance\Hilfsskip by\edtabcolsep%
        \moveright\Hilfsskip\hbox{ #2}}\hss}%
        }

%
```

\rightrtab \rightrtab{⟨*math*⟩}{⟨*text*⟩} for \edaftertab in \rtab.

```
\newcommand{\rightrtab}[2]{%
```

```
        \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
        \disablel@dtabfeet#2}%
        #1\hb@xt@\z@{\vbox{\edtabindent%
        \advance\Hilfsskip by-\wd\hilfsbox%
        \advance\Hilfsskip by\edtabcolsep%
        \moveright\Hilfsskip\hbox{ #2}}\hss}%
        }

%
```

\rtab
\edbeforetab
\edaftertab

\rtab{⟨*body*⟩} typesets ⟨*body*⟩ as an array with the entries right justified.

The process is first to measure the ⟨*body*⟩ to get the column widths, and then in a second pass to typeset the body.

```
\newcommand{\rtab}[1]{%
  \l@dnullfills
    \def\edbeforetab##1##2{\leftrtab{##1}{##2}}%
    \def\edaftertab##1##2{\rightrtab{##1}{##2}}%
    \measurembody{#1}%
  \l@drestorefills
    \variab
    \setmrowright #1\\&\\%
    \enablel@dtabfeet}

%
```

\measurembody

\measurembody{⟨*body*⟩} measures the array ⟨*body*⟩.

```
\newcommand{\measurembody}[1]{%
  \disablel@dtabfeet%
  \l@dcolcount=0%
  \nullsetzen%
  \l@dcolcount=0
  \measuremrow #1\\&\\%
  \global\l@dampcount=1}

%
```

\rtabtext

\rtabtext{⟨*body*⟩} typesets ⟨*body*⟩ as a tabular with the entries right justified.

```
\newcommand{\rtabtext}[1]{%
 \l@dnullfills
    \measuretbody{#1}%
 \l@drestorefills
    \variab
    \settrowright #1\\&\\%
    \enablel@dtabfeet}

%
```

\measuretbody

\measuretbody{⟨*body*⟩} measures the tabular ⟨*body*⟩.

```
\newcommand{\measuretbody}[1]{%
  \disable@notes%
  \disablel@dtabfeet%
  \l@dcolcount=0%
  \nullsetzen%
  \l@dcolcount=0
  \measuretrow #1\\&\\%
  \restore@notes%
  \global\l@dampcount=1}

%
```

\ltab
\edbeforetab
\edaftertab

Array with entries left justified.

```
\newcommand{\ltab}[1]{%
 \l@dnullfills
    \def\edbeforetab##1##2{\leftltab{##1}{##2}}%
    \def\edaftertab##1##2{\rightltab{##1}{##2}}%
    \measurembody{#1}%
  \l@drestorefills
    \variab
    \setmrowleft #1\\&\\%
    \enablel@dtabfeet}

%
```

\ltabtext

Tabular with entries left justified.

```
\newcommand{\ltabtext}[1]{%
  \l@dnullfills
    \measuretbody{#1}%
  \l@drestorefills
    \variab
    \settrowleft #1\\&\\%
    \enablel@dtabfeet}

%
```

\ctab
\edbeforetab
\edaftertab

Array with centered entries.

```
\newcommand{\ctab}[1]{%
  \l@dnullfills
    \def\edbeforetab##1##2{\leftctab{##1}{##2}}%
    \def\edaftertab##1##2{\rightctab{##1}{##2}}%
    \measurembody{#1}%
  \l@drestorefills
    \variab
    \setmrowcenter #1\\&\\%
    \enablel@dtabfeet}

%
```

\ctabtext

Tabular with entries centered.

```
\newcommand{\ctabtext}[1]{%
  \l@dnullfills
    \measuretbody{#1}%
  \l@drestorefills
    \variab
    \settrowcenter #1\\&\\%
    \enablel@dtabfeet}

%
```

\spreadtext

```
\newcommand{\spreadtext}[1]{%\l@dcolcount=\l@dampcount%
   \hb@xt@ \the\l@dcolwidth{\hbox{#1}\hss}}
%
```

\spreadmath

```
\newcommand{\spreadmath}[1]{%
  \hb@xt@ \the\l@dcolwidth{\hbox{$\displaystyle{#1}$}\hss}}

%
```

\HILFSskip
\Hilfsskip

More helpers.

```
\newskip\HILFSskip
\newskip\Hilfsskip

%
```

\EDTABINDENT

```
\newcommand{\EDTABINDENT}{%
    \ifnum\l@dcolcount=30\let\NEXT\relax\l@dcolcount=0%
    \else\stepl@dcolcount%
         \advance\Hilfsskip by\l@dcolwidth%
         \ifdim\l@dcolwidth=0pt\advance\hilfscount\@ne
         \else\advance\Hilfsskip by \the\hilfscount\edtabcolsep%
         \hilfscount=1\fi%
         \let\NEXT=\EDTABINDENT%
    \fi\NEXT}%
%
```

\edtabindent

(was \tabindent)

```
\newcommand{\edtabindent}{%
    \l@dcolcount=0\relax
    \Hilfsskip=0pt%
    \hilfscount=1\relax
    \EDTABINDENT%
    \hilfsskip=\hsize%
    \advance\hilfsskip -\Hilfsskip%
    \Hilfsskip=0.5\hilfsskip%
    }%

%
```

\EDTAB (was \TAB)

```
\def\EDTAB #1|#2|{%
    \setbox\tabhilfbox=\hbox{$\displaystyle{#1}$}%
    \setbox\tabHilfbox=\hbox{$\displaystyle{#2}$}%
    \advance\tabelskip -\wd\tabhilfbox%
    \advance\tabelskip -\wd\tabHilfbox%
    \unhbox\tabhilfbox\hskip\tabelskip%
    \unhbox\tabHilfbox}%

%
```

\EDTABtext (was \TABtext)

```
\def\EDTABtext #1|#2|{%
    \setbox\tabhilfbox=\hbox{#1}%
    \setbox\tabHilfbox=\hbox{#2}%
    \advance\tabelskip -\wd\tabhilfbox%
    \advance\tabelskip -\wd\tabHilfbox%
    \unhbox\tabhilfbox\hskip\tabelskip%
    \unhbox\tabHilfbox}%
%
```

\tabhilfbox
\tabHilfbox Further helpers.

```
\newbox\tabhilfbox
\newbox\tabHilfbox

%
```

XXX.2.4 Environments

edarrayl edarrayc edarrayr The ‘environment’ forms for \ltab, \ctab and \rtab.

```
\newenvironment{edarrayl}{\l@dcollect@body\ltab}{}
\newenvironment{edarrayc}{\l@dcollect@body\ctab}{}
\newenvironment{edarrayr}{\l@dcollect@body\rtab}{}

%
```

edtabularl edtabularc edtabularr The ‘environment’ forms for \ltabtext, \ctabtext and \rtabtext.

```
\newenvironment{edtabularl}{\l@dcollect@body\ltabtext}{}
\newenvironment{edtabularc}{\l@dcollect@body\ctabtext}{}
\newenvironment{edtabularr}{\l@dcollect@body\rtabtext}{}

%
```

XXXI Quotation’s commands

\initnumbering@quote

This macro, called at the beginning of any numbered section, locally redefines the quotation and quote environments, in order to allow their use inside of numbered sections.

\quotation
\endquotation
\quote
\endquote

\initnumbering@quote defines quotation environment.

```
\newcommand{\initnumbering@quote}{
    \ifnoquotation@\else
    \renewcommand{\quotation}{\par\leavevmode%
                                \parindent=1.5em%
                                \skipnumbering%
                                \ifautopar%
                                    \vskip-\parskip%
                                \else%
                                    \vskip\topsep%
                                \fi%
                                \global\leftskip=\leftmargin%
                                \global\rightskip=\leftmargin%
                            }
    \renewcommand{\endquotation}{\par%
                             \global\leftskip=0pt%
                                \global\rightskip=0pt%
                                \leavevmode%
                                \skipnumbering%
                                \ifautopar%
                                    \vskip-\parskip%
                                \else%
                                    \vskip\topsep%
                                \fi%
                                }
    \renewcommand{\quote}{\par\leavevmode%
                                \parindent=0pt%
                                \skipnumbering%
                                \ifautopar%
                                    \vskip-\parskip%
                                \else%
                                    \vskip\topsep%
                                \fi%
                                \global\leftskip=\leftmargin%
                                \global\rightskip=\leftmargin%
    }
    \renewcommand{\endquote}{\par%
                                \global\leftskip=0pt%
                                \global\rightskip=0pt%
                                \leavevmode%
                                \skipnumbering%
                                \ifautopar%
                                    \vskip-\parskip%
                                \else%
                                    \vskip\topsep%
                                \fi%
                                }
    \fi
}
%
```

XXXII Section’s title commands

XXXII.1 Commands to disable some feature

\ledsectnotoc The \ledsectnotoc only disables the \addcontentsline macro.

```
\newcommand{\ledsectnotoc}{\let\addcontentsline\@gobblethree}
%
```

\ledsectnomark The \ledsectnomark only disables the \chaptermark, \sectionmark and \subsectionmark macros.

```
\newcommand{\ledsectnomark}{%
  \let\chaptermark\@gobble%
  \let\sectionmark\@gobble%
  \let\subsectionmark\@gobble%
}
%
```

XXXII.2 General overview

The system of \eledxxxx commands to section text work like this:

1. When one of these commands is called, reledmac writes to an auxiliary files:
    - The section level.
    - The section title.
    - The side (when eledpar is used).
    - The pstart where the command is called.

  - If we have starred version or not.

2. reledmac adds the title of the section to pstart, as normal content. This is to enable critical notes.
3. When LaTeX is run a other time, this file is read. That:
   - Adds the pstart number to a list of pstarts where a sectioning command is used.
   - Defines a command, the name of which contains the pstart number, and which calls the normal LaTeX sectioning command.
4. This last command is called when the pstart is effectively printed.

XXXII.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).

```
\catcode`\#=12
\notbool{@noeled@sec}{%
%
```

\beforeeledchapter

For technical reasons, not yet solved, page-breaking before chapters can't be made automatically by eledmac. Users have to use \beforeeledchapter.

```
\ifl@dmemoir
  \newcommand\beforeeledchapter{%
  \clearforchapter%
}
\else
  \newcommand\beforeeledchapter{%
  \if@openright%
   \cleardoublepage%
  \else%
   \clearpage%
  \fi%
}
\fi
%
```

XXXII.4 Auxiliary commands

\print@leftmargin@eledsection
\print@rightmargin@eledsection

\print@leftmargin@eledsection and \print@rightmargin@eledsection are added by reledmac inside the code of sectioning command, in order to affix lines numbers. They include tests for RTL languages.

```
\def\print@rightmargin@eledsection{%
  \if@eled@sectioning%
    \begingroup%
    \if@RTL%
        \let\llap\rlap%
        \let\leftlinenum\rightlinenum%
        \let\leftlinenumR\rightlinenumR%
        \let\l@drd@ta\l@dld@ta%
        \let\l@drsn@te\l@dlsn@te%
    \fi%
    \hfill\l@drd@ta \csuse{LR}{\l@drsn@te}%
    \endgroup%
  \fi%
}%

\def\print@leftmargin@eledsection{%
  \if@eled@sectioning%
    \leavevmode%
    \begingroup%
    \if@RTL%
        \let\rlap\llap%
        \let\rightlinenum\leftlinenum%
        \let\rightlinenumR\leftlinenumR%
        \let\l@dld@ta\l@drd@ta%
        \let\l@dlsn@te\l@drsn@te%
    \fi%
    \l@dld@ta\csuse{LR}{\l@dlsn@te}%
    \endgroup%
  \fi%
}%

%
```

XXXII.5 Patching standard commands

\M@sect
\@mem@old@ssect
\@makechapterhead
\@makechapterhead
\@makeschapterhead
\@sect
\@ssect

We have to patch LaTeX, book and memoir sectioning commands in order to:

- 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.

```
\AtBeginDocument{%


\pretocmd{\M@sect}
```

```
  {\let\old@edtext=\edtext%
  \let\edtext=\dummy@edtext@showlemma%
  }
  {}
  {}

\apptocmd{\M@sect}
  {\let\edtext=\old@edtext}
  {}
  {}

\patchcmd{\M@sect}
  { #9}
  { #9%
  \print@rightmargin@eledsection%
  }
  {}
  {}

\patchcmd{\M@sect}
  {\hskip #3\relax}
  {\hskip #3\relax%
  \print@leftmargin@eledsection%
  }
  {}
  {}

\patchcmd{\@mem@old@ssect}
  {#5}
  {#5%
  \print@leftmargin@eledsection%
  }
  {}
  {}

\patchcmd{\@mem@old@ssect}
  {\hskip #1}
  {\hskip #1%
  \print@rightmargin@eledsection%
  }
  {}
  {}



\patchcmd{\scr@startchapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
  \if@eled@sectioning\else%
    \ifl@dprintingpages\else%
      \if@openright\cleardoublepage\else\clearpage\fi%No clearpage inside a
```

```
 \Pages: will keep critical notes from printing on the title page. Here for scrbook.
    \fi%
  \fi%
  }
  {}
  {}

\patchcmd{\@makechapterhead}
  {#1}
  {\print@leftmargin@eledsection%
    #1%
  \print@rightmargin@eledsection%
  }
  {}
  {}

\patchcmd{\@makechapterhead}% For BIDI
  {\if@RTL\raggedleft\else\raggedright\fi}%
  {\if@eled@sectioning\else%
    \if@RTL\raggedleft\else\raggedright\fi%
  \fi%
  }%
  {}%
  {}%

\patchcmd{\@makeschapterhead}
  {#1}
  {\print@leftmargin@eledsection%
    #1%
   \print@rightmargin@eledsection%
  }
  {}
  {}

\pretocmd{\@sect}
  {\let\old@edtext=\edtext
  \let\edtext=\dummy@edtext@showlemma%
  }
  {}
  {}

\apptocmd{\@sect}
  {\let\edtext=\old@edtext}
  {}
  {}

\pretocmd{\@ssect}
  {\let\old@edtext=\edtext%
  \let\edtext=\dummy@edtext@showlemma%
  }
  {}
  {}

\apptocmd{\@ssect}
  {\let\edtext=\old@edtext}
  {}
  {}

%
```

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.

```
\@ifpackageloaded{nameref}{

  \patchcmd{\NR@sect}
    {#8}
    {#8%
    \print@rightmargin@eledsection%
    }
    {}
    {}

  \patchcmd{\NR@sect}
    {\hskip #3\relax}
    {\hskip #3\relax%
    \print@leftmargin@eledsection%
    }
    {}
    {}

  \patchcmd{\NR@ssect}
    {#5}
    {#5%
    \print@rightmargin@eledsection%
    }
    {}
    {}

  \patchcmd{\NR@ssect}
    {\hskip #1}
    {\hskip #1%
    \print@leftmargin@eledsection%
    }
    {}
    {}
  }%
  {
  \patchcmd{\@sect}
    {#8}
    {#8%
    \print@rightmargin@eledsection%
    }
    {}
    {}

  \patchcmd{\@sect}
    {\hskip #3\relax}
    {\hskip #3\relax%
    \print@leftmargin@eledsection%
    }
    {}
    {}

  \patchcmd{\@ssect}
    {#5}
    {#5%
    \print@rightmargin@eledsection%
    }
    {}
    {}

  \patchcmd{\@ssect}
    {\hskip #1}
    {\hskip #1%
    \print@leftmargin@eledsection%
    }
    {}
    {}
  }%
}%
%
```

Close the `\notbool{@noeled@sec}` statement. Also, we have finished patching the commands, using # with a catcode equal to 12, so we are restoring the normal catcode for #.

```
{}}%
\protect\catcode`\#=6 %Space NEEDS by \catcode
%
```

`\chapter` We patch the `\chapter` command even if the `noeledsec` option is called, because we can use `\chapter` in the optional argument of a `\pstart` in parallel typesetting.

```
\AtBeginDocument{%
\patchcmd{\chapter}{\clearforchapter}{%
  \if@eled@sectioning\else%
    \ifl@dprintingpages\else%
      \clearforchapter%
    \fi%
  \fi%
  }%
  {}%
  {}%

\patchcmd{\chapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
  \if@eled@sectioning\else%
    \ifl@dprintingpages%
      \endgraf%
    \else%
      \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
    \fi%
  \fi%
  }%
  {}%
  {}%
}%
%
```

\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.

```
\newif\if@eled@sectioning%
%
```

We reopen a new \notbool{@noeled@sec} statement, as we will define the \elesection commands.

```
\notbool{@noeled@sec}{%
%
```

XXXII.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.

```
\newwrite\eled@sectioning@out
%
```

\eledchapter
\eledsection
\eledsubsection
\eledsubsubsection
\eledchapter*
\eledsection*
\eledsubsection*
\eledsubsubsection*

And now, the user sectioning commands, which write to the file, and also add content as a "normal" line.

```
\newcommand{\eledchapter}[2][]{%
  \disable@familiarnotes%
  #2%
  \restore@familiarnotes%
  \ifledRcol%
```

```
    \immediate\write\eled@sectioningR@out{%
      \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{}{R}
      }%
  \else%
    \immediate\write\eled@sectioning@out{%
      \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{}{}
      }%
  \fi%
}

\newcommand{\eledsection}[2][]{%
  \disable@familiarnotes%
  #2%
  \restore@familiarnotes%
  \ifledRcol%
    \immediate\write\eled@sectioningR@out{%
      \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{}{R}
      }%
  \else%
    \immediate\write\eled@sectioning@out{%
      \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{}{}
      }%
  \fi%
}

\newcommand{\eledsubsection}[2][]{%
  \disable@familiarnotes%
  #2%
  \restore@familiarnotes%
  \ifledRcol%
    \immediate\write\eled@sectioningR@out{%
      \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{}{R
}
      }%
  \else%
    \immediate\write\eled@sectioning@out{%
      \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{}{}
      }%
  \fi%
}
\newcommand{\eledsubsubsection}[2][]{%
  \disable@familiarnotes%
  #2%
  \restore@familiarnotes%
  \ifledRcol%
    \immediate\write\eled@sectioningR@out{%
      \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR
}{}{R}
      }%
  \else%
```

```
    \immediate\write\eled@sectioning@out{%
      \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{}{}
      }%
  \fi%
}


\WithSuffix\newcommand\eledchapter*[2][]{%
  \disable@familiarnotes%
  #2%
  \restore@familiarnotes%
  \ifledRcol%
    \immediate\write\eled@sectioningR@out{%
      \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
      }%
  \else%
    \immediate\write\eled@sectioning@out{%
      \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{}
      }%
  \fi%
}

\WithSuffix\newcommand\eledsection*[2][]{%
  \disable@familiarnotes%
  #2%
  \restore@familiarnotes%
  \ifledRcol%
    \immediate\write\eled@sectioningR@out{%
      \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
      }%
  \else%
    \immediate\write\eled@sectioning@out{%
      \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{}
      }%
  \fi%
}

\WithSuffix\newcommand\eledsubsection*[2][]{%
  \disable@familiarnotes%
  #2%
  \restore@familiarnotes%
  \ifledRcol%
    \immediate\write\eled@sectioningR@out{%
      \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
      }%
  \else%
    \immediate\write\eled@sectioning@out{%
      \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL
}{*}{}
      }%
  \fi%
}

\WithSuffix\newcommand\eledsubsubsection*[2][]{%
  \disable@familiarnotes%
  #2%
  \restore@familiarnotes%
  \ifledRcol%
    \immediate\write\eled@sectioningR@out{%
      \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR
}{*}{R}
      }%
  \else%
    \immediate\write\eled@sectioning@out{%
      \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL
}{*}{}
      }%
  \fi%
}
%
```

XXXII.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.

```
\def\eled@chapter#1#2#3#4#5{%
    \ifstrempty{#4}%
      {%
      \ifstrempty{#1}%
        {%
        \csgdef{eled@sectioning@#3#5}{\let\edtext=\dummy@edtext@showlemma\
chapter{#2}}%
        \csgdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\chaptermark
{#2}}%
        }%Need for \pairs, because of using parbox.
        {%
        \csgdef{eled@sectioning@#3#5}{\let\edtext=\dummy@edtext@showlemma\
chapter[#1]{#2}}%
```

```
    \csgdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\chaptermark{#2}}%Need for \pairs, because of using parbox.
    }%
   }%
   {%
   \ifstrempty{#1}%
    {\csgdef{eled@sectioning@#3#5}{\let\edtext=\dummy@edtext@showlemma\chapter*{#2}}}%
    {\csgdef{eled@sectioning@#3#5}{\let\edtext=\dummy@edtext@showlemma\chapter*[#1]{#2}}}%Bug in LaTeX!
   }%
 \listcsgadd{eled@sections#5@@}{#3}%
  }
\def\eled@section#1#2#3#4#5{%
  \ifstrempty{#4}%
   {\ifstrempty{#1}%
    {%
    \csgdef{eled@sectioning@#3#5}{\section{#2}}%
    \csgdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\sectionmark{#2}}%Need for \pairs, because of using parbox.
    }%
    {%
    \csgdef{eled@sectioning@#3#5}{\section[#1]{#2}}%
    \csgdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\sectionmark{#1}}%Need for \pairs, because of using parbox.
    }%
   }%
   {\ifstrempty{#1}%
    {\csgdef{eled@sectioning@#3#5}{\section*{#2}}}%
    {\csgdef{eled@sectioning@#3#5}{\section*[#1]{#2}}}%Bug in LaTeX!
   }
 \listcsgadd{eled@sections#5@@}{#3}%
  }
\def\eled@subsection#1#2#3#4#5{%
  \ifstrempty{#4}%
   {\ifstrempty{#1}%
    {%
    \csgdef{eled@sectioning@#3#5}{\subsection{#2}}%
    \csgdef{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)
    }%
    {%
    \csgdef{eled@sectioning@#3#5}{\subsection[#1]{#2}}%
    \csgdef{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)
    }%
   }%
   {\ifstrempty{#1}%
```

```
    {\csgdef{eled@sectioning@#3#5}{\subsection*{#2}}}%
    {\csgdef{eled@sectioning@#3#5}{\subsection*[#1]{#2}}}%Bug in LaTeX!
   }
 \listcsgadd{eled@sections#5@@}{#3}%
  }
\def\eled@subsubsection#1#2#3#4#5{%
  \ifstrempty{#4}%
   {\ifstrempty{#1}%
    {\csgdef{eled@sectioning@#3#5}{\subsubsection{#2}}}%
    {\csgdef{eled@sectioning@#3#5}{\subsubsection[#1]{#2}}}%
   }%
   {\ifstrempty{#1}%
    {\csgdef{eled@sectioning@#3#5}{\subsubsection*{#2}}}%
    {\csgdef{eled@sectioning@#3#5}{\subsubsection*[#1]{#2}}}%Bug in LaTeX!
   }
 \listcsgadd{eled@sections#5@@}{#3}%
  }

%
```

End of the conditional test about noeledsec option.

```
}{}
%
```

XXXIII 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.

```
\def\normal@page@break{}
%
```

\prev@pb
\prev@nopb

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.

```
\def\l@prev@pb{}
\def\l@prev@nopb{}
%
```

\ledpb
\ledpbnum
\lednopb
\lednopbnum

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.

```
\newcommand{\ledpb}{\write\linenum@out{\string\led@pb}}
\newcommand{\ledpbnum}[1]{\write\linenum@out{\string\led@pbnum{#1}}}
\newcommand{\lednopb}{\write\linenum@out{\string\led@nopb}}
\newcommand{\lednopbnum}[1]{\write\linenum@out{\string\led@nopbnum{#1}}}
%
```

\led@pb
\led@pbnum
\led@nopb
\led@nopbnum

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.

```
\newcommand{\led@pb}{\listxadd{\l@prev@pb}{\the\absline@num}}
\newcommand{\led@pbnum}[1]{\listxadd{\l@prev@pb}{#1}}
\newcommand{\led@nopb}{\listxadd{\l@prev@nopb}{\the\absline@num}}
\newcommand{\led@nopbnum}[1]{\listxadd{\l@prev@nopb}{#1}}
%
```

\ledpbsetting
\led@pb@setting

The \ledpbsetting macro only changes the value of \led@pb@macro, for which the default value is before.

```
\def\led@pb@setting{before}
\newcommand{\ledpbsetting}[1]{\gdef\led@pb@setting{#1}}
%
```

\led@check@pb
\led@check@nopb

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.

```
\newcommand{\led@check@pb}{\xifinlist{\the\absline@num}{\l@prev@pb}{\pagebreak[4]}{}}
\newcommand{\led@check@nopb}{%
    \IfStrEq{\led@pb@setting}{before}{%
      \xifinlist{\the\absline@num}{\l@prev@nopb}%
         {\numdef{\abs@prevline}{\the\absline@num-1}%
         \xifinlist{\abs@prevline}{\normal@page@break}%
           {\nopagebreak[4]\enlargethispage{\baselineskip}}%
           {}}%
         {}}%
       {}%
     {}%
    \IfStrEq{\led@pb@setting}{after}{%
      \xifinlist{\the\absline@num}{\l@prev@nopb}{%
        \xifinlist{\the\absline@num}{\normal@page@break}%
          {\nopagebreak[4]\enlargethispage{\baselineskip}}%
            {}%
}%
```

```
         {}}%
       {}%
     {}%
}
%
```

XXXIV 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.

```
\newcommand{\check@pb@in@verse}{%
    \ifinstanza\iflednopbinverse\ifinserthangingsymbol% Using stanzas and enabling page breaks in verse control, while on a hanging verse.
        \ifnum\page@num=\last@page@num\else%If we have change page
          \IfStrEq{\led@pb@setting}{before}{%
              \numgdef{\abs@line@verse}{\the\absline@num-1}%
              \ledpbnum{\abs@line@verse}%
            }{}%
          \IfStrEq{\led@pb@setting}{after}{%
              \numgdef{\abs@line@verse}{\the\absline@num-1}%
              \lednopbnum{\abs@line@verse}%
          }{}%
        \fi%
    \fi\fi\fi%
}
%
```

XXXV Tools for hyperref package

\Hy@raisedlink@left

The hyperref package provides a \Hy@raisedlink command, to be used to add an anchor to the top of a line and not to the bottom of it.[35]

[35] http://tex.stackexchange.com/a/17138/7712.

However, this command disrupts the line breaking mechanism when it is called before any word. This is why reledmac defines \Hy@raisedlink@left that is called to the left of words, at the beginning of \edtext or inside the \edlabel commands.[36]

```
\def\Hy@raisedlink@left#1{%
    \ifvmode
        #1%
    \else
        \Hy@SaveSpaceFactor
        \llap{\smash{%
        \begingroup
            \let\HyperRaiseLinkLength\@tempdima
            \setlength\HyperRaiseLinkLength\HyperRaiseLinkDefault
            \HyperRaiseLinkHook
        \expandafter\endgroup
        \expandafter\raise\the\HyperRaiseLinkLength\hbox{%
            \Hy@RestoreSpaceFactor
            #1%
            \Hy@SaveSpaceFactor
        }%
        }}%
        \Hy@RestoreSpaceFactor
        \penalty\@M\hskip\z@\relax
    \fi
}
%
```

XXXVI Compatibility with eledmac

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

```
\ifeledmaccompat@%

  \newcommand{\footnormalX}[1]{\arrangementX[#1]{normal}}%
  \newcommand{\footparagraphX}[1]{\arrangementX[#1]{paragraph}}%
  \newcommand{\foottwocolX}[1]{\arrangementX[#1]{twocol}}%
  \newcommand{\footthreecolX}[1]{\XarrangementX[#1]{threecol}}%

  \unless\ifnocritical@
    \newcommand{\footnormal}[1]{\Xarrangement[#1]{normal}}%
    \newcommand{\footparagraph}[1]{\Xarrangement[#1]{paragraph}}%
    \newcommand{\foottwocol}[1]{\Xarrangement[#1]{twocol}}%
    \newcommand{\footthreecol}[1]{\Xarrangement[#1]{threecol}}%
    \let\hsizetwocol\Xhsizetwocol
    \let\hsizethreecol\Xhsizethreecol
    \let\bhookXnote\Xbhooknote
```

[36] The code is inspired by an answer given by @unbonpetit. Thanks to him. http://texnique.fr:80/osqa/questions/781/hyraisedlink-perturbe-la-maniere-dont-se-fait-la-coupure-de-ligne/801.

```
  \let\boxsymlinenum\Xboxsymlinenum
  \let\symlinenum\Xsymlinenum
  \let\beforenumberinfootnote\Xbeforenumber
  \let\afternumberinfootnote\Xafternumber
  \let\beforeXsymlinenum\Xbeforesymlinenum
  \let\afterXsymlinenum\Xaftersymlinenum
  \let\inplaceofnumber\Xinplaceofnumber
  \let\Xlemmaseparator\lemmaseparator
  \let\afterlemmaseparator\Xafterlemmaseparator
  \let\beforelemmaseparator\Xbeforelemmaseparator
  \let\inplaceoflemmaseparator\Xinplaceoflemmaseparator
  \let\txtbeforeXnotes\Xtxtbeforenotes
  \let\afterXrule\Xafterrule
  \let\numberonlyfirstinline\Xnumberonlyfirstinline
  \let\numberonlyfirstintwolines\Xnumberonlyfirstintwolines
  \let\nonumberinfootnote\Xnonumberinfootnote
  \let\pstartinfootnote\Xpstart
  \let\pstartinfootnoteeverytime\Xpstarteverytime
  \let\onlyXpstart\Xonlypstart
  \let\Xnonumberinfootnote\Xnonumber
  \let\nonbreakableafternumber\Xnonbreakableafternumber
  \let\maxhXnotes\Xmaxhnotes
  \let\beforeXnotes\Xbeforenotes
  \let\boxlinenum\Xboxlinenum
  \let\boxlinenumalign\Xboxlinenumaligm
  \let\boxstartlinenum\Xboxstartlinenum
  \let\boxendlinenum\Xboxendlinenum
  \let\twolines\Xtwolines
  \let\morethantwolines\Xmorethantwolines
  \let\twolinesbutnotmore\Xtwolinesbutnotmore
  \let\twolinesonlyinsamepage\Xtwolinesonlyinsamepage
\fi

\unless\ifnofamiliar@
  \let\notesXwidthliketwocolumns\noteswidthliketwocolumnsX
\fi
\newcommandx{\parafootsep}[2][1,usedefault]{%
    \Xparafootsep[#1]{#2}%
    \parafootsepX[#1]{#2}
}%

\newcommandx{\afternote}[2][1,usedefault]{%
    \Xafternote[#1]{#2}%
    \afternoteX[#1]{#2}%
}%

\unless\ifnoend@
  \let\XendXtwolines\Xendtwolines
  \let\XendXmorethantwolines\Xendmorethantwolines
  \let\bhookXendnote\Xendbhooknote
```

```
    \let\boxXendlinenum\Xendboxlinenum%
    \let\boxXendlinenumalign\Xendboxlinenumalign%
    \let\boxXendstartlinenum\Xendboxstartlinenum%
    \let\boxXendendlinenum\Xendboxendlinenum%
    \let\XendXlemmaseparator\Xendlemmaseparator
    \let\XendXbeforelemmaseparator\Xendbeforelemmaseparator
    \let\XendXafterlemmaseparator\Xendafterlemmaseparator
    \let\XendXinplaceoflemmaseparator\Xendinplaceoflemmaseparator
  \fi

  \AtBeginDocument{%
    \ifdef\lineref{}{\let\lineref\edlineref}%
  }%


\fi%
%
```

</code>

Appendix A Things to do when changing versions

Appendix A.1 Migrating from edmac to ledmac

If you have never used edmac, ignore this section. If you have used edmac and are starting on a completely new document, ignore this section. Only read this section if you are converting an original edmac document to use ledmac.

The package still provides the original \text command, but it is (a) deprecated, and (b) its name has been changed[37] 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 \critext{Smith}
\Afootnote{Jones C, D.}/
on Tuesday.
```

1 I saw my friend
2 Smith on Tuesday.

2 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 \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:

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

1 I saw my friend
2 Smith on Tuesday.

2 Smith] Jones C, D.

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

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

The second argument of the \critext macro, ⟨*commands*⟩, is the same as the second argument to the \edtext macro.

It is possible to define aliases for \critext, which can be easier to type. You can make a single character substitute for \critext by saying this:

```
\catcode`\<=\active
```

[37] A name like \text is likely to be defined by other LaTeX packages (it certainly is by the AMS packages) and it seems sensible to try and avoid clashes with other definitions.

```
\let<=\critext
```

Then you might say <{Smith}\variant{Jones}/. This of course destroys the ability to use < in any new macro definitions, so long as it remains in effect; hence it should be used with care.

Changing the character at the end of the command requires more work:

```
\catcode`\<=\active
\def\xtext#1#2>{\critext{#1}{#2}/}
\let<=\xtext
```

This allows you to say <{Smith}\Afootnote{Jones}>.

Aliases for \critext of the first kind shown here also can't be nested—that is, you can't use the alias in the text that forms the first argument to \critext. (See VI p. 123 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 6.7.1 p. 35), and remove any \Xfootnote command.

If you have customized the \XXXXXXfmt command, please check whether you can achieve the same by the commands documented for display options (7 p. 36) or \Xfootnote options (6.2.2 p. 25). Otherwise please add a new ticket on Github to request a new function for doing this.[38]

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]
```

[38] 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. 9.3 p. 51). 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 an 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 5.2.3 p. 18). 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 16.2 p. 67.

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 19.1.3 p. 70 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 memoirare 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.

Deprecated command	*Replaced with*
\addfootins	\newseries
\addfootinsX	\newseries
\critext	\edtext
\falseverse	\newverse
\interparanoteglue	\Xafternote and \afternoteX
\ledchapter	\eledchapter
\ledsection	\eledsection
\ledsetnormalparstuff	\Xledsetnormalparstuff and \ledsetnormalparstuffX
\ledsubsection	\eledsubsection
\ledsubsubsection	\eledsubsubsection
\noeledsec	Package option noeledsec
\noendnotes	Package option noendnotes
\pageparbreak	\ledpb

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.

Deprecated `\renewcommand`	*Replaced with*
`\@led@extranofeet`	`\newseries`
`\apprefprefixmore`	`\setapprefprefixmore`
`\apprefprefixsingle`	`\setapprefprefixsingle`
`\endstanzaextra`	Optional argument of `\&`
`\hangingsymbol`	`\sethangingsymbol`
`\ledfootinsdim`	`\Xmaxhnotes` and `\maxhnotesX`
`\parafootftmsep`	`\Xparafootsep` and `\parafootsepX`
`\notenumfont`	`\Xnotenumfont`, `\Xendnotenumfont` and `\notenumfontX`
`\notefontsetup`	`\Xnotefontsize`, `\Xendnotefontsize` and `\notefontsizeX`
`\sidenotesep`	`\setsidenotsep`
`\startstanzahook`	Optional argument of `\stanza`
`\symplinenum`	`\Xsymlinenum`

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.

Old command	*New command*
`\footparagraph`	`\Xarrangement`
`\footnormal`	`\Xarrangement`
`\foottwocol`	`\Xarrangement`
`\footthreecol`	`\Xarrangement`
`\footparagraphX`	`\arrangementX`
`\footnormalX`	`\arrangementX`
`\foottwocolX`	`\arrangementX`
`\footthreecolX`	`\arrangementX`
`\afterlemmaseparator`	`\Xafterlemmaseparator`
`\afternote`	`\Xafternote` and `\afternoteX`
`\afternumberinfootnote`	`\Xafternumber`
`\afterXrule`	`\Xafterrule`
`\afterXsymlinenum`	`\Xaftersymlinenum`
`\beforelemmaseparator`	`\Xbeforelemmaseparator`
`\beforenumberinfootnote`	`\Xbeforenumber`
`\beforeXnotes`	`\Xbeforenotes`
`\beforeXsymlinenum`	`\Xbeforesymlinenum`

Old command	*New command*
\bhookXnote	\Xbhookendnote
\bhookXnote	\Xbhooknote
\boxendlinenum	\Xboxendlinenum
\boxlinenum	\Xboxlinenum
\boxlinenumalign	\Xboxlinenumalign
\boxstartlinenum	\Xboxstartlinenum
\boxsymlinenum	\Xboxsymlinenum
\boxXendlinenum	\Xendboxlinenum
\boxXendlinenumalign	\Xendboxlinenumalign
\boxXendstartlinenum	\boxXendstartlinenum
\letboxXendendlinenum	\Xendletboxendlinenum
\hsizetwocol	\Xhsizetwocol
\hsizethreecol	\Xhsizethreecol
\inplaceoflemmaseparator	\Xinplaceoflemmaseparator
\inplaceofnumber	\Xinplaceofnumber
\lemmaseparator	\Xlemmaseparator
\maxhXnotes	\Xmaxhnotes
\morethantwolines	\Xmorethantwolines
\nonumberinfootnote	\Xnonumber
\notesXwidthliketwocolumns	\noteswidthliketwocolumnsX
\noXlemmaseparator	\Xnolemmaseparator
\numberonlyfirstinline	\Xnumberonlyfirstinline
\numberonlyfirstintwolines	\Xnumberonlyfirstintwolines
\nonbreakableafternumber	\Xnonbreakableafternumber
\onlyXpstart	\Xonlypstart
\parafootsep	\Xparafootsep and \parafootsepX
\pstartinfootnote	\Xpstart
\pstartinfootnoteeverytime	\Xpsarteverytime
\symlinenum	\Xsymlinenum
\twolines	\Xtwolines
\twolinesbutnotmore	\Xtwolinesbutnotmore
\twolinesonlyinsamepage	\Xtwolinesonlyinsamepage
\txtbeforeXnotes	\Xtxtbeforenotes
\XendXafterlemmaseparator	\Xendafterlemmaseparator
\XendXbeforelemmaseparator	\Xendbeforelemmaseparator
\XendXinplaceoflemmaseparator	\Xendinplaceoflemmaseparator
\XendXlemmaseparator	\Xendlemmaseparator
\XendXmorethantwolines	\Xendmorethantwolines
\XendXtwolines	\Xendtwolines
\Xnonumberinfootnote	\Xnonumber
\lineref	\edlineref

Appendix A.9.6 Endnotes

With `reledmac`, there is now one auxiliary file for every endnotes set (.Aend, .Bend, .Cend etc.). If you have overriden `\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 overriden internal commands, which is wrong, must adapt according to the following. Or better, they should not override any of such commands and use `reledmac` options instead.

- If you have modified `\Xfootfmt`, note that the fourth argument is now mandatory.
- `\unvxh` has been replaced with `\Xunvxh` and `\unvxhX` with two mandatory arguments.

Appendix A.10 Migration to `reledmac` 2.1.0

`Reledmac` 2.1.0 fix some bugs when using `\Xbhooknote` and `\bhooknoteX` not in order to execute code at the beginning of each notes, but to insert content of at the beginning of each notes.

People who use these commands to do it, which is not the original idea, must change the following:

1. Horizontal space is no longer automatically added after the content of the `\Xbhooknote`/`\bhooknoteX` argument. You must include it manually. So instead of `\Xbhooknote{content}`, use `\Xbhooknote{content }`.
2. Indent is no longer automatically added before the content of the `\Xbhooknote`/`\bhooknoteX` argument. If you want to keep it, add `\indent` in the argument of `\Xbhooknote`/`\bhooknoteX`.

Appendix A.11 Migration to `reledmac` 2.1.3

`Reledmac` 2.1.3 fix an historical bug, (style in `ledmac` 0.7!) which doubled the space before the rules of paragraphed familiar footnotes. Consequently, if you use paragraphed familiar footnotes, you should maybe adapt it, playing with `\beforenotesX`.

Appendix A.12 Migration to `reledmac` 2.3.0

Before `reledmac` 2.3.0, for typesetting verse, any empty line was considered a paragraph inside verses. Counting empty lines this created breaking verse, hanging verses, and also added spurious vertical spaces. Version 2.3.0 disables paragraph in stanza. If you want vertical space, use optional argument of `\stanza` or `\endverse`.

Appendix A.13 Migration to reledmac 2.4.0

It is not mandatory, but strongly recommended, to change any \renewcommand{\endashchar}{⟨...⟩} to the use of \Xlinerangeseparator or / and \Xendlinerangeseparator (7.2.4 p. 39).

Appendix A.14 Migration to reledmac 2.5.0

It is strongly recommended to stop redefining \printnpnum and to use the hooks documented in 7.3 p. 42.

\xlineref does not print anymore the side flag (R for right side), because it is incompatible with numerical test. Use \xflagref to obtain it.

The \printlines and \printendlines commands take now an eighth argument, which is the side flag. It is strongly recommended to NEVER redefine these two commands and to use the setting commands instead (or to ask for new setting commands if the actual does not answer to your needs). However, if you have done it, just change your redefinition to have a new argument.

It is strongly recommended to stop redefining \fullstop and to use \Xsublinesep instead.

Appendix A.15 Migration to reledmac 2.7.0

\SErefonlypage (introduced in reledmac 2.5.0) added an parenthesis after the page number. This was just an error, linked to a bad imitation of \SErefwithpage. That has been deleted. And so, the \XendafterpagenumberSErefonlypage to set it was also deleted.

\rigidbalance is split to two new commands: \Xrigidbalance for critical footnotes and \rigidbalanceX for familiar footnotes. If you have redefined it — but why should you have ?—, you should split your single redefinition in two redefinitions.

Appendix A.16 Migration to reledmac 2.7.2

\Xhsize is already defined in the floatrow package. It becomes \Xwidth, and, consequently, \hsizeX becomes \widthX.

The ancient names are temporarily maintained as aliases.

Appendix A.17 Migration to reledmac 2.8.0

Reledmac 2.8.0 fix spurious indents for paragraphed critical and familiar footnotes in ledgroup and minipage. You can re-establish the indent with \Xparinden and \parindentX.

Appendix A.18 Migration to reledmac 2.13.1

Reledmac 2.5.0 added a bug, which makes the right flag to be printed on the right side of critical footnotes, even if not explicitly requested by using \Xlineflag.

Version 2.13.1 solves this issue. Please use `\Xlineflag` if you want to add the right flag.

Appendix A.19 Migration to `reledmac` 2.18.0

After updating reledmac, and before any new compilation, you need to clean your `.aux` files, if you use `\edlabel` or related.

Appendix A.20 Migration to `reledmac` 2.21.0

Previously, there was a bug, which meant that the description in the handbook was incorrect. If you wrote

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

"elephant" was indexed in the main text and in the critical footnotes. With the new version of `reledmac`, it is indexed only in main text. If you also want to index it in critical footnotes, do

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

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 LaTeX*. February 2003. (Available from CTAN in `macros/latex/contrib/footmisc`)

[LW90] John Lavagnino and Dominik Wujastyk. ‘An overview of `edmac`: a PLAIN TeX format for critical editions’. *TUGboat*, **11**, 4, pp. 623–643, November 1990. (Code available from CTAN in `macros/plain/contrib/edmac`)

[Lüc03] Uwe Lück. ‘ednotes — critical edition typesetting with LaTeX’. *TUGboat*, **24**, 2, pp. 224–236, 2003. (Code available from CTAN in `macros/latex/contrib/ednotes`)

[Sul92] Wayne G. Sullivan. *The file `edstanza.doc`*. June 1992. (Available from CTAN in `macros/plain/contrib/edmac`)

[Wil02] Peter Wilson. *The `memoir` class for configurable typesetting*. November 2002. (Available from CTAN in `macros/latex/contrib/memoir`)

[Wil04] Peter Wilson and Maïeul Rouquette. *Parallel typesetting for critical editions: the `eledpar` package*. December 2004. (Available from CTAN in `macros/latex/contrib/ledmmac`)

Index

Symbols

\& *50*
\@EDROWFILL@ 1
\@adv 1
\@advancestanzanumber 1
\@beforeinsertofthisedtext 1
\@doclearpage 1
\@doreinfeetX 1
\@edindex@fornote@ 1
\@edindex@hyperref 1
\@edrowfill@ 1
\@edtext@level 1

\@emptytoks 1
\@fnpos 1
\@footnotemark 1
\@footnotetext 1
\@getfirstseries 1
\@gobblefour 1
\@gobbleseven 1
\@gobblethree 1
\@h 1
\@hangingsymbol 1
\@iiiminipage 1
\@insertstanzanumber 1
\@k 1
\@l@dtempcnta 1
\@l@dtempcntb 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
\@msd 1
\@msd@c 1
\@msd@options@iffullpage 1
\@msdata@list 1
\@nl 1
\@nl@reg 1
\@opXfeet 1
\@pend 1
\@pendR 1
\@ref 1
\@ref@reg 1
\@ref@reg@parse 1
\@sect 1
\@series 1
\@set 1
\@sidenotesep 1
\@ssect 1
\@startstanza 1
\@stopmsd 1
\@stopstanza 1
\@sw 1
\@tag 1
\@wredindex 1

\@xloop 1

\@xympar 1

Classarticle 70

Classbook 70, 343

Classmemoir 199, 259–261, 294, 343, 363, 425, 429

Classscrbook 429

Command*footnote 71

Command\...@footnotemark... 202

Command\...d@ta 149

Command\<hook

    @<series 247

Command\<hookname

    <pseudoseries 249, 250

Command\<type

    footfmt 189

Command\@@line 180

Command\@MM 166, 426

Command\@Rlineflag 296, 426

Command\@SErefprefix 273

Command\@SErefprefixmore 273

Command\@add@ 333

Command\@adv 108

Command\@apprefprefixmore 273

Command\@apprefprefixsingle 273

Command\@beforeinsertofthisedtext 130

Command\@bsphack 262

Command\@doclearpage 260, 261, 419, 429

Command\@doreinfeetX 429

Command\@dprintingcolumns 426

Command\@edindex@hyperref 296, 297

Command\@edtext@level 127

Command\@esphack 262

Command\@fnpos 220, 256

Command\@footnotemark 199, 200, 419, 429

Command\@footnotetext 199, 200, 419

Command\@gobble 125, 126, 242

Command\@gobblefive 428

Command\@gobblefour 425

Command\@gobbleseven 244

Command\@gobblethree 418

Command\@h 183

Command\@hangingsymbol 301

Command\@iiiminipage 285, 287, 418, 429

Command\@iiiminpage 285

Command\@l 424

Command\@l@dtempcnta 151, 153, 161

Command\@l@dtempcntb 153

Command\@l@reg 424

Command\@lab 104, 262, 265, 269, 418

Command\@ldunboxmpfoot 288

COMMAND\@led@extranofeet 364
COMMAND\@ledinnote@command 292, 293
COMMAND\@lemma 130, 132
COMMAND\@lock 98, 301
COMMAND\@lopL 419
COMMAND\@lopR 419
COMMAND\@makecol 256, 257, 259, 429
COMMAND\@mpfnpos 220
COMMAND\@msd 311
COMMAND\@msd@c 311
COMMAND\@msd@options@iffullpage 317
COMMAND\@msdata@list 311, 312
COMMAND\@nl 104–107, 109, 118, 265, 418, 419
COMMAND\@nl@reg 105, 361, 419, 424
COMMAND\@opXfeet 419
COMMAND\@opfeetX 429
COMMAND\@opxtrafeeti 429
COMMAND\@page 106, 265
COMMAND\@pend 419
COMMAND\@pendR 419
COMMAND\@ref 104, 114–116, 119, 120, 125
COMMAND\@ref@later 115, 121
COMMAND\@ref@reg 114, 419
COMMAND\@ref@reg@parsearg 115
COMMAND\@reinserts 256–259, 429
COMMAND\@secondoftwo 72
COMMAND\@sect 346
COMMAND\@series 246
COMMAND\@set 108
COMMAND\@sidenotesep 284
COMMAND\@stopmsd 311
COMMAND\@sw 116, 133, 136, 137
COMMAND\@tag 126, 128, 131
COMMAND\@tempcnta 84
COMMAND\@tempcntb 84
COMMAND\@toksa 90
COMMAND\@toksb 90
COMMAND\@xloop 162
COMMAND\@xympar 277, 429
COMMAND\Aendnote 16, 25
COMMAND\Afootfmt 166
COMMAND\Afootgroup 166
COMMAND\Afootnote 8, 16, 24, 25, 28, 128, 175, 199, 221, 237, 428
COMMAND\Afootstart 166
COMMAND\AtBeginDocument 259
COMMAND\AtEndEveryPend 19, 435
COMMAND\AtEveryPend 18, 19, 53, 143, 426, 427, 429, 435
COMMAND\AtEveryPend* 19
COMMAND\AtEveryPstart 18, 19, 53, 426, 427, 429, 432, 435
COMMAND\AtEveryPstart* 19

COMMAND\AtEveryStanza . . . 53, 433, 435
COMMAND\AtEveryStopStanza . . . 53, 433, 435
COMMAND\AtStartEveryPstart . . . 19, 435
COMMAND\AtStartEveryStanza . . . 53, 435
COMMAND\BeforeEveryStopStanza . . . 435
COMMAND\Bendnote . . . 16, 24
COMMAND\Bfootnote . . . 8, 16, 199, 221, 237
COMMAND\Centering . . . 46
COMMAND\Cfootnote . . . 199
COMMAND\Columns . . . 85, 171
COMMAND\Dfootnote . . . 199
COMMAND\Efootnote . . . 199
COMMAND\Gls . . . 63
COMMAND\Hy@raisedlink . . . 355
COMMAND\Hy@raisedlink@left . . . 356
COMMAND\LTR . . . 46
COMMAND\NR . . . 346
COMMAND\Pages . . . 85, 257, 258
COMMAND\ProcessOptionsX . . . 75
COMMAND\RL . . . 45
COMMAND\RaggedLeft . . . 46
COMMAND\RaggedRight . . . 46
COMMAND\SEonlypage . . . 271, 431
COMMAND\SEref . . . 57–59, 271, 274, 432, 434
COMMAND\SErefonlypage . . . 57–59, 367, 431
COMMAND\SErefwithpage . . . 57, 59, 271, 274, 367, 431, 433
COMMAND\Stanza . . . 424
COMMAND\Waklam . . . 334
COMMAND\X@doreinfeet . . . 258, 429
COMMAND\XR@prefix . . . 277
COMMAND\XR@test . . . 277
COMMAND\XR@test@mac . . . 277
COMMAND\XR@test@mac@test . . . 277
COMMAND\XXXXXXfmt . . . 360
COMMAND\XXXXXfmt . . . 360
COMMAND\Xafterlemmaseparator . . . 43, 364
COMMAND\Xafternote . . . 46, 47, 363, 364
COMMAND\Xafternumber . . . 41, 364
COMMAND\Xafterrule . . . 48, 222, 364, 424, 427
COMMAND\Xaftersymlinenum . . . 41, 364
COMMAND\Xarrangement . . . 37, 47, 71, 167, 168, 248, 364
COMMAND\Xarrangement@footparagraph . . . 173
COMMAND\Xarrangement@normal . . . 168
COMMAND\Xarrangement@paragraph . . . 173
COMMAND\Xbeforeinserting . . . 45, 46
COMMAND\Xbeforelemmaseparator . . . 43, 364
COMMAND\Xbeforenotes . . . 48, 221, 364, 424, 427
COMMAND\Xbeforenumber . . . 39, 41, 364
COMMAND\Xbeforesymlinenum . . . 41, 364
COMMAND\Xbhookendnote . . . 365

Command\Xbhookgroup 47, 431, 432
Command\Xbhooknote 45, 365, 366, 429, 430
Command\Xboxendlinenum 42, 365, 428
Command\Xboxlinenum 41, 42, 365
Command\Xboxlinenumalign 42, 365, 428
Command\Xboxstartlinenum 42, 365, 428
Command\Xboxsymlinenum 41, 42, 365
Command\Xcolalign 46, 427
Command\Xdo@feet 429, 434
Command\Xend 244
Command\XendXafterlemmaseparator 365
Command\XendXbeforelemmaseparator 365
Command\XendXinplaceoflemmaseparator 365
Command\XendXlemmaseparator 365
Command\XendXmorethantwolines 365
Command\XendXtwolines 365
Command\Xendafterenumber 41, 430
Command\Xendafterlemmaseparator 43, 365
Command\Xendafternote 49, 432
Command\Xendafternumber 43
Command\Xendafterpagenumbe 433
Command\Xendafterpagenumber 42, 59
Command\XendafterpagenumberSErefonlypage 367
Command\Xendaftersymlinenum 41, 43, 430
Command\Xendahookinplaceofnumber 43, 430
Command\Xendahooklinenumber 43, 430
Command\Xendbeforelemmaseparator 43, 365
Command\Xendbeforelinenumber 42
Command\Xendbeforenumber 41, 430
Command\Xendbeforepagenumber 42, 58, 59
Command\XendbeforepagenumberSErefonlypage 58
Command\Xendbeforesymlinenum 41, 43, 430
Command\Xendbhookinplaceofnumber 43, 430
Command\Xendbhooklinenumber 42, 430
Command\Xendbhooknote 45
Command\Xendboxendlinenum 42, 428
Command\Xendboxlinenum 42, 365, 426
Command\Xendboxlinenumalign 42, 365, 428
Command\Xendboxstartlinenum 42, 428
Command\Xendboxsymlinenum 42, 430
Command\Xendhangindent 45, 430, 432
Command\Xendinplaceoflemmaseparator 26, 43, 365
Command\Xendinplaceofnumber 41, 429
Command\Xendinplaceofpagenumber 39, 434
Command\Xendinsertsep@ 228
Command\Xendlemmadisablefontselection 44
Command\Xendlemmafont 44, 431
Command\Xendlemmaseparator 26, 43, 365
Command\Xendletboxendlinenum 365
Command\Xendlineflag 59

Command\Xendlineprefixmore 42, 59
Command\Xendlineprefixsingle 42, 59
Command\Xendlinerangeseparator 39, 59, 187, 367, 430
Command\Xendmorethantwolines 26, 40, 59, 365, 427, 428
Command\Xendnonumber 40, 429
Command\Xendnote 224, 243, 244, 427
Command\Xendnotefontsize 44, 364
Command\Xendnotenumfont 42–44, 364
Command\Xendnotes 227
Command\Xendnumberonlyfirstinline 38, 430
Command\Xendnumberonlyfirstintwolines 38, 430
Command\Xendpagenumberonlyfirst 38, 434
Command\Xendpagenumberonlyfirstifsingle 38, 434
Command\Xendpagenumberonlyfirstintwo 38, 434
Command\Xendparagraph 49, 424
Command\Xendsep 49
Command\Xendsublinesep 40, 59, 187
Command\Xendsymlinenum 38, 430
Command\Xendsympagenum 39, 434
Command\Xendtwolines 26, 39, 40, 59, 365, 427, 428
Command\Xendtwolinesbutnotmore 40, 59, 427, 428
Command\Xendtwolinesonlyinsamepage 40, 59, 427, 428
Command\Xendwrapcontent 45, 433
Command\Xendwraplemma 45, 433
Command\Xfootfmt 366
Command\Xfootgroup 172
Command\Xfootins 171
Command\Xfootnote 56, 62, 126, 360, 421, 425, 427, 431, 433
Command\Xfootstarts 172
Command\Xgroupbyline 47, 159, 197
Command\Xgroupbylines 434
Command\Xgroupbylineseparetwolines 47
Command\Xhangindent 45, 430
Command\Xhsize 367, 431, 432
Command\Xhsizethreecol 46, 49, 365
Command\Xhsizetwocol 46, 49, 249, 365
Command\Xinplaceoflemmaseparator 25, 43, 365
Command\Xinplaceofnumber 41, 365, 427, 428
Command\Xinsertparafootsep 177, 179
Command\Xledsetnormalparstuff 362, 363, 427
Command\Xlemmadisablefontselection 44
Command\Xlemmafont 44, 431
Command\Xlemmaseparator 43, 187, 251, 253, 255, 365
Command\Xlineflag 58, 367, 368, 433
Command\Xlinerangeseparator 39, 58, 187, 367, 430
Command\Xmaxhnotes 48, 70, 71, 364, 365, 424, 426
Command\Xmorethantwolines 25, 39, 40, 58, 365, 426
Command\Xnoindent 430
Command\Xnolemmaseparator 43, 255, 365
Command\Xnonbreakableafternumber 41, 365, 422

COMMAND\Xnonumber . . . 40, 365
COMMAND\Xnonumberinfootnote . . . 365
COMMAND\Xnotefontsize . . . 44, 364
COMMAND\Xnotefontsize@⟨*s*⟩ . . . 178, 182, 183
COMMAND\Xnotenumfont . . . 44, 364
COMMAND\Xnoteswidthliketwocolumns . . . 49, 425
COMMAND\Xnumberonlyfirstinline . . . 38, 39, 47, 101, 189, 250, 251, 253, 365, 421, 426, 435
COMMAND\Xnumberonlyfirstintwolines . . . 38, 47, 365, 421, 435
COMMAND\Xonlypstart . . . 40, 365, 421, 426
COMMAND\Xpagelinesep . . . 41, 434
COMMAND\Xparafootsep . . . 47, 101, 364, 365, 433, 435
COMMAND\Xparafootsep@series . . . 177
COMMAND\Xparinden . . . 367
COMMAND\Xparindent . . . 45, 427, 430, 432
COMMAND\Xprenotes . . . 48, 222, 433
COMMAND\Xprenotes@ . . . 171, 222, 421
COMMAND\Xpstart . . . 40, 362, 365, 421, 426
COMMAND\Xpstarteverytime . . . 40, 362, 365, 426
COMMAND\Xragged . . . 47
COMMAND\Xrigidbalance . . . 180, 367, 431
COMMAND\Xstanza . . . 40, 53
COMMAND\Xstanzaseparator . . . 40
COMMAND\Xstorelineinfo . . . 189
COMMAND\Xsublinesep . . . 22, 40, 41, 59, 187, 367
COMMAND\Xsublinesepside . . . 22, 40
COMMAND\Xsymlinenum . . . 38, 47, 364, 365, 428, 435
COMMAND\Xtextbeforenotes . . . 162
COMMAND\Xtoendnotes . . . 27, 244
COMMAND\Xtwolines . . . 25, 39, 40, 59, 194, 195, 249, 365, 426
COMMAND\Xtwolinesappref . . . 249
COMMAND\Xtwolinesbutnotmore . . . 39, 40, 59, 365, 427
COMMAND\Xtwolinesbutnotmoreappref . . . 250
COMMAND\Xtwolinesonlyinsamepage . . . 39, 40, 59, 365, 427
COMMAND\Xtxtbeforenotes . . . 47, 365, 433, 434, 436
COMMAND\Xtxtbeforenotesonlyonce . . . 47, 436
COMMAND\Xunvxh . . . 175, 366
COMMAND\Xwidth . . . 49, 367, 432
COMMAND\Xwrapcontent . . . 45, 433
COMMAND\Xwraplemma . . . 44–46, 433
COMMAND\& . . . 364
COMMAND\⟨*XXX*⟩vfootnote . . . 197
COMMAND\absline@num . . . 98, 150
COMMAND\accent . . . 125
COMMAND\actionlines@list . . . 99, 152
COMMAND\actions@list . . . 99
COMMAND\add@Xgroupbyline . . . 159
COMMAND\add@inserts . . . 99, 158, 159
COMMAND\add@inserts@next . . . 158, 159
COMMAND\add@msd@ . . . 311
COMMAND\add@msdata . . . 311, 312

Command\add@msdata@firstlineofpage 314

Command\add@msddata 311

Command\add@penalties 150, 160

Command\addcontentsline 341

Command\addfootins 360, 363

Command\addfootinsX 360, 363

Command\addtoendnotes 244

Command\advancelabel@refs 264

Command\advanceline 23, 100, 108, 122, 429

Command\affixlin@num 284

Command\affixline@num 153, 156, 157, 419

Command\affixpstart@num 157

Command\afterXrule 364

Command\afterXsymlinenum 364

Command\afterenumber 41

Command\aftergroup 125, 129

Command\afterlemmaseparator 364

Command\afternote 364

Command\afternoteX 47, 363, 364

Command\afternumberinfootnote 364

Command\afterruleX 48, 424, 427

Command\applabel 58, 266, 267, 427, 433

Command\appref 56, 58, 59, 271, 274, 431, 432

Command\apprefprefixmore 58, 364

Command\apprefprefixsingle 58, 364

Command\apprefwithpage 58, 59, 271, 274, 428, 431

Command\arrangementX 37, 71, 203, 248, 364

Command\arrangementX@normal 208

Command\article 15

Command\at@every@pend 143

Command\autopar 18, 140, 144, 145, 218, 420, 422, 423, 427

Command\ballast 71

Command\ballast@count 150, 161

Command\baselineskip 37, 174, 178

Command\beforeXnotes 364

Command\beforeXsymlinenum 364

Command\beforeeledchapter 10, 68, 342

Command\beforeinsertingX 45

Command\beforelemmaseparator 364

Command\beforenotesX 48, 366, 423, 424, 427

Command\beforenumberinfootnote 364

Command\begin 319

Command\beginnumbering 16, 17, 19, 20, 85, 86, 88, 97, 102, 117, 144, 224, 310, 421, 424, 428, 429, 434, 435

Command\bf 421

Command\bfseries 44, 421

Command\bhookXnote 365

Command\bhookgroupX 48, 431

Command\bhooknoteX 45, 366, 429, 430

Command\body 302

Command\bodyfootmarkA 34
Command\book 15
Command\boxXendlinenum 365
Command\boxXendlinenumalign 365
Command\boxXendstartlinenum 365
Command\boxendlinenum 365
Command\boxlinefootnote 191
Command\boxlinenum 365
Command\boxlinenumalign 365
Command\boxstartlinenum 365
Command\boxsymlinenum 365
Command\break 37, 175
Command\brokenpenalty 160
Command\centering 46
Command\ch@ck@l@ck 419
Command\ch@cksub@l@ck 156, 419
Command\chapter 67, 343, 347, 424, 427, 429, 433
Command\chaptermark 341
Command\check@pb@in@verse 355
Command\colalignX 46, 427
Command\collect@body 319
Command\color 433
Command\colorbox 72
Command\columns 49
Command\columnwidth 174, 425
Command\command names 249, 250
Command\copyright 126
Command\correct@Xfootins@box 426
Command\correct@footinsX@box 426
Command\count 181
Command\critex 420
Command\critext 132, 359, 360, 363
Command\csname 76, 135
Command\csquotes 241
Command\ctab 335, 339
Command\ctabtext 340
Command\dcol 328
Command\def 73
Command\detokenize 135
Command\dimen 181
Command\dimexpr 49
Command\discretionary 175
Command\displaywidowpenalty 160
Command\do@Xfeet 257, 419, 429, 434
Command\do@actions 150–152, 419
Command\do@actions@fixedcode 419
Command\do@actions@next 151, 152
Command\do@ballast 150, 161
Command\do@feet@custom@order 256
Command\do@insidelinehook 422

COMMAND\do@line 99, 124, 142, 145, 149, 158, 160, 301, 419, 420, 422, 424
COMMAND\do@linehook 419
COMMAND\do@lockoff 100
COMMAND\do@lockon 100
COMMAND\dodoreinxtrafeet 418
COMMAND\doendnotes 26, 228, 366, 428, 435
COMMAND\doendnotesbysection 26, 228, 244, 428, 435
COMMAND\doennotes 435
COMMAND\doinsidelinehook 24, 425
COMMAND\dolinehook 24, 425
COMMAND\doreinxtrafeeti 429
COMMAND\doreinxtrafeetii 429
COMMAND\doxtrafeet 256, 418
COMMAND\doxtrafeeti 429
COMMAND\doxtrafeetii 429
COMMAND\dummy@ref 125
COMMAND\edaftertab 66, 334, 335
COMMAND\edatleft 66, 332
COMMAND\edatright 66, 333
COMMAND\edbeforetab 66, 334, 335
COMMAND\edfilldimen 333
COMMAND\edfont@info 131
COMMAND\edgls 63, 291
COMMAND\edglsadd 435
COMMAND\edgls... 432
COMMAND\edindex 61–63, 291, 292, 295, 297, 323, 422, 425, 426, 429, 430, 434–436
COMMAND\edindexlab 63
COMMAND\edlabel 55–59, 126, 262, 264, 265, 268, 269, 276, 291, 323, 356, 368, 418, 421–423, 426, 431
COMMAND\edlabelE 57, 267
COMMAND\edlabelS 57, 267
COMMAND\edlabelSE 57
COMMAND\edlineref 55, 262, 365, 426, 428, 431, 435
COMMAND\edmakelabel 57, 276
COMMAND\edpageref 55, 262, 268, 276
COMMAND\edrowfill 334
COMMAND\edsublineref 55
COMMAND\edtabcolsep 327
COMMAND\edtext 6, 24, 25, 27–32, 50, 55–58, 61, 64, 71, 99, 114, 116, 119, 120, 123–132, 134–136, 138, 266, 267, 270, 323, 324, 343, 356, 359, 360, 363, 418, 420, 422, 424–428, 433–436
COMMAND\edtext@level 428
COMMAND\edtextlater 116
COMMAND\edvertdots 67, 333
COMMAND\edvertline 66, 67, 333
COMMAND\elechapter 68
COMMAND\eled@sectioning@out 348
COMMAND\eledchapter 67, 363, 425, 429
COMMAND\eledchapter* 67
COMMAND\eledmac@error 418
COMMAND\eledsection 7, 16, 67, 125, 148, 342, 363, 427, 435
COMMAND\eledsection* 67

Command\eledsubsection 67, 363

Command\eledsubsection* 67

Command\eledsubsubsection 67, 363

Command\eledsubsubsection* 67

Command\eledxxx 10, 68, 348, 424

Command\eledxxxx 341

Command\elesection 348

Command\else 290, 342

Command\empty 84, 154, 262

Command\end 318, 319

Command\end@lemmas 125

Command\endashchar 187

Command\endgraf 142, 177, 218

Command\endlock 22, 100, 123, 306

Command\endminipage 285, 287, 418, 429

Command\endmsdata 33

Command\endnotes 427, 431

Command\endnumbering 16, 17, 20, 85, 86, 88, 419, 428, 434, 435

Command\endprint 224, 227, 244, 362

Command\endstanzaextra 364

Command\endsub 22, 100, 121

Command\endverse 366

Command\everypar 144

Command\extensionchars 69, 85

Command\externaldocument 59, 60, 276

Command\f@x@l@cks 419

Command\falseverse 363, 422, 424

Command\fi 342

Command\firstlinenum 21, 153, 420

Command\firstsublinenum 21, 420

Command\fix@page 105, 106, 419

Command\flag@end 119, 120, 131, 424

Command\flag@end@RTL 120

Command\flag@end@later 121

Command\flag@start 119, 120, 131, 424, 425

Command\flag@start@RTL 120

Command\flag@start@later 121

Command\flagstanza 54

Command\floatingpenalty 166, 426

Command\flush@notes 161, 162

Command\fnpos 36, 220, 423, 434

Command\footfmt 166, 168

Command\footfmt... 203

Command\footfootmarkA 34

Command\footfudgefactor 175

Command\footfudgefiddle 71, 173, 174, 418

Command\footgroup 166

Command\footins 171

Command\footnormal 249, 364, 418

Command\footnormalX 364

Command\footnote 34, 70, 199, 200, 361, 419
Command\footnote@lang 187
Command\footnoteA 16, 34
Command\footnoteB 16
Command\footnoteC 24
Command\footnoteE 34
Command\footnoteX 8, 35, 241, 242, 433
Command\footnoteX@reading 242
Command\footnoteXmark 242, 435
Command\footnoteXmk 255
Command\footnoteXtext 242, 435
Command\footnote⟨X⟩ 126
Command\footnote⟨X⟩mark 35
Command\footnote⟨X⟩mk 35
Command\footnote⟨X⟩nomk 35
Command\footnote⟨X⟩text 35
Command\footnotelang@lua 165
Command\footnotelang@poly 165
Command\footnotemark 35, 242
Command\footnoteoption@ 164, 430
Command\footnoterule 181
Command\footnotesize 44
Command\footnotetext 35, 242
Command\footparagraph 174, 249, 364, 424
Command\footparagraphX 213, 364, 424
Command\footsplitskips 419, 426
Command\footstart 166, 171, 181
Command\footstrut 177
Command\footthreecol 364
Command\footthreecolX 364, 428
Command\foottwocol 364
Command\foottwocolX 364, 428
Command\foreignlanguage 45
Command\fullstop 367
Command\get@edindex@hyperref 296
Command\get@edindex@ledinnote@command 292
Command\get@fnmark 200
Command\get@index@command 423
Command\get@linelistfile 419
Command\get@thisfootnote 207
Command\getline@num 150, 151
Command\gl@p 90
Command\global 104
Command\globaldefs 104
Command\gls 63, 299
Command\hangindentX 45, 427, 430
Command\hangingsymbol 364, 420
Command\hbox 175
Command\hfill 423
Command\hidenumbering 23, 113, 427

COMMAND\hidenumberingonleftpage . . . 23, 113, 433
COMMAND\hidenumberingonrightpage . . . 23, 113, 433
COMMAND\hline . . . 64
COMMAND\hrulefill . . . 334
COMMAND\hsize . . . 38, 171, 174, 176, 182, 185, 219, 419, 425
COMMAND\hsizeX . . . 367, 431, 432
COMMAND\hsizethreecol . . . 365
COMMAND\hsizethreecolX . . . 46, 49
COMMAND\hsizetwocol . . . 365
COMMAND\hsizetwocolX . . . 46, 49
COMMAND\hyperlinkR . . . 296
COMMAND\hyperlinkformat . . . 296
COMMAND\hyperlinkformatR . . . 296
COMMAND\if@RTL . . . 77, 129
COMMAND\if@edtext@ . . . 425, 428
COMMAND\if@eled@sectioning . . . 348
COMMAND\if@firstlineofpage . . . 77
COMMAND\if@firstlineofpageR . . . 77
COMMAND\if@msd@options@fullpage . . . 317
COMMAND\if@msdata@insertedfrompreviouspage . . . 314
COMMAND\if@nobreak . . . 143
COMMAND\if@noneed@Footnote . . . 119
COMMAND\ifXnote@ . . . 85
COMMAND\ifbypage@ . . . 91
COMMAND\ifbypage@R . . . 91
COMMAND\ifbypstart@ . . . 91
COMMAND\ifbypstart@R . . . 91
COMMAND\iffirst@linenum@out@ . . . 117, 118
COMMAND\ifindtl@innote . . . 85
COMMAND\ifindtl@notenumber . . . 85
COMMAND\ifinserthangingsymbol . . . 301
COMMAND\ifinstanza . . . 301
COMMAND\ifistwofollowinglines . . . 195
COMMAND\ifl@d@Xmorethantwolines . . . 192, 426
COMMAND\ifl@d@Xtwolines . . . 192
COMMAND\ifl@d@dash . . . 192
COMMAND\ifl@d@elin . . . 192
COMMAND\ifl@d@esl . . . 192
COMMAND\ifl@d@pnum . . . 192
COMMAND\ifl@d@ssub . . . 192
COMMAND\ifl@dend@X . . . 243
COMMAND\ifl@dmemoir . . . 418
COMMAND\ifl@dpaging . . . 425
COMMAND\ifl@dpairing . . . 84, 420
COMMAND\ifl@dprintingpages . . . 426
COMMAND\ifl@dskipnumber . . . 153
COMMAND\ifl@dstartendok . . . 333
COMMAND\ifl@imakeidx . . . 77
COMMAND\ifledRcol . . . 85, 420
COMMAND\ifledRcol@ . . . 85, 424

Command\iflemmacommand@ 425

Command\ifnoend@ 229

Command\ifnoledgroup@ 290

Command\ifnoteschanged@ 101

Command\ifnumberedpar@ 139

Command\ifnumbering 86, 88

Command\ifnumberingR 85, 420

Command\ifnumberline 130, 153

Command\ifpst@rted 420

Command\ifpst@rtedL 86

Command\ifseriesbefore 247

Command\ifstopmsdata@inserted@ 310

Command\ifsublines@ 98, 110

Command\iftrue 428

Command\ifvmode 264

Command\ifxxx 342

Command\ignorespaces 128

Command\imki@wrindexentry 77

Command\immediate 117, 118, 223

Command\indent 18, 144, 366

Command\index 298, 299, 434

Command\indtl@wrindexentry 77

Command\initnumbering@quote 340, 429

Command\initnumbering@reg 419

Command\initnumbering@sectcmd 429

Command\inplaceoflemmaseparator 365

Command\inplaceofnumber 365

Command\insert 130, 158, 166, 168, 169, 183, 197, 203

Command\insert@Xtxtbeforenotes 163, 183

Command\insert@count 114, 119, 128

Command\insert@countR 128

Command\insert@msdata 311, 317

Command\insert@txtbeforenotesX 163

Command\inserthangingsymbol 423

Command\insertlines@list 99, 114

Command\insertparafootsepX 217

Command\inserts@list 124, 140, 158, 159, 175

Command\interAfootnotelinepenalty 420

Command\interfootnotelinepenalty 420

Command\interlinepenalty 166

Command\interparanoteglue 363

Command\justifying 46

Command\l@advance@parledegroup@beforenormalnotes 429

Command\l@d@@wrindexhyp 425

Command\l@d@add 133

Command\l@d@end 224, 243

Command\l@d@nums 128, 131–133, 192

Command\l@d@section 224

Command\l@d@set 109, 122

Command\l@dampcount 325

Command\l@dbfnote 200, 419
Command\l@dcheckstartend 333
Command\l@dchset@num 109
Command\l@dcolcount 325, 326
Command\l@dcollect@@body 319
Command\l@dcollect@body 318
Command\l@dcsnote 424
Command\l@dcsnotetext 149, 281
Command\l@dcsnotetext@l 149, 281
Command\l@dcsnotetext@r 149, 281
Command\l@ddodoreinxtrafeet 257, 418
Command\l@ddoxtrafeet 257, 418
Command\l@demptyd@ta 420
Command\l@dend@close 223
Command\l@dend@open 223
Command\l@dend@stuff 224
Command\l@denvbody 319
Command\l@dfeetbeginmini 418
Command\l@dfeetendmini 418
Command\l@dgetline@margin 420
Command\l@dgetlock@disp 420
Command\l@dgetref@num 269, 270
Command\l@dgetsidenote@margin 278, 420
Command\l@dgobbeloptarg 425
Command\l@dgobblearg 425
Command\l@dgobbleoptarg 323
Command\l@dlabel@parse 269, 270
Command\l@dld@ta 153, 155
Command\l@dlp@rbox 283
Command\l@dlsn@te 420
Command\l@dlsnote 424
Command\l@dmake@labels 264, 265, 277
Command\l@dmake@labelsR 277
Command\l@dnumpstartsL 86, 420
Command\l@dp@rsefootspec 192
Command\l@dparsefootspec 192
Command\l@dpush@begins 319
Command\l@drd@ta 153, 155
Command\l@dref@undefined 269
Command\l@drsn@te 420
Command\l@drsnote 424
Command\l@dtabaddcols 333
Command\l@dtabnoexpands 418
Command\l@dumboxmpfoot 429
Command\l@dunboxmpfoot 420
Command\l@dzeropenalties 420, 425
Command\l@pb 354
Command\l@prev@nopb 353
Command\l@prev@pb 353
Command\l@reg 361

COMMAND\label 19, 57, 59, 63, 262, 263, 270
COMMAND\label@refs 262
COMMAND\labelpstarttrue 19, 421
COMMAND\labelref@list 262, 265
COMMAND\language 175
COMMAND\last@page@num 419
COMMAND\lastbox 144
COMMAND\lastskip 121
COMMAND\latex@makecol 259
COMMAND\leavevmode 18, 144
COMMAND\led@check@nopb 354
COMMAND\led@check@pb 354
COMMAND\led@nopb 354, 355
COMMAND\led@nopbnum 354
COMMAND\led@pb 354, 355
COMMAND\led@pb@macro 354
COMMAND\led@pbnum 354
COMMAND\led@reinit@index@fornote 299
COMMAND\led@set@index@fornote 298
COMMAND\ledRflag 296
COMMAND\ledchapter 363, 422
COMMAND\ledfootinsdim 364
COMMAND\ledinnernote 60, 279, 424, 433
COMMAND\ledinnote 293, 362, 428
COMMAND\ledinnotemark 62, 362, 427
COMMAND\ledleftnote 60, 279
COMMAND\ledlinenum 96, 420
COMMAND\ledllfill 149
COMMAND\ledlsnotefontsetup 433
COMMAND\ledlsnotesep 60
COMMAND\ledlsnotewidth 60
COMMAND\lednopb 69, 354
COMMAND\lednopbinverse 355
COMMAND\lednopbinversetrue 52, 69
COMMAND\lednopbnum 354
COMMAND\ledouternote 60, 279, 424, 433
COMMAND\ledpb 69, 354, 363
COMMAND\ledpbnum 354
COMMAND\ledpbsetting 69, 354, 430
COMMAND\ledrightnote 60, 279
COMMAND\ledrsnotefontsetup 433
COMMAND\ledrsnotesep 60
COMMAND\ledrsnotewidth 60
COMMAND\ledsection 363
COMMAND\ledsectnomark 341
COMMAND\ledsectnotoc 341
COMMAND\ledsetnormalparstuff 362, 363, 427
COMMAND\ledsetnormalparstuff@common 218
COMMAND\ledsetnormalparstuffX 362, 363, 427
COMMAND\ledsidenote 60, 279, 281

Command\ledsubsection 363
Command\ledsubsubsection 363
Command\ledxxx 424
Command\left 66
Command\leftctab 335
Command\leftheadline 96
Command\leftlinenum 22, 96, 418, 420
Command\leftltab 334
Command\leftnoteupfalse 60
Command\leftpstartnum 157
Command\leftrtab 334
Command\leftsidenote 281
Command\leftskip 171, 174, 175
Command\lemma 3, 25, 27–32, 124, 127–129, 131, 132, 134, 359, 420, 421, 428, 429, 431
Command\lemmaseparator 365
Command\let 28, 50, 126, 130, 242, 259, 306, 418
Command\letboxXendendlinenum 365
Command\line 180, 183
Command\line@list 99, 115, 131
Command\line@list@stuff 86, 102, 117, 418, 420
Command\line@list@version 104
Command\line@margin 92, 155, 278
Command\line@num 97, 98, 100, 153, 418
Command\line@set 132, 133
Command\lineation 21, 91
Command\linebreak 37
Command\linenum 25, 27, 28, 56, 58, 124, 132, 268, 270, 276, 359, 432
Command\linenum@out 117, 262, 265
Command\linenumberlist 21, 84, 154, 418
Command\linenumberstyle 23, 96, 418, 435
Command\linenumincrement 21, 420
Command\linenummargin 21, 92, 278
Command\linenumr@p 96, 418, 420
Command\linenumrep 96, 420
Command\linenumsep 22, 60, 96, 279
Command\linerangesep@ 255
Command\lineref 262, 268, 276, 365, 426
Command\list@clear 90
Command\list@clearing@reg 420
Command\list@create 90
Command\lock@disp 95
Command\lock@off 111
Command\lock@on 110
Command\lockdisp 22, 95
Command\loop 162, 302
Command\ltab 334, 335, 339
Command\ltabtext 340
Command\m@mmf@prepare 200
Command\makeatletter 149
Command\makehboxofhboxes 176, 178

COMMAND\makeindex 61, 295
COMMAND\makelabel 276
COMMAND\managestanza@modulo 303
COMMAND\marginpar 60, 70, 277, 278, 419
COMMAND\marginparwidth 60, 279
COMMAND\markboth 149
COMMAND\mathchardef 302
COMMAND\maxhXnotes 365
COMMAND\maxhnotesX 48, 71, 364, 423, 424, 426–428
COMMAND\maxlinesinpar@list 102
COMMAND\measurembody 336
COMMAND\measuretbody 337
COMMAND\memorybreak 20
COMMAND\morenoexpands 71, 72, 124, 126
COMMAND\morethantwolines 365
COMMAND\mpfnpos 36, 220, 423, 434
COMMAND\mpnormalfootgroup 419
COMMAND\mpnormalvfootnote 419
COMMAND\msdata 33, 309–311, 435
COMMAND\msdataposition 34
COMMAND\multfootsep 34, 199
COMMAND\multiplefootnotemarker 199
COMMAND\musixtex 424
COMMAND\n@num 420, 427
COMMAND\n@num@ref 427
COMMAND\new@line 118, 419
COMMAND\new@series 126
COMMAND\newcommand 28, 73, 199, 265
COMMAND\newcommandx 28, 29
COMMAND\newhookarg@specific 255
COMMAND\newhookcommand@series 249, 250, 427
COMMAND\newhookcommand@series@reload 250
COMMAND\newhookcommand@toggle@reload 250, 425
COMMAND\newhooktoggle@series 250, 427
COMMAND\newhooktoggle@specific 255
COMMAND\newif 427
COMMAND\newline 37
COMMAND\newlinechar 243
COMMAND\newseries 35, 360, 363, 364
COMMAND\newseries@ 234, 235, 248
COMMAND\newverse 53, 54, 363, 424, 435
COMMAND\next 302
COMMAND\next@action 103
COMMAND\next@actionline 103
COMMAND\next@insert 159
COMMAND\nl@regR 105
COMMAND\no@expands 71, 126, 131, 242, 418
COMMAND\noXlemmaseparator 365
COMMAND\nobreak 191
COMMAND\nocritical 235

Command\noeledsec 68, 363

Command\noendnotes 363

Command\noexpand 361

Command\nofamiliar 252

Command\noindent 18, 19, 52, 53, 144, 430

Command\noindentX 430

Command\nomk@ 255

Command\nonbreakableafternumber 365

Command\nonumberinfootnote 365

Command\norelax 50

Command\normal@footnotemarkX 203

Command\normal@page@break 353

Command\normal@pars 218

Command\normalbfnoteX 420

Command\normalbodyfootmarkX 203

Command\normalfont 434

Command\normalfootfmt 50, 170, 177, 187, 224, 432

Command\normalfootfmtX 204

Command\normalfootfootmarkX 204

Command\normalfootgroup 172

Command\normalfootgroupX 205

Command\normalfootnoterule 167

Command\normalfootstart 171, 174

Command\normalfootstartX 205

Command\normalvfootnote 168, 170

Command\normalvfootnote@inserted 169, 170

Command\normalvfootnoteX 203

Command\notbool 342

Command\notefontsetup 364

Command\notefontsizeX 44, 364

Command\notenumfont 364

Command\notenumfontX 44, 364

Command\notesXwidthliketwocolumns 365

Command\noteswidthliketwocolumnsX 49, 365, 425, 427

Command\num@lines 139, 161

Command\numberlinefalse 20

Command\numberlinetrue 20

Command\numberonlyfirstinline 247, 365

Command\numberonlyfirstintwolines 365

Command\numberpstartfalse 19

Command\numberpstarttrue 19, 40, 362, 421, 429, 435

Command\numberstanza 40

Command\numberstanzafalse 53

Command\numberstanzatrue 53

Command\numlabfont 22, 50, 96

Command\one@line 139

Command\onehalfspacing 430

Command\onlyXpstart 365

Command\onlysideX 241

Command\page@action 100, 109

COMMAND\page@start 100, 420
COMMAND\pagecontents 100
COMMAND\pageparbreak 363
COMMAND\pageref 57, 268
COMMAND\par 27, 37, 144, 218
COMMAND\par@line 139, 161
COMMAND\para@footgroup 174
COMMAND\para@footgroupX 216
COMMAND\para@footsetup 174, 418
COMMAND\para@footsetupX 214, 418, 425
COMMAND\para@vfootnoteX 215
COMMAND\parafootfmt 176, 177, 432
COMMAND\parafootfmtX 216
COMMAND\parafootftm 179
COMMAND\parafootftmX 217
COMMAND\parafootftmsep 364
COMMAND\parafootsep 365, 423, 428
COMMAND\parafootsepX 47, 101, 364, 365, 433
COMMAND\parafootstart 174
COMMAND\parafootstartX 214
COMMAND\paravfootnote 175, 178
COMMAND\parfillskip 177
COMMAND\parindent 430
COMMAND\parindentX 45, 367, 430, 432
COMMAND\parshape 71
COMMAND\parskip 19, 53, 144
COMMAND\pausenumbering 20, 88, 89, 102, 104, 145, 423, 425, 432
COMMAND\penalty 177
COMMAND\pend 2, 7, 17–21, 23, 68, 122, 124, 127, 133, 139, 140, 142–145, 157, 158, 361, 423, 424, 433–435
COMMAND\preXnotes 427, 433
COMMAND\prenotesX 48, 223, 427
COMMAND\prepare@Xgroupbyline 197
COMMAND\prepare@Xprenotes 221
COMMAND\prev@nopb 354
COMMAND\prev@pb 354
COMMAND\prevlineX 101
COMMAND\prevpageX@num 101
COMMAND\print@Xfootnoterule 427
COMMAND\print@Xnotes 257, 258
COMMAND\print@Xnotes@forpages 426
COMMAND\print@eledsection 148
COMMAND\print@footnoteXrule 427
COMMAND\print@leftmargin@eledsection 342
COMMAND\print@lemma 188
COMMAND\print@line 146
COMMAND\print@notesX@forpages 426
COMMAND\print@rightmargin@eledsection 342
COMMAND\printendlines 229, 234, 274, 367, 418, 420
COMMAND\printlinefootnote 189, 190, 426

COMMAND\printlinefootnotearea . . . 190, 191, 426
COMMAND\printlinefootnotenumbers . . . 189
COMMAND\printlines . . . 170, 187, 192, 193, 229, 274, 367, 418, 420, 426, 431
COMMAND\printnpnum . . . 362, 367
COMMAND\printpstart . . . 187
COMMAND\protect . . . 126, 361
COMMAND\providecommand . . . 199, 418
COMMAND\pstart . . . 2, 7, 17–21, 23, 67, 68, 109, 122, 127, 133, 139, 140, 142–144, 148, 158, 347, 361, 420–422, 424, 425, 427–429, 432–435
COMMAND\pstartinfootnote . . . 365
COMMAND\pstartinfootnoteeverytime . . . 365
COMMAND\pstartnum . . . 157
COMMAND\pstartref . . . 55, 262, 269, 423, 435
COMMAND\pstarts . . . 421
COMMAND\raggedX . . . 47
COMMAND\raggedleft . . . 46
COMMAND\raggedright . . . 46
COMMAND\raw@text . . . 139, 140
COMMAND\rbracket . . . 43
COMMAND\read@linelist . . . 102–104
COMMAND\ref . . . 57, 59, 63
COMMAND\refformated@ . . . 274
COMMAND\refformatedwithpage . . . 274
COMMAND\relax . . . 19, 109, 130, 151, 159, 306, 323, 361
COMMAND\renewcommand . . . 71, 364, 367
COMMAND\reset@msd@options@iffullpage . . . 317
COMMAND\resetprevline@ . . . 101
COMMAND\resetprevpage@ . . . 101
COMMAND\resumenumbering . . . 20, 85, 88, 89, 102, 104, 145, 420, 424, 425, 432
COMMAND\right . . . 66
COMMAND\rightctab . . . 335
COMMAND\rightlinenum . . . 22, 96, 418, 420
COMMAND\rightltab . . . 335
COMMAND\rightnoteupfalse . . . 60
COMMAND\rightrtab . . . 335
COMMAND\rightsidenote . . . 281
COMMAND\rightskip . . . 171, 174, 175, 177
COMMAND\rightstartnum . . . 157
COMMAND\rigidbalance . . . 180, 181, 183, 367, 431
COMMAND\rigidbalanceX . . . 180, 367, 431
COMMAND\robustify . . . 38
COMMAND\roman . . . 328, 431
COMMAND\rtab . . . 334–336, 339
COMMAND\rtabtext . . . 336, 340
COMMAND\sameword . . . 29–32, 116, 133–135, 138, 426, 428, 430, 433, 435
COMMAND\sameword@inedtext . . . 134, 135
COMMAND\saweword . . . 134
COMMAND\scriptsize . . . 96
COMMAND\section . . . 67, 420, 434
COMMAND\section@num . . . 85

Command\sectionmark 341

Command\select@lemmafont 50, 164

Command\series 234, 235

Command\series@ 234

Command\seriesatbegin 35, 246, 427

Command\seriesatend 35, 247, 427

Command\set@Xtxtbeforenotes 162

Command\set@line 131

Command\set@line@action 100, 110

Command\set@txtbeforenotesX 162

Command\setSErefonlypageprefixmore 58, 273, 432, 435

Command\setSErefonlypageprefixsingle 58, 273, 432, 435

Command\setSErefprefixmore 58

Command\setSErefprefixsingle 58

Command\setapprefprefixmore 58, 364

Command\setapprefprefixsingle 58, 364, 431

Command\setcommand@series 248

Command\sethangingsymbol 52, 301, 364, 430

Command\sethanginsymbol 51

Command\setistwofollowinglines 195

Command\setl@dlprbox 283

Command\setline 23, 100, 105, 108, 122, 126, 142, 429

Command\setlinenum 23, 105, 109, 122, 418

Command\setmsdatalabel 34

Command\setmsdataposition 435

Command\setmsdataseries 34

Command\setprintendlines 229, 231, 420

Command\setprintlines 193, 195, 229, 420

Command\setsidenotesep 61

Command\setsidenotsep 364

Command\setstanzaindent 303

Command\setstanzaindents 51, 303, 361

Command\setstanzapenalties 303

Command\setstanzavalues 303

Command\settoggle@series 248, 421, 425

Command\showlemma 125, 418, 419

Command\showwordrank 32, 135

Command\sidenote@margin 419

Command\sidenotemargin 60, 419, 424

Command\sidenotesep 364

Command\sidepstartnumtrue 19

Command\skip 171

Command\skipnumbering 23, 112, 123, 420, 427, 428

Command\skipnumbering@reg 427

Command\small 44

Command\special 13

Command\splitmaxdepth 166, 182

Command\splitoff 180

Command\splittopskip 166, 182, 183

Command\stanza 22, 23, 52–54, 306, 364, 366, 430, 435

Command\stanza@hang . . . 305
Command\stanza@line . . . 305
Command\stanzaindent . . . 51, 303, 426
Command\stanzaindent* . . . 51
Command\stanzaindentbase . . . 302
Command\stanzanumwrapper . . . 54
Command\startlock . . . 22, 100, 123, 306
Command\startstanzahook . . . 364
Command\startsub . . . 22, 100, 121
Command\stopmsd . . . 311
Command\stopmsdata . . . 33, 309, 310
Command\strip@pt . . . 174
Command\strutbox . . . 182
Command\sub@action . . . 100, 110
Command\sub@lock . . . 98
Command\sub@off . . . 107, 265
Command\sub@on . . . 107, 265
Command\subline@num . . . 97, 98, 100
Command\sublinenum@rep . . . 418
Command\sublinenumberstyle . . . 23, 96, 418
Command\sublinenumincrement . . . 21
Command\sublinenumr@p . . . 96, 418, 420
Command\sublinenumrep . . . 96, 420
Command\sublineref . . . 55, 262, 268
Command\subsectionmark . . . 341
Command\sw@inthisedtext . . . 128
Command\sw@list@inedtext . . . 131, 138
Command\symlinenum . . . 365
Command\symplinenum . . . 364
Command\sza@penalty . . . 305
Command\tag . . . 426
Command\text . . . 359
Command\text⟨*language*⟩ . . . 45
Command\textbeforenotesX . . . 162
Command\textcolor . . . 72
Command\textheight . . . 71
Command\the . . . 418
Command\thefootnoteA . . . 34
Command\thefootnoteX . . . 422
Command\thelabidx . . . 297
Command\thepage . . . 105
Command\thepstart . . . 19
Command\thepstartL . . . 421
Command\thepstartR . . . 421
Command\thestanza . . . 53
Command\this@line@list@version . . . 117
Command\thisfootnote . . . 207
Command\threecol@begin@insert . . . 183
Command\threecolfootfmt . . . 182, 183, 432
Command\threecolfootfmtX . . . 212

Command\threecolfootgroup 181
Command\threecolfootgroupX 212
Command\threecolfootsetup 181
Command\threecolfootsetupX 211
Command\threecolvfootnote 182
Command\threecolvfootnote@inserted 182
Command\threecolvfootnoteX 212
Command\toendnotes 26, 227, 433
Command\twocolfootfmt 432
Command\twocolfootfmtX 210
Command\twocolfootgroupX 210
Command\twocolfootsetupX 209
Command\twocolvfootnoteX 210
Command\twolines 247, 365
Command\twolines@A 247
Command\twolines@B 248
Command\twolines@C 248
Command\twolinesbutnotmore 365
Command\twolinesonlyinsamepage 365
Command\txtbeforeXnotes 365
Command\txtbeforenotesX 47, 436
Command\txtbeforenotesonlyonceX 47, 436
Command\uline 35
Command\unhbox 175
Command\unpenalty 176–178
Command\unskip 177
Command\unvxh 177, 366
Command\unvxhX 366
Command\upbracefill 334
Command\usingcritext 360, 363
Command\usingedtext 360, 363
Command\vAfootnote 166
Command\vadjust 120
Command\variant 28
Command\vbox 142, 144, 175, 180, 220
Command\vfootnote 166, 171, 175, 182
Command\vl@dbfnote 200, 419
Command\vl@disnote 282
Command\vl@dlsnote 282
Command\vl@dosnote 282
Command\vl@drsnote 282
Command\vnumfootnoteX 420
Command\vsize 48, 71
Command\vsplit 160
Command\waklam 334
Command\waklamec 334
Command\wapunktel 334
Command\wastricht 334
Command\widthX 49, 367, 432
Command\wrap@edcrossref 267, 425

Command\wrapcontentX 45, 433
Command\wrapped@bodyfootmarkX 218
Command\wrapped@footfootmarkX 217
Command\x... 56
Command\xabslineref 269
Command\xdef 90, 306
Command\xflagref 56, 269, 367, 431
Command\xleft@appenditem 90, 125
Command\xlineref 56, 367, 431
Command\xpageref 56
Command\xpstartref 56, 423
Command\xr 59
Command\xright@appenditem 90
Command\xsublineref 56
Command\xxref 56, 270, 276, 423, 426, 433
Command\zz@@@ 418
Environmentastanza 431
Environmentedarrayc 339
Environmentedarrayl 339
Environmentedarrayr 339
Environmentedtabularc 340
Environmentedtabularl 340
Environmentedtabularr 340
Environmentledgroup 76, 288, 367, 431
Environmentledgroupsized 289
Package(r)(e)ledmac 36
PackageEledmac 12, 73, 100, 294, 362, 363, 426, 427
PackageEledpar 427
PackageEtoolbox 76
PackageParallel 369
PackageReledmac 366, 367
Packageamsgen 318
Packageamsmath 318
Packagebabel 45, 72, 328, 431
Packagebiblatex 70
Packagebidi 45, 46, 77, 430
Packageccaption 84
Packagecolor 72
Packagecsquotes 435
Packageedmac 1, 6, 10, 12–14, 73, 192, 199, 262, 303, 359, 369, 418
Packageedstanza 1, 13, 301
Packageeledmac 1, 10, 11, 14–16, 62, 199, 290, 294, 321, 343, 356, 360, 362–364, 422, 424, 426
Packageeledpar 84, 166, 341, 369, 420, 424–426
Packageetex 430
Packageetoolbox 89, 90, 133, 234, 247, 256, 281, 342, 353
Packagefancyhdr 259, 434
Packagefloatrow 70, 367
Packagefootmisc 34, 72, 77, 199, 369, 434
Packagegeometry 15
Packageglossaries 63, 299, 431

Packagehandout 425
Packagehyperlink 242
Packagehyperref 56, 127, 217, 218, 263, 296, 346, 355, 423–425, 432, 435
Packageifluatex 76
Packageifxetex 76
Packageimakeidx 61, 70, 77, 290, 294, 363, 422–424, 426
Packageindextols 298
Packageindextool 363
Packageindextools 61, 70, 77, 85, 290, 294, 298, 363, 426, 431, 434
Packageinputenc 135
Packageledarab 72
Packageledmac 1, 10, 13, 14, 72, 89, 294, 359, 360, 363, 366
Packageledpar 72
Packagememoir 76, 294, 363, 369, 425
Packagemorewrites 70
Packagemusixtex 424
Packageperpage 431
Packagepolyglossia 43, 72, 165, 187, 431
Packageragged2e 46, 76
Packagereledmac 1, 2, 11, 12, 14–17, 20, 21, 24, 25, 28–30, 32, 34, 35, 38, 39, 42, 45, 46, 48, 50, 52, 54, 56–64, 68, 70–74, 91, 93, 99, 100, 104, 107, 117, 126, 129, 130, 133, 158, 167, 171, 175, 187, 199, 224, 235, 239, 240, 248, 256, 267, 269, 271, 276, 277, 294, 314, 321, 341, 342, 355, 356, 363, 364, 366–368, 429, 432, 434
Packagereledpar 1, 4, 6, 8, 16, 19, 35, 49, 56, 58, 59, 68–70, 72, 74, 84, 91, 102, 107, 128, 130, 168, 171, 219, 220, 235, 241, 255, 257, 258, 290, 301, 430, 431, 433, 434
Packagesuffix 76
Packagetabmac 1, 13, 369
Packageulem 35
Packageuninormalize 29
Packagexargs 29, 76
Packagexkeyval 74, 255
Packagexr 5, 59, 60, 277, 432
Packagexref 276
Packagexstring 76, 296

A

\absline@num 1
Abu Kamil Shuja' b. Aslam 13
\actionlines@list 1
\actions@list 1
\add@inserts 1
\add@inserts@next 1
\add@msd@ 1
\add@msdata 1
\add@msdata@firstlineofpage 1
\add@penalties 1
\add@Xgroupbyline 1
\addtol@denvbody 1
Adelard II 13
\advancelabel@refs 1

\advanceline 1, *22*
\Aendnote *25*
\affixline@num 1
\affixpstart@num 1
\affixside@note 1
\Afootnote *25*
\afternoteX *46*
\afterruleX *48*
\ampersand 1, *54*
\append@notesX 1
\append@Xnotes 1
\applabel 1, *58*
\appref 1, *58*
\apprefwithpage 1, *58*
\arrangementX 1, *37*
\arrangementX@normal 1
\arrangementX@threecol 1
\arrangementX@twocol 1
\at@every@pend 1
\AtEndEveryPend 1, *19*
\AtEveryPend 1, *19*
\AtEveryPstart 1, *19*
\AtEveryStanza 1, *53*
\AtEveryStopStanza 1, *53*
\AtStartEveryPstart 1, *19*
\AtStartEveryStanza 1, *53*
\autopar 1, *18*

B

\ballast *71*
\ballast@count 1
Beeton, Barbara Ann Neuhaus Friend 18
\beforeeledchapter 1
\BeforeEveryStopStanza 1
\beforeinsertingX *45*
\beforeinsertion@X 1
\beforenotesX *48*
\beginnumbering 1, *16*
\Bendnote *25*
\Bfootnote *25*
\bhookgroupX *47*
\bhooknoteX *45*
\bodyfootmarkA *34*
\boxfootnotenumbers 1
Bredon, Simon 13
Breger, Herbert 13, 321
Brey, Gerhard 13
Busard, Hubert L. L. 13
\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` *25*
`\Cfootnote` *25*
`\ch@ck@l@ck` 1
`\ch@cksub@l@ck` 1
`\chapter` 1
`\check@pb@in@verse` 1
Chester, Robert of 13
Claassens, Geert H. M. 13
`\colalignX` *46*
Copernicus, Nicolaus 13
`\critext` *359*
`\ctab` 1
`\ctabtext` 1

D

Dekker, Dirk-Jan 72
`\Dendnote` *25*
`\Dfootnote` *25*
`\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@feet@custom@order` 1
`\do@feetX` 1
`\do@insidelinehook` 1
`\do@line` 1
`\do@linehook` 1
`\do@lockoff` 1
`\do@lockoffL` 1
`\do@lockon` 1
`\do@lockonL` 1
`\do@Xfeet` 1
`\doedindexlabel` 1
`\doendnotes` 1, *26*
`\doendnotesbysection` 1, *26*

\doinsidelinehook 1, *24*

\dolinehook 1, *24*

\dosplits 1

Downes, Michael 71, 175, 177

\doxtrafeet 1

\dummy@edindex 1

\dummy@edtext 1

\dummy@edtext@showlemma 1

\dummy@ref 1

E

\edaftertab 1, *66*, *334*

edarrayc (environment) *64*

edarrayl (environment) *64*

edarrayr (environment) *64*

\edatleft 1, *66*

\edatright 1, *66*

\edbeforetab 1, *66*, *334*

\edfilldimen 1

\edfont@info 1

\edindex 1, *61*

\edindexlab 1, *63*

\EDLABEL 1

\edlabel 1, *55*

\edlabelE 1, *57*

\edlabelS 1, *57*

\edlabelSE 1, *57*

\edlineref 1, *55*

\edmakelabel 1, *57*

\edpageref 1, *55*

\edrowfill 1, *65*

\EDTAB 1

\edtabcolsep 1, *64*

\EDTABINDENT 1

\edtabindent 1

\EDTABtext 1

edtabularc (environment) *64*

edtabularl (environment) *64*

edtabularr (environment) *64*

\EDTEXT 1

\edtext 1, *24*

\edvertdots 1, *66*

\edvertline 1, *66*

\Eendnote *25*

\Efootnote *25*

\eled@chapter 1

\eled@section 1

\eled@sectioning@out 1

\eled@subsection 1

\eled@subsubsection 1

\eledchapter 1

\eledchapter* 1

\eledsection 1

\eledsection* 1

\eledsubsection 1

\eledsubsection* 1

\eledsubsubsection 1

\eledsubsubsection* 1

\enablel@dtabfeet 1

\end@lemmas 1

\endashchar 1

\endline@num 1

\endlock 1, *22*

\endminipage 1

\endnumbering 1, *16*

\endpage@num 1

\endprint 1

\endquotation 1

\endquote 1

\endsub 1, *22*

\endsubline@num 1

environments:

- edarrayc *64*
- edarrayl *64*
- edarrayr *64*
- edtabularc *64*
- edtabularl *64*
- edtabularr *64*
- ledgroup *54*
- ledgroupsized *54*
- minipage *54*

Euclid 13

\extensionchars 1, *69*

F

\f@x@l@cks 1

Fairbairns, Robin 34

\first@linenum@out@false 1

\first@linenum@out@true 1

\firstlinenum 1, *21*

\firstseriesX@ 1

\firstsublinenum 1, *21*

\firstXseries@ 1

\fix@page 1

\flag@end 1

\flag@end@later 1

\flag@end@RTL 1

\flag@start 1

\flag@start@later 1

\flag@start@RTL 1

\flagstanza 1, *54*
\flush@notes 1
\fnpos 1, *36*
Folkerts, Menso 13
\footfootmarkA *34*
\footfudgefiddle 1, *71*
\footnote 1
\footnoteA *34*
\footnoteB *34*
\footnoteC *34*
\footnoteD *34*
\footnoteE *34*
\footnotelang@lua 1
\footnotelang@poly 1
\footnoteoptions@ 1
\footnoteXmark *35*
\footnoteXtext *35*
\footsplitskips 1
\fullstop 1

G

Gädeke, Nora 13
\get@edindex@hyperref 1
\get@edindex@ledinnote@command 1
\get@fnmark 1
\get@fnmarkX 1
\get@index@command 1
\get@linelistfile 1
\get@sw@txt 1
\get@thisfootnote 1
\get@thisfootnoteX 1
\getline@num 1
\gl@p 1

H

\h@num 1
\hangindentX *45*
\hidenumbering 1, *23*
\hidenumberingonleftpage 1, *23*
\hidenumberingonrightpage 1
\Hilfsbox 1
\hilfsbox 1
\hilfscount 1
\HILFSskip 1
\Hilfsskip 1
\hilfsskip 1
\hsizethreecolX *46*
\hsizetwocolX *46*
\Hy@raisedlink@left 1
\hyperlinkformat 1

\hyperlinkformatR 1
\hyperlinkR 1

I

\if@addsw 1
\if@edtext@secondarg@ 1
\if@eled@sectioning 1
\if@firstlineofpage 1
\if@led@nofoot 1
\if@ledgroup 1
\if@lemmacommand@ 1
\if@msdata@insertedfrompreviouspage 1
\if@noeled@sec 1
\if@noneed@Footnote 1
\if@RTL 1
\ifat@every@pend@star@ 1
\ifat@every@pstart@star@ 1
\ifautopar@pause 1
\ifbypage@ 1
\ifbypage@R 1
\ifbypstart@ 1
\ifbypstart@R 1
\ifeledmaccompat@ 1
\iffirst@linenum@out@ 1
\ifindtl@innote 1
\ifindtl@notenumber 1
\ifinserthangingsymbol 1
\ifinstanza 1
\ifl@d@dash 1
\ifl@d@elin 1
\ifl@d@esl 1
\ifl@d@pnum 1
\ifl@d@ssub 1
\ifl@d@Xmorethantwolines 1
\ifl@d@Xtwolines 1
\ifl@dend@X 1
\ifl@dhidenumber 1
\ifl@dmemoir 1
\ifl@dpaging 1
\ifl@dpairing 1
\ifl@dprintingcolumns 1
\ifl@dprintingpages 1
\ifl@dskipnumber 1
\ifl@dskipversenumber 1
\ifl@dstartendok 1
\ifl@footmisc 1
\ifl@imakeidx 1
\ifl@indextools 1
\ifledfinal 1, *69*
\ifledgroupnotesL@ 1

\ifledgroupnotesR@ 1
\iflednopbinverse 1
\ifledRcol 1
\ifledRcol@ 1
\ifnocritical@ 1
\ifnoend@ 1
\ifnofamiliar@ 1
\ifnoledgroup@ 1
\ifnoquotation@ 1
\ifnoteschanged@ 1
\ifnumberedpar@ 1
\ifnumbering 1
\ifnumberingR 1
\ifnumberline 1
\ifnumberstanza 1
\ifparapparatus@ 1
\ifparledgroup 1
\ifpst@rtedL 1
\ifseriesbefore 1
\ifsidepstartnum 1
\ifstopmsdata@inserted@ 1
\ifsublines@ 1
\ifwidthliketwocolumns 1
\ifXendinsertsep@ 1
\ifxindy@ 1
\ifxindyhyperref@ 1
\initnumbering@quote 1
\initnumbering@reg 1
\insert@count 0, 1
\insert@msdata 1
\insert@txtbeforenotesX 1
\insert@Xtxtbeforenotes 1
\inserthangingymbol 1
\insertlines@list 1
\insertparafootsepX 1
\inserts@list 1

J

Jayaditya 13

K

Kabelschacht, Alois 162

L

\l@advance@parledgroup@beforenormalnotes 1
\l@d@add 1
\l@d@nums 1
\l@d@section 1
\l@d@set 1
\l@d@Xend 1

\l@dampcount 1
\l@dbfnote 1
\l@dcheckcols 1
\l@dcheckstartend 1
\l@dchset@num 1
\l@dcolcount 1
\l@dcollect@@body 1
\l@dcollect@body 1
\l@dcolwidth 1
\l@dcsnote 1
\l@dcsnotetext 1
\l@dcsnotetext@l 1
\l@dcsnotetext@r 1
\l@ddodoreinxtrafeet 1
\l@dedbeginmini 1
\l@dedendmini 1
\l@demptyd@ta 1
\l@dend@close 1
\l@dend@open 1
\l@dend@stuff 1
\l@dend@Xfalse 1
\l@dend@Xtrue 1
\l@denvbody 1
\l@desnote 1
\l@dfambeginmini 1
\l@dfamendmini 1
\l@dfeetbeginmini 1
\l@dfeetendmini 1
\l@dgetline@margin 1
\l@dgetlock@disp 1
\l@dgetref@num 1
\l@dgetsidenote@margin 1
\l@dgobblearg 1
\l@disnote 1
\l@dlabel@parse 1
\l@dld@ta 1
\l@dlp@rbox 1
\l@dlsn@te 1
\l@dlsnote 1
\l@dmake@labels 1
\l@dmodforedtext 1
\l@dnullfills 1
\l@dnumpstartsL 1
\l@doldold@footnotetext 1
\l@dp@rsefootspec 1
\l@dparsedendline 1
\l@dparsedendpage 1
\l@dparsedendsub 1
\l@dparsedstartline 1
\l@dparsedstartpage 1

\l@dparsedstartsub 1
\l@dparsefootspec 1
\l@dpush@begins 1
\l@drd@ta 1
\l@dref@undefined 1
\l@drestorefills 1
\l@drestoreforedtext 1
\l@drp@rbox 1
\l@drsn@te 1
\l@drsnote 1
\l@dsetmaxcolwidth 1
\l@dskipnumberfalse 1
\l@dskipnumbertrue 1
\l@dtabaddcols 1
\l@dtabnoexpands 1
\l@dunboxmpfoot 1
\l@dunhbox@line 1
\l@dzeropenalties 1
\label *57*
\labelpstartfalse 1
\labelpstarttrue 1, *19*
\labelref@list 1
\labelrefsparseline 1
\labelrefsparsesubline 1
\last@page@num 1
Lavagnino, John 12
\led@check@nopb 1
\led@check@pb 1
\led@err@AutoparNotNumbered 1
\led@err@BadAction 1
\led@err@edtextoutsidepstart 1
\led@err@EdtextWithoutFootnote 1
\led@err@FootnoteNotInSecondArgEdtext 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@toendnotes@outsidenumbering 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@PackageAfterEledmac 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@reinit@index@fornote 1
\led@set@index@fornote 1
\led@toksa 1
\led@toksb 1
\led@warn@AppLabelOutSecondArgEdtext 1
\led@warn@BadAction 1
\led@warn@BadAdvancelineLine 1
\led@warn@BadAdvancelineSubline 1
\led@warn@BadLineation 1
\led@warn@BadLinenummargin 1
\led@warn@BadLockdisp 1
\led@warn@BadSetline 1
\led@warn@BadSetlinenum 1
\led@warn@BadSidenotemargin 1
\led@warn@BadSublockdisp 1
\led@warn@DuplicateLabel 1
\led@warn@edinde@outsidenumbering 1
\led@warn@LineFileObsolete 1
\led@warn@NoFile 1
\led@warn@NoIndexFile 1
\led@warn@NoMarginpars 1
\led@warn@RefUndefined 1
\led@warn@SeriesStillExist 1
\led@warning@hsizeX@deprecated 1
\led@warning@msdatawithoutstop 1
\led@warning@preXnotes@deprecated 1
\led@warning@Xhsize@deprecated 1
ledgroup (environment) *54*
ledgroupsized (environment) *54*
\ledinnernote 1, *60*
\ledinnote 1
\ledinnotehyperpage 1
\ledinnotemark 1
\ledleftnote 1, *60*
\ledlinenum 1
\ledllfill 1

\ledlsnotefontsetup 1, *61*
\ledlsnotesep 1, *60*
\ledlsnotewidth 1, *60*
\lednopb 1, *68*
\lednopbinversetrue *69*
\lednopbnum 1
\ledouternote 1, *60*
\ledpb 1, *68*
\ledpbnum 1
\ledpbsetting 1, *69*
\ledrightnote 1, *60*
\ledrlfill 1
\ledrsnotefontsetup 1, *61*
\ledrsnotesep 1, *60*
\ledrsnotewidth 1, *60*
\ledsectnomark 1
\ledsectnotoc 1
\ledsetnormalparstuff@common 1
\ledsetnormalparstuffX 1
\ledsidenote 1, *60*
\leftctab 1
\leftlinenum 1, *22*
\leftltab 1
\leftnoteupfalse *60*
\leftpstartnum 1
\leftrtab 1
Leibniz 13
\lemma 1, *27*
\letsforverteilen 1
\line@list 1
\line@list@stuff 1
\line@list@version 1
\line@margin 1
\line@num 1
\line@set 1
\lineation 1, *21*
\linenum 1, *27*
\linenum@out 1
\linenumberlist 1, *21*
\linenumberstyle 1, *23*
\linenumincrement 1, *21*
\linenummargin 1, *21*
\linenumr@p 1
\linenumrep 1
\linenumsep 1, *22*
\linerangesep@ 1
\list@clear 1
\list@clearing@reg 1
\list@create 1
\lock@disp 1

\lock@off 1
\lock@on 1
\lockdisp 1, *22*
Lorch, Richard 13
\ltab 1
\ltabtext 1
Luecking, Dan 75

M

\m@mmf@check 1
\m@mmf@prepare 1
\M@sect 1
\makehboxofhboxes 1
\managestanza@modulo 1
\maxhnotesX *48*
Mayer, Gyula 13
\measurembody 1
\measuremcell 1
\measuremrow 1
\measuretbody 1
\measuretcell 1
\measuretrow 1
Middleton, Thomas 13, 98
minipage (environment) *54*
Mittelbach, Frank 12, 13
\morenoexpands 1, *71*
\mp@append@notesX 1
\mp@append@Xnotes 1
\mpfnpos 1, *36*
\mpnormalfootgroup 1
\mpnormalfootgroupX 1
\mpnormalvfootnote 1
\mpnormalvfootnote@inserted 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
\msdata 1, *33*
\msdata@c 1
\msdata@cR 1
\multfootsep 1, *34*

\multiplefootnotemarker 1

N

\n@num 1

\n@num@stanza 1

\new@line 1

\newhookarg@specific 1

\newhookcommand@series 1

\newhookcommand@series@reload 1

\newhooktoggle@series 1

\newhooktoggle@series@reload 1

\newhooktoggle@specific 1

\newseries@ 1

\newverse 1

\NEXT 1

\no@expands 1

\noeledsec *68*

\nomk@ 1

\normal@footnotemarkX 1

\normal@page@break 1

\normal@pars 1

\normalbfnoteX 1

\normalbodyfootmarkX 1

\normalfootfmt 1

\normalfootfmtX 1

\normalfootfootmarkX 1

\normalfootgroup 1

\normalfootgroupX 1

\normalfootnoterule 1

\normalfootnoteruleX 1

\normalfootstart 1

\normalfootstartX 1

\normalvfootnote 1

\normalvfootnote@inserted 1

\normalvfootnoteX 1

\notefontsizeX *44*

\notenumfontX *44*

\noteschanged@false 1

\noteschanged@true 1

\noteswidthliketwocolumnsX *49*

\nulledindex 1

\nullsetzen 1

\num@lines 1

\numberedpar@false 1

\numberedpar@true 1

\numberingfalse 1

\numberingtrue 1

\numberlinefalse *20*

\numberlinetrue *20*

\numberpstartfalse 1, *19*

\numberpstarttrue 1, *19*

\numberstanzafalse *53*

\numberstanzatrue *53*

\numlabfont 1, *49*

O

\old@hsize 1

\one@line 1

optionauxdir 15, 433

optioncontinuousnumberingwithcolumns 432

optioninnnote 431

optioninnote 431

optionlinerangesep 255

optionnocritical 431

optionnoeledsec 347, 433

optionnoend 431

optionnopenalties 71

optionnotenumber 431

P

\page@action 1

\page@num 1

\pagelinesep 1, *62*

\pageref *57*

\par@line 1

\para@footgroupX 1

\para@footsetup 1

\para@footsetupX 1

\para@vfootnoteX 1

\parafootfmt 1

\parafootfmtX 1

\parafootgroup 1

\parafootsepX *47*

\parafootstart 1

\parafootstartX 1

\paravfootnote 1

\parindentX *45*

\pausenumbering 1, *20*

\pend 1, *17*

Plato of Tivoli 13

\postbodyfootmark 1

\prebodyfootmark 1

\prenotesX *48*

\prepare@edindex@fornote 1

\prepare@prenotesX 1

\prepare@Xgroupbyline 1

\prepare@Xprenotes 1

\prev@nopb 1

\prev@pb 1

\prevpage@num 1

\print@eledsection 1
\print@footnoteXrule 1
\print@leftmargin@eledsection 1
\print@lemma 1
\print@line 1
\print@notesX 1
\print@rightmargin@eledsection 1
\print@Xfootnoterule 1
\print@Xnotes 1
\printendlines 1
\printlineendnote 1
\printlineendnotearea 1
\printlinefootnote 1
\printlinefootnotearea 1
\printlinefootnotenumbers 1
\printlines 1
\printnpnum 1
\printpstart 1
\printsymlineendnotearea 1
\printsymlinefootnotearea 1
\printXafternumber 1
\printXbeforenumber 1
\pstart 1, *17*
\pstarteref 1
\pstartnum 1
\pstartref *55*

Q

\quotation 1
\quote 1

R

\raggedX *47*
\raw@text 1
\rbracket 1
\read@linelist 1
\ref *57*
\ref@reg@later 1
\Relax 1
\reledmac@error 1
\reledmac@warning 1
\removehboxes 1
\reset@msd@options@iffullpage 1
\resetprevline@ 1, *101*
\resetprevpage@ 1
\resetprevpage@num *101*
\restore@familiarnotes 1
\restore@notes 1
\restore@sidenotes 1
\resumenumbering 1, *20*

\rightctab 1

\rightlinenum 1, *22*

\rightltab 1

\rightnoteupfalse *60*

\rightrtab 1

\rightstartnum 1

\rigidbalance 1

\rigidbalanceX 1

\rtab 1

\rtabtext 1

S

Sacrobosco 13

\sameword 1, *28*

\sameword@inedtext 1

Schöpf, Rainer 13

\section@num 1

\select@@lemmafont 1

\select@lemmafont 1, *50*

\SEref 1, *57*

\SErefonlypage *57*

\SErefwithpage 1, *57*

\series 1

\seriesatbegin 1, *35*

\seriesatend 1, *35*

\set@line 1

\set@line@action 1

\set@txtbeforenotesX 1

\set@Xtxtbeforenotes 1

\setapprefprefixmore *58*

\setapprefprefixsingle *58*

\setcommand@series 1

\sethangingsymbol 1, *52*

\setistwofollowinglines 1

\setl@dlp@rbox 1

\setl@drpr@box 1

\setline 1, *22*

\setlinenum 1, *23*

\setmcellcenter 1

\setmcellleft 1

\setmcellright 1

\setmrowcenter 1

\setmrowleft 1

\setmrowright 1

\setmsdatalabel 1, *34*

\setmsdataposition 1, *34*

\setmsdataseries 1, *34*

\setnoteswidthliketwocolumnsX@ 1

\setnotesXpositionliketwocolumns@ 1

\setprintendlines 1

\setprintlines 1

\setSErefonlypageprefixmore *58*

\setSErefonlypageprefixsingle *58*

\setSErefprefixmore *58*

\setSErefprefixsingle *58*

\setsidenotesep *61*

\setstanzaindents 1, *50*

\setstanzapenalties 1, *52*

\setstanzavalues 1

\settcellcenter 1

\settcellleft 1

\settcellright 1

\settoggle@series 1

\settrowcenter 1

\settrowleft 1

\settrowright 1

\setXnotespositionliketwocolumns@ 1

\setXnoteswidthliketwocolumns@ 1

\showlemma 1, *69*

\showwordrank 1, *32*

\sidenote@margin 1

\sidenotemargin 1, *60*

\sidepstartnumtrue *19*

\skip@lockoff 1

\skipnumbering 1, *23*

\splitoff 1

\spreadmath 1, *65*

\spreadtext 1, *65*

\stanza 1, *50*

\stanza@count 1

\stanza@hang 1

\stanza@line 1

\stanzaindent 1, *51*

\stanzaindent* 1, *51*

\stanzaindentbase 1, *50*

\stanzanumwrapper 1, *54*

\startlock 1, *22*

\startsub 1, *22*

\stepl@dcolcount 1

\stopmsdata 1, *33*

\strip@szacnt 1

\sub@action 1

\sub@lock 1

\sub@off 1

\sub@on 1

\subline@num 1

\sublinenumberstyle 1, *23*

\sublinenumincrement 1, *21*

\sublinenumr@p 1

\sublinenumrep 1

\sublineref 1, *55*
\sublines@false 1
\sublines@true 1
\sublock@disp 1
\sublockdisp 1
Sullivan, Wayne 13, 71, 84, 88, 175, 176, 262, 301
\sza@penalty 1

T

\tabHilfbox 1
\tabhilfbox 1
\theaddcolcount 1
\theedtext 1
\theendpageline 1
\thefootnoteA *34*
Theodosius 13
\thepageline 1
\thepstart 1, *19*
\thestanza 1, *53*
\thestartpageline 1
\this@line@list@version 1
\threecol@begin@insert 1
\threecolfootfmt 1
\threecolfootfmtX 1
\threecolfootgroup 1
\threecolfootgroupX 1
\threecolfootsetup 1
\threecolfootsetupX 1
\threecolvfootnote 1
\threecolvfootnote@inserted 1
\threecolvfootnoteX 1
\toendnotes 1, *26*
\toendnotes* 1
\twocolfootfmt 1
\twocolfootfmtX 1
\twocolfootgroup 1
\twocolfootgroupX 1
\twocolfootsetup 1
\twocolfootsetupX 1
\twocolvfootnote 1
\twocolvfootnote@inserted 1
\twocolvfootnoteX 1
\txtbeforenotesonlyonceX *47*
\txtbeforenotesX *47*

U

\unvxhX 1

V

Vamana 13

\variab 1
\vbfnoteX 1
\vl@dbfnote 1
\vl@dcsnote 1
\vl@disnote 1
\vl@dlsnote 1
\vl@dosnote 1
\vl@drsnote 1
\vnumfootnoteX 1

W

Whitney, Ron 12
\widthX *49*
\wrap@edcrossref 1
\wrapcontentX *45*
\wrapped@bodyfootmarkX 1
\wrapped@footfootmarkX 1
Wujastyk, Dominik 12

X

\X@atbegininsertion 1
\X@beforeinsertion 1
\X@doreinfeet 1
\xabslineref 1
\Xafterlemmaseparator *43*
\Xafternote *46*
\Xafternumber *41*
\Xafterrule *48*
\Xaftersymlinenum *41*
\Xarrangement 1, *37*
\Xarrangement@normal 1
\Xarrangement@paragraph 1
\Xarrangement@threecol 1
\Xarrangement@twocol 1
\Xbeforeinserting *45*
\Xbeforelemmaseparator *43*
\Xbeforenotes *48*
\Xbeforenumber *39*, *41*
\Xbeforesymlinenum *41*
\Xbhookgroup *47*
\Xbhooknote *45*
\Xboxlinenum *41*
\Xboxlinenumalign *42*
\Xboxsymlinenum *41*
\Xcolalign *46*
\xedindex 1
\xedlabel 1
\xedtext 1
\Xendafterenumber *41*
\Xendafterlemmaseparator *43*

\Xendafternote *49*

\Xendafterpagenumber *42*

\Xendaftersymlinenum *41*

\Xendahookinplaceofnumber *43*

\Xendahooklinenumber *43*

\Xendbeforelemmaseparator *43*

\Xendbeforenumber *41*

\Xendbeforepagenumber *42*

\Xendbeforesymlinenum *41*

\Xendbhookinplaceofnumber *43*

\Xendbhooklinenumber *42*

\Xendbhooknote *45*

\Xendboxendlinenumalign *42*

\Xendboxlinenum *42*

\Xendboxlinenumalign *42*

\Xendboxstartlinenumalign *42*

\Xendboxsymlinenum *42*

\Xendhangindent *45*

\Xendinplaceoflemmaseparator *43*

\Xendinplaceofnumber *41*

\Xendinplaceofpagenumber *39*

\Xendlemmadisablefontselection *44*

\Xendlemmafont *44*

\Xendlemmaseparator *43*

\Xendlineprefixmore *42*

\Xendlineprefixsingle *42*

\Xendlinerangeseparator *39*

\Xendmorethantwolines *40*

\Xendnonumber *40*

\Xendnotefontsize *44*

\Xendnotenumfont *44*

\Xendnumberonlyfirstinline *38*

\Xendnumberonlyfirstintwolines *38*

\Xendpagenumberonlyfirst *38*

\Xendparagraph *49*

\Xendsep *49*

\Xendsublinesep *40*

\Xendsymlinenum *38*

\Xendsympagenum *39*

\Xendtwolines *40*

\Xendtwolinesbutnotmore *40*

\Xendwrapcontent *45*

\xflagref 1

\Xgroupbyline *47*

\Xgroupbylineseparetwolines *47*

\Xhangindent *45*

\Xhsizethreecol *46*

\Xhsizetwocol *46*

\Xinplaceoflemmaseparator *43*

\Xinplaceofnumber *41*

\Xinsertparafootsep 1
\Xledsetnormalparstuff 1
\xleft@appenditem 1
\Xlemmadisablefontselection *44*
\Xlemmafont *44*
\Xlemmaseparator *43*
\Xlinerangeseparator *39*
\xlineref 1, *56*
\Xmaxhnotes *48*
\Xmorethantwolines *39*
\Xnolemmaseparator 1, *43*
\Xnonbreakableafternumber *41*
\Xnonumber *40*
\Xnotefontsize *44*
\Xnotenumfont *44*
\Xnoteswidthliketwocolumns *49*
\Xnumberonlyfirstinline *38*
\Xnumberonlyfirstintwolines *38*
\Xonlypstart *40*
\Xpagelinesep *41*
\xpageref 1, *56*
\Xparafootsep *47*
\Xparindent *45*
\Xprenotes 1, *48*
\Xprenotes@ 1
\Xpstart *40*
\Xpstarteverytime *40*
\xpstartref 1, *56*
\XR@test 1
\XR@test@mac 1
\XR@test@mac@test 1
\Xragged *47*
\xright@appenditem 1
\Xrigidbalance 1
\Xstanza *40*
\Xstanzaseparator *40*
\Xstorelineinfo 1
\xsublineref 1, *56*
\Xsublinesep *40*
\Xsymlinenum *38*
\Xtoendnotes *26*
\Xtwolines *39*
\Xtwolinesonlyinsamepage *39*
\Xtxtbeforenotes *47*
\Xtxtbeforenotesonlyonce *47*
\Xunvxh 1
\Xwidth *49*
\Xwrapcontent *45*
\Xwrapendlemma *45*
\Xwraplemma *44*

\xxref 1, *56*

Z

\zz@@@ 1

Change History

v0.1.0.

- General: First public release 1

v0.2.0.

- `\ifl@dmemoir`: Added `\ifl@dmemoir` for memoir class having been used 76
- `\morenoexpands`: Added `\l@dtabnoexpands` to `\no@expands` 126
- `\reledmac@error`: Added `\eledmac@error` and replaced error messages 78
- General: Added tabmac code, and extended indexing 1

v0.2.1.

- `\@lab`: Removed page setting from `\@lab` 265
- `\doxtrafeet`: Renamed `\doxtrafeet` to `\l@ddoxtrafeet` 256
- `\edlabel`: Tweaked `\edlabel` to get correct page numbers 262
- `\l@ddodoreinxtrafeet`: Renamed `\dodoreinxtrafeet` to `\l@ddodoreinxtrafeet` 257
- `\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 126
- `\zz@@@`: Minor change to `\zz@@@` 262
- General: Added text about normal labeling 57
  - Bug fixes and match with mempatch v1.8 1
  - Major changes to insert code when memoir is loaded 259

v0.2.2.

- `\footfudgefiddle`: Added `\footfudgefiddle` 173
- `\line@list@stuff`: Added initial write of page number in `\line@list@stuff` 118
- `\para@footsetup`: Added `\footfudgefiddle` to `\para@footsetup` 174
- `\para@footsetupX`: Added `\footfudgefiddle` to `\para@footsetupX` 214
- General: Improved paragraph footnotes 1
  - New Dekker example 1
  - Used `\providecommand` for `\@gobblethree` to avoid clash with the amsfonts package 84

v0.3.0.

- `\@lab`: Replaced `\the\line@num` by `\linenumr@p\line@num` in `\@lab`, and similar for sub-lines 265
- `\@nl@reg`: Added a bunch of code to `\@nl` for handling `\setlinenum` 105
- `\ledlinenum`: Added `\linenumr@p` and `\sublinenum@rep` to `\leftlinenum` and `\rightlinenum` 96
- `\linenumberlist`: Added `\linenumberlist` mechanism 84
- `\printendlines`: Added `\linenumr@p` and `\sublinenumr@p` to `\printendlines` 231
- `\printlines`: Added `\linenumr@p` and `\sublinenumr@p` to `\printlines` 196
- `\sublinenumr@p`: Added `\linenumberstyle` and `\sublinenumberstyle` 96
- General: Includes edstanza and more 1

v0.3.1.

- General: Not released. Added remarks about the parallel package 1

v0.4.0.

- `\@iiiminipage`: Modified kernel `\@iiiminipage` and `\endminipage` to cater for critical footnotes 287
- `\Xarrangement@normal`: Added minpage footnote setup to `\footnormal` 168
- `\edtext`: Added `\showlemma` to `\edtext` 127
- `\l@dfeetendmini`: Added `\l@dfeetbeginmini`, `\l@dfeetendmini` and all their supporting code 285

- \mpnormalfootgroup: Added \mpnormalfootgroup 172
- \mpnormalvfootnote: Added \mpnormalvfootnote 170
- \showlemma: Added \showlemma 84
- General: Added final/draft options 74
    - Added ledgroup environment 288
    - Added ledgroupsized environment 289
    - Added minipage, etc., support 1

v0.4.1.

- \do@Xfeet: Changed \do@Xfeet code for easier extensions 257
- \edindex: Let eledmac take advantage of memoir's indexing 294
- \print@Xnotes: Added \@opXfeet 257
- General: Added code for changing \@doclearpage 260
    - Not released. Minor editorial improvements and code tweaks 1
    - Only change \@footnotetext and \@footnotemark if memoir not used 199

v0.5.0.

- \@footnotetext: Enabled regular \footnote in numbered text 200
- \@xympar: Eliminated \marginpar disturbance 277
- General: Added left and right side notes 278
    - Added sidenotes, familiar footnotes in numbered text 1

v0.5.1.

- \affixline@num: Changed \affixline@num to cater for sidenotes 153
- \l@dgetsidenote@margin: Added \sidenotemargin and \sidenote@margin 278
- General: Added moveable side note 278
    - Fixed right line numbers killed in v0.5 1
    - Only change \hsize in ledgroupsized environment otherwise page number can be in wrong place 289

v0.6.0.

- \@lopR: Added \@pend,\@pendR, \@lopL and \@lopR in anticipation of parallel processing 107
- \@nl@reg: Added \fix@page to \@nl 105
    - Extended \@nl to include the page number 105
- \fix@page: Added \last@page@num and \fix@page 106
- \get@thisfootnote: Changed \l@dbfnote and \vl@dbfnote as originals could give incorrect markers in the footnotes 200
- \new@line: Extended \new@line to output page numbers 118
- General: Fixed long paragraphs looping 1
    - Fixed minor typos 1
    - Prepared for eledpar package 1

v0.7.0.

- \@nl@reg: Added \@nl@reg 105
- \@ref@reg: Added \@ref@reg 114
- \affixline@num: Added skipnumering to \affixline@num 153
- \do@actions@fixedcode: Added \do@actions@fixedcode 152
- \do@actions@next: Added number skipping to \do@actions 151
- \do@insidelinehook: Added \do@linehook for use in \do@line 149
- \endnumbering: Changed \endnumbering for eledpar 88
- \f@x@l@cks: Added \ch@cksub@l@ck, \ch@ck@l@ck and \f@x@l@cks 156
- \footsplitskips: Added \footsplitskips for use in many footnote styles 166
- \get@linelistfile: Added \get@linelistfile 104
- \initnumbering@reg: Added \initnumbering@reg 86

`\l@advance@parledgroup@beforenormalnotes`: Added `\l@dunboxmpfoot` containing some common code . . . 288
`\l@dcsnotetext@r`: Added `\l@demptyd@ta` . . . 149
`\l@dgetline@margin`: Added `\l@dgetline@margin` . . . 92
`\l@dgetlock@disp`: Added `\l@dgetlock@disp` . . . 95
`\l@dgetsidenote@margin`: Added `\l@dgetsidenote@margin` . . . 278
`\l@dnumpstartsL`: Added `\l@dnumpstartsL`, `\ifl@dpairing` and `\ifpst@rted` for/from eledpar . . . 84
`\l@drsn@te`: Added `\l@dlsn@te` and `\l@drsn@te` for use in `\do@line` . . . 149
`\l@dzeropenalties`: Added `\l@dzeropenalties` . . . 144
`\ledlinenum`: Added `\ledlinenum` for use by `\leftlinenum` and `\rightlinenum` . . 96
`\line@list@stuff`: Deleted `\page@start` from `\line@list@stuff` . . . 118
`\list@clearing@reg`: Added `\list@clearing@reg` . . . 103
`\n@num`: Added `\n@num` . . . 112
`\normalbfnoteX`: Removed extraneous space from `\normalbfnoteX` . . . 206
`\resumenumbering`: Changed `\resumenumbering` for eledpar . . . 89
`\setprintendlines`: Added `\setprintendlines` for use by `\printendlines` . . . 229
`\setprintlines`: Added `\setprintlines` for use by `\printlines` . . . 193
`\skipnumbering`: Added `\skipnumbering` and supports . . . 123
`\sublinenumincrement`: Added `\firstlinenum`, `\linenumincrement`, `\firstsublinenum` and `\linenumincrement` . . . 94
`\sublinenumr@p`: Using `\linenumrep` instead of `\linenumr@p` . . . 96
Using `\sublinenumrep` instead of `\sublinenumr@p` . . . 96
`\vnumfootnoteX`: Removed extraneous space from `\vnumfootnoteX` . . . 208
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 . . . 77
Replaced all `\interAfootnotelinepenalty`, etc., by just `\interfootnotelinepenalty` . . . 1
Tidying up for eledpar and ledarab packages . . . 1

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.
`\l@dnumpstartsL`: Added `\ifledRcol` and `\ifnumberingR` for/from `eledpar` . . . 84
General: For compatibility with eledpar, possibility to use `\autopar` on the right side. . . 1
Possibility to number `\pstart`. . . . 19

- Possibility to number the pstart with the commands `\numberpstarttrue`. 1

v0.12.1.

- General: Don’t number `\pstarts` of stanza. 1
  - The numbering of `\pstarts` restarts on each `\beginnumbering`. 1

v0.13.0.

- `\managestanza@modulo`: New stanzaindentsrepetition counter to repeat stanza indents every *n* verses. 303
- General: New stanzaindentsrepetition counter to repeat stanza indents every *n* verses. 51
  - New stanzaindentsrepetition counter: to repeat stanza indents every *n* verses. 1

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.

- `\edlabel`: Tweaked `\edlabel` to get correct line number if the command is first element of a paragraph. 262
- General: Tweaked `\edlabel` to get correct line number if the command is first element of a paragraph. 1

v0.15.0.

- `\affixline@num`: Line numbering can be disabled. 153
- `\ifinserthangingsymbol`: New management of hangingsymbol insertion, preventing undesirable insertions. 301
- `\printlines`: Line numbering can be reset at each pstart. 196
- General: Line numbering can be reset at each pstart. 91
  - Possibility to print `\pstart` number inside. 19

v0.17.0.

- `\ifinserthangingsymbol`: New new management of hangingsymbol insertion, preventing undesirable insertions. 301

v1.0.0.

- `\morenoexpands`: Change to be compatible with new features 126
- General: `\lemma` can contain commands. 27
  - Debug in lineation command 21
  - New generic commands to customize footnote display. 36, 247
  - Options nonum and nosep in `\Xfootnote`. 25
  - Options of `\Xfootnotes`. 164
  - Possibility to have commands in sidenotes. 60
  - Some compatibility break with eledmac. Change of name: eledmac. 1

v1.0.1.

- General: Correction on `\Xnumberonlyfirstinline` with lineation by pstart or by page. 38

v1.1.0.

- `\Xprenotes`: New skip `\Xprenotes@` 222
- `\settoggle@series`: `\settoggle@series` switch the global value of the toggle, not only the local value. 248
- General: Add `\labelpstarttrue`. 19
  - Add `\Xnumberonlyfirstintwolines` 38
  - Add `\Xpstart` and `\Xonlypstart` 40
  - New hook to add arbitrary code at the beginning of the notes 45
  - New options for block of notes. 47
  - New package option: parapparatus. 1
  - New tools to change order of series 246
  - Sectioning commands. 67

v1.2.0.

`\Xprenotes`: Debug in familiar footnotes (bug introduced by v1.1). 222

`\endquote`: Compatibility of `\ledchapter` with the *memoir* class. 340

v1.3.0.

`\endquote`: *Quotation* and quote environment inside numbered sections. 340

v1.4.0.

`\edtext`: Compatibility of `\edtext` with the right-to-left direction (with Polyglossia). 127

`\ledsetnormalparstuffX`: Direction of footnotes with polyglossia. 218

`\newseries@`: Remembers the language of the lemma, in order to create a correct direction for the footnote separator. 237

`\rbracket`: Switch the right bracket to a left bracket when the lemma is RTL (needs polyglossia or LuaTeX). 187

General: Compatibility with LuaTeX of RTL notes. 1

v1.4.1.

`\affixside@note`: Remove spurious spaces. 284

`\endquote`: New option *noquotation*. 340

`\get@thisfootnote`: Compatibility of standard footnotes with `eledmac` when these footnotes contain any commands. 200

`\labelrefsparsesubline`: Fix bug with `\edlabel`. 264

v1.4.2.

General: Debug with some special classes. 1

v1.4.3.

General: Add `\Xnonbreakableafternumber`. 41

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`. 340

v1.4.8.

General: Corrects a bug with parallel texts introduced by 1.1. 1

v1.4.9.

`\normalbfnoteX`: Allow to redefine `\thefootnoteX` with alpḫ when some packages are loaded. 206

v1.5.0.

`\do@insidelinehook`: Added `\do@insidelinehook` for use in `\do@line` 149

`\edindex`: Compatibility with `imakeidx` package, and possibility to use multiple index with `\edindex`. 294

General: Correct indexing when the call is made in critical notes. 290

v1.5.1.

`\managestanza@modulo`: Correct stanzaindentsrepetition counter 303

`\normalvfootnoteX`: Fix bug with normal familiar footnotes when mixing RTL and LTR text. 203

v1.6.0.

`\newverse`: Add `\falseverse` macro. 306

v1.6.1.

`\AtStartEveryPstart`: Spurious space in `\pstart`. 140

- `\ifinserthangingsymbol`: Hang verse is now not automatically flush right. 301
- `\l@dunhbox@line`: Move the call to `\inserthangingsymbol` to allow use `\hfill` inside. 145
- `\pend`: Spurious space in `\pend`. 142
- General: Corrects a false hanging verse when a verse is exactly the length of a line. 1

v1.7.0.

- General: New features for managing page breaks. 68

v1.8.0.

- `\endquote`: Correction of sectioning commands in parallel texts. 340
- `\get@index@command`: Debug `\get@index@command` and compatibility with `hyperref` package. 293
- `\newhookcommand@series@reload`: Debug `\beforenotesX` and `\maxhnotesX` which did not work. 250
- `\prevpage@num`: Correct `\parafootsep` when using with ledgroup. 179
- General: Compatibility with parledgroup option of eledpar package. 1
  - If `imakeidx` and `hyperref` are loaded, adds hyperref in the index. 290

v1.8.1.

- General: Debug endnotes when more than one series is used (change the position where tools for endnotes are defined). 223

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. 290

v1.9.0.

- `\doxtrafeet`: Add `\fnpos` to choice the order of footnotes. 256
- `\l@dfeetendmini`: Add `\mpfnpos` to choice the order of footnotes in minipage / ledgroup. 285

v1.10.0.

- `\endquote`: Correction of sectioning commands in parallel texts. 340
- General: Add `\pstartref` and `\xpstartref` to refer to a pstart number (extension of `\edlabel`). 1

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 environnment. 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. 44

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. 105
- \AtStartEveryPstart: New optional argument for \pstart, to execute code before it. 140
- \edindex: Use correctly default index when imakeidx is loaded. 294
- \endquote: \ledxxx sectioning commands are deprecated and replaced by \eledxxx commands. 340
- \initnumbering@reg: \beginnumbering is defined only on eledmac, not on eledpar. 86
- \l@dgetsidenote@margin: \sidenotemargin is now directly defined in eledmac to be able to manage eledpar. 278
- \l@disnote: \l@dlsnote,\l@drsnote and \l@dcsnote defined only one time, in eledmac, including needs for eledpar case. 279
- \l@dnumpstartsL: Add \ifledRcol@ for eledpar 84
- \l@dunhbox@line: \do@line is split in more little commands. 146
- \newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work when called after \footparagraphX. 250
  - Debug \Xbeforenotes and \Xmaxhnotes which did not work when called after \footparagraph. 250
- \pend: New optional argument for \pend, to execute code after it. 142
- \stanza: &can have an optional argument: content to be printed after. 306
  - \Stanza can have an optional argument: content to be printed before. 306
  - Add \newverse macro, \falseverse deprecated. 306
- General: Add \ledinnernote and \ledouternote commands. 60
  - Add \Xendparagraph and related settings. 49
  - Add hyperlink to crossref (needs hyperref package). 55
  - Compatibility with musixtex. 1
  - Debug eledmac sectioning command after using \resumenumbering. 1
  - Ensure that imakeidx is loaded *before* eledmac 290
  - New hooks: \Xafterrule and \afterruleX 48
  - New options for ragged-paragraph notes 47
  - New sectioning commands. 67
  - Optional arguments for \pstart and \pend. 18

v1.12.1.

- \wrap@edcrossref: Fix spurious spaces. 267

v1.12.2.

- \l@dunhbox@line: Fix a bug with critical notes at the tops of pages (added by v12.0.0) 145

v1.12.3.

- \flag@end: \flag@start and \flag@end are now defined only one time for eledmac and eledpar 119
  - \flag@start send a error message when a \edtext is done without insert (note) 119
- \reledmac@error: Replaced error messages 78
- General: Add macros for new messages since v0.7 77
  - 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. 60
  - Debug left and right notes (bugs added by 1.12.0) 1
  - Underline lemma in \eledxxx when using draft mode. 1

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 296
  - `\@ssect`: Debug `\eledchapter` in parallel with `memoir` 343
  - `\doinsidelinehook`: Added `\dolinehook` and `\doinsidelinehook` 149
  - `\endnumbering`: Allow to mix parallel columns and normal text when using `\pausenumbering` 88
  - `\l@dgobblearg`: `\l@dgobblearg` becomes `\l@dgobbeloptarg` 323
  - `\l@drestoreforedtext`: Debug optional arguments of `\Xfootnote` in tabular context 324
  - `\resumenumbering`: Debug `\resumenumbering` 89
- v1.12.7.
  - `\wrap@edcrossref`: `\wrap@edcrossref` is now robust 267
- 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 119
- v1.13.0.
  - `\newhooktoggle@series@reload`: Add `\newhookcommand@toggle@reload` 250
  - `\para@footsetupX`: In `\para@footsetupX`, use `\columnwidth` instead of `\hsize` 214
  - `\settoggle@series`: `\settoggle@series` can take an optional arguments to reload series setup. 248
  - General: Add `\Xnoteswidthliketwocolumns` and `\noteswidthliketwocolumnsX` 49
    - Added widthliketwocolumns option 74
- v1.13.1.
  - `\ifat@every@pstart@star@`: Add `\l@dzeropenalties` in `\pstart` 140
  - 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
- v1.13.2.
  - `\l@dnumpstartsL`: Add `\ifl@dpaging` for `eledpar` 84
  - General: Fix bug with normal footnotes, added by v1.13.0. 1
- 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.
  - `\edindex`: Let eledmac take advantage of imakeidx even when memoir class is used 294
  - General: Debug spurious characters before endnotes. 223
    - Delete previous override of `\l@d@@wrindexhyp` at the beginning of a document when hyperref is not loaded. 298
    - Move gobbling command 84
    - Provide `\@gobblefour` 84
- v1.14.1.
  - `\@ssect`: Debug sectioning commands when using both `handout` and `hyperref` package. 346
- v1.14.2.
  - `\@ssect`: Debug `\edtext` after starred sectioning commands when using `memoir` class. 343
- v1.15.0.
  - `\@edtext@level`: New boolean `\if@edtext@`. 127
  - `\arrangementX@threecol`: Correct bug with paragraphed familiar footnotes setting. 213
  - `\endsub`: Restore subline feature (disabled by mistake in v1.8.0). 121
  - `\if@lemmacommand@`: New boolean `\iflemmacommand@`. 132

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`. 19
New tools to prevent ambiguous references in lemma 28

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. 118

v1.16.0.
General: `\edtext` is now defined only in `eledmac`, not in `eledpar`. Debug wrong numbering when using `\sameword` + `eledpar` + `\tag` command. 127
Compatibility of standard footnotes with some biblatex styles. 1
New `\stanzaindent` command. 1

v1.16.1.
`\xlineref`: `\lineref` is not defined if defined by some other package, like lineno. `Eledmac` provides `\edlineref` instead. 268

v1.17.0.
`\edtext`: Error message when calling `\edtext` outside of a numbered paragraph. . . . 127

v1.18.0.
`\@edindex@hyperref`: Fix spurious space with `\edindex` when using `imakeidx/indextools` + hyperref. 296
`\edlabel`: `\edlabel` is now defined only one time for both `eledmac` and `eledpar` . 262
`\l@d@section`: Option parapparatus works for endnotes. 224
`\l@dnumpstartsL`: Add `\ifl@dprintingpages` and `\@dprintingcolumns` for `eledpar` 84
`\print@line`: Compatibility with LuaLaTeX RTL languages. 146
`\printlinefootnote`: Code refactoring in `\printlinefootnote`: the printing of the numbers are factorized in `\printlinefootnotearea` 189
`\printpstart`: Debug `\Xpstart` with parallel pages and columns (eledpar) 187
General: Add `\Xpstarteverytime` 40
Compatibility with LuaLaTeX RTL languages. 1
Debug `\Xonlypstart` when using `\Xnumberonlyfirstinline` and the current line number differs from the previous. 40

v1.19.0.
`\footsplitskips`: `\footsplitskips` doesn't set `\floatingpenalty` to `\@MM` when processing parallel pages. 166
`\xxref`: `\xxref` works also with right side numbers, when `\@Rlineflag` is not empty. 270
General: `\Xmaxhnotes` and `\maxhnotesX` work now for both two-columns and three-columns setting. 1
Compatibility with `eledpar` v.1.13.0. 1

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.
`\printlines`: Added `\ifl@d@Xmorethantwolines` and `\ifl@d@Xmorethantwolines` to `\printlines` 196
`\stanza`: & and \& can be preceded by spaces. 306
`\xxref`: Debug `\xxref` when not loading `eledpar` (fix bug added in 1.19.0). 270
General: Add `\Xendboxlinenum` 42
Add `\Xtwolines` and `\Xmorethantwolines` hooks 39
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 LuaLaTeX. 1

Fix issues with RTL text in notes when using LuaLaTeX. 1

Options fulllines in `\Xfootnote`. 25

The `\newif`s are not followed by boolean values set to false, because it is the TeX default setting. 1

v1.21.0.

`\@edindex@hyperref`: Look at the hyperindex option of hyperref before inserting hyperref 296

`\l@d@section`: `\endnotes` take five arguments. 224

`\ledinnotemark`: Add `\ledinnotemark`. 293

`\ledsetnormalparstuffX`: `\ledsetnormalparstuff` is deprecated and becomes `\ledsetnormalparstuffX` and `\Xledsetnormalparstuff`. 218

`\n@num`: `\n@num@ref` deleted 112

`\n@num` defined only one time for both `Eledmac` and `Eledpar` 112

`\newhookcommand@series`: `\newhookcommand@series` can take an optional argument. 249

`\newhooktoggle@series`: `\newhooktoggle@series` can take an optional argument. 250

`\print@footnoteXrule`: Code refactoring: the spaces after the footnote rules are directly managed in `\print@Xfootnoterule` and `\print@footnoteXrule` 221

`\seriesatend`: Fix spurious space in `\seriesatend` 247

`\skipnumbering`: `\skipnumbering` defined only one time for both `Eledmac` and `Eledpar`. 123

Correct `\skipnumbering` for stanza. 123

Delete `\skipnumbering@reg`. 123

General: `\AtEveryPstart` and `\AtEveryPend` are now compatible with `\autopar` 1

`\Xafterrule` and `\afterruleX` features no longer create problems of overflowing at the bottom of the page. 1

`\chapter` inside optional argument of `\pstart` works when typesetting parallel pages 1

`\preXnotes` and `\prenotesX` features no longer create problems of overflowing at the bottom of the page. 1

`\seriesatbegin` and `\seriesatbegin` more efficient 246

Add `\applabel` and related 57

Add `\beforenotesX` and `\Xbeforenotes` features for notes set in two and three column. 1

Add `\hidenumbering` 23

Add `\Xcolalign` and `\colalignX` hooks 46

Add `\Xendtwolines`, `\Xendmorethantwolines`, `\Xendtwolinesbutnotmore` and `\Xendtwolinesonlyinsamepage`. 40

Add `\Xparindent` and `\hangindentX` 45

Add `\Xtwolinesbutnotmore` and `\Xtwolinesonlyinsamepage`. 1

Add nocritical, noend, nofamiliar and noledgroup options. 1

Add noeledsec package option 1

Debug `\beforenotesX` `\maxhnotesX` `\noteswidthliketwocolumnsX` and `\afterruleX` with footnotes set in two and three columns. 1

Fix bug when a `\Xfootnote` follows a `\Xendnote` in the second argument of `\edtext` (bug added in eledmac 1.0.0). 1

- Fix bug with \maxhnotesX when using \foottwocolX or \footthreecolX. 1
- Fix bug with space between columns with notes in two columns (bug added in v1.13.0). 1
- Fix spurious space after first page number in \doendnotes. oldprintnpnumspace option allows to come back to previous setting 1
- parapparatus option works now with familiar footnotes. 1
- Provide \@gobblefive 84

v1.22.0.

- \ledinnote: \ledinnote takes a first optional argument, which is the label for hyperlinks. 293
- General: Add \doendnotesbysection command. 26
  - Add option for lemma separator inside endnotes 43
  - Adds hyperlink for references to notes in indices. 1
  - Fix conflict between noend package option and edtabularx environments 1
  - Provides support for xindy. 1
  - Standardize endnotes handbook. 26
  - When using hyperref package, internal links in index or with \edlineref are now targeted to the top and not longer to the bottom of the lines they refer to. 1

v1.22.1.

- \prevpage@num: Correct double symbol when using both \parafootsep and \Xsymlinenum. 179
- General: Fix bug (added on v1.22.0) with \Xinplaceofnumber hook. 1

v1.23.0.

- \@edtext@level: The boolean \if@edtext@ becomes the counter \edtext@level. 127
- \SErefwithpage: Debug \Xendtwolines, \Xendmorethantwolines, \Xendtwolinesbutnotmore and \Xendtwolinesonlyinsamepage when using \apprefwithpage. 274
- \lemma: Fix spurious space after \lemma command 131
- \newseries@: Prevent spurious spaces when \Afootnote and similar commands are followed by spaces (bug added on 1.0.0). 237
- \sameword: In order to allow use of \sameword with inputenc, we detokenize its mandatory argument before using it in control sequence names. 135
- General: Add \Xboxlinenumalign and \Xendboxlinenumalign. 42
  - Add \Xboxstartlinenum, \Xendboxstartlinenum, \Xboxendlinenum, \Xendboxendlinenum. 42
  - Allow use of \sameword with inputenc managing of UTF-8. 1
  - Compatibility between nofamiliar/nocriticals option and minipage/ledgroup. 1
  - Error message when using \beginnumbering...\endnumbering without \pstart. 1
  - Fix bug with \sameword when the lemma overlaps multiple line. 28
  - Fix bug with \sameword when the same lemma is used for multiple notes or for nested \edtexts. 28
  - Fix bug with \skipnumbering called immediately after a \pstart. 1
  - Fix error of \iftrue not closed. 1
  - Fix spurious space with \skipnumbering (bug added on v1.21.0). 1
  - New tools to ensure the line-list file uses the right version of commands when upgrading the eledmac version. 1
  - Optional argument of \sameword can be a comma-separated list of \edtext depth. 28

v1.23.1.

- General: Fix bug with \lemma command in the right side. 1

v1.23.2.

- General: Compatibility with LaTeX's release 2015. 1

v1.24.0.

- General: We can reinitialize `\AtEveryPstart` and `\AtEveryPend` 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. 287
- `\@xympar`: Patching `\@xympar` instead of redefining it 277
- `\endminipage`: Patch `\endminipage` instead of redefining it. 287
- `\initnumbering@quote`: `\initnumbering@sectcmd` becomes `\initnumbering@quote` 340
- `\l@advance@parledgroup@beforenormalnotes`: Some conde of `\l@dumboxmpfoot` moved to `\l@advance@parledegroup@beforenormalnotes` 288
- `\newseries@`: One endnotes file by series. 243
- General: `\@makecol`, `\@reinserts` and `\@doclearpage` are patched instead of begin redefined 259
  - `\doxtrafeeti` becomes `\Xdo@feet`; `\doxtrafeetii` becomes `\do@Xfeet`; `\@opxtrafeeti` becomes `\@opfeetX`; `\doreinxtrafeetii` becomes `\X@doreinfeet`; `\doreinxtrafeeti` becomes `\@doreinfeetX`. 259
  - 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 LaTeX’s output macros are now patched and not override. 1
  - Package’s name becomes `reledmac`. 1
  - Patch `\@footnotemark` instead of redefine it 200
  - Suppress indexing command specific to `memoir`. 294

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 beginning 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 beginning of a `\pstart`. 1
  - Fix spacing bug with `\Xbhooknote` and `\bhooknoteX` when using them to insert text and not to execute code. 1

New tools to number stanzas . . . 1

v2.1.1.

General: Fix bug with `\ledpbsetting{before}`. . . . 1

v2.1.2.

General: Fix bug with lineation by pstart and tabular environments (added in 2.1.0). . . . 1

v2.1.3.

`\ledsetnormalparstuffX`: Replaced `\noindent` with `\parindent` set to 0pt. . . . 218

General: `\Xhangindent` and `\hangindentX` work now with all the paragraphs in the note. . . . 1

`\Xnoindent` and `\noindentX` work now again (broken in 2.0.0). . . . 1

Change some internal code in order to provide compatibility with LaTeX release of october 2015 . . . 1

Fix bug which inserted double space before paragraphed familiar notes. . . . 1

Fix bug with `\edindex` when using not-Latin characters without UTF-8 engines . . . 1

v2.2.0.

General: Fix bug with combination of `\onehalfspacing` and two columns and three columns notes typeset. . . . 1

Fix bug with some setting command and optimization option. . . . 1

Fix spurious space with paragraphed critical notes when using LuaLaTeX. . . . 1

Increase line list version number to ensure compatibility with new options of `reledpar` package. . . . 1

New setting tools for endnotes: `\Xendnumberonlyfirstinline`, `\Xendnumberonlyfirstintwolines`, `\Xendsymlinenum`, `\Xendbeforenumber`, `\Xendafterenumber`, `\Xendbeforesymlinenum`, `\Xendaftersymlinenum`, `\Xendboxsymlinenum`, `\Xendhangindent`, `\Xendbhooklinenumber`, `\Xendahooklinenumber`, `\Xendbhookinplaceofnumber`, `\Xendahookinplaceofnumber`. . . . 1

v2.2.1.

General: Compatibility with LaTeXformat 2015/10/01. . . . 1

v2.2.2.

General: Fix bug in `\sethangingsymbol`. . . . 1

Fix bug with old version of `etex`. . . . 1

v2.3.0.

General: Disable empty lines as paragraph in stanza. . . . 1

Fix compatibility of paragraphed footnotes with `bidi` v17.9 and following. . . . 1

Warning message when using some setting commands inside rightside environment (deprecated behavior) . . . 1

v2.3.1.

General: Fix spurious space when using optional argument of `\stanza` (introduced in v2.3.0). . . . 1

v2.4.0.

`\footnoteoptions@`: First argument of `\footnoteoption@` is now mandatory, not optional. . . . 164

General: `\Xbhooknote` and `\bhooknoteX` work with notes in columns. . . . 1

Fix bug of `\parindentX` and `\Xparindent` with two columns and three columns notes. . . . 1

Fix bug with `\sameword` in right side. . . . 1

Fix spurious space in two columns and three columns notes. . . . 1

Fix spurious space when using optional argument of stanza (introduced in v2.3.0). . . . 1

New hooks: `\Xlinerangeseparator` and `\Xendlinerangeseparator`. . . . 39

- Option linerangesep for critical footnotes and endnotes. 39

v2.4.1.

- General: Fix bug with `\appref` and `\apprefwithpage` (introduced in v2.4.0). 1
  - Fix bug with tabular environments when using `babel` or `polyglossia` languages that override LaTeX `\roman` command, like Greek language. 1
  - Fix bug with tabular environments when using `babel` or `polyglossia` languages that override LaTeX `\roman` command, like Greek. 1

v2.5.0.

- `\SErefwithpage`: Debug `\setapprefprefixsingle` 273
- `\edlabel`: Fix bug when calling`\edlabel` in a footnotes of the rightside 262
- `\l@d@section`: `\endnotes` take six arguments. 224
- `\printlines`: `\printlines` takes an eighth argument: the line flag 196
- `\xlineref`: `\xlineref` does not include anymore the side flag. Use `\xflagref` to get it. Not that `\edlineref` still contains the flag. 268
- General: `\apprefwithpage` and `\appref` print double quotation mark when the label was not defined. 1
  - `\apprefwithpage` and `\appref` work with right side crossref. 1
  - `\apprefwithpage` works also when `noend` option is enabled. 1
  - `\appref` and `\apprefwithpage` can take linerangesep optional argument. 1
  - `\edlabel` works now in `\Xfootnote`. 1
  - `\lemma` can be used even when the `nocritical` is enabled. 1
  - Compatibility with new hook and tools of `reledpar` 2.6.0. 1
  - Fix spurious vertical space in `astanza` environment (`reledpar`) 1
  - Log now states ‘There were undefined references‘ when using wrong references in `\edlineref` or edpageref. 1
  - New hooks to customize page and line number appearance in endnotes. 1
  - New hooks: `\Xbhookgroup` and `\bhookgroupX`. 1
  - New tools to easily make cross-reference to a passage defined by a start and an end line 57

v2.6.0.

- General: Adds compatibility with `innnote` and `notenumber` options of `indextools` package. 1
  - Fix bug with footnote counter in `ledgroup` (added in v2.5.0). 1
  - Fix bug, introduced in v2.5.0, with footnote numbering in parallel typesetting when using `perpage` package. 1

v2.7.0.

- `\@k`: `\rigidbalance` is split in `\Xrigidbalance` and `\rigidbalanceX`. 180
- `\l@d@section`: `\endnotes` take seven arguments. 224
- General: Add dash as default page range separator for `\SEonlypage` 1
  - Debug `\SErefonlypage` when referring to only one page. 1
  - Delete parenthesis after `\SErefonlypage`. 1
  - Fix (again) bugs with footnote numbering in parallel typesetting while using `ledgroup` environments (bug added in v2.5.0). 1
  - Fix bug with `\SErefwithpage`. 1
  - Fix bugs in compatibility with `innote` and `notenumber` options of `indextools` package, when indexing outside of a `ledgroup`. 1
  - New commands to make glossaries connected to page and linenumber with the `glossaries` package 1
  - New hooks: `\Xhsize` and `\hsizeX` 49
  - New hooks: `\Xlemmafont` and `\Xendlemmafont` 44

New setting commands: `\setSErefonlypageprefixsingle` and `\setSErefonlypageprefixmore` 1
Warning for duplicate and undefined labels are parsable by latexmk 1
Warning for duplicate labels does not send any more a false line and page number . . . 1
When using `hyperref` package, add link in familiar footnotes between the footnote marks in the text and the footnote marks in the footnote 1
When using `hyperref` package, add links for `\SEref` and related, `\appref` and related. 1
When using `hyperref` package, add links from critical footnotes and critical endnotes to the line of text they refers 1

v2.7.1.
General: Debug `\Xbhookgroup` hooks executed on columnar footnotes (moved to a larger group, to take effect). 1

v2.7.2.
General: `\Xhsize` and `\hsizeX` become `\Xwidth` and `\widthX` 49
Fix problem of hyphenation when using `hyperref` package (added in v2.7.0). 1

v2.8.0.
`\l@d@section`: `\Xendhangindent` and `\Xendafternote` can take values which are relative to the font size of the endnote. 224
General: `reledmac` cross-referencing can take advantage of `xr` package. 1
More `\edgls`… commands. 1
No indentation for paragraphed notes in ledgroup. Can be changed with `\Xparindent` and `\parindentX`. 1

v2.8.1.
General: Warnings for undefined labels are really parsable by latexmk 1

v2.8.2.
General: Fix bug concerning indent in a paragraph immediately following a sectioning command (bug NOT fixed on reledpar) 1
Fix bug with `\AtEveryPstart` added in version 2.0.0. 1
Fix bug with vertical space after the between-sectioning command as optional argument of a `\pstart` and `\pstart` content 1

v2.9.0.
General: Allow continuing line numbering between normal text and parallel text, using `\pausenumbering` and `\resumenumbering` and the `continuousnumberingwithcolumns` option. 1
Fix bug when using `\linenum{page}` and `\pausenumbering`…`\resumenumbering`. . 1
Fix bug with three- and two-column footnote setting (added in v.2.4.0). 1
Fix spurious space inside three-column familiar footnote. 1
Write correct metadata in numbered files when using `\pausenumbering`…`\resumenumbering`. 1

v2.9.1.
General: Fix bug when notes start with “plus” or “minus”. 1

v2.9.2.
General: Fix bug with `hyperref` package when a lemma starts with “plus” or “minus” (bug introduced in v. 2.7.0). 1

v2.9.3.
General: Fix bug with line number position and reset added by v. 2.9.0 1

v2.10.0.
`\print@lemma`: Code refactoring between `\parafootfmt`, `\twocolfootfmt`, `\threecolfootfmt` and `\normalfootfmt`. 188

- General: Add \AtEveryStanza and \AtEveryStopStanza. 1
  - Fix bug in \ledlsnotefontsetup and \ledrsnotefontsetup which could not handle \color command properly. 1
  - More specific error messages. 1
  - New hooks: \Xwrapcontent, \Xendwrapcontent and \wrapcontentX. 45
  - New hooks: \Xwraplemma and \Xendwraplemma 44

v2.10.1.

- General: Add ‘nopenalties‘ option. 1
  - Fix bug introduced in v. 1.4: not paragraphed critical footnotes could prevent marginal line number from being displayed 1

v2.11.0.

- \do@actions@fixedcode: Add action 1010 152
- General: Add new tools to produce an apparatus of manuscripts 1
  - Fix bug in \Xparafootsep in parallel typesetting 1
  - Make \parafootsepX work 1
  - Prevent \Xtxtbeforenotes hook from causing notes to go beyond the bottom margin 1

v2.12.0.

- General: \preXnotes becomes \Xprenotes (naming convention) 1
  - Add \hidenumberingonleftpage and \hidenumberingonrightpage 1
  - Add \toendnotes and related. 1
  - Add auxdir option. 1
  - Fix bug in critical and familiar footnotes when using uppercase letters with accent mark 1
  - Fix bug when using \chapter in optional argument of \pstart in parallel typesetting in combination with the noeledsec option. 1
  - Fix bug with \ledinnernote and \ledouternote in parallel typesetting 1
  - Fix bug with familiar footnote number in optional argument of \pstart or \pend in parallel typesetting 1
  - Fix spurious vertical space in \chapter when used as optional argument of \pstart in parallel tyepsetting. 1
  - Make endnote compatible with \sameword mechanism 1
  - More accurate message to control the position of \Xfootnote and \applabel in the LaTeX code 1

v2.13.0.

- General: Version 2.13.0 never existed. 1

v2.13.1.

- General: In critical footnotes, the right side flag is printed only if requested explicitly with \Xlineflag (bug added in v. 2.5.0). 1

v2.13.2.

- General: Fix bug added in v. 11.2 which could make parallel typesetting not work. 1

v2.13.3.

- General: Makes \Xendafterpagenumbe affecting \SErefwithpage 1

v2.14.0.

- General: Hyperref with the line number inside critical footnotes is correct when using \xxref 1
  - Some internal changes for new features of reledpar. 1

v2.14.1.

- General: Fix bug when using \footnoteX in the first argument of \edtext. 1

v2.14.1a.

- General: Fix problematic typos in the handbook. 1

v2.15.0.
- General: Add ‘byline‘ arrangement. 1
  - Fix `\Xtxtbeforenotes` in ledgroup. 1

v2.15.1.
- General: Fix `\edindex` in tabular environments. 1

v2.15.2.
- General: Fix bug with `fancyhdr` package 3.8 and later. 1

v2.15.3.
- General: Fix bug with `\section` in optional argument of `\pstart` and empty line before `\pend` (bug added in v2.8.2). 1
  - Simplification of the sectioning command code. 1

v2.16.0.
- General: `\Xdo@feet` becomes `\do@Xfeet` 259
  - Add `\Xendpagenumberonlyfirst`, `\Xendpagenumberonlyfirstifsingle`, `\Xendpagenumberonlyfirstintwo`, `\Xendinplaceofpagenumber` and `\Xendsympagenum` hooks. 1
  - Add `\Xpagelinesep` hook. 41
  - Compatibility with new features of `reledpar` 1
  - Deleted dead code. 1
  - Display a warning message if using a version of LaTeXthat is too old. 1
  - Fix bug with `\Xgroupbylines` for notes in two columns 1
  - Fix bug with `\Xtxtbeforenotes` for notes in three or two columns 1
  - Fix bug with ‘notenumber’ option of `indextools` package when indexing texts in familiar footnotes. 1
  - Fix potential bug when using `\edindex` in critical footnotes. 1
  - More explicit error message in case the stanza indentation is not defined. 259
  - New options for `\fnpos` and `\mpfnpos` to set a customized order for familiar and critical footnotes. 1
  - When `\edindex` is called outside of a `\beginnumbering`...`\endnumbering` structure, it is automatically switched to `\index`, with a warning message. 1
  - When indexing texts in familiar footnotes with `\edtext`, refer to the line number where the footnote is called. 1
  - When indexing texts in sidenotes with `\edtext`, refer to the line number where the sidenote is called. 1

v2.16.1.
- General: Fix bug with redefinition of the style of the footnote number (bug added in v2.12.0) 1

v2.16.2.
- General: Error message if `footmisc` is loaded after `reledmac`. 1
  - Fix bug introduced by v2.16.1 when using non-expandable control sequence, like `\normalfont`, in the footnote number style. 1

v2.16.3.
- General: Fix bug with `\SEref` (bug added in v2.7.0). 1

v2.16.4.
- General: Fix bug with vertical space before sectioning command in optional argument of `\pstart` (bug added in v2.15.3). 1

v2.16.5.
- General: Fix potential spurious spaces in endnotes. 1

v2.16.6.

- General: Fix bug with the line number style in `\doennotes` when referring to right side line in parallel typesetting. 1
  - Take into account `\linenumberstyle` when using `\edlineref`. 1

v2.16.7.

- General: Fix bug with `\msdata` when using multiple `\beginnumbering`...`\endnumbering`. 1
  - Fix bug with `\numberpstarttrue` when using multiple `\beginnumbering`...`\endnumbering`. 1

v2.16.8.

- General: Fix bug with `\edindex` in footnotes in parallel typesetting. 1

v2.17.0.

- General: Add `\edglsadd` command. 1
  - Add `\setmsdataposition` setting. 1

v2.17.1.

- General: Fix spurious space in paragraphed footnotes when using LuaLaTeXwithout using Right-To-Left text. 1

v2.17.2.

- General: Change log message when numbered files still don't exist, in order to improve compatibility with *latexmk*. 1

v2.17.3.

- General: Fix bug with `\doendnotesbysection` and `\doendnotes`. 1

v2.17.4.

- General: Fix bug with `\setSErefonlypageprefixsingle` and `\setSErefonlypageprefixmore`. 1

v2.17.5.

- General: Fix bug with `\pstartref` when refering to the left side in parallel typesetting. . 1

v2.18.0.

- General: Fix bug when using a `\edtext` in two lines or more in right-to-left typesetting with XeLaTeX. 1
  - Fix bug when using both `\Xnumberonlyfirstintwolines` or `\Xnumberonlyfirstinline` and `\Xparafootsep` and `\Xsymlinenum`. 1

v2.18.1.

- General: Fix bug when using `\msdata` with LuaLaTeXor with the `hyperref` package. . . . 1

v2.19.0.

- General: Add `\footnoteXmark` and `\footnoteXtext` commands. 1
  - Add better compatibility with the `csquotes` package when using familiar footnotes. . 1
  - Fix bug with paragraph indent after sectioning command. 1

v2.20.0.

- General: Add `\AtStartEveryStanza`, `\BeforeEveryStopStanza`, `\AtEndEveryPend`, `\AtStartEveryPstart`. 1
  - Add second optional argument of `\pstart`, `\pend` and `\stanza`. 1
  - Add starred version of `\AtEveryPstart`, `\AtEveryPend`, `\AtEveryStanza` and `\AtEveryStopStanza`. 1
  - Add third and fourth optional argument of `\newverse`. 1
  - Fix bug when using familiar footnotes in `\eledsection` and related. 1
  - Reset font specification at the beginning of familiar footnotes. 1

v2.21.0.

- General: Add the possibility of nested `\sameword`. 1

Fix bug when using formatting command in the argument of `\edindex` inside `\edtext`. 1

Now, as explained in the handbook, an `\edindex` inside `\edtext` only creates index reference to main text, and not to the critical footnote. 1

v2.22.0.

General: Add `\txtbeforenotesonlyonceX` and `\Xtxtbeforenotesonlyonce` hooks. . 1

Add `\txtbeforenotesX` hook. 1

Fix bug added in v2.16.0 when using `\Xtxtbeforenotes` with paragraphed or normal footnotes. 1

Fix bug with three and two columns critical footnotes, broken in v. 2.17.6. 1