

reledmac

Typeset scholarly editions with L^AT_EX*

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** T_EX 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 v2.24.0, last revised 2017/08/17.

[†]maieul at maieul dot net

1.2.4 <code>reledmac</code>	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 <code>\pstart ... \pend</code>	18
5.2.3 Content before specific <code>\pstart</code> and after specific <code>\pend</code>	18
5.2.4 Content before every <code>\pstart</code> and after every <code>\pend</code>	19
5.2.5 Numbering paragraphs (<code>\pstart</code>)	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 hiding 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 Case setting	29
6.3.3 Notes about input encoding with UTF-8 processor	29
6.3.4 Use with \lemma command	30
6.3.5 Sameword for a group of words	31
6.3.6 Customizing	32
6.3.7 Problems with some macros	32
6.3.8 Automatic sameword annotation	33
6.4 Apparatus of Manuscripts	33
6.4.1 Marking sections of text	34
6.4.2 Layout of the apparatus of manuscripts	34
6.4.3 Settings	35
6.5 Familiar notes	35
6.5.1 Basic use	35
6.5.2 Customizing mark	35
6.5.3 Separator for multiple footnotes	35
6.6 Printing the footnote mark without printing the footnote text	36
6.7 Changing series	36
6.7.1 Create a new series	36
6.7.2 Delete series	36
6.7.3 Series order	36
6.8 Position of critical and familiar footnotes	37
7 Critical apparatus appearance	37
7.1 Notes arrangement in a series	38
7.2 Control line number printing	39
7.2.1 Print line number only at first time	39
7.2.2 Print page number only at first time	39
7.2.3 Arbitrary text before line number	40
7.2.4 Separator for line range	40
7.2.5 Abbreviate line range	40
7.2.6 Disable line number	41
7.2.7 Printing pstart number	41
7.2.8 Printing stanza number	41
7.2.9 Separator between line and subline numbers	41
7.2.10 Separator between page and line numbers	42
7.2.11 Space around number	42
7.2.12 Space around line symbol	42
7.2.13 Space in place of number	42
7.2.14 Boxing line number and line symbol	42
7.3 For endnotes	43
7.4 Arbitrary code around line number	43
7.5 Separator between the lemma and the note	44
7.5.1 For footnotes	44
7.5.2 For endnotes	44
7.6 Font style	45
7.6.1 For line number	45

7.6.2 For the lemma	45
7.6.3 For all notes	45
7.7 Wrapping notes	45
7.7.1 Wrapping lemmas	45
7.7.2 Wrapping contents	46
7.8 Indent of notes content	46
7.9 Arbitrary code at the beginning of notes	46
7.10 Arbitrary code before inserting note	46
7.11 Options for footnotes in columns	47
7.11.1 Alignment	47
7.11.2 Size of the columns	47
7.12 Options for paragraphed footnotes and notes grouped by line	47
7.12.1 Mark separation of notes	47
7.12.2 Ragged text	48
7.13 Options for block of notes	48
7.13.1 Grouping notes by line	48
7.13.2 Text before notes	48
7.13.3 Code before notes	48
7.13.4 Spacing	49
7.13.5 Rule	49
7.13.6 Maximum height	49
7.13.7 Width	50
7.14 Footnotes and the reledpar columns	50
7.15 Endnotes in one paragraph	50
8 Fonts	50
9 Verse	51
9.1 Basic	51
9.2 Define stanza indents	51
9.3 Repeating stanza indents	52
9.4 Manual stanza indent	52
9.5 Stanza breaking	53
9.6 Hanging symbol	53
9.7 Long verse and page break	53
9.8 Content before/after verses	53
9.9 Numbering stanza	54
9.10 Various tools	55
9.11 Notes on empty lines	55
10 Grouping	55

11 Cross referencing	56
11.1 Basic use	56
11.2 Cross-referencing to a critical note	57
11.3 Cross-referencing which return a number in any case	57
11.3.1 Cross-referencing in order to define line number of a critical note	57
11.4 Not automatic cross-referencing	58
11.5 Normal \LaTeX cross-referencing	58
11.6 References to start and end lines	58
11.6.1 Reference to main text lines	58
11.6.2 References to lines that are commented on in the apparatus	58
11.6.3 Settings	59
11.7 Compatibility with <code>xr</code> package	60
12 Side notes	61
12.1 Basics	61
12.2 Setting	61
12.2.1 Width	61
12.2.2 Vertical position	61
12.2.3 Distance to the main text	61
12.2.4 Font	62
12.2.5 Separator between notes	62
13 Indexing	62
13.1 Basics	62
13.2 Referring to critical notes	62
13.3 Separator between page and line numbers	63
13.4 Using <code>xindy</code>	63
13.5 Advanced setting	64
14 Glossary	64
14.1 Preamble setting	64
14.2 Commands	64
15 Tabular material	65
16 Sectioning commands	68
16.1 Sectioning commands without line numbers or critical notes	68
16.2 Sectioning commands with line numbering and critical notes	68
16.3 Optimization	69
17 Quotation environments	69
18 Page breaks	69
18.1 Control page breaking	69
18.2 Prevent page break in a long verses	70

19 Miscellaneous	70
19.1 Known and suspected limitations	71
19.1.1 Non-standard geometry	71
19.1.2 floatrow package compatibility	71
19.1.3 ‘No room for a new’	71
19.1.4 Marginal notes	71
19.1.5 Paragraph shape	72
19.1.6 Paragraphed footnotes	72
19.1.7 Use with other packages	72
19.2 Parallel typesetting	73
I Implementation overview	74
II Preliminaries	74
II.1 Links with original edmac	74
II.2 Package declaration	74
II.3 Package options	75
II.4 Loading packages	77
II.5 Compatibility with LuaTeX	77
II.6 Boolean flags	78
II.7 Messages	79
II.8 Gobbling	85
II.9 Miscellaneous commands	85
II.10 Prepare reledpar	86
II.11 Booleans provided by other optional packages which are required in any case	86
III Sectioning commands	86
IV List macros	91
V Line counting	93
V.1 Choosing the system of lineation	93
V.2 Line number margin	94
V.3 Line number initialization and increment	95
V.4 Line number locking	96
V.5 Line number style	97
V.6 Line number printing	98
V.7 Line number counters and lists	99
V.8 Line number locking counter	100
V.9 Line number associated to lemma	100
V.10 Reading the line-list file	104
V.11 Commands within the line-list file	106
V.12 Writing to the line-list file	118

VI Marking text for notes	126
VI.1 <code>\edtext</code> itself	126
VI.2 Substitute lemma	133
VI.3 Substitute line numbers	134
VI.4 Lemma disambiguation	135
VII Paragraph decomposition and reassembly	142
VII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	142
VII.2 Processing one line	148
VII.2.1 General process	148
VII.2.2 Process for “normal” line	149
VII.2.3 Process for line containing <code>\eledsection</code> command	151
VII.2.4 Hooks	151
VII.2.5 Sidenotes and marginal line number initialization	152
VIII Line and page number computation	153
IX Line number printing	156
X Pstart number printing in side	161
XI Restoring footnotes and penalties	162
XI.1 Add insertions to the vertical list	162
XI.2 Penalties	164
XI.3 Printing leftover notes	165
XI.4 Text before notes	166
XII Critical footnotes	168
XII.1 Fonts	168
XII.2 Individual note options	168
XII.3 Notes language	169
XII.4 General survey of the way we manage notes	170
XII.5 General setup	170
XII.6 Footnotes arrangement	171
XII.6.1 User level macro	171
XII.6.2 Normal footnote	171
XII.6.3 Paragraphed footnotes	177
XII.6.4 Columnar footnotes	184
XII.7 Critical notes presentation	191
XII.7.1 Font tools	191
XII.7.2 Pstart number in footnote	191
XII.7.3 Lemma printing	192
XII.7.4 Line number printing	193
XII.7.5 Footnote grouped by line	201

XIII Familiar footnotes	203
XIII.1 Adjacent footnotes	203
XIII.2 Regular footnotes for numbered texts	204
XIII.3 Footnote formats	206
XIII.4 Footnote arrangement	207
XIII.4.1 User level macro	207
XIII.4.2 Normal footnotes	207
XIII.4.3 Two columns footnotes	213
XIII.4.4 Three columns footnotes	215
XIII.4.5 Paragraphed footnotes	217
XIII.5 Wrapping footnote marks in hyperlink	221
XIV Code common to both critical and familiar footnote in normal arrangement	222
XV Footnotes' width for two columns	223
XVI Footnotes' order	224
XVII Footnotes' rule	224
XVIII Specific skip for first series of footnotes	225
XVIII.0.1 Overview	225
XVIII.0.2 User level command	226
XVIII.0.3 Internal commands	226
XIX Endnotes	227
XIX.1 Internal commands	227
XIX.2 User level commands	231
XIX.2.1 Inserting contents to endnotes	231
XIX.2.2 Printing endnotes	232
XX Generate series of notes	238
XX.1 Test if series is still existing	239
XX.2 Init specific to <code>reledpar</code>	239
XX.3 For critical footnotes	239
XX.3.1 Options	239
XX.3.2 Create inserts, needed to add notes in foot	241
XX.3.3 Create commands for critical apparatus, <code>\Afootnote</code> , <code>\Bfootnote</code> etc.	241
XX.3.4 Set standard display	244
XX.4 For familiar footnotes	244
XX.4.1 Options	244
XX.4.2 Create tools for familiar footnotes (<code>\footnoteX</code>)	245
XX.5 The endnotes	247
XX.5.1 The auxiliary file	247
XX.5.2 The main macro	247

XX.5.3 Tools	248
XX.5.4 Internal commands	248
XX.5.5 The options	249
XX.6 Init standards series (A,B,C,D,E)	250
XXI Setting series display	250
XXI.1 Change series order	250
XXI.2 Test series order	251
XXI.2.1 Get the first series	251
XXI.3 Series setting	251
XXI.3.1 General way of working	251
XXI.3.2 Tools to set options	252
XXI.3.3 Tools to generate options commands	253
XXI.3.4 Options for critical notes	255
XXI.3.5 Options for familiar notes	256
XXI.3.6 Options for endnotes	257
XXI.4 Hooks for a particular footnote	259
XXI.5 Alias	259
XXII Output routine	260
XXII.1 Extra footnotes output	260
XXII.2 Patching standard output's commands	263
XXIII Cross referencing	266
XXIII.1 Compatibility with xref	280
XXIV Side notes	281
XXV Minipages and such	289
XXVI Indexing	294
XXVI.1 Looking on package order	294
XXVI.2 Auxiliary macros for <code>\edindex</code>	294
XXVI.3 Code specific to <code>\edindexin</code> critical footnotes	295
XXVI.4 Analysis of command in indexed text	297
XXVI.5 Code for the formatted index	297
XXVI.6 Main code	298
XXVI.7 Hyperlink	299
XXVI.8 'innote' and 'notenumber' option of <code>indextols</code> package	302
XXVII Glossaries	303
XXVIII Verse	305
XXVIII.1 Hanging symbol management	305
XXVIII.2 Using & character	306
XXVIII.3 Code category setting	306
XXVIII.4 Stanza count and indent	306

XXVIII.5 Numbering stanza	308
XXVIII.6 Stanza number in note	309
XXVIII.7 Main work	309
XXVIII.8 Restore catcode and penalties	312
XXIX Apparatus of Manuscripts	313
XXIX.1 User level macro	313
XXIX.2 Setting macro	314
XXIX.3 Counters and lists	315
XXIX.4 Auxiliary file macros	315
XXIX.5 Action macro	316
XXIX.6 Inserting footnote	321
XXIX.7 Other	321
XXX Arrays and tables	322
XXX.1 Preamble: macro as environment	322
XXX.2 Tabular environments	325
XXX.2.1 Disabling and restoring commands	325
XXX.2.2 Counters, boxes and lengths	329
XXX.2.3 Tabular typesetting	332
XXX.2.4 Environments	343
XXXI Quotation's commands	344
XXXII Section's title commands	345
XXXII.1 Commands to disable some feature	345
XXXII.2 General overview	345
XXXII.3 \beforeeledchapter command	346
XXXII.4 Auxiliary commands	346
XXXII.5 Patching standard commands	347
XXXII.6 Main code of \eledxxx commands	352
XXXII.7 Macros written in the auxiliary file	355
XXXIII Page breaking or no page breaking depending of specific lines	357
XXXIV Long verse: prevents being separated by a page break	359
XXXV Tools for hyperref package	359
XXXVI Compatibility with eledmac	360
Appendix A Things to do when changing versions	363
Appendix A.1 Migrating from edmac to ledmac	363
Appendix A.2 Migration from ledmac to eledmac	364
Appendix A.3 Migration to eledmac 1.5.1	365
Appendix A.4 Migration to eledmac 1.12.0	365
Appendix A.5 Migration to eledmac 17.1	366

Appendix A.6 Migration to eledmac 1.21.0	366
Appendix A.6.1 \Xledsetnormalparstuff and \ledsetnormalparstuffX	366
Appendix A.6.2 Endnotes	366
Appendix A.7 Migration to eledmac 1.22.0	366
Appendix A.8 Migration to eledmac 1.23.0	366
Appendix A.9 Migration from eledmac to reledmac	367
Appendix A.9.1 Risk of ‘no room for a new’	367
Appendix A.9.2 Multiple indices with memoir	367
Appendix A.9.3 Deprecated commands and options	367
Appendix A.9.4 \renewcommand replaced by command	368
Appendix A.9.5 Commands the names of which have been changed	368
Appendix A.9.6 Endnotes	370
Appendix A.9.7 Z Series	370
Appendix A.9.8 Internal commands	370
Appendix A.10 Migration to reledmac 2.1.0	370
Appendix A.11 Migration to reledmac 2.1.3	370
Appendix A.12 Migration to reledmac 2.3.0	370
Appendix A.13 Migration to reledmac 2.4.0	371
Appendix A.14 Migration to reledmac 2.5.0	371
Appendix A.15 Migration to reledmac 2.7.0	371
Appendix A.16 Migration to reledmac 2.7.2	371
Appendix A.17 Migration to reledmac 2.8.0	371
Appendix A.18 Migration to reledmac 2.13.1	371
Appendix A.19 Migration to reledmac 2.18.0	372
Appendix A.20 Migration to reledmac 2.21.0	372
Appendix A.21 Migration ro reledmac 2.24.0	372
References	373
Index	373
Change History	422

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 de-

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

scription by John and Dominik of this version of edmac was published as ‘An overview of edmac: a PLAIN T_EX format for critical editions’, *TUGboat* 11 (1990), pp. 623–643.

From 1991 through 1994, the macros continued to evolve, and were tested at a number of sites. We are very grateful to all the members of the (now defunct) edmac@mailbase.ac.uk discussion group who helped us with smoothing out the bugs and infelicities in the macros. Ron Whitney and our anonymous reviewer at the TUG were both of great help in ironing out last-minute wrinkles, while Ron made some important suggestions which may help to make future versions of edmac even more efficient. Wayne Sullivan, in particular, provided several important fixes and contributions, including adapting the Mittelbach/Schöpf ‘New Font Selection Scheme’ for use with PLAIN T_EX 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 *Rithmarchia* 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 Generali Quinqueecclesiensi 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.

²Gerhard 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).

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

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

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

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

⁷Menso Folkerts, ‘Die *Rithmarchia* des Werinher von Tegernsee’, *ibid.*

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

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

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

¹¹Leibniz, *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>)

¹²Being prepared at Poona and Lausanne Universities.

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

In March 2003 Peter Wilson started an attempt to port `edmac` from TeX to LaTeX. The starting point was `edmac` version 3.16 as documented on 19 July 1994 (available from CTAN). In August 2003 the `tabmac` functions were added; the starting point for these being version 1.0 of October 1996. The `edstanza` (v0.01) functions were added in February 2004. Sidenotes and regular footnotes in numbered text were added in April 2004. This port was called `ledmac` (\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. 37). 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. 364.

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. 367.

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 \XeTeX . 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 \TeX 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 \TeX '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. 49.

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 \TeX auxiliary files (`.aux`, `.toc`, ...).

eledmac-compat help to migrate from `eledmac` to `reledmac` (see Appendix A.9.5 p. 368).

swcaseinsensitive make `\sameword` command case insensitive.

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. 69).

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. 63).

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. 68).

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. 55). 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` 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`.

¹³For 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. 51) or `reledpar` (see 19.2 p. 73).

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

5.2.4 Content before every `\pstart` and after every `\pend`

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

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

```
\setcounter{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 Lua \TeX or Xe \TeX , 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 \TeX .

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

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

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

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

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

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

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

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

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

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

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

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

`\linenumberlist` 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 permissible values for `<location>` are `left`, `right`, `inner`, or `outer`: for example, `\linenummargin{inner}`. The package's default setting is

```
\linenummargin{left}
```

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

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

5.3.5 Other settings

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

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

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

You can change the separator between line number and subline number either by using `\Xsublinesep` without any optional argument (7.2.9 p. 41) 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` The `\startlock` command, used in running text, locks the line number at its current
`\endlock` value, until you insert `\endlock`. It can tell for itself whether you are in a patch of line or sub-line numbering. One use for line-number locking is in printing poetry: there the line numbers should be those of verse lines rather than of printed lines, even when a verse line requires several printed lines. But in this case you may use the `\stanza` mechanism, see 9 p. 51.

`\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` In some cases you may want to modify the line numbers that are automatically calculated.
`\advanceline`

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

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

5.4.4 Line number style

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

Alph Uppercase letters (A ... Z).

alph Lowercase letters (a ... z).

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

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

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

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

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

5.4.5 Skipping and hiding number

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

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

`\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 :	1	I am happy : I saw my friend Smith on
I saw my friend <code>\edtext{Smith}{</code>	2	Tuesday.
<code>\Afootnote{Jones C, D.}}</code>		
on Tuesday.		
		<hr style="width: 20%; margin-left: 0;"/>
	1	Smith] Jones C, D.

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

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

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

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

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` 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. 36.

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. 40).
- `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. 44).
- `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` **Inserting endnotes** The package also maintains five separate series of endnotes.

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

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

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

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

- `fulllines` to disable `\Xendtwolines` and `\Xendmorethantwolines` features for this particular note (cf. 7.2.5 p. 40).
- `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. 44).
- `linangesep= $\langle c \rangle$` to change to $\langle c \rangle$ 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{ $\langle s \rangle$ }`, where $\langle s \rangle$ is the letter of the series to be printed. Put this command where you want the corresponding set of endnotes printed. In this case, all the endnotes of the $\langle s \rangle$ series are printed, for all numbered sections.

\doendnotesbysection However, you may want to print the endnotes of one given series covering the first numbered section, then the endnotes of another given series covering the first numbered section, then the endnotes of the first given series covering the second numbered section, then the endnotes of the second given series covering the second numbered section, and so forth. In this case, use `\doendnotesbysection{ $\langle s \rangle$ }`. For each value of $\langle s \rangle$, 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. 44).

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

toendnotes **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[$\langle series \rangle$]{ $\langle content \rangle$ }` where $\langle series \rangle$ is a comma-separated list of the series of endnotes where $\langle content \rangle$ must be inserted. If $\langle series \rangle$ is empty, then $\langle content \rangle$ 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-expanded¹⁴ 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:

```
\Attoendnotes*{\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          1 I am happy : I saw my friend Smith on
  \edtext{Smith}{\Afootnote{Jones 2 Tuesday.
    C, D.}} on Tuesday.}
{\lemma{I \dots\ Tuesday.}
  \Bfootnote{The date was
    July 16, 1954.}
}
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.

¹⁴The 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. 56) 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

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

¹⁵We 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 Case setting

By default, `\sameword` is sensitive to the case. E.g. “foo” is considered as a different word to “Foo”.

You can use the `swcaseinsensitive` when loading the package to make `\sameword` insensitive to the case, i.e. to consider “foo” as identical “Foo”.

6.3.3 Notes about input encoding with UTF-8 processor

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

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

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

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

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

- If you use \XeTeX , add this line in your preamble: `\XeTeXinputnormalization 1`.
- If you use \LuaTeX , use the `uninormalize` package of Michal Hoftich¹⁶ with the `buffer` option set to true.

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

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

¹⁶<https://github.com/michal-h21/uninormalize>.

6.3.4 Use with `\lemma` command

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

For example, in the following example:

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

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

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

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

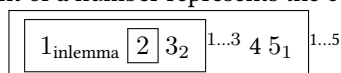
Note that only words that are actually referenced in a `\lemma` need the optional argument. Therefore, the first `\sameword` in the example above should have “1” as its optional argument, to be referenced correctly in the lemma.

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

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

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

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



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

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

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

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

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}%
}%
}
```

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

6.3.5 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}}
```

1 per causam tamen scire causam est per causam cognoscere causam

```
1 causam2] fnote
1 per2 causam3] causam rei B
1 causam4] 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}}
```

1 per causam tamen scire causam est per causam cognoscere causam

```
1 causam2] fnote
1 per causam2] causam rei B
1 causam4] fnote
```

6.3.6 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}%
}
```

6.3.7 Problems with some macros

`\swnoexpands` Macros inside \sameword that are not fully expandable, mainly macros which manipulate font features, write on full or have optional argument, may cause problems during compilation. Custom commands inside \sameword may therefore result in errors saying that “Use of sameword doesn’t match its definition.” To solve this, include a redefinition of your custom commands in the \swnoexpands macro. In order to not include any content of a macro during comparison, identify the command with \@gobble For example:


```

\makeatletter
\appto{\swnoexpands}{%
  \let\somemacro\@gobble%
}
\makeatother

```

This will drop the content of `\somemacro` during comparison.

To include the content of the first and only one argument of a custom command in `sameword` comparison, use the `\@firstofone` command. For example, this is how `\emph` is handled:

```

\makeatletter
\appto{\swnoexpands}{%
  \let\emph\@firstofone%
}
\makeatother

```

To include command which can take optional argument, use `\RenewExpandableDocumentCommand` of `\xparse`. For example, this is how `\edindex` is handled:

```

\makeatletter
\appto{\swnoexpands}{%
  \RenewExpandableDocumentCommand{\edindex}{om}{}%
}
\makeatother

```

6.3.8 Automatic sameword annotation

All potentially ambiguous apparatus entries must be annotated manually. That annotation process is laborious and includes a risk of errors. *Samewords* is a Python script that can automate this step of the process. It can be installed via the *Python Package Index*, but see <https://github.com/stenskjær/samewords> for more info and documentation. The script is still at a beta stage, so comments and questions as well as error reports are very welcome at <https://github.com/stenskjær/samewords/issues>.

Please note that the maintainer of this script is not the maintainer of `reledmac`.

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 \TeX 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. 37). 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` 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 formatted using any of the styles described for the critical footnotes. Note that the 'regular' footnotes have the series letter at the end of the macro name whereas the critical footnotes have the series letter at the start of the name.

6.5.2 Customizing mark

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

`\bodyfootmarkA` `\renewcommand*{\thefootnoteA}{\arabic{footnoteA}}`

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

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

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

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

There are similar command triples for the other series.

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 $\langle X \rangle$ command, `reledmac` provides a `\footnote\langle X \rangle mark` command and a `\footnote\langle X \rangle text` command, equivalent to standard \TeX '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\langle X \rangle nomk` and `\footnote\langle X \rangle 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{\langle s \rangle}` to pull up a given series $\langle s \rangle$ to the beginning, or `\seriesatend{\langle s \rangle}` 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:

$\langle series \rangle \langle type \rangle$

$\langle series \rangle$ is a series letter (A,B,C etc.), while $\langle type \rangle$ 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 $[\langle s \rangle]$, which is the letter of the series — or a list of letters separated by comma — depending on which option is applied. If the optional argument is omitted or empty, the setting will apply to the entire series.

When a length, noted $\langle l \rangle$, is used, it can be stretchable: `a plus b minus c`. The final length m is calculated by \TeX to have: $a - c \leq m \leq a + b$. If you use some relative unit¹⁷, 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. 72.
- `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.

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

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

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

7.2.2 Print page number only at first time

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:

- | | |
|---------------------------------------|--|
| <code>\Xendsympagenum</code> | • <code>\Xendsympagenum[$\langle series \rangle$]{$\langle c \rangle$}</code> to print $\langle c \rangle$ when the page number is not printed. |
| <code>\Xendinplaceofpagenumber</code> | • <code>\Xendinplaceofpagenumber[$\langle series \rangle$]{$\langle l \rangle$}</code> to print a $\langle l \rangle$ length horizontal space in case no symbol is printed instead of the page number. |

7.2.3 Arbitrary text before line number

- | | |
|-----------------------------|--|
| <code>\Xbeforenumber</code> | <code>\Xbeforenumber[$\langle s \rangle$]{$\langle txt \rangle$}</code> allow to insert $\langle txt \rangle$ before the line number, only when the line number is printed, so taking into account <code>\Xnumberonlyfirstinline</code> and similar. |
|-----------------------------|--|

7.2.4 Separator for line range

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

7.2.5 Abbreviate line range

- | | |
|--|--|
| <code>\Xtwolines</code>
<code>\Xmorethantwolines</code> | If a lemma is printed on two subsequent lines, <code>reledmac</code> 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[$\langle s \rangle$]{ $\langle text \rangle$ }` and `\Xmorethantwolines[$\langle s \rangle$]{ $\langle text \rangle$ }`. The $\langle text \rangle$ argument of `\Xtwolines` will be printed if the lemma is on two lines, and the $\langle text \rangle$ argument of `\Xmorethantwolines` will be printed if the lemma is on three or more lines. For example:

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

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

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

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

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.

- | | |
|---------------------------------------|--|
| <code>\Xtwolinesonlyinsamepage</code> | However, you can force print the final page number with <code>\Xtwolinesonlyinsamepage[$\langle series \rangle$]</code> . |
|---------------------------------------|--|

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.

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

7.2.6 Disable line number

`\Xnonumber` You can use `\Xnonumber[⟨s⟩]` if you do not want to have the line number in a footnote.
`\Xendnonumber` `\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` `\Xendbeforenumber` and `\Xendafternumber` are the equivalents of `\Xbeforenumber` and `\Xafternumber` for endnotes.

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

7.2.12 Space around line symbol

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

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

`\Xendbeforesymmlinenum` `\Xendbeforesymmlinenum` and `\Xendaftersymmlinenum` are the equivalents of `\Xbeforesymmlinenum` and `\Xaftersymmlinenum` 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}
```

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

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

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

L to align left (default value);

R to align right;

C to center.

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

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

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

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

`\Xendboxlinenum` `\Xendboxlinenum[⟨s⟩]{⟨l⟩}`, `\Xendboxlinenumalign[⟨s⟩]{⟨text⟩}`, `\Xendboxstartlinenum[⟨s⟩]{⟨l⟩}`, `\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` `\Xendafterpagenumber[⟨s⟩]{⟨text⟩}` defines the text after the page number in endnotes. Default value is `)` (open parenthesis followed by a single space). `\Xendlineprefixsingle[⟨s⟩]{⟨text⟩}` 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.

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

7.5 Separator between the lemma and the note

7.5.1 For footnotes

<code>\Xlemmaseparator</code>	By default, in a footnote, the separator between the lemma and the note is a right bracket (<code>\rbracket</code>) ¹⁸ . You can use <code>\Xlemmaseparator[⟨s⟩]{⟨Xlemmaseparator⟩}</code> to change it. The optional argument can be used to specify the series in which it is used. Note that there is a non-breakable space between the lemma and the separator, but a breakable space between the separator and the following text.
<code>\Xbeforelemmaseparator</code>	Using <code>\Xbeforelemmaseparator[⟨s⟩]{⟨l⟩}</code> you can add some space between lemma and separator. If your lemma separator is empty, this space won't be printed. The default value is 0 em.
<code>\Xafterlemmaseparator</code>	Using <code>\Xafterlemmaseparator[⟨s⟩]{⟨l⟩}</code> you can add some space between separator and note. If your lemma separator is empty, this space will not be printed. The default value is 0.5 em.
<code>\Xnolemmaseparator</code>	You can suppress the lemma separator, using <code>\Xnolemmaseparator[⟨s⟩]</code> , which is simply a alias of <code>\Xlemmaseparator[⟨s⟩]{}</code> .
<code>\Xinplaceoflemmaseparator</code>	With <code>\Xinplaceoflemmaseparator[⟨s⟩]{⟨l⟩}</code> you can add a space if no lemma separator is printed. The default value is 1 em.

7.5.2 For endnotes

<code>\Xendlemmaseparator</code>	By default, there is no separator inside endnotes between the lemma and the content of the note. You can use <code>\Xendlemmaseparator[⟨s⟩]{⟨Xendlemmaseparator⟩}</code> to change this. The optional argument can be used to specify the series in which it is used. A common value of <code>⟨Xendlemmaseparator⟩</code> is <code>\rbracket</code> . Note that there is a non-breakable space between the lemma and the separator, but a breakable space between the separator and the following text.
<code>\Xendbeforelemmaseparator</code>	Using <code>\Xendbeforelemmaseparator[⟨s⟩]{⟨l⟩}</code> you can add some space between the lemma and the separator. If your lemma separator is empty, this space won't be printed. The default value is 0 em.
<code>\Xendafterlemmaseparator</code>	Using <code>\Xendafterlemmaseparator[⟨s⟩]{⟨l⟩}</code> you can add some space between 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.
<code>\Xendinplaceoflemmaseparator</code>	With <code>\Xendinplaceoflemmaseparator[⟨s⟩]{⟨l⟩}</code> you can add some space if you chose to remove the lemma separator. The default value is 0.5 em.

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

7.6 Font style

7.6.1 For line number

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

7.6.2 For the lemma

<code>\lemmadisablefontselection</code>	By default, font of the lemma in footnote is the same as font of the lemma in the main text. For example, if the lemma is in italic in the main text, it is also in italic in note. The <code>\Xlemmadisablefontselection[⟨s⟩]</code> command allows to disable it for a specific series.
<code>\lemmadisablefontselection</code>	By default, font of the lemma in endnote is the same as font of the lemma in the main text. For example, if the lemma is in italic in the main text, it is also in italic in note. The command allows <code>\Xendlemmadisablefontselection[⟨s⟩]</code> to disable it for a specific series.
<code>\Xlemmafont</code> <code>\Xendlemmafont</code>	Use <code>\Xlemmafont[⟨s⟩]{⟨cmd⟩}</code> to apply a \TeX font command to the lemma. For example, to have boldface lemma: <code>\Xlemmafont{\bfseries}</code> <code>\Xendlemmafont[⟨s⟩]{⟨cmd⟩}</code> is the same for endnotes.

7.6.3 For all notes

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

7.7 Wrapping notes

7.7.1 Wrapping lemmas

<code>\Xwraplemma</code>	<code>\Xwraplemma[⟨s⟩]{⟨cmd⟩}</code> is used to wrap, in the footnote, the lemma in a \TeX com-
--------------------------	---

mand. For example, with the `bid` package, to ensure having a lemma written right to left, use `\Xwraplemma{\RL}`.

`\Xwrapendlemma` `\Xendwraplemma[\langle s \rangle]{\langle cmd \rangle}` is the same for endnotes.

7.7.2 Wrapping contents

`\Xwrapcontent` `\Xwrapcontent[\langle s \rangle]{\langle cmd \rangle}` is used to wrap the footnote contents — excluding the lemma — in a \LaTeX command.

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

`\Xendwrapcontent` `\Xendwrapcontent[\langle s \rangle]{\langle cmd \rangle}` is the same for endnotes.

`\wrapcontentX` `\wrapcontentX[\langle s \rangle]{\langle cmd \rangle}` is the same for familiar footnotes.

7.8 Indent of notes content

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

`\parindentX` By default, `reledmac` does not add indentation before the paragraphs inside familiar footnotes. Use `\parindentX[\langle s \rangle]` to enable indentation.

`\Xhangindent` For critical notes NOT paragraphed you can define an indent with `\Xhangindent[\langle s \rangle]{\langle l \rangle}`, 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[\langle s \rangle]{\langle l \rangle}`, 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[\langle s \rangle]{\langle l \rangle}`, 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[\langle s \rangle]{\langle code \rangle}` is to be used at the beginning of the critical footnotes.

`\bhooknoteX` `\bhooknoteX[\langle s \rangle]{\langle code \rangle}` is to be used at the beginning of the familiar footnotes.

`\Xendbhooknote` `\Xendbhooknote[\langle s \rangle]{\langle code \rangle}` is to be used at the beginning of the endnotes.

7.10 Arbitrary code before inserting note

`\Xbeforeinserting` `\Xbeforeinserting[\langle s \rangle]{\langle code \rangle}` and `\beforeinsertingX[\langle s \rangle]{\langle code \rangle}` are very technical commands.

`beforeinsertingX`

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 `bidirectional`, 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. 45)).

Note that the changes are local to the footnote.

7.11 Options for footnotes in columns

7.11.1 Alignment

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

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

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

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

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

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

7.12 Options for paragraphed footnotes and notes grouped by line

7.12.1 Mark separation of notes

`\Xafternote` You can add some horizontal space after a note by using `\Xafternote[⟨s⟩]{⟨l⟩}` (for `\afternoteX`

critical footnotes) or `\afternoteX[⟨s⟩]{⟨l⟩}` (for familiar footnotes). The default value is `1em plus.4em minus.4em`.

`\Xparafootsep` For paragraphed footnotes (see below), you can choose the separator between each
`\parafootsepX` 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 crit-
ical 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. 39).

`\Xgroupbylineseparetwolines` Note that the `\Xafternote` and `\Xparafootsep` settings are used to determine
space and content between footnotes (7.12 p. 47). 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. 39)

7.13.2 Text before notes

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

`\txtbeforenotesX` 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

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

`\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 L^AT_EX footnote rule used by `reledmac` decreases by 3pt. This 3pt decrease is not changed by this command.

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

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

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

Be careful, the standard L^AT_EX 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 todoclist.

7.13.7 Width

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

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

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

Note that changes the width 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. 47.

7.14 Footnotes and the reledpar columns

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

`\noteswidthliketwocolumnsX`

- `\Xnoteswidthliketwocolumns[⟨s⟩]` to create critical notes with a two-column size width.
- `\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. 53)

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.

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

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

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

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

If you want the hanging verse to be flush right, you can use `\sethanginsymbol:` see p. 9.6 p. 53.

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 `stanzaindentrepetition` counter at n . For example:

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

is like

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

Be careful: the feature is changed in `eledmac` 1.5.1. See Appendix A.3 p. 365.

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

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

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

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

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 `\setstanzaindent` for this verse is skipped, and `{<value>}` is used instead.

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

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

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

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

`\sethangingsymbol`

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

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

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

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. 70 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.
```

- | | |
|----------------------------------|--|
| <code>\AtEveryStanza</code> | <ul style="list-style-type: none"> • Use <code>\AtEveryStanza{⟨arg⟩}</code> to automatically add content before the stanza (not in the same paragraph). <p>Note that a <code>\noindent</code> will be inserted before the argument, and, consequently, a <code>\parskip</code>. You can use the starred version of <code>\AtEveryStanza</code> to avoid this <code>\noindent</code>.</p> |
| <code>\AtStartEveryStanza</code> | <ul style="list-style-type: none"> • Use <code>\AtStartEveryStanza</code> to automatically add content at the beginning of stanza (in the same paragraph). • <code>&</code> can be replaced by <code>\newverse</code> with two optional arguments (in brackets). The first will be printed after the current verse, the second before the next verse. <p>A <code>\noindent</code> is automatically inserted before the contents of these optional arguments.</p> <p>Use a third and fourth optional argument to not add these <code>\noindents</code> (to add content respectively after the current verse / before the next verse).</p> <ul style="list-style-type: none"> • Use <code>\AtEveryPend{⟨arg⟩}</code> to automatically add content after verses (including the final one) and <code>\AtEveryPstart{⟨arg⟩}</code> to automatically add content before verses (including the first one). • <code>\&</code> can take an optional argument (in brackets). Its content will be printed after the stanza. |
| <code>\AtEveryStopStanza</code> | <ul style="list-style-type: none"> • Use <code>\AtEveryStopStanza</code> to automatically add content after the end of stanzas (not in the same paragraph). <p>Note that a <code>\noindent</code> will be inserted before the argument, and, consequently, a <code>\parskip</code>. You can use the starred version of <code>\AtEveryStopStanza</code> to avoid this <code>\noindent</code>.</p> |
| <code>\AtStartEveryStanza</code> | <ul style="list-style-type: none"> • Use <code>\AtStartEveryStanza</code> to automatically add content at the end of stanza (in the same paragraph). |

9.9 Numbering stanza

- | | |
|---------------------------------|--|
| <code>\numberstanzatrue</code> | <p>If you want to automatically number stanzas, use <code>\numberstanzatrue</code>. In this case, the line number will restart at each <code>\stanza</code>.</p> <p>If you want to disable this feature again, use <code>\numberstanzafalse</code>.</p> <p>You can use this feature in combination with <code>\Xstanza</code> (7.2.8 p. 41).</p> |
| <code>\numberstanzafalse</code> | |
| <code>\thestanza</code> | <p>You can redefine <code>\thestanza</code> to change the aspect of stanza number. Default value is:</p> |

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

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

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

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

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. 53).

10 Grouping

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

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

You can also put familiar footnotes (see section 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.

`ledgroupsize` The `ledgroupsize` environment is similar to `ledgroup` except that you must specify a width for the environment, as with a `minipage`.
`\begin{ledgroupsize}[⟨pos⟩]{⟨width⟩}`.

The required $\langle width \rangle$ argument is the text width for the environment. The optional $\langle pos \rangle$ argument is for positioning numbered text within the normal textwidth. It may be one of the characters:

- l (left) numbered text is flush left with respect to the normal textwidth. This is the default.
- c (center) numbered text is in the center of the textwidth.
- r (right) numbered text is flush right with respect to the normal textwidth.

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

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

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>}`. $\langle lab \rangle$ can be almost anything you like, including letters, numbers, punctuation, or a combination—anything but spaces; you might type `\edlabel{toves-3}`, for example.¹⁹

`\edpageref` Elsewhere in the text, either before or after the `\edlabel`, you can refer to its

`\edlineref` location with `\edpageref{<lab>}`, `\edlineref{<lab>}`, `\edsublineref{<lab>}` or

`\sublineref` `\pstartref{<lab>}`, that will produce, respectively, the page, line, sub-line and pstart

`\pstartref` 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 `ℳℒX.aux` file. You will need to process your document through `ℳℒX` 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.)

¹⁹More precisely, you should stick to characters in the `ℒX` 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. 58).

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 \TeX is looking for a number, such a warning will lead to a complaint that the number is missing. This is the case for references used within the argument to `\linenum`, for example (see 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}
\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` 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 end-notes.

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

11.6.3 Settings

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

For example, you may use:

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

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

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

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

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

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

Also note that `\setSerefonlypageprefixsingle` is only a shortcut for `\XendbeforepagenumberSerefonlypage` (see 11.6.3 p. 59). 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 \TeX commands `\label` and `\ref` (and related).

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

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

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

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

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

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

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

12.2.2 Vertical position

`\rightnoteupfalse` By default, sidenotes are placed to align with the last line of the note to which it refers.
`\leftnoteupfalse` If you want them 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` The texts are put a distance `\ledlsnotesep` (or `\ledrsnotesep`) into the left (or right)
`\ledrsnotesep` margin. These lengths are initially set to the value of `\linenumsep`.

12.2.4 Font

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

`\ledrsnotefontsetup`

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

These can of course be changed to suit.

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` \TeX provides the `\index{<item>}` command for specifying that $\langle item \rangle$ 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 $\langle item \rangle$ and the current page & line number should be added to the raw index file.

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

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

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

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

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

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 & line number 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 separator²⁰.

```
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-linenummer
```

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:

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

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

1. Use `xindy+hyperref` option when loading the `reledmac` package. When you run (Xe/Lua)TeX with this option, a `.xdy` configuration file will be generated with all the settings needed to allow internal hyperlinking in each index entry which is created by `\edindex`.
2. Use `hyperindex=false` option when loading `hyperref`.
3. Uncomment — by removing the semicolons at the beginning of the relevant lines — some lines in the `<code>.xdy</code>` file provided in the “examples” folder in order to restore internal links in the index to be used by the standard `index` command.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, referring 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. 63).

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>]
```

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

15 Tabular material

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

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

```

edarrayl      \begin{edtabularc}
edarrayc      1 & 2 & 3 \\
edarrayr      a & bb & ccc \\
edtabularl    AAA & BB & C
edtabularc    \end{edtabularc}
edtabularr

```

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` `\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}

```

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

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

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

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

```

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

```

$$\begin{array}{ccccccccc}
1 & & 2 & & 3 & & 4 & & & & 5 \\
Q & & & & & & fd & & h & & qwertziohg \\
v & & wptz & & x & & y & & & & vb \\
g & & nnn & & & & & & \upbracefill & & \\
k & & & & & & & & pq & & dgh \\
1 & & 2 & & 3 & & & & \hrulefill & &
\end{array}$$

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}

```

$$\begin{array}{cccc} 1 & 2 & 3 & 4 \\ a & \perp & & d \\ A & B & C & D \end{array}$$

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

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

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

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

For example:

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

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

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

`\edvertdots`

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

```

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

```

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

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

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

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

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

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

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

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

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

`\ledpb`
`\lednopb`

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

These commands have effect only at the second run.

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

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. 49 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. 69 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`

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

19.1.6 Paraphed footnotes

The restriction on explicit line-breaking in paraphed footnotes, mentioned on 7.1 p. 38, and described in more detail on XII.6.3 p. 179, 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 paraphed 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 paraphed 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. 49), 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 this²³ you will find \TeX whinging ‘Missing { inserted’, and then things start to fall apart. The trick in this case is to specify either:

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

or

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

(`\@secondoftwo` is an internal \TeX macro that takes two arguments and throws away the first one.) The first incantation lets color show in both the main text and footnotes whereas the second one shows color in the main text but kills it in the lemma and footnotes. On the other hand if you use `\textcolor` instead, like

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

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

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

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

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

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

²³Reported 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`.

```

1 %<code>
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{reledmac}[2017/08/17 v2.24.0 typesetting critical editions]
4 %

```

II.3 Package options

`\ifledfinal` Use this to remember which option is used, set and execute the options with final as the default. We use `xkeyval` in order to manage options with argument.

```

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

```

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}

```

```

38 \ExecuteOptionsX{final}
39
40 \newif\ifparapparatus@
41 \DeclareOptionX{parapparatus}{\parapparatus@true}
42
43 \newif\iflednopbinverse
44 \DeclareOptionX{nopbinverse}{\lednopbinversetrue}
45
46 \newif\ifwidthliketwocolumns%
47 \DeclareOptionX{widthliketwocolumns}{\widthliketwocolumnstrue}%
48
49 \newif\ifcontinuousnumberingwithcolumns
50 \DeclareOptionX{continuousnumberingwithcolumns}{\
continuousnumberingwithcolumnstrue}%
51
52 \newif\ifxindy@
53 \DeclareOptionX{xindy}[eledmac-markup-attr.xdy]{%
54   \AtBeginDocument{\immediate\openout\eledmac@xindy@out=#1}%
55   \newwrite\eledmac@xindy@out%
56   \xindy@true%
57   \gdef\eledmacmarkuplocdepth{:depth 1}%
58   \AtEndDocument{\immediate\closeout\eledmac@xindy@out}%
59 }%
60
61 \newif\ifxindyhyperref@
62 \DeclareOptionX{xindy+hyperref}{%
63   \xindyhyperref@true%
64 }%
65
66 \newif\ifeledmaccompat@%
67 \DeclareOptionX{eledmac-compat}{%
68   \eledmaccompat@true%
69 }%
70 \DeclareOptionX{nopenalties}{%
71   \AtBeginDocument{\let\add@penalties\relax}%
72 }
73 \def\l@auxdir{}%
74 \DeclareOptionX{auxdir}{%
75   \xdef\l@auxdir{#1}%
76 }%
77
78 \newif\ifsw@caseinsensitive%
79 \DeclareOptionX{swcaseinsensitive}{%
80   \sw@caseinsensitivetrue%
81 }%
82 %

```

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.

```
83 \ProcessOptionsX*\relax
84
85 %
```

II.4 Loading packages

Loading package `xargs` to declare commands with optional arguments. Loading package `xparse` to declare fully expandable commands with optional argument. Ideally, we should use only `xparse` and not `xargs`. For historical reasons, we use both. `Etoolbox` is also used to make code clearer - for example, in dynamic command names (which can replace `\csname` etc.). Use suffix to declare commands with a starred version, `xstring` to work with strings, `ifluatex` and `ifxetex` to test if `LuaTeX` or `XYTeX` is running, and `ragged2e` to manage ragged justification for paragraphed notes.

```
86 \RequirePackage{xargs}
87 \RequirePackage{xparse}%
88 \RequirePackage{etoolbox}
89 \@ifl@t@r\fmtversion{2015/10/01}
90 {\ifboolexpr{not test{\@ifl@t@r\fmtversion{2016/03/31}} or (test{\
  ifdefstring{\fmtversion}{2016/03/31}} and test {\ifnumless{\patch@level
  }{3}})}}}%
91 {\PackageWarning{reledmac}{You are using a LaTeX version older than
  2016/03/31 patch 3.}%
92 \MessageBreak You are strongly encouraged to use a newer version.}}%
93 {}%
94 }%
95 {\RequirePackage{etex}%
96 \csname reserveinserts\endcsname{32}%
97 }%
98 \RequirePackage{suffix}
99 \RequirePackage{xstring}
100 \RequirePackage{ifluatex}
101 \RequirePackage{ragged2e}
102 \RequirePackage{ifxetex}%
103 %
```

II.5 Compatibility with LuaTeX

Here, we enable some primitives for `LuaTeX`.

```
104 \ifx\directlua\undefined\else%
105 \directlua{tex.enableprimitives("",{"texdir","pdir","bodydir"})}
106 \fi
107 %
```

II.6 Boolean flags

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

```
108 \newif\ifl@dmemoir
109 \@ifclassloaded{memoir}{\l@dmemoirtrue}{\l@dmemoirfalse}
110
111 %
```

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

```
112 \newif\if@ledgroup%
113 %
```

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

```
114 \newif\ifl@imakeidx
115 \@ifpackageloaded{imakeidx}{\l@imakeidxtrue}{\l@imakeidxfalse}%False is the default value
116 %
```

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

```
117 \newif\ifl@indextools%
118 \@ifpackageloaded{indextools}{%
119   \l@indextoolstrue%
120   \l@imakeidxtrue%
121   \let\imki@wrindexentry\indtl@wrindexentry%
122 }{}%
123 %
```

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.

```
124 \newif\ifl@footmisc
125 \@ifpackageloaded{footmisc}{\l@footmisctrue}{\l@footmiscfalse}%False is the default value
126 %
```

`\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.

```
127 \ifdefined\if@RTL{\newif\if@RTL}
128 %
```

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

```
129 \newif\if@firstlineofpage%
130 \newif\if@firstlineofpageR%
131 %
```

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.

```
132 \newcommand{\reledmac@warning}[1]{\PackageWarning{reledmac}{#1}}
133 %
```

`\reledmac@error` Write an error message.

```
134 \newcommand{\reledmac@error}[2]{\PackageError{reledmac}{#1}{#2}}
135 %
```

```
\led@err@NumberingStarted36 \newcommand*{\led@err@NumberingStarted}{%
d@err@NumberingNotStarted37 \reledmac@error{Numbering has already been started}{\@ehc}}
NumberingShouldHaveStarted38 \newcommand*{\led@err@NumberingNotStarted}{%
139 \reledmac@error{Numbering was not started}{\@ehc}}
140 \newcommand*{\led@err@NumberingShouldHaveStarted}{%
141 \reledmac@error{Numbering should already have been started}{\@ehc}}
142 %
```

```
d@err@edtextoutsidestart43 \newcommand*{\led@err@edtextoutsidestart}{%
144 \reledmac@error{\string\edtext\space outside numbered paragraph (\...pstart
\pend)}{\@ehc}}%
145 %
```

```
\led@mess@NotesChanged46 \newcommand*{\led@mess@NotesChanged}{%
147 \typeout{reledmac reminder: }%
148 \typeout{ The number of the footnotes in this section
149 has changed since the last run.}%
150 \typeout{ You will need to run LaTeX two more times
151 before the footnote placement}%
152 \typeout{ and line numbering in this section are
153 correct.}}
154 %
```

```
\led@mess@SectionContinued55 \newcommand*{\led@mess@SectionContinued}[1]{%
156 \message{Section #1 (continuing the previous section)}}
157 %
```

```
d@err@LineationInNumbered58 \newcommand*{\led@err@LineationInNumbered}{%
159 \reledmac@error{You can't use \string\lineation\space within
160 a numbered section}{\@ehc}}
161 %
```

```

\led@warn@BadLineation62 \newcommand*\led@warn@BadLineation}{%
\led@warn@BadLinenummargin63 \reledmac@warning{Bad \string\lineation\space argument}}
\led@warn@BadLockdisp64 \newcommand*\led@warn@BadLinenummargin}{%
\led@warn@BadSublockdisp65 \reledmac@warning{Bad \string\linenummargin\space argument}}
166 \newcommand*\led@warn@BadLockdisp}{%
167 \reledmac@warning{Bad \string\lockdisp\space argument}}
168 \newcommand*\led@warn@BadSublockdisp}{%
169 \reledmac@warning{Bad \string\sublockdisp\space argument}}
170 %

\led@warn@NoFile71 \newcommand*\led@warn@NoFile}[1]{%
172 \reledmac@warning{File `#1' not found}}
173 %

\led@warn@LineFileObsolete74 \newcommand*\led@warn@Obsolete}[1]{%
175 \reledmac@warning{Line-list file #1 was obsolete. We have not read it.
Please run LaTeX again.}}
176 %

\led@warn@BadAdvancelineSubline77 \newcommand*\led@warn@BadAdvancelineSubline}{%
\led@warn@BadAdvancelineLine78 \reledmac@warning{\string\advanceline\space produced a sub-line
179 number less than zero.}}
180 \newcommand*\led@warn@BadAdvancelineLine}{%
181 \reledmac@warning{\string\advanceline\space produced a line
182 number less than zero.}}
183 %

\led@warn@BadSetline84 \newcommand*\led@warn@BadSetline}{%
\led@warn@BadSetlinenum85 \reledmac@warning{Bad \string\setline\space argument}}
186 \newcommand*\led@warn@BadSetlinenum}{%
187 \reledmac@warning{Bad \string\setlinenum\space argument}}
188 %

\led@err@PstartNotNumbered89 \newcommand*\led@err@PstartNotNumbered}{%
\led@err@PstartInPstart90 \reledmac@error{\string\pstart\space must be used within a
\led@err@PendNotNumbered91 numbered section %
\led@err@PendNoPstart92 (\string\...beginnumbering\string\endnumbering)}{\@ehc}}%
\led@err@AutoparNotNumbered93 \newcommand*\led@err@PstartInPstart}{%
\led@err@NumberingWithoutPstart94 \reledmac@error{\string\pstart\space encountered while another
195 \string\pstart\space was in effect}{\@ehc}}
196 \newcommand*\led@err@PendNotNumbered}{%
197 \reledmac@error{\string\pend\space must be used within a
198 numbered section}{\@ehc}}
199 \newcommand*\led@err@PendNoPstart}{%

```



```

200 \reledmac@error{\string\pend\space must follow a \string\pstart}{\@ehc}}
201 \newcommand*{\led@err@AutoparNotNumbered}{%
202 \reledmac@error{\string\autopar\space must be used within a
203 numbered section}{\@ehc}}
204 \newcommand*{\led@err@NumberingWithoutPstart}{%
205 \reledmac@error{\string\beginnumbering...\string\endnumbering\space
without \string\pstart}{\@ehc}}%
206 %

```

```

\led@warn@BadAction07 \newcommand*{\led@warn@BadAction}{%
208 \reledmac@warning{Bad action code, value \next@action.}}
209 %

```

```

\led@warn@DuplicateLabel10 \newcommand*{\led@warn@DuplicateLabel}[1]{%
ppLabelOutSecondArgEdtext11 \reledmac@warning{Duplicate definition of label `#1'\@gobble}%
\led@warn@RefUndefined12 \@latex@warning@no@line{Label `#1' multiply defined}%
\led@warn@RefUndefined13 }%
214 \newcommand*{\led@warn@AppLabelOutSecondArgEdtext}[1]{%
215 \reledmac@warning{\string\applabel\space outside of the second argument
of an \string\edtext\space `#1' on page \thepage.}}%
216 \newcommand*{\led@warn@RefUndefined}[1]{%
217 \G@refundefinedtrue%
218 \reledmac@warning{Reference `#1' on page \thepage\space undefined.%
219 Using `000'.}%
220 \@latex@warning{Reference `#1' undefined\on@line}%
221 }%
222 \newcommand*{\led@warn@pairRefUndefined}[1]{%
223 \G@refundefinedtrue%
224 \reledmac@warning{Reference `#1:start' and/or `#1:end' on page \thepage\
space undefined.
225 Using `??'.}%
226 \@latex@warning{Reference `#1:start' and/or `#1:end' undefined\on@line}%
227 }
228 %

```

```

\led@warn@NoMarginpars29 \newcommand*{\led@warn@NoMarginpars}{%
230 \reledmac@warning{You can't use \string\marginpar\space in numbered text
}}
231 %

```

```

\led@warn@BadSidenotemargin32 \newcommand*{\led@warn@BadSidenotemargin}{%
233 \reledmac@warning{Bad \string\sidenotemmargin\space argument}}
234 %

```

```

\led@warn@NoIndexFile 35 \newcommand*{\led@warn@NoIndexFile}[1]{%
236 \reledmac@warning{Undefined index file #1}}
237 %

```

```

\led@warn@SeriesStillExist 38 \newcommand{\led@warn@SeriesStillExist}[1]{%
239 \reledmac@warning{Series #1 is still existing !}%
240 }%
241 %

```

```

\led@err@BadAction 42 \newcommand*{\led@err@StanzaIndentNotDefined}{%
243 \reledmac@error{You have not defined the indentation for the line \number
\stanza@count}{\@ehc}}%
244 %

```

```

\led@err@ManySidenotes 45 \newcommand{\led@err@ManySidenotes}{%
\led@err@ManyLeftnotes 46 \ifledRcol{%
\led@err@ManyRightnotes 47 \reledmac@warning{\itemcount@space sidenotes on line \the\line@numR\
space p. \the\page@numR}%
248 \else%
249 \reledmac@warning{\itemcount@space sidenotes on line \the\line@num\
space p. \the\page@num}%
250 \fi%
251 }%
252 \newcommand{\led@err@ManyLeftnotes}{%
253 \ifledRcol{%
254 \reledmac@warning{\itemcount@space leftnotes on line \the\line@numR\
space p. \the\page@numR}%
255 \else%
256 \reledmac@warning{\itemcount@space leftnotes on line \the\line@num\
space p. \the\page@num}%
257 \fi%
258 }%
259 \newcommand{\led@err@ManyRightnotes}{%
260 \ifledRcol{%
261 \reledmac@warning{\itemcount@space rightnotes on line \the\line@numR\
space p. \the\page@numR}%
262 \else%
263 \reledmac@warning{\itemcount@space rightnotes on line \the\line@num\
space p. \the\page@num}%
264 \fi%
265 }%
266 %

```

```

\led@err@TooManyColumns 67 \newcommand*{\led@err@TooManyColumns}{%
\led@err@UnequalColumns 68 \reledmac@error{Too many columns}{\@ehc}}
\led@err@LowStartColumn
\led@err@HighEndColumn
\led@err@ReverseColumns

```

```

269 \newcommand*\led@err@UnequalColumns}{%
270   \reledmac@error{Number of columns is not equal to the number
271     in the previous row (or \protect\\ \space forgotten?)}{\@ehc}}
272 \newcommand*\led@err@LowStartColumn}{%
273   \reledmac@error{Start column is too low}{\@ehc}}
274 \newcommand*\led@err@HighEndColumn}{%
275   \reledmac@error{End column is too high}{\@ehc}}
276 \newcommand*\led@err@ReverseColumns}{%
277   \reledmac@error{Start column is greater than end column}{\@ehc}}
278 %

```

```

endnotes@outsidenumbering79 \newcommand\led@err@toendnotes@outsidenumbering}{%
280   \reledmac@error{\string\toendnotes\space and related commands must be
    called inside a numbered text (\string\...beginnumbering\string\endnumbering
    )}{\@ehc}%
281 }%
282 %

```

```

err@EdtextWithoutFootnote83 \newcommand\led@err@EdtextWithoutFootnote}{%
284   \reledmac@error{edtext without Xfootnote. Check syntax}{\@ehc}%
285 }%
286 %

```

```

tNoteNotInSecondArgEdtext87 \newcommand\led@err@FootnoteNotInSecondArgEdtext}[1]{%
288   \reledmac@error{#1 footnote outside of the second argument of an edtext.
    Check syntax}{\@ehc}%
289 }%
290 %

```

```

error@PackageAfterEledmac91 \newcommand\led@error@PackageAfterEledmac}[1]{%
292   \reledmac@error{#1 must be loaded before reledmac}{\@ehc}%
293 }%
294 %

```

```

error@fail@patch@@makecol195 \newcommand\led@error@fail@patch@@makecol}{%
296   \reledmac@error{Fail to patch \string\@makecol\space command}{\@ehc}%
297 }%
298 %

```

```

ror@fail@patch@@reinserts99 \newcommand\led@error@fail@patch@@reinserts}{%
300   \reledmac@error{Fail to patch \string\@reinserts\space command}{\@ehc}%
301 }%
302 %

```

```

\led@error@fail@patch@@doclearpage03 \newcommand{\led@error@fail@patch@@doclearpage}{%
304 \reledmac@error{Fail to patch \string\@doclearpage\space command}{\@ehc}%
305 }%
306 %

```

```

\led@error@fail@patch@@iiiminipage07 \newcommand{\led@error@fail@patch@@iiiminipage}{%
308 \reledmac@error{Fail to patch \string\@iiiminipage\space command}{\@ehc}%
309 }%
310 %

```

```

\led@error@fail@patch@endminipage11 \newcommand{\led@error@fail@patch@endminipage}{%
312 \reledmac@error{Failed to patch the \string\endminipage\space command}{\@ehc}%
313 }%
314 %

```

```

\led@error@fail@patch@endminipage15 \newcommand{\led@error@fail@patch@makeindex}{%
316 \reledmac@error{Failed to patch the \string\makeindex\space command}{\@ehc}%
317 }%
318 %

```

```

\led@warn@edinde@outsidenumbering19 \newcommand{\led@warn@edinde@outsidenumbering}{%
320 \reledmac@warning{\string\edindex\space called outside of \string\
...beginnumbering\string\endnumbering. \MessageBreak Automatically switched
to \string\index.}%
321 }%
322 %

```

```

\led@warning@hsizeX@deprecated23 \newcommand{\led@warning@hsizeX@deprecated}{%
324 \reledmac@warning{\string\hsizeX\space command deprecated, use \string\
widthX\space instead.}%
325 }%
326 %

```

```

\led@warning@Xhsize@deprecated27 \newcommand{\led@warning@Xhsize@deprecated}{%
328 \reledmac@warning{\string\Xhsize\space command deprecated, use \string\
Xwidth\space instead.}%
329 }%
330 %

```

```

\warning@msdatawithoutstop 31 \newcommand{\led@warning@msdatawithoutstop}{%
332 \reledmac@warning{\string\msdata\space without corresponding \string\
stopmsdata}%
333 }%
334 %

```

```

\warning@preXnotes@deprecated 35 \newcommand{\led@warning@preXnotes@deprecated}{%
336 \reledmac@warning@preXnotes@deprecated%
337 }%
338 %

```

II.8 Gobbling

Here, we define some commands which gobble their arguments.

```

\@gobblethree 39 \providecommand*\@gobblethree}[3]{}
\@gobblefour 40 \providecommand*\@gobblefour}[4]{}
\@gobbleseven 41 \providecommand*\@gobbleseven}[7]{}
342 %

```

II.9 Miscellaneous commands

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

```

343 \ifledfinal
344 \newcommand*\showlemma[1]{#1}
345 \else
346 \newcommand*\showlemma[1]{\underline{#1}}
347 \fi
348
349 %

```

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

Initialize it as `\empty`.

```

350 \let\linenumberlist=\empty
351
352 %

```

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

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

```

353 \newcount\@l@tempcnta \newcount\@l@tempcntb
354 %

```

II.10 Prepare reledpar

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

`\ifl@dpaging`

`\ifl@dprintingpages`

`\ifl@dprintingcolumns`

```

355 \newif\ifl@dpairing
356 \newif\ifl@dpaging%
357 \newif\ifl@dprintingpages%
358 \newif\ifl@dprintingcolumns%
359 \newif\ifpst@rtedL
360 \newcount\l@dnumpstartsL
361 %

```

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

`\ifledRcol@`

```

362 \newif\ifledRcol
363 \newif\ifledRcol@
364 %

```

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

```

365 \newif\ifnumberingR
366 %

```

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

```

367 \newif\ifXnote@%
368 %

```

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

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

`\ifindtl@notenumber`

```

369 \providebool{indtl@innote}%
370 \providebool{indtl@notenumber}%
371 %

```

III Sectioning commands

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

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

```

372 \newcount\section@num
373 \section@num=0
374 \let\extensionchars=\empty
375 %

```

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

```

376 \newif\ifnumbering
377 %

```

`\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.

```

378 \newcommand*{\beginnumbering}{%
379   \ifnumbering
380     \led@err@NumberingStarted
381   \endnumbering
382 \fi
383 \global\numberingtrue
384 \global\advance\section@num \@ne
385 \initnumbering@reg
386 \message{Section \the\section@num }%
387 \line@list@stuff{\jobname.\extensionchars\the\section@num}%

```

```

388 \l@dend@stuff
389 \setcounter{pstart}{1}
390 \ifl@dpairing
391   \global\l@dnumpstartsL \z@
392   \global\pst@rtedLfalse
393 %

```

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.

```

394 \else
395   \begingroup
396   \global\@afterindenttrue%In order to reestablish normal feature if the \
  begingroup was not here
397   \initnumbering@quote
398   \ifwidthliketwocolumns%
399     \setwidthliketwocolumns%
400     \csuse{setpositionliketwocolumns@\columns@position}%
401   \fi%
402 \fi
403 \gdef\eled@sections@{ }%
404 \if@noeled@sec\else%
405   \makeatletter%
406   \InputIfFileExists%
407     {\l@auxdir\jobname.eledsec\the\section@num}%
408     {}%
409     {\led@warn@NoFile{\l@auxdir\jobname.eledsec\the\section@num}}%
410   \makeatother%
411   \immediate\openout\eled@sectioning@out=\l@auxdir\jobname.eledsec\the\
  section@num\relax%
412 \fi%
413 }
414 \newcommand*{\initnumbering@reg}{%
415   \global\pst@rtedLfalse
416   \global\l@dnumpstartsL \z@
417   \global\absline@num \z@
418   \gdef\normal@page@break{}
419   \gdef\l@prev@pb{}
420   \gdef\l@prev@nopb{}
421   \global\line@num \z@
422   \global\subline@num \z@
423   \global\@lock \z@
424   \global\sub@lock \z@
425   \global\sublines@false
426   \global\let\next@page@num=\relax
427   \global\let\sub@change=\relax

```



```

428 \global\last@page@num=-10000%
429 \ifdefined\line@numR%
430   \line@numR=\z@%
431   \last@page@numR=\z@%
432 \fi%
433 \resetprevline@
434 \resetprevpage@num
435 \global\stopmsdata@inserted@true%
436 \global\let\@msdata@list\relax%
437 \global\csundef{\@msdata@\add@msd@c @data}%
438 }
439
440 %

```

\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. We define it using \def and not \newcommand because \TeX does not allow defining command whose name starts by “end” except if we are currently creating an environment, which is not the case here.

```

441 \def\endnumbering{%
442   \ifnumbering
443     \global\numberingfalse
444     \normal@pars
445     \ifnum\l@dnumstart=L=0%
446       \led@err@NumberingWithoutPstart%
447     \fi%
448     \ifl@dpairing
449       \global\pstart=L=false
450     \else
451       \ifx\insertlines@list\empty\else
452         \global\noteschanged@true
453       \fi
454       \ifx\line@list\empty\else
455         \global\noteschanged@true
456       \fi
457     \fi
458     \ifnoteschanged@
459       \led@mess@NotesChanged
460     \fi
461   \else
462     \led@err@NumberingNotStarted
463   \fi
464   \autoparfalse
465   \if@noeled@sec\else%
466     \immediate\closeout\eled@sectioning@out%
467   \fi%
468   \ifl@dpairing\else
469     \global\l@dnumstart=L=\z@%
470   \endgroup

```

```

471 \fi
472 }
473 %

```

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

`\resumenumbering` The `\pausenumbering@page@num` counter stores the `\next@page@num` when the `\pausenumbering@page@num` `\pause@numbering` is called.

```

474 \newcount\pausenumbering@page@num%
475 \newcommand{\pausenumbering}{%
476   \ifx\next@page@num\relax%
477     \global\pausenumbering@page@num=0%
478   \else%
479     \global\pausenumbering@page@num=\next@page@num%
480   \fi%
481   \ifautopar\global\autopar@pausetrue\fi%
482   \endnumbering\global\numberingtrue}
483 %

```

The `\resumenumbering` macro is a bit more involved, but not much. It does most of the same things as `\beginnumbering`, but without resetting the various counters. Note that no check is made by `\resumenumbering` to ensure that `\pausenumbering` was actually invoked. The boolean `\ifresumenumbering@start` is set to true. That allows us to avoid resetting the line number at the first line of `\resumenumbering` if the lineation is by page. This boolean is set to false after the first action.

```

484 \newif\ifresumenumbering@start%
485 \newcommand*{\resumenumbering}{%
486   \ifnumbering
487     \ifautopar@pause\autopar\fi
488     \global\pst@rtedLtrue
489     \global\advance\section@num \@ne
490     \global\resumenumbering@starttrue%
491     \led@mess@SectionContinued{the\section@num}%
492     \set@continuousnumberingforL%
493     \line@list@stuff{\jobname.\extensionchars\the\section@num}%
494     \l@dend@stuff
495     \ifl@dpairing\else%
496       \begingroup%
497       \initnumbering@quote%
498       \ifwidthliketwocolumns%
499         \setwidthliketwocolumns%
500         \csuse{setpositionliketwocolumns@\columns@position}%
501       \fi%
502     \fi%
503   \else
504     \led@err@NumberingShouldHaveStarted

```

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

```

505 \endnumbering
506 \beginnumbering
507 \fi}
508
509
510 %

```

`\set@continuousnumberingforL` \set@continuousnumberingforL sets the left line numbers and pstart counters at a \beginnumbering or a \resumenumbering in order to have continuous numbering with single column text.

```

511 \newcommand{\set@continuousnumberingforL}{%
512 \ifcontinuousnumberingwithcolumns%
513 \ifdefined\line@numR%
514 \ifnum\line@numR>\line@num%
515 \global\appto\next@line@list@stuff{%
516 \expandafter\setlinenum\expandafter{\the\line@numR}%
517 }%
518 \fi%
519 \ifnum\last@page@numR>\last@page@num%
520 \global\last@page@num=\last@page@numR%
521 \fi%
522 \fi%
523 \ifl@dpairing%
524 \unless\ifl@dpaging%
525 \global\c@pstartL=\c@pstart%
526 \fi%
527 \fi%
528 \fi%
529 }%
530 %

```

IV List macros

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

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

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

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

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

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

```

531 \newcommand*\list@create}[1]{%
532   \global\let#1=\empty%
533 }%
534 %

```

\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 .

```

535 \newcommand*\list@clear}[1]{%
536   \global\let#1=\empty%
537 }
538 %

```

\xright@appenditem `\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`.

```

539 \newtoks\led@toksa \newtoks\led@toksb
540 \global\led@toksa={\}
541 \long\def\xright@appenditem#1\to#2{%
542   \global\led@toksb=\expandafter{#2}%
543   \xdef#2{\the\led@toksb\the\led@toksa\expandafter{#1}}%
544   \global\led@toksb={}}
545 %

```

\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`.

```

546 \long\def\xleft@appenditem#1\to#2{%
547   \global\led@toksb=\expandafter{#2}%
548   \xdef#2{\the\led@toksa\expandafter{#1}\the\led@toksb}%
549   \global\led@toksb={}}
550 %

```

\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.

```

551 \def\gl@p#1\to#2{\expandafter\gl@poff#1\gl@poff#1#2}
552 \long\def\gl@poff\#1#2\gl@poff#3#4{\gdef#4{#1}\gdef#3{#2}}
553
554 %

```

V Line counting

V.1 Choosing the system of lineation

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

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

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

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

`\ifbypage@`

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

`\bypage@false`

```
555 \newif\ifbypage@
556 \newif\ifbypstart@
557 %
```

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 558 \newif\ifbypage@R
\ifbypstart@R 559 \newif\ifbypstart@R
560 %
```

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

```
561 \newcommand*{\lineation}[1]{%
562 %
```

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

```
563 \ifnumbering
564 \led@err@LineationInNumbered
565 \else
566 %
```

If the argument is page.

```
567 \def\@tempa{#1}\def\@tempb{page}%
568 \ifx\@tempa\@tempb
569 \global\bypage@true
570 \global\bypstart@false
571 \unless\ifnocritical@%
572 \Xpstart[] [false]%
573 \fi%
574 %
```

If the argument is pstart.

```

575 \else
576   \def\@tempb{pstart}%
577   \ifx\@tempa\@tempb
578     \global\bypage@false
579     \global\bystart@true
580     \unless\ifnocritical@%
581       \Xpstart%
582     \fi%
583 %

```

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

```

584 \else
585   \def\@tempb{section}
586   \ifx\@tempa\@tempb
587     \global\bypage@false
588     \global\bystart@false
589     \unless\ifnocritical@%
590       \Xpstart[] [false]%
591     \fi%
592 %

```

In other case, it is an error.

```

593 \else
594   \led@warn@BadLineation
595 \fi
596 \fi
597 \fi
598 \fi}}
599 %

```

V.2 Line number margin

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

```

600 \newcount\line@margin%
601 \newcount\line@margin@columns%Only for parallel typesetting
602 \line@margin@columns=\m@ne%
603
604 \newcommand*{\linenummargin}[1]{%
605   \l@getline@margin{#1}%
606   \ifnum\@l@dttempcntb>\m@ne
607     \ifledRcol
608       \global\line@marginR=\@l@dttempcntb
609       \led@warn@setting@in@rightside{\linenummargin}%
610     \else
611       \global\line@margin=\@l@dttempcntb

```

```

612 \fi
613 \fi}}
614
615 \newcommand*{\l@dgetline@margin}[1]{%
616 \def\@tempa{#1}\def\@tempb{left}%
617 \ifx\@tempa\@tempb
618 \l@dtempcntb \z@
619 \else
620 \def\@tempb{right}%
621 \ifx\@tempa\@tempb
622 \l@dtempcntb \@ne
623 \else
624 \def\@tempb{outer}%
625 \ifx\@tempa\@tempb
626 \l@dtempcntb \tw@
627 \else
628 \def\@tempb{inner}%
629 \ifx\@tempa\@tempb
630 \l@dtempcntb \thr@@
631 \else
632 \led@warn@BadLinenummargin
633 \l@dtempcntb \m@ne
634 \fi
635 \fi
636 \fi
637 \fi}
638
639 %

```

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.

```

640 \newcounter{firstlinenum}
641 \setcounter{firstlinenum}{5}
642 \newcounter{linenumincrement}
643 \setcounter{linenumincrement}{5}
644 %

```

`\c@firstsublinenum`
`\c@sublinenumincrement`

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

```

645 \newcounter{firstsublinenum}
646 \setcounter{firstsublinenum}{5}
647 \newcounter{sublinenumincrement}

```

```

648 \setcounter{sublinenumincrement}{5}
649
650 %

```

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

```

\linenumincrement
\firstsublinenum
\sublinenumincrement
651 \newcommand*\firstlinenum[1]{%
652 \ifledRcol%
653 \setcounter{firstlinenumR}{#1}%
654 \led@warn@setting@in@rightside{\firstlinenum}%
655 \else%
656 \setcounter{firstlinenum}{#1}%
657 \fi%
658 }
659 \newcommand*\linenumincrement[1]{%
660 \ifledRcol%
661 \setcounter{linenumincrementR}{#1}%
662 \led@warn@setting@in@rightside{\linenumincrement}%
663 \else%
664 \setcounter{linenumincrement}{#1}%
665 \fi%
666 }
667 \newcommand*\firstsublinenum[1]{%
668 \ifledRcol%
669 \setcounter{firstsublinenumR}{#1}%
670 \led@warn@setting@in@rightside{\firstsublinenum}%
671 \else%
672 \setcounter{firstsublinenum}{#1}%
673 \fi%
674 }
675 \newcommand*\sublinenumincrement[1]{%
676 \ifledRcol%
677 \setcounter{sublinenumincrementR}{#1}%
678 \led@warn@setting@in@rightside{\sublinenumincrement}%
679 \else%
680 \setcounter{sublinenumincrement}{#1}%
681 \fi%
682 }
683 %
684
685 %

```

V.4 Line number locking

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

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


```

686 \newcount\lock@disp
687 \newcommand{\lockdisp}[1]{%
688   \l@getlock@disp{#1}%
689   \ifnum\@l@tempcntb>\m@ne
690     \global\lock@disp=\@l@tempcntb
691   \else
692     \led@warn@BadLockdisp
693   \fi}}
694 \newcommand*{\l@getlock@disp}[1]{
695   \def\@tempa{#1}\def\@tempb{first}%
696   \ifx\@tempa\@tempb
697     \@l@tempcntb \z@
698   \else
699     \def\@tempb{last}%
700     \ifx\@tempa\@tempb
701       \@l@tempcntb \@ne
702     \else
703       \def\@tempb{all}%
704       \ifx\@tempa\@tempb
705         \@l@tempcntb \tw@
706       \else
707         \@l@tempcntb \m@ne
708       \fi
709     \fi
710   \fi}
711
712 %

```

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

```

713 \newcount\sublock@disp
714 \newcommand{\sublockdisp}[1]{%
715   \l@getlock@disp{#1}%
716   \ifnum\@l@tempcntb>\m@ne
717     \global\sublock@disp=\@l@tempcntb
718   \else
719     \led@warn@BadSublockdisp
720   \fi}}
721
722 %

```

V.5 Line number style

`\linenumberstyle` We provide a mechanism for using different representations of the line numbers, not
`\linenumrep` just the normal arabic.
`\linenumr@p` NOTE: In v0.7 `\linenumrep` and `\sublinenumrep` replaced the internal `\linenumr@p`
`\sublinenumberstyle` and `\sublinenumr@p`.
`\sublinenumrep`
`\sublinenumr@p`

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

```

723 \newcommand*{\linenumberstyle}[1]{%
724   \def\linenumrep##1{\@nameuse{#1}{##1}}
725 \newcommand*{\sublinenumberstyle}[1]{%
726   \def\sublinenumrep##1{\@nameuse{#1}{##1}}
727   %

```

Initialise the number styles to arabic.

```

728 \linenumberstyle{arabic}
729 \let\linenumr@p\linenumrep
730 \sublinenumberstyle{arabic}
731 \let\sublinenumr@p\sublinenumrep
732
733 %

```

V.6 Line number printing

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

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

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

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

```

734 \newlength{\linenumsep}
735 \setlength{\linenumsep}{1pc}
736 \newcommand*{\numlabfont}{\normalfont\scriptsize}
737 \newcommand*{\ledlinenum}{%
738   \bgroup%
739   \ifluatex%
740     \texdir TLT%
741   \fi%
742   \numlabfont\linenumrep{\line@num}%
743   \ifsublines@
744     \ifnum\subline@num>0\relax
745       \unskip%
746       \Xsublinesep@side%
747       \sublinenumrep{\subline@num}%

```

```

748 \fi
749 \fi%
750 \egroup%
751 }%
752
753 \newcommand*{\leftlinenum}{%
754 \ledlinenum
755 \kern\linenumsep}
756 \newcommand*{\rightlinenum}{%
757 \kern\linenumsep
758 \ledlinenum}
759
760 %

```

V.7 Line number counters and lists

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

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

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

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

```

761 \newcount\line@num
762 %

```

`\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.

```

763 \newcount\subline@num
764 %

```

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

`\sublines@true` You may wonder why we do not just use the value of `\subline@num` to determine this—treating anything greater than 0 as an indication that sub-lineation is on. We need a separate flag because sub-lineation can be used together with line-number locking in

`\sublines@false`

odd ways: several pieces of a logical line might be interrupted by pieces of sub-lined 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.

```
765 \newif\ifsublines@
766 %
```

\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.

```
767 \newcount\absline@num
768 %
```

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

V.8 Line number locking counter

\@lock The counts `\@lock` and `\sub@lock` tell us the state of line-number and sub-line-number locking. 0 means we are not within a locked set of lines; 1 means we are at the first line in the set; 2, at some intermediate line; and 3, at the last line.

```
769 \newcount\@lock
770 \newcount\sub@lock
771 %
```

V.9 Line number associated to lemma

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

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

- `\line@list`: the page and line numbers for every lemma marked by `\edtext`. There are seven pieces of information, separated by vertical bars:

1. the starting page,
2. line, and
3. sub-line numbers, followed by the
4. ending page,
5. line, and
6. sub-line numbers, and then the
7. font specifier for the lemma.

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

```
23|35|0|24|3|0|0T1/cmr/m/n.
```

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

- `\insertlines@list`: the line numbers of lines that have footnotes or other insertions. These are the absolute numbers where the corresponding lemmas begin. This list contains one entry for every footnote in the section; one lemma may contribute no footnotes or many footnotes. This list is used by `\add@inserts` within `\do@line`, to tell it where to insert notes.
- `\actionlines@list`: a list of absolute line numbers at which we are to perform special actions; these actions are specified by the `\actions@list` list defined below.
- `\actions@list`: action codes corresponding to the line numbers in `\actionlines@list`. These codes tell `reledmac` what action it is supposed to take at each of these lines. One action, the page-start action, is generated behind the scenes by `reledmac` itself; the others, for specifying sub-lineation, line-number locking, and line-number alteration, are generated only by explicit commands in your input file. The page-start and line-number-alteration actions require arguments, to specify the new values for the page or line numbers; instead of storing those arguments in another list, we have chosen the action-code values so that they can encode both the action and the argument in these cases. Action codes greater than -1000 are page-start actions, and the code value is the page number; action codes less than -5000 specify line numbers, and the code value is a transformed version of the line number; action codes between these two values specify other actions which require no argument.

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

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

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

The action code -1002 specifies the end of sub-lineation. At the next start-of-line, we should clear the sub-line counter and start advancing the line number.

The action codes for starting and ending sub-lineation are added to the list by the `\sub@action` macro, as called to implement the `\startsub` and `\endsub` macros.

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

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

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

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

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

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

```

772 \list@create{\line@list}
773 \list@create{\insertlines@list}
774 \list@create{\actionlines@list}
775 \list@create{\actions@list}
776
777 %

```

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

```

778 \newcount\page@num
779 \newcount\endpage@num
780 \newcount\endline@num
781 \newcount\endsubline@num
782 %

```

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

the line-list used to place the notes will be wrong, and some notes will probably be misplaced. When this happens, we prefer to give a single error message for the whole section rather than messages at every point where we notice the problem, because we do not really know where in the section notes were added or removed, and the solution in any case is simply to run \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.

```
783 \newif\ifnoteschanged@
784 %
```

`\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@85 \newcommand*{\resetprevline@}{%
786   \def\do##1{\global\csundef{prevline##1}}%
787   \dolistloop{\@series}%
788 }
789 %
```

`\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@90 \newcommand*{\resetprevpage@num}{%
791   \def\do##1{%
792     \ifcsdef{prevpage##1@num}{%
793       \global\csname prevpage##1@num\endcsname=\z@%
794       \global\csname prevpage##1@numR\endcsname=\z@%
795     }%
796     {}%
797     \ifcsdef{##1prevpage@num}{%
798       \global\csname ##1prevpage@num\endcsname=\z@%
799       \global\csname ##1prevpage@numR\endcsname=\z@%
800     }%
801     {}%
802   }%
803   \dolistloop{\@series}%
804 }
805 %
```

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.

```

806 \newread\@inputcheck
807 \newcommand*{\read@linelist}[1]{%
808   \ifledRcol%
809     \list@clearing@regR%
810   \else%
811     \list@clearing@reg%
812   \fi%
813 %

```

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).

```

814   \list@clear{\maxlinesinpar@list}
815 %

```

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

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

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

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

```

816 \get@linelistfile{#1}%
817 \@stopmsd%Security if last \endms{} is forgotten
818 \unless\ifledRcol%Get the last line of the last page
819   \csnumgdef{\@lastabsline@forpage@the\page@num}{\the\absline@num}%
820   \csnumgdef{\@lastline@forpage@the\page@num}{\the\line@num}%
821 \else%
822   \csnumgdef{\@lastabsline@forpageR@the\page@numR}{\the\absline@numR}%
823   \csnumgdef{\@lastline@forpageR@the\page@numR}{\the\line@numR}%
824 \fi%
825 \endgroup
826 %

```

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.

```

827 \ifledRcol
828   \global\page@numR=\m@ne
829   \ifx\actionlines@listR\empty
830     \gdef\next@actionlineR{1000000}%
831   \else
832     \gl@p\actionlines@listR\to\next@actionlineR
833     \gl@p\actions@listR\to\next@actionR
834   \fi
835 \else
836   \global\page@num=\m@ne
837   \ifx\actionlines@list\empty
838     \gdef\next@actionline{1000000}%
839   \else
840     \gl@p\actionlines@list\to\next@actionline
841     \gl@p\actions@list\to\next@action
842   \fi
843 \fi
844 }
845 %

```

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

```

846 \newcommand*{\list@clearing@reg}{%
847   \list@clear{\line@list}%
848   \list@clear{\insertlines@list}%
849   \list@clear{\actionlines@list}%
850   \list@clear{\actions@list}%
851   \list@clear{\linesinpar@listL}%
852   \list@clear{\linesonpage@listL}%
853 }%
854 %

```

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

```

855 \newcommand*{\get@linelistfile}[1]{%
856   \InputIfFileExists{\l@auxdir#1}{%
857     \global\noteschanged@false
858     \begingroup
859       \catcode`\[=1 \catcode`\]=2
860       \makeatletter \catcode`\^^M=9}{%
861     \led@warn@NoFile{\l@auxdir#1}%
862     \global\noteschanged@true
863     \begingroup}%
864 }
865 %
866 %

```

This version of `\read@linelist` creates list macros containing data for the entire section, so they could get rather large. It would be no more difficult to read the line-list file incrementally rather than all at once: we could read, at the start of each paragraph, only the commands relating to that paragraph. But this would require that we have two line-lists open at once, one for reading, one for writing, and on systems without version numbers we would have to do some file renaming outside of \TeX for that to work. We have retained this slower approach to avoid that sort of hacking about, but have provided the `\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.

```

867 \newcommand{\line@list@version}[1]{%
868   \IfStrEq{#1}{\this@line@list@version}%
869   }%
870   {\ifledRcol%
871     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
872     \else%
873     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
874     \fi%
875     \endinput%
876   }%
877 }%
878 %

```

`\@nl` `\@nl` does everything related to the start of a new line of numbered text.

`\@nl@reg` In order to get the `\setlinenum` to work Peter Wilson had to slip in some new code at the start of the macro, to get the timing of the actions correct. The problem was that his original naive implementation of `\setlinenum` had a unfortunate tendency to change the number of the last line of the *preceding* paragraph. The new code is sort of

based on the page number handling and `\setline`. It seems that a lot of fiddling with the line number internals is required.

In November 2004 in order to accurately determine page numbers Peter Wilson added these to the macro. It is now:

`\@nl{<page counter number>}{<printed page number>}`

We do not (yet) use the printed number (i.e., the `\thepage`) but it may come in handy later. The macro `\fix@page` checks if a new page has started.

Exactly what `\@nl` does depends on whether right text is being processed. That's why many code is defined in `\@nl@reg` or `\nl@regR`.

```

879
880 \newcommand*{\@nl}[2]{%
881   \fix@page{#1}%
882   \ifledRcol%
883     \@nl@regR%
884   \else%
885     \@nl@reg%
886   \fi%
887 }
888 \newcommand*{\@nl@reg}{%
889   \ifx\l@dchset@num\relax \else
890     \advance\absline@num \@ne
891     \csgdef{l@dchset@num@the\absline@num}{}%To remember this line have
      been marked by a \setlinenum
892     \set@line@action
893     \let\l@dchset@num=\relax
894     \advance\absline@num \m@ne
895     \advance\line@num \m@ne
896   \fi
897 %

```

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

```

898 \advance\absline@num \@ne
899   \ifx\next@page@num\relax \else
900     \page@action
901     \let\next@page@num=\relax
902   \fi
903   \ifx\sub@change\relax \else
904     \ifnum\sub@change>\z@
905       \sublines@true
906     \else
907       \sublines@false
908     \fi
909     \sub@action
910     \let\sub@change=\relax
911   \fi
912 %

```

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

```

913     \ifcase\@lock
914     \or
915     \@lock \tw@
916     \or \or
917     \@lock \z@
918     \fi
919     \ifcase\sub@lock
920     \or
921     \sub@lock \tw@
922     \or \or
923     \sub@lock \z@
924     \fi
925 %

```

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

```

926     \ifsublines@
927     \ifnum\sub@lock<\tw@
928     \advance\subline@num \@ne
929     \fi
930     \else
931     \ifnum\@lock<\tw@
932     \advance\line@num \@ne \subline@num \z@
933     \fi
934     \fi}
935 %
936 %

```

`\last@page@num` `\fix@page` basically replaces `\@page`. It determines whether or not a new page has been started, based on the page values held by `\@n1`.

```

937 \newcount\last@page@num
938 \last@page@num=-10000
939
940 \newcommand*{\fix@page}[1]{%
941   \ifledRcol
942   \ifnum #1=\last@page@numR
943   \else
944     \csnumgdef{\lastabsline@forpageR\the\page@numR}{\the\absline@numR}%
945     \csnumgdef{\lastline@forpageR\the\page@numR}{\the\line@numR}%
946     \ifbypage@R
947     \ifx\l@dchset@num\relax%Not resetting if preceded by a \setlinenum
948     \line@numR \z@ \subline@numR \z@%
949     \fi%
950     \fi
951     \global\page@numR=#1%
952     \global\last@page@numR=#1%
953     \global\def\next@page@numR{#1}%

```

```

954 \fi
955 \else
956 \ifnum #1=\last@page@num
957 \else
958 \csnumgdef{\lastabsline@forpage@the\page@num}{\the\absline@num}%
959 \csnumgdef{\lastline@forpage@the\page@num}{\the\line@num}%
960 \ifbypage@
961 \ifx\l@dchset@num\relax%Not resetting if preceded by a \setlinenum
962 \line@num \z@ \subline@num \z@%
963 \fi%
964 \fi
965 \global\page@num=#1%
966 \global\last@page@num=#1%
967 \global\def\next@page@num{#1}%
968 \listxadd{\normal@page@break}{\the\absline@num}
969 \fi
970 \fi}
971 %

```

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

```

972 \newcommand*\@pend}[1]{}
973 \newcommand*\@pendR}[1]{}
974 \newcommand*\@lopL}[1]{}
975 \newcommand*\@lopR}[1]{}
976
977 %

```

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

```

978 \newcommand*\sub@on{\ifsublines@
979 \let\sub@change=\relax
980 \else
981 \def\sub@change{1}%
982 \fi}
983 \newcommand*\sub@off{\ifsublines@
984 \def\sub@change{-1}%
985 \else
986 \let\sub@change=\relax
987 \fi}
988
989 %

```

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

```

990
991 \newcommand*{\@adv}[1]{%
992   \ifsublines@
993     \ifledRcol
994       \advance\subline@numR by #1\relax
995       \ifnum\subline@numR<\z@
996         \led@warn@BadAdvancelineSubline
997         \subline@numR \z@
998       \fi
999     \else
1000       \advance\subline@num by #1\relax
1001       \ifnum\subline@num<\z@
1002         \led@warn@BadAdvancelineSubline
1003         \subline@num \z@
1004       \fi
1005     \fi
1006   \else
1007     \ifledRcol
1008       \advance\line@numR by #1\relax
1009       \ifnum\line@numR<\z@
1010         \led@warn@BadAdvancelineLine
1011         \line@numR \z@
1012       \fi
1013     \else
1014       \advance\line@num by #1\relax
1015       \ifnum\line@num<\z@
1016         \led@warn@BadAdvancelineLine
1017         \line@num \z@
1018       \fi
1019     \fi
1020   \fi
1021   \set@line@action}
1022
1023 %

```

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

```

1024
1025 \newcommand*{\@set}[1]{%
1026   \ifledRcol
1027     \ifsublines@
1028       \subline@numR=#1\relax
1029     \else
1030       \line@numR=#1\relax
1031     \fi
1032     \set@line@action
1033   \else
1034     \ifsublines@
1035       \subline@num=#1\relax

```

```

1036 \else
1037 \line@num=#1\relax
1038 \fi
1039 \set@line@action
1040 \fi}
1041
1042 %

```

\l@d@set The `\l@d@set{<num>}` macro sets the line number for the next `\pstart` to the value specified as its argument. This is used to implement `\setlinenum`.

\l@dchset@num `\l@dchset@num` is a flag to the `\@nl?` macro. If it is not `\relax` then a linenum change is to be done.

```

1043
1044 \newcommand*{\l@d@set}[1]{%
1045 \ifledRcol
1046 \line@numR=#1\relax
1047 \advance\line@numR \@ne
1048 \def\l@dchset@num{#1}
1049 \else
1050 \line@num=#1\relax
1051 \advance\line@num \@ne
1052 \def\l@dchset@num{#1}
1053 \fi}
1054 \let\l@dchset@num\relax
1055
1056 %

```

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

```

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

```

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

```

1067
1068 \newcommand*{\set@line@action}{%
1069 \ifledRcol
1070 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
1071 \ifsublines@
1072 \@l@tempcnta=-\subline@numR

```

```

1073 \else
1074   \@l@tempcnta=-\line@numR
1075 \fi
1076 \advance\@l@tempcnta by -5000\relax
1077 \xright@appenditem{\the\@l@tempcnta}\to\actions@listR
1078 \else
1079   \xright@appenditem{\the\absline@num}\to\actionlines@list
1080   \ifsublines@
1081     \@l@tempcnta=-\subline@num
1082   \else
1083     \@l@tempcnta=-\line@num
1084   \fi
1085   \advance\@l@tempcnta by -5000\relax
1086   \xright@appenditem{\the\@l@tempcnta}\to\actions@list
1087 \fi}
1088 %

```

\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.

```

1089
1090 \newcommand*{\sub@action}{%
1091   \ifl@Rcol
1092     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
1093     \ifsublines@
1094       \xright@appenditem{-1001}\to\actions@listR
1095     \else
1096       \xright@appenditem{-1002}\to\actions@listR
1097     \fi
1098   \else
1099     \xright@appenditem{\the\absline@num}\to\actionlines@list
1100     \ifsublines@
1101       \xright@appenditem{-1001}\to\actions@list
1102     \else
1103       \xright@appenditem{-1002}\to\actions@list
1104     \fi
1105   \fi}
1106 %

```

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

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

```

1107 \newcommand*{\lock@on}{\futurelet\next\do@lockon}
1108

```



```

1109 \newcommand*{\do@lockon}{%
1110   \ifx\next\lock@off
1111     \global\let\lock@off=\skip@lockoff
1112   \else
1113     \ifledRcol
1114       \do@lockonR
1115     \else
1116       \do@lockonL
1117     \fi
1118   \fi}
1119
1120
1121 \newcommand*{\do@lockonL}{%
1122   \xright@appenditem{\the\absline@num}\to\actionlines@list
1123   \ifsublines@
1124     \xright@appenditem{-1005}\to\actions@list
1125     \ifnum\sub@lock=\z@
1126       \sub@lock \@ne
1127     \else
1128       \ifnum\sub@lock=\thr@@
1129         \sub@lock \@ne
1130       \fi
1131     \fi
1132   \else
1133     \xright@appenditem{-1003}\to\actions@list
1134     \ifnum\@lock=\z@
1135       \@lock \@ne
1136     \else
1137       \ifnum\@lock=\thr@@
1138         \@lock \@ne
1139       \fi
1140     \fi
1141   \fi}
1142
1143 %

```

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

```

\do@lockoff
\do@lockoffL
\skip@lockoff
1144 \newcommand*{\do@lockoffL}{%
1145   \xright@appenditem{\the\absline@num}\to\actionlines@list
1146   \ifsublines@
1147     \xright@appenditem{-1006}\to\actions@list
1148     \ifnum\sub@lock=\tw@
1149       \sub@lock \thr@@
1150     \else
1151       \sub@lock \z@
1152     \fi
1153   \else
1154     \xright@appenditem{-1004}\to\actions@list
1155     \ifnum\@lock=\tw@

```

```

1156 \@lock \thr@@
1157 \else
1158 \@lock \z@
1159 \fi
1160 \fi}
1161
1162 \newcommand*{\do@lockoff}{%
1163 \ifledRcol
1164 \do@lockoffR
1165 \else
1166 \do@lockoffL
1167 \fi}
1168 \newcommand*{\skip@lockoff}{\global\let\lock@off=\do@lockoff}
1169 \global\let\lock@off=\do@lockoff
1170
1171 %

```

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

```

1172 \newcommand*{\n@num}{%
1173 \ifledRcol%
1174 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
1175 \xright@appenditem{-1007}\to\actions@listR
1176 \else%
1177 \xright@appenditem{\the\absline@num}\to\actionlines@list%
1178 \xright@appenditem{-1007}\to\actions@list%
1179 \fi%
1180 }%
1181
1182 %

```

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

```

1183 \newcommand*{\n@num@stanza}{%
1184 \ifledRcol%
1185 \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1186 \xright@appenditem{-1008}\to\actions@listR%
1187 \else%
1188 \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1189 \xright@appenditem{-1008}\to\actions@list%
1190 \fi%
1191 }
1192 %

```

\ifl@dhidenumber `\hidenumbering` hides number in margin. It uses action code 1009. `\hidenumberingonleftpage` and `\hidenumberingonrightpage` are variants, using action code only conditionally.

\hidenumberingonleftpage

\hidenumberingonrightpage

```

\h@num93 \newif\ifl@dhidenumber
1194 \newcommand*{\hidenumbering}{
1195   \ifledRcol%
1196     \write\linenum@outR{\string\hide@num}%
1197   \else%
1198     \write\linenum@out{\string\hide@num}%
1199   \fi%
1200 }%
1201 \newcommand*{\hide@num}{%
1202   \ifledRcol%
1203     \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1204     \xright@appenditem{-1009}\to\actions@listR%
1205   \else%
1206     \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1207     \xright@appenditem{-1009}\to\actions@list%
1208   \fi%
1209 }
1210 \newcommand*{\hidenumberingonleftpage}{%
1211   \ifledRcol%
1212     \write\linenum@outR{\string\hide@num@left}%
1213   \else%
1214     \write\linenum@out{\string\hide@num@left}%
1215   \fi%
1216 }%
1217
1218 \newcommand*{\hide@num@left}{%
1219   \ifledRcol%
1220     \ifodd\page@numR\else%
1221       \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1222       \xright@appenditem{-1009}\to\actions@listR%
1223     \fi%
1224   \else%
1225     \ifodd\page@num\else%
1226       \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1227       \xright@appenditem{-1009}\to\actions@list%
1228     \fi%
1229   \fi%
1230 }%
1231
1232 \newcommand*{\hidenumberingonrightpage}{%
1233   \ifledRcol%
1234     \write\linenum@outR{\string\hide@num@right}%
1235   \else%
1236     \write\linenum@out{\string\hide@num@right}%
1237   \fi%
1238 }%
1239
1240 \newcommand*{\hide@num@right}{%
1241   \ifledRcol%

```

```

1242 \ifodd\page@numR%
1243 \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1244 \xright@appenditem{-1009}\to\actions@listR%
1245 \fi%
1246 \else%
1247 \ifodd\page@num%
1248 \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1249 \xright@appenditem{-1009}\to\actions@list%
1250 \fi%
1251 \fi%
1252 }%
1253
1254 %

```

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

- #1, the number of entries to add to \insertlines@list for this reference. This value, here and within \edtext, which computes it and writes it to the line-list file, will be stored in the count \insert@count.

```

1255 \newcount\insert@count
1256 %

```

- #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.

```

1257 \newcommand*\@dummy@ref}[2]{#2}
1258 %

```

\@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.

```

1259 \newcommand*\@ref}[2]{%
1260 \ifledRcol%
1261 \@ref@regR{#1}{#2}%
1262 \else%
1263 \@ref@reg{#1}{#2}%
1264 \fi%
1265 }%
1266 \newcommand*\@ref@reg}[2]{%
1267 \global\insert@count=#1\relax
1268 \global\advance\@edtext@level by 1%
1269 \loop\ifnum\insert@count>\z@
1270 \xright@appenditem{\the\absline@num}\to\insertlines@list
1271 \global\advance\insert@count \m@ne

```

```

1272 \repeat
1273 %

```

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.

```

1274 \begingroup
1275   \let\@ref=\dummy@ref
1276   \let\@lopL\@gobble
1277   \let\page@action=\relax
1278   \let\sub@action=\relax
1279   \let\set@line@action=\relax
1280   \let\@lab=\relax
1281   \let\@lemma=\relax%
1282   \let\@sw\@gobblethree%
1283   #2
1284   \global\endpage@num=\page@num
1285   \global\endline@num=\line@num
1286   \global\endsubline@num=\subline@num
1287 \endgroup
1288 %

```

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

```

1289 \xright@appenditem%
1290   {\the\page@num|\the\line@num|}%
1291   \ifsublines@ \the\subline@num \else 0\fi}%
1292   \the\endpage@num|\the\endline@num|}%
1293   \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@list
1294 %

```

And now, call `\@ref@reg@parsearg`, which can be also called by `\@ref@later`

```

1295   \@ref@reg@parse{#2}%
1296 %

```

Decrease edtext level counter.

```

1297 \global\advance\@edtext@level by -1%
1298 }
1299 %

```

`\@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.

```

1300 \newcommand{\@ref@reg@parse}[1]{%
1301   \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\@edtext@level\endcsname}%

```

```

1302 \providebool{lemmacommand@the\edtext@level}%
1303 \boolfalse{lemmacommand@the\edtext@level}%
1304 %

```

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

```

1305 #1%
1306 %

```

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.

```

1307 \ifnum\edtext@level>0%
1308 \def\create@this@edtext@level{\expandafter\list@create\expandafter{\csname sw@list@edtext@the\edtext@level\endcsname}}%
1309 \ifcsundef{sw@list@edtext@the\edtext@level}\create@this@edtext@level\fi%
1310 \letcs{\tmp}{sw@list@edtext@the\edtext@level}%
1311 \letcs{\tmpp}{sw@list@edtext@tmp@the\edtext@level}%
1312 \xright@appenditem{\expandonce\tmpp}{to\tmp}%
1313 \global\cslet{sw@list@edtext@the\edtext@level}{\tmp}%
1314 \fi%
1315 %
1316 }
1317 %
1318 %

```

\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.

```

1319 \newcommand{\@ref@later}[1]{%
1320 \global\advance\edtext@level by \ne%
1321 \ifledRcol%
1322 \@ref@reg@parseR{#1}%
1323 \else%
1324 \@ref@reg@parse{#1}%
1325 \fi%
1326 \global\advance\edtext@level by -\ne%
1327 }%
1328 %

```

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.

```

1329 \newwrite\linenum@out
1330 %

```

```

\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.

```

1331 \newif\iffirst@linenum@out@
1332 \first@linenum@out@true
1333 %

```

```
\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.

```

1334 \newcommand{\this@line@list@version}{7}%
1335 %

```

```

\line@list@stuff
\next@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.

```

1336 \let\next@line@list@stuff\relax%
1337 \newcommand*{\line@list@stuff}[1]{%
1338 %

```

First, define a toggle set to true when we are not in the first run.

```

1339 \global\newtoggle{notfirstrun@#1}%
1340 \IfFileExists{\l@auxdir#1}%
1341 {\global\toggletrue{notfirstrun@#1}}%
1342 {\global\togglefalse{notfirstrun@#1}}%
1343 %

```

Use the commands of the previous section to interpret the line-list file from the last run.

```

1344 \read@linelist{#1}%
1345 %

```

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. `reledmac` and `reledpar` can fill the `\next@line@list@stuff` hook between a `\endnumbering` (associated with numbered file n) and a `\beginnumbering` (associated with numbered file $n + 1$). It allows adding content to the numbered file $n + 1$ and not n .

```

1346 \iffirst@linenum@out@
1347   \global\first@linenum@out@false%
1348   \immediate\openout\linenum@out=\l@auxdir#1\relax%
1349   \immediate\write\linenum@out{\string\line@list@version{\
this@line@list@version}}}%
1350   \ifl@dpaging%
1351     \immediate\write\linenum@out{\string\@par@sync@option{\
@par@this@sync@option}}}%
1352   \fi%
1353   \else
1354 %

```

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.

```

1355   \if@minipage%
1356     \leavevmode%
1357     \fi%
1358     \closeout\linenum@out%
1359     \openout\linenum@out=\l@auxdir#1\relax%
1360     \write\linenum@out{\string\line@list@version{\this@line@list@version}}%
1361 %
1362   \ifl@dpaging%
1363     \write\linenum@out{\string\@par@sync@option{\@par@this@sync@option}}%
1364 %
1365   \fi%
1366   \fi%
1367   \next@line@list@stuff%
1368   \global\let\next@line@list@stuff\relax%
1369 }%

```

`\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.

```

1370 \newcommand*{\new@line}{%
1371   \IfStrEq{\led@pb@setting}{after}%
1372   {\xifinlist{\the\absline@num}{\l@prev@nopb}%
1373     {\xifinlist{\the\absline@num}{\normal@page@break}%
1374       {\numdef{\@next@page}{\c@page+\@ne}%
1375         \write\linenum@out{\string\@nl[\@next@page][\@next@page]}}%
1376       }%
1377     {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}}%

```



```

1378 }%
1379 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}}%
1380 {}%
1381 \IfStrEq{\led@pb@setting}{before}%
1382 {\numdef{\next@absline}{\the\absline@num+\@ne}%
1383 \xifinlist{\next@absline}{\l@prev@nopb}%
1384 {\xifinlist{\the\absline@num}{\normal@page@break}%
1385 {\numgdef{\nc@page}{\c@page+\@ne}%
1386 \write\linenum@out{\string\@nl[\nc@page][\nc@page]}}%
1387 }%
1388 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}}%
1389 }%
1390 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}}%
1391 }%
1392 {}%
1393 \IfStrEqCase{\led@pb@setting}{{before}{\relax}{after}{\relax}}{\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1394 }
1395
1396 %

```

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

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

\flag@end

```

1397 \newif\if@noneed@Footnote%
1398
1399 \newcommand*{\flag@start}{%
1400 \ifledRcol%
1401 \edef\next{\write\linenum@outR{%
1402 \string\@ref[\the\insert@countR] []}}%
1403 \next%
1404 \ifnum\insert@countR<1%
1405 \if@noneed@Footnote\else%
1406 \led@err@EdtextWithoutFootnote%
1407 \fi%
1408 \fi%
1409 \else%
1410 \edef\next{\write\linenum@out{%
1411 \string\@ref[\the\insert@count] []}}%
1412 \next%
1413 \ifnum\insert@count<1%
1414 \if@noneed@Footnote\else%
1415 \led@err@EdtextWithoutFootnote%
1416 \fi%
1417 \fi%
1418 \fi}%

```

```

1419
1420 \newcommand*{\flag@end}{%
1421   \ifledRcol%
1422     \write\linenum@outR{}}%
1423   \else%
1424     \write\linenum@out{}}%
1425   \fi}%
1426
1427
1428 %

```

`\flag@start@RTL` With Xe_{La}TeX, 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 typeset in one line, but it is **not** reversed when this first argument is typeset 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.

```

1429 \newcommand{\flag@start@RTL}{%
1430   \edlabel{edtext:start:\csuse{thisedtext@the\@edtext@level}}}%
1431   \IfStrEq{\xabslineref{edtext:start:\csuse{thisedtext@the\@edtext@level}}}{%
1432     {000}}%
1433     {}%
1434     {%
1435       \ifnumequal%
1436         {\xabslineref{edtext:start:\csuse{thisedtext@the\@edtext@level}}}{%
1437           {\xabslineref{edtext:end:\csuse{thisedtext@the\@edtext@level}}}}%
1438         {\flag@end}%
1439         {\flag@start}}%
1440     }%
1441 }%

```

²⁵This problem is caused by the way Xe_{La}TeX manages right-to-left typesetting. 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.

```

1442 \newcommand{\flag@end@RTL}{%
1443   \edlabel{edtext:end:\csuse{thisedtext@the\@edtext@level}}}%
1444   \IfStrEq{\xabslineref{edtext:start:\csuse{thisedtext@the\@edtext@level
1445   }}}%
1446     {000}%
1447     {}%
1448     {%
1449     \ifnumequal%
1450       {\xabslineref{edtext:start:\csuse{thisedtext@the\@edtext@level}}}%
1451       {\xabslineref{edtext:end:\csuse{thisedtext@the\@edtext@level}}}%
1452       {\flag@start}%
1453       {\flag@end}%
1454     }%
1455   }%
1456   %

```

`\flag@start@later` `\flag@start@later` and `\flag@end@later`: these send the `\@ref@later` to the line-list file command to the line-list file

```

1457 \newcommand*{\flag@start@later}{%
1458   \ifledRcol%
1459     \write\linenum@outR{\string\@ref@later []}%
1460   \else%
1461     \write\linenum@out{\string\@ref@later []}%
1462   \fi%
1463 }%
1464 \newcommand{\flag@end@later}{%
1465   \ifledRcol%
1466     \write\linenum@outR{[]}%
1467   \else%
1468     \write\linenum@out{[]}%
1469   \fi%
1470 }
1471 %

```

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

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

```

1472
1473
1474 \newcommand*{\startsub}{\dimen0\lastskip
1475 \ifdim\dimen0>Opt \unskip \fi
1476 \ifledRcol \write\linenum@outR{\string\sub@on}%
1477 \else \write\linenum@out{\string\sub@on}%
1478 \fi
1479 \ifdim\dimen0>Opt \hskip\dimen0 \fi}
1480 \def\endsub{\dimen0\lastskip
1481 \ifdim\dimen0>Opt \unskip \fi
1482 \ifledRcol \write\linenum@outR{\string\sub@off}%
1483 \else \write\linenum@out{\string\sub@off}%
1484 \fi
1485 \ifdim\dimen0>Opt \hskip\dimen0 \fi}
1486
1487 %

```

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

```

1488 \newcommand*{\advanceline}[1]{\leavevmode%
1489 \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
1490 \else \write\linenum@out{\string\@adv[#1]}%
1491 \fi%
1492 }
1493 %

```

\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.

```

1494
1495 \newcommand*{\setline}[1]{%
1496 \leavevmode%
1497 \ifnum#1<\z@
1498 \led@warn@BadSetline
1499 \else
1500 \ifledRcol \write\linenum@outR{\string\@set[#1]}%
1501 \else \write\linenum@out{\string\@set[#1]}%
1502 \fi
1503 \fi}
1504
1505 %

```

\setlinenum You can use `\setlinenum{<num>}` before a `\pstart` to set the visible line-number to a specified positive value. It writes a `\led@set` command to the line-list file.

```

1506
1507 \newcommand*{\setlinenum}[1]{%
1508 \ifnum#1<\z@
1509 \led@warn@BadSetlinenum

```

```

1510 \else
1511 \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
1512 \else \write\linenum@out{\string\l@d@set[#1]} \fi
1513 \fi}
1514
1515 %

```

\startlock You can use **\startlock** or **\endlock** in running text to start or end line number locking at the current line. They decide whether line numbers or sub-line numbers are affected, depending on the current state of the sub-lineation flags.

\endlock

```

1516
1517 \newcommand*{\startlock}{%
1518 \ifledRcol \write\linenum@outR{\string\lock@on}%
1519 \else \write\linenum@out{\string\lock@on}%
1520 \fi}
1521 \def\endlock{%
1522 \ifledRcol \write\linenum@outR{\string\lock@off}%
1523 \else \write\linenum@out{\string\lock@off}%
1524 \fi}
1525 %

```

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

\ifl@dskipversenumber

\l@dskipnumbertrue

\l@dskipnumberfalse

\skipnumbering

```

1526 \newif\ifl@dskipnumber
1527 \newif\ifl@dskipversenumber%
1528 \newcommand*{\skipnumbering}{%
1529 \leavevmode%
1530 \ifledRcol%
1531 \ifinstanza%
1532 \write\linenum@outR{\string\n@num@stanza}%
1533 \else%
1534 \write\linenum@outR{\string\n@num}%
1535 \fi%
1536 \advanceline{-1}%
1537 \else%
1538 \ifinstanza%
1539 \write\linenum@out{\string\n@num@stanza}%
1540 \else%
1541 \write\linenum@out{\string\n@num}%
1542 \fi%
1543 \advanceline{-1}%
1544 \fi%
1545 }%
1546
1547 %

```

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. 148). Instead, the appropriate note-generating command is appended to the list macro `\inserts@list`, and when `\pend` completes the paragraph it inserts all the notes at the proper places.

Note that we do not provide previous-note information, although it is often wanted; your own macros must handle that. We cannot do it correctly without keeping track of what kind of notes have gone past: it is not just a matter of remembering the line numbers associated with the previous invocation of `\edtext`, because that might have been for a different kind of note. It is preferable for your footnote macros to store and recall this kind of information if they need it.

VI.1 `\edtext` itself

The various note-generating macros might want to request that commands be executed not at once, but in close connection with the start or end of the lemma. For example,

footnote numbers in the text should be connected to the end of the lemma; or, instead of a single macro to create a note listing variants, you might want to use several macros in series to create individual variants, which would each add information to a private macro or token register, which in turn would be formatted and output when all of #2 for the lemma has been read.

`\end@lemmas` To 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.

```
1548 \list@create{\end@lemmas}
1549 %
```

`\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.

```
1550 \newcommand{\dummy@edtext}[2]{#1}
1551 %
```

`\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`.

```
1552 \newcommand{\dummy@edtext@showlemma}[2]{\showlemma{#1}}%
1553 %
```

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

`\no@expands` `\morenoexpands` We need to turn off macro expansion for certain sorts of macros we are likely to see within the lemma and within the notes.

The first class is font-changing macros. We suppress expansion for them by letting them become equal to zero.²⁶ This is done because we want to pass into our notes the

²⁶Since ‘control sequences equivalent to characters are not expandable’—*The TeXbook*, answer to Exercise 20.14.

generic commands to change to roman or whatever, and not their expansions that will ask for a particular style at a specified size. The notes may well be in a smaller font, so the command should be expanded later, when the note's environment is in effect.

A second sort to turn off includes a few of the accent macros. Most are not a problem: an accent that is expanded to an `\accent` command may be harder to read but it works just the same. The ones that cause problems are: those that use alignments— \TeX seems to get confused about the difference between alignment parameters and macro parameters; those that use temporary control sequences; and those that look carefully at what the current font is.

(The `\copyright` macro defined in PLAIN \TeX has this sort of problem as well, but is not used enough to bother with. That macro, and any other that causes trouble, will get by all right if you put a `\protect` in front of it in your file.)

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

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

The `\new@series` command also adds `\let\footnote(X)\@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 Lua \TeX or Xe \LaTeX .)

```

1554 \newcommand*{\no@expands}{%
1555   \let\select@lemmafnt=0%
1556   \let\startsub=\relax \let\endsub=\relax
1557   \let\startlock=\relax \let\endlock=\relax
1558   \let\edlabel=\@gobble
1559   \let\setline=\@gobble \let\advanceline=\@gobble
1560   \let\sameword\sameword@inedtext%
1561   \let\edtext=\dummy@edtext
1562   \let\edindex\dummy@edindex%
1563   \l@dtabnoexpands
1564   \morenoexpands}
1565 \let\morenoexpands=\relax

```



```
1566
1567 %
```

\@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.

```
1568 \newcommand{\@tag}{}
1569 %
```

\@edtext@level This counter is increased by 1 at each level of \edtext.

```
1570 \newcount\@edtext@level%
1571 \@edtext@level=0%
1572 %
```

\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.

```
1573 \newif\if@edtext@secondarg%
1574 %
```

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

```
1575 \newcounter{edtext}
1576 \renewcommand{\theedtext}{\edtxt@arabic{edtext}}%
1577 %
```

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

```
1578 \newcommand{\edtext}[2]{\leavevmode%
1579 %
```

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

```
1580 \ifnumberedpar%
1581 %
```

we increment the \@edtext@level \TeX counter to know in which level of \edtext we are.

```
1582 \global\advance\@edtext@level by 1%
1583 %
```

We also increase the edtext \TeX counter to insert a hypertarget if the hyperref package is loaded, and also works with \edtext on right-to-left typesetting with Xe \TeX .

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.

```
1584 \stepcounter{edtext}%
1585 \csxdef{thisedtext@the\@edtext@level}{\theedtext}%
1586 %
```

By default, we do not use `\lemma`

```
1587 \global\@lemmacommand@false%
1588 %
```

```
1589 \begingroup%
1590 %
```

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`.

```
1591 \ifledRcol%
1592 \ifcsvoid{sw@list@edtextR@the\@edtext@level}%
1593 {\global\let\sw@inthisedtext\empty}%
1594 {\expandafter\gl@p\csname sw@list@edtextR@the\@edtext@level\
endcsname\to\sw@inthisedtext}%
1595 \else%
1596 \ifcsvoid{sw@list@edtext@the\@edtext@level}%
1597 {\global\let\sw@inthisedtext\empty}%
1598 {\expandafter\gl@p\csname sw@list@edtext@the\@edtext@level\
endcsname\to\sw@inthisedtext}%
1599 \fi%
1600 %
```

`\@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.

```
1601 \global\renewcommand{\@tag}{%
1602 \no@expands #1%
1603 }%
1604 %
```

`\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`.

```
1605 \set@line%
1606 %
```

`\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`.

```
1607 \ifledRcol \global\insert@countR \z@%
1608 \else \global\insert@count \z@ \fi%
1609 %
```

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.

```

1610     \@edtext@secondarg@true%
1611     \ignorespaces #2\relax%
1612     \@edtext@secondarg@false%
1613 %

```

With \LaTeX , 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.

```

1614     \if@RTL%
1615         \flag@start@RTL%
1616     \else%
1617         \flag@start%
1618     \fi%
1619 %

```

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

```

1620     \if@lemmacommand@%
1621         \ifledRcol%
1622             \write\linenum@outR{\string\@lemma}%
1623         \else%
1624             \write\linenum@out{\string\@lemma}%
1625         \fi%
1626     \fi%
1627 %

```

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.

```

1628     \endgroup%
1629     \ifdef{\hypertarget}%
1630     {%
1631         \Hy@raisedlink@left{\hypertarget{\csuse{thisedtext@the\@edtext@level}:start}}}%
1632         \showlemma{#1}%
1633         \Hy@raisedlink{\hypertarget{\csuse{thisedtext@the\@edtext@level}:end}}}%
1634     }%
1635     {%
1636         \showlemma{#1}%
1637     }%
1638 %

```

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.

```

1639 \ifx\end@lemmas\empty \else%
1640 \gl@p\end@lemmas\to\x@lemma%
1641 \x@lemma%
1642 \global\let\x@lemma=\relax%
1643 \fi%
1644 \if@RTL%
1645 \flag@end@RTL%
1646 \else%
1647 \flag@end%
1648 \fi%
1649 %

```

We switch some flags to false.

- The one that checks having footnotes inside a `\edtext`.
- The one that says we are inside a `\edtext`. In fact, it is not a flag, but a counter which is increased to 1 in each level of `\edtext`.
- The one that says we are inside a `\@lemma`.

```

1650 \global\@noneed@Footnotefalse%
1651 \global\advance\@edtext@level by -1%
1652 \global\@lemmacommand@false%
1653 %

```

We also reset `\@beforeinsertofthisedtext`

```

1654 \global\let\@beforeinsertofthisedtext\relax%
1655 %

```

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

```

1656 \else%
1657 \showlemma{#1} (\textbf{\textsc{Edtext outside numbered paragraph}})\
1658 led@err@edtextoutsidestart%
1659 \fi%
1660 }%
1661 %
1662 %

```

`\@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`.

```

1663 \let\@beforeinsertofthisedtext\relax
1664 %

```

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

```
1665 \newif\ifnumberline
1666 \numberlinetrue
1667 %
```

`\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.

```
1668 \newcommand*{\set@line}{%
1669   \ifledRcol
1670   \ifx\line@listR\empty
1671     \global\noteschanged@true
1672     \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
1673   \else
1674     \gl@p\line@listR\to\@tempb
1675     \xdef\l@d@nums{\@tempb|\edfont@info}%
1676     \global\let\@tempb=\undefined
1677   \fi
1678 \else
1679   \ifx\line@list\empty
1680     \global\noteschanged@true
1681     \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
1682   \else
1683     \gl@p\line@list\to\@tempb
1684     \xdef\l@d@nums{\@tempb|\edfont@info}%
1685     \global\let\@tempb=\undefined
1686   \fi
1687 \fi}
1688
1689 %
```

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

```
1690 \newcommand*{\edfont@info}{\f@encoding/\f@family/\f@series/\f@shape}
1691
1692 %
```

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. 130).

```

1693 \newcommand*{\lemma}[1]{%
1694   \global\@lemmacommand@true%
1695   \global\renewcommand{\@tag}{%
1696     \no@expands #1%
1697   }%
1698   \ignorespaces%
1699 }%
1700 %

```

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

```

1701 \newcommand{\@lemma}{%
1702   \booltrue{lemmacommand@the\@edtext@level}%
1703 }%
1704 %

```

\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.

```

1705 \newif\if@lemmacommand@%
1706 %

```

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. 100): 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.

```

1707 \newcommand*{\linenum}[1]{%
1708   \xdef\@tempa{#1|}|}|}|}|}|}|noexpand\\l@d@nums}%
1709   \global\let\l@d@nums=\empty
1710   \expandafter\line@set\@tempa|\\ignorespaces}
1711 %

```

\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.

```

1712 \def\line@set#1|#2\|#3|#4\|{%
1713   \gdef\@tempb{#1}%
1714   \ifx\@tempb\empty
1715     \l@add{#3}%
1716   \else
1717     \l@add{#1}%
1718   \fi
1719   \gdef\@tempb{#4}%
1720   \ifx\@tempb\empty\else
1721     \l@add{|\}\line@set#2\|#4\|}%
1722   \fi}
1723 %

```

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

```

1724 \newcommand{\l@add}[1]{\xdef\l@d@nums{\l@d@nums#1}}
1725
1726 %

```

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:

bar¹ foo¹ foo² bar² foo³ (A)(B) foo⁴ bar³ (C) foo⁵ (D) bar⁴ (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 characters²⁸, we detokenize only for non- \XeTeX engines. In any case, in \XeTeX a `\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.

```

1727 \newcommand{\get@sw@txt}[1]{%
1728   \begingroup%
1729   \swnoexpands%
1730   %.

```

Using case sensibility option.

```

1731   \ifsw@caseinsensitive%
1732     \def\tmpa##1{\lowercase{##1}}%
1733   \else%
1734     \def\tmpa##1{##1}%
1735   \fi%
1736   %

```

And now, define `\sw@txt`.

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

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

```

1737 \ifxetex%
1738 \@tmpa{\xdef\sw@txt{#1}}%
1739 \else%
1740 \@tmpa{\expandafter\xdef\expandafter\sw@txt\expandafter{\detokenize
1741 {#1}}}%
1742 \fi%
1743 \endgroup%
1744 }%
1745 %

```

Allow some macros inside `\sameword`. We use `\RenewExpandableDocumentCommand` to get expandable command with optional argument. Cf. <https://tex.stackexchange.com/a/384783/7712>.

```

\sw@noexpand\newcommand{\swnoexpands}{%
1746 \let\sameword\@firstofone%Allow to have nested \sameword
1747 \let\emph\@firstofone%
1748 \let\textit\@firstofone%
1749 \let\textbf\@firstofone%
1750 \let\textsc\@firstofone%
1751 \let\framebox\@firstofone%
1752 \RenewExpandableDocumentCommand{\edindex}{om}{}%
1753 \ifdefined\index%
1754 \RenewExpandableDocumentCommand{\index}{om}{}%
1755 \fi%
1756 \let\selectlanguage\gobble%
1757 \let\foreignlanguage\@secondoftwo%
1758 \ifdefined\xpg@loaded%
1759 \renewcommand\do[1]{%
1760 \expandafter\RenewExpandableDocumentCommand\expandafter{\csname
1761 text##1\endcsname}{om}{###2}%
1762 }%
1763 \expandafter\docsvlist\expandafter{\xpg@loaded}%
1764 \fi%
1765 }%
1766 %

```

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

```

1766 \newcommandx{\sameword}[2][1,usedefault]{%
1767 \leavevmode%
1768 \get@sw@txt{#2}%
1769 %

```

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

```

1770 \unless\ifledRcol%
1771 \csnumgdef{sw@\sw@txt}{\csuse{sw@\sw@txt}+\@ne}%
1772 %

```

Then, write its value to the numbered file.

```

1773 \protected@write\linenum@out{}\string\@sw{\sw@txt}{\csuse{sw@\sw@txt
}}{#1}}%
1774 %

```

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

```

1775 \else%
1776 \csnumgdef{sw@\sw@txt}{\csuse{sw@\sw@txt}+\@ne}%
1777 \protected@write\linenum@outR{}\string\@sw{\sw@txt}{\csuse{sw@\sw@txt
}}{#1}}%
1778 \fi%
1779 %

```

And print the word.

```

1780 #2%
1781 }%
1782 %

```

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

```

\if@addsw83 \newif\if@addsw%
1784 %

```

\@sw The command printed in the auxiliary files.

```

1785 \newcommand{\@sw}[3]{%
1786 \get@sw@txt{#1}%
1787 \unless\ifledRcol%
1788 %

```

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

```

1789 \csxdef{sw@\sw@txt @\the\absline@num @\the\section@num}{#2}%
1790 %

```

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

```

1791 \numdef{\prev@line}{\the\absline@num-1}%
1792 \ifcsundef{sw@\sw@txt @\prev@line @\the\section@num}{%
1793 \csnumgdef{sw@\sw@txt @\prev@line @\the\section@num}{#2-1}%
1794 }{}%
1795 %

```

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

```

1796 \numdef{\the@sw}{#2-\csuse{sw@\sw@txt @\prev@line @\the\section@num}}%
1797 %

```

And do the same thing for the right side.

```

1798 \else%
1799 \csxdef{sw@sw@txt @\the\absline@numR @\the\section@numR @R}{#2}%
1800 \numdef{\prev@line}{\the\absline@numR-1}%
1801 \ifcsundef{sw@sw@txt @\prev@line @\the\section@numR @R}{%
1802 \csnumgdef{sw@sw@txt @\prev@line @\the\section@numR @R}{#2-1}%
1803 }{}%
1804 \numdef{\the@sw}{#2-\csuse{sw@sw@txt @\prev@line @\the\section@numR @R
1805 }}%
1806 \fi%
1807 %

```

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

```

1807 \@tempcnta=\@edtext@level
1808 \@whilenum{\@tempcnta>0}\do{%
1809 \ifcsdef{sw@list@edtext@tmp@\the\@tempcnta}%
1810 {%
1811 \addswfalse%
1812 \notbool{lemmacommand@\the\@tempcnta}%
1813 {\addswtrue}%
1814 {\IfStrEq{#3}{inlemma}%
1815 {\addswtrue}%
1816 {%
1817 \def\do##1{%
1818 \ifnumequal{##1}{\the\@tempcnta}%
1819 {\addswtrue\listbreak}%
1820 }%
1821 }%
1822 \docsvlist{#3}%
1823 }%
1824 }%
1825 \if@addsw%
1826 \letcs{\@tmp}{sw@list@edtext@tmp@\the\@tempcnta}%
1827 \ifledRcol%
1828 \xright@appenditem{\the@sw}{\the\absline@numR}\to\@tmp%
1829 \else%
1830 \xright@appenditem{\the@sw}{\the\absline@num}\to\@tmp%
1831 \fi%
1832 \cslet{sw@list@edtext@tmp@\the\@tempcnta}{\@tmp}%
1833 \fi%
1834 }%
1835 }%
1836 \advance\@tempcnta by -1%
1837 }%
1838 }%
1839 %

```

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

```

1840 \newcommandx{\sameword@inedtext}[2][1,usedefault]{%
1841 \get@sw@txt{#2}%

```

```

1842 \unless\ifledRcol@%
1843 %

```

Just a precaution.

```

1844 \ifx\sw@list@inedtext\empty%
1845 \def\the@sw{999}%
1846 \def\this@absline{-99}%
1847 \else%
1848 %

```

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

```

1849 \gl@p\sw@list@inedtext\to\@tmp%
1850 \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1851 \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1852 \fi%
1853 %

```

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

```

1854 \ifcsdef{sw@\sw@txt @\this@absline @\the\section@num}{%
1855 \numdef{\prev@line}{\this@absline-1}%
1856 \numdef{\sw@atthisline}{\csuse{sw@\sw@txt @\this@absline @\the\
section@num}-\csuse{sw@\sw@txt @\prev@line @\the\section@num}}%
1857 }%
1858 {\numdef{\sw@atthisline}{0}}%
1859 %

```

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

```

1860 \ifnumgreater{\sw@atthisline}{1}%
1861 {\showwordrank{#2}{\the@sw}}%
1862 {#2}%
1863 %

```

And the same for right side.

```

1864 \else%
1865 \ifx\sw@list@inedtext\empty%
1866 \def\the@sw{999}%
1867 \def\this@absline{-99}%
1868 \else%
1869 \gl@p\sw@list@inedtext\to\@tmp%
1870 \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1871 \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1872 \fi%
1873 \ifcsdef{sw@\sw@txt @\this@absline @\the\section@numR @R}{%
1874 \numdef{\prev@line}{\this@absline-1}%
1875 \numdef{\sw@atthisline}{\csuse{sw@\sw@txt @\this@absline @\the\
section@numR @R}-\csuse{sw@\sw@txt @\prev@line @\the\section@numR @R}}%
1876 }%

```

```

1877     {\numdef{\sw@atthisline}{0}}%
1878     \ifnumgreater{\sw@atthisline}{1}%
1879         {\showwordrank{#2}{\the@sw}}%
1880         {#2}%
1881     \fi%
1882 }%
1883 %

```

`\showwordrank` Finally, the way the rank will be printed.

```

1884 \newcommand{\showwordrank}[2]{%
1885     #1\textsuperscript{#2}%
1886 }%
1887 %

```

VII Paragraph decomposition and reassembly

In order to be able to count the lines of text and affix line numbers, we add an extra stage of processing for each paragraph. We send the paragraph into a box register, rather than straight onto the vertical list, and when the paragraph ends we slice the paragraph into its component lines; to each line we add any notes or line numbers, add a command to write to the line-list, and then at last send the line to the vertical list. This section contains all the code for this processing.

VII.1 Boxes, counters, `\pstart` and `\pend`

`\raw@text` Here are numbers and flags that are used internally in the course of the paragraph decomposition.

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

```

1888 \newbox\raw@text
1889 \newif\ifnumberedpar@
1890 \newcount\num@lines
1891 \newbox\one@line
1892 \newcount\par@line
1893 %

```

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

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

`\AtStartEveryPstart`

`\numberpstarttrue`

`\numberpstartfalse`

`\labelpstarttrue`

`\labelpstartfalse`

`\thepstart`

`\ifat@every@pstart@star@`

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.

```

1894
1895 \newcommand{\AtStartEveryPstart}[1]{%
1896   \ifstrempy{#1}%
1897     {\gdef\@at@start@every@pstart{}}%
1898     {\gdef\@at@start@every@pstart{#1}}%
1899 }%
1900 \def\@at@start@every@pstart{}%
1901
1902 \newif\ifat@every@pstart@star%
1903 \newcommand{\AtEveryPstart}[1]{%
1904   \ifstrempy{#1}%
1905     {\gdef\@at@every@pstart{}}%
1906     {\gdef\@at@every@pstart{\noindent#1}}%
1907   \global\@at@every@pstart@star@false%
1908 }%
1909 \WithSuffix\newcommand\AtEveryPstart*[1]{%
1910   \ifstrempy{#1}%
1911     {\gdef\@at@every@pstart{}}%
1912     {\gdef\@at@every@pstart{#1}}%
1913   \global\@at@every@pstart@star@true%
1914 }%
1915 \def\@at@every@pstart{}%
1916
1917 \newcounter{pstart}
1918 \renewcommand{\thepstart}{\bfseries\@arabic\c@pstart}. }
1919 \newif\ifnumberpstart
1920 \numberpstartfalse
1921 \newif\iflabelpstart
1922 \labelpstartfalse
1923 \newcommandx*\pstart[2][1,2,usedefault]{%
1924   \normal@pars%
1925   \ifboolexpr{%
1926     test {\ifstrempy{#1}}%
1927     and test {\ifstrempy{#2}}%
1928   }%
1929     {\@at@every@pstart}%
1930     {%
1931       \ifstrempy{#1}{\noindent#1}%
1932       \ifstrempy{#2}{#2}%
1933     }%
1934   \ifautopar%
1935     \autopar%
1936   \fi%
1937   \ifluatex%
1938     \edef\l@luatextextdir@L{\the\textdir}%
1939   \fi%

```

```

1940 \@nobreake>true%
1941 \ifnumbering \else%
1942   \led@err@PstartNotNumbered%
1943   \beginnumbering%
1944 \fi%
1945 \ifnumberedpar@%
1946   \led@err@PstartInPstart%
1947   \pend%
1948 \fi%
1949 \list@clear{\inserts@list}%
1950 \global\let\next@insert=\empty%
1951 \begingroup\normal@pars%
1952 \global\advance \l@dnumpstartsL\@ne
1953 \global\setbox\raw@text=\vbox\bgroup%
1954   \if@nbreak%
1955     \if@afterindent\else%
1956       \noindent%
1957       \global\@afterindenttrue%
1958     \fi%
1959   \fi%
1960   \ifautopar\else%
1961     \ifnumberpstart%
1962       \ifinstanza\else%
1963         \ifsidepstartnum\else%
1964           \thepstart%
1965         \fi%
1966       \fi%
1967     \fi%
1968   \fi%
1969 \numberedpar@true%
1970 \iflabelpstart%
1971   \protected@edef\@currentlabel{\p@pstart\thepstart}%
1972 \fi%
1973 \l@dzeropenalties%
1974 \@at@start@every@pstart%
1975 \ignorespaces%because not automatically ignored if an optional argument
    is used (classical TeX behavior for space after commands)
1976 }
1977 %

```

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

```

1978 \newcommand*{\pend}[2][1,2,usedefault]{\ifnumbering \else%
1979   \led@err@PendNotNumbered%
1980 \fi%
1981 \global\l@dskipversenumberfalse%
1982 \ifnumberedpar@ \else%
1983   \led@err@PendNoPstart%
1984 \fi%
1985 %

```


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.

```

1986 \l@dzeropenalties%
1987 \@at@end@every@pend%
1988 \endgraf\global\num@lines=\prevgraf\egroup%
1989 \global\par@line=0%
1990 %

```

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`.

```

1991 \csnumdef{pstartline}{0}%
1992 \loop\ifvbox\raw@text%
1993   \csnumdef{pstartline}{\pstartline+\@ne}%
1994   \do@line%
1995   \ifbypstart@%
1996     \ifnumequal{\pstartline}{1}{%
1997       \bgroup%
1998       \let\leavevmode\relax%
1999       \setline{1}%
2000       \egroup%
2001       \resetprevline@}{}%
2002     \fi%
2003   \repeat%
2004 %

```

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

```

2005 \flush@notes%
2006 \endgroup%
2007 \ignorespaces%
2008 %

```

Increase `pstart` counter.

```

2009 \ifnumberpstart%
2010   \global\pstartnumtrue%
2011 \fi%
2012 \addtocounter{pstart}{1}%
2013 \ifcontinuousnumberingwithcolumns%
2014   \addtocounter{pstartL}{1}%
2015   \addtocounter{pstartR}{1}%
2016 \fi%
2017 %

```

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

```

2018 \normal@pars%
2019 \ifboolexpr{%
2020     test {\ifstrempy{#1}}%
2021     and test {\ifstrempy{#2}}%
2022 }%
2023     {\at@every@pend}%
2024     {%
2025     \ifstrempy{#1}{\noindent#1}%
2026     \ifstrempy{#2}{#2}%
2027     }%
2028 %

```

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.

```

2029 \@nobreakfalse%
2030 \ifautopar%
2031     \autopar%
2032 \fi%
2033 }
2034 %

```

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%
2035
2036
2037 \newif\ifat@every@pend@star%
2038 \newcommand{\AtEveryPend}[1]{%
2039     \ifstrempy{#1}%
2040     {\gdef\at@every@pend{}}%
2041     {\gdef\at@every@pend{\noindent#1}}%
2042     \global\at@every@pend@star@false%
2043 }%
2044 \WithSuffix\newcommand\AtEveryPend*[1]{%
2045     \ifstrempy{#1}%
2046     {\gdef\at@every@pend{}}%
2047     {\gdef\at@every@pend{#1}}%
2048     \global\at@every@pend@star@true%
2049 }%
2050 \xdef\at@every@pend{}%
2051
2052 %

```

\AtEndEveryPend Here a macro to insert automatically any content at the end of `\pend`, in numbered lines.

```

2053 \newcommand{\AtEndEveryPend}[1]{%
2054     \ifstrempy{#1}%
2055     {\xdef\at@end@every@pend{}}%

```

```

2056 {\gdef\@at@end@every@pend{#1}}%
2057 }%
2058 \def\@at@end@every@pend{}%
2059 %

```

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

```

2060 \newcommand*\l@dzeropenalties{%
2061   \brokenpenalty \z@ \clubpenalty \z@
2062   \displaywidowpenalty \z@ \interlinepenalty \z@ \predisplaypenalty \z@
2063   \postdisplaypenalty \z@ \widowpenalty \z@}
2064
2065 %

```

\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.

```

2066 \newif\ifaautopar
2067 \autoparfalse
2068 \newcommand*\autopar{%
2069   \ifledRcol
2070     \ifnumberingR \else
2071       \led@err@AutoparNotNumbered
2072     \beginnumberingR
2073     \fi
2074   \else
2075     \ifnumbering \else

```

```

2076 \led@err@AutoparNotNumbered
2077 \beginnumbering
2078 \fi
2079 \fi
2080 \autopartrue
2081 \everypar{\setbox0=\lastbox
2082 \endgraf \vskip-\parskip
2083 \pstart \noindent \kern\wd0 \ifnumberpstart\ifinstanza\else\thepstart\
fi\fi
2084 \let\par=\pend}%
2085 \ignorespaces}
2086 %

```

\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.

```

2087 \newcommand*{\normal@pars}{\everypar{}\let\par\endgraf}
2088
2089 %

```

\ifautopar@pause We define a boolean test switched to true at the beginning of the \pausenumbering command if the autopar is enabled. This boolean will be tested at the beginning of \resumenumbering to continue the autopar if needed.

```

2090 \newif\ifautopar@pause
2091 %

```

VII.2 Processing one line

VII.2.1 General process

\do@line The \do@line macro is called by \pend to do all the processing for a single line of text.
\l@dunhbox@line

```

2092 \newcommand*{\l@dunhbox@line}[1]{\unhbox #1}
2093 \newcommand*{\do@line}{%
2094 {\vbadness=10000
2095 \splittopskip=\z@
2096 \do@linehook
2097 \l@demtyd@ta
2098 \global\setbox\one@line=\vsplit\raw@text to\baselineskip}%
2099 \unvbox\one@line \global\setbox\one@line=\lastbox
2100 \getline@num
2101 \IfStrEq{\led@pb@setting}{before}{\led@check@pb\led@check@nopb}{}}
2102 \ifnum\@lock>\@ne
2103 \inserthangingsymboltrue
2104 \else
2105 \inserthangingsymbolfalse
2106 \fi

```

```

2107 \check@pb@in@verse
2108 \ifl@dhidenumber%
2109 \global\l@dhidenumberfalse%
2110 \f@x@l@cks%
2111 \else%
2112 \affixline@num%
2113 \fi%
2114 %

```

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

```

2115 \xifinlist{\the\l@dnumpstartsL}{\eled@sections@@}%
2116 {\print@eledsection}%
2117 {\print@line}%
2118 \IfStrEq{\led@pb@setting}{after}{\led@check@pb\led@check@nopb}{%
2119 }%
2120 %

```

VII.2.2 Process for “normal” line

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

```

2121 \def\print@line{
2122 %

```

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

```

2123 \affixpstart@num%
2124 %

```

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

```

2125 \hb@xt@ \linewidth{%
2126 %

```

User hook.

```

2127 \do@insidelinehook%
2128 %

```

Left line number

```

2129 \l@dld@ta%
2130 %

```

Prepare text to be inserted before notes.

```

2131 \if@firstlineofpage%
2132 \set@Xtxtbeforenotes%
2133 \set@txtbeforenotesX%
2134 \global\@firstlineofpagefalse%
2135 \fi%
2136 %

```

Insert footnotes made of manuscripts data and critical footnotes.

```

2137 \ifdefstring{\ms@data@position}{msdata-regular}{%
2138 \insert@msdata%
2139 \add@inserts%
2140 \add@Xgroupbyline%
2141 }{%
2142 \add@inserts%
2143 \add@Xgroupbyline%
2144 \insert@msdata%
2145 }%
2146 %

```

Insert marginal notes.

```

2147 \affixside@note%
2148 %

```

Print left notes.

```

2149 \l@dlsn@te
2150 %

```

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

```

2151 {\ledllfill\hb@xt@ \wd\one@line{\new@line%
2152 %

```

If we use Lua \TeX then restore the direction.

```

2153 \ifluatex%
2154 \textdir\l@luatextextdir@L%
2155 \fi%
2156 %

```

Insert, if needed, the hanging symbol.

```

2157 \inserthangingsymbol%
2158 %

```

And so, print the line.

```

2159 \l@dunhbox@line{\one@line}}%
2160 %

```

Right line number

```

2161 \ledrlfill\l@drd@ta%
2162 %

```

Print right notes.

```

2163 \l@drsn@te
2164 }}%
2165 %

```

And reinsert penalties (for page breaking)...

```

2166 \add@penalties%
2167 }
2168 %

```

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.

```

2169 \def\print@eledsection{%
2170   \if@firstlineofpage%
2171     \set@Xtxtbeforenotes%
2172     \set@txtbeforenotesX%
2173     \global\@firstlineofpagefalse%
2174   \fi%
2175   \ifdefstring{\ms@data@position}{msdata-regular}{%
2176     \insert@msdata%
2177     \add@inserts%
2178     \add@Xgroupbyline%
2179   }{%
2180     \add@inserts%
2181     \add@Xgroupbyline%
2182     \insert@msdata%
2183   }%
2184   \affixside@note%
2185   \numdef{\temp@}{\l@dnumpstartsL-1}%
2186   \xifinlist{\temp@}{\eled@sections@}{\@nbreaktrue}{\@nbreakfalse}%
2187   \@eled@sectioningtrue%
2188   \csuse{eled@sectioning@the\l@dnumpstartsL}%
2189   \@eled@sectioningfalse%
2190   \global\csundef{eled@sectioning@the\l@dnumpstartsL}%
2191   \if@RTL%
2192     \hspace{-3\paperwidth}%
2193     {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
2194   \else%
2195     \hspace{3\paperwidth}%
2196     {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
2197   \fi%
2198   \vskip-\baselineskip%
2199 }
2200 %

```

VII.2.4 Hooks

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

\do@insidelinehook

```

2201 \newcommand*{\do@linehook}{%
2202 \newcommand*{\do@insidelinehook}{%
2203 %

```

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

```

2204 \newcommand*{\dolinehook}[1]{\gdef\do@linehook{#1}}%
2205 \newcommand*{\doinsidelinehook}[1]{\gdef\do@insidelinehook{#1}}%
2206
2207 %

```

VII.2.5 Sidenotes and marginal line number initialization

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

```

\l@dcsnotetext
2208 \newcommand*{\l@demptyd@ta}{%
\l@dcsnotetext@l
2209 \gdef\l@dld@ta{}%
\l@dcsnotetext@r
2210 \gdef\l@drd@ta{}%
2211 \gdef\l@dcsnotetext@l{}%
2212 \gdef\l@dcsnotetext@r{}%
2213 \gdef\l@dcsnotetext{}%
2214
2215 %

```

`\l@dlsn@te` Zero width boxes of the left and right side notes, together with their kerns, and, even-
`\l@drsn@te` tually, with additional space if we are in parallel columns typesetting.

```

2216 \newcommand{\l@dlsn@te}{%
2217 \ifboolexpr{%
2218     bool {l@dprintingcolumns}%
2219     and bool {ledRcol}%
2220 }{% If we are on a right column
2221     \@tempdima=\@morespace@leftnote@rightcolumn%
2222 }{%
2223     \@tempdima=\z@%
2224 }%
2225 \hb@xt@ \z@{\hss\box\l@dldp@rbox\kern\ledlsnotesep\hskip\@tempdima}%
2226 }%
2227 \newcommand{\l@drsn@te}{%
2228 \ifboolexpr{%
2229     bool {l@dprintingcolumns}%
2230     and not bool {ledRcol}%
2231 }{% If we are on a left column
2232     \@tempdima=\@morespace@rightnote@leftcolumn%
2233 }{%
2234     \@tempdima=\z@%
2235 }%

```



```

2236 \hb@xt@ \z@{\hskip\@tempdima\kern\ledrsnotesep\box\l@drp@rbox\hss}%
2237 }%
2238
2239 %

```

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

```

2240 \newcommand*{\ledllfill}{\hfil}
2241 \newcommand*{\ledrlfill}{\hfil}
2242
2243 %

```

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.

```

2244 \newcommand*{\getline@num}{%
2245   \global\advance\absline@num \@ne%
2246   \do@actions
2247   \do@ballast
2248   \ifnumberline
2249     \ifsublines@
2250       \ifnum\sub@lock<\tw@
2251         \global\advance\subline@num \@ne
2252       \fi
2253     \else
2254       \ifnum\@lock<\tw@
2255         \global\advance\line@num \@ne
2256         \global\subline@num \z@
2257       \fi
2258     \fi
2259   \fi
2260 }
2261 %

```

`\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. 164).

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

`\c@ballast`

```

2262 \newcount\ballast@count
2263 \newcounter{ballast}
2264 \setcounter{ballast}{0}
2265 %

```

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.

```

2266 \newcommand*{\do@ballast}{\global\ballast@count \z@
2267 \begingroup
2268 \advance\absline@num \@ne
2269 \ifnum\next@actionline=\absline@num
2270 \ifnum\next@action>-1001\relax
2271 \global\advance\ballast@count by -\c@ballast
2272 \fi
2273 \fi
2274 \endgroup}
2275 %

```

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

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

```

2276 \newcommand*{\do@actions}{%
2277 \global\let\do@actions@next=\relax
2278 \ifnum\absline@num<\next@actionline\else
2279 %

```

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.

```

2280 \ifnum\next@action>-1001
2281 \global\page@num=\next@action
2282 \ifresumenummering@start%
2283 \setbox0=\hbox{}%Required to get the correct page number, when the
\resumenummering is just after a \newpage
2284 \ifnum\pausenumbering@page@num<\page@num%
2285 \global\resumenummering@startfalse%
2286 \fi%
2287 \fi%
2288 \unless\ifresumenummering@start%
2289 \global\@firstlineofpagetrue%
2290 \fi%
2291 \ifbypage@
2292 \ifcsdef{l@dchset@num@\the\absline@num}%
2293 {%
2294 \global\csundef{l@dchset@num@\the\absline@num}%

```

```

2295     }%
2296     {%
2297     \unless\ifresumenumbering@start%
2298     \global\line@num=\z@ \global\subline@num=\z@%
2299     \resetprevline@%
2300     \fi%
2301     }%
2302 \fi
2303 \global\resumenumbering@startfalse%
2304 \add@msdata@firstlineofpage%
2305 %

```

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`.)

```

2306 \else
2307 \ifnum\next@action<-4999
2308 \@l@dtmpcnta=-\next@action
2309 \advance\@l@dtmpcnta by -5001
2310 \ifsublines@
2311 \global\subline@num=\@l@dtmpcnta
2312 \else
2313 \global\line@num=\@l@dtmpcnta
2314 \fi
2315 %

```

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

```

2316 \else
2317 \@l@dtmpcnta=-\next@action
2318 \advance\@l@dtmpcnta by -1000
2319 \do@actions@fixedcode
2320 \fi
2321 \fi
2322 %

```

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.

```

2323 \ifx\actionlines@list\empty
2324 \gdef\next@actionline{1000000}%
2325 \else
2326 \gl@p\actionlines@list\to\next@actionline
2327 \gl@p\actions@list\to\next@action
2328 \global\let\do@actions@next=\do@actions
2329 \fi
2330 \fi
2331 %

```

Make the recursive call, if necessary.

```

2332 \do@actions@next}
2333
2334 %

```

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

```

2335 \newcommand*{\do@actions@fixedcode}{%
2336   \ifcase\@l@dttempcnta
2337     \or%           % 1001 = starting sublineation
2338       \global\sublines@true
2339     \or%           % 1002 = ending sublineation
2340       \global\sublines@false
2341     \or%           % 1003 = starting locking number
2342       \global\@lock=\@ne
2343     \or%           % 1004 = ending locking number
2344       \ifnum\@lock=\tw@
2345         \global\@lock=\thr@@
2346       \else
2347         \global\@lock=\z@
2348       \fi
2349     \or%           % 1005 = starting locking subnumber
2350       \global\sub@lock=\@ne
2351     \or%           % 1006 = ending locking subnumber
2352       \ifnum\sub@lock=\tw@
2353         \global\sub@lock=\thr@@
2354       \else
2355         \global\sub@lock=\z@
2356       \fi
2357     \or%           % 1007 = skipping numbering
2358       \l@dskipnumbertrue
2359     \or%           % 1008 = skipping numbering in stanza
2360       \l@dskipversenumbertrue%
2361     \or%           % 1009 = hiding number
2362       \l@dhiddenumbertrue
2363     \or%           % 1010 = inserting msdata
2364       \add@msdata%
2365     \else
2366       \led@warn@BadAction
2367     \fi}
2368
2369
2370 %

```

IX Line number printing

`\affixline@num` `\affixline@num` just puts a left line number into `\l@dld@ta` or a right line number into `\l@drd@ta` if required.

To determine whether we need to affix a line number to this line, we compute the following:

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

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

(where *int* truncates a real number to an integer). *m* will be equal to *linenum* only if we are to paste a number on here. However, the formula breaks down for the first line to number (and any before that), so we check that case separately: if `\line@num ≤ \firstlinenum`, we compare the two directly instead of making these calculations.

We compute, in the scratch counter `\@l@tempcnta`, the number of the next line that should be printed with a number (*m* in the above discussion), and move the current line number into the counter `\@l@tempcntb` for comparison.

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

```
2371 \newcommand*{\affixline@num}{%
2372 %
```

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).

```
2373 \ifledgroupnotesL@else
2374 \ifnumberline
2375 \ifl@dskipnumber
2376 \global\l@dskipnumberfalse
2377 \else
2378 \ifsublines@
2379 \l@tempcntb=\subline@num
2380 \ifnum\subline@num>\c@firstsublinenum
2381 \l@tempcnta=\subline@num
2382 \advance\l@tempcnta by-\c@firstsublinenum
2383 \divide\l@tempcnta by\c@sublinenumincrement
2384 \multiply\l@tempcnta by\c@sublinenumincrement
2385 \advance\l@tempcnta by\c@firstsublinenum
2386 \else
2387 \l@tempcnta=\c@firstsublinenum
2388 \fi
2389 %
```

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.

```
2390 \ch@cksub@l@ck
2391 %
```

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

```
2392 \else
2393 \l@tempcntb=\line@num
2394 %
```

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

```

2395         \ifx\linenumberlist\empty
2396           \ifnum\line@num>\c@firstlinenum
2397             \@l@tempcnta=\line@num
2398             \advance\@l@tempcnta by-\c@firstlinenum
2399             \divide\@l@tempcnta by\c@linenumincrement
2400             \multiply\@l@tempcnta by\c@linenumincrement
2401             \advance\@l@tempcnta by\c@firstlinenum
2402           \else
2403             \@l@tempcnta=\c@firstlinenum
2404           \fi
2405         \else
2406 %

```

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

```

2407         \@l@tempcnta=\line@num
2408         \edef\rem@inder{\linenumberlist,\number\line@num,}%
2409         \edef\sc@n@list{\def\noexpand\sc@n@list
2410           #####1,\number\@l@tempcnta,####2|{\def\noexpand\rem@inder
2411 {#####2}}}%
2412         \sc@n@list\expandafter\sc@n@list\rem@inder|}%
2413         \ifx\rem@inder\empty%
2414           \advance\@l@tempcnta\@ne
2415         \fi
2416 %

```

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

```

2417         \ch@ck@l@ck
2418       \fi
2419 %

```

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

```

2420         \ifnum\@l@tempcnta=\@l@tempcntb
2421         \ifl@dskipversenumber\else
2422 %

```

If we got here, we are going to print a line number; so now we need to calculate a number that will tell us which side of the page will get the line number. We start from `\line@margin`, which asks for one side always if it is less than 2; and then if the side does depend on the page number, we simply add the page number to this side code—because the values of `\line@margin` have been devised so that this produces a number that is even for left-margin numbers and odd for right-margin numbers.

For \LaTeX we have to consider two column documents as well. In this case Peter Wilson thought we need to put the numbers at the outside of the column — the left of the first column and the right of the second. Do the `twocolumn` stuff before going on with the original code.

`\l@dld@ta` A left line number is stored in `\l@dld@ta` and a right one in `\l@drd@ta`.
`\l@drd@ta`

```

2423         \if@twocolumn
2424             \if@firstcolumn
2425                 \gdef\l@dld@ta{\llap{\leftlinenum}}}%
2426             \else
2427                 \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
2428             \fi
2429         \else
2430             \ifboolexpr{bool {l@dprintingcolumns} and test {\
2431 ifnumgreater{\line@margin@columns}{\m@ne}}}%
2432                 {\@l@tempcntb=\line@margin@columns}%
2433                 {\@l@tempcntb=\line@margin}%
2434                 \ifnum\@l@tempcntb>\@ne
2435                     \advance\@l@tempcntb \page@num
2436                 \fi
2437             \ifboolexpr{%
2438                 bool {l@dprintingcolumns}%
2439                 and (%
2440                     (test {\ifdefstring{\linenum@OnlyPages@ForColumns}{left
2441 }}%
2442                     and test {\ifnumodd{\page@num}}%
2443                     )%
2444                     or%
2445                     (test {\ifdefstring{\linenum@OnlyPages@ForColumns}{right
2446 }}%
2447                     and not test {\ifnumodd{\page@num}}%
2448                     )%
2449                 )%
2450                 }%
2451                 {%
2452                     \ifodd\@l@tempcntb%
2453                         \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
2454                     \else%
2455                         \gdef\l@dld@ta{\llap{\leftlinenum}}}%
2456                     \fi%
2457                 }%
2458             \fi
2459         \fi

```

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

```

2460         \fix@locks
2461     \fi
2462 \fi
2463 \fi
2464 }
2465

```

2466 %

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

```
2467 \newcommand*{\ch@cksub@l@ck}{%
2468   \ifcase\sub@lock
2469   \or
2470     \ifnum\sublock@disp=\@ne
2471       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2472     \fi
2473   \or
2474     \ifnum\sublock@disp=\tw@ \else
2475       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2476     \fi
2477   \or
2478     \ifnum\sublock@disp=\z@
2479       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2480     \fi
2481   \fi}
2482 %
```

Similarly for line numbers.

```
2483 \newcommand*{\ch@ck@l@ck}{%
2484   \ifcase@lock
2485   \or
2486     \ifnum\lock@disp=\@ne
2487       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2488     \fi
2489   \or
2490     \ifnum\lock@disp=\tw@ \else
2491       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2492     \fi
2493   \or
2494     \ifnum\lock@disp=\z@
2495       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2496     \fi
2497   \fi}
2498 %
```

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

```
2499 \newcommand*{\f@x@l@cks}{%
2500   \ifcase@lock
2501   \or
2502     \global\@lock=\tw@
2503   \or \or
2504     \global\@lock=\z@
```



```

2505 \fi
2506 \ifcase\sub@lock
2507 \or
2508 \global\sub@lock=\tw@
2509 \or \or
2510 \global\sub@lock=\z@
2511 \fi}
2512
2513 %

```

X Pstart number printing in side

In side, the printing of pstart number is running like the printing of line number. There is only some differences:

- The pstarts counter is upgrade in the `\pend` command. Consequently, the `\affixpstart@num` command has not to upgrade it, unlike the `\affixline@num` which upgrades the lines counter.
- To print the pstart number only at the beginning of a pstart, and not in every line, a boolean test is made. The `\pstartnum` boolean is set to TRUE at every `\pend`. It is tried in the `\leftpstartnum` and `\rightpstartnum` commands. After the try, it is set to FALSE.

```

\leftpstartnum14
\rightpstartnum15 \newif\ifsidepstartnum
\ifsidepstartnum16 \newcommand*{\affixpstart@num}{%
2517 \ifsidepstartnum
2518 \if@twocolumn
2519 \if@firstcolumn
2520 \gdef\l@dld@ta{\llap{\leftpstartnum}}}%
2521 \else
2522 \gdef\l@drd@ta{\rlap{\rightpstartnum}}}%
2523 \fi
2524 \else
2525 \l@dttempcntb=\line@margin%
2526 \ifnum\l@dttempcntb>\@ne
2527 \advance\l@dttempcntb \page@num
2528 \fi
2529 \ifodd\l@dttempcntb
2530 \gdef\l@drd@ta{\rlap{\rightpstartnum}}}%
2531 \else
2532 \gdef\l@dld@ta{\llap{\leftpstartnum}}}%
2533 \fi
2534 \fi
2535 \fi
2536
2537 }

```

```

2538 %
2539
2540 \newif\ifpstartnum
2541 \pstartnumtrue
2542 \newcommand*{\leftpstartnum}{%
2543   \ifpstartnum\thepstart
2544   \kern\linenumsep\fi
2545   \global\pstartnumfalse
2546 }
2547 \newcommand*{\rightpstartnum}{%
2548   \ifpstartnum
2549   \kern\linenumsep
2550   \thepstart
2551   \fi
2552   \global\pstartnumfalse
2553 }
2554 %

```

XI Restoring footnotes and penalties

Because of the paragraph decomposition process in order to number line, `reledmac` must hack the standard way \TeX works in order to manage insertion of footnotes, both critical and familiar.

We need to call the `\insert` commands not when the content of `\pstart...\pend` is read by \TeX but when each individual line is typeset.

Consequently, when reading the content of `\pstart...\pend`, we store the insertion (footnotes) in an specific `reledmac`'s list, and we restore them to the vertical list when printing each individual line.

XI.1 Add insertions to the vertical list

`\inserts@list` `\inserts@list` is the list macro that contains the inserts that we save up for one paragraph.

```

2555 \list@create{\inserts@list}
2556 %

```

`\add@inserts` `\add@inserts` is the penultimate macro used by `\do@line`; it takes insertions saved in a list macro and sends them onto the vertical list.

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

```

2557 \newcommand*{\add@inserts}{%
2558   \global\let\add@inserts@next=\relax
2559 %

```

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

```
2560 \ifx\inserts@list\empty \else
2561 %
```

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.

```
2562 \ifx\next@insert\empty
2563 \ifx\insertlines@list\empty
2564 \global\noteschanged@true
2565 \gdef\next@insert{100000}%
2566 \else
2567 \glp\insertlines@list\to\next@insert
2568 \fi
2569 \fi
2570 %
```

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.

```
2571 \ifnum\next@insert=\absline@num
2572 \glp\inserts@list\to\@insert
2573 \@insert
2574 \global\let\@insert=\undefined
2575 \global\let\next@insert=\empty
2576 \global\let\add@inserts@next=\add@inserts
2577 \fi
2578 \fi
2579 %
```

Make the recursive call, if necessary.

```
2580 \add@inserts@next}
2581
2582 %
```

`\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`.

```
2583 \newcommand{\add@Xgroupbyline}{%
2584 \unless\ifnocritical%
2585 \def\do##1{%Looping on the series
2586 \def\do###1{%Looping on the ##1@forinserting command
2587 \ifcsdef{##1@forinserting@###1}{%
2588 \Xbeforeinsertion{##1}%
2589 \if@ledgroup%
2590 \global\setbox\@nameuse{mp##1footins}=\vbox%
```

```

2591 \else%
2592 \insert\csname ##1footins\endcsname%
2593 \fi%
2594 {%
2595 \ifcsdef{Xhsize\csuse{series@display##1}@##1}%
2596 {\hsize \csuse{Xhsize\csuse{series@display##1}@##1}}%
2597 }%
2598 \if@ledgroup%
2599 \unvbox\@nameuse{mp##1footins}%
2600 \fi%
2601 \X@atbegininsertion{##1}%
2602 \ifcsstring{series@display##1}%
2603 {%
2604 \Xledsetnormalparstuff{##1}%
2605 \rule\z@\splittopskip%
2606 }%
2607 }%
2608 \csuse{##1@forinserting@###1}%
2609 \strut\par%
2610 }%
2611 \global\csundef{##1@forinserting@###1}%
2612 }%
2613 {%
2614 }%
2615 \ifcsdef{##1@forinserting}{%
2616 \dolistcslloop{##1@forinserting}%
2617 }{%
2618 \global\csundef{##1@forinserting}%
2619 }%
2620 \dolistloop{\@series}%
2621 \fi%
2622 }%
2623
2624
2625 %

```

XI.2 Penalties

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

In this code, \num@lines is the number of lines in the whole paragraph, and \par@line is the line we are working on at the moment. The count \@l@dttempcnta is used to calculate and accumulate the penalty; it is initially set to the value of \ballast@count, which has been worked out in \do@ballast above (VIII p. 153). Finally, the penalty is checked to see that it does not go below -10000 .

```

2626 \newcommand*{\add@penalties}{\@l@tempcnta=\ballast@count
2627 \ifnum\num@lines>\@ne
2628 \global\advance\par@line \@ne
2629 \ifnum\par@line=\@ne
2630 \advance\@l@tempcnta \clubpenalty
2631 \fi
2632 \@l@tempcntb=\par@line \advance\@l@tempcntb \@ne
2633 \ifnum\@l@tempcntb=\num@lines
2634 \advance\@l@tempcnta \widowpenalty
2635 \fi
2636 \ifnum\par@line<\num@lines
2637 \advance\@l@tempcnta \interlinepenalty
2638 \fi
2639 \fi
2640 \ifnum\@l@tempcnta=\z@
2641 \relax
2642 \else
2643 \ifnum\@l@tempcnta>-10000
2644 \penalty\@l@tempcnta
2645 \else
2646 \penalty -10000
2647 \fi
2648 \fi}
2649
2650 %

```

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. For the first run, we do not flush the notes, as that means all the notes will be added at the end of numbered section, and so, very far of the expected position

```

2651 \newcommand*{\flush@notes}{%
2652 \iftoggle{notfirststrun@jobname.\extensionchars\the\section@num}{%
2653 \@xloop%
2654 \ifx\inserts@list\empty \else%
2655 \gl@p\inserts@list\to\@insert%
2656 \@insert%
2657 \global\let\@insert=\undefined%
2658 \repeat%
2659 }{}%
2660 }%

```

```
2661
2662
2663 %
```

\@xloop \@xloop is a variant of the PLAIN T_EX \loop macro, useful when it's hard to construct a positive test using the T_EX \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 T_EX \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.

```
2664 \def\@xloop#1\repeat{%
2665   \def\body{#1\expandafter\body\fi}%
2666   \body}
2667
2668 %
```

XI.4 Text before notes

\set@Xtxtbeforenotes The \set@Xtxtbeforenotes macro resets the Xtxtbeforenotes@⟨series⟩@typeset boolean to false. Just before the first note of the ⟨series⟩ in a page, the \Xtextbeforenotes will be inserted.

```
2669 \newcommand{\set@Xtxtbeforenotes}{%
2670   \unless\ifnocritical@%
2671     \def\do##1{%
2672       \nottoggle{Xtxtbeforenotesonlyonce@##1}{%
2673         \global\togglefalse{Xtxtbeforenotes@##1@typeset}%
2674       }{}%
2675     }%
2676     \dolistloop{\@series}%
2677   \fi%
2678 }%
2679 %
```

\set@txtbeforenotesX The \set@txtbeforenotesX does the same for the \textbeforenotesX.

```
2680 \newcommand{\set@txtbeforenotesX}{%
2681   \unless\ifnofamiliar@%
2682     \def\do##1{%
2683       \nottoggle{txtbeforenotesonlyonceX@##1}{%
2684         \global\togglefalse{txtbeforenotesX@##1@typeset}%
2685       }{}%
2686     }%
2687     \dolistloop{\@series}%
2688   \fi%
2689 }%
2690 %
```

`\insert@Xtxtbeforenotes` `\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.

```

2691 \newcommand{\insert@Xtxtbeforenotes}[1]{%
2692 \nottoggle{Xtxtbeforenotes@#1@typeset}{%
2693 \global\toggletrue{Xtxtbeforenotes@#1@typeset}%
2694 \ifcsvoid{Xtxtbeforenotes@#1}{-%
2695 \ifcsstring{series@display#1}{paragraph}%
2696 {\noindent\csuse{Xtxtbeforenotes@#1}}%
2697 {\expandafter\insert\csname#1footins\endcsname%
2698 \bgroup%
2699 \noindent%
2700 \ifcsdef{\csuse{series@display#1}@begin@insert}{%
2701 \csuse{\csuse{series@display#1}@begin@insert}{#1}%
2702 }}%
2703 \strut\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}%
2704 \egroup%
2705 }%
2706 }%
2707 }%
2708 {}%
2709 }%
2710
2711
2712 \newcommand{\insert@txtbeforenotesX}[1]{%
2713 \nottoggle{txtbeforenotesX@#1@typeset}{%
2714 \global\toggletrue{txtbeforenotesX@#1@typeset}%
2715 \ifcsvoid{txtbeforenotesX@#1}{-%
2716 \ifcsstring{series@displayX#1}{paragraph}%
2717 {\noindent\csuse{txtbeforenotesX@#1}}%
2718 {\expandafter\insert\csname#1\endcsname%
2719 \bgroup%
2720 \noindent%
2721 \ifcsdef{\csuse{series@displayX#1}@begin@insert}{%
2722 \csuse{\csuse{series@displayX#1}@begin@insert}{#1}%
2723 }}%
2724 \strut\csuse{notefontsizeX@#1}\csuse{txtbeforenotesX@#1}%
2725 \egroup%
2726 }%
2727 }%
2728 }%
2729 {}%
2730 }%
2731
2732
2733 %

```

XII Critical footnotes

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

XII.1 Fonts

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

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

```

2734 \def\select@lemmafont#1|#2|#3|#4|#5|#6|#7|{\select@@lemmafont#7|}
2735 \def\select@@lemmafont#1/#2/#3/#4|{%
2736   {\fontencoding{#1}\fontfamily{#2}\fontseries{#3}\fontshape{#4}%
2737   \selectfont}
2738
2739 %
```

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.

```

2740 \newcommand*\footnoteoptions@[3]{%
2741   \def\do##1{%
2742     \ifstrequal{#1}{L}{% In Leftside
2743       \xright@appenditem{\noexpand\setkeys[mac]{#3footnoteoption}{\
2744       unexpanded{##1}}}{\to\inserts@list%
2745       \global\advance\insert@count \@ne% Increment the left insert
2746       counter.
2747       }%
2748       }%
2749       \xright@appenditem{\noexpand\setkeys[mac]{#3footnoteoption}{\
2750       unexpanded{##1}}}{\to\inserts@listR%
2751       \global\advance\insert@countR \@ne% Increment the right insert
2752       counter insert.
2753       }%
2754       }%
2755       \notblank{#2}{\docsvlist{#2}}{}}% Parsing all options
2756   }
```


XII.3 Notes language

`\footnotelang@lua` `\footnotelang@lua` is called to remember the information about the direction of a lemma when Lua¹TeX is used.

```

2754 \newcommand*{\footnotelang@lua}[1][1=L,usedefault]{%
2755   \ifstrequal{#1}{L}{%
2756     \xright@appenditem{\csxdef{footnote@luatextextdir}{\the\textdir}}{\to\
inserts@list%Know the dir of lemma
2757     \global\advance\insert@count \@ne%
2758     \xright@appenditem{\csxdef{footnote@luatexpardir}{\the\pardir}}{\to\
inserts@list%Know the dir of lemma
2759     \global\advance\insert@count \@ne%
2760   }%
2761   {%
2762     \xright@appenditem{\csxdef{footnote@luatextextdir}{\the\textdir}}{\to\
inserts@listR%Know the dir of lemma
2763     \global\advance\insert@countR \@ne%
2764     \xright@appenditem{\csxdef{footnote@luatexpardir}{\the\pardir}}{\to\
inserts@listR%Know the dir of lemma
2765     \global\advance\insert@countR \@ne%
2766   }%
2767 }
2768 %

```

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

```

2769 \newcommand*{\footnotelang@poly}[1][1=L,usedefault]{%
2770   \ifstrequal{#1}{L}{%
2771     \if@RTL%
2772       \xright@appenditem{\csxdef{footnote@dir}{@RTLtrue}}{\to\
inserts@list%Know the language used in the lemma
2773       \global\advance\insert@count \@ne%
2774     }%
2775     \else
2776       \xright@appenditem{\csxdef{footnote@dir}{@RTLfalse}}{\to\
inserts@list%Know the language of lemma
2777       \global\advance\insert@count \@ne%
2778     }%
2779     \fi%
2780     \xright@appenditem{\csxdef{footnote@lang}{\expandonce\language}}{\to\
inserts@list%Know the language of lemma
2781     \global\advance\insert@count \@ne%
2782   }%
2783   {%
2784     \if@RTL
2785       \xright@appenditem{\csxdef{footnote@dir}{@RTLtrue}}{\to\
inserts@listR%Know the language of lemma
2786       \global\advance\insert@countR \@ne%
2787     }%
2788     \else
2789       \xright@appenditem{\csxdef{footnote@dir}{@RTLfalse}}{\to\
inserts@listR%Know the language of lemma

```

```

2787 \global\advance\insert@countR \@ne%
2788 \fi
2789 \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\language\language}}}{\
to\inserts@listR%Know the language of lemma
2790 \global\advance\insert@countR \@ne%
2791 }%
2792 }
2793 %

```

XII.4 General survey of the way we manage notes

The processing of each note is done by four principal macros: the `\vfootnote` macro takes the text of the footnote and does the `\insert`; it calls on the `\footfmt` macro to select the right fonts, print the line number and lemma, and do any other formatting needed for that individual note. Within the output routine, the two other macros, `\footstart` and `\footgroup`, are called; the first prints extra vertical space and a footnote rule, if desired; the second does any reformatting of the whole set of the footnotes in this series for this page—such as paragraphing or division into columns—and then sends them to the page.

These four macros, and the other macros and parameters shown here, are distinguished by the ‘series letter’ that indicates which set of the footnotes we are dealing with—A, B, C, D, or E. The series letter always precedes the string `foot` in macro and parameter names. Hence, for the A series, the four macros are called `\vAfootnote`, `\Afootfmt`, `\Afootstart`, and `\Afootgroup`.

These macros are changed depending of the footnotes arrangement: “normal”, “paragraphed”, “two columns” or “three columns”.

XII.5 General setup

`\footsplitskips` Some setup code that is common for a variety of the footnotes. The setup is for:

- `\interlinepenalty`.
- `\splittopskip` (skip before last part of notes that flow from one page to another).
- `\splitmaxdepth`.
- `\floatingpenalty`, that is penalty values being added when a long note flows from one page to another. Here, we let it to 0 when we are processing parallel pages in `eledpar`, in order to allow notes to flow from left to right pages and *vice-versa*. Otherwise, we let it to `\@MM`, which is the standard \TeX `\floatingpenalty`.

```

2794 \newcommand*{\footsplitskips}{%
2795 \interlinepenalty=\interfootnotelinepenalty
2796 \unless\ifl@dprintingpages%
2797 \floatingpenalty=\@MM%
2798 \fi%

```

```

2799 \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
2800 \leftskip=\z@skip \rightskip=\z@skip
2801
2802 %

```

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

```

2803 \let\normalfootnoterule=\footnoterule
2804 %

```

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.

```

2805 \newcommand{\Xarrangement}[2][1,usedefault]{%
2806   \def\do##1{%
2807     \csname Xarrangement@#2\endcsname{##1}%
2808   }%
2809   \ifstrempy{#1}%
2810     {%
2811       \dolistloop{\@series}%
2812     }%
2813     {
2814       \docsvlist{#1}%
2815     }%
2816   }%
2817   %

```

XII.6.2 Normal footnote

\Xarrangement@normal We can now define all the parameters for the series of footnotes; initially they use the “normal” footnote formatting.

What we want to do here is to insert something like the following for each footnote series. (This is an example, not part of the actual reledmac code.)

```

\skip\Afootins=12pt plus5pt minus5pt
\count\Afootins=1000
\dimen\Afootins=0.8\vsiz
\let\vAfootnote=\normalvfootnote \let\Afootfmt=\normalfootfmt
\let\Afootstart=\normalfootstart \let\Afootgroup=\normalfootgroup
\let\Afootnoterule=\normalfootnoterule

```

(Read *The TeXbook* in order to understand what are the counter, skip and dimen associated to an insertion.)

Instead of repeating ourselves, we define a `\Xarrangement@normal` macro that makes all these assignments for us, for any given series letter. This command is called when people use `\Xarrangement[⟨series⟩]{normal}`

Now we set up the `\Xarrangement@normal` macro itself. It takes one argument: the footnote series letter.

```

2818 \newcommand*{\Xarrangement@normal}[1]{%
2819   \csgdef{series@display#1}{normal}
2820   \expandafter\let\csname #1footstart\endcsname=\normalfootstart
2821   \expandafter\let\csname v#1footnote\endcsname=\normalvfootnote
2822   \expandafter\let\csname #1footfmt\endcsname=\normalfootfmt
2823   \expandafter\let\csname #1footgroup\endcsname=\normalfootgroup
2824   \expandafter\let\csname #1footnoterule\endcsname=%
2825                                   \normalfootnoterule
2826   \count\csname #1footins\endcsname=1000
2827   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2828   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2829   \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2830   %

```

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.

```

2831   \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
2832   side only
2833   %

```

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).

```

2833   \ifnoledgroup@else%
2834     \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2835     \expandafter\let\csname mp#1footgroup\endcsname=\mpnormalfootgroup
2836     \count\csname mp#1footins\endcsname=1000
2837     \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2838     \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2839     \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2840   \fi
2841 }
2842
2843 %

```

`\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.

```

2844 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnote}[2]{%
2845   \iftoggle{Xgroupbyline@#1}{%In the case we use \Xgroupbyline, the
insertion is done later, in \add@Xgroupbyline.
2846   \prepare@Xgroupbyline{#1}{#2}{\normalvfootnote@inserted}%
2847 }{%In the case we don't use \Xgroupbyline, the insertion is made directly
2848   \X@beforeinsertion{#1}%
2849   \insert\csname #1footins\endcsname{%
2850     \X@atbegininsertion{#1}%
2851     \normalvfootnote@inserted{#1}{#2}%
2852   }%
2853 }%
2854 }%
2855 %

```

\normalvfootnote@inserted The `\normalvfootnote@inserted` macro is expanded to the content to be add to a `\insert` for normal critical footnote.

```

2856 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnote@inserted}[2]{%
2857   \nottoggle{Xgroupbyline@#1}{\noindent}{\csuse{Xhooknote@#1}%
2858   \csuse{Xnotefontsize@#1}%
2859   \footssplitskips
2860   \ifl@dpairing\ifl@dpaging\else%
2861     \setXnoteswidthliketwocolumns@{#1}%
2862   \fi\fi%
2863   \setXnotespositionliketwocolumns@{#1}%
2864   \spaceskip=\z@skip \xspaceskip=\z@skip%
2865   \csname #1footfmt\endcsname #2{#1}%
2866 }%
2867 %

```

```

\X@beforeinsertion 2868 \newcommand{\X@beforeinsertion}[1]{%
2869   \if@ledgroup\else%
2870     \insert@Xtxtbeforenotes{#1}%
2871   \fi%
2872   \csuse{Xbeforeinserting@#1}%
2873 }%
2874 %

```

```

\beforeinsertion@X 2875 \newcommand{\beforeinsertion@X}[1]{%
2876   \if@ledgroup\else%
2877     \insert@txtbeforenotesX{#1}%
2878   \fi%
2879   \csuse{beforeinsertingX@#1}%
2880 }%
2881 %

```

```

\X@atbegininsertion82 \newcommand{\X@atbegininsertion}[1]{%
2883   \hsize=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
2884 }%
2885 %

```

And somewhat different versions of `\normalvfootnote` and `\normalvfootnote@inserted` for minipages.

```

\mpnormalvfootnote86 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\mpnormalvfootnote}[2]{%
2887   \iftoggle{Xgroupbyline@#1}{%
2888     \prepare@Xgroupbyline{#1}{#2}{\mpnormalvfootnote@inserted}%
2889   }%
2890   {%
2891     \global\setbox\@nameuse{mp#1footins}%
2892     \vbox{%
2893       \unvbox\@nameuse{mp#1footins}%
2894       \mpnormalvfootnote@inserted{#1}{#2}%
2895     }%
2896   }%
2897 }%
2898 %
2899 %

```

```

\mpnormalvfootnote@inserted90 \newcommand{\mpnormalvfootnote@inserted}[2]{%
2901   \noindent\csuse{Xbhooknote@#1}%
2902   \csuse{Xnotefontsize@#1}%
2903   \hsize\columnwidth%
2904   \@parboxrestore%
2905   \color@begingroup%
2906   \csname #1footfmt\endcsname #2{#1}\color@endgroup%
2907 }%
2908 %

```

`\normalfootfmt` `\normalfootfmt` is a ‘normal’ macro to take the footnote line and page number information (see V.9 p. 100), 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.

```

2909 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmt}[4]{%
2910   \Xstorelineinfo{#1}{#4}%
2911   \nottoggle{Xgroupbyline@#4}{\Xledsetnormalparstuff{#4}}{}%
2912   \hangindent=\csuse{Xhangindent@#4}%
2913   \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2914   \nottoggle{Xgroupbyline@#4}{\rule{z@\splittopskip}}{}%
2915   {\printlinefootnote{#1}{#4}}%
2916   \print@lemma{#1}{#2}{#4}%

```

```

2917 \csuse{Xwrapcontent@#4}{#3}%
2918 \nottoggle{Xgroupbyline@#4}{\strut\par}{}%
2919 }%
2920 %

```

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

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

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

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

```

2921 \newcommand*{\normalfootstart}[1]{%
2922 %

```

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

```

2923 \ifdimequal{0pt}{\Xprenotes@}{}%
2924   {%
2925     \iftoggle{Xprenotes@}{%
2926       \togglefalse{Xprenotes@}%
2927       \skip\csname #1footins\endcsname=%
2928       \glueexpr\csuse{Xprenotes@}+\csuse{Xafterrule@#1}\relax%
2929     }%
2930   }%
2931 }%
2932 \vskip\skip\csname #1footins\endcsname%
2933 %

```

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. 223 for the general description of the problem.

```

2934 \leftskip0pt \rightskip0pt
2935 \ifl@dpairing\else%
2936   \hsize=\old@hsize%
2937 \fi%

```

```

2938 \setXnoteswidthliketwocolumns@{#1}%
2939 \setXnotespositionliketwocolumns@{#1}%
2940 %

```

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

```

2941 \print@Xfootnoterule{#1}%
2942 }%
2943 %

```

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

```

2944 \newcommand*{\normalfootgroup}[1]{%
2945 \csuse{Xhookgroup@#1}%
2946 \unvbox\csname #1footins\endcsname%
2947 \hsize=\old@hsize%
2948 }%
2949
2950 %

```

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

```

2951 \unless\ifnoledgroup@
2952 \newcommand*{\mpnormalfootgroup}[1]{%
2953 \vskip\skip\@nameuse{mp#1footins}%
2954 \ifl@dpairing\ifparledgroup%
2955 \leavevmode\marks\parledgroup@{begin}%
2956 \marks\parledgroup@series{#1}%
2957 \marks\parledgroup@type{Xfootnote}%
2958 \fi\fi\normalcolor%
2959 \ifparledgroup%
2960 \ifl@dpairing%
2961 \else%
2962 \setXnoteswidthliketwocolumns@{#1}%
2963 \setXnotespositionliketwocolumns@{#1}%
2964 \print@Xfootnoterule{#1}%%
2965 \fi%
2966 \else%
2967 \setXnoteswidthliketwocolumns@{#1}%
2968 \setXnotespositionliketwocolumns@{#1}%
2969 \print@Xfootnoterule{#1}%%
2970 \fi%
2971 \setlength{\parindent}{0pt}%
2972 \csuse{Xhookgroup@#1}%
2973 \unvbox\csname mp#1footins\endcsname}}
2974 \fi
2975 %

```


XII.6.3 Paraphrased footnotes

The paraphrased-footnote option reformats all the footnotes of one series for a page into a single paragraph; this is especially appropriate when the notes are numerous and brief. The code is based on *The TeXbook*, pp. 398–400, with alterations for our environment. This algorithm uses a considerable amount of save-stack space: a T_EX of ordinary size may not be able to handle more than about 100 notes of this kind on a page.

`\Xarrangement@paragraph` The `\Xarrangement@paragraph` macro sets up everything for one series of the footnotes so that they will be paraphrased; it takes the series letter as argument. We include the setting of `\count\footins` to 1000 for the footnote series just in case user is switching to paraphrased 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 paraphrased.

```

2976 \newcommand*{\Xarrangement@paragraph}[1]{%
2977   \csgdef{series@display#1}{paragraph}
2978   \expandafter\let\csname #1footstart\endcsname=\parafootstart
2979   \expandafter\let\csname v#1footnote\endcsname=\paravfootnote
2980   \expandafter\let\csname #1footfmt\endcsname=\parafootfmt
2981   \expandafter\let\csname #1footgroup\endcsname=\parafootgroup
2982   \count\csname #1footins\endcsname=1000
2983   \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
side only
2984   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2985   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2986   \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2987   \para@footsetup{#1}
2988   %

```

And the extra setup for minipages.

```

2989 \ifnoledgroup@else
2990   \expandafter\let\csname mpv#1footnote\endcsname=\mpparavfootnote
2991   \expandafter\let\csname mp#1footgroup\endcsname=\mpparafootgroup
2992   \count\csname mp#1footins\endcsname=1000
2993   \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2994   \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2995   \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2996 \fi
2997 }
2998 %

```

`\footfudgefiddle` For paraphrased footnotes T_EX 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.

```

2999 \providecommand{\footfudgefiddle}{64}
3000 %

```

`\para@footsetup` `\footparagraph` calls the `\para@footsetup` macro to calculate a special fudge factor, which is the ratio of the `\baselineskip` to the `\hsize`. We assume that the proper value of `\baselineskip` for the footnotes (normally 9pt) has been set already. The argument of the macro is again the note series letter.

Peter Wilson thinks that `\columnwidth` should be used here for \TeX not `\hsize`. Peter Wilson have also included `\footfudgefiddle`.

```

3001 \newcommand*{\para@footsetup}[1]{\csuse{Xhookgroup@#1}\csuse{
Xnotefontsize@#1}
3002   \setXnoteswidthliketwocolumns@{#1}%
3003   \ifcempty{Xwidth@#1}%
3004     {}%
3005     {\columnwidth=\expandafter\dimexpr\csuse{Xwidth@#1}\relax}%
3006   \dimen0=\baselineskip
3007   \multiply\dimen0 by 1024
3008   \divide \dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\
relax
3009   \csxdef{#1footfudgefactor}{%
3010     \expandafter\strip@pt\dimen0 }}
3011
3012 %

```

`\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.

```

3013 \newcommand*{\parafootstart}[1]{%
3014   \rightskip=0pt \leftskip=0pt%
3015   \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
3016   \ifdimequal{0pt}{\Xprenotes@}{}%
3017     {%
3018       \iftoggle{Xprenotes@}{%
3019         \togglefalse{Xprenotes@}%
3020         \skip\csname #1footins\endcsname=%
3021         \glueexpr\csuse{Xprenotes@}+\csuse{Xafterrule@#1}\relax%
3022       }%
3023     }%
3024   }%
3025   \vskip\skip\csname #1footins\endcsname%
3026   \setXnoteswidthliketwocolumns@{#1}%
3027   \setXnotespositionliketwocolumns@{#1}%
3028   \print@Xfootnoterule{#1}%
3029   \let\bidirectional@everypar@empty%
3030   \noindent\leavevmode}
3031 %

```

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

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

However, Michael Downes has pointed out that because text in `hboxes` gets typeset in restricted horizontal mode, there are some undesirable side-effects if you later want to break such text across lines. In restricted horizontal mode, where \TeX does not expect to have to break lines, it does not insert certain items like `\discretionary`s. If you later unbox these `hboxes` and stick them together, as the *TeXbook* macros do to make these footnotes, you lose the ability to hyphenate after an explicit hyphen. This can lead to overfull `hboxes` when you would not expect to find them, and to the uninitiated it might be very hard to see why the problem had arisen.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. 175 above). We need to do this, since `\footfudgefactor` is calculated on the assumption that the

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

³⁰See *The TeXbook*, p. 455 (editions after January 1990).

³¹Wayne 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.

```

3032 \newcommand*{\paravfootnote}[2]{%
3033   \csuse{Xbeforeinserting@#1}%
3034   \insert\csname #1footins\endcsname
3035   \bgroup
3036     \csuse{Xnotefontsize@#1}
3037     \footsplitskips
3038     \setbox0=\vbox{\hsize=\maxdimen%
3039       \let\bidir@RTL@everypar\@empty%
3040       \insert@Xtxtbeforenotes{#1}%
3041       \noindent\csuse{Xhooknote@#1}%
3042       \csname #1footfmt\endcsname #2{#1}}%
3043     \setbox0=\hbox{\Xunvvh{0}{#1}}%
3044     \dp0=0pt
3045     \ht0=\csname #1footfudgefactor\endcsname\wd0
3046   %

```

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

```

3047   \if@RTL\noindent \leavevmode\fi\box0%
3048   \penalty0
3049   \egroup}
3050
3051 %

```

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.

```

3052 \newcommand*{\mpparavfootnote}[2]{%
3053   \global\setbox\@nameuse{mp#1footins}\vbox{%
3054     \unvbox\@nameuse{mp#1footins}%
3055     \csuse{Xnotefontsize@#1}
3056     \footsplitskips
3057     \setbox0=\vbox{\hsize=\maxdimen%
3058       \let\bidir@RTL@everypar\@empty%
3059       \insert@Xtxtbeforenotes{#1}%
3060       \noindent\color@begingroup%

```

```

3061 \csuse{Xhooknote@#1}%
3062 \csname #1footfmt\endcsname #2{#1}\color@endgroup}%
3063 \setbox0=\hbox{\Xunvxh{0}{#1}}%
3064 \dp0=\z@
3065 \ht0=\csname #1footfudgefactor\endcsname\wd0
3066 \box0
3067 \penalty0
3068 }}
3069
3070 %

```

\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`.

```

3071 \newcommand*{\Xunvxh}[2]{%
3072 \setbox0=\vbox{\unvbox#1%
3073 \global\setbox1=\lastbox}%
3074 \unhbox1
3075 \unskip % remove \rightskip,
3076 \unskip % remove \parfillskip,
3077 \unpenalty % remove \penalty of 10000,
3078 \hskip\csuse{Xafternote@#2}\relax}% add the glue to go between the notes
3079
3080 %

```

\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).

```

3081 \newcommand*{\parafootfmt}[4]{%
3082 \Xstorelineinfo{#1}{#4}%
3083 \Xinsertparafootsep{#4}%
3084 \ledsetnormalparstuff@common%
3085 \printlinefootnote{#1}{#4}%
3086 \print@lemma{#1}{#2}{#4}%
3087 \csuse{Xwrapcontent@#4}{#3}%
3088 \penalty-10 }
3089 %

```

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.

```

3090 \newcommand*\parafootgroup}[1]{%
3091   \hsize=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
3092   \unvbox\csname #1footins\endcsname
3093   \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
3094   \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
3095   \makehboxofhboxes
3096   \setbox0=\hbox{\unhbox0 \removehboxes}%
3097   \csuse{Xbhookgroup@#1}%
3098   \csuse{Xnotefontsize@#1}%
3099   \unhbox0\par%
3100   \global\hsize=\old@hsize%
3101 }%
3102
3103 %

```

`\mpparafootgroup` The minipage version.

```

3104 \newcommand*\mpparafootgroup}[1]{%
3105   \setXnoteswidthliketwocolumns@{#1}%
3106   \vskip\skip\@nameuse{mp#1footins}
3107   \ifl@dpairing\ifparledgroup%
3108     \leavevmode\marks\parledgroup@{begin}%
3109     \marks\parledgroup@series{#1}%
3110     \marks\parledgroup@type{Xfootnote}%
3111   \fi\fi\normalcolor
3112   \ifparledgroup%
3113     \ifl@dpairing%
3114     \else%
3115       \setXnoteswidthliketwocolumns@{#1}%
3116       \setXnotespositionliketwocolumns@{#1}%
3117       \print@Xfootnoterule{#1}%
3118     \fi%
3119   \else%
3120     \setXnoteswidthliketwocolumns@{#1}%
3121     \setXnotespositionliketwocolumns@{#1}%
3122     \print@Xfootnoterule{#1}%
3123   \fi%
3124   \unvbox\csname mp#1footins\endcsname
3125   \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
3126   \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
3127   \makehboxofhboxes
3128   \setbox0=\hbox{\unhbox0 \removehboxes}%
3129   \csuse{Xbhookgroup@#1}%
3130   \csuse{Xnotefontsize@#1}%

```

```

3131 \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
3132 \unhbox0\par}}
3133
3134 %

```

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

```

\makehboxofhboxes 35 \newcommand*\makehboxofhboxes{\setbox0=\hbox{}}%
\removehboxes 36 \loop
3137 \unpenalty
3138 \setbox2=\lastbox
3139 \ifhbox2
3140 \setbox0=\hbox{\box2\unhbox0}%
3141 \repeat}
3142
3143 \newcommand*\removehboxes{\setbox0=\lastbox
3144 \ifhbox0{\removehboxes}\unhbox0 \fi}
3145
3146 %

```

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

```

\prevpage@num 47 \newcommand{\Xinsertparafootsep}[1]{%
\Xinsertparafootsep 48 \iflabeledRcol{%
3149 \ifnumequal{\csuse{#1prevpage@numR}}{\page@numR}%
3150 {\ifcsdef{prevline#1}% Be sur \prevline#1 exists.
3151 {\ifcsequal{prevline#1}{lineinfo@}%
3152 {\ifcseempty{Xsymlinenum@#1}{\csuse{Xparafootsep@#1}}{}}%
3153 {\csuse{Xparafootsep@#1}}}%
3154 }%
3155 {\csuse{Xparafootsep@#1}}%
3156 }%
3157 {}%
3158 \global\csname #1prevpage@numR\endcsname=\page@numR%
3159 \else%
3160 \ifnumequal{\csuse{#1prevpage@num}}{\page@num}%
3161 {\ifcsdef{prevline#1}% Be sur \prevline#1 exists.
3162 {\ifcsequal{prevline#1}{lineinfo@}%
3163 {\ifcseempty{Xsymlinenum@#1}{\csuse{Xparafootsep@#1}}{}}%
3164 {\csuse{Xparafootsep@#1}}}%
3165 }%
3166 {\csuse{Xparafootsep@#1}}%
3167 }%
3168 {}%
3169 \global\csname #1prevpage@num\endcsname=\page@num%
3170 \fi%

```

```

3171 }
3172 %

```

XII.6.4 Columnar footnotes

Common tools

`\rigidbalance` We will now define macros for three-column notes and two-column notes. Both sets of macros will use `\rigidbalance`, which splits a box (#1) into a number (#2) of columns, each with a space (#3) between the top baseline and the top of the `\vbox`. The `\Xrigidbalance` macro is taken from *The TeXbook*, p. 397, with a slight change to the syntax of the arguments so that they do not depend on white space. Note also the extra unboxing in `\splitoff`, which allows the new `\vbox` to have its natural height as it goes into the alignment.

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

We do not call directly `\rigidbalance`, but we call `\Xrigidbalance` for critical notes and `\rigidbalanceX` for familiar notes. Both of them call `\rigidbalance`.

```

3173 \newcount\@k \newdimen\@h
3174 \newcommand*\Xrigidbalance}[3]{%
3175     \hsize=\expandafter\dimexpr\csuse{Xwidth@\@currentseries}\relax%
3176     \rigidbalance{#1}{#2}{#3}%
3177 }%
3178
3179 \newcommand*\rigidbalanceX}[3]{%
3180     \hsize=\expandafter\dimexpr\csuse{widthX@\@currentseries}\relax%
3181     \rigidbalance{#1}{#2}{#3}%
3182 }%
3183
3184 \newcommand*\rigidbalance}[3]{%
3185     \setbox0=\box#1 \@k=#2 \@h=#3%
3186     \@@line{\splittopskip=\@h \vbadness=\@M \hfilneg
3187       \valign{##\vfil\cr\dosplits}}}%
3188
3189 \newcommand*\dosplits{\ifnum\@k>0 \noalign{\hfil}\splitoff
3190   \global\advance\@k-1\cr\dosplits\fi}
3191
3192 \newcommand*\splitoff{\dimen0=\ht0
3193   \divide\dimen0 by\@k \advance\dimen0 by\@h
3194   \setbox2 \vsplit0 to \dimen0
3195   \unvbox2 }
3196
3197 %

```

Three columns


```

\Xarrangement@threecol  \newcommand*{\Xarrangement@threecol}[1]{%
3199 \csgdef{series@display#1}{threecol}
3200 \expandafter\let\csname v#1footnote\endcsname=\threecolvfootnote
3201 \expandafter\let\csname #1footfmt\endcsname=\threecolfootfmt
3202 \expandafter\let\csname #1footgroup\endcsname=\threecolfootgroup
3203 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
3204 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
3205 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
3206 \threecolfootsetup{#1}
3207 %

```

The additional setup for minipages.

```

3208 \ifnoledgroup@else
3209 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
3210 \expandafter\let\csname mp#1footgroup\endcsname=\mpthreecolfootgroup
3211 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
3212 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
3213 \mpthreecolfootsetup{#1}
3214 \fi
3215 }
3216 %
3217 %

```

The `\footstart` and `\footnoterule` macros for these notes assume the normal values (XII.6.2 p. 175 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.

```

3218 \newcommand*{\threecolfootsetup}[1]{%
3219 \count\csname #1footins\endcsname 333
3220 \csxdef{default@#1footins}{333}%Use this to confine the notes to one
side only
3221 \multiply\dimen\csname #1footins\endcsname \thr@@}
3222 %

```

`\mpthreecolfootsetup` The setup for minipages.

```

3223 \newcommand*\mpthreecolfootsetup}[1]{%
3224   \count\csname mp#1footins\endcsname 333
3225   \multiply\dimen\csname mp#1footins\endcsname \thr@@}
3226
3227 %

```

\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@<s> ensures that the \splittopskip and \splitmaxdepth take their values from the right \strutbox: the one used in a footnotes. Note especially the importance of temporarily reducing the \hsize to 0.3 of its normal value. This determines the widths of the individual columns. So if the normal \hsize is (say) 10 cm, then each column will be $0.3 \times 10 = 3$ cm wide, leaving a gap of 1 cm spread equally between columns (i.e., .5 cm between each).

The arguments are #1 the note series letter and #1 the full text of the note (including numbers, lemma and text).

```

3228 \notbool{parapparatus@}\newcommand*\newcommand*\threecolvfootnote}[2]{%
3229   \iftoggle{Xgroupbyline@#1}{%
3230     \prepare@Xgroupbyline{#1}{#2}\threecolvfootnote@inserted}%
3231   }%
3232   {%
3233     \X@beforeinsertion{#1}%
3234     \insert\csname #1footins\endcsname{%
3235       \threecolvfootnote@inserted{#1}{#2}%
3236     }%
3237   }%
3238 }%
3239 %

```

```

\threecolvfootnote@inserted 3240 \notbool{parapparatus@}\newcommand*\newcommand*\threecolvfootnote@inserted}[2]{%
3241   \hsize=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
3242   \noindent\csuse{Xhooknote@#1}%
3243   \csuse{Xnotefontsize@#1}%
3244   \footsplitskip%
3245   \csname #1footfmt\endcsname #2{#1}%
3246 }%
3247 %

```

\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.

```

3248 \notbool{parapparatus@}\newcommand*\newcommand*\threecolfootfmt}[4]{%
3249   \Xstorelineinfo{#1}{#4}%
3250   \threecol@begin@insert{#4}%
3251   \hspace{\parindent}%

```

```

3252 \printlinefootnote{#1}{#4}%
3253 \print@lemma{#1}{#2}{#4}%
3254 \csuse{Xwrapcontent@#4}{#3}%
3255 \nottoggle{Xgroupbyline@#4}%
3256   {\strut\par\allowbreak}%
3257   {}%
3258 }%
3259 %

```

\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`.

```

3260 \newcommand{\threecol@begin@insert}[1]{%
3261   \normal@pars%
3262   \nottoggle{Xgroupbyline@#1}%
3263   {\hspace \csuse{Xhsizethreecol@#1}}%
3264   {}%
3265   \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
3266   \tolerance=5000%
3267   \hangindent=\csuse{Xhangindent@#1}%
3268   \everypar{\hangindent=\csuse{Xhangindent@#1}}%
3269   \@tempdima=\parindent%
3270   \csuse{Xcolalign@#1}%
3271   \parindent=\@tempdima%
3272   \strut%
3273 }%
3274 %

```

\threecolfootgroup And here is the `footgroup` macro that is called within the output routine to regroup the notes into three columns. Once again, the call to `\Xnotefontsize@{s}` is there to ensure that it is the right `\splittopskip`—the one used in footnotes—which is used to provide the third argument for `\rigidbalance`. This third argument (`\@h`) is the `topskip` for the box containing the text of the footnotes, and does the job of making sure the top lines of the columns line up horizontally. In *The TeXbook*, p. 398, Donald Knuth suggests retrieving the output of `\rigidbalance`, putting it back into the insertion box, and then printing the box. Here, we just print the `\line` which comes out of `\rigidbalance` directly, without any re-boxing.

```

3275 \newcommand*{\threecolfootgroup}[1]{%
3276   \csuse{Xhookgroup@#1}\par%
3277   \splittopskip=\ht\strutbox
3278   \expandafter
3279   \Xrigidbalance\csname #1footins\endcsname \thr@@ \splittopskip}
3280 %

```

\mpthreecolfootgroup The setup for minipages.

```

3281 \newcommand*{\mpthreecolfootgroup}[1]{%

```

```

3282 \vskip\skip\@nameuse{mp#1footins}
3283 \ifl@dpairing\ifparledgroup%
3284   \leavevmode\marks\parledgroup@{begin}%
3285   \marks\parledgroup@series{#1}%
3286   \marks\parledgroup@type{Xfootnote}%
3287 \fi\fi\normalcolor
3288 \ifparledgroup%
3289   \ifl@dpairing%
3290   \else%
3291     \setXnoteswidthliketwocolumns@{#1}%
3292     \setXnotespositionliketwocolumns@{#1}%
3293     \print@Xfootnoterule{#1}%
3294   \fi%
3295 \else%
3296   \setXnoteswidthliketwocolumns@{#1}%
3297   \setXnotespositionliketwocolumns@{#1}%
3298   \print@Xfootnoterule{#1}%
3299 \fi%
3300 \csuse{Xbhookgroup@#1}\par%
3301 \splittopskip=\ht\strutbox
3302 \expandafter
3303 \Xrigidbalance\csname mp#1footins\endcsname \thr@@ \splittopskip}}
3304
3305 %

```

Two columns

```

\Xarrangement@twocol 3306 \newcommand*\Xarrangement@twocol}[1]{%
3307   \csgdef{series@display#1}{twocol}
3308   \expandafter\let\csname v#1footnote\endcsname=\twocolvfootnote
3309   \expandafter\let\csname #1footfmt\endcsname=\twocolfootfmt
3310   \expandafter\let\csname #1footgroup\endcsname=\twocolfootgroup
3311   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
3312   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
3313   \advance\skip\csname #1footins\endcsname by\csuse{Xafterterrule@#1}%
3314   \twocolfootsetup{#1}
3315 %

```

The additional setup for minipages.

```

3316 \ifnoledgroup@else
3317   \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
3318   \expandafter\let\csname mp#1footgroup\endcsname=\mptwocolfootgroup
3319   \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
3320   \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterterrule@#1}%
3321   \mptwocolfootsetup{#1}
3322 \fi
3323 }
3324
3325 %

```

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

```

\twocolvfootnote@inserted
\twocolfootfmt
\twocolfootgroup
3326 \newcommand*{\twocolfootsetup}[1]{%
3327   \count\csname #1footins\endcsname 500
3328   \csxdef{default@#1footins}{500}%
3329   \multiply\dimen\csname #1footins\endcsname \tw@}
3330   %

3331 \notbool{parapparatus@}{\newcommand*{\newcommand}{\twocolvfootnote}[2]{%
3332   \iftoggle{Xgroupbyline@#1}{%
3333     \prepare@Xgroupbyline{#1}{#2}{\twocolvfootnote@inserted}%
3334   }{%
3335     \X@beforeinsertion{#1}%
3336     \insert\csname #1footins\endcsname{%
3337       \twocolvfootnote@inserted{#1}{#2}%
3338     }%
3339   }%
3340 }%

3341 \notbool{parapparatus@}{\newcommand*{\newcommand}{\twocolvfootnote@inserted}[2]{%
3342   \hsiz=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
3343   \noindent\csuse{Xhooknote@#1}%
3344   \csuse{Xnotefontsize@#1}%
3345   \footsplitskips%
3346   \csname #1footfmt\endcsname #2{#1}%
3347 }%
3348   %

3349 \notbool{parapparatus@}{\newcommand*{\newcommand}{\twocolfootfmt}[4]{% 4th
   arg is optional, for backward compatibility
3350   \Xstorelineinfo{#1}{#4}%
3351   \twocol@begin@insert{#4}%
3352   \hspace{\parindent}%
3353   \printlinefootnote{#1}{#4}%
3354   \print@lemma{#1}{#2}{#4}%
3355   \csuse{Xwrapcontent@#4}{#3}%
3356   \nottoggle{Xgroupbyline@#4}%
3357     {\strut\par\allowbreak}%
3358   {}%
3359 }%
3360   %

3361 \newcommand{\twocol@begin@insert}[1]{%
3362   \normal@pars%
3363   \hsiz \csuse{Xsizetwocol@#1}%
3364   \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
3365   \tolerance=5000%
3366   \hangindent=\csuse{Xhangindent@#1}%

```

```

3367 \everypar{\hangindent=\csuse{Xhangindent@#1}}%
3368 \@tempdima=\parindent%
3369 \csuse{Xcolalign@#1}%
3370 \parindent=\@tempdima%
3371 \strut%
3372 }%
3373
3374 \newcommand*{\twocolfootgroup}[1]{%
3375   \csuse{Xbhookgroup@#1}\par%
3376   \splittopskip=\ht\strutbox
3377   \expandafter
3378   \Xrigidbalance\csname #1footins\endcsname \tw@ \splittopskip}
3379
3380 %

```

`\mptwocolfootsetup` The versions for minipages.

`\mptwocolfootgroup`

```

3381 \newcommand*{\mptwocolfootgroup}[1]{%
3382   \count\csname mp#1footins\endcsname 500
3383   \multiply\dimen\csname mp#1footins\endcsname \tw@}
3384 %
3385
3386 \newcommand*{\mptwocolfootgroup}[1]{%
3387   \vskip\skip\@nameuse{mp#1footins}
3388   \ifl@dpairing\ifparledgroup%
3389     \leavevmode\marks\parledgroup@{begin}%
3390     \marks\parledgroup@series{#1}%
3391     \marks\parledgroup@type{Xfootnote}%
3392   \fi\fi\normalcolor
3393   \ifparledgroup%
3394     \ifl@dpairing%
3395       \else%
3396         \setXnoteswidthliketwocolumns@{#1}%
3397         \setXnotespositionliketwocolumns@{#1}%
3398         \print@Xfootnoterule{#1}%
3399       \fi%
3400     \else%
3401       \setXnoteswidthliketwocolumns@{#1}%
3402       \setXnotespositionliketwocolumns@{#1}%
3403       \print@Xfootnoterule{#1}%
3404     \fi%
3405     \csuse{Xbhookgroup@#1}\par%
3406     \splittopskip=\ht\strutbox
3407     \expandafter
3408     \Xrigidbalance\csname mp#1footins\endcsname \tw@ \splittopskip}}
3409 %

```

XII.7 Critical notes presentation

Here, we define some commons macro which are used in order to print a critical notes, that is a note with 1) line number 2) lemma 3) lemma separator 4) text associated to the lemma.

XII.7.1 Font tools

`\endashchar` The fonts that are used for printing notes might not have the character mapping we expect: for example, the Computer Modern font that contains old-style numerals does not contain an en-dash or square brackets, and its period and comma are in odd locations. To allow use of the standard footnote macros with such fonts, we use the following macros for certain characters.

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

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 `\Xlinrangeseparator`, `\Xendlinrangeseparator`, `\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.

```

3410 \def\endashchar{\textnormal{--}}
3411
3412 \newcommand*{\fullstop}{\textnormal{.}}
3413 \def\Xsublinesep@side{\fullstop}
3414
3415 \newcommand*{\rbracket}{\textnormal{%
3416   \csuse{text\csuse{footnote@lang}}{%
3417     \ifluatex%
3418       \ifdefstring{\footnote@luatextextdir}{TRT}{\thinspace[]}{\thinspace
3419     ]}%
3419     \else%
3420     \thinspace}%
3421     \fi}%
3422   }%
3423 }
3424
3425 %

```

XII.7.2 Pstart number in footnote

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

```

3426 \newcommand{\printpstart}[0]{%

```

```

3427 \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
3428 \ifledRcol%
3429 \thepstartR%
3430 \else%
3431 \thepstartL%
3432 \fi%
3433 }{%
3434 \thepstart%
3435 }%
3436 }
3437 %

```

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).

```

3438 %
3439 \newcommand{\print@lemma}[3]{%
3440 \bgroup%
3441 \nottoggle{Xlemmadisablefontselection@#3}%
3442 {\select@lemmafnt#1|}%
3443 }%
3444 \bgroup%
3445 \csuse{Xlemmafnt@#3}%Deprecated
3446 \csuse{Xwraplemma@#3}{#2}%
3447 \egroup%
3448 \egroup%
3449 \iftoggle{nosep@}{%
3450 \hskip\csuse{Xinplaceoflemmaseparator@#3}%
3451 \relax%
3452 }%
3453 {\ifcempty{Xlemmaseparator@#3}%
3454 {%
3455 \hskip\csuse{Xinplaceoflemmaseparator@#3}%
3456 \relax%
3457 }%
3458 {%
3459 \nobreak%
3460 \hskip\csuse{Xbeforelemmaseparator@#3}%
3461 \csuse{Xlemmaseparator@#3}%
3462 \hskip\csuse{Xafterlemmaseparator@#3}%
3463 \relax%
3464 }%
3465 }%
3466 }%
3467 %

```


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.

```

3468 \newcommand{\Xstorelineinfo}[2]{%
3469   \l@dp@rsefootspec#1|%
3470   \iftoggle{Xnumberonlyfirstintwolines@#2}{%
3471     \xdef\lineinfo@{\l@dparsedstartline - \l@dparsedstartsub - \
l@dparsedendline - \l@dparsedendsub}%
3472   }%
3473   {%
3474     \xdef\lineinfo@{\l@dparsedstartline - \l@dparsedstartsub}%
3475   }%
3476 }%
3477 %

```

\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`.

```

3478 \newcommand{\printlinefootnote}[2]{%
3479   \iftoggle{nonum@}{%Try if the line number must printed for this specific
not (by default, yes)
3480     \hspace{\csuse{Xinplaceofnumber@#2}}%
3481   }%
3482   {%
3483     {%
3484       \iftoggle{Xnonumber@#2}%Try if the line number must printed (by
default, yes)
3485       {%
3486         \hspace{\csuse{Xinplaceofnumber@#2}}%
3487       }%
3488       {%
3489         {\iftoggle{Xnumberonlyfirstinline@#2}% If for this series the
line number must be printed only in the first time.
3490           {%
3491             \ifcsdef{prevline#2}%
3492             {%Be sure the \prevline exists.
3493             \ifcsequal{prevline#2}{\lineinfo@}%Try it
3494             {%
3495             \ifcsempy{Xsymlinenum@#2}%
3496             {%
3497               \hspace{\csuse{Xinplaceofnumber@#2}}%
3498             }%
3499             {\printsymlinefootnotearea{#2}}%
3500           }%

```

```

3501         {%
3502         \printlinefootnotearea{#1}{#2}%
3503         }%
3504     }%
3505     {%
3506     \printlinefootnotearea{#1}{#2}%
3507     }%
3508 }%
3509 {%
3510 \printlinefootnotearea{#1}{#2}%
3511 }%
3512 \csxdef{prevline#2}{\lineinfo@}%
3513 }%
3514 }%
3515 }%
3516 }%
3517 }
3518 %

```

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

```

3519 \newcommand{\printsymlinefootnotearea}[1]{%
3520 \hspace{\csuse{Xbeforesymlinewidth@#1}}%
3521 \csuse{Xnotenumfont@#1}%
3522 \ifdimequal{\csuse{Xboxsymlinewidth@#1}}{\z@}%
3523 {\csuse{Xsymlinewidth@#1}}%
3524 {\hbox to \csuse{Xboxsymlinewidth@#1}%
3525 {\csuse{Xsymlinewidth@#1}\hfill}%
3526 }%
3527 \hspace{\csuse{Xaftersymlinewidth@#1}}%
3528 }%
3529 %

```

\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.)

```

3530 \newcommand{\printlinefootnotearea}[2]{%
3531 \printXbeforenumber{#2}%
3532 \csuse{Xnotenumfont@#2}%
3533 \boxfootnotenumbers{#1}{#2}%
3534 \printXafternumber{#2}%
3535 }%
3536 %

```

\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.

```

3537 \newcommand{\boxfootnotenumbers}[2]{%
3538   \ifdimequal{\csuse{Xboxlinenum@#2}}{0pt}{%
3539     \printlinefootnotenumbers{#1}{#2}%
3540   }%
3541   {%
3542     \hbox to \csuse{Xboxlinenum@#2}%
3543     {%
3544       \IfSubStr{RC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
3545       \printlinefootnotenumbers{#1}{#2}%
3546       \IfSubStr{LC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
3547     }%
3548   }%
3549 }%
3550 %

```

\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.

```

3551 \newcommand{\printlinefootnotenumbers}[2]{%
3552   \xdef\@currentseries{#2}%
3553   \ifboolexpr{%
3554     (togl{Xpstart@#2} and bool{numberpstart})%
3555     or togl{Xpstarteverytime@#2}}%
3556   {\printpstart}{}%
3557   \iftoggle{Xstanza@#2}{%
3558     \ifnumberstanza%
3559     \printstanza%
3560     \csuse{Xstanzaseparator@#2}%
3561   }%
3562   \iftoggle{Xonlypstart@#2}{%
3563     \csuse{Xtxtbeforenumber@#2}%
3564     \printlines#1|\ifledRcol@{\@Rlineflag\fi}|}%
3565   }%
3566 }%
3567 %

```

\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.)

```

3568 \newcommand{\printXbeforenumber}[1]{%
3569   \hspace{\csuse{Xbeforenumber@#1}}%
3570 }%
3571 %

```

\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

```

3572 \newcommand{\printXafternumber}[1]{%
3573   \iftoggle{Xnonbreakableafternumber@#1}{\nobreak}{}%

```

```

3574 \hspace{\csuse{Xafternumber@#1}}%
3575 }%
3576 %

```

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. 100: 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 77 \newif\ifl@d@pnum
\ifl@d@ssub 78 \newif\ifl@d@ssub
\ifl@d@elin 79 \newif\ifl@d@elin
\ifl@d@esl 80 \newif\ifl@d@esl
\ifl@d@dash 81 \newif\ifl@d@dash
\ifl@d@Xtwolines 82 \newif\ifl@d@Xtwolines%
\ifl@d@Xmorethantwolines 83 \newif\ifl@d@Xmorethantwolines%
3584 %

```

```

\l@dparsefootspec \l@dparsefootspec{<spec>}{<lemma>}{<text>} parses a footnote specification. <lemma>
\l@dp@rsefootspec and <text> are the lemma and text respectively. <spec> is the line and page num-
\l@dparsedstartpage ber and lemma font specifier in \l@d@nums style format. The real work is done by
\l@dparsedstartline \l@dp@rsefootspec which defines macros holding the numeric values. In many cases,
\l@dparsedstartsub this last command is called directly. Just a reminder of the arguments:
\l@dparsedendpage \printlines #1 | #2 | #3 | #4 | #5 | #6 | #7
\l@dparsedendline \printlines start-page | line | subline | end-page | line | subline | fontflag
\l@dparsedendsub \newcommand*{\l@dparsefootspec}[3]{\l@dp@rsefootspec#1|}
3585
3586 \def\l@dp@rsefootspec#1|#2|#3|#4|#5|#6|#7|{%
3587 \gdef\l@dparsedstartpage{#1}%
3588 \gdef\l@dparsedstartline{#2}%

```

```

3589 \gdef\l@dparsedstartsub{#3}%
3590 \gdef\l@dparsedendpage{#4}%
3591 \gdef\l@dparsedendline{#5}%
3592 \gdef\l@dparsedendsub{#6}%
3593 }
3594 %

```

Initialise the several number value macros.

```

3595 \def\l@dparsedstartpage{0}%
3596 \def\l@dparsedstartline{0}%
3597 \def\l@dparsedstartsub{0}%
3598 \def\l@dparsedendpage{0}%
3599 \def\l@dparsedendline{0}%
3600 \def\l@dparsedendsub{0}%
3601
3602 %

```

\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`.

```

3603 \newcommand*{\setprintlines}[6]{%
3604   \l@d@pnumfalse \l@d@dashfalse
3605 %

```

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

```

3606 \ifbypage@
3607   \ifnum#4=#1 \else
3608     \l@d@pnumtrue
3609     \l@d@dashtrue
3610   \fi
3611 \fi
3612 %

```

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.

```

3613 \ifl@d@pnum \l@d@elintrue \else \l@d@elinfalse \fi
3614 \ifnum#2=#5 \else
3615   \l@d@elintrue
3616   \l@d@dashtrue
3617 \fi
3618 %

```

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

```

3619 \l@d@ssubfalse
3620 \ifnum#3=0 \else
3621   \l@d@ssubtrue
3622 \fi
3623 %

```

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.

```

3624 \l@d@eslfalse
3625 \ifnum#6=0 \else
3626   \ifnum#6=#3
3627     \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
3628   \else
3629     \l@d@esltrue
3630     \l@d@dashtrue
3631   \fi
3632 \fi%
3633 %

```

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

```

3634 \ifl@d@dash%
3635 \ifboolexpr{togl{fulllines@} or test{\ifcempty{Xtwolines@}\@currentseries}}}%
3636 {}%
3637 {}%
3638 \setistwofollowinglines{#1}{#2}{#4}{#5}%
3639 \ifboolexpr{%
3640   (%
3641     togl {Xtwolinesbutnotmore@\@currentseries}%
3642     and not%
3643     (%
3644       bool {istwofollowinglines@}%
3645     )%
3646   )%
3647   or%
3648   (%
3649     (not test{\ifnumequal{#1}{#4}})%
3650     and togl{Xtwolinesonlyinsamepage@\@currentseries}%
3651   )%
3652   }%
3653 {}%
3654 {}%
3655 \l@d@dashfalse%
3656 \l@d@Xtwolinesttrue%
3657 \l@d@elinfalse%
3658 \l@d@eslfalse%
3659 \ifcempty{Xmorethantwolines@\@currentseries}%
3660 {}%
3661 {\ifistwofollowinglines@\else%
3662   \l@d@Xmorethantwolinesttrue%
3663   \fi%
3664 }%
3665 }%
3666 }%

```

```

3667 \fi%
3668 %

```

End of \setprintlines.

```

3669 }%
3670 %

```

`\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.

```

3671 \newif\ifistwofollowinglines@%
3672 \newcommand{\setistwofollowinglines}[4]{%
3673   \ifcsdef{lastlinenumberon@#1}%
3674     {\numdef{\tmp}{\csuse{lastlinenumberon@#1}}}%
3675     {\numdef{\tmp}{0}}}%
3676   \istwofollowinglines@false%
3677   \ifnumequal{#4-#2}{1}%
3678     {\istwofollowinglines@true}%
3679     {\ifbypage@%
3680       \ifnumequal{#3-#1}{1}%
3681       {%
3682         \ifnumequal{#2}{\tmp}%
3683         {\ifnumequal{#4}{1}{\istwofollowinglines@true}{}}%
3684         {}%
3685       }%
3686       {}%
3687     }%
3688   }%
3689 }%
3690 %

```

`\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

```
3691 \def\printlines#1|#2|#3|#4|#5|#6|#7|#8|{%
3692   \begingroup%
3693   %
```

If we use Lua_T_EX, ensure we use good text's direction.

```
3694   \ifluatex%
3695     \edef\@tmp{\the\textdir}%
3696     \ifdefstring{\@tmp}{TLT}{\textdir TLT}%Test in order to prevent
3697     \fi%
3698   %
```

Decide which part of line number components we will print.

```
3699   \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
3700   %
```

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.

```
3701   \ifdimequal{\csuse{Xboxstartlinenum@\@currentseries}}{0pt}%
3702     {\bgroup}%
3703     {\leavevmode\hbox to \csuse{Xboxstartlinenum@\@currentseries}\bgroup\
3704     hfill}%
3705     \ifl@d@pnum%
3706       \wrap@edcrossref{\@this@crossref@start}{#1}%
3707       \csuse{Xpagelinesep@\@currentseries}%
3708       \fi%
3709       \wrap@edcrossref{\@this@crossref@start}{%
3710       \linenumrep{#2}%
3711       \iftoggle{Xlineflag@\@currentseries}{#8}{}%
3712       }%
3713       \ifl@d@ssub%
3714       \csuse{Xsublinesep@\@currentseries}%
3715       \wrap@edcrossref{\@this@crossref@start}{\sublinenumrep{#3}}%
3716       \fi
3717     \egroup%
3718   %
```

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

```
3718   \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
3719     {\bgroup}%
3720     {\hbox to \csuse{Xboxendlinenum@\@currentseries}\bgroup}%
3721     \ifl@d@Xtwolines%
```



```

3722 \ifl@d@Xmorethantwolines%
3723 \csuse{Xmorethantwolines@ \@currentseries}%
3724 \else%
3725 \csuse{Xtwolines@ \@currentseries}%
3726 \fi%
3727 \else%
3728 \ifl@d@dash%
3729 \ifdefined\linangesep%
3730 \linangesep%
3731 \else%
3732 \csuse{Xlinangeseparator@ \@currentseries}%
3733 \fi%
3734 \fi%
3735 \ifl@d@pnum%
3736 \wrap@edcrossref{\@this@crossref@end}{#4}%
3737 \csuse{Xpagelinesep@ \@currentseries}%
3738 \fi%
3739 \ifl@d@elin%
3740 \wrap@edcrossref{\@this@crossref@end}{%
3741 \linenumrep{#5}%
3742 \iftoggle{Xlineflag@ \@currentseries}{#8}{}}%
3743 }%
3744 \fi%
3745 \ifl@d@esl%
3746 \ifl@d@elin%
3747 \csuse{Xsublinesep@ \@currentseries}%
3748 \fi%
3749 \wrap@edcrossref{\@this@crossref@end}{\sublinenumrep{#6}}%
3750 \fi%
3751 \fi%
3752 \ifdimequal{\csuse{Xboxendlinenum@ \@currentseries}}{0pt}%
3753 {}%
3754 {\hfill}%Prevent underfull hbox
3755 \egroup%
3756 \endgroup%
3757 }%
3758 %

```

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`.

```

3759 \newcommand{\prepare@Xgroupbyline}[3]{%
3760 \iftoggle{Xgroupbylineseparetwolines@#1}{%
3761 \l@dparsfootspec#2%
3762 \ifcsdef{#1@forinserting@ \l@dparsedendpage- \l@dparsedendline- \l@dparsedendsub}%
3763 {%
3764 \csgappto%

```

```

3765     {#1@forinserting@\l@dparsedendpage-\l@dparsedendline-\
l@dparsedendsub}%
3766     {%
3767         \ifcempty{Xsymlinenum@#1}%
3768         {\csuse{Xparafootsep@#1}}%
3769         {}%
3770         #3{#1}{#2}%
3771         \hskip\csuse{Xafternote@#1}\relax%
3772     }%
3773 }%
3774 {%
3775     \csdef%
3776     {#1@forinserting@\l@dparsedendpage-\l@dparsedendline-\
l@dparsedendsub}%
3777     {%
3778         #3{#1}{#2}%
3779         \hskip\csuse{Xafternote@#1}\relax%
3780     }%
3781 }%
3782 \listcsxadd{#1@forinserting}{\l@dparsedendpage-\l@dparsedendline-\
l@dparsedendsub}%
3783 }{%
3784     \ifcsdef{#1@forinserting@all}{%
3785         \csgappto%
3786         {#1@forinserting@all}%
3787         {%
3788             \ifcempty{Xsymlinenum@#1}%
3789             {\csuse{Xparafootsep@#1}}%
3790             {}%
3791             #3{#1}{#2}%
3792             \hskip\csuse{Xafternote@#1}\relax%
3793         }%
3794     }%
3795     {%
3796         \csdef%
3797         {#1@forinserting@all}%
3798         {%
3799             #3{#1}{#2}%
3800             \hskip\csuse{Xafternote@#1}\relax%
3801         }%
3802     }%
3803     \listcsgadd{#1@forinserting}{all}%
3804 }%
3805 }%
3806 %

```

XIII Familiar footnotes

XIII.1 Adjacent footnotes

The original edmac provided users with five series of critical footnotes (`\Afootnote` `\Bfootnote` `\Cfootnote` `\Dfootnote` `\Efootnote`), and \LaTeX provides a single numbered footnote. The `reledmac` package uses the edmac mechanism to provide six series of numbered footnotes.

First, though, the `footmisc` package has an option whereby two or more consecutive `\footnotes` have their marks separated by commas. This seemed to Peter Wilson such a useful ability that it was provided automatically by `eledmac`.

Maïeul Rouquette has maintained this feature in `reledmac`, despite he thought that is not directly in relationship with the aim of `reledmac`.

`\multiplefootnotemarker` 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`.

```
3807 \providecommand*\multiplefootnotemarker}{3sp}
3808 \providecommand*\multfootsep}{\textsuperscript{\normalfont,}}
3809
3810 %
```

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

```
3811 \providecommand*\m@mmf@prepare}{%
3812 \kern-\multiplefootnotemarker
3813 \kern\multiplefootnotemarker\relax}
3814 %
```

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

```
3815 \providecommand*\m@mmf@check}{%
3816 \ifdim\lastkern=\multiplefootnotemarker\relax
3817 \edef\@x@sf{\the\spacefactor}%
3818 \unkern
3819 \multfootsep
3820 \spacefactor\@x@sf\relax
3821 \fi}
3822
3823 %
```

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

```
3824 \@ifclassloaded{memoir}{\}%
3825 %
```

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

```
3826 \apptocmd{\@footnotetext}{\m@mmf@prepare}{}{}
3827 %
```

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

```
3828
3829 \patchcmd{\@footnotemark}
3830   {\nobreak}
3831   {\m@mmf@check
3832    \nobreak
3833   }
3834   {}{}
3835 \patchcmd{\@footnotemark}
3836   {\@makefnmark}
3837   {\@makefnmark
3838    \m@mmf@prepare
3839   }
3840   {}{}
3841 %
```

Finished the modifications for the non-memoir case.

```
3842 }
3843
3844 %
```

XIII.2 Regular footnotes for numbered texts

`\l@doldold@footnotetext` In order to enable the regular `\footnotes` in numbered text we have to play around
`\@footnotetext` with its `\@footnotetext`, using different forms for when in numbered or regular text.

```
3845 \pretocmd{\@footnotetext}{%
3846   \ifnumberedpar@
3847     \edtext{}{\l@dbfnote{#1}}}%
3848   \else
3849   }{}{}
3850 \apptocmd{\@footnotetext}{\fi}{}{}%
3851 %
```

`\l@dbfnote` `\l@dbfnote` adds the footnote to the insert list, and `\v1@dbfnote` calls the original
`\v1@dbfnote` `\@footnotetext`. We also patch `\footnote` in order to get the correct footnote
`\v1@dbfnote` numbers when typesetting parallel texts. This is moved into a `\get@fnmark` command.
`\footnote`
`\get@fnmark`
`\get@thisfootnote`

```

3852 \patchcmd%
3853   {\footnote}%
3854   {\stepcounter\@mpfn}%
3855   {%
3856   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
3857   l@dprintingcolumns}}{%
3858     \global\advance\footnote@reading by \@ne%
3859     \get@thisfootnote%
3860     \get@fnmark{\thisc@footnote}%
3861     \ifcsdef{footnotereading\the\footnote@reading=typeset}%
3862     {\setcounter{\@mpfn}{\csuse{footnotereading\the\footnote@reading=
typeset}}}%
3863     {\setcounter{\@mpfn}{\footnote@reading}}%
3864   }{%
3865     \stepcounter\@mpfn%
3866   }%
3867 }%
3868 {}
3869 {}
3870
3871 \newcommand{\get@thisfootnote}{%
3872   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
3873     \protected@xdef\thisc@footnote{\the\footnote@reading}%
3874   }{%
3875     \protected@xdef\thisc@footnote{\the\c@footnote}%
3876   }%
3877 }%
3878
3879 \newcommand{\l@dbfnote}[1]{%
3880   \get@thisfootnote%
3881   \gdef\@tag{#1\relax}%
3882   \ifledRcol%
3883     \xright@appenditem{%
3884       \ifdefined\Hy@footnote@currentHref%
3885         \noexpand\def\noexpand\Hy@footnote@currentHref{\
Hy@footnote@currentHref}%
3886       \fi%
3887       \noexpand\vl@dbfnote{{\expandonce\@tag}}{\thisc@footnote}%
3888     }%
3889     \to\inserts@listR
3890     \global\advance\insert@countR \@ne%
3891   \else%
3892     \xright@appenditem{%
3893       \ifdefined\Hy@footnote@currentHref%
3894         \noexpand\def\noexpand\Hy@footnote@currentHref{\
Hy@footnote@currentHref}%
3895       \fi%
3896       \noexpand\vl@dbfnote{{\expandonce\@tag}}{\thisc@footnote}%

```

```

3897     }%
3898         \to\inserts@list
3899     \global\advance\insert@count \@ne%
3900     \fi
3901     \ignorespaces%
3902 }%
3903
3904 \newcommand{\get@fnmark}[1]{%
3905     \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
3906         l@dprintingcolumns}}}%
3907     {%
3908         \stepcounter{footnote@typeset}%
3909         \setcounter{footnote}{\c@footnote@typeset}%
3910         \immediate\write\@mainaux{%
3911             \csgdef{footnotereading#1=typeset}{\the\c@footnote@typeset}%
3912             }%
3913         \def\@thefnmark{\thefootnote}%
3914     }%
3915     \setcounter{footnote}{#1}%
3916     \def\@thefnmark{\thefootnote}%
3917 }%
3918 }%
3919
3920 \newcommand{\vl@dbfnote}[2]{%
3921     \get@fnmark{#2}%
3922     \@footnotetext{#1}%
3923 }%
3924 %

```

XIII.3 Footnote formats

Some of the code for the various formats is remarkably similar to that in section ??.

The following macros generally set things up for the ‘standard’ footnote format.

`\prebodyfootmark` Two convenience macros for use by `\...@footnotemark...` macros.
`\postbodyfootmark`

```

3925 \newcommand*\prebodyfootmark{%
3926     \leavevmode
3927     \ifhmode
3928         \edef\@x@sf{\the\spacefactor}%
3929         \m@mmf@check
3930         \nobreak
3931     \fi}
3932 \newcommand*\postbodyfootmark{%
3933     \m@mmf@prepare
3934     \ifhmode\spacefactor\@x@sf\fi\relax}
3935
3936 %

```

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.

```

3937 \newcommandx{\arrangementX}[2][1,usedefault]{%
3938   \def\do##1{%
3939     \csname arrangementX@#2\endcsname{##1}%
3940   }%
3941   \ifstrempy{#1}%
3942     {%
3943       \dolistloop{\@series}%
3944     }%
3945     {
3946       \docsvlist{#1}%
3947     }%
3948 }%
3949 %

```

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.

```

3950 \newcommand*{\normal@footnotemarkX}[1]{%
3951   \prebodyfootmark
3952   \wrapped@bodyfootmarkX{#1}%
3953   \postbodyfootmark}
3954
3955 %

```

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

```

3956 \newcommand*{\normalbodyfootmarkX}[1]{%
3957   \hbox{\textsuperscript{\normalfont\@nameuse{@thefnmark#1}}}}
3958 %

```

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

```

3959 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnoteX}[2]{%
3960   \beforeinsertion@X{#1}%
3961   \insert\@nameuse{footins#1}\bgroup
3962     \reset@font%
3963     \hsize=\expandafter\dimexpr\csuse{widthX@#1}\relax%
3964     \noindent\csuse{bhooknoteX@#1}%
3965     \csuse{notefontsizeX@#1}%
3966     \footplitskips

```

```

3967 \ifl@dpairing\ifl@dpaging\else%
3968   \setnoteswidthliketwocolumnsX@{#1}%
3969   \fi\fi%
3970   \setnotesXpositionliketwocolumns@{#1}%
3971   \spaceskip=\z@skip \xspaceskip=\z@skip
3972   \csuse{\csuse{footnote@dir}}\@nameuse{footfmt#1}{#1}{#2}\egroup}
3973
3974 %

```

`\mpnormalvfootnoteX` The minipage version.

```

3975 \newcommand*{\mpnormalvfootnoteX}[2]{%
3976   \get@thisfootnoteX{#1}%
3977   \get@fnmarkX{#1}{\thisc@footnote}%
3978   \edef\this@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
3979   \global\setbox\@nameuse{mpfootins#1}\vbox{%
3980     \unvbox\@nameuse{mpfootins#1}
3981     \noindent\csuse{bhooknoteX@#1}%
3982     \csuse{notefontsizeX@#1}%
3983     \hsize\columnwidth
3984     \@parboxrestore
3985     \color@begingroup
3986     \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}}
3987
3988 %

```

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

```

3989 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmtX}[2]{%
3990   \ifluatex%
3991     \texdir\footnote@luatextextdir%
3992     \pardir\footnote@luatexpardir%
3993     \par%
3994   \fi%
3995   \protected@edef\@currentlabel{%
3996     \@nameuse{@thefnmark#1}%
3997   }%
3998   \ledsetnormalparstuffX{#1}%
3999   \hangindent=\csuse{hangindentX@#1}%
4000   \everypar{\hangindent=\csuse{hangindentX@#1}}%
4001   \rule\z@\splittopskip%
4002   {{\csuse{notenumfontX@#1}\wrapped@footfootmarkX{#1}}%
4003     \csuse{wrapcontentX@#1}{#2}%
4004   \strut\par}}
4005
4006 %

```

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


```

4007 \newcommand*{\normalfootfootmarkX}[1]{%
4008   \textsuperscript{\@nameuse{@thefnmark#1}}}
4009
4010 %

```

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

```

4011 \newcommand*{\normalfootstartX}[1]{%
4012   \ifdimequal{Opt}{\prenotesX@}{}%
4013   {%
4014     \iftoggle{prenotesX@}{%
4015       \togglefalse{prenotesX@}%
4016       \skip\csname footins#1\endcsname=%
4017       \glueexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
4018     }%
4019     {}%
4020   }%
4021   \vskip\skip\csname footins#1\endcsname%
4022   \leftskip=\z@
4023   \rightskip=\z@
4024   \ifl@dpairing\else%
4025     \hsize=\old@hsize%
4026     \fi%
4027   \setnoteswidthliketwocolumnsX@{#1}%
4028   \setnotesXpositionliketwocolumns@{#1}%
4029   \print@footnoteXrule{#1}%
4030 }%
4031
4032 %

```

\normalfootnoteruleX The rule drawn before the footnote series group.

```

4033 \let\normalfootnoteruleX=\footnoterule
4034
4035 %

```

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

```

4036 \newcommand*{\normalfootgroupX}[1]{%
4037   \csuse{bhookgroupX@#1}%
4038   \unvbox\@nameuse{footins#1}%
4039   \hsize=\old@hsize%
4040 }%
4041
4042 %

```

\mpnormalfootgroupX The minipage version.

```

4043 \newcommand*{\mpnormalfootgroupX}[1]{%
4044 \vskip\skip\@nameuse{mpfootins#1}
4045 \ifl@dpairing\ifparledgroup%
4046 \leavevmode\marks\parledgroup@{begin}%
4047 \marks\parledgroup@series{#1}%
4048 \marks\parledgroup@type{footnoteX}%
4049 \fi\fi\normalcolor
4050 \ifparledgroup%
4051 \ifl@dpairing%
4052 \else%
4053 \setnoteswidthliketwocolumnsX@{#1}%
4054 \setnotesXpositionliketwocolumns@{#1}%
4055 \print@footnoteXrule{#1}%
4056 \fi%
4057 \else%
4058 \setnoteswidthliketwocolumnsX@{#1}%
4059 \setnotesXpositionliketwocolumns@{#1}%
4060 \print@footnoteXrule{#1}%
4061 \fi%
4062 \csuse{bhookgroupX@#1}%
4063 \unvbox\@nameuse{mpfootins#1}}
4064
4065 %

```

`\normalbfnoteX`₄₀₆₆

```

4067 \newcommand{\normalbfnoteX}[2]{%
4068 \get@thisfootnoteX{#1}%
4069 \ifl@Rcol%
4070 \ifluatex
4071 \footnotelang@lua[R]%
4072 \fi
4073 \@ifundefined{xpg@main@language}%if polyglossia
4074 {}%
4075 {\footnotelang@poly[R]}%
4076 \xright@appenditem{%
4077 \noexpand\led@set@index@fornote{#1}%
4078 \noexpand\prepare@edindex@fornote{\l@d@nums}%
4079 \unexpanded{\def\this@footnoteX@reading}{\the\csname footnote#1
@reading\endcsname}%
4080 \noexpand\vbfnoteX{#1}{#2}{\thisc@footnote}%
4081 \noexpand\led@reinit@index@fornote%
4082 \unexpanded{\advance\@edindex@fornote@m@ne}%
4083 }%
4084 \to\inserts@listR
4085 \global\advance\insert@countR \@ne%
4086 \else%
4087 \ifluatex
4088 \footnotelang@lua%
4089 \fi

```

```

4090 \ifundefined{xpg@main@language}%if polyglossia
4091 {}%
4092 {\footnotelang@poly}%
4093 \xright@appenditem{%
4094 \noexpand\led@set@index@fornote{#1}%
4095 \noexpand\prepare@edindex@fornote{\led@nums}%
4096 \unexpanded{\def\this@footnoteX@reading}{\the\csname footnote#1
@reading\endcsname}%
4097 \noexpand\vbfnoteX{#1}{#2}{\thisc@footnote}%
4098 \noexpand\led@reinit@index@fornote%
4099 \unexpanded{\advance\@edindex@fornote@\m@ne}%
4100 }%
4101 \to\inserts@list
4102 \global\advance\insert@count \@ne%
4103 \fi
4104 \ignorespaces}
4105 %
4106 %

```

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

```

4107 \newcommand{\get@thisfootnoteX}[1]{%
4108 \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
4109 \protected@xdef\thisc@footnote{\the\csname footnote#1@reading\
endcsname}%
4110 }{%
4111 \protected@xdef\thisc@footnote{\the\csname c@footnote#1\endcsname}%
4112 }%
4113 }%
4114 %

```

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

```

4115 \newcommand{\vbfnoteX}[3]{%
4116 \get@fnmarkX{#1}{#3}%
4117 \@nameuse{regvfootnote#1}{#1}{#2}%
4118 }%
4119 %
4120 %

```

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

```

4121 \newcommand{\get@fnmarkX}[2]{%
4122 \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
4123 {%
4124 \stepcounter{footnote#1@typeset}%
4125 \setcounter{footnote#1}{\value{footnote#1@typeset}}}%

```

```

4126 \namedef{@thefnmark#1}{\csuse{thefootnote#1}}%
4127 \immediate\write\@mainaux{%
4128 \csgdef{footnote#1reading#2=typeset}{\the\csname c@footnote#1
@typeset\endcsname}%
4129 }%
4130 }%
4131 {%
4132 \setcounter{footnote#1}{#2}%
4133 \namedef{@thefnmark#1}{\csuse{thefootnote#1}}%
4134 }%
4135 }
4136 %
4137 %

```

```

\newcommand{\vnumfootnoteX}[2]{%
4139 \ifnumberedpar@
4140 \edtext{}{\normalbfnoteX{#1}{#2}}%
4141 \else
4142 \def\this@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
4143 \get@thisfootnoteX{#1}%
4144 \get@fnmarkX{#1}{\expandonce\thisc@footnote}%
4145 \@nameuse{regvfootnote#1}{#1}{#2}%
4146 \fi}
4147
4148 %

```

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

```

4149 \newcommand*{\arrangementX@normal}[1]{%
4150 \csgdef{series@displayX#1}{normal}
4151 \expandafter\let\csname footstart#1\endcsname=\normalfootstartX
4152 \namedef{@footnotemark#1}{\normal@footnotemarkX{#1}}
4153 \namedef{bodyfootmark#1}{\normalbodyfootmarkX{#1}}
4154 \expandafter\let\csname regvfootnote#1\endcsname=\normalvfootnoteX
4155 \expandafter\let\csname vfootnote#1\endcsname=\vnumfootnoteX
4156 \expandafter\let\csname footfmt#1\endcsname=\normalfootfmtX
4157 \namedef{footfootmark#1}{\normalfootfootmarkX{#1}}
4158 \expandafter\let\csname footgroup#1\endcsname=\normalfootgroupX
4159 \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
4160 \count\csname footins#1\endcsname=1000
4161 \csxdef{default@footins#1}{1000}%Use to have note only for one side
4162 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
4163 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
4164 \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
4165 %

```

Additions for minipages.

```

4166 \ifnoledgroup@else%
4167 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
4168 \expandafter\let\csname mpfootgroup#1\endcsname=\mpnormalfootgroupX
4169 \count\csname mpfootins#1\endcsname=1000
4170 \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
4171 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
4172 \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
4173 \fi
4174 }
4175
4176 %

```

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 177 \newcommand*\arrangementX@twocol}[1]{%
4178 \csgdef{series@displayX#1}{twocol}
4179 \expandafter\let\csname regvfootnote#1\endcsname=\twocolvfootnoteX
4180 \expandafter\let\csname footfmt#1\endcsname=\twocolfootfmtX
4181 \expandafter\let\csname footgroup#1\endcsname=\twocolfootgroupX
4182 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
4183 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
4184 \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
4185 \twocolfootsetupX{#1}
4186 \ifnoledgroup@else%
4187 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
4188 \expandafter\let\csname mpfootgroup#1\endcsname=\mptwocolfootgroupX
4189 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
4190 \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}
4191 \mptwocolfootsetupX{#1}
4192 \fi%
4193 }
4194
4195 %

```

```

\twocolfootsetupX \twocolfootsetupX{<series>}
\mptwocolfootsetupX
4196 \newcommand*\twocolfootsetupX}[1]{%
4197 \count\csname footins#1\endcsname 500
4198 \csxdef{default@footins#1}{500}%Use this to confine the notes to one
side only
4199 \multiply\dimen\csname footins#1\endcsname by \tw@}
4200 \newcommand*\mptwocolfootsetupX}[1]{%
4201 \count\csname mpfootins#1\endcsname 500
4202 \multiply\dimen\csname mpfootins#1\endcsname by \tw@}
4203
4204 %

```

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

```

4205 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolvfootnoteX}[2]{%
4206   \beforeinsertion@X{#1}%
4207   \insert\csname footins#1\endcsname\bgroup%
4208     \hspace=\expandafter\dimexpr\csuse{widthX@#1}\relax%
4209     \noindent\csuse{bhooknoteX@#1}%
4210     \csuse{notefontsizeX@#1}%
4211     \footssplitsskip%
4212     \spaceskip=\z@skip \xspaceskip=\z@skip%
4213     \@nameuse{footfmt#1}{#1}{#2}\egroup}
4214
4215 %

```

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

```

4216 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolfootfmtX}[2]{%
4217   \protected@edef\@currentlabel{%
4218     \@nameuse{@thefnmark#1}%
4219   }%
4220   \normal@pars%
4221   \hangindent=\csuse{hangindentX@#1}%
4222   \everypar{\hangindent=\csuse{hangindentX@#1}}%
4223   \hspace \csuse{hsizetwocolX@#1}%
4224   \nottoggle{parindentX@#1}{\parindent=\z@}{}%
4225   \tolerance=5000\relax%
4226   \par%
4227   \@tempdima=\parindent%
4228   \csuse{colalignX@#1}%
4229   \parindent=\@tempdima%
4230   {\hspace{\parindent}%
4231     \csuse{notenumfontX@#1}\wrapped@footfootmarkX{#1}\strut%
4232     \csuse{wrapcontentX@#1}{#2}%
4233     \strut\par}%
4234   \allowbreak%
4235 }%
4236
4237 %

```

`\twocolfootgroupX` `\twocolfootgroupX{<series>}`
`\mptwocolfootgroupX`

```

4238 \newcommand*{\twocolfootgroupX}[1]{\csuse{bhookgroupX@#1}\csuse{
4239   notefontsizeX@#1}
4240   \splittopskip=\ht\strutbox
4241   \expandafter
4242   \rigidbalanceX\csname footins#1\endcsname \tw@ \splittopskip}}
4243
4244 \newcommand*{\mptwocolfootgroupX}[1]{%
4245   \vskip\skip\@nameuse{mpfootins#1}
4246   \ifl@dpairing\ifparledgroup%

```

```

4246 \leavevmode\marks\parledgroup@{begin}%
4247 \marks\parledgroup@series{#1}%
4248 \marks\parledgroup@type{footnoteX}%
4249 \fi\fi\normalcolor
4250 \ifparledgroup%
4251 \ifl@dpairing%
4252 \else%
4253 \setnoteswidthliketwocolumnsX@{#1}%
4254 \setnotesXpositionliketwocolumns@{#1}%
4255 \print@footnoteXrule{#1}%
4256 \fi%
4257 \else%
4258 \setnoteswidthliketwocolumnsX@{#1}%
4259 \setnotesXpositionliketwocolumns@{#1}%
4260 \print@footnoteXrule{#1}%
4261 \fi%
4262 \csuse{bhookgroupX@#1}%
4263 \splittopskip=\ht\strutbox
4264 \expandafter
4265 \rigidbalanceX\csname mpfootins#1\endcsname \tw@ \splittopskip}}
4266
4267 %

```

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 4268 \newcommand*{\arrangementX@threecol}[1]{%
4269 \csgdef{series@displayX#1}{threecol}
4270 \expandafter\let\csname regvfootnote#1\endcsname=\threecolvfootnoteX
4271 \expandafter\let\csname footfmt#1\endcsname=\threecolfootfmtX
4272 \expandafter\let\csname footgroup#1\endcsname=\threecolfootgroupX
4273 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
4274 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
4275 \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
4276 \threecolfootsetupX{#1}
4277 \ifnoledgroup@ \else%
4278 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
4279 \expandafter\let\csname mpfootgroup#1\endcsname=\mpthreecolfootgroupX
4280 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
4281 \advance\skip\csname mpfootins#1\endcsname by \csuse{afterruleX@#1}
4282 \mpthreecolfootsetupX{#1}
4283 \fi%
4284 }
4285
4286 %

```

```

\threecolfootsetupX \threecolfootsetupX{<series>}
\mpthreecolfootsetupX

```

```

4287 \newcommand*{\threecolfootsetupX}[1]{%
4288   \count\csname footins#1\endcsname 333
4289   \csxdef{default@footins#1}{333}%Use this to confine the notes to one
side only
4290   \multiply\dimen\csname footins#1\endcsname by \thr@@}
4291 \newcommand*{\mpthreecolfootsetupX}[1]{%
4292   \count\csname mpfootins#1\endcsname 333
4293   \multiply\dimen\csname mpfootins#1\endcsname by \thr@@}
4294
4295 %

```

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

```

4296 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolvfootnoteX}[2]{%
%
4297   \beforeinsertion@X{#1}%
4298   \insert\csname footins#1\endcsname\bgroup%
4299     \hsize=\expandafter\dimexpr\csuse{widthX@#1}\relax%
4300     \noindent\csuse{bhooknoteX@#1}%
4301     \csuse{notefontsizeX@#1}%
4302     \footsplitskips%
4303     \@nameuse{footfmt#1}{#1}{#2}\egroup}
4304
4305 %

```

`\threecolfootfmtX` `\threecolfootfmtX{<series>}`

```

4306 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmtX}[2]{%
4307   \protected@edef\@currentlabel{%
4308     \@nameuse{@thefnmark#1}%
4309   }%
4310   \hangindent=\csuse{hangindentX@#1}%
4311   \everypar{\hangindent=\csuse{hangindentX@#1}}%
4312   \normal@pars%
4313   \hsize \csuse{hsizethreecolX@#1}%
4314   \nottoggle{parindentX@#1}{\parindent=\z@}{}%
4315   \tolerance=5000\relax%
4316   \@tempdima=\parindent%
4317   \csuse{colalignX@#1}%
4318   \parindent=\@tempdima%
4319   {\hspace{\parindent}%
4320   \csuse{notenumfontX@#1}\wrapped@footfootmarkX{#1}\strut%
4321     \csuse{wrapcontentX@#1}{#2}%
4322     \strut\par}\allowbreak}
4323
4324 %

```

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


```

4325 \newcommand*\threecolfootgroupX}[1]{\csuse{bhookgroupX@#1}\csuse{
notefontsizeX@#1}
4326 \splittopskip=\ht\strutbox
4327 \expandafter
4328 \rigidbalanceX\csname footins#1\endcsname \thr@@ \splittopskip}}
4329
4330 \newcommand*\mpthreecolfootgroupX}[1]{\%
4331 \vskip\skip\@nameuse{mpfootins#1}
4332 \ifl@dpairing\ifparledgroup
4333 \leavevmode\marks\parledgroup@{begin}%
4334 \marks\parledgroup@series{#1}%
4335 \marks\parledgroup@type{footnoteX}%
4336 \fi\fi\normalcolor
4337 \ifparledgroup%
4338 \ifl@dpairing%
4339 \else%
4340 \setnoteswidthliketwocolumnsX@{#1}%
4341 \setnotesXpositionliketwocolumns@{#1}%
4342 \print@footnoteXrule{#1}%
4343 \fi%
4344 \else%
4345 \setnoteswidthliketwocolumnsX@{#1}%
4346 \setnotesXpositionliketwocolumns@{#1}%
4347 \print@footnoteXrule{#1}%
4348 \fi%
4349 \csuse{bhookgroupX@#1}%
4350 \splittopskip=\ht\strutbox
4351 \expandafter
4352 \rigidbalanceX\csname mpfootins#1\endcsname \thr@@ \splittopskip}}
4353
4354 \%

```

XIII.4.5 Paragraphed footnotes

The following macros set footnotes as one paragraph.

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

```

4355 \newcommand*\arrangementX@paragraph}[1]{\%
4356 \csgdef{series@displayX#1}{paragraph}%
4357 \expandafter\let\csname footstart#1\endcsname=\parafootstartX
4358 \expandafter\let\csname regvfootnote#1\endcsname=\para@vfootnoteX
4359 \expandafter\let\csname footfmt#1\endcsname=\parafootfmtX
4360 \expandafter\let\csname footgroup#1\endcsname=\para@footgroupX
4361 \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
4362 \count\csname footins#1\endcsname=1000
4363 \csxdef{default@footins#1}{1000}%Use this to confine the notes to one
side only
4364 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
4365 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%

```

```

4366 \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
4367 \para@footsetupX{#1}
4368 \ifnoledgroup\else
4369   \expandafter\let\csname mpvfootnote#1\endcsname=\mppara@vfootnoteX
4370   \expandafter\let\csname mpfootgroup#1\endcsname=\mppara@footgroupX
4371   \count\csname mpfootins#1\endcsname=1000
4372   \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
4373   \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
4374   \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
4375 \fi
4376 }
4377
4378 %

```

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

```

4379 \newcommand*{\para@footsetupX}[1]{\csuse{bhookgroupX@#1}\csuse{
notefontsizeX@#1}
4380 \setnoteswidthliketwocolumnsX@{#1}%
4381 \ifcsempy{widthX@#1}%
4382   {}%
4383   {\columnwidth=\expandafter\dimexpr\csuse{widthX@#1}\relax}%
4384 \dimen0=\baselineskip
4385 \multiply\dimen0 by 1024
4386 \divide\dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\relax
4387 %
4388 \expandafter
4389 \xdef\csname footfudgefactor#1\endcsname{%
\expandafter\strip@pt\dimen0 }}
4390
4391 %

```

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

```

4392 \newcommand*{\parafootstartX}[1]{%
4393   \ifdimequal{Opt}{\prenotesX@}{}%
4394   {%
4395     \iftoggle{prenotesX@}{%
4396       \togglefalse{prenotesX@}%
4397       \skip\csname footins#1\endcsname=%
4398       \glueexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
4399     }%
4400   }%
4401 }%
4402 \leftskip=\z@
4403 \rightskip=\z@
4404 \nottoggle{parindentX@#1}{\parindent=\z@}{}%
4405 \vskip\skip@nameuse{footins#1}%
4406 \setnoteswidthliketwocolumnsX@{#1}%
4407 \setnotesXpositionliketwocolumns@{#1}%

```

```

4408 \print@footnoteXrule{#1}%
4409 }
4410
4411 %

```

```

\para@vfootnoteX \para@vfootnoteX{<series>}{<text>}
\mppara@vfootnoteX

```

```

4412 \newcommand*{\para@vfootnoteX}[2]{%
4413   \csuse{beforeinsertingX@#1}%
4414   \insert\csname footins#1\endcsname%
4415   \bgroup
4416     \csuse{notefontsizeX@#1}
4417     \footsplitskips
4418     \setbox0=\vbox{\hsize=\maxdimen%
4419       \let\bidir@RTL@everypar\@empty%
4420       \insert@txtbeforenotesX{#1}%
4421       \noindent\csuse{bhooknoteX@#1}%
4422       \@nameuse{footfmt#1}{#1}{#2}}%
4423     \setbox0=\hbox{\unvvhX{0}{#1}}%
4424     \dp0=\z@
4425     \ht0=\csname footfudgefactor#1\endcsname\wd0
4426     \box0
4427     \penalty0
4428   \egroup}
4429 \newcommand*{\mppara@vfootnoteX}[2]{%
4430   \get@thisfootnoteX{#1}%
4431   \get@fnmarkX{#1}{\thisc@footnote}%
4432   \edef\thisc@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
4433   \global\setbox\@nameuse{mpfootins#1}\vbox{%
4434     \unvvhX\@nameuse{mpfootins#1}
4435     \csuse{notefontsizeX@#1}
4436     \footsplitskips
4437     \setbox0=\vbox{\hsize=\maxdimen%
4438       \let\bidir@RTL@everypar\@empty%
4439       \noindent\color@begingroup%
4440       \csuse{bhooknoteX@#1}%
4441       \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}%
4442     \setbox0=\hbox{\unvvhX{0}{#1}}%
4443     \dp0=\z@
4444     \ht0=\csname footfudgefactor#1\endcsname\wd0
4445     \box0
4446     \penalty0}}
4447
4448 %

```

```

\unvvhX49 \newcommand*{\unvvhX}[2]{% 2th is optional for retro-compatibility
4450   \setbox0=\vbox{\unvvhX#1%
4451     \global\setbox1=\lastbox}%
4452   \unhbox1

```

```

4453 \unskip          % remove \rightskip,
4454 \unskip          % remove \parfillskip,
4455 \unpenalty       % remove \penalty of 10000,
4456 \hskip\csuse{afternoteX@#2}%
4457 \relax}% but add the glue to go between the notes
4458
4459 %

```

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

```

4460 \newcommand*{\parafootfmtX}[2]{%
4461   \protected@edef\@currentlabel{%
4462     \@nameuse{@thefnmark#1}%
4463   }%
4464   \insertparafootsepX{#1}%
4465   \ledsetnormalparstuff@common%
4466   {\csuse{notenumfontX@#1}%
4467    \csuse{notenumfontX@#1}%
4468    \wrapped@footfootmarkX{#1}%
4469    \strut%
4470    \csuse{wrapcontentX@#1}{#2}%
4471    \penalty-10}}
4472
4473 %

```

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

`\mppara@footgroupX`

```

4474 \newcommand*{\para@footgroupX}[1]{%
4475   \hsize=\expandafter\dimexpr\csuse{widthX@#1}\relax%
4476   \unvbox\csname footins#1\endcsname
4477   \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
4478   \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
4479   \makehboxofhboxes
4480   \setbox0=\hbox{\unhbox0 \removehboxes}%
4481   \csuse{bhookgroupX@#1}
4482   \csuse{notefontsizeX@#1}
4483   \unhbox0\par}
4484
4485 \newcommand*{\mppara@footgroupX}[1]{%
4486   \setnoteswidthliketwocolumnsX@{#1}%
4487   \vskip\skip\@nameuse{mpfootins#1}
4488   \ifl@dpairing\ifparledgroup
4489     \leavevmode%
4490     \leavevmode\marks\parledgroup@{begin}%
4491     \marks\parledgroup@series{#1}%
4492     \marks\parledgroup@type{footnoteX}%
4493     \fi\fi\normalcolor
4494     \ifparledgroup%
4495       \ifl@dpairing%
4496       \else%

```

```

4497 \setnoteswidthliketwocolumnsX@{#1}%
4498 \setnotesXpositionliketwocolumns@{#1}%
4499 \print@footnoteXrule{#1}%
4500 \fi%
4501 \else%
4502 \setnoteswidthliketwocolumnsX@{#1}%
4503 \setnotesXpositionliketwocolumns@{#1}%
4504 \print@footnoteXrule{#1}%
4505 \fi%
4506 \unvbox\csname mpfootins#1\endcsname
4507 \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
4508 \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
4509 \makehboxofhboxes
4510 \setbox0=\hbox{\unhbox0 \removehboxes}%
4511 \csuse{bhookgroupX@#1}%
4512 \csuse{notefontsizeX@#1}%
4513 \nottoggle{parindentX@#1}{\parindent=\z@}{}%
4514 \unhbox0\par}}
4515 %
4516 %

```

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

```

\insertparafootsepX17 \newcommand{\insertparafootsepX}[1]{%
4518 \ifledRcol%
4519 \ifnumequal{\csuse{prevpage#1@numR}}{\page@numR}%
4520 {\csuse{Xparafootsep@#1}}%
4521 {}%
4522 \global\csname prevpage#1@numR\endcsname=\page@numR%
4523 \else%
4524 \ifnumequal{\csuse{prevpage#1@num}}{\page@num}%
4525 {\csuse{Xparafootsep@#1}}%
4526 {}%
4527 \global\csname prevpage#1@num\endcsname=\page@num%
4528 \fi%
4529 }
4530 %

```

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.

```

4531 \newcommand{\wrapped@footfootmarkX}[1]{%
4532 \ifdefined\hypertarget%
4533 \hyperlink%
4534 {@bodyfootmark#1@\this@footnoteX@reading}%

```

```

4535     {\@nameuse{footfootmark#1}}}%
4536     \Hy@raisedlink{%
4537       \hypertarget%
4538       {@footnotemark#1\this@footnoteX@reading}%
4539       {}%
4540     }%
4541   \else%
4542     \@nameuse{footfootmark#1}%
4543   \fi%
4544 }%
4545 %

```

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

```

4546 \newcommand{\wrapped@bodyfootmarkX}[1]{%
4547   \ifdefined\hypertarget%
4548     \hyperlink%
4549     {@footnotemark#1\expandafter\the\csname footnote#1@reading\
4550 endcsname}%
4551     {\@nameuse{bodyfootmark#1}}}%
4552   \Hy@raisedlink{%
4553     \hypertarget%
4554     {@bodyfootmark#1\expandafter\the\csname footnote#1@reading\
4555 endcsname}%
4556     {}%
4557   }%
4558   \else%
4559     \@nameuse{bodyfootmark#1}%
4560   \fi%
4561 }%
4562 %

```

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

`\par` should always be redefined to `\endgraf` within the format macro (this is what `\normal@pars` does), to override tricky material in the main text to get the lines numbered automatically (as set up by `\autopar`, for example).

In the case of footnote arranged in a “normal” way, we also must set some setting for paragraph indent and text direction when using `Lua®TeX`.

That why we have defined `\ledsetnormalparstuff@common` in order to make this setting for both familiar and critical notes. This command is called by command to make specific setting to critical or familiar footnote.

```

\ledsetnormalparstuff@common61 \newcommand*{\ledsetnormalparstuff@common}{%
\Xledsetnormalparstuff62 \ifluatex%
\ledsetnormalparstuffX

```

```

4563 \ifdefstring{\footnote@luatextextdir}{TLT}{}%
4564 {\textdir\footnote@luatextextdir}%
4565 \pardir\footnote@luatexpardir%
4566 \fi%
4567 \csuse{\csuse{footnote@dir}}%
4568 \normal@pars%
4569 \parfillskip \z@ \@plus 1fil}%
4570
4571 \newcommand*{\Xledsetnormalparstuff}[1]{%
4572 \ledsetnormalparstuff@common%
4573 \nottoggle{Xparindent@#1}{\parindent=\z@}{\hspace{\parindent}}%
4574 }%
4575
4576 \newcommand*{\ledsetnormalparstuffX}[1]{%
4577 \ledsetnormalparstuff@common%
4578 \nottoggle{parindentX@#1}{\parindent=\z@}{\hspace{\parindent}}%
4579 }%
4580 %

```

XV Footnotes' width for two columns

We define here some commands which make sense only with `reledpar`, but must be called when defining notes parameters. These commands change the width of block notes to allow them to have the same size than two parallel columns.

`\old@hsize` These two commands are called at the beginning of critical or familiar notes groups. They set, if the option is enabled, the `\hsize`. They are also called at the on the setup for paragraphed notes.

`\noteswidthliketwocolumns@`
`\oteswidthliketwocolumnsX@`

```

4581
4582 \newdimen\old@hsize%
4583 \AtBeginDocument{\old@hsize=\hsize}%
4584
4585 \newcommand{\setXnoteswidthliketwocolumns@}[1]{%
4586 \global\let\hsize@fornote=\hsize%
4587 \global\old@hsize=\hsize%
4588 \let\old@columnwidth=\columnwidth%
4589 \iftoggle{Xnoteswidthliketwocolumns@#1}%
4590 {%
4591 \setwidthliketwocolumns%
4592 \global\let\hsize@fornote=\hsize%
4593 }%
4594 {}%
4595 \let\hsize=\hsize@fornote%
4596 \let\columnwidth=\old@columnwidth%
4597 }%
4598
4599 \newcommand{\setnoteswidthliketwocolumnsX@}[1]{%
4600 \global\let\hsize@fornote=\hsize%

```

```

4601 \global\old@hsize=\hsize%
4602 \let\old@columnwidth=\columnwidth%
4603 \iftoggle{noteswidthliketwocolumnsX@#1}%
4604 {%
4605   \setwidthliketwocolumns%
4606   \global\let\hsize@fornote=\hsize%
4607 }%
4608 {}%
4609 \let\hsize=\hsize@fornote%
4610 \let\columnwidth=\old@columnwidth%
4611 }%
4612
4613 %

```

`\setXnotespositionliketwocolumns@` 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`.

`\setnotesXpositionliketwocolumns@`

```

4614 \newcommand{\setXnotespositionliketwocolumns@}[1]{%
4615   \iftoggle{Xnoteswidthliketwocolumns@#1}{%
4616     \csuse{setnotespositionliketwocolumns@\columns@position}%
4617   }{}%
4618 }%
4619
4620 \newcommand{\setnotesXpositionliketwocolumns@}[1]{%
4621   \iftoggle{noteswidthliketwocolumnsX@#1}{%
4622     \csuse{setnotespositionliketwocolumns@\columns@position}%
4623   }{}%
4624 }%
4625
4626 %

```

XVI Footnotes' order

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

`\mpfnpos`

```

4627 \def\@fnpos{familiar-critical}
4628 \def\@mpfnpos{critical-familiar}
4629 \newcommand{\fnpos}[1]{\xdef\@fnpos{#1}}
4630 \newcommand{\mpfnpos}[1]{\xdef\@mpfnpos{#1}}
4631 %

```

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@Xfootnoterule32 \newcommand{\print@Xfootnoterule}[1]{%
\print@footnoterule33 \vskip-\csuse{Xafterrule@#1}%Because count in \dimen\csuse{#1footins}
4634 \nointerlineskip%
4635 \moveleft-\leftskip\vbox{\csuse{#1footnoterule}}%
4636 \nointerlineskip%
4637 \vskip\csuse{Xafterrule@#1}%
4638 }%
4639
4640 \newcommand{\print@footnoterule}[1]{%
4641 \vskip-\csuse{afterruleX@#1}%Because count in \dimen\csuse{footins#1}
4642 \nointerlineskip%
4643 \moveleft-\leftskip\vbox{\csuse{footnoterule#1}}%
4644 \nointerlineskip%
4645 \vskip\csuse{afterruleX@#1}%
4646 }%
4647
4648 %

```

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@` 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.

```

4649 \newtoggle{Xprenotes@}%
4650 \toggletrue{Xprenotes@}%
4651 \newcommand{\Xprenotes@}{Opt}%
4652 \newcommand*{\Xprenotes}[1]{\renewcommand{\Xprenotes@}{#1}}%
4653 \newcommand{\preXnotes}[1]{\led@warning@preXnotes@deprecated\Xprenotes{#1}}
    %For compatibility
4654 %

```

The same, but for familiar footnotes.

```

\Xprenotes 55 \newtoggle{prenotesX@}
\Xprenotes@56 \toggletrue{prenotesX@}
4657 \newcommand{\prenotesX@}{Opt}
4658 \newcommand*{\prenotesX}[1]{\renewcommand{\prenotesX@}{#1}}
4659 %

```

XVIII.0.3 Internal commands

```

firstXseries@60 \gdef\firstXseries@{}
prepare@Xprenotes61 \newcommand{\prepare@Xprenotes}[1]{%
4662   \ifdimequal{Opt}{\Xprenotes@}%
4663   }%
4664   {%
4665     \IfStrEq{\firstXseries@}{}{%
4666       \global\skip\csuse{#1footins}=\Xprenotes@%
4667       \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
4668       \gdef\firstXseries@{#1}%
4669     }%
4670     {%
4671       \ifseriesbefore{#1}{\firstXseries@}%
4672       {%
4673         \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@\firstXseries@}%
4674         \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
4675         \gdef\firstXseries@{#1}%
4676       }%
4677     }%

```

```

4678 }%
4679 }%
4680 }
4681 %

```

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

```

firstseriesX@ \gdef\firstseriesX@{}
prepare@prenotesX@ \newcommand{\prepare@prenotesX}[1]{%
4684 \ifdimequal{0pt}{\prenotesX@}%
4685 {}%
4686 {%
4687 \IfStrEq{\firstseriesX@}{}%
4688 \global\skip\csuse{footins#1}=\prenotesX@%
4689 \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
4690 \gdef\firstseriesX@{#1}%
4691 }%
4692 {%
4693 \ifseriesbefore{#1}{\firstseriesX@}%
4694 {%
4695 \global\skip\csuse{footins#1}=\csuse{beforenotesX@\firstseriesX@}%
4696 \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
4697 \gdef\firstXseries@{#1}%
4698 }%
4699 {}%
4700 }%
4701 }%
4702 }
4703 %

```

XIX Endnotes

First, check the `noend` option.

```

4704 \ifbool{noend@}{}%Used instead of \ifnoend@ to prevent expansion problem
4705 %

```

XIX.1 Internal commands

`\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.

```

4706 \newcommand{\l@dend@open}[1]{%
4707   \global\booltrue{l@dend@#1}%
4708   \expandafter\immediate%
4709   \expandafter\openout%
4710   \csname l@d@#1end\endcsname%
4711   =\l@auxdir\jobname.#1end\relax%
4712 }%
4713 \newcommand{\l@dend@close}[1]{%
4714   \global\boolfalse{l@dend@#1}%
4715   \expandafter\immediate%
4716   \expandafter\closeout\csname l@d@#1end\endcsname%
4717 }%
4718
4719 %

```

\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.

```

4720 \newcommand{\l@dend@stuff}{%
4721   \def\do##1{%
4722     \ifbool{l@dend@##1}{}%
4723     {\l@dend@open{##1}}%
4724     \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname{\
string\l@d@section{\the\section@num}\@percentchar}%
4725   }%
4726   \dolistloop{@series}%
4727 }%
4728
4729 %

```

\endprint The \endprint here is nearly identical in its functioning to \normalfootfmt.
\l@d@section The endnote file also contains \l@d@section commands, which supply the section numbers from the main text; standard reledmac does nothing with this information, but it is there if you want to write custom macros to do something with it. Arguments are:

- #1 Line numbers and font selection.
- #2 Lemma.
- #3 Note content.
- #4 Series.
- #5 Optional argument of \Xendnote.
- #6 Side (L or R).
- #7 Label for cross-referencing.

```

4730 \global\newbool{parapparatus@}{\long}\def\endprint#1#2#3#4#5#6#7{
4731 \csuse{Xendbhooknote@#4}%
4732 \csuse{Xendnotefontsize@#4}%
4733 \hangindent=\csuse{Xendhangindent@#4}%
4734 \ifXendinsertsep%
4735 \hskip\csuse{Xendafternote@#4}\relax%
4736 \csuse{Xendsep@#4}%
4737 \else%
4738 \iftoggle{Xendparagraph@#4}%
4739 {\global\Xendinsertsep@true}%
4740 {}%
4741 \fi%
4742 \xdef\@currentseries{#4}%
4743 \def\do##1{%
4744 \setkeys[mac]{truefootnoteoption}{##1}%
4745 }%
4746 \notblank{#5}{\docsvlist{#5}}{}%
4747 \IfStrEq{#6}{R}{\ledRcol@true}{}%
4748 \def\@this@crossref@start{#7:start}%
4749 \def\@this@crossref@end{#7:end}%
4750 \printlineendnote{#1}{#4}%
4751 \IfStrEq{#6}{R}{\ledRcol@false}{}%
4752 \undef\@this@crossref@start%
4753 \undef\@this@crossref@end%
4754 \nottoggle{Xendlemmadisablefontselection@#4}%
4755 {\select@lemmafont#1}%
4756 {}%
4757 \bgroup%
4758 \csuse{Xendlemmafont@#4}%
4759 \csuse{Xendwraplemma@#4}{#2}%
4760 \egroup%
4761 \ifboolexpr{
4762 togl {nosep@}%
4763 or test{\ifcempty{Xendlemmaseparator@#4}}%
4764 }%
4765 {\hskip\csuse{Xendinplaceoflemmaseparator@#4}\relax}%
4766 {\nobreak%
4767 \hskip\csuse{Xendbeforelemmaseparator@#4}%
4768 \csuse{Xendlemmaseparator@#4}%
4769 \hskip\csuse{Xendafterlemmaseparator@#4}%
4770 \relax%
4771 }%
4772 \csuse{Xendwrapcontent@#4}{#3}%
4773 \nottoggle{Xendparagraph@#4}{\par}{}%
4774 \def\do##1{%
4775 \setkeys[mac]{falsefootnoteoption}{##1}%
4776 }%
4777 \notblank{#5}{\docsvlist{#5}}{}%
4778 }}%
4779

```

```

4780 \let\l@d@section=\@gobble
4781
4782 %

```

\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.

```

4783 \newcommand{\printlineendnote}[2]{%
4784   \l@dp@rsefootspec#1|{%
4785     \iftoggle{Xendnumberonlyfirstintwolines@#2}{%
4786       \xdef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
4787         \l@dparsedstartsub - \l@dparsedendpage - \l@dparsedendline - \
4788         \l@dparsedendsub}%
4789       }%
4790       {%
4791         \xdef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
4792         \l@dparsedstartsub}%
4793       }%
4794       \ifboolexpr{%
4795         togl {nonum@}%
4796         or togl {Xendnonumber@#2}}%
4797       {%
4798         \iftoggle{Xendnumberonlyfirstinline@#2}%
4799         {\ifcsequal{prevendline#2}%
4800          {\ifcsequal{prevendline#2}{\lineinfo@}%
4801           {%
4802             \csuse{Xendbhookinplaceofnumber@#2}%
4803             \ifcsequal{Xendsymmlinenumber@#2}%
4804             {\hspace{\csuse{Xendinplaceofnumber@#2}}}%
4805             {\printsymlineendnotearea{#2}}}%
4806             \csuse{Xendahookinplaceofnumber@2}%
4807             }%
4808             {\printlineendnotearea{#1}{#2}}}%
4809             {\printlineendnotearea{#1}{#2}}}%
4810             {\printlineendnotearea{#1}{#2}}}%We keep every time line
4811             \csxdef{prevendline#2}{\lineinfo@}%
4812             }%
4813             }%
4814             %

```

```

\printsymlineendnotearea#1\newcommand{\printsymlineendnotearea}[1]{%
4816   \hspace{\csuse{Xendbeforemsymmlinenumber@#1}}%
4817   \csuse{Xendnotenumfont@#1}%
4818   \ifdimequal{\csuse{Xendboxsymmlinenumber@#1}}{\z@}%
4819   {\csuse{Xendsymmlinenumber@#1}}%

```

```

4820     {\hbox to \csuse{Xendboxsymlinenum@#1}%
4821       {\csuse{Xendsymlinenum@#1}\hfill}%
4822     }%
4823     \hspace{\csuse{Xendaftersymlinenum@#1}}%
4824   }%
4825   %

```

\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.)

```

4826 \newcommand{\printlineendnotearea}[2]{%
4827   \csuse{Xendbhooklinenumber@#2}%
4828   \hspace{\csuse{Xendbeforenumber@#2}}%
4829   \bgroup%
4830     \csuse{Xendnotenumfont@#2}%
4831     \ifdimequal{\csuse{Xendboxlinenum@#2}}{0pt}%
4832       {\printendlines#1||\ifledRcol@\@Rlineflag\fi}%
4833       {\leavevmode%
4834         \hbox to \csuse{Xendboxlinenum@#2}%
4835         {%
4836           \IfSubStr{RC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}{}%
4837           \printendlines#1||\ifledRcol@\@Rlineflag\fi%
4838           \IfSubStr{LC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}{}%
4839         }}%
4840     \egroup%
4841     \hspace{\csuse{Xendafternumber@#2}}%
4842     \csuse{Xendahooklinenumber@#2}%
4843   }%
4844   %

```

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 4845 \newcommandx{\toendnotes}[2][1,usedefault]{%
\toendnotes* 4846   \ifboolexpr{bool{numbering} or bool{numberingR}}{%
4847     \def\do##1{%
4848       \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname%
4849       {\unexpanded{#2}\@percentchar}%
4850     }%
4851     \ifstrempy{#1}%
4852       {\dolistloop{\@series}}%

```

```

4853     {\docsvlist{#1}}%
4854   }\led@err@toendnotes@outsidenumbering}%
4855 }%
4856 \WithSuffix\newcommandx\toendnotes*[2][1,usedefault]{%
4857   \ifboolexpr{bool{numbering} or bool{numberingR}}{%
4858     \def\do##1{%
4859       \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname%
4860       {#2\@percentchar}%
4861     }%
4862     \ifstrempy{#1}%
4863     {\dolistloop{\@series}}%
4864     {\docsvlist{#1}}%
4865   }\led@err@toendnotes@outsidenumbering}%
4866 }%
4867 %

```

XIX.2.2 Printing endnotes

`\doendnotes` `\doendnotes` is the command you use to print one series of endnotes; it takes one argument: the series letter of the note series you want to print. `\Xendinsertsep@` is set to true at the first note of the series, and to false at the last one.

```

4868 \newif\ifXendinsertsep@%
4869 \newcommand*{\doendnotes}[1]{%
4870   \l@dend@close{#1}%
4871   \begingroup
4872     \csxdef{prevpagenum@#1}{}%
4873     \csxdef{prevpagerange@#1}{}%
4874     \makeatletter
4875     \expandafter\let\csname #1end\endcsname=\endprint
4876     \input\l@auxdir\jobname.#1end%
4877     \global\Xendinsertsep@false%
4878   \endgroup}
4879 %

```

`\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.

```

4880 \newcommand*{\doendnotesbysection}[1]{%
4881   \l@dend@close{#1}%
4882   \csxdef{prevpagenum@#1}{}%
4883   \csxdef{prevpagerange@#1}{}%
4884   \global\expandafter\advance\csname #1end@bysection\endcsname by 1%
4885   \begingroup%
4886     \makeatletter%
4887     \def\l@d@section##1{%

```



```

4888 \ifnumequal{##1}{\csname #1end@bysection\endcsname}%
4889 {\cslet{#1end}{\endprint}}}%
4890 {\cslet{#1end}{\@gobbleseven}}}%
4891 }%
4892 \input\l@auxdir\jobname.#1end%
4893 \global\Xendinsertsep@false%
4894 \endgroup%
4895 }%
4896 %

```

We close now the conditional period, which depends on `\ifnoend@`, because the following commands can be used by other commands than those specific to endnotes.

```

4897 }%
4898 %

```

`\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.

```

4899 \newcommand*{\setprintendlines}[6]{%
4900 \l@d@pnumfalse \l@d@dashfalse
4901 \ifnum#4=#1 \else
4902 \l@d@pnumtrue
4903 \l@d@dashtrue
4904 \fi
4905 %

```

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.

```

4906 \ifl@d@pnum \l@d@elintrue \else \l@d@elinfalse \fi
4907 \ifnum#2=#5 \else
4908 \l@d@elintrue
4909 \l@d@dashtrue
4910 \fi
4911 %

```

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

```

4912 \l@d@ssubfalse
4913 \ifnum#3=0 \else
4914 \l@d@ssubtrue
4915 \fi
4916 %

```

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.

```

4917 \l@d@eslfalse
4918 \ifnum#6=0 \else
4919     \ifnum#6=#3
4920         \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
4921     \else
4922         \l@d@esltrue
4923         \l@d@dashtrue
4924     \fi
4925 \fi%
4926 %

4927 \ifl@d@dash%
4928     \ifboolexpr{togl{fulllines@} or test{\ifcempty{Xendtwolines@}
@currentseries}}}%
4929     {%
4930     {%
4931     \setistwofollowinglines{#1}{#2}{#4}{#5}%
4932     \ifboolexpr{%
4933         (%
4934             togl {Xendtwolinesbutnotmore@\@currentseries}%
4935             and not%
4936             (%
4937                 bool {istwofollowinglines@}%
4938             )%
4939         )%
4940         or%
4941         (%
4942             (not test{\ifnumequal{#1}{#4}})%
4943             and togl{Xendtwolinesonlyinsamepage@\@currentseries}%
4944         )%
4945     }%
4946     }%
4947     {%
4948     \l@d@dashfalse%
4949     \l@d@Xtwolinesttrue%
4950     \l@d@elinfalse%
4951     \l@d@eslfalse%
4952     \ifcempty{Xendmoreethantwolines@\@currentseries}%
4953     {%
4954     {\ifistwofollowinglines@\else%
4955         \l@d@Xmoreethantwolinesttrue%
4956     \fi%
4957     }%
4958     }%
4959     }%
4960 \fi%
4961 %

```

End of `\setprintendlines`.

```
4962 }%
4963 %
```

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

```
4964 \def\printendlines#1|#2|#3|#4|#5|#6|#7|#8|{%
4965   \begingroup
4966   \setprintendlines{#1}{#2}{#3}{#4}{#5}{#6}%
4967   %
```

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.

```
4968   \ifdimequal{\csuse{Xendboxstartlinenum@\@currentseries}}{0pt}%
4969     {\bgroup}%
4970     {\leavevmode\hbox to \csuse{Xendboxstartlinenum@\@currentseries}\bgroup
\hfill}}%
4971   %
```

Then, print the starting page number-

```
4972   \ifboolexpr{%
4973     (%
4974       test{\ifcsstring{prevpagenum@\@currentseries}{#1}}%
4975       and not%
4976       (togl{Xendpagenumberonlyfirstifsingle@\@currentseries} and bool{
1@d@pnum})}%
4977     )%
4978     or%
4979     (%
4980       test {\ifcsstring{prevpagerange@\@currentseries}{#1-#4}}%
4981     )%
4982   }%
4983   {%
4984     \ifcsempy{Xendsympagenum@\@currentseries}%
4985       {\hspace{\csuse{Xendinplaceofpagenumber@\@currentseries}}}%
4986       {\csuse{Xendsympagenum@\@currentseries}}%
4987   }%
4988   {%
4989     \wrap@edcrossref{\@this@crossref@start}{\printnpnum{#1}}%
4990   }%
4991   %
```

Then, determine what must be printed before the start line.

```
4992   \ifl@d@dash%
4993     \ifl@d@pnum%
```

```

4994 \csuse{Xendlineprefixsingle@\@currentseries}%
4995 \else%
4996 \ifcempty{Xendlineprefixmore@\@currentseries}%
4997 {\csuse{Xendlineprefixsingle@\@currentseries}}%
4998 {\csuse{Xendlineprefixmore@\@currentseries}}}%
4999 \fi%
5000 \else%
5001 \csuse{Xendlineprefixsingle@\@currentseries}%
5002 \fi%
5003 %

```

The print the starting line, followed, if needed, by the side flag and the starting sub line number.

```

5004 \wrap@edcrossref{\@this@crossref@start}{%
5005 \ifledRcol@%
5006 \linenumrepR{#2}%
5007 \else%
5008 \linenumrep{#2}%
5009 \fi%
5010 }%
5011 \iftoggle{Xendlineflag@\@currentseries}{\ifledRcol@\@Rlineflag\fi}{}%
5012 \ifl@d@ssub%
5013 \csuse{Xendsublinesep@\@currentseries}%
5014 \wrap@edcrossref{\@this@crossref@start}{%
5015 \ifledRcol@%
5016 \sublinenumrepR{#3}%
5017 \else%
5018 \sublinenumrep{#3}%
5019 \fi%
5020 }%
5021 \fi%
5022 %

```

Close the box.

```

5023 \egroup%
5024 %

```

Open the box for the ending line number.

```

5025 \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
5026 {\bgroup}%
5027 {\hbox to \csuse{Xendboxendlinenum@\@currentseries}\bgroup}%
5028 %

```

Print the dash + the ending line number, or the line number range symbol.

```

5029 \ifl@d@Xtwolines%
5030 \ifl@d@Xmorethantwolines%
5031 \csuse{Xendmorethantwolines@\@currentseries}%
5032 \else%
5033 \csuse{Xendtwwolines@\@currentseries}%

```

```

5034 \fi%
5035 \else%
5036 \ifl@d@dash%
5037 \ifdefined\linrangesep@%
5038 \linrangesep@%
5039 \else%
5040 \csuse{Xendlinrangeseparator@}\@currentseries}%
5041 \fi%
5042 \fi%
5043 %

```

Print the ending page number.

```

5044 \ifl@d@pnum%
5045 \ifcsstring{prevpagerange@\@currentseries}{#1-#4}%
5046 {%
5047 \ifcsempy{Xendsympagenum@\@currentseries}%
5048 {\hspace{\csuse{Xendingplaceofpagenumber@\@currentseries}}}%
5049 {\csuse{Xendsympagenum@\@currentseries}}}%
5050 }%
5051 {%
5052 \wrap@edcrossref{\@this@crossref@end}\printnpnum{#4}%
5053 }%
5054 \fi%
5055 %

```

Print the ending line number, with, if needed, the line prefix, and followed by the side flag and the subline number.

```

5056 \ifl@d@elin%
5057 \ifl@d@pnum\csuse{Xendlineprefixsingle@\@currentseries}\fi%
5058 \wrap@edcrossref{\@this@crossref@end}{%
5059 \ifledRcol@%
5060 \linenumrepR{#5}%
5061 \else%
5062 \linenumrep{#5}%
5063 \fi%
5064 }%
5065 \iftoggle{Xendlineflag@\@currentseries}{\ifledRcol@\@Rlineflag\fi}{%
5066 \fi%
5067 \ifl@d@esl%
5068 \ifl@d@elin%
5069 \csuse{Xendsublinesep@\@currentseries}%
5070 \fi%
5071 \wrap@edcrossref{\@this@crossref@end}{%
5072 \ifledRcol@%
5073 \sublinenumrepR{#6}%
5074 \else%
5075 \sublinenumrep{#6}%
5076 \fi%
5077 }%
5078 \fi%

```

```
5079 \fi%
5080 %
```

Close the ending line box.

```
5081 \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
5082 {}%
5083 {\hfill}%Prevent underfull hbox
5084 \egroup%
5085 %
```

And, finally, save, if needed, the current page number for the Xendpagenumberonlyfirst hooks.

```
5086 \iftoggle{Xendpagenumberonlyfirst@\@currentseries}%
5087 {\iftoggle{Xendpagenumberonlyfirstintwo@\@currentseries}%
5088 {\csxdef{prevpagerange@\@currentseries}{#1-#4}}%
5089 {\csxdef{prevpagenum@\@currentseries}{#4}}%
5090 }%
5091 {}%
5092 %
```

Now, the end of \printendlines macro.

```
5093 \endgroup%
5094 }%
5095
5096 %
```

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

```
5097 \newcommand*{\printnpnum}[1]{\csuse{Xendbeforepagenumber@\@currentseries}
5098 }#1\csuse{Xendafterpagenumber@\@currentseries}}
5099 %
```

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@.

```
5100 \newcommand{\newseries}[1]{%
5101 \def\do##1{\newseries@{##1}}%
5102 \docsvlist{#1}
5103 }
5104 %
```

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

```

5105 \newcommand{\@series}{}
5106 %

```

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

```

\newseries@07 \newcommand{\newseries@}[1]{
5108 %

```

XX.1 Test if series is still existing

```

5109 \xifinlist{#1}{\@series}{\led@warn@SeriesStillExist{#1}}%
5110 {%
5111 %

```

XX.2 Init specific to reledpar

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

```

5112 \ifdefined\newseries@par%
5113 \newseries@par{#1}%
5114 \fi%
5115 %

```

XX.3 For critical footnotes

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

```

5116 \unless\ifnocritical@
5117 %

```

XX.3.1 Options

```

5118 \newtoggle{Xlineflag@#1}
5119 \newtoggle{Xparindent@#1}
5120 \newtoggle{Xlemmadisablefontselection@#1}
5121 \csgdef{Xwrapcontent@#1}{}%
5122 \csgdef{Xbeforeinserting@#1}{}%
5123 \csgdef{Xhangindent@#1}{Opt}%
5124 \csgdef{Xragged@#1}{}%
5125 \csgdef{Xhsizetwocol@#1}{0.45 \hsize}%
5126 \csgdef{Xhsizethreecol@#1}{.3 \hsize}%
5127 \csgdef{Xcolalign@#1}{\raggedright}%
5128 \csgdef{Xnotenumfont@#1}{\normalfont}%
5129 \csgdef{Xnotefontsize@#1}{\footnotesize}%
5130 \csgdef{Xhooknote@#1}{}%

```

```

5131 \csgdef{Xbhookgroup@#1}{}%
5132
5133 \csgdef{Xboxlinenum@#1}{Opt}%
5134 \csgdef{Xboxlinenumalign@#1}{L}%
5135
5136 \csgdef{Xboxstartlinenum@#1}{Opt}%
5137 \csgdef{Xboxendlinenum@#1}{Opt}%
5138
5139 \csgdef{Xboxsymlinenum@#1}{Opt}%
5140 \newtoggle{Xnumberonlyfirstinline@#1}%
5141 \newtoggle{Xgroupbyline@#1}%
5142 \newtoggle{Xgroupbylineseparetwolines@#1}%
5143 \newtoggle{Xnumberonlyfirstintwolines@#1}%
5144 \csgdef{Xtwolines@#1}{}%
5145 \csgdef{Xmorethantwolines@#1}{}%
5146 \csgdef{Xsublinesep@#1}{\fullstop}%
5147 \csgdef{Xpagelinesep@#1}{\csname Xsublinesep@#1\endcsname}%for
backward compatibility, call Xsublinesep@#1
5148 \newtoggle{Xtwolinesbutnotmore@#1}%
5149 \newtoggle{Xtwolinesonlyinsamepage@#1}%
5150 \newtoggle{Xonlypstart@#1}%
5151 \newtoggle{Xpstarteverytime@#1}%
5152 \newtoggle{Xpstart@#1}%
5153 \newtoggle{Xstanza@#1}%
5154 \csgdef{Xstanzaseparator@#1}{}%
5155 \csgdef{Xsymlinenum@#1}{}%
5156 \newtoggle{Xnonumber@#1}%
5157 \csgdef{Xbeforenumber@#1}{Opt}%
5158 \csgdef{Xtxtbeforenumber@#1}{}%
5159 \csgdef{Xafternumber@#1}{0.5em}%
5160 \newtoggle{Xnonbreakableafternumber@#1}%
5161 \csgdef{Xbeforesymlinenum@#1}{\csuse{Xbeforenumber@#1}}%
5162 \csgdef{Xaftersymlinenum@#1}{\csuse{Xafternumber@#1}}%
5163 \csgdef{Xinplaceofnumber@#1}{1em}%
5164 \global\cslet{Xlemmaseparator@#1}{\rbracket}%
5165 \csgdef{Xbeforelemmaseparator@#1}{0em}%
5166 \csgdef{Xafterlemmaseparator@#1}{0.5em}%
5167 \csgdef{Xinplaceoflemmaseparator@#1}{1em}%
5168 \csgdef{Xbeforenotes@#1}{1.2em \@plus .6em \@minus .6em}
5169 \csgdef{Xafterrule@#1}{Opt}
5170
5171 \csgdef{Xtxtbeforenotes@#1}{%
5172 \newtoggle{Xtxtbeforenotes@#1@typeset}}%Not directly used by user,
but internal
5173 \newtoggle{Xtxtbeforenotesonlyonce@#1}%
5174
5175 \csgdef{Xmaxhnotes@#1}{0.8\vsizer}
5176 \newtoggle{Xnoteswidthliketwocolumns@#1}%
5177 \csgdef{Xparafootsep@#1}{}%
5178 \csgdef{Xafternote@#1}{1em plus.4em minus.4em}

```



```

5179 \csgdef{Xlinrangeseparator@#1}{\endashchar}%
5180
5181 \csgdef{Xlemmafont@#1}{}%
5182 \csgdef{Xwraplemma@#1}{%
5183 \csgdef{Xwidth@#1}{\hsize}%
5184 %

```

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

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

```

5185 \expandafter\newinsert\csname #1footins\endcsname%
5186 \unless\ifnoledgroup%
5187 \expandafter\newinsert\csname mp#1footins\endcsname%
5188 \fi%
5189 %

```

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.

```

5190 \global\newbool{parapparatus@}{\expandafter\newcommand\expandafter
5191 *}{\expandafter\newcommand}\csname #1footnote\endcsname[2][]{%
5192 \if@edtext@secondarg%
5193 \ifledRcol%
5194 \ifcsstring{Xonlyside@#1}{L}{\
5195 led@error@note@called@onrightside{#1footnote}}}%
5196 \else%
5197 \ifcsstring{Xonlyside@#1}{R}{\
5198 led@error@note@called@onleftside{#1footnote}}}%
5199 \fi%
5200 \beginngroup%
5201 \newcommand{\content}{##2}%
5202 \ifnumberedpar%
5203 \ifledRcol%
5204 \ifluatex%
5205 \footnotelang@lua[R]%
5206 \fi%
5207 \@ifundefined{xpg@main@language}%if polyglossia
5208 {}%
5209 {\footnotelang@poly[R]}%
5210 \footnoteoptions@{R}{##1}{true}%
5211 \xright@appenditem{%
5212 \ifbool{indtl@innote}%
5213 {\unexpanded{\let\index\nindex}}%
5214 {}%
5215 \ifbool{indtl@notenumber}%
5216 {\unexpanded{\let\index\nindex}}%There is no note
5217 ...number so
5218 {}%

```

```

5215         \noexpand\Xnote@true%
5216         \noexpand\prepare@Xprenotes{#1}%
5217         \noexpand\prepare@edindex@fornote{\l@d@nums}%
5218         \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.
5219         \noexpand\setcounter{stanzaR}{\the\c@stanzaR}%Save
stanzaR counter for footnote
5220         \unexpanded{\def\@this@crossref@start}{\theedtext:
start}%
5221         \unexpanded{\def\@this@crossref@end}{\theedtext:end}%
5222         \expandonce{\@beforeinsertofthisedtext}% Internal for
now, no reason to make it public
5223         \noexpand\csuse{v#1footnote}{#1}%
5224         {\l@d@nums}{\expandonce\@tag}{\expandonce\content}}
%
5225         \noexpand\Xnote@false%
5226         \unexpanded{\advance\@edindex@fornote@m@one}%
5227         \unexpanded{\undef\@this@crossref@start}%
5228         \unexpanded{\undef\@this@crossref@end}%
5229         \ifbool{indtl@innote}%
5230         {\unexpanded{\let\index\orig@@index}}%
5231         {}%
5232         \ifbool{indtl@notenumber}%
5233         {\unexpanded{\let\index\orig@@index}}%
5234         {}%
5235         }\to\inserts@listR
5236         \footnoteoptions@{R}{#1}{false}%
5237         \global\advance\insert@countR \@ne%
5238     \else%
5239         \ifluatex%
5240         \footnotelang@lua%
5241         \fi%
5242         \@ifundefined{xpg@main@language}%if polyglossia
5243         {}%
5244         {\footnotelang@poly}%
5245         \footnoteoptions@{L}{#1}{true}%
5246         \xright@appenditem{%
5247         \ifbool{indtl@innote}%
5248         {\unexpanded{\let\index\nindex}}%
5249         {}%
5250         \ifbool{indtl@notenumber}%
5251         {\unexpanded{\let\index\nindex}}%There is no note
...number so
5252         {}%
5253         \noexpand\Xnote@true%
5254         \noexpand\prepare@Xprenotes{#1}%
5255         \noexpand\prepare@edindex@fornote{\l@d@nums}%

```

```

5256 \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.
5257 \ifl@dpairing%
5258 \noexpand\setcounter{stanzaL}{\the\c@stanzaL}%Save
stanzaR counter for footnote
5259 \fi%
5260 \unexpanded{\def\@this@crossref@start}{\theedtext:
start}%
5261 \unexpanded{\def\@this@crossref@end}{\theedtext:end}%
5262 \expandonce{\@beforeinsertofthisedtext}%Internal for
now, no reason to make it public
5263 \noexpand\csuse{v#1footnote}%
5264 {#1}%
5265 {{\l@d@nums}{\expandonce\@tag}{\expandonce\content
}}%
5266 \unexpanded{\undef\@this@crossref@start}%
5267 \unexpanded{\undef\@this@crossref@end}%
5268 \noexpand\Xnote@false%
5269 \unexpanded{\advance\@edindex@fornote@m@ne}%
5270 \ifbool{indtl@innote}%
5271 {\unexpanded{\let\index\orig@index}}%
5272 {}%
5273 \ifbool{indtl@notenumber}%
5274 {\unexpanded{\let\index\orig@index}}%
5275 {}%
5276 }\to\inserts@list
5277 \global\advance\insert@count \@ne%
5278 \footnoteoptions@{L}{##1}{false}%
5279 \fi
5280 \else
5281 \csuse{v#1footnote}{#1}{0|0|0|0|0|0|0|0}{##1}%
5282 \fi%
5283 \endgroup%
5284 \else%
5285 \led@err@FootnoteNotInSecondArgEdtext{#1}%
5286 \fi%
5287 \ignorespaces%
5288 }
5289 %

```

Create counter used to determine on which page the previous note was called.

```

5290 \expandafter\newcount\csname #1prevpage@num\endcsname%
5291 \expandafter\newcount\csname #1prevpage@numR\endcsname%
5292 %

```

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

```

5293 \global\csletcs{#1@footnote}{#1footnote}
5294 %

```

XX.3.4 Set standard display

```
5295 \Xarrangement@normal{#1}%
5296 %
```

End of for critical footnotes.

```
5297 \fi
5298 %
```

XX.4 For familiar footnotes

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

```
5299 \unless\ifnofamiliar@
5300 %
```

XX.4.1 Options

```
5301 \newtoggle{parindentX@#1}
5302 \csgdef{wrapcontentX@#1}{}%
5303 \csgdef{hangindentX@#1}{Opt}%
5304 \csgdef{beforeinsertingX@#1}{}%
5305 \csgdef{raggedX@#1}{}%
5306 \csgdef{hsizetwocolX@#1}{0.45 \hsize}%
5307 \csgdef{hsizethreecolX@#1}{.3 \hsize}%
5308 \csgdef{colalignX@#1}{\raggedright}%
5309 \csgdef{notenumfontX@#1}{\normalfont}%
5310 \csgdef{notefontsizeX@#1}{\footnotesize}%
5311 \csgdef{bhooknoteX@#1}{}%
5312 \csgdef{bhookgroupX@#1}{}%
5313 \csgdef{afterruleX@#1}{Opt}
5314 \csgdef{beforenotesX@#1}{1.2em \@plus .6em \@minus .6em}
5315 \csgdef{maxhnotesX@#1}{0.8\vsizex}%
5316 \newtoggle{noteswidthliketwocolumnsX@#1}%
5317 \csgdef{parafootsepX@#1}{}%
5318 \csgdef{afternoteX@#1}{1em plus.4em minus.4em}
5319 \csgdef{widthX@#1}{\hsizex}%
5320 \csgdef{txtbeforenotesX@#1}{}%
5321 \newtoggle{txtbeforenotesX@#1@typeset}%Not directly used by user,
but internal
5322 \newtoggle{txtbeforenotesonlyonceX@#1}%
5323 % End of for familiar footnotes.
5324 % \subsubsection{Create inserts, needed to add notes in foot}
5325 % As regards inserts, see chapter 15 of the TeXBook by D. Knuth.
5326 % \begin{macrocode}
5327 \expandafter\newinsert\csname footins#1\endcsname%
5328 \unless\ifnoledgroup@%
5329 \expandafter\newinsert\csname mpfootins#1\endcsname%
5330 \fi%
```

5331 %

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.

```
5332
5333 \global\expandafter\newcommand\csname footnote#1\endcsname[1]{%
5334 \begingroup%
5335 \prepare@prenotesX{#1}%
5336 \newcommand{\content}{##1}%
5337 %
```

If we use the \csquotes package, we reset quotation level.

```
5338 \ifdefined\csq@qllevel%
5339 \csq@qllevel=0\relax%
5340 \fi%
5341 %
```

If we are preparing parallel typesetting, we cannot just increase the footnote counter. Read reledpar's handbook about that (V.1.2 p. 53).

```
5342 \global\expandafter\advance\csname footnote#1@reading\
endcsname by \@ne%
5343 \ifboolexpr{bool{!l@dpairing} or bool{!l@dprintingpages} or
bool{!l@dprintingcolumns}}{%
5344 \ifcsdef{footnote#1reading\the\csname footnote#1@reading\
endcsname=typeset}%
5345 {\setcounter{footnote#1}{\csuse{footnote#1reading\the\
csname footnote#1@reading\endcsname=typeset}}}%
5346 {\setcounter{footnote#1}{\the\csname footnote#1@reading
\endcsname}}}%
5347 }{%
5348 \stepcounter{footnote#1}%
5349 }%
5350 %
```

We also have to check consistency with \onlysideX setting.

```
5351 \ifledRcol%
5352 \ifcsstring{onlysideX@#1}{L}{\
led@error@note@called@onrightside{footnote#1}}}%
5353 \else%
5354 \ifcsstring{onlysideX@#1}{R}{\
led@error@note@called@onleftside{footnote#1}}}%
5355 \fi%
5356 %
```

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

```
5357 \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
5358 \nottoggle{nomk@}%Nomk is set to true when using \
footnoteXnomk with \parpackage
```

```

5359         {\csuse{@footnotemark#1}}%
5360         {}%
5361     \ifluatex%
5362         \xdef\footnote@luatextextdir{\the\textdir}%
5363         \xdef\footnote@luatexpardir{\the\pardir}%
5364     \fi%
5365     \if@ledgroup%
5366         \led@set@index@fornote{#1}%
5367     \fi%
5368     \csuse{vfootnote#1}{#1}{\expandonce\content}\m@mmf@prepare%
5369     \ifbool{indtl@innote}%
5370         {\let\index\orig@@index}%
5371         {}%
5372     \ifbool{indtl@notenumber}%
5373         {\let\index\orig@@index}%
5374         {}%
5375     \endgroup%
5376 }
5377 %

```

Then define the counters. The \TeX 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. 53).

```

5378     \newcounter{footnote#1}
5379     \global\expandafter\renewcommand\csname thefootnote#1\endcsname{\
arabic{footnote#1}}
5380     \expandafter\newcount\csname footnote#1@reading\endcsname%
5381 %

```

Create counter used to determine on which page the previous note was called.

```

5382     \expandafter\newcount\csname prevpage#1@num\endcsname%
5383     \expandafter\newcount\csname prevpage#1@numR\endcsname%
5384 %

```

Add `\let\footnoteX\@gobble` to `\no@expands`.

```

5385     \expandafter\gappto\expandafter\no@expands\expandafter{\expandafter\
let\csname footnote#1\endcsname\@gobble}%
5386 %

```

And now, define `\footnoteXmark` and `\footnoteXtext`, equivalent to classical `\footnotemark` and `\footnotetext`.

```

5387     \expandafter\newcommand\csname footnote#1mark\endcsname{%
5388         \begingroup%
5389         \prepare@prenotesX{#1}%
5390         \stepcounter{footnote#1}%
5391         \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
5392         \csuse{@footnotemark#1}%
5393         \m@mmf@prepare%

```

```

5394     \endgroup%
5395   }%
5396   \expandafter\newcommand\csname footnote#1text\endcsname[1]{%
5397     \begingroup%
5398     \csuse{vfootnote#1}{#1}{\expandonce{##1}}%
5399     \endgroup%
5400   }%
5401 %

```

Do not forget to initialize the series.

```

5402   \arrangementX@normal{#1}%
5403   \fi
5404 %

```

XX.5 The endnotes

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

```

5405   \unless\ifnoend@
5406 %

```

XX.5.1 The auxiliary file

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

```

5407   \expandafter\newwrite\csname l@d@#1end\endcsname%
5408   \expandafter\newif\csname ifl@dend@#1\endcsname%
5409 %

```

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.

```

5410
5411   \global\expandafter\newcommandx\csname #1endnote\endcsname[2][1,
usedefault]{%
5412     \bgroup%
5413     \newlinechar='40%
5414     \global\@noneed@Footnotetrue%
5415     \newcommand{\content}{##2}%
5416     \stepcounter{labidx}%
5417     \expandafter\immediate\expandafter\write\csname l@d@#1end\
endcsname{%

```

```

5418 \unexpanded{\def\sw@list@inedtext}{\expandafter\unexpanded\
expandafter{\sw@inthisedtext}}\@percentchar\space%Explicit space, to add a
linebreak in the output file
5419 \expandafter\string\csname #1end\endcsname%
5420 {\ifnumberedpar@l@d@nums\fi}%
5421 {\ifnumberedpar@expandonce\@tag\fi}%
5422 {\expandonce\content}%
5423 {\#1}%
5424 {\unexpanded{##1}}%
5425 {\ifledRcol R\else L\fi}%
5426 {\theedtext}%
5427 \@percentchar%
5428 }%
5429 \egroup%
5430 \ignorespaces%
5431 }%
5432 %

```

XX.5.3 Tools

The `\Xtoendnotes` command inserts any arbitrary content into the endnote file. It is an alias of the more generalist `\addtoendnotes`

```

5433 \global\expandafter\newcommand\csname #1toendnotes\endcsname[1]{%
5434 \toendnotes[#1]{##1}%
5435 }%
5436
5437 \expandafter\WithSuffix\expandafter\newcommand\csname #1toendnotes\
5438 endcsname*[1]{%
5439 \toendnotes*[#1]{##1}%
5440 }%
5441
5442 %

```

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.

```

5443
5444 \global\cslet{#1end}{\@gobbleseven}
5445 %

```

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

```

5446 \global\expandafter\newcount\csname #1end@bysection\endcsname%
5447 %

```


XX.5.5 The options

```

5448 \csgdef{Xendwraplemma@#1}{%
5449 \csgdef{Xendwrapcontent@#1}{}%
5450 \csgdef{Xendtwolines@#1}{}%
5451 \csgdef{Xendmorethantwolines@#1}{}%
5452 \newtoggle{Xendtwolinesbutnotmore@#1}{}%
5453 \newtoggle{Xendtwolinesonlyinsamepage@#1}{}%
5454 \newtoggle{Xendlemmadisablefontselection@#1}{}%
5455 \csgdef{Xendnotenumfont@#1}{\normalfont}%
5456 \csgdef{Xendnotefontsize@#1}{\footnotesize}%
5457 \csgdef{Xendbhooknote@#1}{}%
5458
5459 \csgdef{Xendsublinesep@#1}{\fullstop}%
5460
5461 \csgdef{Xendbeforenumber@#1}{Opt}
5462 \csgdef{Xendafternumber@#1}{0.5em}
5463
5464 \csgdef{Xendboxlinenum@#1}{Opt}%
5465 \csgdef{Xendboxlinenumalign@#1}{L}%
5466
5467 \csgdef{Xendboxstartlinenum@#1}{Opt}%
5468 \csgdef{Xendboxendlinenum@#1}{Opt}%
5469
5470 \csgdef{Xendlemmaseparator@#1}{}%
5471 \csgdef{Xendbeforelemmaseparator@#1}{0em}%
5472 \csgdef{Xendafterlemmaseparator@#1}{0.5em}%
5473 \csgdef{Xendinplaceoflemmaseparator@#1}{0.5em}%
5474
5475 \newtoggle{Xendparagraph@#1}%
5476 \csgdef{Xendafternote@#1}{1em plus.4em minus.4em}%
5477 \csgdef{Xendsep@#1}{}%
5478
5479 \csgdef{Xendinplaceofnumber@#1}{Opt}%
5480 \newtoggle{Xendnonumber@#1}%
5481
5482 \csgdef{Xendhangindent@#1}{Opt}%
5483 \newtoggle{Xendnumberonlyfirstinline@#1}%
5484 \newtoggle{Xendnumberonlyfirstintwolines@#1}%
5485
5486 \csgdef{Xendbeforesymlinenum@#1}{\csuse{Xendbeforenumber@#1}}%
5487 \csgdef{Xendaftersymlinenum@#1}{\csuse{Xendafternumber@#1}}%
5488 \csgdef{Xendsymlinenum@#1}{}%
5489 \csgdef{Xendboxsymlinenum@#1}{Opt}%
5490
5491 \csgdef{Xendbhooklinenumber@#1}{}%
5492 \csgdef{Xendehooklinenumber@#1}{}%
5493 \csgdef{Xendbhookinplaceofnumber@#1}{}%
5494 \csgdef{Xendehookinplaceofnumber@#1}{}%
5495

```

```

5496 \csgdef{Xendlinrangeseparator@#1}{\endashchar}%
5497
5498 \csgdef{Xendbeforepagenumber@#1}{p.}%
5499 \csgdef{Xendafterpagenumber@#1}{) }%
5500 \csgdef{Xendlineprefixsingle@#1}{}%
5501 \csgdef{Xendlineprefixmore@#1}{}%
5502
5503 \newtoggle{Xendlineflag@#1}
5504
5505 \csgdef{Xendlemmafont@#1}{}%
5506
5507 \newtoggle{Xendpagenumberonlyfirst@#1}%
5508 \newtoggle{Xendpagenumberonlyfirstifsingle@#1}%
5509 \newtoggle{Xendpagenumberonlyfirstintwo@#1}%
5510 \csgdef{Xendsympagenum@#1}{}%
5511 \csgdef{Xendinplaceofpagenumber@#1}{0pt}%
5512
5513 %

```

End of endnotes declaration

```

5514 \fi%
5515 %

```

Dump series in \@series

```

5516 \listxadd{\@series}{#1}
5517 }
5518 }% End of \newseries
5519 %

```

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

```

5520 \expandafter\newseries\expandafter{\default@series}
5521 %

```

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.

```

5522 \newcommand{\seriesatbegin}[1]{%
5523   \StrDel{\@series}{#1}[\@series]%
5524   \edef\@new{%
5525     \listxadd{\@new}{#1}%
5526     \listxadd{\@new}{\@series}%
5527   \xdef\@series{\@new}%
5528 }
5529 %

```

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

```

5530 \newcommand{\seriesatend}[1]{%
5531   \StrDel{\@series}{#1}[\@series]%
5532   \edef\@new{%
5533     \listadd{\@new}{\@series}%
5534     \listadd{\@new}{#1}%
5535     \xdef\@series{\@new}%
5536   }
5537   %

```

XXI.2 Test series order

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

```

5538 \newcommand{\ifseriesbefore}[4]{%
5539   \StrPosition{\@series}{#1}[\@first]%
5540   \StrPosition{\@series}{#2}[\@second]%
5541   \ifnumgreater{\@second}{\@first}{#3}{#4}%
5542 }
5543 %

```

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}{%
5545   \ifempty{\@series}%
5546   {\xdef\@firstseries{}}%
5547   {\StrChar{\@series}{1}[\@firstseries]}%
5548 }%
5549 %

```

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`.

```

5550 \newcommandx{\settoggle@series}[5][4,5,usedefault]{%
5551   \def\do##1{%
5552     \global\settoggle{#2@##1}{#3}%
5553     \ifstrequal{#4}{critical}{
5554       \csuse{Xarrangement@}\csuse{series@display##1}}{##1}%
5555     }{}
5556     \ifstrequal{#4}{familiar}{
5557       \csuse{arrangementX@}\csuse{series@displayX##1}}{##1}%
5558     }{}
5559   }%
5560   \ifstreempty{#1}{%
5561     \dolistloop{\@series}%
5562     \ifstreempty{#5}{}{%
5563       \docsvlist{#5}%
5564     }
5565   }%
5566   {%
5567     \docsvlist{#1}%
5568   }%
5569 }
5570 %

```

`\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`.

```

5571 \newcommandx{\setcommand@series}[5][4,5,usedefault]{%
5572   \def\do##1{
5573     \csgdef{#2@##1}{#3}
5574     \ifstrequal{#4}{critical}{%
5575       \csuse{Xarrangement@}\csuse{series@display##1}}{##1}%
5576     }{}
5577     \ifstrequal{#4}{familiar}{%
5578       \csuse{arrangementX@}\csuse{series@displayX##1}}{##1}%
5579     }{}%
5580   }%
5581   \ifstreempty{#1}{%
5582     \dolistloop{\@series}%
5583     \ifstreempty{#5}{}{%
5584       \docsvlist{#5}
5585     }
5586   }%
5587   {%
5588     \docsvlist{#1}%
5589   }%
5590 }%
5591 %

```

XXI.3.3 Tools to generate options commands

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

```

5592 \newcommandx{\newhookcommand@series}[2][2,usedefault]{%
5593   \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][]{%
5594     \setcommand@series{##1}{#1}{##2}[][#2]%
5595   }%
5596   \ifstreempty{#2}{}{%
5597     \def\do##1{%

```

```

5598 \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname
[1]{%
5599 \csuse{#1}[##1]{###1}%
5600 }%
5601 }%
5602 \docsvlist{#2}%
5603 }%
5604 }
5605 %

```

\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`.

```

5606 \newcommandx{\newhooktoggle@series}[2][2,usedefault]{%
5607 \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={
true},usedefault]{%
5608 \settoggle@series{##1}{#1}{##2}[#2]%
5609 }%
5610 \ifstrepty{#2}{-}{%
5611 \def\do##1{%
5612 \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname{%
5613 \csuse{#1}[##1]%
5614 }%
5615 }%
5616 \docsvlist{#2}%
5617 }%
5618 }
5619 %

```

\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

```

5620 \newcommand{\newhooktoggle@series@reload}[2]{%
5621 \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={
true},usedefault]{%
5622 \settoggle@series{##1}{#1}{##2}[#2]%
5623 }%
5624 }%
5625 %

```

\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.

```

5626 \newcommand{\newhookcommand@series@reload}[2]{%
5627 \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][1]{%
5628 \setcommand@series{##1}{#1}{##2}[#2]%
5629 }%

```

```
5630 }
5631 %
```

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.

```
5632 \unless\ifnocritical@
5633 \newhookcommand@series{Xwrapcontent}%
5634 \newhookcommand@series{Xbeforeinserting}%
5635 \newhookcommand@series{Xlemmafont}%
5636 \newhookcommand@series{Xwraplemma}%
5637 \newhooktoggle@series{Xparindent}
5638 \newhookcommand@series{Xhangindent}
5639 \newhookcommand@series{Xragged}
5640 \newhookcommand@series{Xhsizetwocol}
5641 \newhookcommand@series{Xhsizethreecol}
5642 \newhookcommand@series{Xcolalign}%
5643 \newhookcommand@series{Xnotenumfont}
5644 \newhookcommand@series{Xbhooknote}
5645 \newhookcommand@series@reload{Xbhookgroup}{critical}
5646 \newhookcommand@series{Xboxsymlinenum}%
5647 \newhookcommand@series{Xsymlinenum}
5648 \newhookcommand@series{Xbeforenumber}
5649 \newhookcommand@series{Xtxtbeforenumber}
5650 \newhookcommand@series{Xafternumber}
5651 \newhookcommand@series{Xbeforesymlinenum}
5652 \newhookcommand@series{Xaftersymlinenum}
5653 \newhookcommand@series{Xinplaceofnumber}
5654 \newhookcommand@series{Xlemmaseparator}
5655 \newhookcommand@series{Xbeforelemmaseparator}
5656 \newhookcommand@series{Xafterlemmaseparator}
5657 \newhookcommand@series{Xinplaceoflemmaseparator}
5658 \newhookcommand@series{Xtxtbeforenotes}
5659 \newhooktoggle@series{Xtxtbeforenotesonlyonce}%
5660 \newhookcommand@series@reload{Xafterrule}{critical}
5661 \newhooktoggle@series{Xnumberonlyfirstinline}
5662 \newhooktoggle@series{Xnumberonlyfirstintwolines}
5663 \newhooktoggle@series{Xgroupbyline}%
5664 \newhooktoggle@series{Xgroupbylineseparatetwolines}%
5665 \newhooktoggle@series{Xnonumber}
5666 \newhooktoggle@series{Xpstart}
5667 \newhooktoggle@series{Xpstarteverytime}%
5668
5669 \newhooktoggle@series{Xstanza}%
5670 \newhookcommand@series{Xstanzaseparator}%
5671
5672 \newhooktoggle@series{Xonlypstart}
```

```

5673 \newhooktoggle@series{Xnonbreakableafternumber}
5674 \newhooktoggle@series{Xlemmadisablefontselection}
5675 \newhookcommand@series@reload{Xmaxhnotes}{critical}
5676 \newhookcommand@series@reload{Xbeforenotes}{critical}
5677 \newhooktoggle@series@reload{Xnoteswidthliketwocolumns}{critical}%
5678 \newhookcommand@series@reload{Xnotefontsize}{critical}
5679
5680 \newhookcommand@series{Xboxlinenum}%
5681 \newhookcommand@series{Xboxlinenumalign}%
5682
5683 \newhookcommand@series{Xboxstartlinenum}%
5684 \newhookcommand@series{Xboxendlinenum}%
5685
5686 \newhookcommand@series{Xafternote}%
5687 \newhookcommand@series{Xparafootsep}
5688
5689 \newhookcommand@series@reload{Xwidth}{critical}%
5690
5691 \ifundef{\Xhsize}%
5692 {%
5693   \newcommandx{\Xhsize}[2][1,usedefault]{%
5694     \led@warning@Xhsize@deprecated%
5695     \Xwidth[#1]{#2}%
5696   }%
5697 }%
5698 {}%
5699 \fi
5700 \newhooktoggle@series{Xlineflag}[appref,SEref]
5701 \newhookcommand@series{Xtwolines}[appref,SEref]
5702 \newhookcommand@series{Xmorethantwolines}[appref,SEref]
5703 \newhookcommand@series{Xsublinesep}[appref,SEref,side]%
5704 \newhookcommand@series{Xpagelinesep}[appref,SEref,side]%
5705 \newhooktoggle@series{Xtwolinesbutnotmore}[appref,SEref]
5706 \newhooktoggle@series{Xtwolinesonlyinsamepage}[appref,SEref]
5707 \newhookcommand@series{Xlinerrangeseparator}[appref,SEref]
5708 %

```

XXI.3.5 Options for familiar notes

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

```

5709 \unless\ifnofamiliar@
5710   \newhookcommand@series{wrapcontentX}%
5711   \newhookcommand@series{beforeinsertingX}%
5712   \newhooktoggle@series{parindentX}
5713   \newhookcommand@series{hangindentX}
5714   \newhookcommand@series{raggedX}
5715   \newhookcommand@series{hsizetwocolX}
5716   \newhookcommand@series{hsizethreecolX}

```



```

5717 \newhookcommand@series{colalignX}%
5718 \newhookcommand@series{notenumfontX}
5719 \newhookcommand@series{bhooknoteX}
5720 \newhookcommand@series@reload{bhookgroupX}{familiar}
5721 \newhookcommand@series@reload{beforenotesX}{familiar}
5722 \newhookcommand@series@reload{maxhnotesX}{familiar}
5723 \newhooktoggle@series@reload{noteswidthliketwocolumnsX}{familiar}%
5724 \newhookcommand@series@reload{afterruleX}{familiar}
5725 \newhookcommand@series@reload{notefontsizeX}{familiar}
5726 \newhookcommand@series{afternoteX}
5727 \newhookcommand@series{parafootsepX}
5728 \newhookcommand@series{txtbeforenotesX}%
5729 \newhooktoggle@series{txtbeforenotesonlyonceX}%
5730 \newhookcommand@series@reload{widthX}{familiar}%
5731 \ifundef{\hsizeX}%
5732   {%
5733     \newcommandx{\hsizeX}[2][1,usedefault]{%
5734       \led@warning@hsizeX@deprecated%
5735       \widthX[#1]{#2}%
5736     }%
5737   }%
5738 {}%
5739 \fi
5740 %

```

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.

```

5741 \unless\ifnoend@
5742 \newhookcommand@series{Xendwraplemma}
5743 \newhookcommand@series{Xendwrapcontent}
5744 \newhookcommand@series{Xendnotenumfont}
5745 \newhookcommand@series{Xendlemmafont}%
5746 \newhookcommand@series{Xendbhooknote}
5747
5748 \newhookcommand@series{Xendboxlinenum}%
5749 \newhookcommand@series{Xendboxlinenumalign}%
5750
5751 \newhookcommand@series{Xendboxstartlinenum}%
5752 \newhookcommand@series{Xendboxendlinenum}%
5753
5754 \newhookcommand@series{Xendnotefontsize}
5755 \newhooktoggle@series{Xendlemmadisablefontselection}
5756 \newhookcommand@series{Xendlemmaseparator}
5757 \newhookcommand@series{Xendbeforelemmaseparator}
5758 \newhookcommand@series{Xendafterlemmaseparator}
5759 \newhookcommand@series{Xendinplaceoflemmaseparator}
5760

```

```

5761 \newhookcommand@series{Xendbeforenumber}%
5762 \newhookcommand@series{Xendafternumber}%
5763
5764 \newhooktoggle@series{Xendparagraph}
5765 \newhookcommand@series{Xendafternote}
5766 \newhookcommand@series{Xendsep}
5767
5768 \newhookcommand@series{Xendinplaceofnumber}%
5769 \newhooktoggle@series{Xendnonumber}%
5770
5771 \newhooktoggle@series{Xendnumberonlyfirstinline}%
5772 \newhooktoggle@series{Xendnumberonlyfirstintwolines}%
5773
5774 \newhookcommand@series{Xendsymmlinenumber}%
5775 \newhookcommand@series{Xendbeforesymmlinenumber}%
5776 \newhookcommand@series{Xendaftersymmlinenumber}%
5777 \newhookcommand@series{Xendboxsymmlinenumber}%
5778
5779 \newhookcommand@series{Xendbhooklinenumber}%
5780 \newhookcommand@series{Xendahooklinenumber}%
5781 \newhookcommand@series{Xendbhookinplaceofnumber}%
5782 \newhookcommand@series{Xendahookinplaceofnumber}%
5783
5784 \newhookcommand@series{Xendhangindent}%
5785
5786 \newhooktoggle@series{Xendpagenumberonlyfirst}%
5787 \newhooktoggle@series{Xendpagenumberonlyfirstifsingle}%
5788 \newhooktoggle@series{Xendpagenumberonlyfirstintwo}%
5789 \newhookcommand@series{Xendsympagenumber}%
5790 \newhookcommand@series{Xendinplaceofpagenumber}%
5791
5792 \fi
5793 \newhooktoggle@series{Xendlineflag}[apprefwithpage,SErefwithpage]
5794 \newhookcommand@series{Xendt看olines}[apprefwithpage,SErefwithpage]
5795 \newhookcommand@series{Xendmoreethant看olines}[apprefwithpage,SErefwithpage]
5796 \newhooktoggle@series{Xendt看olinesbutnotmore}[apprefwithpage,SErefwithpage]
5797 \newhooktoggle@series{Xendt看olinesonlyinsamepage}[apprefwithpage,
SErefwithpage]
5798 \newhookcommand@series{Xendlinerangeseparator}[apprefwithpage,SErefwithpage
]
5799 \newhookcommand@series{Xendbeforepagenumber}[apprefwithpage,SErefwithpage,
SErefonlypage]
5800 \newhookcommand@series{Xendafterpagenumber}[apprefwithpage,SErefwithpage]
5801 \newhookcommand@series{Xendlineprefixsingle}[apprefwithpage,SErefwithpage]
5802 \newhookcommand@series{Xendlineprefixmore}[apprefwithpage,SErefwithpage]
5803 \newhookcommand@series{Xendsublinesep}[apprefwithpage,SErefwithpage]
5804
5805 %

```

XXI.4 Hooks for a particular footnote

\newhooktoggle@specific `\newhooktoggle@specific` is a generic command to create boolean hook specific to a note.

```

5806 \newcommand{\newhooktoggle@specific}[1]{%
5807   \newtoggle{#1}%
5808   \define@key[mac]{truefootnoteoption}{#1}[]{\global\settoggle{#1}{true}}%
When enabling footnote option
5809   \define@key[mac]{falsefootnoteoption}{#1}[]{\global\settoggle{#1}{false}
5810   }}
5811 }
```

\newhookarg@specific `\newhookarg@specific` is a generic command to create argumen hook specific to a note.

```

5812 \newcommand{\newhookarg@specific}[1]{%
5813   \define@key[mac]{truefootnoteoption}{#1}{\global\def\linrangesep@{##1}}%
When enabling footnote option
5814   \define@key[mac]{falsefootnoteoption}{#1}{\global\undef\linrangesep@}%
When
5815 }
5816 %
```

And now, we define some hooks specific to a note.

```

5817 \newhooktoggle@specific{fulllines}%
5818 \newhooktoggle@specific{nonum}
5819 \newhooktoggle@specific{nosep}
5820 \newhookarg@specific{linrangesep}
5821 %
```

linrangesep@ `\linrangesep@` is defined by the option `linrangesep` of critical notes to change temporarily the line range separator for a specific line. As we have to define it before typesetting the line and undefine it after, we use the family of `xkeyval` package's key.

```

5822 %
```

\nomk@ `\nomk@` toggle is used by `reledpar` to remove the footnote mark in the text when using `\footnoteXmk`. Read `reledpar` handbook.

```

5823 \newtoggle{nomk}%
5824 %
```

XXI.5 Alias

\Xnolemmaseparator `\Xnolemmaseparator[⟨series⟩]` is just an alias for `\Xlemmaseparator[⟨series⟩]{}`.

```

5825 \newcommandx*{\Xnolemmaseparator}[1][1]{\Xlemmaseparator[#1]}
5826 %
```

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 \TeX 's kernel. Since `reledmac`, we use `etoolbox`'s patching commands instead of overriding. It should provides better compatibility with other package which modify these commands

`\doextrafeet` `\doextrafeet` is the code extending `\@makecol` to cater for the extra `reledmac` feet. We have two categories of extra footnotes. By default, we order the footnote inserts so that the regular footnotes of \TeX are first, then familiar familiar footnotes and finally the critical footnotes.

```

5827 \newcommand*{\l@ddoxtrafeet}{%
5828   \IfStrEq{familiar-critical}{\@fnpos}
5829   {\do@feetX\do@Xfeet}%
5830   {%
5831     \IfStrEq{critical-familiar}{\@fnpos}%
5832     {\do@Xfeet\do@feetX}%
5833     {%
5834       \setbox\@outputbox \vbox{%
5835         \unvbox\@outputbox%
5836         \do@feet@custom@order{\@fnpos}%
5837       }%
5838     }%
5839   }%
5840 }%
5841
5842 %

```

`\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, \TeX could perform an infinite run.

```

5843 \newcommand{\do@feet@custom@order}[2]{%
5844   \def\do##1{%
5845     \edef\@notesseries{\@firstoftwo##1}%
5846     \edef\@notetype{\@secondoftwo##1}%
5847     \ifdefstring{\@notetype}{critical}%
5848       {\csuse{#1append@Xnotes}{\@notesseries}}%
5849       {\ifdefstring{\@notetype}{familiar}%
5850        {\csuse{#1append@notesX}{\@notesseries}}%
5851        {}%
5852      }%
5853   }%
5854   \expandafter\docsvlist\expandafter{#2}%
5855 }%

```

```
5856 %
```

\do@Xfeet \do@Xfeet is the code extending \@makecol to cater to the extra critical feet.

```
5857 \newcommand*\do@Xfeet}{%
5858   \setbox\@outputbox \vbox{%
5859     \unvbox\@outputbox
5860     \op@Xfeet}}
5861 %
```

\@opXfeet 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

```
\append@Xnotes
\print@Xnotes
5862 \newcommand{\append@Xnotes}[1]{%
5863   \ifvoid\csuse{#1footins}\else%
5864     \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@#1}%
5865     \global\advance\skip\csuse{#1footins} by\csuse{Xafterrule@#1}%
5866     \print@Xnotes{#1}%
5867   \fi%
5868 }%
5869 %
```

The normal way to add one series, \print@Xnotes, is replaced by reledpar when using \Pages.

```
5870 \newcommand\print@Xnotes[1]{%
5871   \xdef\@currentseries{#1}%
5872   \csuse{#1footstart}{#1}%
5873   \csuse{#1footgroup}{#1}%
5874 }%
5875 %
```

We print all series of notes by looping on them. We check before printing them that they are not voided.

```
5876 \newcommand*\@opXfeet}{%
5877   \unless\ifnocritical@%
5878     \gdef\firstXseries@{}%
5879     \def\do##1{%
5880       \append@Xnotes{##1}%
5881     }%
5882     \dolistloop{\@series}%
5883   \fi%
5884 }%
5885 %
```

\l@ddodoreinextrafeet \l@ddodoreinextrafeet is the code for catering for the extra footnotes within \@reinserts. We use the same category and ordering as in \l@ddoxtrafeet.

```
5886 \newcommand*\l@ddodoreinextrafeet}{%
5887   \IfStrEq{familiar-critical}{\@fnpos}
```

```

5888     {\@doreinfeetX\X@doreinfeet}%
5889     {%
5890     \IfStrEq{critical-familiar}{\@fnpos}%
5891       {\X@doreinfeet\@doreinfeetX}%
5892       {\@doreinfeetX\X@doreinfeet}%
5893     }%
5894   }
5895
5896   %

```

\X@doreinfeet \X@doreinfeet is the code for catering for the extra critical footnotes within \@reinserts.

```

5897 \newcommand*{\X@doreinfeet}{%
5898   \unless\ifnocritical@%
5899   \def\do##1{%
5900     \ifvoid\csuse{##1footins}\else%
5901       \insert\csuse{##1footins}{\unvbox\csuse{##1footins}}%
5902     \fi}%
5903   \dolistloop{\@series}
5904   \fi%
5905 }
5906
5907 %

```

\print@notesX We have to add all the new kinds of familiar footnotes to the output routine. A macro
\append@notesX which appends the familiar footnotes of one series onto the output routine, also adding
\do@feetX vertical skip before notes.

```

\@doreinfeetX
5908 \newcommand{\append@notesX}[1]{%
5909   \ifvoid\csuse{footins#1}\else%
5910     \global\skip\csuse{footins#1}=\csuse{beforenotesX@#1}%
5911     \global\advance\skip\csuse{footins#1} by\csuse{afterruleX@#1}%
5912     \print@notesX{#1}%
5913   \fi%
5914 }%
5915 %

```

The normal way to print one series of notes. \print@Xnotes is replaced by reledpar when using \Pages.

```

5916 \newcommand\print@notesX[1]{%
5917   \xdef\@currentseries{#1}%
5918   \csuse{footstart#1}{#1}%
5919   \csuse{footgroup#1}{#1}%
5920 }%
5921 %

```

We print all the series of notes by looping on them. We check before printing them that they are not voided.

```

5922 \newcommand*{\do@feetX}{%

```

```

5923 \unless\ifnofamiliar@%
5924 \gdef\firstseriesX@{}%
5925 \setbox\@outputbox \vbox{%
5926 \unvbox\@outputbox%
5927 \def\do##1{%
5928 \append@notesX{##1}%
5929 }%
5930 \dolistloop{\@series}}%
5931 \fi%
5932 }%
5933
5934 \newcommand{\@doreinfeetX}{%
5935 \unless\ifnofamiliar@%
5936 \def\do##1{%
5937 \ifvoid\csuse{footins##1}\else
5938 \insert%
5939 \csuse{footins##1}
5940 {\unvbox\csuse{footins##1}}%
5941 \fi%
5942 }%
5943 \dolistloop{\@series}%
5944 \fi%
5945 }%
5946
5947 %

```

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.)

```

5948 \@ifclassloaded{memoir}{%
5949 %
5950 memoir is loaded so we use memoir's built in hooks.
5951 \g@addto@macro{\m@mddoextrafeet}{\l@dddoextrafeet}%
5952 \g@addto@macro{\m@mddodoreinextrafeet}{\l@ddodoreinextrafeet}%
5953 }{%
5954 %

```

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`.

```

5954 \ifbool{expr}%
5955   test{\@ifpackageloaded{fancyhdr}}%
5956   and test {\ifdef{\latex@makecol}}%
5957 }{%
5958   \patchcmd%
5959     {\latex@makecol}%
5960     {\xdef\@freelist{\@freelist\@midlist}}%
5961     {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
5962     {}%
5963     {\led@error@fail@patch@@@makecol}%
5964 }{%
5965   \patchcmd%
5966     {\@makecol}%
5967     {\xdef\@freelist{\@freelist\@midlist}}%
5968     {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
5969     {}%
5970     {\led@error@fail@patch@@@makecol}%
5971 }%
5972
5973 \patchcmd%
5974   {\@reinserts}%
5975   {\ifvbox}%
5976   {\l@ddodoreinextrafeet\ifvbox}%
5977   {}%
5978   {\led@error@fail@patch@@@reinserts}%
5979 }
5980
5981 %

```

It turns out that `\@doclearpage` also needs modifying.

`\if@led@nofoot` We have to check if there are any leftover feet.

```

5982 \newif\if@led@nofoot
5983
5984 %

```

```

5985 \@ifclassloaded{memoir}{%
5986 %

```

If the memoir class is loaded we hook into its modified `\@doclearpage`.

```

\@mem@extranofeet \g@addto@macro{\@mem@extranofeet}{%
5988   \def\do#1{%
5989     \unless\ifnocritical@%
5990       \ifvoid\csuse{#1footins}\else\@mem@nofootfalse\fi%
5991     \fi%
5992     \unless\ifnofamiliar@%
5993       \ifvoid\csuse{footins#1}\else\@mem@nofootfalse\fi%
5994     \fi%

```



```

5995 }
5996 \dolistloop{\@series}%
5997 }%
5998 }-%
5999 %

```

As memoir is not loaded we have patch \@doclearpage.

```

\@led@testifnofoot%00 \newcommand*{\@led@testifnofoot}{%
\@doclearpage%01 \@led@nofoottrue%
6002 \ifvoid\footins\else%
6003 \@led@nofootfalse%
6004 \fi%
6005 \def\do##1{%
6006 \unless\ifnocritical@%
6007 \ifvoid\csuse{##1footins}\else%
6008 \@led@nofootfalse%
6009 \fi%
6010 \fi%
6011 \unless\ifnofamiliar@%
6012 \ifvoid\csuse{footins##1}\else%
6013 \@led@nofootfalse%
6014 \fi%
6015 \fi%
6016 }%
6017 \dolistloop{\@series}%
6018 }-%
6019
6020 \pretocmd%
6021 {\@doclearpage}%
6022 {\@led@testifnofoot}%
6023 {}%
6024 {\led@error@fail@patch@@doclearpage}%
6025
6026 \patchcmd%
6027 {\@doclearpage}%
6028 {\ifvoid\footins}%
6029 {\if@led@nofoot}%
6030 {}%
6031 {\led@error@fail@patch@@doclearpage}%
6032
6033 }
6034
6035 %

```

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.

```
6036 \list@create{\labelref@list}
6037 %
```

\zz@@@ A convenience macro to zero three labeling counters in one go.

```
6038 \newcommand*{\zz@@@}{000|000|000}% Set three counters to zero in one go
6039
6040 %
```

\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 \TeX write methods for the `.aux` file.

Jesse Billett³⁴ found that the original code could be off by several pages. This version, hopefully cures that, and also allows for non-arabic page numbering.

```
6041 \newcommand*{\edlabel}[1]{%
6042   \leavevmode%
6043   \@bsphack%
6044   \ifbool{expr{bool{ledRcol} or bool{ledRcol@}}}{%
6045     \ifXnote{%
6046       \protected@write\@auxout{%
6047         {\string\l@dmake@labelsR\space\thepage|\l@dparsedstartline|\l@dparsedstartsub||\the\c@pstartR|{#1}}}%
6048         \ifdef{\hypertarget}%
6049           {\Hy@raisedlink{\hypertarget{#1}}}%
6050           {}%
6051       \else%
```

³³The 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.

```

6052 \write\linenum@outR{\string\@lab}%
6053 \ifx\labelref@listR\empty%
6054 \xdef\label@refs{\zz@@@}%
6055 \else%
6056 \gl@p\labelref@listR\to\label@refs%
6057 \fi%
6058 \ifvmode%
6059 \advancelabel@refs%
6060 \fi%
6061 %

```

Use code from the kernel `\label` command to write the correct page number. Also define an `hypertarget` if `hyperref` package is loaded.

```

6062 \protected@write\@auxout{}%
6063 {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstartR
|{\#1}}}%
6064 \ifdef\hypertarget%
6065 {\Hy@raisedlink{\hypertarget{\#1}}}%
6066 {}%
6067 \fi%
6068 }{%
6069 \ifXnote@%
6070 \ifl@dpairing%pstart or pstartL?
6071 \protected@write\@auxout{}%
6072 {\string\l@dmake@labels\space\thepage|\l@dparsedstartline|\
l@dparsedstartsub||\the\c@pstartL|{\#1}}}%
6073 \ifdef\hypertarget%
6074 {\Hy@raisedlink{\hypertarget{\#1}}}%
6075 {}%
6076 \else%
6077 \protected@write\@auxout{}%
6078 {\string\l@dmake@labels\space\thepage|\l@dparsedstartline|\
l@dparsedstartsub||\the\c@pstartL|{\#1}}}%
6079 \ifdef\hypertarget%
6080 {\Hy@raisedlink{\hypertarget{\#1}}}%
6081 {}%
6082 \fi%
6083 \else%
6084 \write\linenum@out{\string\@lab}%
6085 \ifx\labelref@list\empty%
6086 \xdef\label@refs{\zz@@@}%
6087 \else%
6088 \gl@p\labelref@list\to\label@refs%
6089 \fi%
6090 \ifvmode%
6091 \advancelabel@refs%
6092 \fi%
6093 \ifl@dpairing%Pstart or PstartL?
6094 \protected@write\@auxout{}%
6095 {\string\l@dmake@labels\space\thepage|\label@refs|\the\c@pstartL

```

```

|{#1}}%
6096     \ifdef{\hypertarget}%
6097         {\Hy@raisedlink{\hypertarget{#1}{}}}%
6098         {}%
6099     \else%
6100         \protected@write\@auxout{}%
6101         {\string\l@dmake@labels\space\thepage|\label@refs|\the\c@pstart
|{#1}}%
6102         \ifdef{\hypertarget}%
6103             {\Hy@raisedlink{\hypertarget{#1}{}}}%
6104             {}%
6105         \fi%
6106     \fi%
6107 }%
6108 \@esphack}%
6109
6110 %

```

`\advancelabel@refs` In cases where `\edlabel` is the first element in a paragraph, we have a problem with line counts, because line counts change only at the first horizontal box of the paragraph. 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.

```

6111 \newcounter{line}%
6112 \newcounter{subline}%
6113 \newcounter{absline}%
6114 \newcommand{\advancelabel@refs}{%
6115     \setcounter{line}{\expandafter\labelrefsparseline\label@refs}%
6116     \stepcounter{line}%
6117     \setcounter{absline}{\expandafter\labelrefsparseabsline\label@refs}%
6118     \stepcounter{absline}%
6119     \ifsublines%
6120         \setcounter{subline}{\expandafter\labelrefsparsesubline\label@refs}%
6121         \stepcounter{subline}{1}%
6122     \def\label@refs{\theline|\thesubline|\theabsline}%
6123     \else%
6124         \def\label@refs{\theline|0|\theabsline}%
6125     \fi%
6126 }
6127 \def\labelrefsparseline#1|#2|#3{#1}%
6128 \def\labelrefsparsesubline#1|#2|#3{#2}%
6129 \def\labelrefsparseabsline#1|#2|#3{#3}%
6130 %

```

`\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.

```

6131 \newcommand*{\l@dmake@labels}{}
6132 \def\l@dmake@labels#1|#2|#3|#4|#5|#6{%
6133   \expandafter\ifx\csname the@label\csuse{XR@prefix}#6\endcsname%
6134   \relax%
6135   \else%
6136     \led@warn@DuplicateLabel{\csuse{XR@prefix}#6}%
6137   \fi
6138   \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
6139   \expandafter\gdef\csname the@label\csuse{XR@prefix}#6\endcsname
{\#1|#2|#3|#4|#5|\relax}%
6140   \ignorespaces}
6141
6142 %

```

TeX reads the aux file at both the beginning and end of the document, so we have to switch off duplicate label checking after the first time the file is read.

```

6143 \AtBeginDocument{%
6144   \def\l@dmake@labels#1|#2|#3|#4|#5|#6{%
6145   }
6146
6147 %

```

\@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`, `\@n1`, and the `\sub@on` and `\sub@off` commands appearing in the `\linenum@out` file.

TeX uses the page counter for page numbers. However, it appears that this is not the right place to grab the page number. That task is now done in the `\edlabel` macro. This version of `\@lab` appends just the current line and sub-line numbers to `\labelref@list`.

```

6148
6149 \newcommand*{\@lab}{%
6150   \ifledRcol
6151     \xright@appenditem{\linenumr@p{\line@numR}}|%
6152     \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi|\the\
absline@numR}%
6153     \to\labelref@listR
6154   \else
6155     \xright@appenditem{\linenumr@p{\line@num}}|%
6156     \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi|\the\absline@num}%
6157   %

```

```

6157 \to\labelref@list
6158 \fi}
6159 %

```

\applabel \applabel, if called in \edtext will insert automatically both a start and an end label for the current edtext lines.

```

6160 \newcommand*{\applabel}[1]{%
6161 \if@edtext@secondarg%
6162 %

```

Label should not be already defined.

```

6163 \ifcsundef{the@label#1}{%
6164 \csdef{the@label#1}{applabel}%
6165 }%
6166 {%
6167 \led@warn@DuplicateLabel{#1 (applabel)}%
6168 }%
6169 %

```

Parse the \edtext line numbers.

```

6170 \expandafter\l@dp@rsefootspec\l@d@nums|}%
6171 %

```

Use the L^AT_EX standard hack for label.

```

6172 \@bsphack%
6173 %

```

And now, write the data in the auxiliary file.

```

6174 \ifledRcol%
6175 \protected@write\@auxout{%
6176 {\string\l@dmake@labelsR\space\l@dparsedstartpage|\
l@dparsedstartline|\l@dparsedstartsub||\the\c@pstartR|{#1:start}}}%
6177 \ifdef{\hypertarget}%
6178 {\Hy@raisedlink{\hypertarget{#1:start}}}%
6179 }%
6180 \protected@write\@auxout{%
6181 {\string\l@dmake@labelsR\space\l@dparsedendpage|\l@dparsedendline
||\l@dparsedendsub|\the\c@pstartR|{#1:end}}}%
6182 \else%
6183 \ifl@dpairing%pstart or pstartL?
6184 \protected@write\@auxout{%
6185 {\string\l@dmake@labels\space\l@dparsedstartpage|\
l@dparsedstartline|\l@dparsedstartsub||\the\c@pstartL|{#1:start}}}%
6186 \ifdef{\hypertarget}%
6187 {\Hy@raisedlink{\hypertarget{#1:start}}}%
6188 }%
6189 \protected@write\@auxout{%
6190 {\string\l@dmake@labels\space\l@dparsedendpage|\
l@dparsedendline|\l@dparsedendsub||\the\c@pstartL|{#1:end}}}%

```

```

6191         \else%
6192             \protected@write\@auxout{}%
6193             {\string\l@dmake@labels\space\l@dparsedstartpage|\
6194             l@dparsedstartline|\l@dparsedstartsub||\the\c@pstart|{#1:start}}%
6195             \ifdef{\hypertarget}%
6196                 {\Hy@raisedlink{\hypertarget{#1:start}}{}}%
6197                 {}%
6198             \protected@write\@auxout{}%
6199             {\string\l@dmake@labels\space\l@dparsedendpage|\
6200             l@dparsedendline|\l@dparsedendsub||\the\c@pstart|{#1:end}}%
6201         \fi%
6202     \fi%
6203 %

```

Use the \TeX standard hack for label.

```

6202     \@esphack%
6203 %

```

Warning if `\applabel` is called outside of `\edtext`.

```

6204     \else%
6205         \led@warn@AppLabelOutSecondArgEdtext{#1}%
6206     \fi%
6207 %

```

End of `\applabel`

```

6208 }%
6209 %

```

`\edlabels` `\edlabelS` and `\edlabelE` are just used to mark the beginning and the end of a passage.

```

6210 \edlabelE \newcommand{\edlabelS}[1]{%
6211 \edlabelSE \edlabel{#1:start}%
6212 }
6213 \newcommand{\edlabelE}[1]{%
6214 \edlabel{#1:end}%
6215 }
6216 \newcommand{\edlabelSE}[1]{%
6217 \edlabelS{#1}%
6218 \edlabelE{#1}%
6219 }
6220 %

```

`\wrap@edcrossref` `\wrap@edcrossref` is called around all `reledmac` crossref commands, except those which start with `x`. It adds the hyperlink.

```

6221 \newrobustcmd{\wrap@edcrossref}[2]{%
6222 \ifdef{\hyperlink}%
6223     {\hyperlink{#1}{#2}}%
6224     {#2}%
6225 }
6226 %

```

\edpageref If the specified label exists, \edpageref gives its page number.

\xpageref For this reference command, as for the other two, a special version with prefix x is provided for use in places where the command is to be scanned as a number, as in \linenum. These special versions have two limitations: they do not print error messages if the reference is unknown, and they can't appear as the first label or reference command in the file; you must ensure that a \edlabel or a normal reference command appears first, or these x-commands will always return zeros.

TeX already defines a \pageref, so changing the name to \edpageref.

```
6227 \newcommand*\edpageref[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{1}{#1}}}%
6228 \newcommand*\xpageref[1]{\l@dgetref@num{1}{#1}}%
6229 %
6230 %
```

\edlineref If the specified label exists, \lineref gives its line number.

\xlineref

```
6231 \newcommand*\edlineref[1]{%
6232   \l@dref@undefined{#1}%
6233   \wrap@edcrossref{#1}{%
6234     \providetoggle{label@#1@ledRcol}%Required for the first run, when the
label has not yet been parsed on the .aux file
6235     \iftoggle{label@#1@ledRcol}%
6236       {\linenumrepR{\l@dgetref@num{2}{#1}}}%
6237       {\linenumrep{\l@dgetref@num{2}{#1}}}%
6238     \xflagref{#1}%
6239   }%
6240 }%
6241 \newcommand*\xlineref[1]{\l@dgetref@num{2}{#1}}%
6242 %
6243 %
```

\sublineref If the specified label exists, \sublineref gives its sub-line number.

\xsublineref

```
6244 \newcommand*\sublineref[1]{%
6245   \l@dref@undefined{#1}%
6246   \wrap@edcrossref{#1}{%
6247     \providetoggle{label@#1@ledRcol}%Required for the first run, when the
label has not yet been parsed on the .aux file
6248     \iftoggle{label@#1@ledRcol}%
6249       {\sublinenumrepR{\l@dgetref@num{3}{#1}}}%
6250       {\sublinenumrep{\l@dgetref@num{3}{#1}}}%
6251     }%
6252   }%
6253 \newcommand*\xsublineref[1]{\l@dgetref@num{3}{#1}}%
6254 %
6255 %
```

\xabslineref If the specified label exists, \xabslineref gives its absolute line number. That is used usually only by some reledmac internal macros.


```

6256 \newcommand*{\xabslineref}[1]{\l@dgetref@num{4}{#1}}%
6257 %

```

\pstartref If the specified label exists, \pstartref gives its pstart number.

```

\pstartref
6258 \newcommand*{\pstartref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{5}{#1}}}%
6259 \newcommand*{\xpstartref}[1]{\l@dgetref@num{5}{#1}}%
6260 %
6261 %

```

\xflagref \xflagref finds the side flag of any ref defined with \edlabel.

```

6262 \newcommand*{\xflagref}[1]{\l@dgetref@num{6}{#1}}%
6263 %

```

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.

```

6264 \newcommand*{\l@dref@undefined}[1]{%
6265   \expandafter\ifx\csname the@label#1\endcsname\relax
6266     \led@warn@RefUndefined{#1}%
6267   \fi}
6268 %
6269 %

```

\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.

```

6270 \newcommand*{\l@dgetref@num}[2]{%
6271   \expandafter
6272   \ifx\csname the@label#2\endcsname \relax
6273     000%
6274   \else
6275     \expandafter\expandafter\expandafter
6276     \l@dlabel@parse\csname the@label#2\endcsname|#1%
6277   \fi}
6278 %
6279 %

```

`\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`.)

```

6280 \newcommand*{\l@dlabel@parse}{%
6281 \def\l@dlabel@parse#1|#2|#3|#4|#5|#6|#7{%
6282   \ifcase #7%
6283     \or #1%
6284     \or #2%
6285     \or #3%
6286     \or #4%
6287     \or #5%
6288     \or #6%
6289   \fi}
6290 %

```

`\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.

```

6291 \newcommand*{\xxref}[2]{%
6292   {%
6293     \expandafter\ifx\csname the@label#1\endcsname \relax%
6294       \expandafter\let\csname the@@label#1\endcsname\zz@@@%
6295     \else%
6296       \expandafter\def\csname the@@label#1\endcsname{\l@dgetref@num
6297 {1}{#1}|\l@dgetref@num{2}{#1}|\l@dgetref@num{3}{#1}}%
6298     \fi%
6299     \expandafter\ifx\csname the@label#2\endcsname \relax%
6300       \expandafter\let\csname the@@label#2\endcsname\zz@@@%
6301     \else%
6302       \expandafter\def\csname the@@label#2\endcsname{\l@dgetref@num
6303 {1}{#2}|\l@dgetref@num{2}{#2}|\l@dgetref@num{3}{#2}}%
6304     \fi%
6305     \letcs{\@tempa}{the@@label#1}%
6306     \letcs{\@tempb}{the@@label#2}%
6307     \global\appto\@beforeinsertofthisedtext{\def\@this@crossref@start{#1}}%
6308     \global\appto\@beforeinsertofthisedtext{\def\@this@crossref@end{#2}}%
6309     \linenum{\@tempa|
6310     \@tempb}}%

```

```
6309
6310 %
```

`\appref` `\SEref`, `\apprefwithpage`, `\SErefwithpage` and `\SEonlypage` print cross-ref to some start / end lines defined by specific commands. It prints the lines as they should be printed in the apparatus (critical notes for not suffixed versions, endnotes for suffixed versions).

`\SErefwithpage` Here we define hooks similar to some those related to critical footnotes or endnotes.

`\SErefwithpage` So, first declare the default value of the hooks for the pseudo-series. Also declare the internal toggle which are switch by `reledmac`.

```
6311 \def\Xtwolines@appref{}%
6312 \def\Xtwolines@SEref{}%
6313
6314 \def\Xmorethantwolines@appref{}%
6315 \def\Xmorethantwolines@SEref{}%
6316
6317 \def\Xlinerangeseparator@appref{\endashchar}%
6318 \def\Xlinerangeseparator@SEref{\endashchar}%
6319
6320 \def\Xsublinesep@appref{\fullstop}%
6321 \def\Xsublinesep@SEref{\fullstop}%
6322
6323 \def\Xpagelinesep@appref{\fullstop}%
6324 \def\Xpagelinesep@SEref{\fullstop}%
6325
6326
6327 \newtoggle{Xtwolinesbutnotmore@appref}%
6328 \newtoggle{Xtwolinesbutnotmore@SEref}%
6329
6330 \newtoggle{Xtwolinesonlyinsamepage@appref}%
6331
6332 \newtoggle{Xtwolinesonlyinsamepage@SEref}%
6333
6334 \newtoggle{Xlineflag@appref}%
6335 \toggletrue{Xlineflag@appref}%Here exception
6336 \newtoggle{Xlineflag@SEref}%
6337 \toggletrue{Xlineflag@SEref}%Here exception
6338
6339 \def\Xendtwolines@apprefwithpage{}%
6340 \def\Xendtwolines@SErefwithpage{}%
6341
6342 \def\Xendmorethantwolines@apprefwithpage{}%
6343 \def\Xendmorethantwolines@SErefwithpage{}%
6344
6345 \def\Xendlinerangeseparator@apprefwithpage{\endashchar}
6346 \def\Xendlinerangeseparator@SErefwithpage{\endashchar}
6347 \def\Xendlinerangeseparator@SErefonlypage{\endashchar}
6348
```

```

6349 \def\Xendbeforepagenumber@apprefwithpage{p.}%
6350 \def\Xendbeforepagenumber@Serefwithpage{p.}%
6351 \def\Xendbeforepagenumber@SEonlypage{p.}%
6352
6353 \def\Xendafterpagenumber@apprefwithpage{} }%
6354 \def\Xendafterpagenumber@Serefwithpage{} }%
6355
6356
6357 \def\Xendlineprefixsingle@apprefwithpage{}%
6358 \def\Xendlineprefixsingle@Serefwithpage{}%
6359
6360 \def\Xendlineprefixmore@apprefwithpage{}%
6361 \def\Xendlineprefixmore@Serefwithpage{}%
6362
6363 \newtoggle{Xendtwolinesbutnotmore@apprefwithpage}%
6364 \newtoggle{Xendtwolinesbutnotmore@Serefwithpage}%
6365
6366 \def\Xendsublinesep@apprefwithpage{\fullstop}%
6367 \def\Xendsublinesep@Serefwithpage{\fullstop}%
6368
6369 \newtoggle{Xendtwolinesonlyinsamepage@apprefwithpage}%
6370 \newtoggle{Xendtwolinesonlyinsamepage@Serefwithpage}%
6371
6372 \newtoggle{Xendlineflag@apprefwithpage}
6373 \toggletrue{Xendlineflag@apprefwithpage}%Here, exception
6374 \newtoggle{Xendlineflag@Serefwithpage}
6375 \toggletrue{Xendlineflag@Serefwithpage}%Here, exception
6376
6377 %

```

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.

```

6378
6379 \gdef\Xboxstartlinenum@appref{0pt}
6380 \gdef\Xboxstartlinenum@Seref{0pt}
6381
6382 \gdef\Xboxendlinenum@appref{0pt}
6383 \gdef\Xboxendlinenum@Seref{0pt}
6384
6385 \gdef\Xendboxstartlinenum@apprefwithpage{0pt}
6386 \gdef\Xendboxstartlinenum@Serefwithpage{0pt}
6387
6388 \gdef\Xendboxendlinenum@apprefwithpage{0pt}
6389 \gdef\Xendboxendlinenum@Serefwithpage{0pt}
6390
6391 \newtoggle{Xendpagenumberonlyfirst@apprefwithpage}
6392 \newtoggle{Xendpagenumberonlyfirst@Serefwithpage}
6393
6394 \newtoggle{Xendpagenumberonlyfirstifsingle@apprefwithpage}

```

```

6395 \newtoggle{Xendpagenumberonlyfirstifsingle@Serefwithpage}
6396
6397 \newtoggle{Xendpagenumberonlyfirstintwo@apprefwithpage}
6398 \newtoggle{Xendpagenumberonlyfirstintwo@Serefwithpage}
6399
6400 \gdef\Xendsympagenum@apprefwithpage{}
6401 \gdef\Xendsympagenum@Serefwithpage{}
6402
6403 \gdef\Xendinplaceofpagenumber@apprefwithpage{}
6404 \gdef\Xendinplaceofpagenumber@Serefwithpage{}
6405
6406 %

```

Now, declare the default values of \@apprefprefixsingle and \@apprefprefixmore, \@Serefprefix, \@Serefprefixmore and the commands which defines them.

```

6407 \newcommand\@apprefprefixsingle{}%
6408 \newcommand\@Serefprefixsingle{}%
6409
6410 \newcommand\@apprefprefixmore{}%
6411 \newcommand\@Serefprefixmore{}%
6412
6413 \newcommand{\setapprefprefixsingle}[1]{%
6414   \gdef\@apprefprefixsingle{#1}%
6415 }
6416 \newcommand{\setSerefprefixsingle}[1]{%
6417   \gdef\@Serefprefixsingle{#1}%
6418 }
6419
6420 \newcommand{\setapprefprefixmore}[1]{%
6421   \gdef\@apprefprefixmore{#1}%
6422 }
6423 \newcommand{\setSerefprefixmore}[1]{%
6424   \gdef\@Serefprefixmore{#1}%
6425 }
6426
6427 %

```

And not \setSerefonlypageprefixsingle and \setSerefonlypageprefixmore.

```

6428 \newcommand{\setSerefonlypageprefixsingle}[1]{%
6429   \gdef\Serefonlypage@prefixsingle{#1}%
6430 }%
6431 \newcommand{\setSerefonlypageprefixmore}[1]{%
6432   \gdef\Serefonlypage@prefixmore{#1}%
6433 }%
6434 %

```

And now, the main commands: \appref, \apprefwithpage, \Seref and \Serefwithpage. These commands call \reformatted@ and \reformattedwithpage, which calls \printlines and \printendlines . That is why we have previously declared all hooks values tested inside these last commands.

```

6435 \newcommandx{\appref}[2][1,usedefault]{\reformatted@{#1}{#2}{appref}}
6436 \newcommandx{\Seref}[2][1,usedefault]{\reformatted@{#1}{#2}{Seref}}
6437
6438
6439 \newcommandx{\apprefwithpage}[2][1,usedefault]{\reformattedwithpage@
6440 {#1}{#2}{appref}}
6441 \newcommandx{\Serefwithpage}[2][1,usedefault]{\reformattedwithpage@
6442 {#1}{#2}{Seref}}
6443 \newcommandx{\Serefonlypage}[2][1,usedefault]{\reformattedonlypage@
6444 {#1}{#2}{Seref}}
6445
6446 \newcommand{\reformatted@}[3]{%
6447   \def\do##1{%
6448     \setkeys[mac]{truefootnoteoption}{##1}%
6449   }%
6450   \notblank{#1}{\docsvlist{#1}}{%}%
6451   \xdef\@currentseries{#3}%
6452   \ifcsempy{@#3prefixmore}%
6453     {\@apprefprefixsingle}%
6454     {%
6455       \IfEq{\xlineref{#2:start}}{\xlineref{#2:end}}%
6456       {\csuse{@#3prefixsingle}}%
6457       {\csuse{@#3prefixmore}}%
6458     }%
6459   \ifboolexpr{%
6460     test{\ifcsundef{the@label#2:start}}%
6461     or test{\ifcsundef{the@label#2:end}}%
6462   }%
6463   {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}%
6464   {%
6465     \def\@this@crossref@start{#2:start}%
6466     \def\@this@crossref@end{#2:end}%
6467     \printlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:
6468 start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}|\relax|\
6469 xflagref{#2:start}}|}%
6470     \undef\@this@crossref@end%
6471     \undef\@this@crossref@start%
6472   }%
6473   \def\do##1{%
6474     \setkeys[mac]{falsefootnoteoption}{##1}%
6475   }%
6476   \notblank{#1}{\docsvlist{#1}}{%}%
6477 }%
6478
6479 \newcommand{\reformattedwithpage@}[3]{%
6480   \def\do##1{%
6481     \setkeys[mac]{truefootnoteoption}{##1}%
6482   }%
6483   \notblank{#1}{\docsvlist{#1}}{%}%
6484 }%

```

```

6480 \xdef\@currentseries{#3withpage}%
6481 \ifboolexpr{%
6482   test{\ifcsundef{the@label#2:start}}%
6483   or test{\ifcsundef{the@label#2:end}}}%
6484 }%
6485 {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}%
6486 {%
6487   \def\@this@crossref@start{#2:start}%
6488   \def\@this@crossref@end{#2:end}%
6489   \printendlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:
start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}|\relax|\
xflagref{#2:start}|\%
6490   \undef\@this@crossref@end%
6491   \undef\@this@crossref@start%
6492 }%
6493 \def\do##1{%
6494   \setkeys[mac]{falsefootnoteoption}{##1}%
6495 }%
6496 \notblank{#1}{\docsvlist{#1}}{%}%
6497 }%
6498
6499 \newcommand{\reformattedonlypage@}[3]{%
6500   \def\do##1{%
6501     \setkeys[mac]{truefootnoteoption}{##1}%
6502   }%
6503   \notblank{#1}{\docsvlist{#1}}{%}%
6504   \xdef\@currentseries{#3onlypage}%
6505   \ifboolexpr{%
6506     test{\ifcsundef{the@label#2:start}}%
6507     or test{\ifcsundef{the@label#2:end}}}%
6508   }%
6509   {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}%
6510   {\ifnumequal{\xpageref{#2:end}}{\xpageref{#2:start}}%
6511     {%
6512       \ifcsvoid{#3onlypage@prefixsingle}%
6513       {}%
6514       {\csletcs{Xendbeforepagenumber@#3onlypage}{#3onlypage@prefixsingle
}}%
6515       \printnpnum{%
6516         \wrap@edcrossref{#2:start}{\xpageref{#2:start}}}%
6517       }%
6518     }%
6519   {%
6520     \ifcsvoid{#3onlypage@prefixmore}%
6521     {}%
6522     {\csletcs{Xendbeforepagenumber@#3onlypage}{#3onlypage@prefixmore}}%
6523     \ifdefined\linrangesep@%
6524       \printnpnum{%
6525         \wrap@edcrossref{#2:start}{\xpageref{#2:start}}}%
6526         \linrangesep@%

```

```

6527 \wrap@edcrossref{#2:end}{\xpageref{#2:end}}%
6528 }%
6529 \else%
6530 \printnpnum{%
6531 \wrap@edcrossref{#2:start}{\xpageref{#2:start}}%
6532 \csuse{Xendlinrangeseparator@}\@currentseries}%
6533 \wrap@edcrossref{#2:end}{\xpageref{#2:end}}%
6534 }%
6535 \fi%
6536 }%
6537 }%
6538 \def\do##1{%
6539 \setkeys[mac]{falsefootnoteoption}{##1}%
6540 }%
6541 \notblank{#1}{\docsvlist{#1}}{ }%
6542 }%
6543 %

```

\edmakelabel Sometimes the `\edlabel` command cannot be used to specify exactly the page and line desired; you can use the `\edmakelabel` macro make your own label. For example, if you insert `\edmakelabel{elephant}{10|25|0}` you will have created a new label, and a later call to `\edpageref{elephant}` would print ‘10’ and `\lineref{elephant}` would print ‘25’. The sub-line number here is zero. `\edmakelabel` takes a label, followed by a page and a line number(s) as arguments. \TeX defines a `\makelabel` macro which is used in lists. Peter Wilson has changed the name to `\edmakelabel`.

```

6544 \newcommand*{\edmakelabel}[2]{\expandafter\xdef\csname the@label#1\
endcsname{#2}}
6545
6546 %

```

(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. 134 and V.9 p. 100), 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`.

```

6547 \pretocmd{\XR@test}{%
6548 {\XR@test@mac+++#1#2#3#4+++}%
6549 }%
6550 }%
6551 %

```


`\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`.

```
6552 \long\def\xR@test@mac+++#1+++{\XR@test@mac@test#1}
6553 %
```

`\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`.

```
6554 \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
6555 \ifx#1\l@dmake@labels%
6556 \l@dmake@labels#2%
6557 \else
6558 \ifx#1\l@dmake@labelsR%
6559 \l@dmake@labelsR #2%
6560 \fi%
6561 \fi%
6562 }%
6563 %
```

XXIV Side notes

Regular `\marginpar`s do not work inside numbered text — they do not produce any note but do put an extra unnumbered blank line into the text.

`\@xympar` Changing `\@xympar` a little at least ensures that `\marginpar`s in numbered text do not disturb the flow.

```
6564 \pretocmd{\@xympar}%
6565 {\ifnumberedpar@
6566 \led@warn@NoMarginpars
6567 \esphack
6568 \else}%
6569 {}%
6570 {}%
6571
6572 \apptocmd{\@xympar}%
6573 {\fi}%
6574 {}
6575 {}
6576
6577 %
```

We provide side notes as replacement for `\marginpar` in numbered text.

`\sidenote@margin` These are the sidenote equivalents to `\line@margin` and `\linenummargin` for specifying which margin. The default is the right margin (opposite to the default for line
`\sidenotemargin`
`\l@getsidenote@margin`

numbers). `\l@dgetsidenote@margin` returns the number associated to side note margin:

left: 0

right: 1

outer: 2

inner: 3

```

6578 \newcount\sidenote@margin
6579 \newcommand*{\sidenotemargin}[1]{%
6580   \l@dgetsidenote@margin{#1}%
6581   \ifnum\@l@tempcntb>\m@ne
6582     \ifledRcol
6583       \global\sidenote@marginR=\@l@tempcntb
6584     \else
6585       \global\sidenote@margin=\@l@tempcntb
6586     \fi
6587   \fi}}
6588 \newcommand*{\l@dgetsidenote@margin}[1]{%
6589   \def\@tempa{#1}\def\@tempb{left}%
6590   \ifx\@tempa\@tempb
6591     \@l@tempcntb \z@
6592   \else
6593     \def\@tempb{right}%
6594     \ifx\@tempa\@tempb
6595       \@l@tempcntb \@ne
6596     \else
6597       \def\@tempb{outer}%
6598       \ifx\@tempa\@tempb
6599         \@l@tempcntb \tw@
6600       \else
6601         \def\@tempb{inner}%
6602         \ifx\@tempa\@tempb
6603           \@l@tempcntb \thr@@
6604         \else
6605           \led@warn@BadSidenotemargin
6606           \@l@tempcntb \m@ne
6607         \fi
6608       \fi
6609     \fi
6610   \fi}
6611 \sidenotemargin{right}
6612
6613 %

```

`\l@dlp@rbox` We need two boxes to store sidenote texts.
`\l@drp@rbox`

```

6614 \newbox\l@dlp@rbox
6615 \newbox\l@drp@rbox
6616
6617 %

```

`\ledlsnotewidth` `\ledrsnotewidth` These specify the width of the left/right boxes (initialised to `\marginparwidth`), their distance from the text (initialised to `\linenumsep`), and the fonts used.

```

\ledlsnotesep 6618 \newdimen\ledlsnotewidth \ledlsnotewidth=\marginparwidth
\ledrsnotesep 6619 \newdimen\ledrsnotewidth \ledrsnotewidth=\marginparwidth
\ledlsnotefontsetup 6620 \newdimen\ledlsnotesep \ledlsnotesep=\linenumsep
\ledrsnotefontsetup 6621 \newdimen\ledrsnotesep \ledrsnotesep=\linenumsep
6622 \newcommand*\ledlsnotefontsetup{\raggedleft\footnotesize}
6623 \newcommand*\ledrsnotefontsetup{\raggedright\footnotesize}
6624
6625 %

```

`\ledleftnote` `\ledrightnote`, `\ledinnernote`, `\ledouternote` are the user commands for left, right, inner and outer sidenotes. The two last one are just alias for the two first one, depending of the page number. `\ledsidenote{<text>}` is the command for a moveable sidenote.

```

\ledsidenote 6626 \newcommand*\ledleftnote[1]{\edtext{}\l@dlsnote{#1}}
6627 \newcommand*\ledrightnote[1]{\edtext{}\l@drsnote{#1}}
6628 \newcommand*\ledsidenote[1]{\edtext{}\l@dcnote{#1}}%
6629 \newcommand*\ledinnernote[1]{\edtext{}\l@disnote{#1}}%
6630 \newcommand*\ledouternote[1]{\edtext{}\l@dosnote{#1}}%
6631 %

```

`\l@dlsnote` `\l@drsnote` . The ‘footnotes’ for left, right, and moveable sidenotes. The whole scheme is reminiscent of the critical footnotes code.

```

\l@dcnote 6632 \newif\ifrightrightnoteup
\l@desnote 6633 \rightrightnoteuptrue
\l@disnote 6634
6635 \newcommand*\l@dlsnote[1]{%
6636 \begingroup%
6637 \newcommand{\content}{#1}%
6638 \ifnumberedpar@
6639 \ifledRcol%
6640 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}%
6641 \to\inserts@listR
6642 \global\advance\insert@countR \@ne%
6643 \else%
6644 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}%
6645 \to\inserts@list
6646 \global\advance\insert@count \@ne%
6647 \fi
6648 \fi%
6649 \ignorespaces%

```

```

6650 \endgroup%
6651 }%
6652
6653 \newcommand*{\l@drsnote}[1]{%
6654 \begingroup%
6655 \newcommand{\content}{#1}%
6656 \ifnumberedpar@
6657 \ifledRcol%
6658 \xright@appenditem{\noexpand\vl@drsnote{\expandonce\content}}{%
6659 \to\inserts@listR
6660 \global\advance\insert@countR \@ne%
6661 \else%
6662 \xright@appenditem{\noexpand\vl@drsnote{\expandonce\content}}{%
6663 \to\inserts@list
6664 \global\advance\insert@count \@ne%
6665 \fi
6666 \fi\ignorespaces%
6667 \endgroup%
6668 }%
6669
6670 \newcommand*{\l@dcsnote}[1]{%
6671 \begingroup%
6672 \newcommand{\content}{#1}%
6673 \ifnumberedpar@
6674 \ifledRcol%
6675 \xright@appenditem{\noexpand\vl@dcsnote{\expandonce\content}}{%
6676 \to\inserts@listR
6677 \global\advance\insert@countR \@ne%
6678 \else%
6679 \xright@appenditem{\noexpand\vl@dcsnote{\expandonce\content}}{%
6680 \to\inserts@list
6681 \global\advance\insert@count \@ne%
6682 \fi
6683 \fi\ignorespaces%
6684 \endgroup%
6685 }%
6686
6687 \newcommand*{\l@disnote}[1]{%
6688 \begingroup%
6689 \newcommand{\content}{#1}%
6690 \ifnumberedpar@%
6691 \ifledRcol%
6692 \xright@appenditem{\noexpand\vl@disnote{\expandonce\content}}{%
6693 \to\inserts@listR%
6694 \global\advance\insert@countR \@ne%
6695 \else%
6696 \xright@appenditem{\noexpand\vl@disnote{\expandonce\content}}{%
6697 \to\inserts@list%
6698 \global\advance\insert@count \@ne%
6699 \fi%

```

```

6700 \fi\ignorespaces%
6701 \endgroup%
6702 }%
6703
6704 \newcommand*{\l@dosnote}[1]{%
6705 \begingroup%
6706 \newcommand{\content}{#1}%
6707 \ifnumberedpar%
6708 \ifledRcol%
6709 \xright@appenditem{\noexpand\l@dosnote{\expandonce\content}}{%
6710 \to\inserts@listR%
6711 \global\advance\insert@countR \@ne%
6712 \else%
6713 \xright@appenditem{\noexpand\l@dosnote{\expandonce\content}}{%
6714 \to\inserts@list%
6715 \global\advance\insert@count \@ne%
6716 \fi%
6717 \fi\ignorespaces%
6718 \endgroup%
6719 }%
6720
6721 %

```

\vl@dlsnote Put the left/right text into boxes, but just save the moveable text. **\l@dcsnotetext**, **\vl@drsnote** **\l@dcsnotetext@l** and **\l@dcsnotetext@r** are etoolbox's lists which will store the content of side notes. We store the content in lists, because we need to loop later on them, in case many sidenote co-exist for the same line. That is there some special test to do, in order to:

- Store the content of **\ledsidenote** to **\l@dcsnotetext** in any cases.
- Store the content of **\rightsidenote** to:
 - **\l@dcsnotetext** if **\ledsidenote** is to be put on right.
 - **\l@dcsnotetext@r** if **\ledsidenote** is to be put on left.
- Store the content of **\leftsidenote** to:
 - **\l@dcsnotetext** if **\ledsidenote** is to be put on left.
 - **\l@dcsnotetext@l** if **\ledsidenote** is to be put on right.

\vl@disnote and **\vl@dosnote** just call **\vl@dlsnote** or **\vl@drsnote**, depending of the page.

```

6722 \newcommand*{\vl@dlsnote}[1]{%
6723 \ifledRcol%
6724 \@l@dttempcntb=\sidenote@marginR%
6725 \ifnum\@l@dttempcntb>\@ne%
6726 \advance\@l@dttempcntb by\page@numR%
6727 \fi%

```

```

6728 \else%
6729 \@l@dttempcntb=\sidenote@margin%
6730 \ifnum\@l@dttempcntb>\@ne%
6731 \advance\@l@dttempcntb by\page@num%
6732 \fi%
6733 \fi%
6734 \ifodd\@l@dttempcntb%
6735 \listgadd{\l@dcsnotetext@l}{#1}%
6736 \else%
6737 \listgadd{\l@dcsnotetext}{#1}%
6738 \fi
6739 }
6740 \newcommand*{\vl@drsnote}[1]{%
6741 \ifledRcol@%
6742 \@l@dttempcntb=\sidenote@marginR%
6743 \ifnum\@l@dttempcntb>\@ne%
6744 \advance\@l@dttempcntb by\page@numR%
6745 \fi%
6746 \else%
6747 \@l@dttempcntb=\sidenote@margin%
6748 \ifnum\@l@dttempcntb>\@ne%
6749 \advance\@l@dttempcntb by\page@num%
6750 \fi%
6751 \fi%
6752 \ifodd\@l@dttempcntb%
6753 \listgadd{\l@dcsnotetext}{#1}%
6754 \else%
6755 \listgadd{\l@dcsnotetext@r}{#1}%
6756 \fi%
6757 }
6758 \newcommand*{\vl@dcsnote}[1]{\listgadd{\l@dcsnotetext}{#1}}
6759
6760 \newcommand{\vl@disnote}[1]{%
6761 \ifledRcol@%
6762 \@tempcnta=\page@numR%
6763 \else%
6764 \@tempcnta=\page@num%
6765 \fi%
6766 \ifodd\@tempcnta% ODD => right page => inner side = left side
6767 \vl@dlsnote{#1}%
6768 \else%
6769 \vl@drsnote{#1}%
6770 \fi%
6771 }%
6772
6773 \newcommand{\vl@dosnote}[1]{%
6774 \ifledRcol@%
6775 \@tempcnta=\page@numR%
6776 \else%
6777 \@tempcnta=\page@num%

```

```

6778 \fi%
6779 \ifodd\@tempcnta% ODD => right page => outer side = right side
6780 \vl@drsnote{#1}%
6781 \else%
6782 \vl@dlsnote{#1}%
6783 \fi%
6784 }%
6785
6786 %

```

`\setl@dlp@rbox` `\setl@dlprbox{<lednums>}{<tag>}{<text>}` puts `<text>` into the `\l@dlp@rbox` box. And similarly for the right side box. It is these boxes that finally get displayed in the margins.

```

6787 \newcommand*\setl@dlp@rbox}[1]{%
6788 \begingroup%
6789 \parindent\z@\hsize=\ledlsnotewidth%
6790 \ledlsnotefontsetup%We kept it outside of the vbox, because can affect
the ragging
6791 \global\setbox\l@dlp@rbox%
6792 \ifleftnoteup%
6793 =\vbox to\z@{\ledlsnotefontsetup\vss #1}%We put \
ledlsnotefontsetup inside footnote because required for color command. Note
the {} to keep setting local.
6794 \else%
6795 =\vbox to 0.70\baselineskip{\ledlsnotefontsetup\strut#1\vss}}%
6796 \fi%
6797 \endgroup%
6798 }
6799
6800 \newcommand*\setl@drp@rbox}[1]{%
6801 \begingroup%
6802 \parindent\z@\hsize=\ledrsnotewidth%
6803 \ledrsnotefontsetup%We kept it outside of the vbox, because can affect
the ragging
6804 \global\setbox\l@drp@rbox%
6805 \ifrightrightnoteup%
6806 =\vbox to\z@{\ledrsnotefontsetup\vss#1}%We put \ledrsnotefontsetup
inside footnote because required for color command. Note the {} to keep
setting local.
6807 \else%
6808 =\vbox to0.7\baselineskip{\ledrsnotefontsetup\strut#1\vss}}%
6809 \fi%
6810 \endgroup%
6811 }%
6812 \newif\ifleftnoteup
6813 \leftnoteuptrue
6814 %

```

`\@sidenotesep` This macro is used to separate sidenotes of the same line.

```

6815 \newcommand{\setsidenotesep}[1]{\gdef\@sidenotesep{#1}}
6816 \newcommand{\@sidenotesep}{, }
6817 %

```

\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.

```

6818 \newcommand*{\affixside@note}{%
6819   \prepare@edindex@for@note{\the\page@num|\the\line@num|\the\subline@num|\
the\page@num|\the\line@num|\the\subline@num}%
6820   \def\sidenotecontent@{}%
6821   \numgdef{\itemcount@}{0}%
6822   \def\do##1{%
6823     \ifnumequal{\itemcount@}{0}%
6824       {%
6825         \appto\sidenotecontent@{##1}}% Not print not separator before
the 1st note
6826       {\appto\sidenotecontent@{\@sidenotesep ##1}%
6827       }%
6828       \numgdef{\itemcount@}{\itemcount@+\@ne}%
6829   }%
6830   \dolistloop{\l@dcnotes@text}%
6831   \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
6832   %

```

And we do the same for left and right notes (not movable).

```

6833 \gdef\@templ@d{%
6834 \gdef\@templ@n{\l@dcnotes@text\l@dcnotes@text@1\l@dcnotes@text@r}%
6835 \ifx\@templ@d\@templ@n \else%
6836 \if@twocolumn%
6837   \if@firstcolumn%
6838     \setl@dlp@rbox{##1}{\sidenotecontent@}%
6839   \else%
6840     \setl@drp@rbox{\sidenotecontent@}%
6841   \fi%
6842 \else%
6843   \@l@tempcntb=\sidenote@margin%
6844   \ifnum\@l@tempcntb>\@ne%
6845     \advance\@l@tempcntb by\page@num%
6846   \fi%
6847   \ifodd\@l@tempcntb%
6848     \setl@drp@rbox{\sidenotecontent@}%
6849   \gdef\sidenotecontent@{}%
6850   \numgdef{\itemcount@}{0}%
6851   \dolistloop{\l@dcnotes@text@1}%
6852   \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%

```



```

6853     \setl@dlp@rbox{\sidenotecontent@}%
6854   \else%
6855     \setl@dlp@rbox{\sidenotecontent@}%
6856     \gdef\sidenotecontent@{%
6857       \numgdef{\itemcount@}{0}%
6858       \dolistloop{\l@dcstotetext@r}%
6859       \ifnumgreater{\itemcount@}{1}{\led@err@ManyRighnotes}{}%
6860       \setl@drp@rbox{\sidenotecontent@}%
6861     \fi%
6862   \fi%
6863 \fi%
6864 \advance\@edindex@fornote@\m@ne%
6865 }
6866 %

```

XXV Minipages and such

We can put footnotes into minipages. The preparatory code has been set up earlier, all that remains is to ensure that it is available inside a minipage box. This requires some alteration to the kernel code, specifically the `\@iiiminipage` and `\endminipage` macros. We will arrange this so that additional series can be easily added.

`\l@dfeetbeginmini` These will be the hooks in `\@iiiminipage` and `\endminipage`.
`\l@dfeetendmini` They can be extended to handle other things if necessary.

```

6867 \ifnoledgroup@ \else%
6868 \newcommand*{\l@dfeetbeginmini}{\@ledgrouptrue\l@dedbeginmini\l@dfambeginmini}
6869 \newcommand*{\l@dfeetendmini}{%
6870   \IfStrEq{critical-familiar}{\@mpfnpos}%
6871   {\l@dedendmini\l@dfamendmini}%
6872   {%
6873     \IfStrEq{familiar-critical}{\@mpfnpos}%
6874     {\l@dfamendmini\l@dedendmini}%
6875     {\do@feet@custom@order{mp@}{\@mpfnpos}}%
6876   }%
6877 }%
6878 %

```

`\l@dedbeginmini` These handle the initiation and closure of critical footnotes in a minipage environment.
`\l@dedendmini`
`\mp@append@Xnotes`

```

6879 \newcommand*{\l@dedbeginmini}{%
6880   \unless\ifnocritical@%
6881   \def\do##1{%
6882     \csletcs{v##1footnote}{mpv##1footnote}%
6883   }%
6884   \dolistloop{\@series}%
6885 \fi%
6886 }

```

```

6887 \newcommand*{\l@dedendmini}{%
6888   \unless\ifnocritical@%
6889     \ifl@dpairing%
6890       \ifledRcol%
6891         \flush@notesR%
6892       \else%
6893         \flush@notes%
6894       \fi%
6895   \fi
6896   \def\do##1{%
6897     \mp@append@Xnotes{##1}%
6898   }%
6899   \dolistloop{\@series}%
6900 \fi%
6901 }%
6902 \newcommand{\mp@append@Xnotes}[1]{%
6903 \ifvoid\csuse{mp#1footins}\else%
6904   \ifl@dpairing%
6905     \ifparledgroup%
6906       \ifledRcol%
6907         \dingdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+\skip
\@nameuse{mp#1footins}}%
6908       \else%
6909         \dingdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL+\
skip\@nameuse{mp#1footins}}%
6910       \fi%
6911     \fi%
6912   \fi%
6913   \ifcsstring{series@display#1}{paragraph}{%
6914     \setbox\@nameuse{mp#1footins}=\vbox{%
6915       \csuse{Xnotefontsize@#1}%
6916       \ifcsdef{Xhsize}\csuse{series@display#1}@#1}%
6917       \hsize\csuse{Xhsize}\csuse{series@display#1}@#1}%
6918     }{%
6919       \noindent\csuse{Xtxtbeforenotes@#1}%
6920       \unvbox\@nameuse{mp#1footins}%
6921       \@parboxrestore%
6922     }%
6923   }%
6924   \csuse{mp#1footgroup}{#1}%
6925 \fi%
6926 }%
6927 %

```

`\l@dfambeginmini` These handle the initiation and closure of familiar footnotes in a minipage environment.

`\l@dfamendmini`

`\mp@append@notesX`

```

6928 \newcommand*{\l@dfambeginmini}{%
6929   \unless\ifnofamiliar@%
6930     \def\do##1{\csletcs{vfootnote##1}{mpvfootnote##1}}%
6931     \dolistloop{\@series}%

```

```

6932 \fi%
6933 }%
6934
6935 \newcommand*{\l@dfamendmini}{%
6936 \unless\ifnofamiliar%
6937 \def\do##1{%
6938 \mp@append@notesX{##1}%
6939 }%
6940 \dolistloop{\@series}%
6941 \fi%
6942 }%
6943 \newcommand{\mp@append@notesX}[1]{%
6944 \ifvoid\csuse{mpfootins#1}\else%
6945 \ifcsstring{series@displayX#1}{paragraph}{%
6946 \setbox\@nameuse{mpfootins#1}=\vbox{%
6947 \csuse{notefontsizeX@#1}%
6948 \ifcsdef{hsize\csuse{series@display#1}X@#1}{%
6949 \hsize\csuse{hsize\csuse{series@display#1}X@#1}%
6950 }{}%
6951 \noindent\csuse{txtbeforenotesX@#1}%
6952 \unvbox\@nameuse{mpfootins#1}%
6953 \@parboxrestore%
6954 }%
6955 }%
6956 \csuse{mpfootgroup#1}{#1}%
6957 \fi%
6958 }%
6959 %

```

`\@iiiminipage` This is our extended form of the kernel `\@iiiminipage` defined in `ltboxes.dtx`.

```

6960 \patchcmd%
6961 {\@iiiminipage}%
6962 {\let\@footnotetext\mpfootnotetext}%
6963 {\let\@footnotetext\mpfootnotetext\l@dfetbeginmini}%
6964 {}%
6965 {\led@error@fail@patch@iiiminipage}%
6966 %

```

`\endminipage` This is our extended form of the kernel `\endminipage` defined in `ltboxes.dtx`.

```

6967 \patchcmd%
6968 {\endminipage}%
6969 {\footnoterule}%
6970 {\footnoterule\l@advance@parledgroup@beforenormalnotes}%
6971 {}%
6972 {\led@error@fail@patch@endminipage}
6973
6974 \patchcmd%
6975 {\endminipage}%

```

```

6976 {\@minipagefalse}%
6977 {\l@dfteetendmini\@minipagefalse}%
6978 }}%
6979 {\led@error@fail@patch@endminipage}
6980
6981 %

```

`\l@dunboxmpfoot` `\@ldunboxmpfoot` insert normal footnotes for ledgroup.
`\advance@parledgroup@beforenormalnotes`

```

6982 \newcommand*{\l@dunboxmpfoot}{%
6983   \vskip\skip\@mpfootins
6984   \normalcolor
6985   \footnoterule
6986   \l@advance@parledgroup@beforenormalnotes
6987   \unvbox\@mpfootins%
6988 }
6989 %

```

When using parallel ledgroup, we need to store the vertical space added before footnote, in order to compensate them between left and right pages.

```

6990 \newcommand{\l@advance@parledgroup@beforenormalnotes}{%
6991   \ifparledgroup
6992     \ifl@pairing
6993       \ifledRcol
6994         \dimgdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+\
skip\@mpfootins}
6995       \else
6996         \dimgdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL+\
skip\@mpfootins}
6997       \fi
6998     \fi
6999   \fi
7000 }
7001 %

```

`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.

```

7002 \newenvironment{ledgroup}{%
7003   \resetprevpage@num%
7004   \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@%
7005   \let\@footnotetext\@mpfootnotetext
7006   \l@dfteetbeginmini%
7007 }{%
7008   \par
7009   \unskip
7010   \ifvoid\@mpfootins\else
7011

```

```

7012 \l@dunboxmpfoot
7013 \fi
7014 \l@dfeetendmini%
7015 \@ledgroupfalse%
7016 }
7017
7018
7019 %

```

```

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

```

This environment puts footnotes at the end, even if that happens to be in the middle of a page, or crossing a page boundary. It is a sort of unboxed, variable $\langle width \rangle$ minipage. The optional $\langle pos \rangle$ controls the sideways position of numbered text.

```

7020 \newenvironment{ledgroupsize}[2][1]{%
7021 %

```

Set the various text measures.

```

7022 \hsize #2\relax
7023 %

```

Initialize fills for centering.

```

7024 \let\ledllfill\hfil
7025 \let\ledrlfill\hfil
7026 \def\@tempa{#1}\def\@tempb{1}%
7027 %

```

Left adjusted numbered lines

```

7028 \ifx\@tempa\@tempb
7029 \let\ledllfill\relax
7030 \else
7031 \def\@tempb{r}%
7032 \ifx\@tempa\@tempb
7033 %

```

Right adjusted numbered lines

```

7034 \let\ledrlfill\relax
7035 \fi
7036 \fi
7037 %

```

Set up the footnoting.

```

7038 \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
7039 \let\@footnotetext\@mpfootnotetext
7040 \l@dfeetbeginmini%
7041 }{%
7042 \par
7043 \unskip
7044 \ifvoid\@mpfootins\else
7045 \l@dunboxmpfoot

```

```

7046 \fi
7047 \l@dfeetendmini%
7048 }
7049
7050 %

```

Close the \ifnoledgroup@\else.

```

7051 \fi%
7052 %

```

`\ifledgroupnotesL@` These boolean tests check if we are in the notes of a ledgroup. If we are, we do not
`\ifledgroupnotesR@` number the lines. It could be useful for parallel ledgroup of `reledpar`.

```

7053 \newif\ifledgroupnotesL@
7054 \newif\ifledgroupnotesR@
7055 %

```

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`.

```

7056 \AtBeginDocument{%
7057   \unless\ifl@imakeidx%
7058     \ifpackageloaded{imakeidx}{\led@error@PackageAfterEledmac{imakeidx}}{}
7059   %
7060   \fi%
7061   \unless\ifl@indextools%
7062     \ifpackageloaded{indextools}{\led@error@PackageAfterEledmac{indextools}}{}
7063   %
7064   \fi%
7065   \unless\ifl@footmisc%
7066     \ifpackageloaded{footmisc}{\led@error@PackageAfterEledmac{footmisc}}{}
7067   %
7068   \fi%
7069 }
7070 %

```

XXVI.2 Auxiliary macros for `\edindex`

`\pagelinesep` In order to get a correct line number we have to use the label/ref mechanism. These
`\edindexlab` macros are for that.
`\c@labidx`

```

7068 \newcommand{\pagelinesep}{-}
7069 \newcommand{\edindexlab}{${&}}
7070 \newcounter{labidx}
7071 \setcounter{labidx}{0}
7072
7073 %

```

`\doedindexlabel` This macro sets an `\edlabel`.

```

7074 \newcommand{\doedindexlabel}{%
7075   \stepcounter{labidx}%
7076   \edlabel{\edindexlab\thelabidx}%
7077 }
7078
7079 %

```

`\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`.

```

7080 \newcounter{pageline}%
7081 \renewcommand{\thepageline}{%
7082   \thepage%
7083   \pagelinesep%
7084   \xlineref{\edindexlab\thelabidx}%
7085 }
7086 %

```

`\thestartpageline` These macros make up the page/line start/end number when the `\edindex` command
`\theendpageline` is called in critical notes.

```

7087 \newcommand{\thestartpageline}{%
7088   \l@dparsedstartpage%
7089   \pagelinesep%
7090   \l@dparsedstartline%
7091 }
7092 \newcommand{\theendpageline}{%
7093   \l@dparsedendpage%
7094   \pagelinesep%
7095   \l@dparsedendline%
7096 }
7097 %

```

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 side-note), and decremented at the end of each note. If its value is greater than 0, that means we are inside a note.

```

7098 \newcount\@edindex@fornote@
7099 %

```

\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 |.

```

7100 \newcommand{\prepare@edindex@fornote}[1]{%
7101     \l@dp@rsefootspec#1|}%
7102     \advance\@edindex@fornote@\@ne%
7103 }
7104 %

```

\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.

```

7105 \newcommand{\get@edindex@ledinnote@command}{%
7106     \ifxindy%
7107         \gdef\@ledinnote@command{%
7108             ledinnote\thelabidx%
7109         }%
7110         \ifxindyhyperref%
7111             \immediate\write\eledmac@xindy@out{%
7112                 (define-attributes ("ledinnote\thelabidx"))^^J
7113                 \space\space(markup-locref^^J
7114                 \eledmacmarkuplocrefdepth^^J
7115                 :open "\string\ledinnote[\edindexlab\thelabidx]{\@index@command
7116             }{"^^J
7117                 :close "}"^^J
7118                 :attr "ledinnote\thelabidx"^^J
7119             )
7120         }%
7121     \else%
7122         \immediate\write\eledmac@xindy@out{%
7123             (define-attributes ("ledinnote\thelabidx"))^^J
7124             \space\space(markup-locref^^J
7125             \eledmacmarkuplocrefdepth^^J
7126             :open "\string\ledinnote{\@index@command}{"^^J
7127             :close "}"^^J
7128             :attr "ledinnote\thelabidx"^^J
7129         )
7130     }%
7131 \fi%

```

If we do not use xindy option, `\@ledinnote@command` will produce something like `ledinnote{formattingcommand}`.


```

7132 \else%
7133 \gdef\@ledinnote@command{%
7134     ledinnote[\edindexlab\thelabidx]{\@index@command}%
7135 }%
7136 \fi%
7137 }
7138 %

```

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 |.

```

7139 \def\get@index@command#1|#2+{%
7140 \gdef\@index@txt{#1}%
7141 \gdef\@index@command{#2}%
7142 \xdef\@index@parenthesis{%
7143 \IfBeginWith{\@index@command}{(}{%
7144 \StrGobbleLeft{\@index@command}{1}{\@index@command}%
7145 \global\let\@index@command\@index@command%
7146 \xdef\@index@parenthesis{(%
7147 }){}%
7148 \IfBeginWith{\@index@command}{)}{%
7149 \StrGobbleLeft{\@index@command}{1}{\@index@command}%
7150 \global\let\@index@command\@index@command%
7151 \xdef\@index@parenthesis{)}}%
7152 }){}%
7153 }
7154 %

```

XXVI.5 Code for the formatted index

`\ledinnote` These macros are used to specify that an index reference points to a note. Arguments of `\ledinnote` are: #1 (optional): the label for the hyperlink, #2: command applied to the number, #3: the number itself.

`\ledinnotehyperpage`

`\ledinnotemark`

```

7155 \newcommandx{\ledinnote}[3][1,usedefault]{%
7156 \ifboolexpr{%
7157     test{\ifdefequal{\iftrue}{\ifHy@hyperindex}}%
7158     or%
7159     bool {xindyhyperref@}%
7160 }%
7161 {%
7162 \csuse{#2}{\hyperlink{#1}{\ledinnotemark{#3}}}%
7163 }%
7164 {%
7165 \csuse{#2}{\ledinnotemark{#3}}%
7166 }%
7167 }%

```

```

7168 \newcommand{\ledinnotehyperpage}[2]{\csuse{#1}{\ledinnotemark{\hyperpage
      {#2}}}}%
7169 \newcommand{\ledinnotemark}[1]{#1\emph{n}}%
7170 %

```

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
7171 \newcommandx{\@wredindex}[2][1=\expandonce\jobname,usedefault]{%#1 = the
      index name, #2 = the text
7172 \begingroup%
7173 \let\emph\@firstofone%
7174 \let\textbf\@firstofone%
7175 \let\textit\@firstofone%
7176 \let\textmd\@firstofone%
7177 \let\textnormal\@firstofone%
7178 \let\textrm\@firstofone%
7179 \let\textsc\@firstofone%
7180 \let\textsf\@firstofone%
7181 \let\textsl\@firstofone%
7182 \let\texttt\@firstofone%
7183 \let\textup\@firstofone%
7184 \xdef\@tmp{#2}%To be used in IfSubStr instead of #2 directly. Avoid
      some expansion bugs (for example with \edindex{textsc{something}})
7185 \endgroup%
7186 \ifl@imakeidx%
7187 \ifnum\@edindex@fornote@>\z@%
7188 \IfSubStr[1]{\@tmp}{|}{\get@index@command#2+}{\get@index@command#2|+}%
      %
7189 \get@edindex@ledinnote@command%
7190 \expandafter\imki@wrindexentry{#1}{\@index@txt|(\@ledinnote@command
      )}{\thestartpageline}%
7191 \expandafter\imki@wrindexentry{#1}{\@index@txt|)\@ledinnote@command
      }{\theendpageline}%
7192 \else%
7193 \get@edindex@hyperref{#2}%
7194 \imki@wrindexentry{#1}{\@index@txt\@edindex@hyperref}{\thepageline}%
7195 \fi%
7196 \else%

```

```

7197 \ifnum\@edindex@fornote@>\z@%
7198 \IfSubStr[1]{\@tmp}{|}{\get@index@command#2+}{\get@index@command#2|+}
%
7199 \get@edindex@ledinnote@command%
7200 \expandafter\protected@write\@indexfile{}%
7201 {\string\indexentry{\@index@txt|(\@ledinnote@command){\thestartpageline}
7202 }%
7203 \expandafter\protected@write\@indexfile{}%
7204 {\string\indexentry{\@index@txt|)\@ledinnote@command}{\theendpageline}
7205 }%
7206 \else%
7207 \protected@write\@indexfile{}%
7208 {\string\indexentry{#2}{\thepageline}
7209 }%
7210 \fi%
7211 \fi%
7212 \endgroup
7213 \@esphack%
7214 }
7215 %

```

Need to add the definition of `\edindex` to `\makeindex`, and initialise `\edindex` to do nothing.

```

7216 \pretocmd{\makeindex}{%
7217 \def\edindex{%
7218 \ifboolexpr{bool{numbering} or bool{numberingR} or bool{
7219 l@printingpages} or bool{l@printingcolumns}}{%
%
7220 \@bsphack%
7221 \doedindexlabel%
7222 \beginngroup%
7223 \@sanitize%
7224 \wredindex%
7225 }%
7226 {%
7227 \led@warn@edinde@outsidenumbering%
7228 \index%
7229 }%
7230 }%
7231 {}%
7232 {\led@error@fail@patch@makeindex}%
7233 \newcommand{\edindex}[1]{\@bsphack\@esphack}
7234 \newcommand{\dummy@edindex}[2][1=\expandonce\jobname,usedefault]{}%
7235 %

```

XXVI.7 Hyperlink

`\hyperlinkformat` `\hyperlinkformat` command is to be used to have both a internal hyperlink and a format, when indexing.

```

7236 \newcommand{\hyperlinkformat}[3]{%
7237   \ifstrempy{#1}%
7238     {\hyperlink{#2}{#3}}%
7239     {\csuse{#1}{\hyperlink{#2}{#3}}}%
7240   }%
7241 %

```

\hyperlinkR \hyperlinkR command is to be used to create a internal hyperlink and \ledRflag, when indexing.

```

7242 \newcommand{\hyperlinkR}[2]{%
7243   \hyperlink{#1}{#2\@Rlineflag}%
7244 }%
7245 %
7246 %

```

\hyperlinkformatR \hyperlinkformatR command is to be used to create a internal hyperlink, a format and a \@Rlineflag, when indexing.

```

7247 \newcommand{\hyperlinkformatR}[3]{%
7248   \hyperlinkformat{#1}{#2}{#3\@Rlineflag}%
7249 }%
7250 %
7251 %

```

\get@edindex@hyperref \get@edindex@hyperref is to be used to define the \@edindex@hyperref macro, which, in index, links to the point where the index was called (with hyperref).

```

7252 \newcommand{\get@edindex@hyperref}[1]{%
7253 %

```

We have to disable temporary spaces to work through a xstring bug (or feature?)

```

7254 \edef\temp@{%
7255   \catcode`\ =9 %space need for catcode
7256   \detokenize{#1}%For active character in unicode
7257   \catcode`\ =10 % space need for catcode
7258 }%
7259 %

```

Now, we define \@edindex@hyperref if the hyperindex of hyperref is enabled.

```

7260 \ifdefequal{\iftrue}{\ifHy@hyperindex}{%
7261   \IfSubStr{\temp@}{|}%
7262     {\get@index@command#1+%
7263       \ifledRcol%
7264         \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
7265           \hyperlinkformatR{\@index@command}%
7266           {\edindexlab\thelabidx}}%
7267       \else%
7268         \gdef\@edindex@hyperref{|\@index@parenthesis %space kept

```

```

7269     hyperlinkformat{\@index@command}%
7270     {\edindexlab\thelabidx}}}%
7271     \fi%
7272   }%
7273   {\get@index@command#1|+}%
7274   \ifledRcol%
7275     \gdef\@edindex@hyperref{|hyperlinkR{\edindexlab\thelabidx}}}%
7276   \else%
7277     \gdef\@edindex@hyperref{|hyperlink{\edindexlab\thelabidx}}}%
7278   \fi%
7279   }%
7280 }%
7281 %

7282 % If we use both xindy and hyperref, first get the \protect\cs{
index@command} command.
7283 % Then define \protect\cs{@edindex@hyperref} in the form \verb+eledmacXXX+
7284 % \begin{macrocode}
7285 {\ifxindyhyperref%
7286   \IfSubStr{\temp@}{|}%
7287     {\get@index@command#1+}%
7288     {\get@index@command#1|+}%
7289   \gdef\@edindex@hyperref{|eledmac\thelabidx}%
7290 }%

```

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`.

```

7291   \IfStrEq{\@index@parenthesis}{(}%
7292   {%
7293     \csxdef{xindyparenthesis@\@index@txt}{\thelabidx}%
7294     \gdef\@edindex@hyperref{| (eledmac\thelabidx}%
7295   }%
7296   {}%
7297 %

```

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`.

```

7298   \IfStrEq{\@index@parenthesis}{)}%
7299   {%
7300     \xdef\@edindex@hyperref{|)eledmac\csuse{xindyparenthesis@
\@index@txt}}}%
7301     \global\csundef{xindyparenthesis@\@index@txt}%
7302   }%
7303 %

```

Write in the `.xdy` file the attributes of the location.

```

7304   {%

```

```

7305 \immediate\write\eledmac@xindy@out{%
7306   (define-attributes ("eledmac\thelabidx"))^^J
7307   \space\space(markup-locref^^J
7308     \eledmacmarkuplocdepth^^J
7309     :open "\string\hyperlink%
7310       \ifledRcol R\fi%
7311       {\edindexlab\thelabidx}%
7312       {\ifdefempty{\@index@command}%
7313         {}%
7314         {\@backslashchar\@index@command}%
7315       }^^J
7316     :close "}")^^J
7317     :attr "eledmac\thelabidx"^^J
7318   )
7319 }%
7320 }%
7321 %

```

And now, in any other case.

```

7322 \else%
7323   \gdef\@index@txt{#1}%
7324   \gdef\@edindex@hyperref{}%
7325   \fi%
7326 }%
7327 }
7328 %

```

XXVI.8 ‘innote’ and ‘notenumber’ option of indextols package

`\led@set@index@fornote` The `\led@set@index@fornote` is called when a familiar footnote is inserted — and not when it is read — and changes the `\index` command depending of the option of the `indextools` package. Its only argument is the note series.

```

7329 \newcommand{\led@set@index@fornote}[1]{%
7330   \ifbool{indtl@innote}%
7331     {\let\index\nindex}%
7332     {}%
7333   \ifbool{indtl@notenumber}%
7334     {%
7335     \renewcommand{\index}[2][\indtl@jobname]{%
7336       \orig@index[##1]{%
7337         ##2|innotenumber{\this@footnoteX@reading}%
7338       }%
7339     }%
7340   }%
7341   {}%
7342 }%
7343 %

```

`\led@reinit@index@fornote` The `\led@reinit@index@fornote` just reset the default value of `\index`.

```

7344 \newcommand{\led@reinit@index@fornote}{%
7345   \ifbool{indtl@innote}%
7346     {\let\index\orig@index}%
7347     {}%
7348   \ifbool{indtl@notenumber}%
7349     {\let\index\orig@index}%
7350     {}%
7351 }%
7352 %

```

XXVII Glossaries

Here, we define the `\gls`-like commands prefixed by `ed`, only if the package `glossaries` is loaded.

```

7353 \AtBeginDocument{%
7354   \@ifpackageloaded{glossaries}{%
7355     %

```

First those which arguments are `[<options>]{<label>}[<insert>]`.

```

7356   \renewcommand{\do}[1]{%
7357     \expandafter\DeclareRobustCommand\csname ed#1\endcsname[3][1,3,
usedefault]{%
7358       \doedindexlabel%
7359       \csname#1\endcsname[counter=pageline,##1]{##2}[##3]%
7360     }%
7361     \expandafter\WithSuffix\expandafter\DeclareRobustCommand\csname ed
#1\endcsname*[3][1,3,usedefault]{%
7362       \doedindexlabel%
7363       \csname#1\endcsname*[counter=pageline,##1]{##2}[##3]%
7364     }%
7365   }%
7366   \docsvlist{%
7367     gls,%
7368     Gls,%
7369     GLS,%
7370     glspl,%
7371     Glspl,%
7372     GLSpl,%
7373     glstext,%
7374     Glstext,%
7375     GLStext,%
7376     Glsfirst,%
7377     GLSfirst,%
7378     glsplural,%
7379     Glsplural,%
7380     GLSplural,%

```

```

7381 glsfirstplural,%
7382 Glsfirstplural,%
7383 GLSfirstplural,%
7384 glsname,%
7385 Glsname,%
7386 GLSname,%
7387 glssymbol,%
7388 Glsymbol,%
7389 GLSsymbol,%
7390 glsdesc,%
7391 Glsdesc,%
7392 GLSdesc,%
7393 glsuseri,%
7394 Glsuseri,%
7395 GLSuseri,%
7396 glsuserii,%
7397 Glsuserii,%
7398 GLSuserii,%
7399 glsuseriii,%
7400 Glsuseriii,%
7401 GLSuseriii,%
7402 glsuseriv,%
7403 Glsuseriv,%
7404 GLSuseriv,%
7405 glsuserv,%
7406 Glsuserv,%
7407 GLSuserv,%
7408 glsuservi,%
7409 Glsuservi,%
7410 GLSuservi%
7411 }%
7412 %

```

Then those which arguments are [*options*]{*label*}{*link text*}.

```

7413 \renewcommand{\do}[1]{%
7414 \expandafter\DeclareRobustCommand\csname ed#1\endcsname[3][1,
usedefault]{%
7415 \doedindexlabel%
7416 \csname#1\endcsname[counter=pageline,##1]{##2}{##3}%
7417 }%
7418 \expandafter\WithSuffix\expandafter\DeclareRobustCommand\csname ed
#1\endcsname*[3][1,usedefault]{%
7419 \doedindexlabel%
7420 \csname#1\endcsname*[counter=pageline,##1]{##2}{##3}%
7421 }%
7422 }%
7423 \docsvlist{glsdisp,glslink}%
7424 %

```

Then those which arguments are [*options*]{*label*}.


```

7425 \renewcommand{\do}[1]{%
7426 \expandafter\DeclareRobustCommand\csname ed#1\endcsname[2][1,
usedefault]{%
7427 \doedindexlabel%
7428 \csname#1\endcsname[counter=pageline,##1]{##2}%
7429 }%
7430 \expandafter\WithSuffix\expandafter\DeclareRobustCommand\csname ed
#1\endcsname*[2][1,usedefault]{%
7431 \doedindexlabel%
7432 \csname#1\endcsname*[counter=pageline,##1]{##2}%
7433 }%
7434 }%
7435 \docsvlist{glsadd}%
7436 }{}%
7437 }%
7438 %

```

XXVIII Verse

The original code is principally Wayne Sullivan's code from `edstanza`. However, the code has been many time modified by Maïeul Rouquette in order to obtain new features and improved compatibility with `reledpar`.

XXVIII.1 Hanging symbol management

`\@hangingsymbol` The macro `\@hangingsymbol` is used to insert a symbol on each hanging of verses. It is set by user level macro `\sethangingsymbol`.
`\ifinstanza` For example, in french typographie the symbol is '['. We obtain it by the next code:

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

The `\ifinstanza` boolean is used to be sure that we are in a stanza part.

```

7439 \def\@hangingsymbol{}
7440 \newcommand*\sethangingsymbol[1]{%
7441 \gdef\@hangingsymbol{#1}%
7442 }%
7443 \newif\ifinstanza
7444 %

```

`\inserthangingsymbol` The boolean `\ifinserthangingsymbol` is set to TRUE when `\@lock` is greater than 1, i.e. when we are not in the first line of a verse. The switch of `\ifinserthangingsymbol` is made in `\do@line` before the printing of line but after the line number calculation.

```

7445 \newif\ifinserthangingsymbol
7446 \newcommand{\inserthangingsymbol}{%
7447 \ifinserthangingsymbol%
7448 \ifinstanza%

```

```

7449 \hangingsymbol%
7450 \fi%
7451 \fi%
7452 }
7453 %

```

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.

```

7454 \newcommand*{\ampersand}{\char`\&}
7455
7456 %

```

XXVIII.3 Code category setting

\stanza@count Before we can define the main macros we need to save and reset some category codes.
\stanzaindentbase To save the current values we use `\next` and `\body` from the `\loop` macro.

```

7457 \chardef\body=\catcode`\@
7458 \catcode`\@=11
7459 \chardef\next=\catcode`\&
7460 \catcode`\&=\active
7461
7462 %

```

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.

```

7463 \newcount\stanza@count
7464 \newlength{\stanzaindentbase}
7465 \setlength{\stanzaindentbase}{20pt}
7466
7467 %

```

\strip@szacnt The indentations of stanza lines are non-negative integer multiples of the unit called
\setstanzavalues `\stanzaindentbase`. To make it easier for the user to specify these numbers, some list macros are defined. These take numerical values in a list separated by commas and assign the values to special control sequences using `\mathchardef`. Though this does limit the range from 0 to 32767, it should suffice for most applications, including *penalties*, which will be discussed below.

```

7468 \def\strip@szacnt#1,#2|{\def\@tempb{#1}\def\@tempa{#2|}}
7469 \newcommand*{\setstanzavalues}[2]{\def\@tempa{#2,,|}%
7470   \stanza@count\z@
7471   \def\next{\expandafter\strip@szacnt\@tempa
7472     \ifx\@tempb\empty\let\next\relax\else
7473     \expandafter\mathchardef\csname #1@\number\stanza@count
7474     \@endcsname\@tempb\relax
7475     \advance\stanza@count\@ne\fi\next}%
7476   \next}
7477
7478 %

```

\setstanzaindents In the original edmac, `\setstanzavalues{sza}{⟨...⟩}` had to be called to set the indents, and similarly `\setstanzavalues{szp}{⟨...⟩}` to set the penalties. `\setstanzaindents` and `\setstanzapenalties` macros are a convenience to give the user one less thing to worry about (misspelling the first argument).

```

7479 \newcommand*{\setstanzaindents}[1]{\setstanzavalues{sza}{#1}}
7480 \newcommand*{\setstanzapenalties}[1]{\setstanzavalues{szp}{#1}}
7481 %
7482 %

```

\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.

```

7483 \newcounter{stanzaindentsrepetition}
7484 \newcount\stanza@modulo
7485
7486 \newcommand*{\managestanza@modulo}[0]{%
7487   \advance\stanza@modulo\@ne%
7488   \ifnum\stanza@modulo>\value{stanzaindentsrepetition}%
7489     \stanza@modulo\@ne%
7490   \fi%
7491 }
7492 %

```

\stanzaindent The macro `\stanzaindent`, when called at the beginning of a verse, changes the indentation normally defined for this verse by `\setstanzaindent`. The starred version **\stanzaindent*** skips the current verse for the repetition of stanza indent.

```

7493 \newcommand{\stanzaindent}[1]{%
7494   \hspace{\dimexpr#1\stanzaindentbase-\parindent\relax}%
7495   \ignorespaces%
7496 }%
7497 \WithSuffix\newcommand\stanzaindent*[1]{%
7498   \stanzaindent{#1}%

```

```

7499 \global\advance\stanza@modulo-\@ne%
7500 \ifnum\stanza@modulo=0%
7501   \global\stanza@modulo=\value{stanzaindentrepetition}%
7502 \fi%
7503 \ignorespaces%
7504 }%
7505 %

```

XXVIII.5 Numbering stanza

Here, macro for numbering stanza. First, the stanza counter.

```

\thestanza\newcounter{stanza}
7507 \renewcommand{\thestanza}{%
7508   \textbf{\arabic{stanza}}%
7509 }
7510 %

```

`\ifnumberstanza` Then, macro to activate automatically numbering of stanza.

```

7511 \newif\ifnumberstanza%
7512 %

```

`\@insertstanzanumber` Now, macro called at the first line of verse of a stanza.

```

7513 \newcommand{\@insertstanzanumber}[0]{%
7514   \ifnumberstanza%
7515     \ifl@dpairing%
7516       \ifledRcol%
7517         \stanzanumwrapper{\thestanzaR}%
7518       \else%
7519         \stanzanumwrapper{\thestanzaL}%
7520       \fi%
7521     \else%
7522       \stanzanumwrapper{\thestanza}%
7523     \fi%
7524     \setline{1}%
7525   \fi%
7526 }%
7527 %

```

`\@advancestanzanumber` Also a command to advance the counter of stanza.

```

7528 \newcommand{\@advancestanzanumber}[0]{%
7529   \ifnumberstanza%
7530     \ifl@dpairing%
7531       \ifledRcol%
7532         \addtocounter{stanzaR}{1}%
7533       \else%

```

```

7534 \addtocounter{stanzaL}{1}%
7535 \fi%
7536 \else%
7537 \addtocounter{stanza}{1}%
7538 \fi%
7539 \fi%
7540 }%
7541 %

```

`\stanzanumwrapper` And finally, the wrapper for stanza number

```

7542 \newcommand{\stanzanumwrapper}[1]{%
7543 \flagstanza{#1}%
7544 }%
7545 %

```

XXVIII.6 Stanza number in note

Here, the command called when printing stanza number in notes.

```

7546 \newcommand{\printstanza}[0]{%
7547 \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
7548 l@dprintingcolumns}}{%
7549 \ifledRcol{%
7549 \thestanzaR%
7550 \else%
7551 \thestanzaL%
7552 \fi%
7553 }{%
7554 \thestanza%
7555 }%
7556 }
7557 %

```

XXVIII.7 Main work

`\stanza@line` Now we arrive at the main works. `\stanza@line` sets the indentation for the line and starts a numbered paragraph—each line is treated as a paragraph. `\stanza@hang` sets the hanging indentation to be used if the stanza line requires more than one print line.

If it is known that each stanza line will fit on one print line, it is advisable to set the hanging indentation to zero. `\sza@penalty` places the specified penalty following each stanza line. By default, this facility is turned off so that no penalty is included. However, the user may initiate these penalties to indicate good and bad places in the stanza for page breaking.

```

7558 \newcommandx{\stanza@line}[2][1,2,usedefault]{%
7559 \ifnum\value{stanzaindentrepetition}=0
7560 \ifcsdef{sza@number\stanza@count @}{%
7561 {%

```

```

7562 \parindent=\csname sza@\number\stanza@count @\endcsname\
stanzaindentbase%
7563 }{%
7564 \led@err@StanzaIndentNotDefined%
7565 }%
7566 \else
7567 \ifcsdef{sza@\number\stanza@modulo @}{%
7568 \parindent=\csname sza@\number\stanza@modulo @\endcsname\
stanzaindentbase%
7569 \managestanza@modulo%
7570 }%
7571 {%
7572 \led@err@StanzaIndentNotDefined%
7573 }%
7574 \fi
7575 \pstart[#1][#2]\stanza@hang\ignorespaces%
7576 }%
7577 \xdef\stanza@hang{\noexpand\leavevmode\noexpand\startlock
7578 \hangindent\expandafter
7579 \noexpand\csname sza@0@\endcsname\stanzaindentbase
7580 \hangafter\@ne}
7581 \def\sza@penalty{\count@\csname szp@\number\stanza@count @\endcsname
7582 \ifnum\count@>\M\advance\count@-\M\penalty-\else
7583 \penalty\fi\count@}
7584 %

```

\@startstanza Now we have the components of the \stanza macro, which appears at the start of a
 \stanza group of lines. This macro initializes the count and checks to see if hanging indentation
 \@stopstanza and penalties are to be included. Hanging indentation suspends the line count, so that
 \AtEveryStopStanza the enumeration is by verse line rather than by print line. If the print line count is
 \AtEveryStanza desired, invoke \let\startlock\relax and do the same for \endlock. Here and
 \AtStartEveryStanza above we have used \xdef to make the stored macros take up a bit less space, but it also
 \BeforeEveryStopStanza makes them more obscure to the reader. Lines of the stanza are delimited by ampersands
 \newverse &. The last line of the stanza must end with \&.

```

7585 \xdef\@startstanza[#1][#2]{%
7586 \noexpand\instanzatrue\expandafter
7587 \begingroup%
7588 \catcode`\noexpand\&\active%
7589 \global\stanza@count\@ne\stanza@modulo\@ne
7590 \noexpand\ifnum\expandafter\noexpand
7591 \csname sza@0@\endcsname=\z@\let\noexpand\stanza@hang\relax
7592 \let\noexpand\endlock\relax\noexpand\else\interlinepenalty
7593 \@M\rightskip\z@ plus 1fil\relax\noexpand\fi\noexpand\ifnum
7594 \expandafter\noexpand\csname szp@0@\endcsname=\z@
7595 \let\noexpand\sza@penalty\relax\noexpand\fi%
7596 \def\noexpand&{%
7597 \noexpand\newverse[] []}%
7598 \def\noexpand\&{\noexpand\@stopstanza}%

```

```

7599 \noexpand\@advancestanzanumber%
7600 \noexpand\stanza@line[#1][#2]%
7601 \noexpand\@insertstanzanumber%
7602 \let\par\relax\ignorespaces%No paragraph in verses
7603 }
7604
7605 \newcommandx{\stanza}[2][1,2,usedefault]{%
7606   \ifboolexpr{%
7607     not test{\ifdefvoid{\at@every@stanza}}%
7608     and test{\ifstrempy{#1}}%
7609     and test{\ifstrempy{#2}}}%
7610   {\@startstanza[][\at@every@stanza]\at@start@every@stanza}%
7611   {\@startstanza[#1][#2]\at@start@every@stanza}%
7612 }%
7613
7614 \newcommandx{\@stopstanza}[2][1,2,usedefault]{%
7615   \unskip%
7616   \endlock%
7617   \ifboolexpr{%
7618     not test{\ifdefvoid{\at@every@stop@stanza}}%
7619     and test{\ifstrempy{#1}}%
7620     and test{\ifstrempy{#2}}}%
7621     {\before@every@stop@stanza\pend[][\at@every@stop@stanza]}%
7622     {\before@every@stop@stanza\pend[#1][#2]}%
7623   \endgroup%
7624   \instanzafalse%
7625 }
7626
7627 \newcommand{\AtEveryStopStanza}[1]{%
7628   \ifstrempy{#1}%
7629     {\gdef\at@every@stop@stanza{}}%
7630     {\gdef\at@every@stop@stanza{\noindent#1}}%
7631 }%
7632 \WithSuffix\newcommand\AtEveryStopStanza*[1]{%
7633   \ifstrempy{#1}%
7634     {\gdef\at@every@stop@stanza{}}%
7635     {\gdef\at@every@stop@stanza{#1}}%
7636 }%
7637 \def\at@every@stop@stanza{}%
7638
7639 \newcommand{\AtEveryStanza}[1]{%
7640   \ifstrempy{#1}%
7641     {\gdef\at@every@stanza{}}%
7642     {\gdef\at@every@stanza{\noindent#1}}%
7643 }%
7644 \WithSuffix\newcommand\AtEveryStanza*[1]{%
7645   \ifstrempy{#1}%
7646     {\gdef\at@every@stanza{}}%
7647     {\gdef\at@every@stanza{#1}}%
7648 }%

```

```

7649
7650
7651
7652 \newcommand{\AtStartEveryStanza}[1]{%
7653   \ifstrempy{#1}%
7654     {\gdef\at@start@every@stanza{}}%
7655     {\gdef\at@start@every@stanza{#1}}%
7656   }%
7657 \def\at@start@every@stanza{}%
7658
7659 \newcommand{\BeforeEveryStopStanza}[1]{%
7660   \ifstrempy{#1}%
7661     {\gdef\before@every@stop@stanza{}}%
7662     {\gdef\before@every@stop@stanza{#1}}%
7663   }%
7664 \def\before@every@stop@stanza{}%
7665
7666 \newcommandx*\newverse[4][1,2,3,4,usedefault]{%
7667   \unskip%
7668   \endlock\pend[#1][#3]\sza@penalty\global%
7669   \advance\stanza@count\@ne\stanza@line[#2][#4]%
7670   }
7671
7672 %

```

\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`.

```

7673 \newcommand*\flagstanza[2][\stanzaindentbase]{%
7674   \hskip -#1\llap{#2}\hskip #1\ignorespaces}
7675
7676 %

```

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.

```

7677 \catcode`\&=\next
7678 \catcode`\@=\body
7679 \setstanzavalues{szp}{0}
7680
7681 %

```


XXIX Apparatus of Manuscripts

XXIX.1 User level macro

`\msdata` The user level `\msdata` command only writes the manuscripts data in numbered auxiliary file. There are two associated etoolbox counters.

```

\msdata@cR
\msdata@cR
\msdata@cR
7682 \def\msdata@c{}%
7683 \def\msdata@cR{}%
7684 \newcommand{\msdata}[1]{%
7685   \leavevmode%
7686   \unless\ifstopmsdata@inserted@%
7687     \stopmsdata%
7688     \led@warning@msdatawithoutstop%
7689   \fi%
7690   \global\stopmsdata@inserted@false%
7691   \unless\ifledRcol%
7692     \numgdef{\msdata@c}{\msdata@c+1}%
7693     \ifdef{\hypertarget}{%
7694       \edlabel{\msdata@c:start:msdata}%
7695     }{}%
7696     \protected@write\linenum@out{}{%
7697       \string\@msd{#1}%
7698     }%
7699   \else%
7700     \numgdef{\msdata@cR}{\msdata@cR+1}%
7701     \ifdef{\hypertarget}{%
7702       \edlabel{\msdata@cR:start:msdata}%
7703     }{}%
7704     \protected@write\linenum@outR{}{%
7705       \string\@msd{#1}%
7706     }%
7707   \fi%
7708 }%
7709 %

```

`\stopmsdata` The user level `\stopmsdata` command only writes information about the end of manuscripts data in numbered auxiliary file.

```

7710 \newcommand{\stopmsdata}[0]{%
7711   \leavevmode%
7712   \unless\ifledRcol%
7713     \protected@write\linenum@out{}{%
7714       \string\@stopmsd%
7715     }%
7716     \ifdef{\hypertarget}{%
7717       \edlabel{\msdata@c:end:msdata}%
7718     }{}%
7719   \else%
7720     \protected@write\linenum@outR{}{%

```

```

7721     \string\@stopmsd%
7722     }%
7723     \ifdef{\hypertarget}{%
7724         \edlabel{\msdata@cR:end:msdata}%
7725     }{}%
7726     \fi%
7727     \global\stopmsdata@inserted@true%
7728 }%
7729 %

```

\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`

```

7730 \newif\ifstopmsdata@inserted@%
7731 %

```

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?

```

7732 \newcommand{\setmsdataseries}[1]{%
7733     \gdef\@msdata@series{#1}%
7734 }%
7735 \def\@msdata@series{A}%
7736 %

```

\setmsdataposition The label for the manuscripts data.

```

7737 \def\ms@data@position{msdata-regular}%
7738 \newcommand{\setmsdataposition}[1]{%
7739     \gdef\ms@data@position{#1}%
7740 }%
7741 %

```

\setmsdatalabel The label for the manuscripts data.

```

7742 \def\ms@data@label{Ms.}%
7743 \newcommand{\setmsdatalabel}[1]{%
7744     \gdef\ms@data@label{#1}%
7745 }%
7746 %

```

XXIX.3 Counters and lists

\@msd@c \@msd@c is a counter incremented at each \@msd read in auxiliary file.

```
7747 \numdef{\@msd@c}{0}
7748 \numdef{\@msd@cR}{0}
7749 %
```

\add@msd@ \add@msd@ is a counter incremented at each \add@msd@data, that is at each time we prepare the insertion of manuscripts data footnote.

```
7750 \numdef{\add@msd@c}{0}%
7751 \numdef{\add@msd@cR}{0}%
7752 %
```

\@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.

```
7753 \def\@msdata@list{}%
7754 %
```

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.

```
7755 \newcommand{\@msd}[1]{%
7756   \unless\ifledRcol%
7757     \global\numdef{\@msd@c}{\@msd@c+\@ne}%
7758     \csgdef{\@msdata@\@msd@c @data}{#1}%
7759     \csxdef{\@msdata@\@msd@c @linenumber}{\the\line@num}%
7760     \csxdef{\@msdata@\@msd@c @abslinenumber}{\the\absline@num}%
7761     \xright@appenditem{\the\absline@num}\to\actionlines@list%
7762     \xright@appenditem{-1010}\to\actions@list%
7763   \else%
7764     \global\numdef{\@msd@cR}{\@msd@cR+\@ne}%
7765     \csgdef{\@msdata@\@msd@cR @dataR}{#1}%
7766     \csxdef{\@msdata@\@msd@cR @linenumberR}{\the\line@numR}%
7767     \csxdef{\@msdata@\@msd@cR @abslinenumberR}{\the\absline@numR}%
7768     \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
7769     \xright@appenditem{-1010}\to\actions@listR%
7770   \fi%
7771 }%
7772 %
```

\@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.

```

7773 \newcommand{\@stopmsd}[0]{%
7774   \unless\ifledRcol%
7775     \ifcsundef{@msdata@\@msd@c @stoplinenumber}{%
7776       \csxdef{@msdata@\@msd@c @stopabslinenumber}{\the\absline@num}%
7777       \csxdef{@msdata@\@msd@c @stoplinenumber}{\the\line@num}%
7778     }{}%
7779   \else%
7780     \ifcsundef{@msdata@\@msd@cR @stoplinenumberR}{%
7781       \csxdef{@msdata@\@msd@cR @stopabslinenumberR}{\the\absline@numR}%
7782       \csxdef{@msdata@\@msd@cR @stoplinenumberR}{\the\line@numR}%
7783     }%
7784     {}%
7785   \fi%
7786 }%
7787 %

```

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.

```

7788 \newcommand{\add@msdata}{%
7789   \bgroup%
7790   \normalfont%
7791   \unless\ifledRcol%
7792     \numgdef{\add@msd@c}{\add@msd@c+\@one}%
7793     \ifcsdef{@msdata@\add@msd@c @data}{%
7794       \letcs{\@data}{@msdata@\add@msd@c @data}%
7795       \edef\l@d@nums{%
7796         000| % Start page = we don't print it
7797         \csuse{@msdata@\add@msd@c @linenumber}| % Start line number
7798         000| % Start subline number, for now, not used
7799         000| % End page number, we don't print it
7800         \ifnumless{\csuse{@msdata@\add@msd@c @stopabslinenumber}}{\csuse{
@lastabsline@forpage@\the\page@num}}}%
7801         {\csuse{@msdata@\add@msd@c @stoplinenumber}}}%End line number if
in the same page
7802         {\csuse{@lastline@forpage@\the\page@num}}}%Otherwiser, last
number of the page
7803         |%
7804         000| % End sub line number, for now, not used
7805         \edfont@info%Font
7806       }%
7807       \@msd@options@fullpagefalse%
7808       \if@firstlineofpage%Try if the data are for the full page. If yes
, will add options to the list.
7809       \unless\if@msdata@insertedfrompreviouspage%

```

```

7810         \ifnumless{\csuse{@lastabsline@forpage@the\page@num}}{\csuse
{ @msdata@ \add@msd@c @stopabslinenumber}+\@ne}%
7811         {%
7812         \numdef{\@tmp}{\add@msd@c+\@ne}%
7813         \ifcsdef{@msdata@\@tmp @abslinenumber}%
7814         {\ifnumequal{\csuse{@msdata@\@tmp @abslinenumber}}{\csuse{
@lastabsline@forpage@the\page@num}}}%
7815         {}%
7816         {\@msd@options@fullpagetrue}%
7817         }%
7818         {\@msd@options@fullpagetrue}%
7819         }%
7820         {}%
7821         \fi%
7822     \fi%
7823     \listxadd{\@msdata@list}{%
7824     \@msd@options@iffullpage%
7825     \ifluatex%
7826     \csxdef{footnote@luatextextdir}{\the\textdir}%
7827     \csxdef{footnote@luatexpardir}{\the\pardir}%
7828     \fi%
7829     \csdef{@this@crossref@start}{\add@msd@c:start:msdata}%
7830     \csdef{@this@crossref@end}{\add@msd@c:end:msdata}%
7831     \noexpand\csuse{v\@msdata@series footnote}{\@msdata@series}{\{
expandonce\l@d@nums}{\ms@data@label}{\expandonce\data}}}%
7832     \reset@msd@options@iffullpage%
7833     }%
7834     }%
7835     {}%
7836     \else%
7837     \numgdef{\add@msd@cR}{\add@msd@cR+\@ne}%
7838     \ifcsdef{@msdata@\add@msd@cR @dataR}{%
7839     \letcs{\@data}{\@msdata@\add@msd@cR @dataR}%
7840     \edef\l@d@nums{%
7841     000| % Start page = we don't print it
7842     \csuse{@msdata@\add@msd@cR @linenumberR}| % Start line number
7843     000| % Start subline number, for now, not used
7844     000| % End page number, we don't print it
7845     \ifnumless{\csuse{@msdata@\add@msd@cR @stopabslinenumberR}}{\
csuse{@lastline@forpageR@the\page@numR}}}%
7846     {\csuse{@msdata@\add@msd@cR @stoplinenumberR}} % End line number
if in the same page
7847     {\csuse{@lastline@forpageR@the\page@numR}} % Otherwiser, last
number of the page
7848     | %
7849     000| % End sub line number, for now, not used
7850     \edef@info{Font
7851     }%
7852     \@msd@options@fullpagefalse%
7853     \if@firstlineofpageR%

```

```

7854 \unless\if@msdata@insertedfrompreviouspage%
7855 \ifnumless{\csuse{@lastabsline@forpageR@the\page@numR}}{\
csuse{@msdata@add@msd@c @stopabslinenumberR}+\@ne}%
7856 {%
7857 \numdef{\@tmp}{\add@msd@cR+\@ne}%
7858 \ifcsdef{@msdata@\@tmp @abslinenumberR}%
7859 {\ifnumequal{\csuse{@msdata@\@tmp @abslinenumberR}}{\csuse{
@lastabsline@forpageR@the\page@numR}}}%
7860 {}}%
7861 {\@msd@options@fullpagetrue}%
7862 }%
7863 {\@msd@options@fullpagetrue}%
7864 }%
7865 {}%
7866 \fi%
7867 \fi%
7868 \listxadd{\@msdata@list}{%
7869 \@msd@options@iffullpage%
7870 \ifluatex%
7871 \csxdef{footnote@luatextextdir}{\the\textdir}%
7872 \csxdef{footnote@luatexpardir}{\the\pardir}%
7873 \fi%
7874 \csdef{@this@crossref@start}{\add@msd@cR:start:msdata}%
7875 \csdef{@this@crossref@end}{\add@msd@cR:end:msdata}%
7876 \noexpand\csuse{v\@msdata@series footnote}{\@msdata@series}{\
expandonce\l@d@nums}{\ms@data@label}{\expandonce\@data}}%
7877 \reset@msd@options@iffullpage%
7878 }%
7879 }%
7880 {}%
7881 \fi%
7882 \egroup%
7883 }%
7884 %

```

`\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.

```

7885 \newif\if@msdata@insertedfrompreviouspage%
7886 %

```

`\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.

```

7887 \newcommand{\add@msdata@firstlineofpage}{%
7888 \bgroup%
7889 \normalfont%
7890 \unless\ifledRcol@%
7891 \ifcsdef{@msdata@\add@msd@c @data}{%

```

```

7892 \ifnumless{\the\absline@num-\@ne}{\csuse{@msdata@\add@msd@c
@stopabslinenumber}}%
7893 {%
7894 \global\@msdata@insertedfrompreviouspagetrue%
7895 \letcs{@data}{@msdata@\add@msd@c @data}%
7896 \edef\l@d@nums{%
7897 000|% Start page = we don't print it
7898 \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
7899 000|% Start subline number, for now, not used
7900 000|% End page number, we don't print it
7901 \ifnumless{\csuse{@msdata@\add@msd@c @stopabslinenumber}}{\csuse{
@lastabsline@forpage@\the\page@num}}%
7902 {\csuse{@msdata@\add@msd@c @stoplinenumber}}%End line number if
in the same page
7903 {\csuse{@lastline@forpage@\the\page@num}}%Otherwise, last
number of the page
7904 |%
7905 000|% End sub line number, for now, not used
7906 \edefont@info%Font
7907 }%
7908 \@msd@options@fullpagefalse%
7909 \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
7910 {%
7911 \numdef{\@tmp}{\add@msd@c+\@ne}%
7912 \ifcsdef{@msdata@\@tmp @abslinenumber}%
7913 {\ifnumequal{\csuse{@msdata@\@tmp @abslinenumber}}{\csuse{
@lastabsline@forpage@\the\page@num}}%
7914 {}%
7915 {\@msd@options@fullpagetrue}%
7916 }%
7917 {\@msd@options@fullpagetrue}%
7918 }%
7919 {}%
7920 \listxadd{\@msdata@list}{%
7921 \@msd@options@iffullpage%
7922 \ifluatex%
7923 \csxdef{footnote@luatextextdir}{\the\textdir}%
7924 \csxdef{footnote@luatexpardir}{\the\pardir}%
7925 \fi%
7926 \csdef{@this@crossref@start}{\add@msd@c:start:msdata}%
7927 \csdef{@this@crossref@end}{\add@msd@c:end:msdata}%
7928 \noexpand\csuse{v\@msdata@series footnote}{\@msdata@series}{\{
expandonce\l@d@nums}{\ms@data@label}{\expandonce\data}}%
7929 \reset@msd@options@iffullpage%
7930 }%
7931 }%

```

```

7932     {\global\@msdata@insertedfrompreviouspagefalse}%
7933   }{}%
7934   \else%
7935     \ifcsdef{@msdata@\add@msd@cR @dataR}{%
7936       \ifnumless{\the\absline@numR-\@one}{\csuse{@msdata@\add@msd@cR
@stopabslinenumberR}}%
7937       {%
7938         \global\@msdata@insertedfrompreviouspagetrue%
7939         \letcs{\@data}{@msdata@\add@msd@cR @dataR}%
7940         \edef\l@d@nums{%
7941           000}% Start page = we don't print it
7942         \numexpr\the\line@numR+\@one\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
7943         000}% Start subline number, for now, not used
7944         000}% End page number, we don't print it
7945         \ifnumless{\csuse{@msdata@\add@msd@cR @stopabslinenumberR}}{\
csuse{@lastline@forpageR@\the\page@numR}}%
7946         {\csuse{@msdata@\add@msd@cR @stoplinenumberR}}%End line number
if in the same page
7947         {\csuse{@lastline@forpageR@\the\page@numR}}%Otherwise, last
number of the page
7948         |%
7949         000}% End sub line number, for now, not used
7950         \edfont@info%Font
7951       }%
7952       \@msd@options@fullpagefalse%
7953       \ifnumless{\csuse{@lastabsline@forpageR@\the\page@numR}}{\csuse{
@msdata@\add@msd@cR @stopabslinenumberR}+\@one}%
7954       {%
7955         \numdef{\@tmp}{\add@msd@cR+\@one}%
7956         \ifcsdef{@msdata@\@tmp @abslinenumberR}%
7957         {\ifnumequal{\csuse{@msdata@\@tmp @abslinenumberR}}{\csuse{
@lastabsline@forpageR@\the\page@numR}}}%
7958         {}%
7959         {\@msd@options@fullpagetrue}%
7960       }%
7961       {\@msd@options@fullpagetrue}%
7962     }%
7963   {}%
7964   \listxadd{\@msdata@list}{%
7965     \@msd@options@iffullpage%
7966     \ifluatex%
7967       \csxdef{footnote@luatextextdir}{\the\textdir}%
7968       \csxdef{footnote@luatexpardir}{\the\pardir}%
7969     \fi%
7970     \csdef{@this@crossref@start}{\add@msd@cR:start:msdata}%
7971     \csdef{@this@crossref@end}{\add@msd@cR:end:msdata}%
7972     \noexpand\csuse{v@msdata@series footnote}{\@msdata@series}{\
expandonce\l@d@nums}{\ms@data@label}{\expandonce\@data}}%

```



```

7973 \reset@msd@options@iffullpage%
7974 }%
7975 }%
7976 {\global\@msdata@insertedfrompreviouspagefalse}%
7977 }{}%
7978 \fi%
7979 \egroup%
7980 }%
7981 %

```

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%2 \newcommand{\insert@msdata}{%
7983 \def\do##1{##1}%
7984 \dolistloop{\@msdata@list}%
7985 \global\let\@msdata@list\relax%
7986 }%
7987 %

```

XXIX.7 Other

`\@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.

```

7988 \newif\if@msd@options@fullpage%
7989 \newcommand{\@msd@options@iffullpage}[0]{%
7990 \if@msd@options@fullpage%
7991 \noexpand\toggletrue{nonum@}%
7992 \ifdefvoid{\ms@data@label}%
7993 {\noexpand\toggletrue{nosep@}}%
7994 }%
7995 \fi%
7996 }%
7997 \newcommand{\reset@msd@options@iffullpage}[0]{%
7998 \noexpand\togglefalse{nonum@}%
7999 \noexpand\togglefalse{nosep@}%
8000 }%
8001 %

```

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.

```
8002 \newtoks\@emptytoks
8003
8004 %
```

The rest is from `amsmath`.

`\l@denvbody` A token register to contain the body.

```
8005 \newtoks\l@denvbody
8006
8007 %
```

`\addtol@denvbody` `\addtol@denvbody{arg}` adds `arg` to the token register `\l@denvbody`.

```
8008 \newcommand{\addtol@denvbody}[1]{%
8009   \global\l@denvbody\expandafter{\the\l@denvbody#1}}
8010
8011 %
```

`\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`.

```
8012 \newcommand{\l@dcollect@body}[1]{%
8013   \l@denvbody{\expandafter#1\expandafter{\the\l@denvbody}}}%
8014   \edef\processl@denvbody{\the\l@denvbody\noexpand\end{\@currenvir}}}%
8015   \l@denvbody\@emptytoks \def\l@dbegin@stack{b}%
8016   \begingroup
8017     \expandafter\let\csname\@currenvir\endcsname\l@dcollect@@body
8018     \edef\processl@denvbody{\expandafter\noexpand\csname\@currenvir\endcsname}%
8019     \processl@denvbody%
8020   }%
8021
8022 %
```

`\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.

```
8023 \def\l@dpush@begins#1\begin#2{%
8024   \ifx\end#2\else b\expandafter\l@dpush@begins\fi}
8025
8026 %
```

`\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.

```
8027 \def\l@dcollect@@body#1\end#2{%
8028   \edef\l@dbegin@stack{\l@dpush@begins#1\begin\end
8029     \expandafter\@gobble\l@dbegin@stack}%
8030   \ifx\@empty\l@dbegin@stack
8031     \endgroup
8032     \@checkend{#2}%
8033     \addtol@denvbody{#1}%
8034   \else
8035     \addtol@denvbody{#1\end{#2}}%
8036   \fi
8037   \processl@denvbody % A little tricky! Note the grouping
8038 }
8039
8040 %
```

There was a question on CTT about how to use `\collect@body` for a macro taking an argument. The following is part of that thread.

From: Heiko Oberdiek <oberdiek@uni-freiburg.de>
 Newsgroups: comp.text.tex
 Subject: Re: Using `\collect@body` with commands that take >1 argument
 Date: Fri, 08 Aug 2003 09:03:20 +0200

eed132@psu.edu (Evan) wrote:
 > I'm trying to make a new Latex environment that acts like the
 > `\colorbox` command that is part of the `color` package. I looked through
 > the FAQ and ran across this bit about using the `\collect@body` command
 > that is part of AMSLaTeX:
 > <http://www.tex.ac.uk/cgi-bin/texfaq2html?label=cmdasenv>
 >
 > It almost works. If I do something like the following:
 > `\newcommand{\redbox}[1]{\colorbox{red}{#1}}`
 >
 > `\makeatletter`
 > `\newenvironment{redbox}{\collect@body \redbox}{}`

You will get an error message: Command `\redbox` already defined.
Thus you must rename either the command `\redbox` or the environment name.

```
> \begin{coloredbox}{blue}
>   Yadda yadda yadda... this is on a blue background...
> \end{coloredbox}
> and can't figure out how to make the \collect@body take this.

> \collect@body \colorbox{red}
> \collect@body {\colorbox{red}}
```

The argument of `\collect@body` has to be one token exactly.

```
\documentclass{article}
\usepackage{color}
\usepackage{amsmath}

\newcommand{\redbox}[1]{\colorbox{red}{#1}}
\makeatletter
\newenvironment{coloredbox}[1]{%
  \def\next@{\colorbox{#1}}%
  \collect@body\next@
}{%

% ignore spaces at begin and end of environment
\newenvironment{coloredboxII}[1]{%
  \def\next@{\mycoloredbox{#1}}%
  \collect@body\next@
}{%
\newcommand{\mycoloredbox}[2]{%
  \colorbox{#1}{\ignorespaces#2\unskip}%
}

% support of optional color model argument
\newcommand\coloredboxIII\endcsname{}
\def\coloredboxIII#1#2{%
  \@coloredboxIII{#1}%
}
\def\@coloredboxIII#1#2{%
  \def\next@{\mycoloredboxIII{#1}{#2}}%
  \collect@body\next@
}
\newcommand{\mycoloredboxIII}[3]{%
  \colorbox{#1}{#2}{\ignorespaces#3\unskip}%
}

\makeatother
```

```

\begin{document}
  Black text before
  \begin{coloredbox}{blue}
    Hello World
  \end{coloredbox}
  Black text after

  Black text before
  \begin{coloredboxII}{blue}
    Hello World
  \end{coloredboxII}
  Black text after

  Black text before
  \begin{coloredboxIII}[rgb]{0,0,1}
    Hello World
  \end{coloredboxIII}
  Black text after

\end{document}

Yours sincerely
Heiko <oberdiek@uni-freiburg.de>

```

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.

```

8041 \newcommand*{\l@dtabnoexpands}{%
8042   \let\rtab=0%
8043   \let\ctab=0%
8044   \let\ltab=0%
8045   \let\rtabtext=0%
8046   \let\ltabtext=0%
8047   \let\ctabtext=0%
8048   \let\edbeforetab=0%
8049   \let\edaftertab=0%
8050   \let\edatleft=0%

```

```

8051 \let\edatright=0%
8052 \let\edvertline=0%
8053 \let\edvertdots=0%
8054 \let\edrowfill=0%
8055 }
8056
8057 %

```

\disable@familiarnotes Macros to disable and restore familiar notes, to prevent them from printing multiple times in edtabularx and edarrayx environments.

\restore@familiarnotes

```

8058 \newcommand{\disable@familiarnotes}{%
8059   \unless\ifnofamiliar@%
8060     \def\do##1{%
8061       \csletcs{footnote@@##1}{footnote##1}%
8062       \expandafter\renewcommand \csname footnote##1\endcsname[1]{%
8063         \protected@csxdef{@thefnmark##1}{\csuse{thefootnote##1}}%
8064         \csuse{@footnotemark##1}%
8065       }%
8066     }%
8067     \dolistloop{\@series}%
8068   \fi%
8069 }%
8070 \newcommand{\restore@familiarnotes}{%
8071   \unless\ifnofamiliar@%
8072     \def\do##1{%
8073       \csletcs{footnote##1}{footnote@@##1}%
8074     }%
8075     \dolistloop{\@series}%
8076   \fi%
8077 }%
8078
8079 %

```

\disable@sidenotes The same, for side notes.

\restore@sidenotes

```

8080 \newcommand{\disable@sidenotes}{%
8081   \let\@@ledrightnote\ledrightnote%
8082   \let\@@ledleftnote\ledleftnote%
8083   \let\@@ledsidenote\ledsidenote%
8084   \let\ledrightnote@gobble%
8085   \let\ledleftnote@gobble%
8086   \let\ledsidenote@gobble%
8087 }%
8088 \newcommand{\restore@sidenotes}{%
8089   \let\ledrightnote\@@ledrightnote%
8090   \let\ledleftnote\@@ledleftnote%
8091   \let\ledsidenote\@@ledsidenote%
8092 }%
8093 %

```

`\disable@notes` Disable/restore side and familiar notes.

```
\restore@notes
8094 \newcommand{\disable@notes}{%
8095   \disable@sidenotes%
8096   \disable@familiarnotes%
8097 }%
8098 \newcommand{\restore@notes}{%
8099   \restore@sidenotes%
8100   \restore@familiarnotes%
8101 }%
8102 %
```

`\EDTEXT` We need to be able to modify the `\edtext` macros and also restore their original definitions.
`\xedtext`

```
8103 \let\EDTEXT=\edtext
8104 \newcommand{\xedtext}[2]{\EDTEXT{#1}{#2}}
8105 %
```

`\EDLABEL` We need to be able to modify and restore the `\edlabel` macro.

```
\xedlabel
8106 \let\EDLABEL=\edlabel
8107 \newcommand*{\xedlabel}[1]{\EDLABEL{#1}}
8108 %
```

`\xedindex` Macros supporting modification and restoration of `\edindex`.

```
\nulledindex
8109 \AtBeginDocument{\let\xedindex\edindex}%
8110 \newcommand{\nulledindex}[2][\jobname]{\@bsphack\@esphack}
8111
8112 %
```

`\@line@num` Macro supporting restoration of `\linenum`.

```
8113 \let\@line@num=\linenum
8114 %
```

`\l@dgobblearg` `\l@dgobbleoptarg[⟨arg⟩]{⟨arg⟩}` replaces these two arguments (first is optional) by `\relax`.

```
8115 \newcommand*{\l@dgobbleoptarg}[2][\relax]%
8116
8117 %
```

`\Relax` `\let\Relax=\relax`

`\NEXT` `\let\NEXT=\next`

```
8120
8121 %
```

`\l@modforedtext` Modify and restore various macros for when `\edtext` is used.
`\l@drestoreforedtext`

```

8122 \newcommand{\l@modforedtext}{%
8123   \let\edtext\relax
8124   \def\do##1{\global\csletcs{##1footnote}{\l@dgobbleoptarg}}%
8125   \dolistloop{\@series}%
8126   \let\edindex\nulledindex
8127   \let\linenum@gobble}
8128 \newcommand{\l@drestoreforedtext}{%
8129   \def\do##1{\global\csletcs{##1footnote}{##1@footnote}}
8130   \dolistloop{\@series}%
8131   \let\edindex\xedndex}
8132 %

```

`\l@dnnullfills` Nullify and restore some column fillers, etc.

`\l@drestorefills`

```

8133 \newcommand{\l@dnnullfills}{%
8134   \def\edlabel##1{%
8135     \def\edrowfill##1##2##3}%
8136   }
8137 \newcommand{\l@drestorefills}{%
8138   \def\edrowfill##1##2##3{\@EDROWFILL@{##1}{##2}{##3}}%
8139   }
8140
8141 %

```

`\letsforverteilen` Gathers some lets and other code that is common to the `*verteilen*` macros.

```

8142 \newcommand{\letsforverteilen}{%
8143   \let\edtext\xeddtext
8144   \let\edindex\xedndex
8145   \def\do##1{\global\csletcs{##1footnote}{##1@footnote}}
8146   \dolistloop{\@series}%
8147   \let\linenum@line@num
8148   \hilfsskip=\l@dcolwidth%
8149   \advance\hilfsskip by -\wd\hilfsbox
8150   \def\edlabel##1{\xedlabel{##1}}
8151
8152 %

```

`\disablel@dtabfeet` Declarations for using or using `\edtext` inside tabulars. The default at this point is for
`\enablel@dtabfeet` `\edtext`.

```

8153 \newcommand\disablel@dtabfeet{\l@modforedtext}%
8154 \newcommand\enablel@dtabfeet{\l@drestoreforedtext}%
8155 %

```


XXX.2.2 Counters, boxes and lengths

`\l@dampcount` `\l@dampcount` is a counter for the & column dividers and `\l@dcolcount` is a counter for the columns.

```

8156 \newcount\l@dampcount
8157 \l@dampcount=1\relax
8158 \newcount\l@dcolcount
8159 \l@dcolcount=0\relax
8160
8161 %

```

`\hilfsbox` Some (temporary) helper items.

```

\hilfsskip
\Hilfsbox
\hilfscount
8162 \newbox\hilfsbox
8163 \newskip\hilfsskip
8164 \newbox\Hilfsbox
8165 \newcount\hilfscount
8166
8167 %

```

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).

```

8168 \newdimen\dcoli
8169 \newdimen\dcolii
8170 \newdimen\dcoliii
8171 \newdimen\dcoliv
8172 \newdimen\dcolv
8173 \newdimen\dcolvi
8174 \newdimen\dcolvii
8175 \newdimen\dcolviii
8176 \newdimen\dcolix
8177 \newdimen\dcolx
8178 \newdimen\dcolxi
8179 \newdimen\dcolxii
8180 \newdimen\dcolxiii
8181 \newdimen\dcolxiv
8182 \newdimen\dcolxv
8183 \newdimen\dcolxvi
8184 \newdimen\dcolxvii
8185 \newdimen\dcolxviii
8186 \newdimen\dcolxix
8187 \newdimen\dcolxx
8188 \newdimen\dcolxxi
8189 \newdimen\dcolxxii
8190 \newdimen\dcolxxiii
8191 \newdimen\dcolxxiv
8192 \newdimen\dcolxxv
8193 \newdimen\dcolxxvi
8194 \newdimen\dcolxxvii

```

```

8195 \newdimen\dcollxxviii
8196 \newdimen\dcollxxix
8197 \newdimen\dcollxxx
8198 \newdimen\dcollerr    % added for error handling
8199
8200 %

```

\l@dcollwidth This is a cunning way of storing the columnwidths indexed by the column number \l@dcollcount, like an array. (was \Dimenzuordnung)

```

8201 \newcommand{\l@dcollwidth}{\ifcase \the\l@dcollcount \dcoli %???
8202 \or \dcoli \or \dcolii \or \dcoliii
8203 \or \dcoliv \or \dcolv \or \dcolvi
8204 \or \dcolvii \or \dcolviii \or \dcolix \or \dcolx
8205 \or \dcolxi \or \dcolxii \or \dcolxiii
8206 \or \dcolxiv \or \dcolxv \or \dcolxvi
8207 \or \dcolxvii \or \dcolxviii \or \dcolxix \or \dcolxx
8208 \or \dcolxxi \or \dcolxxii \or \dcolxxiii
8209 \or \dcolxxiv \or \dcolxxv \or \dcolxxvi
8210 \or \dcolxxvii \or \dcolxxviii \or \dcolxxix \or \dcolxxx
8211 \else \dcollerr \fi}
8212
8213 %

```

\step1@dcollcount This increments the column counter, and issues an error message if it is too large.

```

8214 \newcommand*\step1@dcollcount{\advance\l@dcollcount\@ne
8215 \ifnum\l@dcollcount>30\relax
8216 \led@err@TooManyColumns
8217 \fi}
8218
8219 %

```

\l@dsetmaxcollwidth Sets the column width to the maximum value seen so far.

```

8220 \newcommand{\l@dsetmaxcollwidth}{%
8221 \ifdim\l@dcollwidth < \wd\hilsbox
8222 \l@dcollwidth = \wd\hilsbox
8223 \else \relax \fi}
8224
8225 %

```

\measuremcell Measure (recursively) the width required for a math cell.

```

8226 \def\measuremcell #1{%
8227 \ifx #1\\ \ifnum\l@dcollcount=0\let\NEXT\relax%
8228 \else\l@dcheckcols%
8229 \l@dcollcount=0%
8230 \let\NEXT\measuremcell%
8231 \fi%

```

```

8232 \else\setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
8233 \step1@dcolcount%
8234 \l@setmaxcolwidth%
8235 \let\NEXT\measuremcell%
8236 \fi\NEXT}
8237
8238 %

```

\measuretcell Measure (recursively) the width required for a text cell.

```

8239 \def\measuretcell #1{%
8240 \ifx #1\ \ifnum\l@dcolcount=0\let\NEXT\relax%
8241 \else\l@dcheckcols%
8242 \l@dcolcount=0%
8243 \let\NEXT\measuretcell%
8244 \fi%
8245 \else\setbox\hilfsbox=\hbox{#1}%
8246 \step1@dcolcount%
8247 \l@setmaxcolwidth%
8248 \let\NEXT\measuretcell%
8249 \fi\NEXT}
8250
8251 %

```

\measuremrow Measure (recursively) the width required for a math row.

```

8252 \def\measuremrow #1{%
8253 \ifx #1&\let\NEXT\relax%
8254 \else\measuremcell #1&\&\&%
8255 \let\NEXT\measuremrow%
8256 \fi\NEXT}
8257 %

```

\measuretrow Measure (recursively) the width required for a text row.

```

8258 \def\measuretrow #1{%
8259 \ifx #1&\let\NEXT\relax%
8260 \else\measuretcell #1&\&\&%
8261 \let\NEXT\measuretrow%
8262 \fi\NEXT}
8263
8264 %

```

\edtabcolsep The length \edtabcolsep controls the distance between columns.

```

8265 \newskip\edtabcolsep
8266 \global\edtabcolsep=10pt
8267
8268 %

```

```
\variab69 \newcommand{\variab}{\relax}
```

```
8270
```

```
8271 %
```

\l@dccheckcols Check that the number of columns is consistent.

```
8272 \newcommand*{\l@dccheckcols}{%
8273   \ifnum\l@dcolcount=1\relax
8274   \else
8275     \ifnum\l@dampcount=1\relax
8276     \else
8277       \ifnum\l@dcolcount=\l@dampcount\relax
8278       \else
8279         \l@d@err@UnequalColumns
8280       \fi
8281     \fi
8282     \l@dampcount=\l@dcolcount
8283   \fi}
8284
8285 %
```

\edfilldimen A length.

```
8286 \newdimen\edfilldimen
8287 \edfilldimen=0pt
8288
8289 %
```

\c@addcolcount A counter to hold the number of a column. We use a roman number so that we can grab the column dimension from `\dcol`. We do not use the `\roman` \TeX command, because some packages, like `babel` can override it in some specific cases (Greek, for example).

\theadcolcount

```
8290 \newcounter{addcolcount}
8291 \renewcommand{\theadcolcount}{\romannumeral \c@addcolcount}
8292 %
```

XXX.2.3 Tabular typesetting

\setmcellright Typeset (recursively) cells of display math right justified.

```
8293 \def\setmcellright #1&{\def\edlabel##1}%
8294   \let\edindex\nulledindex
8295   \ifx #1\\ \ifnum\l@dcolcount=0%\removelastskip
8296     \let\Next\relax%
8297   \else\l@dcolcount=0%
8298     \let\Next=\setmcellright%
8299   \fi%
8300 \else%
8301   \disablel@dtabfeet%
```

```

8302         \stepl@dcolcount%
8303         \disable@notes%
8304         \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
8305         \restore@notes%
8306         \letsforverteilen%
8307         \hskip\hilfsskip$\displaystyle{#1}$%
8308         \hskip\edtabcolsep%
8309         \let\Next=\setmcellright%
8310     \fi\Next}
8311
8312 %

```

\settcellright Typeset (recursively) cells of text right justified.

```

8313 \def\settcellright #1&{\def\edlabel##1{}%
8314     \let\edindex\nulledindex
8315     \ifx #1\\ \ifnum\l@dcolcount=0\removelastskip
8316         \let\Next\relax%
8317     \else\l@dcolcount=0%
8318         \let\Next=\settcellright%
8319     \fi%
8320 \else%
8321     \disablel@dtabfeet%
8322     \stepl@dcolcount%
8323     \disable@notes%
8324     \setbox\hilfsbox=\hbox{#1}%
8325     \restore@notes%
8326     \letsforverteilen%
8327     \hskip\hilfsskip#1%
8328     \hskip\edtabcolsep%
8329     \let\Next=\settcellright%
8330 \fi\Next}
8331 %

```

\setmcellleft Typeset (recursively) cells of display math left justified.

```

8332 \def\setmcellleft #1&{\def\edlabel##1{}%
8333     \let\edindex\nulledindex
8334     \ifx #1\\ \ifnum\l@dcolcount=0 \let\Next\relax%
8335     \else\l@dcolcount=0%
8336         \let\Next=\setmcellleft%
8337     \fi%
8338 \else \disablel@dtabfeet%
8339     \stepl@dcolcount%
8340     \disable@notes%
8341     \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
8342     \restore@notes%
8343     \letsforverteilen%
8344     $\displaystyle{#1}$\hskip\hilfsskip\hskip\edtabcolsep%
8345     \let\Next=\setmcellleft%

```

```

8346 \fi\Next}
8347
8348 %

```

\settcclleft Typeset (recursively) cells of text left justified.

```

8349 \def\settcclleft #1{\def\edlabel##1{%
8350   \let\edindex\nulledindex
8351   \ifx #1\\ \ifnum\l@dc@lcount=0 \let\Next\relax%
8352     \else\l@dc@lcount=0%
8353       \let\Next=\settcclleft%
8354     \fi%
8355   \else \disablel@dtabfeet%
8356     \stepl@dc@lcount%
8357     \disable@notes%
8358     \setbox\hilfsbox=\hbox{#1}%
8359     \restore@notes%
8360     \letsforverteilen%
8361     #1\hskip\hilfsskip\hskip\edtabcolsep%
8362     \let\Next=\settcclleft%
8363   \fi\Next}
8364 %

```

\setmcellcenter Typeset (recursively) cells of display math centered.

```

8365 \def\setmcellcenter #1{\def\edlabel##1{%
8366   \let\edindex\nulledindex
8367   \ifx #1\\ \ifnum\l@dc@lcount=0\let\Next\relax%
8368     \else\l@dc@lcount=0%
8369       \let\Next=\setmcellcenter%
8370     \fi%
8371   \else \disablel@dtabfeet%
8372     \stepl@dc@lcount%
8373     \disable@notes%
8374     \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
8375     \restore@notes%
8376     \letsforverteilen%
8377     \hskip 0.5\hilfsskip$\displaystyle{#1}$\hskip0.5\hilfsskip%
8378     \hskip\edtabcolsep%
8379     \let\Next=\setmcellcenter%
8380   \fi\Next}
8381
8382 %

```

\settcclcenter Typeset (recursively) cells of text centered.

```

8383 \def\settcclcenter #1{\def\edlabel##1{%
8384   \let\edindex\nulledindex
8385   \ifx #1\\ \ifnum\l@dc@lcount=0 \let\Next\relax%
8386     \else\l@dc@lcount=0%

```

```

8387         \let\Next=\settcellcenter%
8388         \fi%
8389     \else \disablel@dtabfeet%
8390         \stepl@dcolcount%
8391         \disable@notes%
8392         \setbox\hilfsbox=\hbox{#1}%
8393         \restore@notes%
8394         \letsforverteilen%
8395         \hskip 0.5\hilfsskip #1\hskip 0.5\hilfsskip%
8396         \hskip\edtabcolsep%
8397         \let\Next=\settcellcenter%
8398     \fi\Next}
8399
8400 %

```

```

\NEXT01 \let\NEXT=\relax

```

```

8402
8403 %

```

\setmrowright Typeset (recursively) rows of right justified math.

```

8404 \def\setmrowright #1\{ %
8405     \ifx #1& \let\NEXT\relax
8406     \else \centerline{\setmcellright #1&\\&\\&}
8407         \let\NEXT=\setmrowright
8408     \fi\NEXT}
8409 %

```

\settroright Typeset (recursively) rows of right justified text.

```

8410 \def\settroright #1\{ %
8411     \ifx #1& \let\NEXT\relax
8412     \else \centerline{\settcellright #1&\\&\\&}
8413         \let\NEXT=\settroright
8414     \fi\NEXT}
8415
8416 %

```

\setmrowleft Typeset (recursively) rows of left justified math.

```

8417 \def\setmrowleft #1\{ %
8418     \ifx #1& \let\NEXT\relax
8419     \else \centerline{\setmcellleft #1&\\&\\&}
8420         \let\NEXT=\setmrowleft
8421     \fi\NEXT}
8422 %

```

\settroright Typeset (recursively) rows of left justified text.

```

8423 \def\settrorleft #1\{\%
8424   \ifx #1& \let\NEXT\relax
8425   \else \centerline{\settcclleft #1&\&\&\&}
8426   \let\NEXT=\settrorleft
8427   \fi\NEXT}
8428
8429 %

```

\setmrowcenter Typeset (recursively) rows of centered math.

```

8430 \def\setmrowcenter #1\{\%
8431   \ifx #1& \let\NEXT\relax%
8432   \else \centerline{\setmcellcenter #1&\&\&\&}
8433   \let\NEXT=\setmrowcenter
8434   \fi\NEXT}
8435 %

```

\settrorcenter Typeset (recursively) rows of centered text.

```

8436 \def\settrorcenter #1\{\%
8437   \ifx #1& \let\NEXT\relax
8438   \else \centerline{\settrcellcenter #1&\&\&\&}
8439   \let\NEXT=\settrorcenter
8440   \fi\NEXT}
8441
8442 %

```

\nullsetzen `\newcommand{\nullsetzen}{\%`

```

8444   \stepl@dc@colcount%
8445   \l@dc@colwidth=0pt%
8446   \ifnum\l@dc@colcount=30\let\NEXT\relax%
8447   \l@dc@colcount=0\relax
8448   \else\let\NEXT\nullsetzen%
8449   \fi\NEXT}
8450
8451 %

```

\edatleft `\edatleft[⟨math⟩]{⟨symbol⟩}{⟨len⟩}`. Left ⟨symbol⟩, 2⟨len⟩ high with prepended ⟨math⟩ vertically centered.

```

8452 \newcommand{\edatleft}[3][\@empty]{\%
8453   \ifx#1\@empty
8454     \vbox to 10pt{\vss\hbox{$\left#2\vrule width0pt height #3
8455       depth 0pt \right. $\hss}\vfil}
8456   \else
8457     \vbox to 4pt{\vss\hbox{$#1\left#2\vrule width0pt height #3
8458       depth 0pt \right. $\}\vfil}
8459   \fi}
8460 %

```


\edatright `\edatright[$\langle math \rangle$]{ $\langle symbol \rangle$ }{ $\langle len \rangle$ }`. Right $\langle symbol \rangle$, $2\langle len \rangle$ high with appended $\langle math \rangle$ vertically centered.

```

8461 \newcommand{\edatright}[3][\@empty]{%
8462   \ifx#1\@empty
8463     \vbox to 10pt{\vss\hbox{$\left.\vrule width0pt height #3
8464       depth 0pt \right#2 $\hss}\vfil}
8465   \else
8466     \vbox to 4pt{\vss\hbox{$\left.\vrule width0pt height #3
8467       depth 0pt \right#2 #1 $\vfil}
8468   \fi}
8469
8470 %

```

\edvertline `\edvertline{ $\langle len \rangle$ }` vertical line $\langle len \rangle$ high.

```

8471 \newcommand{\edvertline}[1]{\vbox to 8pt{\vss\hbox{\vrule height #1}\vfil}}
8472
8473 %

```

\edvertdots `\edvertdots{ $\langle len \rangle$ }` vertical dotted line $\langle len \rangle$ high.

```

8474 \newcommand{\edvertdots}[1]{\vbox to 1pt{\vss\vbox to #1%
8475   {\cleaders\hbox{$\math\hbox{.}\vbox to 0.5em{ }\vfil}}}}
8476
8477 %

```

\l@dtabaddcols `\l@dtabaddcols{ $\langle startcol \rangle$ }{ $\langle endcol \rangle$ }` adds the widths of the columns $\langle startcol \rangle$ through $\langle endcol \rangle$ to `\edfilldimen`. It is a \TeX style reimplementaion of the original `\@add@`.

```

8478 \newcommand{\l@dtabaddcols}[2]{%
8479   \l@dccheckstartend{#1}{#2}%
8480   \ifl@dstartendok
8481     \setcounter{addcolcount}{#1}%
8482     \@whilenum \value{addcolcount}<#2\relax \do
8483     {\advance\edfilldimen by \the \csname dcol\theadcolcount\endcsname
8484      \advance\edfilldimen by \edtabcolsep
8485      \stepcounter{addcolcount}}%
8486     \advance\edfilldimen by \the \csname dcol\theadcolcount\endcsname
8487   \fi
8488 }
8489
8490 %

```

\ifl@dstartendok `\l@dccheckstartend{ $\langle startcol \rangle$ }{ $\langle endcol \rangle$ }` checks that the values of $\langle startcol \rangle$ and $\langle endcol \rangle$ are sensible. If they are then `\ifl@dstartendok` is set TRUE, otherwise it is set FALSE.

```

8491 \newif\ifl@dstartendok
8492 \newcommand{\l@dccheckstartend}[2]{%

```

```

8493 \l@dstartendoktrue
8494 \ifnum #1<\@ne
8495   \l@dstartendokfalse
8496   \led@err@LowStartColumn
8497 \fi
8498 \ifnum #2>30\relax
8499   \l@dstartendokfalse
8500   \led@err@HighEndColumn
8501 \fi
8502 \ifnum #1>#2\relax
8503   \l@dstartendokfalse
8504   \led@err@ReverseColumns
8505 \fi
8506 }
8507
8508 %

```

`\edrowfill` `\edrowfill{<startcol>}{<endcol>}` fill fills columns `<startcol>` to `<endcol>` inclusive with `<fill>` (e.g. `\hrulefill`, `\upbracefill`). This is a \TeX style reimplementation and generalization of the original `\waklam`, `\Waklam`, `\waklamec`, `\wastricht` and `\wapunktel` macros.

```

8509 \newcommand*{\edrowfill}[3]{%
8510   \l@dtabaddcols{#1}{#2}%
8511   \hb@xt@ \the\l@dcolwidth{\hb@xt@ \the\edfilldimen{#3}\hss}}
8512 \let\@edrowfill=\edrowfill
8513 \def\@EDROWFILL@#1#2#3{\@edrowfill@{#1}{#2}{#3}}
8514
8515 %

```

`\edbeforetab` The macro `\edbeforetab{<text>}{<math>}` puts `<text>` at the left margin before array cell entry `<math>`. Conversely, the macro `\edaftertab{<math>}{<text>}` puts `<text>` at the right margin after array cell entry `<math>`. `\edbeforetab` should be in the first column and `\edaftertab` in the last column. The following macros support these.

`\leftltab` `\leftltab{<text>}` for `\edbeforetab` in `\ltab`.

```

8516 \newcommand{\leftltab}[1]{%
8517   \hb@xt@\z@{\vbox{\edtabindent%
8518     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
8519
8520 %

```

`\leftrtab` `\leftrtab{<text>}{<math>}` for `\edbeforetab` in `\rtab`.

```

8521 \newcommand{\leftrtab}[2]{%
8522   #2\hb@xt@\z@{\vbox{\edtabindent%
8523     \advance\Hilfsskip by\dcoli%
8524     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
8525
8526 %

```

`\leftctab` `\leftctab{<text>}{<math>}` for `\edbeforetab` in `\ctab`.

```

8527 \newcommand{\leftctab}[2]{%
8528     \hb@xt@\z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
8529     \advance\Hilfsskip by 0.5\dcoli%
8530     \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
8531     \disablel@dtabfeet$\displaystyle{#2}$}%
8532     \advance\Hilfsskip by -0.5\wd\hilfsbox%
8533     \moveleft\Hilfsskip\hbox{\ #1}}\hss}%
8534     #2}
8535
8536 %

```

`\rightctab` `\rightctab{<math>}{<text>}` for `\edaftertab` in `\ctab`.

```

8537 \newcommand{\rightctab}[2]{%
8538     \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
8539     \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
8540     #1\hb@xt@\z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
8541     \advance\Hilfsskip by 0.5\l@dcolwidth%
8542     \advance\Hilfsskip by -\wd\hilfsbox%
8543     \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
8544     \disablel@dtabfeet$\displaystyle{#1}$}%
8545     \advance\Hilfsskip by -0.5\wd\hilfsbox%
8546     \advance\Hilfsskip by \edtabcolsep%
8547     \moveright\Hilfsskip\hbox{ #2}}\hss}%
8548     }
8549
8550 %

```

`\rightltab` `\rightltab{<math>}{<text>}` for `\edaftertab` in `\ltab`.

```

8551 \newcommand{\rightltab}[2]{%
8552     \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
8553     \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
8554     #1\hb@xt@\z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
8555     \advance\Hilfsskip by\l@dcolwidth%
8556     \advance\Hilfsskip by-\wd\hilfsbox%
8557     \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
8558     \disablel@dtabfeet$\displaystyle{#1}$}%
8559     \advance\Hilfsskip by-\wd\hilfsbox%
8560     \advance\Hilfsskip by\edtabcolsep%
8561     \moveright\Hilfsskip\hbox{ #2}}\hss}%
8562     }
8563
8564 %

```

`\rightrtab` `\rightrtab{<math>}{<text>}` for `\edaftertab` in `\rtab`.

```

8565 \newcommand{\rightrtab}[2]{%

```

```

8566 \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
8567 \disablel@dtabfeet#2}%
8568 #1\hb@xt@{z@{\vbox{\edtabindent%
8569 \advance\Hilfsskip by-\wd\hilfsbox%
8570 \advance\Hilfsskip by\edtabcolsep%
8571 \moveright\Hilfsskip\hbox{ #2}}\hss}%
8572 }
8573
8574 %

```

\rtab `\rtab{<body>}` typesets `<body>` as an array with the entries right justified.
\edbeforetab The process is first to measure the `<body>` to get the column widths, and then in a
\edaftertab second pass to typeset the body.

```

8575 \newcommand{\rtab}[1]{%
8576 \l@dnnullfills
8577 \def\edbeforetab##1##2{\lefttrtab{##1}{##2}}%
8578 \def\edaftertab##1##2{\righttrtab{##1}{##2}}%
8579 \measurebody{#1}%
8580 \l@drestorefills
8581 \variab
8582 \setmrowright #1\\&\\%
8583 \enablel@dtabfeet}
8584
8585 %

```

\measurebody `\measurebody{<body>}` measures the array `<body>`.

```

8586 \newcommand{\measurebody}[1]{%
8587 \disablel@dtabfeet%
8588 \l@dcolcount=0%
8589 \nullsetzen%
8590 \l@dcolcount=0
8591 \measuremrow #1\\&\\%
8592 \global\l@dampcount=1}
8593
8594 %

```

\rtabtext `\rtabtext{<body>}` typesets `<body>` as a tabular with the entries right justified.

```

8595 \newcommand{\rtabtext}[1]{%
8596 \l@dnnullfills
8597 \measuretbody{#1}%
8598 \l@drestorefills
8599 \variab
8600 \settrrowright #1\\&\\%
8601 \enablel@dtabfeet}
8602
8603 %

```

`\measuretbody` `\measuretbody{<body>}` measures the tabular `<body>`.

```

8604 \newcommand{\measuretbody}[1]{%
8605   \disable@notes%
8606   \disablel@dtabfeet%
8607   \l@dc@colcount=0%
8608   \nullsetzen%
8609   \l@dc@colcount=0
8610   \measuretrrow #1\\&\\%
8611   \restore@notes%
8612   \global\l@dampcount=1}
8613
8614 %

```

`\ltab` Array with entries left justified.

```

\edbeforetab
\edaftertab
8615 \newcommand{\ltab}[1]{%
8616   \l@dnullfills
8617   \def\edbeforetab##1##2{\leftltab{##1}{##2}}%
8618   \def\edaftertab##1##2{\rightltab{##1}{##2}}%
8619   \measuretbody{#1}%
8620   \l@drestorefills
8621   \variab
8622   \setmrowleft #1\\&\\%
8623   \enablel@dtabfeet}
8624
8625 %

```

`\ltabtext` Tabular with entries left justified.

```

8626 \newcommand{\ltabtext}[1]{%
8627   \l@dnullfills
8628   \measuretbody{#1}%
8629   \l@drestorefills
8630   \variab
8631   \settrrowleft #1\\&\\%
8632   \enablel@dtabfeet}
8633
8634 %

```

`\ctab` Array with centered entries.

```

\edbeforetab
\edaftertab
8635 \newcommand{\ctab}[1]{%
8636   \l@dnullfills
8637   \def\edbeforetab##1##2{\leftctab{##1}{##2}}%
8638   \def\edaftertab##1##2{\rightctab{##1}{##2}}%
8639   \measuretbody{#1}%
8640   \l@drestorefills
8641   \variab
8642   \setmrowcenter #1\\&\\%

```

```

8643 \enablel@dtabfeet}
8644
8645 %

```

\ctabtext Tabular with entries centered.

```

8646 \newcommand{\ctabtext}[1]{%
8647 \l@dnnullfills
8648 \measuretbody{#1}%
8649 \l@drestorefills
8650 \variab
8651 \settrrowcenter #1\\&\\%
8652 \enablel@dtabfeet}
8653
8654 %

```

```

\spreadtext55 \newcommand{\spreadtext}[1]{%\l@dcolcount=\l@dampcount%
8656 \hb@xt@ \the\l@dcolwidth{\hbox{#1}\hss}}
8657 %

```

```

\spreadmath58 \newcommand{\spreadmath}[1]{%
8659 \hb@xt@ \the\l@dcolwidth{\hbox{$\displaystyle{#1}$}\hss}}
8660
8661 %

```

\HILFSskip More helpers.

```

\Hilfsskip
8662 \newskip\HILFSskip
8663 \newskip\Hilfsskip
8664
8665 %

```

```

\EDTABINDENT166 \newcommand{\EDTABINDENT}{%
8667 \ifnum\l@dcolcount=30\let\NEXT\relax\l@dcolcount=0%
8668 \else\step\l@dcolcount%
8669 \advance\Hilfsskip by\l@dcolwidth%
8670 \ifdim\l@dcolwidth=0pt\advance\hilfscount\@ne
8671 \else\advance\Hilfsskip by \the\hilfscount\edtabcolsep%
8672 \hilfscount=1\fi%
8673 \let\NEXT=\EDTABINDENT%
8674 \fi\NEXT}%
8675 %

```

\edtabindent (was \tabindent)

```

8676 \newcommand{\edtabindent}{%
8677     \l@dcolcount=0\relax
8678     \Hilfsskip=0pt%
8679     \hilfscount=1\relax
8680     \EDTABINDENT%
8681     \hilfsskip=\hsize%
8682     \advance\hilfsskip -\Hilfsskip%
8683     \Hilfsskip=0.5\hilfsskip%
8684 }%
8685
8686 %

```

\EDTAB (was \TAB)

```

8687 \def\EDTAB #1|#2|{%
8688     \setbox\tabhilfbox=\hbox{$\displaystyle{#1}$}%
8689     \setbox\tabHilfbox=\hbox{$\displaystyle{#2}$}%
8690     \advance\tabelskip -\wd\tabhilfbox%
8691     \advance\tabelskip -\wd\tabHilfbox%
8692     \unhbox\tabhilfbox\hskip\tabelskip%
8693     \unhbox\tabHilfbox}%
8694
8695 %

```

\EDTABtext (was \TABtext)

```

8696 \def\EDTABtext #1|#2|{%
8697     \setbox\tabhilfbox=\hbox{#1}%
8698     \setbox\tabHilfbox=\hbox{#2}%
8699     \advance\tabelskip -\wd\tabhilfbox%
8700     \advance\tabelskip -\wd\tabHilfbox%
8701     \unhbox\tabhilfbox\hskip\tabelskip%
8702     \unhbox\tabHilfbox}%
8703 %

```

\tabhilfbox Further helpers.

\tabHilfbox

```

8704 \newbox\tabhilfbox
8705 \newbox\tabHilfbox
8706
8707 %

```

XXX.2.4 Environments

`edarrayl edarrayc edarrayr` The ‘environment’ forms for `\ltab`, `\ctab` and `\rtab`.

```

8708 \newenvironment{edarrayl}{\l@dcollect@body\ltab}{}
8709 \newenvironment{edarrayc}{\l@dcollect@body\ctab}{}
8710 \newenvironment{edarrayr}{\l@dcollect@body\rtab}{}
8711
8712 %

```

edtabularl edtabularc edtabularr The ‘environment’ forms for \ltabtext, \ctabtext and \rtabtext.

```

8713 \newenvironment{edtabularl}{\l@collect@body\ltabtext}{}
8714 \newenvironment{edtabularc}{\l@collect@body\ctabtext}{}
8715 \newenvironment{edtabularr}{\l@collect@body\rtabtext}{}
8716
8717 %

```

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 \initnumbering@quote defines quotation environment.
\endquotation
\quote      8718 \newcommand{\initnumbering@quote}{
\endquote   8719 \ifnoquotation@else
            8720 \renewcommand{\quotation}{\par\leavevmode%
            8721 \parindent=1.5em%
            8722 \skipnumbering%
            8723 \ifautopar%
            8724 \vskip-\parskip%
            8725 \else%
            8726 \vskip\topsep%
            8727 \fi%
            8728 \global\leftskip=\leftmargin%
            8729 \global\rightskip=\leftmargin%
            8730 }
            8731 \renewcommand{\endquotation}{\par%
            8732 \global\leftskip=0pt%
            8733 \global\rightskip=0pt%
            8734 \leavevmode%
            8735 \skipnumbering%
            8736 \ifautopar%
            8737 \vskip-\parskip%
            8738 \else%
            8739 \vskip\topsep%
            8740 \fi%
            8741 }
            8742 \renewcommand{\quote}{\par\leavevmode%
            8743 \parindent=0pt%
            8744 \skipnumbering%
            8745 \ifautopar%
            8746 \vskip-\parskip%
            8747 \else%
            8748 \vskip\topsep%
            8749 \fi%

```



```

8750             \global\leftskip=\leftmargin%
8751             \global\rightskip=\leftmargin%
8752         }
8753     \renewcommand{\endquote}{\par%
8754         \global\leftskip=0pt%
8755         \global\rightskip=0pt%
8756         \leavevmode%
8757         \skipnumbering%
8758         \ifautopar%
8759             \vskip-\parskip%
8760         \else%
8761             \vskip\topsep%
8762         \fi%
8763     }
8764 \fi
8765 }
8766 %

```

XXXII Section's title commands

XXXII.1 Commands to disable some feature

\ledsectnotoc The \ledsectnotoc only disables the \addcontentsline macro.

```

8767 \newcommand{\ledsectnotoc}{\let\addcontentsline\@gobblethree}
8768 %

```

\ledsectnomark The \ledsectnomark only disables the \chaptermark, \sectionmark and \subsectionmark macros.

```

8769 \newcommand{\ledsectnomark}{%
8770     \let\chaptermark\@gobble%
8771     \let\sectionmark\@gobble%
8772     \let\subsectionmark\@gobble%
8773 }
8774 %

```

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).

```
8775 \catcode`\#=12
8776 \notbool{@noeled@sec}{%
8777 %
```

`\beforeeledchapter` For technical reasons, not yet solved, page-breaking before chapters can't be made automatically by `eledmac`. Users have to use `\beforeeledchapter`.

```
8778 \ifl@dmemoir
8779   \newcommand\beforeeledchapter{%
8780     \clearforchapter%
8781   }
8782 \else
8783   \newcommand\beforeeledchapter{%
8784     \if@openright%
8785       \cleardoublepage%
8786     \else%
8787       \clearpage%
8788     \fi%
8789   }
8790 \fi
8791 %
```

XXXII.4 Auxiliary commands

`\print@leftmargin@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.

```

8792 \def\print@rightmargin@eledsection{%
8793   \if@eled@sectioning%
8794     \begingroup%
8795     \if@RTL%
8796       \let\llap\rlap%
8797       \let\leftlinenum\rightlinenum%
8798       \let\leftlinenumR\rightlinenumR%
8799       \let\l@drd@ta\l@dld@ta%
8800       \let\l@drsn@te\l@dlsn@te%
8801     \fi%
8802     \hfill\l@drd@ta \csuse{LR}{\l@drsn@te}%
8803     \endgroup%
8804   \fi%
8805 }%
8806
8807 \def\print@leftmargin@eledsection{%
8808   \if@eled@sectioning%
8809     \leavevmode%
8810     \begingroup%
8811     \if@RTL%
8812       \let\rlap\llap%
8813       \let\rightlinenum\leftlinenum%
8814       \let\rightlinenumR\leftlinenumR%
8815       \let\l@dld@ta\l@drd@ta%
8816       \let\l@dlsn@te\l@drsn@te%
8817     \fi%
8818     \l@dld@ta\csuse{LR}{\l@dlsn@te}%
8819     \endgroup%
8820   \fi%
8821 }%
8822
8823 %

```

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.

```

8824 \AtBeginDocument{%
8825
8826
8827 \pretocmd{\M@sect}

```

```

8828 {\let\old@edtext=\edtext%
8829 \let\edtext=\dummy@edtext@showlemma%
8830 }
8831 {}
8832 {}
8833
8834 \apptocmd{\M@sect}
8835 {\let\edtext=\old@edtext}
8836 {}
8837 {}
8838
8839 \patchcmd{\M@sect}
8840 { #9}
8841 { #9%
8842 \print@rightmargin@eledsection%
8843 }
8844 {}
8845 {}
8846
8847 \patchcmd{\M@sect}
8848 {\hskip #3\relax}
8849 {\hskip #3\relax%
8850 \print@leftmargin@eledsection%
8851 }
8852 {}
8853 {}
8854
8855 \patchcmd{\@mem@old@ssect}
8856 {#5}
8857 {#5%
8858 \print@leftmargin@eledsection%
8859 }
8860 {}
8861 {}
8862
8863 \patchcmd{\@mem@old@ssect}
8864 {\hskip #1}
8865 {\hskip #1%
8866 \print@rightmargin@eledsection%
8867 }
8868 {}
8869 {}
8870
8871
8872
8873 \patchcmd{\scr@startchapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
8874 \if@eled@sectioning\else%
8875 \ifl@dprintingpages\else%
8876 \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%
8877 \fi%
8878 \fi%
8879 }
8880 {}
8881 {}
8882
8883 \patchcmd{\@makechapterhead}
8884 {#1}
8885 {\print@leftmargin@eledsection%
8886   #1%
8887 \print@rightmargin@eledsection%
8888 }
8889 {}
8890 {}
8891
8892 \patchcmd{\@makechapterhead}% For BIDI
8893 {\if@RTL\raggedleft\else\raggedright\fi}%
8894 {\if@eled@sectioning\else%
8895   \if@RTL\raggedleft\else\raggedright\fi%
8896 \fi%
8897 }%
8898 {}%
8899 {}%
8900
8901 \patchcmd{\@makeschapterhead}
8902 {#1}
8903 {\print@leftmargin@eledsection%
8904   #1%
8905 \print@rightmargin@eledsection%
8906 }
8907 {}
8908 {}
8909
8910 \pretocmd{\@sect}
8911 {\let\old@edtext=\edtext
8912 \let\edtext=\dummy@edtext@showlemma%
8913 }
8914 {}
8915 {}
8916
8917 \apptocmd{\@sect}
8918 {\let\edtext=\old@edtext}
8919 {}
8920 {}
8921
8922 \pretocmd{\@ssect}
8923 {\let\old@edtext=\edtext%
8924 \let\edtext=\dummy@edtext@showlemma%

```

```

8925 }
8926 {}
8927 {}
8928
8929 \apptocmd{\@ssect}
8930 {\let\edtext=\old@edtext}
8931 {}
8932 {}
8933
8934 %

```

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.

```

8935 \@ifpackageloaded{nameref}{
8936
8937   \patchcmd{\NR@sect}
8938     {#8}
8939     {#8%
8940       \print@rightmargin@eledsection%
8941     }
8942     {}
8943     {}
8944
8945   \patchcmd{\NR@sect}
8946     {\hskip #3\relax}
8947     {\hskip #3\relax%
8948       \print@leftmargin@eledsection%
8949     }
8950     {}
8951     {}
8952
8953   \patchcmd{\NR@ssect}
8954     {#5}
8955     {#5%
8956       \print@rightmargin@eledsection%
8957     }
8958     {}
8959     {}
8960
8961   \patchcmd{\NR@ssect}
8962     {\hskip #1}
8963     {\hskip #1%
8964       \print@leftmargin@eledsection%
8965     }
8966     {}
8967     {}
8968   }%
8969   {
8970     \patchcmd{\@sect}
8971     {#8}

```

```

8972     {#8%
8973     \print@rightmargin@eledsection%
8974     }
8975     {}
8976     {}
8977
8978     \patchcmd{\@sect}
8979     {\hskip #3\relax}
8980     {\hskip #3\relax%
8981     \print@leftmargin@eledsection%
8982     }
8983     {}
8984     {}
8985
8986     \patchcmd{\@ssect}
8987     {#5}
8988     {#5%
8989     \print@rightmargin@eledsection%
8990     }
8991     {}
8992     {}
8993
8994     \patchcmd{\@ssect}
8995     {\hskip #1}
8996     {\hskip #1%
8997     \print@leftmargin@eledsection%
8998     }
8999     {}
9000     {}
9001     }%
9002 }%
9003 %

```

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 `#`.

```

9004 {}}%
9005 \protect\catcode`\#=6 %Space NEEDS by \catcode
9006 %

```

\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.

```

9007 \AtBeginDocument{%
9008 \patchcmd{\chapter}{\clearforchapter}{%
9009 \if@eled@sectioning\else%
9010 \ifl@dprintingpages\else%
9011 \clearforchapter%
9012 \fi%
9013 \fi%

```

```

9014 }%
9015 {}%
9016 {}%
9017
9018 \patchcmd{\chapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
9019 \if@eled@sectioning\else%
9020 \ifl@dprintingpages%
9021 \endgraf%
9022 \else%
9023 \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
9024 \fi%
9025 \fi%
9026 }%
9027 {}%
9028 {}%
9029 }%
9030 %

```

\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.

```

9031 \newif\if@eled@sectioning%
9032 %

```

We reopen a new `\notbool{@noeled@sec}` statement, as we will define the `\elesection` commands.

```

9033 \notbool{@noeled@sec}{%
9034 %

```

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.

```

9035 \newwrite\eled@sectioning@out
9036 %

```

\eledchapter **\eledsection** And now, the user sectioning commands, which write to the file, and also add content as a “normal” line.

```

\eledsubsection
\eledsubsubsection
\eledchapter*
\eledsection*
\eledsubsection*
\eledsubsubsection*

```

```

9037 \newcommand{\eledchapter}[2][]{%
9038 \disable@familiarnotes%
9039 #2%
9040 \restore@familiarnotes%
9041 \ifledRcol%

```



```

9042 \immediate\write\eled@sectioningR@out{%
9043 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
9044 }%
9045 \else%
9046 \immediate\write\eled@sectioning@out{%
9047 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{}}
9048 }%
9049 \fi%
9050 }
9051
9052 \newcommand{\eledsection}[2][{}]{%
9053 \disable@familiarnotes%
9054 #2%
9055 \restore@familiarnotes%
9056 \ifledRcol%
9057 \immediate\write\eled@sectioningR@out{%
9058 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
9059 }%
9060 \else%
9061 \immediate\write\eled@sectioning@out{%
9062 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{}}
9063 }%
9064 \fi%
9065 }
9066
9067 \newcommand{\eledsubsection}[2][{}]{%
9068 \disable@familiarnotes%
9069 #2%
9070 \restore@familiarnotes%
9071 \ifledRcol%
9072 \immediate\write\eled@sectioningR@out{%
9073 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
9074 }%
9075 \else%
9076 \immediate\write\eled@sectioning@out{%
9077 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{}}
9078 }%
9079 \fi%
9080 }
9081 \newcommand{\eledsubsubsection}[2][{}]{%
9082 \disable@familiarnotes%
9083 #2%
9084 \restore@familiarnotes%
9085 \ifledRcol%
9086 \immediate\write\eled@sectioningR@out{%
9087 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}
9088 }{R}
9089 }%
9090 \else%

```

```

9090 \immediate\write\eled@sectioning@out{%
9091 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL
9092 }{}{}
9093 }%
9094 \fi%
9095 }
9096
9097 \WithSuffix\newcommand\eledchapter*[2][]{%
9098 \disable@familiarnotes%
9099 #2%
9100 \restore@familiarnotes%
9101 \ifledRcol%
9102 \immediate\write\eled@sectioningR@out{%
9103 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
9104 }%
9105 \else%
9106 \immediate\write\eled@sectioning@out{%
9107 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{L}
9108 }%
9109 \fi%
9110 }
9111
9112 \WithSuffix\newcommand\eledsection*[2][]{%
9113 \disable@familiarnotes%
9114 #2%
9115 \restore@familiarnotes%
9116 \ifledRcol%
9117 \immediate\write\eled@sectioningR@out{%
9118 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
9119 }%
9120 \else%
9121 \immediate\write\eled@sectioning@out{%
9122 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{L}
9123 }%
9124 \fi%
9125 }
9126
9127 \WithSuffix\newcommand\eledsubsection*[2][]{%
9128 \disable@familiarnotes%
9129 #2%
9130 \restore@familiarnotes%
9131 \ifledRcol%
9132 \immediate\write\eled@sectioningR@out{%
9133 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
9134 }%
9135 \else%
9136 \immediate\write\eled@sectioning@out{%
9137 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL

```

```

}{{*}}{}
9138   }%
9139   \fi%
9140 }
9141
9142 \WithSuffix\newcommand\eledsubsubsection*[2] [] {%
9143   \disable@familiarnotes%
9144   #2%
9145   \restore@familiarnotes%
9146   \ifledRcol%
9147     \immediate\write\eled@sectioningR@out{%
9148       \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsR
9149 }{{*}}{R}
9149     }%
9150   \else%
9151     \immediate\write\eled@sectioning@out{%
9152       \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL
9153 }{{*}}{}
9153     }%
9154   \fi%
9155 }
9156 %

```

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.

```

9157 \def\eled@chapter#1#2#3#4#5{%
9158   \ifstrempy{#4}%
9159   {%
9160     \ifstrempy{#1}%
9161     {%
9162       \csgdef{eled@sectioning@#3#5}{\let\edtext=\dummy@edtext@showlemma\
chapter{#2}}}%
9163       \csgdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\chaptermark
{#2}}}%
9164     }%Need for \pairs, because of using parbox.
9165     {%
9166       \csgdef{eled@sectioning@#3#5}{\let\edtext=\dummy@edtext@showlemma\
chapter[#1]{#2}}}%

```

```

9167 \csgdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\chaptermark
9168 {#2}}%Need for \pairs, because of using parbox.
9169 }%
9170 {%
9171 \ifstrepty{#1}%
9172 {\csgdef{eled@sectioning@#3#5}{\let\edtext=\dummy@edtext@showlemma\
chapter*{#2}}}%
9173 {\csgdef{eled@sectioning@#3#5}{\let\edtext=\dummy@edtext@showlemma\
chapter*{#1}{#2}}}%Bug in LaTeX!
9174 }%
9175 \listcsgadd{eled@sections#5@@}{#3}%
9176 }
9177 \def\eled@section#1#2#3#4#5{%
9178 \ifstrepty{#4}%
9179 {\ifstrepty{#1}%
9180 {%
9181 \csgdef{eled@sectioning@#3#5}{\section{#2}}%
9182 \csgdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\sectionmark
9183 {#2}}%Need for \pairs, because of using parbox.
9184 }%
9185 {\csgdef{eled@sectioning@#3#5}{\section[#1]{#2}}%
9186 \csgdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\sectionmark
9187 {#1}}%Need for \pairs, because of using parbox.
9188 }%
9189 {\ifstrepty{#1}%
9190 {\csgdef{eled@sectioning@#3#5}{\section*{#2}}}%
9191 {\csgdef{eled@sectioning@#3#5}{\section*{#1}{#2}}}%Bug in LaTeX!
9192 }
9193 \listcsgadd{eled@sections#5@@}{#3}%
9194 }
9195 \def\eled@subsection#1#2#3#4#5{%
9196 \ifstrepty{#4}%
9197 {\ifstrepty{#1}%
9198 {%
9199 \csgdef{eled@sectioning@#3#5}{\subsection{#2}}%
9200 \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)
9201 }%
9202 {%
9203 \csgdef{eled@sectioning@#3#5}{\subsection[#1]{#2}}%
9204 \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)
9205 }%
9206 }%
9207 {\ifstrepty{#1}%

```

```

9208     {\csgdef{eled@sectioning@#3#5}{\subsection*{#2}}}%
9209     {\csgdef{eled@sectioning@#3#5}{\subsection*{#1}{#2}}}%Bug in LaTeX!
9210   }
9211   \listcsgadd{eled@sections#5@@}{#3}%
9212   }
9213 \def\eled@subsubsection#1#2#3#4#5{%
9214   \ifstrepty{#4}%
9215     {\ifstrepty{#1}%
9216       {\csgdef{eled@sectioning@#3#5}{\subsubsection{#2}}}%
9217       {\csgdef{eled@sectioning@#3#5}{\subsubsection{#1}{#2}}}%
9218     }%
9219     {\ifstrepty{#1}%
9220       {\csgdef{eled@sectioning@#3#5}{\subsubsection*{#2}}}%
9221       {\csgdef{eled@sectioning@#3#5}{\subsubsection*{#1}{#2}}}%Bug in
LaTeX!
9222   }
9223   \listcsgadd{eled@sections#5@@}{#3}%
9224   }
9225
9226 %

```

End of the conditional test about noeledsec option.

```

9227 }{}
9228 %

```

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.

```

9229 \def\normal@page@break{}
9230 %

```

\prev@pb The \l@prev@pb macro is a etoolbox list, which contains the lines in which page breaks occur (before or after). The \l@prev@nopb macro is a etoolbox list, which contains the lines with NO page break before or after.

```

9231 \def\l@prev@pb{}
9232 \def\l@prev@nopb{}
9233 %

```

`\ledpb` The `\ledpb` macro writes the call to `\led@pb` in line-list file. The `\ledpbnum` macro writes the call to `\led@pbnum` in line-list file. The `\lednopb` macro writes the call to `\led@nopb` in line-list file. The `\lednopbnum` macro writes the call to `\led@nopbnum` in line-list file.

```

9234 \newcommand{\ledpb}{\write\linenum@out{\string\led@pb}}
9235 \newcommand{\ledpbnum}[1]{\write\linenum@out{\string\led@pbnum{#1}}}
9236 \newcommand{\lednopb}{\write\linenum@out{\string\led@nopb}}
9237 \newcommand{\lednopbnum}[1]{\write\linenum@out{\string\led@nopbnum{#1}}}
9238 %

```

`\led@pb` The `\led@pb` adds the absolute line number in the `\prev@pb` list. The `\led@pbnum` adds the argument in the `\prev@pb` list. The `\led@nopb` adds the absolute line number in the `\prev@nopb` list. The `\led@nopbnum` adds the argument in the `\prev@nopb` list.

```

9239 \newcommand{\led@pb}{\listxadd{\l@prev@pb}{\the\absline@num}}
9240 \newcommand{\led@pbnum}[1]{\listxadd{\l@prev@pb}{#1}}
9241 \newcommand{\led@nopb}{\listxadd{\l@prev@nopb}{\the\absline@num}}
9242 \newcommand{\led@nopbnum}[1]{\listxadd{\l@prev@nopb}{#1}}
9243 %

```

`\ledpbsetting` The `\ledpbsetting` macro only changes the value of `\led@pb@macro`, for which the default value is before.

`\led@pb@setting`

```

9244 \def\led@pb@setting{before}
9245 \newcommand{\ledpbsetting}[1]{\gdef\led@pb@setting{#1}}
9246 %

```

`\led@check@pb` The `\led@check@pb` and `\led@check@nopb` are called before or after each line. They check if a page break must occur, depending on the current line and on the content of `\l@pb`.

`\led@check@nopb`

```

9247 \newcommand{\led@check@pb}{\xifinlist{\the\absline@num}{\l@prev@pb}{\pagebreak[4]}}
9248 \newcommand{\led@check@nopb}{%
9249   \IfStrEq{\led@pb@setting}{before}{%
9250     \xifinlist{\the\absline@num}{\l@prev@nopb}%
9251     {\numdef{\abs@prevline}{\the\absline@num-1}%
9252     \xifinlist{\abs@prevline}{\normal@page@break}%
9253     {\nopagebreak[4]\enlargethispage{\baselineskip}}%
9254     {}}%
9255   {}}%
9256   {}%
9257   {}%
9258   \IfStrEq{\led@pb@setting}{after}{%
9259     \xifinlist{\the\absline@num}{\l@prev@nopb}%
9260     \xifinlist{\the\absline@num}{\normal@page@break}%
9261     {\nopagebreak[4]\enlargethispage{\baselineskip}}%
9262     {}}%
9263 }%

```

```

9264     {}}%
9265     {}%
9266     {}%
9267 }
9268 %

```

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.

```

9269 \newcommand{\check@pb@in@verse}{%
9270   \ifinstanza\iflednopbinverse\ifinserthangingsymbol% Using stanzas and
enabling page breaks in verse control, while on a hanging verse.
9271   \ifnum\page@num=\last@page@num\else%If we have change page
9272   \IfStrEq{\led@pb@setting}{before}{%
9273     \numdef{\abs@line@verse}{\the\absline@num-1}%
9274     \ledpbnum{\abs@line@verse}%
9275   }{}%
9276   \IfStrEq{\led@pb@setting}{after}{%
9277     \numdef{\abs@line@verse}{\the\absline@num-1}%
9278     \lednopbnum{\abs@line@verse}%
9279   }{}%
9280   \fi%
9281 \fi\fi\fi%
9282 }
9283 %

```

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

```

9284 \def\Hy@raisedlink@left#1{%
9285   \ifvmode
9286     #1%
9287   \else
9288     \Hy@SaveSpaceFactor
9289     \llap{\smash{%
9290       \begingroup
9291         \let\HyperRaiseLinkLength\@tempdima
9292         \setlength\HyperRaiseLinkLength\HyperRaiseLinkDefault
9293         \HyperRaiseLinkHook
9294       \expandafter\endgroup
9295       \expandafter\raise\the\HyperRaiseLinkLength\hbox{%
9296         \Hy@RestoreSpaceFactor
9297         #1%
9298         \Hy@SaveSpaceFactor
9299       }%
9300     }}%
9301     \Hy@RestoreSpaceFactor
9302     \penalty\@M\hskip\z@ \relax
9303   \fi
9304 }
9305 %

```

XXXVI Compatibility with eledmac

Here, we define some commands for the `eledmac-compat` option.

```

9306 \ifeledmaccompat@%
9307
9308   \newcommand{\footnormalX}[1]{\arrangementX[#1]{normal}}%
9309   \newcommand{\footparagraphX}[1]{\arrangementX[#1]{paragraph}}%
9310   \newcommand{\foottwocolX}[1]{\arrangementX[#1]{twocol}}%
9311   \newcommand{\footthreecolX}[1]{\XarrangementX[#1]{threecol}}%
9312
9313   \unless\ifnocritical@
9314     \newcommand{\footnormal}[1]{\Xarrangement[#1]{normal}}%
9315     \newcommand{\footparagraph}[1]{\Xarrangement[#1]{paragraph}}%
9316     \newcommand{\foottwocol}[1]{\Xarrangement[#1]{twocol}}%
9317     \newcommand{\footthreecol}[1]{\Xarrangement[#1]{threecol}}%
9318     \let\hsizetwocol\Xhsizetwocol
9319     \let\hsizethreecol\Xhsizethreecol
9320     \let\bhookXnote\Xbhooknote

```

³⁶The 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>.


```

9321 \let\boxsymlinenum\Xboxsymlinenum
9322 \let\symlinenum\Xsymlinenum
9323 \let\beforenumberinfootnote\Xbeforenumber
9324 \let\afternumberinfootnote\Xafternumber
9325 \let\beforeXsymlinenum\XbeforeXsymlinenum
9326 \let\afterXsymlinenum\XafterXsymlinenum
9327 \let\inplaceofnumber\Xinplaceofnumber
9328 \let\Xlemmaseparator\lemmaseparator
9329 \let\afterlemmaseparator\Xafterlemmaseparator
9330 \let\beforelemmaseparator\Xbeforelemmaseparator
9331 \let\inplaceoflemmaseparator\Xinplaceoflemmaseparator
9332 \let\txbeforeXnotes\Xtxbeforenotes
9333 \let\afterXrule\Xafterrule
9334 \let\numberonlyfirstinline\Xnumberonlyfirstinline
9335 \let\numberonlyfirstintwolines\Xnumberonlyfirstintwolines
9336 \let\nonumberinfootnote\Xnonumberinfootnote
9337 \let\pstartinfootnote\Xpstart
9338 \let\pstartinfootnoteeverytime\Xpstarteverytime
9339 \let\onlyXpstart\Xonlypstart
9340 \let\Xnonumberinfootnote\Xnonumber
9341 \let\Xnonbreakableafternumber\Xnonbreakableafternumber
9342 \let\maxhXnotes\Xmaxhnotes
9343 \let\beforeXnotes\Xbeforenotes
9344 \let\boxlinenum\Xboxlinenum
9345 \let\boxlinenumalign\Xboxlinenumalign
9346 \let\boxstartlinenum\Xboxstartlinenum
9347 \let\boxendlinenum\Xboxendlinenum
9348 \let\twolines\Xtwolines
9349 \let\morethantwolines\Xmorethantwolines
9350 \let\twolinesbutnotmore\Xtwolinesbutnotmore
9351 \let\twolinesonlyinsamepage\Xtwolinesonlyinsamepage
9352 \fi
9353
9354 \unless\ifnofamiliar@
9355 \let\notesXwidthliketwocolumns\noteswidthliketwocolumnsX
9356 \fi
9357 \newcommandx{\parafootsep}[2][1,usedefault]{%
9358 \Xparafootsep[#1]{#2}%
9359 \parafootsepX[#1]{#2}
9360 }%
9361
9362 \newcommandx{\afternote}[2][1,usedefault]{%
9363 \Xafternote[#1]{#2}%
9364 \afternoteX[#1]{#2}%
9365 }%
9366
9367 \unless\ifnoend@
9368 \let\XendXtwolines\Xendtwolines
9369 \let\XendXmorethantwolines\Xendmorethantwolines
9370 \let\XhookXendnote\Xendhooknote

```

```

9371 \let\boxXendlinenum\Xendboxlinenum%
9372 \let\boxXendlinenumalign\Xendboxlinenumalign%
9373 \let\boxXendstartlinenum\Xendboxstartlinenum%
9374 \let\boxXendendlinenum\Xendboxendlinenum%
9375 \let\XendXlemmaseparator\Xendlemmaseparator
9376 \let\XendXbeforelemmaseparator\Xendbeforelemmaseparator
9377 \let\XendXafterlemmaseparator\Xendafterlemmaseparator
9378 \let\XendXinplaceoflemmaseparator\Xendinplaceoflemmaseparator
9379 \fi
9380
9381 \AtBeginDocument{%
9382   \ifdef\lineref{}\let\lineref\edlineref}%
9383 }%
9384
9385
9386 \fi%
9387 %

```

</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 changed³⁷ to `\critext`; use the `\edtext` macro instead. However, if you do use `\critext` (the new name for `\text`), the following is a reminder.

`\critext` Within numbered paragraphs, footnotes and endnotes are generated by forms of the `\critext` macro:

```
\critext{⟨lemma⟩}⟨commands⟩/
```

The `⟨lemma⟩` argument is the lemma in the main text: `\critext` both prints this as part of the text, and makes it available to the `⟨commands⟩` you specify to generate notes. The `/` at the end terminates the command; it is part of the macro's definition so that spaces after the macro will be treated as significant.

For example:

<pre>I saw my friend \critext{Smith} \Afootnote{Jones C, D.}/ on Tuesday.</pre>	<pre>1 I saw my friend 2 Smith on Tuesday. 2 Smith] Jones C, D.</pre>
---	---

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

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

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

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

The second argument of the `\critext` macro, `⟨commands⟩`, is the same as the second argument to the `\edtext` macro.

It is possible to define aliases for `\critext`, which can be easier to type. You can make a single character substitute for `\critext` by saying this:

```
\catcode`\<=\active
```

³⁷A name like `\text` is likely to be defined by other \TeX packages (it certainly is by the AMS packages) and it seems sensible to try and avoid clashes with other definitions.

```
\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. 126 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. 36), and remove any `\Xfootnote` command.

If you have customized the `\XXXXXfmt` command, please check whether you can achieve the same by the commands documented for display options (7 p. 37) 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. 52). This bug had two consequences:

1. `stanzaindentsrepetition` did not work when its value was greater than 2.
2. `stanzaindentsrepetition` worked wrong when its value was equal to 2.

So, if you used `stanzaindentsrepetition` with a value equal to 2, you had to change your `\setstanzaindents`. Explanation:

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

This code, in versions prior to 1.5.1, made the first line have an indentation of 0, the second line of 1, the third verse of 0, the fourth verse of 1 and so forth.

But this code should have instead achieved quite the contrary: the first line would have an indentation of 1, the second line of 0, the third line of 1, the fourth line of 0 and so forth.

So version 1.5.1 corrected this bug. If you want to keep the former presentation, you must change:

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

to:

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

Appendix A.4 Migration to eledmac 1.12.0

The migration to eledmac 1.12.0 is easy:

- You must first delete all the auxiliary files, then compile your document three times as usual.
- If you have modified `\l@reg`, which is not advisable, you must rename it to `\@nl@reg`.

There is an additional problem. If you have put text into brackets just after `\pstart` or `\pend`, this text will be considered to be an optional argument of `\pstart` or `\pend` (see 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. 68.

Appendix A.5 Migration to eledmac 17.1

This version changes the default setting of `\Xpstart`. Henceforth, `pstart` numbers will be printed in footnotes within the section of text where you have called `\numberpstarttrue`.

We do not see any reason to print them in the other sections. However, if you want to print the `pstart` numbers in all of the footnotes, whatever the section, without having to use `\numberpstarttrue`, you can use `\Xpstarteverytime`.

Appendix A.6 Migration to eledmac 1.21.0

Appendix A.6.1 `\Xledsetnormalparstuff` and `\ledsetnormalparstuffX`

The `\ledsetnormalparstuff` has been split into two different commands:

- `\Xledsetnormalparstuff` for critical notes;
- `\ledsetnormalparstuffX` for familiar notes.

Both commands can take an optional argument which is the series letter. If you have redefined `\ledsetnormalparstuff` or any of the commands which call them, you must change them accordingly.

Appendix A.6.2 Endnotes

In any case, delete the `.end` file before the next run.

The previous version of Eledmac had a bug: there were two spaces between the starting page number and the starting line number, but only one space between the ending page number and the ending line number.

As a matter of fact, a spurious space was added after the first `\printnnum`. This spurious space has been deleted. However, if you want to keep the previous spurious space, you may load the package with the `oldprintnnumspace` option.

If you have redefined `\endprint`, you must:

- Contact us and ask for the feature that required your hack, in order to avoid such a hack in the future.
- Use the new fifth argument.
- Add `\xdef\@currentseries{#4}` at the beginning of your own command.

Appendix A.7 Migration to eledmac 1.22.0

The `\ledinnote` command now takes a first optional argument, which is the label for the hyperreference. If you have redefined it, change your redefinition, and check whether you can avoid this redefinition by only redefining `\ledinnotemark`.

Appendix A.8 Migration to eledmac 1.23.0

You must delete the numbered auxiliary files before compiling with the new version of eledmac.

Appendix A.9 Migration from eledmac to reledmac

There are many changes in reledmac which require the user to make modifications.

Appendix A.9.1 Risk of ‘no room for a new’

The risk to obtain a ‘no room for a new something’ error is greater in reledmac than it is in eledmac. See 19.1.3 p. 71 in order to know how to limit it.

Appendix A.9.2 Multiple indices with memoir

Eledmac and ledmac used the specific indexing tools of the memoir class designed to produce multiple indices. However, eledmac could also use imakeidx or indextools tools independently of the memoir class. This system forced to maintain redundant code. Since reledmac, we use only the imakeidx or indextools tools.

Consequently: Users of memoir are invited to use indextool or imakeidx to produce multiple indices.

Appendix A.9.3 Deprecated commands and options

The table of deprecated commands and their alternatives follows. Note that the way some commands must be used may have changed. Please read the handbook.

<i>Deprecated command</i>	<i>Replaced with</i>
<code>\addfootins</code>	<code>\newseries</code>
<code>\addfootinsX</code>	<code>\newseries</code>
<code>\critext</code>	<code>\edtext</code>
<code>\falseverse</code>	<code>\newverse</code>
<code>\interparanoteglue</code>	<code>\Xafternote</code> and <code>\afternoteX</code>
<code>\ledchapter</code>	<code>\eledchapter</code>
<code>\ledsection</code>	<code>\eledsection</code>
<code>\ledsetnormalparstuff</code>	<code>\Xledsetnormalparstuff</code> and <code>\ledsetnormalparstuffX</code>
<code>\ledsubsection</code>	<code>\eledsubsection</code>
<code>\ledsubsubsection</code>	<code>\eledsubsubsection</code>
<code>\noeledsec</code>	Package option <code>noeledsec</code>
<code>\noendnotes</code>	Package option <code>noendnotes</code>
<code>\pageparbreak</code>	<code>\ledpb</code>

The `ledsecnolinenumber` option has been removed, because it was related to deprecated commands.

The `oldprintnpnumspace` option has been removed too, because it was related to a historical bug. The `\usingedtext` and `\usingcritext` commands are also deprecated.

Appendix A.9.4 `\renewcommand` replaced by command

Many uses of `\renewcommand` have been replaced with uses of specific commands. Please read handbook about specific commands.

<i>Deprecated <code>\renewcommand</code></i>	<i>Replaced with</i>
<code>\@led@extranofeet</code>	<code>\newseries</code>
<code>\apprefprefixmore</code>	<code>\setapprefprefixmore</code>
<code>\apprefprefixsingle</code>	<code>\setapprefprefixsingle</code>
<code>\endstanzaextra</code>	Optional argument of <code>\&</code>
<code>\hangingsymbol</code>	<code>\sethangingsymbol</code>
<code>\ledfootinsdim</code>	<code>\Xmaxhnotes</code> and <code>\maxhnotesX</code>
<code>\parafootftmsep</code>	<code>\Xparafootsep</code> and <code>\parafootsepX</code>
<code>\notenumfont</code>	<code>\Xnotenumfont</code> , <code>\Xendnotenumfont</code> and <code>\notenumfontX</code>
<code>\notefontsetup</code>	<code>\Xnotefontsize</code> , <code>\Xendnotefontsize</code> and <code>\notefontsizeX</code>
<code>\sidenotesep</code>	<code>\setsidenotsep</code>
<code>\startstanzahook</code>	Optional argument of <code>\stanza</code>
<code>\symplinenum</code>	<code>\Xsymplinenum</code>

Appendix A.9.5 Commands the names of which have been changed

In order to help the migration from `eledmac` to `reledmac`, you may load `reledmac` with `eledmac-compat` option. However, it is advised not to, and to change the command names themselves instead. In many cases, you use only a few of them, except the `\footparagraph` command.

<i>Old command</i>	<i>New command</i>
<code>\footparagraph</code>	<code>\Xarrangement</code>
<code>\footnormal</code>	<code>\Xarrangement</code>
<code>\foottwocol</code>	<code>\Xarrangement</code>
<code>\footthreecol</code>	<code>\Xarrangement</code>
<code>\footparagraphX</code>	<code>\arrangementX</code>
<code>\footnormalX</code>	<code>\arrangementX</code>
<code>\foottwocolX</code>	<code>\arrangementX</code>
<code>\footthreecolX</code>	<code>\arrangementX</code>
<code>\afterlemmaseparator</code>	<code>\Xafterlemmaseparator</code>
<code>\afternote</code>	<code>\Xafternote</code> and <code>\afternoteX</code>
<code>\afternumberinfootnote</code>	<code>\Xafternumber</code>
<code>\afterXrule</code>	<code>\Xafterrule</code>
<code>\afterXsymplinenum</code>	<code>\Xaftersymplinenum</code>
<code>\beforelemmaseparator</code>	<code>\Xbeforelemmaseparator</code>
<code>\beforenumberinfootnote</code>	<code>\Xbeforenumber</code>
<code>\beforeXnotes</code>	<code>\Xbeforenotes</code>
<code>\beforeXsymplinenum</code>	<code>\Xbeforesymplinenum</code>

<i>Old command</i>	<i>New command</i>
<code>\bhookXnote</code>	<code>\Xbhookendnote</code>
<code>\bhookXnote</code>	<code>\Xbhooknote</code>
<code>\boxendlinenum</code>	<code>\Xboxendlinenum</code>
<code>\boxlinenum</code>	<code>\Xboxlinenum</code>
<code>\boxlinenumalign</code>	<code>\Xboxlinenumalign</code>
<code>\boxstartlinenum</code>	<code>\Xboxstartlinenum</code>
<code>\boxsymlinenum</code>	<code>\Xboxsymlinenum</code>
<code>\boxXendlinenum</code>	<code>\Xendboxlinenum</code>
<code>\boxXendlinenumalign</code>	<code>\Xendboxlinenumalign</code>
<code>\boxXendstartlinenum</code>	<code>\boxXendstartlinenum</code>
<code>\letboxXendendlinenum</code>	<code>\Xendletboxendlinenum</code>
<code>\hsizetwocol</code>	<code>\Xhsizetwocol</code>
<code>\hsizethreecol</code>	<code>\Xhsizethreecol</code>
<code>\inplaceoflemmaseparator</code>	<code>\Xinplaceoflemmaseparator</code>
<code>\inplaceofnumber</code>	<code>\Xinplaceofnumber</code>
<code>\lemmaseparator</code>	<code>\Xlemmaseparator</code>
<code>\maxhXnotes</code>	<code>\Xmaxhnotes</code>
<code>\morethantwolines</code>	<code>\Xmorethantwolines</code>
<code>\nonumberinfootnote</code>	<code>\Xnonumber</code>
<code>\notesXwidthliketwocolumns</code>	<code>\noteswidthliketwocolumnsX</code>
<code>\noXlemmaseparator</code>	<code>\Xnolemmaseparator</code>
<code>\numberonlyfirstinline</code>	<code>\Xnumberonlyfirstinline</code>
<code>\numberonlyfirstintwolines</code>	<code>\Xnumberonlyfirstintwolines</code>
<code>\nonbreakableafternumber</code>	<code>\Xnonbreakableafternumber</code>
<code>\onlyXpstart</code>	<code>\Xonlypstart</code>
<code>\parafootsep</code>	<code>\Xparafootsep</code> and <code>\parafootsepX</code>
<code>\pstartinfootnote</code>	<code>\Xpstart</code>
<code>\pstartinfootnoteeverytime</code>	<code>\Xpstarteverytime</code>
<code>\symlinenum</code>	<code>\Xsymlinenum</code>
<code>\twolines</code>	<code>\Xtwolines</code>
<code>\twolinesbutnotmore</code>	<code>\Xtwolinesbutnotmore</code>
<code>\twolinesonlyinsamepage</code>	<code>\Xtwolinesonlyinsamepage</code>
<code>\txtbeforeXnotes</code>	<code>\Xtxtbeforenotes</code>
<code>\XendXafterlemmaseparator</code>	<code>\Xendafterlemmaseparator</code>
<code>\XendXbeforelemmaseparator</code>	<code>\Xendbeforelemmaseparator</code>
<code>\XendXinplaceoflemmaseparator</code>	<code>\Xendinplaceoflemmaseparator</code>
<code>\XendXlemmaseparator</code>	<code>\Xendlemmaseparator</code>
<code>\XendXmorethantwolines</code>	<code>\Xendmorethantwolines</code>
<code>\XendXtwolines</code>	<code>\Xendtwolines</code>
<code>\Xnonumberinfootnote</code>	<code>\Xnonumber</code>
<code>\lineref</code>	<code>\edlineref</code>

Appendix A.9.6 Endnotes

With `reledmac`, there is now one auxiliary file for every endnotes set (`.Aend`, `.Bend`, `.Cend` etc.). If you have overridden `\doendnotes` (which you would not have done) you must adapt your code.

Appendix A.9.7 Z Series

The ‘Z’ series of notes has been removed. Only five series are provided now by default: A, B, C, D, E.

Appendix A.9.8 Internal commands

Users who have overridden internal commands, which is wrong, must adapt according to the following. Or better, they should not override any of such commands and use `reledmac` options instead.

- If you have modified `\Xfootfmt`, note that the fourth argument is now mandatory.
- `\unvxh` has been replaced with `\Xunvxh` and `\unvxhX` with two mandatory arguments.

Appendix A.10 Migration to `reledmac` 2.1.0

`Reledmac` 2.1.0 fix some bugs when using `\Xbhooknote` and `\bhooknoteX` not in order to execute code at the beginning of each notes, but to insert content of at the beginning of each notes.

People who use these commands to do it, which is not the original idea, must change the following:

1. Horizontal space is no longer automatically added after the content of the `\Xbhooknote/\bhooknoteX` argument. You must include it manually. So instead of `\Xbhooknote{content}`, use `\Xbhooknote{content }.`
2. Indent is no longer automatically added before the content of the `\Xbhooknote/\bhooknoteX` argument. If you want to keep it, add `\indent` in the argument of `\Xbhooknote/\bhooknoteX`.

Appendix A.11 Migration to `reledmac` 2.1.3

`Reledmac` 2.1.3 fix an historical bug, (style in `ledmac` 0.7!) which doubled the space before the rules of paragraphed familiar footnotes. Consequently, if you use paragraphed familiar footnotes, you should maybe adapt it, playing with `\beforenotesX`.

Appendix A.12 Migration to `reledmac` 2.3.0

Before `reledmac` 2.3.0, for typesetting verse, any empty line was considered a paragraph inside verses. Counting empty lines this created breaking verse, hanging verses, and also added spurious vertical spaces. Version 2.3.0 disables paragraph in stanza. If you want vertical space, use optional argument of `\stanza` or `\endverse`.

Appendix A.13 Migration to reledmac 2.4.0

It is not mandatory, but strongly recommended, to change any `\renewcommand{\endashchar}{\langle...\rangle}` to the use of `\Xlinangeseparator` or `/` and `\Xendlinangeseparator` (7.2.4 p. 40).

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. 43.

`\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.}}
```

Appendix A.21 Migration to reledmac 2.24.0

When using `\labelpstarttrue`, a spurious space^a was added after the `pstart` number (only on normal typesetting, not on parallel typesetting). The new version of the package delete this spurious space. If you consider that it was NOT a spurious space, you should add it manually in your definition of `\thepstart`.

References

- [Bre96] Herbert Breger. `tabmac`. October 1996. (Available from CTAN in `macros/plain/contrib/tabmac`)
- [Bur01] John Burt. ‘Typesetting critical editions of poetry’. *TUGboat*, **22**, 4, pp 353–361, December 2001. (Code available from CTAN in `macros/latex/contrib/poemscol`)
- [Eck03] Matthias Eckermann. *The Parallel-Package*. April 2003. (Available from CTAN in `macros/latex/contrib/parallel`)
- [Fai03] Robin Fairbairns. *footmisc—a portmanteau package for customising footnotes in L^AT_EX*. February 2003. (Available from CTAN in `macros/latex/contrib/footmisc`)
- [LW90] John Lavagnino and Dominik Wujastyk. ‘An overview of `edmac`: a PLAIN TeX format for critical editions’. *TUGboat*, **11**, 4, pp. 623–643, November 1990. (Code available from CTAN in `macros/plain/contrib/edmac`)
- [Lüc03] Uwe Lück. ‘`ednotes`—critical edition typesetting with LaTeX’. *TUGboat*, **24**, 2, pp. 224–236, 2003. (Code available from CTAN in `macros/latex/contrib/ednotes`)
- [Sul92] Wayne G. Sullivan. *The file `edstanza.doc`*. June 1992. (Available from CTAN in `macros/plain/contrib/edmac`)
- [Wil02] Peter Wilson. *The memoir class for configurable typesetting*. November 2002. (Available from CTAN in `macros/latex/contrib/memoir`)
- [Wil04] Peter Wilson and Maïeul Rouquette. *Parallel typesetting for critical editions: the `eledpar` package*. December 2004. (Available from CTAN in `macros/latex/contrib/ledmmac`)

Index

Symbols

<code>\&</code>	51
<code>\@EDROWFILL@</code>	1
<code>\@adv</code>	1
<code>\@advancestanzanumber</code>	1
<code>\@beforeinsertofthisedtext</code>	1
<code>\@doclearpage</code>	1
<code>\@doreinfeetX</code>	1
<code>\@edindex@fornote@</code>	1
<code>\@edindex@hyperref</code>	1
<code>\@edrowfill@</code>	1
<code>\@edtext@level</code>	1

\@emptytoks	1
\@fnpos	1
\@footnotemark	1
\@footnotetext	1
\@getfirstseries	1
\@gobblefour	1
\@gobbleseven	1
\@gobblethree	1
\@h	1
\@hangingsymbol	1
\@iiiminipage	1
\@insertstanza	1
\@k	1
\@l@dttempcnta	1
\@l@dttempcntb	1
\@lab	1
\@led@testifnofoot	1
\@lemma	1
\@line@num	1
\@lock	1
\@lopL	1
\@lopR	1
\@makechapterhead	1
\@makeschapterhead	1
\@mem@extranofeet	1
\@mem@old@ssect	1
\@mpfnpos	1
\@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	71
CLASSbook	71, 347
CLASSmemoir	203, 263–265, 298, 347, 367, 429, 433
CLASSscrbook	433
COMMAND*footnote	72
COMMAND\...\@footnotemark...	206
COMMAND\...d@ta	152
COMMAND\<hook	
@<series	251
COMMAND\<hookname	
<pseudoserries	253, 254
COMMAND\<type	
footfmt	193
COMMAND\@@line	184
COMMAND\@MM	170, 430
COMMAND\@Rlineflag	300, 430
COMMAND\@Serefprefix	277
COMMAND\@Serefprefixmore	277
COMMAND\@add@	337
COMMAND\@adv	109
COMMAND\@apprefprefixmore	277
COMMAND\@apprefprefixsingle	277
COMMAND\@beforeinsertofthisedtext	132
COMMAND\@bsphack	266
COMMAND\@doclearpage	264, 265, 423, 433
COMMAND\@doreinfeetX	433
COMMAND\@dprintingcolumns	430
COMMAND\@edindex@hyperref	300, 301
COMMAND\@edtext@level	129
COMMAND\@esphack	266
COMMAND\@firstofone	33
COMMAND\@fnpos	224, 260
COMMAND\@footnotemark	203, 204, 423, 433
COMMAND\@footnotetext	203, 204, 423
COMMAND\@gobble	32, 127, 128, 246
COMMAND\@gobblefive	432
COMMAND\@gobblefour	429
COMMAND\@gobbleseven	248
COMMAND\@gobblethree	422
COMMAND\@h	187
COMMAND\@hangingsymbol	305
COMMAND\@iiiminipage	289, 291, 422, 433
COMMAND\@iiiminipage	289
COMMAND\@l	428
COMMAND\@l@tempcnta	155, 157, 164
COMMAND\@l@tempcntb	157
COMMAND\@l@reg	428
COMMAND\@lab	106, 266, 269, 273, 422

COMMAND\@ldunboxmpfoot	292
COMMAND\@led@extranofeet	368
COMMAND\@ledinnote@command	296
COMMAND\@lemma	132, 134
COMMAND\@lock	100, 305
COMMAND\@lopL	423
COMMAND\@lopR	423
COMMAND\@makecol	260, 261, 263, 433
COMMAND\@mpfnpos	224
COMMAND\@msd	315
COMMAND\@msd@c	315
COMMAND\@msd@options@iffullpage	321
COMMAND\@msdata@list	315, 316
COMMAND\@nl	106–109, 111, 120, 269, 422, 423
COMMAND\@nl@reg	107, 365, 423, 428
COMMAND\@opXfeet	423
COMMAND\@opfeetX	433
COMMAND\@opextrafeeti	433
COMMAND\@page	108, 269
COMMAND\@pend	423
COMMAND\@pendR	423
COMMAND\@ref	106, 116–118, 121, 122, 127
COMMAND\@ref@later	117, 123
COMMAND\@ref@reg	116, 423
COMMAND\@ref@reg@parsearg	117
COMMAND\@reinserts	260–263, 433
COMMAND\@secondoftwo	73
COMMAND\@sect	350
COMMAND\@series	250
COMMAND\@set	110
COMMAND\@sidenotesep	288
COMMAND\@stopmsd	315
COMMAND\@sw	117, 118, 135, 139, 140
COMMAND\@tag	129, 130, 133
COMMAND\@tempcnta	85
COMMAND\@tempcntb	85
COMMAND\@toksa	92
COMMAND\@toksb	92
COMMAND\@xloop	166
COMMAND\@xympar	281, 433
COMMAND\Aendnote	16, 25
COMMAND\Afootfmt	170
COMMAND\Afootgroup	170
COMMAND\Afootnote	8, 16, 24, 25, 28, 131, 179, 203, 225, 241, 432
COMMAND\Afootstart	170
COMMAND\AtBeginDocument	263
COMMAND\AtEndEveryPend	19, 439
COMMAND\AtEveryPend	18, 19, 54, 146, 430, 431, 433, 439
COMMAND\AtEveryPend*	19
COMMAND\AtEveryPstart	18, 19, 54, 430, 431, 433, 436, 439

COMMAND\AtEveryPstart*	19
COMMAND\AtEveryStanza	54, 437, 439
COMMAND\AtEveryStopStanza	54, 437, 439
COMMAND\AtStartEveryPstart	19, 439
COMMAND\AtStartEveryStanza	54, 439
COMMAND\BeforeEveryStopStanza	439
COMMAND\Bendnote	16, 24
COMMAND\Bfootnote	8, 16, 203, 225, 241
COMMAND\Centering	47
COMMAND\Cfootnote	203
COMMAND\Columns	86, 175
COMMAND\Dfootnote	203
COMMAND\Efootnote	203
COMMAND\Gls	64
COMMAND\Hy@raisedlink	359
COMMAND\Hy@raisedlink@left	360
COMMAND\LTR	47
COMMAND\NR	350
COMMAND\Pages	86, 261, 262
COMMAND\ProcessOptionsX	76
COMMAND\RL	46
COMMAND\RaggedLeft	47
COMMAND\RaggedRight	47
COMMAND\RenewExpandableDocumentCommand	33, 138
COMMAND\SEonlypage	275, 435
COMMAND\SEref	58–60, 275, 277, 436, 438
COMMAND\SErefonlypage	58–60, 371, 435
COMMAND\SErefwithpage	58, 60, 275, 277, 371, 435, 437
COMMAND\Stanza	428
COMMAND\Waklam	338
COMMAND\X@doreinfeet	262, 433
COMMAND\XR@prefix	281
COMMAND\XR@test	280, 281
COMMAND\XR@test@mac	281
COMMAND\XR@test@mac@test	281
COMMAND\XXXXXXfmt	364
COMMAND\XXXXXXfmt	364
COMMAND\Xafterlemmaseparator	44, 368
COMMAND\Xafternote	47, 48, 367, 368
COMMAND\Xafternumber	42, 368
COMMAND\Xafterrule	49, 226, 368, 428, 431
COMMAND\Xaftersymlinenum	42, 368
COMMAND\Xarrangement	38, 48, 72, 171, 172, 252, 368
COMMAND\Xarrangement@footparagraph	177
COMMAND\Xarrangement@normal	172
COMMAND\Xarrangement@paragraph	177
COMMAND\Xbeforeinserting	46, 47
COMMAND\Xbeforelemmaseparator	44, 368
COMMAND\Xbeforenotes	49, 225, 368, 428, 431
COMMAND\Xbeforenumber	40, 42, 368

COMMAND\Xbeforesymlinenum	42, 368
COMMAND\Xbhookendnote	369
COMMAND\Xbhookgroup	48, 435, 436
COMMAND\Xbhooknote	46, 369, 370, 434
COMMAND\Xboxendlinenum	43, 369, 432
COMMAND\Xboxlinenum	42, 43, 369
COMMAND\Xboxlinenumalign	43, 369, 432
COMMAND\Xboxstartlinenum	43, 369, 432
COMMAND\Xboxsymlinenum	42, 43, 369
COMMAND\Xcolalign	47, 431
COMMAND\Xdo@feet	433, 438
COMMAND\Xend	248
COMMAND\XendXafterlemmaseparator	369
COMMAND\XendXbeforelemmaseparator	369
COMMAND\XendXinplaceoflemmaseparator	369
COMMAND\XendXlemmaseparator	369
COMMAND\XendXmorethantwolines	369
COMMAND\XendXtwolines	369
COMMAND\Xendafterenumber	42, 434
COMMAND\Xendafterlemmaseparator	44, 369
COMMAND\Xendafternote	50, 436
COMMAND\Xendafternumber	44
COMMAND\Xendafterpagenumbe	437
COMMAND\Xendafterpagenumber	43, 60
COMMAND\XendafterpagenumberSErefonlypage	371
COMMAND\Xendaftersymlinenum	42, 44, 434
COMMAND\Xendahookinplaceofnumber	44, 434
COMMAND\Xendahooklinenum	44, 434
COMMAND\Xendbeforelemmaseparator	44, 369
COMMAND\Xendbeforelinenum	43
COMMAND\Xendbeforenumber	42, 434
COMMAND\Xendbeforepagenumber	43, 59, 60
COMMAND\XendbeforepagenumberSErefonlypage	59
COMMAND\Xendbeforesymlinenum	42, 44, 434
COMMAND\Xendbhookinplaceofnumber	44, 434
COMMAND\Xendbhooklinenum	43, 434
COMMAND\Xendbhooknote	46
COMMAND\Xendboxendlinenum	43, 432
COMMAND\Xendboxlinenum	43, 369, 430
COMMAND\Xendboxlinenumalign	43, 369, 432
COMMAND\Xendboxstartlinenum	43, 432
COMMAND\Xendboxsymlinenum	43, 434
COMMAND\Xendhangindent	46, 434, 436
COMMAND\Xendinplaceoflemmaseparator	26, 44, 369
COMMAND\Xendinplaceofnumber	42, 433
COMMAND\Xendinplaceofpagenumber	40, 438
COMMAND\Xendinsertsep@	232
COMMAND\Xendlemmadisablefontselection	45
COMMAND\Xendlemmafont	45, 436
COMMAND\Xendlemmaseparator	26, 44, 369

COMMAND\Xendletboxendlinenum	369
COMMAND\Xendlineflag	60
COMMAND\Xendlineprefixmore	43, 60
COMMAND\Xendlineprefixsingle	43, 60
COMMAND\Xendlinerangeseparator	40, 60, 191, 371, 435
COMMAND\Xendmorethantwolines	26, 41, 60, 369, 431, 432
COMMAND\Xendnonumber	41, 433
COMMAND\Xendnote	228, 247, 248, 431
COMMAND\Xendnotefontsize	45, 368
COMMAND\Xendnotenumfont	43–45, 368
COMMAND\Xendnotes	231
COMMAND\Xendnumberonlyfirstinline	39, 434
COMMAND\Xendnumberonlyfirstintwolines	39, 434
COMMAND\Xendpagenumberonlyfirst	39, 438
COMMAND\Xendpagenumberonlyfirstifsingle	39, 438
COMMAND\Xendpagenumberonlyfirstintwo	39, 438
COMMAND\Xendparagraph	50, 428
COMMAND\Xendsep	50
COMMAND\Xendsublinesep	41, 60, 191
COMMAND\Xendsymlinenum	39, 434
COMMAND\Xendsympagenum	40, 438
COMMAND\Xendtwolines	26, 40, 41, 60, 369, 431, 432
COMMAND\Xendtwolinesbutnotmore	41, 60, 431, 432
COMMAND\Xendtwolinesonlyinsamepage	41, 60, 431, 432
COMMAND\Xendwrapcontent	46, 437
COMMAND\Xendwraplemma	46, 437
COMMAND\Xfootfmt	370
COMMAND\Xfootgroup	176
COMMAND\Xfootins	175
COMMAND\Xfootnote	57, 63, 129, 364, 425, 429, 431, 435, 437
COMMAND\Xfootstarts	176
COMMAND\Xgroupbyline	48, 163, 201
COMMAND\Xgroupbylines	438
COMMAND\Xgroupbylineseparetwolines	48
COMMAND\Xhangindent	46, 434
COMMAND\Xhsize	371, 435, 436
COMMAND\Xsizethreecol	47, 50, 369
COMMAND\Xsizetwocol	47, 50, 253, 369
COMMAND\Xinplaceoflemmaseparator	25, 44, 369
COMMAND\Xinplaceofnumber	42, 369, 431, 432
COMMAND\Xinsertparafootsep	181, 183
COMMAND\Xledsetnormalparstuff	366, 367, 431
COMMAND\Xlemmadisablefontselection	45
COMMAND\Xlemmafont	45, 436
COMMAND\Xlemmaseparator	44, 191, 255, 257, 259, 369
COMMAND\Xlineflag	59, 371, 372, 437
COMMAND\Xlinerangeseparator	40, 59, 191, 371, 435
COMMAND\Xmaxhnotes	49, 71, 72, 368, 369, 428, 430
COMMAND\Xmorethantwolines	25, 40, 41, 59, 369, 430
COMMAND\Xnoindent	434

COMMAND\Xnolemmaseparator	44, 259, 369
COMMAND\Xnonbreakableafternumber	42, 369, 426
COMMAND\Xnonumber	41, 369
COMMAND\Xnonumberinfootnote	369
COMMAND\Xnotefontsize	45, 368
COMMAND\Xnotefontsize@⟨s⟩	182, 186, 187
COMMAND\Xnotenumfont	45, 368
COMMAND\Xnoteswidthliketwocolumns	50, 429
COMMAND\Xnumberonlyfirstinline	39, 40, 48, 103, 193, 254, 255, 257, 369, 425, 430, 439
COMMAND\Xnumberonlyfirstintwolines	39, 48, 369, 425, 439
COMMAND\Xonlypstart	41, 369, 425, 430
COMMAND\Xpagelinesep	42, 438
COMMAND\Xparafootsep	48, 103, 368, 369, 437, 439
COMMAND\Xparafootsep@series	181
COMMAND\Xparinden	371
COMMAND\Xparindent	46, 431, 434, 436
COMMAND\Xprenotes	49, 226, 437
COMMAND\Xprenotes@	175, 226, 425
COMMAND\Xpstart	41, 366, 369, 425, 430
COMMAND\Xpstarteverytime	41, 366, 369, 430
COMMAND\Xragged	48
COMMAND\Xrigidbalance	184, 371, 435
COMMAND\Xstanza	41, 54
COMMAND\Xstanzaseparator	41
COMMAND\Xstorelineinfo	193
COMMAND\Xsublinesep	22, 41, 42, 60, 191, 371
COMMAND\Xsublinesepside	22, 41
COMMAND\Xsymlinenum	39, 48, 368, 369, 432, 439
COMMAND\Xtextbeforenotes	166
COMMAND\Xtoendnotes	27, 248
COMMAND\Xtwolines	25, 40, 41, 60, 198, 199, 253, 369, 430
COMMAND\Xtwolinesappref	253
COMMAND\Xtwolinesbutnotmore	40, 41, 60, 369, 431
COMMAND\Xtwolinesbutnotmoreappref	254
COMMAND\Xtwolinesonlyinsamepage	40, 41, 60, 369, 431
COMMAND\Xtxtbeforenotes	48, 369, 437, 438, 440
COMMAND\Xtxtbeforenotesonlyonce	48, 440
COMMAND\Xunvxh	179, 370
COMMAND\Xwidth	50, 371, 436
COMMAND\Xwrapcontent	46, 437
COMMAND\Xwraplemma	45–47, 437
COMMAND\&	368
COMMAND\⟨XXX⟩vfootnote	201
COMMAND\absline@num	100, 154
COMMAND\accent	128
COMMAND\actionlines@list	101, 155
COMMAND\actions@list	101
COMMAND\add@Xgroupbyline	163
COMMAND\add@inserts	101, 162, 163
COMMAND\add@inserts@next	162, 163

COMMAND\add@msd@	315
COMMAND\add@msdata	315, 316
COMMAND\add@msdata@firstlineofpage	318
COMMAND\add@msddata	315
COMMAND\add@penalties	153, 164
COMMAND\addcontentsline	345
COMMAND\addfootins	364, 367
COMMAND\addfootinsX	364, 367
COMMAND\addtoendnotes	248
COMMAND\advancelabel@refs	268
COMMAND\advanceline	23, 102, 109, 124, 433
COMMAND\affixlin@num	288
COMMAND\affixline@num	156, 160, 161, 423
COMMAND\affixpstart@num	161
COMMAND\afterXrule	368
COMMAND\afterXsymlinenum	368
COMMAND\afterenumber	42
COMMAND\aftergroup	127, 131
COMMAND\afterlemmaseparator	368
COMMAND\afternote	368
COMMAND\afternoteX	48, 367, 368
COMMAND\afternumberinfootnote	368
COMMAND\afterruleX	49, 428, 431
COMMAND\applabel	59, 270, 271, 431, 437
COMMAND\appref	57, 59, 60, 275, 277, 435, 436
COMMAND\apprefprefixmore	59, 368
COMMAND\apprefprefixsingle	59, 368
COMMAND\apprefwithpage	59, 60, 275, 277, 432, 435
COMMAND\arrangementX	38, 72, 207, 252, 368
COMMAND\arrangementX@normal	212
COMMAND\article	15
COMMAND\at@every@pend	146
COMMAND\autopar	18, 142, 147, 148, 222, 424, 426, 427, 431
COMMAND\ballast	72
COMMAND\ballast@count	153, 164
COMMAND\baselineskip	38, 178, 182
COMMAND\beforeXnotes	368
COMMAND\beforeXsymlinenum	368
COMMAND\beforeeledchapter	10, 69, 346
COMMAND\beforeinsertingX	46
COMMAND\beforelemmaseparator	368
COMMAND\beforenotesX	49, 370, 427, 428, 431
COMMAND\beforenumberinfootnote	368
COMMAND\begin	323
COMMAND\beginnumbering	16, 17, 19, 20, 86, 87, 90, 91, 99, 104, 119, 120, 147, 228, 314, 425, 428, 432, 433, 438, 439
COMMAND\bf	425
COMMAND\bfseries	45, 425
COMMAND\bhookXnote	369
COMMAND\bhookgroupX	49, 435

COMMAND\bhooknoteX	46, 370, 434
COMMAND\body	306
COMMAND\bodyfootmarkA	35
COMMAND\book	15
COMMAND\boxXendlinenum	369
COMMAND\boxXendlinenumalign	369
COMMAND\boxXendstartlinenum	369
COMMAND\boxendlinenum	369
COMMAND\boxlinefootnote	195
COMMAND\boxlinenum	369
COMMAND\boxlinenumalign	369
COMMAND\boxstartlinenum	369
COMMAND\boxsymlinenum	369
COMMAND\break	38, 179
COMMAND\brokenpenalty	164
COMMAND\centering	47
COMMAND\ch@ck@l@ck	423
COMMAND\ch@cksub@l@ck	160, 423
COMMAND\chapter	68, 347, 351, 428, 431, 433, 437
COMMAND\chaptermark	345
COMMAND\check@pb@in@verse	359
COMMAND\colalignX	47, 431
COMMAND\collect@body	323
COMMAND\color	437
COMMAND\colorbox	73
COMMAND\columns	50
COMMAND\columnwidth	178, 429
COMMAND\command names	253, 254
COMMAND\copyright	128
COMMAND\correct@Xfootins@box	430
COMMAND\correct@footinsX@box	430
COMMAND\count	185
COMMAND\critex	424
COMMAND\critext	134, 363, 364, 367
COMMAND\csname	77, 137
COMMAND\csquotes	245
COMMAND\ctab	339, 343
COMMAND\ctabtext	344
COMMAND\dc col	332
COMMAND\def	74, 89
COMMAND\detokenize	137
COMMAND\dimen	185
COMMAND\dimexpr	50
COMMAND\discretionary	179
COMMAND\displaywidowpenalty	164
COMMAND\do@Xfeet	261, 423, 433, 438
COMMAND\do@actions	153, 154, 156, 423
COMMAND\do@actions@fixedcode	423
COMMAND\do@actions@next	154, 155
COMMAND\do@ballast	153, 154, 164

COMMAND\do@feet@custom@order	260
COMMAND\do@insidelinehook	426
COMMAND\do@line	101, 126, 145, 148, 151, 153, 162, 164, 305, 423, 424, 426, 428
COMMAND\do@linehook	423
COMMAND\do@lockoff	102
COMMAND\do@lockon	102
COMMAND\dodoreintrafeet	422
COMMAND\doendnotes	26, 232, 370, 432, 439
COMMAND\doendnotesbysection	26, 232, 248, 432, 439
COMMAND\doennotes	439
COMMAND\doinsidelinehook	24, 429
COMMAND\dolinehook	24, 429
COMMAND\doreintrafeeti	433
COMMAND\doreintrafeetii	433
COMMAND\doxtrafeet	260, 422
COMMAND\doxtrafeeti	433
COMMAND\doxtrafeetii	433
COMMAND\dummy@ref	127
COMMAND\edaftertab	67, 338, 339
COMMAND\edatleft	67, 336
COMMAND\edatright	67, 337
COMMAND\edbeforetab	67, 338, 339
COMMAND\edfilldimen	337
COMMAND\edfont@info	133
COMMAND\edgls	64, 295
COMMAND\edglsadd	439
COMMAND\edgls...	436
COMMAND\edindex	33, 62–64, 294, 295, 299, 301, 327, 426, 429, 430, 433, 434, 438–440
COMMAND\edindexlab	64
COMMAND\edlabel	56–60, 128, 266, 268, 269, 272, 273, 280, 295, 327, 360, 372, 422, 425–427, 430, 435, 440
COMMAND\edlabelE	58, 271
COMMAND\edlabelS	58, 271
COMMAND\edlabelSE	58
COMMAND\edlineref	56, 266, 369, 430, 432, 435, 439
COMMAND\edmakelabel	58, 280
COMMAND\edpageref	56, 266, 272, 280
COMMAND\edrowfill	338
COMMAND\edsublineref	56
COMMAND\edtabcolsep	331
COMMAND\edtext	7, 24, 25, 27, 28, 30–32, 51, 56–59, 62, 65, 72, 100, 101, 116, 118, 121, 122, 126–134, 136, 137, 139–141, 270, 271, 274, 327, 328, 347, 360, 363, 364, 367, 422, 424, 426, 428–432, 438–440
COMMAND\edtext@level	432
COMMAND\edtextlater	118
COMMAND\edvertdots	68, 337
COMMAND\edvertline	67, 68, 337
COMMAND\elechapter	69
COMMAND\eled@sectioning@out	352
COMMAND\eledchapter	68, 367, 429, 433

COMMAND\eledchapter*	68
COMMAND\eledmac@error	422
COMMAND\eledsection	7, 16, 68, 127, 151, 346, 367, 431, 439
COMMAND\eledsection*	68
COMMAND\eledsubsection	68, 367
COMMAND\eledsubsection*	68
COMMAND\eledsubsubsection	68, 367
COMMAND\eledsubsubsection*	68
COMMAND\eledxxx	10, 69, 352, 428
COMMAND\eledxxxx	345
COMMAND\elsection	352
COMMAND\else	294, 346
COMMAND\emph	33
COMMAND\empty	85, 157, 158, 266
COMMAND\end	322, 323
COMMAND\end@lemmas	127
COMMAND\endashchar	191
COMMAND\endgraf	145, 181, 222
COMMAND\endlock	22, 102, 125, 310
COMMAND\endminipage	289, 291, 422, 433
COMMAND\endmsdata	34
COMMAND\endnotes	431, 435
COMMAND\endnumbering	17, 20, 86, 87, 89, 90, 120, 423, 432, 438, 439
COMMAND\endprint	228, 231, 248, 366
COMMAND\endstanzaextra	368
COMMAND\endsub	22, 102, 123
COMMAND\endverse	370
COMMAND\everypar	147
COMMAND\extensionchars	70, 87
COMMAND\externaldocument	60, 61, 280
COMMAND\f@x@l@cks	423
COMMAND>falseverse	367, 426, 428
COMMAND\fi	346
COMMAND\firstlinenum	21, 157, 424
COMMAND\firstsublinenum	21, 424
COMMAND\fix@page	107, 108, 423
COMMAND\flag@end	121, 122, 133, 428
COMMAND\flag@end@RTL	122
COMMAND\flag@end@later	123
COMMAND\flag@start	121, 122, 133, 428, 429
COMMAND\flag@start@RTL	122
COMMAND\flag@start@later	123
COMMAND\flagstanza	55
COMMAND\floatingpenalty	170, 430
COMMAND\flush@notes	165, 166
COMMAND\fnpos	37, 224, 427, 438
COMMAND\footfmt	170, 172
COMMAND\footfmt...	207
COMMAND\footfootmarkA	35
COMMAND\footfudgefactor	179

COMMAND\footfudgefiddle	72, 177, 178, 422
COMMAND\footgroup	170
COMMAND\footins	175
COMMAND\footnormal	253, 368, 422
COMMAND\footnormalX	368
COMMAND\footnote	35, 71, 203, 204, 365, 423
COMMAND\footnote@lang	191
COMMAND\footnoteA	16, 35
COMMAND\footnoteB	16
COMMAND\footnoteC	24
COMMAND\footnoteE	35
COMMAND\footnoteX	8, 36, 245, 246, 438
COMMAND\footnoteX@reading	246
COMMAND\footnoteXmark	246, 439
COMMAND\footnoteXmk	259
COMMAND\footnoteXtext	246, 439
COMMAND\footnote⟨X⟩	128
COMMAND\footnote⟨X⟩mark	36
COMMAND\footnote⟨X⟩mk	36
COMMAND\footnote⟨X⟩nomk	36
COMMAND\footnote⟨X⟩text	36
COMMAND\footnotelang@lua	169
COMMAND\footnotelang@poly	169
COMMAND\footnotemark	36, 246
COMMAND\footnoteoption@	168, 434
COMMAND\footnoterule	185
COMMAND\footnotesize	45
COMMAND\footnotetext	36, 246
COMMAND\footparagraph	178, 253, 368, 428
COMMAND\footparagraphX	217, 368, 428
COMMAND\footsplitskips	423, 430
COMMAND\footstart	170, 175, 185
COMMAND\footstrut	181
COMMAND\footthreecol	368
COMMAND\footthreecolX	368, 432
COMMAND\foottwocol	368
COMMAND\foottwocolX	368, 432
COMMAND\foreignlanguage	46
COMMAND\fullstop	371
COMMAND\get@edindex@hyperref	300
COMMAND\get@edindex@ledinnote@command	296
COMMAND\get@fnmark	204
COMMAND\get@index@command	427
COMMAND\get@linelistfile	423
COMMAND\get@thisfootnote	211
COMMAND\getline@num	153, 155
COMMAND\gl@p	92
COMMAND\global	106
COMMAND\globaldefs	106
COMMAND\gls	64, 303

COMMAND\hangindentX	46, 431, 434
COMMAND\hangingsymbol	368, 424
COMMAND\hbox	179
COMMAND\hfill	427
COMMAND\hidenumbering	23, 114, 431
COMMAND\hidenumberingonleftpage	23, 114, 437
COMMAND\hidenumberingonrightpage	23, 114, 437
COMMAND\hline	65
COMMAND\hrulefill	338
COMMAND\hsize	39, 175, 178, 180, 186, 189, 223, 423, 429
COMMAND\hsizeX	371, 435, 436
COMMAND\hsizethreecol	369
COMMAND\hsizethreecolX	47, 50
COMMAND\hsizetwocol	369
COMMAND\hsizetwocolX	47, 50
COMMAND\hyperlinkR	300
COMMAND\hyperlinkformat	299
COMMAND\hyperlinkformatR	300
COMMAND\if@RTL	78, 131
COMMAND\if@edtext@	429, 432
COMMAND\if@eled@sectioning	352
COMMAND\if@firstlineofpage	78
COMMAND\if@firstlineofpageR	78
COMMAND\if@msd@options@fullpage	321
COMMAND\if@msdata@insertedfrompreviouspage	318
COMMAND\if@nobreak	146
COMMAND\if@noneed@Footnote	121
COMMAND\ifXnote@	86
COMMAND\ifbypage@	93
COMMAND\ifbypage@R	93
COMMAND\ifbypstart@	93
COMMAND\ifbypstart@R	93
COMMAND\iffirst@linenum@out@	119, 120
COMMAND\ifindtl@innote	86
COMMAND\ifindtl@notenumber	86
COMMAND\ifinserthangingsymbol	305
COMMAND\ifinstanza	305
COMMAND\ifstwofollowinglines	199
COMMAND\ifl@d@Xmorethantwolines	196, 430
COMMAND\ifl@d@Xtwolines	196
COMMAND\ifl@d@dash	196
COMMAND\ifl@d@elin	196
COMMAND\ifl@d@esl	196
COMMAND\ifl@d@pnum	196
COMMAND\ifl@d@ssub	196
COMMAND\ifl@dend@X	247
COMMAND\ifl@dmemoir	422
COMMAND\ifl@dpaging	429
COMMAND\ifl@dpairing	86, 424
COMMAND\ifl@dprintingpages	430

COMMAND\ifl@dskipnumber	157
COMMAND\ifl@dstartendok	337
COMMAND\ifl@imakeidx	78
COMMAND\ifledRcol	86, 424
COMMAND\ifledRcol@	86, 428
COMMAND\iflemmacommand@	429
COMMAND\ifnoend@	233
COMMAND\ifnoledgroup@	294
COMMAND\ifnoteschanged@	103
COMMAND\ifnumberedpar@	142
COMMAND\ifnumbering	87, 90
COMMAND\ifnumberingR	86, 424
COMMAND\ifnumberline	133, 157
COMMAND\ifpst@rted	424
COMMAND\ifpst@rtedL	87
COMMAND\ifresumenumbering@start	90
COMMAND\ifseriesbefore	251
COMMAND\ifstopmsdata@inserted@	314
COMMAND\ifsublines@	99, 112
COMMAND\iftrue	432
COMMAND\ifvmode	268
COMMAND\ifxxx	346
COMMAND\ignorespaces	131
COMMAND\imki@wrindexentry	78
COMMAND\immediate	119, 120, 227
COMMAND\indent	18, 147, 370
COMMAND\index	302, 303, 438
COMMAND\indtl@wrindexentry	78
COMMAND\initnumbering@quote	344, 433
COMMAND\initnumbering@reg	423
COMMAND\initnumbering@sectcmd	433
COMMAND\inplaceoflemmaseparator	369
COMMAND\inplaceofnumber	369
COMMAND\insert	132, 162, 170, 172, 173, 187, 201, 207
COMMAND\insert@Xtxtbeforenotes	167, 187
COMMAND\insert@count	116, 121, 130
COMMAND\insert@countR	130
COMMAND\insert@msdata	315, 321
COMMAND\insert@txtbeforenotesX	167
COMMAND\inserthangingsymbol	427
COMMAND\insertlines@list	101, 116
COMMAND\insertparafootsepX	221
COMMAND\inserts@list	126, 142, 162, 163, 179
COMMAND\interAfootnotelinepenalty	424
COMMAND\interfootnotelinepenalty	424
COMMAND\interlinepenalty	170
COMMAND\interparanoteglue	367
COMMAND\justifying	47
COMMAND\l@advance@parledegroupp@beforenormalnotes	433
COMMAND\l@d@wrindexhyp	429

COMMAND\l@d@add	135
COMMAND\l@d@end	228, 247
COMMAND\l@d@nums	130, 133–135, 196
COMMAND\l@d@section	228
COMMAND\l@d@set	111, 124
COMMAND\l@dampcount	329
COMMAND\l@dbfnote	204, 423
COMMAND\l@dcheckstartend	337
COMMAND\l@dchset@num	111
COMMAND\l@dcolcount	329, 330
COMMAND\l@dcollect@@body	323
COMMAND\l@dcollect@body	322
COMMAND\l@dcsnote	428
COMMAND\l@dcsnotetext	152, 285
COMMAND\l@dcsnotetext@l	152, 285
COMMAND\l@dcsnotetext@r	152, 285
COMMAND\l@ddodorextrafeet	261, 422
COMMAND\l@ddoxtrafeet	261, 422
COMMAND\l@demptyd@ta	424
COMMAND\l@dend@close	227
COMMAND\l@dend@open	227
COMMAND\l@dend@stuff	228
COMMAND\l@denvbody	323
COMMAND\l@dfeetbeginmini	422
COMMAND\l@dfeetendmini	422
COMMAND\l@dgetline@margin	424
COMMAND\l@dgetlock@disp	424
COMMAND\l@dgetref@num	273, 274
COMMAND\l@dgetsidenote@margin	282, 424
COMMAND\l@dgobbeloptarg	429
COMMAND\l@dgonblearg	429
COMMAND\l@dgonbleoptarg	327
COMMAND\l@dlabel@parse	273, 274
COMMAND\l@dld@ta	156, 158
COMMAND\l@dlp@rbox	287
COMMAND\l@dlsn@te	424
COMMAND\l@dlsnote	428
COMMAND\l@dmake@labels	268, 269, 281
COMMAND\l@dmake@labelsR	281
COMMAND\l@dnumpstartsL	87, 424
COMMAND\l@dp@rsefootspec	196
COMMAND\l@dp@rsefootspec	196
COMMAND\l@dpush@begins	323
COMMAND\l@d@rd@ta	156, 158
COMMAND\l@dref@undefined	273
COMMAND\l@d@rsn@te	424
COMMAND\l@d@rsnote	428
COMMAND\l@dtabaddcols	337
COMMAND\l@dtabnoexpands	422
COMMAND\l@dumboxmpfoot	433

COMMAND\l@dunboxmpfoot	424
COMMAND\l@dzeropenalties	424, 429
COMMAND\l@pb	358
COMMAND\l@prev@nopb	357
COMMAND\l@prev@pb	357
COMMAND\l@reg	365
COMMAND\label	19, 58, 60, 64, 266, 267, 274
COMMAND\label@refs	266
COMMAND\labelstarttrue	19, 372, 425, 440
COMMAND\labelref@list	266, 269
COMMAND\language	179
COMMAND\last@page@num	423
COMMAND\lastbox	147
COMMAND\lastskip	123
COMMAND\latex@makecol	263
COMMAND\leavevmode	18, 147
COMMAND\led@check@nopb	358
COMMAND\led@check@pb	358
COMMAND\led@nopb	358, 359
COMMAND\led@nopbnum	358
COMMAND\led@pb	358, 359
COMMAND\led@pb@macro	358
COMMAND\led@pbnum	358
COMMAND\led@reinit@index@fornote	303
COMMAND\led@set@index@fornote	302
COMMAND\ledRflag	300
COMMAND\ledchapter	367, 426
COMMAND\ledfootinsdim	368
COMMAND\ledinnernote	61, 283, 428, 437
COMMAND\ledinnote	297, 366, 432
COMMAND\ledinnotemark	63, 366, 431
COMMAND\ledleftnote	61, 283
COMMAND\ledlinenum	98, 424
COMMAND\ledllfill	153
COMMAND\ledlsnotefontsetup	437
COMMAND\ledlsnotesep	61
COMMAND\ledlsnotewidth	61
COMMAND\lednopb	70, 358
COMMAND\lednopbinverse	359
COMMAND\lednopbinversetrue	53, 70
COMMAND\lednopbnum	358
COMMAND\ledouternote	61, 283, 428, 437
COMMAND\ledpb	70, 358, 367
COMMAND\ledpbnum	358
COMMAND\ledpbsetting	70, 358, 434
COMMAND\ledrightnote	61, 283
COMMAND\ledrsnotefontsetup	437
COMMAND\ledrsnotesep	61
COMMAND\ledrsnotewidth	61
COMMAND\ledsection	367

COMMAND\ledsectnomark	345
COMMAND\ledsectnotoc	345
COMMAND\ledsetnormalparstuff	366, 367, 431
COMMAND\ledsetnormalparstuff@common	222
COMMAND\ledsetnormalparstuffX	366, 367, 431
COMMAND\ledsidenote	61, 283, 285
COMMAND\ledsubsection	367
COMMAND\ledsubsubsection	367
COMMAND\ledxxx	428
COMMAND\left	67
COMMAND\leftctab	339
COMMAND\leftheadline	98
COMMAND\leftlinenum	22, 98, 422, 424
COMMAND\leftltab	338
COMMAND\leftnoteupfalse	61
COMMAND\leftpstartnum	161
COMMAND\leftrtab	338
COMMAND\leftsidenote	285
COMMAND\leftskip	175, 178, 179
COMMAND\lemma	3, 25, 27, 28, 30–32, 126, 130, 131, 133, 134, 136, 363, 424, 425, 432, 433, 435
COMMAND\lemmaseparator	369
COMMAND\let	28, 51, 128, 132, 246, 263, 310, 422
COMMAND\letboxXendendlinenum	369
COMMAND\line	184, 187
COMMAND\line@list	100, 117, 133
COMMAND\line@list@stuff	87, 104, 119, 422, 424
COMMAND\line@list@version	106
COMMAND\line@margin	94, 158, 281
COMMAND\line@num	99–101, 157, 422
COMMAND\line@set	134, 135
COMMAND\lineation	21, 93, 440
COMMAND\linebreak	38
COMMAND\lineneation	436
COMMAND\linenum	25, 27, 28, 57, 59, 126, 134, 272, 274, 280, 363
COMMAND\linenum@out	118, 119, 266, 269
COMMAND\linenumberlist	21, 85, 157, 158, 422
COMMAND\linenumberstyle	23, 98, 422, 439
COMMAND\linenumincrement	21, 424
COMMAND\linenummargin	21, 94, 281
COMMAND\linenumr@p	97, 422, 424
COMMAND\linenumrep	97, 98, 424
COMMAND\linenumsep	22, 61, 98, 283
COMMAND\linerangesep@	259
COMMAND\lineref	266, 272, 280, 369, 430
COMMAND\list@clear	92
COMMAND\list@clearing@reg	424
COMMAND\list@create	92
COMMAND\lock@disp	96
COMMAND\lock@off	113
COMMAND\lock@on	112

COMMAND\lockdisp	22, 96
COMMAND\loop	166, 306
COMMAND\ltab	338, 339, 343
COMMAND\ltabtext	344
COMMAND\m@mmf@prepare	204
COMMAND\makeatletter	152
COMMAND\makehboxoffhboxes	180, 182
COMMAND\makeindex	62, 299
COMMAND\makelabel	280
COMMAND\managestanza@modulo	307
COMMAND\marginpar	61, 71, 281, 423
COMMAND\marginparwidth	61, 283
COMMAND\markboth	151
COMMAND\mathchardef	306
COMMAND\maxhXnotes	369
COMMAND\maxhnotesX	49, 72, 368, 427, 428, 430–432
COMMAND\maxlinesinpar@list	104
COMMAND\measurembdy	340
COMMAND\measuretbody	341
COMMAND\memorybreak	20
COMMAND\morenoexpands	72, 73, 126, 128
COMMAND\morethantwolines	369
COMMAND\mpfnpos	37, 224, 427, 438
COMMAND\mpnnormalfootgroup	423
COMMAND\mpnnormalvfootnote	423
COMMAND\msdata	34, 313–315, 439
COMMAND\msdataposition	35
COMMAND\multfootsep	35, 203
COMMAND\multiplefootnotemarker	203
COMMAND\musixtex	428
COMMAND\n@num	424, 431
COMMAND\n@num@ref	431
COMMAND\new@line	120, 423
COMMAND\new@series	128
COMMAND\newcommand	28, 74, 89, 203, 269
COMMAND\newcommandx	28, 29
COMMAND\newhookarg@specific	259
COMMAND\newhookcommand@series	253, 254, 431
COMMAND\newhookcommand@series@reload	254
COMMAND\newhookcommand@toggle@reload	254, 429
COMMAND\newhooktoggle@series	254, 431
COMMAND\newhooktoggle@specific	259
COMMAND\newif	431
COMMAND\newline	38
COMMAND\newlinechar	247
COMMAND\newseries	36, 364, 367, 368
COMMAND\newseries@	238, 239, 252
COMMAND\newverse	54, 55, 367, 428, 439
COMMAND\next	306
COMMAND\next@action	105

COMMAND\next@actionline	105
COMMAND\next@insert	163
COMMAND\next@line@list@stuff	120, 440
COMMAND\next@page@num	90
COMMAND\nl@regR	107
COMMAND\no@expands	72, 128, 133, 246, 422
COMMAND\noXlemmaseparator	369
COMMAND\nobreak	195
COMMAND\nocritical	239
COMMAND\noledsec	69, 367
COMMAND\noendnotes	367
COMMAND\noexpand	365
COMMAND\nofamiliar	256
COMMAND\noindent	18, 19, 53, 54, 147, 434
COMMAND\noindentX	434
COMMAND\nomk@	259
COMMAND\nonbreakableafternumber	369
COMMAND\nonumberinfootnote	369
COMMAND\norelax	51
COMMAND\normal@footnotemarkX	207
COMMAND\normal@page@break	357
COMMAND\normal@pars	222
COMMAND\normalbfnoteX	424
COMMAND\normalbodyfootmarkX	207
COMMAND\normalfont	438
COMMAND\normalfootfmt	51, 174, 181, 191, 228, 437
COMMAND\normalfootfmtX	208
COMMAND\normalfootfootmarkX	208
COMMAND\normalfootgroup	176
COMMAND\normalfootgroupX	209
COMMAND\normalfootnoterule	171
COMMAND\normalfootstart	175, 178
COMMAND\normalfootstartX	209
COMMAND\normalvfootnote	172, 174
COMMAND\normalvfootnote@inserted	173, 174
COMMAND\normalvfootnoteX	207
COMMAND\notbool	346
COMMAND\notfontsetup	368
COMMAND\notfontsizeX	45, 368
COMMAND\notenumfont	368
COMMAND\notenumfontX	45, 368
COMMAND\notesXwidthliketwocolumns	369
COMMAND\noteswidthliketwocolumnsX	50, 369, 429, 431
COMMAND\num@lines	142, 164
COMMAND\numberlinefalse	20
COMMAND\numberlinetrue	20
COMMAND\numberonlyfirstinline	251, 369
COMMAND\numberonlyfirstintwolines	369
COMMAND\numberpstartfalse	19
COMMAND\numberpstarttrue	19, 41, 366, 425, 433, 439

COMMAND\numberstanza	41
COMMAND\numberstanzafalse	54
COMMAND\numberstanzatrue	54
COMMAND\numlabfont	22, 51, 98
COMMAND\one@line	142
COMMAND\onehalfspacing	434
COMMAND\onlyXpstart	369
COMMAND\onlysideX	245
COMMAND\page@action	101, 111
COMMAND\page@start	101, 424
COMMAND\pagecontents	101
COMMAND\pageparbreak	367
COMMAND\pageref	58, 272
COMMAND\par	27, 38, 147, 222
COMMAND\par@line	142, 164
COMMAND\para@footgroup	178
COMMAND\para@footgroupX	220
COMMAND\para@footsetup	178, 422
COMMAND\para@footsetupX	218, 422, 429
COMMAND\para@vfootnoteX	219
COMMAND\parafootfmt	180, 181, 437
COMMAND\parafootfmtX	220
COMMAND\parafootftm	183
COMMAND\parafootftmX	221
COMMAND\parafootftmsep	368
COMMAND\parafootsep	369, 427, 432
COMMAND\parafootsepX	48, 103, 368, 369, 437
COMMAND\parafootstart	178
COMMAND\parafootstartX	218
COMMAND\paravfootnote	179, 182
COMMAND\parfillskip	181
COMMAND\parindent	434
COMMAND\parindentX	46, 371, 434, 436
COMMAND\parshape	72
COMMAND\parskip	19, 54, 147
COMMAND\pause@numbering	90
COMMAND\pausenumbering	20, 90, 104, 106, 148, 427, 429, 436
COMMAND\pausenumbering@page@num	90
COMMAND\penalty	181
COMMAND\pend	2, 7, 17–21, 23, 69, 124, 126, 129, 135, 142–148, 161, 162, 365, 427, 428, 437–439
COMMAND\preXnotes	431, 437
COMMAND\prenotesX	49, 227, 431
COMMAND\prepare@Xgroupbyline	201
COMMAND\prepare@Xprenotes	225
COMMAND\prev@nopb	358
COMMAND\prev@pb	358
COMMAND\prevlineX	103
COMMAND\prevpageX@num	103
COMMAND\print@Xfootnoterule	431
COMMAND\print@Xnotes	261, 262

COMMAND\print@Xnotes@forpages	430
COMMAND\print@eledsection	151
COMMAND\print@footnoteXrule	431
COMMAND\print@leftmargin@eledsection	346
COMMAND\print@lemma	192
COMMAND\print@line	149
COMMAND\print@notesX@forpages	430
COMMAND\print@rightmargin@eledsection	346
COMMAND\printendlines	233, 238, 277, 371, 422, 424
COMMAND\printlinefootnote	193, 194, 430
COMMAND\printlinefootnotearea	194, 195, 430
COMMAND\printlinefootnotenumbers	193
COMMAND\printlines	174, 191, 196, 197, 233, 277, 371, 422, 424, 430, 435
COMMAND\printnpnum	366, 371
COMMAND\printpstart	191
COMMAND\protect	128, 365
COMMAND\providecommand	203, 422
COMMAND\pstart	2, 7, 17–21, 23, 68, 69, 111, 124, 129, 135, 142, 143, 145–147, 151, 162, 351, 365, 424–426, 428, 429, 431–433, 436–440
COMMAND\pstartinfootnote	369
COMMAND\pstartinfootnoteeverytime	369
COMMAND\pstartnum	161
COMMAND\pstartref	56, 266, 273, 427, 439
COMMAND\pstarts	425
COMMAND\raggedX	48
COMMAND\raggedleft	47
COMMAND\raggedright	47
COMMAND\raw@text	142
COMMAND\rbracket	44
COMMAND\read@linelist	104–106
COMMAND\ref	58, 60, 64
COMMAND\refformated@	277
COMMAND\refformatedwithpage	277
COMMAND\relax	19, 111, 132, 154, 162, 310, 327, 365
COMMAND\renewcommand	72, 368, 371
COMMAND\reset@msd@options@iffullpage	321
COMMAND\resetprevline@	103
COMMAND\resetprevpage@	103
COMMAND\resumenumbering	20, 86, 90, 91, 104, 106, 148, 424, 428, 429, 436
COMMAND\right	67
COMMAND\rightctab	339
COMMAND\rightlinenum	22, 98, 422, 424
COMMAND\rightltab	339
COMMAND\rightnoteupfalse	61
COMMAND\rightrtab	339
COMMAND\rightsidenote	285
COMMAND\rightright	175, 178, 179, 181
COMMAND\rightstartnum	161
COMMAND\rigidbalance	184, 185, 187, 371, 435
COMMAND\rigidbalanceX	184, 371, 435

COMMAND\robustify	39
COMMAND\roman	332, 435
COMMAND\rtab	338–340, 343
COMMAND\rtabtext	340, 344
COMMAND\sameword	15, 29–32, 118, 135–138, 140, 430, 432, 434, 437, 440
COMMAND\sameword@inedtext	136, 137
COMMAND\saweword	136
COMMAND\scriptsize	98
COMMAND\section	68, 424, 438
COMMAND\section@num	86
COMMAND\sectionmark	345
COMMAND\select@lemmafont	51, 168
COMMAND\series	238, 239
COMMAND\series@	238
COMMAND\seriesatbegin	36, 250, 431
COMMAND\seriesatend	36, 251, 431
COMMAND\set@Xtxtbeforenotes	166
COMMAND\set@continuousnumberingforl	91
COMMAND\set@line	133
COMMAND\set@line@action	102, 111
COMMAND\set@txtbeforenotesX	166
COMMAND\setSErefonlypageprefixmore	59, 277, 436, 439
COMMAND\setSErefonlypageprefixsingle	59, 277, 436, 439
COMMAND\setSErefprefixmore	59
COMMAND\setSErefprefixsingle	59
COMMAND\setapprefprefixmore	59, 368
COMMAND\setapprefprefixsingle	59, 368, 435
COMMAND\setcommand@series	252
COMMAND\sethangingsymbol	53, 305, 368, 434
COMMAND\sethanginsymbol	52
COMMAND\setistwofollowinglines	199
COMMAND\setl@dlprbox	287
COMMAND\setline	23, 102, 107, 110, 124, 128, 145, 433
COMMAND\setlinenum	23, 106, 111, 124, 422, 440
COMMAND\setmsdatalabel	35
COMMAND\setmsdataposition	439
COMMAND\setmsdataseries	35
COMMAND\setprintendlines	233, 235, 424
COMMAND\setprintlines	197, 199, 233, 424
COMMAND\setsidenotesep	62
COMMAND\setsidenotsep	368
COMMAND\setstanzaindent	307
COMMAND\setstanzaindents	52, 307, 365
COMMAND\setstanzapenalties	307
COMMAND\setstanzavalues	307
COMMAND\settoggle@series	252, 425, 429
COMMAND\showlemma	127, 422, 423
COMMAND\showwordrank	32, 137
COMMAND\sidenote@margin	423
COMMAND\sidenotemargin	61, 423, 428

COMMAND\sidenotesep	368
COMMAND\sidepstartnumtrue	19
COMMAND\skip	175
COMMAND\skipnumbering	23, 114, 125, 424, 431, 432
COMMAND\skipnumbering@reg	431
COMMAND\small	45
COMMAND\somemacro	33
COMMAND\special	13
COMMAND\splitmaxdepth	170, 186
COMMAND\splitoff	184
COMMAND\splittopskip	170, 186, 187
COMMAND\stanza	22, 23, 53–55, 310, 368, 370, 434, 439
COMMAND\stanza@hang	309
COMMAND\stanza@line	309
COMMAND\stanzaindent	52, 307, 430
COMMAND\stanzaindent*	52
COMMAND\stanzaindentbase	306
COMMAND\stanzanumwrapper	55
COMMAND\startlock	22, 102, 125, 310
COMMAND\startstanzahook	368
COMMAND\startsub	22, 102, 123
COMMAND\stopmsd	315
COMMAND\stopmsdata	34, 313, 314
COMMAND\strip@pt	178
COMMAND\strutbox	186
COMMAND\sub@action	102, 112
COMMAND\sub@lock	100
COMMAND\sub@off	109, 269
COMMAND\sub@on	109, 269
COMMAND\subline@num	99–101
COMMAND\sublinenum@rep	422
COMMAND\sublinenumberstyle	23, 98, 422
COMMAND\sublinenumincrement	21
COMMAND\sublinenumr@p	97, 422, 424
COMMAND\sublinenumrep	97, 98, 424
COMMAND\sublineref	56, 266, 272
COMMAND\subsectionmark	345
COMMAND\sw@inthisedtext	130
COMMAND\sw@list@inedtext	133, 141
COMMAND\sw@txt	137
COMMAND\swnoexpands	32, 440
COMMAND\symlinenum	369
COMMAND\symplinenum	368
COMMAND\sza@penalty	309
COMMAND>tag	430
COMMAND>text	363
COMMAND>text⟨ <i>language</i> ⟩	46
COMMAND>textbeforenotesX	166
COMMAND>textcolor	73
COMMAND>textheight	72

COMMAND\the	422
COMMAND\thefootnoteA	35
COMMAND\thefootnoteX	426
COMMAND\thelabidx	301
COMMAND\thepage	107
COMMAND\thepstart	19, 372
COMMAND\thepstartL	425
COMMAND\thepstartR	425
COMMAND\thestanza	54
COMMAND\this@line@list@version	119
COMMAND\thisfootnote	211
COMMAND\threecol@begin@insert	187
COMMAND\threecolfootfmt	186, 187, 437
COMMAND\threecolfootfmtX	216
COMMAND\threecolfootgroup	185
COMMAND\threecolfootgroupX	216
COMMAND\threecolfootsetup	185
COMMAND\threecolfootsetupX	215
COMMAND\threecolvfootnote	186
COMMAND\threecolvfootnote@inserted	186
COMMAND\threecolvfootnoteX	216
COMMAND\toendnotes	26, 231, 437
COMMAND\twocolfootfmt	437
COMMAND\twocolfootfmtX	214
COMMAND\twocolfootgroupX	214
COMMAND\twocolfootsetupX	213
COMMAND\twocolvfootnoteX	214
COMMAND\twolines	251, 369
COMMAND\twolines@A	251
COMMAND\twolines@B	252
COMMAND\twolines@C	252
COMMAND\twolinesbutnotmore	369
COMMAND\twolinesonlyinsamepage	369
COMMAND\txtbeforeXnotes	369
COMMAND\txtbeforenotesX	48, 440
COMMAND\txtbeforenotesonlyonceX	48, 440
COMMAND\uline	36
COMMAND\unhbox	179
COMMAND\unpenalty	180–182
COMMAND\unskip	181
COMMAND\unvxh	181, 370
COMMAND\unvxhX	370
COMMAND\upbracefill	338
COMMAND\usingcritext	364, 367
COMMAND\usingedtext	364, 367
COMMAND\vAfootnote	170
COMMAND\vadjust	122
COMMAND\variant	28
COMMAND\vbox	145, 147, 179, 184, 224
COMMAND\vfootnote	170, 175, 179, 186

COMMAND\vl@dbfnote	204, 423
COMMAND\vl@disnote	285
COMMAND\vl@dlsnote	285
COMMAND\vl@dosnote	285
COMMAND\vl@drsnote	285
COMMAND\vnumfootnoteX	424
COMMAND\vsizer	49, 72
COMMAND\vsplit	164
COMMAND\waklam	338
COMMAND\waklamec	338
COMMAND\wapunktel	338
COMMAND\wastricht	338
COMMAND\widthX	50, 371, 436
COMMAND\wrap@edcrossref	271, 429
COMMAND\wrapcontentX	46, 437
COMMAND\wrapped@bodyfootmarkX	222
COMMAND\wrapped@footfootmarkX	221
COMMAND\x...	57
COMMAND\xabslineref	272
COMMAND\xdef	92, 310
COMMAND\xflagref	57, 273, 371, 435
COMMAND\xleft@appenditem	92, 127
COMMAND\xlineref	57, 371, 435
COMMAND\xpageref	57
COMMAND\xparse	33
COMMAND\xpstartref	57, 427
COMMAND\xr	60
COMMAND\xright@appenditem	92
COMMAND\xsublineref	57
COMMAND\xxref	57, 274, 280, 427, 430, 437
COMMAND\zz@@@	422
ENVIRONMENTastanza	435
ENVIRONMENTedarrayc	343
ENVIRONMENTedarrayl	343
ENVIRONMENTedarrayr	343
ENVIRONMENTedtabularc	344
ENVIRONMENTedtabularl	344
ENVIRONMENTedtabularr	344
ENVIRONMENTledgroup	78, 292, 371, 435
ENVIRONMENTledgroupsize	293
PACKAGE(r)(e)ledmac	37
PACKAGEEledmac	12, 74, 101, 298, 366, 367, 430, 431
PACKAGEEledpar	431
PACKAGEEtoolbox	77
PACKAGEParallel	373
PACKAGEReledmac	370, 371
PACKAGEamsgen	322
PACKAGEamsmath	322
PACKAGEbabel	46, 73, 332, 435
PACKAGEbiblatex	71

PACKAGEbidi	46, 47, 78, 434
PACKAGEccaption	85
PACKAGEcolor	73
PACKAGEcsquotes	439
PACKAGEedmac	1, 6, 10, 12–14, 74, 196, 203, 266, 307, 363, 373, 422
PACKAGEedstanza	1, 14, 305
PACKAGEeledmac	1, 10, 11, 14–16, 63, 203, 294, 298, 325, 347, 360, 364, 366–368, 426, 428, 430
PACKAGEeledpar	86, 170, 345, 373, 424, 428–430
PACKAGEetex	434
PACKAGEetoolbox	91, 135, 238, 251, 260, 285, 346, 357
PACKAGEfancyhdr	263, 438
PACKAGEfloatrow	71, 371
PACKAGEfootmisc	35, 73, 78, 203, 373, 438
PACKAGEgeometry	15
PACKAGEglossaries	64, 303, 435
PACKAGEhandout	429
PACKAGEhyperlink	246
PACKAGEhyperref	57, 129, 221, 222, 267, 300, 350, 359, 427–429, 436, 439
PACKAGEifluatex	77
PACKAGEifxetex	77
PACKAGEimakeidx	62, 71, 78, 294, 298, 367, 426–428, 430
PACKAGEindextols	302
PACKAGEindextool	367
PACKAGEindextools	62, 71, 78, 86, 294, 298, 302, 367, 430, 435, 438
PACKAGEinputenc	137
PACKAGEledarab	73
PACKAGEledmac	1, 10, 13, 14, 73, 91, 298, 363, 364, 367, 370
PACKAGEledpar	73
PACKAGEMemoir	78, 298, 367, 373, 429
PACKAGEMorewrites	71
PACKAGEMusixtex	428
PACKAGEperpage	435
PACKAGEpolyglossia	44, 73, 169, 191, 435
PACKAGERagged2e	47, 77
PACKAGEReledmac	1, 2, 11, 12, 14–17, 20, 21, 24, 25, 28–30, 33, 35, 36, 39, 40, 43, 46, 47, 49, 51, 53, 55, 57–65, 69, 71–75, 93, 95, 101, 102, 105, 106, 109, 118–120, 128, 131, 132, 135, 162, 171, 175, 179, 191, 203, 228, 239, 243, 244, 252, 260, 271, 272, 275, 280, 298, 318, 325, 345, 346, 359, 360, 367, 368, 370–372, 433, 436, 438
PACKAGEReledpar	1, 4, 6, 8, 16, 19, 36, 50, 57, 59, 60, 69–71, 73, 75, 86, 93, 104, 109, 120, 130, 132, 172, 175, 223, 224, 239, 245, 259, 261, 262, 294, 305, 434, 435, 437, 438, 440
PACKAGESuffix	77
PACKAGETabmac	1, 13, 14, 373
PACKAGEulem	36
PACKAGEuninormalize	29
PACKAGExargs	29, 77
PACKAGExkeyval	75, 259
PACKAGExparse	77
PACKAGEXr	5, 60, 61, 280, 281, 436
PACKAGEXref	280
PACKAGEXstring	77, 300

A

<code>\absline@num</code>	1
Abu Kamil Shuja' b. Aslam	13
<code>\actionlines@list</code>	1
<code>\actions@list</code>	1
<code>\add@inserts</code>	1
<code>\add@inserts@next</code>	1
<code>\add@msd@</code>	1
<code>\add@msdata</code>	1
<code>\add@msdata@firstlineofpage</code>	1
<code>\add@penalties</code>	1
<code>\add@Xgroupbyline</code>	1
<code>\addtol@denbody</code>	1
Adelard II	13
<code>\advancelabel@refs</code>	1
<code>\advanceline</code>	1, 22
<code>\Aendnote</code>	25
<code>\affixline@num</code>	1
<code>\affixpstart@num</code>	1
<code>\affixside@note</code>	1
<code>\Afootnote</code>	25
<code>\afternoteX</code>	47
<code>\afterruleX</code>	49
<code>\ampersand</code>	1, 55
<code>\append@notesX</code>	1
<code>\append@Xnotes</code>	1
<code>\applabel</code>	1, 59
<code>\appref</code>	1, 59
<code>\apprefwithpage</code>	1, 59
<code>\arrangementX</code>	1, 38
<code>\arrangementX@normal</code>	1
<code>\arrangementX@threecol</code>	1
<code>\arrangementX@twocol</code>	1
<code>\at@every@pend</code>	1
<code>\AtEndEveryPend</code>	1, 19
<code>\AtEveryPend</code>	1, 19
<code>\AtEveryPstart</code>	1, 19
<code>\AtEveryStanza</code>	1, 54
<code>\AtEveryStopStanza</code>	1, 54
<code>\AtStartEveryPstart</code>	1, 19
<code>\AtStartEveryStanza</code>	1, 54
<code>\autopar</code>	1, 18

B

<code>\ballast</code>	72
<code>\ballast@count</code>	1
Beeton, Barbara Ann Neuhaus Friend	18
<code>\beforeeledchapter</code>	1
<code>\BeforeEveryStopStanza</code>	1
<code>\beforeinsertingX</code>	46

<code>\beforeinsertion@X</code>	1
<code>\beforenotesX</code>	49
<code>\beginnumbering</code>	1, 16
<code>\Bendnote</code>	25
<code>\Bfootnote</code>	25
<code>\bhookgroupX</code>	48
<code>\bhooknoteX</code>	46
<code>\bodyfootmarkA</code>	35
<code>\boxfootnotenumbers</code>	1
Bredon, Simon	13
Breger, Herbert	13, 325
Brey, Gerhard	13
Busard, Hubert L. L.	13
<code>\bypage@false</code>	1
<code>\bypage@true</code>	1
<code>\bypstart@false</code>	1
<code>\bypstart@true</code>	1

C

<code>\c@addcolcount</code>	1
<code>\c@ballast</code>	1
<code>\c@firstlinenum</code>	1
<code>\c@firstsublinenum</code>	1
<code>\c@labidx</code>	1
<code>\c@linenumincrement</code>	1
<code>\c@sublinenumincrement</code>	1
<code>\Cendnote</code>	25
<code>\Cfootnote</code>	25
<code>\ch@ck@l@ck</code>	1
<code>\ch@cksub@l@ck</code>	1
<code>\chapter</code>	1
<code>\check@pb@in@verse</code>	1
Chester, Robert of	13
Claassens, Geert H. M.	13
<code>\colalignX</code>	47
Copernicus, Nicolaus	13
<code>\critext</code>	363
<code>\ctab</code>	1
<code>\ctabtext</code>	1

D

Dekker, Dirk-Jan	73
<code>\Dendnote</code>	25
<code>\Dfootnote</code>	25
<code>\disable@familiarnotes</code>	1
<code>\disable@notes</code>	1
<code>\disable@sidenotes</code>	1
<code>\disablel@dtabfeet</code>	1
<code>\do@actions</code>	1
<code>\do@actions@fixedcode</code>	1

<code>\do@actions@next</code>	1
<code>\do@ballast</code>	1
<code>\do@feet@custom@order</code>	1
<code>\do@feetX</code>	1
<code>\do@insidelinehook</code>	1
<code>\do@line</code>	1
<code>\do@linehook</code>	1
<code>\do@lockoff</code>	1
<code>\do@lockoffL</code>	1
<code>\do@lockon</code>	1
<code>\do@lockonL</code>	1
<code>\do@Xfeet</code>	1
<code>\doedindexlabel</code>	1
<code>\doendnotes</code>	1, 26
<code>\doendnotesbysection</code>	1, 26
<code>\doinsidelinehook</code>	1, 24
<code>\dolinehook</code>	1, 24
<code>\dosplits</code>	1
Downes, Michael	72, 179, 181
<code>\doextrafeet</code>	1
<code>\dummy@edindex</code>	1
<code>\dummy@edtext</code>	1
<code>\dummy@edtext@showlemma</code>	1
<code>\dummy@ref</code>	1

E

<code>\edaftertab</code>	1, 67, 338
<code>edarrayc</code> (environment)	65
<code>edarrayl</code> (environment)	65
<code>edarrayr</code> (environment)	65
<code>\edatleft</code>	1, 67
<code>\edatright</code>	1, 67
<code>\edbeforetab</code>	1, 67, 338
<code>\edfilldimen</code>	1
<code>\edfont@info</code>	1
<code>\edindex</code>	1, 62
<code>\edindexlab</code>	1, 64
<code>\EDLABEL</code>	1
<code>\edlabel</code>	1, 56
<code>\edlabelE</code>	1, 58
<code>\edlabelS</code>	1, 58
<code>\edlabelSE</code>	1, 58
<code>\edlineref</code>	1, 56
<code>\edmakelabel</code>	1, 58
<code>\edpageref</code>	1, 56
<code>\edrowfill</code>	1, 66
<code>\EDTAB</code>	1
<code>\edtabcolsep</code>	1, 65
<code>\EDTABINDENT</code>	1
<code>\edtabindent</code>	1

<code>\EDTABtext</code>	1
<code>edtabularc</code> (environment)	65
<code>edtabularl</code> (environment)	65
<code>edtabularr</code> (environment)	65
<code>\EDTEXT</code>	1
<code>\edtext</code>	1, 24
<code>\edvertdots</code>	1, 67
<code>\edvertline</code>	1, 67
<code>\Eendnote</code>	25
<code>\Efootnote</code>	25
<code>\eled@chapter</code>	1
<code>\eled@section</code>	1
<code>\eled@sectioning@out</code>	1
<code>\eled@subsection</code>	1
<code>\eled@subsubsection</code>	1
<code>\eledchapter</code>	1
<code>\eledchapter*</code>	1
<code>\eledsection</code>	1
<code>\eledsection*</code>	1
<code>\eledsubsection</code>	1
<code>\eledsubsection*</code>	1
<code>\eledsubsubsection</code>	1
<code>\eledsubsubsection*</code>	1
<code>\enablel@dtabfeet</code>	1
<code>\end@lemmas</code>	1
<code>\endashchar</code>	1
<code>\endline@num</code>	1
<code>\endlock</code>	1, 22
<code>\endminipage</code>	1
<code>\endnumbering</code>	1, 16
<code>\endpage@num</code>	1
<code>\endprint</code>	1
<code>\endquotation</code>	1
<code>\endquote</code>	1
<code>\endsub</code>	1, 22
<code>\endsubline@num</code>	1
environments:	
<code>edarrayc</code>	65
<code>edarrayl</code>	65
<code>edarrayr</code>	65
<code>edtabularc</code>	65
<code>edtabularl</code>	65
<code>edtabularr</code>	65
<code>ledgroup</code>	55
<code>ledgroupsize</code>	55
<code>minipage</code>	55
<code>Euclid</code>	13
<code>\extensionchars</code>	1, 70

F

\f@x@l@cks	1
Fairbairns, Robin	35
\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, 55
\flush@notes	1
\fnpos	1, 37
Folkerts, Menso	13
\footfootmarkA	35
\footfudgefiddle	1, 72
\footnote	1
\footnoteA	35
\footnoteB	35
\footnoteC	35
\footnoteD	35
\footnoteE	35
\footnotelang@lua	1
\footnotelang@poly	1
\footnoteoptions@	1
\footnoteXmark	36
\footnoteXtext	36
\footplitskips	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	46
\hidenumbering	1, 23
\hidenumberingonleftpage	1, 23
\hidenumberingonrightpage	1
\Hilfsbox	1
\hilfsbox	1
\hilfscount	1
\HILFSskip	1
\Hilfsskip	1
\hilfsskip	1
\hsizethreecolX	47
\hsizetwocolX	47
\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

<code>\ifl@d@Xtwolines</code>	1
<code>\ifl@dend@X</code>	1
<code>\ifl@dhidenumber</code>	1
<code>\ifl@dmemoir</code>	1
<code>\ifl@dpaging</code>	1
<code>\ifl@dpairing</code>	1
<code>\ifl@dprintingcolumns</code>	1
<code>\ifl@dprintingpages</code>	1
<code>\ifl@dskipnumber</code>	1
<code>\ifl@dskipversenumber</code>	1
<code>\ifl@dstartendok</code>	1
<code>\ifl@footmisc</code>	1
<code>\ifl@imakeidx</code>	1
<code>\ifl@indextools</code>	1
<code>\ifledfinal</code>	1, 70
<code>\ifledgroupnotesL@</code>	1
<code>\ifledgroupnotesR@</code>	1
<code>\iflednopbinverse</code>	1
<code>\ifledRcol</code>	1
<code>\ifledRcol@</code>	1
<code>\ifnocritical@</code>	1
<code>\ifnoend@</code>	1
<code>\ifnofamiliar@</code>	1
<code>\ifnoledgroup@</code>	1
<code>\ifnoquotation@</code>	1
<code>\ifnoteschanged@</code>	1
<code>\ifnumberedpar@</code>	1
<code>\ifnumbering</code>	1
<code>\ifnumberingR</code>	1
<code>\ifnumberline</code>	1
<code>\ifnumberstanza</code>	1
<code>\ifparapparatus@</code>	1
<code>\ifparledgroup</code>	1
<code>\ifpst@rtedL</code>	1
<code>\ifresumenumbering@start</code>	1
<code>\ifseriesbefore</code>	1
<code>\ifsidepstartnum</code>	1
<code>\ifstopmsdata@inserted@</code>	1
<code>\ifsublines@</code>	1
<code>\ifwidthliketwocolumns</code>	1
<code>\ifXendinsertsep@</code>	1
<code>\ifxindy@</code>	1
<code>\ifxindyhyperref@</code>	1
<code>\initnumbering@quote</code>	1
<code>\initnumbering@reg</code>	1
<code>\insert@count</code>	0, 1
<code>\insert@msdata</code>	1
<code>\insert@txtbeforenotesX</code>	1
<code>\insert@Xtxtbeforenotes</code>	1
<code>\inserthangingymbol</code>	1

<code>\insertlines@list</code>	1
<code>\insertparafootsepX</code>	1
<code>\inserts@list</code>	1

J

Jayaditya	13
-----------------	----

K

Kabelschacht, Alois	166
---------------------------	-----

L

<code>\l@advance@parledgroup@beforenormalnotes</code>	1
<code>\l@d@add</code>	1
<code>\l@d@nums</code>	1
<code>\l@d@section</code>	1
<code>\l@d@set</code>	1
<code>\l@d@Xend</code>	1
<code>\l@dampcount</code>	1
<code>\l@dbfnote</code>	1
<code>\l@dcheckcols</code>	1
<code>\l@dcheckstartend</code>	1
<code>\l@dchset@num</code>	1
<code>\l@dcolcount</code>	1
<code>\l@dcollect@body</code>	1
<code>\l@dcollect@body</code>	1
<code>\l@dcolwidth</code>	1
<code>\l@dcsnote</code>	1
<code>\l@dcsnotetext</code>	1
<code>\l@dcsnotetext@l</code>	1
<code>\l@dcsnotetext@r</code>	1
<code>\l@ddodoreinextrafeet</code>	1
<code>\l@dedbeginmini</code>	1
<code>\l@dedendmini</code>	1
<code>\l@demptyd@ta</code>	1
<code>\l@dend@close</code>	1
<code>\l@dend@open</code>	1
<code>\l@dend@stuff</code>	1
<code>\l@dend@Xfalse</code>	1
<code>\l@dend@Xtrue</code>	1
<code>\l@denvbody</code>	1
<code>\l@desnote</code>	1
<code>\l@dfambeginmini</code>	1
<code>\l@dfamendmini</code>	1
<code>\l@dfeetbeginmini</code>	1
<code>\l@dfeetendmini</code>	1
<code>\l@dgetline@margin</code>	1
<code>\l@dgetlock@disp</code>	1
<code>\l@dgetref@num</code>	1
<code>\l@dgetsidenote@margin</code>	1
<code>\l@dobblearg</code>	1

<code>\l@disnote</code>	1
<code>\l@dlabel@parse</code>	1
<code>\l@dld@ta</code>	1
<code>\l@dlp@rbox</code>	1
<code>\l@dlsn@te</code>	1
<code>\l@dlsnote</code>	1
<code>\l@dmake@labels</code>	1
<code>\l@dmodforedtext</code>	1
<code>\l@dnullfills</code>	1
<code>\l@dnumstartsl</code>	1
<code>\l@doldold@footnotetext</code>	1
<code>\l@dp@rsefootspec</code>	1
<code>\l@dparsedendline</code>	1
<code>\l@dparsedendpage</code>	1
<code>\l@dparsedendsub</code>	1
<code>\l@dparsedstartline</code>	1
<code>\l@dparsedstartpage</code>	1
<code>\l@dparsedstartsub</code>	1
<code>\l@dparsefootspec</code>	1
<code>\l@dpush@begins</code>	1
<code>\l@drd@ta</code>	1
<code>\l@dref@undefined</code>	1
<code>\l@drestorefills</code>	1
<code>\l@drestoreforedtext</code>	1
<code>\l@drp@rbox</code>	1
<code>\l@drsn@te</code>	1
<code>\l@drsnote</code>	1
<code>\l@dsetmaxcolwidth</code>	1
<code>\l@dskipnumberfalse</code>	1
<code>\l@dskipnumbertrue</code>	1
<code>\l@dtabaddcols</code>	1
<code>\l@dtabnoexpands</code>	1
<code>\l@dunboxmpfoot</code>	1
<code>\l@dunhbox@line</code>	1
<code>\l@dzeropenalties</code>	1
<code>\label</code>	58
<code>\labelstartfalse</code>	1
<code>\labelstarttrue</code>	1, 19
<code>\labelref@list</code>	1
<code>\labelrefsparseline</code>	1
<code>\labelrefsparsesubline</code>	1
<code>\last@page@num</code>	1
Lavagnino, John	12
<code>\led@check@nopb</code>	1
<code>\led@check@pb</code>	1
<code>\led@err@AutoparNotNumbered</code>	1
<code>\led@err@BadAction</code>	1
<code>\led@err@edtextoutsidepstart</code>	1
<code>\led@err@EdtextWithoutFootnote</code>	1
<code>\led@err@FootnoteNotInSecondArgEdtext</code>	1

<code>\led@err@HighEndColumn</code>	1
<code>\led@err@LineationInNumbered</code>	1
<code>\led@err@LowStartColumn</code>	1
<code>\led@err@ManyLeftnotes</code>	1
<code>\led@err@ManyRightnotes</code>	1
<code>\led@err@ManySidenotes</code>	1
<code>\led@err@NumberingNotStarted</code>	1
<code>\led@err@NumberingShouldHaveStarted</code>	1
<code>\led@err@NumberingStarted</code>	1
<code>\led@err@NumberingWithoutPstart</code>	1
<code>\led@err@PendNoPstart</code>	1
<code>\led@err@PendNotNumbered</code>	1
<code>\led@err@PstartInPstart</code>	1
<code>\led@err@PstartNotNumbered</code>	1
<code>\led@err@ReverseColumns</code>	1
<code>\led@err@toendnotes@outsidenumbering</code>	1
<code>\led@err@TooManyColumns</code>	1
<code>\led@err@UnequalColumns</code>	1
<code>\led@error@fail@patch@@doclearpage</code>	1
<code>\led@error@fail@patch@@iiiminipage</code>	1
<code>\led@error@fail@patch@@makecol</code>	1
<code>\led@error@fail@patch@@reinserts</code>	1
<code>\led@error@fail@patch@endminipage</code>	1
<code>\led@error@PackageAfterEledmac</code>	1
<code>\led@mess@NotesChanged</code>	1
<code>\led@mess@SectionContinued</code>	1
<code>\led@nopb</code>	1
<code>\led@nopbnum</code>	1
<code>\led@pb</code>	1
<code>\led@pb@setting</code>	1
<code>\led@pbnum</code>	1
<code>\led@reinit@index@fornote</code>	1
<code>\led@set@index@fornote</code>	1
<code>\led@toksa</code>	1
<code>\led@toksb</code>	1
<code>\led@warn@AppLabelOutSecondArgEdtext</code>	1
<code>\led@warn@BadAction</code>	1
<code>\led@warn@BadAdvancelineLine</code>	1
<code>\led@warn@BadAdvancelineSubline</code>	1
<code>\led@warn@BadLineation</code>	1
<code>\led@warn@BadLinenummargin</code>	1
<code>\led@warn@BadLockdisp</code>	1
<code>\led@warn@BadSetline</code>	1
<code>\led@warn@BadSetlinenum</code>	1
<code>\led@warn@BadSidenotemargin</code>	1
<code>\led@warn@BadSubblockdisp</code>	1
<code>\led@warn@DuplicateLabel</code>	1
<code>\led@warn@edinde@outsidenumbering</code>	1
<code>\led@warn@LineFileObsolete</code>	1
<code>\led@warn@NoFile</code>	1

<code>\led@warn@NoIndexFile</code>	1
<code>\led@warn@NoMarginpars</code>	1
<code>\led@warn@RefUndefined</code>	1
<code>\led@warn@SeriesStillExist</code>	1
<code>\led@warning@hsizeX@deprecated</code>	1
<code>\led@warning@msdatawithoutstop</code>	1
<code>\led@warning@preXnotes@deprecated</code>	1
<code>\led@warning@Xhsize@deprecated</code>	1
<code>ledgroup (environment)</code>	55
<code>ledgroupsize (environment)</code>	55
<code>\ledinnernote</code>	1, 61
<code>\ledinnote</code>	1
<code>\ledinnotehyperpage</code>	1
<code>\ledinnotemark</code>	1
<code>\ledleftnote</code>	1, 61
<code>\ledlinenum</code>	1
<code>\ledllfill</code>	1
<code>\ledlsnotefontsetup</code>	1, 62
<code>\ledlsnotesep</code>	1, 61
<code>\ledlsnotewidth</code>	1, 61
<code>\lednopb</code>	1, 69
<code>\lednopbinversetrue</code>	70
<code>\lednopbnum</code>	1
<code>\ledouternote</code>	1, 61
<code>\ledpb</code>	1, 69
<code>\ledpbnum</code>	1
<code>\ledpbsetting</code>	1, 70
<code>\ledrightnote</code>	1, 61
<code>\ledrlfill</code>	1
<code>\ledrsnotefontsetup</code>	1, 62
<code>\ledrsnotesep</code>	1, 61
<code>\ledrsnotewidth</code>	1, 61
<code>\ledsectnomark</code>	1
<code>\ledsectnotoc</code>	1
<code>\ledsetnormalparstuff@common</code>	1
<code>\ledsetnormalparstuffX</code>	1
<code>\ledsidenote</code>	1, 61
<code>\leftctab</code>	1
<code>\leftlinenum</code>	1, 22
<code>\leftltab</code>	1
<code>\leftnoteupfalse</code>	61
<code>\leftpstartnum</code>	1
<code>\leftrtab</code>	1
<code>Leibniz</code>	13
<code>\lemma</code>	1, 27
<code>\letsforverteilen</code>	1
<code>\line@list</code>	1
<code>\line@list@stuff</code>	1
<code>\line@list@version</code>	1
<code>\line@margin</code>	1

<code>\line@num</code>	1
<code>\line@set</code>	1
<code>\lineation</code>	1, 21
<code>\linenum</code>	1, 27
<code>\linenum@out</code>	1
<code>\linenumberlist</code>	1, 21
<code>\linenumberstyle</code>	1, 23
<code>\linenumincrement</code>	1, 21
<code>\linenummargin</code>	1, 21
<code>\linenumr@p</code>	1
<code>\linenumrep</code>	1
<code>\linenumsep</code>	1, 22
<code>\linerangesep@</code>	1
<code>\list@clear</code>	1
<code>\list@clearing@reg</code>	1
<code>\list@create</code>	1
<code>\lock@disp</code>	1
<code>\lock@off</code>	1
<code>\lock@on</code>	1
<code>\lockdisp</code>	1, 22
Lorch, Richard	13
<code>\ltab</code>	1
<code>\ltabtext</code>	1
Luecking, Dan	76

M

<code>\m@mmf@check</code>	1
<code>\m@mmf@prepare</code>	1
<code>\M@sect</code>	1
<code>\makehboxofhboxes</code>	1
<code>\managestanza@modulo</code>	1
<code>\maxhnotesX</code>	49
Mayer, Gyula	13
<code>\measurembody</code>	1
<code>\measuremcell</code>	1
<code>\measuremrow</code>	1
<code>\measuretbody</code>	1
<code>\measuretcell</code>	1
<code>\measuretrow</code>	1
Middleton, Thomas	13, 100
minipage (environment)	55
Mittelbach, Frank	12, 13
<code>\morenoexpands</code>	1, 72
<code>\mp@append@notesX</code>	1
<code>\mp@append@Xnotes</code>	1
<code>\mpfnpos</code>	1, 37
<code>\mpnormalfootgroup</code>	1
<code>\mpnormalfootgroupX</code>	1
<code>\mpnormalvfootnote</code>	1
<code>\mpnormalvfootnote@inserted</code>	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, 34
\msdata@c	1
\msdata@cR	1
\multfootsep	1, 35
\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
\next@line@list@stuff	1
\no@expands	1
\noeledsec	69
\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

<code>\normalvfootnote</code>	1
<code>\normalvfootnote@inserted</code>	1
<code>\normalvfootnoteX</code>	1
<code>\notefontsizeX</code>	45
<code>\notenumfontX</code>	45
<code>\noteschanged@false</code>	1
<code>\noteschanged@true</code>	1
<code>\noteswidthliketwocolumnsX</code>	50
<code>\nulledindex</code>	1
<code>\nullsetzen</code>	1
<code>\num@lines</code>	1
<code>\numberedpar@false</code>	1
<code>\numberedpar@true</code>	1
<code>\numberingfalse</code>	1
<code>\numberingtrue</code>	1
<code>\numberlinefalse</code>	20
<code>\numberlinetrue</code>	20
<code>\numberpstartfalse</code>	1, 19
<code>\numberpstarttrue</code>	1, 19
<code>\numberstanzafalse</code>	54
<code>\numberstanzatrue</code>	54
<code>\numlabfont</code>	1, 50

O

<code>\old@hsize</code>	1
<code>\one@line</code>	1
<code>optionauxdir</code>	15, 437
<code>optioncontinuousnumberingwithcolumns</code>	436
<code>optioninnnote</code>	435
<code>optioninnote</code>	435
<code>optionlinrangesep</code>	259
<code>optionnocritical</code>	435
<code>optionnoeledsec</code>	351, 437
<code>optionnoend</code>	435
<code>optionnopenalties</code>	72
<code>optionnotenumber</code>	435
<code>optionswcaseinsensitive</code>	29, 440

P

<code>\page@action</code>	1
<code>\page@num</code>	1
<code>\pagelinesep</code>	1, 63
<code>\pageref</code>	58
<code>\par@line</code>	1
<code>\para@footgroupX</code>	1
<code>\para@footsetup</code>	1
<code>\para@footsetupX</code>	1
<code>\para@vfootnoteX</code>	1
<code>\parafootfmt</code>	1
<code>\parafootfmtX</code>	1

<code>\parafootgroup</code>	1
<code>\parafootsepX</code>	48
<code>\parafootstart</code>	1
<code>\parafootstartX</code>	1
<code>\paravfootnote</code>	1
<code>\parindentX</code>	46
<code>\pausenumbering</code>	1, 20
<code>\pausenumbering@page@num</code>	1
<code>\pend</code>	1, 17
Plato of Tivoli	13
<code>\postbodyfootmark</code>	1
<code>\prebodyfootmark</code>	1
<code>\prenotesX</code>	49
<code>\prepare@edindex@fornote</code>	1
<code>\prepare@prenotesX</code>	1
<code>\prepare@Xgroupbyline</code>	1
<code>\prepare@Xprenotes</code>	1
<code>\prev@nopb</code>	1
<code>\prev@pb</code>	1
<code>\prevpage@num</code>	1
<code>\print@eledsection</code>	1
<code>\print@footnoteXrule</code>	1
<code>\print@leftmargin@eledsection</code>	1
<code>\print@lemma</code>	1
<code>\print@line</code>	1
<code>\print@notesX</code>	1
<code>\print@rightmargin@eledsection</code>	1
<code>\print@Xfootnoterule</code>	1
<code>\print@Xnotes</code>	1
<code>\printendlines</code>	1
<code>\printlineendnote</code>	1
<code>\printlineendnotearea</code>	1
<code>\printlinefootnote</code>	1
<code>\printlinefootnotearea</code>	1
<code>\printlinefootnotenumbers</code>	1
<code>\printlines</code>	1
<code>\printnpnum</code>	1
<code>\printpstart</code>	1
<code>\printsymlineendnotearea</code>	1
<code>\printsymlinefootnotearea</code>	1
<code>\printXafternumber</code>	1
<code>\printXbeforenumber</code>	1
<code>\pstart</code>	1, 17
<code>\pstarteref</code>	1
<code>\pstartnum</code>	1
<code>\pstartref</code>	56

Q

<code>\quotation</code>	1
<code>\quote</code>	1

R

<code>\raggedX</code>	48
<code>\raw@text</code>	<u>1</u>
<code>\rbracket</code>	<u>1</u>
<code>\read@linelist</code>	<u>1</u>
<code>\ref</code>	58
<code>\ref@reg@later</code>	<u>1</u>
<code>\Relax</code>	<u>1</u>
<code>\reledmac@error</code>	<u>1</u>
<code>\reledmac@warning</code>	<u>1</u>
<code>\removehboxes</code>	<u>1</u>
<code>\reset@msd@options@iffullpage</code>	<u>1</u>
<code>\resetprevline@</code>	<u>1</u> , 103
<code>\resetprevpage@</code>	<u>1</u>
<code>\resetprevpage@num</code>	103
<code>\restore@familiarnotes</code>	<u>1</u>
<code>\restore@notes</code>	<u>1</u>
<code>\restore@sidenotes</code>	<u>1</u>
<code>\resumenumbering</code>	<u>1</u> , 20
<code>\rightctab</code>	<u>1</u>
<code>\rightlinenum</code>	<u>1</u> , 22
<code>\rightltab</code>	<u>1</u>
<code>\rightnoteupfalse</code>	61
<code>\rightrtab</code>	<u>1</u>
<code>\rightstartnum</code>	<u>1</u>
<code>\rigidbalance</code>	<u>1</u>
<code>\rigidbalanceX</code>	<u>1</u>
<code>\rtab</code>	<u>1</u>
<code>\rtabtext</code>	<u>1</u>

S

Sacrobosco	13
<code>\sameword</code>	<u>1</u> , 28
<code>\sameword@inedtext</code>	<u>1</u>
Schöpf, Rainer	13
<code>\section@num</code>	<u>1</u>
<code>\select@@lemmafont</code>	<u>1</u>
<code>\select@lemmafont</code>	<u>1</u> , 51
<code>\SEref</code>	<u>1</u> , 58
<code>\SErefonlypage</code>	58
<code>\SErefwithpage</code>	<u>1</u> , 58
<code>\series</code>	<u>1</u>
<code>\seriesatbegin</code>	<u>1</u> , 36
<code>\seriesatend</code>	<u>1</u> , 36
<code>\set@continuousnumberingforL</code>	<u>1</u>
<code>\set@line</code>	<u>1</u>
<code>\set@line@action</code>	<u>1</u>
<code>\set@txtbeforenotesX</code>	<u>1</u>
<code>\set@Xtxtbeforenotes</code>	<u>1</u>
<code>\setapprefprefixmore</code>	59

<code>\setapprefprefixsingle</code>	59
<code>\setcommand@series</code>	1
<code>\sethangingsymbol</code>	1, 53
<code>\setistwofollowinglines</code>	1
<code>\setl@dlp@rbox</code>	1
<code>\setl@drpr@box</code>	1
<code>\setline</code>	1, 22
<code>\setlinenum</code>	1, 23
<code>\setmcellcenter</code>	1
<code>\setmcellleft</code>	1
<code>\setmcellright</code>	1
<code>\setmrowcenter</code>	1
<code>\setmrowleft</code>	1
<code>\setmrowright</code>	1
<code>\setmsdatalabel</code>	1, 35
<code>\setmsdataposition</code>	1, 35
<code>\setmsdataseries</code>	1, 35
<code>\setnoteswidthliketwocolumnsX@</code>	1
<code>\setnotesXpositionliketwocolumns@</code>	1
<code>\setprintendlines</code>	1
<code>\setprintlines</code>	1
<code>\setSerefonlypageprefixmore</code>	59
<code>\setSerefonlypageprefixsingle</code>	59
<code>\setSerefprefixmore</code>	59
<code>\setSerefprefixsingle</code>	59
<code>\setsidenotesep</code>	62
<code>\setstanzaindents</code>	1, 51
<code>\setstanzapenalties</code>	1, 53
<code>\setstanzavalues</code>	1
<code>\settccllcenter</code>	1
<code>\settccllleft</code>	1
<code>\settccllright</code>	1
<code>\settoggle@series</code>	1
<code>\setthrowcenter</code>	1
<code>\setthrowleft</code>	1
<code>\setthrowright</code>	1
<code>\setXnotespositionliketwocolumns@</code>	1
<code>\setXnoteswidthliketwocolumns@</code>	1
<code>\showlemma</code>	1, 70
<code>\showwordrank</code>	1, 32
<code>\sidenote@margin</code>	1
<code>\sidenotemargin</code>	1, 61
<code>\sidepstartnumtrue</code>	19
<code>\skip@lockoff</code>	1
<code>\skipnumbering</code>	1, 23
<code>\splitoff</code>	1
<code>\spreadmath</code>	1, 66
<code>\spreadtext</code>	1, 66
<code>\stanza</code>	1, 51
<code>\stanza@count</code>	1

<code>\stanza@hang</code>	1
<code>\stanza@line</code>	1
<code>\stanzaindent</code>	1, 52
<code>\stanzaindent*</code>	1, 52
<code>\stanzaindentbase</code>	1, 51
<code>\stanzanumwrapper</code>	1, 55
<code>\startlock</code>	1, 22
<code>\startsub</code>	1, 22
<code>\step1@dcolcount</code>	1
<code>\stopmsdata</code>	1, 34
<code>\strip@szacnt</code>	1
<code>\sub@action</code>	1
<code>\sub@lock</code>	1
<code>\sub@off</code>	1
<code>\sub@on</code>	1
<code>\subline@num</code>	1
<code>\sublinenumberstyle</code>	1, 23
<code>\sublinenumincrement</code>	1, 21
<code>\sublinenumr@p</code>	1
<code>\sublinenumrep</code>	1
<code>\sublineref</code>	1, 56
<code>\sublines@false</code>	1
<code>\sublines@true</code>	1
<code>\sublock@disp</code>	1
<code>\sublockdisp</code>	1
Sullivan, Wayne	13, 14, 72, 85, 90, 179, 180, 266, 305
<code>\sw@noexpand</code>	1
<code>\swnoexpands</code>	32
<code>\sza@penalty</code>	1

T

<code>\tabHilfbox</code>	1
<code>\tabhilfbox</code>	1
<code>\theadcolcount</code>	1
<code>\theadtext</code>	1
<code>\theendpageline</code>	1
<code>\thefootnoteA</code>	35
Theodosius	13
<code>\thepageline</code>	1
<code>\thepstart</code>	1, 19
<code>\thestanza</code>	1, 54
<code>\thestartpageline</code>	1
<code>\this@line@list@version</code>	1
<code>\threecol@begin@insert</code>	1
<code>\threecolfootfmt</code>	1
<code>\threecolfootfmtX</code>	1
<code>\threecolfootgroup</code>	1
<code>\threecolfootgroupX</code>	1
<code>\threecolfootsetup</code>	1
<code>\threecolfootsetupX</code>	1

<code>\threecolvfootnote</code>	1
<code>\threecolvfootnote@inserted</code>	1
<code>\threecolvfootnoteX</code>	1
<code>\toendnotes</code>	1, 26
<code>\toendnotes*</code>	1
<code>\twocolfootfmt</code>	1
<code>\twocolfootfmtX</code>	1
<code>\twocolfootgroup</code>	1
<code>\twocolfootgroupX</code>	1
<code>\twocolfootsetup</code>	1
<code>\twocolfootsetupX</code>	1
<code>\twocolvfootnote</code>	1
<code>\twocolvfootnote@inserted</code>	1
<code>\twocolvfootnoteX</code>	1
<code>\txtbeforenotesonlyonceX</code>	48
<code>\txtbeforenotesX</code>	48

U

<code>\unvxhX</code>	1
----------------------------	---

V

Vamana	13
<code>\variab</code>	1
<code>\vbfnoteX</code>	1
<code>\vl@dbfnote</code>	1
<code>\vl@dcfnote</code>	1
<code>\vl@disnote</code>	1
<code>\vl@dlsnote</code>	1
<code>\vl@dosnote</code>	1
<code>\vl@drsnote</code>	1
<code>\vnumfootnoteX</code>	1

W

Whitney, Ron	13
<code>\widthX</code>	50
<code>\wrap@edcrossref</code>	1
<code>\wrapcontentX</code>	46
<code>\wrapped@bodyfootmarkX</code>	1
<code>\wrapped@footfootmarkX</code>	1
Wujastyk, Dominik	12

X

<code>\X@atbegininsertion</code>	1
<code>\X@beforeinsertion</code>	1
<code>\X@doreinfeet</code>	1
<code>\xabslineref</code>	1
<code>\Xafterlemmaseparator</code>	44
<code>\Xafternote</code>	47
<code>\Xafternumber</code>	42
<code>\Xafterrule</code>	49

\Xaftersymlinenum	42
\Xarrangement	1, 38
\Xarrangement@normal	1
\Xarrangement@paragraph	1
\Xarrangement@threecol	1
\Xarrangement@twocol	1
\Xbeforeinserting	46
\Xbeforelemmaseparator	44
\Xbeforenotes	49
\Xbeforenumber	40, 42
\Xbeforesymlinenum	42
\Xbhookgroup	48
\Xbhooknote	46
\Xboxlinenum	42
\Xboxlinenumalign	43
\Xboxsymlinenum	42
\Xcolalign	47
\xedindex	1
\xedlabel	1
\xedtext	1
\Xendafterenumber	42
\Xendafterlemmaseparator	44
\Xendafternote	50
\Xendafterpagenumber	43
\Xendaftersymlinenum	42
\Xendahookinplaceofnumber	44
\Xendahooklinenumber	44
\Xendbeforelemmaseparator	44
\Xendbeforenumber	42
\Xendbeforepagenumber	43
\Xendbeforesymlinenum	42
\Xendbhookinplaceofnumber	44
\Xendbhooklinenumber	43
\Xendbhooknote	46
\Xendboxendlinenumalign	43
\Xendboxlinenum	43
\Xendboxlinenumalign	43
\Xendboxstartlinenumalign	43
\Xendboxsymlinenum	43
\Xendhangindent	46
\Xendinplaceofflemmaseparator	44
\Xendinplaceofnumber	42
\Xendinplaceofpagenumber	40
\Xendlemmadisablefontselection	45
\Xendlemmafont	45
\Xendlemmaseparator	44
\Xendlineprefixmore	43
\Xendlineprefixsingle	43
\Xendlinerangeseparator	40
\Xendmorethantwolines	41

<code>\Xendnonumber</code>	41
<code>\Xendnotefontsize</code>	45
<code>\Xendnotenumfont</code>	45
<code>\Xendnumberonlyfirstinline</code>	39
<code>\Xendnumberonlyfirstintwolines</code>	39
<code>\Xendpagenumberonlyfirst</code>	39
<code>\Xendparagraph</code>	50
<code>\Xendsep</code>	50
<code>\Xendsublinesep</code>	41
<code>\Xendsymlinenum</code>	39
<code>\Xendsympagenum</code>	40
<code>\Xendtwolines</code>	41
<code>\Xendtwolinesbutnotmore</code>	41
<code>\Xendwrapcontent</code>	46
<code>\xflagref</code>	<u>1</u>
<code>\Xgroupbyline</code>	48
<code>\Xgroupbylineseparetwolines</code>	48
<code>\Xhangindent</code>	46
<code>\Xhsizethreecol</code>	47
<code>\Xhsizetwocol</code>	47
<code>\Xinplaceoflemmaseparator</code>	44
<code>\Xinplaceofnumber</code>	42
<code>\Xinsertparafootsep</code>	<u>1</u>
<code>\Xledsetnormalparstuff</code>	<u>1</u>
<code>\xleft@appenditem</code>	<u>1</u>
<code>\Xlemmadisablefontselection</code>	45
<code>\Xlemmafont</code>	45
<code>\Xlemmaseparator</code>	44
<code>\Xlinerrangeseparator</code>	40
<code>\xlineref</code>	<u>1</u> , 57
<code>\Xmaxhnotes</code>	49
<code>\Xmorethantwolines</code>	40
<code>\Xnolemmaseparator</code>	<u>1</u> , 44
<code>\Xnonbreakableafternumber</code>	42
<code>\Xnonumber</code>	41
<code>\Xnotefontsize</code>	45
<code>\Xnotenumfont</code>	45
<code>\Xnoteswidthliketwocolumns</code>	50
<code>\Xnumberonlyfirstinline</code>	39
<code>\Xnumberonlyfirstintwolines</code>	39
<code>\Xonlypstart</code>	41
<code>\Xpagelinesep</code>	42
<code>\xpageref</code>	<u>1</u> , 57
<code>\Xparafootsep</code>	48
<code>\Xparindent</code>	46
<code>\Xprenotes</code>	<u>1</u> , 49
<code>\Xprenotes@</code>	<u>1</u>
<code>\Xpstart</code>	41
<code>\Xpstarteverytime</code>	41
<code>\xpstartref</code>	<u>1</u> , 57

\XR@test	<u>1</u>
\XR@test@mac	<u>1</u>
\XR@test@mac@test	<u>1</u>
\Xragged	48
\xright@appenditem	<u>1</u>
\Xrigidbalance	<u>1</u>
\Xstanza	41
\Xstanzaseparator	41
\Xstorelineinfo	<u>1</u>
\xsublineref	<u>1</u> , 57
\Xsublinesep	41
\Xsymlinenum	39
\Xtoendnotes	26
\Xtwolines	40
\Xtwolinesonlyinsamepage	40
\Xtxtbeforenotes	48
\Xtxtbeforenotesonlyonce	48
\Xunvxh	<u>1</u>
\Xwidth	50
\Xwrapcontent	46
\Xwrapendlemma	46
\Xwraplemma	45
\xxref	<u>1</u> , 57

Z

\zz@@@	<u>1</u>
--------------	----------

Change History

v0.1.0.	
General: First public release	1
v0.2.0.	
\ifl@dmemoir: Added \ifl@dmemoir for memoir class having been used	78
\morenoexpands: Added \l@dtabnoexpands to \no@expands	128
\reledmac@error: Added \eledmac@error and replaced error messages	79
General: Added tabmac code, and extended indexing	1
v0.2.1.	
\@lab: Removed page setting from \@lab	269
\doxtrafeet: Renamed \doxtrafeet to \l@ddoxtrafeet	260
\edlabel: Tweaked \edlabel to get correct page numbers	266
\l@ddodoreinxtrafeet: Renamed \dodoreinxtrafeet to \l@ddodoreinxtrafeet	261
\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	128
\zz@@@: Minor change to \zz@@@	266
General: Added text about normal labeling	58
Bug fixes and match with mempatch v1.8	1
Major changes to insert code when memoir is loaded	263
v0.2.2.	
\footfudgefiddle: Added \footfudgefiddle	177
\next@line@list@stuff: Added initial write of page number in \line@list@stuff	120
\para@footsetup: Added \footfudgefiddle to \para@footsetup	178
\para@footsetupX: Added \footfudgefiddle to \para@footsetupX	218
General: Improved paragraph footnotes	1
New Dekker example	1
Used \providecommand for \@gobblethree to avoid clash with the amsfonts package	85
v0.3.0.	
\@lab: Replaced \the\line@num by \linenumr@p\line@num in \@lab, and similar for sub-lines	269
\@nl@reg: Added a bunch of code to \@nl for handling \setlinenum	107
\ledlinenum: Added \linenumr@p and \sublinenum@rep to \leftlinenum and \rightlinenum	98
\linenumberlist: Added \linenumberlist mechanism	85
\printendlines: Added \linenumr@p and \sublinenumr@p to \printendlines	235
\printlines: Added \linenumr@p and \sublinenumr@p to \printlines	200
\sublinenumr@p: Added \linenumberstyle and \sublinenumberstyle	98
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	291
\Xarrangement@normal: Added minpage footnote setup to \footnormal	172
\edtext: Added \showlemma to \edtext	129
\l@dfeetendmini: Added \l@dfeetbeginmini, \l@dfeetendmini and all their supporting code	289

\mpnormalfootgroup: Added \mpnormalfootgroup	176
\mpnormalvfootnote: Added \mpnormalvfootnote	174
\showlemma: Added \showlemma	85
General: Added final/draft options	75
Added ledgroup environment	292
Added ledgroupsize environment	293
Added minipage, etc., support	1
v0.4.1.	
\do@Xfeet: Changed \do@Xfeet code for easier extensions	261
\edindex: Let eledmac take advantage of memoir's indexing	298
\print@Xnotes: Added \op@Xfeet	261
General: Added code for changing \doclearpage	264
Not released. Minor editorial improvements and code tweaks	1
Only change \@footnotetext and \@footnotemark if memoir not used	203
v0.5.0.	
\@footnotetext: Enabled regular \footnote in numbered text	204
\@xympar: Eliminated \marginpar disturbance	281
General: Added left and right side notes	281
Added sidenotes, familiar footnotes in numbered text	1
v0.5.1.	
\affixline@num: Changed \affixline@num to cater for sidenotes	157
\l@edgetsidenote@margin: Added \sidenotemargin and \sidenote@margin	282
General: Added moveable side note	281
Fixed right line numbers killed in v0.5	1
Only change \hsize in ledgroupsize environment otherwise page number can be in wrong place	293
v0.6.0.	
\@lopR: Added \@pend, \@pendR, \@lopL and \@lopR in anticipation of parallel processing	109
\@nl@reg: Added \fix@page to \@nl	107
Extended \@nl to include the page number	107
\fix@page: Added \last@page@num and \fix@page	108
\get@thisfootnote: Changed \l@dbfnote and \vl@dbfnote as originals could give incorrect markers in the footnotes	204
\new@line: Extended \new@line to output page numbers	120
General: Fixed long paragraphs looping	1
Fixed minor typos	1
Prepared for eledpar package	1
v0.7.0.	
\@nl@reg: Added \@nl@reg	107
\@ref@reg: Added \@ref@reg	116
\affixline@num: Added skipnumering to \affixline@num	157
\do@actions@fixedcode: Added \do@actions@fixedcode	156
\do@actions@next: Added number skipping to \do@actions	154
\do@insidelinehook: Added \do@linehook for use in \do@line	151
\endnumbering: Changed \endnumbering for eledpar	89
\fix@l@cks: Added \ch@cksub@l@ck, \ch@ck@l@ck and \fix@l@cks	160
\footplitskips: Added \footplitskips for use in many footnote styles	170
\get@linelistfile: Added \get@linelistfile	105
\initnumbering@reg: Added \initnumbering@reg	87

\l@advance@parledgroup@beforenormalnotes: Added \l@dunboxmpfoot containing some common code	292
\l@dcsnotetext@r: Added \l@emptyd@ta	152
\l@dgetline@margin: Added \l@dgetline@margin	94
\l@dgetlock@disp: Added \l@dgetlock@disp	96
\l@dgetsidenote@margin: Added \l@dgetsidenote@margin	282
\l@dnumstartsL: Added \l@dnumstartsL, \ifl@dpairing and \ifpst@rted for/from eledpar	86
\l@drsn@te: Added \l@dlsn@te and \l@drsn@te for use in \do@line	152
\l@dzeropenalties: Added \l@dzeropenalties	147
\ledlinenum: Added \ledlinenum for use by \leftlinenum and \rightlinenum	98
\list@clearing@reg: Added \list@clearing@reg	105
\n@num: Added \n@num	114
\next@line@list@stuff: Deleted \page@start from \line@list@stuff	120
\normalbfnoteX: Removed extraneous space from \normalbfnoteX	210
\pausenumbering@page@num: Changed \resumenumbering for eledpar	90
\setprintendlines: Added \setprintendlines for use by \printendlines	233
\setprintlines: Added \setprintlines for use by \printlines	197
\skipnumbering: Added \skipnumbering and supports	125
\sublinenumincrement: Added \firstlinenum, \linenumincrement, \firstsublinenum and \linenumincrement	96
\sublinenumr@p: Using \linenumrep instead of \linenumr@p	98
Using \sublinenumrep instead of \sublinenumr@p	98
\vnumfootnoteX: Removed extraneous space from \vnumfootnoteX	212
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	79
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@dnumstartsL: Added \ifledRcol and \ifnumberingR for/from eledpar	86
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 <code>\numberpstarttrue</code>	1
v0.12.1.	
General: Don't number <code>\pstarts</code> of stanza.	1
The numbering of <code>\pstarts</code> restarts on each <code>\beginnumbering</code>	1
v0.13.0.	
<code>\managestanza@modulo</code> : New <code>stanzaindentsrepetition</code> counter to repeat stanza indents every n verses.	307
General: New <code>stanzaindentsrepetition</code> counter to repeat stanza indents every n verses.	52
New <code>stanzaindentsrepetition</code> counter: to repeat stanza indents every n verses.	1
v0.13.1.	
General: <code>\thepstartL</code> and <code>\thepstartR</code> use now <code>\bfseries</code> and not <code>\bf</code> , which is deprecated and makes conflicts with memoir class.	1
v0.14.0.	
<code>\edlabel</code> : Tweaked <code>\edlabel</code> to get correct line number if the command is first element of a paragraph.	266
General: Tweaked <code>\edlabel</code> to get correct line number if the command is first element of a paragraph.	1
v0.15.0.	
<code>\affixline@num</code> : Line numbering can be disabled.	157
<code>\ifinserthangingsymbol</code> : New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions.	305
<code>\printlines</code> : Line numbering can be reset at each <code>pstart</code>	200
General: Line numbering can be reset at each <code>pstart</code>	93
Possibility to print <code>\pstart</code> number inside.	19
v0.17.0.	
<code>\ifinserthangingsymbol</code> : New new management of <code>hangingsymbol</code> insertion, preventing undesirable insertions.	305
v1.0.0.	
<code>\morenoexpands</code> : Change to be compatible with new features	128
General: <code>\lemma</code> can contain commands.	27
Debug in lineation command	21
New generic commands to customize footnote display.	37, 251
Options <code>nonum</code> and <code>nosep</code> in <code>\Xfootnote</code>	25
Options of <code>\Xfootnotes</code>	168
Possibility to have commands in sidenotes.	61
Some compatibility break with <code>eledmac</code> . Change of name: <code>eledmac</code>	1
v1.0.1.	
General: Correction on <code>\Xnumberonlyfirstinline</code> with lineation by <code>pstart</code> or by page.	39
v1.1.0.	
<code>\Xprenotes</code> : New skip <code>\Xprenotes@</code>	226
<code>\settoggle@series</code> : <code>\settoggle@series</code> switch the global value of the toggle, not only the local value.	252
General: Add <code>\labelpstarttrue</code>	19
Add <code>\Xnumberonlyfirstintwolines</code>	39
Add <code>\Xpstart</code> and <code>\Xonlypstart</code>	41
New hook to add arbitrary code at the beginning of the notes	46
New options for block of notes.	48
New package option: <code>parapparatus</code>	1
New tools to change order of series	250
Sectioning commands.	68

v1.2.0.	
<code>\Xprenotes</code> :	Debug in familiar footnotes (bug introduced by v1.1). 226
<code>\endquote</code> :	Compatibility of <code>\ledchapter</code> with the <i>memoir</i> class. 344
v1.3.0.	
<code>\endquote</code> :	<i>Quotation</i> and quote environment inside numbered sections. 344
v1.4.0.	
<code>\edtext</code> :	Compatibility of <code>\edtext</code> with the right-to-left direction (with Polyglossia). 129
<code>\ledsetnormalparstuffX</code> :	Direction of footnotes with polyglossia. 222
<code>\newseries@</code> :	Remembers the language of the lemma, in order to create a correct direction for the footnote separator. 241
<code>\rbracket</code> :	Switch the right bracket to a left bracket when the lemma is RTL (needs polyglossia or LuaTeX). 191
General:	Compatibility with LuaTeX of RTL notes. 1
v1.4.1.	
<code>\affixside@note</code> :	Remove spurious spaces. 288
<code>\endquote</code> :	New option <i>noquotation</i> 344
<code>\get@thisfootnote</code> :	Compatibility of standard footnotes with <code>eledmac</code> when these footnotes contain any commands. 204
<code>\labelrefsparsesubline</code> :	Fix a bug with <code>\edlabel</code> 268
v1.4.2.	
General:	Debug with some special classes. 1
v1.4.3.	
General:	Add <code>\Xnonbreakableafternumber</code> 42
Spurious space after familiar footnotes. 1
v1.4.4.	
General:	Label inside familiar footnotes. 1
v1.4.5.	
General:	Bug with <code>komasscript</code> + <code>eledpar</code> + <code>chapter</code> 1
v1.4.6.	
General:	Bug with <code>memoir</code> class introduced by 1.4.5. 1
v1.4.7.	
<code>\endquote</code> :	Compatibility of sectioning commands with <code>\autopar</code> 344
v1.4.8.	
General:	Corrects a bug with parallel texts introduced by 1.1. 1
v1.4.9.	
<code>\normalbfnoteX</code> :	Allow to redefine <code>\thefootnoteX</code> with <code>alph</code> when some packages are loaded. 210
v1.5.0.	
<code>\do@insidelinehook</code> :	Added <code>\do@insidelinehook</code> for use in <code>\do@oline</code> 151
<code>\edindex</code> :	Compatibility with <code>imakeidx</code> package, and possibility to use multiple index with <code>\edindex</code> 298
General:	Correct indexing when the call is made in critical notes. 294
v1.5.1.	
<code>\managestanza@modulo</code> :	Correct <code>stanzaindentsrepetition</code> counter 307
<code>\normalvfootnoteX</code> :	Fix a bug with normal familiar footnotes when mixing RTL and LTR text. 207
v1.6.0.	
<code>\newverse</code> :	Add <code>\falseverse</code> macro. 310
v1.6.1.	
<code>\AtStartEveryPstart</code> :	Spurious space in <code>\pstart</code> 142

\ifinserthangingsymbol: Hang verse is now not automatically flush right.	305
\l@dunhbox@line: Move the call to \inserthangingsymbol to allow use \hfill inside.	148
\pend: Spurious space in \pend.	144
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.	69
v1.8.0.	
\endquote: Correction of sectioning commands in parallel texts.	344
\get@index@command: Debug \get@index@command and compatibility with hyperref package.	297
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work.	254
\prevpage@num: Correct \parafootsep when using with ledgroup.	183
General: Compatibility with parledgroup option ofeledpar package.	1
If imakeidx and hyperref are loaded, adds hyperref in the index.	294
v1.8.1.	
General: Debug endnotes when more than one series is used (change the position where tools for endnotes are defined).	227
v1.8.2.	
General: Debug compatibility problem with hebrew option of babel package.	1
v1.8.3.	
General: Fixes spurious spaces added by v1.7.0.	1
v1.8.5.	
General: Debug indexing in right column, witheledpar.	294
v1.9.0.	
\doxtrafeet: Add \fnpos to choice the order of footnotes.	260
\l@dfeetendmini: Add \mpfnpos to choice the order of footnotes in minipage / ledgroup.	289
v1.10.0.	
\endquote: Correction of sectioning commands in parallel texts.	344
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 environment.	1
v1.10.5.	
General: Debug of \pausenumbering.	1
Debug of \xxref.	1
v1.10.6.	
General: Debug of interaction between \autopar and \pausenumbering.	1
v1.11.0.	
General: Add hooks to disable the font selection for lemma in footnote.	45
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.	106
\AtStartEveryPstart: New optional argument for \pstart, to execute code before it.	142
\edindex: Use correctly default index when imakeidx is loaded.	298
\endquote: \ledxxx sectioning commands are deprecated and replaced by \eledxxx commands.	344
\initnumbering@reg: \beginnumbering is defined only on eledmac, not on eledpar.	87
\l@dgetsidenote@margin: \sidenotemargin is now directly defined in eledmac to be able to manage eledpar.	282
\l@disnote: \l@dlsnote, \l@drsnote and \l@dcsnote defined only one time, in eledmac, including needs for eledpar case.	283
\l@dnumpstartsL: Add \ifledRcol@ for eledpar	86
\l@dunhbox@line: \do@line is split in more little commands.	149
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work when called after \footparagraphX.	254
Debug \Xbeforenotes and \Xmaxhnotes which did not work when called after \footparagraph.	254
\pend: New optional argument for \pend, to execute code after it.	144
\stanza: &can have an optional argument: content to be printed after.	310
\Stanza can have an optional argument: content to be printed before.	310
Add \newverse macro, \falseverse deprecated.	310
General: Add \ledinnernote and \ledouternote commands.	61
Add \Xendparagraph and related settings.	50
Add hyperlink to crossref (needs hyperref package).	56
Compatibility with musixtex.	1
Debug eledmac sectioning command after using \resumenumbering.	1
Ensure that imakeidx is loaded <i>before</i> eledmac	294
New hooks: \Xafterrule and \afterruleX	49
New options for ragged-paragraph notes	48
New sectioning commands.	68
Optional arguments for \pstart and \pend.	18

v1.12.1.

\wrap@edcrossref: Fix spurious spaces.	271
--	-----

v1.12.2.

\l@dunhbox@line: Fix a bug with critical notes at the tops of pages (added by v12.0.0)	148
--	-----

v1.12.3.

\flag@end: \flag@start and \flag@end are now defined only one time for eledmac and eledpar	121
\flag@start send a error message when a \edtext is done without insert (note)	121
\reledmac@error: Replaced error messages	79
General: Add macros for new messages since v0.7	79
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.	61
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	300
\@ssect: Debug \eledchapter in parallel with memoir	347
\doinsidelinehook: Added \dolinehook and \doinsidelinehook	152
\endnumbering: Allow to mix parallel columns and normal text when using	
\pausenumbering	89
\l@dgobblearg: \l@dgobblearg becomes \l@gobbeloptarg	327
\l@drestoreforedtext: Debug optional arguments of \Xfootnote in tabular context	328
\pausenumbering@page@num: Debug \resumenumbering	90
v1.12.7.	
\wrap@edcrossref: \wrap@edcrossref is now robust	271
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	121
v1.13.0.	
\newhooktoggle@series@reload: Add \newhookcommand@toggle@reload	254
\para@footsetupX: In \para@footsetupX, use \columnwidth instead of \hsize	218
\settoggle@series: \settoggle@series can take an optional arguments to reload	
series setup.	252
General: Add \Xnoteswidthliketwocolumns and \noteswidthliketwocolumnsX	50
Added widthliketwocolumns option	75
v1.13.1.	
\ifat@every@pstart@star@: Add \l@dzeroopenalties in \pstart	143
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	86
General: Fix a 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 a bug with index when memoir class is used without hyperref	1
v1.14.0.	
\edindex: Let \eledmac take advantage of \makeidx even when memoir class is used	298
General: Debug spurious characters before endnotes.	227
Delete previous override of \l@d@wrindexhyp at the beginning of a document	
when hyperref is not loaded.	302
Move gobbling command	85
Provide \@gobblefour	85
v1.14.1.	
\@ssect: Debug sectioning commands when using both handout and hyperref	
package.	350
v1.14.2.	
\@ssect: Debug \edtext after starred sectioning commands when using memoir class.	347
v1.15.0.	
\@edtext@level: New boolean \if@edtext@.	129
\arrangementX@threecol: Correct bug with paragraphed familiar footnotes setting.	217
\endsub: Restore subline feature (disabled by mistake in v1.8.0).	123
\if@lemmacommand@: New boolean \iflemmacommand@.	134

General: Fix a bug with footnotes layout when using some options of the geometry package (bug add by v1.13.0).	1
New commands <code>\AtEveryPstart</code> and <code>\AtEveryPend</code> .	19
New tools to prevent ambiguous references in lemma	28
v1.15.1.	
<code>\next@line@list@stuff</code> : Revert modification of 1.5.2 which makes bug with numbering. Leave vertical mode to solve spurious space before minipage.	120
v1.16.0.	
General: <code>\edtext</code> is now defined only in <code>eledmac</code> , not in <code>eledpar</code> . Debug wrong numbering when using <code>\sameword + eledpar + \tag</code> command.	129
Compatibility of standard footnotes with some biblatex styles.	1
New <code>\stanzaindent</code> command.	1
v1.16.1.	
<code>\xlineref</code> : <code>\lineref</code> is not defined if defined by some other package, like <code>lineno</code> . Eledmac provides <code>\edlineref</code> instead.	272
v1.17.0.	
<code>\edtext</code> : Error message when calling <code>\edtext</code> outside of a numbered paragraph.	129
v1.18.0.	
<code>\@edindex@hyperref</code> : Fix spurious space with <code>\edindex</code> when using <code>imakeidx/indextools + hyperref</code> .	300
<code>\edlabel</code> : <code>\edlabel</code> is now defined only one time for both <code>eledmac</code> and <code>eledpar</code>	266
<code>\l@d@section</code> : Option <code>parapparatus</code> works for endnotes.	228
<code>\l@dnumppstart:L</code> : Add <code>\ifl@dprintingpages</code> and <code>\@dprintingcolumns</code> for <code>eledpar</code>	86
<code>\print@line</code> : Compatibility with Lua \TeX RTL languages.	149
<code>\printlinefootnote</code> : Code refactoring in <code>\printlinefootnote</code> : the printing of the numbers are factorized in <code>\printlinefootnotearea</code>	193
<code>\printpstart</code> : Debug <code>\Xpstart</code> with parallel pages and columns (<code>eledpar</code>)	191
General: Add <code>\Xpstarteverytime</code>	41
Compatibility with Lua \TeX RTL languages.	1
Debug <code>\Xonlypstart</code> when using <code>\Xnumberonlyfirstinline</code> and the current line number differs from the previous.	41
v1.19.0.	
<code>\footssplitskips</code> : <code>\footssplitskips</code> doesn't set <code>\floatingpenalty</code> to <code>\@MM</code> when processing parallel pages.	170
<code>\xxref</code> : <code>\xxref</code> works also with right side numbers, when <code>\@Rlineflag</code> is not empty.	274
General: <code>\Xmaxhnotes</code> and <code>\maxhnotesX</code> work now for both two-columns and three-columns setting.	1
Compatibility with <code>eledpar v1.13.0</code> .	1
v1.19.1.	
General: Call <code>\correct@footinsX@box</code> and <code>\correct@Xfootins@box</code> directly in <code>\print@notesX@forpages</code> and <code>\print@Xnotes@forpages</code> , that is in <code>eledpar</code> .	1
v1.20.0.	
<code>\printlines</code> : Added <code>\ifl@d@Xmorethantwolines</code> and <code>\ifl@d@Xmorethantwolines</code> to <code>\printlines</code>	200
<code>\stanza</code> : <code>&</code> and <code>&</code> can be preceded by spaces.	310
<code>\xxref</code> : Debug <code>\xxref</code> when not loading <code>eledpar</code> (Fix a bug added in 1.19.0).	274
General: Add <code>\Xendboxlinenum</code>	43
Add <code>\Xtwolines</code> and <code>\Xmorethantwolines</code> hooks	40
Add series option.	1

Correct <code>\Xinplaceofnumber</code> hook.	1
Explicit error message when calling <code>\Xfootnote</code> outside of <code>\edtext</code>	1
Fix a bug with line number typesetting direction when using <code>\eledsection</code> and similar commands for RTL texts with <code>Lua\TeX</code>	1
Fix issues with RTL text in notes when using <code>Lua\TeX</code>	1
Options fulllines in <code>\Xfootnote</code>	25
The <code>\newifs</code> are not followed by boolean values set to false, because it is the <code>\TeX</code> default setting.	1
v1.21.0.	
<code>\@edindex@hyperref</code> : Look at the hyperindex option of hyperref before inserting hyperref	300
<code>\l@d@section</code> : <code>\endnotes</code> take five arguments.	228
<code>\ledinnotemark</code> : Add <code>\ledinnotemark</code>	297
<code>\ledsetnormalparstuffX</code> : <code>\ledsetnormalparstuff</code> is deprecated and becomes <code>\ledsetnormalparstuffX</code> and <code>\Xledsetnormalparstuff</code>	222
<code>\n@num</code> : <code>\n@num@ref</code> deleted	114
<code>\n@num</code> defined only one time for both <code>Eledmac</code> and <code>Eledpar</code>	114
<code>\newhookcommand@series</code> : <code>\newhookcommand@series</code> can take an optional argument.	253
<code>\newhooktoggle@series</code> : <code>\newhooktoggle@series</code> can take an optional argument.	254
<code>\print@footnoteXrule</code> : Code refactoring: the spaces after the footnote rules are directly managed in <code>\print@Xfootnoterule</code> and <code>\print@footnoteXrule</code>	225
<code>\seriesatend</code> : Fix spurious space in <code>\seriesatend</code>	251
<code>\skipnumbering</code> : <code>\skipnumbering</code> defined only one time for both <code>Eledmac</code> and <code>Eledpar</code>	125
Correct <code>\skipnumbering</code> for stanza.	125
Delete <code>\skipnumbering@reg</code>	125
General: <code>\AtEveryPstart</code> and <code>\AtEveryPend</code> are now compatible with <code>\autopar</code> . . .	1
<code>\Xafterrule</code> and <code>\afterruleX</code> features no longer create problems of overflowing at the bottom of the page.	1
<code>\chapter</code> inside optional argument of <code>\pstart</code> works when typesetting parallel pages	1
<code>\preXnotes</code> and <code>\prenotesX</code> features no longer create problems of overflowing at the bottom of the page.	1
<code>\seriesatbegin</code> and <code>\seriesatbegin</code> more efficient	250
Add <code>\applabel</code> and related	58
Add <code>\beforenotesX</code> and <code>\Xbeforenotes</code> features for notes set in two and three column.	1
Add <code>\hidenumbering</code>	23
Add <code>\Xcolalign</code> and <code>\colalignX</code> hooks	47
Add <code>\Xendtwolines</code> , <code>\Xendmorethantwolines</code> , <code>\Xendtwolinesbutnotmore</code> and <code>\Xendtwolinesonlyinsamepage</code>	41
Add <code>\Xparindent</code> and <code>\hangindentX</code>	46
Add <code>\Xtwolinesbutnotmore</code> and <code>\Xtwolinesonlyinsamepage</code>	1
Add <code>nocritical</code> , <code>noend</code> , <code>nofamiliar</code> and <code>noledgroup</code> options.	1
Add <code>noeledsec</code> package option	1
Debug <code>\beforenotesX</code> <code>\maxhnotesX</code> <code>\noteswidthliketwocolumnsX</code> and <code>\afterruleX</code> with footnotes set in two and three columns.	1
Fix a bug when a <code>\Xfootnote</code> follows a <code>\Xendnote</code> in the second argument of <code>\edtext</code> (bug added in <code>eledmac 1.0.0</code>).	1

Fix a bug with <code>\maxhnotesX</code> when using <code>\foottwocolX</code> or <code>\footthreecolX</code> .	1
Fix a bug with space between columns with notes in two columns (bug added in v1.13.0).	1
Fix spurious space after first page number in <code>\doendnotes</code> . <code>oldprintnpnumspace</code> option allows to come back to previous setting	1
<code>parapparatus</code> option works now with familiar footnotes.	1
Provide <code>\@gobblefive</code>	85
v1.22.0.	
<code>\ledinnote</code> : <code>\ledinnote</code> takes a first optional argument, which is the label for hyperlinks.	297
General: Add <code>\doendnotesbysection</code> command.	26
Add option for lemma separator inside endnotes	44
Adds hyperlink for references to notes in indices.	1
Fix conflict between <code>noend</code> package option and <code>edtabularx</code> environments	1
Provides support for <code>xindy</code> .	1
Standardize endnotes handbook.	26
When using <code>hyperref</code> package, internal links in index or with <code>\edlineref</code> are now targeted to the top and not longer to the bottom of the lines they refer to.	1
v1.22.1.	
<code>\prevpage@num</code> : Correct double symbol when using both <code>\parafootsep</code> and <code>\Xsymlinenum</code> .	183
General: Fix a bug (added on v1.22.0) with <code>\Xinplaceofnumber</code> hook.	1
v1.23.0.	
<code>\@edtext@level</code> : The boolean <code>\if@edtext@</code> becomes the counter <code>\edtext@level</code> .	129
<code>\Serefwithpage</code> : Debug <code>\Xendtwolines</code> , <code>\Xendmorethantwolines</code> , <code>\Xendtwolinesbutnotmore</code> and <code>\Xendtwolinesonlyinsamepage</code> when using <code>\apprefwithpage</code> .	277
<code>\lemma</code> : Fix spurious space after <code>\lemma</code> command	133
<code>\newseries@</code> : Prevent spurious spaces when <code>\Afootnote</code> and similar commands are followed by spaces (bug added on 1.0.0).	241
<code>\sameword</code> : In order to allow use of <code>\sameword</code> with <code>inputenc</code> , we detokenize its mandatory argument before using it in control sequence names.	138
General: Add <code>\Xboxlinenumalign</code> and <code>\Xendboxlinenumalign</code> .	43
Add <code>\Xboxstartlinenum</code> , <code>\Xendboxstartlinenum</code> , <code>\Xboxendlinenum</code> , <code>\Xendboxendlinenum</code> .	43
Allow use of <code>\sameword</code> with <code>inputenc</code> managing of UTF-8.	1
Compatibility between <code>nofamiliar/nocriticals</code> option and <code>minipage/ledgroup</code> .	1
Error message when using <code>\beginnumbering... \endnumbering</code> without <code>\pstart</code> .	1
Fix a bug with <code>\sameword</code> when the lemma overlaps multiple line.	28
Fix a bug with <code>\sameword</code> when the same lemma is used for multiple notes or for nested <code>\edtexts</code> .	28
Fix a bug with <code>\skipnumbering</code> called immediately after a <code>\pstart</code> .	1
Fix error of <code>\iftrue</code> not closed.	1
Fix spurious space with <code>\skipnumbering</code> (bug added on v1.21.0).	1
New tools to ensure the line-list file uses the right version of commands when upgrading the <code>eledmac</code> version.	1
Optional argument of <code>\sameword</code> can be a comma-separated list of <code>\edtext</code> depth.	28
v1.23.1.	
General: Fix a bug with <code>\lemma</code> command in the right side.	1

v1.23.2.	
General: Compatibility with L ^A T _E X'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 a 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.	291
\@xympar: Patching \@xympar instead of redefining it	281
\endminipage: Patch \endminipage instead of redefining it.	291
\initnumbering@quote: \initnumbering@sectcmd becomes \initnumbering@quote	344
\l@advance@parledgroup@beforenormalnotes: Some conde of \l@dumboxmpfoot moved to \l@advance@parledegroupp@beforenormalnotes	292
\newseries@: One endnotes file by series.	247
General: \@makecol, \@reinserts and \@doclearpage are patched instead of begin redefined	263
\doxtrafeeti becomes \Xdo@feet; \doxtrafeetii becomes \do@Xfeet; \@opxtrafeeti becomes \@opfeetX; \doreinxtrafeetii becomes \X@doreinfeet; \doreinxtrafeeti becomes \@doreinfeetX.	263
Add \Xendinplaceofnumber hook.	1
Add \Xendnonumber hook.	1
Add nonum option for endnotes.	1
Fix a bug when printing only one series of endnotes, but wanted to keep endnotes for other series.	1
In order to have a more consistent name's convention, many names has been changed.	1
Many L ^A T _E X's output macros are now patched and not override.	1
Package's name becomes reledmac.	1
Patch \@footnotemark instead of redefine it	204
Suppress indexing command specific to memoir.	298
v2.0.1.	
General: Fix a bug in eledmac-compat option	1
Fix incompatibility between optional argument of \pstart and \numberpstarttrue	1
v2.1.0.	
General: Fix a bug with \advanceline at the beginning of a \pstart.	1
Fix a bug with \chapter in optional argument of \pstart in parallel typesetting with scrbook.	1
Fix a bug with \eledchapter in parallel typesetting with scrbook.	1
Fix a bug with \setline at the beginning of a \pstart.	1

Fix spacing bug with <code>\Xbhooknote</code> and <code>\bhooknoteX</code> when using them to insert text and not to execute code.	1
New tools to number stanzas	1
v2.1.1.	
General: Fix a bug with <code>\ledpbsetting{before}</code>	1
v2.1.2.	
General: Fix a bug with lineation by <code>pstart</code> and <code>tabular</code> environments (added in 2.1.0). . . .	1
v2.1.3.	
<code>\ledsetnormalparstuffX</code> : Replaced <code>\noindent</code> with <code>\parindent</code> set to 0pt.	222
General: <code>\Xhangindent</code> and <code>\hangindentX</code> work now with all the paragraphs in the note.	1
<code>\Xnoindent</code> and <code>\noindentX</code> work now again (broken in 2.0.0).	1
Change some internal code in order to provide compatibility with \TeX release of october 2015	1
Fix a bug which inserted double space before paragraphed familiar notes.	1
Fix a bug with <code>\edindex</code> when using not-Latin characters without UTF-8 engines	1
v2.2.0.	
General: Fix a bug with combination of <code>\onehalfspacing</code> and two columns and three columns notes typeset.	1
Fix a bug with some setting command and optimization option.	1
Fix spurious space with paragraphed critical notes when using $\text{Lua}\TeX$	1
Increase line list version number to ensure compatibility with new options of <code>reledpar</code> package.	1
New setting tools for endnotes: <code>\Xendnumberonlyfirstinline</code> , <code>\Xendnumberonlyfirstintwoline</code> , <code>\Xendsymmlinenumber</code> , <code>\Xendbeforenumber</code> , <code>\Xendafternumber</code> , <code>\Xendbeforesymmlinenumber</code> , <code>\Xendaftersymmlinenumber</code> , <code>\Xendboxsymmlinenumber</code> , <code>\Xendhangindent</code> , <code>\Xendbhooklinenumber</code> , <code>\Xendahooklinenumber</code> , <code>\Xendbhookinplaceofnumber</code> , <code>\Xendahookinplaceofnumber</code>	1
v2.2.1.	
General: Compatibility with TeXformat 2015/10/01.	1
v2.2.2.	
General: Fix a bug in <code>\sethangingsymbol</code>	1
Fix a bug with old version of <code>etex</code>	1
v2.3.0.	
General: Disable empty lines as paragraph in stanza.	1
Fix incompatibility of paragraphed footnotes with <code> bidi v17.9</code> and following.	1
Warning message when using some setting commands inside <code>rightsides</code> environment (deprecated behavior)	1
v2.3.1.	
General: Fix spurious space when using optional argument of <code>\stanza</code> (introduced in v2.3.0).	1
v2.4.0.	
<code>\footnoteoptions@</code> : First argument of <code>\footnoteoption@</code> is now mandatory, not optional.	168
General: <code>\Xbhooknote</code> and <code>\bhooknoteX</code> work with notes in columns.	1
Fix a bug of <code>\parindentX</code> and <code>\Xparindent</code> with two columns and three columns notes.	1
Fix a bug with <code>\sameword</code> in right side.	1
Fix spurious space in two columns and three columns notes.	1

Fix spurious space when using optional argument of stanza (introduced in v2.3.0). . . .	1
New hooks: <code>\Xlinangeseparator</code> and <code>\Xendlinangeseparator</code>	40
Option <code>linangesep</code> for critical footnotes and endnotes.	40
v2.4.1.	
General: Fix a bug with <code>\appref</code> and <code>\apprefwithpage</code> (introduced in v2.4.0).	1
Fix a bug with tabular environments when using <code>babel</code> or <code>polyglossia</code> languages that override \TeX <code>\roman</code> command, like Greek language.	1
Fix a bug with tabular environments when using <code>babel</code> or <code>polyglossia</code> languages that override \TeX <code>\roman</code> command, like Greek.	1
v2.5.0.	
<code>\Serefwithpage</code> : Debug <code>\setapprefprefixsingle</code>	277
<code>\edlabel</code> : Fix a bug when calling <code>\edlabel</code> in a footnotes of the rightside	266
<code>\ld@section</code> : <code>\endnotes</code> take six arguments.	228
<code>\printlines</code> : <code>\printlines</code> takes an eighth argument: the line flag	200
<code>\xlineref</code> : <code>\xlineref</code> does not include anymore the side flag. Use <code>\xflagref</code> to get it. Not that <code>\edlineref</code> still contains the flag.	272
General: <code>\apprefwithpage</code> and <code>\appref</code> print double quotation mark when the label was not defined.	1
<code>\apprefwithpage</code> and <code>\appref</code> work with right side crossref.	1
<code>\apprefwithpage</code> works also when <code>noend</code> option is enabled.	1
<code>\appref</code> and <code>\apprefwithpage</code> can take <code>linangesep</code> optional argument.	1
<code>\edlabel</code> works now in <code>\Xfootnote</code>	1
<code>\lemma</code> can be used even when the <code>nocritical</code> is enabled.	1
Compatibility with new hook and tools of <code>reledpar</code> 2.6.0.	1
Fix spurious vertical space in <code>astanza</code> environment (<code>reledpar</code>)	1
Log now states ‘There were undefined references’ when using wrong references in <code>\edlineref</code> or <code>edpageref</code>	1
New hooks to customize page and line number appearance in endnotes.	1
New hooks: <code>\Xhookgroup</code> and <code>\hookgroupX</code>	1
New tools to easily make cross-reference to a passage defined by a start and an end line	58
v2.6.0.	
General: Adds compatibility with <code>innnote</code> and <code>notenumber</code> options of <code>indextools</code> package.	1
Fix a bug with footnote counter in <code>ledgroup</code> (added in v2.5.0).	1
Fix bug, introduced in v2.5.0, with footnote numbering in parallel typesetting when using <code>perpage</code> package.	1
v2.7.0.	
<code>\@k</code> : <code>\rigidbalance</code> is split in <code>\Xrigidbalance</code> and <code>\rigidbalanceX</code>	184
<code>\ld@section</code> : <code>\endnotes</code> take seven arguments.	228
General: Add dash as default page range separator for <code>\SEonlypage</code>	1
Debug <code>\Serefonlypage</code> when referring to only one page.	1
Delete parenthesis after <code>\Serefonlypage</code>	1
Fix (again) bugs with footnote numbering in parallel typesetting while using <code>ledgroup</code> environments (bug added in v2.5.0).	1
Fix a bug with <code>\Serefwithpage</code>	1
Fix bugs in compatibility with <code>innnote</code> and <code>notenumber</code> options of <code>indextools</code> package, when indexing outside of a <code>ledgroup</code>	1
New commands to make glossaries connected to page and linenumber with the <code>glossaries</code> package	1
New hooks: <code>\Xhsize</code> and <code>\hsizeX</code>	50

New hooks: <code>\Xlemmafont</code> and <code>\Xendlemmafont</code>	45
New setting commands: <code>\setSErefonlypageprefixsingle</code> and <code>\setSErefonlypageprefixmore</code>	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 <code>hyperref</code> package, add link in familiar footnotes between the footnote marks in the text and the footnote marks in the footnote	1
When using <code>hyperref</code> package, add links for <code>\SEref</code> and <code>related</code> , <code>\appref</code> and <code>related</code>	1
When using <code>hyperref</code> package, add links from critical footnotes and critical endnotes to the line of text they refers	1
v2.7.1.	
General: Debug <code>\Xhookgroup</code> hooks executed on columnar footnotes (moved to a larger group, to take effect).	1
v2.7.2.	
General: <code>\Xhsize</code> and <code>\hsizeX</code> become <code>\Xwidth</code> and <code>\widthX</code>	50
Fix problem of hyphenation when using <code>hyperref</code> package (added in v2.7.0).	1
v2.8.0.	
<code>\l@d@section</code> : <code>\Xendhangindent</code> and <code>\Xendafternote</code> can take values which are relative to the font size of the endnote.	228
General: <code>reledmac</code> cross-referencing can take advantage of <code>xr</code> package.	1
More <code>\edgls...</code> commands.	1
No indentation for paragraphed notes in <code>ledgroup</code> . Can be changed with <code>\Xparindent</code> and <code>\parindentX</code>	1
v2.8.1.	
General: Warnings for undefined labels are really parsable by latexmk	1
v2.8.2.	
General: Fix a bug concerning indent in a paragraph immediately following a sectioning command (bug NOT fixed on <code>reledpar</code>)	1
Fix a bug with <code>\AtEveryPstart</code> added in version 2.0.0.	1
Fix a bug with vertical space after the between-sectioning command as optional argument of a <code>\pstart</code> and <code>\pstart</code> content	1
v2.9.0.	
General: Allow continuing line numbering between normal text and parallel text, using <code>\pausenumbering</code> and <code>\resumenumbering</code> and the <code>continuousnumberingwithcolumns</code> option.	1
Fix a bug when using <code>\lineneation{page}</code> and <code>\pausenumbering...resumenumbering</code>	1
Fix a bug with three- and two-column footnote setting (added in v2.4.0).	1
Fix spurious space inside three-column familiar footnote.	1
Write correct metadata in numbered files when using <code>\pausenumbering...resumenumbering</code>	1
v2.9.1.	
General: Fix a bug when notes start with “plus” or “minus”.	1
v2.9.2.	
General: Fix a bug with <code>hyperref</code> package when a lemma starts with “plus” or “minus” (bug introduced in v. 2.7.0).	1
v2.9.3.	
General: Fix a 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.	192
General: Add \AtEveryStanza and \AtEveryStopStanza.	1
Fix a bug in \ledlsnotefontsetup and \ledrsnotefontsetup which could not handle \color command properly.	1
More specific error messages.	1
New hooks: \Xwrapcontent, \Xendwrapcontent and \wrapcontentX.	46
New hooks: \Xwraplemma and \Xendwraplemma	45
v2.10.1.	
General: Add ‘nopenalties’ option.	1
Fix a 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	156
General: Add new tools to produce an apparatus of manuscripts	1
Fix a 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 a bug in critical and familiar footnotes when using uppercase letters with accent mark	1
Fix a bug when using \chapter in optional argument of \pstart in parallel typesetting in combination with the noeledsec option.	1
Fix a bug with \ledinnernote and \ledouternote in parallel typesetting	1
Fix a 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 typesetting.	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 a 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 a bug when using <code>\footnoteX</code> in the first argument of <code>\edtext</code> .	1
v2.14.1a.	
General: Fix problematic typos in the handbook.	1
v2.15.0.	
General: Add ‘byline’ arrangement.	1
Fix <code>\Xtxtbeforenotes</code> in <code>ledgroup</code> .	1
v2.15.1.	
General: Fix <code>\edindex</code> in tabular environments.	1
v2.15.2.	
General: Fix a bug with <code>fancyhdr</code> package 3.8 and later.	1
v2.15.3.	
General: Fix a bug with <code>\section</code> in optional argument of <code>\pstart</code> and empty line before <code>\pend</code> (bug added in v2.8.2).	1
Simplification of the sectioning command code.	1
v2.16.0.	
General: <code>\Xdo@feet</code> becomes <code>\do@Xfeet</code>	263
Add <code>\Xendpagenumberonlyfirst</code> , <code>\Xendpagenumberonlyfirstifsingle</code> , <code>\Xendpagenumberonlyfirstintwo</code> , <code>\Xendinplaceofpagenumber</code> and <code>\Xendsympagenum</code> hooks.	1
Add <code>\Xpagelinesep</code> hook.	42
Compatibility with new features of <code>reledpar</code>	1
Deleted dead code.	1
Display a warning message if using a version of \TeX that is too old.	1
Fix a bug with <code>\Xgroupbylines</code> for notes in two columns	1
Fix a bug with <code>\Xtxtbeforenotes</code> for notes in three or two columns	1
Fix a bug with ‘notenumber’ option of <code>indextools</code> package when indexing texts in familiar footnotes.	1
Fix potential bug when using <code>\edindex</code> in critical footnotes.	1
More explicit error message in case the stanza indentation is not defined.	263
New options for <code>\fnpos</code> and <code>\mpfnpos</code> to set a customized order for familiar and critical footnotes.	1
When <code>\edindex</code> is called outside of a <code>\beginnumbering... \endnumbering</code> structure, it is automatically switched to <code>\index</code> , with a warning message.	1
When indexing texts in familiar footnotes with <code>\edtext</code> , refer to the line number where the footnote is called.	1
When indexing texts in sidenotes with <code>\edtext</code> , refer to the line number where the sidenote is called.	1
v2.16.1.	
General: Fix a bug with redefinition of the style of the footnote number (bug added in v2.12.0)	1
v2.16.2.	
General: Error message if <code>footmisc</code> is loaded after <code>reledmac</code> .	1
Fix a bug introduced by v2.16.1 when using non-expandable control sequence, like <code>\normalfont</code> , in the footnote number style.	1
v2.16.3.	
General: Fix a bug with <code>\Seref</code> (bug added in v2.7.0).	1
v2.16.4.	
General: Fix a bug with vertical space before sectioning command in optional argument of <code>\pstart</code> (bug added in v2.15.3).	1

v2.16.5.	
General: Fix potential spurious spaces in endnotes.	1
v2.16.6.	
General: Fix a bug with the line number style in <code>\doennotes</code> when referring to right side line in parallel typesetting.	1
Take into account <code>\linenumberstyle</code> when using <code>\edlineref</code>	1
v2.16.7.	
General: Fix a bug with <code>\msdata</code> when using multiple <code>\beginnumbering... \endnumbering</code>	1
Fix a bug with <code>\numberpstarttrue</code> when using multiple <code>\beginnumbering... \endnumbering</code>	1
v2.16.8.	
General: Fix a bug with <code>\edindex</code> in footnotes in parallel typesetting.	1
v2.17.0.	
General: Add <code>\edglsadd</code> command.	1
Add <code>\setmsdataposition</code> setting.	1
v2.17.1.	
General: Fix spurious space in paragraphed footnotes when using Lua \TeX without 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 <i>latexmk</i>	1
v2.17.3.	
General: Fix a bug with <code>\doendnotesbysection</code> and <code>\doendnotes</code>	1
v2.17.4.	
General: Fix a bug with <code>\setSErefonlypageprefixsingle</code> and <code>\setSErefonlypageprefixmore</code>	1
v2.17.5.	
General: Fix a bug with <code>\pstartref</code> when referring to the left side in parallel typesetting.	1
v2.18.0.	
General: Fix a bug when using a <code>\edtext</code> in two lines or more in right-to-left typesetting with \TeX	1
Fix a bug when using both <code>\Xnumberonlyfirstintwolines</code> or <code>\Xnumberonlyfirstinline</code> and <code>\Xparafootsep</code> and <code>\Xsymlinenum</code>	1
v2.18.1.	
General: Fix a bug when using <code>\msdata</code> with Lua \TeX or with the <code>hyperref</code> package.	1
v2.19.0.	
General: Add <code>\footnoteXmark</code> and <code>\footnoteXtext</code> commands.	1
Add better compatibility with the <code>csquotes</code> package when using familiar footnotes.	1
Fix a bug with paragraph indent after sectioning command.	1
v2.20.0.	
General: Add <code>\AtStartEveryStanza</code> , <code>\BeforeEveryStopStanza</code> , <code>\AtEndEveryPend</code> , <code>\AtStartEveryPstart</code>	1
Add second optional argument of <code>\pstart</code> , <code>\pend</code> and <code>\stanza</code>	1
Add starred version of <code>\AtEveryPstart</code> , <code>\AtEveryPend</code> , <code>\AtEveryStanza</code> and <code>\AtEveryStopStanza</code>	1
Add third and fourth optional argument of <code>\newverse</code>	1
Fix a bug when using familiar footnotes in <code>\eledsection</code> and related.	1
Reset font specification at the beginning of familiar footnotes.	1

v2.21.0.	
General: Add the possibility of nested <code>\sameword</code> .	1
Fix a bug when using formatting command in the argument of <code>\edindex</code> inside <code>\edtext</code> .	1
Now, as explained in the handbook, an <code>\edindex</code> inside <code>\edtext</code> only creates index reference to main text, and not to the critical footnote.	1
v2.22.0.	
General: Add <code>\txtbeforenotesonlyonceX</code> and <code>\Xtxtbeforenotesonlyonce</code> hooks.	1
Add <code>\txtbeforenotesX</code> hook.	1
Fix a bug added in v2.16.0 when using <code>\Xtxtbeforenotes</code> with paragraphed or normal footnotes.	1
Fix a bug with three and two columns critical footnotes, broken in v. 2.17.6.	1
v2.22.1.	
General: Compatibility with new version of <code>reledpar</code> .	1
Fix a bug with some commands inside <code>\sameword</code> .	1
v2.22.2.	
General: Do not print footnotes at the first run.	1
v2.23.0.	
General: Add <code>swcaseinsensitive</code> option	1
Optimisation of the code added on v2.22.2 to not print footnotes at the first run.	1
v2.24.0.	
<code>\next@line@list@stuff</code> : Add <code>\next@line@list@stuff</code> internal hook.	119
General: Add <code>\swnoexpands</code> macro to avoid problems with not fully expandable macro inside <code>\sameword</code> .	1
Fix a bug with <code>\edlabel</code> at the beginning of a <code>\pstart</code> .	1
Fix spurious space with <code>\labelpstarttrue</code>	1
When a <code>\setlinenum</code> is used, it is stronger than the <code>\lineation{page}</code> setting.	1