

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	9
1.1 Aim of the package	9
1.2 History	10
1.2.1 edmac	10
1.2.2 ledmac	12
1.2.3 eledmac	12

*This file (**reledmac.dtx**) has version number v2.2.1, last revised 2015/10/05.

[†]maieul at maieul dot net

1.2.4 <code>reledmac</code>	12
1.3 List of works edited with (r)(e)ledmac	12
2 How the package works	12
3 Options	13
3.1 Specific features	13
3.2 Optimizing package performance	13
4 Text lines and paragraphs numbering	14
4.1 Text lines numbering	14
4.2 Paragraphs	15
4.2.1 Basics	15
4.2.2 Automatically producing <code>\pstart ... \pend</code>	15
4.2.3 Content before specific <code>\pstart</code> and after specific <code>\pend</code>	16
4.2.4 Content before every <code>\pstart</code> and after every <code>\pend</code>	16
4.2.5 Numbering paragraphs (<code>\pstart</code>)	16
4.2.6 Languages written in Right to Left	17
4.2.7 Memory limits	17
4.3 Lineation commands	17
4.3.1 Disabling lineation	17
4.3.2 Setting lineation start and step	18
4.3.3 Setting lineation reset	18
4.3.4 Setting line number margin	18
4.3.5 Other settings	19
4.4 Changing the line numbers	19
4.4.1 Sublineation	19
4.4.2 Locking lineation	19
4.4.3 Setting and changing line number	19
4.4.4 Line number style	20
4.4.5 Skipping and hiding number	20
4.4.6 Execute code at each line	20
5 Apparatus commands	21
5.1 Terminology	21
5.2 Critical notes	21
5.2.1 The lemma	21
5.2.2 Footnotes	22
5.2.3 Endnotes	22
5.2.4 Paragraph in critical apparatus	23
5.2.5 Change lemma and line number	23
5.2.6 Changing the names of commands for critical apparatus	24
5.3 Disambiguation of identical words in the apparatus	25
5.3.1 Basic use	25
5.3.2 Notes about input encoding with UTF-8 processor	25
5.3.3 Use with <code>\lemma</code> command	26

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

7 Fonts	39
8 Verse	40
8.1 Basic	40
8.2 Define stanza indents	40
8.3 Repeating stanza indents	40
8.4 Manual stanza indent	41
8.5 Stanza breaking	41
8.6 Hanging symbol	42
8.7 Long verse and page break	42
8.8 Content before/after verses	42
8.9 Numbering stanza	42
8.10 Various tools	43
9 Grouping	43
10 Cross referencing	43
10.1 Basic use	44
10.2 Refer to a critical notes	44
10.3 Cross-referencing which return a number in any case	44
10.3.1 Cross-referencing in order to define line number of a critical note	45
10.4 Not automatic cross-referencing	45
10.5 Normal L ^A T _E X cross-referencing	45
10.6 References to lines commented in the apparatus	45
11 Side notes	46
11.1 Basics	46
11.2 Setting	47
11.2.1 Width	47
11.2.2 Vertical position	47
11.2.3 Distance to the main text	47
11.2.4 Separator between notes	47
12 Indexing	47
12.1 Basics	47
12.2 Separator between page and line numbers	48
12.3 Using xindy	48
12.4 Advanced setting	49
13 Tabular material	49
14 Sectioning commands	52
14.1 Sectioning commands without line numbers or critical notes	52
14.2 Sectioning commands with line numbering and critical notes	53
14.3 Optimization	54
15 Quotation environments	54

16 Page breaks	54
16.1 Control page breaking	54
16.2 Prevent page break in a long verses	54
17 Miscellaneous	55
17.1 Known and suspected limitations	55
17.2 ‘No room for a new’	55
17.3 Marginal notes	56
17.4 Paragraph shape	56
17.5 Paragraphed footnotes	56
17.6 Use with other packages	57
17.7 Parallel typesetting	58
I Implementation overview	59
II Preliminaries	59
II.1 Links with original edmac	59
II.2 Package declaration	59
II.3 Package options	60
II.4 Loading packages	61
II.5 Compatibility with Lua \TeX	62
II.6 Boolean flags	62
II.7 Messages	63
II.8 Gobbling	68
II.9 Miscellaneous commands	68
II.10 Prepare reledpar	69
III Sectioning commands	69
IV List macros	73
V Line counting	74
V.1 Choosing the system of lineation	74
V.2 Line number margin	76
V.3 Line number initialization and increment	77
V.4 Line number locking	78
V.5 Line number style	79
V.6 Line number printing	79
V.7 Line number counters and lists	80
V.8 Line number locking counter	82
V.9 Line number associated to lemma	82
V.10 Reading the line-list file	85
V.11 Commands within the line-list file	87
V.12 Writing to the line-list file	98

VI Marking text for notes	103
VI.1 <code>\edtext</code> itself	104
VI.2 Substitute lemma	110
VI.3 Substitute line numbers	111
VI.4 Lemma disambiguation	112
VII Paragraph decomposition and reassembly	118
VII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	118
VII.2 Processing one line	123
VII.2.1 General process	123
VII.2.2 Process for “normal” line	124
VII.2.3 Process for line containing <code>\eledsection</code> command	125
VII.2.4 Hooks	126
VII.2.5 Sidenotes and marginal line number initialization	126
VIII Line and page number computation	127
IX Line number printing	130
X Pstart number printing in side	134
XI Restoring footnotes and penalties	135
XI.1 Add insertions to the vertical list	135
XI.2 Penalties	136
XI.3 Printing leftover notes	137
XII Critical footnotes	138
XII.1 Fonts	138
XII.2 Individual note options	138
XII.3 Notes language	139
XII.4 General survey of the way we manage notes	140
XII.5 General setup	140
XII.6 Footnotes arrangement	141
XII.6.1 User level macro	141
XII.6.2 Normal footnote	142
XII.6.3 Paragraphed footnotes	146
XII.6.4 Columnar footnotes	153
XII.7 Critical notes presentation	158
XII.7.1 Font tools	158
XII.7.2 Pstart number in footnote	159
XII.7.3 Line number printing	159
XIII Familiar footnotes	167
XIII.1 Adjacent footnotes	167
XIII.2 Regular footnotes for numbered texts	169
XIII.3 Footnote formats	170
XIII.4 Footnote arrangement	170

XIII.4.1 User level macro	170
XIII.4.2 Normal footnotes	171
XIII.4.3 Two columns footnotes	175
XIII.4.4 Three columns footnotes	177
XIII.4.5 Paragraphed footnotes	179
XIV Code common to both critical and familiar footnote in normal arrangement	183
XV Footnotes' width for two columns	184
XVI Footnotes' order	185
XVII Footnotes' rule	185
XVIII Specific skip for first series of footnotes	186
XVIII.0.1 Overview	186
XVIII.0.2 User level command	186
XVIII.0.3 Internal commands	187
XIX Endnotes	188
XX Generate series of notes	195
XX.1 Test if series is still existing	196
XX.2 Init specific to <code>reledpar</code>	196
XX.3 For critical footnotes	196
XX.3.1 Options	196
XX.3.2 Create inserts, needed to add notes in foot	197
XX.3.3 Create commands for critical apparatus, <code>\Afootnote</code> , <code>\Bfootnote</code> etc.	197
XX.3.4 Set standard display	199
XX.4 For familiar footnotes	199
XX.4.1 Options	199
XX.4.2 Create tools for familiar footnotes (<code>\footnotex</code>)	200
XX.5 The endnotes	201
XX.5.1 The auxiliary file	201
XX.5.2 The main macro	201
XX.5.3 The options	202
XX.6 Init standards series (A,B,C,D,E)	203
XXI Setting series display	203
XXI.1 Change series order	203
XXI.2 Test series order	204
XXI.2.1 Get the first series	204
XXI.3 Series setting	204
XXI.3.1 General way of working	204
XXI.3.2 Tools to set options	205

XXI.3.3 Tools to generate options commands	206
XXI.3.4 Options for critical notes	208
XXI.3.5 Options for familiar notes	209
XXI.3.6 Options for endnotes	209
XXI.4 Hooks for a particular footnote	210
XXI.5 Alias	211
XXII Output routine	211
XXII.0.1 Page number management	211
XXII.0.2 Extra footnotes output	211
XXII.0.3 Standard output's commands patching	214
XXIII Cross referencing	216
XXIV Side notes	225
XXV Minipages and such	232
XXVI Indexing	236
XXVII Verse	243
XXVII.1 Hanging symbol management	243
XXVII.2 Using & character	244
XXVII.3 Code category setting	244
XXVII.4 Stanza count and indent	244
XXVII.5 Numbering stanza	246
XXVII.6 Stanza number in note	247
XXVII.7 Main work	247
XXVII.8 Restore catcode and penalties	249
XXVIII Arrays and tables	249
XXVIII.1 Preamble: macro as environment	249
XXVIII.2 Tabular environments	253
XXVIII.2.1 Disabling and restoring commands	253
XXVIII.2.2 Counters, boxes and lengths	256
XXVIII.2.3 Tabular typesetting	260
XXVIII.2.4 Environments	271
XXIX Quotation's commands	271
XXX Section's title commands	273
XXX.1 Commands to disable some feature	273
XXX.2 General overview	273
XXX.3 \beforeeledchapter command	274
XXX.4 Auxiliary commands	274
XXX.5 Patching standard commands	275
XXX.6 Main code of \eledxxx commands	280

XXX.7 Macros written in the auxiliary file	282
XXXI Page breaking or no page breaking depending of specific lines	284
XXXII Long verse: prevents being separated by a page break	286
XXXIII Compatibility with eledmac	287
Appendix A Some things to do when changing version	289
Appendix A.1 Migrating from edmac to ledmac	289
Appendix A.2 Migration from ledmac to eledmac	290
Appendix A.3 Migration to eledmac 1.5.1	291
Appendix A.4 Migration to eledmac 1.12.0	291
Appendix A.5 Migration to eledmac 17.1	292
Appendix A.6 Migration to eledmac 1.21.0	292
Appendix A.6.1 \Xledsetnormalparstuffand\ledsetnormalparstuffX	292
Appendix A.6.2 Endnotes	292
Appendix A.7 Migration to eledmac 1.22.0	292
Appendix A.8 Migration to eledmac 1.23.0	292
Appendix A.9 Migration from eledmac to reledmac	293
Appendix A.9.1 Risk of ‘no room for a new’	293
Appendix A.9.2 Multiple indices with memoir	293
Appendix A.9.3 Deprecated commands and options	293
Appendix A.9.4 \renewcommandreplaced by command	294
Appendix A.9.5 Commands the names of which have been changed . . .	294
Appendix A.9.6 Endnotes	296
Appendix A.9.7 Z Series	296
Appendix A.9.8 Internal commands	296
Appendix A.10 Migration to reledmac 2.1.0	296
Appendix A.11 Migration to reledmac 2.1.3	296
References	297
Index	297
Change History	337

1 Introduction

1.1 Aim of the package

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

- automatic stepped line numbering, by page, section or paragraph;
- sub-lineation within the main series of line numbers;

- variant readings automatically keyed to line numbers;
- caters to both prose and verse;
- multiple series of footnotes and endnotes;
- block or columnar formatting of the footnotes;
- simple tabular material may be line numbered;
- indexing keyed to page and line numbers.

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

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

Any suggestions for new features are welcome.

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

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

1.2 History

1.2.1 `edmac`

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

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

The present version was developed in the summer of 1990, with the intent of adding necessary features, streamlining and documenting the code, and further generalizing it to make it easily adaptable to the needs of editors in different disciplines. John did most of the general reworking and documentation, with the financial assistance of the Division of the Humanities and Social Sciences, California Institute of Technology. Dominik

adapted the code to the conventions of Frank Mittelbach's doc option, and added some documentation, multiple-column footnotes, cross-references, and crop marks.¹ A description by John and Dominik of this version of edmac was published as 'An overview of edmac: a PLAIN 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.

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).

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: Schiphower 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.

1.2.2 `ledmac`

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

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

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

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

1.2.3 `eledmac`

Important changes were put in version 1.0, to make `ledmac` more easily extensible (see 6 p. 29). These changes can trigger small problems with the old customization. That is why a new name was selected: `eledmac` (extended `ledmac`).

To migrate from `ledmac` to `eledmac`, please read Appendix A.2 p. 290.

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. 293.

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

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

2 How the package works

The `reledmac` package is a three-pass package like L^AT_EX itself. Although your textual apparatus and line numbers will be printed on the first run, it takes two more passes through L^AT_EX to be sure that everything is correctly placed. If you make any subsequent changes altering the number of lines or notes, the input file may similarly require three

passes to get everything to the right place. `reledmac` will tell you that you need to make more runs when it detects changes, but it does not expend the labor to check this thoroughly. If you have problems with a line or two misnumbered at the top of a page, try running `ℒTEX` once or twice more.

A file may mix *numbered* and *unnumbered* text.

Numbered text is printed with marginal line numbers and can include footnotes and endnotes that are referenced to those line numbers: this is how you will want to print the text that you are editing.

Unnumbered text is not printed with line numbers, and you can't use `reledmac`'s note commands with it: this is appropriate for introductions and other material added by the editor around the edited text.

3 Options

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

3.1 Specific features

draft underlines lemmas in the main text.

eledmac-compat help to migrate from `eledmac` to `reledmac` (see Appendix A.9.5 p. 294).

nopbinverse prevents page breaks inside verses.

noquotation by default, the quotation environment is redefined inside numbered text. You can disable this redefinition with `noquotation` (see 15 p. 54).

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

xindy and `xindy+hyperref` are for selecting `xindy` as the index processor (12.3 p. 48).

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

3.2 Optimizing package performance

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

noeledsec disables tools for `\eledsection` and related commands (14.2 p. 53).

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

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

noledgroup `reledmac` allows use of a series of critical notes and a new series of normal notes inside `minipage` and `ledgroup` environments (see 9 p. 43). However, such features use up computer memory, at the expense of other processing needs. So if you do not need this feature, use `noledgroup` option. This should make `reledmac` faster.

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

4 Text lines and paragraphs numbering

4.1 Text lines numbering

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

```
\beginnumbering
Text
\endnumbering
```

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

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

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

4.2 Paragraphs

4.2.1 Basics

`\pstart` Within a numbered section, each paragraph of numbered text must be marked using the
`\pend` `\pstart` and `\pend` commands like this:

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

Text that appears within a numbered section but is not marked with `\pstart` and `\pend` will not be numbered.

The following example shows the proper section and paragraph markup and the kind of output that would typically be generated:

```
\beginnumbering
\pstart
This is a sample paragraph, with
lines numbered automatically.
\pend

\pstart
This paragraph too has its
lines automatically numbered.
\pend

The lines of this paragraph are
not numbered.

\pstart
And here the numbering begins
again.
\pend
\endnumbering
```

4.2.2 Automatically producing `\pstart ... \pend`

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

```

\begingroup
  \beginnumbering
  \autopar

  A paragraph of numbered text.

  Another paragraph of numbered
  text.

  \endnumbering
\endgroup

```

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

4.2.3 Content before specific `\pstart` and after specific `\pend`

Both `\pstart` and `\pend` can take a optional argument in brackets. Its content will be printed before the beginning of `\pstart` / after the end of `\pend` instead of the argument of `\AtEveryPstart` / `\AtEveryPend`. If you need to start a `\pstart` with brackets, or to add brackets after a `\pend`, just add a `\relax` between `\pstart ... \pend` and the brackets.

This feature is also useful when typesetting verses (see 8 p. 40) or `reledpar` (see 17.7 p. 58).

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

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

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

4.2.5 Numbering paragraphs (`\pstart`)

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

On each `\beginnumbering` the numbering restarts.

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

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

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

4.2.6 Languages written in Right to Left

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

4.2.7 Memory limits

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

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

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

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

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

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

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

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

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

4.3 Lineation commands

4.3.1 Disabling lineation

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

4.3.2 Setting lineation start and step

`\firstlinenum`
`\linenumincrement`

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

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

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

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

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

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

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

```
\def\linenumberlist{}
```

the standard numbering sequence is applied. The standard sequence is that specified by the combination of the `firstlinenum`, `linenumincrement`, `firstsublinenum` and `linenumincrement` counter values.

4.3.3 Setting lineation reset

`\lineation`

Lines can be numbered either by page, by `pstart` or by section; you specify this using the `\lineation{<arg>}` macro, where `<arg>` is either `page`, `pstart` or `section`.

You may only use this command at places where numbering is not in effect; you can't change the lineation system within a section. You can change it between sections: they don't all have to use the same lineation system. The package's standard setting is `\lineation{section}`. If the lineation is by `pstart`, the `pstart` number will be printed before the line number in the notes.

4.3.4 Setting line number margin

`\linenummargin`

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

```
\linenummargin{left}
```

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

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

4.3.5 Other settings

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

4.4 Changing the line numbers

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

4.4.1 Sublineation

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

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

4.4.2 Locking lineation

`\startlock` `\endlock` The `\startlock` command, used in running text, locks the line number at its current value, until you insert `\endlock`. It can tell for itself whether you are in a patch of line or sub-line numbering. One use for line-number locking is in printing poetry: there the line numbers should be those of verse lines rather than of printed lines, even when a verse line requires several printed lines. But in this case you may use the `\stanza` mechanism, see 8 p. 40.

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

4.4.3 Setting and changing line number

`\setline` `\advanceline` In some cases you may want to modify the line numbers that are automatically calculated: if you are printing only fragments of a work but want to print line numbers appropriate to a complete version, for example. The `\setline{<num>}` and `\advanceline{<num>}` commands may be used to change the current line's number (or the sub-line number, if sub-lineation is currently on). They change both the marginal

line numbers and the line numbers passed to the notes. `\setline` takes one argument, the value to which you want the line number set; it must be 0 or greater. `\advance` takes one argument, an amount that should be added to the current line number; it may be positive or negative.

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

4.4.4 Line number style

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

Alph Uppercase letters (A ... Z).

alph Lowercase letters (a ... z).

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

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

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

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

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

4.4.5 Skipping and hiding number

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

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

4.4.6 Execute code at each line

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

5 Apparatus commands

5.1 Terminology

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

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

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

So :

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

5.2 Critical notes

5.2.1 The lemma

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

```
\edtext{⟨lemma⟩}{⟨commands⟩}
```

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

For example:

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

1 Smith] Jones C, D.

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

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

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

1 Smith] Jones C, D.

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

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

5.2.2 Footnotes

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

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

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

An optional argument can be added before the text of the footnote. Its value is a comma-separated list of options. The available options are:

- `fulllines` to disable `\Xtwolines` and `\Xmorethantwolines` features for this note (cf. 6.2.2 p. 30).
- `nonum` to disable line numbering for this note.
- `nosep` to disable the lemma separator for this note.

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

5.2.3 Endnotes

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

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

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

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

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

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

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

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

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

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

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

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

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

5.2.4 Paragraph in critical apparatus

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

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

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

5.2.5 Change lemma and line number

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

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

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

1 Smith] Jones C, D.

1-2 I ... Tuesday.] The date was July 16, 1954.

`\linenum`

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

```
\linenum{⟨start page⟩|⟨s. line⟩|⟨s. sub-l.⟩|⟨end p.⟩|⟨e. l.⟩|⟨e. sub-l.⟩|⟨font⟩|}
```

However, you can retain the value computed by `reledmac` for any number by simply omitting it; and you can omit a sequence of vertical bars at the end of the argument. For example, `\linenum{|||23}` changes only the ending page number of the current lemma.

This command does not change the marginal line numbers in any way; it just changes the numbers passed to the notes. Its use comes in situations that `\edtext` has trouble dealing with for whatever reason. If you need notes for overlapping passages that aren't nested, for instance, you can use `\lemma` and `\linenum` to generate such notes despite the limitations of `\edtext`. If the `⟨lemma⟩` argument to `\edtext` is extremely long, you may run out of memory; here again you can specify a note with an abbreviated lemma using `\lemma` and `\linenum`. The numbers used in `\linenum` need not be entered manually; you can use the 'x-' symbolic cross-referencing commands below (10 p. 43) to compute them automatically.

Similarly, being able to manually change the lemma's font specifier in the notes might be important if you were using multiple scripts or languages. The form of the font specifier is three separate codes separated by `/` characters, giving the family, series, and shape codes as defined within NFSS.

5.2.6 Changing the names of commands for critical apparatus

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

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

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

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

5.3 Disambiguation of identical words in the apparatus

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

5.3.1 Basic use

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

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

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

5.3.2 Notes about input encoding with UTF-8 processor

If you use UTF-8 processor, like $\text{Xe}\text{L}\text{A}\text{T}\text{E}\text{X}$ or $\text{Lua}\text{L}\text{A}\text{T}\text{E}\text{X}$, 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 $\text{Xe}\text{L}\text{A}\text{T}\text{E}\text{X}$, add this line in your preamble: `\XeTeXinputnormalization 1`.
- If you use $\text{Lua}\text{L}\text{A}\text{T}\text{E}\text{X}$, use the `uninormalize` package of Michal Hoftich¹⁵ with the `buffer` option set to true.

With these tools, $\text{Xe}\text{L}\text{A}\text{T}\text{E}\text{X}$ / $\text{Lua}\text{L}\text{A}\text{T}\text{E}\text{X}$ will dynamically normalize unicode input when reading the file. Consequently, you will have no problems with the `\sameword` command.

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

5.3.3 Use with `\lemma` command

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

For example in the following example:

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

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

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

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

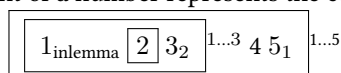
Note that only words that are actually referenced in a `\lemma` need the optional argument. Therefore, the first `\sameword` in the example above should have “1” as its optional argument, to be referenced correctly in the lemma.

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

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

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

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



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

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

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

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

5.3.4 Customizing

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

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

5.4 Familiar notes

5.4.1 Basic use

`\footnoteA` As well as the standard L^AT_EX footnotes generated via `\footnote`, the package also provides five series of additional footnotes called `\footnoteA` through `\footnoteE`. These

`\footnoteB` have the familiar marker in the text, and the marked text at the foot of the page can be

`\footnoteC` formatted using any of the styles described for the critical footnotes. Note that the ‘reg-

`\footnoteD` ular’ footnotes have the series letter at the end of the macro name whereas the critical

`\footnoteE` footnotes have the series letter at the start of the name.

5.4.2 Customizing mark

`\thefootnoteA` Each series uses a set of macros for styling the marks. The mark numbering scheme of

`\bodyfootmarkA` series A is defined by the `\thefootnoteA` macro; the default is:

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

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

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

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

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

There are similar command triples for the other series.

5.4.3 Separator for multiple footnotes

The `footmisc` package [Fai03] by Robin Fairbairns has an option whereby sequential footnote marks in the text can be separated by commas^{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.

5.5 Changing series

5.5.1 Create a new series

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

5.5.2 Delete series

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

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

5.5.3 Series order

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

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

5.6 Position of critical and familiar footnotes

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

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

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

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

6 Critical apparatus appearance

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

When a length (*l*) is used, it can be stretchable: *a plus b minus c*. The final length *m* is calculated by \TeX to have $a - c \leq m \leq a + b$ units. If you use some relative unit¹⁶, it will be relative to the font size of the footnote, except for commands concerning the place kept by the notes, including blank spaces.

When a length, noted *l*, is used, it can be stretchable: *a plus b minus c*. The final length *m* is calculated by \TeX to have: $a - c \leq m \leq a + b$. If you use some relative unit¹⁷, it will be relative to font size of the footnote, except for commands concerning the place kept by the notes — including blank space.

There is also name convention:

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

6.1 Notes arrangement in a series

`\Xarrangement`
`\arrangementX`

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

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

The value of *a* can be one of the following

- `paragraph` formats all the footnotes of a series as a single paragraph. If you use this arrangement, you are strongly encouraged to read 17.5 p. 56.
- `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!¹⁸

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

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

¹⁸See XII.6.3 p. 149 for further information.

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

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

6.2 Control line number printing

6.2.1 Print line number only at first time

`\Xnumberonlyfirstinline`

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

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

`\Xnumberonlyfirstintwolines`

Suppose you have a lemma on line 2 and a lemma between line 2 and line 3. With `\Xnumberonlyfirstinline`, the second lemma is considered to be on the same line as the first lemma. But if you use both `\Xnumberonlyfirstinline[⟨s⟩]` and `\Xnumberonlyfirstintwolines[⟨s⟩]`, a distinction is made. Use the command `\Xnumberonlyfirstintwolines[⟨s⟩][false]` to disable this. `⟨s⟩` can be empty if you want to disable it for every series.

`\Xsymlinenum`

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

`\Xendnumberonlyfirstinline`

`\Xendnumberonlyfirstintwolines`

`\Xendsymlinenum`

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

6.2.2 Abbreviate line range

`\Xtwolines`

`\Xmorethantwolines`

If a lemma is printed on two subsequent lines, `reledmac` will print the first and the last line numbers. Instead of this, it is also possible to print an abbreviation which stands for “line 1 and subsequent line(s)”.

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

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

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

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

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

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

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

However, you can force print the final page number with

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

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

You can disable `\Xtwolines` and related for a specific note by using the ‘[fulllines]’ argument in the note macro cf. 5.2.2 p. 22.

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

`\Xendtwolinesbutnotmore`;

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

`\Xtwolinesbutnotmore`; `\Xtwolinesonlyinsamepage`.

6.2.3 Disable line number

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

6.2.4 Printing pstart number

You can use `\Xpstart[$\langle s \rangle$]` if you want to print the pstart number in the footnote, before the line and subline number. Use `\Xpstart[$\langle s \rangle$][false]` to disable this. $\langle s \rangle$ can be empty if you want to disable it for every series. Note that when you change the lineation system, the option is automatically switched :

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

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

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

6.2.5 Printing stanza number

`\Xstanza` You can use `\Xstanza[⟨s⟩]` if you want to print the stanza number in the footnote, before the line and subline number. Use `\Xstanza[⟨s⟩][false]` to disable this. `⟨s⟩` can be empty if you want to disable it for every series.

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

`\Xstanzaseparator`

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

6.2.6 Space around number

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

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

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

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

6.2.7 Space around line symbol

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

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

`\Xendbeforemsymlinenum` `\Xendaftersymlinenum` are the equivalents of `\Xbeforemsymlinenum` and `\Xaftersymlinenum` for the endnotes.

6.2.8 Space in place of number

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

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

6.2.9 Boxing line number and line symbol

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

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


`\Xboxlinenum{1em}`

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

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

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

L to align left (default value);

R to align right;

C to center.

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

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

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

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

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

6.3 Arbitrary code around line number

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

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

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

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

6.4 Separator between the lemma and the note

6.4.1 For footnotes

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

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

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

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

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

6.4.2 For endnotes

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

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

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

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

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

6.5 Font style

6.5.1 For line number

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

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

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

6.5.2 For the lemma

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

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

6.5.3 For all notes

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

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

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

6.6 Indent of notes content

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

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

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

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

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

6.7 Arbitrary code at the beginning of notes

The three next commands add arbitrary code at the beginning of notes. As the name's space is local to the notes, you can use it to redefine some style inside the notes. For

example, if you don't want the pstart number to be in bold, use :

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

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

6.8 Options for footnotes in columns

6.8.1 Alignment

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

`<code>` 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`).

6.8.2 Size of the columns

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

<pre>\Xhsizetwocol \Xsizethreecol \hsizetwocolX \hsizethreecolX</pre>	<pre>\Xhsizetwocol[⟨s⟩]{⟨l⟩}</pre> is used to change width of a column when critical notes are displaying in two columns. Defaut value is <code>.45 \hsizetwocol</code> . <pre>\Xsizethreecol[⟨s⟩]{⟨l⟩}</pre> is used to change width of a column when critical notes are displaying in three columns. Defaut value is <code>.3 \hsizetwocol</code> . <pre>\hsizetwocolX[⟨s⟩]{⟨l⟩}</pre> is used to change width of a column when familiar notes are displaying in two columns. Defaut value is <code>.45 \hsizetwocolX</code> . <pre>\hsizethreecolX[⟨s⟩]{⟨l⟩}</pre> is used to change width of a column when familiar notes are displaying in three columns. Defaut value is <code>.3 \hsizetwocolX</code> .
---	---

6.9 Options for paragraphed footnotes

6.9.1 Mark separation of notes

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

`\afternoteX`

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

`\parafootsepX` Note that if the symbol defined by `\Xsymlinenum` must be used at the beginning of a note, the `\Xparafootsep` / `\parafootsepX` is not used before this note.

6.9.2 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.

6.10 Options for block of notes

6.10.1 Text before notes

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

6.10.2 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.

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

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

6.10.3 Rule

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

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

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

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

6.10.4 Maximum height

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

`\maxhnotesX` `\maxhnotesX[⟨s⟩]{⟨l⟩}` is the same as previous, but for familiar footnotes.
Note that in many cases, you should call these commands, because the `\vsize` in the preamble is not the same as `\vsize` after the preamble. That why we recommend to you to add in your preamble

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

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

6.11 Footnotes and the `reledpar` columns

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

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

6.12 Endnotes in one paragraph

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

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

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

7 Fonts

One of the most important features of the appearance of the notes, and indeed of your whole document, will be the fonts used. We will first describe the commands that give you control over the use of fonts in the different structural elements of the document, especially within the notes, and then in subsequent sections specify how these commands are used.

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

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

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

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

`\endashchar` A relatively trivial matter relates to punctuation. In your footnotes, there will sometimes be spans of line numbers like this: 12–34, or lines with sub-line numbers like this: 55.6. The en-dash and the full stop are taken from the same font as the numbers, and it all works nicely. But what if you wanted to use old-style numbers, like 12 and 34? These look nice in an edition, but when you use the fonts provided by PLAIN T_EX they are taken from a math font which does not have the en-dash or full stop in the same places as a text font. If you (or your macros) just typed `$\oldstyle 12--34$` or `$\oldstyle 55.6$` you would get ‘12”34’ and ‘55>6’. So we define `\endashchar` and `\fullstop`, which produce an en-dash and a full stop respectively from the normal document font, whatever font you are using for the numbers. These two macros are used in the macros which format the line numbers in the margins and footnotes, instead of explicit punctuation. We also define an `\rbracket` macro for the right square bracket printed at the end of the lemma in many styles of textual notes (including `reledmac`’s standard style). For `polyglossia`, when the lemma is RTL, the bracket automatically switches to a left bracket.

`\fullstop`
`\rbracket`

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

`reledmac` uses `\select@lemmafонт` in a standard footnote format macro called

`\normalfootfmt`. The footnote formats for each of the layers A to E are `\let` equal to `\normalfootfmt`. So all the layers of the footnotes are formatted in the same way.

8 Verse

8.1 Basic

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

8.2 Define stanza indents

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

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

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

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

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

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

Enumeration is by stanza lines, not by print lines. In the above example the lines are indented one unit, two units, one unit, two units, with 3 units of hanging indentation in case a stanza line is too long to fit on one print line.

8.3 Repeating stanza indents

Since version 0.13, if the indentation is repeated every n verses of the stanza, you can define only the n first indentations, and indicate that they are repeated, defining the value of the `stanzaindentrepetition` counter at n . For example:

```
\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. 291.

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

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

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

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

8.4 Manual stanza indent

`\stanzaindent`
`\stanzaindent*`

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

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

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

8.5 Stanza breaking

`\setstanzapenalties`

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

The command

```
\setstanzapenalties{1,5000,10100,5000,0}
```

results in a penalty of 5000 being placed after the first and third lines of the stanza, and a penalty of −100 after the second.

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

8.6 Hanging symbol

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

`\sethangingsymbol`

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

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

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

8.7 Long verse and page break

If you want to prevent page breaks inside long verses, use the option `nopbinverse` when loading package, or use `\lednopbinversetrue`. Read 16.2 p. 54 for further details.

8.8 Content before/after verses

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

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

8.9 Numbering stanza

`\numberstanzatrue`
`\numberstanzafalse`

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

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

You can use this feature in combination with `\Xstanza` (6.2.5 p. 32).

`thestanza`

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

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

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

`\stanzanumwrapper`

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

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

8.10 Various tools

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

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

9 Grouping

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

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

You can also put familiar footnotes (see section 5.4) in a `minipage` but unlike with `\footnote` the numbering scheme is unaltered.

`ledgroup` Minipages, of course, are not broken across pages. Footnotes in a `ledgroup` environment are typeset at the end of the environment, as with `minipages`, but the environment includes normal page breaks. The environment makes no change to the `textwidth` so it appears as normal text; it just might be that footnotes appear in the middle of a page, with text above and below.

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

`\begin{ledgroupsize}[⟨pos⟩]{⟨width⟩}`.

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

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

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

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

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

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

10 Cross referencing

The package provides a simple cross-referencing facility that allows you to mark places in the text with labels, and generate page and line number references to those places

elsewhere using those labels.

10.1 Basic use

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

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

`\edlineref`

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

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

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

10.2 Refer to a critical notes

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

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

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

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

10.3 Cross-referencing which return a number in any case

`\xpageref` Where `#1` stands for the reference.

`\xlineref` However, there are situations in which you will want `reledmac` to return a number without displaying any warning messages about undefined labels or the like: if you want to use the reference in a context where \TeX is looking for a number, such a warning will lead to a complaint that the number is missing. This is the case for references used within the argument to `\linenum`, for example (see 5.2.5 p. 24).

`\xsublineref`

`\xpstartref`

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

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

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

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

`\xxref` The macros `\xxref` and `\edmakelabel` let you manipulate numbers and labels in ways which you may find helpful in tricky situations.

The `\xxref{<lab1>}{<lab2>}` command generates a reference to a sequence of lines, for use in the second argument of `\edtext`. It takes two arguments, both of which are labels: e.g., `\xxref{mouse}{elephant}`. It calls `\linenum` (q.v., 5.2.5 p. 24 above) and sets the beginning page, line and subline numbers to those of the place where `\edlabel{mouse}` was placed, and the ending numbers to those where `\edlabel{elephant}` occurs.

10.4 Not automatic cross-referencing

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

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

10.5 Normal L^AT_EX cross-referencing

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

10.6 References to lines commented in the apparatus

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

`\applabel` If you use `\applabel{<label>}` inside the second argument of a `\edtext`, `reledmac`

will add a `\edlabel` at the beginning and end of the marked passage. The label at the beginning of the passage will have the title `\label: start`, while the label at the end will have the title `\label: end`.

If you use `\linenum` (5.2.5 p. 24) to refer to these labels, `reledmac` will use your line settings to refer to the passage.

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

If you use `\setapprefprefixsingle{\prefix}`, `\prefix` will be printed before the line numbers of a `\appref`-reference. If you use `\setapprefprefixmore{\prefix}`, `\prefix` will be printed before the line numbers, if you refer to more than one line.

For example, you may use:

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

Note that if you have not used `\setapprefprefixmore` is empty, argument of `\setapprefprefixsingle` will be used in any case.

If you use `\Xtwo lines`, `\Xmore than two lines`, `\Xtwo lines but not more` and/or `\Xtwo lines only in same page` (6.2.2 p. 30) *without the optional series argument*, the setting will also be available for `\appref`.

The commands `\Xtwo lines appref{\text}`, `\Xmore than two lines appref{\text}`, `\Xtwo lines but not more appref` `\Xtwo lines only in same page appref` can also be used, if you only want to change the reference style of `\appref`.

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

If you use one of `\Xend two lines`, `\Xend more than two lines`, `\Xend two lines but not more`, `\Xend more than two lines apprefwithpage` `\Xend two lines only in same page` (6.2.2 p. 31) *without the optional series argument*, the setting will also be available for `\apprefwithpage`.

The commands `\Xend two lines appref{\text}`, `\Xend more than two lines appref{\text}`, `\Xend two lines but not more appref`, `\Xend two lines only in same page appref` can also be used, if you only want to change the reference style of `\apprefwithpage`.

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

11 Side notes

11.1 Basics

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

`\ledinnernote{\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
`\ledrightnote`
`\ledsidenote`
`\sidenotemargin`

setting of `\sidenotemargin{⟨location⟩}`. The permissible value for `⟨location⟩` is one out of the list `left`, `right`, `inner`, or `outer`, for example `\sidenotemargin{outer}`. The package's default setting is

```
\sidenotemargin{right}
```

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

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

11.2 Setting

11.2.1 Width

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

11.2.2 Vertical position

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

11.2.3 Distance to the main text

`\ledlsnotesep` The texts are put a distance `\ledlsnotesep` (or `\ledrsnotesep`) into the left (or right)
`\ledrsnotesep` margin. These lengths are initially set to the value of `\linenumsep`.
`\ledlsnotefontsetup` These macros specify how the sidenote texts are to be typeset. The initial definitions
`\ledrsnotefontsetup` are:

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

These can of course be changed to suit.

11.2.4 Separator between notes

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

12 Indexing

12.1 Basics

`\edindex` L^AT_EX provides the `\index{⟨item⟩}` command for specifying that `⟨item⟩` and the current page number should be added to the raw index (`idx`) file. The `\edindex{⟨item⟩}` macro

can be used in numbered text to specify that $\langle item \rangle$ and the current page & linenumber should be added to the raw index file.

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

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

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

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

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

12.2 Separator between page and line numbers

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

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

Consequently, if you want to use an other `\pagelinesep`, you have to configure your `.ist` index style file. For example if you use `:` as separator²⁰.

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

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

12.3 Using xindy

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

`pagenumber-linenummer`

An example of such a file is provided in the “examples” folder. Read the `xindy` handbook to learn how to use it.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.

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>.

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

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

1. Use `xindy+hyperref` option when loading the `reledmac` package. When you run (Xe/Lua) \TeX with this option, a `.xdy` configuration file will be generated with all the settings needed to allow internal hyperlinking in each index entry which is created by `\edindex`.
2. Use `hyperindex=false` option when loading `hyperref`.
3. Uncomment — by removing the semicolons at the beginning of the relevant lines — some lines in the `<code>.xdy</code>` file provided in the “examples” folder in order to restore internal links in the index to be used by the standard `index` command.²²

12.4 Advanced setting

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

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

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

13 Tabular material

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

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

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

1	2	3
a	bb	ccc
AAA	BB	C

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

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

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

1	2	3	4
	$F + G + C$		
<i>a</i>	<i>bb</i>	<i>ccc</i>	<i>dddd</i>

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

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

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

```
\begin{edtabularr}
```

```

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

```

1	2	3	4	5
Q		fd	h	qwertziohg
v	wptz	x	y	vb
g	nnn	$\overbrace{\hspace{10em}}$		
	$\underbrace{\hspace{10em}}$		pq	dgh
k		l	co	ghweropjklmnbvcxys
1	2	3	$\rule{10em}{0.4pt}$	

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

```

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

```

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

```

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

```

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

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

```

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

```

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

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

For example:

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

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

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

`\edvertdots`

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

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

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

14 Sectioning commands

14.1 Sectioning commands without line numbers or critical notes

The standard sectioning commands (`\chapter`, `\section` etc.) can be used inside numbered text. In this case, you must call them as an optional argument of `\pstart` (4.2.3

p. 16):

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

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

14.2 Sectioning commands with line numbering and critical notes

You have to use the following commands:

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

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

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

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

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

14.3 Optimization

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

15 Quotation environments

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

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

16 Page breaks

16.1 Control page breaking

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

`\ledpb`
`\lednopb`

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

These commands have effect only at the second run.

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

`\ledpbsetting`

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

16.2 Prevent page break in a long verses

`\lednopbinversettrue`

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

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

17 Miscellaneous

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

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

`\showlemma` The lemma within the text is printed via `\showlemma{lemma}`. Normally, or with the ‘final’ option, the definition of `\showlemma` is:
`\newcommand*{\showlemma}[1]{#1}`
 so it just produces its argument. With the ‘draft’ option it is defined as
`\newcommand*{\showlemma}[1]{\textit{#1}}`
 so that its argument is typeset in an italic font, which may make it easier to check that all lemmas have been treated.

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

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

17.1 Known and suspected limitations

17.2 ‘No room for a new’

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

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

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

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

‘no room for a new write’ can be caused by with multiple indexes. In this case, use `indextools` of `imakeidx` with the `splitindex` option, in order to obtain only

one .idx file. If that does not solve your problem, you can use `morewrites` package. That should solve the problem, but \TeX will be slower.

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

17.3 Marginal notes

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

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

17.4 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.

17.5 Paragraphed footnotes

The restriction on explicit line-breaking in paragraphed footnotes, mentioned in a footnote ?? p. ??, and described in more detail on XII.6.3 p. 148, really is a nuisance if that is something you need to do. There are some possible solutions, described by Michael Downes, but this area remains unsatisfactory.

`\footfudgefiddle` For paragraphed footnotes \TeX has to estimate the amount of space required. If it underestimates this then the notes may get too long and run off the bottom of the text block. `\footfudgefiddle` can be increased from its default 64 (say, to 68) to increase the estimate. You have to use `\renewcommand` for this, like:
`\renewcommand{\footfudgefiddle}{68}` Note that you must call it *before* `\Xarrangement{paragraph}` or `\arrangementX{paragraph}`.

Any settings to 'geometry' must be made before `\Xarrangement` / `\arrangementX`.

Finally, in many cases you should use `\Xmaxhnotes` and / or `\maxhnotesX` (6.10.4 p. 38), in order to define the maximum height relative to `\textheight` and not to `\vsize`, because the `\vsize` value is not the same inside and outside of the preamble.

17.6 Use with other packages

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

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

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

If you actually try 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.

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

17.7 Parallel typesetting

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

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

I Implementation overview

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

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

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

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

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

II Preliminaries

II.1 Links with original `edmac`

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

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

II.2 Package declaration

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

```

1 %<*code>
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{reledmac}[2015/10/05 v2.2.1 typeset critical edition]%
4 %

```

II.3 Package options

```

\ifledfinal Use this to remember which option is used, set and execute the options with final as the
\ifnocritical@ default. We use xkeyval in order to manage options with argument.
\if@noeled@sec \RequirePackage{xkeyval}
\ifnoend@ %
\ifnofamiliar@
\ifnoledgroup@ The parledgroup option is for reledpar. However, it has consequence on reledmac
\ifparapparatus@ internal command. So we need to define the boolean now.
\ifnoquotation@ \newif\ifparledgroup
\iflednopbinverse %
\ifparledgroup
\ifwidthliketwocolumns And now, the options of reledmac.
\ifxindy@ \DeclareOptionX{series}[A,B,C,D,E]{\xdef\default@series{#1}}
\ifxindyhyperref@ \ExecuteOptionsX{series}%
\ifeledmaccompat@
12 \newif\if@noeled@sec%
13 \DeclareOptionX{noeledsec}{\@noeled@sectrue}
14
15 \newif\ifnocritical@%
16 \DeclareOptionX{nocritical}{\nocritical@true}%
17
18 \newif\ifnofamiliar@%
19 \DeclareOptionX{nofamiliar}{\nofamiliar@true}%
20
21 \newif\ifnoledgroup@%
22 \DeclareOptionX{noledgroup}{\noledgroup@true}%
23
24 \newif\ifnoend@%
25 \DeclareOptionX{noend}{%
26 \let\l@dend@open\@gobble%
27 \let\l@dend@close\relax%
28 \global\let\l@dend@stuff=\relax%
29 \noend@true%
30 }%
31
32 \newif\ifnoquotation@
33 \DeclareOptionX{noquotation}{\noquotation@true}
34
35
36 \newif\ifledfinal
37 \DeclareOptionX{final}{\ledfinaltrue}
38 \DeclareOptionX{draft}{\ledfinalfalse}

```

```

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

```

We use the starred form of `\ProcessOptionsX` which executes options in the order listed in the source file: class options, then listed package options, so a package option can override a class option with the same name. This was suggested by Dan Luecking in the `ctt` thread *Class/package option processing*, on 27 February 2004.

```

69 \ProcessOptionsX*\relax
70
71 %

```

II.4 Loading packages

Loading package `xargs` to declare commands with optional arguments. `Etoolbox` is also used to make code clearer - for example, in dynamic command names (which can replace `\csname` etc.). Use suffix to declare commands with a starred version, `xstring` to work with strings, `ifluatex` and `ifxetex` to test if `LuaTeX` or `XYTeX` is running, and `ragged2e` to manage ragged justification for paragraphed notes.

```

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

```

```

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

```

II.5 Compatibility with Lua_T_EX

Here, we enable some primitives for Lua_T_EX.

```

83 \ifx\directlua\undefined\else%
84   \directlua{tex.enableprimitives("",{"texdir","pdir","bodydir"})}
85 \fi
86 %

```

II.6 Boolean flags

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

```

87 \newif\ifl@dmemoir
88 \ifclassloaded{memoir}{\l@dmemoirtrue}{\l@dmemoirfalse}
89
90 %

```

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

```

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

```

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

```

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

```

False is the default value. We consider indextools as a variant of imakeidx. That is why we set \ifl@imakeidx to true. We also let \imki@wrindexentry to \indtl@wrindexentry.

\if@RTL The \if@RTL is defined by the bidi package, which is sometimes loaded by *polyglossia*. But we define it as well if the bidi package is not loaded.

```

101 \ifdef{\if@RTL}{\newif\if@RTL}
102 %

```

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.

```

103 \newcommand{\reledmac@warning}[1]{\PackageWarning{reledmac}{#1}}
104 %

```

`\reledmac@error` Write an error message.

```

105 \newcommand{\reledmac@error}[2]{\PackageError{reledmac}{#1}{#2}}
106 %

```

```

\led@err@NumberingStarted 107 \newcommand*\led@err@NumberingStarted}{%
d@err@NumberingNotStarted 108 \reledmac@error{Numbering has already been started}{\@ehc}}
NumberingShouldHaveStarted 109 \newcommand*\led@err@NumberingNotStarted}{%
110 \reledmac@error{Numbering was not started}{\@ehc}}
111 \newcommand*\led@err@NumberingShouldHaveStarted}{%
112 \reledmac@error{Numbering should already have been started}{\@ehc}}
113 %

```

```

d@err@edtextoutsidestart 114 \newcommand*\led@err@edtextoutsidestart}{%
115 \reledmac@error{\string\edtext\space outside numbered paragraph (\pstart\
ldots\pend)}{\@ehc}}%
116 %

```

```

\led@mess@NotesChanged 117 \newcommand*\led@mess@NotesChanged}{%
118 \typeout{reledmac reminder: }%
119 \typeout{ The number of the footnotes in this section
120 has changed since the last run.}%
121 \typeout{ You will need to run LaTeX two more times
122 before the footnote placement}%
123 \typeout{ and line numbering in this section are
124 correct.}}
125 %

```

```

\led@mess@SectionContinued 126 \newcommand*\led@mess@SectionContinued}[1]{%
127 \message{Section #1 (continuing the previous section)}}
128 %

```

```

\led@err@LineationInNumbered29 \newcommand*\led@err@LineationInNumbered}{%
130 \reledmac@error{You can't use \string\lineation\space within
131 a numbered section}{\@ehc}}
132 %

\led@warn@BadLineation33 \newcommand*\led@warn@BadLineation}{%
\led@warn@BadLinenummargin34 \reledmac@warning{Bad \string\lineation\space argument}}
\led@warn@BadLockdisp35 \newcommand*\led@warn@BadLinenummargin}{%
\led@warn@BadSublockdisp36 \reledmac@warning{Bad \string\linenummargin\space argument}}
137 \newcommand*\led@warn@BadLockdisp}{%
138 \reledmac@warning{Bad \string\lockdisp\space argument}}
139 \newcommand*\led@warn@BadSublockdisp}{%
140 \reledmac@warning{Bad \string\sublockdisp\space argument}}
141 %

\led@warn@NoLineFile42 \newcommand*\led@warn@NoLineFile}[1]{%
143 \reledmac@warning{Can't find line-list file #1}}
144 %

\led@warn@LineFileObsolete45 \newcommand*\led@warn@Obsolete}[1]{%
146 \reledmac@warning{Line-list file #1 was obsolete. We have not read it.
Please run LaTeX again.}}
147 %

\led@warn@BadAdvancelineSubline48 \newcommand*\led@warn@BadAdvancelineSubline}{%
\led@warn@BadAdvancelineLine49 \reledmac@warning{\string\advanceline\space produced a sub-line
150 number less than zero.}}
151 \newcommand*\led@warn@BadAdvancelineLine}{%
152 \reledmac@warning{\string\advanceline\space produced a line
153 number less than zero.}}
154 %

\led@warn@BadSetline55 \newcommand*\led@warn@BadSetline}{%
\led@warn@BadSetlinenum56 \reledmac@warning{Bad \string\setline\space argument}}
157 \newcommand*\led@warn@BadSetlinenum}{%
158 \reledmac@warning{Bad \string\setlinenum\space argument}}
159 %

\led@err@PstartNotNumbered60 \newcommand*\led@err@PstartNotNumbered}{%
\led@err@PstartInPstart61 \reledmac@error{\string\pstart\space must be used within a
\led@err@PendNotNumbered62 numbered section}{\@ehc}}
\led@err@PendNoPstart63 \newcommand*\led@err@PstartInPstart}{%
\led@err@AutoparNotNumbered64 \reledmac@error{\string\pstart\space encountered while another
\led@err@NumberingWithoutPstart

```



```

165 \string\pstart\space was in effect}{\@ehc}}
166 \newcommand*{\led@err@PendNotNumbered}{%
167 \reledmac@error{\string\pend\space must be used within a
168 numbered section}{\@ehc}}
169 \newcommand*{\led@err@PendNoPstart}{%
170 \reledmac@error{\string\pend\space must follow a \string\pstart}{\@ehc}}
171 \newcommand*{\led@err@AutoparNotNumbered}{%
172 \reledmac@error{\string\autopar\space must be used within a
173 numbered section}{\@ehc}}
174 \newcommand*{\led@err@NumberingWithoutPstart}{%
175 \reledmac@error{\string\beginnumbering...\string\endnumbering\space
without \string\pstart}{\@ehc}}%
176 %

```

```

\led@warn@BadAction 177 \newcommand*{\led@warn@BadAction}{%
178 \reledmac@warning{Bad action code, value \next@action.}}
179 %

```

```

\led@warn@DuplicateLabel 180 \newcommand*{\led@warn@DuplicateLabel}[1]{%
\led@warn@AppLabelOutEdtext 181 \reledmac@warning{Duplicate definition of label `#1' on page \the\pageno
\led@warn@RefUndefined .}}
182 \newcommand*{\led@warn@AppLabelOutEdtext}[1]{%
183 \reledmac@warning{\string\applabel\space outside of \string\edtext\space
`#1' on page \the\pageno.}}%
184 \newcommand*{\led@warn@RefUndefined}[1]{%
185 \reledmac@warning{Reference `#1' on page \the\pageno\space undefined.
186 Using `000'.}}
187 %

```

```

\led@warn@NoMarginpars 188 \newcommand*{\led@warn@NoMarginpars}{%
189 \reledmac@warning{You can't use \string\marginpar\space in numbered text
}}
190 %

```

```

\led@warn@BadSidenotemargin 191 \newcommand*{\led@warn@BadSidenotemargin}{%
192 \reledmac@warning{Bad \string\sidenotemmargin\space argument}}
193 %

```

```

\led@warn@NoIndexFile 194 \newcommand*{\led@warn@NoIndexFile}[1]{%
195 \reledmac@warning{Undefined index file #1}}
196 %

```

```

\led@warn@SeriesStillExist97 \newcommand{\led@warn@SeriesStillExist}[1]{%
198   \reledmac@warning{Series #1 is still existing !}%
199 }%
200 %

\led@err@ManySidenotes01 \newcommand{\led@err@ManySidenotes}{%
\led@err@ManyLeftnotes02   \ifledRcol{%
\led@err@ManyRightnotes03   \reledmac@warning{\itemcount@\space sidenotes on line \the\line@numR\
space p. \the\page@numR}%
204   \else%
205   \reledmac@warning{\itemcount@\space sidenotes on line \the\line@num\
space p. \the\page@num}%
206   \fi%
207 }%
208 \newcommand{\led@err@ManyLeftnotes}{%
209   \ifledRcol{%
210   \reledmac@warning{\itemcount@\space leftnotes on line \the\line@numR\
space p. \the\page@numR}%
211   \else%
212   \reledmac@warning{\itemcount@\space leftnotes on line \the\line@num\
space p. \the\page@num}%
213   \fi%
214 }%
215 \newcommand{\led@err@ManyRightnotes}{%
216   \ifledRcol{%
217   \reledmac@warning{\itemcount@\space rightnotes on line \the\line@numR\
space p. \the\page@numR}%
218   \else%
219   \reledmac@warning{\itemcount@\space rightnotes on line \the\line@num\
space p. \the\page@num}%
220   \fi%
221 }%
222 %

\led@err@TooManyColumns23 \newcommand*\led@err@TooManyColumns{%
\led@err@UnequalColumns24   \reledmac@error{Too many columns}{\@ehc}}
\led@err@LowStartColumn25 \newcommand*\led@err@UnequalColumns{%
\led@err@HighEndColumn26   \reledmac@error{Number of columns is not equal to the number
\led@err@ReverseColumns27   in the previous row (or \protect\\ \space forgotten?)}{\@ehc}}
228 \newcommand*\led@err@LowStartColumn{%
229   \reledmac@error{Start column is too low}{\@ehc}}
230 \newcommand*\led@err@HighEndColumn{%
231   \reledmac@error{End column is too high}{\@ehc}}
232 \newcommand*\led@err@ReverseColumns{%
233   \reledmac@error{Start column is greater than end column}{\@ehc}}
234 %

```

```

err@EdtextWithoutFootnote35 \newcommand{\led@err@EdtextWithoutFootnote}{%
236 \reledmac@error{edtext without Xfootnote. Check syntaxis.}\@ehc}%
237 }%
238 %

err@FootnoteWithoutEdtext39 \newcommand{\led@err@FootnoteWithoutEdtext}{%
240 \reledmac@error{Xfootnote without edtext. Check syntax.}\@ehc}%
241 }%
242 %

error@ImakeidxAfterEledmac43 \newcommand{\led@error@ImakeidxAfterEledmac}{%
244 \reledmac@error{Imakeidx must be loaded before reledmac.}\@ehc}%
245 }%
246 %

or@IndextoolsAfterEledmac47 \newcommand{\led@error@IndextoolsAfterEledmac}{%
248 \reledmac@error{Indextools must be loaded before reledmac.}\@ehc}%
249 }%
250 %

error@fail@patch@@makecol51 \newcommand{\led@error@fail@patch@@makecol}{%
252 \reledmac@error{Fail to patch \string\@makecol\space command.}\@ehc}%
253 }%
254 %

ror@fail@patch@@reinserts55 \newcommand{\led@error@fail@patch@@reinserts}{%
256 \reledmac@error{Fail to patch \string\@reinserts\space command.}\@ehc}%
257 }%
258 %

r@fail@patch@@doclearpage59 \newcommand{\led@error@fail@patch@@doclearpage}{%
260 \reledmac@error{Fail to patch \string\@doclearpage\space command.}\@ehc}
%
261 }%
262 %

r@fail@patch@@iiiminipage63 \newcommand{\led@error@fail@patch@@iiiminipage}{%
264 \reledmac@error{Fail to patch \string\@iiiminipage\space command.}\@ehc}
%
265 }%
266 %

```

```

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

```

II.8 Gobbling

Here, we define some commands which gobble their arguments.

```

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

```

II.9 Miscellaneous commands

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

```

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

```

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

Initialize it as `\empty`.

```

282 \let\linenumberlist=\empty
283
284 %

```

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

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

```

285 \newcount\@l@tempcnta \newcount\@l@tempcntb
286 %

```

II.10 Prepare reledpar

`\ifnumberingR` 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@dpairing`

`\ifl@dpaging`

`\l@dpagingtrue` 287 `\newif\ifl@dpairing`

`\l@dpagingfalse` 288 `\newif\ifl@dpaging%`

`\ifl@dprintingpages` 289 `\newif\ifl@dprintingpages%`

`\l@dprintingpagestrue` 290 `\newif\ifl@dprintingcolumns%`

`\l@dprintingpagesfalse` 291 `\newif\ifpst@rtedL`

`\ifl@dprintingcolumns` 292 `\newcount\l@dnumpstartsL`

`\l@dprintingcolumnstrue` 293 `%`

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

`\l@dpairingtrue`

`\l@dpairingfalse`

`\ifpst@rtedL`

`\pst@rtedLtrue` 294 `\newif\ifledRcol`

`\pst@rtedLfalse` 295 `\newif\ifledRcol@`

`\l@dnumpstartsL` 296 `%`

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

`\ifledRcol@` 297 `\newif\ifnumberingR`

298 `%`

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 `\resumenummering` 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.

299 `\newcount\section@num`

300 `\section@num=0`

301 `\let\extensionchars=\empty`

302 `%`

`\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.

```
303 \newif\ifnumbering
304 %
```

`\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.

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

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.

```

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

```

\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.

```

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

```

```

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

```

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

```

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

```

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.

```

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

```

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


```

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

```

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.

```

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

```

`\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 .

```

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

```

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

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

`\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`.

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

`\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.

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

V Line counting

V.1 Choosing the system of lineation

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

`\ifbypstart@` The `\ifbypage@` and `\ifbypstart@` flag specify the current lineation system:
`\bypstart@true` • line-of-page: `bypstart@ = false` and `bypage@ = true`.
`\bypstart@false` • line-of-pstart: `bypstart@ = true` and `bypage@ = false`.
`\ifbypage@`
`\bypage@true` reledmac will use the line-of-section system unless instructed otherwise.
`\bypage@false`

```
440 \newif\ifbypage@
441 \newif\ifbypstart@
442 %
```

The `\ifbypage@R` and `\ifbypstart@R` flag specify the current lineation for right side in case of using `reledpar`. They are now defined because they are used in some specific code. `reledpar` will use the line-of-section system unless instructed otherwise.

```
\ifbypage@R43 \newif\ifbypage@R
\ifbypstart@R44 \newif\ifbypstart@R
445 %
```

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

```
446 \newcommand*{\lineation}[1]{\{
447 %
```

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

```
448 \ifnumbering
449 \led@err@LineationInNumbered
450 \else
451 %
```

If the argument is `page`.

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

If the argument is `pstart`.

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

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

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

```

476 \fi%
477 %

```

In other case, it is an error.

```

478 \else
479 \led@warn@BadLineation
480 \fi
481 \fi
482 \fi
483 \fi}}
484 %

```

V.2 Line number margin

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

```

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

```

```

515         \@l@dttempcntb \m@ne
516         \fi
517     \fi
518 \fi
519 \fi}
520
521 %

```

V.3 Line number initialization and increment

`\c@firstlinenum` The following counters tell reledmac which lines should be printed with line numbers. `firstlinenum` is the number of the first line in each section that gets a number; `linenumincrement` is the difference between successive numbered lines. The initial values of these counters produce labels on lines 5, 10, 15, etc. `linenumincrement` must be at least 1.

```

522 \newcounter{firstlinenum}
523 \setcounter{firstlinenum}{5}
524 \newcounter{linenumincrement}
525 \setcounter{linenumincrement}{5}
526 %

```

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

```

527 \newcounter{firstsublinenum}
528 \setcounter{firstsublinenum}{5}
529 \newcounter{sublinenumincrement}
530 \setcounter{sublinenumincrement}{5}
531
532 %

```

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

```

533 \newcommand*{\firstlinenum}[1]{%
534     \ifledRcol%
535         \setcounter{firstlinenumR}{#1}%
536     \else%
537         \setcounter{firstlinenum}{#1}%
538     \fi%
539 }
540
541 \newcommand*{\linenumincrement}[1]{%
542     \ifledRcol%
543         \setcounter{linenumincrementR}{#1}%
544     \else%
545         \setcounter{linenumincrement}{#1}%
546     \fi%
547 }

```

```

548 \newcommand*\firstsublinenum}[1]{%
549   \ifledRcol%
550     \setcounter{firstsublinenumR}{#1}%
551   \else%
552     \setcounter{firstsublinenum}{#1}%
553   \fi%
554 }
555 \newcommand*\sublinenumincrement}[1]{%
556   \ifledRcol%
557     \setcounter{sublinenumincrementR}{#1}%
558   \else%
559     \setcounter{sublinenumincrement}{#1}%
560   \fi%
561 }
562
563 %

```

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.

```

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

```

```

588 \fi}
589
590 %

```

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

```

591 \newcount\sublock@disp
592 \newcommand{\sublockdisp}[1]{\%
593   \l@getlock@disp{#1}%
594   \ifnum\l@dttempcntb>\m@ne
595     \global\sublock@disp=\l@dttempcntb
596   \else
597     \led@warn@BadSublockdisp
598   \fi}}
599
600 %

```

V.5 Line number style

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

`\linenumrep`
`\linenumr@p`

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

`\sublinenumberstyle`
`\sublinenumrep`
`\sublinenumr@p`

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

```

601 \newcommand*{\linenumberstyle}[1]{\%
602   \def\linenumrep##1{\@nameuse{#1}{##1}}}
603 \newcommand*{\sublinenumberstyle}[1]{\%
604   \def\sublinenumrep##1{\@nameuse{#1}{##1}}}
605 %

```

Initialise the number styles to arabic.

```

606 \linenumberstyle{arabic}
607 \let\linenumr@p\linenumrep
608 \sublinenumberstyle{arabic}
609 \let\sublinenumr@p\sublinenumrep
610
611 %

```

V.6 Line number printing

`\leftlinenum` `\leftlinenum` and `\rightlinenum` are the macros that are called to print marginal line numbers on a page, for left- and right-hand margins respectively. They are made easy to access and change, since you may want to change the styling in some way. These

`\rightlinenum`
`\linenumsep`
`\numlabfont`
`\ledlinenum`

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.

```

612 \newlength{\linenumsep}
613 \setlength{\linenumsep}{1pc}
614 \newcommand*{\numlabfont}{\normalfont\scriptsize}
615 \newcommand*{\ledlinenum}{%
616   \bgroup%
617   \ifluatex%
618     \texdir TLT%
619   \fi%
620   \numlabfont\linenumrep{\line@num}%
621   \ifsublines@
622     \ifnum\subline@num>0\relax
623       \unskip\fullstop\sublinenumrep{\subline@num}%
624     \fi
625   \fi%
626   \egroup%
627 }%
628
629 \newcommand*{\leftlinenum}{%
630   \ledlinenum
631   \kern\linenumsep}
632 \newcommand*{\rightlinenum}{%
633   \kern\linenumsep
634   \ledlinenum}
635
636 %

```

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`.

```
637 \newcount\line@num
638 %
```

`\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.

```
639 \newcount\subline@num
640 %
```

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

`\sublines@true` You may wonder why we do not just use the value of `\subline@num` to determine this—treating anything greater than 0 as an indication that sub-lineation is on. We need a separate flag because sub-lineation can be used together with line-number locking in odd ways: several pieces of a logical line might be interrupted by pieces of sub-lineated text, and those sub-line numbers should not return to zero until the next change in the major line number. This is common in the typesetting of English Renaissance verse drama, in which stage directions are given sub-line numbers: a single line of verse may be interrupted by several stage directions.

`\sublines@false`

```
641 \newif\ifsublines@
642 %
```

`\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.

```
643 \newcount\absline@num
644 %
```

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.

```
645 \newcount\@lock
646 \newcount\sub@lock
647 %
```

V.9 Line number associated to lemma

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

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

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

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

```
23|35|0|24|3|0|OT1/cmr/m/n.
```

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

- `\insertlines@list`: the line numbers of lines that have footnotes or other insertions. These are the absolute numbers where the corresponding lemmas begin. This list contains one entry for every footnote in the section; one lemma may contribute no footnotes or many footnotes. This list is used by `\add@inserts` within `\do@line`, to tell it where to insert notes.
- `\actionlines@list`: a list of absolute line numbers at which we are to perform special actions; these actions are specified by the `\actions@list` list defined below.
- `\actions@list`: action codes corresponding to the line numbers in `\actionlines@list`. These codes tell `reledmac` what action it is supposed to take at each of these lines. One action, the page-start action, is generated behind the scenes by `reledmac`

itself; the others, for specifying sub-lineation, line-number locking, and line-number alteration, are generated only by explicit commands in your input file. The page-start and line-number-alteration actions require arguments, to specify the new values for the page or line numbers; instead of storing those arguments in another list, we have chosen the action-code values so that they can encode both the action and the argument in these cases. Action codes greater than -1000 are page-start actions, and the code value is the page number; action codes less than -5000 specify line numbers, and the code value is a transformed version of the line number; action codes between these two values specify other actions which require no argument.

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

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

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

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

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

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

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

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

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

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

```
648 \list@create{\line@list}
649 \list@create{\insertlines@list}
650 \list@create{\actionlines@list}
651 \list@create{\actions@list}
652
653 %
```

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

```
\endpage@num
\endline@num
\endsubline@num
654 \newcount\page@num
655 \newcount\endpage@num
656 \newcount\endline@num
657 \newcount\endsubline@num
658 %
```

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

```
659 \newif\ifnoteschanged@
660 %
```

`\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@ 661 \newcommand*{\resetprevline@}{%
662 \def\do##1{\global\csundef{prevline##1}}}%
663 \dolistloop{\@series}%
664 }
665 %
```

`\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@ 66 \newcommand*{\resetprevpage@num}{%
667   \def\do##1{\ifcsdef{prevpage##1@num}{\global\csname prevpage##1@num\
endcsname=0}{}}%
668   \dolistloop{\@series}%
669 }
670 %

```

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.

```

671 \newread\@inputcheck
672 \newcommand*{\read@linelist}[1]{%
673   \ifledRcol%
674   \list@clearing@regR%
675   \else%
676   \list@clearing@reg%
677   \fi%
678 %

```

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).

```

679 \list@clear{\maxlinesinpar@list}
680 %

```

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

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

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

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

```

681 \get@linelistfile{#1}%
682 \endgroup
683 %

```

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.

```

684 \ifledRcol
685   \global\page@numR=\m@ne
686   \ifx\actionlines@listR\empty
687     \gdef\next@actionlineR{1000000}%
688   \else
689     \glp\actionlines@listR\to\next@actionlineR
690     \glp\actions@listR\to\next@actionR
691   \fi
692 \else
693   \global\page@num=\m@ne
694   \ifx\actionlines@list\empty
695     \gdef\next@actionline{1000000}%
696   \else
697     \glp\actionlines@list\to\next@actionline
698     \glp\actions@list\to\next@action
699   \fi
700 \fi
701 }
702 %

```

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

```

703 \newcommand*{\list@clearing@reg}{%
704   \list@clear{\line@list}%
705   \list@clear{\insertlines@list}%
706   \list@clear{\actionlines@list}%
707   \list@clear{\actions@list}%
708   \list@clear{\linesinpar@listL}%
709   \list@clear{\linesonpage@listL}%
710 }%
711 %

```

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

```

712 \newcommand*{\get@linelistfile}[1]{%
713   \InputIfFileExists{#1}{%
714     \global\noteschanged@false
715     \begingroup
716       \catcode`\[=1 \catcode`\]=2
717       \makeatletter \catcode`\^M=9}{%

```

```

718 \led@warn@NoLineFile{#1}%
719 \global\noteschanged@true
720 \begingroup}%
721 }
722
723 %

```

This version of `\read@linelist` creates list macros containing data for the entire section, so they could get rather large. It would be no more difficult to read the line-list file incrementally rather than all at once: we could read, at the start of each paragraph, only the commands relating to that paragraph. But this would require that we have two line-lists open at once, one for reading, one for writing, and on systems without version numbers we would have to do some file renaming outside of \TeX for that to work. We have retained this slower approach to avoid that sort of hacking about, but have provided the `\pausenumbers` and `\resumenumbers` macros to help you if you run into macro memory limitations (see 4.2.7 p. 17 above).

V.11 Commands within the line-list file

This section defines the commands that can appear within a line-list file. They all have very short names because we are likely to be writing very large numbers of them out. One macro, `\@nl`, is especially short, since it will be written to the line-list file once for every line of text in a numbered section. (Another of these commands, `\@lab`, will be introduced in a later section, among the cross-referencing commands it is associated with.)

When these commands modify the various page and line counters, they deliberately do not use `\global`. This is because we want them to affect only the counter values within the current group when nested calls of `\@ref` occur. (The code assumes throughout that the value of `\globaldefs` is zero.)

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

`\line@list@version` The `\line@list@version` check if the line-list file does not 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.

```

724 \newcommand{\line@list@version}[1]{%
725   \IfStrEq{#1}{\this@line@list@version}%
726   {}%
727   {\ifledRcol%
728     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
729     \else%
730     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
731     \fi%
732   \endinput%
733   }%
734 }%
735 %

```

`\@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`.

```

736
737 \newcommand*{\@nl}[2]{%
738   \fix@page{#1}%
739   \ifledRcol%
740     \@nl@regR%
741   \else%
742     \@nl@reg%
743   \fi%
744 }
745 \newcommand*{\@nl@reg}{%
746   \ifx\l@dchset@num\relax \else
747     \advance\absline@num \@ne
748     \set@line@action
749     \let\l@dchset@num=\relax
750     \advance\absline@num \m@ne
751     \advance\line@num \m@ne
752   \fi
753 %

```

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

```

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

```



```

767 \fi
768 %

```

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

```

769 \ifcase\@lock
770 \or
771 \@lock \tw@
772 \or \or
773 \@lock \z@
774 \fi
775 \ifcase\sub@lock
776 \or
777 \sub@lock \tw@
778 \or \or
779 \sub@lock \z@
780 \fi
781 %

```

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

```

782 \ifsublines@
783 \ifnum\sub@lock<\tw@
784 \advance\subline@num \@ne
785 \fi
786 \else
787 \ifnum\@lock<\tw@
788 \advance\line@num \@ne \subline@num \z@
789 \fi
790 \fi}
791 %
792

```

`\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`.

```

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

```

```

808 \ifnum #1=\last@page@num
809 \else
810 \ifbypage@
811 \line@num \z@ \subline@num \z@
812 \fi
813 \page@num=#1\relax
814 \last@page@num=#1\relax
815 \def\next@page@num{#1}%
816 \listxadd{normal@page@break}{\the\absline@num}
817 \fi
818 \fi}
819 %

```

\@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.

```

820 \@lopR \newcommand*{\@pend}[1]{}
821 \newcommand*{\@pendR}[1]{}
822 \newcommand*{\@lopL}[1]{}
823 \newcommand*{\@lopR}[1]{}
824
825 %

```

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

```

826 \newcommand*{\sub@on}{\ifsublines@
827 \let\sub@change=\relax
828 \else
829 \def\sub@change{1}%
830 \fi}
831 \newcommand*{\sub@off}{\ifsublines@
832 \def\sub@change{-1}%
833 \else
834 \let\sub@change=\relax
835 \fi}
836
837 %

```

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

```

838
839 \newcommand*{\@adv}[1]{%
840 \ifsublines@
841 \ifledRcol
842 \advance\subline@numR by #1\relax
843 \ifnum\subline@numR<\z@

```

```

844     \led@warn@BadAdvancelineSubline
845     \subline@numR \z@
846     \fi
847   \else
848     \advance\subline@num by #1\relax
849     \ifnum\subline@num<\z@
850       \led@warn@BadAdvancelineSubline
851       \subline@num \z@
852     \fi
853   \fi
854 \else
855   \ifledRcol
856     \advance\line@numR by #1\relax
857     \ifnum\line@numR<\z@
858       \led@warn@BadAdvancelineLine
859       \line@numR \z@
860     \fi
861   \else
862     \advance\line@num by #1\relax
863     \ifnum\line@num<\z@
864       \led@warn@BadAdvancelineLine
865       \line@num \z@
866     \fi
867   \fi
868 \fi
869 \set@line@action}
870
871 %

```

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

```

872
873 \newcommand*{\@set}[1]{%
874   \ifledRcol
875     \ifsublines@
876       \subline@numR=#1\relax
877     \else
878       \line@numR=#1\relax
879     \fi
880     \set@line@action
881   \else
882     \ifsublines@
883       \subline@num=#1\relax
884     \else
885       \line@num=#1\relax
886     \fi
887     \set@line@action
888   \fi}
889

```

890 %

\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 line number change is to be done.

```
891
892 \newcommand*{\l@d@set}[1]{%
893   \ifledRcol
894     \line@numR=#1\relax
895     \advance\line@numR \@ne
896     \def\l@dchset@num{#1}
897   \else
898     \line@num=#1\relax
899     \advance\line@num \@ne
900     \def\l@dchset@num{#1}
901   \fi}
902 \let\l@dchset@num\relax
903
904 %
```

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

```
905
906 \newcommand*{\page@action}{%
907   \ifledRcol
908     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
909     \xright@appenditem{\next@page@numR}\to\actions@listR
910   \else
911     \xright@appenditem{\the\absline@num}\to\actionlines@list
912     \xright@appenditem{\next@page@num}\to\actions@list
913   \fi}
914 %
```

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

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

```

927 \xright@appenditem{\the\absline@num}\to\actionlines@list
928 \ifsublines@
929   \@l@dttempcnta=-\subline@num
930 \else
931   \@l@dttempcnta=-\line@num
932 \fi
933 \advance\@l@dttempcnta by -5000\relax
934 \xright@appenditem{\the\@l@dttempcnta}\to\actions@list
935 \fi}
936 %

```

\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.

```

937
938 \newcommand*{\sub@action}{%
939   \ifledRcol
940     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
941     \ifsublines@
942       \xright@appenditem{-1001}\to\actions@listR
943     \else
944       \xright@appenditem{-1002}\to\actions@listR
945     \fi
946   \else
947     \xright@appenditem{\the\absline@num}\to\actionlines@list
948     \ifsublines@
949       \xright@appenditem{-1001}\to\actions@list
950     \else
951       \xright@appenditem{-1002}\to\actions@list
952     \fi
953   \fi}
954 %

```

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

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

```

955 \newcommand*{\lock@on}{\futurelet\next\do@lockon}
956
957 \newcommand*{\do@lockon}{%
958   \ifx\next\lock@off
959     \global\let\lock@off=\skip@lockoff
960   \else
961     \ifledRcol
962       \do@lockonR

```

```

963 \else
964 \do@lockonL
965 \fi
966 \fi}
967
968
969 \newcommand*{\do@lockonL}{%
970 \xright@appenditem{\the\absline@num}\to\actionlines@list
971 \ifsublines@
972 \xright@appenditem{-1005}\to\actions@list
973 \ifnum\sub@lock=\z@
974 \sub@lock \@ne
975 \else
976 \ifnum\sub@lock=\thr@@
977 \sub@lock \@ne
978 \fi
979 \fi
980 \else
981 \xright@appenditem{-1003}\to\actions@list
982 \ifnum\@lock=\z@
983 \@lock \@ne
984 \else
985 \ifnum\@lock=\thr@@
986 \@lock \@ne
987 \fi
988 \fi
989 \fi}
990
991 %

```

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

```

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

```

```

1010 \newcommand*{\do@lockoff}{%
1011   \ifledRcol
1012     \do@lockoffR
1013   \else
1014     \do@lockoffL
1015   \fi}
1016 \newcommand*{\skip@lockoff}{\global\let\lock@off=\do@lockoff}
1017 \global\let\lock@off=\do@lockoff
1018
1019 %

```

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

```

1020 \newcommand*{\n@num}{%
1021   \ifledRcol%
1022     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
1023     \xright@appenditem{-1007}\to\actions@listR
1024   \else%
1025     \xright@appenditem{\the\absline@num}\to\actionlines@list%
1026     \xright@appenditem{-1007}\to\actions@list%
1027   \fi%
1028 }%
1029
1030 %

```

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

```

1031 \newcommand*{\n@num@stanza}{%
1032   \ifledRcol%
1033     \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1034     \xright@appenditem{-1008}\to\actions@listR%
1035   \else%
1036     \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1037     \xright@appenditem{-1008}\to\actions@list%
1038   \fi%
1039 }
1040 %

```

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

\hidenumbering

```

1041 \h@num#1 \newif\ifl@dhiddenumber
1042 \newcommand*{\hidenumbering}{
1043   \ifledRcol%
1044     \write\linenum@outR{\string\hide@num}%
1045   \else%
1046     \write\linenum@out{\string\hide@num}%
1047   \fi%

```

```

1048 }%
1049 \newcommand*\hide@num{%
1050   \ifledRcol%
1051     \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1052     \xright@appenditem{-1009}\to\actions@listR%
1053   \else%
1054     \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1055     \xright@appenditem{-1009}\to\actions@list%
1056   \fi%
1057 }
1058 %

```

\@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.

```

1059 \newcount\insert@count
1060 %

```

- #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.

```

1061 \newcommand*\dummy@ref}[2]{#2}
1062 %

```

\@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.

```

1063 \newcommand*\@ref}[2]{%
1064   \ifledRcol%
1065     \@ref@regR{#1}{#2}%
1066   \else%
1067     \@ref@reg{#1}{#2}%
1068   \fi%
1069 }%
1070 \newcommand*\@ref@reg}[2]{%
1071   \global\insert@count=#1\relax
1072   \global\advance\@edtext@level by 1%
1073   \loop\ifnum\insert@count>\z@
1074     \xright@appenditem{\the\absline@num}\to\insertlines@list
1075     \global\advance\insert@count \m@ne
1076   \repeat
1077 %

```


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.

```

1078 \begingroup
1079   \let\@ref=\dummy@ref
1080   \let\@lopL\@gobble
1081   \let\page@action=\relax
1082   \let\sub@action=\relax
1083   \let\set@line@action=\relax
1084   \let\@lab=\relax
1085   \let\@lemma=\relax%
1086   \let\@sw\@gobblethree%
1087   #2
1088   \global\endpage@num=\page@num
1089   \global\endline@num=\line@num
1090   \global\endsubline@num=\subline@num
1091 \endgroup
1092 %

```

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

```

1093   \xright@appenditem%
1094     {\the\page@num|\the\line@num}%
1095     \ifsublines@ \the\subline@num \else 0\fi}%
1096     \the\endpage@num|\the\endline@num}%
1097     \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@list
1098 %

```

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.

```

1099   \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\
@edtext@level\endcsname}%
1100   \providebool{lemmacommand@\the\@edtext@level}%
1101   \boolfalse{lemmacommand@\the\@edtext@level}%
1102 %

```

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

```

1103   #2%
1104 %

```

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.

```

1105   \ifnum\@edtext@level>0%
1106     \def\create@this@edtext@level{\expandafter\list@create\expandafter{\
csname sw@list@edtext@\the\@edtext@level\endcsname}}%
1107     \ifcsundef{sw@list@edtext@\the\@edtext@level}{\create@this@edtext@level
}\}%

```

```

1108 \letcs{\@tmp}{sw@list@edtext@the\@edtext@level}%
1109 \letcs{\@tmpp}{sw@list@edtext@tmp@the\@edtext@level}
1110 \xright@appenditem{\expandonce\@tmpp}{to\@tmp%
1111 \global\cslet{sw@list@edtext@the\@edtext@level}{\@tmp}%
1112 \fi%
1113 %

Decrease edtext level counter.

1114 \global\advance\@edtext@level by -1%
1115 %

1116 }
1117
1118 %

```

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`.

```

1119 \newwrite\linenum@out
1120 %

```

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

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

```

1121 \newif\iffirst@linenum@out@
1122 \iffirst@linenum@out@true
1123 %

```

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

```

1124 \newcommand{\this@line@list@version}{3}%
1125 %

```

\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.

```

1126 \newcommand*{\line@list@stuff}[1]{%
1127 %

```

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

```

1128 \read@linelist{#1}%
1129 %

```

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.

```

1130 \iffirst@linenum@out@
1131 \immediate\closeout\linenum@out%
1132 \global\first@linenum@out@false%
1133 \immediate\openout\linenum@out=#1\relax%
1134 \immediate\write\linenum@out{\string\line@list@version{\
this@line@list@version}}%
1135 \ifl@dpaging%
1136 \immediate\write\linenum@out{\string\@par@sync@option{\
@par@this@sync@option}}%
1137 \fi%
1138 \else
1139 %

```

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.

```

1140 \if@minipage%
1141 \leavevmode%
1142 \fi%
1143 \closeout\linenum@out%
1144 \openout\linenum@out=#1\relax%
1145 \fi}
1146
1147 %

```

\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.

```

1148 \newcommand*{\new@line}{%
1149 \IfStrEq{\led@pb@setting}{after}%
1150 {\xifinlist{\the\absline@num}{\l@prev@nopb}%
1151 {\xifinlist{\the\absline@num}{\normal@page@break}%

```

```

1152     {\numgdef{\@next@page}{\c@page+1}%
1153     \write\linenum@out{\string\@nl[\@next@page][\@next@page]}}%
1154   }%
1155   {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1156 }%
1157 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1158 {}%
1159 \IfStrEq{\led@pb@setting}{before}%
1160 {\numdef{\next@absline}{\the\absline@num+1}%
1161 \xifinlist{\next@absline}{\l@prev@nopb}%
1162 {\xifinlist{\the\absline@num}{\normal@page@break}%
1163 {\numgdef{\nc@page}{\c@page+1}%
1164 \write\linenum@out{\string\@nl[\nc@page][\nc@page]}}%
1165 }%
1166 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1167 }%
1168 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1169 }%
1170 {}%
1171 \IfStrEqCase{\led@pb@setting}{before}{\relax}{after}{\relax}}{\write\
linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1172 }
1173
1174 %

```

\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

```

1175 \newif\if@noneed@Footnote%
1176
1177 \newcommand*{\flag@start}{%
1178   \ifledRcol%
1179     \edef\next{\write\linenum@out{%
1180       \string\@ref[\the\insert@countR] [ ]}%
1181     \next%
1182     \ifnum\insert@countR<1%
1183       \if@noneed@Footnote\else%
1184         \led@err@EdtextWithoutFootnote%
1185       \fi%
1186     \fi%
1187   \else%
1188     \edef\next{\write\linenum@out{%
1189       \string\@ref[\the\insert@count] [ ]}%
1190     \next%
1191     \ifnum\insert@count<1%
1192       \if@noneed@Footnote\else%

```

```

1193 \led@err@EdtextWithoutFootnote%
1194 \fi%
1195 \fi%
1196 \fi}%
1197
1198 %

```

\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.

```

1199
1200
1201 \newcommand*{\startsub}{\dimen0\lastskip
1202 \ifdim\dimen0>0pt \unskip \fi
1203 \ifledRcol \write\linenum@outR{\string\sub@on}%
1204 \else \write\linenum@out{\string\sub@on}%
1205 \fi
1206 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
1207 \def\endsub{\dimen0\lastskip
1208 \ifdim\dimen0>0pt \unskip \fi
1209 \ifledRcol \write\linenum@outR{\string\sub@off}%
1210 \else \write\linenum@out{\string\sub@off}%
1211 \fi
1212 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
1213
1214 %

```

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

```

1215 \newcommand*{\advanceline}[1]{\leavevmode%
1216 \ifledRcol \write\linenum@outR{\string\adv[#1]}%
1217 \else \write\linenum@out{\string\adv[#1]}%
1218 \fi%
1219 }
1220 %

```

\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.

```

1221 \newcommand*\setline}[1]{%
1222 \leavevmode%
1223 \ifnum#1<\z@
1224 \led@warn@BadSetline
1225 \else
1226 \ifledRcol \write\linenum@outR{\string\@set[#1]}%
1227 \else \write\linenum@out{\string\@set[#1]}%
1228 \fi
1229 \fi}
1230
1231
1232 %

```

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

```

1233 \newcommand*\setlinenum}[1]{%
1234 \ifnum#1<\z@
1235 \led@warn@BadSetlinenum
1236 \else
1237 \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
1238 \else \write\linenum@out{\string\l@d@set[#1]} \fi
1239 \fi}
1240
1241
1242 %

```

\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.

```

1243 \newcommand*\startlock{%
1244 \ifledRcol \write\linenum@outR{\string\lock@on}%
1245 \else \write\linenum@out{\string\lock@on}%
1246 \fi}
1247 \def\endlock{%
1248 \ifledRcol \write\linenum@outR{\string\lock@off}%
1249 \else \write\linenum@out{\string\lock@off}%
1250 \fi}
1251
1252 %

```

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

```

1253 \l@dskipnumbertrue \newif\ifl@dskipnumber
1254 \l@dskipnumberfalse \newif\ifl@dskipversenumber%
1255 \skipnumbering \newcommand*\skipnumbering{%
1256 \leavevmode%
1257 \ifledRcol%

```

```

1258 \ifistanza%
1259 \write\linenum@outR{\string\n@num@stanza}%
1260 \else%
1261 \write\linenum@outR{\string\n@num}%
1262 \fi%
1263 \advanceline{-1}%
1264 \else%
1265 \ifistanza%
1266 \write\linenum@out{\string\n@num@stanza}%
1267 \else%
1268 \write\linenum@out{\string\n@num}%
1269 \fi%
1270 \advanceline{-1}%
1271 \fi%
1272 }%
1273
1274 %

```

VI Marking text for notes

The `\edtext` macro is used to create all footnotes and endnotes, as well as to print the portion of the main text to which a given note or notes is keyed. The idea is to have that lemma appear only once in the `.tex` file: all instances of it in the main text and in the notes are copied from that one appearance.

The `\edtext` macro takes two arguments.

```
\edtext{#1}{#2}
```

- `#1` is the piece of the main text being glossed; it gets added to the main text, and is also used as a lemma for notes to it.
- `#2` is a series of subsidiary macros that generate various kinds of notes.

The `\edtext` macro may be used (somewhat) recursively; that is, `\edtext` may be used within its own first argument. The code would be much simpler without this feature, but nested notes will commonly be necessary: it is quite likely that we will have an explanatory note for a long passage and notes on variants for individual words within that passage. The situation we can't handle is overlapping notes that are not nested: for example, one note covering lines 10–15, and another covering 12–18. You can handle such cases by using the `\lemma` and `\linenum` macros within `#2`: they alter the copy of the lemma and the line numbers that are passed to the notes, and hence allow you to overcome any limitations of this system, albeit with extra effort.

The recursive operation of `\edtext` will fail if you try to use a copy that is called something other than `\edtext`. In order to handle recursion, `\edtext` needs to redefine its own definition temporarily at one point, and that does not work if the macro you are calling is not actually named `\edtext`. There is no problem as long as `\edtext` is not invoked in the first argument. If you want to call `\edtext` something else, it is best

to create instead a macro that expands to an invocation of `\edtext`, rather than copying `\edtext` and giving it a new name; otherwise you will need to add an appropriate definition for your new macro to `\morenoexpands`.

Side effects of our line-numbering code make it impossible to use the usual footnote macros directly within a paragraph whose lines are numbered (see comments to `\do@line`, VII.2.1 p. 123). Instead, the appropriate note-generating command is appended to the list macro `\inserts@list`, and when `\pend` completes the paragraph it inserts all the notes at the proper places.

Note that we do not provide previous-note information, although it is often wanted; your own macros must handle that. We 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.

```
1275 \list@create{\end@lemmas}
1276 %
```

`\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.


```
1277 \newcommand{\dummy@edtext}[2]{#1}
1278 %
```

`\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`.

```
1279 \newcommand{\dummy@edtext@showlemma}[2]{\showlemma{#1}}%
1280 %
```

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

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

The first class is font-changing macros. We suppress expansion for them by letting them become equal to zero.²⁵ This is done because we want to pass into our notes the generic commands to change to roman or whatever, and not their expansions that will ask for a particular style at a specified size. The notes may well be in a smaller font, so the command should be expanded later, when the note's environment is in effect.

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

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

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

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

(A related problem, not addressed by these two macros, is that of characters whose category code is changed by any the macros used in the arguments to `\edtext`. Since the category codes are set when the arguments are scanned, macros that depend on changing them will not work. We have most often encountered this with characters that are made 'active' within text in some, but not all, of the languages used within

²⁵Since 'control sequences equivalent to characters are not expandable'—*The TeXbook*, answer to Exercise 20.14.

the document. One way around the problem, if it takes this form, is to ensure that those characters are *always* active; within languages that make no special use of them, their associated control sequences should simply return the proper character. A simpler solution is to avoid active character, using Lua_{TeX} or Xe_{La}TeX.)

```

1281 \newcommand*{\no@expands}{%
1282   \let\select@lemmafnt=0%
1283   \let\startsub=\relax \let\endsub=\relax
1284   \let\startlock=\relax \let\endlock=\relax
1285   \let\edlabel=\@gobble
1286   \let\setline=\@gobble \let\advanceline=\@gobble
1287   \let\sameword\sameword@inedtext%
1288   \let\edtext=\dummy@edtext
1289   \l@dtabnoexpands
1290   \morenoexpands}
1291 \let\morenoexpands=\relax
1292
1293 %

```

\@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.

```

1294 \newcommand{\@tag}{}
1295 %

```

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

```

1296 \newcount\@edtext@level%
1297 \@edtext@level=0%
1298 %

```

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

```

1299 \newcommand{\edtext}[2]{\leavevmode%
1300 %

```

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

```

1301   \ifnumberedpar@%
1302 %

```

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

```

1303   \global\advance\@edtext@level by 1%
1304 %

```

By default, we do not use \lemma

```

1305   \global\@lemmacommand@false%
1306 %

```

```

1307 \begingroup%
1308 %

```

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`.

```

1309 \ifledRcol%
1310 \ifcsundef{sw@list@edtextR@the\@edtext@level}%
1311 {\global\let\sw@inthisedtext\empty}%
1312 {\ifcseempty{sw@list@edtextR@the\@edtext@level}%
1313 {\global\let\sw@inthisedtext\empty}%
1314 {\expandafter\glp\csname sw@list@edtextR@the\@edtext@level\endcsname\
to\sw@inthisedtext}%
1315 }%
1316 \else%
1317 \ifcsundef{sw@list@edtext@\the\@edtext@level}%
1318 {\global\let\sw@inthisedtext\empty}%
1319 {\ifcseempty{sw@list@edtext@\the\@edtext@level}%
1320 {\global\let\sw@inthisedtext\empty}%
1321 {\expandafter\glp\csname sw@list@edtext@\the\@edtext@level\endcsname\
to\sw@inthisedtext}%
1322 }%
1323 \fi%
1324 %

```

\@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.

```

1325 \global\renewcommand{\@tag}{%
1326 \no@expands #1%
1327 }%
1328 %

```

\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`.

```

1329 \set@line%
1330 %

```

`\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`.

```

1331 \ifledRcol \global\insert@countR \z@%
1332 \else \global\insert@count \z@ \fi%
1333 %

```

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.

```
1334 \ignorespaces #2\relax%
1335 %
```

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

```
1336 \ifundefined{xpg@main@language}{%if not polyglossia
1337 \flag@start}%
1338 {\if@RTL\flag@end\else\flag@start\fi%
1339 }%
1340 %
```

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

```
1341 \if@lemmacommand%
1342 \ifledRcol%
1343 \write\linenum@outR{\string\@lemma}%
1344 \else%
1345 \write\linenum@out{\string\@lemma}%
1346 \fi%
1347 \fi%
1348 %
```

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.

```
1349 \endgroup%
1350 \showlemma{#1}%
1351 %
```

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.

```
1352 \ifx\end@lemmas\empty \else%
1353 \gl@p\end@lemmas\to\x@lemma%
1354 \x@lemma%
1355 \global\let\x@lemma=\relax%
1356 \fi%
1357 \ifundefined{xpg@main@language}{%if not polyglossia
1358 \flag@end}%

```

```

1359     {\if@RTL\flag@start\else\flag@end\fi% With polyglossia, you must
        track whether the language reads left to right (English) or right to left
        (Arabic).
1360     }%
1361 %

```

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`.

```

1362     \global\@noneed@Footnotefalse%
1363     \global\advance\@edtext@level by -1%
1364     \global\@lemmacommand@false%
1365 %

```

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

```

1366     \else%
1367     \showlemma{#1} (\textbf{\textsc{Edtext outside numbered paragraph}})\
        led@err@edtextoutsidestart%
1368     \fi%
1369 }%
1370
1371 \newcommand*{\flag@end}{%
1372     \ifledRcol%
1373     \write\linenum@outR{}}%
1374     \else%
1375     \write\linenum@out{}}%
1376     \fi}%
1377 %
1378 %

```

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

```

1379 \newif\ifnumberline
1380 \numberlinetrue
1381 %

```

`\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.

```

1382 \newcommand*{\set@line}{%
1383   \ifledRcol
1384     \ifx\line@listR\empty
1385       \global\noteschanged@true
1386       \xdef\l@d@nums{000|000|000|000|000|000|000|\edfont@info}%
1387     \else
1388       \glp\line@listR\to\@tempb
1389       \xdef\l@d@nums{\@tempb|\edfont@info}%
1390       \global\let\@tempb=undefined
1391     \fi
1392   \else
1393     \ifx\line@list\empty
1394       \global\noteschanged@true
1395       \xdef\l@d@nums{000|000|000|000|000|000|000|\edfont@info}%
1396     \else
1397       \glp\line@list\to\@tempb
1398       \xdef\l@d@nums{\@tempb|\edfont@info}%
1399       \global\let\@tempb=undefined
1400     \fi
1401   \fi}
1402
1403 %

```

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

```

1404 \newcommand*{\edfont@info}{\f@encoding/\f@family/\f@series/\f@shape}
1405
1406 %

```

VI.2 Substitute lemma

`\lemma` The `\lemma{<text>}` macro allows you to change the lemma that is passed on to the notes. Read about `\@tag` in normal `\edtext` macro for more details about `\sw@list@inedtext` and `\no@expands` (VI.1 p. 107).

```

1407 \unless\ifnocritical@
1408 \newcommand*{\lemma}[1]{%
1409   \global\@lemmacommand@true%
1410   \global\renewcommand{\@tag}{%
1411     \no@expands #1%
1412   }%
1413   \ignorespaces%
1414 }%
1415 %

```

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

```

1416 \newcommand{\@lemma}{%
1417   \booltrue{lemmacommand@the\edtext@level}%
1418 }%
1419 \fi
1420 %

```

\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.

```

1421 \newif\if@lemmacommand@%
1422 %

```

VI.3 Substitute line numbers

\linenum The \linenum macro can change any or all of the page and line numbers that are passed on to the notes.

As argument \linenum takes a set of seven parameters separated by vertical bars, in the format used internally for \l@d@nums (see V.9 p. 82): the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma. However, you can omit any parameters you do not want to change, and you can omit a string of vertical bars at the end of the argument. Hence \linenum{18|4|0|18|7|1|0} is an invocation that changes all the parameters, but \linenum{|3} only changes the starting line number, and leaves the rest unaltered.

We use \ as an internal separator for the macro parameters.

```

1423 \newcommand*{\linenum}[1]{%
1424   \xdef\@tempa{#1|}|}|}|}|}|noexpand\\l@d@nums}%
1425   \global\let\l@d@nums=\empty
1426   \expandafter\line@set\@tempa|\\ignorespaces}
1427 %

```

\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.

```

1428 \def\line@set#1|#2|#3|#4|{%
1429   \gdef\@tempb{#1}%
1430   \ifx\@tempb\empty
1431     \l@d@add{#3}%
1432   \else
1433     \l@d@add{#1}%
1434   \fi
1435   \gdef\@tempb{#4}%
1436   \ifx\@tempb\empty\else

```

```

1437 \l@d@add{|\}\line@set#2\#4\%
1438 \fi}
1439 %

```

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

```

1440 \newcommand{\l@d@add}[1]{\xdef\l@d@nums{\l@d@nums#1}}
1441
1442 %

```

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 `eledmac` section (the `.1`, `.2...` file).
- **When this auxiliary file is read at the second run**, different operations are achieved:

1. Get the rank of each `\sameword` in a line (relative rank) from the rank of each `\sameword` in all the numbered section (absolute rank):
 - For each paired `\sameword` argument and absolute line number, a counter is defined. Its value corresponds to the number of times `\sameword{<argument>}` is called from the beginning of the lineation to the end of the current line. We also store the same data for the preceding absolute line number, if it does not have `\sameword{<argument>}`.
 - For each `\sameword` having the same argument, we subtract from its absolute rank the number stored for the paired `\sameword` argument and previous absolute line number. Consequently, we obtain the relative rank.
 - See the following example which explain how for same `\sameword` absolute ranks are transformed to relative rank.

At line 1:

absolute rank 1 becomes relative rank $1-0 = 1$
 1 is stored for this `\sameword` and the line 1

At line 2:

absolute rank 2 becomes relative rank $2-1 = 1$
 absolute rank 3 becomes relative rank $3-2 = 2$


```

3 is stored for this \sameword and the line 2
At line 3:
no \sameword for this line.
3 is stored for this \sameword and the line 3
At line 4:
absolute rank 4 becomes relative rank 4-3 = 1
3 is stored for this \sameword and the line 4

```

2. Create lists of lists of `\sameword` by depth of `\edtext`. That is: create a list for `\edtext` of level 1, a list for `\edtext` of level 2, a list for `\edtext` of level 3 etc. For each `\edtext` in these list, we store all the relative rank of `\saweword` which are called as lemma information, that is 1) or called in the first argument of `\sameword` 2) or called in the `\lemma` macro of the second argument of `\sameword` AND marked by the optional argument of `\saweword` in first argument of `\edtext`.

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

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 level of `\edtext`.

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

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

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

- At the second run, when a critical notes is called, we associate it to the next item of the list associated to is `\edtext` level. So, in the previous example:
 - Critical notes (A) and (B) are associated with $\{1, 2, 2, 3\}$.
 - Critical note (C) is associated with $\{1, 2, 2, 3, 4, 3\}$.
 - Critical note (D) is associated with $\{5\}$.
 - Critical note (E) is associated with $\{5, 4\}$.
- At the second run, when a critical note is printed:
 - The `\sameword` command is let `\sameword@inedtext`.
 - At each call of this `\sameword@inedtext`, we step to the next element of the list associated to the note. Let it be r .
 - For the word marked by `\sameword`, we calculate how many time it is called in its line. To do it:
 - * We get the absolute line number of the current `\sameword`. This absolute line number was stored with list of relative rank for the current

`\edtext`. That means, in the previous example, that, if the absolute line number of `\edtext` was 1, that critical notes (A) and (B) were not associated with $\{1, 2, 2, 3\}$ but with $\{(1, 1), (2, 1), (2, 1), (3, 1)\}$. Such method to know the absolute line number associated to a `\sameword` is required because a `\edtext` can be overlap many lines, but `\sameword` can't get it.

- * We get the value associated, when reading the auxiliary file, to the pair compose by the current marked word and the current absolute line number. Let this value be n .
- If $n > 1$, that mean the current word appears more than once time in its line. In this case, we call `\showwordrank` with the word as first argument and r as second argument. If the word is called only once, we just print it.

After theory, implementation.

`\get@sw@txt` As the argument of `\sameword` can contain active character if we use `inputenc` with `utf8` option instead of native UTF-8 engine, we store its detokenized content in a macro in order to allow dynamic name of macro with `\csname`.²⁶

Because there is a bug with `\detokenize` and \LaTeX when using non BMP characters²⁷, we detokenize only for not \LaTeX engines. In any case, in \LaTeX , a `\csname` construction can contain UTF-8 characters without a problem, as UTF-8 characters are not managed with category code, but instead read directly as UTF-8 characters.

```

1443 \newcommand{\get@sw@txt}[1]{%
1444   \ifxetex%
1445     \xdef\sw@txt{#1}%
1446   \else%
1447     \expandafter\xdef\expandafter\sw@txt\expandafter{\detokenize{#1}}%
1448   \fi%
1449 }%
1450 %

```

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

```

1451 \newcommand{\sameword}[2][1,usedefault]{%
1452   \leavevmode%
1453   \get@sw@txt{#2}%
1454 %

```

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

```

1455   \unless\ifledRcol%
1456     \csnumgdef{sw@\sw@txt}{\csuse{sw@\sw@txt}+1}%
1457 %

```

Then, write its value to the numbered file.

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

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

```

1458 \protected@write\linenum@out{}\string\@sw{\sw@txt}{\csuse{sw@\sw@txt
}}{#1}}%
1459 %

```

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

```

1460 \else%
1461 \csnumgdef{sw@\sw@txt@R}{\csuse{sw@\sw@txt@R}+1}%
1462 \protected@write\linenum@outR{}\string\@sw{\sw@txt}{\csuse{sw@\
sw@txt@R}}{#1}}%
1463 \fi%
1464 %

```

And print the word.

```

1465 #2%
1466 }%
1467 %

```

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

```

\if@addsw68 \newif\if@addsw%
1469 %

```

\@sw The command printed in the auxiliary files.

```

1470 \newcommand{\@sw}[3]{%
1471 \get@sw@txt{#1}%
1472 \unless\ifledRcol%
1473 %

```

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

```

1474 \csxdef{sw@\sw@txt @\the\absline@num @\the\section@num}{#2}%
1475 %

```

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

```

1476 \numdef{\prev@line}{\the\absline@num-1}%
1477 \ifcsundef{sw@\sw@txt @\prev@line @\the\section@num}{%
1478 \csnumgdef{sw@\sw@txt @\prev@line @\the\section@num}{#2-1}%
1479 }{}%
1480 %

```

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

```

1481 \numdef{\the@sw}{#2-\csuse{sw@\sw@txt @\prev@line @\the\section@num}}%
1482 %

```

And do the same thing for the right side.

```

1483 \else%
1484 \csxdef{sw@sw@txt @\the\absline@numR @\the\section@numR @R}{#2}%
1485 \numdef{\prev@line}{\the\absline@numR-1}%
1486 \ifcsundef{sw@sw@txt @\prev@line @\the\section@numR @R}{%
1487 \csnumgdef{sw@sw@txt @\prev@line @\the\section@numR @R}{#2-1}%
1488 }{}%
1489 \numdef{\the@sw}{#2-\csuse{sw@sw@txt @\prev@line @\the\section@numR @R
}}%
1490 \fi%
1491 %

```

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

```

1492 \@tempcnta=\@edtext@level
1493 \@whilenum{\@tempcnta>0}\do{%
1494 \ifcsdef{sw@list@edtext@tmp@\the\@tempcnta}%
1495 {%
1496 \addswfalse%
1497 \notbool{lemmacommand@\the\@tempcnta}%
1498 {\addswtrue}%
1499 {\IfStrEq{#3}{inlemma}%
1500 {\addswtrue}%
1501 {%
1502 \def\do##1{%
1503 \ifnumequal{##1}{\the\@tempcnta}%
1504 {\addswtrue\listbreak}%
1505 }%
1506 }%
1507 \docsvlist{#3}%
1508 }%
1509 }%
1510 \if@addsw%
1511 \letcs{\@tmp}{sw@list@edtext@tmp@\the\@tempcnta}%
1512 \ifledRcol%
1513 \xright@appenditem{\the@sw}{\the\absline@numR}\to\@tmp%
1514 \else%
1515 \xright@appenditem{\the@sw}{\the\absline@num}\to\@tmp%
1516 \fi%
1517 \cslet{sw@list@edtext@tmp@\the\@tempcnta}{\@tmp}%
1518 \fi%
1519 }%
1520 }%
1521 \advance\@tempcnta by -1%
1522 }%
1523 }%
1524 %

```

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

```

1525 \newcommandx{\sameword@inedtext}[2][1,usedefault]{%
1526 \get@sw@txt{#2}%

```

```

1527 \unless\ifledRcol@%
1528 %

```

Just a precaution.

```

1529 \ifx\sw@list@inedtext\empty%
1530 \def\the@sw{999}%
1531 \def\this@absline{-99}%
1532 \else%
1533 %

```

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

```

1534 \gl@p\sw@list@inedtext\to\@tmp%
1535 \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1536 \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1537 \fi%
1538 %

```

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

```

1539 \ifcsdef{sw@\sw@txt @\this@absline @\the\section@num}{%
1540 \numdef{\prev@line}{\this@absline-1}%
1541 \numdef{\sw@atthisline}{\csuse{sw@\sw@txt @\this@absline @\the\
section@num}-\csuse{sw@\sw@txt @\prev@line @\the\section@num}}%
1542 }%
1543 {\numdef{\sw@atthisline}{0}}%
1544 %

```

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

```

1545 \ifnumgreater{\sw@atthisline}{1}%
1546 {\showwordrank{#2}{\the@sw}}%
1547 {#2}%
1548 %

```

And the same for right side.

```

1549 \else%
1550 \ifx\sw@list@inedtext\empty%
1551 \def\the@sw{999}%
1552 \def\this@absline{-99}%
1553 \else%
1554 \gl@p\sw@list@inedtext\to\@tmp%
1555 \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1556 \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1557 \fi%
1558 \ifcsdef{sw@\sw@txt @\this@absline @\the\section@numR @R}{%
1559 \numdef{\prev@line}{\this@absline-1}%
1560 \numdef{\sw@atthisline}{\csuse{sw@\sw@txt @\this@absline @\the\
section@numR @R}-\csuse{sw@\sw@txt @\prev@line @\the\section@numR @R}}%
1561 }%

```

```

1562     {\numdef{\sw@atthisline}{0}}%
1563     \ifnumgreater{\sw@atthisline}{1}%
1564         {\showwordrank{#2}{\the@sw}}%
1565         {#2}%
1566     \fi%
1567 }%
1568 %

```

```

\showwordrank % Finally, the way the rank will be printed.
1570 \newcommand{\showwordrank}[2]{%
1571     #1\textsuperscript{#2}%
1572 }%
1573 %

```

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

Here are numbers and flags that are used internally in the course of the paragraph decomposition.

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

```

1574 \newbox\raw@text
1575 \newif\ifnumberedpar@
1576 \newcount\num@lines
1577 \newbox\one@line
1578 \newcount\par@line
1579 %

```

`\pstart` starts the paragraph by clearing the `\inserts@list` list and other relevant variables, and then arranges for the subsequent text to go into the `\raw@text` box. `\pstart` needs to appear at the start of every paragraph that is to be numbered; the `\autopar` command below may be used to insert these commands automatically.

```

\AtEveryPstart
\numberpstarttrue
\numberpstartfalse
\labelpstarttrue
\labelpstartfalse
\thepstart

```

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

```

1580
1581 \newcommand{\AtEveryPstart}[1]{%
1582   \ifstrempy{#1}%
1583     {\xdef\at@every@pstart{}}%
1584     {\gdef\at@every@pstart{\noindent#1}}%
1585 }%
1586 \xdef\at@every@pstart{}%
1587
1588 \newcounter{pstart}
1589 \renewcommand{\thepstart}{\bfseries\@arabic\c@pstart}. }
1590 \newif\ifnumberpstart
1591 \numberpstartfalse
1592 \newif\iflabelpstart
1593 \labelpstartfalse
1594 \newcommandx*{\pstart}[1][1]{%
1595   \normal@pars%
1596   \ifstrempy{#1}{\at@every@pstart}{\noindent#1}%
1597   \ifautopar%
1598     \autopar%
1599   \fi%
1600   \ifluatex%
1601     \edef\l@luatextextdir@L{\the\textdir}%
1602   \fi%
1603   \if@nobreak%
1604     \let\@oldnobreak\@nobreaktrue%
1605   \else%
1606     \let\@oldnobreak\@nobreakfalse%
1607   \fi%
1608   \@nobreaktrue%
1609   \ifnumbering \else%
1610     \led@err@PstartNotNumbered%
1611     \beginnumbering%
1612   \fi%
1613   \ifnumberedpar@%
1614     \led@err@PstartInPstart%
1615     \pend%
1616   \fi%
1617   \list@clear{\inserts@list}%
1618   \global\let\next@insert=\empty%
1619   \begingroup\normal@pars%
1620   \global\advance \l@dnumpstartsL\@ne
1621   \global\setbox\raw@text=\vbox\bgroup%
1622     \ifautopar\else%
1623     \ifnumberpstart%
1624       \ifinstanza\else%
1625       \ifsidepstartnum\else%
```

```

1626     \thepstart%
1627     \fi%
1628     \fi%
1629     \fi%
1630     \fi%
1631     \numberedpar@true%
1632     \iflabelpstart\protected@edef\@currentlabel%
1633         {\p@pstart\thepstart}
1634     \fi%
1635     \l@dzeropenalties%
1636     \ignorespaces%because not automatically ignored if an optional argument
is used (classical TeX behavior for space after commands)
1637 }
1638 %

```

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

```

1639 \newcommandx*{\pend}[1][1]{\ifnumbering \else%
1640     \led@err@PendNotNumbered%
1641     \fi%
1642     \global\l@dskipversenumberfalse%
1643     \ifnumberedpar@ \else%
1644         \led@err@PendNoPstart%
1645     \fi%
1646 %

```

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.

```

1647     \l@dzeropenalties%
1648     \endgraf\global\num@lines=\prevgraf\egroup%
1649     \global\par@line=0%
1650 %

```

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`.

```

1651     \csnumdef{pstartline}{0}%
1652     \loop\ifvbox\raw@text%
1653         \csnumdef{pstartline}{\pstartline+1}%
1654         \do@line%
1655         \ifbypstart@%
1656             \ifnumequal{\pstartline}{1}{%
1657                 \bgroup%
1658                 \let\leavevmode\relax%

```



```

1659     \setline{1}%
1660     \egroup%
1661     \resetprevline@{}%
1662     \fi%
1663     \repeat%
1664 %

```

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

```

1665     \flush@notes%
1666     \endgroup%
1667     \ignorespaces%
1668 %

```

Increase pstart counter.

```

1669     \ifnumberpstart%
1670         \pstartnumtrue%
1671     \fi%
1672     \addtocounter{pstart}{1}%
1673 %

```

Restore paragraph, nobreak setting and autopar setting.

```

1674     \normal@pars%
1675     \@oldnobreak%
1676     \ifautopar%
1677         \autopar%
1678     \fi%
1679 %

```

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

```

1680     \ifstrempy{#1}{\at@every@pend}{\noindent#1}%
1681 }
1682 %
1683 %

```

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.

```

\AtEveryPend84
\at@every@pend85 \newcommand{\AtEveryPend}[1]{%
1686     \ifstrempy{#1}%
1687         {\xdef\at@every@pend{}}%
1688         {\gdef\at@every@pend{\noindent#1}}%
1689     }%
1690     \xdef\at@every@pend{}%
1691 %
1692 %

```

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

```

1693 \newcommand*{\l@dzzeropenalties}{%
1694   \brokenpenalty \z@ \clubpenalty \z@
1695   \displaywidowpenalty \z@ \interlinepenalty \z@ \predisplaypenalty \z@
1696   \postdisplaypenalty \z@ \widowpenalty \z@}
1697
1698 %

```

`\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.

```

1699 \newif\ifautopar
1700 \autoparfalse
1701 \newcommand*{\autopar}{
1702   \ifledRcol
1703     \ifnumberingR \else
1704       \led@err@AutoparNotNumbered
1705     \beginnumberingR
1706     \fi
1707   \else
1708     \ifnumbering \else
1709       \led@err@AutoparNotNumbered
1710     \beginnumbering
1711     \fi
1712   \fi
1713   \autopartrue
1714   \everypar{\setbox0=\lastbox
1715     \endgraf \vskip-\parskip

```

```

1716 \pstart \noindent \kern\wd0 \ifnumberpstart\ifinstanza\else\thepstart\
fi\fi
1717 \let\par=\pend}%
1718 \ignorespaces}
1719 %

```

\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.

```

1720 \newcommand*{\normal@pars}{\everypar{}\let\par\endgraf}
1721
1722 %

```

\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.

```

1723 \newif\ifautopar@pause
1724 %

```

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

```

1725 \newcommand*{\l@dunhbox@line}[1]{\unhbox #1}
1726 \newcommand*{\do@line}{%
1727   {\vbadness=10000
1728     \splittopskip=\z@
1729     \do@linehook
1730   \l@demtyd@ta
1731     \global\setbox\one@line=\vsplit\raw@text to\baselineskip}%
1732   \unvbox\one@line \global\setbox\one@line=\lastbox
1733   \getline@num
1734   \IfStrEq{\led@pb@setting}{before}{\led@check@pb\led@check@nopb}{\}
1735   \ifnum\@lock>\@one
1736     \inserthangingsymboltrue
1737   \else
1738     \inserthangingsymbolfalse
1739   \fi
1740   \check@pb@in@verse
1741   \ifl@dhidenumber%
1742     \global\l@dhidenumberfalse%
1743     \f@x@l@cks%
1744   \else%
1745     \affixline@num%
1746   \fi%
1747   %

```

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

```

1748 \xifinlist{\the\l@dumpstartsL}{\eled@sections@@}%
1749     {\print@eledsection}%
1750     {\print@line}%
1751 \IfStrEq{\led@pb@setting}{after}{\led@check@pb\led@check@nopb}{\}
1752 }%
1753 %

```

VII.2.2 Process for “normal” line

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

```

1754 \def\print@line{
1755 %

```

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

```

1756     \affixpstart@num%
1757 %

```

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

```

1758     \hb@xt@ \linewidth{\%
1759 %

```

User hook.

```

1760     \do@insidelinehook%
1761 %

```

Left line number

```

1762     \l@dld@ta%
1763 %

```

Restore marginal and footnotes.

```

1764     \add@inserts\affixside@note%
1765 %

```

Print left notes.

```

1766     \l@dlsn@te
1767 %

```

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

```

1768     {\ledllfill\hb@xt@ \wd\one@line{\new@line%
1769 %

```

If we use Lua_{TEX} then restore the direction.

```

1770     \ifluatex%
1771     \textdir\l@luatextextdir@L%
1772     \fi%
1773 %

```

Insert, if needed, the hanging symbol.

```
1774 \inserthangingsymbol %Space kept for backward compatibility
1775 %
```

And so, print the line.

```
1776 \l@dunhbox@line{\one@line}}%
1777 %
```

Right line number

```
1778 \ledrlfill\l@drd@ta%
1779 %
```

Print right notes.

```
1780 \l@drsn@te
1781 }}%
1782 %
```

And reinsert penalties (for page breaking)...

```
1783 \add@penalties%
1784 }
1785 %
```

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.

```
1786 \def\print@eledsection{%
1787 \add@inserts\affixside@note%
1788 \numdef{\temp@}{\l@dnumpstartsL-1}%
1789 \xifinlist{\temp@}{\eled@sections@@}{\@nbreaktrue}{\@nbreakfalse}%
1790 \@eled@sectioningtrue%
1791 \csuse{eled@sectioning@the\l@dnumpstartsL}%
1792 \@eled@sectioningfalse%
1793 \global\csundef{eled@sectioning@the\l@dnumpstartsL}%
1794 \if@RTL%
1795 \hspace{-3\paperwidth}%
1796 {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1797 \else%
1798 \hspace{3\paperwidth}%
1799 {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
1800 \fi%
1801 \vskip-\baselineskip%
1802 }
1803 %
```

VII.2.4 Hooks

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

```
1804 \newcommand*\do@linehook{}{}
1805 \newcommand*\do@insidelinehook{}{}
1806 %
```

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

```
1807 \newcommand*\dolinehook[1]{\gdef\do@linehook{#1}}%
1808 \newcommand*\doinsidelinehook[1]{\gdef\do@insidelinehook{#1}}%
1809
1810 %
```

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.

```
1811 \l@dcsnotetext \newcommand*\l@demptyd@ta{}{}%
1812 \l@dcsnotetext@l \gdef\l@dld@ta{}{}%
1813 \l@dcsnotetext@r \gdef\l@drd@ta{}{}%
1814 \gdef\l@dcsnotetext@l{}{}%
1815 \gdef\l@dcsnotetext@r{}{}%
1816 \gdef\l@dcsnotetext{}{}
1817
1818 %
```

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

```
1819 \l@dlsn@te \newcommand*\l@dlsn@te{}{}%
1820 \hb@xt@ \z@{\hss\box\l@dldp@rbox\kern\ledlsnotesep}}
1821 \l@dlsn@te \newcommand*\l@dlsn@te{}{}%
1822 \hb@xt@ \z@{\kern\ledrsnotesep\box\l@drp@rbox\hss}}
1823
1824 %
```

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

```
1825 \newcommand*\ledllfill{\hfil}
1826 \newcommand*\ledrlfill{}{}
1827
1828 %
```

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.

```

1829 \newcommand*{\getline@num}{%
1830   \global\advance\absline@num \@ne%
1831   \do@actions
1832   \do@ballast
1833   \ifnumberline
1834     \ifsublines@
1835       \ifnum\sub@lock<\tw@
1836         \global\advance\subline@num \@ne
1837       \fi
1838     \else
1839       \ifnum\@lock<\tw@
1840         \global\advance\line@num \@ne
1841         \global\subline@num \z@
1842       \fi
1843     \fi
1844   \fi
1845 }
1846 %

```

`\do@ballast` The real work in the macro above is done in `\do@actions`, but before we plunge into that, let's get `\do@ballast` out of the way. This macro looks to see if there is an action to be performed on the *next* line, and if it is going to be a page break action, `\do@ballast` decreases the count `\ballast@count` counter by the amount of ballast. This means, in practice, that when `\add@penalties` assigns penalties at this point, \TeX will be given extra encouragement to break the page here (see XI.2 p. 136).

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

`\c@ballast`

```

1847 \newcount\ballast@count
1848 \newcounter{ballast}
1849 \setcounter{ballast}{0}
1850 %

```

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.

```

1851 \newcommand*{\do@ballast}{\global\ballast@count \z@
1852   \begingroup
1853     \advance\absline@num \@ne
1854     \ifnum\next@actionline=\absline@num
1855       \ifnum\next@action>-1001\relax
1856         \global\advance\ballast@count by -\c@ballast
1857       \fi
1858     \fi

```

```

1859 \endgroup}
1860 %

```

`\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`.

```

1861 \newcommand*{\do@actions}{%
1862   \global\let\do@actions@next=\relax
1863   \ifnum\absline@num<\next@actionline\else
1864   %

```

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.

```

1865   \ifnum\next@action>-1001
1866     \global\page@num=\next@action
1867     \ifbypage@
1868       \global\line@num=\z@ \global\subline@num=\z@
1869       \resetprevline@
1870     \fi
1871   %

```

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`.)

```

1872   \else
1873     \ifnum\next@action<-4999
1874       \@l@dttempcnta=-\next@action
1875       \advance\@l@dttempcnta by -5001
1876       \ifsublines@
1877         \global\subline@num=\@l@dttempcnta
1878       \else
1879         \global\line@num=\@l@dttempcnta
1880       \fi
1881   %

```

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

```

1882   \else
1883     \@l@dttempcnta=-\next@action
1884     \advance\@l@dttempcnta by -1000
1885     \do@actions@fixedcode
1886   \fi
1887 \fi
1888 %

```


Now we get information about the next action off the list, and then set `\do@actions@next` so that we will call ourselves recursively: the next action might also be for this line.

There is no warning if we find `\actionlines@list` empty, since that will always happen near the end of the section.

```

1889 \ifx\actionlines@list\empty
1890 \gdef\next@actionline{1000000}%
1891 \else
1892 \gl@p\actionlines@list\to\next@actionline
1893 \gl@p\actions@list\to\next@action
1894 \global\let\do@actions@next=\do@actions
1895 \fi
1896 \fi
1897 %

```

Make the recursive call, if necessary.

```

1898 \do@actions@next}
1899
1900 %

```

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

```

1901 \newcommand*{\do@actions@fixedcode}{%
1902 \ifcase\@l@dttempcnta
1903 \or% % 1001
1904 \global\sublines@true
1905 \or% % 1002
1906 \global\sublines@false
1907 \or% % 1003
1908 \global\@lock=\@ne
1909 \or% % 1004
1910 \ifnum\@lock=\tw@
1911 \global\@lock=\thr@@
1912 \else
1913 \global\@lock=\z@
1914 \fi
1915 \or% % 1005
1916 \global\sub@lock=\@ne
1917 \or% % 1006
1918 \ifnum\sub@lock=\tw@
1919 \global\sub@lock=\thr@@
1920 \else
1921 \global\sub@lock=\z@
1922 \fi
1923 \or% % 1007
1924 \l@dskipnumbertrue
1925 \or% % 1008
1926 \l@dskipversenumbertrue%
1927 \or% % 1009
1928 \l@dhiddenumbertrue

```

```

1929 \else
1930 \led@warn@BadAction
1931 \fi}
1932
1933
1934 %

```

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.

```

1935 \newcommand*{\affixline@num}{\%
1936 %

```

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).

```

1937 \ifledgroupnotesL\else
1938 \ifnumberline
1939 \ifl@dskipnumber
1940 \global\l@dskipnumberfalse
1941 \else
1942 \ifsublines@
1943 \l@tempcntb=\subline@num
1944 \ifnum\subline@num>\c@firstsublinenum
1945 \l@tempcnta=\subline@num
1946 \advance\l@tempcnta by-\c@firstsublinenum
1947 \divide\l@tempcnta by\c@sublinenumincrement
1948 \multiply\l@tempcnta by\c@sublinenumincrement
1949 \advance\l@tempcnta by\c@firstsublinenum
1950 \else
1951 \l@tempcnta=\c@firstsublinenum
1952 \fi
1953 %

```

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.

```
1954 \ch@cksub@l@ck
1955 %
```

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

```
1956 \else
1957 \@l@tempcntb=\line@num
1958 %
```

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

```
1959 \ifx\linenumberlist\empty
1960 \ifnum\line@num>\c@firstlinenum
1961 \@l@tempcnta=\line@num
1962 \advance\@l@tempcnta by-\c@firstlinenum
1963 \divide\@l@tempcnta by\c@linenumincrement
1964 \multiply\@l@tempcnta by\c@linenumincrement
1965 \advance\@l@tempcnta by\c@firstlinenum
1966 \else
1967 \@l@tempcnta=\c@firstlinenum
1968 \fi
1969 \else
1970 %
```

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

```
1971 \@l@tempcnta=\line@num
1972 \edef\rem@inder{\,\linenumberlist,\number\line@num,}%
1973 \edef\sc@n@list{\def\noexpand\sc@n@list
1974 ###1,\number\@l@tempcnta,###2|\def\noexpand\rem@inder
1975 {####2}}}%
1976 \sc@n@list\expandafter\sc@n@list\rem@inder|
1977 \ifx\rem@inder\empty%
1978 \advance\@l@tempcnta\@ne
1979 \fi
1980 %
```

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

```
1981 \ch@ck@l@ck
1982 \fi
1983 %
```

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

```
1984 \ifnum\@l@tempcnta=\@l@tempcntb
1985 \ifl@dskipversenumber\else
1986 %
```

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`

```

1987         \if@twocolumn
1988             \if@firstcolumn
1989                 \gdef\l@dld@ta{\llap{\leftlinenum}}}%
1990             \else
1991                 \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
1992             \fi
1993         \else
1994             \@l@tempcntb=\line@margin
1995             \ifnum\@l@tempcntb>\@ne
1996                 \advance\@l@tempcntb \page@num
1997             \fi
1998             \ifodd\@l@tempcntb
1999                 \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
2000             \else
2001                 \gdef\l@dld@ta{\llap{\leftlinenum}}}%
2002             \fi
2003         \fi
2004     \fi
2005 \fi
2006 %

```

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

```

2007         \f@x@l@cks
2008     \fi
2009 \fi
2010 \fi
2011 }
2012 %
2013 %

```

`\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.

```

2014 \newcommand*{\ch@cksub@l@ck}{%
2015   \ifcase\sub@lock
2016     \or
2017       \ifnum\sublock@disp=\@ne
2018         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2019       \fi
2020     \or
2021       \ifnum\sublock@disp=\tw@ \else
2022         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2023       \fi
2024     \or
2025       \ifnum\sublock@disp=\z@
2026         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2027       \fi
2028   \fi}
2029 %

```

Similarly for line numbers.

```

2030 \newcommand*{\ch@ck@l@ck}{%
2031   \ifcase\@lock
2032     \or
2033       \ifnum\lock@disp=\@ne
2034         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2035       \fi
2036     \or
2037       \ifnum\lock@disp=\tw@ \else
2038         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2039       \fi
2040     \or
2041       \ifnum\lock@disp=\z@
2042         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2043       \fi
2044   \fi}
2045 %

```

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

```

2046 \newcommand*{\f@x@l@cks}{%
2047   \ifcase\@lock
2048     \or
2049       \global\@lock=\tw@
2050     \or \or
2051       \global\@lock=\z@
2052     \fi
2053   \ifcase\sub@lock
2054     \or
2055       \global\sub@lock=\tw@
2056     \or \or
2057       \global\sub@lock=\z@
2058     \fi}

```

2059
2060 %

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.

```

\leftpstartnum61
\rightpstartnum62 \newif\ifsidepstartnum
\ifsidepstartnum63 \newcommand*{\affixpstart@num}{%
2064   \ifsidepstartnum
2065     \if@twocolumn
2066       \if@firstcolumn
2067         \gdef\l@dld@ta{\llap{\leftpstartnum}}}%
2068       \else
2069         \gdef\l@drd@ta{\rlap{\rightpstartnum}}}%
2070       \fi
2071     \else
2072       \l@dttempcntb=\line@margin
2073       \ifnum\l@dttempcntb>\@ne
2074         \advance\l@dttempcntb \page@num
2075       \fi
2076       \ifodd\l@dttempcntb
2077         \gdef\l@drd@ta{\rlap{\rightpstartnum}}}%
2078       \else
2079         \gdef\l@dld@ta{\llap{\leftpstartnum}}}%
2080       \fi
2081     \fi
2082   \fi
2083 }
2084 %
2085
2086
2087 \newif\ifpstartnum
2088 \pstartnumtrue
2089 \newcommand*{\leftpstartnum}{
2090   \ifpstartnum\thepstart
2091   \kern\linenumsep\fi

```

```

2092 \global\pstartnumfalse
2093 }
2094 \newcommand*{\rightpstartnum}{
2095   \ifpstartnum
2096     \kern\linenumsep
2097     \thepstart
2098     \fi
2099   \global\pstartnumfalse
2100 }
2101 %

```

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.

```

2102 \list@create{\inserts@list}
2103 %

```

`\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`.

```

2104 \newcommand*{\add@inserts}{%
2105   \global\let\add@inserts@next=\relax
2106   %

```

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

```

2107   \ifx\inserts@list\empty \else
2108   %

```

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.

```

2109 \ifx\next@insert\empty
2110 \ifx\insertlines@list\empty
2111 \global\noteschanged@true
2112 \gdef\next@insert{100000}%
2113 \else
2114 \gl@p\insertlines@list\to\next@insert
2115 \fi
2116 \fi
2117 %

```

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

```

2118 \ifnum\next@insert=\absline@num
2119 \gl@p\inserts@list\to\@insert
2120 \@insert
2121 \global\let\@insert=\undefined
2122 \global\let\next@insert=\empty
2123 \global\let\add@inserts@next=\add@inserts
2124 \fi
2125 \fi
2126 %

```

Make the recursive call, if necessary.

```

2127 \add@inserts@next}
2128
2129 %

```

XI.2 Penalties

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

In this code, `\num@lines` is the number of lines in the whole paragraph, and `\par@line` is the line we are working on at the moment. The count `\@l@dttempcnta` is used to calculate and accumulate the penalty; it is initially set to the value of `\ballast@count`, which has been worked out in `\do@ballast` above (VIII p. 127). Finally, the penalty is checked to see that it does not go below -10000 .

```

2130 \newcommand*{\add@penalties}{\@l@dttempcnta=\ballast@count
2131 \ifnum\num@lines>\@ne
2132 \global\advance\par@line \@ne
2133 \ifnum\par@line=\@ne

```



```

2134 \advance\@l@dttempcnta \clubpenalty
2135 \fi
2136 \@l@dttempcntb=\par@line \advance\@l@dttempcntb \@ne
2137 \ifnum\@l@dttempcntb=\num@lines
2138 \advance\@l@dttempcnta \widowpenalty
2139 \fi
2140 \ifnum\par@line<\num@lines
2141 \advance\@l@dttempcnta \interlinepenalty
2142 \fi
2143 \fi
2144 \ifnum\@l@dttempcnta=\z@
2145 \relax
2146 \else
2147 \ifnum\@l@dttempcnta>-10000
2148 \penalty\@l@dttempcnta
2149 \else
2150 \penalty -10000
2151 \fi
2152 \fi}
2153
2154 %

```

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.

```

2155 \newcommand*{\flush@notes}{%
2156 \@xloop
2157 \ifx\inserts@list\empty \else
2158 \glp\inserts@list\to\@insert
2159 \@insert
2160 \global\let\@insert=\undefined
2161 \repeat}
2162
2163 %

```

\@xloop `\@xloop` is a variant of the PLAIN \TeX `\loop` macro, useful when it's hard to construct a positive test using the \TeX `\if` commands—as in `\flush@notes` above. One types `\@xloop ... \if ... \else ... \repeat`, and the action following `\else` is repeated as long as the `\if` test fails. (This macro will work wherever the PLAIN \TeX `\loop` is used, too, so we could just call it `\loop`; but it seems preferable not to change the definitions of any of the standard macros.)

This variant of `\loop` was introduced by Alois Kabelschacht in *TUGboat* **8** (1987), pp. 184–5.

```

2164 \def\@xloop#1\repeat{%
2165   \def\body{#1\expandafter\body\fi}%
2166   \body}
2167
2168 %

```

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.

```

2169 \def\select@lemmafont#1|#2|#3|#4|#5|#6|#7|{\select@@lemmafont#7|}
2170 \def\select@@lemmafont#1/#2/#3/#4|{%
2171   {\fontencoding{#1}\fontfamily{#2}\fontseries{#3}\fontshape{#4}%
2172   \selectfont}
2173
2174 %

```

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.

```

2175 \newcommand*{\footnoteoptions@}[3][1=L,usedefault]{%
2176   \def\do##1{%
2177     \ifstrequal{#1}{L}{% In Leftside
2178       \xright@appenditem{\global\noexpand\settoggle{##1@}{#3}}\to\
inserts@list% Switch toggle, in all case
2179       \global\advance\insert@count \@ne% Increment the left insert
counter.
2180     }%

```

```

2181     {%
2182         \xright@appenditem{\global\noexpand\settoggle{##1@}{#3}}\to\
inserts@listR% Switch toggle, in all case
2183         \global\advance\insert@countR \@one% Increment the right insert
counter insert.
2184     }%
2185 }%
2186 \notblank{#2}{\docsvlist{#2}}}% Parsing all options
2187 }
2188 %

```

XII.3 Notes language

`\footnotelang@lua` `\footnotelang@lua` is called to remember the information about the direction of a lemma when Lua¹TeX is used.

```

2189 \newcommand*{\footnotelang@lua}[1][1=L,usedefault]{%
2190     \ifstrequal{#1}{L}{%
2191         \xright@appenditem{\csxdef{footnote@luatextextdir}{\the\textdir}}\to\
inserts@list%Know the dir of lemma
2192         \global\advance\insert@count \@one%
2193         \xright@appenditem{\csxdef{footnote@luatexpardir}{\the\pardir}}\to\
inserts@list%Know the dir of lemma
2194         \global\advance\insert@count \@one%
2195     }%
2196     {%
2197         \xright@appenditem{\csxdef{footnote@luatextextdir}{\the\textdir}}\to\
inserts@listR%Know the dir of lemma
2198         \global\advance\insert@countR \@one%
2199         \xright@appenditem{\csxdef{footnote@luatexpardir}{\the\pardir}}\to\
inserts@listR%Know the dir of lemma
2200         \global\advance\insert@countR \@one%
2201     }%
2202 }
2203 %

```

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

```

2204 \newcommand*{\footnotelang@poly}[1][1=L,usedefault]{%
2205     \ifstrequal{#1}{L}{%
2206         \if@RTL%
2207             \xright@appenditem{\csxdef{footnote@dir}{@RTLtrue}}\to\
inserts@list%Know the language used in the lemma
2208             \global\advance\insert@count \@one%
2209         \else
2210             \xright@appenditem{\csxdef{footnote@dir}{@RTLfalse}}\to\
inserts@list%Know the language of lemma
2211             \global\advance\insert@count \@one%

```

```

2212 \fi%
2213 \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\language}}}\
to\inserts@list%Know the language of lemma
2214 \global\advance\insert@count \@ne%
2215 }%
2216 {%
2217 \if@RTL
2218 \xright@appenditem{{\csxdef{footnote@dir}{@RTLtrue}}}\to\
inserts@listR%Know the language of lemma
2219 \global\advance\insert@countR \@ne%
2220 \else
2221 \xright@appenditem{{\csxdef{footnote@dir}{@RTLfalse}}}\to\
inserts@listR%Know the language of lemma
2222 \global\advance\insert@countR \@ne%
2223 \fi
2224 \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\language}}}\
to\inserts@listR%Know the language of lemma
2225 \global\advance\insert@countR \@ne%
2226 }%
2227 }
2228 %

```

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`.

```

2229 \newcommand*{\footplitskips}{%
2230   \interlinepenalty=\interfootnotelinepenalty
2231   \unless\ifl@dprintingpages%
2232     \floatingpenalty=\@MM%
2233   \fi%
2234   \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
2235   \leftskip=\z@skip \rightskip=\z@skip}
2236
2237 %

```

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

```

2238 \let\normalfootnoterule=\footnoterule
2239 %

```

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.

```

2240 \newcommandx{\Xarrangement}[2][1,usedefault]{%
2241   \def\do##1{%
2242     \csname Xarrangement@#2\endcsname{##1}%
2243   }%
2244   \ifstrepty{#1}%
2245     {%
2246       \dolistloop{\@series}%
2247     }%
2248   {
2249     \docsvlist{#1}%
2250   }%
2251 }%
2252 %

```

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.

```
2253 \newcommand*{\Xarrangement@normal}[1]{%
2254   \csgdef{series@display#1}{normal}
2255   \expandafter\let\csname #1footstart\endcsname=\normalfootstart
2256   \expandafter\let\csname v#1footnote\endcsname=\normalvfootnote
2257   \expandafter\let\csname #1footfmt\endcsname=\normalfootfmt
2258   \expandafter\let\csname #1footgroup\endcsname=\normalfootgroup
2259   \expandafter\let\csname #1footnoterule\endcsname=%
2260                                   \normalfootnoterule
2261   \count\csname #1footins\endcsname=1000
2262   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2263   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2264   \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2265   %
```

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.

```
2266   \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
2267   side only
2268   %
```

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).

```
2268   \ifnoledgroup@else%
2269     \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2270     \expandafter\let\csname mp#1footgroup\endcsname=\mpnormalfootgroup
2271     \count\csname mp#1footins\endcsname=1000
```

```

2272 \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2273 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2274 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2275 \fi
2276 }
2277
2278 %

```

\normalvfootnote We now begin a series of commands that do ‘normal’ footnote formatting: a format much like that implemented in PLAIN T_EX, 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.

```

2279 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnote}[2]{%
2280 \insert\csname #1footins\endcsname\bgroup
2281 \noindent\csuse{Xhooknote@#1}%
2282 \csuse{Xnotefontsize@#1}%
2283 \footsplitskips
2284 \ifl@dpairing\ifl@dpadding\else%
2285 \setXnoteswidthliketwocolumns@{#1}%
2286 \fi\fi%
2287 \setXnotespositionliketwocolumns@{#1}%
2288 \spaceskip=\z@skip \xspaceskip=\z@skip
2289 \csname #1footfmt\endcsname #2{#1}\egroup}
2290 %

```

\mpnormalvfootnote And a somewhat different version for minipages.

```

2291 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\mpnormalvfootnote}[2]{%
2292 \global\setbox\@nameuse{mp#1footins}\vbox{%
2293 \unvbox\@nameuse{mp#1footins}
2294 \noindent\csuse{Xhooknote@#1}%
2295 \csuse{Xnotefontsize@#1}%
2296 \hsize\columnwidth
2297 \@parboxrestore
2298 \color@begingroup
2299 \csname #1footfmt\endcsname #2{#1}\color@endgroup}}
2300
2301 %

```

\normalfootfmt `\normalfootfmt` is a ‘normal’ macro to take the footnote line and page number information (see V.9 p. 82), and the desired text, and output what’s to be printed. Argument #1 contains the line and page number information and lemma font specifier; #2 is the lemma; #3 is the note’s text. This version is very rudimentary—it uses `\printlines` to print just the range of line numbers, followed by a square bracket, the lemma, and the note text.

```

2302

```

```

2303
2304 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmt}[4]{%
2305   \Xledsetnormalparstuff{#4}%
2306   \hangindent=\csuse{Xhangindent@#4}
2307   \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2308   \strut{\printlinefootnote{#1}{#4}}%
2309   {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemmafnt#1|#2}{#2}}%
2310   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcempty{
Xlemmaseparator@#4}%
2311     {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2312     {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}\relax%
2313   }}%
2314   #3\strut\par}
2315 %

```

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

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

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

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

```

2316 \newcommand*{\normalfootstart}[1]{%
2317 %

```

The first series of notes printed in a page can have a specific skip before it. In order to insert this specific skip without overlap the bottom margin of the page, Maïeul Rouquette have defined an algorithm explained in XVIII p. 186. Here is part of this algorithm, when the block of notes are ready to be printed.

```

2318   \ifdimequal{0pt}{\preXnotes@}{}%
2319   {%
2320     \iftoggle{preXnotes@}{%
2321       \togglefalse{preXnotes@}%
2322       \skip\csname #1footins\endcsname=%
2323       \dimexpr\csuse{preXnotes@}+\csuse{Xafterterrule@#1}\relax%
2324     }%
2325   }%

```



```

2326 }%
2327 \vskip\skip\csize #1footins\endcsname%
2328 %

```

And now, the problem of left and right skip for notes. Especially when using one feature of `reledpar` which allows to have the footnotes horizontal size as the size of columns printed by `\Columns`. Read XV p. 184 for the general description of the problem.

```

2329 \leftskipOpt \rightskipOpt
2330 \ifl@dpairing\else%
2331   \hsize=\old@hsize%
2332 \fi%
2333 \setXnoteswidthliketwocolumns@{#1}%
2334 \setXnotespositionliketwocolumns@{#1}%
2335 %

```

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

```

2336 \print@Xfootnoterule{#1}%
2337 \noindent\leavevmode}
2338 %

```

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

```

2339 \newcommand*{\normalfootgroup}[1]{%
2340   {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}%
2341   \unvbox\csize #1footins\endcsname%
2342   \hsize=\old@hsize%
2343 }%
2344
2345 %

```

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

```

2346 \unless\ifnoledgroup@
2347 \newcommand*{\mpnormalfootgroup}[1]{%
2348   \vskip\skip\@nameuse{mp#1footins}
2349   \ifl@dpairing\ifparledgroup%
2350     \leavevmode\marks\parledgroup@{begin}%
2351     \marks\parledgroup@series{#1}%
2352     \marks\parledgroup@type{Xfootnote}%
2353   \fi\fi\normalcolor%
2354   \ifparledgroup%
2355     \ifl@dpairing%
2356     \else%
2357       \setXnoteswidthliketwocolumns@{#1}%
2358       \setXnotespositionliketwocolumns@{#1}%
2359       \print@Xfootnoterule{#1}%
2360     \fi%

```

```

2361 \else%
2362   \setXnoteswidthliketwocolumns@{#1}%
2363   \setXnotespositionliketwocolumns@{#1}%
2364   \printXfootnoterule{#1}%
2365 \fi%
2366 \setlength{\parindent}{0pt}
2367 {\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}
2368 \unvbox\csname mp#1footins\endcsname}}
2369 \fi
2370 %

```

XII.6.3 Paragraphed footnotes

The paragraphed-footnote option reformats all the footnotes of one series for a page into a single paragraph; this is especially appropriate when the notes are numerous and brief. The code is based on *The TeXbook*, pp. 398–400, with alterations for our environment. This algorithm uses a considerable amount of save-stack space: a TeX of ordinary size may not be able to handle more than about 100 notes of this kind on a page.

\Xarrangement@paragraph The `\Xarrangement@paragraph` macro sets up everything for one series of the footnotes so that they will be paragraphed; it takes the series letter as argument. We include the setting of `\count\footins` to 1000 for the footnote series just in case user is switching to paragraphed footnotes after having columnar ones, since they change this value (see below).

The argument of `\Xarrangement@footparagraph` is the letter denoting the series of notes to be paragraphed.

```

2371 \newcommand*{\Xarrangement@paragraph}[1]{%
2372   \csgdef{series@display#1}{paragraph}
2373   \expandafter\newcount\csname #1prevpage@num\endcsname
2374   \expandafter\let\csname #1footstart\endcsname=\parafootstart
2375   \expandafter\let\csname v#1footnote\endcsname=\paravfootnote
2376   \expandafter\let\csname #1footfmt\endcsname=\parafootfmt
2377   \expandafter\let\csname #1footgroup\endcsname=\parafootgroup
2378   \count\csname #1footins\endcsname=1000
2379   \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
side only
2380   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2381   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2382   \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2383   \para@footsetup{#1}
2384 %

```

And the extra setup for minipages.

```

2385 \ifnoledgroup@else
2386   \expandafter\let\csname mpv#1footnote\endcsname=\mpparavfootnote
2387   \expandafter\let\csname mp#1footgroup\endcsname=\mpparafootgroup
2388   \count\csname mp#1footins\endcsname=1000
2389   \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}

```

```

2390 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2391 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2392 \fi
2393 }
2394 %

```

\footfudgefiddle For paragraphed footnotes \TeX has to estimate the amount of space required. If it underestimates this then the notes may get too long and run off the bottom of the text block. `\footfudgefiddle` can be increased from its default 64 (say, to 70) to increase the estimate.

```

2395 \providecommand{\footfudgefiddle}{64}
2396 %

```

\para@footsetup `\footparagraph` calls the `\para@footsetup` macro to calculate a special fudge factor, which is the ratio of the `\baselineskip` to the `\hsize`. We assume that the proper value of `\baselineskip` for the footnotes (normally 9pt) has been set already. The argument of the macro is again the note series letter.

Peter Wilson thinks that `\columnwidth` should be used here for \LaTeX not `\hsize`. Peter Wilson have also included `\footfudgefiddle`.

```

2397 \newcommand*{\para@footsetup}[1]{\csuse{Xnotefontsize@#1}
2398 \setXnoteswidthliketwocolumns@{#1}%
2399 \dimen0=\baselineskip
2400 \multiply\dimen0 by 1024
2401 \divide \dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\
relax
2402 \csxdef{#1footfudgefactor}{%
2403 \expandafter\strip@pt\dimen0 }}
2404
2405 %

```

`\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.

```

2406 \newcommand*{\parafootstart}[1]{%
2407 \rightskip=0pt \leftskip=0pt%
2408 \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
2409 \ifdimequal{0pt}{\preXnotes@}{}%
2410 {%
2411 \iftoggle{preXnotes@}{%
2412 \togglefalse{preXnotes@}%
2413 \skip\csname #1footins\endcsname=%

```

```

2414         \dimexpr\csuse{preXnotes@}+\csuse{Xaftererrule@#1}\relax%
2415     }%
2416     {}%
2417 }%
2418 \vskip\skip\csname #1footins\endcsname%
2419 \setXnoteswidthliketwocolumns@{#1}%
2420 \setXnotespositionliketwocolumns@{#1}%
2421 \print@Xfootnoterule{#1}%%
2422 \noindent\leavevmode}
2423 %

```

`\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.²⁸

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.²⁹ 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.³⁰ 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

²⁸Michael Downes, 'Line Breaking in `\unhboxed` Text', *TUGboat* 11 (1990), pp. 605–612.

²⁹See *The TeXbook*, p. 455 (editions after January 1990).

³⁰Wayne 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.

really can't put an explicit line-break into a note built in a `\vbox` the way we are doing.³¹ In other words, be very careful not to use `\break`, or `\penalty-10000`, or any equivalent inside your para-footnote. If you do, most of the note will probably disappear. You *are* allowed to make strong suggestions; in fact `\penalty-9999` will be quite okay. Just do not make the break mandatory. We have not applied any of Michael's solutions here, since we feel that the problem is exiguous, and `reledmac` is quite baroque enough already. If you think you are having this problem, look up Michael's solutions.

One more thing; we set `\leftskip` and `\rightskip` to zero. This has the effect of neutralizing any such skips which may apply to the main text (cf. XII.6.2 p. 144 above). We need to do this, since `\footfudgefactor` is calculated on the assumption that the notes are `\hsize` wide.

So, finally, here is the modified foot-paragraph code, which sets the footnote in vertical mode so that language and discretionary nodes are included.

```

2424 \newcommand*{\paravfootnote}[2]{%
2425   \insert\csname #1footins\endcsname
2426   \bgroup
2427     \csuse{Xnotefontsize@#1}
2428     \footsplitskips
2429     \setbox0=\vbox{\hsize=\maxdimen
2430       \noindent\csuse{Xhooknote@#1}%
2431       \csname #1footfmt\endcsname #2{#1}}%
2432     \setbox0=\hbox{\Xunvxh{0}{#1}}%
2433     \dp0=0pt
2434     \ht0=\csname #1footfudgefactor\endcsname\wd0
2435   %

```

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

```

2436   \if@RTL\noindent \leavevmode\fi\box0%
2437   \penalty0
2438   \egroup}
2439
2440 %

```

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.

³¹'Line Breaking', p. 610.

```

2441 \newcommand*\mpparavfootnote}[2]{%
2442   \global\setbox\@nameuse{mp#1footins}\vbox{%
2443     \unvbox\@nameuse{mp#1footins}%
2444     \csuse{Xnotefontsize@#1}%
2445     \footsplitskips
2446     \setbox0=\vbox{\hsize=\maxdimen
2447       \noindent\color@begingroup%
2448       \csuse{Xbhooknote@#1}%
2449       \csname #1footfmt\endcsname #2{#1}\color@endgroup}%
2450     \setbox0=\hbox{\Xunvxh{0}{#1}}%
2451     \dp0=\z@
2452     \ht0=\csname #1footfudgefactor\endcsname\wd0
2453     \box0
2454     \penalty0
2455   }}
2456
2457 %

```

\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 T_EX automatically attaches to the end of paragraphs. When T_EX 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`.

```

2458 \newcommand*\Xunvxh}[2]{%
2459   \setbox0=\vbox{\unvbox#1}%
2460   \global\setbox1=\lastbox}%
2461   \unhbox1
2462   \unskip           % remove \rightskip,
2463   \unskip           % remove \parfillskip,
2464   \unpenalty        % remove \penalty of 10000,
2465   \hskip\csuse{Xafternote@#2}} % but add the glue to go between the notes
2466
2467 %

```

\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).

```

2468 \newcommand*\parafootfmt}[4]{%
2469   \Xinsertparafootsep{#4}%
2470   \ledsetnormalparstuff@common%
2471   \printlinefootnote{#1}{#4}%
2472   {\nottoggle{Xlemmadisablefontselection@#4}}{\select@lemmafont#1|#2}{#2}}%
2473   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcempty{
Xlemmaseparator@#4}}%

```

```

2474 {\hskip\csuse{Xinplaceoflemmaseparator@#4}}}%
2475 {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}}%
2476 }}%
2477 #3\penalty-10 }
2478 %

```

Note that in the above definition, the penalty of -10 encourages a line break between notes, so that notes have a slight tendency to begin on new lines. The `\Xinsertparafootsep` command is used to insert the `\Xparafootsep@series` between each note in the *same* page.

\parafootgroup This footgroup code is modelled on the macros in *The TeXbook*, p. 399. The only difference is the `\unpenalty` in `\makehboxofhboxes`, which is there to remove the penalty of 0 which was added to the end of each footnote by `\para@vfootnote`.

The call to `\Xnotefontsize@<s>` is to ensure that the correct `\baselineskip` for the footnotes is used. The argument is the note series letter.

```

2479 \newcommand*{\parafootgroup}[1]{%
2480 \unvbox\curname #1footins\endcurname
2481 \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
2482 \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2483 \makehboxofhboxes
2484 \setbox0=\hbox{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\
unhbox0 \removehboxes}%
2485 \csuse{Xnotefontsize@#1}
2486 \unhbox0\par%
2487 \global\hsize=\old@hsize%
2488 }%
2489
2490 %

```

\mpparafootgroup The minipage version.

```

2491 \newcommand*{\mpparafootgroup}[1]{%
2492 \setXnoteswidthliketwocolumns@{#1}%
2493 \vskip\skip\@nameuse{mp#1footins}
2494 \ifl@dpairing\ifparledgroup%
2495 \leavevmode\marks\parledgroup@{begin}%
2496 \marks\parledgroup@series{#1}%
2497 \marks\parledgroup@type{Xfootnote}%
2498 \fi\fi\normalcolor
2499 \ifparledgroup%
2500 \ifl@dpairing%
2501 \else%
2502 \setXnoteswidthliketwocolumns@{#1}%
2503 \setXnotespositionliketwocolumns@{#1}%
2504 \print@Xfootnoterule{#1}%%
2505 \fi%
2506 \else%

```

```

2507 \setXnoteswidthliketwocolumns@{#1}%
2508 \setXnotespositionliketwocolumns@{#1}%
2509 \print@Xfootnoterule{#1}%
2510 \fi%
2511 \unvbox\csname mp#1footins\endcsname
2512 \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
2513 \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2514 \makehboxofhboxes
2515 \setbox0=\hbox{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\
unhbox0 \removehboxes}%
2516 \csuse{Xnotefontsize@#1}
2517 \unhbox0\par}}
2518
2519 %

```

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

```

\makehboxofhboxes20 \newcommand*{\makehboxofhboxes}{\setbox0=\hbox{}%
\removehboxes21 \loop
2522 \unpenalty
2523 \setbox2=\lastbox
2524 \ifhbox2
2525 \setbox0=\hbox{\box2\unhbox0}%
2526 \repeat}
2527
2528 \newcommand*{\removehboxes}{\setbox0=\lastbox
2529 \ifhbox0{\removehboxes}\unhbox0 \fi}
2530
2531 %

```

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

```

\prevpage@num32 \newcommand{\Xinsertparafootsep}[1]{%
\Xinsertparafootsep33 \ifnumequal{\csuse{#1prevpage@num}}{\page@num}%
2534 {\ifcsdef{prevline#1}% Be sur \prevline#1 exists.
2535 {\ifnumequal{\csuse{prevline#1}}{\line@num}%
2536 {\ifcsempy{Xsymlinenum@#1}{\csuse{Xparafootsep@#1}}{}}%
2537 {\csuse{Xparafootsep@#1}}%
2538 }%
2539 {\csuse{Xparafootsep@#1}}%
2540 }%
2541 {}%
2542 \global\csname #1prevpage@num\endcsname=\page@num%
2543 }
2544 %

```


XII.6.4 Columnar footnotes

Common tools

`\rigidbalance` We will now define macros for three-column notes and two-column notes. Both sets of macros will use `\rigidbalance`, which splits a box (#1) into a number (#2) of columns, each with a space (#3) between the top baseline and the top of the `\vbox`. The `\dosplits` `\rigidbalance` macro is taken from *The TeXbook*, p. 397, with a slight change to the `\splitoff` syntax of the arguments so that they do not depend on white space. Note also the extra unboxing in `\splitoff`, which allows the new `\vbox` to have its natural height as it goes into the alignment.

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

```

2545 \newcount\@k \newdimen\@h
2546 \newcommand*\rigidbalance}[3]{\setbox0=\box#1 \@k=#2 \@h=#3
2547 \@@line{\splittopskip=\@h \vbadness=\@M \hfilneg
2548 \valign{##\vfil\cr\dosplits}}
2549
2550 \newcommand*\dosplits{\ifnum\@k>0 \noalign{\hfil}\splitoff
2551 \global\advance\@k-1\cr\dosplits\fi}
2552
2553 \newcommand*\splitoff{\dimen0=\ht0
2554 \divide\dimen0 by\@k \advance\dimen0 by\@h
2555 \setbox2 \vsplit0 to \dimen0
2556 \unvbox2 }
2557
2558 %

```

Three columns

```

\Xarrangement@threecol 2549 \newcommand*\Xarrangement@threecol}[1]{%
2550 \csgdef{series@display#1}{threecol}
2551 \expandafter\let\csname v#1footnote\endcsname=\threecolvfootnote
2552 \expandafter\let\csname #1footfmt\endcsname=\threecolfootfmt
2553 \expandafter\let\csname #1footgroup\endcsname=\threecolfootgroup
2554 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2555 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2556 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2557 \threecolfootsetup{#1}
2558 %

```

The additional setup for minipages.

```

2569 \ifnoledgroup@else
2570 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2571 \expandafter\let\csname mp#1footgroup\endcsname=\mpthreecolfootgroup
2572 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2573 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2574 \mpthreecolfootsetup{#1}

```

```

2575 \fi
2576 }
2577
2578 %

```

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

`\threecolfootsetup` The `\threecolfootsetup` macro calculates and sets some numbers for three-column footnotes.

We set the `\count` of the foot insert to 333. Each footnote can be thought of as contributing only one third of its height to the page, since the footnote insertion has been made as a long narrow column, which then gets trisected by the `\rigidbalance` routine (inside `\threecolfootgroup`). These new, shorter columns are saved in a box, and then that box is *put back* into the footnote insert, replacing the original collection of the footnotes. This new box is, therefore, only about a third of the height of the original one.

The `\dimen` value for this note series has to change in the inverse way: it needs to be three times the actual limit on the amount of space these notes are allowed to fill on the page, because when \TeX is accumulating material for the page and checking that limit, it does not apply the `\count` scaling.

```

2579 \newcommand*{\threecolfootsetup}[1]{%
2580   \count\csname #1footins\endcsname 333
2581   \csxdef{default@#1footins}{333}%Use this to confine the notes to one
side only
2582   \multiply\dimen\csname #1footins\endcsname \thr@@}
2583 %

```

`\mpthreecolfootsetup` The setup for minipages.

```

2584 \newcommand*{\mpthreecolfootsetup}[1]{%
2585   \count\csname mp#1footins\endcsname 333
2586   \multiply\dimen\csname mp#1footins\endcsname \thr@@}
2587
2588 %

```

`\threecolvfootnote` `\threecolvfootnote` is the `\vfootnote` command for three-column notes. The call to `\Xnotefontsize@<s>` ensures that the `\splittopskip` and `\splitmaxdepth` take their values from the right `\strutbox`: the one used in a footnotes. Note especially the importance of temporarily reducing the `\hsize` to 0.3 of its normal value. This determines the widths of the individual columns. So if the normal `\hsize` is, say, 10 cm, then each column will be $0.3 \times 10 = 3$ cm wide, leaving a gap of 1 cm spread equally between columns (i.e., .5 cm between each).

The arguments are #1 the note series letter and #1 the full text of the note (including numbers, lemma and text).

```

2589 \notbool{parapparatus@}{\newcommand*}{\newcommand*}{\threecolvfootnote}[2]{%
2590   \insert\csname #1footins\endcsname\bgroup

```

```

2591 \csuse{Xnotefontsize@#1}
2592 \footsplitskip
2593 \csname #1footfmt\endcsname #2{#1}\egroup}
2594 %

```

\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.

```

2595 \notbool{parapparatus0}{\newcommand*{\threecolfootfmt}[4]{%
2596 \normal@pars
2597 \hsize \csuse{Xhsizethreecol@#4}
2598 \nottoggle{Xparindent@#4}{\parindent=\z@}{%
2599 \tolerance=5000
2600 \hangindent=\csuse{Xhangindent@#4}
2601 \leavevmode
2602 \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2603 \@tempdima=\parindent%
2604 \csuse{Xcolalign@#4}%
2605 \parindent=\@tempdima%
2606 \strut{\printlinefootnote{#1}{#4}}%
2607 {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemmafont#1|#2}{#2}}%
2608 \iftoggle{nosep0}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsempy{
Xlemmaseparator@#4}%
2609 {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2610 {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}%
2611 }}%
2612 #3\strut\par\allowbreak}
2613 %

```

\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.

```

2614 \newcommand*{\threecolfootgroup}[1]{\csuse{Xnotefontsize@#1}%
2615 \noindent\csuse{Xtxtbeforenotes@#1}}\par%
2616 \splittopskip=\ht\strutbox
2617 \expandafter
2618 \rigidbalance\csname #1footins\endcsname \thr@@ \splittopskip}
2619 %

```

\mpthreecolfootgroup The setup for minipages.

```

2620 \newcommand*\mpthreecolfootgroup}[1]{%
2621 \vskip\skip\@nameuse{mp#1footins}
2622 \ifl@dpairing\ifparledgroup%
2623 \leavevmode\marks\parledgroup@{begin}%
2624 \marks\parledgroup@series{#1}%
2625 \marks\parledgroup@type{Xfootnote}%
2626 \fi\fi\normalcolor
2627 \ifparledgroup%
2628 \ifl@dpairing%
2629 \else%
2630 \setXnoteswidthliketwocolumns@{#1}%
2631 \setXnotespositionliketwocolumns@{#1}%
2632 \print@Xfootnoterule{#1}%
2633 \fi%
2634 \else%
2635 \setXnoteswidthliketwocolumns@{#1}%
2636 \setXnotespositionliketwocolumns@{#1}%
2637 \print@Xfootnoterule{#1}%
2638 \fi%
2639 {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}\par
2640 \splittopskip=\ht\strutbox
2641 \expandafter
2642 \rigidbalance\csname mp#1footins\endcsname \thr@@ \splittopskip}}
2643
2644 %

```

Two columns

```

\Xarrangement@twocol 2645 \newcommand*\Xarrangement@twocol}[1]{%
2646 \csgdef{series@display#1}{twocol}
2647 \expandafter\let\csname v#1footnote\endcsname=\twocolvfootnote
2648 \expandafter\let\csname #1footfmt\endcsname=\twocolfootfmt
2649 \expandafter\let\csname #1footgroup\endcsname=\twocolfootgroup
2650 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2651 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2652 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2653 \twocolfootsetup{#1}
2654 %

```

The additional setup for minipages.

```

2655 \ifnoledgroup@else
2656 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2657 \expandafter\let\csname mp#1footgroup\endcsname=\mptwocolfootgroup
2658 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2659 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2660 \mptwocolfootsetup{#1}
2661 \fi
2662 }
2663

```

```

2664 %

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

2665 \newcommand*{\twocolfootsetup}[1]{%
2666 \count\csname #1footins\endcsname 500
2667 \csxdef{default@#1footins}{500}%Use this to confine the notes to one
side only
2668 \multiply\dimen\csname #1footins\endcsname \tw@}
2669 %

2670 \notbool{parapparatus@}{\newcommand*{\newcommand}{\twocolvfootnote}[2]{\
insert\csname #1footins\endcsname\bgroup
2671 \csuse{Xnotefontsize@#1}
2672 \footplitskips
2673 \csname #1footfmt\endcsname #2{#1}\egroup}
2674 %

2675 \notbool{parapparatus@}{\newcommand*{\newcommand}{\twocolfootfmt}[4]{% 4th
arg is optional, for backward compatibility
2676 \normal@pars
2677 \hsiz \csuse{Xhsizetwocol@#4}
2678 \nottoggle{Xparindent@#4}{\parindent=\z@}{\}
2679 \tolerance=5000
2680 \hangindent=\csuse{Xhangindent@#4}
2681 \leavevmode
2682 \everypar{\hangindent=\csuse{Xhangindent@#4}}}%
2683 \@tempdima=\parindent%
2684 \csuse{Xcolalign@#4}%
2685 \parindent=\@tempdima%
2686 \strut{\printlinefootnote{#1}{#4}}}%
2687 {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemmafnt#1|#2}{#2}}}%
2688 \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsemt{
Xlemmaseparator@#4}%
2689 {\hskip\csuse{Xinplaceoflemmaseparator@#4}}}%
2690 {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}%
2691 }}%
2692 #3\strut\par\allowbreak}
2693 %

2694 \newcommand*{\twocolfootgroup}[1]{\csuse{Xnotefontsize@#1}
2695 \noindent\csuse{Xtxtbeforenotes@#1}}\par%
2696 \splittopskip=\ht\strutbox
2697 \expandafter
2698 \rigidbalance\csname #1footins\endcsname \tw@ \splittopskip}
2699 %
2700 %

```

`\mptwocolfootsetup` The versions for minipages.

`\mptwocolfootgroup`

```

2701 \newcommand*\mptwocolfootsetup}[1]{%
2702   \count\csname mp#1footins\endcsname 500
2703   \multiply\dimen\csname mp#1footins\endcsname \tw@}
2704 %

2705 \newcommand*\mptwocolfootgroup}[1]{%
2706   \vskip\skip\@nameuse{mp#1footins}
2707   \ifl@dpairing\ifparledgroup%
2708     \leavevmode\marks\parledgroup@{begin}%
2709     \marks\parledgroup@series{#1}%
2710     \marks\parledgroup@type{Xfootnote}%
2711   \fi\fi\normalcolor
2712   \ifparledgroup%
2713     \ifl@dpairing%
2714     \else%
2715       \setXnoteswidthliketwocolumns@{#1}%
2716       \setXnotespositionliketwocolumns@{#1}%
2717       \print@Xfootnoterule{#1}%
2718     \fi%
2719   \else%
2720     \setXnoteswidthliketwocolumns@{#1}%
2721     \setXnotespositionliketwocolumns@{#1}%
2722     \print@Xfootnoterule{#1}%
2723   \fi%
2724   {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}\par
2725   \splittopskip=\ht\strutbox
2726   \expandafter
2727   \rigidbalance\csname mp#1footins\endcsname \tw@ \splittopskip}}
2728
2729 %
```

XII.7 Critical notes presentation

Here, we define some commons macro which are used in order to print a critical notes, that is a note with 1) line number 2) lemma 3) lemma separator 4) text associated to the lemma.

XII.7.1 Font tools

`\endashchar`

`\fullstop`

`\rbracket`

The fonts that are used for printing notes might not have the character mapping we expect: for example, the Computer Modern font that contains old-style numerals does not contain an en-dash or square brackets, and its period and comma are in odd locations. To allow use of the standard footnote macros with such fonts, we use the following macros for certain characters.

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

`\normalfootfmt` and the other footnote macros for controlling the format of the footnotes.

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

```

2730 \def\endashchar{\textnormal{--}}
2731 \newcommand*\fullstop{\textnormal{.}}
2732 \newcommand*\rbracket{\textnormal{%
2733   \csuse{text\csuse{footnote@lang}}{%
2734     \ifluatex%
2735       \ifdefstring{\footnote@luatextextdir}{TRT}{\thinspace[]}{\thinspace
2736     }%
2737     \else%
2738       \thinspace}%
2739     \fi}%
2740 }
2741
2742 %

```

XII.7.2 Pstart number in footnote

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

```

2743 \newcommand{\printpstart}[0]{%
2744   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
2745     l@dprintingcolumns}}{%
2746     \ifledRcol%
2747       \thepstartR%
2748     \else%
2749       \thepstartL%
2750     \fi%
2751   }%
2752   \thepstart%
2753 }
2754 %

```

XII.7.3 Line number printing

`\printlinefootnote` The `\printlinefootnote` macro is called in each `\<type>footfmt` command. It controls whether the line number is printed or not, according to the series options. Its first argument is the information about lines; its second is the series of the footnote. The printing of the line number is shared in `\printlinefootnotenumbers`.

```

2755 \newcommand{\printlinefootnote}[2]{%
2756   \def\extractline@##1|##2|##3|##4|##5|##6|##7|{##2}%
2757   \def\extractsubline@##1|##2|##3|##4|##5|##6|##7|{##3}%
2758   \def\extractendline@##1|##2|##3|##4|##5|##6|##7|{##5}%

```

```

2759 \def\extractendsubline@##1|##2|##3|##4|##5|##6|##7|{##6}%
2760 \iftoggle{Xnumberonlyfirstintwolines@#2}{%
2761   \edef\lineinfo@{\extractline@ #1| - \extractsubline@ #1| - \
extractendline@ #1| - \extractendsubline@ #1|}%
2762   }%
2763   {%
2764     \edef\lineinfo@{\extractline@ #1| - \extractsubline@ #1|}%
2765     }%
2766   \iftoggle{nonum@}{%Try if the line number must printed for this specific
not (by default, yes)
2767     \hspace{\csuse{Xinplaceofnumber@#2}}}%
2768     }%
2769     {%
2770       {%
2771         \iftoggle{Xnonumber@#2}%Try if the line number must printed (by
default, yes)
2772         {%
2773           \hspace{\csuse{Xinplaceofnumber@#2}}}%
2774           }%
2775           {%
2776             {\iftoggle{Xnumberonlyfirstinline@#2}% If for this series the
line number must be printed only in the first time.
2777               {%
2778                 \ifcsdef{prevline#2}%
2779                 {%Be sure the \prevline exists.
2780                 \ifcsequal{prevline#2}{\lineinfo@}%Try it
2781                 {%
2782                   \ifcsequal{Xsymlinenumber@#2}%
2783                   {%
2784                     \hspace{\csuse{Xinplaceofnumber@#2}}}%
2785                     }%
2786                     {\printsymlinefootnotearea{#2}}}%
2787                     }%
2788                     {%
2789                       \printlinefootnotearea{#1}{#2}%
2790                       }%
2791                     }%
2792                     {%
2793                       \printlinefootnotearea{#1}{#2}%
2794                       }%
2795                     }%
2796                     {%
2797                       \printlinefootnotearea{#1}{#2}%
2798                       }%
2799                       \csxdef{prevline#2}{\lineinfo@}%
2800                       }%
2801                     }%
2802                     }%
2803                     }%
2804   }

```


2805 %

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

```
2806 \newcommand{\printsymlinefootnotearea}[1]{%
2807   \hspace{\csuse{Xbeforesymlinenum@#1}}%
2808   \csuse{Xnotenumfont@#1}%
2809   \ifdimequal{\csuse{Xboxsymlinenum@#1}}{\z@}%
2810     {\csuse{Xsymlinenum@#1}}%
2811     {\hbox to \csuse{Xboxsymlinenum@#1}%
2812       {\csuse{Xsymlinenum@#1}\hfill}}%
2813   }%
2814   \hspace{\csuse{Xaftersymlinenum@#1}}%
2815 }%
2816 %
```

\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.)

```
2817 \newcommand{\printlinefootnotearea}[2]{%
2818   \printXbeforenumber{#2}%
2819   \csuse{Xnotenumfont@#2}%
2820   \boxfootnotenumbers{#1}{#2}%
2821   \printXafternumber{#2}%
2822 }%
2823 %
```

\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.

```
2824 \newcommand{\boxfootnotenumbers}[2]{%
2825   \ifdimequal{\csuse{Xboxlinenum@#2}}{0pt}{%
2826     \printlinefootnotenumbers{#1}{#2}%
2827   }%
2828   {%
2829     \hbox to \csuse{Xboxlinenum@#2}%
2830     {%
2831       \IfSubStr{RC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2832       \printlinefootnotenumbers{#1}{#2}%
2833       \IfSubStr{LC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2834     }%
2835   }%
2836 }%
2837 %
```

\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.

```

2838 \newcommand{\printlinefootnotenumbers}[2]{%
2839   \xdef\@currentseries{#2}%
2840   \ifboolexpr{%
2841     (togl{Xpstart@#2} and bool{numberpstart})%
2842     or togl{Xpstarteverytime@#2}}}%
2843   {\printpstart}{}%
2844   \iftoggle{Xstanza@#2}{%
2845     \ifnumberstanza%
2846       \printstanza%
2847       \csuse{Xstanzaseparator@#2}%
2848     \fi%
2849   }{ }%
2850   \iftoggle{Xonlypstart@#2}{ }{\printlines#1|}%
2851 }%
2852 %

```

\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.)

```

2853 \newcommand{\printXbeforenumber}[1]{%
2854   \hspace{\csuse{Xbeforenumber@#1}}%
2855 }%
2856 %

```

\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

```

2857 \newcommand{\printXafternumber}[1]{%
2858   \iftoggle{Xnonbreakableafternumber@#1}{\nobreak}{ }%
2859   \hspace{\csuse{Xafternumber@#1}}%
2860 }%
2861 %

```

If we have decided to print the line number in a specific notes, the `\printlines` macro prints the line numbers for a note—which, in the general case, is a rather complicated task. The seven parameters of the argument are the line numbers as stored in `\l@d@nums`, in the form described on V.9 p. 82: the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma.

edmac' creator have defined six boolean in order to know which component of line number description we have to print:

- `\ifl@d@pnum` for page numbers;
- `\ifl@d@ssub` for starting sub-line;

- `\ifl@d@elin` for ending line;
- `\ifl@d@esl` for ending sub-line; and
- `\ifl@d@dash` for the dash between the starting and ending groups.

There is no boolean for the line number because it is always printed.

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@pnum62 \newif\ifl@d@pnum
\ifl@d@ssub63 \newif\ifl@d@ssub
\ifl@d@elin64 \newif\ifl@d@elin
\ifl@d@esl65 \newif\ifl@d@esl
\ifl@d@dash66 \newif\ifl@d@dash
\ifl@d@Xtwolines67 \newif\ifl@d@Xtwolines%
\ifl@d@Xmorethantwolines68 \newif\ifl@d@Xmorethantwolines%
69 %

\l@d@parsefootspec \l@d@parsefootspec{<spec>}{<lemma>}{<text>} parses a footnote specification. <lemma>
\l@d@p@rsefootspec and <text> are the lemma and text respectively. <spec> is the line and page num-
\l@d@p@rsefootspec ber and lemma font specifier in \l@d@nums style format. The real work is done by
\l@d@p@rsefootspec \l@d@p@rsefootspec which defines macros holding the numeric values. Just a reminder
\l@d@p@rsefootspec of the arguments:
\l@d@p@rsefootspec \printlines #1 | #2 | #3 | #4 | #5 | #6 | #7
\l@d@p@rsefootspec \printlines start-page | line | subline | end-page | line | subline | font
\l@d@p@rsefootspec \l@d@p@rsefootspec
2870 \newcommand*{\l@d@parsefootspec}[3]{\l@d@p@rsefootspec#1|}
2871 \def\l@d@p@rsefootspec#1|#2|#3|#4|#5|#6|#7|{%
2872 \gdef\l@d@p@rsefootspec#1|{%
2873 \gdef\l@d@p@rsefootspec#2|{%
2874 \gdef\l@d@p@rsefootspec#3|{%
2875 \gdef\l@d@p@rsefootspec#4|{%
2876 \gdef\l@d@p@rsefootspec#5|{%
2877 \gdef\l@d@p@rsefootspec#6|{%
2878 }
2879 %

```

Initialise the several number value macros.

```

2880 \def\l@d@p@rsefootspec#1|{%
2881 \def\l@d@p@rsefootspec#2|{%
2882 \def\l@d@p@rsefootspec#3|{%
2883 \def\l@d@p@rsefootspec#4|{%
2884 \def\l@d@p@rsefootspec#5|{%
2885 \def\l@d@p@rsefootspec#6|{%
2886 }
2887 %

```

\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`.

```
2888 \newcommand*\setprintlines}[6]{%
2889   \l@dpnumfalse \l@ddashfalse
2890   %
```

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

```
2891   \ifbypage@
2892     \ifnum#4=#1 \else
2893       \l@dpnumtrue
2894       \l@ddashtrue
2895     \fi
2896   \fi
2897   %
```

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.

```
2898   \ifl@dpnum \l@d@elintrue \else \l@d@elinfalse \fi
2899   \ifnum#2=#5 \else
2900     \l@d@elintrue
2901     \l@ddashtrue
2902   \fi
2903   %
```

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

```
2904   \l@d@ssubfalse
2905   \ifnum#3=0 \else
2906     \l@d@ssubtrue
2907   \fi
2908   %
```

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.

```
2909   \l@d@eslfalse
2910   \ifnum#6=0 \else
2911     \ifnum#6=#3
2912       \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
2913     \else
2914       \l@d@esltrue
2915       \l@ddashtrue
2916     \fi
2917   \fi%
2918   %
```

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

```

2919 \ifl@dash%
2920 \ifbool{expr}{togl{fulllines@} or test{\ifcsemt{Xtwolines@
@currentseries}}}%
2921 {}%
2922 {%
2923 \setistwofollowinglines{#1}{#2}{#4}{#5}%
2924 \ifbool{expr}%
2925 (%
2926     togl {Xtwolinesbutnotmore@ \@currentseries}%
2927     and not%
2928     (%
2929         bool {istwofollowinglines@}%
2930     )%
2931 )%
2932 or%
2933 (%
2934     (not test{\ifnumequal{#1}{#4}})%
2935     and togl{Xtwolinesonlyinsamepage@ \@currentseries}%
2936 )%
2937 }%
2938 {}%
2939 {%
2940 \l@d@dashfalse%
2941 \l@d@Xtwolinestrue%
2942 \l@d@elinfalse%
2943 \l@d@eslfalse%
2944 \ifcsemt{Xmorethantwolines@ \@currentseries}%
2945 {%
2946     {\ifistwofollowinglines@ \else%
2947         \l@d@Xmorethantwolinestrue%
2948     }%
2949 }%
2950 }%
2951 }%
2952 \fi%
2953 %

```

End of \setprintlines.

```

2954 }%
2955 %

```

\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.

```

2956 \newif\ifistwofollowinglines%
2957 \newcommand{\setistwofollowinglines}[4]{%
2958   \ifcsdef{lastlinenumberon@#1}%
2959     {\numdef{\tmp}{\csuse{lastlinenumberon@#1}}}%
2960     {\numdef{\tmp}{0}}%
2961   \istwofollowinglines@false%
2962   \ifnumequal{#4-#2}{1}%
2963     {\istwofollowinglines@true}%
2964     {\ifbypage%
2965       \ifnumequal{#3-#1}{1}%
2966       {%
2967         \ifnumequal{#2}{\tmp}%
2968         {\ifnumequal{#4}{1}{\istwofollowinglines@true}{}}%
2969         {}%
2970       }%
2971       {}%
2972     \fi%
2973   }%
2974 }%
2975 %

```

`\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`.

```

2976 \def\printlines#1|#2|#3|#4|#5|#6|#7|{%
2977   \begingroup%
2978   %

```

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

```

2979 \ifluatex%
2980   \edef\@tmp{\the\textdir}%
2981   \ifdefstring{\@tmp}{TLT}{\textdir TLT}%Test in order to prevent
2982   spurious space (bug #397)
2983   \fi%
2984 %

```

Decide which part of line number components we will print.

```

2984 \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
2985 %

```

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.

```

2986 \ifdimequal{\csuse{Xboxstartlinenum@\@currentseries}}{0pt}%
2987   {\bgroup}%
2988   {\leavevmode\hbox to \csuse{Xboxstartlinenum@\@currentseries}\bgroup\
hfill}%
2989   \ifl@d@pnum #1\fullstop\fi
2990   \linenumrep{#2}
2991   \ifl@d@ssub \fullstop \sublinenumrep{#3}\fi
2992   \egroup%
2993 %

```

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

```

2994 \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
2995   {\bgroup}%
2996   {\hbox to \csuse{Xboxendlinenum@\@currentseries}\bgroup}%
2997   \ifl@d@Xtwolines%
2998     \ifl@d@Xmorethantwolines%
2999       \csuse{Xmorethantwolines@\@currentseries}%
3000     \else%
3001       \csuse{Xtwolines@\@currentseries}%
3002     \fi%
3003   \else%
3004     \ifl@d@dash \endashchar\fi%
3005     \ifl@d@pnum #4\fullstop\fi%
3006     \ifl@d@elin \linenumrep{#5}\fi%
3007     \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumrep{#6}\fi%
3008   \fi%
3009   \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
3010     {}%
3011     {\hfill}%Prevent underfull hbox
3012   \egroup%
3013   \endgroup%
3014 }%
3015 %

```

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 num-

bered 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`.

```
3016 \providecommand*\multiplefootnotemarker}{3sp}
3017 \providecommand*\multfootsep{\textsuperscript{\normalfont,}}
3018
3019 %
```

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

```
3020 \providecommand*\m@mmf@prepare}{%
3021   \kern-\multiplefootnotemarker
3022   \kern\multiplefootnotemarker\relax}
3023 %
```

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

```
3024 \providecommand*\m@mmf@check}{%
3025   \ifdim\lastkern=\multiplefootnotemarker\relax
3026     \edef\@x@sf{\the\spacefactor}%
3027     \unkern
3028     \multfootsep
3029     \spacefactor\@x@sf\relax
3030   \fi}
3031
3032 %
```

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

```
3033 \@ifclassloaded{memoir}{-}{%
3034 %
```

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

```
3035 \apptocmd{\@footnotetext}{\m@mmf@prepare}{-}{-}
3036 %
```


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

```

3037
3038 \patchcmd{\@footnotemark}
3039   {\nobreak}
3040   {\m@mmf@check
3041   \nobreak
3042   }
3043   {}{}
3044 \patchcmd{\@footnotemark}
3045   {\@makefnmark}
3046   {\@makefnmark
3047   \m@mmf@prepare
3048   }
3049   {}{}
3050 %

```

Finished the modifications for the non-memoir case.

```

3051 }
3052
3053 %

```

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.

```

3054 \pretocmd{\@footnotetext}{%
3055   \ifnumberedpar@
3056   \edtext{}{\l@dbfnote{#1}}}%
3057   \else
3058   }{}{}
3059 \apptocmd{\@footnotetext}{\fi}{}{}%
3060 %

```

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

```

3061
3062 \newcommand{\l@dbfnote}[1]{%
3063   \ifnumberedpar@
3064   \gdef\@tag{#1\relax}%
3065   \ifledRcol%
3066     \xright@appenditem{\noexpand\vl@dbfnote{\expandonce\@tag}}{\@
3067     thefnmark}}}%
3068     \to\inserts@listR
3069     \global\advance\insert@countR \@ne%
3070   \else%

```

```

3070 \xright@appenditem{\noexpand\vl@dbfnote{\{expandonce\@tag\}}{\@theftmark}}%
3071 \to\inserts@list
3072 \global\advance\insert@count \@ne%
3073 \fi
3074 \fi\ignorespaces}
3075
3076 \newcommand{\vl@dbfnote}[2]{%
3077 \def\@theftmark{#2}%
3078 \@footnotetext{#1}%
3079 }%
3080 %

```

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`

```

3081 \newcommand*\prebodyfootmark{%
3082 \leavevmode
3083 \ifhmode
3084 \edef\@x@sf{\the\spacefactor}%
3085 \m@mmf@check
3086 \nobreak
3087 \fi}
3088 \newcommand*\postbodyfootmark{%
3089 \m@mmf@prepare
3090 \ifhmode\spacefactor\@x@sf\fi\relax}
3091
3092 %

```

XIII.4 Footnote arrangement

XIII.4.1 User level macro

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

```

3093 \newcommandx{\arrangementX}[2][1,usedefault]{%
3094 \def\do##1{%
3095 \csname arrangementX@#2\endcsname{##1}%
3096 }%
3097 \ifstrempy{#1}%
3098 {%
3099 \dolistloop{\@series}%
3100 }%
3101 {

```

```

3102 \docsvlist{#1}%
3103 }%
3104 }%
3105 %

```

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.

```

3106 \newcommand*{\normal@footnotemarkX}[1]{%
3107 \prebodyfootmark
3108 \@nameuse{bodyfootmark#1}%
3109 \postbodyfootmark}
3110
3111 %

```

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

```

3112 \newcommand*{\normalbodyfootmarkX}[1]{%
3113 \hbox{\textsuperscript{\normalfont\@nameuse{@thefnmark#1}}}}
3114 %

```

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

```

3115 \notbool{parapparatus}{\newcommand*{\newcommand}{\normalvfootnoteX}[2]{%
3116 \insert\@nameuse{footins#1}\bgroup
3117 \noindent\csuse{bhooknoteX@#1}%
3118 \csuse{notefontsizeX@#1}%
3119 \footsplitskips
3120 \ifl@dpairing\ifl@dpaging\else%
3121 \setnoteswidthliketwocolumnsX@{#1}%
3122 \fi\fi%
3123 \setnotesXpositionliketwocolumns@{#1}%
3124 \spaceskip=\z@skip \xspaceskip=\z@skip
3125 \csuse{\csuse{footnote@dir}}\@nameuse{footfmt#1}{#1}{#2}\egroup}
3126
3127 %

```

`\mpnormalvfootnoteX` The minipage version.

```

3128 \newcommand*{\mpnormalvfootnoteX}[2]{%
3129 \global\setbox\@nameuse{mpfootins#1}\vbox{%
3130 \unvbox\@nameuse{mpfootins#1}
3131 \noindent\csuse{bhooknoteX@#1}%
3132 \csuse{notefontsizeX@#1}%
3133 \hsize\columnwidth
3134 \@parboxrestore

```

```

3135 \color@begingroup
3136 \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}}
3137
3138 %

```

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

```

3139 \notbool{parapparatus@}{\newcommand*{\newcommand}{\normalfootfmtX}[2]{%
3140 \ifluatex%
3141 \texdir\footnote@luatexttexdir%
3142 \pardir\footnote@luatexpardir%
3143 \par%
3144 \fi%
3145 \protected@edef\currentlabel{%
3146 \@nameuse{@thefnmark#1}%
3147 }%
3148 \ledsetnormalparstuffX{#1}%
3149 \hangindent=\csuse{hangindentX@#1}%
3150 \everypar{\hangindent=\csuse{hangindentX@#1}}%
3151 {\csuse{notenumfontX@#1}\@nameuse{footfootmark#1}}\strut%
3152 #2\strut\par}}
3153
3154 %

```

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

```

3155 \newcommand*{\normalfootfootmarkX}[1]{%
3156 \textsuperscript{\@nameuse{@thefnmark#1}}}
3157
3158 %

```

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

```

3159 \newcommand*{\normalfootstartX}[1]{%
3160 \ifdimequal{Opt}{\prenotesX@}{}%
3161 {%
3162 \iftoggle{prenotesX@}{%
3163 \togglefalse{prenotesX@}%
3164 \skip\csname footins#1\endcsname=%
3165 \dimexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
3166 }%
3167 }%
3168 }%
3169 \vskip\skip\csname footins#1\endcsname%
3170 \leftskip=\z@
3171 \rightskip=\z@
3172 \ifl@dpairing\else%
3173 \hsize=\old@hsize%

```

```

3174 \fi%
3175 \setnoteswidthliketwocolumnsX@{#1}%
3176 \setnotesXpositionliketwocolumns@{#1}%
3177 \print@footnoteXrule{#1}%
3178 }%
3179
3180 %

```

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

```

3181 \let\normalfootnoteruleX=\footnoterule
3182
3183 %

```

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

```

3184 \newcommand*{\normalfootgroupX}[1]{%
3185 \unvbox\@nameuse{footins#1}%
3186 \hsize=\old@hsize%
3187 }%
3188
3189 %

```

`\mpnormalfootgroupX` The minipage version.

```

3190 \newcommand*{\mpnormalfootgroupX}[1]{%
3191 \vskip\skip\@nameuse{mpfootins#1}
3192 \ifl@dpairing\ifparledgroup%
3193 \leavevmode\marks\parledgroup@{begin}%
3194 \marks\parledgroup@series{#1}%
3195 \marks\parledgroup@type{footnoteX}%
3196 \fi\fi\normalcolor
3197 \ifparledgroup%
3198 \ifl@dpairing%
3199 \else%
3200 \setnoteswidthliketwocolumnsX@{#1}%
3201 \setnotesXpositionliketwocolumns@{#1}%
3202 \print@footnoteXrule{#1}%
3203 \fi%
3204 \else%
3205 \setnoteswidthliketwocolumnsX@{#1}%
3206 \setnotesXpositionliketwocolumns@{#1}%
3207 \print@footnoteXrule{#1}%
3208 \fi%
3209 \unvbox\@nameuse{mpfootins#1}}
3210
3211 %

```

```

\normalbfnoteX12
3213 \newcommand{\normalbfnoteX}[2]{%
3214   \ifnumberedpar@
3215   \ifledRcol%
3216   \ifluatex
3217     \footnotelang@lua[R]%
3218   \fi
3219   \@ifundefined{xpg@main@language}%if polyglossia
3220     {}%
3221     {\footnotelang@poly[R]}%
3222   \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
3223   \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\expandonce\
thisfootnote}}}%
3224
3225   \to\inserts@listR
3226   \global\advance\insert@countR \@ne%
3227 \else%
3228   \ifluatex
3229     \footnotelang@lua%
3230   \fi
3231   \@ifundefined{xpg@main@language}%if polyglossia
3232     {}%
3233     {\footnotelang@poly}%
3234   \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
3235   \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\expandonce\
thisfootnote}}}%
3236
3237   \to\inserts@list
3238   \global\advance\insert@count \@ne%
3239 \fi
3240 \fi\ignorespaces}
%
```

```

\vbfnoteX41 \newcommand{\vbfnoteX}[3]{%
3242   \@namedef{@thefnmark#1}{#3}%
3243   \@nameuse{regvfootnote#1}{#1}{#2}}
3244
3245 %
```

```

\vnumfootnoteX46 \newcommand{\vnumfootnoteX}[2]{%
3247   \ifnumberedpar@
3248     \edtext{}{\normalbfnoteX{#1}{#2}}%
3249   \else
3250     \@nameuse{regvfootnote#1}{#1}{#2}%
3251   \fi}
3252
3253 %
```

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

```

3254 \newcommand*{\arrangementX@normal}[1]{%
3255   \csgdef{series@displayX#1}{normal}
3256   \expandafter\let\csname footstart#1\endcsname=\normalfootstartX
3257   \expandafter\newcount\csname prevpage#1@num\endcsname
3258   \@namedef{@footnotemark#1}{\normal@footnotemarkX{#1}}
3259   \@namedef{bodyfootmark#1}{\normalbodyfootmarkX{#1}}
3260   \expandafter\let\csname regvfootnote#1\endcsname=\normalvfootnoteX
3261   \expandafter\let\csname vfootnote#1\endcsname=\vnumfootnoteX
3262   \expandafter\let\csname footfmt#1\endcsname=\normalfootfmtX
3263   \@namedef{footfootmark#1}{\normalfootfootmarkX{#1}}
3264   \expandafter\let\csname footgroup#1\endcsname=\normalfootgroupX
3265   \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3266   \count\csname footins#1\endcsname=1000
3267   \csxdef{default@footins#1}{1000}%Use to have note only for one side
3268   \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3269   \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3270   \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
3271   %

```

Additions for minipages.

```

3272   \ifnoledgroup@{\else%
3273     \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3274     \expandafter\let\csname mpfootgroup#1\endcsname=\mpnormalfootgroupX
3275     \count\csname mpfootins#1\endcsname=1000
3276     \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
3277     \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3278     \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
3279   \fi
3280 }
3281
3282 %

```

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@twocol33 \newcommand*{\arrangementX@twocol}[1]{%
3284   \csgdef{series@displayX#1}{twocol}
3285   \expandafter\let\csname regvfootnote#1\endcsname=\twocolvfootnoteX
3286   \expandafter\let\csname footfmt#1\endcsname=\twocolfootfmtX
3287   \expandafter\let\csname footgroup#1\endcsname=\twocolfootgroupX
3288   \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3289   \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3290   \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3291   \twocolfootsetupX{#1}

```

```

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

```

```

\twocolfootsetupX \twocolfootsetupX{<series>}
\mptwocolfootsetupX
3302 \newcommand*{\twocolfootsetupX}[1]{%
3303 \count\csname footins#1\endcsname 500
3304 \csxdef{default@footins#1}{500}%Use this to confine the notes to one
side only
3305 \multiply\dimen\csname footins#1\endcsname by \tw@}
3306 \newcommand*{\mptwocolfootsetupX}[1]{%
3307 \count\csname mpfootins#1\endcsname 500
3308 \multiply\dimen\csname mpfootins#1\endcsname by \tw@}
3309
3310 %

```

```

\twocolvfootnoteX \twocolvfootnoteX{<series>}
3311 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolvfootnoteX}[2]{%
3312 \insert\csname footins#1\endcsname\bgroup
3313 \csuse{notefontsizeX@#1}
3314 \footssplit skips
3315 \spaceskip=\z@skip \xspaceskip=\z@skip
3316 \@nameuse{footfmt#1}{#1}{#2}\egroup}
3317
3318 %

```

```

\twocolfootfmtX \twocolfootfmtX{<series>}
3319 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolfootfmtX}[2]{%
3320 \protected@edef\@currentlabel{%
3321 \@nameuse{@thefnmark#1}%
3322 }%
3323 \normal@pars
3324 \hangindent=\csuse{hangindentX@#1}%
3325 \everypar{\hangindent=\csuse{hangindentX@#1}}%
3326 \hsize \csuse{hsizetwocolX@#1}
3327 \nottoggle{parindentX@#1}{\parindent=\z@}{\}
3328 \tolerance=5000\relax
3329 \leavevmode
3330 \@tempdima=\parindent%
3331 \csuse{colalignX@#1}%

```



```

3332 \parindent=\@tempdima%
3333 {\csuse{notenumfontX@#1}\@nameuse{footfootmark#1}\strut%
3334 #2\strut\par}\allowbreak}
3335
3336 %

```

`\twocolfootgroupX` `\twocolfootgroupX{<series>}`
`\mptwocolfootgroupX`

```

3337 \newcommand*{\twocolfootgroupX}[1]{\csuse{notefontsizeX@#1}
3338 \splittopskip=\ht\strutbox
3339 \expandafter
3340 \rigidbalance\csname footins#1\endcsname \tw@ \splittopskip}}
3341 \newcommand*{\mptwocolfootgroupX}[1]{\csuse{notefontsizeX@#1}
3342 \vskip\skip\@nameuse{mpfootins#1}
3343 \ifl@dpairing\ifparledgroup%
3344 \leavevmode\marks\parledgroup@{begin}%
3345 \marks\parledgroup@series{#1}%
3346 \marks\parledgroup@type{footnoteX}%
3347 \fi\fi\normalcolor
3348 \ifparledgroup%
3349 \ifl@dpairing%
3350 \else%
3351 \setnoteswidthliketwocolumnsX@{#1}%
3352 \setnotesXpositionliketwocolumns@{#1}%
3353 \print@footnoteXrule{#1}%
3354 \fi%
3355 \else%
3356 \setnoteswidthliketwocolumnsX@{#1}%
3357 \setnotesXpositionliketwocolumns@{#1}%
3358 \print@footnoteXrule{#1}%
3359 \fi%
3360 \splittopskip=\ht\strutbox
3361 \expandafter
3362 \rigidbalance\csname mpfootins#1\endcsname \tw@ \splittopskip}}
3363
3364 %

```

XIII.4.4 Three columns footnotes

The following macros set footnotes in three columns. It is assumed that the length of each footnote is less than the column width.

```

\arrangementX@threecol  \newcommand*{\arrangementX@threecol}[1]{%
3366 \csgdef{series@displayX#1}{threecol}
3367 \expandafter\let\csname regvfootnote#1\endcsname=\threecolvfootnoteX
3368 \expandafter\let\csname footfmt#1\endcsname=\threecolfootfmtX
3369 \expandafter\let\csname footgroup#1\endcsname=\threecolfootgroupX
3370 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3371 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%

```

```

3372 \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3373 \threecolfootsetupX{#1}
3374 \ifnoledgroup@{\else%
3375   \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3376   \expandafter\let\csname mpfootgroup#1\endcsname=\mpthreecolfootgroupX
3377   \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3378   \advance\skip\csname mpfootins#1\endcsname by \csuse{afterruleX@#1}
3379   \mpthreecolfootsetupX{#1}
3380 \fi%
3381 }
3382
3383 %

```

```

\threecolfootsetupX \threecolfootsetupX{<series>}
\mpthreecolfootsetupX
3384 \newcommand*{\threecolfootsetupX}[1]{%
3385   \count\csname footins#1\endcsname 333
3386   \csxdef{default@footins#1}{333}%Use this to confine the notes to one
side only
3387   \multiply\dimen\csname footins#1\endcsname by \thr@@}
3388 \newcommand*{\mpthreecolfootsetupX}[1]{%
3389   \count\csname mpfootins#1\endcsname 333
3390   \multiply\dimen\csname mpfootins#1\endcsname by \thr@@}
3391
3392 %

```

```

\threecolvfootnoteX \threecolvfootnoteX{<series>}{<text>}
3393 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolvfootnoteX}[2]{
%
3394   \insert\csname footins#1\endcsname\bgroup
3395     \csuse{notefontsizeX@#1}
3396     \footsplitskips
3397     \@nameuse{footfmt#1}{#1}{#2}\egroup}
3398
3399 %

```

```

\threecolfootfmtX \threecolfootfmtX{<series>}
3400 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmtX}[2]{%
3401   \protected@edef\@currentlabel{%
3402     \@nameuse{@thefnmark#1}%
3403   }%
3404   \hangindent=\csuse{hangindentX@#1}%
3405   \everypar{\hangindent=\csuse{hangindentX@#1}}%
3406   \normal@pars
3407   \hsize \csuse{hsizethreecolX@#1}
3408   \nottoggle{parindentX@#1}{\parindent=\z@}{ } %
3409   \tolerance=5000\relax
3410   \leavevmode

```

```

3411 \@tempdima=\parindent%
3412 \csuse{colalignX@#1}%
3413 \parindent=\@tempdima%
3414 {\csuse{notenumfontX@#1}\@nameuse{footfootmark#1}\strut%
3415 #2\strut\par}\allowbreak}
3416
3417 %

\threecolfootgroupX \threecolfootgroupX{<series>}
\mpthreecolfootgroupX
3418 \newcommand*{\threecolfootgroupX}[1]{\csuse{notefontsizeX@#1}
3419 \splittopskip=\ht\strutbox
3420 \expandafter
3421 \rigidbalance\csname footins#1\endcsname \thr@@ \splittopskip}}
3422 \newcommand*{\mpthreecolfootgroupX}[1]{\%
3423 \vskip\skip\@nameuse{mpfootins#1}
3424 \ifl@dpairing\ifparledgroup
3425 \leavevmode\marks\parledgroup@{begin}%
3426 \marks\parledgroup@series{#1}%
3427 \marks\parledgroup@type{footnoteX}%
3428 \fi\fi\normalcolor
3429 \ifparledgroup%
3430 \ifl@dpairing%
3431 \else%
3432 \setnoteswidthliketwocolumnsX@{#1}%
3433 \setnotesXpositionliketwocolumns@{#1}%
3434 \print@footnoteXrule{#1}%
3435 \fi%
3436 \else%
3437 \setnoteswidthliketwocolumnsX@{#1}%
3438 \setnotesXpositionliketwocolumns@{#1}%
3439 \print@footnoteXrule{#1}%
3440 \fi%
3441 \splittopskip=\ht\strutbox
3442 \expandafter
3443 \rigidbalance\csname mpfootins#1\endcsname \thr@@ \splittopskip}}
3444
3445 %

```

XIII.4.5 Paragraphed footnotes

The following macros set footnotes as one paragraph.

```

\arrangementX@threecol \footparagraphX{<series>}

3446 \newcommand*{\arrangementX@paragraph}[1]{%
3447 \csgdef{series@displayX#1}{paragraph}%
3448 \expandafter\newcount\csname #1prevpage@num\endcsname
3449 \expandafter\let\csname footstart#1\endcsname=\parafootstartX
3450 \expandafter\let\csname regvfootnote#1\endcsname=\para@vfootnoteX

```

```

3451 \expandafter\let\csname footfmt#1\endcsname=\parafootfmtX
3452 \expandafter\let\csname footgroup#1\endcsname=\para@footgroupX
3453 \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3454 \count\csname footins#1\endcsname=1000
3455 \csxdef{default@footins#1}{1000}%Use this to confine the notes to one
side only
3456 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3457 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3458 \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
3459 \para@footsetupX{#1}
3460 \ifnoledgroup@else
3461 \expandafter\let\csname mpvfootnote#1\endcsname=\mppara@vfootnoteX
3462 \expandafter\let\csname mpfootgroup#1\endcsname=\mppara@footgroupX
3463 \count\csname mpfootins#1\endcsname=1000
3464 \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
3465 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3466 \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
3467 \fi
3468 }
3469
3470 %

```

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

```

3471 \newcommand*{\para@footsetupX}[1]{\csuse{notefontsizeX@#1}
3472 \setnoteswidthliketwocolumnsX@{#1}%
3473 \dimen0=\baselineskip
3474 \multiply\dimen0 by 1024
3475 \divide\dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\relax
%
3476 \expandafter
3477 \xdef\csname footfudgefactor#1\endcsname{%
3478 \expandafter\strip@pt\dimen0 }}
3479
3480 %

```

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

```

3481 \newcommand*{\parafootstartX}[1]{%
3482 \ifdimequal{Opt}{\prenotesX@}{}%
3483 {%
3484 \iftoggle{prenotesX@}{%
3485 \togglefalse{prenotesX@}%
3486 \skip\csname footins#1\endcsname=%
3487 \dimexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
3488 }%
3489 }%
3490 }%
3491 \leftskip=\z@
3492 \rightskip=\z@

```

```

3493 \nottoggle{parindentX@#1}{\parindent=\z@}{}%
3494 \vskip\skip\@nameuse{footins#1}%
3495 \setnoteswidthliketwocolumnsX@{#1}%
3496 \setnotesXpositionliketwocolumns@{#1}%
3497 \print@footnoteXrule{#1}%
3498 }
3499
3500 %

```

`\para@vfootnoteX` `\para@vfootnoteX{<series>}{<text>}`
`\mppara@vfootnoteX`

```

3501 \newcommand*{\para@vfootnoteX}[2]{%
3502   \insert\csname footins#1\endcsname
3503   \bgroup
3504     \csuse{notefontsizeX@#1}
3505     \footsplitskips
3506     \setbox0=\vbox{\hsize=\maxdimen
3507       \noindent\csuse{bhooknoteX@#1}%
3508       \@nameuse{footfmt#1}{#1}{#2}}%
3509     \setbox0=\hbox{\unvxhX{0}{#1}}%
3510     \dp0=\z@
3511     \ht0=\csname footfudgefactor#1\endcsname\wd0
3512     \box0
3513     \penalty0
3514   \egroup}
3515 \newcommand*{\mppara@vfootnoteX}[2]{%
3516   \global\setbox\@nameuse{mpfootins#1}\vbox{%
3517     \unvbox\@nameuse{mpfootins#1}
3518     \csuse{notefontsizeX@#1}
3519     \footsplitskips
3520     \setbox0=\vbox{\hsize=\maxdimen
3521       \noindent\color@begingroup%
3522       \csuse{bhooknoteX@#1}%
3523       \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}%
3524     \setbox0=\hbox{\unvxhX{0}{#1}}%
3525     \dp0=\z@
3526     \ht0=\csname footfudgefactor#1\endcsname\wd0
3527     \box0
3528     \penalty0}}
3529
3530 %

```

`\unvxhX`₃₁ `\newcommand*{\unvxhX}[2]{% 2th is optional for retro-compatibility`

```

3532   \setbox0=\vbox{\unvbox#1%
3533     \global\setbox1=\lastbox}%
3534   \unhbox1
3535   \unskip           % remove \rightskip,
3536   \unskip           % remove \parfillskip,
3537   \unpenalty        % remove \penalty of 10000,

```

```

3538 \hskip\csuse{afternoteX@#2}} % but add the glue to go between the notes
3539
3540 %

```

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

```

3541 \newcommand*{\parafootfmtX}[2]{%
3542   \protected@edef\@currentlabel{%
3543     \@nameuse{@thefnmark#1}%
3544   }%
3545   \insertparafootsepX{#1}%
3546   \ledsetnormalparstuff@common%
3547   {\csuse{notenunfontX@#1}%
3548    \csuse{notenunfontX@#1}%
3549    \@nameuse{footfootmark#1}%
3550    \strut%
3551    #2\penalty-10}}
3552
3553 %

```

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

```

3554 \newcommand*{\para@footgroupX}[1]{%
3555   \unvbox\csname footins#1\endcsname
3556   \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3557   \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3558   \makeboxofhboxes
3559   \setbox0=\hbox{\unhbox0 \removehboxes}%
3560   \csuse{notefontsizeX@#1}
3561   \unhbox0\par}
3562 \newcommand*{\mppara@footgroupX}[1]{%
3563   \setnoteswidthliketwocolumnsX@{#1}%
3564   \vskip\skip\@nameuse{mpfootins#1}
3565   \ifl@dpairing\ifparledgroup
3566     \leavevmode%
3567     \leavevmode\marks\parledgroup@{begin}%
3568     \marks\parledgroup@series{#1}%
3569     \marks\parledgroup@type{footnoteX}%
3570   \fi\fi\normalcolor
3571   \ifparledgroup%
3572     \ifl@dpairing%
3573     \else%
3574       \setnoteswidthliketwocolumnsX@{#1}%
3575       \setnotesXpositionliketwocolumns@{#1}%
3576       \print@footnoteXrule{#1}%
3577     \fi%
3578   \else%
3579     \setnoteswidthliketwocolumnsX@{#1}%
3580     \setnotesXpositionliketwocolumns@{#1}%
3581     \print@footnoteXrule{#1}%

```

```

3582 \fi%
3583 \unvbox\csname mpfootins#1\endcsname
3584 \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3585 \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3586 \makehboxofhboxes
3587 \setbox0=\hbox{\unhbox0 \removehboxes}%
3588 \csuse{notefontsizeX@#1}
3589 \unhbox0\par}}
3590
3591 %

```

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

```

\prevpage@num%2 \newcommand{\insertparafootsepX}[1]{%
\Xinsertparafootsep%3 \ifnumequal{\csuse{prevpage#1@num}}{\page@num}%
3594 {\csuse{parafootsepX@#1}}%
3595 {}%
3596 }
3597 %

```

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

`\par` should always be redefined to `\endgraf` within the format macro (this is what `\normal@pars` does), to override tricky material in the main text to get the lines numbered automatically (as set up by `\autopar`, for example).

In the case of footnote arranged in a “normal” way, we also must set some setting for paragraph indent and text direction when using `LuaLaTeX`.

That why we have defined `\ledsetnormalparstuff@common` in order to make this setting for both familiar and critical notes. This command is called by command to make specific setting to critical or familiar footnote.

```

\ledsetnormalparstuff@common%8 \newcommand*{\ledsetnormalparstuff@common}{%
\Xledsetnormalparstuff%9 \ifluatex%
\ledsetnormalparstuffX%0 \textdir\footnote@luatextextdir%
3601 \pardir\footnote@luatexpardir%
3602 \fi%
3603 \csuse{\csuse{footnote@dir}}%
3604 \normal@pars%
3605 \parfillskip \z@ \@plus 1fil}%
3606
3607 \newcommand*{\Xledsetnormalparstuff}[1]{%
3608 \ledsetnormalparstuff@common%
3609 \nottoggle{Xparindent@#1}{\parindent=\z@}{\hspace{\parindent}}%

```

```

3610 }%
3611
3612 \newcommand*{\ledsetnormalparstuffX}[1]{%
3613   \ledsetnormalparstuff@common%
3614   \nottoggle{parindentX@#1}{\parindent=\z@}{\hspace{\parindent}}%
3615 }%
3616 %

```

XV Footnotes' width for two columns

We define here some commands which make sense only with `reledpar`, but must be called when defining notes parameters. These commands change the width of block notes to allow them to have the same size than two parallel columns.

`\old@hsize` These two commands are called at the beginning of critical or familiar notes groups. They set, if the option is enabled, the `\hsize`. They are also called at the on the setup for paragraphed notes.

`\setXnoteswidthliketwocolumns@`
`\setnoteswidthliketwocolumnsX@`

```

3617
3618 \newdimen\old@hsize%
3619 \AtBeginDocument{\old@hsize=\hsize}%
3620
3621 \newcommand{\setXnoteswidthliketwocolumns@}[1]{%
3622   \global\let\hsize@fornote=\hsize%
3623   \global\old@hsize=\hsize%
3624   \iftoggle{Xnoteswidthliketwocolumns@#1}%
3625     {%
3626       \csuse{setwidthliketwocolumns@\columns@position}%
3627       \global\let\hsize@fornote=\hsize%
3628     }%
3629   {%
3630     \let\hsize=\hsize@fornote%
3631     \let\columnwidth=\hsize@fornote%
3632   }%
3633
3634 \newcommand{\setnoteswidthliketwocolumnsX@}[1]{%
3635   \global\let\hsize@fornote=\hsize%
3636   \global\old@hsize=\hsize%
3637   \iftoggle{noteswidthliketwocolumnsX@#1}%
3638     {%
3639       \csuse{setwidthliketwocolumns@\columns@position}%
3640       \global\let\hsize@fornote=\hsize%
3641     }%
3642   {%
3643     \let\hsize=\hsize@fornote%
3644     \let\columnwidth=\hsize@fornote%
3645   }%
3646
3647 %

```


`\setnotespositionliketwocolumns@` These two commands set the position of the critical / familiar footnotes, depending on the hooks `Xnoteswidthliketwocolumns` and `noteswidthliketwocolumnsX`. They call commands which are defined only in `reledpar`, because this feature has no sens without `reledpar`.

```

3648 \newcommand{\setXnotespositionliketwocolumns@}[1]{%
3649   \iftoggle{Xnoteswidthliketwocolumns@#1}{%
3650     \csuse{setnotespositionliketwocolumns@\columns@position}%
3651   }{}%
3652 }%
3653
3654 \newcommand{\setnotesXpositionliketwocolumns@}[1]{%
3655   \iftoggle{noteswidthliketwocolumnsX@#1}{%
3656     \csuse{setnotespositionliketwocolumns@\columns@position}%
3657   }{}%
3658 }%
3659
3660 %

```

XVI Footnotes' order

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

```

\@fnpos
\@mpfnpos
3661 \def\@fnpos{familiar-critical}
3662 \def\@mpfnpos{critical-familiar}
3663 \newcommand{\fnpos}[1]{\xdef\@fnpos{#1}}
3664 \newcommand{\mpfnpos}[1]{\xdef\@mpfnpos{#1}}
3665 %

```

XVII Footnotes' rule

Because the footnotes' rules can be shifted to the right when footnotes are set like two columns, we do not print them directly, but we put them in a `\vbox`.

```

\print@Xfootnoterule 3666 \newcommand{\print@Xfootnoterule}[1]{%
\print@footnoteXrule 3667   \vskip-\csuse{Xafterterrule@#1}%Because count in \dimen\csuse{#1footins}
3668   \nointerlineskip%
3669   \moveleft-\leftskip\vbox{\csuse{#1footnoterule}}%
3670   \nointerlineskip%
3671   \vskip\csuse{Xafterterrule@#1}%
3672 }%
3673
3674 \newcommand{\print@footnoteXrule}[1]{%
3675   \vskip-\csuse{afterterruleX@#1}%Because count in \dimen\csuse{footins#1}
3676   \nointerlineskip%
3677   \moveleft-\leftskip\vbox{\csuse{footnoterule#1}}%

```

```

3678 \nointerlineskip%
3679 \vskip\csuse{afterruleX@#1}%
3680 }%
3681
3682 %

```

XVIII Specific skip for first series of footnotes

XVIII.0.1 Overview

`\Xbeforenotes` inserts a specific skip for the first series of notes in a page. As we can't know in advance which series will be the first, we call `\prepare@preXnotes` before inserting any critical notes, in order to prevent page number overlapping.

1. If it is the first note of the current page, it changes the footnote skip for the series to the value specified to `\Xbeforenotes`. It also keeps the series of the note as the first one of the current page.
2. If it is not the first note of the current page:
 - If the current series is printed after the series kept as the first of the current page, then nothing happens.
 - If the current series is printed before the series kept as the first of the current page, then it changes the footnote skip of the current series to the value normally used by the series which was marked as the first of the page. It also keeps the current series as the new first one of the current page.

For example, suppose the series order is A,B. We call first a `\Bfootnote` and a `\Afootnote`. The only skips used are, finally, the skip specific to the first series of the page, and the skip for the B series. If we have not called `\Afootnote`, the only skip used is the skip specific to the first series of the page.

That is perfect.

The series skip and the first series of the current page are reset before the footnotes are printed. Then, the footstart macros manage the problem of the first series of the page.

After the rule, the space which is defined by `\Xafterrule` does not depend on whether the series is the first one of the page or not. So we use its normal value for each series.

And now, implementation !

XVIII.0.2 User level command

`\preXnotes@` If user redefines `\preXnotes@`, via `\preXnotes` to a value greater than 0 pt, this skip will be added before first series notes instead of the notes skip.

```

3683 \newtoggle{preXnotes@}
3684 \toggletrue{preXnotes@}
3685 \newcommand{\preXnotes@}{0pt}

```

```

3686 \newcommand*{\preXnotes}[1]{\renewcommand{\preXnotes@}{#1}}
3687 %

```

The same, but for familiar footnotes.

```

\preXnotes 88 \newtoggle{prenotesX@}
\preXnotes 89 \toggletrue{prenotesX@}
3690 \newcommand{\preXnotesX@}{Opt}
3691 \newcommand*{\preXnotesX}[1]{\renewcommand{\preXnotesX@}{#1}}
3692 %

```

XVIII.0.3 Internal commands

```

firstXseries@ 93 \gdef\firstXseries@{}
prepare@preXnotes 94 \newcommand{\prepare@preXnotes}[1]{%
3695 \ifdimequal{Opt}{\preXnotes@}%
3696 }%
3697 {%
3698 \IfStrEq{\firstXseries@}{}%
3699 \global\skip\csuse{#1footins}=\preXnotes@%
3700 \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
3701 \gdef\firstXseries@{#1}%
3702 }%
3703 {%
3704 \ifseriesbefore{#1}{\firstXseries@}%
3705 {%
3706 \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@\firstXseries@}%
3707 \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
3708 \gdef\firstXseries@{#1}%
3709 }%
3710 }%
3711 }%
3712 }%
3713 }
3714 %

```

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

```

firstseriesX@ 15 \gdef\firstseriesX@{}
prepare@prenotesX 16 \newcommand{\prepare@prenotesX}[1]{%
3717 \ifdimequal{Opt}{\prenotesX@}%
3718 }%
3719 {%
3720 \IfStrEq{\firstseriesX@}{}%
3721 \global\skip\csuse{footins#1}=\prenotesX@%

```

```

3722 \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
3723 \gdef\firstseriesX@{#1}%
3724 }%
3725 {%
3726 \ifseriesbefore{#1}{\firstseriesX@}%
3727 {%
3728 \global\skip\csuse{footins#1}=\csuse{beforenotesX@\firstseriesX@}%
3729 \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
3730 \gdef\firstXseries@{#1}%
3731 }%
3732 {}%
3733 }%
3734 }%
3735 }
3736 %

```

XIX Endnotes

First, check the noend option.

```

3737 \ifbool{noend@}{\%Used instead of \ifnoend@ to prevent expansion problem
3738 %

```

\l@dend@open **\l@dend@close** **\l@dend@stuff** **\l@dend@open** and **\l@dend@close** are the macros that are used to open and close the endnote file. Note that all our writing to this file is `\immediate`: all page and line numbers for the endnotes are generated by the same mechanism we use for the footnotes, so that there is no need to defer any writing to catch information from the output routine. The argument of these two command is the series letter.

```

3739 \newcommand{\l@dend@open}[1]{%
3740 \global\booltrue{l@dend@#1}%
3741 \expandafter\immediate%
3742 \expandafter\openout%
3743 \csname l@d@#1end\endcsname%
3744 =\jobname.#1end\relax%
3745 }%
3746 \newcommand{\l@dend@close}[1]{%
3747 \global\boolfalse{l@dend@#1}%
3748 \expandafter\immediate%
3749 \expandafter\closeout\csname l@d@#1end\endcsname%
3750 }%
3751
3752 %

```

\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.

```

3753 \newcommand{\l@dend@stuff}{%
3754   \def\do##1{%
3755     \ifbool{l@dend@##1}{}%
3756     {\l@dend@open{##1}}%
3757     \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname{\
string\l@d@section{\the\section@num}}}%
3758   }%
3759   \dolistloop{\@series}%
3760 }%
3761
3762 %

```

`\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`.

```

3763 \global\newbool{parapparatus@}{\long}\def\endprint#1#2#3#4#5{%
3764   \hangindent=\csuse{Xendhangindent@#4}%
3765   \ifXendinsertsep@%
3766     \hskip\csuse{Xendafternote@#4}%
3767     \csuse{Xendsep@#4}%
3768   \else%
3769     \iftoggle{Xendparagraph@#4}%
3770       {\global\Xendinsertsep@true}%
3771     }%
3772   \fi%
3773   \xdef\@currentseries{#4}%
3774   \def\do##1{%
3775     \toggletrue{##1@}%
3776   }%
3777   \notblank{#5}{\docsvlist{#5}}{%}%
3778   \csuse{Xendbhooknote@#4}%
3779   \csuse{Xendnotefontsize@#4}%
3780   \printlineendnote{#1}{#4}%
3781   \nottoggle{Xendlemmadisablefontselection@#4}%
3782     {\select@lemmafont#1|#2}%
3783     {#2}%
3784   \ifboolexpr{%

```

```

3785     togl {nosep@}%
3786     or test{\ifcsemtyp{Xendlemmaseparator@#4}}%
3787     }%
3788     {\hskip\csuse{Xendinplaceoflemmaseparator@#4}}%
3789     {\nobreak%
3790     \hskip\csuse{Xendbeforelemmaseparator@#4}%
3791     \csuse{Xendlemmaseparator@#4}%
3792     \hskip\csuse{Xendafterlemmaseparator@#4}%
3793     }%
3794     #3%
3795     \nottoggle{Xendparagraph@#4}{\par}{}%
3796     \def\do##1{%
3797     \togglefalse{##1@}%
3798     }%
3799     \notblank{#5}{\docsvlist{#5}}{}%
3800 }}%
3801
3802 \let\l@d@section=\@gobble
3803
3804 %

```

\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.

```

3805 \newcommand{\printlineendnote}[2]{%
3806     \l@dp@rsefootspec#1|%
3807     \iftoggle{Xendnumberonlyfirstintwolines@#2}{%
3808         \edef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
3809         \l@dparsedstartsub - \l@dparsedendpage - \l@dparsedendline - \
3810         \l@dparsedendsub}%
3811         }%
3812         {%
3813         \edef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
3814         \l@dparsedstartsub}%
3815         }%
3816         \ifboolexpr{%
3817         togl {nonum@}%
3818         or togl {Xendnonumber@#2}%
3819         }%
3820         {\hspace{\csuse{Xendinplaceofnumber@#2}}}%
3821         {%
3822         \iftoggle{Xendnumberonlyfirstinline@#2}%
3823         {\ifcsemtyp{Xendnumberonlyfirstinline@#2}%
3824         {\ifcsequal{Xendnumberonlyfirstinline@#2}{lineinfo@}%
3825         }%
3826         \csuse{Xendhookinplaceofnumber@#2}%
3827         \ifcsemtyp{Xendsymlineum@#2}%
3828         {\hspace{\csuse{Xendinplaceofnumber@#2}}}%
3829         {\printsymlineendnotearea{#2}}}%
3830         }%
3831     }%
3832 }%

```

```

3827         \csuse{Xendahookinplaceofnumber@2}%
3828     }%
3829     {\printlineendnotearea{#1}{#2}}}%
3830     {\printlineendnotearea{#1}{#2}}%
3831 }%
3832 {\printlineendnotearea{#1}{#2}}%We keep every time line
3833 \csxdef{prevendline#2}{\lineinfo@}%
3834 }%
3835 }%
3836 %

```

```

\printsymlineendnotearea \newcommand{\printsymlineendnotearea}[1]{%
3838     \hspace{\csuse{Xendbeforenum@#1}}%
3839     \csuse{Xendnotenumfont@#1}%
3840     \ifdimequal{\csuse{Xendboxnum@#1}}{\z@}%
3841         {\csuse{Xendsymnum@#1}}%
3842         {\hbox to \csuse{Xendboxnum@#1}%
3843             {\csuse{Xendsymnum@#1}\hfill}%
3844         }%
3845     \hspace{\csuse{Xendaftersymnum@#1}}%
3846 }%
3847 %

```

\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.)

```

3848 \newcommand{\printlineendnotearea}[2]{%
3849     \csuse{Xendbhooklinenumber@#2}%
3850     \hspace{\csuse{Xendbeforenumber@#2}}%
3851     \bgroup%
3852     \csuse{Xendnotenumfont@#2}%
3853     \ifdimequal{\csuse{Xendboxlinenum@#2}}{0pt}%
3854         {\printendlines#1}%
3855         {\leavevmode%
3856             \hbox to \csuse{Xendboxlinenum@#2}%
3857             {%
3858                 \IfSubStr{RC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}{}%
3859                 \printendlines#1}%
3860             \IfSubStr{LC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}{}%
3861             }%
3862     \egroup%
3863     \hspace{\csuse{Xendafternumber@#2}}%
3864     \csuse{Xendahooklinenumber@#2}%
3865 }%
3866 %

```

`\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.

```
3867 \newcommand*{\setprintendlines}[6]{%
3868   \l@dpnumfalse \l@ddashfalse
3869   \ifnum#4=#1 \else
3870     \l@dpnumtrue
3871     \l@ddashtrue
3872   \fi
3873   %
```

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.

```
3874   \ifl@dpnum \l@d@elintrue \else \l@d@elinfalse \fi
3875   \ifnum#2=#5 \else
3876     \l@d@elintrue
3877     \l@d@dashtrue
3878   \fi
3879   %
```

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

```
3880   \l@d@ssubfalse
3881   \ifnum#3=0 \else
3882     \l@d@ssubtrue
3883   \fi
3884   %
```

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.

```
3885   \l@d@eslfalse
3886   \ifnum#6=0 \else
3887     \ifnum#6=#3
3888       \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
3889     \else
3890       \l@d@esltrue
3891       \l@d@dashtrue
3892     \fi
3893   \fi%
3894   %
3895   \ifl@d@dash%
```



```

3896 \ifboolexpr{togl{fulllines@} or test{\ifcempty{Xendtwolines@}\
@currentseries}}}%
3897 {}%
3898 {%
3899 \setistwofollowinglines{#1}{#2}{#4}{#5}%
3900 \ifboolexpr{%
3901 (%
3902     togl {Xendtwolinesbutnotmore@\@currentseries}%
3903     and not%
3904     (%
3905         bool {istwofollowinglines@}%
3906     )%
3907 )%
3908 or%
3909 (%
3910     (not test{\ifnumequal{#1}{#4}})%
3911     and togl{Xendtwolinesonlyinsamepage@\@currentseries}%
3912 )%
3913 }%
3914 {}%
3915 {%
3916 \l@d@dashfalse%
3917 \l@d@Xtwolinestrue%
3918 \l@d@elinfalse%
3919 \l@d@eslfalse%
3920 \ifcempty{Xendmorethantwolines@\@currentseries}%
3921 {%
3922     {\ifistwofollowinglines@\else%
3923         \l@d@Xmorethantwolinestrue%
3924     }%
3925 }%
3926 }%
3927 }%
3928 \fi%
3929 %

```

End of \setprintendlines.

```

3930 }%
3931 %

```

\printendlines Now we are ready to print it all.

```

3932 \def\printendlines#1|#2|#3|#4|#5|#6|#7|{\begingroup
3933 \setprintendlines{#1}{#2}{#3}{#4}{#5}{#6}%
3934 %

```

The only subtlety left here is when to print a period between numbers. But the only instance in which this is tricky is for the ending sub-line number: it could be coming after the starting sub-line number (in which case we want only the dash) or after an ending line number (in which case we need to insert a period).

So, first, print the start lines.

```

3935 \ifdimequal{\csuse{Xendboxstartlinenum@\@currentseries}}{0pt}%
3936   {\bgroup}%
3937   {\leavevmode\hbox to \csuse{Xendboxstartlinenum@\@currentseries}\bgroup
\hfill}%
3938   \printnpnum{#1}%
3939   \linenumrep{#2}%
3940   \ifl@d@ssub \fullstop \sublinenumrep{#3}\fi
3941   \egroup%
3942 %

```

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

```

3943 \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
3944   {\bgroup}%
3945   {\hbox to \csuse{Xendboxendlinenum@\@currentseries}\bgroup}%
3946   \ifl@d@Xtwolines%
3947     \ifl@d@Xmorethantwolines%
3948       \csuse{Xendmorethantwolines@\@currentseries}%
3949     \else%
3950       \csuse{Xendtwolines@\@currentseries}%
3951     \fi%
3952   \else%
3953     \ifl@d@dash \endashchar\fi%
3954     \ifl@d@pnum \printnpnum{#4}\fi%
3955     \ifl@d@elin \linenumrep{#5}\fi%
3956     \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumrep{#6}\fi%
3957   \fi%
3958   \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
3959   {}%
3960   {\hfill}%Prevent underfull hbox
3961   \egroup%
3962   \endgroup%
3963 }%
3964 %
3965 %

```

\printnpnum A macro to print a page number in an endnote.

```

3966 \newcommand*\printnpnum[1]{p.#1} }
3967
3968 %

```

\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.

```

3969 \newif\ifXendinsertsep@%
3970 \newcommand*\doendnotes[1]{%
3971   \l@dend@close{#1}%

```

```

3972 \begingroup
3973   \makeatletter
3974   \expandafter\let\csname #1end\endcsname=\endprint
3975   \input\jobname.#1end%
3976   \global\Xendinsertsep@false%
3977 \endgroup}
3978 %

```

\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.

```

3979 \newcommand*{\doendnotesbysection}[1]{%
3980   \l@dend@close{#1}%
3981   \global\expandafter\advance\csname #1end@bysection\endcsname by 1%
3982   \begingroup%
3983     \makeatletter%
3984     \def\l@d@section##1{%
3985       \ifnumequal{##1}{\csname #1end@bysection\endcsname}%
3986       {\cslet{#1end}{\endprint}}%
3987       {\cslet{#1end}{\@gobblefive}}%
3988     }%
3989     \input\jobname.#1end%
3990     \global\Xendinsertsep@false%
3991   \endgroup%
3992 }%
3993 %

```

End of section for end notes

```

3994 }%
3995 %

```

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@.

```

3996 \newcommand{\newseries}[1]{%
3997   \def\do##1{\newseries@{##1}}%
3998   \docsvlist{#1}
3999 }
4000 %

```

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

```
4001 \newcommand{\@series}{}
4002 %
```

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

```
\newseries@03 \newcommand{\newseries@}[1]{
4004 %
```

XX.1 Test if series is still existing

```
4005 \xifinlist{#1}{\@series}{\led@warn@SeriesStillExist{#1}}%
4006 {%
4007 %
```

XX.2 Init specific to reledpar

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

```
4008 \ifdefined\newseries@par%
4009 \newseries@par{#1}%
4010 \fi%
4011 %
```

XX.3 For critical footnotes

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

```
4012 \unless\ifnocritical@
4013 %
```

XX.3.1 Options

```
4014 \newtoggle{Xparindent@#1}
4015 \newtoggle{Xlemmadisablefontselection@#1}
4016 \csgdef{Xhangindent@#1}{Opt}%
4017 \csgdef{Xragged@#1}{}%
4018 \csgdef{Xhsizetwocol@#1}{0.45 \hsize}%
4019 \csgdef{Xhsizethreecol@#1}{.3 \hsize}%
4020 \csgdef{Xcolalign@#1}{\raggedright}%
4021 \csgdef{Xnotenumfont@#1}{\normalfont}%
4022 \csgdef{Xnotefontsize@#1}{\footnotesize}%
4023 \csgdef{Xbhooknote@#1}{}%
4024
4025 \csgdef{Xboxlinenum@#1}{Opt}%
4026 \csgdef{Xboxlinenumalign@#1}{L}%

```

```

4027 \csgdef{Xboxstartlinenum@#1}{Opt}%
4028 \csgdef{Xboxendlinenum@#1}{Opt}%
4029
4030
4031 \csgdef{Xboxsymlinenum@#1}{Opt}%
4032 \newtoggle{Xnumberonlyfirstinline@#1}%
4033 \newtoggle{Xnumberonlyfirstintwolines@#1}%
4034 \csgdef{Xtwolines@#1}{}%
4035 \csgdef{Xmorethantwolines@#1}{}%
4036 \newtoggle{Xtwolinesbutnotmore@#1}%
4037 \newtoggle{Xtwolinesonlyinsamepage@#1}%
4038 \newtoggle{Xonlypstart@#1}%
4039 \newtoggle{Xpstarteverytime@#1}%
4040 \newtoggle{Xpstart@#1}%
4041 \newtoggle{Xstanza@#1}%
4042 \csgdef{Xstanzaseparator@#1}{}%
4043 \csgdef{Xsymlinenum@#1}{}%
4044 \newtoggle{Xnonumber@#1}%
4045 \csgdef{Xbeforenumber@#1}{Opt}%
4046 \csgdef{Xafternumber@#1}{0.5em}%
4047 \newtoggle{Xnonbreakableafternumber@#1}%
4048 \csgdef{Xbeforesymlinenum@#1}{\csuse{Xbeforenumber@#1}}%
4049 \csgdef{Xaftersymlinenum@#1}{\csuse{Xafternumber@#1}}%
4050 \csgdef{Xinplaceofnumber@#1}{1em}%
4051 \global\cslet{Xlemmaseparator@#1}{\rbracket}%
4052 \csgdef{Xbeforelemmaseparator@#1}{0em}%
4053 \csgdef{Xafterlemmaseparator@#1}{0.5em}%
4054 \csgdef{Xinplaceofflemmaseparator@#1}{1em}%
4055 \csgdef{Xbeforenotes@#1}{1.2em \@plus .6em \@minus .6em}%
4056 \csgdef{Xafterrule@#1}{Opt}%
4057 \csgdef{Xtxtbeforenotes@#1}{}%
4058 \csgdef{Xmaxhnotes@#1}{0.8\vsizel}%
4059 \newtoggle{Xnoteswidthliketwocolumns@#1}%
4060 \csgdef{Xparafootsep@#1}{}%
4061 \csgdef{Xafternote@#1}{1em plus.4em minus.4em}%
4062 %

```

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

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

```

4063 \expandafter\newinsert\csname #1footins\endcsname%
4064 \unless\ifnoledgroup%
4065 \expandafter\newinsert\csname mp#1footins\endcsname%
4066 \fi%
4067 %

```

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.

```

4068 \global\newbool{parapparatus@}\expandafter\newcommand\expandafter
*}{\expandafter\newcommand}\csname #1footnote\endcsname[2] [] {%
4069 \ifnum\@edtext@level>0%
4070 \beginngroup%
4071 \newcommand{\content}{##2}%
4072 \ifnumberedpar%
4073 \ifledRcol%
4074 \ifluatex%
4075 \footnotelang@lua[R]%
4076 \fi%
4077 \@ifundefined{xpg@main@language}%if polyglossia
4078 {}%
4079 {\footnotelang@poly[R]}%
4080 \footnoteoptions@[R]{##1}{true}%
4081 \xright@appenditem{%
4082 \noexpand\prepare@preXnotes{#1}%
4083 \noexpand\prepare@edindex@fornote{\l@d@nums}%
4084 \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.
4085 \noexpand\setcounter{stanzaR}{\the\c@stanzaR}%Save
stanzaR counter for footnote
4086 \noexpand\csuse{v#1footnote}{#1}%
4087 {\l@d@nums}{\expandonce\@tag}{\expandonce\content}}
%
4088 }\to\inserts@listR
4089 \footnoteoptions@[R]{##1}{false}%
4090 \global\advance\insert@countR \@ne%
4091 \else%
4092 \ifluatex%
4093 \footnotelang@lua%
4094 \fi%
4095 \@ifundefined{xpg@main@language}%if polyglossia
4096 {}%
4097 {\footnotelang@poly}%
4098 \footnoteoptions@{##1}{true}%
4099 \xright@appenditem{%
4100 \noexpand\prepare@preXnotes{#1}%
4101 \noexpand\prepare@edindex@fornote{\l@d@nums}%
4102 \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.
4103 \ifl@dpairing%
4104 \noexpand\setcounter{stanzaL}{\the\c@stanzaL}%Save
stanzaR counter for footnote
4105 \fi%
4106 \noexpand\csuse{v#1footnote}{#1}%

```

```

4107             {{\l@d@nums}{\expandonce\@tag}{\expandonce\content}}
4108 %
4109             }\to\inserts@list
4110             \global\advance\insert@count \@ne%
4111             \footnoteoptions@{##1}{false}%
4112 \fi
4113 \else
4114 \csuse{v#1footnote}{#1}{0|0|0|0|0|0|0|0}{#1}%
4115 \fi%
4116 \endgroup%
4117 \else%
4118 \led@err@FootnoteWithoutEdtext%
4119 \fi%
4120 \ignorespaces%
4121 }
4122 %

```

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

```

4122 \global\csletcs{#1@@footnote}{#1footnote}
4123 %

```

XX.3.4 Set standard display

```

4124 \Xarrangement@normal{#1}%
4125 %

```

End of for critical footnotes.

```

4126 \fi
4127 %

```

XX.4 For familiar footnotes

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

```

4128 \unless\ifnofamiliar@
4129 %

```

XX.4.1 Options

```

4130 \newtoggle{parindentX@#1}
4131 \csgdef{hangindentX@#1}{Opt}%
4132 \csgdef{raggedX@#1}{}%
4133 \csgdef{hsizetwocolX@#1}{0.45 \hsize}%
4134 \csgdef{hsizethreecolX@#1}{.3 \hsize}%
4135 \csgdef{colalignX@#1}{\raggedright}%
4136 \csgdef{notenumfontX@#1}{\normalfont}%
4137 \csgdef{notefontsizeX@#1}{\footnotesize}%

```

```

4138 \csgdef{bhooknoteX@#1}{}%
4139 \csgdef{afterruleX@#1}{Opt}
4140 \csgdef{beforenotesX@#1}{1.2em \@plus .6em \@minus .6em}
4141 \csgdef{maxhnotesX@#1}{0.8\vsizex}%
4142 \newtoggle{noteswidthliketwocolumnsX@#1}%
4143 \csgdef{parafootsepX@#1}{}%
4144 \csgdef{afternoteX@#1}{1em plus.4em minus.4em}
4145 % End of for familiar footnotes.
4146 % \subsubsection{Create inserts, needed to add notes in foot}
4147 % As regards inserts, see chapter 15 of the TeXBook by D. Knuth.
4148 % \begin{macrocode}
4149 \expandafter\newinsert\csname footins#1\endcsname%
4150 \unless\ifnoledgroup@%
4151 \expandafter\newinsert\csname mpfootins#1\endcsname%
4152 \fi%
4153 %

```

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.

```

4154 \global\expandafter\newcommand\csname footnote#1\endcsname[1]{%
4155 \begingroup%
4156 \prepare@prenotesX{#1}%
4157 \newcommand{\content}{##1}%
4158 \stepcounter{footnote#1}%
4159 \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
4160 \nottoggle{nomk@}%Nomk is set to true when using \
footnoteXnomk with \parpackage
4161 {\csuse{@footnotemark#1}}%
4162 {}%
4163 \ifluatex%
4164 \xdef\footnote@luatextextdir{\the\textdir}%
4165 \xdef\footnote@luatexpardir{\the\pardir}%
4166 \fi%
4167 \csuse{vfootnote#1}{#1}{\expandonce\content}\m@mmf@prepare%
4168 \endgroup%
4169 }
4170 %
4171 %

```

Then define the counters.

```

4172 \newcounter{footnote#1}
4173 \global\expandafter\renewcommand\csname thefootnote#1\endcsname{\
arabic{footnote#1}}
4174 %

```

Do not forget to initialize series

```

4175 \arrangementX@normal{#1}%
4176 \fi

```



```
4177 %
```

XX.5 The endnotes

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

```
4178 \unless\ifnoend@
4179 %
```

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`

```
4180 \expandafter\newwrite\csname l@d@#1end\endcsname%
4181 \expandafter\newif\csname ifl@dend@#1\endcsname%
4182 %
```

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.

```
4183
4184 \global\expandafter\newcommandx\csname #1endnote\endcsname[2][1,
usedefault]{%
4185 \bgroup%
4186 \newlinechar='40%
4187 \global\@noneed@Footnotetrue%
4188 \newcommand{\content}{##2}%
4189 \expandafter\immediate\expandafter\write\csname l@d@#1end\
endcsname{%
4190 \expandafter\string\csname #1end\endcsname%
4191 {\ifnumberedpar@l@d@nums\fi}%
4192 {\ifnumberedpar@\expandonce\@tag\fi}%
4193 {\expandonce\content}%
4194 {\#1}%
4195 {\##1}%
4196 \@percentchar%
4197 }%
4198 \egroup%
4199 \ignorespaces%
4200 }%
4201 %
```

`\Xendnote` commands called `\Xend` commands on to the endnote file; these are analogous to the various `footfmt` commands above, and they take the same arguments. When we process this file, we want to pick out the notes of one series and ignore all the rest. To do that, we equate the end command for the series we want to `\endprint`, and leave the rest equated to `\@gobblefive`, which just skips over its five arguments.

```
4202
4203 \global\cslet{#1end}{\@gobblefive}
4204 %
```

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

```
4205 \global\expandafter\newcount\csname #1end@bysection\endcsname%
4206 %
```

XX.5.3 The options

```
4207 \csgdef{Xendtwolines@#1}{}%
4208 \csgdef{Xendmorethantwolines@#1}{}%
4209 \newtoggle{Xendtwolinesbutnotmore@#1}{}%
4210 \newtoggle{Xendtwolinesonlyinsamepage@#1}{}%
4211 \newtoggle{Xendlemmadisablefontselection@#1}%
4212 \csgdef{Xendnotenumfont@#1}{\normalfont}%
4213 \csgdef{Xendnotefontsize@#1}{\footnotesize}%
4214 \csgdef{Xendbhooknote@#1}{}%
4215
4216 \csgdef{Xendbeforenumber@#1}{Opt}%
4217 \csgdef{Xendafternumber@#1}{0.5em}%
4218
4219 \csgdef{Xendboxlinenum@#1}{Opt}%
4220 \csgdef{Xendboxlinenumalign@#1}{L}%
4221
4222 \csgdef{Xendboxstartlinenum@#1}{Opt}%
4223 \csgdef{Xendboxendlinenum@#1}{Opt}%
4224
4225 \csgdef{Xendlemmaseparator@#1}{}%
4226 \csgdef{Xendbeforelemmaseparator@#1}{0em}%
4227 \csgdef{Xendafterlemmaseparator@#1}{0.5em}%
4228 \csgdef{Xendinplaceoflemmaseparator@#1}{0.5em}%
4229
4230 \newtoggle{Xendparagraph@#1}%
4231 \csgdef{Xendafternote@#1}{1em plus.4em minus.4em}%
4232 \csgdef{Xendsep@#1}{}%
4233
4234 \csgdef{Xendinplaceofnumber@#1}{Opt}%
4235 \newtoggle{Xendnonumber@#1}%
4236
4237 \csgdef{Xendhangindent@#1}{Opt}%
4238 \newtoggle{Xendnumberonlyfirstinline@#1}%
4239 \newtoggle{Xendnumberonlyfirstintwolines@#1}%

```

```

4240 \csgdef{Xendbeforesymlinewidth@#1}{\csuse{Xendbeforenumber@#1}}%
4241 \csgdef{Xendaftersymlinewidth@#1}{\csuse{Xendafternumber@#1}}%
4242 \csgdef{Xendsymlinewidth@#1}{}%
4243 \csgdef{Xendboxsymlinewidth@#1}{Opt}%
4244
4245 \csgdef{Xendbhooklinewidth@#1}{}%
4246 \csgdef{Xendehooklinewidth@#1}{}%
4247 \csgdef{Xendbhookinplaceofnumber@#1}{}%
4248 \csgdef{Xendehookinplaceofnumber@#1}{}%
4249
4250 %

```

End of endnotes declaration

```

4251 \fi%
4252 %

```

Dump series in \@series

```

4253 \listxadd{\@series}{#1}
4254 }
4255 }% End of \newseries
4256 %

```

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

```

4257 \expandafter\newseries\expandafter{\default@series}
4258 %

```

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.

```

4259 \newcommand{\seriesatbegin}[1]{%
4260 \StrDel{\@series}{#1}[\@series]%
4261 \edef\@new{}%
4262 \listead{\@new}{#1}%
4263 \listead{\@new}{\@series}%
4264 \xdef\@series{\@new}%
4265 }
4266 %

```

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

```

4267 \newcommand{\seriesatend}[1]{%
4268 \StrDel{\@series}{#1}[\@series]%
4269 \edef\@new{}%
4270 \listead{\@new}{\@series}%

```

```

4271 \listeadadd{\@new}{#1}%
4272 \xdef\@series{\@new}%
4273 }
4274 %

```

XXI.2 Test series order

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

```

4275 \newcommand{\ifseriesbefore}[4]{%
4276   \StrPosition{\@series}{#1}[\@first]%
4277   \StrPosition{\@series}{#2}[\@second]%
4278   \ifnumgreater{\@second}{\@first}{#3}{#4}%
4279 }
4280 %

```

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}{%
4282   \ifdefempty{\@series}%
4283   {\xdef\@firstseries{}}%
4284   {\StrChar{\@series}{1}[\@firstseries]}%
4285 }%
4286 %

```

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`.

```

4287 \newcommandx{\settoggle@series}[5][4,5,usedefault]{%
4288   \def\do##1{%
4289     \global\settoggle{#2@##1}{#3}%
4290     \ifstrequal{#4}{critical}{
4291       \csuse{Xarrangement@\csuse{series@display##1}}{##1}%
4292     }{}
4293     \ifstrequal{#4}{familiar}{
4294       \csuse{arrangementX@\csuse{series@displayX##1}}{##1}%
4295     }{}
4296   }%
4297   \ifstreempty{#1}{%
4298     \dolistloop{\@series}%
4299     \ifstreempty{#5}{}{%
4300       \docsvlist{#5}%
4301     }
4302   }%
4303   {%
4304     \docsvlist{#1}%
4305   }%
4306 }
4307 %

```

`\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`.

```

4308 \newcommandx{\setcommand@series}[5][4,5,usedefault]{%
4309   \def\do##1{
4310     \csgdef{#2@##1}{#3}
4311     \ifstrequal{#4}{critical}{%
4312       \csuse{Xarrangement@}\csuse{series@display##1}}{##1}%
4313     }{}
4314     \ifstrequal{#4}{familiar}{%
4315       \csuse{arrangementX@}\csuse{series@displayX##1}}{##1}%
4316     }{}%
4317   }%
4318   \ifstrempy{#1}{%
4319     \dolistloop{\@series}%
4320     \ifstrempy{#5}{}{%
4321       \docsvlist{#5}
4322     }
4323   }%
4324   {%
4325     \docsvlist{#1}%
4326   }%
4327 }%
4328 %

```

XXI.3.3 Tools to generate options commands

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

```

4329 \newcommandx{\newhookcommand@series}[2][2,usedefault]{%
4330   \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][]{%
4331     \setcommand@series{##1}{#1}{##2}[][#2]%
4332   }%
4333   \ifstrempy{#2}{}{%
4334     \def\do##1{%
4335       \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname
4336       [1]{%
4337         \csuse{#1}[##1]{####1}%

```

```

4338 }%
4339 \docsvlist{#2}%
4340 }%
4341 }
4342 %

```

\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.

```

4343 \newcommand{\newhooktoggle@series}[2][2,usedefault]{%
4344 \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][1,2={
true},usedefault]{%
4345 \settoggle@series{##1}{#1}{##2}[][#2]%
4346 }%
4347 \ifstrempy{#2}{-}{%
4348 \def\do##1{%
4349 \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname{%
4350 \csuse{#1}[##1]%
4351 }%
4352 }%
4353 \docsvlist{#2}%
4354 }%
4355 }
4356 %

```

\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

```

4357 \newcommand{\newhooktoggle@series@reload}[2]{%
4358 \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][1,2={
true},usedefault]{%
4359 \settoggle@series{##1}{#1}{##2}[#2]%
4360 }%
4361 }%
4362 %

```

\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.

```

4363 \newcommand{\newhookcommand@series@reload}[2]{%
4364 \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][]{%
4365 \setcommand@series{##1}{#1}{##2}[#2]%
4366 }%
4367 }
4368 %

```

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.

```

4369 \unless\ifnocritical@
4370   \newhooktoggle@series{Xparindent}
4371   \newhookcommand@series{Xtwolines}[appref]
4372   \newhookcommand@series{Xmorethantwolines}[appref]
4373   \newhooktoggle@series{Xtwolinesbutnotmore}[appref]
4374   \newhooktoggle@series{Xtwolinesonlyinsamepage}[appref]
4375   \newhookcommand@series{Xhangindent}
4376   \newhookcommand@series{Xragged}
4377   \newhookcommand@series{Xhsizetwocol}
4378   \newhookcommand@series{Xhsizethreecol}
4379   \newhookcommand@series{Xcolalign}%
4380   \newhookcommand@series{Xnotenumfont}
4381   \newhookcommand@series{Xbhooknote}
4382   \newhookcommand@series{Xboxsymlinenum}%
4383   \newhookcommand@series{Xsymlinenum}
4384   \newhookcommand@series{Xbeforenumber}
4385   \newhookcommand@series{Xafternumber}
4386   \newhookcommand@series{Xbeforesymlinenum}
4387   \newhookcommand@series{Xaftersymlinenum}
4388   \newhookcommand@series{Xinplaceofnumber}
4389   \newhookcommand@series{Xlemmaseparator}
4390   \newhookcommand@series{Xbeforelemmaseparator}
4391   \newhookcommand@series{Xafterlemmaseparator}
4392   \newhookcommand@series{Xinplaceoflemmaseparator}
4393   \newhookcommand@series{Xtxtbeforenotes}
4394   \newhookcommand@series@reload{Xafterrule}{critical}
4395   \newhooktoggle@series{Xnumberonlyfirstinline}
4396   \newhooktoggle@series{Xnumberonlyfirstintwolines}
4397   \newhooktoggle@series{Xnonumber}
4398   \newhooktoggle@series{Xpstart}
4399   \newhooktoggle@series{Xpstarteverytime}%
4400
4401   \newhooktoggle@series{Xstanza}%
4402   \newhookcommand@series{Xstanzaseparator}%
4403
4404   \newhooktoggle@series{Xonlypstart}
4405   \newhooktoggle@series{Xnonbreakableafternumber}
4406   \newhooktoggle@series{Xlemmadisablefontselection}
4407   \newhookcommand@series@reload{Xmaxhnotes}{critical}
4408   \newhookcommand@series@reload{Xbeforenotes}{critical}
4409   \newhooktoggle@series@reload{Xnoteswidthliketwocolumns}{critical}%
4410   \newhookcommand@series{Xnotefontsize}
4411
4412   \newhookcommand@series{Xboxlinenum}%

```



```

4413 \newhookcommand@series{Xboxlinenumalign}%
4414
4415 \newhookcommand@series{Xboxstartlinenum}%
4416 \newhookcommand@series{Xboxendlinenum}%
4417
4418 \newhookcommand@series{Xafternote}%
4419 \newhookcommand@series{Xparafootsep}
4420
4421 \fi
4422 %

```

XXI.3.5 Options for familiar notes

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

```

4423 \unless\ifnofamiliar@
4424 \newhooktoggle@series{parindentX}
4425 \newhookcommand@series{hangindentX}
4426 \newhookcommand@series{raggedX}
4427 \newhookcommand@series{hsizetwocolX}
4428 \newhookcommand@series{hsizethreecolX}
4429 \newhookcommand@series{colalignX}%
4430 \newhookcommand@series{notenumfontX}
4431 \newhookcommand@series{bhooknoteX}
4432 \newhookcommand@series@reload{beforenotesX}{familiar}
4433 \newhookcommand@series@reload{maxhnotesX}{familiar}
4434 \newhooktoggle@series@reload{noteswidthliketwocolumnsX}{familiar}%
4435 \newhookcommand@series@reload{afterruleX}{familiar}
4436 \newhookcommand@series{notefontsizeX}
4437 \newhookcommand@series{afternoteX}
4438 \newhookcommand@series{parafootsepX}
4439 \fi
4440 %

```

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.

```

4441 \unless\ifnoend@
4442 \newhookcommand@series{Xendtwolines}[apprefwithpage]
4443 \newhookcommand@series{Xendmoreethantwolines}[apprefwithpage]
4444 \newhooktoggle@series{Xendtwolinesbutnotmore}[apprefwithpage]
4445 \newhooktoggle@series{Xendtwolinesonlyinsamepage}[apprefwithpage]
4446 \newhookcommand@series{Xendnotenumfont}
4447 \newhookcommand@series{Xendbhooknote}
4448
4449 \newhookcommand@series{Xendboxlinenum}%

```

```

4450 \newhookcommand@series{Xendboxlinenumalign}%
4451
4452 \newhookcommand@series{Xendboxstartlinenum}%
4453 \newhookcommand@series{Xendboxendlinenum}%
4454
4455 \newhookcommand@series{Xendnotefontsize}
4456 \newhooktoggle@series{Xendlemmadisablefontselection}
4457 \newhookcommand@series{Xendlemmaseparator}
4458 \newhookcommand@series{Xendbeforelemmaseparator}
4459 \newhookcommand@series{Xendafterlemmaseparator}
4460 \newhookcommand@series{Xendingplaceoflemmaseparator}
4461
4462 \newhookcommand@series{Xendbeforenumber}%
4463 \newhookcommand@series{Xendafternumber}%
4464
4465 \newhooktoggle@series{Xendparagraph}
4466 \newhookcommand@series{Xendafternote}
4467 \newhookcommand@series{Xendsep}
4468
4469 \newhookcommand@series{Xendingplaceofnumber}%
4470 \newhooktoggle@series{Xendnonumber}%
4471
4472 \newhooktoggle@series{Xendnumberonlyfirstinline}%
4473 \newhooktoggle@series{Xendnumberonlyfirstintwolines}%
4474
4475 \newhookcommand@series{Xendsymmlinenumber}%
4476 \newhookcommand@series{Xendbeforesymmlinenumber}%
4477 \newhookcommand@series{Xendaftersymmlinenumber}%
4478 \newhookcommand@series{Xendboxsymmlinenumber}%
4479
4480 \newhookcommand@series{Xendbhooklinenumber}%
4481 \newhookcommand@series{Xendahooklinenumber}%
4482 \newhookcommand@series{Xendbhookinplaceofnumber}%
4483 \newhookcommand@series{Xendahookinplaceofnumber}%
4484
4485 \newhookcommand@series{Xendhangindent}%
4486 \fi
4487 %

```

XXI.4 Hooks for a particular footnote

\fulllines@ \fulllines@ toggle is used to print the full lines references, and not the abbreviated form defined by \Xtwolines and \Xmorethantwolines.

```

4488 \newtoggle{fulllines@}%
4489 %

```

\nonum@ \nonum@ toggle is used to disable line number printing in a particular footnote.

```
4490 \newtoggle{nonum@}
4491 %
```

\nosep@ \nonum@ toggle is used to disable the lemma separator in a particular footnote.

```
4492 \newtoggle{nosep@}
4493 %
```

\nomk@ \nomk@ toggle is used by reledpar to remove the footnote mark in the text when using \footnoteXmk. Read reledpar handbook.

```
4494 \newtoggle{nomk@}%
4495 %
```

XXI.5 Alias

\Xnolemmaseparator \Xnolemmaseparator[*<series>*] is just an alias for \Xlemmaseparator[*<series>*]{ }.

```
4496 \newcommand*{\Xnolemmaseparator}[1][1]{\Xlemmaseparator[#1]{}}
4497 %
```

XXII Output routine

Now we begin the output routine and associated things.

XXII.0.1 Page number management

\pageno \pageno is a page number, starting at 1, and \advancepageno increments the number.
\advancepageno

```
4498 \countdef\pageno=0 \pageno=1
4499 \newcommand*{\advancepageno}{\ifnum\pageno<\z@ \global\advance\pageno\m@ne
4500 \else\global\advance\pageno\@ne\fi}
4501
4502 %
```

XXII.0.2 Extra footnotes output

With luck we might only have to change \@makecol and \@reinserts of the \TeX 's kernel. Since reledmac, we use etoolbox's patching commands instead of overriding. It should provides better compatibility with other package which modify these commands

\doextrafeet \doextrafeet is the code extending \@makecol to cater for the extra reledmac feet. We have two categories of extra footnotes. By default, we order the footnote inserts so that the regular footnotes of \TeX are first, then familiar familiar footnotes and finally the critical footnotes.

```

4503 \newcommand*\l@ddoxtrafeet}{%
4504   \IfStrEq{familiar-critical}{\@fnpos}
4505     {\do@feetX\Xdo@feet}%
4506     {%
4507       \IfStrEq{critical-familiar}{\@fnpos}%
4508         {\Xdo@feet\do@feetX}%
4509         {\do@feetX\Xdo@feet}%
4510     }%
4511 }%
4512
4513 %

```

\Xdo@feet \Xdo@feet is the code extending \@makecol to cater for the extra critical feet.

```

4514 \newcommand*\Xdo@feet}{%
4515   \setbox\@outputbox \vbox{%
4516     \unvbox\@outputbox
4517     \@opXfeet}}
4518 %

```

\@opXfeet The extra critical feet to be added to the output. The normal way to add one series, **\print@Xnotes** \print@Xnotes, is replaced by reledpar when using \Pages.

```

4519 \newcommand\print@Xnotes[1]{%
4520   \csuse{#1footstart}{#1}%
4521   \csuse{#1footgroup}{#1}%
4522 }%
4523 %

```

We print all series of notes by looping on them. We check before printing them that they are not voided.

```

4524 \newcommand*\@opXfeet}{%
4525   \unless\ifnocritical@%
4526     \gdef\firstXseries@{}%
4527     \def\do##1{%
4528       \ifvoid\csuse{##1footins}\else%
4529         \global\skip\csuse{##1footins}=\csuse{Xbeforenotes@##1}%
4530         \global\advance\skip\csuse{##1footins} by\csuse{Xafterrule@##1}%
4531         \print@Xnotes{##1}%
4532       \fi%
4533     }%
4534     \dolistloop{\@series}%
4535   \fi%
4536 }%
4537 %

```

\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.

```

4538 \newcommand*{\l@ddodoreinextrafeet}{%
4539   \IfStrEq{familiar-critical}{\@fnpos}
4540   {\@doreinfeetX\X@doreinfeet}%
4541   {%
4542     \IfStrEq{critical-familiar}{\@fnpos}%
4543     {\X@doreinfeet\@doreinfeetX}%
4544     {\@doreinfeetX\X@doreinfeet}%
4545   }%
4546 }
4547
4548 %

```

\X@doreinfeet \X@doreinfeet is the code for catering for the extra critical footnotes within \@reinserts.

```

4549 \newcommand*{\X@doreinfeet}{%
4550   \unless\ifnocritical@%
4551   \def\do##1{%
4552     \ifvoid\csuse{##1footins}\else%
4553     \insert\csuse{##1footins}{\unvbox\csuse{##1footins}}%
4554     \fi}%
4555   \dolistloop{\@series}
4556   \fi%
4557 }
4558
4559 %

```

\print@notesX We have to add all the new kinds of familiar footnotes to the output routine. The normal way to add one series. \print@Xnotes is replaced by reledpar when using \Pages.

\do@feetX

```

4560 \newcommand\print@notesX[1]{%
4561   \csuse{footstart#1}{#1}%
4562   \csuse{footgroup#1}{#1}%
4563 }%
4564 %

```

We print all the series of notes by looping on them. We check before printing them that they are not voided.

```

4565 \newcommand*{\do@feetX}{%
4566   \unless\ifnofamiliar@%
4567   \gdef\firstseriesX@{}%
4568   \setbox\@outputbox \vbox{%
4569     \unvbox\@outputbox%
4570     \def\do##1{%
4571       \ifvoid\csuse{footins##1}\else%
4572       \global\skip\csuse{footins##1}=\csuse{beforenotesX@##1}%
4573       \global\advance\skip\csuse{footins##1} by\csuse{afterruleX@##1}%
4574       \print@notesX{##1}%
4575       \fi%
4576     }%
4577     \dolistloop{\@series}}%

```

```

4578 \fi%
4579 }%
4580
4581 \newcommand{\@doreinfeetX}{%
4582 \unless\ifnofamiliar@%
4583 \def\do##1{%
4584 \ifvoid\csuse{footins##1}\else
4585 \insert%
4586 \csuse{footins##1}
4587 {\unvbox\csuse{footins##1}}}%
4588 \fi%
4589 }%
4590 \dolistloop{\@series}%
4591 \fi%
4592 }%
4593
4594 %

```

XXII.0.3 Standard output's commands patching

The memoir class does not use the ‘standard’ versions of `\@makecol` and `\@reinserts`, due to its sidebar insert. We had better add that code if memoir is used. (It can be awkward dealing with `\if` code within `\if` code, so don't use `\ifl@dmemoir` here.)

```

4595 \@ifclassloaded{memoir}{%
4596 %
4597
4598 \g@addto@macro{\m@mdoextrafeet}{\l@ddoxtrafeet}%
4599 \g@addto@macro{\m@mdodoreinextrafeet}{\l@ddodoreinxtrafeet}%
4600 }{%
4601 %

```

memoir is loaded so we use memoir's built in hooks.

memoir has not been loaded, so patch `\@makecol` and `\@reinserts`.

```

4601 \@ifpackageloaded{fancyhdr}{%
4602 \patchcmd%
4603 {\latex@makecol}%
4604 {\xdef\@freelist{\@freelist\@midlist}}}%
4605 {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
4606 {}%
4607 {\led@error@fail@patch@makecol}%
4608 }{%
4609 \patchcmd%
4610 {\@makecol}%
4611 {\xdef\@freelist{\@freelist\@midlist}}}%
4612 {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
4613 {}%
4614 {\led@error@fail@patch@makecol}%

```

```

4615 }%
4616
4617 \patchcmd%
4618 {\@reinserts}%
4619 {\ifvbox}%
4620 {\l@ddodoreinextrafeet\ifvbox}%
4621 {}%
4622 {\led@error@fail@patch@\@reinserts}%
4623 }
4624
4625 %

```

It turns out that \@doclearpage also needs modifying.

`\if@led@nofoot` We have to check if there are any leftover feet.

```

4626 \newif\if@led@nofoot
4627
4628 %

```

```

4629 \@ifclassloaded{memoir}{%
4630 %

```

If the memoir class is loaded we hook into its modified \@doclearpage.

```

\@mem@extranofeet%31 \g@addto@macro{\@mem@extranofeet}{%%
4632 \def\do#1{%
4633 \unless\ifnocritical@%
4634 \ifvoid\csuse{#1footins}\else\@mem@nofootfalse\fi%
4635 \fi%
4636 \unless\ifnofamiliar@%
4637 \ifvoid\csuse{footins#1}\else\@mem@nofootfalse\fi%
4638 \fi%
4639 }
4640 \dolistloop{\@series}%
4641 }%
4642 }{%
4643 %

```

As memoir is not loaded we have patch \@doclearpage.

```

\@led@testifnofoot%44 \newcommand*\@led@testifnofoot{%
\@doclearpage%45 \@led@nofoottrue%
4646 \ifvoid\footins\else%
4647 \@led@nofootfalse%
4648 \fi%
4649 \def\do##1{%
4650 \unless\ifnocritical@%
4651 \ifvoid\csuse{##1footins}\else%
4652 \@led@nofootfalse%

```

```

4653     \fi%
4654     \fi%
4655     \unless\ifnofamiliar@%
4656         \ifvoid\csuse{footins##1}\else%
4657             \@led@nofootfalse%
4658         \fi%
4659     \fi%
4660 }%
4661 \dolistloop{\@series}%
4662 }%
4663
4664 \pretocmd%
4665 {\@docclearpage}%
4666 {\@led@testifnofoot}%
4667 {}%
4668 {\led@error@fail@patch@@docclearpage}%
4669
4670 \patchcmd%
4671 {\@docclearpage}%
4672 {\ifvoid\footins}%
4673 {\if@led@nofoot}%
4674 {}%
4675 {\led@error@fail@patch@@docclearpage}%
4676
4677 }
4678
4679 %

```

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.

```

4680 \list@create{\labelref@list}
4681 %

```

\zz@@@ A convenience macro to zero two labeling counters in one go.


```

4682 \newcommand*{\zz@@@}{000|000} % set two counters to zero in one go
4683
4684 %

```

`\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.³²

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 Billett³³ found that the original code could be off by several pages. This version, hopefully cures that, and also allows for non-arabic page numbering.

```

4685 \newcommand*{\edlabel}[1]{%
4686   \ifl@dpairing\ifautopar%
4687   \strut%
4688   \fi\fi%
4689   \@bsphack%
4690   \ifledRcol%
4691     \write\linenum@out{\string\@lab}%
4692     \ifx\labelref@listR\empty%
4693       \xdef\label@refs{\zz@@@}%
4694     \else%
4695       \glp\labelref@listR\to\label@refs%
4696     \fi%
4697     \ifvmode%
4698       \advance\label@refs%
4699     \fi%
4700   %

```

Use code from the kernel `\label` command to write the correct page number. Also define an `hypertarget` if `hyperref` package is loaded.

```

4701   \protected@write\@auxout{%
4702     {\string\l@dmake@labelsRspace\thepage|\label@refs|\the\c@pstartR
4703     |{#1}}%
4704     \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1}}}{}}%
4705   \else%
4706     \write\linenum@out{\string\@lab}%
4707     \ifx\labelref@list\empty%
4708       \xdef\label@refs{\zz@@@}%
4709     \else%
4710       \glp\labelref@list\to\label@refs%
4711     \fi%
4712     \ifvmode%
4713       \advance\label@refs%

```

³²The remaining macros in this section were kindly revised by Wayne Sullivan, who substantially improved their efficiency and flexibility.

³³(jdb43@cam.ac.uk) via the ctt thread ‘ledmac cross referencing’, 25 August 2003.

```

4713 \fi%
4714 \protected@write\@auxout{}%
4715 {\string\l@dmake@labels\space\thepage\label@refs\the\c@pstart|{#1}}%
4716 \ifdef\hypertarget{\Hy@raisedlink{\hypertarget{#1}-}}}%
4717 \fi%
4718 \@esphack}%
4719
4720 %

```

`\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.

```

4721 \newcounter{line}%
4722 \newcounter{subline}%
4723 \newcommand{\advancelabel@refs}{%
4724   \setcounter{line}{\expandafter\labelrefsparseline\label@refs}%
4725   \stepcounter{line}%
4726   \ifsublines@%
4727     \setcounter{subline}{\expandafter\labelrefsparsesubline\label@refs}%
4728   %
4729     \stepcounter{subline}{1}%
4730     \def\label@refs{\theline|\thesubline}%
4731   \else%
4732     \def\label@refs{\theline|0}%
4733   \fi%
4734 }
4735 \def\labelrefsparseline#1|#2{#1}
4736 \def\labelrefsparsesubline#1|#2{#2}
4737 %

```

`\l@dmake@labels` The `\l@dmake@labels` macro gets executed when the labels file is read. For each label it defines a macro, whose name is made up partly from the label you supplied, that contains the page, line and sub-line numbers. But first it checks to see whether the label has already been used (and complains if it has).

The initial use of `\newcommand` is to catch if `\l@dmake@labels` has been previously defined (by a class or package).

#1 page number, #2 line number, #3 sub-line number, #4 pstart number, #5 label.

```

4737 \newcommand*{\l@dmake@labels}{%
4738 \def\l@dmake@labels#1|#2|#3|#4|#5{%
4739   \expandafter\ifx\csname the@label#5\endcsname \relax\else
4740     \led@warn@DuplicateLabel{#5}%
4741   \fi
4742   \expandafter\gdef\csname the@label#5\endcsname{#1|#2|#3|#4}%
4743   \ignorespaces}
4744
4745 %

```

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.

```

4746 \AtBeginDocument{%
4747   \def\l@make@labels#1|#2|#3|#4|#5{%
4748 }
4749
4750 %

```

\@lab The \@lab command, which appears in the \linenum@out file, appends the current values of page, line and sub-line to the \labelref@list. These values are defined by the earlier \@page, \@nl, and the \sub@on and \sub@off commands appearing in the \linenum@out file.

TeX uses the page counter for page numbers. However, it appears that this is not the right place to grab the page number. That task is now done in the \edlabel macro. This version of \@lab appends just the current line and sub-line numbers to \labelref@list.

```

4751
4752 \newcommand*{\@lab}{%
4753   \ifledRcol
4754     \xright@appenditem{\linenumr@p{\line@numR}}{%
4755       \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
4756     \to\labelref@listR
4757   \else
4758     \xright@appenditem{\linenumr@p{\line@num}}{%
4759       \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
4760     \to\labelref@list
4761   \fi}
4762 %

```

\applabel \applabel, if called in \edtext will insert automatically both a start and an end label for the current edtext lines.

```

4763 \newcommand*{\applabel}[1]{%
4764   \ifnum\@edtext@level>0%
4765 %

```

Label should not be already defined.

```

4766   \ifcsundef{the@label#1}{%
4767     \csdef{the@label#1}{\applabel}%
4768   }%
4769   {%
4770     \led@warn@DuplicateLabel{#1 (\applabel)}%
4771   }%
4772 %

```

Parse the \edtext line numbers.

```

4773   \expandafter\l@dp@rsefootspec\l@d@nums|%
4774 %

```

Use the \TeX standard hack for label.

```
4775 \bsphack%
4776 %
```

And now, write the data in the auxiliary file.

```
4777 \ifledRcol%
4778 \protected@write\@auxout{}%
4779 {\string\l@dmake@labelsR\space\l@dparsedstartpage|\l@dparsedstartline|\l@dparsedstartsub|\the\c@pstartR|{#1:start}}%
4780 \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1:start}}}{}}}%
4781 \protected@write\@auxout{}%
4782 {\string\l@dmake@labelsR\space\l@dparsedendpage|\l@dparsedendline|\l@dparsedendsub|\the\c@pstartR|{#1:end}}}%
4783 \else%
4784 \protected@write\@auxout{}%
4785 {\string\l@dmake@labels\space\l@dparsedstartpage|\l@dparsedstartline|\l@dparsedstartsub|\the\c@pstart|{#1:start}}%
4786 \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1:start}}}{}}}%
4787 \protected@write\@auxout{}%
4788 {\string\l@dmake@labels\space\l@dparsedendpage|\l@dparsedendline|\l@dparsedendsub|\the\c@pstart|{#1:end}}}%
4789 \fi%
4790 %
```

Use the \TeX standard hack for label.

```
4791 \esphack%
4792 %
```

Warning if `\applabel` is called outside of `\edtext`.

```
4793 \else%
4794 \led@warn@AppLabelOutEdtext{#1}%
4795 \fi%
4796 %
```

End of `\applabel`

```
4797 }%
4798 %
```

`\wrap@edcrossref` `\wrap@edcrossref` is called around all `reledmac` crossref commands, except those which start with `x`. It adds the hyperlink.

```
4799 \newrobustcmd{\wrap@edcrossref}[2]{%
4800 \ifdef{\hyperlink}%
4801 {\hyperlink{#1}{#2}}%
4802 {#2}%
4803 }
4804 %
```

`\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.

LaTeX already defines a `\pageref`, so changing the name to `\edpageref`.

```
4805 \newcommand*\edpageref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{1}{#1}}}
4806 \newcommand*\xpageref}[1]{\l@dgetref@num{1}{#1}}
4807
4808 %
```

`\edlineref` If the specified label exists, `\lineref` gives its line number.

`\xlineref`

```
4809 \newcommand*\edlineref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{2}{#1}}}%
4810 \newcommand*\xlineref}[1]{\l@dgetref@num{2}{#1}}%
4811
4812 %
```

`\sublineref` If the specified label exists, `\sublineref` gives its sub-line number.

`\xsublineref`

```
4813 \newcommand*\sublineref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{3}{#1}}}
4814 \newcommand*\xsublineref}[1]{\l@dgetref@num{3}{#1}}
4815
4816 %
```

`\pstarteref` If the specified label exists, `\pstartref` gives its pstart number.

`\xpstartref`

```
4817 \newcommand*\pstartref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{4}{#1}}}
4818 \newcommand*\xpstartref}[1]{\l@dgetref@num{4}{#1}}
4819
4820 %
```

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.

```
4821 \newcommand*\l@dref@undefined}[1]{%
4822 \expandafter\ifx\csname the@label#1\endcsname\relax
```

```

4823 \led@warn@RefUndefined{#1}%
4824 \fi}
4825
4826 %

```

\l@dgetref@num Next, `\l@dgetref@num` fetches the number we want. It has two arguments: the first is simply a digit, specifying whether to fetch a page (1), line (2), sub-line (3) or (4) pstart number. (This switching is done by calling `\l@dlabel@parse`.) The second argument is the label-macro, which because of the `\@lab` macro above is defined to be a string of the type 123|456|789.

```

4827 \newcommand*{\l@dgetref@num}[2]{%
4828 \expandafter
4829 \ifx\csname the@label#2\endcsname \relax
4830 000%
4831 \else
4832 \expandafter\expandafter\expandafter
4833 \l@dlabel@parse\csname the@label#2\endcsname|#1%
4834 \fi}
4835
4836 %

```

\l@dlabel@parse Notice that we slipped another `|` delimiter into the penultimate line of `\l@dgetref@num`, to keep the ‘switch-number’ separate from the reference numbers. This `|` is used as another parameter delimiter by `\l@dlabel@parse`, which extracts the appropriate number from its first arguments. The `|`-delimited arguments consist of the expanded label-macro (three reference numbers), followed by the switch-number (1, 2, 3 or 4) which defines which of the earlier three numbers to pick out. (It was earlier given as the first argument of `\l@dgetref@num`.)

```

4837 \newcommand*{\l@dlabel@parse}{%
4838 \def\l@dlabel@parse#1|#2|#3|#4|#5{%
4839 \ifcase #5%
4840 \or #1%
4841 \or #2%
4842 \or #3%
4843 \or #4%
4844 \fi}
4845 %

```

\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.

```

4846 \newcommand*{\xxref}[2]{%
4847   {%
4848     \expandafter\ifx\csname the@label#1\endcsname \relax%
4849     \expandafter\let\csname the@label#1\endcsname\zz@@@%
4850     \else%
4851     \expandafter\def\csname the@label#1\endcsname{\l@dgetref@num
4852 {1}{#1}|\l@dgetref@num{2}{#1}|\l@dgetref@num{3}{#1}}%
4853     \fi%
4854     \expandafter\ifx\csname the@label#2\endcsname \relax%
4855     \expandafter\let\csname the@label#2\endcsname\zz@@@%
4856     \else%
4857     \expandafter\def\csname the@label#2\endcsname{\l@dgetref@num
4858 {1}{#2}|\l@dgetref@num{2}{#2}|\l@dgetref@num{3}{#2}}%
4859     \fi%
4860     \ifdefined\@Rlineflag%
4861     \StrDel{\csuse{the@label#1}}{\@Rlineflag}[\@tempa]%
4862     \StrDel{\csuse{the@label#2}}{\@Rlineflag}[\@tempb]%
4863     \else%
4864     \letcs{\@tempa}{the@label#1}%
4865     \letcs{\@tempb}{the@label#2}%
4866     \fi%
4867     \linenum{\@tempa}%
4868     \@tempb}}%
4869 %

```

\appref \appref prints a crossref to some lines of the apparatus defined by \applabel. It prints the lines as they should be printed in the apparatus.

\apprefwithpage If \@apprefprefixsingle is not empty, it prints it before the line number. If \@apprefprefixmore is not empty, it prints it before the line numbers when the first line is not the same as the last line. \apprefwithpage prints a crossref to some lines of the apparatus defined by \applabel. It always prints the page number, as it should be printed in the end notes. The \Xtwolinesappref and \Xmorethantwolinesappref are similar to the footnote hooks and \Xtwolines \Xmorethantwolines.

So, first declare the default value of the hooks for the pseudo-series appref. Also declare the internal toggle which are switch by reledmac.

```

4869 \xdef\Xtwolines@appref{}%
4870 \xdef\Xmorethantwolines@appref{}%
4871 \newtoggle{Xtwolinesbutnotmore@appref}%
4872 \newtoggle{Xtwolinesonlyinsamepage@appref}%
4873
4874 \xdef\Xendtwolines@apprefwithpage{}%
4875 \xdef\Xendmorethantwolines@apprefwithpage{}%
4876 \newtoggle{Xendtwolinesbutnotmore@apprefwithpage}%
4877 \newtoggle{Xendtwolinesonlyinsamepage@apprefwithpage}%
4878
4879 %

```

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.

```

4880
4881 \xdef\Xboxstartlinenum@appref{Opt}
4882 \xdef\Xboxendlinenum@appref{Opt}
4883
4884 \xdef\Xendboxstartlinenum@apprefwithpage{Opt}
4885 \xdef\Xendboxendlinenum@apprefwithpage{Opt}
4886
4887 %

```

Now, declare the default value of `\@apprefprefixsingle` and `\@apprefprefixmore`, and the commands which defines them

```

4888 \newcommand\@apprefprefixsingle{}%
4889 \newcommand\@apprefprefixmore{}%
4890
4891 \newcommand{\apprefprefixsingle}[1]{%
4892   \gdef\@apprefprefixsingle{#1}%
4893 }
4894
4895 \newcommand{\setapprefprefixmore}[1]{%
4896   \gdef\@apprefprefixmore{#1}%
4897 }
4898
4899 %

```

And now, the main commands: `\appref` and `\apprefwithpage`. These commands call `\printlines` and `\printendlines`. That is why we have previously declared all hooks values tested inside these last commands.

```

4900 \newcommandx{\appref}[2][1,usedefault]{%
4901   \IfStrEq{#1}{fulllines}%
4902     {\toggletrue{fulllines@}}%
4903     {}%
4904   \xdef\@currentseries{appref}%
4905   \ifdefempty{\@apprefprefixmore}%
4906     {\@apprefprefixsingle}%
4907     {%
4908       \IfEq{\xlineref{#2:start}}{\xlineref{#2:end}}%
4909       {\@apprefprefixsingle}%
4910       {\@apprefprefixmore}%
4911     }%
4912   \printlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:start}
4913   }|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}||%
4914   \togglefalse{fulllines@}%
4915 }%
4916
4917 % \changes{v1.23.0}{2015/05/18}{Debug \protect\cs{Xendtwolines}, \protect\
4918   cs{Xendmorethantwolines}, \protect\cs{Xendtwolinesbutnotmore} and \protect\

```



```

cs{Xendtwolinesonlyinsamepage} when using \protect\cs{apprefwithpage}.}
4917 \newcommand{\apprefwithpage}[2][1,usedefault]{%
4918   \IfStrEq{#1}{fulllines}%
4919     {\toggletrue{fulllines@}}%
4920     {}%
4921   \xdef\@currentseries{apprefwithpage}%
4922   \printendlines\pageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:
start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}||%
4923   \togglefalse{fulllines@}%
4924 }%
4925 %

```

\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`.

```

4926 \newcommand*{\edmakelabel}[2]{\expandafter\xdef\csname the@label#1\
endcsname{#2}}
4927
4928 %

```

(If you are only going to refer to such a label using `\xxref`, then you can omit entries in the same way as with `\linenum` (see VI.3 p. 111 and V.9 p. 82), since `\xxref` makes a call to `\linenum` in order to do its work.)

XXIV Side notes

Regular `\marginpars` do not work inside numbered text — they do not produce any note but do put an extra unnumbered blank line into the text.

\@xympar Changing `\@xympar` a little at least ensures that `\marginpars` in numbered text do not disturb the flow.

```

4929 \pretocmd{\@xympar}%
4930   {\ifnumberedpar@
4931     \led@warn@NoMarginpars
4932     \@esphack
4933   \else}%
4934 {}%
4935 {}%
4936
4937 \apptocmd{\@xympar}%
4938   {\fi}%
4939 {}
4940 {}

```

```
4941
4942 %
```

We provide side notes as replacement for `\marginpar` in numbered text.

`\sidenote@margin` These are the sidenote equivalents to `\line@margin` and `\linenummargin` for specifying which margin. The default is the right margin (opposite to the default for line numbers). `\l@dgetsidenote@margin` returns the number associated to side note margin:

left : 0

right : 1

outer : 2

inner : 3

```
4943 \newcount\sidenote@margin
4944 \newcommand*{\sidenotemargin}[1]{\%
4945   \l@dgetsidenote@margin{#1}%
4946   \ifnum\@l@tempcntb>\m@ne
4947     \ifledRcol
4948       \global\sidenote@marginR=\@l@tempcntb
4949     \else
4950       \global\sidenote@margin=\@l@tempcntb
4951     \fi
4952   \fi}}
4953 \newcommand*{\l@dgetsidenote@margin}[1]{\%
4954   \def\@tempa{#1}\def\@tempb{left}%
4955   \ifx\@tempa\@tempb
4956     \@l@tempcntb \z@
4957   \else
4958     \def\@tempb{right}%
4959     \ifx\@tempa\@tempb
4960       \@l@tempcntb \@ne
4961     \else
4962       \def\@tempb{outer}%
4963       \ifx\@tempa\@tempb
4964         \@l@tempcntb \tw@
4965       \else
4966         \def\@tempb{inner}%
4967         \ifx\@tempa\@tempb
4968           \@l@tempcntb \thr@@
4969         \else
4970           \led@warn@BadSidenotemargin
4971           \@l@tempcntb \m@ne
4972         \fi
4973       \fi
4974     \fi
```

```

4975 \fi}
4976 \sidenotemargin{right}
4977
4978 %

```

`\l@dlp@rbox` We need two boxes to store sidenote texts.

```

\l@drp@rbox
4979 \newbox\l@dlp@rbox
4980 \newbox\l@drp@rbox
4981
4982 %

```

`\ledlsnotewidth` These specify the width of the left/right boxes (initialised to `\marginparwidth`), their distance from the text (initialised to `\linenumsep`), and the fonts used.

```

\ledrsnotewidth
\ledlsnotesep
4983 \newdimen\ledlsnotewidth \ledlsnotewidth=\marginparwidth
\ledrsnotesep
4984 \newdimen\ledrsnotewidth \ledrsnotewidth=\marginparwidth
\ledlsnotefontsetup
4985 \newdimen\ledlsnotesep \ledlsnotesep=\linenumsep
\ledrsnotefontsetup
4986 \newdimen\ledrsnotesep \ledrsnotesep=\linenumsep
4987 \newcommand*\ledlsnotefontsetup{\raggedleft\footnotesize}
4988 \newcommand*\ledrsnotefontsetup{\raggedright\footnotesize}
4989
4990 %

```

`\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
4991 \newcommand*\ledleftnote[1]{\edtext{}\l@dlsnote{#1}}
4992 \newcommand*\ledrightnote[1]{\edtext{}\l@drsnote{#1}}
4993
4994 \newcommand*\ledinnernote[1]{%
4995 \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
4996 \ledleftnote{#1}%
4997 \else%
4998 \ledrightnote{#1}%
4999 \fi%
5000 }
5001
5002 \newcommand*\ledouternote[1]{%
5003 \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
5004 \ledrightnote{#1}%
5005 \else%
5006 \ledleftnote{#1}%
5007 \fi%
5008 }
5009

```

```

5010 \newcommand*{\ledsidenote}[1]{\edtext{}{\l@dcsnote{#1}}}
5011 %

```

\l@dlsnote . The ‘footnotes’ for left, right, and moveable sidenotes. The whole scheme is reminiscent of the critical footnotes code.

```

\l@dcsnote
5012 \newif\ifrightrightnoteup
5013 \rightnoteuptrue
5014
5015 \newcommand*{\l@dlsnote}[1]{%
5016 \begingroup%
5017 \newcommand{\content}{#1}%
5018 \ifnumberedpar@
5019 \ifledRcol%
5020 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}{%
5021 \to\inserts@listR
5022 \global\advance\insert@countR \@ne%
5023 \else%
5024 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}{%
5025 \to\inserts@list
5026 \global\advance\insert@count \@ne%
5027 \fi
5028 \fi\ignorespaces\endgroup}
5029
5030 \newcommand*{\l@drsnote}[1]{%
5031 \begingroup%
5032 \newcommand{\content}{#1}%
5033 \ifnumberedpar@
5034 \ifledRcol%
5035 \xright@appenditem{\noexpand\l@drsnote{\expandonce\content}}{%
5036 \to\inserts@listR
5037 \global\advance\insert@countR \@ne%
5038 \else%
5039 \xright@appenditem{\noexpand\l@drsnote{\expandonce\content}}{%
5040 \to\inserts@list
5041 \global\advance\insert@count \@ne%
5042 \fi
5043 \fi\ignorespaces\endgroup}
5044
5045 \newcommand*{\l@dcsnote}[1]{%
5046 \begingroup%
5047 \newcommand{\content}{#1}%
5048 \ifnumberedpar@
5049 \ifledRcol%
5050 \xright@appenditem{\noexpand\l@dcsnote{\expandonce\content}}{%
5051 \to\inserts@listR
5052 \global\advance\insert@countR \@ne%
5053 \else%
5054 \xright@appenditem{\noexpand\l@dcsnote{\expandonce\content}}{%
5055 \to\inserts@list

```

```

5056 \global\advance\insert@count \@ne%
5057 \fi
5058 \fi\ignorespaces\endgroup}
5059
5060 %

```

\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.

```

5061 \newcommand*{\vl@dlsnote}[1]{%
5062 \ifledRcol{%
5063 \l@dtempcntb=\sidenote@marginR%
5064 \ifnum\l@dtempcntb>\@ne%
5065 \advance\l@dtempcntb by\page@numR%
5066 \fi%
5067 \else%
5068 \l@dtempcntb=\sidenote@margin%
5069 \ifnum\l@dtempcntb>\@ne%
5070 \advance\l@dtempcntb by\page@num%
5071 \fi%
5072 \fi%
5073 \ifodd\l@dtempcntb%
5074 \listgadd{\l@dcsnotetext@l}{#1}%
5075 \else%
5076 \listgadd{\l@dcsnotetext}{#1}%
5077 \fi
5078 }
5079 \newcommand*{\vl@drsnote}[1]{%
5080 \ifledRcol{%
5081 \l@dtempcntb=\sidenote@marginR%
5082 \ifnum\l@dtempcntb>\@ne%
5083 \advance\l@dtempcntb by\page@numR%
5084 \fi%
5085 \else%

```

```

5086 \l@dttempcntb=\sidenote@margin%
5087 \ifnum\l@dttempcntb>\@ne%
5088 \advance\l@dttempcntb by\page@num%
5089 \fi%
5090 \fi%
5091 \ifodd\l@dttempcntb%
5092 \listgadd{\l@dcsnotetext}{#1}%
5093 \else%
5094 \listgadd{\l@dcsnotetext@r}{#1}%
5095 \fi%
5096 }
5097 \newcommand*{\vl@dcsnote}[1]{\listgadd{\l@dcsnotetext}{#1}}
5098
5099 %

```

\setl@dlp@rbox `\setl@dlp@rbox{<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.

```

5100 \newcommand*{\setl@dlp@rbox}[1]{%
5101 {\parindent\z@\hsize=\ledlsnotewidth\ledlsnotefontsetup
5102 \global\setbox\l@dlp@rbox
5103 \ifleftnoteup
5104 =\vbox to\z@{\vss #1}%
5105 \else
5106 =\vbox to 0.70\baselineskip{\strut#1\vss}%
5107 \fi}}
5108 \newcommand*{\setl@drp@rbox}[1]{%
5109 {\parindent\z@\hsize=\ledrsnotewidth\ledrsnotefontsetup
5110 \global\setbox\l@drp@rbox
5111 \ifrightrightnoteup
5112 =\vbox to\z@{\vss#1}%
5113 \else
5114 =\vbox to0.7\baselineskip{\strut#1\vss}%
5115 \fi}}
5116 \newif\ifleftnoteup
5117 \leftnoteuptrue
5118 %

```

\@sidenotesep This macro is used to separate sidenotes of the same line.

```

5119 \newcommand{\setsidenotesep}[1]{\gdef\@sidenotesep{#1}}
5120 \newcommand{\@sidenotesep}{, }
5121 %

```

\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.

```

5122 \newcommand*{\affixside@note}{%
5123   \def\sidenotecontent@{%
5124     \numgdef{\itemcount@}{0}%
5125     \def\do##1{%
5126       \ifnumequal{\itemcount@}{0}%
5127         {%
5128           \appto\sidenotecontent@{##1}}% Not print not separator before
the 1st note
5129         {\appto\sidenotecontent@{\@sidenotesep ##1}%
5130         }%
5131       \numgdef{\itemcount@}{\itemcount@+1}%
5132     }%
5133   \dolistloop{\l@dcstotetext}%
5134   \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
5135   %

```

And we do the same for left and right notes (not movable).

```

5136 \gdef\@templ@d{%
5137 \gdef\@templ@n{\l@dcstotetext\l@dcstotetext@1\l@dcstotetext@r}%
5138 \ifx\@templ@d\@templ@n \else%
5139   \if@twocolumn%
5140     \if@firstcolumn%
5141       \setl@dlp@rbox{##1}{\sidenotecontent@}%
5142     \else%
5143       \setl@drp@rbox{\sidenotecontent@}%
5144     \fi%
5145   \else%
5146     \l@dttempcntb=\sidenote@margin%
5147     \ifnum\l@dttempcntb>\@ne%
5148       \advance\l@dttempcntb by\page@num%
5149     \fi%
5150     \ifodd\l@dttempcntb%
5151       \setl@drp@rbox{\sidenotecontent@}%
5152       \gdef\sidenotecontent@{%
5153         \numgdef{\itemcount@}{0}%
5154         \dolistloop{\l@dcstotetext@1}%
5155         \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
5156         \setl@dlp@rbox{\sidenotecontent@}%
5157       \else%
5158         \setl@dlp@rbox{\sidenotecontent@}%
5159         \gdef\sidenotecontent@{%
5160           \numgdef{\itemcount@}{0}%
5161           \dolistloop{\l@dcstotetext@r}%
5162           \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
5163           \setl@drp@rbox{\sidenotecontent@}%
5164         \fi%
5165       \fi%
5166     \fi%
5167   }
5168   %

```

XXV Minipages and such

We can put footnotes into minipages. The preparatory code has been set up earlier, all that remains is to ensure that it is available inside a minipage box. This requires some alteration to the kernel code, specifically the `\@iiminipage` and `\endminipage` macros. We will arrange this so that additional series can be easily added.

`\l@dfeetbeginmini` These will be the hooks in `\@iiminipage` and `\endminipage`.
`\l@dfeetendmini` They can be extended to handle other things if necessary.

```

5169 \ifnoledgroup@else%
5170 \newcommand*{\l@dfeetbeginmini}{\l@dedbeginmini\l@dfambeginmini}
5171 \newcommand*{\l@dfeetendmini}{%
5172   \IfStrEq{critical-familiar}{\@mpfnpos}%
5173   {\l@dedendmini\l@dfamendmini}%
5174   {%
5175     \IfStrEq{familiar-critical}{\@mpfnpos}%
5176     {\l@dfamendmini\l@dedendmini}%
5177     {\l@dedendmini\l@dfamendmini}%
5178   }%
5179 }%
5180 %

```

`\l@dedbeginmini` These handle the initiation and closure of critical footnotes in a minipage environment.

```

\l@dedendmini
5181 \newcommand*{\l@dedbeginmini}{%
5182   \unless\ifnocritical@%
5183   \def\do##1{\csletcs{v##1footnote}{mpv##1footnote}}%
5184   \dolistloop{\@series}%
5185   \fi%
5186 }
5187 \newcommand*{\l@dedendmini}{%
5188   \unless\ifnocritical@%
5189   \ifl@dpairing%
5190     \ifledRcol%
5191       \flush@notesR%
5192     \else%
5193       \flush@notes%
5194     \fi%
5195   \fi
5196   \def\do##1{%
5197     \ifvoid\csuse{mp##1footins}\else%
5198     \ifl@dpairing\ifparledgroup%
5199       \ifledRcol%
5200         \dingdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+
skip\@nameuse{mp##1footins}}%
5201       \else%
5202         \dingdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL
+skip\@nameuse{mp##1footins}}%
5203       \fi%

```



```

5204     \fi\fi%
5205     \csuse{mp##1footgroup}{##1}%
5206     \fi}%
5207     \dolistloop{\@series}%
5208     \fi%
5209 }%
5210
5211 %

```

\l@dfambeginmini These handle the initiation and closure of familiar footnotes in a minipage environment.

\l@dfamendmini

```

5212 \newcommand*\l@dfambeginmini{%
5213   \unless\ifnofamiliar%
5214     \def\do##1{\csletcs{vfootnote##1}{mpvfootnote##1}}%
5215     \dolistloop{\@series}%
5216   \fi%
5217 }%
5218
5219 \newcommand*\l@dfamendmini{%
5220   \unless\ifnofamiliar%
5221     \def\do##1{%
5222       \ifvoid\csuse{mpfootins##1}\else%
5223         \csuse{mpfootgroup##1}{##1}%
5224       \fi}%
5225     \dolistloop{\@series}%
5226   \fi%
5227 }%
5228 %

```

\@iiiminipage This is our extended form of the kernel `\@iiiminipage` defined in `ltboxes.dtx`.

```

5229 \patchcmd%
5230   {\@iiiminipage}%
5231   {\let\@footnotetext\@mpfootnotetext}%
5232   {\let\@footnotetext\@mpfootnotetext\l@dfeetbeginmini}%
5233   {}%
5234   {\led@error@fail@patch@\@iiiminipage}%
5235   %

```

\endminipage This is our extended form of the kernel `\endminipage` defined in `ltboxes.dtx`.

```

5236 \patchcmd%
5237   {\endminipage}%
5238   {\footnoterule}%
5239   {\footnoterule\l@advance@parledgroup@beforenormalnotes}%
5240   {}%
5241   {\led@error@fail@patch@endminipage}%
5242
5243 \patchcmd%
5244   {\endminipage}%

```

```

5245 {\@minipagefalse}%
5246 {\l@dfeetendmini\@minipagefalse}%
5247 }%
5248 {\led@error@fail@patch@endminipage}
5249
5250 %

```

`\l@dunboxmpfoot` `\@ldunboxmpfoot` insert normal footnotes for ledgroup.
`\advance@parledgroup@beforenormalnotes`

```

5251 \newcommand*{\l@dunboxmpfoot}{%
5252   \vskip\skip\@mpfootins
5253   \normalcolor
5254   \footnoterule
5255   \l@advance@parledgroup@beforenormalnotes
5256   \unvbox\@mpfootins%
5257 }
5258 %

```

When using parallel ledgroup, we need to store the vertical space added before footnote, in order to compensate them between left and right pages.

```

5259 \newcommand{\l@advance@parledgroup@beforenormalnotes}{%
5260   \ifparledgroup
5261     \ifl@dpairing
5262       \ifledRcol
5263         \dimgdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+\
skip\@mpfootins}
5264       \else
5265         \dimgdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL+\
skip\@mpfootins}
5266       \fi
5267     \fi
5268   \fi
5269 }
5270 %

```

`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.

```

5271 \newenvironment{ledgroup}{%
5272   \resetprevpage@num%
5273   \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@%
5274   \let\@footnotetext\@mpfootnotetext
5275   \l@dfeetbeginmini%
5276 }{%
5277   \par
5278   \unskip
5279   \ifvoid\@mpfootins\else
5280     \l@dunboxmpfoot

```

```

5281 \fi
5282 \l@dfeetendmini%
5283 }
5284
5285 %

```

`\ledgroupsize` `\begin{ledgroupsize}` [*pos*] {*width*}

This environment puts footnotes at the end, even if that happens to be in the middle of a page, or crossing a page boundary. It is a sort of unboxed, variable *width* minipage. The optional *pos* controls the sideways position of numbered text.

```

5286 \newenvironment{ledgroupsize}[2][1]{%
5287 %

```

Set the various text measures.

```

5288 \hsize #2\relax
5289 %% \textwidth #2\relax
5290 %% \columnwidth #2\relax
5291 %

```

Initialize fills for centering.

```

5292 \let\ledllfill\hfil
5293 \let\ledrlfill\hfil
5294 \def\@tempa{#1}\def\@tempb{1}%
5295 %

```

Left adjusted numbered lines

```

5296 \ifx\@tempa\@tempb
5297 \let\ledllfill\relax
5298 \else
5299 \def\@tempb{r}%
5300 \ifx\@tempa\@tempb
5301 %

```

Right adjusted numbered lines

```

5302 \let\ledrlfill\relax
5303 \fi
5304 \fi
5305 %

```

Set up the footnoting.

```

5306 \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
5307 \let\@footnotetext\@mpfootnotetext
5308 \l@dfeetbeginmini%
5309 }{
5310 \par
5311 \unskip
5312 \ifvoid\@mpfootins\else
5313 \l@dunboxmpfoot
5314 \fi

```

```

5315 \l@dfetendmini%
5316 }
5317
5318 %

```

Close the \ifnoledgroup@\else.

```

5319 \fi%
5320 %

```

`\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`.

```

5321 \newif\ifledgroupnotesL@
5322 \newif\ifledgroupnotesR@
5323 %

```

XXVI Indexing

Here is some code for indexing using page and line numbers.

First, ensure that `imakeidx` or `indextools` is loaded *before* `eledmac`.

```

5324 \AtBeginDocument{%
5325   \unless\ifl@imakeidx%
5326     \@ifpackageloaded{imakeidx}{\led@error@ImakeidxAfterEledmac}{}%
5327   \fi%
5328   \unless\ifl@indextools%
5329     \@ifpackageloaded{indextools}{\led@error@indextoolsAfterEledmac}{}%
5330   \fi%
5331 }
5332 %

```

`\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
5333 \newcommand{\pagelinesep}{-}
5334 \newcommand{\edindexlab}{${}}
5335 \newcounter{labidx}
5336 \setcounter{labidx}{0}
5337
5338 %

```

`\doedindexlabel` This macro sets an `\edlabel`.

```

5339 \newcommand{\doedindexlabel}{%
5340   \stepcounter{labidx}%
5341   \edlabel{\edindexlab\thelabidx}%
5342 }
5343
5344 %

```

`\thepageline` This macro makes up the page/line number combo from the label/ref.

```

5345 \newcommand{\thepageline}{%
5346   \thepage%
5347   \pagelinesep%
5348   \xlineref{\edindexlab\thelabidx}%
5349 }
5350 %

```

`\thestartpageline` These macros make up the page/line start/end number when the `\edindex` command is called in critical notes.
`\theendpageline`

```

5351 \newcommand{\thestartpageline}{%
5352   \l@dparsedstartpage%
5353   \pagelinesep%
5354   \l@dparsedstartline%
5355 }
5356 \newcommand{\theendpageline}{%
5357   \l@dparsedendpage%
5358   \pagelinesep%
5359   \l@dparsedendline%
5360 }
5361 %

```

`\if@edindex@fornote@true` This boolean test is switching at the beginning of each critical note, to allow index referring to this note.

```

5362 \newif\if@edindex@fornote@
5363 %

```

`\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 |.

```

5364 \newcommand{\prepare@edindex@fornote}[1]{%
5365   \l@dp@rsefootspec#1|%
5366   \@edindex@fornote@true%
5367 }
5368 %

```

`\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.

```

5369 \newcommand{\get@edindex@ledinnote@command}{%
5370   \ifxindy@%
5371     \gdef\@ledinnote@command{%
5372       ledinnote\thelabidx%

```

```

5373 }%
5374 \ifxindyhyperref%
5375   \immediate\write\eledmac@xindy@out{%
5376     (define-attributes ("ledinnote\thelabidx"))^^J
5377     \space\space(markup-locref^^J
5378     \eledmacmarkuplocdepth^^J
5379     :open "\string\ledinnote[\edindexlab\thelabidx]{\@index@command
5380 }{"^^J
5381   :close "}"^^J
5382   :attr "ledinnote\thelabidx"^^J
5383   )
5384 }%
5385 \else%
5386   \immediate\write\eledmac@xindy@out{%
5387     (define-attributes ("ledinnote\thelabidx"))^^J
5388     \space\space(markup-locref^^J
5389     \eledmacmarkuplocdepth^^J
5390     :open "\string\ledinnote{\@index@command}"^^J
5391     :close "}"^^J
5392     :attr "ledinnote\thelabidx"^^J
5393   )
5394 }%
5395 \fi%
5396 %

```

If we do not use xindy option, \@ledinnote@command will produce something like ledinnote{formattingcommand}.

```

5396 \else%
5397   \gdef\@ledinnote@command{%
5398     ledinnote[\edindexlab\thelabidx]{\@index@command}%
5399   }%
5400 \fi%
5401 }
5402 %

```

\get@index@command This macro is used to analyse if a text to be indexed has a command after a |.

```

5403 \def\get@index@command#1|#2+{%
5404   \gdef\@index@txt{#1}%
5405   \gdef\@index@command{#2}%
5406   \xdef\@index@parenthesis{}%
5407   \IfBeginWith{\@index@command}{(}{%
5408     \StrGobbleLeft{\@index@command}{1}[\@index@command@]%
5409     \global\let\@index@command\@index@command@%
5410     \xdef\@index@parenthesis{(%
5411   }{}%
5412   \IfBeginWith{\@index@command}{)}{%
5413     \StrGobbleLeft{\@index@command}{1}[\@index@command@]%
5414     \global\let\@index@command\@index@command@%
5415     \xdef\@index@parenthesis{)%

```

```

5416 }{}%
5417 }
5418 %

```

\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.

\ledinnothyperpage

\ledinnotemark

```

5419 \newcommand{\ledinnote}[3][1,usedefault]{%
5420   \ifboolexpr{%
5421     test{\ifdefequal{\iftrue}{\ifHy@hyperindex}}%
5422     or%
5423     bool {xindyhyperref@}%
5424   }%
5425   {%
5426     \csuse{#2}{\hyperlink{#1}{\ledinnotemark{#3}}}%
5427   }%
5428   {%
5429     \csuse{#2}{\ledinnotemark{#3}}%
5430   }%
5431 }%
5432 \newcommand{\ledinnothyperpage}[2]{\csuse{#1}{\ledinnotemark{\hyperpage
5433   {#2}}}%
5434 \newcommand{\ledinnotemark}[1]{#1\emph{n}}%
5435 %

```

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

```

5435 \newcommand{\@wredindex}[2][1=\expandonce\jobname,usedefault]{%#1 = the
5436   index name, #2 = the text
5437   \global\let\old@Rlineflag\@Rlineflag%
5438   \gdef\@Rlineflag{}%
5439   \ifl@imakeidx%
5440     \if@edindex@fornote@%
5441       \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
5442       \get@edindex@ledinnote@command%
5443       \expandafter\imki@wrindexentry{#1}{\@index@txt|(\@ledinnote@command
5444         }{\thestartpageline}%
5445       \expandafter\imki@wrindexentry{#1}{\@index@txt|)\@ledinnote@command
5446         }{\theendpageline}%

```

```

5444 \else%
5445 \get@edindex@hyperref{#2}%
5446 \imki@wrindexentry{#1}{\@index@txt\@edindex@hyperref}{\thepageline}%
5447 \fi%
5448 \else%
5449 \if@edindex@fornote%
5450 \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
5451 \get@edindex@ledinnote@command%
5452 \expandafter\protected@write\@indexfile{%
5453 {\string\indexentry{\@index@txt|(\@ledinnote@command){\thestartpageline}
5454 }%
5455 \expandafter\protected@write\@indexfile{%
5456 {\string\indexentry{\@index@txt|)\@ledinnote@command}{\theendpageline}
5457 }%
5458 \else%
5459 \protected@write\@indexfile{%
5460 {\string\indexentry{#2}{\thepageline}
5461 }%
5462 \fi%
5463 \fi%
5464 \endgroup
5465 \global\let\@Rlineflag\old@Rlineflag%
5466 \@esphack%
5467 }
5468 %

```

Need to add the definition of `\edindex` to `\makeindex`, and initialise `\edindex` to do nothing.

```

5469 \pretocmd{\makeindex}{%
5470 \def\edindex{\@bsphack
5471 \doedindexlabel
5472 \begingroup
5473 \@sanitize
5474 \@wredindex}}{}{}
5475 \newcommand{\edindex}[1]{\@bsphack\@esphack}
5476 %

```

`\hyperlinkformat` `\hyperlinkformat` command is to be used to have both a internal hyperlink and a format, when indexing.

```

5477 \newcommand{\hyperlinkformat}[3]{%
5478 \ifstrempy{#1}%
5479 {\hyperlink{#2}{#3}}%
5480 {\csuse{#1}{\hyperlink{#2}{#3}}%
5481 }}
5482 %

```

`\hyperlinkR` `\hyperlinkR` command is to be used to create a internal hyperlink and `\ledRflag`, when indexing.


```

5483 \newcommand{\hyperlinkR}[2]{%
5484   \hyperlink{#1}{#2\@Rlineflag}%
5485 }%
5486
5487 %

```

\hyperlinkformatR \hyperlinkformatR command is to be used to create a internal hyperlink, a format and a \@Rlineflag, when indexing.

```

5488 \newcommand{\hyperlinkformatR}[3]{%
5489   \hyperlinkformat{#1}{#2}{#3\@Rlineflag}%
5490 }%
5491
5492 %

```

\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.

```

5493 \newcommand{\get@edindex@hyperref}[1]{%
5494 %

```

We have to disable temporary spaces to work through a xstring bug (or feature?)

```

5495 \edef\temp@{%
5496   \catcode\ =9 %space need for catcode
5497   \detokenize{#1}%For active character in unicode
5498   \catcode\ =10 % space need for catcode
5499 }%
5500 %

```

Now, we define \@edindex@hyperref if the hyperindex of hyperref is enabled.

```

5501 \ifdefequal{\iftrue}{\ifHy@hyperindex}{%
5502   \IfSubStr{\temp@}{|}%
5503   {\get@index@command#1+%
5504     \ifledRcol%
5505       \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
5506         hyperlinkformatR{\@index@command}%
5507         {\edindexlab\thelabidx}}%
5508     \else%
5509       \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
5510         hyperlinkformat{\@index@command}%
5511         {\edindexlab\thelabidx}}%
5512     \fi%
5513   }%
5514   {\get@index@command#1|+%
5515     \ifledRcol%
5516       \gdef\@edindex@hyperref{|\hyperlinkR{\edindexlab\thelabidx}}%
5517     \else%
5518       \gdef\@edindex@hyperref{|\hyperlink{\edindexlab\thelabidx}}%
5519     \fi%
5520   }%

```

```

5521 }%
5522 %

5523 % If we use both xindy and hyperref, first get the \protect\cs{
index@command} command.
5524 % Then define \protect\cs{@edindex@hyperref} in the form \verb+eledmacXXX+
5525 % \begin{macrocode}
5526 {\ifxindyhyperref%
5527 \IfSubStr{\temp@}{|}%
5528 {\get@index@command#1+}%
5529 {\get@index@command#1|+}%
5530 \gdef\@edindex@hyperref{|eledmac\thelabidx}%
5531 %

```

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`.

```

5532 \IfStrEq{\@index@parenthesis}{(}%
5533 {%
5534 \csxdef{xindyparenthesis@\@index@txt}{\thelabidx}%
5535 \gdef\@edindex@hyperref{| (eledmac\thelabidx}%
5536 }%
5537 {}%
5538 %

```

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`.

```

5539 \IfStrEq{\@index@parenthesis}{)}%
5540 {%
5541 \xdef\@edindex@hyperref{|)eledmac\csuse{xindyparenthesis@\@index@txt}}%
5542 \global\csundef{xindyparenthesis@\@index@txt}%
5543 }%
5544 %

```

Write in the `.xdy` file the attributes of the location.

```

5545 {%
5546 \immediate\write\eledmac@xindy@out{%
5547 (define-attributes ("eledmac\thelabidx"))^^J
5548 \space\space(markup-locref^^J
5549 \eledmacmarkuplocrefdepth^^J
5550 :open "\string\hyperlink%
5551 \ifledRcol R\fi%
5552 {\edindexlab\thelabidx}%
5553 {\ifdefempty{\@index@command}%
5554 }%
5555 {\@backslashchar\@index@command}%
5556 {"^^J

```

```

5557         :close "}}"^^J
5558         :attr "eledmac\thelabidx"^^J
5559     )
5560 }%
5561 }%
5562 %

```

And now, in any other case.

```

5563 \else%
5564     \gdef\@index@txt{#1}%
5565     \gdef\@edindex@hyperref{}%
5566 \fi%
5567 }%
5568 }
5569 %

```

XXVII Verse

The original code is principally Wayne Sullivan's code from `edstanza`. However, the code has been many time modified by Maïeul Rouquette in order to obtain new features and improved compatibility with `reledpar`.

XXVII.1 Hanging symbol management

`\@hangingsymbol` The macro `\@hangingsymbol` is used to insert a symbol on each hanging of verses. It is set by user level macro `\sethangingsymbol`.
`\ifinstanza` For example, in french typographie the symbol is '['. We obtain it by the next code:

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

The `\ifinstanza` boolean is used to be sure that we are in a stanza part.

```

5570 \def\@hangingsymbol{}
5571 \newcommand*{\sethangingsymbol}[1]{%
5572     \gdef\@hangingsymbol{#1}%
5573 }%
5574 \newif\ifinstanza
5575 %

```

`\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.

```

5576 \newif\ifinserthangingsymbol
5577 \newcommand{\inserthangingsymbol}{%
5578     \ifinserthangingsymbol%
5579         \ifinstanza%

```

```

5580 \hangingsymbol%
5581 \fi%
5582 \fi%
5583 }
5584 %

```

XXVII.2 Using & character

\ampersand Within a stanza the `\&` macro is going to be usurped. We need an alias in case an `&` needs to be typeset in a stanza. Define it rather than letting it in case some other package has already defined it.

```

5585 \newcommand*\ampersand{\char`\&}
5586
5587 %

```

XXVII.3 Code category setting

\stanza@count Before we can define the main macros we need to save and reset some category codes.
\stanzaindentbase To save the current values we use `\next` and `\body` from the `\loop` macro.

```

5588 \chardef\body=\catcode`\@
5589 \catcode`\@=11
5590 \chardef\next=\catcode`\&
5591 \catcode`\&=\active
5592
5593 %

```

XXVII.4 Stanza count and indent

A count register is allocated for counting lines in a stanza; also allocated is a dimension register which is used to specify the base value for line indentation; all stanza indentations are multiples of this value. The default value of `\stanzaindentbase` is 20pt.

```

5594 \newcount\stanza@count
5595 \newlength{\stanzaindentbase}
5596 \setlength{\stanzaindentbase}{20pt}
5597
5598 %

```

\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.

```

5599 \def\strip@szacnt#1,#2|{\def\@tempb{#1}\def\@tempa{#2|}}
5600 \newcommand*{\setstanzavalues}[2]{\def\@tempa{#2,,|}%
5601   \stanza@count\z@
5602   \def\next{\expandafter\strip@szacnt\@tempa
5603     \ifx\@tempb\empty\let\next\relax\else
5604     \expandafter\mathchardef\csname #1@number\stanza@count
5605     \@endcsname\@tempb\relax
5606     \advance\stanza@count\@ne\fi\next}%
5607   \next}
5608
5609 %

```

\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).

```

5610 \newcommand*{\setstanzaindents}[1]{\setstanzavalues{sza}{#1}}
5611 \newcommand*{\setstanzapenalties}[1]{\setstanzavalues{szp}{#1}}
5612 %
5613 %

```

\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.

```

5614 \newcounter{stanzaindentsrepetition}
5615 \newcount\stanza@modulo
5616
5617 \newcommand*{\managestanza@modulo}[0]{
5618   \advance\stanza@modulo\@ne
5619   \ifnum\stanza@modulo>\value{stanzaindentsrepetition}
5620     \stanza@modulo\@ne
5621   \fi
5622 }
5623 %

```

\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.

```

5624 \newcommand{\stanzaindent}[1]{%
5625   \hspace{\dimexpr#1\stanzaindentbase-\parindent\relax}%
5626   \ignorespaces%
5627 }%
5628 \WithSuffix\newcommand\stanzaindent*[1]{%
5629   \stanzaindent{#1}%

```

```

5630 \global\advance\stanza@modulo-\@ne%
5631 \ifnum\stanza@modulo=0%
5632 \global\stanza@modulo=\value{stanza\indentsrepetition}%
5633 \fi%
5634 \ignorespaces%
5635 }%
5636 %

```

XXVII.5 Numbering stanza

Here, macro for numbering stanza. First, the stanza counter.

```

\thestanza\newcounter{stanza}
5638 \renewcommand{\thestanza}{%
5639 \textbf{\arabic{stanza}}}%
5640 }
5641 %

```

`\ifnumberstanza` Then, macro to activate automatically numbering of stanza.

```

5642 \newif\ifnumberstanza%
5643 %

```

`\@insertstanzanumber` Now, macro called at the first line of verse of a stanza.

```

5644 \newcommand{\@insertstanzanumber}[0]{%
5645 \ifnumberstanza%
5646 \ifl@dpairing%
5647 \ifledRcol%
5648 \stanzanumwrapper{\thestanzaR}%
5649 \else%
5650 \stanzanumwrapper{\thestanzaL}%
5651 \fi%
5652 \else%
5653 \stanzanumwrapper{\thestanza}%
5654 \fi%
5655 \setline{1}%
5656 \fi%
5657 }%
5658 %

```

`\@advancestanzanumber` Also a command to advance the counter of stanza.

```

5659 \newcommand{\@advancestanzanumber}[0]{%
5660 \ifnumberstanza%
5661 \ifl@dpairing%
5662 \ifledRcol%
5663 \addtocounter{stanzaR}{1}%
5664 \else%

```

```

5665     \addtocounter{stanzaL}{1}%
5666     \fi%
5667   \else%
5668     \addtocounter{stanza}{1}%
5669     \fi%
5670   \fi%
5671 }%
5672 %

```

`\stanzanumwrapper` And finally, the wrapper for stanza number

```

5673 \newcommand{\stanzanumwrapper}[1]{%
5674   \flagstanza{#1}%
5675 }%
5676 %

```

XXVII.6 Stanza number in note

Here, the command called when printing stanza number in notes.

```

5677 \newcommand{\printstanza}[0]{%
5678   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
5679     l@dprintingcolumns}}{%
5680     \ifledRcol{%
5681       \thestanzaR%
5682     }%
5683   }%
5684   \thestanzaL%
5685 }%
5686 %
5687 }
5688 %

```

XXVII.7 Main work

`\stanza@line` Now we arrive at the main works. `\stanza@line` sets the indentation for the line and starts a numbered paragraph—each line is treated as a paragraph. `\stanza@hang` sets the hanging indentation to be used if the stanza line requires more than one print line.

If it is known that each stanza line will fit on one print line, it is advisable to set the hanging indentation to zero. `\sza@penalty` places the specified penalty following each stanza line. By default, this facility is turned off so that no penalty is included. However, the user may initiate these penalties to indicate good and bad places in the stanza for page breaking.

```

5689 \newcommand{\stanza@line}[1][1]{
5690   \ifnum\value{stanzaindentrepetition}=0
5691     \parindent=\csname sza@number\stanza@count
5692       @\endcsname\stanzaindentbase

```

```

5693 \else
5694 \parindent=\csname sza@\number\stanza@modulo
5695 @\endcsname\stanzaindentbase
5696 \managestanza@modulo
5697 \fi
5698 \pstart[#1]\stanza@hang\ignorespaces}
5699 \xdef\stanza@hang{\noexpand\leavevmode\noexpand\startlock
5700 \hangindent\expandafter
5701 \noexpand\csname sza@0@\endcsname\stanzaindentbase
5702 \hangafter\@ne}
5703 \def\sza@penalty{\count@\csname szp@\number\stanza@count @\endcsname
5704 \ifnum\count@>\@M\advance\count@-\@M\penalty-\else
5705 \penalty\fi\count@}
5706 %

```

`\@startstanza` Now we have the components of the `\stanza` macro, which appears at the start of a group of lines. This macro initializes the count and checks to see if hanging indentation and penalties are to be included. Hanging indentation suspends the line count, so that the enumeration is by verse line rather than by print line. If the print line count is desired, invoke `\let\startlock\relax` and do the same for `\endlock`. Here and above we have used `\xdef` to make the stored macros take up a bit less space, but it also makes them more obscure to the reader. Lines of the stanza are delimited by ampersands `&`. The last line of the stanza must end with `\&`.

```

5707 \xdef\@startstanza[#1]{%
5708 \noexpand\instanzatrue\expandafter
5709 \begingroup%
5710 \catcode`\noexpand\&\active%
5711 \global\stanza@count\@ne\stanza@modulo\@ne
5712 \noexpand\ifnum\expandafter\noexpand
5713 \csname sza@0@\endcsname=\z@\let\noexpand\stanza@hang\relax
5714 \let\noexpand\endlock\relax\noexpand\else\interlinepenalty
5715 \@M\rightskip\z@ plus 1fil\relax\noexpand\fi\noexpand\ifnum
5716 \expandafter\noexpand\csname szp@0@\endcsname=\z@
5717 \let\noexpand\sza@penalty\relax\noexpand\fi%
5718 \def\noexpand&{%
5719 \noexpand\newverse[] []}%
5720 \def\noexpand\&{\noexpand\@stopstanza}%
5721 \noexpand\@advancestanza@number%
5722 \noexpand\stanza@line[#1]\noexpand\@insertstanza@number}
5723
5724 \newcommandx{\stanza}[1][1,usedefault]{\@startstanza[#1]}
5725
5726 \newcommandx{\@stopstanza}[1][1,usedefault]{%
5727 \unskip%
5728 \endlock%
5729 \pend[#1]%
5730 \endgroup%
5731 \instanzafalse%

```



```

5732 }
5733
5734 \newcommand*{\newverse}[2][1,2,usedefault]{%
5735   \unskip%
5736   \endlock\pend[#1]\sza@penalty\global%
5737   \advance\stanza@count\@ne\stanza@line[#2]%
5738 }
5739
5740 %

```

\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`.

```

5741 \newcommand*{\flagstanza}[2][\stanzaindentbase]{%
5742   \hskip -#1\llap{#2}\hskip #1\ignorespaces}
5743
5744 %

```

XXVII.8 Restore catcode and penalties

The ampersand & is used to mark the end of each stanza line, except the last, which is marked with `\&`. This means that `\halign` may not be used directly within a stanza line. This does not affect macros involving alignments defined outside `\stanza \&`. Since these macros usurp the control sequence `\&`, the replacement `\ampersand` is defined to be used if this symbol is needed in a stanza. Also we reset the modified category codes and initialize the penalty default.

```

5745 \catcode`\&=\next
5746 \catcode`\@=\body
5747 \setstanzavalues{szp}{0}
5748
5749 %

```

XXVIII Arrays and tables

XXVIII.1 Preamble: macro as environment

The following is borrowed, and renamed, from the `amsmath` package. See also the CTT thread ‘`eeq` and `amstex`’, 1995/08/31, started by Keith Reckdahl and ended definitively by David M. Jones.

Several of the `[math]` macros scan their body twice. This means we must collect all text in the body of an environment form before calling the macro.

\@emptytoks This is actually defined in the `amsgen` package.

```

5750 \newtoks\@emptytoks
5751
5752 %

```

The rest is from amsmath.

\l@denbody A token register to contain the body.

```
5753 \newtoks\l@denbody
5754
5755 %
```

\addtol@denbody \addtol@denbody{arg} adds arg to the token register \l@denbody.

```
5756 \newcommand{\addtol@denbody}[1]{%
5757   \global\l@denbody\expandafter\the\l@denbody#1}
5758
5759 %
```

\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 env#1{...} as a macro that processes #1, then the environment form, \begin{env} would call \l@dcollect@body\env.

```
5760 \newcommand{\l@dcollect@body}[1]{%
5761   \l@denbody{\expandafter#1\expandafter\the\l@denbody}}%
5762   \edef\processl@denbody{\the\l@denbody\noexpand\end{\@currenvir}}%
5763   \l@denbody\emptytoks \def\l@dbegin@stack{b}%
5764   \begingroup
5765     \expandafter\let\csname\@currenvir\endcsname\l@dcollect@@body
5766     \edef\processl@denbody{\expandafter\noexpand\csname\@currenvir\endcsname}%
5767     \processl@denbody%
5768   }%
5769
5770 %
```

\l@dpush@begins When adding a piece of the current environment's contents to \l@denbody, we scan it to check for additional \begin tokens, and add a 'b' to the stack for any that we find.

```
5771 \def\l@dpush@begins#1\begin#2{%
5772   \ifx\end#2\else b\expandafter\l@dpush@begins\fi}
5773
5774 %
```

\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.

```

5775 \def\l@dcollect@@body#1\end#2{%
5776   \edef\l@dbegin@stack{\l@dpush@begins#1\begin\end
5777     \expandafter\@gobble\l@dbegin@stack}%
5778   \ifx\@empty\l@dbegin@stack
5779     \endgroup
5780     \@checkend{#2}%
5781     \addtol@denbody{#1}%
5782   \else
5783     \addtol@denbody{#1\end{#2}}%
5784   \fi
5785   \processl@denbody % A little tricky! Note the grouping
5786 }
5787
5788 %

```

There was a question on CTT about how to use `\collect@body` for a macro taking an argument. The following is part of that thread.

From: Heiko Oberdiek <oberdiek@uni-freiburg.de>
 Newsgroups: comp.text.tex
 Subject: Re: Using `\collect@body` with commands that take >1 argument
 Date: Fri, 08 Aug 2003 09:03:20 +0200

eed132@psu.edu (Evan) wrote:
 > I'm trying to make a new Latex environment that acts like the
 > `\colorbox` command that is part of the color package. I looked through
 > the FAQ and ran across this bit about using the `\collect@body` command
 > that is part of AMSLaTeX:
 > <http://www.tex.ac.uk/cgi-bin/texfaq2html?label=cmdasenv>
 >
 > It almost works. If I do something like the following:
 > `\newcommand{\redbox}[1]{\colorbox{red}{#1}}`
 >
 > `\makeatletter`
 > `\newenvironment{redbox}{\collect@body \redbox}{}`

You will get an error message: Command `\redbox` already defined.
 Thus you must rename either the command `\redbox` or the environment
 name.

```

> \begin{coloredbox}{blue}
>   Yadda yadda yadda... this is on a blue background...
> \end{coloredbox}
> and can't figure out how to make the \collect@body take this.

> \collect@body \colorbox{red}
> \collect@body {\colorbox{red}}

```

The argument of `\collect@body` has to be one token exactly.

```

\documentclass{article}
\usepackage{color}
\usepackage{amsmath}

\newcommand{\redbox}[1]{\colorbox{red}{#1}}
\makeatletter
\newenvironment{coloredbox}[1]{%
  \def\next@{\colorbox{#1}}%
  \collect@body\next@
}{%

% ignore spaces at begin and end of environment
\newenvironment{coloredboxII}[1]{%
  \def\next@{\mycoloredbox{#1}}%
  \collect@body\next@
}{%}
\newcommand{\mycoloredbox}[2]{%
  \colorbox{#1}{\ignorespaces#2\unskip}%
}

% support of optional color model argument
\newcommand\coloredboxIII\endcsname{}
\def\coloredboxIII#1#{%
  \@coloredboxIII{#1}%
}
\def\@coloredboxIII#1#2{%
  \def\next@{\mycoloredboxIII{#1}{#2}}%
  \collect@body\next@
}
\newcommand{\mycoloredboxIII}[3]{%
  \colorbox#1{#2}{\ignorespaces#3\unskip}%
}

\makeatother

\begin{document}
  Black text before
  \begin{coloredbox}{blue}
    Hello World
  \end{coloredbox}
  Black text after

  Black text before
  \begin{coloredboxII}{blue}
    Hello World
  \end{coloredboxII}
  Black text after

  Black text before
  \begin{coloredboxIII}[rgb]{0,0,1}

```

```

Hello World
\end{coloredboxIII}
Black text after

\end{document}

Yours sincerely
Heiko <oberdiek@uni-freiburg.de>

```

XXVIII.2 Tabular environments

This is based on the work by Herbert Breger in developing `tabmac.tex`.

The original `tabmac.tex` file was void of comments or any explanatory text other than the above notice. The algorithm is Breger's. Peter Wilson have made some cosmetic changes to the original code and reimplemented some things so they are more LaTeX-like. All the commentary are from Peter Wilson, as are any mistake or errors.

However, Maïeul Rouquette has modified code in order to add new features of `eledmac` and `reledmac`.

XXVIII.2.1 Disabling and restoring commands

`\l@dtabnoexpands` More no expansion for critical and familiar footnotes in tabular environment.

```

5789 \newcommand*{\l@dtabnoexpands}{%
5790   \let\rtab=0%
5791   \let\ctab=0%
5792   \let\ltab=0%
5793   \let\rtabtext=0%
5794   \let\ltabtext=0%
5795   \let\ctabtext=0%
5796   \let\edbeforetab=0%
5797   \let\edaftertab=0%
5798   \let\edatleft=0%
5799   \let\edatright=0%
5800   \let\edvertline=0%
5801   \let\edvertdots=0%
5802   \let\edrowfill=0%
5803 }
5804
5805 %

```

`\disable@familiarnotes` Macros to disable and restore familiar notes, to prevent them from printing multiple times in `edtabularx` and `edarrayx` environments.

```

5806 \newcommand{\disable@familiarnotes}{%
5807   \unless\ifnofamiliar%
5808     \def\do##1{%
5809       \csletcs{footnote@##1}{footnote##1}%

```

```

5810     \expandafter\renewcommand \csname footnote##1\endcsname[1]{%
5811         \protected@csxdef{theftmark##1}{\csuse{thefootnote##1}}%
5812         \csuse{@footnotemark##1}%
5813     }%
5814 }%
5815 \dolistloop{\@series}%
5816 \fi%
5817 }%
5818 \newcommand{\restore@familiarnotes}{%
5819     \unless\ifnofamiliar%
5820     \def\do##1{%
5821         \csletcs{footnote##1}{footnote@@##1}%
5822     }%
5823     \dolistloop{\@series}%
5824     \fi%
5825 }%
5826
5827 %

```

`\disable@sidenotes` The same, for side notes.

```

\restore@sidenotes
5828 \newcommand{\disable@sidenotes}{%
5829     \let\@@ledrightnote\ledrightnote%
5830     \let\@@ledleftnote\ledleftnote%
5831     \let\@@ledsidenote\ledsidenote%
5832     \let\ledrightnote@gobble%
5833     \let\ledleftnote@gobble%
5834     \let\ledsidenote@gobble%
5835 }%
5836 \newcommand{\restore@sidenotes}{%
5837     \let\ledrightnote\@@ledrightnote%
5838     \let\ledleftnote\@@ledleftnote%
5839     \let\ledsidenote\@@ledsidenote%
5840 }%
5841 %

```

`\disable@notes` Disable/restore side and familiar notes.

```

\restore@notes
5842 \newcommand{\disable@notes}{%
5843     \disable@sidenotes%
5844     \disable@familiarnotes%
5845 }%
5846 \newcommand{\restore@notes}{%
5847     \restore@sidenotes%
5848     \restore@familiarnotes%
5849 }%
5850 %

```

`\EDTEXT` We need to be able to modify the `\edtext` macros and also restore their original definitions.
`\xedtext`

```

5851 \let\EDTEXT=\edtext
5852 \newcommand{\xedtext}[2]{\EDTEXT{#1}{#2}}
5853 %

```

\EDLABEL We need to be able to modify and restore the `\edlabel` macro.

```

\xedlabel
5854 \let\EDLABEL=\edlabel
5855 \newcommand*{\xedlabel}[1]{\EDLABEL{#1}}
5856 %

```

\EDINDEX Macros supporting modification and restoration of `\edindex`.

```

\xedindex
\nulledindex
5857 \let\EDINDEX=\edindex
5858 \newcommand{\xedindex}{\@bsphack%
5859   \ifnextchar [{\l@d@index}{\l@d@index[\jobname]}}
5860 \newcommand{\nulledindex}[2][\jobname]{\@bsphack\@esphack}
5861
5862 %

```

\@line@@num Macro supporting restoration of `\linenum`.

```

5863 \let\@line@@num=\linenum
5864 %

```

\l@dgobblearg `\l@dgobbleoptarg[⟨arg⟩]{⟨arg⟩}` replaces these two arguments (first is optional) by `\relax`.

```

5865 \newcommand*{\l@dgobbleoptarg}[2][\relax]{%
5866
5867 %

```

\Relax `\let\Relax=\relax`

\NEXT `\let\NEXT=\next`

```

5870
5871 %

```

\l@dmodforedtext Modify and restore various macros for when `\edtext` is used.

```

\l@drestoreforedtext
5872 \newcommand{\l@dmodforedtext}{%
5873   \let\edtext\relax
5874   \def\do##1{\global\csletcs{##1footnote}{\l@dgobbleoptarg}}%
5875   \dolistloop{\@series}%
5876   \let\edindex\nulledindex
5877   \let\linenum\@gobble}
5878 \newcommand{\l@drestoreforedtext}{%
5879   \def\do##1{\global\csletcs{##1footnote}{##1@footnote}}
5880   \dolistloop{\@series}%
5881   \let\edindex\xedindex}
5882 %

```

`\l@dnnullfills` Nullify and restore some column fillers, etc.

```
\l@drestorefills
5883 \newcommand{\l@dnnullfills}{%
5884   \def\edlabel##1{%
5885     \def\edrowfill##1##2##3{%
5886   }
5887 \newcommand{\l@drestorefills}{%
5888   \def\edrowfill##1##2##3{\@EDROWFILL@{##1}{##2}{##3}}%
5889 }
5890
5891 %
```

`\letsforverteilen` Gathers some lets and other code that is common to the **verteilen** macros.

```
5892 \newcommand{\letsforverteilen}{%
5893   \let\edtext\xedtext
5894   \let\edindex\xedindex
5895   \def\do##1{\global\csletcs{##1footnote}{##1@@footnote}}
5896   \dolistloop{\@series}%
5897   \let\linenum\@line@num
5898   \hilfsskip=\l@dcwidth%
5899   \advance\hilfsskip by -\wd\hilfsbox
5900   \def\edlabel##1{\xedlabel{##1}}
5901
5902 %
```

`\disablel@dtabfeet` Declarations for using or using `\edtext` inside tabulars. The default at this point is for `\edtext`.

```
5903 \newcommand\disablel@dtabfeet{\l@dmodforedtext}%
5904 \newcommand\enablel@dtabfeet{\l@drestoreforedtext}%
5905 %
```

XXVIII.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.

```
5906 \newcount\l@dampcount
5907 \l@dampcount=1\relax
5908 \newcount\l@dcolcount
5909 \l@dcolcount=0\relax
5910
5911 %
```

`\hilfsbox` Some (temporary) helper items.

```
\hilfsskip
5912 \newbox\hilfsbox
\Hilfsbox
5913 \newskip\hilfsskip
\hilfscount
5914 \newbox\Hilfsbox
```



```

5915 \newcount\hilfscount
5916
5917 %

```

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).

```

5918 \newdimen\dcoli
5919 \newdimen\dcolii
5920 \newdimen\dcoliii
5921 \newdimen\dcoliv
5922 \newdimen\dcolv
5923 \newdimen\dcolvi
5924 \newdimen\dcolvii
5925 \newdimen\dcolviii
5926 \newdimen\dcolix
5927 \newdimen\dcolx
5928 \newdimen\dcolxi
5929 \newdimen\dcolxii
5930 \newdimen\dcolxiii
5931 \newdimen\dcolxiv
5932 \newdimen\dcolxv
5933 \newdimen\dcolxvi
5934 \newdimen\dcolxvii
5935 \newdimen\dcolxviii
5936 \newdimen\dcolxix
5937 \newdimen\dcolxx
5938 \newdimen\dcolxxi
5939 \newdimen\dcolxxii
5940 \newdimen\dcolxxiii
5941 \newdimen\dcolxxiv
5942 \newdimen\dcolxxv
5943 \newdimen\dcolxxvi
5944 \newdimen\dcolxxvii
5945 \newdimen\dcolxxviii
5946 \newdimen\dcolxxix
5947 \newdimen\dcolxxx
5948 \newdimen\dcolerr % added for error handling
5949
5950 %

```

`\l@dcolwidth` This is a cunning way of storing the columnwidths indexed by the column number `\l@dcolcount`, like an array. (was `\Dimenzuordnung`)

```

5951 \newcommand{\l@dcolwidth}{\ifcase \the\l@dcolcount \dcoli %???
5952 \or \dcoli \or \dcolii \or \dcoliii
5953 \or \dcoliv \or \dcolv \or \dcolvi
5954 \or \dcolvii \or \dcolviii \or \dcolix \or \dcolx
5955 \or \dcolxi \or \dcolxii \or \dcolxiii
5956 \or \dcolxiv \or \dcolxv \or \dcolxvi

```

```

5957 \or \dcolxvii \or \dcolxviii \or \dcolxix \or \dcolxx
5958 \or \dcolxxi \or \dcolxxii \or \dcolxxiii
5959 \or \dcolxxiv \or \dcolxxv \or \dcolxxvi
5960 \or \dcolxxvii \or \dcolxxviii \or \dcolxxix \or \dcolxxx
5961 \else \dcolerr \fi}
5962
5963 %

```

\step1@dcolcount This increments the column counter, and issues an error message if it is too large.

```

5964 \newcommand*{\step1@dcolcount}{\advance\l@dcolcount\@ne
5965 \ifnum\l@dcolcount>30\relax
5966 \led@err@TooManyColumns
5967 \fi}
5968
5969 %

```

\l@dsetmaxcolwidth Sets the column width to the maximum value seen so far.

```

5970 \newcommand{\l@dsetmaxcolwidth}{%
5971 \ifdim\l@dcolwidth < \wd\hilfsbox
5972 \l@dcolwidth = \wd\hilfsbox
5973 \else \relax \fi}
5974
5975 %

```

\measuremcell Measure (recursively) the width required for a math cell.

```

5976 \def\measuremcell #1{%
5977 \ifx #1\ \ifnum\l@dcolcount=0\let\NEXT\relax%
5978 \else\l@dcheckcols%
5979 \l@dcolcount=0%
5980 \let\NEXT\measuremcell%
5981 \fi%
5982 \else\setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
5983 \step1@dcolcount%
5984 \l@dsetmaxcolwidth%
5985 \let\NEXT\measuremcell%
5986 \fi\NEXT}
5987
5988 %

```

\measuretcell Measure (recursively) the width required for a text cell.

```

5989 \def\measuretcell #1{%
5990 \ifx #1\ \ifnum\l@dcolcount=0\let\NEXT\relax%
5991 \else\l@dcheckcols%
5992 \l@dcolcount=0%
5993 \let\NEXT\measuretcell%
5994 \fi%

```

```

5995 \else\setbox\hilfsbox=\hbox{#1}%
5996 \step1@dcolcount%
5997 \l@dsetmaxcolwidth%
5998 \let\NEXT\measuretcell%
5999 \fi\NEXT}
6000
6001 %

```

\measuremrow Measure (recursively) the width required for a math row.

```

6002 \def\measuremrow #1\{%
6003 \ifx #1&\let\NEXT\relax%
6004 \else\measuremcell #1&\&\&%
6005 \let\NEXT\measuremrow%
6006 \fi\NEXT}
6007 %

```

\measuretrow Measure (recursively) the width required for a text row.

```

6008 \def\measuretrow #1\{%
6009 \ifx #1&\let\NEXT\relax%
6010 \else\measuretcell #1&\&\&%
6011 \let\NEXT\measuretrow%
6012 \fi\NEXT}
6013
6014 %

```

\edtabcolsep The length `\edtabcolsep` controls the distance between columns.

```

6015 \newskip\edtabcolsep
6016 \global\edtabcolsep=10pt
6017
6018 %

```

\variab₁₉ \newcommand{\variab}{\relax}

```

6020
6021 %

```

\l@dcheckcols Check that the number of columns is consistent.

```

6022 \newcommand*{\l@dcheckcols}{%
6023 \ifnum\l@dcolcount=1\relax
6024 \else
6025 \ifnum\l@dampcount=1\relax
6026 \else
6027 \ifnum\l@dcolcount=\l@dampcount\relax
6028 \else
6029 \l@d@err@UnequalColumns

```

```

6030     \fi
6031     \fi
6032     \l@dampcount=\l@dcolcount
6033 \fi}
6034
6035 %

```

\edfilldimen A length.

```

6036 \newdimen\edfilldimen
6037 \edfilldimen=0pt
6038
6039 %

```

\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.

\theadcolcount

```

6040 \newcounter{addcolcount}
6041 \renewcommand{\theadcolcount}{\roman{addcolcount}}
6042 %

```

XXVIII.2.3 Tabular typesetting

\setmcellright Typeset (recursively) cells of display math right justified.

```

6043 \def\setmcellright #1{\def\edlabel##1{}}%
6044     \let\edindex\nulledindex
6045     \ifx #1\\ \ifnum\l@dcolcount=0%\removelastskip
6046         \let\Next\relax%
6047     \else\l@dcolcount=0%
6048         \let\Next=\setmcellright%
6049     \fi%
6050 \else%
6051     \disablel@dtabfeet%
6052     \stepl@dcolcount%
6053     \disable@notes%
6054     \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6055     \restore@notes%
6056     \letsforverteilen%
6057     \hskip\hilfsskip$\displaystyle{#1}$%
6058     \hskip\edtabcolsep%
6059     \let\Next=\setmcellright%
6060 \fi\Next}
6061
6062 %

```

\settcellright Typeset (recursively) cells of text right justified.

```

6063 \def\settcclright #1{\def\edlabel##1{}}%
6064 \let\edindex\nulledindex
6065 \ifx #1\ \ifnum\l@dcclcount=0%\removelastskip
6066 \let\Next\relax%
6067 \else\l@dcclcount=0%
6068 \let\Next=\settcclright%
6069 \fi%
6070 \else%
6071 \disablel@dtabfeet%
6072 \step1@dcclcount%
6073 \disable@notes%
6074 \setbox\hilfsbox=\hbox{#1}%
6075 \restore@notes%
6076 \letsforverteilen%
6077 \hskip\hilfsskip#1%
6078 \hskip\edtabcolsep%
6079 \let\Next=\settcclright%
6080 \fi\Next}
6081 %

```

\setmclleft Typeset (recursively) cells of display math left justified.

```

6082 \def\setmclleft #1{\def\edlabel##1{}}%
6083 \let\edindex\nulledindex
6084 \ifx #1\ \ifnum\l@dcclcount=0 \let\Next\relax%
6085 \else\l@dcclcount=0%
6086 \let\Next=\setmclleft%
6087 \fi%
6088 \else \disablel@dtabfeet%
6089 \step1@dcclcount%
6090 \disable@notes%
6091 \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6092 \restore@notes%
6093 \letsforverteilen%
6094 $\displaystyle{#1}$\hskip\hilfsskip\hskip\edtabcolsep%
6095 \let\Next=\setmclleft%
6096 \fi\Next}
6097 %
6098 %

```

\settcclleft Typeset (recursively) cells of text left justified.

```

6099 \def\settcclleft #1{\def\edlabel##1{}}%
6100 \let\edindex\nulledindex
6101 \ifx #1\ \ifnum\l@dcclcount=0 \let\Next\relax%
6102 \else\l@dcclcount=0%
6103 \let\Next=\settcclleft%
6104 \fi%
6105 \else \disablel@dtabfeet%
6106 \step1@dcclcount%

```

```

6107         \disable@notes%
6108         \setbox\hilfsbox=\hbox{#1}%
6109         \restore@notes%
6110         \letsforverteilen%
6111         #1\hskip\hilfsskip\hskip\edtabcolsep%
6112         \let\Next=\settcclleft%
6113     \fi\Next}
6114 %

```

\setmcellcenter Typeset (recursively) cells of display math centered.

```

6115 \def\setmcellcenter #1&{\def\edlabel##1{}}%
6116     \let\edindex\nulledindex
6117     \ifx #1\\ \ifnum\l@dcolcount=0\let\Next\relax%
6118         \else\l@dcolcount=0%
6119             \let\Next=\setmcellcenter%
6120         \fi%
6121     \else \disablel@dtabfeet%
6122         \stepl@dcolcount%
6123         \disable@notes%
6124         \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6125         \restore@notes%
6126         \letsforverteilen%
6127         \hskip 0.5\hilfsskip$\displaystyle{#1}$\hskip0.5\hilfsskip%
6128         \hskip\edtabcolsep%
6129         \let\Next=\setmcellcenter%
6130     \fi\Next}
6131
6132 %

```

\settcclcenter Typeset (recursively) cells of text centered.

```

6133 \def\settcclcenter #1&{\def\edlabel##1{}}%
6134     \let\edindex\nulledindex
6135     \ifx #1\\ \ifnum\l@dcolcount=0 \let\Next\relax%
6136         \else\l@dcolcount=0%
6137             \let\Next=\settcclcenter%
6138         \fi%
6139     \else \disablel@dtabfeet%
6140         \stepl@dcolcount%
6141         \disable@notes%
6142         \setbox\hilfsbox=\hbox{#1}%
6143         \restore@notes%
6144         \letsforverteilen%
6145         \hskip 0.5\hilfsskip #1\hskip 0.5\hilfsskip%
6146         \hskip\edtabcolsep%
6147         \let\Next=\settcclcenter%
6148     \fi\Next}
6149
6150 %

```

```
\NEXT 51 \let\NEXT=\relax
```

```
6152  
6153 %
```

\setmrowright Typeset (recursively) rows of right justified math.

```
6154 \def\setmrowright #1\{\%  
6155   \ifx #1& \let\NEXT\relax  
6156   \else \centerline{\setmcellright #1&\\&\\&}  
6157         \let\NEXT=\setmrowright  
6158   \fi\NEXT}  
6159 %
```

\settroright Typeset (recursively) rows of right justified text.

```
6160 \def\settroright #1\{\%  
6161   \ifx #1& \let\NEXT\relax  
6162   \else \centerline{\settcclright #1&\\&\\&}  
6163         \let\NEXT=\settroright  
6164   \fi\NEXT}  
6165  
6166 %
```

\setmrowleft Typeset (recursively) rows of left justified math.

```
6167 \def\setmrowleft #1\{\%  
6168   \ifx #1& \let\NEXT\relax  
6169   \else \centerline{\setmcellleft #1&\\&\\&}  
6170         \let\NEXT=\setmrowleft  
6171   \fi\NEXT}  
6172 %
```

\settrorleft Typeset (recursively) rows of left justified text.

```
6173 \def\settrorleft #1\{\%  
6174   \ifx #1& \let\NEXT\relax  
6175   \else \centerline{\settcclleft #1&\\&\\&}  
6176         \let\NEXT=\settrorleft  
6177   \fi\NEXT}  
6178  
6179 %
```

\setmrowcenter Typeset (recursively) rows of centered math.

```
6180 \def\setmrowcenter #1\{\%  
6181   \ifx #1& \let\NEXT\relax%  
6182   \else \centerline{\setmcellcenter #1&\\&\\&}  
6183         \let\NEXT=\setmrowcenter  
6184   \fi\NEXT}  
6185 %
```

\setrowcenter Typeset (recursively) rows of centered text.

```

6186 \def\setrowcenter #1\\{%
6187   \ifx #1& \let\NEXT\relax
6188   \else \centerline{\setcellcenter #1&\\&\\&}
6189   \let\NEXT=\setrowcenter
6190   \fi\NEXT}
6191
6192 %

```

```

\newcommand{\nullsetzen}%
6194   \step1@dcolcount%
6195   \l@dcolwidth=0pt%
6196   \ifnum\l@dcolcount=30\let\NEXT\relax%
6197   \l@dcolcount=0\relax
6198   \else\let\NEXT\nullsetzen%
6199   \fi\NEXT}
6200
6201 %

```

\edatleft $\edatleft[\langle math \rangle]{\langle symbol \rangle}{\langle len \rangle}$. Left $\langle symbol \rangle$, $2\langle len \rangle$ high with prepended $\langle math \rangle$ vertically centered.

```

6202 \newcommand{\edatleft}[3][\@empty]{%
6203   \ifx#1\@empty
6204     \vbox to 10pt{\vss\hbox{\$ \left#2\vrule width0pt height #3
6205       depth 0pt \right. \$\hss}\vfil}
6206   \else
6207     \vbox to 4pt{\vss\hbox{\$#1\left#2\vrule width0pt height #3
6208       depth 0pt \right. \$}\vfil}
6209   \fi}
6210 %

```

\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.

```

6211 \newcommand{\edatright}[3][\@empty]{%
6212   \ifx#1\@empty
6213     \vbox to 10pt{\vss\hbox{\$ \left.\vrule width0pt height #3
6214       depth 0pt \right#2 \$\hss}\vfil}
6215   \else
6216     \vbox to 4pt{\vss\hbox{\$ \left.\vrule width0pt height #3
6217       depth 0pt \right#2 #1 \$}\vfil}
6218   \fi}
6219
6220 %

```

\edvertline $\edvertline{\langle len \rangle}$ vertical line $\langle len \rangle$ high.


```

6221 \newcommand{\edvertline}[1]{\vbox to 8pt{\vss\hbox{\vrule height #1}\vfil}}
6222
6223 %

```

\edvertdots `\edvertdots{<len>}` vertical dotted line <len> high.

```

6224 \newcommand{\edvertdots}[1]{\vbox to 1pt{\vss\vbox to #1%
6225     {\cleaders\hbox{$\math\hbox{.}\vbox to 0.5em{ }$}\vfil}}}
6226
6227 %

```

\l@dtabaddcols `\l@dtabaddcols{<startcol>}{<endcol>}` adds the widths of the columns <startcol> through <endcol> to `\edfilldimen`. It is a \TeX style reimplementation of the original `\@add@`.

```

6228 \newcommand{\l@dtabaddcols}[2]{%
6229     \l@dcheckstartend{#1}{#2}%
6230     \ifl@dstartendok
6231     \setcounter{addcolcount}{#1}%
6232     \@whilenum \value{addcolcount}<#2\relax \do
6233     {\advance\edfilldimen by \the \csname dcol\theaddcolcount\endcsname
6234     \advance\edfilldimen by \edtabcolsep
6235     \stepcounter{addcolcount}}%
6236     \advance\edfilldimen by \the \csname dcol\theaddcolcount\endcsname
6237     \fi
6238 }
6239
6240 %

```

\ifl@dstartendok `\l@dcheckstartend{<startcol>}{<endcol>}` checks that the values of <startcol> and <endcol> are sensible. If they are then `\ifl@dstartendok` is set TRUE, otherwise it is set FALSE.

```

6241 \newif\ifl@dstartendok
6242 \newcommand{\l@dcheckstartend}[2]{%
6243     \l@dstartendoktrue
6244     \ifnum #1<\@ne
6245     \l@dstartendokfalse
6246     \led@err@LowStartColumn
6247     \fi
6248     \ifnum #2>30\relax
6249     \l@dstartendokfalse
6250     \led@err@HighEndColumn
6251     \fi
6252     \ifnum #1>#2\relax
6253     \l@dstartendokfalse
6254     \led@err@ReverseColumns
6255     \fi
6256 }
6257
6258 %

```

`\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.

```
6259 \newcommand*{\edrowfill}[3]{%
6260   \l@dtabaddcols{#1}{#2}%
6261   \hb@xt@ \the\l@dcolwidth{\hb@xt@ \the\edfilldimen{#3}\hss}}
6262 \let\@edrowfill=\edrowfill
6263 \def\@EDROWFILL@#1#2#3{\@edrowfill@{#1}{#2}{#3}}
6264
6265 %
```

`\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`.

```
6266 \newcommand{\leftltab}[1]{%
6267   \hb@xt@ \z@{\vbox{\edtabindent%
6268     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
6269
6270 %
```

`\leftrtab` `\leftrtab{<text>}{<math>}` for `\edbeforetab` in `\rtab`.

```
6271 \newcommand{\leftrtab}[2]{%
6272   #2\hb@xt@ \z@{\vbox{\edtabindent%
6273     \advance\Hilfsskip by\dcoli%
6274     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
6275
6276 %
```

`\leftctab` `\leftctab{<text>}{<math>}` for `\edbeforetab` in `\ctab`.

```
6277 \newcommand{\leftctab}[2]{%
6278   \hb@xt@ \z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6279     \advance\Hilfsskip by 0.5\dcoli%
6280     \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
6281       \disablel@dtabfeet$\displaystyle{#2}$}%
6282     \advance\Hilfsskip by -0.5\wd\hilfsbox%
6283     \moveleft\Hilfsskip\hbox{\ #1}}\hss}%
6284   #2}
6285
6286 %
```

`\rightctab` `\rightctab{<math>}{<text>}` for `\edaftertab` in `\ctab`.

```

6287 \newcommand{\rightctab}[2]{%
6288     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6289         \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
6290         #1\hb@xt@{\z@\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6291         \advance\Hilfsskip by 0.5\l@dcolwidth%
6292         \advance\Hilfsskip by -\wd\hilfsbox%
6293         \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6294             \disablel@dtabfeet$\displaystyle{#1}$}%
6295             \advance\Hilfsskip by -0.5\wd\hilfsbox%
6296             \advance\Hilfsskip by \edtabcolsep%
6297             \moveright\Hilfsskip\hbox{ #2}}\hss}%
6298         }
6299     }
6300 %

```

\rightltab `\rightltab{<math><math>}{<text>}` for `\edaftertab` in `\ltab`.

```

6301 \newcommand{\rightltab}[2]{%
6302     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6303         \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
6304         #1\hb@xt@{\z@\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6305         \advance\Hilfsskip by \l@dcolwidth%
6306         \advance\Hilfsskip by -\wd\hilfsbox%
6307         \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6308             \disablel@dtabfeet$\displaystyle{#1}$}%
6309             \advance\Hilfsskip by -\wd\hilfsbox%
6310             \advance\Hilfsskip by \edtabcolsep%
6311             \moveright\Hilfsskip\hbox{ #2}}\hss}%
6312         }
6313     }
6314 %

```

\rightrtab `\rightrtab{<math><math>}{<text>}` for `\edaftertab` in `\rtab`.

```

6315 \newcommand{\rightrtab}[2]{%
6316     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6317         \disablel@dtabfeet#2}%
6318         #1\hb@xt@{\z@\vbox{\edtabindent%
6319         \advance\Hilfsskip by -\wd\hilfsbox%
6320         \advance\Hilfsskip by \edtabcolsep%
6321         \moveright\Hilfsskip\hbox{ #2}}\hss}%
6322     }
6323 %
6324 %

```

\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.

```

6325 \newcommand{\rtab}[1]{%
6326   \l@dnnullfills
6327   \def\edbeforetab##1##2{\lefttab{##1}{##2}}%
6328   \def\edaftertab##1##2{\righttab{##1}{##2}}%
6329   \measurebody{#1}%
6330   \l@drestorefills
6331   \variab
6332   \setmrowright #1\\&\\%
6333   \enablel@dtabfeet}
6334
6335 %

```

\measurebody `\measurebody{<body>}` measures the array `<body>`.

```

6336 \newcommand{\measurebody}[1]{%
6337   \disablel@dtabfeet%
6338   \l@dcolcount=0%
6339   \nullsetzen%
6340   \l@dcolcount=0
6341   \measuremrow #1\\&\\%
6342   \global\l@dampcount=1}
6343
6344 %

```

\rtabtext `\rtabtext{<body>}` typesets `<body>` as a tabular with the entries right justified.

```

6345 \newcommand{\rtabtext}[1]{%
6346   \l@dnnullfills
6347   \measuretbody{#1}%
6348   \l@drestorefills
6349   \variab
6350   \settroright #1\\&\\%
6351   \enablel@dtabfeet}
6352
6353 %

```

\measuretbody `\measuretbody{<body>}` measures the tabular `<body>`.

```

6354 \newcommand{\measuretbody}[1]{%
6355   \disable@notes%
6356   \disablel@dtabfeet%
6357   \l@dcolcount=0%
6358   \nullsetzen%
6359   \l@dcolcount=0
6360   \measuretror #1\\&\\%
6361   \restore@notes%
6362   \global\l@dampcount=1}
6363
6364 %

```

\ltab Array with entries left justified.

```

\edbeforetab
\edaftertab
6365 \newcommand{\ltab}[1]{%
6366 \l@dnnullfills
6367 \def\edbeforetab##1##2{\leftltab{##1}{##2}}%
6368 \def\edaftertab##1##2{\rightltab{##1}{##2}}%
6369 \measurebody{#1}%
6370 \l@drestorefills
6371 \variab
6372 \setmrowleft #1\\&\\%
6373 \enablel@dtabfeet}
6374
6375 %

```

\ltabtext Tabular with entries left justified.

```

6376 \newcommand{\ltabtext}[1]{%
6377 \l@dnnullfills
6378 \measuretbody{#1}%
6379 \l@drestorefills
6380 \variab
6381 \settrrowleft #1\\&\\%
6382 \enablel@dtabfeet}
6383
6384 %

```

\ctab Array with centered entries.

```

\edbeforetab
\edaftertab
6385 \newcommand{\ctab}[1]{%
6386 \l@dnnullfills
6387 \def\edbeforetab##1##2{\leftctab{##1}{##2}}%
6388 \def\edaftertab##1##2{\rightctab{##1}{##2}}%
6389 \measurebody{#1}%
6390 \l@drestorefills
6391 \variab
6392 \setmrowcenter #1\\&\\%
6393 \enablel@dtabfeet}
6394
6395 %

```

\ctabtext Tabular with entries centered.

```

6396 \newcommand{\ctabtext}[1]{%
6397 \l@dnnullfills
6398 \measuretbody{#1}%
6399 \l@drestorefills
6400 \variab
6401 \settrrowcenter #1\\&\\%
6402 \enablel@dtabfeet}
6403
6404 %

```

```
\spreadtext \newcommand{\spreadtext}[1]{%\l@dcolcount=\l@dampcount%
6406 \hb@xt@ \the\l@dcolwidth{\hbox{#1}\hss}}
6407 %
```

```
\spreadmath \newcommand{\spreadmath}[1]{%
6409 \hb@xt@ \the\l@dcolwidth{\hbox{$\displaystyle{#1}$}\hss}}
6410
6411 %
```

\HILFSskip More helpers.

```
\Hilfsskip
6412 \newskip\HILFSskip
6413 \newskip\Hilfsskip
6414
6415 %
```

```
\EDTABINDENT \newcommand{\EDTABINDENT}{%
6417 \ifnum\l@dcolcount=30\let\NEXT\relax\l@dcolcount=0%
6418 \else\step\l@dcolcount%
6419 \advance\Hilfsskip by\l@dcolwidth%
6420 \ifdim\l@dcolwidth=0pt\advance\hilfscount\@ne
6421 \else\advance\Hilfsskip by \the\hilfscount\edtabcolsep%
6422 \hilfscount=1\fi%
6423 \let\NEXT=\EDTABINDENT%
6424 \fi\NEXT}%
6425 %
```

\edtabindent (was \tabindent)

```
6426 \newcommand{\edtabindent}{%
6427 \l@dcolcount=0\relax
6428 \Hilfsskip=0pt%
6429 \hilfscount=1\relax
6430 \EDTABINDENT%
6431 \hilfsskip=\hsize%
6432 \advance\hilfsskip -\Hilfsskip%
6433 \Hilfsskip=0.5\hilfsskip%
6434 }%
6435
6436 %
```

\EDTAB (was \TAB)

```
6437 \def\EDTAB #1|#2|{%
6438 \setbox\tabhilfbox=\hbox{$\displaystyle{#1}$}%
6439 \setbox\tabHilfbox=\hbox{$\displaystyle{#2}$}%
6440 \advance\tabelskip -\wd\tabhilfbox%
```

```

6441 \advance\tabelskip -\wd\tabHilfbox%
6442 \unhbox\tabhilfbox\hskip\tabelskip%
6443 \unhbox\tabHilfbox}%
6444
6445 %

```

`\EDTABtext` (was `\TABtext`)

```

6446 \def\EDTABtext #1|#2|{%
6447   \setbox\tabhilfbox=\hbox{#1}%
6448   \setbox\tabHilfbox=\hbox{#2}%
6449   \advance\tabelskip -\wd\tabhilfbox%
6450   \advance\tabelskip -\wd\tabHilfbox%
6451   \unhbox\tabhilfbox\hskip\tabelskip%
6452   \unhbox\tabHilfbox}%
6453 %

```

`\tabhilfbox` Further helpers.

```

\tabHilfbox
6454 \newbox\tabhilfbox
6455 \newbox\tabHilfbox
6456
6457 %

```

XXVIII.2.4 Environments

`edarrayl edarrayc edarrayr` The ‘environment’ forms for `\ltab`, `\ctab` and `\rtab`.

```

6458 \newenvironment{edarrayl}{\l@ddcollect@body\ltab}{}
6459 \newenvironment{edarrayc}{\l@ddcollect@body\ctab}{}
6460 \newenvironment{edarrayr}{\l@ddcollect@body\rtab}{}
6461
6462 %

```

`edtabularl edtabularc edtabularr` The ‘environment’ forms for `\ltabtext`, `\ctabtext` and `\rtabtext`.

```

6463 \newenvironment{edtabularl}{\l@ddcollect@body\ltabtext}{}
6464 \newenvironment{edtabularc}{\l@ddcollect@body\ctabtext}{}
6465 \newenvironment{edtabularr}{\l@ddcollect@body\rtabtext}{}
6466
6467 %

```

XXIX Quotation’s commands

`\initnumbering@quote` This macro, called at the beginning of any numbered section, locally redefines the quotation and quote environments, in order to allow their use inside of numbered sections.

```

\quotation \initnumbering@quote defines quotation environment.
\endquotation
\quote
\endquote
\newcommand{\initnumbering@quote}{
  \ifnoquotation@else
  \renewcommand{\quotation}{\par\leavevmode%
    \parindent=1.5em%
    \skipnumbering%
    \ifautopar%
      \vskip-\parskip%
    \else%
      \vskip\topsep%
    \fi%
    \global\leftskip=\leftmargin%
    \global\rightskip=\leftmargin%
  }
  \renewcommand{\endquotation}{\par%
    \global\leftskip=0pt%
    \global\rightskip=0pt%
    \leavevmode%
    \skipnumbering%
    \ifautopar%
      \vskip-\parskip%
    \else%
      \vskip\topsep%
    \fi%
  }
  \renewcommand{\quote}{\par\leavevmode%
    \parindent=0pt%
    \skipnumbering%
    \ifautopar%
      \vskip-\parskip%
    \else%
      \vskip\topsep%
    \fi%
    \global\leftskip=\leftmargin%
    \global\rightskip=\leftmargin%
  }
  \renewcommand{\endquote}{\par%
    \global\leftskip=0pt%
    \global\rightskip=0pt%
    \leavevmode%
    \skipnumbering%
    \ifautopar%
      \vskip-\parskip%
    \else%
      \vskip\topsep%
    \fi%
  }
  \fi
}

```



```
6516 %
```

XXX Section's title commands

XXX.1 Commands to disable some feature

`\ledsectnotoc` The `\ledsectnotoc` only disables the `\addcontentsline` macro.

```
6517 \newcommand{\ledsectnotoc}{\let\addcontentsline@gobblethree}
6518 %
```

`\ledsectnomark` The `\ledsectnomark` only disables the `\chaptermark`, `\sectionmark` and `\subsectionmark` macros.

```
6519 \newcommand{\ledsectnomark}{%
6520   \let\chaptermark@gobble%
6521   \let\sectionmark@gobble%
6522   \let\subsectionmark@gobble%
6523 }
6524 %
```

XXX.2 General overview

The system of `\eledxxxx` commands to section text work like this:

1. When one of these commands is called, `reledmac` writes to an auxiliary files:
 - The section level.
 - The section title.
 - The side (when `eledpar` is used).
 - The pstart where the command is called.
 - If we have starred version or not.
2. `reledmac` adds the title of the section to pstart, as normal content. This is to enable critical notes.
3. When \TeX is run a other time, this file is read. That:
 - Adds the pstart number to a list of pstarts where a sectioning command is used.
 - Defines a command, the name of which contains the pstart number, and which calls the normal \TeX sectioning command.
4. This last command is called when the pstart is effectively printed.

XXX.3 \beforeeledchapter command

We do not define commands for \eledsection and related if the noeledsec option is loaded. We use etoolbox tests and not the \ifxxx... \else... \fi structure to prevent problem of expansions with command after the \ifxxx which contains \fi. As we patch command inside this test, we need to change the category code of # character before \notbool statement, because the second argument is read with the standard catcode (read *The TeXbook* to understand when the catcode's change has effect).

```
6525 \catcode`\#=12
6526 \notbool{noeled@sec}{%
6527 %
```

\beforeeledchapter For technical reasons, not yet solved, page-breaking before chapters can't be made automatically by eledmac. Users have to use \beforeeledchapter.

```
6528 \ifl@dmemoir
6529 \newcommand\beforeeledchapter{%
6530 \clearforchapter%
6531 }
6532 \else
6533 \newcommand\beforeeledchapter{%
6534 \if@openright%
6535 \cleardoublepage%
6536 \else%
6537 \clearpage%
6538 \fi%
6539 }
6540 \fi
6541 %
```

XXX.4 Auxiliary commands

\if@eled@sectioning The boolean \if@eled@sectioning is set to true when a sectioning command is called by a \eledxxx command, and set to false after. It is used to enable/disable line number printing.

```
6542 \newif\if@eled@sectioning
6543 %
```

\print@leftmargin@eledsection **\print@rightmargin@eledsection** \print@leftmargin@eledsection and \print@rightmargin@eledsection are added by reledmac inside the code of sectioning command, in order to affix lines numbers. They include tests for RTL languages.

```
6544 \def\print@rightmargin@eledsection{%
6545 \if@eled@sectioning%
6546 \begingroup%
6547 \if@RTL%
6548 \let\llap\rlap%
6549 \let\leftlinenum\rightlinenum%
```

```

6550 \let\leftlinenum\rightlinenumR%
6551 \let\l@drd@ta\l@dld@ta%
6552 \let\l@drsn@te\l@dlsn@te%
6553 \fi%
6554 \hfill\l@drd@ta \csuse{LR}{\l@drsn@te}%
6555 \endgroup%
6556 \fi%
6557 }%
6558
6559 \def\print@leftmargin@eledsection{%
6560 \if@eled@sectioning%
6561 \leavevmode%
6562 \begingroup%
6563 \if@RTL%
6564 \let\rlap\llap%
6565 \let\rightlinenum\leftlinenum%
6566 \let\rightlinenumR\leftlinenumR%
6567 \let\l@dld@ta\l@drd@ta%
6568 \let\l@dlsn@te\l@drsn@te%
6569 \fi%
6570 \l@dld@ta\csuse{LR}{\l@dlsn@te}%
6571 \endgroup%
6572 \fi%
6573 }%
6574
6575 %

```

XXX.5 Patching standard commands

`\chapter` 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.

```

6576 \AtBeginDocument{%
6577 \patchcmd{\chapter}{\clearforchapter}{%
6578 \if@eled@sectioning\else%
6579 \ifl@dprintingpages\else%
6580 \clearforchapter%
6581 \fi%
6582 \fi%
6583 }
6584 {}
6585 {}

```

```

6586
6587
6588 \pretocmd{\M@sect}
6589   {\let\old@edtext=\edtext%
6590    \let\edtext=\dummy@edtext@showlemma%
6591   }
6592   {}
6593   {}
6594
6595 \apptocmd{\M@sect}
6596   {\let\edtext=\old@edtext}
6597   {}
6598   {}
6599
6600 \patchcmd{\M@sect}
6601   { #9}
6602   { #9%
6603    \print@rightmargin@eledsection%
6604   }
6605   {}
6606   {}
6607
6608 \patchcmd{\M@sect}
6609   {\hskip #3\relax}
6610   {\hskip #3\relax%
6611    \print@leftmargin@eledsection%
6612   }
6613   {}
6614   {}
6615
6616 \patchcmd{\@mem@old@ssect}
6617   {#5}
6618   {#5%
6619    \print@leftmargin@eledsection%
6620   }
6621   {}
6622   {}
6623
6624 \patchcmd{\@mem@old@ssect}
6625   {\hskip #1}
6626   {\hskip #1%
6627    \print@rightmargin@eledsection%
6628   }
6629   {}
6630   {}
6631
6632 \patchcmd{\chapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
6633   \if@eled@sectioning\else%
6634     \ifl@dprintingpages\else%
6635       \if@openright\cleardoublepage\else\clearpage\fi}%No clearpage inside a

```

```

\Pages: will keep critical notes from printing on the title page. Here for
classical classes
    \fi%
6636 \fi%
6637 \fi%
6638 }%
6639 {}%
6640 {}%
6641
6642 \patchcmd{\scr@startchapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
\if@eled@sectioning\else%
6643 \ifl@dprintingpages\else%
6644 \if@openright\cleardoublepage\else\clearpage\fi}%No clearpage inside a
\Pages: will keep critical notes from printing on the title page. Here for
scrbook.
6646 \fi%
6647 \fi%
6648 }
6649 {}
6650 {}
6651
6652 \patchcmd{\@makechapterhead}
6653 {#1}
6654 {\print@leftmargin@eledsection%
6655 #1%
6656 \print@rightmargin@eledsection%
6657 }
6658 {}
6659 {}
6660
6661 \patchcmd{\@makechapterhead}% For BIDI
6662 {\if@RTL\raggedleft\else\raggedright\fi}%
6663 {\if@eled@sectioning\else%
6664 \if@RTL\raggedleft\else\raggedright\fi%
6665 \fi%
6666 }%
6667 {}%
6668 {}%
6669
6670 \patchcmd{\@makeschapterhead}
6671 {#1}
6672 {\print@leftmargin@eledsection%
6673 #1%
6674 \print@rightmargin@eledsection%
6675 }
6676 {}
6677 {}
6678
6679 \pretocmd{\@sect}
6680 {\let\old@edtext=\edtext

```

```

6681 \let\edtext=\dummy@edtext@showlemma%
6682 }
6683 {}
6684 {}
6685
6686 \apptocmd{\@sect}
6687 {\let\edtext=\old@edtext}
6688 {}
6689 {}
6690
6691 \pretocmd{\@ssect}
6692 {\let\old@edtext=\edtext%
6693 \let\edtext=\dummy@edtext@showlemma%
6694 }
6695 {}
6696 {}
6697
6698 \apptocmd{\@ssect}
6699 {\let\edtext=\old@edtext}
6700 {}
6701 {}
6702
6703 %

```

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.

```

6704 \@ifpackageloaded{nameref}{
6705
6706   \patchcmd{\NR@sect}
6707     {#8}
6708     {#8%
6709     \print@rightmargin@eledsection%
6710     }
6711     {}
6712     {}
6713
6714   \patchcmd{\NR@sect}
6715     {\hskip #3\relax}
6716     {\hskip #3\relax%
6717     \print@leftmargin@eledsection%
6718     }
6719     {}
6720     {}
6721
6722   \patchcmd{\NR@ssect}
6723     {#5}
6724     {#5%
6725     \print@rightmargin@eledsection%
6726     }
6727     {}

```

```

6728 {}
6729
6730 \patchcmd{\NR@ssect}
6731 {\hskip #1}
6732 {\hskip #1%
6733 \print@leftmargin@eledsection%
6734 }
6735 {}
6736 {}
6737 }%
6738 {
6739 \patchcmd{\@sect}
6740 {#8}
6741 {#8%
6742 \print@rightmargin@eledsection%
6743 }
6744 {}
6745 {}
6746
6747 \patchcmd{\@sect}
6748 {\hskip #3\relax}
6749 {\hskip #3\relax%
6750 \print@leftmargin@eledsection%
6751 }
6752 {}
6753 {}
6754
6755 \patchcmd{\@ssect}
6756 {#5}
6757 {#5%
6758 \print@rightmargin@eledsection%
6759 }
6760 {}
6761 {}
6762
6763 \patchcmd{\@ssect}
6764 {\hskip #1}
6765 {\hskip #1%
6766 \print@leftmargin@eledsection%
6767 }
6768 {}
6769 {}
6770 }%
6771 }
6772 %

```

Now, we have finished to patch the commands, using # with a catcode equals to 12. We close the `\notbool{@noeled@sec}` statement, restore the normal catcode for # and reopen a new `\notbool{@noeled@sec}` statement.

```

6773 {}}%

```

```

6774 \protect\catcode`\#=6 %Space NEEDS by \catcode
6775 \notbool{@noeled@sec}{%
6776 %

```

XXX.6 Main code of \eledxxx commands

`\eled@sectioning@out` `\eled@sectioning@out` is the output file, to dump the pstarts where a sectioning command is used.

```

6777 \newwrite\eled@sectioning@out
6778 %

```

`\eledchapter` `\eledsection` And now, the user sectioning commands, which write to the file, and also add content as a “normal” line.

```

\eledsubsection \newcommand{\eledchapter}[2] [] {%
\eledsubsubsection #2%
\eledchapter* \ifledRcol%
\eledsection* \immediate\write\eled@sectioningR@out{%
\eledsubsection* \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
\eledsubsubsection* }%
\else%
\immediate\write\eled@sectioning@out{%
\string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{R}
}%
\fi%
}
\newcommand{\eledsection}[2] [] {%
#2%
\ifledRcol%
\immediate\write\eled@sectioningR@out{%
\string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
}%
\else%
\immediate\write\eled@sectioning@out{%
\string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{R}
}%
\fi%
}
\newcommand{\eledsubsection}[2] [] {%
#2%
\ifledRcol%
\immediate\write\eled@sectioningR@out{%
\string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
}%
\else%

```



```

6812 \immediate\write\eled@sectioning@out{%
6813 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{-}{-}
6814 }%
6815 \fi%
6816 }
6817 \newcommand{\eledsubsubsection}[2][{}]{%
6818 #2%
6819 \ifledRcol%
6820 \immediate\write\eled@sectioningR@out{%
6821 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR
6822 }{-}{R}
6823 }%
6824 \else%
6825 \immediate\write\eled@sectioning@out{%
6826 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL
6827 }{-}{-}
6828 }%
6829 \fi%
6830 }
6831 \WithSuffix\newcommand\eledchapter*[2][{}]{%
6832 #2%
6833 \ifledRcol%
6834 \immediate\write\eled@sectioningR@out{%
6835 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
6836 }%
6837 \else%
6838 \immediate\write\eled@sectioning@out{%
6839 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{-}
6840 }%
6841 \fi%
6842 }
6843
6844 \WithSuffix\newcommand\eledsection*[2][{}]{%
6845 #2%
6846 \ifledRcol%
6847 \immediate\write\eled@sectioningR@out{%
6848 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
6849 }%
6850 \else%
6851 \immediate\write\eled@sectioning@out{%
6852 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{-}
6853 }%
6854 \fi%
6855 }
6856
6857 \WithSuffix\newcommand\eledsubsection*[2][{}]{%
6858 #2%
6859 \ifledRcol%

```

```

6860 \immediate\write\eled@sectioningR@out{%
6861 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsR}{*}{
R}
6862 }%
6863 \else%
6864 \immediate\write\eled@sectioning@out{%
6865 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL
}{*}{
6866 }%
6867 \fi%
6868 }
6869
6870 \WithSuffix\newcommand\eledsubsubsection*[2][ ]{%
6871 #2%
6872 \ifledRcol%
6873 \immediate\write\eled@sectioningR@out{%
6874 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsR
}{*}{R}
6875 }%
6876 \else%
6877 \immediate\write\eled@sectioning@out{%
6878 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL
}{*}{
6879 }%
6880 \fi%
6881 }
6882 %

```

XXX.7 Macros written in the auxiliary file

`\eled@chapter`
`\eled@section`
`\eled@subsection`
`\eled@subsubsection`

The sectioning macros, called in the auxiliary file. They have five arguments:

1. Optional arguments of \LaTeX sectioning command.
2. Mandatory arguments of \LaTeX sectioning command.
3. Pstart number.
4. Side: R if right, nothing if left.
5. Starred or not.

```

6883 \def\eled@chapter#1#2#3#4#5{%
6884 \ifstrempy{#4}%
6885 {%
6886 \ifstrempy{#1}%
6887 {%
6888 \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter{#2}}%

```

```

6889 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
chaptermark{#2}}%
6890 }%Need for \pairs, because of using parbox.
6891 {%
6892 \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter{#1}{#2}}%
6893 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
chaptermark{#2}}%Need for \pairs, because of using parbox.
6894 }%
6895 }%
6896 {%
6897 \ifstrempy{#1}%
6898 {\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#2}}}%
6899 {\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#1}{#2}}}%Bug in LaTeX!
6900 }%
6901 \listcsadd{eled@sections#5@@}{#3}%
6902 }
6903 \def\eled@section#1#2#3#4#5{%
6904 \ifstrempy{#4}%
6905 {\ifstrempy{#1}%
6906 {%
6907 \global\csdef{eled@sectioning@#3#5}{\section{#2}}%
6908 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
sectionmark{#2}}%Need for \pairs, because of using parbox.
6909 }%
6910 {%
6911 \global\csdef{eled@sectioning@#3#5}{\section{#1}{#2}}%
6912 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
sectionmark{#1}}%Need for \pairs, because of using parbox.
6913 }%
6914 }%
6915 {\ifstrempy{#1}%
6916 {\global\csdef{eled@sectioning@#3#5}{\section*{#2}}}%
6917 {\global\csdef{eled@sectioning@#3#5}{\section*{#1}{#2}}}%Bug in
LaTeX!
6918 }
6919 \listcsadd{eled@sections#5@@}{#3}%
6920 }
6921 \def\eled@subsection#1#2#3#4#5{%
6922 \ifstrempy{#4}%
6923 {\ifstrempy{#1}%
6924 {%
6925 \global\csdef{eled@sectioning@#3#5}{\subsection{#2}}%
6926 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\csuse
{subsectionmark}{#2}}%Need for \pairs, because of using parbox. \csuse in
case of \subsectionmark is not defined (book)
6927 }%
6928 {%

```

```

6929 \global\csdef{eled@sectioning@#3#5}{\subsection[#1]{#2}}%
6930 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\csuse
{subsectionmark}{#1}}%Need for \pairs, because of using parbox. \csuse in
case of \subsectionmark is not defined (book)
6931 }%
6932 }%
6933 {\ifstrempy{#1}%
6934 {\global\csdef{eled@sectioning@#3#5}{\subsection*{#2}}}%
6935 {\global\csdef{eled@sectioning@#3#5}{\subsection*{#1}{#2}}}%Bug in
LaTeX!
6936 }
6937 \listcsadd{eled@sections#5@@}{#3}%
6938 }
6939 \def\eled@subsubsection#1#2#3#4#5{%
6940 \ifstrempy{#4}%
6941 {\ifstrempy{#1}%
6942 {\global\csdef{eled@sectioning@#3#5}{\subsubsection{#2}}}%
6943 {\global\csdef{eled@sectioning@#3#5}{\subsubsection[#1]{#2}}}%
6944 }%
6945 {\ifstrempy{#1}%
6946 {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#2}}}%
6947 {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#1}{#2}}}%Bug
in LaTeX!
6948 }
6949 \listcsadd{eled@sections#5@@}{#3}%
6950 }
6951
6952 %

End of the conditional test about noeledsec option.

6953 }{}
6954 %

```

XXXI Page breaking or no page breaking depending of specific lines

By default, page breaks are automatic. However, the user can define lines which will force page breaks, or prevent page breaks around one specific line. On the first run, the line-list file records the line number of where the page break is being changed (either forced, or prevented). On the next run, page breaks occur either before or after this line, depending on how the user sets the command. The default setting is after the line.

\normal@page@break \normal@page@break is an etoolbox list which contains the absolute line number of the last line, for each page.

```

6955 \def\normal@page@break{}
6956 %

```

\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.

```
6957 \def\l@prev@pb{}
6958 \def\l@prev@nopb{}
6959 %
```

\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.

```
6960 \newcommand{\ledpb}{\write\linenum@out{\string\led@pb}}
6961 \newcommand{\ledpbnum}[1]{\write\linenum@out{\string\led@pbnum{#1}}}
6962 \newcommand{\lednopb}{\write\linenum@out{\string\led@nopb}}
6963 \newcommand{\lednopbnum}[1]{\write\linenum@out{\string\led@nopbnum{#1}}}
6964 %
```

\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.

```
6965 \newcommand{\led@pb}{\listxadd{\l@prev@pb}{\the\absline@num}}
6966 \newcommand{\led@pbnum}[1]{\listxadd{\l@prev@pb}{#1}}
6967 \newcommand{\led@nopb}{\listxadd{\l@prev@nopb}{\the\absline@num}}
6968 \newcommand{\led@nopbnum}[1]{\listxadd{\l@prev@nopb}{#1}}
6969 %
```

\ledpbsetting The \ledpbsetting macro only changes the value of \led@pb@macro, for which the default value is before.

```
6970 \def\led@pb@setting{before}
6971 \newcommand{\ledpbsetting}[1]{\gdef\led@pb@setting{#1}}
6972 %
```

\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.

```
6973 \newcommand{\led@check@pb}{\xifinlist{\the\absline@num}{\l@prev@pb}{\pagebreak[4]}}
6974 \newcommand{\led@check@nopb}{%
6975   \IfStrEq{\led@pb@setting}{before}{%
6976     \xifinlist{\the\absline@num}{\l@prev@nopb}{%
6977       {\numdef{\abs@prevline}{\the\absline@num-1}}%
6978       \xifinlist{\abs@prevline}{\normal@page@break}%
6979       {\nopagebreak[4]\enlargethispage{\baselineskip}}%
6980     }%
6981   }%
}
```

```

6982     {}%
6983   {}%
6984   \IfStrEq{\led@pb@setting}{after}{%
6985     \xifinlist{\the\absline@num}{\l@prev@nopb}{%
6986       \xifinlist{\the\absline@num}{\normal@page@break}%
6987       {\nopagebreak[4]\enlargethispage{\baselineskip}}%
6988     }%
6989 }%
6990   {}}%
6991   {}%
6992   {}%
6993 }
6994 %

```

XXXII Long verse: prevents being separated by a page break

\iflednopbinverse The `\lednopbinverse` boolean is set to false by default. If set to true, `reledmac` will automatically prevent page breaks inside verse. The declaration is made at the beginning of the file, because it is used as a package option.

\check@pb@in@verse The `\check@pb@in@verse` checks if a verse is broken in two page. If true, it adds:

- The absolute line number of the first line of the verse -1 in the `\led@pb` list, if the page break must occur before the verse.
- The absolute line number of the first line of the verse -1 in the `\led@nopb` list, if the page break must occur after the verse.

```

6995 \newcommand{\check@pb@in@verse}{%
6996   \ifinstanza\iflednopbinverse\ifinserthangingsymbol% Using stanzas and
        enabling page breaks in verse control, while on a hanging verse.
6997   \ifnum\page@num=\last@page@num\else%If we have change page
6998     \IfStrEq{\led@pb@setting}{before}{%
6999       \numgdef{\abs@line@verse}{\the\absline@num-1}%
7000       \ledpbnum{\abs@line@verse}%
7001     }{}%
7002     \IfStrEq{\led@pb@setting}{after}{%
7003       \numgdef{\abs@line@verse}{\the\absline@num-1}%
7004       \lednopbnum{\abs@line@verse}%
7005     }{}%
7006   \fi%
7007   \fi\fi\fi%
7008 }
7009 %

```

XXXIII Compatibility with eledmac

Here, we define some command for the eledmac-compat option.

```

7010 \ifeledmaccompat@%
7011
7012 \newcommand{\footnormalX}[1]{\arrangementX[#1]{normal}}%
7013 \newcommand{\footparagraphX}[1]{\arrangementX[#1]{paragraph}}%
7014 \newcommand{\foottwocolX}[1]{\arrangementX[#1]{twocol}}%
7015 \newcommand{\footthreecolX}[1]{\arrangementX[#1]{threecol}}%
7016
7017 \unless\ifnocritical@
7018   \newcommand{\footnormal}[1]{\Xarrangement[#1]{normal}}%
7019   \newcommand{\footparagraph}[1]{\Xarrangement[#1]{paragraph}}%
7020   \newcommand{\foottwocol}[1]{\Xarrangement[#1]{twocol}}%
7021   \newcommand{\footthreecol}[1]{\Xarrangement[#1]{threecol}}%
7022   \let\hsizetwocol\Xhsizetwocol
7023   \let\hsizethreecol\Xhsizethreecol
7024   \let\bhookXnote\Xbhooknote
7025   \let\boxsymlinenum\Xboxsymlinenum
7026   \let\symlinenum\Xsymlinenum
7027   \let\beforenumberinfootnote\Xbeforenumber
7028   \let\afternumberinfootnote\Xafternumber
7029   \let\beforeXsymlinenum\Xbeforesymlinenum
7030   \let\afterXsymlinenum\Xaftersymlinenum
7031   \let\inplaceofnumber\Xinplaceofnumber
7032   \let\Xlemmaseparator\lemmaseparator
7033   \let\afterlemmaseparator\Xafterlemmaseparator
7034   \let\beforelemmaseparator\Xbeforelemmaseparator
7035   \let\inplaceoflemmaseparator\Xinplaceoflemmaseparator
7036   \let\txbeforeXnotes\Xtxbeforenotes
7037   \let\afterXrule\Xafterrule
7038   \let\numberonlyfirstinline\Xnumberonlyfirstinline
7039   \let\numberonlyfirstintwolines\Xnumberonlyfirstintwolines
7040   \let\nonumberinfootnote\Xnonumberinfootnote
7041   \let\pstartinfootnote\Xpstart
7042   \let\pstartinfootnoteeverytime\Xpstarteverytime
7043   \let\onlyXpstart\Xonlypstart
7044   \let\Xnonumberinfootnote\Xnonumber
7045   \let\nonbreakableafternumber\Xnonbreakableafternumber
7046   \let\maxhXnotes\Xmaxhnotes
7047   \let\beforeXnotes\Xbeforenotes
7048   \let\boxlinenum\Xboxlinenum
7049   \let\boxlinenumalign\Xboxlinenumalign
7050   \let\boxstartlinenum\Xboxstartlinenum
7051   \let\boxendlinenum\Xboxendlinenum
7052   \let\twolines\Xtwolines
7053   \let\morethantwolines\Xmorethantwolines
7054   \let\twolinesbutnotmore\Xtwolinesbutnotmore
7055   \let\twolinesonlyinsamepage\Xtwolinesonlyinsamepage

```

```

7056 \fi
7057
7058 \unless\ifnofamiliar@
7059 \let\notesXwidthliketwocolumns\noteswidthliketwocolumnsX
7060 \fi
7061 \newcommandx{\parafootsep}[2][1,usedefault]{%
7062   \Xparafootsep[#1]{#2}%
7063   \parafootsepX[#1]{#2}
7064 }%
7065
7066 \newcommandx{\afternote}[2][1,usedefault]{%
7067   \Xafternote[#1]{#2}%
7068   \afternoteX[#1]{#2}%
7069 }%
7070
7071 \unless\ifnoend@
7072 \let\XendXtwolines\Xendtwolines
7073 \let\XendXmoreethantwolines\Xendmoreethantwolines
7074 \let\bhookXendnote\Xendbhooknote
7075 \let\boxXendlinenum\Xendboxlinenum%
7076 \let\boxXendlinenumalign\Xendboxlinenumalign%
7077 \let\boxXendstartlinenum\Xendboxstartlinenum%
7078 \let\boxXendendlinenum\Xendboxendlinenum%
7079 \let\XendXlemmaseparator\Xendlemmaseparator
7080 \let\XendXbeforelemmaseparator\Xendbeforelemmaseparator
7081 \let\XendXafterlemmaseparator\Xendafterlemmaseparator
7082 \let\XendXinplaceoflemmaseparator\Xendinplaceoflemmaseparator
7083 \fi
7084
7085 \AtBeginDocument{%
7086   \ifdef\lineref{}\let\lineref\edlineref}%
7087 }%
7088
7089
7090 \fi%
7091 %

```

</code>

Appendix A Some things to do when changing version

Appendix A.1 Migrating from edmac to ledmac

If you have never used edmac, ignore this section. If you have used edmac and are starting on a completely new document, ignore this section. Only read this section if you are converting an original edmac document to use ledmac.

The package still provides the original `\text` command, but it is (a) deprecated, and (b) its name has been changed³⁴ to `\critext`; use the `\edtext` macro instead. However, if you do use `\critext` (the new name for `\text`), the following is a reminder.

`\critext` Within numbered paragraphs, footnotes and endnotes are generated by forms of the `\critext` macro:

```
\critext{⟨lemma⟩}⟨commands⟩/
```

The `⟨lemma⟩` argument is the lemma in the main text: `\critext` both prints this as part of the text, and makes it available to the `⟨commands⟩` you specify to generate notes. The `/` at the end terminates the command; it is part of the macro's definition so that spaces after the macro will be treated as significant.

For example:

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

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

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

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

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

The second argument of the `\critext` macro, `⟨commands⟩`, is the same as the second argument to the `\edtext` macro.

It is possible to define aliases for `\critext`, which can be easier to type. You can make a single character substitute for `\critext` by saying this:

³⁴A name like `\text` is likely to be defined by other \TeX packages (it certainly is by the AMS packages) and it seems sensible to try and avoid clashes with other definitions.

```
\catcode`\<=\active
\let<=\critext
```

Then you might say `<{Smith}\variant{Jones}/`. This of course destroys the ability to use `<` in any new macro definitions, so long as it remains in effect; hence it should be used with care.

Changing the character at the end of the command requires more work:

```
\catcode`\<=\active
\def\xtext#1#2>{\critext{#1}{#2}/}
\let<=\xtext
```

This allows you to say `<{Smith}\Afootnote{Jones}>`.

Aliases for `\critext` of the first kind shown here also can't be nested—that is, you can't use the alias in the text that forms the first argument to `\critext`. (See VI p. 103 to find out why.) Aliases of the second kind may be nested without any problem.

If you really have to use `\critext` in any of the tabular or array environments, then `\edtext` must not be used in the same environment. If you use `\critext` in one of these environments then you have to issue the declaration `\usingcritext` beforehand. The declaration `\usingedtext` must be issued to revert to the default assumption that `\edtext` will be used.

Appendix A.2 Migration from ledmac to eledmac

In `eledmac`, some changes were made in the code to allow easy customization. This may cause problems for people who have already made their own. The next sections explain how to handle this.

If you have created your own series using `\addfootins` and `\addfootinsX`, you must use instead the `\newseries` command (see 5.5.1 p. 28), and remove any `\Xfootnote` command.

If you have customized the `\XXXXXfmt` command, please check whether you can achieve the same by the commands documented for display options (6 p. 29) or `\Xfootnote` options (5.2.2 p. 22). Otherwise please add a new ticket on Github to request a new function for doing this.³⁵

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]
```

³⁵<https://github.com/maieul/ledmac/issues>

If you do not make that, you will get a spurious [X], where X is series letter.

If you used a `\protect` command inside a `\footnote` command inside a numbered section, you must change the `\protect` to `\noexpand`. Otherwise the command after the `\protect` will be discarded.

Appendix A.3 Migration to eledmac 1.5.1

The version 1.5.1 corrects a bug in `stanzaindentsrepetition` (cf. 8.3 p. 40). This bug had two consequences:

1. `stanzaindentsrepetition` did not work when its value was greater than 2.
2. `stanzaindentsrepetition` worked wrong when its value was equal to 2.

So, if you used `stanzaindentsrepetition` with a value equal to 2, you had to change your `\setstanzaindents`. Explanation:

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

This code, in versions prior to 1.5.1, made the first line have an indentation of 0, the second line of 1, the third verse of 0, the fourth verse of 1 and so forth.

But this code should have instead achieved quite the contrary: the first line would have an indentation of 1, the second line of 0, the third line of 1, the fourth line of 0 and so forth.

So version 1.5.1 corrected this bug. If you want to keep the former presentation, you must change:

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

to:

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

Appendix A.4 Migration to eledmac 1.12.0

The migration to eledmac 1.12.0 is easy:

- You must first delete all the auxiliary files, then compile your document three times as usual.
- If you have modified `\l@reg`, which is not advisable, you must rename it to `\@nl@reg`.

There is an additional problem. If you have put text into brackets just after `\pstart` or `\pend`, this text will be considered to be an optional argument of `\pstart` or `\pend` (see 4.2.3 p. 16). If so, add a `\relax` between `\pstart`/`\pend` and the first bracket.

The version 1.12.0 also introduce a better way to handle sectional divisions inside numbered text. Please read 14.2 p. 53.

Appendix A.5 Migration to eledmac 17.1

This version changes the default setting of `\Xpstart`. Henceforth, pstart numbers will be printed in footnotes within the section of text where you have called `\numberpstarttrue`.

We do not see any reason to print them in the other sections. However, if you want to print the pstart numbers in all of the footnotes, whatever the section, without having to use `\numberpstarttrue`, you can use `\Xpstarteverytime`.

Appendix A.6 Migration to eledmac 1.21.0

Appendix A.6.1 `\Xledsetnormalparstuff` and `\ledsetnormalparstuffX`

The `\ledsetnormalparstuff` has been split into two different commands:

- `\Xledsetnormalparstuff` for critical notes;
- `\ledsetnormalparstuffX` for familiar notes.

Both commands can take an optional argument which is the series letter. If you have redefined `\ledsetnormalparstuff` or any of the commands which call them, you must change them accordingly.

Appendix A.6.2 Endnotes

In any case, delete the `.end` file before the next run.

The previous version of Eledmac had a bug: there were two spaces between the starting page number and the starting line number, but only one space between the ending page number and the ending line number.

As a matter of fact, a spurious space was added after the first `\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 17.2 p. 55 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 `\Xhooknote` and `\hooknoteX` 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 `\Xhooknote/\hooknoteX` argument. You must include it manually. So instead of `\Xhooknote{content}`, use `\Xhooknote{content }`.
2. Indent is no longer automatically added before the content of the `\Xhooknote/\hooknoteX` argument. If you want to keep it, add `\indent` in the argument of `\Xhooknote/\hooknoteX`.

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`.

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 `edstanz.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>	40
<code>\@EDROWFILL@</code>	1
<code>\@adv</code>	1
<code>\@advancestanzanumber</code>	1
<code>\@apprefprefixmore</code>	1
<code>\@apprefprefixsingle</code>	1
<code>\@doclearpage</code>	1
<code>\@doreinfeetX</code>	1
<code>\@edindex@hyperref</code>	1
<code>\@edrowfill@</code>	1
<code>\@edtext@level</code>	1

\@emptytoks	1
\@fnpos	1
\@footnotemark	1
\@footnotetext	1
\@getfirstseries	1
\@gobblefive	1
\@gobblefour	1
\@gobblethree	1
\@h	1
\@hangingsymbol	1
\@iiiminipage	1
\@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
\@nl	1
\@nl@reg	1
\@opXfeet	1
\@pend	1
\@pendR	1
\@ref	1
\@ref@reg	1
\@sect	1
\@series	1
\@set	1
\@sidenotesep	1
\@ssect	1
\@startstanza	1
\@stopstanza	1
\@sw	1
\@tag	1
\@wredindex	1
\@xloop	1
\@xympar	1
CLASSbook	275
CLASSmemoir	168, 214, 215, 239, 275, 293, 344, 348
CLASSscrbook	348
COMMAND*footnote	57

COMMAND\...\@footnotemark...	170
COMMAND\...d@ta	126
COMMAND\<hook	
@<series	204
COMMAND\<hookname	
<pseudoseries	206, 207
COMMAND\<type	
footfmt	159
COMMAND\@@line	153
COMMAND\@MM	141, 345
COMMAND\@Rlineflag	241, 345
COMMAND\@add@	265
COMMAND\@adv	90
COMMAND\@apprefprefixmore	223, 224
COMMAND\@apprefprefixsingle	223, 224
COMMAND\@bspback	217
COMMAND\@doclearpage	215, 338, 348
COMMAND\@doreinfeetX	348
COMMAND\@dprintingcolumns	345
COMMAND\@edindex@hyperref	241, 242
COMMAND\@edtext@	106
COMMAND\@esphack	217
COMMAND\@fnpos	185
COMMAND\@footnotemark	168, 169, 338, 348
COMMAND\@footnotetext	168, 169, 338
COMMAND\@gobble	105
COMMAND\@gobblefive	202, 346
COMMAND\@gobblefour	344
COMMAND\@gobblethree	337
COMMAND\@h	155
COMMAND\@hangingsymbol	243
COMMAND\@iiiminipage	232, 233, 337, 348
COMMAND\@iiiminipage	232
COMMAND\@l	343
COMMAND\@l@tempcnta	128, 130, 136
COMMAND\@l@tempcntb	130
COMMAND\@l@reg	343
COMMAND\@lab	87, 217, 219, 222, 337
COMMAND\@ldunboxmpfoot	234
COMMAND\@led@extranofeet	294
COMMAND\@ledinnote@command	237, 238
COMMAND\@lemma	109, 111
COMMAND\@lock	82, 243
COMMAND\@lopL	338
COMMAND\@lopR	338
COMMAND\@makecol	211, 212, 214, 348
COMMAND\@mpfnpos	185
COMMAND\@nl	87–90, 92, 99, 219, 337, 338
COMMAND\@nl@reg	88, 291, 338, 343
COMMAND\@opXfeet	338

COMMAND\@opfeetX	348
COMMAND\@opxtrafeeti	348
COMMAND\@page	89, 219
COMMAND\@pend	338
COMMAND\@pendR	338
COMMAND\@ref	87, 96, 97, 100, 104
COMMAND\@ref@reg	96, 338
COMMAND\@reinserts	211–214, 348
COMMAND\@secondoftwo	57
COMMAND\@sect	278
COMMAND\@series	203
COMMAND\@set	91
COMMAND\@sidenotesepp	230
COMMAND\@sw	97, 112, 115, 116
COMMAND\@tag	106, 107, 110
COMMAND\@tempcnta	68
COMMAND\@tempcntb	68
COMMAND\@toksa	74
COMMAND\@toksb	74
COMMAND\@xloop	137
COMMAND\@xympar	225, 348
COMMAND\Aendnote	14, 22
COMMAND\Afootfmt	140
COMMAND\Afootgroup	140
COMMAND\Afootnote	7, 13, 21, 22, 24, 108, 148, 167, 186, 197, 347
COMMAND\Afootstart	140
COMMAND\AtEveryPend	16, 121, 344, 346, 347
COMMAND\AtEveryPstart	16, 344, 346, 347
COMMAND\Bendnote	14, 21
COMMAND\Bfootnote	7, 13, 167, 186, 197
COMMAND\Centering	36
COMMAND\Cfootnote	167
COMMAND\Columns	69, 145
COMMAND\Dfootnote	167
COMMAND\Efootnote	167
COMMAND\NR	278
COMMAND\Pages	69, 212, 213
COMMAND\ProcessOptionsX	61
COMMAND\RaggedLeft	36
COMMAND\RaggedRight	36
COMMAND\Stanza	343
COMMAND\Waklam	266
COMMAND\X@doreinfeet	213, 348
COMMAND\XXXXXXfmt	290
COMMAND\XXXXXXfint	290
COMMAND\Xafterlemmaseparator	34, 294
COMMAND\Xafternote	37, 293, 294
COMMAND\Xafternumber	32, 294
COMMAND\Xafterrule	38, 186, 294, 343, 346
COMMAND\Xaftersymlinenum	32, 294

COMMAND\Xarrangement	29, 56, 141, 142, 205, 294
COMMAND\Xarrangement@footparagraph	146
COMMAND\Xarrangement@normal	142
COMMAND\Xarrangement@paragraph	146
COMMAND\Xbeforelemmaseparator	34, 294
COMMAND\Xbeforenotes	37, 186, 294, 343, 346
COMMAND\Xbeforenumber	32, 294
COMMAND\Xbeforesymlinenum	32, 294
COMMAND\Xbhookendnote	295
COMMAND\Xbhooknote	36, 295, 296, 348
COMMAND\Xboxendlinenum	33, 295, 347
COMMAND\Xboxlinenum	32, 33, 295
COMMAND\Xboxlinenumalign	33, 295, 347
COMMAND\Xboxstartlinenum	33, 295, 347
COMMAND\Xboxsymlinenum	33, 295
COMMAND\Xcolalign	36, 346
COMMAND\Xdo@feet	212, 338, 348
COMMAND\Xend	202
COMMAND\XendXafterlemmaseparator	295
COMMAND\XendXbeforelemmaseparator	295
COMMAND\XendXinplaceoflemmaseparator	295
COMMAND\XendXlemmaseparator	295
COMMAND\XendXmorethantwolines	295
COMMAND\XendXtwolines	295
COMMAND\Xendafterenumber	32, 349
COMMAND\Xendafterlemmaseparator	34, 295
COMMAND\Xendafternote	38
COMMAND\Xendafternumber	33
COMMAND\Xendaftersymlinenum	32, 33, 349
COMMAND\Xendahookinplaceofnumber	33, 349
COMMAND\Xendahooklinenumber	33, 349
COMMAND\Xendbeforelemmaseparator	34, 295
COMMAND\Xendbeforelinenumber	33
COMMAND\Xendbeforenumber	32, 349
COMMAND\Xendbeforesymlinenum	32, 33, 349
COMMAND\Xendbhookinplaceofnumber	33, 349
COMMAND\Xendbhooklinenumber	33, 349
COMMAND\Xendbhooknote	36
COMMAND\Xendboxendlinenum	33, 347
COMMAND\Xendboxlinenum	33, 295, 345
COMMAND\Xendboxlinenumalign	33, 295, 347
COMMAND\Xendboxstartlinenum	33, 347
COMMAND\Xendboxsymlinenum	33, 349
COMMAND\Xendhangindent	35, 349
COMMAND\Xendinplaceoflemmaseparator	34, 295
COMMAND\Xendinplaceofnumber	32, 348
COMMAND\Xendinsertsep@	194
COMMAND\Xendlemmadisablefontselection	35
COMMAND\Xendlemmaseparator	23, 34, 295
COMMAND\Xendletboxendlinenum	295

COMMAND\Xendmorethantwolines	22, 31, 46, 295, 346
COMMAND\Xendmorethantwolinesappref	46
COMMAND\Xendnonumber	31, 348
COMMAND\Xendnote	189, 201, 202, 346
COMMAND\Xendnotefontsize	35, 294
COMMAND\Xendnotenumfont	33, 34, 294
COMMAND\Xendnumberonlyfirstinline	30, 349
COMMAND\Xendnumberonlyfirstintwolines	30, 349
COMMAND\Xendparagraph	38, 343
COMMAND\Xendsep	38
COMMAND\Xendsymlinenum	30, 349
COMMAND\Xendtwolines	22, 31, 46, 295, 346
COMMAND\Xendtwolinesappref	46
COMMAND\Xendtwolinesbutnotmore	31, 46, 346
COMMAND\Xendtwolinesbutnotmoreappref	46
COMMAND\Xendtwolinesonlyinsamepage	31, 46, 346
COMMAND\Xendtwolinesonlyinsamepageappref	46
COMMAND\Xfootfmt	296
COMMAND\Xfootgroup	145
COMMAND\Xfootins	144
COMMAND\Xfootnote	44, 106, 290, 340, 344–346
COMMAND\Xfootstarts	145
COMMAND\Xhangindent	35, 349
COMMAND\Xhsizethreecol	36, 295
COMMAND\Xhsizetwocol	36, 206, 295
COMMAND\Xinplaceoflemmaseparator	34, 295
COMMAND\Xinplaceofnumber	32, 295, 345, 347
COMMAND\Xinsertparafootsep	151, 152
COMMAND\Xledsetnormalparstuff	292, 293, 346
COMMAND\Xlemmadisablefontselection	35
COMMAND\Xlemmaseparator	34, 208, 209, 211, 295
COMMAND\Xmaxhnotes	38, 56, 294, 295, 343, 345
COMMAND\Xmorethantwolines	22, 30, 31, 46, 210, 223, 295, 345
COMMAND\Xmorethantwolinesappref	46, 223
COMMAND\Xnoindent	349
COMMAND\Xnolemmaseparator	34, 211, 295
COMMAND\Xnonbreakableafternumber	32, 295, 341
COMMAND\Xnonumber	31, 295
COMMAND\Xnonumberinfootnote	295
COMMAND\Xnotefontsize	35, 294
COMMAND\Xnotefontsize@{s}	151, 154, 155
COMMAND\Xnotenumfont	34, 294
COMMAND\Xnoteswidthliketwocolumns	38, 344
COMMAND\Xnumberonlyfirstinline	30, 84, 207–209, 295, 340, 345
COMMAND\Xnumberonlyfirstintwolines	30, 295, 340
COMMAND\Xonlypstart	31, 295, 340, 345
COMMAND\Xparafootsep	37, 85, 294, 295
COMMAND\Xparafootsep@series	151
COMMAND\Xparindent	35, 346
COMMAND\Xpstart	31, 292, 295, 340, 345

COMMAND\Xpstarteverytime	31, 292, 295, 345
COMMAND\Xragged	37
COMMAND\Xstanza	32, 42
COMMAND\Xstanzaseparator	32
COMMAND\Xsymlinenum	30, 37, 294, 295, 347
COMMAND\Xtwolines	22, 30, 31, 46, 164, 165, 206, 210, 223, 295, 345
COMMAND\Xtwolinesappref	46, 206, 223
COMMAND\Xtwolinesbutnotmore	31, 46, 295, 346
COMMAND\Xtwolinesbutnotmoreappref	46, 207
COMMAND\Xtwolinesonlyinsamepage	31, 46, 295, 346
COMMAND\Xtwolinesonlyinsamepageappref	46
COMMAND\Xtxtbeforenotes	37, 295
COMMAND\Xunvxh	148, 296
COMMAND\&	294
COMMAND\absline@num	81, 127
COMMAND\accent	105
COMMAND\actionlines@list	82, 129
COMMAND\actions@list	82
COMMAND\add@inserts	82, 135
COMMAND\add@inserts@next	135, 136
COMMAND\add@penalties	127, 136
COMMAND\addcontentsline	273
COMMAND\addfootins	290, 293
COMMAND\addfootinsX	290, 293
COMMAND\advancelabel@refs	218
COMMAND\advanceline	19, 20, 84, 90, 101, 348
COMMAND\advancepageno	211
COMMAND\affixlin@num	230
COMMAND\affixline@num	130, 132, 134, 338
COMMAND\affixpstart@num	134
COMMAND\afterXrule	294
COMMAND\afterXsymlinenum	294
COMMAND\afterenumber	32
COMMAND\aftergroup	104, 108
COMMAND\afterlemmaseparator	294
COMMAND\afternote	294
COMMAND\afternoteX	37, 293, 294
COMMAND\afternumberinfootnote	294
COMMAND\afterruleX	38, 343, 346
COMMAND\applabel	45, 219, 220, 223, 346
COMMAND\appref	46, 223, 224
COMMAND\apprefprefixmore	294
COMMAND\apprefprefixsingle	294
COMMAND\apprefwithpage	46, 223, 224
COMMAND\arrangementX	29, 56, 170, 205, 294
COMMAND\arrangementX@normal	175
COMMAND\at@every@pend	121
COMMAND\autopar	15, 16, 118, 122, 123, 183, 339, 341, 342, 346
COMMAND\ballast	56
COMMAND\ballast@count	127, 136

COMMAND\baselineskip	29, 147, 151
COMMAND\beforeXnotes	294
COMMAND\beforeXsymlinenum	294
COMMAND\beforeelectedchapter	8, 53, 274
COMMAND\beforelemmaseparator	294
COMMAND\beforenotesX	37, 296, 342, 343, 346
COMMAND\beforenumberinfootnote	294
COMMAND\begin	250
COMMAND\beginnumbering	14, 16, 17, 69, 70, 72, 81, 85, 99, 122, 188, 340, 343, 347, 348
COMMAND\bf	340
COMMAND\bfseries	34, 35, 340
COMMAND\bhookXnote	295
COMMAND\bhooknoteX	36, 296, 348
COMMAND\body	244
COMMAND\bodyfootmarkA	27
COMMAND\boxXendlinenum	295
COMMAND\boxXendlinenumalign	295
COMMAND\boxXendstartlinenum	295
COMMAND\boxendlinenum	295
COMMAND\boxlinefootnote	162
COMMAND\boxlinenum	295
COMMAND\boxlinenumalign	295
COMMAND\boxstartlinenum	295
COMMAND\boxsymlinenum	295
COMMAND\break	29, 149
COMMAND\brokenpenalty	136
COMMAND\centering	36
COMMAND\ch@ck@l@ck	339
COMMAND\ch@cksub@l@ck	132, 339
COMMAND\chapter	52, 275, 343, 346, 348
COMMAND\chaptermark	273
COMMAND\check@pb@in@verse	286
COMMAND\colalignX	36, 346
COMMAND\collect@body	251
COMMAND\colorbox	57
COMMAND\columns	38
COMMAND\columnwidth	147, 344
COMMAND\command names	206, 207
COMMAND\copyright	105
COMMAND\correct@Xfootins@box	345
COMMAND\correct@footinsX@box	345
COMMAND\count	154
COMMAND\critex	339
COMMAND\critext	111, 289, 290, 293
COMMAND\csname	61, 114
COMMAND\ctab	266, 271
COMMAND\ctabtext	271
COMMAND\dcoll	260
COMMAND\def	59
COMMAND\detokenize	114

COMMAND\dimen	154
COMMAND\discretionary	148
COMMAND\displaywidowpenalty	136
COMMAND\do@actions	127–129, 339
COMMAND\do@actions@fixedcode	338
COMMAND\do@actions@next	128, 129
COMMAND\do@ballast	127, 136
COMMAND\do@feetX	348
COMMAND\do@insidelinehook	341
COMMAND\do@line	82, 104, 120, 123, 126, 135, 136, 243, 339, 341, 343
COMMAND\do@linehook	339
COMMAND\do@lockoff	83
COMMAND\do@lockon	83
COMMAND\dodoreintrafeet	337
COMMAND\doendnotes	22, 194, 195, 296, 346
COMMAND\doendnotesbysection	23, 195, 202, 347
COMMAND\doinsidelinehook	20, 344
COMMAND\dolinehook	20, 344
COMMAND\doreintrafeeti	348
COMMAND\doreintrafeetii	348
COMMAND\doxtrafeet	211, 337
COMMAND\doxtrafeeti	348
COMMAND\doxtrafeetii	348
COMMAND\dummy@ref	104
COMMAND\edaftertab	52, 266, 267
COMMAND\edatleft	51, 264
COMMAND\edatright	51, 52, 264
COMMAND\edbforetab	52, 266
COMMAND\edfilldimen	265
COMMAND\edfont@info	110
COMMAND\edindex	49, 237, 240, 242, 255, 341, 344, 345, 348, 349
COMMAND\edindexlab	49
COMMAND\edlabel	44–46, 105, 216–219, 221, 225, 236, 255, 337, 340–342, 345
COMMAND\edlineref	44, 216, 295, 345, 347
COMMAND\edmakelabel	45, 225
COMMAND\edpageref	44, 216, 221, 225
COMMAND\edrowfill	266
COMMAND\edtabcolsep	259
COMMAND\edtext	6, 21–24, 26, 27, 39, 44, 45, 50, 57, 82, 96, 97, 100, 103–111, 113–117, 219, 220, 222, 254, 256, 275, 289, 290, 293, 338, 339, 341, 343–347
COMMAND\edtext@level	347
COMMAND\edvertdots	52, 265
COMMAND\edvertline	52, 264
COMMAND\elechapter	53
COMMAND\eled@sectioning@out	280
COMMAND\eledchapter	53, 293, 344, 348
COMMAND\eledchapter*	53
COMMAND\eledmac@error	337
COMMAND\eledsection	6, 14, 53, 105, 125, 274, 293, 345
COMMAND\eledsection*	53

COMMAND\eledsubsection	53, 293
COMMAND\eledsubsection*	53
COMMAND\eledsubsubsection	53, 293
COMMAND\eledsubsubsection*	53
COMMAND\eledxxx	8, 54, 274, 280, 343
COMMAND\eledxxxx	273
COMMAND\else	236, 274
COMMAND\empty	68, 131, 217
COMMAND\end	250
COMMAND\end@lemmas	104
COMMAND\endashchar	39, 158
COMMAND\endgraf	120, 150, 183
COMMAND\endlock	19, 83, 102, 248
COMMAND\endminipage	232, 233, 337, 348
COMMAND\endnotes	346
COMMAND\endnumbering	14, 17, 69–72, 339, 347
COMMAND\endprint	189, 191, 202, 292
COMMAND\endstanzaextra	294
COMMAND\endsub	19, 83, 101
COMMAND\everypar	122
COMMAND\extensionchars	55, 69
COMMAND\f@x@l@cks	339
COMMAND>falseverse	293, 341, 343
COMMAND\fi	274
COMMAND\firstlinenum	18, 130, 339
COMMAND\firstsublinenum	18, 339
COMMAND\fix@page	88, 89, 338
COMMAND\flag@end	100, 109, 343
COMMAND\flag@start	100, 109, 343, 344
COMMAND\flagstanza	43
COMMAND\floatingpenalty	141, 345
COMMAND\flush@notes	137
COMMAND\fnpos	185, 342
COMMAND\footfmt	140, 143
COMMAND\footfmt...	171
COMMAND\footfootmarkA	27
COMMAND\footfudgefactor	149
COMMAND\footfudgefiddle	56, 147, 337
COMMAND\footgroup	140
COMMAND\footins	144
COMMAND\footnormal	206, 294, 338
COMMAND\footnormalX	294
COMMAND\footnote	27, 56, 168, 169, 291, 338
COMMAND\footnote@lang	159
COMMAND\footnoteA	14, 27
COMMAND\footnoteB	14
COMMAND\footnoteC	21
COMMAND\footnoteE	27
COMMAND\footnoteX	7, 200
COMMAND\footnoteXmk	211

COMMAND\footnotelang@lua	139
COMMAND\footnotelang@poly	139
COMMAND\footnoteoption@	138
COMMAND\footnoterule	154
COMMAND\footnotesize	35
COMMAND\footparagraph	147, 206, 294, 343
COMMAND\footparagraphX	179, 294, 343
COMMAND\footsplitskips	339, 345
COMMAND\footstart	140, 144, 154
COMMAND\footstrut	150
COMMAND\footthreecol	294
COMMAND\footthreecolX	294, 346
COMMAND\foottwocol	294
COMMAND\foottwocolX	294, 346
COMMAND\fulllines@	210
COMMAND\fullstop	39
COMMAND\get@edindex@hyperref	241
COMMAND\get@edindex@ledinnote@command	237
COMMAND\get@index@command	342
COMMAND\get@linelistfile	339
COMMAND\getline@num	127, 128
COMMAND\gl@p	74
COMMAND\global	87
COMMAND\globaldefs	87
COMMAND\hangindentX	35, 346, 349
COMMAND\hangingsymbol	294, 339
COMMAND\hbox	148
COMMAND\hfill	342
COMMAND\hidenumbering	20, 95, 346
COMMAND\hline	50
COMMAND\hrulefill	266
COMMAND\hsize	30, 144, 147, 149, 154, 157, 184, 338, 344
COMMAND\hsizethreecol	295
COMMAND\hsizethreecolX	36
COMMAND\hsizetwocol	295
COMMAND\hsizetwocolX	36
COMMAND\hyperlinkR	240
COMMAND\hyperlinkformat	240
COMMAND\hyperlinkformatR	241
COMMAND\if@RTL	62
COMMAND\if@edtext@	344, 347
COMMAND\if@eled@sectioning	274
COMMAND\if@noneed@Footnote	100
COMMAND\ifbypage@	74
COMMAND\ifbypage@R	75
COMMAND\ifbypstart@	74
COMMAND\ifbypstart@R	75
COMMAND\iffirst@linenum@out@	98, 99
COMMAND\ifinserthangingsymbol	243
COMMAND\ifinstanza	243

COMMAND\ifistwofollowinglines	165, 166
COMMAND\ifl@d@Xmorethantwolines	163, 346
COMMAND\ifl@d@Xtwolines	163
COMMAND\ifl@d@dash	163
COMMAND\ifl@d@elin	163
COMMAND\ifl@d@esl	163
COMMAND\ifl@d@pnum	162
COMMAND\ifl@d@ssub	162
COMMAND\ifl@dend@X	201
COMMAND\ifl@dmemoir	337
COMMAND\ifl@dpaging	344
COMMAND\ifl@dpairing	69, 339
COMMAND\ifl@dprintingpages	345
COMMAND\ifl@dskipnumber	130
COMMAND\ifl@dstartendok	265
COMMAND\ifl@imakeidx	62
COMMAND\ifledRcol	69, 340
COMMAND\ifledRcol@	69, 343
COMMAND\iflemmacommand@	345
COMMAND\ifnoledgroup@	236
COMMAND\ifnoteschanged@	84
COMMAND\ifnumberedpar@	118
COMMAND\ifnumbering	70, 72
COMMAND\ifnumberingR	69, 340
COMMAND\ifnumberline	109, 130
COMMAND\ifpst@rted	339
COMMAND\ifpst@rtedL	70
COMMAND\ifseriesbefore	204
COMMAND\ifsublines@	81, 93
COMMAND\iftrue	347
COMMAND\ifvmode	218
COMMAND\ifxxx	274
COMMAND\ignorespaces	108
COMMAND\imki@wrindexentry	62
COMMAND\immediate	98, 99, 188
COMMAND\indent	16, 122, 296
COMMAND\indtl@wrindexentry	62
COMMAND\initnumbering@quote	272, 348
COMMAND\initnumbering@reg	339
COMMAND\initnumbering@sectcmd	348
COMMAND\inplaceoflemmaseparator	295
COMMAND\inplaceofnumber	295
COMMAND\insert	135, 140, 143, 171
COMMAND\insert@count	96, 100, 107
COMMAND\insert@countR	107
COMMAND\inserthangingsymbol	342
COMMAND\insertlines@list	82, 96
COMMAND\insertparafootsepX	183
COMMAND\inserts@list	104, 118, 135, 148
COMMAND\interAfootnotelinepenalty	338

COMMAND\interfootnotelinepenalty	338
COMMAND\interlinepenalty	140
COMMAND\interparanoteglue	293
COMMAND\justifying	36
COMMAND\l@advance@parledegroupp@beforenormalnotes	348
COMMAND\l@d@w@rindexhyp	344
COMMAND\l@d@add	112
COMMAND\l@d@end	188, 201
COMMAND\l@d@nums	107, 109, 111, 112, 162, 163
COMMAND\l@d@section	189
COMMAND\l@d@set	92, 102
COMMAND\l@dampcount	256
COMMAND\l@dbfnote	169, 338
COMMAND\l@dcheckstartend	265
COMMAND\l@dchset@num	92
COMMAND\l@dcolcount	256, 257
COMMAND\l@dcollect@@body	250
COMMAND\l@dcollect@body	250
COMMAND\l@dcsnote	343
COMMAND\l@dcsnotetext	126, 229
COMMAND\l@dcsnotetext@l	126, 229
COMMAND\l@dcsnotetext@r	126, 229
COMMAND\l@ddodorextrafeet	212, 337
COMMAND\l@ddoxtrafeet	212, 337
COMMAND\l@demptyd@ta	339
COMMAND\l@dend@close	188
COMMAND\l@dend@open	188
COMMAND\l@dend@stuff	188
COMMAND\l@denvbody	250
COMMAND\l@dfeetbeginmini	338
COMMAND\l@dfeetendmini	338
COMMAND\l@dgetline@margin	339
COMMAND\l@dgetlock@disp	339
COMMAND\l@dgetref@num	222
COMMAND\l@dgetsidenote@margin	226, 339
COMMAND\l@dgobbleoptarg	344
COMMAND\l@dgobblearg	344
COMMAND\l@dgobbleoptarg	255
COMMAND\l@dlabel@parse	222
COMMAND\l@dld@ta	130, 132
COMMAND\l@dlp@rbox	230
COMMAND\l@dlsn@te	339
COMMAND\l@dlsnote	343
COMMAND\l@dmake@labels	218
COMMAND\l@dnumpsstartsL	70, 339
COMMAND\l@dp@rsefootspec	163
COMMAND\l@dparsefootspec	163
COMMAND\l@dpush@begins	250
COMMAND\l@d@drd@ta	130, 132
COMMAND\l@dref@undefined	221

COMMAND\l@drsn@te	339
COMMAND\l@drsnote	343
COMMAND\l@dtabaddcols	265
COMMAND\l@dtabnoexpands	337
COMMAND\l@dumboxmpfoot	348
COMMAND\l@dunboxmpfoot	339
COMMAND\l@dzeropenalties	339, 344
COMMAND\l@pb	285
COMMAND\l@prev@nopb	285
COMMAND\l@prev@pb	285
COMMAND\l@reg	291
COMMAND\label	16, 45, 49, 217, 222
COMMAND\label@refs	217
COMMAND\labelstarttrue	16, 340
COMMAND\labelref@list	216, 217, 219
COMMAND\language	148
COMMAND\last@page@num	338
COMMAND\lastbox	122
COMMAND\lastskip	101
COMMAND\leavevmode	16, 122
COMMAND\led@check@nopb	285
COMMAND\led@check@pb	285
COMMAND\led@nopb	285, 286
COMMAND\led@nopbnum	285
COMMAND\led@pb	285, 286
COMMAND\led@pb@macro	285
COMMAND\led@pbnum	285
COMMAND\ledRflag	240
COMMAND\ledchapter	293, 341
COMMAND\ledfootinsdim	294
COMMAND\ledinnernote	46, 227, 343
COMMAND\ledinnote	239, 292, 347
COMMAND\ledinnotemark	44, 292, 346
COMMAND\ledleftnote	47, 227
COMMAND\ledlinenum	80, 339
COMMAND\ledllfill	126
COMMAND\ledlsnotesep	47
COMMAND\ledlsnotewidth	47
COMMAND\lednopb	54, 285
COMMAND\lednopbinverse	286
COMMAND\lednopbinversetrue	42, 54
COMMAND\lednopbnum	285
COMMAND\ledouternote	46, 227, 343
COMMAND\ledpb	54, 285, 293
COMMAND\ledpbnum	285
COMMAND\ledpbsetting	55, 285, 348
COMMAND\ledrightnote	47, 227
COMMAND\ledrsnotesep	47
COMMAND\ledrsnotewidth	47
COMMAND\ledsection	293

COMMAND\ledsectnomark	273
COMMAND\ledsectnotoc	273
COMMAND\ledsetnormalparstuff	292, 293, 346
COMMAND\ledsetnormalparstuff@common	183
COMMAND\ledsetnormalparstuffX	292, 293, 346
COMMAND\ledsidenote	46, 47, 227, 229
COMMAND\ledsubsection	293
COMMAND\ledsubsubsection	293
COMMAND\ledxxx	343
COMMAND\left	51
COMMAND\leftctab	266
COMMAND\leftheadline	80
COMMAND\leftlinenum	19, 79, 337, 339
COMMAND\leftltab	266
COMMAND\leftnoteupfalse	47
COMMAND\leftstartnum	134
COMMAND\leftrtab	266
COMMAND\leftsidenote	229
COMMAND\leftskip	144, 147, 149
COMMAND\lemma	2, 22–24, 26, 27, 103, 106, 108, 110, 111, 113, 289, 339, 340, 347, 348
COMMAND\lemmaseparator	295
COMMAND\let	24, 40, 248, 337
COMMAND\letboxXendendlinum	295
COMMAND\line	153, 155
COMMAND\line@list	82, 97, 109, 110
COMMAND\line@list@stuff	70, 85, 99, 337, 339
COMMAND\line@list@version	87
COMMAND\line@margin	76, 132, 226
COMMAND\line@num	81, 83, 130, 337
COMMAND\line@set	111, 112
COMMAND\lineation	18, 75
COMMAND\linebreak	29
COMMAND\linenum	22, 24, 44–46, 103, 111, 221, 222, 225, 289
COMMAND\linenum@out	98, 217, 219
COMMAND\linenumberlist	18, 68, 131, 337
COMMAND\linenumberstyle	20, 79, 337
COMMAND\linenumincrement	18, 339
COMMAND\linenummargin	18, 76, 226
COMMAND\linenumr@p	79, 337, 339
COMMAND\linenumrep	79, 339
COMMAND\linenumsep	19, 47, 80, 227
COMMAND\lineref	216, 221, 225, 295, 345
COMMAND\list@clear	73
COMMAND\list@clearing@reg	339
COMMAND\list@create	73
COMMAND\lock@disp	78
COMMAND\lock@off	94
COMMAND\lock@on	93
COMMAND\lockdisp	19, 78
COMMAND\loop	137, 138, 244

COMMAND\ltab	266, 267, 271
COMMAND\ltabtext	271
COMMAND\m@mmf@prepare	168
COMMAND\makeatletter	126
COMMAND\makehboxoffhboxes	149, 151
COMMAND\makeindex	48, 240
COMMAND\makelabel	225
COMMAND\managestanza@modulo	245
COMMAND\marginpar	46, 56, 225, 226, 338
COMMAND\marginparwidth	47, 227
COMMAND\markboth	126
COMMAND\mathchardef	244
COMMAND\maxhXnotes	295
COMMAND\maxhnotesX	38, 56, 294, 342, 343, 345, 346
COMMAND\maxlinesinpar@list	85
COMMAND\measurebody	268
COMMAND\measuretbody	268
COMMAND\memorybreak	17
COMMAND\morenoexpands	57, 104, 105
COMMAND\morethantwolines	295
COMMAND\mpfnpos	185, 342
COMMAND\mpnormalfootgroup	338
COMMAND\mpnormalvfootnote	338
COMMAND\multfootsep	28, 168
COMMAND\multiplefootnotemarker	168
COMMAND\musixtex	343
COMMAND\n@num	339, 346
COMMAND\n@num@ref	346
COMMAND\new@line	99, 338
COMMAND\newcommand	24, 59, 168, 218
COMMAND\newcommandx	24
COMMAND\newhookcommand@series	206, 207, 346
COMMAND\newhookcommand@series@reload	207
COMMAND\newhookcommand@toggle@reload	207, 344
COMMAND\newhooktoggle@series	207, 346
COMMAND\newif	346
COMMAND\newline	29
COMMAND\newlinechar	201
COMMAND\newseries	28, 290, 293, 294
COMMAND\newseries@	195, 196, 204
COMMAND\newverse	42, 293, 343
COMMAND\next	244
COMMAND\next@action	86
COMMAND\next@actionline	86
COMMAND\next@insert	136
COMMAND\nl@regR	88
COMMAND\no@expands	57, 110, 337
COMMAND\noXlemmaseparator	295
COMMAND\nobreak	162
COMMAND\nocritical	196

COMMAND\noeledsec	54, 293
COMMAND\noendnotes	293
COMMAND\noexpand	291
COMMAND\nofamiliar	209
COMMAND\noindent	16, 122, 349
COMMAND\noindentX	349
COMMAND\nomk@	211
COMMAND\nonbreakableafternumber	295
COMMAND\nonum@	210, 211
COMMAND\nonumberinfootnote	295
COMMAND\normal@footnotemarkX	171
COMMAND\normal@page@break	284
COMMAND\normal@pars	183
COMMAND\normalbfnoteX	339
COMMAND\normalbodyfootmarkX	171
COMMAND\normalfootfmt	40, 143, 150, 159, 189
COMMAND\normalfootfmtX	172
COMMAND\normalfootfootmarkX	172
COMMAND\normalfootgroup	145
COMMAND\normalfootgroupX	173
COMMAND\normalfootnoterule	141
COMMAND\normalfootstart	144, 147
COMMAND\normalfootstartX	172
COMMAND\normalvfootnote	143
COMMAND\normalvfootnoteX	171
COMMAND\notbool	274
COMMAND\notfontsetup	294
COMMAND\notfontsizeX	35, 294
COMMAND\notenumfont	294
COMMAND\notenumfontX	35, 294
COMMAND\notesXwidthliketwocolumns	295
COMMAND\noteswidthliketwocolumnsX	38, 295, 344, 346
COMMAND\num@lines	118, 136
COMMAND\numberlinefalse	17
COMMAND\numberlinetrue	17
COMMAND\numberonlyfirstinline	204, 295
COMMAND\numberonlyfirstintwolines	295
COMMAND\numberpstartfalse	16
COMMAND\numberpstarttrue	16, 31, 292, 339, 348
COMMAND\numberstanza	32
COMMAND\numberstanzafalse	42
COMMAND\numberstanzatrue	42
COMMAND\numlabfont	19, 39, 80
COMMAND\one@line	118
COMMAND\onehalfspacing	349
COMMAND\onlyXpstart	295
COMMAND\page@action	83, 92
COMMAND\page@start	83, 339
COMMAND\pagecontents	83
COMMAND\pagelinesep	48

COMMAND\pageno	211
COMMAND\pageparbreak	293
COMMAND\pageref	45, 221
COMMAND\par	23, 29, 122, 183
COMMAND\par@line	118, 136
COMMAND\para@footgroup	147
COMMAND\para@footgroupX	182
COMMAND\para@footsetup	147, 337
COMMAND\para@footsetupX	180, 337, 344
COMMAND\para@vfootnote	151
COMMAND\para@vfootnoteX	181
COMMAND\parafootfmt	149, 150
COMMAND\parafootfmtX	182
COMMAND\parafootftm	152
COMMAND\parafootftmX	183
COMMAND\parafootftmsep	294
COMMAND\parafootsep	295, 342, 347
COMMAND\parafootsepX	37, 85, 294, 295
COMMAND\parafootstart	147
COMMAND\parafootstartX	180
COMMAND\paravfootnote	148
COMMAND\parfillskip	150
COMMAND\parindent	349
COMMAND\parindentX	35
COMMAND\parshape	56
COMMAND\parskip	122
COMMAND\pausenumbering	17, 72, 85, 87, 123, 342, 344
COMMAND\penalty	150
COMMAND\pend	2, 6, 15–18, 20, 53, 54, 101, 104, 106, 112, 118–123, 134, 135, 291, 342, 343
COMMAND\preXnotes	37, 186, 346
COMMAND\preXnotes@	144, 186, 340
COMMAND\prenotesX	37, 187, 346
COMMAND\prepare@preXnotes	186
COMMAND\prev@nopb	285
COMMAND\prev@pb	285
COMMAND\prevlineX	84
COMMAND\prevpageX@num	85
COMMAND\print@Xfootnoterule	346
COMMAND\print@Xnotes	212, 213
COMMAND\print@Xnotes@forpages	345
COMMAND\print@eledsection	125
COMMAND\print@footnoteXrule	346
COMMAND\print@leftmargin@eledsection	274
COMMAND\print@line	124
COMMAND\print@notesX@forpages	345
COMMAND\print@rightmargin@eledsection	274
COMMAND\printendlines	192, 224, 337, 339
COMMAND\printlinefootnote	159, 161, 345
COMMAND\printlinefootnotearea	161, 162, 345
COMMAND\printlinefootnotenumbers	159

COMMAND\printlines	143, 158, 162, 164, 192, 224, 337, 339, 346
COMMAND\printnpnum	23, 292
COMMAND\printpstart	159
COMMAND\protect	105, 291
COMMAND\providecommand	168, 337
COMMAND\pstart	2, 6, 15–18, 20, 52–54, 92, 101, 102, 106, 112, 118–122, 125, 135, 291, 339, 340, 342–344, 346–348
COMMAND\pstartinfootnote	295
COMMAND\pstartinfootnoteeverytime	295
COMMAND\pstartnum	134
COMMAND\pstartref	44, 216, 221, 342
COMMAND\pstarts	340
COMMAND\raggedX	37
COMMAND\raggedleft	36
COMMAND\raggedright	36
COMMAND\raw@text	118
COMMAND\rbracket	34, 39
COMMAND\read@linelist	85–87
COMMAND\ref	45, 49
COMMAND\relax	16, 92, 128, 135, 248, 255, 291
COMMAND\renewcommand	56, 294
COMMAND\resetprevline@	84
COMMAND\resetprevpage@	85
COMMAND\resumenumbering	17, 69, 72, 85, 87, 123, 339, 343, 344
COMMAND\right	51
COMMAND\rightctab	266
COMMAND\rightlinenum	19, 79, 337, 339
COMMAND\rightltab	267
COMMAND\rightnoteupfalse	47
COMMAND\rightrtab	267
COMMAND\rightrightnote	229
COMMAND\rightrightskip	144, 147, 149, 150
COMMAND\rightrightstartnum	134
COMMAND\rigidbalance	153–155
COMMAND\robustify	30
COMMAND\rtab	266, 267, 271
COMMAND\rtabtext	268, 271
COMMAND\sameword	25–27, 112–114, 116, 345, 347
COMMAND\sameword@inedtext	113
COMMAND\saweword	113
COMMAND\scriptsize	80
COMMAND\section	52, 339
COMMAND\section@num	69
COMMAND\sectionmark	273
COMMAND\select@lemmafont	39, 138
COMMAND\series	195, 196
COMMAND\series@	195
COMMAND\seriesatbegin	28, 203, 346
COMMAND\seriesatend	28, 203, 347
COMMAND\set@line	109

COMMAND\set@line@action	83, 92
COMMAND\setapprefprefixmore	46, 294
COMMAND\setapprefprefixsingle	46, 294
COMMAND\setcommand@series	205
COMMAND\sethangingsymbol	42, 243, 294
COMMAND\sethanginsymbol	40
COMMAND\setistwofollowinglines	165
COMMAND\setl@dlprbox	230
COMMAND\setline	19, 20, 84, 88, 91, 101, 105, 120, 348
COMMAND\setlinenum	20, 88, 92, 102, 337
COMMAND\setprintendlines	192, 193, 339
COMMAND\setprintlines	164, 165, 192, 339
COMMAND\setsidenoteseq	47
COMMAND\setsidenotsep	294
COMMAND\setstanzaindent	245
COMMAND\setstanzaindents	41, 245, 291
COMMAND\setstanzapenalties	245
COMMAND\setstanzavalues	245
COMMAND\settoggle@series	205, 340, 344
COMMAND\showlemma	105, 338
COMMAND\showwordrank	27, 114
COMMAND\sidenote@margin	338
COMMAND\sidenotemargin	47, 338, 343
COMMAND\sidenoteseq	294
COMMAND\sidesstartnumtrue	16
COMMAND\skip	144
COMMAND\skipnumbering	20, 95, 102, 339, 347
COMMAND\skipnumbering@reg	347
COMMAND\small	35
COMMAND\special	11
COMMAND\splitmaxdepth	141, 154
COMMAND\splitoff	153
COMMAND\splittopskip	140, 154, 155
COMMAND\stanza	19, 20, 42, 248, 294
COMMAND\stanza@hang	247
COMMAND\stanza@line	247
COMMAND\stanzaindent	41, 245, 345
COMMAND\stanzaindent*	41
COMMAND\stanzaindentbase	244
COMMAND\stanzanumwrapper	42
COMMAND\startlock	19, 83, 102, 248
COMMAND\startstanzahook	294
COMMAND\startsub	19, 83, 101
COMMAND\strip@pt	147
COMMAND\strutbox	154
COMMAND\sub@action	83, 93
COMMAND\sub@lock	82
COMMAND\sub@off	90, 219
COMMAND\sub@on	90, 219
COMMAND\subline@num	81, 83

COMMAND\sublinenum@rep	337
COMMAND\sublinenumberstyle	20, 79, 337
COMMAND\sublinenumincrement	18
COMMAND\sublinenumr@p	79, 337, 339
COMMAND\sublinenumrep	79, 339
COMMAND\sublineref	44, 216, 221
COMMAND\subsectionmark	273
COMMAND\sw@inthisedtext	107
COMMAND\sw@list@inedtext	110, 117
COMMAND\symlinenum	295
COMMAND\symplinum	294
COMMAND\sza@penalty	247
COMMAND>tag	345
COMMAND\text	289
COMMAND\textcolor	57
COMMAND\textheight	56
COMMAND\the	337
COMMAND\thefootnoteA	27
COMMAND\thefootnoteX	341
COMMAND\thelabidx	242
COMMAND\thepage	88
COMMAND\thepstart	16
COMMAND\thepstartL	340
COMMAND\thepstartR	340
COMMAND\thestanza	42
COMMAND>this@line@list@version	98
COMMAND\threecolfootfmt	155
COMMAND\threecolfootfmtX	178
COMMAND\threecolfootgroup	154
COMMAND\threecolfootgroupX	179
COMMAND\threecolfootsetup	154
COMMAND\threecolfootsetupX	178
COMMAND\threecolvfootnote	154
COMMAND\threecolvfootnoteX	178
COMMAND\twocolfootfmtX	176
COMMAND\twocolfootgroupX	177
COMMAND\twocolfootsetupX	176
COMMAND\twocolvfootnoteX	176
COMMAND\twolines	204, 295
COMMAND\twolines@A	204
COMMAND\twolines@B	204
COMMAND\twolines@C	204
COMMAND\twolinesbutnotmore	295
COMMAND\twolinesonlyinsamepage	295
COMMAND\xtbeforeXnotes	295
COMMAND\unhbox	148
COMMAND\unpenalty	149–151
COMMAND\unskip	150
COMMAND\unvxh	150, 296
COMMAND\unvxhX	296

COMMAND\upbracefill	266
COMMAND\usingcritext	290, 293
COMMAND\usingedtext	290, 293
COMMAND\vAfootnote	140
COMMAND\variant	24
COMMAND\vbox	120, 122, 148, 149, 153, 185
COMMAND\vfootnote	140, 144, 148, 154
COMMAND\vl@dbfnote	169, 338
COMMAND\vnumfootnoteX	339
COMMAND\vsizer	38, 56
COMMAND\vsplit	136
COMMAND\waklam	266
COMMAND\waklamec	266
COMMAND\wapunktel	266
COMMAND\wastricht	266
COMMAND\wrap@edcrossref	220, 344
COMMAND\x...	45
COMMAND\xdef	74, 248
COMMAND\xleft@appenditem	74, 104
COMMAND\xlineref	45
COMMAND\xpageref	45
COMMAND\xpstartref	45, 342
COMMAND\xright@appenditem	74
COMMAND\xsublineref	45
COMMAND\xxref	45, 222, 225, 342, 345, 346
COMMAND\zz@@@	337
ENVIRONMENTedarrayc	271
ENVIRONMENTedarrayl	271
ENVIRONMENTedarrayr	271
ENVIRONMENTedtabularc	271
ENVIRONMENTedtabularl	271
ENVIRONMENTedtabularr	271
ENVIRONMENTledgroup	234
ENVIRONMENTledgroupsize	235
PACKAGE(r)(e)ledmac	28
PACKAGEEledmac	10, 59, 83, 239, 292, 293, 345–347
PACKAGEEledpar	346, 347
PACKAGEEtoolbox	61
PACKAGEParallel	297
PACKAGEReledmac	296
PACKAGEamsgen	249
PACKAGEamsmath	249, 250
PACKAGEbabel	58
PACKAGEbiblatex	55
PACKAGEbidi	62
PACKAGEcaption	68
PACKAGEcolor	57
PACKAGEedmac	1, 5, 9–12, 59, 162, 167, 168, 217, 245, 289, 297, 337
PACKAGEedstanza	1, 12, 243
PACKAGEeledmac	1, 9, 12–14, 48, 112, 168, 236, 239, 253, 275, 287, 290, 292–294, 341, 343, 345

PACKAGEeledpar	69, 141, 273, 297, 340, 343–346
PACKAGEetoolbox	73, 112, 195, 204, 211, 229, 274, 284, 285
PACKAGEfootmisc	28, 57, 168, 297
PACKAGEhandout	344
PACKAGEhyperref	45, 217, 241, 278, 342–344
PACKAGEifluatex	61
PACKAGEifxetex	61
PACKAGEimakeidx	48, 55, 62, 236, 239, 293, 341–343, 345
PACKAGEindextool	293
PACKAGEindextools	48, 55, 62, 236, 239, 293, 345
PACKAGEinputenc	114
PACKAGEledarab	58
PACKAGEledmac	1, 9, 12, 58, 73, 239, 289, 290, 293, 296
PACKAGEeledpar	58
PACKAGEMemoir	62, 239, 293, 297, 344
PACKAGEMorewrites	56
PACKAGEMusixtex	343
PACKAGEpolyglossia	39, 58, 108, 139, 159
PACKAGERagged2e	36, 61
PACKAGEReledmac	1, 2, 9, 10, 12–14, 17, 18, 20–22, 24–26, 28, 30, 33, 35–39, 42, 44–46, 48, 49, 54–57, 59, 60, 74, 77, 82, 84, 86, 87, 90, 98, 105, 135, 142, 144, 149, 168, 189, 196, 199, 204, 211, 220, 223, 239, 253, 273, 274, 286, 293, 294, 296, 348
PACKAGEReledpar	1, 3, 5, 7, 13, 16, 38, 54, 55, 58, 60, 69, 75, 85, 90, 107, 142, 145, 184, 185, 196, 211–213, 236, 243, 349
PACKAGESuffix	61
PACKGETabmac	1, 12, 297
PACKAGEuninormalize	25
PACKAGEXargs	24, 61
PACKAGEXkeyval	60
PACKAGEXstring	61, 241

A

\absline@num	1
Abu Kamil Shuja' b. Aslam	11
\actionlines@list	1
\actions@list	1
\add@inserts	1
\add@inserts@next	1
\add@penalties	1
\addtol@denvbody	1
Adelard II	11
\advancelabel@refs	1
\advanceline	1, 19
\advancepageno	1
\Aendnote	22
\affixline@num	1
\affixpstart@num	1
\affixside@note	1
\Afootnote	22
\afternoteX	37

<code>\afterruleX</code>	38
<code>\ampersand</code>	1, 43
<code>\applabel</code>	1, 45
<code>\appref</code>	1, 46
<code>\apprefwithpage</code>	1, 46
<code>\arrangementX</code>	1, 29
<code>\arrangementX@normal</code>	1
<code>\arrangementX@threecol</code>	1
<code>\arrangementX@twocol</code>	1
<code>\at@every@pend</code>	1
<code>\AtEveryPend</code>	1, 16
<code>\AtEveryPstart</code>	1, 16
<code>\autopar</code>	1, 15

B

<code>\ballast</code>	56
<code>\ballast@count</code>	1
Beeton, Barbara Ann Neuhaus Friend	16
<code>\beforeeledchapter</code>	1
<code>\beforenotesX</code>	37
<code>\beginnumbering</code>	1, 14
<code>\Bendnote</code>	22
<code>\Bfootnote</code>	22
<code>\bhooknoteX</code>	36
<code>\bodyfootmarkA</code>	27
<code>\boxfootnotenumbers</code>	1
Bredon, Simon	11
Breger, Herbert	11, 12, 253
Brey, Gerhard	11
Busard, Hubert L. L.	11
<code>\bypage@false</code>	1
<code>\bypage@true</code>	1
<code>\bypstart@false</code>	1
<code>\bypstart@true</code>	1

C

<code>\c@addcolcount</code>	1
<code>\c@ballast</code>	1
<code>\c@firstlinenum</code>	1
<code>\c@firstsublinenum</code>	1
<code>\c@labidx</code>	1
<code>\c@linenumincrement</code>	1
<code>\c@sublinenumincrement</code>	1
<code>\Cendnote</code>	22
<code>\Cfootnote</code>	22
<code>\ch@ck@l@ck</code>	1
<code>\ch@cksub@l@ck</code>	1
<code>\chapter</code>	1
<code>\check@pb@in@verse</code>	1
Chester, Robert of	11

Claassens, Geert H. M.	11
\colalignX	36
Copernicus, Nicolaus	11
\critext	289
\ctab	1
\ctabtext	1

D

Dekker, Dirk-Jan	57
\Dendnote	22
\Dfootnote	22
\disable@familiarnotes	1
\disable@notes	1
\disable@sidenotes	1
\disablel@dtabfeet	1
\do@actions	1
\do@actions@fixedcode	1
\do@actions@next	1
\do@ballast	1
\do@feetX	1
\do@insidelinehook	1
\do@line	1
\do@linehook	1
\do@lockoff	1
\do@lockoffL	1
\do@lockon	1
\do@lockonL	1
\doedindexlabel	1
\doendnotes	1, 22
\doendnotesbysection	1, 22
\doinsidelinehook	1, 20
\dolinehook	1, 20
\dosplits	1
Downes, Michael	56, 148, 150
\doxtrafeet	1
\dummy@edtext	1
\dummy@edtext@showlemma	1
\dummy@ref	1

E

\edaftertab	1, 52, 266
edarrayc (environment)	49
edarrayl (environment)	49
edarrayr (environment)	49
\edatleft	1, 51
\edatright	1, 51
\edbeforetab	1, 52, 266
\edfilldimen	1
\edfont@info	1
\EDINDEX	1

<code>\edindex</code>	1, 47
<code>\edindexlab</code>	1, 49
<code>\EDLABEL</code>	1
<code>\edlabel</code>	1, 44
<code>\edlineref</code>	1, 44
<code>\edmakelabel</code>	1, 45
<code>\edpageref</code>	1, 44
<code>\edrowfill</code>	1, 50
<code>\EDTAB</code>	1
<code>\edtabcolsep</code>	1, 50
<code>\EDTABINDENT</code>	1
<code>\edtabindent</code>	1
<code>\EDTABtext</code>	1
<code>edtabularc (environment)</code>	49
<code>edtabularl (environment)</code>	49
<code>edtabularr (environment)</code>	49
<code>\EDTEXT</code>	1
<code>\edtext</code>	1, 21
<code>\edvertdots</code>	1, 52
<code>\edvertline</code>	1, 52
<code>\Endnote</code>	22
<code>\Efootnote</code>	22
<code>\eled@chapter</code>	1
<code>\eled@section</code>	1
<code>\eled@sectioning@out</code>	1
<code>\eled@subsection</code>	1
<code>\eled@subsubsection</code>	1
<code>\eledchapter</code>	1
<code>\eledchapter*</code>	1
<code>\eledsection</code>	1
<code>\eledsection*</code>	1
<code>\eledsubsection</code>	1
<code>\eledsubsection*</code>	1
<code>\eledsubsubsection</code>	1
<code>\eledsubsubsection*</code>	1
<code>\enablel@dtabfeet</code>	1
<code>\end@lemmas</code>	1
<code>\endashchar</code>	1, 39
<code>\endline@num</code>	1
<code>\endlock</code>	1, 19
<code>\endminipage</code>	1
<code>\endnumbering</code>	1, 14
<code>\endpage@num</code>	1
<code>\endprint</code>	1
<code>\endquotation</code>	1
<code>\endquote</code>	1
<code>\endsub</code>	1, 19
<code>\endsubline@num</code>	1
environments:	
<code>edarrayc</code>	49

edarrayl	49
edarrayr	49
edtabularc	49
edtabularl	49
edtabularr	49
ledgroup	43
ledgroupsize	43
minipage	43
Euclid	11
\extensionchars	<u>1</u> , 55

F

\f@x@l@cks	<u>1</u>
Fairbairns, Robin	28
\first@linenum@out@false	<u>1</u>
\first@linenum@out@true	<u>1</u>
\firstlinenum	<u>1</u> , 18
\firstseriesX@	<u>1</u>
\firstsublinenum	<u>1</u> , 18
\firstXseries@	<u>1</u>
\fix@page	<u>1</u>
\flag@end	<u>1</u>
\flag@start	<u>1</u>
\flagstanza	<u>1</u> , 43
\flush@notes	<u>1</u>
\fnpos	<u>1</u> , 28
Folkerts, Menso	11
\footfootmarkA	27
\footfudgefiddle	<u>1</u> , 56
\footnoteA	27
\footnoteB	27
\footnoteC	27
\footnoteD	27
\footnoteE	27
\footnotelang@lua	<u>1</u>
\footnotelang@poly	<u>1</u>
\footnoteoptions@	<u>1</u>
\footsplitskips	<u>1</u>
\fulllines@	<u>1</u>
\fullstop	<u>1</u> , 39

G

Gädeke, Nora	11
\get@edindex@hyperref	<u>1</u>
\get@edindex@ledinnote@command	<u>1</u>
\get@index@command	<u>1</u>
\get@linelistfile	<u>1</u>
\get@sw@txt	<u>1</u>
\getline@num	<u>1</u>
\gl@p	<u>1</u>

H

\h@num	1
\hangindentX	35
\hidenumbering	1, 20
\Hilfsbox	1
\hilfsbox	1
\hilfscount	1
\HILFSskip	1
\Hilfsskip	1
\hilfsskip	1
\hsizethreecolX	36
\hsizetwocolX	36
\hyperlinkformat	1
\hyperlinkformatR	1
\hyperlinkR	1

I

\if@addsw	1
\if@edindex@fornote@true	1
\if@eled@sectioning	1
\if@led@nofoot	1
\if@lemmacommand@	1
\if@noeled@sec	1
\if@noneed@Footnote	1
\if@RTL	1
\ifautopar@pause	1
\ifbypage@	1
\ifbypage@R	1
\ifbypstart@	1
\ifbypstart@R	1
\ifeledmaccompat@	1
\iffirst@linenum@out@	1
\ifinserthangingsymbol	1
\ifinstanza	1
\ifl@d@dash	1
\ifl@d@elin	1
\ifl@d@esl	1
\ifl@d@pnum	1
\ifl@d@ssub	1
\ifl@d@Xmorethantwolines	1
\ifl@d@Xtwolines	1
\ifl@dend@X	1
\ifl@dhidenum	1
\ifl@dmemoir	1
\ifl@dpaging	1
\ifl@dpairing	1
\ifl@dprintingcolumns	1
\ifl@dprintingpages	1
\ifl@dskipnumber	1
\ifl@dskipversenumber	1

<code>\ifl@dstartendok</code>	1
<code>\ifl@imakeidx</code>	1
<code>\ifl@indextools</code>	1
<code>\ifledfinal</code>	1, 55
<code>\ifledgroupnotesL@</code>	1
<code>\ifledgroupnotesR@</code>	1
<code>\iflednopbinverse</code>	1
<code>\ifledRcol</code>	1
<code>\ifledRcol@</code>	1
<code>\ifnocritical@</code>	1
<code>\ifnoend@</code>	1
<code>\ifnofamiliar@</code>	1
<code>\ifnoledgroup@</code>	1
<code>\ifnoquotation@</code>	1
<code>\ifnoteschanged@</code>	1
<code>\ifnumberedpar@</code>	1
<code>\ifnumbering</code>	1
<code>\ifnumberingR</code>	1
<code>\ifnumberline</code>	1
<code>\ifnumberstanza</code>	1
<code>\ifparapparatus@</code>	1
<code>\ifparledgroup</code>	1
<code>\ifpst@rtedL</code>	1
<code>\ifseriesbefore</code>	1
<code>\ifsidepstartnum</code>	1
<code>\ifsublines@</code>	1
<code>\ifwidthliketwocolumns</code>	1
<code>\ifXendinsertsep@</code>	1
<code>\ifxindy@</code>	1
<code>\ifxindyhyperref@</code>	1
<code>\initnumbering@quote</code>	1
<code>\initnumbering@reg</code>	1
<code>\insert@count</code>	0, 1
<code>\inserthangingymbol</code>	1
<code>\insertlines@list</code>	1
<code>\inserts@list</code>	1

J

Jayaditya	11
-----------------	----

K

Kabelschacht, Alois	138
---------------------------	-----

L

<code>\l@advance@parledgroup@beforenormalnotes</code>	1
<code>\l@d@add</code>	1
<code>\l@d@nums</code>	1
<code>\l@d@section</code>	1
<code>\l@d@set</code>	1
<code>\l@d@Xend</code>	1

<code>\l@dampcount</code>	1
<code>\l@dbfnote</code>	1
<code>\l@dcheckcols</code>	1
<code>\l@dcheckstartend</code>	1
<code>\l@dchset@num</code>	1
<code>\l@dcolcount</code>	1
<code>\l@dcollect@body</code>	1
<code>\l@dcollect@body</code>	1
<code>\l@dcolwidth</code>	1
<code>\l@dcsnote</code>	1
<code>\l@dcsnotetext</code>	1
<code>\l@dcsnotetext@l</code>	1
<code>\l@dcsnotetext@r</code>	1
<code>\l@ddodoreinextrafeet</code>	1
<code>\l@dedbeginmini</code>	1
<code>\l@dedendmini</code>	1
<code>\l@emptyd@ta</code>	1
<code>\l@dend@close</code>	1
<code>\l@dend@open</code>	1
<code>\l@dend@stuff</code>	1
<code>\l@dend@Xfalse</code>	1
<code>\l@dend@Xtrue</code>	1
<code>\l@denvbody</code>	1
<code>\l@dfambeginmini</code>	1
<code>\l@dfamendmini</code>	1
<code>\l@dfeetbeginmini</code>	1
<code>\l@dfeetendmini</code>	1
<code>\l@dgetline@margin</code>	1
<code>\l@dgetlock@disp</code>	1
<code>\l@dgetref@num</code>	1
<code>\l@dgetsidenote@margin</code>	1
<code>\l@dobblearg</code>	1
<code>\l@dlabel@parse</code>	1
<code>\l@dld@ta</code>	1
<code>\l@dlp@rbox</code>	1
<code>\l@dlsn@te</code>	1
<code>\l@dlsnote</code>	1
<code>\l@dmake@labels</code>	1
<code>\l@dmodfordtext</code>	1
<code>\l@dnullfills</code>	1
<code>\l@dnumpstartsL</code>	1
<code>\l@doldold@footnotetext</code>	1
<code>\l@dp@rsefootspec</code>	1
<code>\l@dpagingfalse</code>	1
<code>\l@dpagingtrue</code>	1
<code>\l@dpairingfalse</code>	1
<code>\l@dpairingtrue</code>	1
<code>\l@dparsedendline</code>	1
<code>\l@dparsedendpage</code>	1
<code>\l@dparsedendsub</code>	1

<code>\l@dparsedstartline</code>	1
<code>\l@dparsedstartpage</code>	1
<code>\l@dparsedstartsub</code>	1
<code>\l@dparsefootspec</code>	1
<code>\l@dpprintingcolumnfalse</code>	1
<code>\l@dpprintingcolumnstrue</code>	1
<code>\l@dpprintingpagesfalse</code>	1
<code>\l@dpprintingpagestrue</code>	1
<code>\l@dpush@begins</code>	1
<code>\l@drd@ta</code>	1
<code>\l@dref@undefined</code>	1
<code>\l@drestorefills</code>	1
<code>\l@dstoreforedtext</code>	1
<code>\l@drp@rbox</code>	1
<code>\l@drsn@te</code>	1
<code>\l@drsnote</code>	1
<code>\l@dsetmaxcolwidth</code>	1
<code>\l@dskipnumberfalse</code>	1
<code>\l@dskipnumbertrue</code>	1
<code>\l@dtabaddcols</code>	1
<code>\l@dtabnoexpands</code>	1
<code>\l@dunboxmpfoot</code>	1
<code>\l@dunhbox@line</code>	1
<code>\l@dzeropenalties</code>	1
<code>\label</code>	45
<code>\labelstartfalse</code>	1
<code>\labelstarttrue</code>	1, 16
<code>\labelref@list</code>	1
<code>\labelrefsparseline</code>	1
<code>\labelrefsparsesubline</code>	1
<code>\last@page@num</code>	1
Lavagnino, John	10
<code>\led@check@nopb</code>	1
<code>\led@check@pb</code>	1
<code>\led@err@AutoparNotNumbered</code>	1
<code>\led@err@edtextoutsidepstart</code>	1
<code>\led@err@EdtextWithoutFootnote</code>	1
<code>\led@err@FootnoteWithoutEdtext</code>	1
<code>\led@err@HighEndColumn</code>	1
<code>\led@err@LineationInNumbered</code>	1
<code>\led@err@LowStartColumn</code>	1
<code>\led@err@ManyLeftnotes</code>	1
<code>\led@err@ManyRightnotes</code>	1
<code>\led@err@ManySidenotes</code>	1
<code>\led@err@NumberingNotStarted</code>	1
<code>\led@err@NumberingShouldHaveStarted</code>	1
<code>\led@err@NumberingStarted</code>	1
<code>\led@err@NumberingWithoutPstart</code>	1
<code>\led@err@PendNoPstart</code>	1
<code>\led@err@PendNotNumbered</code>	1

<code>\led@err@PstartInPstart</code>	1
<code>\led@err@PstartNotNumbered</code>	1
<code>\led@err@ReverseColumns</code>	1
<code>\led@err@TooManyColumns</code>	1
<code>\led@err@UnequalColumns</code>	1
<code>\led@error@fail@patch@@docclearpage</code>	1
<code>\led@error@fail@patch@@iiiminipage</code>	1
<code>\led@error@fail@patch@@makecol</code>	1
<code>\led@error@fail@patch@@reinserts</code>	1
<code>\led@error@fail@patch@endminipage</code>	1
<code>\led@error@ImakeidxAfterEledmac</code>	1
<code>\led@error@IndextoolsAfterEledmac</code>	1
<code>\led@mess@NotesChanged</code>	1
<code>\led@mess@SectionContinued</code>	1
<code>\led@nopb</code>	1
<code>\led@nopbnum</code>	1
<code>\led@pb</code>	1
<code>\led@pb@setting</code>	1
<code>\led@pbnum</code>	1
<code>\led@toksa</code>	1
<code>\led@toksb</code>	1
<code>\led@warn@AppLabelOutEdtext</code>	1
<code>\led@warn@BadAction</code>	1
<code>\led@warn@BadAdvancelineLine</code>	1
<code>\led@warn@BadAdvancelineSubline</code>	1
<code>\led@warn@BadLineation</code>	1
<code>\led@warn@BadLinenummargin</code>	1
<code>\led@warn@BadLockdisp</code>	1
<code>\led@warn@BadSetline</code>	1
<code>\led@warn@BadSetlinenum</code>	1
<code>\led@warn@BadSidenotemargin</code>	1
<code>\led@warn@BadSublockdisp</code>	1
<code>\led@warn@DuplicateLabel</code>	1
<code>\led@warn@LineFileObsolete</code>	1
<code>\led@warn@NoIndexFile</code>	1
<code>\led@warn@NoLineFile</code>	1
<code>\led@warn@NoMarginpars</code>	1
<code>\led@warn@RefUndefined</code>	1
<code>\led@warn@SeriesStillExist</code>	1
<code>ledgroup (environment)</code>	43
<code>ledgroupsize (environment)</code>	43
<code>\ledinnernote</code>	1, 46
<code>\ledinnote</code>	1
<code>\ledinnotehyperpage</code>	1
<code>\ledinnotemark</code>	1
<code>\ledleftnote</code>	1, 46
<code>\ledlinenum</code>	1
<code>\ledllfill</code>	1
<code>\ledlsnotefontsetup</code>	1, 47
<code>\ledlsnotesep</code>	1, 47

<code>\ledlsnotewidth</code>	1, 47
<code>\lednopb</code>	1, 54
<code>\lednopbinversettrue</code>	54
<code>\lednopbnum</code>	1
<code>\ledouternote</code>	46
<code>\ledouterote</code>	1
<code>\ledpb</code>	1, 54
<code>\ledpbnum</code>	1
<code>\ledpbsetting</code>	1, 54
<code>\ledrightnote</code>	1, 46
<code>\ledrlfill</code>	1
<code>\ledrsnotefontsetup</code>	1, 47
<code>\ledrsnotesep</code>	1, 47
<code>\ledrsnotewidth</code>	1, 47
<code>\ledsectnomark</code>	1
<code>\ledsectnotoc</code>	1
<code>\ledsetnormalparstuff@common</code>	1
<code>\ledsetnormalparstuffX</code>	1
<code>\ledsidenote</code>	1, 46
<code>\leftctab</code>	1
<code>\leftlinenum</code>	1, 19
<code>\leftltab</code>	1
<code>\leftnoteupfalse</code>	47
<code>\leftpstartnum</code>	1
<code>\leftrtab</code>	1
<code>Leibniz</code>	11
<code>\lemma</code>	1, 23
<code>\letsforverteilen</code>	1
<code>\line@list</code>	1
<code>\line@list@stuff</code>	1
<code>\line@list@version</code>	1
<code>\line@margin</code>	1
<code>\line@num</code>	1
<code>\line@set</code>	1
<code>\lineation</code>	1, 18
<code>\linenum</code>	1, 24
<code>\linenum@out</code>	1
<code>\linenumberlist</code>	1, 18
<code>\linenumberstyle</code>	1, 20
<code>\linenumincrement</code>	1, 18
<code>\linenummargin</code>	1, 18
<code>\linenumr@p</code>	1
<code>\linenumrep</code>	1
<code>\linenumsep</code>	1, 19
<code>\list@clear</code>	1
<code>\list@clearing@reg</code>	1
<code>\list@create</code>	1
<code>\lock@disp</code>	1
<code>\lock@off</code>	1
<code>\lock@on</code>	1

<code>\lockdisp</code>	1, 19
Lorch, Richard	11
<code>\ltab</code>	1
<code>\ltabtext</code>	1
Luecking, Dan	61

M

<code>\m@mmf@check</code>	1
<code>\m@mmf@prepare</code>	1
<code>\M@sect</code>	1
<code>\makehboxofhboxes</code>	1
<code>\managestanza@modulo</code>	1
<code>\maxhnotesX</code>	38
Mayer, Gyula	11
<code>\measurebody</code>	1
<code>\measurecell</code>	1
<code>\measuremrow</code>	1
<code>\measuretbody</code>	1
<code>\measuretcell</code>	1
<code>\measuretrow</code>	1
Middleton, Thomas	11, 81
<code>minipage</code> (environment)	43
Mittelbach, Frank	11
<code>\morenoexpands</code>	1, 57
<code>\mpfnpos</code>	1, 28
<code>\mpnormalfootgroup</code>	1
<code>\mpnormalfootgroupX</code>	1
<code>\mpnormalvfootnote</code>	1
<code>\mpnormalvfootnoteX</code>	1
<code>\mppara@footgroupX</code>	1
<code>\mppara@vfootnoteX</code>	1
<code>\mpparafootgroup</code>	1
<code>\mpparavfootnote</code>	1
<code>\mpthreecolfootgroup</code>	1
<code>\mpthreecolfootgroupX</code>	1
<code>\mpthreecolfootsetup</code>	1
<code>\mpthreecolfootsetupX</code>	1
<code>\mptwocolfootgroup</code>	1
<code>\mptwocolfootgroupX</code>	1
<code>\mptwocolfootsetup</code>	1
<code>\mptwocolfootsetupX</code>	1
<code>\multfootsep</code>	1, 28
<code>\multiplefootnotemarker</code>	1

N

<code>\n@num</code>	1
<code>\n@num@stanza</code>	1
<code>\new@line</code>	1
<code>\newhookcommand@series</code>	1
<code>\newhookcommand@series@reload</code>	1

<code>\newhooktoggle@series</code>	1
<code>\newhooktoggle@series@reload</code>	1
<code>\newseries@</code>	1
<code>\newverse</code>	1
<code>\NEXT</code>	1
<code>\no@expands</code>	1
<code>\noeledsec</code>	54
<code>\nomk@</code>	1
<code>\nonum@</code>	1
<code>\normal@footnotemarkX</code>	1
<code>\normal@page@break</code>	1
<code>\normal@pars</code>	1
<code>\normalbfnoteX</code>	1
<code>\normalbodyfootmarkX</code>	1
<code>\normalfootfmt</code>	1
<code>\normalfootfmtX</code>	1
<code>\normalfootfootmarkX</code>	1
<code>\normalfootgroup</code>	1
<code>\normalfootgroupX</code>	1
<code>\normalfootnoterule</code>	1
<code>\normalfootnoteruleX</code>	1
<code>\normalfootstart</code>	1
<code>\normalfootstartX</code>	1
<code>\normalvfootnote</code>	1
<code>\normalvfootnoteX</code>	1
<code>\nosep@</code>	1
<code>\notefontsizeX</code>	35
<code>\notenumfontX</code>	35
<code>\noteschanged@false</code>	1
<code>\noteschanged@true</code>	1
<code>\noteswidthliketwocolumnsX</code>	38
<code>\nulledindex</code>	1
<code>\nullsetzen</code>	1
<code>\num@lines</code>	1
<code>\numberedpar@false</code>	1
<code>\numberedpar@true</code>	1
<code>\numberingfalse</code>	1
<code>\numberingtrue</code>	1
<code>\numberlinefalse</code>	17
<code>\numberlinetrue</code>	17
<code>\numberpstartfalse</code>	1, 16
<code>\numberpstarttrue</code>	1, 16
<code>\numberstanzafalse</code>	42
<code>\numberstanzatrue</code>	42
<code>\numlabfont</code>	1, 39

O

<code>\old@hsize</code>	1
<code>\one@line</code>	1

P

\page@action	1
\page@num	1
\pagelinesep	1, 48
\pageno	1
\pageref	45
\par@line	1
\para@footgroupX	1
\para@footsetup	1
\para@footsetupX	1
\para@vfootnoteX	1
\parafootfmt	1
\parafootfmtX	1
\parafootgroup	1
\parafootsepX	37
\parafootstart	1
\parafootstartX	1
\paravfootnote	1
\parindentX	35
\pausenumbering	1, 17
\pend	1, 15
Plato of Tivoli	11
\postbodyfootmark	1
\prebodyfootmark	1
\prenotesX	37
\prepare@edindex@fornote	1
\prepare@prenotesX	1
\prepare@preXnotes	1
\prev@nopb	1
\prev@pb	1
\prevpage@num	1
\preXnotes	1, 37
\preXnotes@	1
\print@eledsection	1
\print@footnoteXrule	1
\print@leftmargin@eledsection	1
\print@line	1
\print@notesX	1
\print@rightmargin@eledsection	1
\print@Xfootnoterule	1
\print@Xnotes	1
\printendlines	1
\printlineendnote	1
\printlineendnotearea	1
\printlinefootnote	1
\printlinefootnotearea	1
\printlinefootnotenumbers	1
\printlines	1
\printnpnum	1
\printpstart	1

<code>\printsymlineendnotearea</code>	<u>1</u>
<code>\printsymlinefootnotearea</code>	<u>1</u>
<code>\printXafternumber</code>	<u>1</u>
<code>\printXbeforenumber</code>	<u>1</u>
<code>\pst@rtedLfalse</code>	<u>1</u>
<code>\pst@rtedLtrue</code>	<u>1</u>
<code>\pstart</code>	<u>1</u> , 15
<code>\pstarteref</code>	<u>1</u>
<code>\pstartnum</code>	<u>1</u>
<code>\pstartref</code>	44

Q

<code>\quotation</code>	<u>1</u>
<code>\quote</code>	<u>1</u>

R

<code>\raggedX</code>	37
<code>\raw@text</code>	<u>1</u>
<code>\rbracket</code>	<u>1</u> , 39
<code>\read@linelist</code>	<u>1</u>
<code>\ref</code>	45
<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>\resetprevline@</code>	<u>1</u> , 84
<code>\resetprevpage@</code>	<u>1</u>
<code>\resetprevpage@num</code>	85
<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> , 17
<code>\rightctab</code>	<u>1</u>
<code>\rightlinenum</code>	<u>1</u> , 19
<code>\rightltab</code>	<u>1</u>
<code>\rightnoteupfalse</code>	47
<code>\rightrtab</code>	<u>1</u>
<code>\rightstartnum</code>	<u>1</u>
<code>\rigidbalance</code>	<u>1</u>
<code>\rtab</code>	<u>1</u>
<code>\rtabtext</code>	<u>1</u>

S

<code>Sacrobosco</code>	11
<code>\sameword</code>	<u>1</u> , 25
<code>\sameword@inedtext</code>	<u>1</u>
Schöpf, Rainer	11
<code>\section@num</code>	<u>1</u>
<code>\select@lemmafонт</code>	<u>1</u>
<code>\select@lemmafонт</code>	<u>1</u> , 39

<code>\series</code>	1
<code>\seriesatbegin</code>	1, 28
<code>\seriesatend</code>	1, 28
<code>\set@line</code>	1
<code>\set@line@action</code>	1
<code>\setapprefprefixmore</code>	1, 46
<code>\setapprefprefixsingle</code>	1, 46
<code>\setcommand@series</code>	1
<code>\sethangingsymbol</code>	1, 42
<code>\setistwofollowinglines</code>	1
<code>\setl@dlp@rbox</code>	1
<code>\setl@drpr@box</code>	1
<code>\setline</code>	1, 19
<code>\setlinenum</code>	1, 20
<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>\setnoteswidthliketwocolumnsX@</code>	1
<code>\setnotesXpositionliketwocolumns@</code>	1
<code>\setprintendlines</code>	1
<code>\setprintlines</code>	1
<code>\setsidenotesep</code>	47
<code>\setstanzaindents</code>	1, 40
<code>\setstanzapenalties</code>	1, 41
<code>\setstanzavalues</code>	1
<code>\settcellcenter</code>	1
<code>\settcellleft</code>	1
<code>\settcellright</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, 55
<code>\showwordrank</code>	1, 27
<code>\sidenote@margin</code>	1
<code>\sidenotemargin</code>	1, 46
<code>\sidepstartnumtrue</code>	16
<code>\skip@lockoff</code>	1
<code>\skipnumbering</code>	1, 20
<code>\splitoff</code>	1
<code>\spreadmath</code>	1, 50
<code>\spreadtext</code>	1, 50
<code>\stanza</code>	1, 40
<code>\stanza@count</code>	1
<code>\stanza@chang</code>	1

<code>\stanza@line</code>	1
<code>\stanzaindent</code>	1, 41
<code>\stanzaindent*</code>	1, 41
<code>\stanzaindentbase</code>	1, 40
<code>\stanzanumwrapper</code>	1, 42
<code>\startlock</code>	1, 19
<code>\startsub</code>	1, 19
<code>\stepl@dcollcount</code>	1
<code>\strip@szacnt</code>	1
<code>\sub@action</code>	1
<code>\sub@lock</code>	1
<code>\sub@off</code>	1
<code>\sub@on</code>	1
<code>\subline@num</code>	1
<code>\sublinenumberstyle</code>	1, 20
<code>\sublinenumincrement</code>	1, 18
<code>\sublinenumr@p</code>	1
<code>\sublinenumrep</code>	1
<code>\sublineref</code>	1, 44
<code>\sublines@false</code>	1
<code>\sublines@true</code>	1
<code>\sublock@disp</code>	1
<code>\sublockdisp</code>	1
Sullivan, Wayne	11, 12, 56, 68, 72, 148, 149, 217, 243
<code>\sza@penalty</code>	1

T

<code>\tabHilfbox</code>	1
<code>\tabhilfbox</code>	1
<code>\theadcolcount</code>	1
<code>\theendpageline</code>	1
<code>\thefootnoteA</code>	27
Theodosius	11
<code>\thepageline</code>	1
<code>\thepstart</code>	1, 16
<code>\thestanza</code>	1, 42
<code>\thestartpageline</code>	1
<code>\this@line@list@version</code>	1
<code>\threecolfootfmt</code>	1
<code>\threecolfootfmtX</code>	1
<code>\threecolfootgroup</code>	1
<code>\threecolfootgroupX</code>	1
<code>\threecolfootsetup</code>	1
<code>\threecolfootsetupX</code>	1
<code>\threecolvfootnote</code>	1
<code>\threecolvfootnoteX</code>	1
<code>\twocolfootfmt</code>	1
<code>\twocolfootfmtX</code>	1
<code>\twocolfootgroup</code>	1
<code>\twocolfootgroupX</code>	1

<code>\twocolfootsetup</code>	1
<code>\twocolfootsetupX</code>	1
<code>\twocolvfootnote</code>	1
<code>\twocolvfootnoteX</code>	1

U

<code>\unvxhX</code>	1
----------------------------	---

V

Vamana	11
<code>\variab</code>	1
<code>\vbfnoteX</code>	1
<code>\vl@dbfnote</code>	1
<code>\vl@dcnote</code>	1
<code>\vl@dlsnote</code>	1
<code>\vl@drsnote</code>	1
<code>\vnumfootnoteX</code>	1

W

Whitney, Ron	11
<code>\wrap@edcrossref</code>	1
Wujastyk, Dominik	10

X

<code>\X@doreinfeet</code>	1
<code>\Xafterlemmaseparator</code>	34
<code>\Xafternote</code>	37
<code>\Xafternumber</code>	32
<code>\Xafterrule</code>	38
<code>\Xaftersymlinenum</code>	32
<code>\Xarrangement</code>	1, 29
<code>\Xarrangement@normal</code>	1
<code>\Xarrangement@paragraph</code>	1
<code>\Xarrangement@threecol</code>	1
<code>\Xarrangement@twocol</code>	1
<code>\Xbeforelemmaseparator</code>	34
<code>\Xbeforenotes</code>	37
<code>\Xbeforenumber</code>	32
<code>\Xbeforesymlinenum</code>	32
<code>\Xbhooknote</code>	36
<code>\Xboxlinenum</code>	32
<code>\Xboxlinenumalign</code>	33
<code>\Xboxsymlinenum</code>	33
<code>\Xcolalign</code>	36
<code>\Xdo@feet</code>	1
<code>\xedindex</code>	1
<code>\xedlabel</code>	1
<code>\xedtext</code>	1
<code>\Xendafterrenumber</code>	32
<code>\Xendafterlemmaseparator</code>	34

\Xendafternote	38
\Xendaftersymlinenum	32
\Xendahookinplaceofnumber	33
\Xendahooklinenum	33
\Xendbeforelemmaseparator	34
\Xendbeforenumber	32
\Xendbeforesymlinenum	32
\Xendbhookinplaceofnumber	33
\Xendbhooklinenum	33
\Xendbhooknote	36
\Xendboxendlinenumalign	33
\Xendboxlinenum	33
\Xendboxlinenumalign	33
\Xendboxstartlinenumalign	33
\Xendboxsymlinenum	33
\Xendhangindent	35
\Xendinplaceoflemmaseparator	34
\Xendinplaceofnumber	32
\Xendlemmadisablefontselection	35
\Xendlemmaseparator	34
\Xendmorethantwolines	31
\Xendmorethantwolinesapprefwithpage	46
\Xendnonumber	31
\Xendnotefontsize	35
\Xendnotenumfont	34
\Xendnumberonlyfirstinline	30
\Xendnumberonlyfirstintwolines	30
\Xendparagraph	38
\Xendsep	38
\Xendsymlinenum	30
\Xendtwolines	31
\Xendtwolinesapprefwithpage	46
\Xendtwolinesbutnotmore	31
\Xendtwolinesbutnotmoreapprefwithpage	46
\Xendtwolinesonlyinsamepageapprefwithpage	46
\Xhangindent	35
\Xhsizethreecol	36
\Xhsizetwocol	36
\Xinplaceoflemmaseparator	34
\Xinplaceofnumber	32
\Xinsertparafootsep	1
\Xledsetnormalparstuff	1
\xleft@appenditem	1
\Xlemmadisablefontselection	35
\Xlemmaseparator	34
\xlineref	1, 44
\Xmaxhnotes	38
\Xmorethantwolines	30
\Xmorethantwolinesappref	46
\Xnolemmaseparator	1, 34

\Xnonbreakableafternumber	32
\Xnonumber	31
\Xnotefontsize	35
\Xnotenumfont	34
\Xnoteswidthliketwocolumns	38
\Xnumberonlyfirstinline	30
\Xnumberonlyfirstintwolines	30
\Xonlypstart	31
\xpageref	<u>1</u> , 44
\Xparafootsep	37
\Xparindent	35
\Xpstart	31
\Xpstarteverytime	31
\xpstartref	<u>1</u> , 44
\Xragged	37
\xright@appenditem	<u>1</u>
\Xstanza	32
\Xstanzaseparator	32
\xsublineref	<u>1</u> , 44
\Xsymlinenum	30
\Xtwolines	30
\Xtwolinesappref	46
\Xtwolinesbutnotmoreappref	46
\Xtwolinesonlyinsamepage	31, 46
\Xtxtbeforenotes	37
\Xunvxh	<u>1</u>
\xxref	<u>1</u> , 45
Z	
\zz@@@	<u>1</u>

Change History

v0.1.0.	
General: First public release	1
v0.2.0.	
General: Added tabmac code, and extended indexing	1
\ifl@dmemoir: Added \ifl@dmemoir for memoir class having been used	62
\morenoexpands: Added \l@dtabnoexpands to \no@expands	106
\reledmac@error: Added \eledmac@error and replaced error messages	63
v0.2.1.	
\@lab: Removed page setting from \@lab	219
General: Added text about normal labeling	45
Bug fixes and match with mempatch v1.8	1
Major changes to insert code when memoir is loaded	214
\dextrafeet: Renamed \dextrafeet to \l@ddextrafeet	211
\edlabel: Tweaked \edlabel to get correct page numbers	217
\l@ddodoreinxtrafeet: Renamed \dodoreinxtrafeet to \l@ddodoreinxtrafeet	212
\morenoexpands: Removed some \lets from \no@expands. These were in edmac but Peter Wilson feels that they should not have been as they disabled page/line refs in a footnotes	106
\zz@@@: Minor change to \zz@@@	216
v0.2.2.	
General: Improved paragraph footnotes	1
New Dekker example	1
Used \providecommand for \@gobblethree to avoid clash with the amsfonts pack- age	68
\footfudgefiddle: Added \footfudgefiddle	147
\line@list@stuff: Added initial write of page number in \line@list@stuff	99
\para@footsetup: Added \footfudgefiddle to \para@footsetup	147
\para@footsetupX: Added \footfudgefiddle to \para@footsetupX	180
v0.3.0.	
\@lab: Replaced \the\line@num by \linenumr@p\line@num in \@lab, and similar for sub-lines	219
\@nl@reg: Added a bunch of code to \@nl for handling \setlinenum	88
General: Includes edstanza and more	1
\ledlinenum: Added \linenumr@p and \sublinenum@rep to \leftlinenum and \rightlinenum	80
\linenumberlist: Added \linenumberlist mechanism	68
\printendlines: Added \linenumr@p and \sublinenumr@p to \printendlines	193
\printlines: Added \linenumr@p and \sublinenumr@p to \printlines	167
\sublinenumr@p: Added \linenumberstyle and \sublinenumberstyle	79
v0.3.1.	
General: Not released. Added remarks about the parallel package	1
v0.4.0.	
\@iiiminipage: Modified kernel \@iiiminipage and \endminipage to cater for criti- cal footnotes	233
General: Added final/draft options	60
Added ledgroup environment	234
Added ledgroupsized environment	235
Added minipage, etc., support	1

\edtext: Added \showlemma to \edtext	106
\l@dfeetendmini: Added \l@dfeetbeginmini, \l@dfeetendmini and all their supporting code	232
\mpnormalfootgroup: Added \mpnormalfootgroup	145
\mpnormalvfootnote: Added \mpnormalvfootnote	143
\showlemma: Added \showlemma	68
\Xarrangement@normal: Added minpage footnote setup to \footnormal	142
v0.4.1.	
General: Added code for changing \@docclearpage	215
Not released. Minor editorial improvements and code tweaks	1
Only change \@footnotetext and \@footnotemark if memoir not used	168
\edindex: Let eledmac take advantage of memoir's indexing	239
\print@Xnotes: Added \@opXfeet	212
\Xdo@feet: Changed \Xdo@feet code for easier extensions	212
v0.5.0.	
\@footnotetext: Enabled regular \footnote in numbered text	169
\@xympar: Eliminated \marginpar disturbance	225
General: Added left and right side notes	226
Added sidenotes, familiar footnotes in numbered text	1
v0.5.1.	
General: Added moveable side note	226
Fixed right line numbers killed in v0.5	1
Only change \hsize in ledgroupsized environment otherwise page number can be in wrong place	235
\affixline@num: Changed \affixline@num to cater for sidenotes	130
\l@dgetsidenote@margin: Added \sidenotemargin and \sidenote@margin	226
v0.6.0.	
\@lopR: Added \@pend, \@pendR, \@lopL and \@lopR in anticipation of parallel processing	90
\@nl@reg: Added \fix@page to \@nl	88
Extended \@nl to include the page number	88
General: Fixed long paragraphs looping	1
Fixed minor typos	1
Prepared for eledpar package	1
\fix@page: Added \last@page@num and \fix@page	89
\new@line: Extended \new@line to output page numbers	99
\l@dbfnote: Changed \l@dbfnote and \l@dbfnote as originals could give incorrect markers in the footnotes	169
v0.7.0.	
\@nl@reg: Added \@nl@reg	88
\@ref@reg: Added \@ref@reg	96
General: eledmac having been available for 2 years, deleted the commented out original edmac texts	1
Maïeul Rouquette new maintainer	1
Made macros of all messages	63
Replaced all \interfootnotelinepenalty, etc., by just \interfootnotelinepenalty	1
Tidying up for eledpar and ledarab packages	1
\affixline@num: Added skipnumering to \affixline@num	130
\do@actions@fixedcode: Added \do@actions@fixedcode	129

\do@actions@next: Added number skipping to \do@actions	128
\do@insidelinehook: Added \do@linehook for use in \do@line	126
\endnumbering: Changed \endnumbering for eledpar	71
\fx@l@cks: Added \ch@cksub@l@ck, \ch@ck@l@ck and \fx@l@cks	132
\footsplitskips: Added \footsplitskips for use in many footnote styles	141
\get@linelistfile: Added \get@linelistfile	86
\ifledRcol@: Added \l@dunpstartsL, \ifl@dpairing and \ifpst@rted for/from eledpar	69
\initnumbering@reg: Added \initnumbering@reg	70
\l@advance@parledgroup@beforenormalnotes: Added \l@dunboxmpfoot containing some common code	234
\l@dcsnotetext@r: Added \l@demptyd@ta	126
\l@dgetline@margin: Added \l@dgetline@margin	76
\l@dgetlock@disp: Added \l@dgetlock@disp	78
\l@dgetsidenote@margin: Added \l@dgetsidenote@margin	226
\l@drsn@te: Added \l@dlsn@te and \l@drsn@te for use in \do@line	126
\l@dzeropenalties: Added \l@dzeropenalties	121
\ledlinenum: Added \ledlinenum for use by \leftlinenum and \rightlinenum	80
\line@list@stuff: Deleted \page@start from \line@list@stuff	99
\list@clearing@reg: Added \list@clearing@reg	86
\n@num: Added \n@num	95
\normalbfnoteX: Removed extraneous space from \normalbfnoteX	173
\resumenumbering: Changed \resumenumbering for eledpar	72
\setprintendlines: Added \setprintendlines for use by \printendlines	192
\setprintlines: Added \setprintlines for use by \printlines	164
\skipnumbering: Added \skipnumbering and supports	102
\sublinenumincrement: Added \firstlinenum, \linenumincrement, \firstsublinenum and \linenumincrement	77
\sublinenumr@p: Using \linenumrep instead of \linenumr@p	79
Using \sublinenumrep instead of \sublinenumr@p	79
\vnumfootnoteX: Removed extraneous space from \vnumfootnoteX	174
v0.8.0.	
General: Bug on endnotes fixed: in a // text, all endnotes will print and be placed at the ends of columns ()	1
v0.8.1.	
General: Bug on \edtext ; \critex ; \lemma fixed: we can now us non-switching commands	1
v0.9.0.	
General: No more ledpatch. All patches are now in the main file.	1
v0.9.1.	
General: Fix some bugs linked to integrating ledpatch on the main file.	1
v0.10.0.	
General: Corrections to \section and other titles in numbered sections	1
v0.11.0.	
General: Makes it possible to add a symbol on each verse's hanging, as in French typography. Redefines the command \hangingsymbol to define the character.	1
v0.12.0.	
General: For compatibility with eledpar, possibility to use \autopar on the right side.	1
Possibility to number \pstart.	16
Possibility to number the pstart with the commands \numberpstarttrue.	1

\ifledRcol@: Added \ifledRcol and \ifnumberingR for/from eledpar	69
v0.12.1.	
General: Don't number \pstarts of stanza.	1
The numbering of \pstarts restarts on each \beginnumbering.	1
v0.13.0.	
General: New stanzaindentsrepetition counter to repeat stanza indents every n verses.	40
New stanzaindentsrepetition counter: to repeat stanza indents every n verses.	1
\managestanza@modulo: New stanzaindentsrepetition counter to repeat stanza indents every n verses.	245
v0.13.1.	
General: \thepstartL and \thepstartR use now \bfseries and not \bf, which is deprecated and makes conflicts with memoir class.	1
v0.14.0.	
General: Tweaked \edlabel to get correct line number if the command is first element of a paragraph.	1
\edlabel: Tweaked \edlabel to get correct line number if the command is first element of a paragraph.	217
v0.15.0.	
General: Line numbering can be reset at each pstart.	74
Possibility to print \pstart number inside.	16
\affixline@num: Line numbering can be disabled.	130
\ifinserthangingsymbol: New management of hangingsymbol insertion, preventing undesirable insertions.	243
\printlines: Line numbering can be reset at each pstart.	166
v0.17.0.	
\ifinserthangingsymbol: New new management of hangingsymbol insertion, preventing undesirable insertions.	243
v1.0.0.	
General: \lemma can contain commands.	23
Debug in lineation command	18
New generic commands to customize footnote display.	29, 204
Options nonum and noseup in \Xfootnote.	22
Options of \Xfootnotes.	138
Possibility to have commands in sidenotes.	46
Some compatibility break with eledmac. Change of name: eledmac.	1
\morenoexpands: Change to be compatible with new features	106
v1.0.1.	
General: Correction on \Xnumberonlyfirstinline with lineation by pstart or by page. 30	
v1.1.0.	
General: Add \labelpstarttrue.	16
Add \Xnumberonlyfirstintwolines	30
Add \Xpstart and \Xonlypstart	31
New hook to add arbitrary code at the beginning of the notes	35
New options for block of notes.	37
New package option: parapparatus.	1
New tools to change order of series	203
Sectioning commands.	52
\preXnotes: New skip \preXnotes@	186
\settoggle@series: \settoggle@series switch the global value of the toggle, not only the local value.	205

v1.2.0.	
\endquote:	Compatibility of \ledchapter with the <i>memoir</i> class. 272
\preXnotes:	Debug in familiar footnotes (bug introduced by v1.1). 186
v1.3.0.	
\endquote:	<i>Quotation</i> and quote environment inside numbered sections. 272
v1.4.0.	
General:	Compatibility with LuaTeX of RTL notes. 1
\edtext:	Compatibility of \edtext with the right-to-left direction (with Polyglossia). 106
\ledsetnormalparstuffX:	Direction of footnotes with polyglossia. 183
\newseries@:	Remembers the language of the lemma, in order to create a correct direction for the footnote separator. 197
\rbracket:	Switch the right bracket to a left bracket when the lemma is RTL (needs polyglossia or LuaTeX). 159
v1.4.1.	
\affixside@note:	Remove spurious spaces. 230
\endquote:	New option <i>noquotation</i> 272
\labelrefsparsesubline:	Fix bug with \edlabel. 218
\vl@dbfnote:	Compatibility of standard footnotes with eledmac when these footnotes contain any commands. 169
v1.4.2.	
General:	Debug with some special classes. 1
v1.4.3.	
General:	Add \Xnonbreakableafternumber. 32
Spurious space after familiar footnotes. 1
v1.4.4.	
General:	Label inside familiar footnotes. 1
v1.4.5.	
General:	Bug with komasscript + eledpar + chapter. 1
v1.4.6.	
General:	Bug with memoir class introduced by 1.4.5. 1
v1.4.7.	
\endquote:	Compatibility of sectioning commands with \autopar. 272
v1.4.8.	
General:	Corrects a bug with parallel texts introduced by 1.1. 1
v1.4.9.	
\normalbfnoteX:	Allow to redefine \thefootnoteX with alph when some packages are loaded. 173
v1.5.0.	
General:	Correct indexing when the call is made in critical notes. 236
\do@insidelinehook:	Added \do@insidelinehook for use in \do@line 126
\edindex:	Compatibility with imakeidx package, and possibility to use multiple index with \edindex. 239
v1.5.1.	
\managestanza@modulo:	Correct stanzaindentsrepetition counter 245
\normalvfootnoteX:	Fix bug with normal familiar footnotes when mixing RTL and LTR text. 171
v1.6.0.	
\newverse:	Add \falseverse macro. 248
v1.6.1.	
General:	Corrects a false hanging verse when a verse is exactly the length of a line. 1

\AtEveryPstart: Spurious space in \pstart.	118
\ifinserthangingsymbol: Hang verse is now not automatically flush right.	243
\l@dunhbox@line: Move the call to \inserthangingsymbol to allow use \hfill in-side.	123
\pend: Spurious space in \pend.	120
v1.7.0.	
General: New features for managing page breaks.	54
v1.8.0.	
General: Compatibility with parledgroup option of eledpar package.	1
If imakeidx and hyperref are loaded, adds hyperref in the index.	236
\endquote: Correction of sectioning commands in parallel texts.	272
\get@index@command: Debug \get@index@command and compatibility with hyperref package.	238
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work.	207
\prevpage@num: Correct \parafootsep when using with ledgroup.	152
v1.8.1.	
General: Debug endnotes when more than one series is used (change the position where tools for endnotes are defined).	188
v1.8.2.	
General: Debug compatibility problem with hebrew option of babel package.	1
v1.8.3.	
General: Fixes spurious spaces added by v1.7.0.	1
v1.8.5.	
General: Debug indexing in right column, with eledpar.	236
v1.9.0.	
\doxtrafeet: Add \fnpos to choice the order of footnotes.	211
\l@dfeetendmini: Add \mpfnpos to choice the order of footnotes in minipage / led-group.	232
v1.10.0.	
General: Add \pstartref and \xpstartref to refer to a pstart number (extension of \edlabel).	1
\endquote: Correction of sectioning commands in parallel texts.	272
v1.10.1.	
General: Compatibility with cleveref.	1
v1.10.2.	
General: Compatibility of stanza with v1.8a of babel-greek.	1
v1.10.3.	
General: Debug of cross-referencing.	1
v1.10.4.	
General: Debug of critical notes in edtabular environment.	1
v1.10.5.	
General: Debug of \pausenumbering.	1
Debug of \xxref.	1
v1.10.6.	
General: Debug of interaction between \autopar and \pausenumbering.	1
v1.11.0.	
General: Add hooks to disable the font selection for lemma in footnote.	35
v1.11.1.	
General: Correct a bug when a critical note starts with plus or minus.	1

v1.12.0.

\@nl@reg: To ensure compatibility with \musixtex, \@l becomes \@l. Consequently, \@l@reg becomes \@nl@reg.	88
General: Add \ledinnernote and \ledouternote commands.	46
Add \Xendparagraph and related settings.	38
Add hyperlink to crossref (needs hyperref package).	43
Compatibility with musixtex.	1
Debug eledmac sectioning command after using \resumenumbering.	1
Ensure that imakeidx is loaded <i>before</i> eledmac	236
New hooks: \Xafterrule and \afterruleX	38
New options for ragged-paragraph notes	37
New sectioning commands.	52
Optional arguments for \pstart and \pend.	16
\AtEveryPstart: New optional argument for \pstart, to execute code before it. ...	118
\edindex: Use correctly default index when imakeidx is loaded.	239
\endquote: \ledxxx sectioning commands are deprecated and replaced by \eledxxx commands.	272
\ifledRcol@: Add \ifledRcol@ for eledpar	69
\initnumbering@reg: \beginnumbering is defined only on eledmac, not on eledpar.	70
\l@dcnote: \l@dlsnote, \l@drsnote and \l@dcnote defined only one time, in eledmac, including needs for eledpar case.	228
\l@dgetsidenote@margin: \sidenotemargin is now directly defined in eledmac to be able to manage eledpar.	226
\l@dunhbox@line: \do@line is split in more little commands.	124
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work when called after \footparagraphX.	207
Debug \Xbeforenotes and \Xmaxhnotes which did not work when called after \footparagraph.	207
\pend: New optional argument for \pend, to execute code after it.	120
\stanza: &can have an optional argument: content to be printed after.	248
\Stanza can have an optional argument: content to be printed before.	248
Add \newverse macro, \falseverse deprecated.	248
v1.12.1.	
\wrap@edcrossref: Fix spurious spaces.	220
v1.12.2.	
\l@dunhbox@line: Fix a bug with critical notes at the tops of pages (added by v12.0.0) ...	123
v1.12.3.	
General: Add macros for new messages since v0.7	63
Correct bug with side and familiar notes in tabular environments.	1
Debug \eledxxx with some paper size	1
Debug \ledinnernote and \ledouternote commands in the top of pages.	46
Debug left and right notes (bugs added by 1.12.0)	1
Underline lemma in \eledxxx when using draft mode.	1
\flag@end: \flag@start and \flag@end are now defined only one time for eledmac and eledpar	100
\flag@start send a error message when a \edtext is done without insert (note) ...	100
\reledmac@error: Replaced error messages	63
v1.12.4.	
General: Debug spurious page breaks before \chapter (bug added in 1.12.0)	1

v1.12.5.	
\@edindex@hyperref: Debug \edindex when hyperref is not loaded	241
\@ssect: Debug \eledchapter in parallel with memoir	275
\doinsidelinehook: Added \dolinehook and \doinsidelinehook	126
\endnumbering: Allow to mix parallel columns and normal text when using \pausenumbering	71
\l@dgobblearg: \l@dgobblearg becomes \l@dgobbeloptarg	255
\l@drestoreforedtext: Debug optional arguments of \Xfootnote in tabular context	255
\resumenumbering: Debug \resumenumbering	72
v1.12.7.	
\wrap@edcrossref: \wrap@edcrossref is now robust	220
v1.12.8.	
\flag@end: \flag@start do not send a error message when a \edtext is done without insert (note) but have a endnote	100
v1.13.0.	
General: Add \Xnoteswidthliketwocolumns and \noteswidthliketwocolumnsX	38
Added widthliketwocolumns option	60
\newhooktoggle@series@reload: Add \newhookcommand@toggle@reload	207
\para@footsetupX: In \para@footsetupX, use \columnwidth instead of \hsize	180
\settoggle@series: \settoggle@series can take an optional arguments to reload series setup.	205
v1.13.1.	
General: Coming back of page and line breaking penalties's management, deleted by error in v0.17.	1
Debug quotation environment inside of a \pstart preceded by a sectioning command.	1
\thepstart: Add \l@dzeropenalties in \pstart	119
v1.13.2.	
General: Fix bug with normal footnotes, added by v1.13.0.	1
\ifledRcol@: Add \ifl@dpadding for eledpar	69
v1.13.3.	
General: Fix extra spaces with paragraphed footnotes, added by v1.13.0.	1
v1.13.4.	
General: Fix bug with index when memoir class is used without hyperref	1
v1.14.0.	
General: Debug spurious characters before endnotes.	188
Delete previous override of \l@d@@wrindexhyp at the beginning of a document when hyperref is not loaded.	243
Move gobbling command	68
Provide \@gobblefour	68
\edindex: Let eledmac take advantage of imakeidx even when memoir class is used	239
v1.14.1.	
\@ssect: Debug sectioning commands when using both handout and hyperref package.	278
v1.14.2.	
\@ssect: Debug \edtext after starred sectioning commands when using memoir class.	275
v1.15.0.	
\@edtext@level: New boolean \if@edtext@.	106
General: Fix bug with footnotes layout when using some options of the geometry package (bug add by v1.13.0).	1
New commands \AtEveryPstart and \AtEveryPend.	16

New tools to prevent ambiguous references in lemma	25
\arrangementX@threecol: Correct bug with paragraphed familiar footnotes setting.	179
\endsub: Restore subline feature (disabled by mistake in v1.8.0).	101
\if@lemmacommand@: New boolean \iflemmacommand@.	111
v1.15.1.	
\line@list@stuff: Revert modification of 1.5.2 which makes bug with numbering. Leave vertical mode to solve spurious space before minipage.	99
v1.16.0.	
General: \edtext is now defined only in eledmac, not in eledpar. Debug wrong num- bering when using \sameword + eledpar + \tag command.	106
Compatibility of standard footnotes with some biblatex styles.	1
New \stanzaindent command.	1
v1.16.1.	
\edlineref: \lineref is not defined if defined by some other package, like lineno. Eledmac provides \edlineref instead.	221
v1.17.0.	
\edtext: Error message when calling \edtext outside of a numbered paragraph.	106
v1.18.0.	
\@edindex@hyperref: Fix spurious space with \edindex when using imakeidx/indextools + hyperref.	241
General: Add \Xpstarteverytime	31
Compatibility with Lua \TeX RTL languages.	1
Debug \Xonlypstart when using \Xnumberonlyfirstinline and the current line number differs from the previous.	31
\edlabel: \edlabel is now defined only one time for both eledmac and eledpar	217
\ifledRcol@: Add \ifl@dprintingpages and \@dprintingcolumns for eledpar	69
\l@d@section: Option parapparatus works for endnotes.	189
\print@line: Compatibility with Lua \TeX RTL languages.	124
\printlinefootnote: Code refactoring in \printlinefootnote: the printing of the numbers are factorized in \printlinefootnotearea	159
\printpstart: Debug \Xpstart with parallel pages and columns (eledpar)	159
v1.19.0.	
General: \Xmaxhnotes and \maxhnotesX work now for both two-columns and three- columns setting.	1
Compatibility with eledpar v1.13.0.	1
\footplitskips: \footplitskips doesn't set \floatingpenalty to \@MM when processing parallel pages.	141
\xxref: \xxref works also with right side numbers, when \@Rlineflag is not empty.	222
v1.19.1.	
General: Call \correct@footinsX@box and \correct@Xfootins@box directly in \print@notesX@forpages and \print@Xnotes@forpages, that is in eledpar.	1
v1.20.0.	
General: Add \Xendboxlinenum	33
Add \Xtwolines and \Xmorethantwolines hooks	30
Add series option.	1
Correct \Xinplaceofnumber hook.	1
Explicit error message when calling \Xfootnote outside of \edtext.	1
Fix bug with line number typesetting direction when using \eledsection and similar commands for RTL texts with Lua \TeX	1
Fix issues with RTL text in notes when using Lua \TeX	1

Options fulllines in <code>\Xfootnote</code> .	22
The <code>\newifs</code> are not followed by boolean values set to false, because it is the \TeX default setting.	1
<code>\printlines</code> : Added <code>\ifl@d@Xmorethantwolines</code> and <code>\ifl@d@Xmorethantwolines</code> to <code>\printlines</code>	167
<code>\stanza</code> : <code>&</code> and <code>\&</code> can be preceded by spaces.	248
<code>\xxref</code> : Debug <code>\xxref</code> when not loading <code>eledpar</code> (fix bug added in 1.19.0).	222
v1.21.0.	
<code>\@edindex@hyperref</code> : Look at the hyperindex option of hyperref before inserting hyperref	241
General: <code>\AtEveryPstart</code> and <code>\AtEveryPend</code> are now compatible with <code>\autopar</code>	1
<code>\Xafterrule</code> and <code>\afterruleX</code> features no longer create problems of overflowing at the bottom of the page.	1
<code>\chapter</code> inside optional argument of <code>\pstart</code> works when typesetting parallel pages	1
<code>\preXnotes</code> and <code>\prenotesX</code> features no longer create problems of overflowing at the bottom of the page.	1
<code>\seriesatbegin</code> and <code>\seriesatbegin</code> more efficient	203
Add <code>\applabel</code> and related	45
Add <code>\beforenotesX</code> and <code>\Xbeforenotes</code> features for notes set in two and three column.	1
Add <code>\hidenumbering</code>	20
Add <code>\Xcolalign</code> and <code>\colalignX</code> hooks	36
Add <code>\Xendtwolines</code> , <code>\Xendmorethantwolines</code> , <code>\Xendtwolinesbutnotmore</code> and <code>\Xendtwolinesonlyinsamepage</code> .	31
Add <code>\Xparindent</code> and <code>\hangindentX</code>	35
Add <code>\Xtwolinesbutnotmore</code> and <code>\Xtwolinesonlyinsamepage</code> .	1
Add <code>nocritical</code> , <code>noend</code> , <code>nofamiliar</code> and <code>noledgroup</code> options.	1
Add <code>noeledsec</code> package option	1
Debug <code>\beforenotesX</code> <code>\maxhnotesX</code> <code>\noteswidthliketwocolumnsX</code> and <code>\afterruleX</code> with footnotes set in two and three columns.	1
Fix bug when a <code>\Xfootnote</code> follows a <code>\Xendnote</code> in the second argument of <code>\edtext</code> (bug added in <code>eledmac</code> 1.0.0).	1
Fix bug with <code>\maxhnotesX</code> when using <code>\foottwocolX</code> or <code>\footthreecolX</code> .	1
Fix bug with space between columns with notes in two columns (bug added in v1.13.0).	1
Fix spurious space after first page number in <code>\doendnotes</code> . <code>oldprintnnumspace</code> option allows to come back to previous setting	1
<code>parapparatus</code> option works now with familiar footnotes.	1
Provide <code>\@gobblefive</code>	68
<code>\l@d@section</code> : <code>\endnotes</code> take five arguments.	189
<code>\ledinnotemark</code> : Add <code>\ledinnotemark</code> .	239
<code>\ledsetnormalparstuffX</code> : <code>\ledsetnormalparstuff</code> is deprecated and becomes <code>\ledsetnormalparstuffX</code> and <code>\Xledsetnormalparstuff</code> .	183
<code>\n@num</code> : <code>\n@num@ref</code> deleted	95
<code>\n@num</code> defined only one time for both <code>Eledmac</code> and <code>Eledpar</code>	95
<code>\newhookcommand@series</code> : <code>\newhookcommand@series</code> can take an optional argument.	206
<code>\newhooktoggle@series</code> : <code>\newhooktoggle@series</code> can take an optional argument.	207
<code>\print@footnoteXrule</code> : Code refactoring: the spaces after the footnote rules are directly managed in <code>\print@Xfootnoterule</code> and <code>\print@footnoteXrule</code>	185

<code>\seriesatend</code> : Fix spurious space in <code>\seriesatend</code>	203
<code>\skipnumbering</code> : <code>\skipnumbering</code> defined only one time for both <code>Eledmac</code> and <code>Eledpar</code>	102
Correct <code>\skipnumbering</code> for stanza.	102
Delete <code>\skipnumbering@reg</code>	102
v1.22.0.	
General: Add <code>\doendnotesbysection</code> command.	22
Add option for lemma separator inside endnotes	34
Adds hyperlink for references to notes in indices.	1
Fix conflict between <code>noend</code> package option and <code>edtabularx</code> environments	1
Provides support for <code>xindy</code>	1
Standardize endnotes handbook.	22
When using <code>hyperref</code> package, internal links in index or with <code>\edlineref</code> are now targeted to the top and not longer to the bottom of the lines they refer to.	1
<code>\ledinnote</code> : <code>\ledinnote</code> takes a first optional argument, which is the label for hyper- links.	239
v1.22.1.	
General: Fix bug (added on v1.22.0) with <code>\Xinplaceofnumber</code> hook.	1
<code>\prevpage@num</code> : Correct double symbol when using both <code>\parafootsep</code> and <code>\Xsymllinenum</code>	152
v1.23.0.	
<code>\@edtext@level</code> : The boolean <code>\if@edtext@</code> becomes the counter <code>\edtext@level</code>	106
General: Add <code>\Xboxlinenumalign</code> and <code>\Xendboxlinenumalign</code>	33
Add <code>\Xboxstartlinenum</code> , <code>\Xendboxstartlinenum</code> , <code>\Xboxendlinenum</code> , <code>\Xendboxendlinenum</code>	33
Allow use of <code>\sameword</code> with <code>inputenc</code> managing of UTF-8.	1
Compatibility between <code>nofamiliar/nocriticals</code> option and <code>minipage/ledgroup</code>	1
Error message when using <code>\beginnumbering... \endnumbering</code> without <code>\pstart</code>	1
Fix bug with <code>\sameword</code> when the lemma overlaps multiple line.	25
Fix bug with <code>\sameword</code> when the same lemma is used for multiple notes or for nested <code>\edtexts</code>	25
Fix bug with <code>\skipnumbering</code> called immediately after a <code>\pstart</code>	1
Fix error of <code>\iftrue</code> not closed.	1
Fix spurious space with <code>\skipnumbering</code> (bug added on v1.21.0).	1
New tools to ensure the line-list file uses the right version of commands when upgrad- ing the <code>eledmac</code> version.	1
Optional argument of <code>\sameword</code> can be a comma-separated list of <code>\edtext</code> depth.	25
<code>\lemma</code> : Fix spurious space after <code>\lemma</code> command	110
<code>\newseries@</code> : Prevent spurious spaces when <code>\Afootnote</code> and similar commands are followed by spaces (bug added on 1.0.0).	197
<code>\sameword</code> : In order to allow use of <code>\sameword</code> with <code>inputenc</code> , we detokenize its manda- tory argument before using it in control sequence names.	114
v1.23.1.	
General: Fix bug with <code>\lemma</code> command in the right side.	1
v1.23.2.	
General: Compatibility with \LaTeX 's release 2015.	1
v1.24.0.	
General: We can reinitialize <code>\AtEveryPstart</code> and <code>\AtEveryPend</code> providing to it an empty argument.	1

v1.24.1.	
General: \lemma is disabled when using ‘nocritical’ option.	1
v1.24.2.	
General: Fix incompatibility between ‘nofamiliar’ option and ‘memoir’ package.	1
v1.24.3.	
General: Restore marginal numbers and notes with sectioning command (bug introduced in v1.21.0)	1
v1.24.4.	
General: Fix spurious space with \edindex when using xindy+hyperref option.	1
v1.24.5.	
General: Fix bug of indent, when a added in 1.1.0, when a \beginnumbering immediately follow a sectioning command.	1
v2.0.0.	
\@iiiminipage: Patch \@iiiminipage instead of redefining it.	233
\@xympar: Patching \@xympar instead of redefining it	225
General: \@makecol, \@reinserts and \@docclearpage are patched instead of begin redefined	214
\doxtrafeeti becomes \do@feetX; \doxtrafeetii becomes \Xdo@feet; \@opxtrafeeti becomes \@opfeetX; \doreinxtrafeetii becomes \X@doreinfeet; \doreinxtrafeeti becomes \@doreinfeetX.	214
Add \Xendinplaceofnumber hook.	1
Add \Xendnonumber hook.	1
Add nonum option for endnotes.	1
Fix bug when printing only one series of endnotes, but wanted to keep endnotes for other series.	1
In order to have a more consistent name’s convention, many names has been changed.	1
Many L ^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	168
Suppress indexing command specific to memoir.	239
\endminipage: Patch \endminipage instead of redefining it.	233
\initnumbering@quote: \initnumbering@sectcmd becomes \initnumbering@quote	271
\l@advance@parledgroup@beforenormalnotes: Some conde of \l@dumboxmpfoot moved to \l@advance@parledegroupp@beforenormalnotes	234
\newseries@: One endnotes file by series.	201
v2.0.1.	
General: Fix bug in eledmac-compat option	1
Fix incompatibility between optional argument of \pstart and \numberpstarttrue	1
v2.1.0.	
General: Fix bug with \advanceline at the beginning of a \pstart.	1
Fix bug with \chapter in optional argument of \pstart in parallel typesetting with scrbook.	1
Fix bug with \eledchapter in parallel typesetting with scrbook.	1
Fix bug with \setline at the beginning of a \pstart.	1
Fix spacing bug with \Xbhooknote and \bhooknoteX when using them to insert text and not to execute code.	1
New tools to number stanzas	1
v2.1.1.	
General: Fix bug with \ledpbsetting{before}.	1

v2.1.2.	
General: Fix bug with lineation by <code>pstart</code> and <code>tabular</code> environments (added in 2.1.0).	1
v2.1.3.	
General: <code>\Xhangindent</code> and <code>\hangindentX</code> work now with all the paragraphs in the note.	1
<code>\Xnoindent</code> and <code>\noindentX</code> work now again (broken in 2.0.0).	1
Change some internal code in order to provide compatibility with \TeX release of october 2015	1
Fix bug which inserted double space before paragraphed familiar notes.	1
Fix bug with <code>\edindex</code> when using not-Latin characters without UTF-8 engines	1
<code>\ledsetnormalparstuffX</code> : Replaced <code>\noindent</code> with <code>\parindent</code> set to 0pt.	183
v2.2.0.	
General: Fix bug with combination of <code>\onehalfspacing</code> and two columns and three columns notes typeset.	1
Fix bug with some setting command and optimization option.	1
Fix spurious space with paragraphed critical notes when using <code>Lua\TeX</code> .	1
Increase line list version number to ensure compatibility with new options of <code>reledpar</code> package.	1
New setting tools for endnotes: <code>\Xendnumberonlyfirstinline</code> , <code>\Xendnumberonlyfirstintwolines</code> , <code>\Xendsymmlinenumber</code> , <code>\Xendbeforenumber</code> , <code>\Xendafternumber</code> , <code>\Xendbeforemsymmlinenumber</code> , <code>\Xendaftersymmlinenumber</code> , <code>\Xendboxsymmlinenumber</code> , <code>\Xendhangindent</code> , <code>\Xendbhooklinenumber</code> , <code>\Xendahooklinenumber</code> , <code>\Xendbhookinplaceofnumber</code> , <code>\Xendahookinplaceofnumber</code> .	1
v2.2.1.	
General: Compatibility with \TeX format 2015/10/01.	1