

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 tool in order to typeset scholarly edition. It is base on the **eledmac** package, which was based on **ledmac** package, which was based on **edmac** T_EX package.

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

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

To report bugs or request a new feature, please go to ledmac GitHub page and click on “New Issue”: <https://github.com/maieul/ledmac/issues/>. You must create an account on github.com 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 in:
<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 v2.1.1, last revised 2015/08/13.

[†]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 Optimize package's performance	13
4 Text lines and paragraphs numbering	14
4.1 Text lines numbering	14
4.2 Paragraphs	15
4.2.1 Basis	15
4.2.2 Producing automatically <code>\pstart... \pend</code>	15
4.2.3 Content before specific <code>\pstart</code> and after specific <code>\pend</code>	16
4.2.4 Content before every <code>\pstart</code> and after every <code>\pend</code>	16
4.2.5 Numbering paragraphs (<code>\pstart</code>)	16
4.2.6 Languages written in Right to Left	17
4.2.7 Memory limits	17
4.3 Lineation commands	17
4.3.1 Disabling lineation	17
4.3.2 Setting lineation start and step	18
4.3.3 Setting lineation reset	18
4.3.4 Setting line number margin	18
4.3.5 Other settings	19
4.4 Changing the line numbers	19
4.4.1 Sublineation	19
4.4.2 Locking lineation	19
4.4.3 Setting and changing line number	19
4.4.4 Line number style	20
4.4.5 Skipping and hiding number	20
4.4.6 Execute code at each line	20
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	24
5.3.1 Basic use	25
5.3.2 Notes about input encoding with UTF-8 processor	25
5.3.3 Use with <code>\lemma</code> command	25

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

8 Verse	39
8.1 Basic	39
8.2 Define stanza indents	39
8.3 Repeating stanza indents	39
8.4 Manual stanza indent	40
8.5 Stanza breaking	40
8.6 Hanging symbol	41
8.7 Long verse and page break	41
8.8 Hanging symbol	41
8.9 Content before/after verses	41
8.10 Numbering stanza	41
8.11 Various tools	42
9 Grouping	42
10 Cross referencing	43
10.1 Basic use	43
10.2 Refer to a critical notes	43
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	44
10.4 Not automatic cross-referencing	44
10.5 Normal \LaTeX cross-referencing	44
10.6 References to lines commented in the apparatus	45
11 Side notes	46
11.1 Basis	46
11.2 Setting	46
11.2.1 Width	46
11.2.2 Vertical position	46
11.2.3 Distance to the main text	46
11.2.4 Separator between notes	47
12 Indexing	47
12.1 Basis	47
12.2 Separator between page and line numbers	47
12.3 Using xindy	48
12.4 Advanced setting	48
13 Tabular material	48
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	52
14.3 Optimization	53
15 Quotation environments	53

16 Page breaks	53
16.1 Control page breaking	53
16.2 Prevent page break in a long verses	54
17 Miscellaneous	54
17.1 Known and suspected limitations	55
17.2 ‘No room for a new’	55
17.2.1 Marginal notes	55
17.2.2 Paragraph shape	55
17.2.3 Paragraphed footnotes	55
17.2.4 Use with other packages	56
17.3 Parallel typesetting	57
I Implementation overview	58
II Preliminaries	58
II.1 Links with original edmac	58
II.2 Package declaration	58
II.3 Package options	59
II.4 Loading packages	60
II.5 Boolean flags	61
II.6 Messages	61
II.7 Gobbling	67
II.8 Miscellaneous commands	67
II.9 Prepare reledpar	67
III Sectioning commands	68
IV List macros	72
V Line counting	73
V.1 Choosing the system of lineation	73
V.2 Line number margin	75
V.3 Line number initialization and increment	76
V.4 Line number locking	77
V.5 Line number style	78
V.6 Line number printing	78
V.7 Line number counters and lists	79
V.8 Line number locking counter	80
V.9 Line number associated to lemma	81
V.10 Reading the line-list file	84
V.11 Commands within the line-list file	86
V.12 Writing to the line-list file	97

VI Marking text for notes	102
VI.1 <code>\edtext</code> itself	103
VI.2 Substitute lemma	109
VI.3 Substitute line numbers	110
VI.4 Lemma disambiguation	111
VII Paragraph decomposition and reassembly	117
VII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	117
VII.2 Processing one line	122
VII.2.1 General process	122
VII.2.2 Process for “normal” line	122
VII.2.3 Process for line containing <code>\eledsection</code> command	124
VII.2.4 Hooks	124
VII.2.5 Sidenotes and marginal line number initialization	124
VIII Line and page number computation	125
IX Line number printing	128
X Pstart number printing in side	132
XI Restoring footnotes and penalties	133
XI.1 Add insertions to the vertical list	134
XI.2 Penalties	135
XI.3 Printing leftover notes	135
XII Critical footnotes	136
XII.1 Fonts	136
XII.2 Individual note options	137
XII.3 Notes language	137
XII.4 General survey of the way we manage notes	138
XII.5 General setup	139
XII.6 Footnotes arrangement	140
XII.6.1 User level macro	140
XII.6.2 Normal footnote	140
XII.6.3 Paragraphed footnotes	144
XII.6.4 Columnar footnotes	151
XII.7 Critical notes presentation	156
XII.7.1 Font tools	157
XII.7.2 Pstart number in footnote	157
XII.7.3 Line number printing	158
XIII Familiar footnotes	165
XIII.1 Adjacent footnotes	165
XIII.2 Regular footnotes for numbered texts	167
XIII.3 Footnote formats	168
XIII.4 Footnote arrangement	168

XIII.4.1 User level macro	168
XIII.4.2 Normal footnotes	169
XIII.4.3 Two columns footnotes	173
XIII.4.4 Three columns footnotes	175
XIII.4.5 Paragraphed footnotes	177
XIV Code common to both critical and familiar footnote in normal arrangement	181
XV Footnotes' width for two columns	182
XVI Footnotes' order	183
XVII Footnotes' rule	183
XVIII Specific skip for first series of footnotes	184
XVIII.0.1 Overview	184
XVIII.0.2 User level command	184
XVIII.0.3 Internal commands	185
XIX Endnotes	186
XX Generate series of notes	192
XX.1 Test if series is still existing	193
XX.2 Init specific to <code>reledpar</code>	193
XX.3 For critical footnotes	193
XX.3.1 Options	193
XX.3.2 Create inserts, needed to add notes in foot	194
XX.3.3 Create commands for critical apparatus, <code>\Afootnote</code> , <code>\Bfootnote</code> etc.	194
XX.3.4 Set standard display	196
XX.4 For familiar footnotes	196
XX.4.1 Options	196
XX.4.2 Create tools for familiar footnotes (<code>\footnotex</code>)	197
XX.5 The endnotes	197
XX.5.1 The auxiliary file	198
XX.5.2 The main macro	198
XX.5.3 The options	199
XX.6 Init standards series (A,B,C,D,E)	199
XXI Setting series display	200
XXI.1 Change series order	200
XXI.2 Test series order	200
XXI.2.1 Get the first series	200
XXI.3 Series setting	201
XXI.3.1 General way of working	201
XXI.3.2 Tools to set options	201

XXI.3.3 Tools to generate options commands	203
XXI.3.4 Options for critical notes	204
XXI.3.5 Options for familiar notes	205
XXI.3.6 Options for endnotes	206
XXI.4 Hooks for a particular footnote	206
XXI.5 Alias	207
XXII Output routine	207
XXII.0.1 Page number management	207
XXII.0.2 Extra footnotes output	207
XXII.0.3 Standard output's commands patching	210
XXIII Cross referencing	212
XXIV Side notes	221
XXV Minipages and such	228
XXVI Indexing	232
XXVII Verse	239
XXVII.1 Hanging symbol management	239
XXVII.2 Using & character	240
XXVII.3 Code category setting	240
XXVII.4 Stanza count and indent	240
XXVII.5 Numbering stanza	242
XXVII.6 Stanza number in note	243
XXVII.7 Main work	243
XXVII.8 Restore catcode and penalties	245
XXVIII Arrays and tables	245
XXVIII.1 Preamble: macro as environment	245
XXVIII.2 Tabular environments	249
XXVIII.2.1 Disabling and restoring commands	249
XXVIII.2.2 Counters, boxes and lengths	252
XXVIII.2.3 Tabular typesetting	256
XXVIII.2.4 Environments	267
XXIX Quotation's commands	267
XXX Section's title commands	269
XXX.1 Commands to disable some feature	269
XXX.2 General overview	269
XXX.3 \beforeeledchapter command	270
XXX.4 Auxiliary commands	270
XXX.5 Patching standard commands	271
XXX.6 Main code of \eledxxx commands	276

XXX.7 Macros written in the auxiliary file	278
XXXI Page breaking or no page breaking depending of specific lines	280
XXXII Long verse: prevents being separated by a page break	282
XXXIII Compatibility with eledmac	283
Appendix A Some things to do when changing version	285
Appendix A.1 Migrating from edmac to ledmac	285
Appendix A.2 Migration from ledmac to eledmac	286
Appendix A.3 Migration to eledmac 1.5.1	287
Appendix A.4 Migration to eledmac 1.12.0	287
Appendix A.5 Migration to eledmac 17.1	288
Appendix A.6 Migration to eledmac 1.21.0	288
Appendix A.6.1 \Xledsetnormalparstuff and \ledsetnormalparstuffX	288
Appendix A.6.2 Endnotes	288
Appendix A.7 Migration to eledmac 1.22.0	288
Appendix A.8 Migration to eledmac 1.23.0	288
Appendix A.9 Migration from eledmac to reledmac	289
Appendix A.9.1 Risk of ‘no room for a new’	289
Appendix A.9.2 Multiple indices with memoir	289
Appendix A.9.3 Deprecated commands and options	289
Appendix A.9.4 \renewcommandreplaced by command	290
Appendix A.9.5 Commands the names of which have been changed . . .	290
Appendix A.9.6 Endnotes	292
Appendix A.9.7 Z Series	292
Appendix A.9.8 Internal commands	292
Appendix A.10 Migration to reledmac 2.1.0	292
References	293
Index	293
Change History	332

1 Introduction

1.1 Aim of the package

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

Eledmac 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 some other \LaTeX packages for critical edition typesetting. However, the aim of reledmac is to provide a “all in one” and flexible tool in the field of critical edition.

Any suggestion for new features are welcome.

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

We do not suggest that you need to read the source code for this package in order to use it; we provide this code primarily for reference, and many of our comments on it repeat material that is also found in the earlier sections.

But no documentation, however thorough, can cover every question that comes up, and many can be answered quickly by consultation of the code. On a first reading, we suggest that you should read only the general documentation in sections 2, unless you are particularly interested in the innards of reledmac.

1.2 History

1.2.1 edmac

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

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

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

documentation, multiple-column footnotes, cross-references, and crop marks.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 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.

At the time of writing (1994), we are pleased to be able to say that edmac is being used for real-life book production of several interesting editions, such as the Latin texts of Euclid’s *Elements*,² an edition of the letters of Nicolaus Copernicus,³ Simon Bredon’s *Arithmetica*,⁴ a Latin translation by Plato of Tivoli of an Arabic astrolabe text,⁵ a Latin translation of part II of the Arabic *Algebra* by Abū Kāmil Shujā’ b. Aslam,⁶ the Latin *Rithmachia* of Werinher von Tegernsee,⁷ a middle-Dutch romance epic on the Crusades,⁸ a seventeenth-century Hungarian politico-philosophical tract,⁹ an anonymous Latin compilation from Hungary entitled *Sermones Compilati in Studio 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 *Rithmachia* 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. Is it more and more powerful and flexible, but also more and more divergent from original TeX macro.

1.2.3 `eledmac`

Important changes were put in version 1.0, to make `ledmac` more easily extensible (see 6 p. 28). These changes can trigger small problems with the old customization.

That is why a new name was selected: `eledmac` (extended `ledmac`).

To migrate from `ledmac` to `eledmac`, please read Appendix Appendix A.2 (p. 286).

1.2.4 `reledmac`

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

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

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

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

To migrate from `eledmac` to `reledmac`, please read Appendix ?? (p. ??).

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

A collaborative list of works edited with (r)(e)ledmac is available on https://www.zotero.org/groups/critical_editions_typeset_with_edmac_ledmac_and_eledmac/items. 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 even on the first run, it takes two more passes through L^AT_EX to be sure that everything gets to its right place. Any changes you make to the input file may similarly require three passes to get everything to the right

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

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

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

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

3 Options

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

3.1 Specific features

`draft` underlines lemmas in the main text.

`eledmac-compat` help to migrate from `eledmac` to `reledmac` (see Appendix A.9.5 p. 290).

`nopbinverse` prevents page break inside verses.

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

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

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

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

3.2 Optimize package's performance

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

`noeledsec` disables tools for `\eledsection` and related commands (14.2 p. 52).

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

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

noledgroup `reledmac` allows to use of series of critical notes and new series of normal notes inside `minipage` and `ledgroup` environments (see 9 p. 42). However, such features use up computer memory, at the expense of other processing needs. So if you do not need this feature, use `noledgroup` option. This should make `reledmac` faster.

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

4 Text lines and paragraphs numbering

4.1 Text lines numbering

`\beginnumbering` `\endnumbering` Each section of numbered text must be preceded by `\beginnumbering` and followed by `\endnumbering`, like this:

```
\beginnumbering
text
\endnumbering
```

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

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

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

4.2 Paragraphs

4.2.1 Basis

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

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

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

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

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

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

The lines of this paragraph are
not numbered.

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

4.2.2 Producing automatically `\pstart...``\pend`

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

```

\begingroup
  \beginnumbering
  \autopar

  A paragraph of numbered text.

  Another paragraph of numbered
  text.

  \endnumbering
\endgroup

```

`\autopar` fails, however, on paragraphs that start with a `{` or with any other command that starts a new group before it generates any text. Such paragraphs need to be started explicitly, before the new group is opened, using `\indent`, `\noindent`, or `\leavevmode`, or using `\pstart` itself.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` by brackets, or to add brackets after a `\pend`, just add a `\relax` between `\pstart`/`\pend` and the brackets.

This feature is also useful when typesetting verses (see 8 p. 39) or `eledpar` (see 17.3 p. 57).

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

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

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

4.2.5 Numbering paragraphs (`\pstart`)

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

On each `\beginnumbering` the numbering restarts.

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

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

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 in right to left, we Lua_{La}TeX or Xe_{La}TeX, so you must switch text direction `\before` the `\pstart` command.

4.2.7 Memory limits

`\pausenumbering`
`\resumenumbering`

This paragraph is kept for history, but problem described below should not appear with recent version of \LaTeX . `reledmac` stores a lot of information about line numbers and footnotes in memory as it goes through a numbered section. But at the end of such a section, it empties its memory out, so to speak. If your text has a very long numbered section it is possible that your \LaTeX may reach its memory limit. There are two solutions to this. The first is to get a larger \LaTeX with increased memory.

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

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

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

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

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

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

4.3 Lineation commands

4.3.1 Disabling lineation

`\numberlinefalse`
`\numberlinetrue`

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

4.3.2 Setting lineation start and step

`\firstlinenum`
`\linenumincrement`

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

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

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

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

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

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

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

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

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

4.3.3 Setting lineation reset

`\lineation`

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

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

4.3.4 Setting line number margin

`\linenummargin`

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

```
\linenummargin{left}
```

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

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

4.3.5 Other settings

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

4.4 Changing the line numbers

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

4.4.1 Sublineation

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

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

4.4.2 Locking lineation

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

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

4.4.3 Setting and changing line number

`\setline` In some cases you may want to modify the line numbers that are automatically calculated:
`\advanceline` 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 to advanced feature for user. The argument passed to `\dolinehook{<arg>}` will be executed before slicing a new line in the paragraph. The argument passed to `\doinsidelinehook{<arg>}` will be executed before printing a new line. In many case, the second of two is more useful than the first. The file `examples/2-line_numbers_in_header.tex` provides an example of use in order to print the first and last line number of a page in the header.

5 Apparatus commands

5.1 Terminology

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

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

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

So :

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

5.2 Critical notes

5.2.1 The lemma

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

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

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

For example:

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

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

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

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

However, `\edtext` cannot handle overlapping but unnested notes—for example, one note covering lines 10–15, and another covering 12–18; a `\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 `eledmac`.

`\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.

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 section.

`\doendnotesbysection`

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

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

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

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

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

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

5.2.4 Paragraph in critical apparatus

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

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

5.2.5 Change lemma and line number

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

<pre>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.} }</pre>	<pre>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.</pre>
---	--

`\linenum` You can use `\linenum{⟨arg⟩}` to change the line numbers passed to the notes. The notes are actually given 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). 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 one number, 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 footnotes. 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 say something you find more meaningful, like `\variant`.

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

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.

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`.

5.3.1 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 \XeTeX or \LuaTeX , there should not be any glitches. However, pay attention to how characters are encoded. Similar-looking characters may be represented differently in unicode numbering.

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

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

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

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

- If you use \XeTeX , add this line in your preamble: `\XeTeXinputnormalization 1`.
- If you use \LuaTeX , use the `uninormalize` package of Michal Hoftich¹⁵ with the `buffer` option set to true.

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

5.3.3 Use with `\lemma` command

If you use the `\lemma` command, `eledmac` 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}}
```

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

```

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

```

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

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

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

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

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

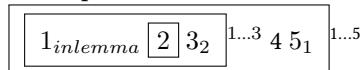
```

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

```

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

In the following schema, 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 ‘regular’
`\footnoteD` 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 Eledmac 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 six series of critical footnotes you can create extra series, using `\newseries` command. For example to create G and H series `\newseriesG,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, B, C, D, 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 concern all series.

When a length, noted `<l>`, is used, it can be stretchable: `a plus b minus c`. The final length `m` is calculated by \TeX to have: $a - c \leq m \leq a + b$. If you use some relative unity¹⁶, it will be relative to fontsize of the footnote, except for commands concerning

¹⁶Like `em` which is the width of a `mg`.

the place kept by the notes — including blank space.

There is also name convention:

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

6.1 Notes arrangement in a series

`\Xarrangement`
`\arrangementX`

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

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

The value of `⟨a⟩` can be one of the following

- `paragraph` formats all the footnotes of a series as a single paragraph;
- `twocol` formats them as separate paragraphs, but in two columns;
- `threecol`, in three columns.
- `normal`, restore normal arrangement.

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

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

`\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 `\footparagraph` 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⟩]`.

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

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

`\Xnumberonlyfirstintwotlines`

`\Xsymlinenum`

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 `\Xnumberonlyfirstintwotlines[⟨s⟩]`, the distinction is made. Use `\Xnumberonlyfirstintwotlines[⟨s⟩]` to disable this (⟨s⟩ can be empty if you want to disable it for every series). For setting a particular symbol in place of the line number, you can use `\Xsymlinenum[⟨s⟩]{⟨symbol⟩}` in combination with `\Xnumberonlyfirstinline[⟨s⟩]`. From the second lemma of the same line, the symbol will be used instead of the line number. Note that any command called in `⟨symbol⟩` must be robust. Use `\robustify` to robustify a not robust command.

6.2.2 Abbreviate line range

`\Xtwotlines`
`\Xmorethantwotlines`

If a lemma is printed on two subsequent lines, Eledmac 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 `\Xtwotlines[⟨s⟩]{⟨text⟩}` and `\Xmorethantwotlines[⟨s⟩]{⟨text⟩}`. The `⟨text⟩` argument of `\Xtwotlines` will be printed if the lemma is on two lines, and the `⟨text⟩` argument of `\Xmorethantwotlines` will be printed if the lemma is on three or more lines. For example:

```
\Xtwotlines{sq.}
\Xmorethantwotlines{sqq.}
```

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

`\Xmorethantwotlines`

If you use `\Xtwotlines` without setting `\Xmorethantwotlines`, the `⟨text⟩` argument of `\Xtwotlines` 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 `\Xtwotlinesbutnotmore[⟨series⟩]`.

It is possible to disable `\Xtwotlinesbutnotmore[⟨series⟩]` with `\Xtwotlinesbutnotmore[⟨series⟩][false]`.

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

`\Xtwotlinesonlyinsamepage`

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

Use `\Xtwotlinesonlyinsamepage[⟨series⟩][false]` to disable this.

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

`\Xendtwotlines`
`\Xendmorethantwotlines`
`\Xendtwotlinesbutnotmore`

For endnotes, use `\Xendtwotlines`; `\Xendmorethantwotlines`; `\Xendtwotlinesbutnotmore`; `\Xendtwotlinesonlyinsamepage` instead of `\Xtwotlines`; `\Xmorethantwotlines`; `\Xtwotlinesbutnotmore`; `\Xtwotlinesonlyinsamepage`.

6.2.3 Disable line number

`\Xnonumber`

You can use `\Xnonumber[⟨s⟩]` if you do not want to have the line number in a foot-

`\Xendnonumber` note. To cancel it, use `\Xnonumber[⟨s⟩][false]`. `\Xendnonumber[⟨s⟩]` is the same for endnote.

6.2.4 Printing pstart number

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

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

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

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

6.2.5 Printing stanza number

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

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

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

6.2.6 Space around number

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

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

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

6.2.7 Space around line symbol

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

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

6.2.8 Space in place of number

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

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

6.2.9 Boxing line number and line symbol

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

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

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

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

L to align left (default value);

R to align right;

C to center.

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

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

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

12-23
24ff.

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

6.3 Separator between the lemma and the note

6.3.1 For footnotes

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

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

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

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

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

6.3.2 For endnotes

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

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

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

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

6.4 Font style

6.4.1 For line number

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

6.4.2 For the lemma

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

6.5 Styles of notes content

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

6.6 Arbitrary code at the beginning of notes

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

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

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

6.7 Options for footnotes in columns

6.7.1 Alignment

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

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

`\justifying` to have text justified, as usual with \LaTeX . You can also let `<code>` empty.

`\raggedright` to have text left aligned, but *without hyphenation*. That is the default `eledmac` setting.

`\RaggedRight` to have text left aligned *with hyphenation*.

`\raggedleft` to have text right aligned, but *without hyphenation*.

`\RaggedLeft` to have text right aligned *with hyphenation*.

`\centering` to have text centered, but *without hyphenation*.

`\Centering` to have text centered *with hyphenation*.

6.7.2 Size of the columns

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

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

6.8 Options for paragraphed footnotes

6.8.1 Mark separation of notes

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

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

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

6.8.2 Ragging

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

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

6.9 Options for block of notes

6.9.1 Text before notes

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

6.9.2 Spacing

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

Be careful, the standard L^AT_EX footnote rule, which is used by eledmac, 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 eledmac, decreases 3pt. These 3pt are not changed by this command.

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

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

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

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

6.9.3 Rule

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

Be careful, the standard L^AT_EX footnote rule, which is used by `eledmac`, 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 `\beforenotesX[<s>]{<l>}`. The default value is 0pt.

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

6.9.4 Maximum height

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

`\maxhnotesX` `\maxhnotesX[<s>]{<l>}` is the same as previous, but for familiar footnotes.

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

6.10 Footnotes and the `reledpar` columns

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

`\noteswidthliketwocolumnsX`

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

6.11 Endnotes in one paragraph

`\Xendparagraph` By default, any new endnote starts a new paragraph. Use `\Xendparagraph[<series>]` 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.

`\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.

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

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

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

If it is known that each stanza line will fit on more than 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. 41.

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

8.3 Repeating stanza indents

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

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

is like

```
\setstanzaindents{0,1,0,1,0,1,0,1,0,1,0}
```


Be careful: the feature is changed in eledmac 1.5.1. See Appendix A.3 p. 287.

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

`\sethangingsymbol`

It is possible to insert a symbol in each line of hanging verse, as in French typography for '['. To insert it in `reledmac`, use macro `\hangingsymbol⟨h⟩` with this code. In the example of French typography, do

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

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

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

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 Hanging symbol

`\sethangingsymbol`

It is possible to insert a symbol on each line of hanging verse, as in French typography for '['. To insert in `reledmac`, use macro `\sethangingsymbol{⟨symbol⟩}`. Fro exemple:

```
\hangingsymbol{[,}
```

8.9 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.

8.10 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. 31).

`thestanza`

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

```
\renewcommand{\thestanza}{%
```

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

You can change the value of the `stanza` counter with the usual commands of \TeX .
`\stanzanumwrapper` You can redefine `\stanzanumwrapper` in order to modify the way the stanza number is inserted in the flow of text. Default value is:

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

8.11 Various tools

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

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

9 Grouping

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

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

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

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

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

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

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

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

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

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

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

10 Cross referencing

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

10.1 Basic use

`\edlabel` First you place a label in the text using the command `\edlabel{<lab>}`. *<lab>* can be almost anything you like, including letters, numbers, punctuation, or a combination—anything but spaces; you might say `\edlabel{toves-3}`, for example.¹⁸

`\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

`\edlineref` page, line, sub-line and pstart on which the `\edlabel{<lab>}` command occurred.

`\sublineref` An `\edlabel` command may appear in the main text, or in the first argument of

`\pstartref` `\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 `ℒTeX.aux` file. You will need to process your document through `ℒTeX` twice in order for the references to be resolved.

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

10.2 Refer to a critical notes

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

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

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

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

¹⁸More precisely, you should stick to characters in the `ℒTeX` categories of “letter” and “other”.

10.3 Cross-referencing which return a number in any case

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

Where #1 stands for the reference.

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

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. 23 above) and sets the beginning page, line and subline numbers to those of the place where `\edlabel{mouse}` was placed, and the ending numbers to those where `\edlabel{elephant}` occurs.

10.4 Not automatic cross-referencing

`\edmakelabel`

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

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

10.5 Normal \TeX cross-referencing

`\label`
`\ref`
`\pageref`

The normal `\label`, `\ref` and `\pageref` macros may be used within numbered text,

and operate in the familiar fashion.

10.6 References to lines commented in the apparatus

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

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

If you use `\linenum` (5.2.5 p. 23) to refer to these labels, `reledmac` will use your line settings to refer to the passage.

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

`\setapprefprefixsingle` 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.

`\Xtwoinesappref` If you use `\Xtwoines`, `\Xmorethantwoines`, `\Xtwoinesbutnotmore` and/or `\Xtwoinesonlyinsamepage` (6.2.2 p. 30) *without the optional series argument*, the setting will also be available for `\appref`.

`\Xmorethantwoinesappref` The commands `\Xtwoinesappref{<text>}`, `\Xmorethantwoinesappref{<text>}`, `\Xtwoinesbutnotmoreappref` `\Xtwoinesonlyinsamepageappref` 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>}`.

`\Xendtwoinesapprefwithpage` If you use one of `\Xendtwoines`, `\Xendmorethantwoines`, `\Xendtwoinesbutnotmore`, `\Xendtwoinesonlyinsamepage` (6.2.2 p. 30) *without the optional series argument*, the setting will also be available for `\apprefwithpage`.

`\Xendtwoinesbutnotmoreapprefwithpage` The commands `\Xendtwoinesappref{<text>}`, `\Xendmorethantwoinesappref{<text>}`, `\Xendtwoinesbutnotmoreappref`, `\Xendtwoinesonlyinsamepageappref` can also be used, if you only want to change the reference style of `\apprefwithpage`.

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

11 Side notes

11.1 Basis

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

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

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

`\ledrightnote` The package's default setting is

`\ledsidenote` `\sidenotemargin{right}`

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

If two, say, `\ledleftnote`, commands are called in the same line the second `<text>` will obliterate the first. There is no problem though with having both a left and a right sidenote on the same line.

11.2 Setting

11.2.1 Width

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

`\ledrsnotewidth`

11.2.2 Vertical position

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

`\leftnoteupfalse`

11.2.3 Distance to the main text

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

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

`\ledlsnotefontsetup`

`\ledrsnotefontsetup`

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

These can of course be changed to suit.

11.2.4 Separator between notes

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

12 Indexing

12.1 Basis

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

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

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

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

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

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

12.2 Separator between page and line numbers

`\pagelinesep` The page & linenummer 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 separator¹⁹.

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

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

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

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.²⁰

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

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

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

1. Use `xindy+hyperref` option when loading the `reledmac` package. When you run (Xe/Lua) \LaTeX 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.²¹

12.4 Advanced setting

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

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

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

13 Tabular material

\LaTeX ’s normal `tabular` and `array` environments cannot be used where line numbering is being done; more precisely, they can be used but with odd results, so don’t use

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

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

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.
`edtabularr`

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

1	2	3
a	bb	ccc
AAA	BB	C

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

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

For example:

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

produces the following parallel pair of verses.

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

`\edtabcolsep` The distance between the columns is controlled by the length `\edtabcolsep`.
`\spreadmath` `\spreadmath{<math>}` typesets `{<math>}` but the `{<math>}` has no effect on the
`\spreadtext` calculation of column widths. `\spreadtext{<text>}` is the analagous command for use
in `edtabular` environments.

```

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

```

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

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

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

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

```

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

```

$$\begin{array}{cccccc}
1 & 2 & 3 & 4 & & 5 \\
Q & & & fd & h & qwertziohg \\
v & wptz & x & y & & vb \\
g & nnn & & & \underbrace{\hspace{2cm}} & \\
k & & & pq & & dgh \\
& \underbrace{\hspace{2cm}} & l & co & ghweropjklmbvcxys & \\
1 & 2 & 3 & & \underline{\hspace{2cm}} &
\end{array}$$

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

```

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

```

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

```

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

```

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

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

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

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

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

For example:

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

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

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

`\edvertdots`

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

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

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

14 Sectioning commands

14.1 Sectioning commands without line numbers or critical notes

The standard sectioning commands (`\chapter`, `\section` etc.) can be used inside numbered text. In this case, you must call them as an optional argument of `\pstart` (4.2.3 p. 16):

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

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

14.2 Sectioning commands with line numbering and critical notes

You have to use the following commands:

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

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

```
\pstart
\eledsection*{xxxx\ledsidenote{section}}
```

```

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

```

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

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

14.3 Optimization

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

15 Quotation environments

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

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

16 Page breaks

16.1 Control page breaking

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

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

These commands have effect only at the second run.

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

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

16.2 Prevent page break in a long verses

`\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, not more. It works at the third run, or at fourth run with `eledpar`. By default, when a long verse runs normally between two pages, a page break will be placed at the beginning of the verse. However, if you have added `\ledpbsetting{after}`, the page break will be placed at the end of the long verse, and the page containing the long verse will have one extra line.

17 Miscellaneous

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

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

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

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

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

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

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

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

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

17.1 Known and suspected limitations

17.2 ‘No room for a new’

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

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

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

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

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

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

17.2.1 Marginal notes

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

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

17.2.2 Paragraph shape

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

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

17.2.3 Paragraphed footnotes

The restriction on explicit line-breaking in paragraphed footnotes, mentioned in a footnote 17 p. 29, and described in more detail on XII.6.3 p. 147, really is a nuisance if that

is something you need to do. There are some possible solutions, described by Michael Downes, but this area remains unsatisfactory.

`\footfudgefiddle`

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

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

17.2.4 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 this²² 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:

²²Reported by Dirk-Jan Dekker in the CTT thread 'Incompatibility of "color" package' on 2003/08/28.


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

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

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

17.3 Parallel typesetting

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

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

I Implementation overview

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

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

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

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

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

II Preliminaries

II.1 Links with original `edmac`

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

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

II.2 Package declaration

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

```

1 %<*code>
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{reledmac}[2015/08/13 v2.1.1 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 for paragraphed notes.

```

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

```

```

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

```

II.5 Boolean flags

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

```

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

```

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

```

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

```

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

```

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

```

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

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

```

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

```

II.6 Messages

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

`\reledmac@warning` Write a warning message.

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

`\reledmac@error` Write an error message.

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

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

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

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

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

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

```

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

```
109 %
```

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

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

```
112 %
```

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

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

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

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

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

```
121 %
```

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

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

```
124 %
```

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

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

```
128 %
```

```

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

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

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

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

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

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

```

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```



```

\led@err@ManySidenotes97 \newcommand{\led@err@ManySidenotes}{%
\led@err@ManyLeftnotes98   \ifledRcol{%
\led@err@ManyRightnotes99   \reledmac@warning{\itemcount@\space sidenotes on line \the\line@numR\
space p. \the\page@numR}%
200   \else%
201   \reledmac@warning{\itemcount@\space sidenotes on line \the\line@num\
space p. \the\page@num}%
202   \fi%
203 }%
204 \newcommand{\led@err@ManyLeftnotes}{%
205   \ifledRcol{%
206   \reledmac@warning{\itemcount@\space leftnotes on line \the\line@numR\
space p. \the\page@numR}%
207   \else%
208   \reledmac@warning{\itemcount@\space leftnotes on line \the\line@num\
space p. \the\page@num}%
209   \fi%
210 }%
211 \newcommand{\led@err@ManyRightnotes}{%
212   \ifledRcol{%
213   \reledmac@warning{\itemcount@\space rightnotes on line \the\line@numR\
space p. \the\page@numR}%
214   \else%
215   \reledmac@warning{\itemcount@\space rightnotes on line \the\line@num\
space p. \the\page@num}%
216   \fi%
217 }%
218 %

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

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

```

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```

II.7 Gobbling

Here, we define some commands which gobble their arguments.

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

II.8 Miscellaneous commands

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

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

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

Initialize it as `\empty`.

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

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

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

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

II.9 Prepare reledpar

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

`\ifl@pairing`

`\ifl@dpaging`

`\l@dpagingtrue`

`\l@dpagingfalse`

`\ifl@printingpages`

`\l@printingpagestrue`

`\l@printingpagesfalse`

`\ifl@printingcolumns`

`\l@printingcolumnstrue`

`\l@printingcolumnsfalse`

`\l@dpairingtrue`

`\l@dpairingfalse`

`\ifpst@rtedL`

`\pst@rtedLtrue`

`\pst@rtedLfalse`

`\l@dnumpstartsl`

`\ifledRcol`

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

```

288 \newcount\l@dnumpstartsL
289 %

```

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

```

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

```

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

```

293 \newif\ifnumberingR
294 %

```

III Sectioning commands

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

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

```

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

```

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

`\numberingtrue`
`\numberingfalse`

```

299 \newif\ifnumbering
300 %

```

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

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

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

```

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

```

The tools for section's title commands are called:

- Define an empty list of pstart number where sectioning commands are called.
- Input auxiliary file with the description of section titles.
- Open the same auxiliary file to write in.

```

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

```

```

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

```

`\endnumbering` `\endnumbering` must follow the last text for a numbered section. It takes care of notifying you when changes have been noted in the input that require running the file through again to move everything to the right place.

```

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

```

```

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

```

`\pausenumbering` The `\pausenumbering` macro is just the same as `\endnumbering`, but with the `\ifnumbering` flag set to true, to show that numbering continues across the gap.²³

`\resumenumbering`

```

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

```

The `\resumenumbering` macro is a bit more involved, but not much. It does most of the same things as `\beginnumbering`, but without resetting the various counters. Note that no check is made by `\resumenumbering` to ensure that `\pausenumbering` was actually invoked.

```

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

```

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

```

407 \beginnumbering
408 \fi}
409
410
411 %

```

IV List macros

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

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

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

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

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

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

```

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

```

\list@clear The `\list@clear` macro just initializes a list to the empty list; it is no different from `\list@create` in its effect, but it is in its semantic .

```

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

```

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

```

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

```



```

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

```

\xleft@appenditem \xleft@appenditem expands an item and appends it to the left end of a list macro; it is otherwise identical to \xright@appenditem.

```

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

```

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

```

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

```

V Line counting

V.1 Choosing the system of lineation

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

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

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

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

```

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

```

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

```

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

```

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

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

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

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

If the argument is page.

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

If the argument is pstart.

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

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

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

In other case, it is an error.

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

```

479 \fi}}
480 %

```

V.2 Line number margin

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

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

`\l@dgetline@margin`

```

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

```

V.3 Line number initialization and increment

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

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

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

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

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

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

```

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

```

V.4 Line number locking

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

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

```

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

```

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

```

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

```

```

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

```

V.5 Line number style

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

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

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

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

`\sublinenumr@p`

```

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

```

Initialise the number styles to arabic.

```

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

```

V.6 Line number printing

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

`\rightlinenum`

`\linenumsep`

`\numlabfont`

`\ledlinenum`

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

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

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

```

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

```

V.7 Line number counters and lists

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

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

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

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

```

633 \newcount\line@num

```

634 %

`\subline@num` The count `\subline@num` stores a sub-line number that qualifies `\line@num`. For example, line 10 might have sub-line numbers 1, 2 and 3, which might be printed as lines 10.1, 10.2, 10.3.

635 `\newcount\subline@num`
636 %

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

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

`\sublines@false`

637 `\newif\ifsublines@`
638 %

`\absline@num` The count `\absline@num` stores the absolute number of lines since the start of the section: that is, the number we have actually printed, no matter what numbers we attached to them. This value is never printed on an output page, though `\line@num` will often be equal to it. It is used internally to keep track of where notes are to appear and where new pages start: using this value rather than `\line@num` is a lot simpler, because it does not depend on the lineation system in use.

639 `\newcount\absline@num`
640 %

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

V.8 Line number locking counter

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

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

V.9 Line number associated to lemma

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

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

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

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

```
23|35|0|24|3|0|OT1/cm/r/n.
```

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

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

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

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

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

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

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

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

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

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

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

An action code of -5000 or less sets the current visible line number (either the line number or the sub-line number, whichever is currently being advanced) to a specific positive value. The value of the code is $-(5000 + n)$, where n is the value (always ≥ 0) assigned to the current line number. Action codes of this type are added to the list by the `\set@line@action` macro, as called to implement the `\advanceline` and `\setline` macros: this action only occurs when the user has specified some change to the line numbers using those macros. Normally `reledmac` computes the visible line numbers from the absolute line numbers with reference to the other action codes and the settings they invoke; it does not require an entry in the action-code list for every line.

Here are the commands to create these lists:

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

```

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

```

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

`\endpage@num`

`\endline@num`

`\endsubline@num`

```

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

```

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

`\noteschanged@true`

`\noteschanged@false`

```

655 \newif\ifnoteschanged@
656 %

```

`\resetprevline@` Inside the apparatus, at each note, the line number is stored in a macro called `\prevlineX`, where X is the letter of the current series. This macro is called when using `\Xnumberonlyfirstinline`. This macro must be reset at the same time as the line number. The `\resetprevline@` does this resetting for every series.

```

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

```

`\resetprevpage@num` Inside the apparatus, at each note, the page number is stored in a macro called `\prevpageX@num`, where X is the letter of the current series. This macro is called when using `\Xparafootsep` or `\parafootsepX`. This macro must be reset at the beginning of each numbered section. The `\resetprevpage@` command resets this macro for every series.

```

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

```

```

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

```

V.10 Reading the line-list file

`\read@linelist` `\read@linelist{⟨file⟩}` is the control sequence that is called by `\beginnumbering` (via `\line@list@stuff`) to open and process a line-list file; its argument is the name of the file. . First, it clear all previous line's list.

```

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

```

When using `reledpar`, make sure that the `\maxlinesinpar@list` is empty (otherwise things will be thrown out of kilter if there is any old stuff still hanging in there).

```

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

```

Now get the file and interpret it. When the file is there we start a new group and make some special definitions we will need to process it. It is a sequence of \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 \TeX context. We ignore carriage returns, since if we are in horizontal mode they can get interpreted as spaces to be printed.

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

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

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

```

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

```

When the reading is done, we are all through with the line-list file. All the information we needed from it will now be encoded in our list macros.

Finally, we initialize the `\next@actionline` and `\next@action` macros, which specify where and what the next action to be taken is.

```

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

```

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

```

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

```

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

```

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

```

This version of `\read@linelist` creates list macros containing data for the entire section, so they could get rather large. It would be no more difficult to read the line-list

file incrementally rather than all at once: we could read, at the start of each paragraph, only the commands relating to that paragraph. But this would require that we have two line-lists open at once, one for reading, one for writing, and on systems without version numbers we would have to do some file renaming outside of \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 say `\global`. This is because we want them to affect only the counter values within the current group when nested calls of `\@ref` occur. (The code assumes throughout that the value of `\globaldefs` is zero.)

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

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

```

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

```

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

`\@nl@reg` In order to get the `\setlinenum` to work Peter Wilson had to slip in some new code at the start of the macro, to get the timing of the actions correct. The problem was that his original naive implementation of `\setlinenum` had a unfortunate tendency to change the number of the last line of the *preceding* paragraph. The new code is sort of based on the page number handling and `\setline`. It seems that a lot of fiddling with the line number internals is required.

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

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

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

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

```

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

```

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

```

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

```

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

```

765   \ifcase\@lock
766     \or

```

```

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

```

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

```

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

```

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

```

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

```



```

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

```

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

\@lopL **\@lopR**

```

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

```

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

```

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

```

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

```

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

```

```

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

```

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

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

```

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

```

\sub@action \sub@action adds an entry to the action-code list to turn sub-lineation on or off, according to the current value of the \ifsublines@ flag.

```

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

```

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

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

```

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

```

```

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

```

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

```

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

```

```

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

```

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

```

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

```

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

```

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

```

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

\hidenum

```

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

```

```

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

```

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

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

```

1055 \newcount\insert@count
1056 %

```

- #2, a sequence of other line-list-file commands, executed to determine the ending line-number. (This may also include other \@ref commands, corresponding to uses of \edtext within the first argument of another instance of \edtext.)

\dummy@ref When nesting of \@ref commands does occur, it is necessary to temporarily redefine \@ref within \@ref, so that we are only doing one of these at a time.

```

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

```

\@ref@reg The first thing \@ref (i.e. \@ref@reg) itself does is to add the specified number of items to the \insertlines@list list.

```

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

```

Next, process the second argument to determine the page and line numbers for the end of this lemma. We temporarily equate \@ref to a different macro that just executes its argument, so that nested \@ref commands are just skipped this time. Some other macros need to be temporarily redefined to suppress their action.

```

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

```

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

```

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

```

Create a list which stores every second argument of each `\@sw` in this lemma, at this level. Also set the boolean about the use of lemma in this edtext level to false.

```

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

```

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

```

1099   #2%
1100 %

```

Now, we store the list of `\@sw` of this current `\edtext` as an element of the global list of list of `\@sw` for a `\edtext` depth.

```

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

```



```

1108 \fi%
1109 %

Decrease edtext level counter.

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

1112 }
1113
1114 %

```

V.12 Writing to the line-list file

We have now defined all the counters, lists, and commands involved in reading the line-list file at the start of a section. Now we will cover the commands that `reledmac` uses within the text of a section to write commands out to the line-list.

`\linenum@out` The file will be opened on output stream `\linenum@out`.

```

1115 \newwrite\linenum@out
1116 %

```

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

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

```

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

```

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

```

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

```

\line@list@stuff The `\line@list@stuff{<file>}` macro, which is called by `\beginnumbering`, performs all the line-list operations needed at the start of a section. Its argument is the name of the line-list file.

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

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

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

Now close the current output line-list file, if any, and open a new one. The first time we open a line-list file for output, we do it using `\immediate`, and clear the `\iffirst@linenum@out@` flag.

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

If we get here, then this is not the first line-list we have seen, so we do not open or close the files immediately.

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

\new@line The `\new@line` macro sends the `\@nl` command to the line-list file, to mark the start of a new text line, and its page number.

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

```

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

```

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

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

\flag@end

```

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

```

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

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

```

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

```

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

```

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

```

\setline You can use `\setline{<num>}` in running text (i.e., within `\pstart... \pend`) to set the current visible line-number to a specified positive value.

```

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

```

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

```

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

```

VI Marking text for notes

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

The `\edtext` macro takes two arguments.

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

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

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

The recursive operation of `\edtext` will fail if you try to use a copy that is called something other than `\edtext`. In order to handle recursion, `\edtext` needs to redefine its own definition temporarily at one point, and that does not work if the macro you are calling is not actually named `\edtext`. There is no problem as long as `\edtext` is not invoked in the first argument. If you want to call `\edtext` something else, it is best to create instead a macro that expands to an invocation of `\edtext`, rather than copying `\edtext` and giving it a new name; otherwise you will need to add an appropriate definition for your new macro to `\morenoexpands`.

Side effects of our line-numbering code make it impossible to use the usual footnote macros directly within a paragraph whose lines are numbered (see comments to

`\do@line`, VII.2.1 p. 122). Instead, the appropriate note-generating command is appended to the list macro `\inserts@list`, and when `\pend` completes the paragraph it inserts all the notes at the proper places.

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

VI.1 `\edtext` itself

The various note-generating macros might want to request that commands be executed not at once, but in close connection with the start or end of the lemma. For example, footnote numbers in the text should be connected to the end of the lemma; or, instead of a single macro to create a note listing variants, you might want to use several macros in series to create individual variants, which would each add information to a private macro or token register, which in turn would be formatted and output when all of #2 for the lemma has been read.

`\end@lemmas` To accomodate this, we provide a list macro to which macros may add commands that should subsequently be executed at the end of the lemma when that lemma is added to the text of the paragraph. A macro should add its contribution to `\end@lemmas` by using `\xleft@appenditem`. (Anything that needs to be done at the *start* of the lemma may be handled using `\aftergroup`, since the commands specified within `\edtext`'s second argument are executed within a group that ends just before the lemma is added to the main text.)

`\end@lemmas` is intended for the few things that need to be associated with the end of the lemma, like footnote numbers. Such numbers are not implemented in the current version, and indeed no use is currently made of `\end@lemmas` or of the `\aftergroup` trick. The general approach would be to define a macro to be used within the second argument of `\edtext` that would add the appropriate command to `\end@lemmas`.

Commands that are added to this list should always take care not to do anything that adds possible line-breaks to the output; otherwise line numbering could be thrown off.

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

`\dummy@edtext` We now need to define a number of macros that allow us to weed out nested instances of `\edtext`, and other problematic macros, from our lemma. This is similar to what we did in reading the line-list file using `\dummy@ref` and various redefinitions—and that is because nested `\edtexts` macros create nested `\@ref` entries in the line-list file.

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

`\dummy@edtext@showlemma` Some time, we want to obtain only the first argument of `\edtext`, while also wrapping it in `\showlemma`. For example, when printing a `\eledsection`.

```

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

```

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

`\no@expands`
`\morenoexpands`

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

The first class is font-changing macros. We suppress expansion for them by letting them become equal to zero.²⁴ 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 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 $\text{Lua}\TeX$ or $\text{Xe}\mathbb{L}\TeX$.)

```

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

```

²⁴Since 'control sequences equivalent to characters are not expandable'—*The TeXbook*, answer to Exercise 20.14.


```

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

```

\@tag Now, we define an empty \@tag command. It will be redefine by \edtext: its value is the first argument. It will be used by the \Xfootnote commands.

```

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

```

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

```

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

```

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

```

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

```

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

```

1294 \ifnumberedpar%
1295 %

```

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

```

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

```

By default, we do not use \lemma

```

1298 \global\@lemmacommand@false%
1299 %

```

```

1300 \begingroup%
1301 %

```

We get the next series of samewords data in the list of samewords data for the current edtext level. We push them inside \sw@inthisedtext.

```

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

```

\@tag Our normal lemma is just argument #1; but that argument could have further invocations of `\edtext` within it. We get a copy of the lemma without any `\edtext` macros within it by temporarily redefining `\edtext` to just copy its first argument and ignore the other, and then expand #1 into `\@tag`, our lemma.

This is done within a group that starts here, in order to get the original `\edtext` restored; within this group we have also turned off the expansion of those control sequences commonly found within text that can cause trouble for us.

```

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

```

\l@d@nums Prepare more data for the benefit of note-generating macros: the line references and font specifier for this lemma go to `\l@d@nums`.

```

1322     \set@line%
1323 %

```

`\insert@count` will be altered by the note-generating macros: it counts the number of deferred footnotes or other insertions generated by this instance of `\edtext`. If we are in a right column (`reledpar`), we use `\insert@countR` instead of `\insert@count`.

```

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

```

Now process the note-generating macros in argument #2 (i.e., `\Afootnote`, `\lemma`, etc.). `\ignorespaces` is here to skip over any spaces that might appear at the start of #2; otherwise they wind up in the main text. Footnote and other macros that are

used within #2 should all end with `\ignorespaces` as well, to skip any spaces between macros when several are used in series.

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

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

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

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

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

Finally, we are ready to admit the first argument into the current paragraph.

It is important that we generate and output all the notes for this chunk of text *before* putting the text into the paragraph: notes that are referenced by line number should generally be tied to the start of the passage they gloss, not the end. That should all be done within the expansion of #2 above, or in `\aftergroup` commands within that expansion.

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

Finally, we add any insertions that are associated with the *end* of the lemma. Footnotes that are identified by symbols rather than by where the lemma begins in the main text need to be done here, and not above.

```
1345 \ifx\end@lemmas\empty \else%
1346 \gl@p\end@lemmas\to\x@lemma%
1347 \x@lemma%
1348 \global\let\x@lemma=\relax%
1349 \fi%
1350 \@ifundefined{xpg@main@language}{%if not polyglossia
1351 \flag@end}%
1352 {\if@RTL\flag@start\else\flag@end\fi% With polyglossia, you must
track whether the language reads left to right (English) or right to left
(Arabic).
1353 }%
1354 %
```

We switch to false some flags.

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

```

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

```

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

```

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

```

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

```

1372 \newif\ifnumberline
1373 \numberlinetrue
1374 %

```

`\set@line` The `\set@line` macro is called by `\edtext` to put the line-reference field and font specifier for the current block of text into `\l@d@nums`.

One instance of `\edtext` may generate several notes, or it may generate none — it is legitimate for argument #2 to `\edtext` to be empty. But `\flag@start` and `\flag@end` induce the generation of a single entry in `\line@list` during the next run, and it is vital to also remove one and only one `\line@list` entry here.

If no more lines are listed in `\line@list`, something is wrong — probably just some change in the input. We set all the numbers to zeros, following an old publishing convention for numerical references that have not yet been resolved.

```

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

```

```

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

```

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

```

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

```

VI.2 Substitute lemma

\lemma The `\lemma{<text>}` macro allows you to change the lemma that is passed on to the notes. Read about `\@tag` in normal `\edtext` macro for more details about `\sw@list@inedtext` and `\no@expands` (VI.1 p. 106).

```

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

```

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

```

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

```

```

1412 \fi
1413 %

```

\if@lemmacommand@ This boolean is set to TRUE inside a `\edtext` (or `\critext`) when a `\lemma` command is called. That is useful for some commands which can have a different behavior if the lemma in the note is different from the lemma in the main text.

```

1414 \newif\if@lemmacommand@%
1415 %

```

VI.3 Substitute line numbers

\linenum The `\linenum` macro can change any or all of the page and line numbers that are passed on to the notes.

As argument `\linenum` takes a set of seven parameters separated by vertical bars, in the format used internally for `\l@d@nums` (see V.9 p. 81): the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma. However, you can omit any parameters you do not want to change, and you can omit a string of vertical bars at the end of the argument. Hence `\linenum{18|4|0|18|7|1|0}` is an invocation that changes all the parameters, but `\linenum{|3}` only changes the starting line number, and leaves the rest unaltered.

We use `\\` as an internal separator for the macro parameters.

```

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

```

\line@set `\linenum` calls `\line@set` to do the actual work; it looks at the first number in the argument to `\linenum`, sets the corresponding value in `\l@d@nums`, and then calls itself to process the next number in the `\linenum` argument, if there are more numbers in `\l@d@nums` to process.

```

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

```

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

```

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

```

VI.4 Lemma disambiguation

The mechanism which counts the occurrence of a same word in a same line is quite complex, because, when \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{\langle argument \rangle}+` is called from the beginning of the lineation to the end of the current line. We also store the same data for the preceding absolute line number, if it does not have `\sameword{\langle argument \rangle}`.
 - For each `\sameword` having the same argument, we subtract from its absolute rank the number stored for the paired `\sameword` argument and previous absolute line number. Consequently, we obtain the relative rank.
 - See the following example which explain how for same `\sameword` absolute ranks are transformed to relative rank.

```

At line 1:
absolute rank 1 becomes relative rank 1-0 = 1
1 is stored for this \sameword and the line 1
At line 2:
absolute rank 2 becomes relative rank 2-1 = 1
absolute rank 3 becomes relative rank 3-2 = 2
3 is stored for this \sameword and the line 2
At line 3:
no \sameword for this line.
3 is stored for this \sameword and the line 3
At line 4:
absolute rank 4 becomes relative rank 4-3 = 1
3 is stored for this \sameword and the line 4

```

2. Create lists of lists of `\sameword` by depth of `\edtext`. That is: create a list for `\edtext` of level 1, a list for `\edtext` of level 2, a list for `\edtext` of level 3 etc. For each `\edtext` in these list, we store all the relative rank of `\sameword` which are called as lemma information, that is 1) or called in the first argument of `\sameword` 2) or called in the `\lemma` macro of the second argument of `\sameword` AND marked by the optional argument of `\sameword` in first argument of `\edtext`.

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

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`.²⁵

Because there is a bug with `\detokenize` and \LaTeX when using non BMP characters²⁶, 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.

```

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

```

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

```

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

```

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

```

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

```

Then, write its value to the numbered file.

```

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

```

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

```

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

```

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

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

And print the word.

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

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

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

\@sw The command printed in the auxiliary files.

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

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

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

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

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

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

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

And do the same thing for the right side.

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

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

```

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

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

```

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

Just a precaution.

```

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

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

```

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

```

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

```

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

```

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

```

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

```

And the same for right side.

```

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

```

`\showwordrank`^{k62} % Finally, the way the rank will be printed.

```

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

```

```

1565 }%
1566 %

```

VII Paragraph decomposition and reassembly

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

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

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

`\ifnumberedpar@` When we first form the paragraph, it goes into a box register, `\raw@text`, instead

`\numberedpar@true` of onto the current vertical list. The `\ifnumberedpar@` flag will be `true` while a paragraph is being processed in that way. `\num@lines` will store the number of lines in the

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

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

`\one@line` paragraph.

`\par@line`

```

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

```

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

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

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

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

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

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

`\thepstart` paragraph.

```

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

```

```

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

```

```
1631 %
```

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

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

We set all the usual interline penalties to zero and then immediately call \endgraf to end the paragraph; this ensures that there will be no large interline penalties to prevent us from slicing the paragraph into pieces. These penalties revert to the values that you set when the group for the \vbox ends. Then we call \do@line to slice a line off the top of the paragraph, add a line number and footnotes, and restore it to the page; we keep doing this until there are not any more lines left.

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

We check if lineation is by pstart: in this case, we reset line number, but only in the second line of the pstart. We can't reset line number at the beginning of \pstart, as \setline is parsed at the end of previous \pend, and so, we must do it at the end of first line of pstart.

```
1644 \csnumdef{pstartline}{0}%
1645 \loop\ifvbox\raw@text%
1646   \csnumdef{pstartline}{\pstartline+1}%
1647   \do@line%
1648   \ifbypstart@%
1649     \ifnumequal{\pstartline}{1}{\setline{1}\resetprevline@}{}%
1650   \fi%
1651 \repeat%
1652 %
```

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

```
1653 \flush@notes%
1654 \endgroup%
1655 \ignorespaces%
1656 %
```

Increase pstart counter.

```
1657 \ifnumberpstart%
1658   \pstartnumtrue%
1659 \fi%
1660 \addtocounter{pstart}{1}%
1661 %
```

Restore paragraph, nobreak setting and autopar setting.

```

1662 \normal@pars%
1663 \@oldnobreak%
1664 \ifautopar%
1665 \autopar%
1666 \fi%
1667 %

```

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

```

1668 \ifstrepty{#1}{\at@every@pend}{\noindent#1}%
1669 }
1670
1671 %

```

Here, two macros to insert content after every `\pend`, between numbered line. `\AtEveryPend` is the user macro, `\at@every@pend` is macro set by it.

```

\AtEveryPend 72
\at@every@pend 73 \newcommand{\AtEveryPend}[1]{%
1674 \ifstrepty{#1}%
1675 {\xdef\at@every@pend{}}%
1676 {\gdef\at@every@pend{\noindent#1}}%
1677 }%
1678 \xdef\at@every@pend{}%
1679
1680 %

```

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

```

1681 \newcommand*\l@dzzeropenalties{%
1682 \brokenpenalty \z@ \clubpenalty \z@
1683 \displaywidowpenalty \z@ \interlinepenalty \z@ \predisplaypenalty \z@
1684 \postdisplaypenalty \z@ \widowpenalty \z@}
1685
1686 %

```

`\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.

```

1687 \newif\ifautopar
1688 \autoparfalse
1689 \newcommand*\autopar{%
1690   \ifledRcol
1691     \ifnumberingR \else
1692       \led@err@AutoparNotNumbered
1693       \beginnumberingR
1694       \fi
1695     \else
1696       \ifnumbering \else
1697         \led@err@AutoparNotNumbered
1698         \beginnumbering
1699         \fi
1700       \fi
1701       \autopartrue
1702       \everypar{\setbox0=\lastbox
1703         \endgraf \vskip-\parskip
1704         \pstart \noindent \kern\wd0 \ifnumberpstart\ifinstanza\else\thepstart\fi\fi
1705         \let\par=\pend}%
1706       \ignorespaces}
1707 %

```

\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.

```

1708 \newcommand*\normal@pars{\everypar{}\let\par\endgraf}
1709
1710 %

```

\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.

```

1711 \newif\ifautopar@pause
1712 %

```

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`

```

1713 \newcommand*{\l@dunhbox@line}[1]{\unhbox #1}
1714 \newcommand*{\do@line}{%
1715   {\vbadness=10000
1716     \splittopskip=\z@
1717     \do@linehook
1718   \l@demptyd@ta
1719     \global\setbox\one@line=\vsplit\raw@text to\baselineskip}%
1720   \unvbox\one@line \global\setbox\one@line=\lastbox
1721   \getline@num
1722   \IfStrEq{\led@pb@setting}{before}{\led@check@pb\led@check@nopb}{%}
1723   \ifnum\@lock>\@ne
1724     \inserthangingsymboltrue
1725   \else
1726     \inserthangingsymbolfalse
1727   \fi
1728   \check@pb@in@verse
1729   \ifl@dhidenumber%
1730     \global\l@dhidenumberfalse%
1731     \f@x@l@cks%
1732   \else%
1733     \affixline@num%
1734   \fi%
1735   %

```

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

```

1736 \xifinlist{\the\l@dunpstartsL}{\eled@sections@}%
1737   {\print@eledsection}%
1738   {\print@line}%
1739   \IfStrEq{\led@pb@setting}{after}{\led@check@pb\led@check@nopb}{%}
1740   }%
1741   %

```

VII.2.2 Process for “normal” line

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

```

1742 \def\print@line{
1743   %

```

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

```

1744   \affixpstart@num%
1745   %

```

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

```
1746 \hb@xt@ \linewidth{%
1747 %
```

User hook.

```
1748 \do@insidelinehook%
1749 %
```

Left line number

```
1750 \l@dld@ta%
1751 %
```

Restore marginal and footnotes.

```
1752 \add@inserts\affixside@note%
1753 %
```

Print left notes.

```
1754 \l@dlsn@te
1755 %
```

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

```
1756 {\ledllfill\hb@xt@ \wd\one@line{\new@line%
1757 %
```

If we use Lua_{TeX} then restore the direction.

```
1758 \ifluatex%
1759 \luatextextdir\l@luatextextdir@L%
1760 \fi%
1761 %
```

Insert, if needed, the hanging symbol.

```
1762 \inserthangingsymbol %Space keep for backward compatibility
1763 %
```

And so, print the line.

```
1764 \l@dunhbox@line{\one@line}}%
1765 %
```

Right line number

```
1766 \ledrlfill\l@drd@ta%
1767 %
```

Print right notes.

```
1768 \l@drsn@te
1769 }}%
1770 %
```

And reinsert penalties (for page breaking)...

```
1771 \add@penalties%
1772 }
1773 %
```

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.

```

1774 \def\print@eledsection{%
1775   \add@inserts\affixside@note%
1776   \numdef{\temp@}{\l@dnumpstartsL-1}%
1777   \xifinlist{\temp@}{\eled@sections@@}{\@nbreaktrue}{\@nbreakfalse}%
1778   \@eled@sectioningtrue%
1779   \csuse{eled@sectioning@\the\l@dnumpstartsL}%
1780   \@eled@sectioningfalse%
1781   \global\csundef{eled@sectioning@\the\l@dnumpstartsL}%
1782   \if@RTL%
1783     \hspace{-3\paperwidth}%
1784     {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1785   \else%
1786     \hspace{3\paperwidth}%
1787     {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
1788   \fi%
1789   \vskip-\baselineskip%
1790 }
1791 %

```

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.

```

1792 \newcommand*\do@linehook{}
1793 \newcommand*\do@insidelinehook{}
1794 %

```

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

```

1795 \newcommand*\dolinehook[1]{\gdef\do@linehook{#1}}%
1796 \newcommand*\doinsidelinehook[1]{\gdef\do@insidelinehook{#1}}%
1797
1798 %

```

VII.2.5 Sidenotes and marginal line number initialization

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

```

\l@dcsnotetext
\l@dcsnotetext@l
\l@dcsnotetext@r

```

```

1799 \newcommand*{\l@emptyd@ta}{%
1800   \gdef\l@dld@ta{%
1801     \gdef\l@drd@ta{%
1802       \gdef\l@dcsnotetext@l{%
1803         \gdef\l@dcsnotetext@r{%
1804           \gdef\l@dcsnotetext{}
1805
1806 %

```

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

\l@drsn@te

```

1807 \newcommand{\l@dlsn@te}{%
1808   \hb@xt@ \z@{\hss\box\l@dldp@rbox\kern\ledlsnotesep}}
1809 \newcommand{\l@drsn@te}{%
1810   \hb@xt@ \z@{\kern\ledrsnotesep\box\l@drp@rbox\hss}}
1811
1812 %

```

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

\ledrlfill

```

1813 \newcommand*{\ledllfill}{\hfil}
1814 \newcommand*{\ledrlfill}{}
1815
1816 %

```

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.

```

1817 \newcommand*{\getline@num}{%
1818   \global\advance\absline@num \@ne%
1819   \do@actions
1820   \do@ballast
1821   \ifnumberline
1822     \ifsublines@
1823       \ifnum\sub@lock<\tw@
1824         \global\advance\subline@num \@ne
1825       \fi
1826     \else
1827       \ifnum\@lock<\tw@
1828         \global\advance\line@num \@ne
1829         \global\subline@num \z@
1830       \fi
1831     \fi
1832   \fi
1833 }
1834 %

```

`\do@ballast` The real work in the macro above is done in `\do@actions`, but before we plunge into that, let's get `\do@ballast` out of the way. This macro looks to see if there is an action to be performed on the *next* line, and if it is going to be a page break action, `\do@ballast` decreases the count `\ballast@count` counter by the amount of ballast. This means, in practice, that when `\add@penalties` assigns penalties at this point, TeX will be given extra encouragement to break the page here (see XI.2 p. 135).

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

```
1835 \newcount\ballast@count
1836 \newcounter{ballast}
1837 \setcounter{ballast}{0}
1838 %
```

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.

```
1839 \newcommand*{\do@ballast}{\global\ballast@count \z@
1840 \begingroup
1841 \advance\absline@num \@ne
1842 \ifnum\next@actionline=\absline@num
1843 \ifnum\next@action>-1001\relax
1844 \global\advance\ballast@count by -\c@ballast
1845 \fi
1846 \fi
1847 \endgroup}
1848 %
```

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

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

```
1849 \newcommand*{\do@actions}{%
1850 \global\let\do@actions@next=\relax
1851 \ifnum\absline@num<\next@actionline\else
1852 %
```

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.

```
1853 \ifnum\next@action>-1001
1854 \global\page@num=\next@action
1855 \ifbypage@
1856 \global\line@num=\z@ \global\subline@num=\z@
1857 \resetprevline@
1858 \fi
1859 %
```

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`.)

```

1860 \else
1861 \ifnum\next@action<-4999
1862 \@l@dttempcnta=-\next@action
1863 \advance\@l@dttempcnta by -5001
1864 \ifsublines@
1865 \global\subline@num=\@l@dttempcnta
1866 \else
1867 \global\line@num=\@l@dttempcnta
1868 \fi
1869 %

```

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

```

1870 \else
1871 \@l@dttempcnta=-\next@action
1872 \advance\@l@dttempcnta by -1000
1873 \do@actions@fixedcode
1874 \fi
1875 \fi
1876 %

```

Now we get information about the next action off the list, and then set `\do@actions@next` so that we will call ourself recursively: the next action might also be for this line.

There is no warning if we find `\actionlines@list` empty, since that will always happen near the end of the section.

```

1877 \ifx\actionlines@list\empty
1878 \gdef\next@actionline{1000000}%
1879 \else
1880 \gl@p\actionlines@list\to\next@actionline
1881 \gl@p\actions@list\to\next@action
1882 \global\let\do@actions@next=\do@actions
1883 \fi
1884 \fi
1885 %

```

Make the recursive call, if necessary.

```

1886 \do@actions@next}
1887
1888 %

```

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

```

1889 \newcommand*{\do@actions@fixedcode}{%
1890 \ifcase\@l@dttempcnta
1891 \or% % 1001
1892 \global\sublines@true

```

```

1893 \or% % 1002
1894 \global\sublines@false
1895 \or% % 1003
1896 \global\@lock=\@ne
1897 \or% % 1004
1898 \ifnum\@lock=\tw@
1899 \global\@lock=\thr@@
1900 \else
1901 \global\@lock=\z@
1902 \fi
1903 \or% % 1005
1904 \global\sub@lock=\@ne
1905 \or% % 1006
1906 \ifnum\sub@lock=\tw@
1907 \global\sub@lock=\thr@@
1908 \else
1909 \global\sub@lock=\z@
1910 \fi
1911 \or% % 1007
1912 \l@dskipnumbertrue
1913 \or% % 1008
1914 \l@dskipversenumbertrue%
1915 \or% % 1009
1916 \l@dhiddenumbertrue
1917 \else
1918 \led@warn@BadAction
1919 \fi}
1920
1921
1922 %

```

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@drrd@ta` if required.

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

$$\begin{aligned}
 n &= \text{int}((\text{linenum} - \text{firstlinenum}) / \text{linenumincrement}) \\
 m &= \text{firstlinenum} + (n \times \text{linenumincrement})
 \end{aligned}$$

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

We compute, in the scratch counter `\@l@dttempcnta`, 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@dttempcntb` for comparison.

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

```
1923 \newcommand*{\affixline@num}{%
1924 %
```

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).

```
1925 \ifledgroupnotesL@else
1926 \ifnumberline
1927 \ifl@dskipnumber
1928 \global\l@dskipnumberfalse
1929 \else
1930 \ifsublines@
1931 \@l@tempcntb=\subline@num
1932 \ifnum\subline@num>\c@firstsublinenum
1933 \@l@tempcnta=\subline@num
1934 \advance\@l@tempcnta by-\c@firstsublinenum
1935 \divide\@l@tempcnta by\c@sublinenumincrement
1936 \multiply\@l@tempcnta by\c@sublinenumincrement
1937 \advance\@l@tempcnta by\c@firstsublinenum
1938 \else
1939 \@l@tempcnta=\c@firstsublinenum
1940 \fi
1941 %
```

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.

```
1942 \ch@cksub@l@ck
1943 %
```

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

```
1944 \else
1945 \@l@tempcntb=\line@num
1946 %
```

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

```
1947 \ifx\linenumberlist\empty
1948 \ifnum\line@num>\c@firstlinenum
1949 \@l@tempcnta=\line@num
1950 \advance\@l@tempcnta by-\c@firstlinenum
1951 \divide\@l@tempcnta by\c@linenumincrement
1952 \multiply\@l@tempcnta by\c@linenumincrement
1953 \advance\@l@tempcnta by\c@firstlinenum
1954 \else
1955 \@l@tempcnta=\c@firstlinenum
1956 \fi
1957 \else
1958 %
```

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

```

1959      \l@ldtempcnta=\line@num
1960      \edef\rem@inder{,\linenumberlist,\number\line@num,}%
1961      \edef\sc@n@list{\def\noexpand\sc@n@list
1962      #####1,\number\l@ldtempcnta,#####2|\def\noexpand\rem@inder
1963      {#####2}}}%
1964      \sc@n@list\expandafter\sc@n@list\rem@inder|}%
1965      \ifx\rem@inder\empty%
1966      \advance\l@ldtempcnta\@ne
1967      \fi
1968  %

```

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

```

1969      \ch@ck@l@ck
1970      \fi
1971  %

```

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

```

1972      \ifnum\l@ldtempcnta=\l@ldtempcntb
1973      \ifl@dskipversenumber\else
1974  %

```

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 $\mathcal{E}\mathcal{T}\mathcal{E}\mathcal{X}$ 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@ld@ta` A left line number is stored in `\l@ld@ta` and a right one in `\l@d@rd@ta`.

```

1975      \l@d@rd@ta
1976      \if@twocolumn
1977      \if@firstcolumn
1978      \gdef\l@ld@ta{\llap{\leftlinenum}}}%
1979      \else
1980      \gdef\l@d@rd@ta{\rlap{\rightlinenum}}}%
1981      \fi
1982      \else
1983      \l@ldtempcntb=\line@margin
1984      \ifnum\l@ldtempcntb>\@ne
1985      \advance\l@ldtempcntb \page@num
1986      \fi
1987      \ifodd\l@ldtempcntb

```

```

1987 \gdef\l@drd@ta{\rlap{{\rightlinenum}}}%
1988 \else
1989 \gdef\l@dld@ta{\llap{{\leftlinenum}}}%
1990 \fi
1991 \fi
1992 \fi
1993 \fi
1994 %

```

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

```

1995 \f@x@l@cks
1996 \fi
1997 \fi
1998 \fi
1999 }
2000 %
2001 %

```

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

```

2002 \newcommand*{\ch@cksub@l@ck}{%
2003 \ifcase\sub@lock
2004 \or
2005 \ifnum\sublock@disp=\@ne
2006 \l@l@tempcntb=\z@ \l@l@tempcnta=\@ne
2007 \fi
2008 \or
2009 \ifnum\sublock@disp=\tw@ \else
2010 \l@l@tempcntb=\z@ \l@l@tempcnta=\@ne
2011 \fi
2012 \or
2013 \ifnum\sublock@disp=\z@
2014 \l@l@tempcntb=\z@ \l@l@tempcnta=\@ne
2015 \fi
2016 \fi}
2017 %

```

Similarly for line numbers.

```

2018 \newcommand*{\ch@ck@l@ck}{%
2019 \ifcase\l@lock
2020 \or
2021 \ifnum\lock@disp=\@ne
2022 \l@l@tempcntb=\z@ \l@l@tempcnta=\@ne
2023 \fi
2024 \or
2025 \ifnum\lock@disp=\tw@ \else

```

```

2026         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2027         \fi
2028         \or
2029         \ifnum\lock@disp=\z@
2030         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2031         \fi
2032     \fi}
2033 %

```

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

```

2034 \newcommand*{\f@x@l@cks}{%
2035     \ifcase\@lock
2036     \or
2037     \global\@lock=\tw@
2038     \or \or
2039     \global\@lock=\z@
2040     \fi
2041     \ifcase\sub@lock
2042     \or
2043     \global\sub@lock=\tw@
2044     \or \or
2045     \global\sub@lock=\z@
2046     \fi}
2047 %
2048 %

```

X Pstart number printing in side

In side, the printing of pstart number is running like the printing of line number. There is only some differences:

`\affixpstart@num`
`\pstartnum`

- The pstarts counter is upgrade in the `\pend` command. Consequently, the `\affixpstart@num` command has not to upgrade it, unlike the `\affixline@num` which upgrades the lines counter.
- To print the pstart number only at the beginning of a pstart, and not in every line, a boolean test is made. The `\pstartnum` boolean is set to TRUE at every `\pend`. It is tried in the `\leftpstartnum` and `\rightpstartnum` commands. After the try, it is set to FALSE.

```

\leftpstartnum49
\rightpstartnum50 \newif\ifsidepstartnum
\ifsidepstartnum51 \newcommand*{\affixpstart@num}{%
2052     \ifsidepstartnum
2053     \if@twocolumn
2054     \if@firstcolumn
2055         \gdef\l@dld@ta{\llap{\leftpstartnum}}}%

```

```

2056         \else
2057             \gdef\l@drd@ta{\rlap{{\rightpstartnum}}}%
2058         \fi
2059     \else
2060         \@l@dttempcntb=\line@margin
2061         \ifnum\@l@dttempcntb>\@ne
2062             \advance\@l@dttempcntb \page@num
2063         \fi
2064         \ifodd\@l@dttempcntb
2065             \gdef\l@drd@ta{\rlap{{\rightpstartnum}}}%
2066         \else
2067             \gdef\l@dld@ta{\llap{{\leftpstartnum}}}%
2068         \fi
2069     \fi
2070 \fi
2071
2072 }
2073 %
2074
2075 \newif\ifpstartnum
2076 \pstartnumtrue
2077 \newcommand*{\leftpstartnum}{
2078     \ifpstartnum\thepstart
2079     \kern\linenumsep\fi
2080     \global\pstartnumfalse
2081 }
2082 \newcommand*{\rightpstartnum}{
2083     \ifpstartnum
2084     \kern\linenumsep
2085     \thepstart
2086     \fi
2087     \global\pstartnumfalse
2088 }
2089 %

```

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.

```
2090 \list@create{\inserts@list}
2091 %
```

`\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`.

```
2092 \newcommand*{\add@inserts}{%
2093   \global\let\add@inserts@next=\relax
2094   %
```

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

```
2095   \ifx\inserts@list\empty \else
2096   %
```

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.

```
2097   \ifx\next@insert\empty
2098     \ifx\insertlines@list\empty
2099       \global\noteschanged@true
2100       \gdef\next@insert{100000}%
2101     \else
2102       \gl@p\insertlines@list\to\next@insert
2103     \fi
2104   \fi
2105   %
```

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.

```
2106   \ifnum\next@insert=\absline@num
2107     \gl@p\inserts@list\to\@insert
2108     \@insert
2109     \global\let\@insert=\undefined
2110     \global\let\next@insert=\empty
2111     \global\let\add@inserts@next=\add@inserts
2112   \fi
2113 \fi
2114 %
```

Make the recursive call, if necessary.

```
2115 \add@inserts@next}
2116
2117 %
```

XI.2 Penalties

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

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

```
2118 \newcommand*{\add@penalties}{\@l@tempcnta=\ballast@count
2119 \ifnum\num@lines>\@ne
2120 \global\advance\par@line \@ne
2121 \ifnum\par@line=\@ne
2122 \advance\@l@tempcnta \clubpenalty
2123 \fi
2124 \@l@tempcntb=\par@line \advance\@l@tempcntb \@ne
2125 \ifnum\@l@tempcntb=\num@lines
2126 \advance\@l@tempcnta \widowpenalty
2127 \fi
2128 \ifnum\par@line<\num@lines
2129 \advance\@l@tempcnta \interlinepenalty
2130 \fi
2131 \fi
2132 \ifnum\@l@tempcnta=z@
2133 \relax
2134 \else
2135 \ifnum\@l@tempcnta>-10000
2136 \penalty\@l@tempcnta
2137 \else
2138 \penalty -10000
2139 \fi
2140 \fi}
2141
2142 %
```

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.

```

2143 \newcommand*{\flush@notes}{%
2144   \@xloop
2145   \ifx\inserts@list\empty \else
2146     \glp\inserts@list\to\@insert
2147     \@insert
2148     \global\let\@insert=\undefined
2149   \repeat}
2150
2151 %

```

\@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 says $\text{\@xloop} \dots \text{\if} \dots \text{\else} \dots \text{\repeat}$, 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.

```

2152 \def\@xloop#1\repeat{%
2153   \def\body{#1\expandafter\body\fi}%
2154   \body}
2155
2156 %

```

XII Critical footnotes

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

XII.1 Fonts

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

`\select@lemmafont` `\select@lemmafont` 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.

```

2157 \def\select@lemmafont#1|#2|#3|#4|#5|#6|#7|{\select@lemmafont#7|}
2158 \def\select@lemmafont#1/#2/#3/#4|{%
2159   {\fontencoding{#1}\fontfamily{#2}\fontseries{#3}\fontshape{#4}%
2160    \selectfont}
2161
2162 %

```

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.

```

2163 \newcommand*{\footnoteoptions@}[3][1=L,usedefault]{%
2164   \def\do##1{%
2165     \ifstrequal{#1}{L}{% In Leftside
2166       \xright@appenditem{\global\noexpand\settoggle{##1@}{#3}}\to\
inserts@list% Switch toggle, in all case
2167       \global\advance\insert@count \One% Increment the left insert
counter.
2168     }%
2169     {%
2170       \xright@appenditem{\global\noexpand\settoggle{##1@}{#3}}\to\
inserts@listR% Switch toggle, in all case
2171       \global\advance\insert@countR \One% Increment the right insert
counter insert.
2172     }%
2173   }%
2174   \notblank{#2}{\docsvlist{#2}}}% Parsing all options
2175 }
2176 %

```

XII.3 Notes language

`\footnotelang@lua` `\footnotelang@lua` is called to remember the information about the direction of a lemma when Lua_{TeX} is used.

```

2177 \newcommand*{\footnotelang@lua}[1][1=L,usedefault]{%
2178   \ifstrequal{#1}{L}{%
2179     \xright@appenditem{\csxdef{footnote@luatextextdir}{\the\luatextextdir
}}\to\inserts@list%Know the dir of lemma
2180     \global\advance\insert@count \One%
2181     \xright@appenditem{\csxdef{footnote@luatexpardir}{\the\luatexpardir
}}\to\inserts@list%Know the dir of lemma
2182     \global\advance\insert@count \One%
2183   }%

```

```

2184 {%
2185 \xright@appenditem{{\csxdef{footnote@luatextextdir}{\the\luatextextdir
}}} \to\inserts@listR%Know the dir of lemma
2186 \global\advance\insert@countR \@ne%
2187 \xright@appenditem{{\csxdef{footnote@luatexpardir}{\the\luatexpardir
}}} \to\inserts@listR%Know the dir of lemma
2188 \global\advance\insert@countR \@ne%
2189 }%
2190 }
2191 %

```

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

```

2192 \newcommandx*{\footnotelang@poly}[1][1=L,usedefault]{%
2193 \ifstrequal{#1}{L}{%
2194 \if@RTL%
2195 \xright@appenditem{{\csxdef{footnote@dir}{@RTLtrue}}}\to\
inserts@list%Know the language used in the lemma
2196 \global\advance\insert@count \@ne%
2197 \else
2198 \xright@appenditem{{\csxdef{footnote@dir}{@RTLfalse}}}\to\
inserts@list%Know the language of lemma
2199 \global\advance\insert@count \@ne%
2200 \fi%
2201 \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\language name}}}\
to\inserts@list%Know the language of lemma
2202 \global\advance\insert@count \@ne%
2203 }%
2204 {%
2205 \if@RTL
2206 \xright@appenditem{{\csxdef{footnote@dir}{@RTLtrue}}}\to\
inserts@listR%Know the language of lemma
2207 \global\advance\insert@countR \@ne%
2208 \else
2209 \xright@appenditem{{\csxdef{footnote@dir}{@RTLfalse}}}\to\
inserts@listR%Know the language of lemma
2210 \global\advance\insert@countR \@ne%
2211 \fi
2212 \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\language name}}}\
to\inserts@listR%Know the language of lemma
2213 \global\advance\insert@countR \@ne%
2214 }%
2215 }
2216 %

```

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`.

```

2217 \newcommand*{\footsplitskips}{%
2218   \interlinepenalty=\interfootnotelinepenalty
2219   \unless\ifl@dprintingpages%
2220     \floatingpenalty=\@MM%
2221   \fi%
2222   \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
2223   \leftskip=\z@skip \rightskip=\z@skip}
2224
2225 %

```

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

```

2226 \let\normalfootnoterule=\footnoterule
2227 %

```

XII.6 Footnotes arrangement

XII.6.1 User level macro

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

```

2228 \newcommandx{\Xarrangement}[2][1,usedefault]{%
2229   \def\do##1{%
2230     \csname Xarrangement@#2\endcsname{##1}%
2231   }%
2232   \ifstrempy{#1}%
2233     {%
2234       \dolistloop{@series}%
2235     }%
2236     {
2237       \docsvlist{#1}%
2238     }%
2239   }%
2240   %

```

XII.6.2 Normal footnote

`\Xarrangement@normal` We can now define all the parameters for the series of footnotes; initially they use the “normal” footnote formatting.

What we want to do here is to say something like the following for each footnote series. (This is an example, not part of the actual `reledmac` code.)

```

\skip\Afootins=12pt plus5pt minus5pt
\count\Afootins=1000
\dimen\Afootins=0.8\vsiz
\let\vAfootnote=\normalvfootnote \let\Afootfmt=\normalfootfmt
\let\Afootstart=\normalfootstart \let\Afootgroup=\normalfootgroup
\let\Afootnoterule=\normalfootnoterule

```

(Read *The TeXbook* in order to understand what are the counter, skip and dimen associated to an insertion.)

Instead of repeating ourselves, we define a `\Xarrangement@normal` macro that makes all these assignments for us, for any given series letter. This command is called when people use `\Xarrangement[s]{normal}`

Now we set up the `\Xarrangement@normal` macro itself. It takes one argument: the footnote series letter.

```

2241 \newcommand*{\Xarrangement@normal}[1]{%
2242   \csgdef{series@display#1}{normal}
2243   \expandafter\let\csname #1footstart\endcsname=\normalfootstart
2244   \expandafter\let\csname #1footnote\endcsname=\normalvfootnote
2245   \expandafter\let\csname #1footfmt\endcsname=\normalfootfmt

```

```

2246 \expandafter\let\csname #1footgroup\endcsname=\normalfootgroup
2247 \expandafter\let\csname #1footnoterule\endcsname=%
2248                                     \normalfootnoterule
2249 \count\csname #1footins\endcsname=1000
2250 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2251 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2252 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2253 %

```

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.

```

2254 \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
side only
2255 %

```

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).

```

2256 \ifnoledgroup@else%
2257 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2258 \expandafter\let\csname mp#1footgroup\endcsname=\mpnormalfootgroup
2259 \count\csname mp#1footins\endcsname=1000
2260 \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2261 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2262 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2263 \fi
2264 }
2265 %
2266 %

```

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

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

```

2267 \notbool{parapparatus@}\newcommand*{\newcommand}{\normalvfootnote}[2]{%
2268 \insert\csname #1footins\endcsname\bgroup
2269 \noindent\csuse{Xbhooknote@#1}%
2270 \csuse{Xnotefontsize@#1}%
2271 \footsplitskips
2272 \ifl@dpairing\ifl@dpaging\else%
2273 \setXnoteswidthliketwocolumns@{#1}%
2274 \fi\fi%
2275 \setXnotespositionliketwocolumns@{#1}%
2276 \spaceskip=\z@skip \xspaceskip=\z@skip
2277 \csname #1footfmt\endcsname #2{#1}\egroup}
2278 %

```

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

```

2279 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\mpnnormalvfootnote}[2]{%
2280   \global\setbox\@nameuse{mp#1footins}\vbox{%
2281     \unvbox\@nameuse{mp#1footins}
2282     \noindent\csuse{Xhooknote@#1}%
2283     \csuse{Xnotefontsize@#1}%
2284     \hsize\columnwidth
2285     \@parboxrestore
2286     \color@begingroup
2287     \csname #1footfmt\endcsname #2{#1}\color@endgroup}}
2288
2289 %

```

`\normalfootfmt` `\normalfootfmt` is a ‘normal’ macro to take the footnote line and page number information (see V.9 p. 81), and the desired text, and output what’s to be printed. Argument #1 contains the line and page number information and lemma font specifier; #2 is the lemma; #3 is the note’s text. This version is very rudimentary—it uses `\printlines` to print just the range of line numbers, followed by a square bracket, the lemma, and the note text.

```

2290
2291
2292 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmt}[4]{%
2293   \Xledsetnormalparstuff{#4}%
2294   \hangindent=\csuse{Xhangindent@#4}
2295   \strut{\printlinefootnote{#1}{#4}}%
2296   {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemfont#1|#2}{#2}}%
2297   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsemtty{
Xlemmaseparator@#4}%
2298     {\hskip\csuse{Xinplaceoflemmaseparator@#4}}}%
2299     {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}\relax%
2300   }}%
2301   #3\strut\par}
2302 %

```

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

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

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

The `\leftskip` and `\rightskip` values are both zeroed here. Similarly, these skips are cancelled in the `\vfootnote` macros for the various types of notes. Strictly speaking, this is necessary only if you are using paragraphed footnotes, but we have put it

here and in the other `\vfootnote` macros too so that the behavior of `reledmac` in this respect is general across all footnote types. What this means is that any `\leftskip` and `\rightskip` you specify applies to the main text, but not the footnotes. The footnotes continue to be of width `\hsize`.

```
2303 \newcommand*{\normalfootstart}[1]{%
2304 %
```

The first series of notes printed in a page can have a specific skip before it. In order to insert this specific skip without overlap the bottom margin of the page, Maïeul Rouquette have defined an algorithm explained in XVIII p. 184. Here is part of this algorithm, when the block of notes are ready to be printed.

```
2305 \ifdimequal{0pt}{\preXnotes@}{}%
2306 {%
2307   \iftoggle{preXnotes@}{%
2308     \togglefalse{preXnotes@}%
2309     \skip\csname #1footins\endcsname=%
2310     \dimexpr\csuse{preXnotes@}+\csuse{Xafterrule@#1}\relax%
2311   }%
2312   {%
2313   }%
2314   \vskip\skip\csname #1footins\endcsname%
2315 %
```

And now, the problem of left and right skip for notes. Especially when using one feature of `reledpar` which allows to have the footnotes horizontal size as the size of columns printed by `\Columns`. Read XV p. 182 for the general description of the problem.

```
2316 \leftskip0pt \rightskip0pt
2317 \ifl@dpairing\else%
2318   \hsize=\old@hsize%
2319 \fi%
2320 \setXnoteswidthliketwocolumns@{#1}%
2321 \setXnotespositionliketwocolumns@{#1}%
2322 %
```

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

```
2323 \print@Xfootnoterule{#1}%
2324 \noindent\leavevmode}
2325 %
```

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

```
2326 \newcommand*{\normalfootgroup}[1]{%
2327   {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}%
2328   \unvbox\csname #1footins\endcsname%
2329   \hsize=\old@hsize%
2330   }%
2331 %
2332 %
```

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

```

2333 \unless\ifnoledgroup@
2334 \newcommand*{\mpnormalfootgroup}[1]{%
2335   \vskip\skip\@nameuse{mp#1footins}
2336   \ifl@dpairing\ifparledgroup%
2337     \leavevmode\marks\parledgroup@{begin}%
2338     \marks\parledgroup@series{#1}%
2339     \marks\parledgroup@type{Xfootnote}%
2340   \fi\fi\normalcolor%
2341   \ifparledgroup%
2342     \ifl@dpairing%
2343     \else%
2344       \setXnoteswidthliketwocolumns@{#1}%
2345       \setXnotespositionliketwocolumns@{#1}%
2346       \print@Xfootnoterule{#1}%%
2347     \fi%
2348   \else%
2349     \setXnoteswidthliketwocolumns@{#1}%
2350     \setXnotespositionliketwocolumns@{#1}%
2351     \print@Xfootnoterule{#1}%%
2352   \fi%
2353   \setlength{\parindent}{0pt}
2354   {\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}
2355   \unvbox\csname mp#1footins\endcsname}}
2356 \fi
2357 %

```

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.

```

2358 \newcommand*{\Xarrangement@paragraph}[1]{%
2359   \csgdef{series@display#1}{paragraph}
2360   \expandafter\newcount\csname #1prevpage@num\endcsname
2361   \expandafter\let\csname #1footstart\endcsname=\parafootstart

```



```

2362 \expandafter\let\csname v#1footnote\endcsname=\paravfootnote
2363 \expandafter\let\csname #1footfmt\endcsname=\parafootfmt
2364 \expandafter\let\csname #1footgroup\endcsname=\parafootgroup
2365 \count\csname #1footins\endcsname=1000
2366 \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
side only
2367 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2368 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2369 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2370 \para@footsetup{#1}
2371 %

```

And the extra setup for minipages.

```

2372 \ifnoledgroup@else
2373 \expandafter\let\csname mpv#1footnote\endcsname=\mpparavfootnote
2374 \expandafter\let\csname mp#1footgroup\endcsname=\mpparafootgroup
2375 \count\csname mp#1footins\endcsname=1000
2376 \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2377 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2378 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2379 \fi
2380 }
2381 %

```

\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.

```

2382 \providecommand{\footfudgefiddle}{64}
2383 %

```

\para@footsetup `\footparagraph` calls the `\para@footsetup` macro to calculate a special fudge factor, which is the ratio of the `\baselineskip` to the `\hsize`. We assume that the proper value of `\baselineskip` for the footnotes (normally 9 pt) has been set already. The argument of the macro is again the note series letter.

Peter Wilson thinks that `\columnwidth` should be used here for \LaTeX not `\hsize`. Peter Wilson have also included `\footfudgefiddle`.

```

2384 \newcommand*{\para@footsetup}[1]{\csuse{Xnotefontsize@#1}
2385 \setXnoteswidthliketwocolumns@{#1}%
2386 \dimen0=\baselineskip
2387 \multiply\dimen0 by 1024
2388 \divide \dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\
relax
2389 \csxdef{#1footfudgefactor}{%
2390 \expandafter\strip@pt\dimen0 }}
2391
2392 %

```

`\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.

```

2393 \newcommand*{\parafootstart}[1]{%
2394   \rightskip=0pt \leftskip=0pt \parindent=0pt
2395   \ifdimequal{0pt}{\preXnotes@}{}%
2396   {%
2397     \iftoggle{preXnotes@}{%
2398       \togglefalse{preXnotes@}%
2399       \skip\csname #1footins\endcsname=%
2400       \dimexpr\csuse{preXnotes@}+\csuse{Xafterrule@#1}\relax%
2401     }%
2402   }%
2403 }%
2404 \vskip\skip\csname #1footins\endcsname%
2405 \setXnoteswidthliketwocolumns@{#1}%
2406 \setXnotespositionliketwocolumns@{#1}%
2407 \print@Xfootnoterule{#1}%
2408 \noindent\leavevmode}
2409 %

```

`\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.²⁷

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.²⁸ So changes from one language to another will not invoke the proper hyphenation rules in

²⁷Michael Downes, ‘Line Breaking in `\unhboxed` Text’, *TUGboat* **11** (1990), pp. 605–612.

²⁸See *The TeXbook*, p. 455 (editions after January 1990).

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.²⁹ Michael's unboxing macro is called `\Xunvxh`: `unvbox`, extract the last line, and `unhbox` it.

Doing things this way has an important consequence: as Michael pointed out, you really can't put an explicit line-break into a note built in a `\vbox` the way we are doing.³⁰ In other words, be very careful not to say `\break`, or `\penalty-10000`, or any equivalent inside your para-footnote. If you do, most of the note will probably disappear. You *are* allowed to make strong suggestions; in fact `\penalty-9999` will be quite okay. Just do not make the break mandatory. We have not applied any of Michael's solutions here, since we feel that the problem is exiguous, and `reledmac` is quite baroque enough already. If you think you are having this problem, look up Michael's solutions.

One more thing; we set `\leftskip` and `\rightskip` to zero. This has the effect of neutralizing any such skips which may apply to the main text (cf. XII.6.2 p. 142 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.

```

2410 \newcommand*{\paravfootnote}[2]{%
2411   \insert\csname #1footins\endcsname
2412   \bgroup
2413     \csuse{Xnotefontsize@#1}
2414     \footsplitskips
2415     \setbox0=\vbox{\hsize=\maxdimen
2416       \noindent\csuse{Xhooknote@#1}%
2417       \csname #1footfmt\endcsname #2{#1}}%
2418     \setbox0=\hbox{\Xunvxh{0}{#1}}%
2419     \dp0=0pt
2420     \ht0=\csname #1footfudgefactor\endcsname\wd0
2421   %

```

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

```

2422   \if@RTL\noindent \leavevmode\fi\box0%
2423   \penalty0

```

²⁹Wayne 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.

³⁰'Line Breaking', p. 610.

```

2424 \egroup}
2425
2426 %

```

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 T_EX attempts to split foot paragraphs. After trying out such a split (see *The TeXbook*, p. 124), T_EX 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.

```

2427 \newcommand*{\mpparavfootnote}[2]{%
2428   \global\setbox\@nameuse{mp#1footins}\vbox{%
2429     \unvbox\@nameuse{mp#1footins}%
2430     \csuse{Xnotefontsize@#1}
2431     \footsplitskip
2432     \setbox0=\vbox{\hsize=\maxdimen
2433       \noindent\color@begingroup%
2434       \csuse{Xbhooknote@#1}%
2435       \csname #1footfmt\endcsname #2{#1}\color@endgroup}%
2436     \setbox0=\hbox{\Xunvxh{0}{#1}}%
2437     \dp0=\z@
2438     \ht0=\csname #1footfudgefactor\endcsname\wd0
2439     \box0
2440     \penalty0
2441   }}
2442
2443 %

```

`\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`.

```

2444 \newcommand*{\Xunvxh}[2]{%
2445   \setbox0=\vbox{\unvbox#1%
2446     \global\setbox1=\lastbox}%
2447   \unhbox1
2448   \unskip           % remove \rightskip,
2449   \unskip           % remove \parfillskip,
2450   \unpenalty        % remove \penalty of 10000,
2451   \hskip\csuse{Xafternote@#2}} % but add the glue to go between the notes

```

2452
2453 %

`\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).

```
2454 \newcommand*{\parafootfmt}[4]{%
2455   \Xinsertparafootsep{#4}%
2456   \Xledsetnormalparstuff{#4}%
2457   \printlinefootnote{#1}{#4}%
2458   {\nottoggle{Xlemmadisablefontselection@#4}}{\select@lemmafnt#1|#2}{#2}}%
2459   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsemtyp{
Xlemmaseparator@#4}%
2460     {\hskip\csuse{Xinplaceoflemmaseparator@#4}}}%
2461     {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}%
2462   }}%
2463   #3\penalty-10 }
2464 %
```

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.

```
2465 \newcommand*{\parafootgroup}[1]{%
2466   \unvbox\csname #1footins\endcsname
2467   \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
2468   \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2469   \makehboxofhboxes
2470   \setbox0=\hbox{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\
unhbox0 \removehboxes}%
2471   \csuse{Xnotefontsize@#1}
2472   \noindent\unhbox0\par%
2473   \global\hsize=\old@hsize%
2474   }%
2475
2476 %
```

`\mpparafootgroup` The minipage version.

```

2477 \newcommand*\mpparafootgroup}[1]{%
2478   \setXnoteswidthliketwocolumns@{#1}%
2479   \vskip\skip\@nameuse{mp#1footins}
2480   \ifl@dpairing\ifparledgroup%
2481     \leavevmode\marks\parledgroup@{begin}%
2482     \marks\parledgroup@series{#1}%
2483     \marks\parledgroup@type{Xfootnote}%
2484   \fi\fi\normalcolor
2485   \ifparledgroup%
2486     \ifl@dpairing%
2487     \else%
2488       \setXnoteswidthliketwocolumns@{#1}%
2489       \setXnotespositionliketwocolumns@{#1}%
2490       \print@Xfootnoterule{#1}%
2491     \fi%
2492   \else%
2493     \setXnoteswidthliketwocolumns@{#1}%
2494     \setXnotespositionliketwocolumns@{#1}%
2495     \print@Xfootnoterule{#1}%
2496   \fi%
2497   \unvbox\csname mp#1footins\endcsname
2498   \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
2499   \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2500   \makehboxofhboxes
2501   \setbox0=\hbox{{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\unhbox0 \removehboxes}%
2502   \csuse{Xnotefontsize@#1}
2503   \noindent\unhbox0\par}}
2504
2505 %

```

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

```

\makehboxofhboxes\newcommand*\makehboxofhboxes{\setbox0=\hbox{}}%
\removehboxes\loop
2508   \unpenalty
2509   \setbox2=\lastbox
2510   \ifhbox2
2511     \setbox0=\hbox{\box2\unhbox0}%
2512   \repeat}
2513
2514 \newcommand*\removehboxes{\setbox0=\lastbox
2515   \ifhbox0{\removehboxes}\unhbox0 \fi}
2516
2517 %

```

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

```

\prevpage@num18 \newcommand{\Xinsertparafootsep}[1]{%
\Xinsertparafootsep19 \ifnumequal{\csuse{#1prevpage@num}}{\page@num}%
2520 {\ifcsdef{prevline#1}% Be sur \prevline#1 exists.
2521 {\ifnumequal{\csuse{prevline#1}}{\line@num}%
2522 {\ifcsempy{Xsymlinenum@#1}{\csuse{Xparafootsep@#1}}{}}%
2523 {\csuse{Xparafootsep@#1}}}%
2524 }%
2525 {\csuse{Xparafootsep@#1}}%
2526 }%
2527 {}%
2528 \global\csname #1prevpage@num\endcsname=\page@num%
2529 }
2530 %

```

XII.6.4 Columnar footnotes

Common tools

\rigidbalance We will now define macros for three-column notes and two-column notes. Both sets
\dosplits of macros will use \rigidbalance, which splits a box (#1) into into a number (#2) of
\splitoff columns, each with a space (#3) between the top baseline and the top of the \vbox. The
\@h \rigidbalance macro is taken from *The TeXbook*, p. 397, with a slight change to the
\@k 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.

```

2531 \newcount\@k \newdimen\@h
2532 \newcommand*\@rigidbalance}[3]{\setbox0=\box#1 \@k=#2 \@h=#3
2533 \@@line{\splittopskip=\@h \badness=\@M \hfilneg
2534 \valign{##\vfil\cr\dosplits}}}%
2535
2536 \newcommand*\@dosplits{\ifnum\@k>0 \noalign{\hfil}\splitoff
2537 \global\advance\@k-1\cr\dosplits\fi}%
2538
2539 \newcommand*\@splitoff{\dimen0=\ht0
2540 \divide\dimen0 by\@k \advance\dimen0 by\@h
2541 \setbox2 \vsplit0 to \dimen0
2542 \unvbox2 }
2543
2544 %

```

Three columns

```

\Xarrangement@threecol45 \newcommand*\Xarrangement@threecol}[1]{%
2546 \csgdef{series@display#1}{threecol}
2547 \expandafter\let\csname v#1footnote\endcsname=\threecolvfootnote

```

```

2548 \expandafter\let\csname #1footfmt\endcsname=\threecolfootfmt
2549 \expandafter\let\csname #1footgroup\endcsname=\threecolfootgroup
2550 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2551 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2552 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2553 \threecolfootsetup{#1}
2554 %

```

The additional setup for minipages.

```

2555 \ifnoledgroup@else
2556 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2557 \expandafter\let\csname mp#1footgroup\endcsname=\mpthreecolfootgroup
2558 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2559 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2560 \mpthreecolfootsetup{#1}
2561 \fi
2562 }
2563
2564 %

```

The `\footstart` and `\footnoterule` macros for these notes assume the normal values (XII.6.2 p. 142 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.

```

2565 \newcommand*{\threecolfootsetup}[1]{%
2566 \count\csname #1footins\endcsname 333
2567 \csxdef{default@#1footins}{333}%Use this to confine the notes to one
side only
2568 \multiply\dimen\csname #1footins\endcsname \thr@@}
2569 %

```

`\mpthreecolfootsetup` The setup for minipages.

```

2570 \newcommand*{\mpthreecolfootsetup}[1]{%
2571 \count\csname mp#1footins\endcsname 333
2572 \multiply\dimen\csname mp#1footins\endcsname \thr@@}

```


2573
2574 %

\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).

```
2575 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolvfootnote}[2]{%
2576   \insert\csname #1footins\endcsname\bgroup
2577   \csuse{Xnotefontsize@#1}
2578   \footplitskips
2579   \csname #1footfmt\endcsname #2{#1}\egroup}
2580 %
```

\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.

```
2581 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmt}[4]{%
2582   \normal@pars
2583   \hsize \csuse{Xhsizethreecol@#4}
2584   \nottoggle{Xparindent@#4}{\parindent=\z@}{%
2585     \tolerance=5000
2586     \hangindent=\csuse{Xhangindent@#4}
2587     \leavevmode
2588     \csuse{Xcolalign@#4}%
2589     \strut{\printlinefootnote{#1}{#4}}%
2590     {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemmafnt#1|#2}{#2}}%
2591     \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcempty{
Xlemmaseparator@#4}%
2592       {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2593       {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}%
2594     }}%
2595     #3\strut\par\allowbreak}
2596 %
```

\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.

```

2597 \newcommand*{\threecolfootgroup}[1]{\csuse{Xnotefontsize@#1}%
2598   \noindent\csuse{Xtxtbeforenotes@#1}\par%
2599   \splittopskip=\ht\strutbox
2600   \expandafter
2601   \rigidbalance\csname #1footins\endcsname \thr@@ \splittopskip}}
2602 %

```

`\mpthreecolfootgroup` The setup for minipages.

```

2603 \newcommand*{\mpthreecolfootgroup}[1]{%
2604   \vskip\skip\@nameuse{mp#1footins}
2605   \ifl@dpairing\ifparledgroup%
2606     \leavevmode\marks\parledgroup@{begin}%
2607     \marks\parledgroup@series{#1}%
2608     \marks\parledgroup@type{Xfootnote}%
2609     \fi\fi\normalcolor
2610     \ifparledgroup%
2611       \ifl@dpairing%
2612         \else%
2613           \setXnoteswidthliketwocolumns@{#1}%
2614           \setXnotespositionliketwocolumns@{#1}%
2615           \print@Xfootnoterule{#1}%
2616         \fi%
2617       \else%
2618         \setXnoteswidthliketwocolumns@{#1}%
2619         \setXnotespositionliketwocolumns@{#1}%
2620         \print@Xfootnoterule{#1}%
2621       \fi%
2622     {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}\par
2623     \splittopskip=\ht\strutbox
2624     \expandafter
2625     \rigidbalance\csname mp#1footins\endcsname \thr@@ \splittopskip}}
2626 %
2627 %

```

Two columns

```

\Xarrangement@twocol28 \newcommand*{\Xarrangement@twocol}[1]{%
2629   \csgdef{series@display#1}{twocol}
2630   \expandafter\let\csname v#1footnote\endcsname=\twocolvfootnote
2631   \expandafter\let\csname #1footfmt\endcsname=\twocolfootfmt
2632   \expandafter\let\csname #1footgroup\endcsname=\twocolfootgroup
2633   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2634   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%

```

```

2635 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2636 \twocolfootsetup{#1}
2637 %

```

The additional setup for minipages.

```

2638 \ifnoledgroup@else
2639 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2640 \expandafter\let\csname mp#1footgroup\endcsname=\mptwocolfootgroup
2641 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2642 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2643 \mptwocolfootsetup{#1}
2644 \fi
2645 }
2646
2647 %

```

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

```

\twocolfootnote
\twocolfootfmt
\twocolfootgroup
2648 \newcommand*{\twocolfootsetup}[1]{%
2649 \count\csname #1footins\endcsname 500
2650 \csxdef{default@#1footins}{500}%Use this to confine the notes to one
side only
2651 \multiply\dimen\csname #1footins\endcsname \tw@}
2652 %

2653 \notbool{parapparatus@}{\newcommand*{\twocolvfootnote}[2]{\
insert\csname #1footins\endcsname\bgroup
2654 \csuse{Xnotefontsize@#1}
2655 \footssplitsskip
2656 \csname #1footfmt\endcsname #2{#1}\egroup}
2657 %

2658 \notbool{parapparatus@}{\newcommand*{\twocolfootfmt}[4]{% 4th
arg is optional, for backward compatibility
2659 \normal@pars
2660 \hsiz \csuse{Xhsizetwocol@#4}
2661 \nottoggle{Xparindent@#4}{\parindent=\z@}{\
2662 \tolerance=5000
2663 \hangindent=\csuse{Xhangindent@#4}
2664 \leavevmode
2665 \csuse{Xcolalign@#4}%
2666 \strut{\printlinefootnote{#1}{#4}}%
2667 {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemmafnt#1|#2}{#2}}%
2668 \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsempy{
Xlemmaseparator@#4}%
2669 {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2670 {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}}%

```

```

2671 }}%
2672 #3\strut\par\allowbreak}
2673 %

2674 \newcommand*{\twocolfootgroup}[1]{\csuse{Xnotefontsize@#1}
2675 \noindent\csuse{Xtxtbeforenotes@#1}\par%
2676 \splittopskip=\ht\strutbox
2677 \expandafter
2678 \rigidbalance\csname #1footins\endcsname \tw@ \splittopskip}}
2679
2680 %

```

`\mptwocolfootsetup` The versions for minipages.

`\mptwocolfootgroup`

```

2681 \newcommand*{\mptwocolfootsetup}[1]{%
2682 \count\csname mp#1footins\endcsname 500
2683 \multiply\dimen\csname mp#1footins\endcsname \tw@}
2684 %

2685 \newcommand*{\mptwocolfootgroup}[1]{%
2686 \vskip\skip\@nameuse{mp#1footins}
2687 \ifl@dpairing\ifparledgroup%
2688 \leavevmode\marks\parledgroup@{begin}%
2689 \marks\parledgroup@series{#1}%
2690 \marks\parledgroup@type{Xfootnote}%
2691 \fi\fi\normalcolor
2692 \ifparledgroup%
2693 \ifl@dpairing%
2694 \else%
2695 \setXnoteswidthliketwocolumns@{#1}%
2696 \setXnotespositionliketwocolumns@{#1}%
2697 \print@Xfootnoterule{#1}%
2698 \fi%
2699 \else%
2700 \setXnoteswidthliketwocolumns@{#1}%
2701 \setXnotespositionliketwocolumns@{#1}%
2702 \print@Xfootnoterule{#1}%
2703 \fi%
2704 {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}\par
2705 \splittopskip=\ht\strutbox
2706 \expandafter
2707 \rigidbalance\csname mp#1footins\endcsname \tw@ \splittopskip}}
2708
2709 %

```

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.

```

2710 \def\endashchar{\textnormal{--}}
2711 \newcommand*\fullstop{\textnormal{.}}
2712 \newcommand*\rbracket{\textnormal{%
2713   \csuse{text\csuse{footnote@lang}}{%
2714     \ifluatex%
2715       \ifdefstring{\footnote@luatextextdir}{TRT}{\thinspace[]}{\thinspace
2716     }%
2717     \else%
2718       \thinspace}%
2719     \fi}%
2720 }}%
2721 %
2722 %

```

XII.7.2 Pstart number in footnote

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

```

2723 \newcommand{\printpstart}[0]{%
2724   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
2725     l@dprintingcolumns}}{%
2726     \ifledRcol%
2727       \thepstartR%
2728     \else%
2729       \thepstartL%
2730     \fi%
2731   }{%
2732     \thepstart%
2733   }%
2734 %

```

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`.

```

2735 \newcommand{\printlinefootnote}[2]{%
2736   \def\extractline@##1|##2|##3|##4|##5|##6|##7|{##2}%
2737   \def\extractsubline@##1|##2|##3|##4|##5|##6|##7|{##3}%
2738   \def\extractendline@##1|##2|##3|##4|##5|##6|##7|{##5}%
2739   \def\extractendsubline@##1|##2|##3|##4|##5|##6|##7|{##6}%
2740   \iftoggle{Xnumberonlyfirstintwolines@#2}{%
2741     \edef\lineinfo@{\extractline@ #1| - \extractsubline@ #1| - \
extractendline@ #1| - \extractendsubline@ #1|}%
2742   }%
2743   {%
2744     \edef\lineinfo@{\extractline@ #1| - \extractsubline@ #1|}%
2745   }%
2746   \iftoggle{nonum@}{%Try if the line number must printed for this specific
not (by default, yes)
2747     \hspace{\csuse{Xinplaceofnumber@#2}}%
2748   }%
2749   {%
2750     {%
2751       \iftoggle{Xnonumber@#2}{%Try if the line number must printed (by
default, yes)
2752       {%
2753         \hspace{\csuse{Xinplaceofnumber@#2}}%
2754       }%
2755       {%
2756         {\iftoggle{Xnumberonlyfirstinline@#2}% If for this series the
line number must be printed only in the first time.
2757         {%
2758           \ifcsdef{prevline#2}%
2759           {%Be sure the \prevline exists.
2760           \ifcsequal{prevline#2}{\lineinfo@}{%Try it
2761           {%
2762             \ifcsequal{Xsymlinenum@#2}%
2763             {%
2764               \hspace{\csuse{Xinplaceofnumber@#2}}%
2765             }%
2766             {\hspace{\csuse{Xbeforeasymlinenum@#2}}\csuse{
Xnotenumfont@#2}%
2767             \ifdimequal{\csuse{Xboxsymlinenum@#2}}{0pt}%
2768             {\csuse{Xsymlinenum@#2}}%
2769             {\hbox to \csuse{Xboxsymlinenum@#2}{\csuse{
Xsymlinenum@#2}\hfill}}%
2770             \hspace{\csuse{Xaftersymlinenum@#2}}}%
2771           }%

```

```

2772         {%
2773         \printlinefootnotearea{#1}{#2}%
2774         }%
2775     }%
2776     {%
2777     \printlinefootnotearea{#1}{#2}%
2778     }%
2779 }%
2780 {%
2781 \printlinefootnotearea{#1}{#2}%
2782 }%
2783 \csxdef{prevline#2}{\lineinfo@}%
2784 }%
2785 }%
2786 }%
2787 }%
2788 }
2789 %

```

\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.)

```

2790 \newcommand{\printlinefootnotearea}[2]{%
2791 \printXbeforenumber{#2}%
2792 \csuse{Xnotenumfont@#2}%
2793 \boxfootnotenumbers{#1}{#2}%
2794 \printXafternumber{#2}%
2795 }%
2796 %

```

\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.

```

2797 \newcommand{\boxfootnotenumbers}[2]{%
2798 \ifdimequal{\csuse{Xboxlinenum@#2}}{0pt}{%
2799 \printlinefootnotenumbers{#1}{#2}%
2800 }%
2801 {%
2802 \hbox to \csuse{Xboxlinenum@#2}%
2803 {%
2804 \IfSubStr{RC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2805 \printlinefootnotenumbers{#1}{#2}%
2806 \IfSubStr{LC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2807 }%
2808 }%
2809 }%
2810 %

```

\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.

```

2811 \newcommand{\printlinefootnotenumbers}[2]{%
2812   \xdef\@currentseries{#2}%
2813   \ifboolexpr{%
2814     (togl{Xpstart@#2} and bool{numberpstart})%
2815     or togl{Xpstarteverytime@#2}}}%
2816   {\printpstart}{}%
2817   \iftoggle{Xstanza@#2}{%
2818     \ifnumberstanza%
2819       \printstanza%
2820       \csuse{Xstanzaseparator@#2}%
2821     \fi%
2822   }{ }%
2823   \iftoggle{Xonlypstart@#2}{ }{\printlines#1|}%
2824 }%
2825 %

```

\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.)

```

2826 \newcommand{\printXbeforenumber}[1]{%
2827   \hspace{\csuse{Xbeforenumber@#1}}%
2828 }%
2829 %

```

\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

```

2830 \newcommand{\printXafternumber}[1]{%
2831   \iftoggle{Xnonbreakableafternumber@#1}{\nobreak}{ }%
2832   \hspace{\csuse{Xafternumber@#1}}%
2833 }%
2834 %

```

If we have decided to print the line number in a specific notes, the `\printlines` macro prints the line numbers for a note—which, in the general case, is a rather complicated task. The seven parameters of the argument are the line numbers as stored in `\l@d@nums`, in the form described on V.9 p. 81: the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma.

edmac' creator have defined six boolean in order to know which component of line number description we have to print:

- `\ifl@d@pnum` for page numbers;
- `\ifl@d@ssub` for starting sub-line;

- `\ifl@d@elin` for ending line;
- `\ifl@d@esl` for ending sub-line; and
- `\ifl@d@dash` for the dash between the starting and ending groups.

There is no boolean for the line number because it is always printed.

Maïeul Rouquette has added `\ifl@d@Xtwolines` and `\ifl@d@Xmorethantwolines` to print a symbol which stands for “and subsequent” when there are two, three or more lines.

```

\ifl@d@pnum 35 \newif\ifl@d@pnum
\ifl@d@ssub 36 \newif\ifl@d@ssub
\ifl@d@elin 37 \newif\ifl@d@elin
\ifl@d@esl 38 \newif\ifl@d@esl
\ifl@d@dash 39 \newif\ifl@d@dash
\ifl@d@Xtwolines 40 \newif\ifl@d@Xtwolines%
\ifl@d@Xmorethantwolines 41 \newif\ifl@d@Xmorethantwolines%
2842 %

\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@n@ums 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
2843 \newcommand*{\l@d@p@rsefootspec}[3]{\l@d@p@rsefootspec#1|}
2844 \def\l@d@p@rsefootspec#1|#2|#3|#4|#5|#6|#7|{%
2845 \gdef\l@d@p@rsefootspec#1}%
2846 \gdef\l@d@p@rsefootspec#2}%
2847 \gdef\l@d@p@rsefootspec#3}%
2848 \gdef\l@d@p@rsefootspec#4}%
2849 \gdef\l@d@p@rsefootspec#5}%
2850 \gdef\l@d@p@rsefootspec#6}%
2851 }
2852 %

```

Initialise the several number value macros.

```

2853 \def\l@d@p@rsefootspec{0}%
2854 \def\l@d@p@rsefootspec{0}%
2855 \def\l@d@p@rsefootspec{0}%
2856 \def\l@d@p@rsefootspec{0}%
2857 \def\l@d@p@rsefootspec{0}%
2858 \def\l@d@p@rsefootspec{0}%
2859 %
2860 %

```

\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`.

```
2861 \newcommand*\setprintlines}[6]{%
2862   \l@dpnumfalse \l@ddashfalse
2863   %
```

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

```
2864   \ifbypage@
2865     \ifnum#4=#1 \else
2866       \l@dpnumtrue
2867       \l@ddashtrue
2868     \fi
2869   \fi
2870   %
```

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.

```
2871   \ifl@dpnum \l@d@elintrue \else \l@d@elinfalse \fi
2872   \ifnum#2=#5 \else
2873     \l@d@elintrue
2874     \l@ddashtrue
2875   \fi
2876   %
```

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

```
2877   \l@d@ssubfalse
2878   \ifnum#3=0 \else
2879     \l@d@ssubtrue
2880   \fi
2881   %
```

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.

```
2882   \l@d@eslfalse
2883   \ifnum#6=0 \else
2884     \ifnum#6=#3
2885       \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
2886     \else
2887       \l@d@esltrue
2888       \l@ddashtrue
2889     \fi
2890   \fi%
2891   %
```

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

```

2892 \ifl@dash%
2893 \ifbool{expr}{togl{fulllines@} or test{\ifcsemt{Xtwolines@
@currentseries}}}%
2894 {}%
2895 {%
2896 \setistwofollowinglines{#1}{#2}{#4}{#5}%
2897 \ifbool{expr}%
2898 (%
2899     togl {Xtwolinesbutnotmore@ \@currentseries}%
2900     and not%
2901     (%
2902         bool {istwofollowinglines@}%
2903     )%
2904 )%
2905 or%
2906 (%
2907     (not test{\ifnumequal{#1}{#4}})%
2908     and togl{Xtwolinesonlyinsamepage@ \@currentseries}%
2909 )%
2910 }%
2911 {}%
2912 {%
2913 \l@dashfalse%
2914 \l@Xtwolinestrue%
2915 \l@delinfalse%
2916 \l@delselfalse%
2917 \ifcsemt{Xmorethantwolines@ \@currentseries}%
2918 {%
2919     {\ifistwofollowinglines@ \else%
2920         \l@Xmorethantwolinestrue%
2921     }%
2922 }%
2923 }%
2924 }%
2925 \fi%
2926 %

```

End of \setprintlines.

```

2927 }%
2928 %

```

\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.

```

2929 \newif\ifistwofollowinglines%
2930 \newcommand{\setistwofollowinglines}[4]{%
2931   \ifcsdef{lastlinenumberon@#1}%
2932     {\numdef{\tmp}{\csuse{lastlinenumberon@#1}}}%
2933     {\numdef{\tmp}{0}}%
2934   \istwofollowinglines@false%
2935   \ifnumequal{#4-#2}{1}%
2936     {\istwofollowinglines@true}%
2937     {\ifbypage%
2938       \ifnumequal{#3-#1}{1}%
2939       {%
2940         \ifnumequal{#2}{\tmp}%
2941         {\ifnumequal{#4}{1}{\istwofollowinglines@true}}}%
2942       {}%
2943     }%
2944   }%
2945   \fi%
2946 }%
2947 }%
2948 %

```

`\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.

```

2949 \def\printlines#1|#2|#3|#4|#5|#6|#7|{%
2950   \begingroup%
2951 %

```

If we use Lua_T_EX, ensure we use good text's direction.

```

2952 \ifluatex%
2953   \luatextextdir TLT%
2954 \fi%
2955 %

```

Decide which part of line number components we will print.

```

2956 \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
2957 %

```

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.

```

2958 \ifdimequal{\csuse{Xboxstartlinenum@\@currentseries}}{0pt}%
2959   {\bgroup}%
2960   {\leavevmode\hbox to \csuse{Xboxstartlinenum@\@currentseries}\bgroup\
hfill}%
2961   \ifl@d@pnum #1\fullstop\fi
2962   \linenumrep{#2}
2963   \ifl@d@ssub \fullstop \sublinenumrep{#3}\fi
2964   \egroup%
2965 %

```

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

```

2966 \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
2967   {\bgroup}%
2968   {\hbox to \csuse{Xboxendlinenum@\@currentseries}\bgroup}%
2969   \ifl@d@Xtwolines%
2970     \ifl@d@Xmorethantwolines%
2971       \csuse{Xmorethantwolines@\@currentseries}%
2972     \else%
2973       \csuse{Xtwolines@\@currentseries}%
2974     \fi%
2975   \else%
2976     \ifl@d@dash \endashchar\fi%
2977     \ifl@d@pnum #4\fullstop\fi%
2978     \ifl@d@elin \linenumrep{#5}\fi%
2979     \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumrep{#6}\fi%
2980   \fi%
2981   \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
2982   {}%
2983   {\hfill}%Prevent underfull hbox
2984   \egroup%
2985   \endgroup%
2986 }%
2987 %

```

XIII Familiar footnotes

XIII.1 Adjacent footnotes

The original edmac provided users with five series of critical footnotes (`\Afootnote` `\Bfootnote` `\Cfootnote` `\Dfootnote` `\Efootnote`), and \LaTeX provides a single numbered footnote. The `reledmac` package uses the edmac mechanism to provide six series of numbered footnotes.

First, though, the `footmisc` package has an option whereby two or more consecutive `\footnotes` have their marks separated by commas. This seemed to Peter Wilson such a useful ability that it was provided automatically by `eledmac`.

Maïeul Rouquette has maintained this feature in `reledmac`, despite he thought that is not directly in relationship with the aim of `reledmac`.

`\multiplefootnotemarker` These macros may have been defined by the `memoir` class, are provided by the `footmisc` package and perhaps by other footnote packages. That is why we use `\providecommand` and not `\newcommand`.

`\multfootsep`

```
2988 \providecommand*\multiplefootnotemarker}{3sp}
2989 \providecommand*\multfootsep{\textsuperscript{\normalfont,}}
2990
2991 %
```

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

```
2992 \providecommand*\m@mmf@prepare}{%
2993   \kern-\multiplefootnotemarker
2994   \kern\multiplefootnotemarker\relax}
2995 %
```

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

```
2996 \providecommand*\m@mmf@check}{%
2997   \ifdim\lastkern=\multiplefootnotemarker\relax
2998     \edef\@x@sf{\the\spacefactor}%
2999     \unkern
3000     \multfootsep
3001     \spacefactor\@x@sf\relax
3002   \fi}
3003
3004 %
```

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

```
3005 \@ifclassloaded{memoir}{}{%
3006 %
```

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

```
3007 \apptocmd{\@footnotetext}{\m@mmf@prepare}{}{}
3008 %
```

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

```

3009
3010 \patchcmd{\@footnotemark}
3011   {\nobreak}
3012   {\m@mmf@check
3013   \nobreak
3014   }
3015   {}{}
3016 \patchcmd{\@footnotemark}
3017   {\@makefnmark}
3018   {\@makefnmark
3019   \m@mmf@prepare
3020   }
3021   {}{}
3022 %

```

Finished the modifications for the non-memoir case.

```

3023 }
3024
3025 %

```

XIII.2 Regular footnotes for numbered texts

`\l@doldold@footnotetext` In order to enable the regular `\footnotes` in numbered text we have to play around with its `\@footnotetext`, using different forms for when in numbered or regular text.

```

3026 \pretocmd{\@footnotetext}{%
3027   \ifnumberedpar@
3028   \edtext{}{\l@dbfnote{#1}}}%
3029   \else
3030   }{}{}
3031 \apptocmd{\@footnotetext}{\fi}{}{}%
3032 %

```

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

```

3033
3034 \newcommand{\l@dbfnote}[1]{%
3035   \ifnumberedpar@
3036   \gdef\@tag{#1\relax}%
3037   \ifledRcol%
3038     \xright@appenditem{\noexpand\vl@dbfnote{\expandonce\@tag}}{\@theftnmark}}}%
3039     \to\inserts@listR
3040     \global\advance\insert@countR \@ne%
3041   \else%
3042     \xright@appenditem{\noexpand\vl@dbfnote{\expandonce\@tag}}{\@theftnmark}}}%

```

```

3043 \to\inserts@list
3044 \global\advance\insert@count \@ne%
3045 \fi
3046 \fi\ignorespaces}
3047
3048 \newcommand{\vl@dbfnote}[2]{%
3049 \def\@thefnmark{#2}%
3050 \@footnotetext{#1}%
3051 }%
3052 %

```

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
3053 \newcommand*{\prebodyfootmark}{%
3054 \leavevmode
3055 \ifhmode
3056 \edef\@x@sf{\the\spacefactor}%
3057 \m@mmf@check
3058 \nobreak
3059 \fi}
3060 \newcommand{\postbodyfootmark}{%
3061 \m@mmf@prepare
3062 \ifhmode\spacefactor\@x@sf\fi\relax}
3063
3064 %

```

XIII.4 Footnote arrangement

XIII.4.1 User level macro

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

```

3065 \newcommandx{\arrangementX}[2][1,usedefault]{%
3066 \def\do##1{%
3067 \csname arrangementX@#2\endcsname{##1}%
3068 }%
3069 \ifstrempy{#1}%
3070 {%
3071 \dolistloop{\@series}%
3072 }%
3073 {
3074 \docsvlist{#1}%
3075 }%

```



```
3076 }%
3077 %
```

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.

```
3078 \newcommand*{\normal@footnotemarkX}[1]{%
3079   \prebodyfootmark
3080   \@nameuse{bodyfootmark#1}%
3081   \postbodyfootmark}
3082
3083 %
```

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

```
3084 \newcommand*{\normalbodyfootmarkX}[1]{%
3085   \hbox{\textsuperscript{\normalfont\@nameuse{@thefnmark#1}}}}
3086 %
```

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

```
3087 \notbool{parapparatus@}{\newcommand*{\newcommand}{\normalvfootnoteX}[2]{%
3088   \insert\@nameuse{footins#1}\bgroup
3089   \noindent\csuse{bhooknoteX@#1}%
3090   \csuse{notefontsizeX@#1}%
3091   \footsplitskips
3092   \ifl@dpairing\ifl@dpaging\else%
3093     \setnoteswidthliketwocolumnsX@{#1}%
3094   \fi\fi%
3095   \setnotesXpositionliketwocolumns@{#1}%
3096   \spaceskip=\z@skip \xspaceskip=\z@skip
3097   \csuse{\csuse{footnote@dir}}\@nameuse{footfmt#1}{#1}{#2}\egroup}
3098
3099 %
```

`\mpnormalvfootnoteX` The minipage version.

```
3100 \newcommand*{\mpnormalvfootnoteX}[2]{%
3101   \global\setbox\@nameuse{mpfootins#1}\vbox{%
3102     \unvbox\@nameuse{mpfootins#1}
3103     \noindent\csuse{bhooknoteX@#1}%
3104     \csuse{notefontsizeX@#1}%
3105     \hsize\columnwidth
3106     \@parboxrestore
3107     \color@begingroup
3108     \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}}
```

3109

3110 %

`\normalfootfmtX` `\normalfootfmtX{⟨series⟩}{⟨text⟩}` typesets the footnote text, prepended by the marker.

```

3111 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmtX}[2]{%
3112   \ifluatex%
3113     \luatextextdir\footnote@luatextextdir%
3114     \luatexpardir\footnote@luatexpardir%
3115     \par%
3116   \fi%
3117   \protected@edef\@currentlabel{%
3118     \@nameuse{@thefnmark#1}%
3119   }%
3120   \ledsetnormalparstuffX{#1}%
3121   \hangindent=\csuse{hangindentX@#1}%
3122   {{\csuse{notenunfontX@#1}\@nameuse{footfootmark#1}}\strut%
3123     #2\strut\par}}
3124
3125 %

```

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

```

3126 \newcommand*{\normalfootfootmarkX}[1]{%
3127   \textsuperscript{\@nameuse{@thefnmark#1}}}
3128
3129 %

```

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

```

3130 \newcommand*{\normalfootstartX}[1]{%
3131   \ifdimequal{Opt}{\prenotesX@}{}%
3132   {%
3133     \iftoggle{prenotesX@}{%
3134       \togglefalse{prenotesX@}%
3135       \skip\csname footins#1\endcsname=%
3136       \dimexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
3137     }%
3138     {}%
3139   }%
3140   \vskip\skip\csname footins#1\endcsname%
3141   \leftskip=\z@
3142   \rightskip=\z@
3143   \ifl@dpairing\else%
3144     \hsize=\old@hsize%
3145   \fi%
3146   \setnoteswidthliketwocolumnsX@{#1}%
3147   \setnotesXpositionliketwocolumns@{#1}%

```

```

3148 \print@footnoteXrule{#1}%
3149 }%
3150
3151 %

```

\normalfootnoteruleX The rule drawn before the footnote series group.

```

3152 \let\normalfootnoteruleX=\footnoterule
3153
3154 %

```

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

```

3155 \newcommand*\normalfootgroupX[1]{%
3156 \unvbox\@nameuse{footins#1}%
3157 \hsize=\old@hsize%
3158 }%
3159
3160 %

```

\mpnormalfootgroupX The minipage version.

```

3161 \newcommand*\mpnormalfootgroupX[1]{%
3162 \vskip\skip\@nameuse{mpfootins#1}
3163 \ifl@dpairing\ifparledgroup%
3164 \leavevmode\marks\parledgroup@{begin}%
3165 \marks\parledgroup@series{#1}%
3166 \marks\parledgroup@type{footnoteX}%
3167 \fi\fi\normalcolor
3168 \ifparledgroup%
3169 \ifl@dpairing%
3170 \else%
3171 \setnoteswidthliketwocolumnsX@{#1}%
3172 \setnotesXpositionliketwocolumns@{#1}%
3173 \print@footnoteXrule{#1}%
3174 \fi%
3175 \else%
3176 \setnoteswidthliketwocolumnsX@{#1}%
3177 \setnotesXpositionliketwocolumns@{#1}%
3178 \print@footnoteXrule{#1}%
3179 \fi%
3180 \unvbox\@nameuse{mpfootins#1}}
3181
3182 %

```

\normalbfnoteX₈₃

```

3184 \newcommand{\normalbfnoteX}[2]{%
3185 \ifnumberedpar@

```

```

3186 \ifledRcol%
3187 \ifluatex
3188 \footnotelang@lua[R]%
3189 \fi
3190 \ifundefined{xpg@main@language}%if polyglossia
3191 {}%
3192 {\footnotelang@poly[R]}%
3193 \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
3194 \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\expandonce\
thisfootnote}}%
3195 \to\inserts@listR
3196 \global\advance\insert@countR \@ne%
3197 \else%
3198 \ifluatex
3199 \footnotelang@lua%
3200 \fi
3201 \ifundefined{xpg@main@language}%if polyglossia
3202 {}%
3203 {\footnotelang@poly}%
3204 \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
3205 \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\expandonce\
thisfootnote}}%
3206 \to\inserts@list
3207 \global\advance\insert@count \@ne%
3208 \fi
3209 \fi\ignorespaces}
3210
3211 %

```

```

\vbfnoteX12 \newcommand{\vbfnoteX}[3]{%
3213 \@namedef{@thefnmark#1}{#3}%
3214 \@nameuse{regvfootnote#1}{#1}{#2}}
3215
3216 %

```

```

\vnumfootnoteX17 \newcommand{\vnumfootnoteX}[2]{%
3218 \ifnumberedpar@
3219 \edtext{}{\normalbfnoteX{#1}{#2}}%
3220 \else
3221 \@nameuse{regvfootnote#1}{#1}{#2}%
3222 \fi}
3223
3224 %

```

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

```

3225 \newcommand*\arrangementX@normal}[1]{%
3226   \csgdef{series@displayX#1}{normal}
3227   \expandafter\let\csname footstart#1\endcsname=\normalfootstartX
3228   \expandafter\newcount\csname prevpage#1@num\endcsname
3229   \@namedef{footnotemark#1}{\normal@footnotemarkX{#1}}
3230   \@namedef{bodyfootmark#1}{\normalbodyfootmarkX{#1}}
3231   \expandafter\let\csname regvfootnote#1\endcsname=\normalvfootnoteX
3232   \expandafter\let\csname vfootnote#1\endcsname=\vnumfootnoteX
3233   \expandafter\let\csname footfmt#1\endcsname=\normalfootfmtX
3234   \@namedef{footfootmark#1}{\normalfootfootmarkX{#1}}
3235   \expandafter\let\csname footgroup#1\endcsname=\normalfootgroupX
3236   \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3237   \count\csname footins#1\endcsname=1000
3238   \csxdef{default@footins#1}{1000}%Use to have note only for one side
3239   \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3240   \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3241   \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
3242   %

```

Additions for minipages.

```

3243   \ifnoleddgroup@else%
3244     \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3245     \expandafter\let\csname mpfootgroup#1\endcsname=\mpnormalfootgroupX
3246     \count\csname mpfootins#1\endcsname=1000
3247     \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
3248     \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3249     \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
3250   \fi
3251 }
3252
3253 %

```

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@twocol54 \newcommand*\arrangementX@twocol}[1]{%
3255   \csgdef{series@displayX#1}{twocol}
3256   \expandafter\let\csname regvfootnote#1\endcsname=\twocolvfootnoteX
3257   \expandafter\let\csname footfmt#1\endcsname=\twocolfootfmtX
3258   \expandafter\let\csname footgroup#1\endcsname=\twocolfootgroupX
3259   \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3260   \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3261   \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3262   \twocolfootsetupX{#1}
3263   \ifnoleddgroup@else%
3264     \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3265     \expandafter\let\csname mpfootgroup#1\endcsname=\mptwocolfootgroupX

```

```

3266 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3267 \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}
3268 \mptwocolfootsetupX{#1}
3269 \fi%
3270 }
3271
3272 %

```

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

```

3273 \newcommand*{\twocolfootsetupX}[1]{%
3274 \count\csname footins#1\endcsname 500
3275 \csxdef{default@footins#1}{500}%Use this to confine the notes to one
side only
3276 \multiply\dimen\csname footins#1\endcsname by \tw@}
3277 \newcommand*{\mptwocolfootsetupX}[1]{%
3278 \count\csname mpfootins#1\endcsname 500
3279 \multiply\dimen\csname mpfootins#1\endcsname by \tw@}
3280
3281 %

```

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

```

3282 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolvfootnoteX}[2]{%
3283 \insert\csname footins#1\endcsname\bgroup
3284 \csuse{notefontsizeX@#1}
3285 \footsplitskips
3286 \spaceskip=\z@skip \xspaceskip=\z@skip
3287 \@nameuse{footfmt#1}{#1}{#2}\egroup}
3288
3289 %

```

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

```

3290 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolfootfmtX}[2]{%
3291 \protected@edef\@currentlabel{%
3292 \@nameuse{thefnmark#1}%
3293 }%
3294 \normal@pars
3295 \hangindent=\csuse{hangindentX@#1}%
3296 \hsize \csuse{hsizetwocolX@#1}
3297 \nottoggle{parindentX@#1}{\parindent=\z@}{}
3298 \tolerance=5000\relax
3299 \leavevmode
3300 \csuse{colalignX@#1}%
3301 {\csuse{notenumfontX@#1}\@nameuse{footfootmark#1}\strut%
3302 #2\strut\par}\allowbreak}
3303
3304 %

```

```

\twocolfootgroupX \twocolfootgroupX{<series>}
\mptwocolfootgroupX
3305 \newcommand*{\twocolfootgroupX}[1]{\csuse{notefontsizeX@#1}
3306 \splittopskip=\ht\strutbox
3307 \expandafter
3308 \rigidbalance\csname footins#1\endcsname \tw@ \splittopskip}}
3309 \newcommand*{\mptwocolfootgroupX}[1]{\%
3310 \vskip\skip\@nameuse{mpfootins#1}
3311 \ifl@dpairing\ifparledgroup%
3312 \leavevmode\marks\parledgroup@{begin}%
3313 \marks\parledgroup@series{#1}%
3314 \marks\parledgroup@type{footnoteX}%
3315 \fi\fi\normalcolor
3316 \ifparledgroup%
3317 \ifl@dpairing%
3318 \else%
3319 \setnoteswidthliketwocolumnsX@{#1}%
3320 \setnotesXpositionliketwocolumns@{#1}%
3321 \print@footnoteXrule{#1}%
3322 \fi%
3323 \else%
3324 \setnoteswidthliketwocolumnsX@{#1}%
3325 \setnotesXpositionliketwocolumns@{#1}%
3326 \print@footnoteXrule{#1}%
3327 \fi%
3328 \splittopskip=\ht\strutbox
3329 \expandafter
3330 \rigidbalance\csname mpfootins#1\endcsname \tw@ \splittopskip}}
3331
3332 %

```

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
3333 \newcommand*{\arrangementX@threecol}[1]{\%
3334 \csgdef{series@displayX#1}{threecol}
3335 \expandafter\let\csname regvfootnote#1\endcsname=\threecolvfootnoteX
3336 \expandafter\let\csname footfmt#1\endcsname=\threecolfootfmtX
3337 \expandafter\let\csname footgroup#1\endcsname=\threecolfootgroupX
3338 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3339 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3340 \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3341 \threecolfootsetupX{#1}
3342 \ifnoledgroup\else%
3343 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3344 \expandafter\let\csname mpfootgroup#1\endcsname=\mpthreecolfootgroupX
3345 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%

```

```

3346 \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}
3347 \mpthreecolfootsetupX{#1}
3348 \fi%
3349 }
3350
3351 %

```

```

\threecolfootsetupX \threecolfootsetupX{<series>}
\mpthreecolfootsetupX
3352 \newcommand*{\threecolfootsetupX}[1]{%
3353 \count\csname footins#1\endcsname 333
3354 \csxdef{default@footins#1}{333}%Use this to confine the notes to one
side only
3355 \multiply\dimen\csname footins#1\endcsname by \thr@@}
3356 \newcommand*{\mpthreecolfootsetupX}[1]{%
3357 \count\csname mpfootins#1\endcsname 333
3358 \multiply\dimen\csname mpfootins#1\endcsname by \thr@@}
3359
3360 %

```

```

\threecolvfootnoteX \threecolvfootnoteX{<series>}{<text>}
3361 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolvfootnoteX}[2]{
%
3362 \insert\csname footins#1\endcsname\bgroup
3363 \csuse{notefontsizeX@#1}
3364 \footplitskips
3365 \@nameuse{footfmt#1}{#1}{#2}\egroup}
3366
3367 %

```

```

\threecolfootfmtX \threecolfootfmtX{<series>}
3368 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmtX}[2]{%
3369 \protected@edef\@currentlabel{%
3370 \@nameuse{@thefnmark#1}%
3371 }%
3372 \hangindent=\csuse{hangindentX@#1}%
3373 \normal@pars
3374 \hsize \csuse{hsizethreecolX@#1}
3375 \nottoggle{parindentX@#1}{\parindent=\z@}{ } %
3376 \tolerance=5000\relax
3377 \leavevmode
3378 \csuse{colalignX@#1}%
3379 {\csuse{notenumberX@#1}\@nameuse{footfootmark#1}\strut%
3380 #2\strut\par}\allowbreak}
3381
3382 %

```

```

\threecolfootgroupX \threecolfootgroupX{<series>}
\mpthreecolfootgroupX

```



```

3383 \newcommand*{\threecolfootgroupX}[1]{\csuse{notefontsizeX@#1}
3384 \splittopskip=\ht\strutbox
3385 \expandafter
3386 \rigidbalance\csname footins#1\endcsname \thr@@ \splittopskip}}
3387 \newcommand*{\mpthreecolfootgroupX}[1]{\{%
3388 \vskip\skip\@nameuse{mpfootins#1}
3389 \ifl@dpairing\ifparledgroup
3390 \leavevmode\marks\parledgroup@{begin}%
3391 \marks\parledgroup@series{#1}%
3392 \marks\parledgroup@type{footnoteX}%
3393 \fi\fi\normalcolor
3394 \ifparledgroup%
3395 \ifl@dpairing%
3396 \else%
3397 \setnoteswidthliketwocolumnsX@{#1}%
3398 \setnotesXpositionliketwocolumns@{#1}%
3399 \print@footnoteXrule{#1}%
3400 \fi%
3401 \else%
3402 \setnoteswidthliketwocolumnsX@{#1}%
3403 \setnotesXpositionliketwocolumns@{#1}%
3404 \print@footnoteXrule{#1}%
3405 \fi%
3406 \splittopskip=\ht\strutbox
3407 \expandafter
3408 \rigidbalance\csname mpfootins#1\endcsname \thr@@ \splittopskip}}
3409
3410 %

```

XIII.4.5 Paragraphed footnotes

The following macros set footnotes as one paragraph.

`\arrangementX@threecol` `\footparagraphX{<series>}`

```

3411 \newcommand*{\arrangementX@paragraph}[1]{\%
3412 \csgdef{series@displayX#1}{paragraph}%
3413 \expandafter\newcount\csname #1prevpage@num\endcsname
3414 \expandafter\let\csname footstart#1\endcsname=\parafootstartX
3415 \expandafter\let\csname regvfootnote#1\endcsname=\para@vfootnoteX
3416 \expandafter\let\csname footfmt#1\endcsname=\parafootfmtX
3417 \expandafter\let\csname footgroup#1\endcsname=\para@footgroupX
3418 \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3419 \count\csname footins#1\endcsname=1000
3420 \csxdef{default@footins#1}{1000}%Use this to confine the notes to one
side only
3421 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3422 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3423 \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
3424 \para@footsetupX{#1}

```

```

3425 \ifnoledgroup@else
3426   \expandafter\let\csname mpvfootnote#1\endcsname=\mppara@vfootnoteX
3427   \expandafter\let\csname mpfootgroup#1\endcsname=\mppara@footgroupX
3428   \count\csname mpfootins#1\endcsname=1000
3429   \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
3430   \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3431   \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
3432 \fi
3433 }
3434
3435 %

```

`\para@footsetupX` `\para@footsetupX{<series>}`

```

3436 \newcommand*{\para@footsetupX}[1]{\csuse{notefontsizeX@#1}
3437   \setnoteswidthliketwocolumnsX@{#1}%
3438   \dimen0=\baselineskip
3439   \multiply\dimen0 by 1024
3440   \divide\dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\relax
3441   %
3442   \expandafter
3443   \xdef\csname footfudgefactor#1\endcsname{%
3444     \expandafter\strip@pt\dimen0 }}
3445   %

```

`\parafootstartX` `\parafootstartX{<series>}`

```

3446 \newcommand*{\parafootstartX}[1]{%
3447   \ifdimequal{Opt}{\prenotesX@}{}%
3448   {%
3449     \iftoggle{prenotesX@}{%
3450       \togglefalse{prenotesX@}%
3451       \skip\csname footins#1\endcsname=%
3452       \dimexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
3453     }%
3454   }%
3455   }%
3456   \vskip\skip\csname footins#1\endcsname%
3457   \leftskip=\z@
3458   \rightskip=\z@
3459   \parindent=\z@
3460   \vskip\skip\@nameuse{footins#1}%
3461   \setnoteswidthliketwocolumnsX@{#1}%
3462   \setnotesXpositionliketwocolumns@{#1}%
3463   \print@footnoteXrule{#1}%
3464 }
3465
3466 %

```

`\para@vfootnoteX` `\para@vfootnoteX{<series>}{<text>}`
`\mppara@vfootnoteX`

```

3467 \newcommand*{\para@vfootnoteX}[2]{%
3468   \insert\csname footins#1\endcsname
3469   \bgroup
3470     \csuse{notefontsizeX@#1}
3471     \footsplitskips
3472     \setbox0=\vbox{\hsize=\maxdimen
3473       \noindent\csuse{bhooknoteX@#1}%
3474       \@nameuse{footfmt#1}{#1}{#2}}%
3475     \setbox0=\hbox{\unvXH{0}{#1}}%
3476     \dp0=\z@
3477     \ht0=\csname footfudgefactor#1\endcsname\wd0
3478     \box0
3479     \penalty0
3480   \egroup}
3481 \newcommand*{\mppara@vfootnoteX}[2]{%
3482   \global\setbox\@nameuse{mpfootins#1}\vbox{%
3483     \unvbox\@nameuse{mpfootins#1}
3484     \csuse{notefontsizeX@#1}
3485     \footsplitskips
3486     \setbox0=\vbox{\hsize=\maxdimen
3487       \noindent\color@begingroup%
3488       \csuse{bhooknoteX@#1}%
3489       \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}%
3490     \setbox0=\hbox{\unvXH{0}{#1}}%
3491     \dp0=\z@
3492     \ht0=\csname footfudgefactor#1\endcsname\wd0
3493     \box0
3494     \penalty0}}
3495 %
3496 %

```

```

\unvXH{0}{#1} \newcommand*{\unvXH}[2]{% 2th is optional for retro-compatibility
3498   \setbox0=\vbox{\unvbox#1%
3499   \global\setbox1=\lastbox}%
3500   \unhbox1
3501   \unskip           % remove \rightskip,
3502   \unskip           % remove \parfillskip,
3503   \unpenalty        % remove \penalty of 10000,
3504   \hskip\csuse{afternoteX@#2} % but add the glue to go between the notes
3505 %
3506 %

```

`\parafootfmtX` `\parafootfmtX{<series>}`

```

3507 \newcommand*{\parafootfmtX}[2]{%
3508   \protected@edef\@currentlabel{%
3509     \@nameuse{@thefnmark#1}%

```

```

3510 }%
3511 \insertparafootsepX{#1}%
3512 \ledsetnormalparstuffX{#1}%
3513 {\csuse{notenumfontX@#1}%
3514  \csuse{notenumfontX@#1}%
3515  \@nameuse{footfootmark#1}%
3516  \strut%
3517  #2\penalty-10}}
3518
3519 %

```

```

\para@footgroupX \para@footgroupX{<series>}
\mppara@footgroupX
3520 \newcommand*{\para@footgroupX}[1]{%
3521  \unvbox\csname footins#1\endcsname
3522  \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3523  \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3524  \makehboxofhboxes
3525  \setbox0=\hbox{\unhbox0 \removehboxes}%
3526  \csuse{notefontsizeX@#1}
3527  \noindent\unhbox0\par}
3528 \newcommand*{\mppara@footgroupX}[1]{%
3529  \setnoteswidthliketwocolumnsX@{#1}%
3530  \vskip\skip\@nameuse{mpfootins#1}
3531  \ifl@dpairing\ifparledgroup
3532    \leavevmode%
3533    \leavevmode\marks\parledgroup@{begin}%
3534    \marks\parledgroup@series{#1}%
3535    \marks\parledgroup@type{footnoteX}%
3536    \fi\fi\normalcolor
3537    \ifparledgroup%
3538      \ifl@dpairing%
3539        \else%
3540          \setnoteswidthliketwocolumnsX@{#1}%
3541          \setnotesXpositionliketwocolumns@{#1}%
3542          \print@footnoteXrule{#1}%
3543          \fi%
3544        \else%
3545          \setnoteswidthliketwocolumnsX@{#1}%
3546          \setnotesXpositionliketwocolumns@{#1}%
3547          \print@footnoteXrule{#1}%
3548          \fi%
3549        \unvbox\csname mpfootins#1\endcsname
3550        \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3551        \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3552        \makehboxofhboxes
3553        \setbox0=\hbox{\unhbox0 \removehboxes}%
3554        \csuse{notefontsizeX@#1}
3555        \noindent\unhbox0\par}}
3556

```

```
3557 %
```

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

```
\prevpage@num58 \newcommand{\insertparafootsepX}[1]{%
\Xinsertparafootsep59 \ifnumequal{\csuse{prevpage#1@num}}{\page@num}%
3560 {\csuse{parafootsepX@#1}}%
3561 {}%
3562 }
3563 %
```

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

`\par` should always be redefined to `\endgraf` within the format macro (this is what `\normal@pars` does), to override tricky material in the main text to get the lines numbered automatically (as set up by `\autopar`, for example).

In the case of footnote arranged in a “normal” way, we also must set some setting for paragraph indent and text direction when using Lua[®]TeX.

That why we have defined `\ledsetnormalparstuff@common` in order to make this setting for both familiar and critical notes. This command is called by command to make specific setting to critical or familiar footnote.

```
dsetnormalparstuff@common64 \newcommand*{\ledsetnormalparstuff@common}{%
\Xledsetnormalparstuff65 \ifluatex%
\ledsetnormalparstuffX66 \luatextextdir\footnote@luatextextdir%
3567 \luatexpardir\footnote@luatexpardir%
3568 \fi%
3569 \csuse{\csuse{footnote@dir}}%
3570 \normal@pars%
3571 \parfillskip \z@ \@plus 1fil}%
3572
3573 \newcommand*{\Xledsetnormalparstuff}[1]{%
3574 \ledsetnormalparstuff@common%
3575 \nottoggle{Xparindent@#1}{\noindent}{}%\noindent and and not \parindent=0
pt to avoid to break the (bad) change made when moving from ledmac to
eledmac
3576 }%
3577
3578 \newcommand*{\ledsetnormalparstuffX}[1]{%
3579 \ledsetnormalparstuff@common%
3580 \nottoggle{parindentX@#1}{\noindent}{}%\noindent and and not \parindent=0
pt to avoid to break the (bad) change made when moving from ledmac to
eledmac
```

```
3581 }%
3582 %
```

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@`

```
3583
3584 \newdimen\old@hsize%
3585 \AtBeginDocument{\old@hsize=\hsize}%
3586
3587 \newcommand{\setXnoteswidthliketwocolumns@}[1]{%
3588   \global\let\hsize@fornote=\hsize%
3589   \global\old@hsize=\hsize%
3590   \iftoggle{Xnoteswidthliketwocolumns@#1}%
3591     {%
3592       \csuse{setwidthliketwocolumns@\columns@position}%
3593       \global\let\hsize@fornote=\hsize%
3594     }%
3595     {}%
3596   \let\hsize=\hsize@fornote%
3597   \let\columnwidth=\hsize@fornote%
3598 }%
3599
3600 \newcommand{\setnoteswidthliketwocolumnsX@}[1]{%
3601   \global\let\hsize@fornote=\hsize%
3602   \global\old@hsize=\hsize%
3603   \iftoggle{noteswidthliketwocolumnsX@#1}%
3604     {%
3605       \csuse{setwidthliketwocolumns@\columns@position}%
3606       \global\let\hsize@fornote=\hsize%
3607     }%
3608     {}%
3609   \let\hsize=\hsize@fornote%
3610   \let\columnwidth=\hsize@fornote%
3611 }%
3612
3613 %
```

`\setXnotespositionliketwocolumns@` These two commands set the position of the critical / familiar footnotes, depending on the hooks `Xnoteswidthliketwocolumns` and `noteswidthliketwocolumnsX`. They call commands which are defined only in `reledpar`, because this feature has no sens without `reledpar`.

`\setnotesXpositionliketwocolumns@`

```

3614 \newcommand{\setXnotespositionliketwocolumns@}[1]{%
3615   \iftoggle{Xnoteswidthliketwocolumns@#1}{%
3616     \csuse{setnotespositionliketwocolumns@\columns@position}%
3617   }{}%
3618 }%
3619
3620 \newcommand{\setnotesXpositionliketwocolumns@}[1]{%
3621   \iftoggle{noteswidthliketwocolumnsX@#1}{%
3622     \csuse{setnotespositionliketwocolumns@\columns@position}%
3623   }{}%
3624 }%
3625
3626 %

```

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
3627 \def\@fnpos{familiar-critical}
3628 \def\@mpfnpos{critical-familiar}
3629 \newcommand{\fnpos}[1]{\xdef\@fnpos{#1}}
3630 \newcommand{\mpfnpos}[1]{\xdef\@mpfnpos{#1}}
3631 %

```

XVII Footnotes' rule

Because the footnotes' rules can be shifted to the right when footnotes are set like two columns, we do not print them directly, but we put them in a `\vbox`.

```

\print@Xfootnoterule32 \newcommand{\print@Xfootnoterule}[1]{%
\print@footnotexrule33 \vskip-\csuse{Xafterterrule@#1}%Because count in \dimen\csuse{#1footins}
3634 \nointerlineskip%
3635 \moveleft-\leftskip\vbox{\csuse{#1footnoterule}}%
3636 \nointerlineskip%
3637 \vskip\csuse{Xafterterrule@#1}%
3638 }%
3639
3640 \newcommand{\print@footnotexrule}[1]{%
3641 \vskip-\csuse{afterterruleX@#1}%Because count in \dimen\csuse{footins#1}
3642 \nointerlineskip%
3643 \moveleft-\leftskip\vbox{\csuse{footnoterule#1}}%
3644 \nointerlineskip%
3645 \vskip\csuse{afterterruleX@#1}%
3646 }%
3647
3648 %

```

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.

```

3649 \newtoggle{preXnotes@}
3650 \toggletrue{preXnotes@}
3651 \newcommand{\preXnotes@}{0pt}
3652 \newcommand*{\preXnotes}[1]{\renewcommand{\preXnotes@}{#1}}
3653 %

```

The same, but for familiar footnotes.


```

\preXnotes54 \newtoggle{prenotesX@}
\preXnotes55 \toggletrue{prenotesX@}
3656 \newcommand{\prenotesX@}{Opt}
3657 \newcommand*{\prenotesX}[1]{\renewcommand{\prenotesX@}{#1}}
3658 %

```

XVIII.0.3 Internal commands

```

firstXseries59 \gdef\firstXseries@{}
prepare@preXnotes60 \newcommand{\prepare@preXnotes}[1]{%
3661 \ifdimequal{Opt}{\preXnotes@}%
3662 }%
3663 {%
3664 \IfStrEq{\firstXseries@}{}{%
3665 \global\skip\csuse{#1footins}=\preXnotes@%
3666 \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
3667 \gdef\firstXseries@{#1}%
3668 }%
3669 {%
3670 \ifseriesbefore{#1}{\firstXseries@}%
3671 {%
3672 \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@\firstXseries@}%
3673 \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
3674 \gdef\firstXseries@{#1}%
3675 }%
3676 }%
3677 }%
3678 }%
3679 }
3680 %

```

The same thing is required for familiar notes and \prenotesX.

```

firstseriesX81 \gdef\firstseriesX@{}
prepare@prenotesX82 \newcommand{\prepare@prenotesX}[1]{%
3683 \ifdimequal{Opt}{\prenotesX@}%
3684 }%
3685 {%
3686 \IfStrEq{\firstseriesX@}{}{%
3687 \global\skip\csuse{footins#1}=\prenotesX@%
3688 \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
3689 \gdef\firstseriesX@{#1}%
3690 }%
3691 {%
3692 \ifseriesbefore{#1}{\firstseriesX@}%

```

```

3693     {%
3694     \global\skip\csuse{footins#1}=\csuse{beforenotesX@\firstseriesX@}%
3695     \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
3696     \gdef\firstXseries@{#1}%
3697     }%
3698     {}%
3699     }%
3700     }%
3701 }
3702 %

```

XIX Endnotes

First, check the noend option.

```

3703 \ifbool{noend@}{-}{%Used instead of \ifnoend@ to prevent expansion problem
3704 %

```

\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.

```

3705 \newcommand{\l@dend@open}[1]{%
3706   \global\booltrue{l@dend@#1}%
3707   \expandafter\immediate%
3708   \expandafter\openout%
3709   \csname l@d@#1end\endcsname%
3710   =\jobname.#1end\relax%
3711 }%
3712 \newcommand{\l@dend@close}[1]{%
3713   \global\boolfalse{l@dend@#1}%
3714   \expandafter\immediate%
3715   \expandafter\closeout\csname l@d@#1end\endcsname%
3716 }%
3717
3718 %

```

\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.

```

3719 \newcommand{\l@dend@stuff}{%
3720   \def\do##1{%
3721     \ifbool{l@dend@##1}{%
3722       {\l@dend@open{##1}}%

```

```

3723 \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname{\
string\l@d@section{\the\section@num}}}%
3724 }%
3725 \dolistloop{\@series}%
3726 }%
3727
3728 %

```

\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`.

```

3729 \global\newbool{parapparatus@}\def\endprint#1#2#3#4#5{%
3730 \ifXendinsertsep%
3731 \hskip\csuse{Xendafternote@#4}%
3732 \csuse{Xendsep@#4}%
3733 \else%
3734 \iftoggle{Xendparagraph@#4}%
3735 {\global\Xendinsertsep@true}%
3736 {}%
3737 \fi%
3738 \xdef\@currentseries{#4}%
3739 \def\do##1{%
3740 \toggletrue{##1}%
3741 }%
3742 \notblank{#5}{\docsvlist{#5}}}%
3743 \csuse{Xendbhooknote@#4}%
3744 \csuse{Xendnotefontsize@#4}%
3745 \ifboolexpr{%
3746 togl {nonum@}%
3747 or togl {Xendnonumber@#4}%
3748 }%
3749 {\hspace{\csuse{Xendinplaceofnumber@#4}}}%
3750 {\printlineendnotearea{#1}{#4}}%
3751 \nottoggle{Xendlemmadisablefontselection@#4}%
3752 {\select@lemmafont#1|#2}%
3753 {#2}%
3754 \ifboolexpr{%

```

```

3755     togl {nosep@}%
3756     or test{\ifcsemt{Xendlemmaseparator@#4}}%
3757     }%
3758     {\hskip\csuse{Xendinplaceoflemmaseparator@#4}}%
3759     {\nobreak%
3760     \hskip\csuse{Xendbeforelemmaseparator@#4}%
3761     \csuse{Xendlemmaseparator@#4}%
3762     \hskip\csuse{Xendafterlemmaseparator@#4}%
3763     }%
3764     #3%
3765     \nottoggle{Xendparagraph@#4}{\par}{}%
3766     \def\do##1{%
3767     \togglefalse{##1@}%
3768     }%
3769     \notblank{#5}{\docsvlist{#5}}{}%
3770 }}%
3771
3772 \let\l@d@section=\@gobble
3773
3774 %

```

\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.)

```

3775 \newcommand{\printlineendnotearea}[2]{%
3776   \bgroup%
3777   \csuse{Xendnotenumfont@#2}%
3778   \ifdimequal{\csuse{Xendboxlinenum@#2}}{0pt}%
3779   {\printendlines#1|}%
3780   {\leavevmode%
3781     \hbox to \csuse{Xendboxlinenum@#2}%
3782     {%
3783       \IfSubStr{RC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}{}%
3784       \printendlines#1|}%
3785       \IfSubStr{LC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}{}%
3786     }%
3787   \egroup%
3788   \enspace%
3789 }%
3790 %

```

\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.

```

3791 \newcommand*{\setprintendlines}[6]{%
3792   \l@dpnumfalse \l@ddashfalse
3793   \ifnum#4=#1 \else
3794     \l@dpnumtrue
3795     \l@ddashtrue
3796   \fi
3797   %

```

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.

```

3798   \ifl@dpnum \l@d@elintrue \else \l@d@elinfalse \fi
3799   \ifnum#2=#5 \else
3800     \l@d@elintrue
3801     \l@ddashtrue
3802   \fi
3803   %

```

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

```

3804   \l@d@ssubfalse
3805   \ifnum#3=0 \else
3806     \l@d@ssubtrue
3807   \fi
3808   %

```

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.

```

3809   \l@d@eslfalse
3810   \ifnum#6=0 \else
3811     \ifnum#6=#3
3812       \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
3813     \else
3814       \l@d@esltrue
3815       \l@ddashtrue
3816     \fi
3817   \fi%
3818   %

```

```

3819   \ifl@d@dash%
3820   \ifboolexpr{togl{fulllines@} or test{\ifcsemtty{Xendtwolines@}\
@currentseries}}}%
3821   {}%
3822   {%
3823   \setistwofollowinglines{#1}{#2}{#4}{#5}%
3824   \ifboolexpr{%

```

```

3825      (%)
3826      togl {Xendtwolinesbutnotmore@\@currentseries}%
3827      and not%
3828      (%)
3829      bool {istwofollowinglines@}%
3830      )%
3831    )%
3832    or%
3833    (%)
3834      (not test{\ifnumequal{#1}{#4}})%
3835      and togl{Xendtwolinesonlyinsamepage@\@currentseries}%
3836      )%
3837    }%
3838    {}%
3839    {%
3840      \l@d@dashfalse%
3841      \l@d@Xtwolinestrue%
3842      \l@d@elinfalse%
3843      \l@d@eslfalse%
3844      \ifcempty{Xendmorethantwolines@\@currentseries}%
3845      {%
3846        \ifistwofollowinglines@\else%
3847          \l@d@Xmorethantwolinestrue%
3848          \fi%
3849        }%
3850      }%
3851    }%
3852  \fi%
3853 %

```

End of \setprintendlines.

```

3854 }%
3855 %

```

\printendlines Now we are ready to print it all.

```

3856 \def\printendlines#1|#2|#3|#4|#5|#6|#7|{\begingroup
3857   \setprintendlines{#1}{#2}{#3}{#4}{#5}{#6}%
3858 %

```

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.

```

3859   \ifdimequal{\csuse{Xendboxstartlinenum@\@currentseries}}{0pt}%
3860   {\bgroup}%
3861   {\leavevmode\hbox to \csuse{Xendboxstartlinenum@\@currentseries}\bgroup
\hfill}%

```

```

3862 \printnpnum{#1}%
3863 \linenumrep{#2}%
3864 \ifl@d@ssub \fullstop \sublinenumrep{#3}\fi
3865 \egroup%
3866 %

```

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

```

3867 \ifdimequal{\csuse{Xendboxendlinenum@}\@currentseries}{0pt}%
3868 {\bgroup}%
3869 {\hbox to \csuse{Xendboxendlinenum@}\@currentseries}\bgroup}%
3870 \ifl@d@Xtwolines%
3871 \ifl@d@Xmorethantwolines%
3872 \csuse{Xendmorethantwolines@}\@currentseries}%
3873 \else%
3874 \csuse{Xendtwolines@}\@currentseries}%
3875 \fi%
3876 \else%
3877 \ifl@d@dash \endashchar\fi%
3878 \ifl@d@pnum \printnpnum{#4}\fi%
3879 \ifl@d@elin \linenumrep{#5}\fi%
3880 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumrep{#6}\fi%
3881 \fi%
3882 \ifdimequal{\csuse{Xendboxendlinenum@}\@currentseries}{0pt}%
3883 {}%
3884 {\hfill}%Prevent underfull hbox
3885 \egroup%
3886 \endgroup%
3887 }%
3888 %
3889 %

```

\printnpnum A macro to print a page number in an endnote.

```

3890 \newcommand*{\printnpnum}[1]{p.#1} }
3891
3892 %

```

\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.

```

3893 \newif\ifXendinsertsep@%
3894 \newcommand*{\doendnotes}[1]{%
3895 \l@dend@close{#1}%
3896 \begingroup
3897 \makeatletter
3898 \expandafter\let\csname #1end\endcsname=\endprint
3899 \input\jobname.#1end%
3900 \global\Xendinsertsep@false%
3901 \endgroup}

```

```
3902 %
```

\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.

```
3903 \newcommand*{\doendnotesbysection}[1]{%
3904   \l@dend@close{#1}%
3905   \global\expandafter\advance\csname #1end@bysection\endcsname by 1%
3906   \begingroup%
3907     \makeatletter%
3908     \def\l@d@section##1{%
3909       \ifnumequal{##1}{\csname #1end@bysection\endcsname}%
3910         {\cslet{#1end}{\endprint}}%
3911         {\cslet{#1end}{\@gobblefive}}%
3912       }%
3913       \input\jobname.#1end%
3914       \global\Xendinsertsep@false%
3915     \endgroup%
3916   }%
3917   %
```

End of section for end notes

```
3918 }%
3919 %
```

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@.

```
3920 \newcommand{\newseries}[1]{%
3921   \def\do##1{\newseries@{##1}}%
3922   \docsvlist{#1}
3923 }
3924 %
```

\@series The \series@ macro is an etoolbox list, which contains the name of all series.

```
3925 \newcommand{\@series}{}
3926 %
```

The command \newseries@\series creates a new series of the footnote.


```
\newseries@27 \newcommand{\newseries@}[1]{
3928 %
```

XX.1 Test if series is still existing

```
3929 \xifinlist{#1}{\@series}{\led@warn@SeriesStillExist{#1}}%
3930 {%
3931 %
```

XX.2 Init specific to reledpar

When calling `\newseries@` after having loaded `reledpar`, we need to load specific setting.

```
3932 \ifdefined\newseries@par%
3933 \newseries@par{#1}%
3934 \fi%
3935 %
```

XX.3 For critical footnotes

Critical footnotes are those which start with letters. We look for the `\nocritical` option of `reledmac`.

```
3936 \unless\ifnocritical@
3937 %
```

XX.3.1 Options

```
3938 \newtoggle{Xparindent@#1}
3939 \newtoggle{Xlemmadisablefontselection@#1}
3940 \csgdef{Xhangindent@#1}{Opt}%
3941 \csgdef{Xragged@#1}{}%
3942 \csgdef{Xhsizetwocol@#1}{0.45 \hsize}%
3943 \csgdef{Xhsizethreecol@#1}{.3 \hsize}%
3944 \csgdef{Xcolalign@#1}{\raggedright}%
3945 \csgdef{Xnotenumfont@#1}{\normalfont}%
3946 \csgdef{Xnotefontsize@#1}{\footnotesize}%
3947 \csgdef{Xbhooknote@#1}{}%
3948
3949 \csgdef{Xboxlinenum@#1}{Opt}%
3950 \csgdef{Xboxlinenumalign@#1}{L}%
3951
3952 \csgdef{Xboxstartlinenum@#1}{Opt}%
3953 \csgdef{Xboxendlinenum@#1}{Opt}%
3954
3955 \csgdef{Xboxsymlinenum@#1}{Opt}%
3956 \newtoggle{Xnumberonlyfirstinline@#1}%
```

```

3957 \newtoggle{Xnumberonlyfirstintwelines@#1}%
3958 \csgdef{Xtwelines@#1}{}%
3959 \csgdef{Xmorethantwelines@#1}{}%
3960 \newtoggle{Xtwelinesbutnotmore@#1}%
3961 \newtoggle{Xtwelinesonlyinsamepage@#1}%
3962 \newtoggle{Xonlypstart@#1}%
3963 \newtoggle{Xpstarteverytime@#1}%
3964 \newtoggle{Xpstart@#1}%
3965 \newtoggle{Xstanza@#1}%
3966 \csgdef{Xstanzaseparator@#1}{}%
3967 \csgdef{Xsymlinenum@#1}{}%
3968 \newtoggle{Xnonumber@#1}%
3969 \csgdef{Xbeforenumber@#1}{0pt}%
3970 \csgdef{Xafternumber@#1}{0.5em}%
3971 \newtoggle{Xnonbreakableafternumber@#1}%
3972 \csgdef{Xbeforesymlinenum@#1}{\csuse{Xbeforenumber@#1}}%
3973 \csgdef{Xaftersymlinenum@#1}{\csuse{Xafternumber@#1}}%
3974 \csgdef{Xinplaceofnumber@#1}{1em}%
3975 \global\cslet{Xlemmaseparator@#1}{\rbracket}%
3976 \csgdef{Xbeforelemmaseparator@#1}{0em}%
3977 \csgdef{Xafterlemmaseparator@#1}{0.5em}%
3978 \csgdef{Xinplaceoflemmaseparator@#1}{1em}%
3979 \csgdef{Xbeforenotes@#1}{1.2em \@plus .6em \@minus .6em}%
3980 \csgdef{Xafterrule@#1}{0pt}%
3981 \csgdef{Xtxtbeforenotes@#1}{}%
3982 \csgdef{Xmaxhnotes@#1}{0.8\vsizel}%
3983 \newtoggle{Xnoteswidthliketwocolumns@#1}%
3984 \csgdef{Xparafootsep@#1}{}%
3985 \csgdef{Xafternote@#1}{1em plus.4em minus.4em}%
3986 %

```

XX.3.2 Create inserts, needed to add notes in foot

As regards inserts, see chapter 15 of *The TeXbook* by D. Knuth.

```

3987 \expandafter\newinsert\csname #1footins\endcsname%
3988 \unless\ifnoledgroup%
3989 \expandafter\newinsert\csname mp#1footins\endcsname%
3990 \fi%
3991 %

```

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.

```

3992 \global\newcommand{\parapparatus@}{\expandafter\newcommand\expandafter
3993 *}{\expandafter\newcommand}\csname #1footnote\endcsname[2][{}]{%
3994 \ifnum\@edtext@level>0%
3995 \begingroup%
3996 \newcommand{\content}{##2}%

```

```

3996         \ifnumberedpar{%
3997             \ifledRcol%
3998                 \ifluatex%
3999                     \footnotelang@lua[R]%
4000             \fi%
4001             \@ifundefined{xpg@main@language}%if polyglossia
4002                 {}%
4003                 {\footnotelang@poly[R]}%
4004             \footnoteoptions@{R}{##1}{true}%
4005             \xright@appenditem{%
4006                 \noexpand\prepare@preXnotes{#1}%
4007                 \noexpand\prepare@edindex@fornote{\l@d@nums}%
4008                 \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.
4009                 \noexpand\setcounter{stanzaR}{\the\c@stanzaR}%Save
stanzaR counter for footnote
4010                 \noexpand\csuse{v#1footnote}{#1}%
4011                 {\l@d@nums}{\expandonce\@tag}{\expandonce\content}}
%
4012             }\to\inserts@listR
4013             \footnoteoptions@{R}{##1}{false}%
4014             \global\advance\insert@countR \@ne%
4015         \else%
4016             \ifluatex%
4017                 \footnotelang@lua%
4018             \fi%
4019             \@ifundefined{xpg@main@language}%if polyglossia
4020                 {}%
4021                 {\footnotelang@poly}%
4022             \footnoteoptions@{##1}{true}%
4023             \xright@appenditem{%
4024                 \noexpand\prepare@preXnotes{#1}%
4025                 \noexpand\prepare@edindex@fornote{\l@d@nums}%
4026                 \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.
4027                 \ifl@d@pairing%
4028                     \noexpand\setcounter{stanzaL}{\the\c@stanzaL}%Save
stanzaR counter for footnote
4029                 \fi%
4030                 \noexpand\csuse{v#1footnote}{#1}%
4031                 {\l@d@nums}{\expandonce\@tag}{\expandonce\content}}
%
4032             }\to\inserts@list
4033             \global\advance\insert@count \@ne%
4034             \footnoteoptions@{##1}{false}%
4035         \fi

```



```

4068 \csgdef{afternoteX@#1}{1em plus.4em minus.4em}
4069 % End of for familiar footnotes.
4070 % \subsubsection{Create inserts, needed to add notes in foot}
4071 % As regards inserts, see chapter 15 of the TeXBook by D. Knuth.
4072 % \begin{macrocode}
4073 \expandafter\newinsert\csname footins#1\endcsname%
4074 \unless\ifnoledgroup%
4075 \expandafter\newinsert\csname mpfootins#1\endcsname%
4076 \fi%
4077 %

```

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.

```

4078 \global\expandafter\newcommand\csname footnote#1\endcsname[1]{%
4079 \begingroup%
4080 \prepare@prenotesX{#1}%
4081 \newcommand{\content}{##1}%
4082 \stepcounter{footnote#1}%
4083 \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
4084 \nottoggle{nomk@}%Nomk is set to true when using \
footnoteXnomk with \parpackage
4085 {\csuse{@footnotemark#1}}%
4086 {}%
4087 \ifluatex%
4088 \xdef\footnote@luatextextdir{\the\luatextextdir}%
4089 \xdef\footnote@luatexpardir{\the\luatexpardir}%
4090 \fi%
4091 \csuse{vfootnote#1}{#1}{\expandonce\content}\m@mmf@prepare%
4092 \endgroup%
4093 }
4094 %
4095 %

```

Then define the counters.

```

4096 \newcounter{footnote#1}
4097 \global\expandafter\renewcommand\csname thefootnote#1\endcsname{\
arabic{footnote#1}}
4098 %

```

Do not forget to initialize series

```

4099 \arrangementX@normal{#1}%
4100 \fi
4101 %

```

XX.5 The endnotes

Endnotes are commands like \Xendnote, where X is a series letter. First, we check for the noend options.

```

4102 \unless\ifnoend@
4103 %

```

XX.5.1 The auxiliary file

Endnotes of all varieties are saved up in a file, one by series, typically named $\langle jobname \rangle$.Xend. $\backslash l@d@Xend$ is the output stream number for this file, and $\backslash ifl@dend@X$ is a flag that is true when the file is open.

```

\l@d@Xend
\ifl@dend@X
\l@dend@Xtrue
\l@dend@Xfalse
4104 \expandafter\newwrite\csname l@d@#1end\endcsname%
4105 \expandafter\newif\csname ifl@dend@#1\endcsname%
4106 %

```

XX.5.2 The main macro

The $\backslash Xendnote$ macro functions to write one endnote to the .Xend file. We change $\backslash newlinechar$ 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.

```

4107
4108 \global\expandafter\newcommandx\csname #1endnote\endcsname[2][1,
usedefault]{%
4109 \bgroup%
4110 \newlinechar='40%
4111 \global\@noneed@Footnotetrue%
4112 \newcommand{\content}{##2}%
4113 \expandafter\immediate\expandafter\write\csname l@d@#1end\
endcsname{%
4114 \expandafter\string\csname #1end\endcsname%
4115 {\ifnumberedpar@l@d@nums\fi}%
4116 {\ifnumberedpar@\expandonce\@tag\fi}%
4117 {\expandonce\content}%
4118 {\#1}%
4119 {\##1}%
4120 \@percentchar%
4121 }%
4122 \egroup%
4123 \ignorespaces%
4124 }%
4125 %

```

$\backslash Xendnote$ commands called $\backslash Xend$ 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 $\backslash endprint$, and leave the rest equated to $\backslash @gobblefive$, which just skips over its five arguments.

```

4126
4127 \global\cslet{#1end}{\@gobblefive}

```

```

4128 %
We need to store the number of times \doendnotesbysection is called for one series.
4129 \global\expandafter\newcount\csname #1end@bysection\endcsname%
4130 %

```

XX.5.3 The options

```

4131 \csgdef{Xendtwolines@#1}{}%
4132 \csgdef{Xendmorethantwolines@#1}{}%
4133 \newtoggle{Xendtwolinesbutnotmore@#1}{}%
4134 \newtoggle{Xendtwolinesonlyinsamepage@#1}{}%
4135 \newtoggle{Xendlemmadisablefontselection@#1}{}%
4136 \csgdef{Xendnotenumfont@#1}{\normalfont}%
4137 \csgdef{Xendnotefontsize@#1}{\footnotesize}%
4138 \csgdef{Xendbhooknote@#1}{}%
4139
4140 \csgdef{Xendboxlinenum@#1}{0pt}%
4141 \csgdef{Xendboxlinenumalign@#1}{L}%
4142
4143 \csgdef{Xendboxstartlinenum@#1}{0pt}%
4144 \csgdef{Xendboxendlinenum@#1}{0pt}%
4145
4146 \csgdef{Xendlemmaseparator@#1}{}%
4147 \csgdef{Xendbeforelemmaseparator@#1}{0em}%
4148 \csgdef{Xendafterlemmaseparator@#1}{0.5em}%
4149 \csgdef{Xendinplaceoflemmaseparator@#1}{0.5em}%
4150
4151 \newtoggle{Xendparagraph@#1}{}%
4152 \csgdef{Xendafternote@#1}{1em plus.4em minus.4em}%
4153 \csgdef{Xendsep@#1}{}%
4154
4155 \csgdef{Xendinplaceofnumber@#1}{0pt}%
4156 \newtoggle{Xendnonumber@#1}{}%
4157 %

```

End of endnotes declaration

```

4158 \fi%
4159 %

```

Dump series in \@series

```

4160 \listxadd{\@series}{#1}
4161 }
4162 }% End of \newseries
4163 %

```

XX.6 Init standards series (A,B,C,D,E)

```

4164 \expandafter\newseries\expandafter{\default@series}
4165 %

```

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.

```

4166 \newcommand{\seriesatbegin}[1]{%
4167   \StrDel{\@series}{#1}[\@series]%
4168   \edef\@new{}%
4169   \listadd{\@new}{#1}%
4170   \listadd{\@new}{\@series}%
4171   \xdef\@series{\@new}%
4172 }
4173 %

```

\seriesatend And `\seriesatend` moves the series to the end of the list.

```

4174 \newcommand{\seriesatend}[1]{%
4175   \StrDel{\@series}{#1}[\@series]%
4176   \edef\@new{}%
4177   \listadd{\@new}{\@series}%
4178   \listadd{\@new}{#1}%
4179   \xdef\@series{\@new}%
4180 }
4181 %

```

XXI.2 Test series order

\ifseriesbefore `\ifseriesbefore{<seriesA>}{<seriesB>}{<true>}{<false>}` expands `<true>` if `<seriesA>` is printed before `<seriesB>`, expands `<false>` otherwise.

```

4182 \newcommand{\ifseriesbefore}[4]{%
4183   \StrPosition{\@series}{#1}[\@first]%
4184   \StrPosition{\@series}{#2}[\@second]%
4185   \ifnumgreater{\@second}{\@first}{#3}{#4}%
4186 }
4187 %

```

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}{%
4189   \ifdefempty{\@series}%
4190   {\xdef\@firstseries{}}%

```



```

4191 {\StrChar{\@series}{1}[\@firstseries]}\%
4192 }%
4193 %

```

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`.

```

4194 \newcommandx{\settoggle@series}[5][4,5,usedefault]{%
4195   \def\do##1{%
4196     \global\settoggle{#2@##1}{#3}%
4197     \ifstrequal{#4}{reload}%
4198       {%

```

```

4199     \csuse{Xarrangement@\csuse{series@display##1}}{##1}%
4200     \csuse{arrangementX@\csuse{series@displayX##1}}{##1}%
4201     }%
4202     {}%
4203     }%
4204     \ifstrempy{#1}{%
4205         \dolistloop{\@series}%
4206         \ifstrempy{#5}{}{%
4207             \docsvlist{#5}%
4208         }
4209     }%
4210     {%
4211         \docsvlist{#1}%
4212     }%
4213 }
4214 %

```

`\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`.

```

4215 \newcommandx{\setcommand@series}[5][4,5,usedefault]{%
4216     \def\do##1{
4217         \csgdef{#2@##1}{#3}
4218         \ifstrequal{#4}{reload}{
4219             \csuse{Xarrangement@\csuse{series@display##1}}{##1}%
4220             \csuse{arrangementX@\csuse{series@displayX##1}}{##1}%
4221         }{}
4222         \ifstrempy{#1}{%
4223             \dolistloop{\@series}%
4224             \ifstrempy{#5}{}{%
4225                 \docsvlist{#5}
4226             }
4227         }%
4228         {%
4229             \docsvlist{#1}%
4230         }%
4231     }%
4232 }%

```

XXI.3.3 Tools to generate options commands

`\newhookcommand@series` `\newhookcommand@series\command names` is a generic command to add new commands for hooks, like `\Xhsizetwocol`. The first argument is the name of the hook, the second a comma separated list of pseudo-series where the hook can be used, like `appref` in the case of `\Xtwolines`. The second argument is also used to create commands named `\<hookname><pseudoseris>`, like `\Xtwolinesappref`.

```

4233 \newcommandx{\newhookcommand@series}[2][2,usedefault]{%
4234   \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][]{%
4235     \setcommand@series{##1}{#1}{##2}[][#2]%
4236   }%
4237   \ifstrepty{#2}{-}{%
4238     \def\do##1{%
4239       \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname
4240       [1]{%
4241         \csuse{#1}{##1}{####1}%
4242       }%
4243     }%
4244   }%
4245 }
4246 %

```

`\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><pseudoseris>`, like `\Xtwolinesbutnotmoreappref`.

```

4247 \newcommandx{\newhooktoggle@series}[2][2,usedefault]{%
4248   \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={
4249     true},usedefault]{%
4250     \settoggle@series{##1}{#1}{##2}[][#2]%
4251   }%
4252   \ifstrepty{#2}{-}{%
4253     \def\do##1{%
4254       \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname{%
4255         \csuse{#1}{##1}%
4256       }%
4257     }%
4258   }%
4259 }
4260 %

```

`\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.

```

4261 \newcommand{\newhooktoggle@series@reload}[1]{%
4262   \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={
true},usedefault]{%

```

```

4263 \settoggle@series{##1}{#1}{##2}[reload]%
4264 }%
4265 }%
4266 %

```

`\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.

```

4267 \newcommand{\newhookcommand@series@reload}[1]{%
4268 \global\expandafter\newcommand\expandafter*\csname #1\endcsname [2] [] {%
4269 \setcommand@series{##1}{#1}{##2}[reload]%
4270 }%
4271 }
4272 %

```

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.

```

4273 \unless\ifnocritical@
4274 \newhooktoggle@series{Xparindent}
4275 \newhookcommand@series{Xtwolines}[appref]
4276 \newhookcommand@series{Xmorethantwolines}[appref]
4277 \newhooktoggle@series{Xtwolinesbutnotmore}[appref]
4278 \newhooktoggle@series{Xtwolinesonlyinsamepage}[appref]
4279 \newhookcommand@series{Xhangindent}
4280 \newhookcommand@series{Xragged}
4281 \newhookcommand@series{Xhsizetwocol}
4282 \newhookcommand@series{Xhsizethreecol}
4283 \newhookcommand@series{Xcolalign}%
4284 \newhookcommand@series{Xnotenumfont}
4285 \newhookcommand@series{Xbhooknote}
4286 \newhookcommand@series{Xboxsymlinenum}%
4287 \newhookcommand@series{Xsymlinenum}
4288 \newhookcommand@series{Xbeforenumber}
4289 \newhookcommand@series{Xafternumber}
4290 \newhookcommand@series{Xbeforesymlinenum}
4291 \newhookcommand@series{Xaftersymlinenum}
4292 \newhookcommand@series{Xinplaceofnumber}
4293 \newhookcommand@series{Xlemmaseparator}
4294 \newhookcommand@series{Xbeforelemmaseparator}
4295 \newhookcommand@series{Xafterlemmaseparator}
4296 \newhookcommand@series{Xinplaceoflemmaseparator}
4297 \newhookcommand@series{Xtxtbeforenotes}
4298 \newhookcommand@series@reload{Xafterrule}
4299 \newhooktoggle@series{Xnumberonlyfirstinline}
4300 \newhooktoggle@series{Xnumberonlyfirstintwolines}
4301 \newhooktoggle@series{Xnonumber}

```

```

4302 \newhooktoggle@series{Xpstart}
4303 \newhooktoggle@series{Xpstarteverytime}%
4304
4305 \newhooktoggle@series{Xstanza}%
4306 \newhookcommand@series{Xstanzaseparator}%
4307
4308 \newhooktoggle@series{Xonlypstart}
4309 \newhooktoggle@series{Xnonbreakableafternumber}
4310 \newhooktoggle@series{Xlemmadisablefontselection}
4311 \newhookcommand@series@reload{Xmaxhnotes}
4312 \newhookcommand@series@reload{Xbeforenotes}
4313 \newhooktoggle@series@reload{Xnoteswidthliketwocolumns}%
4314 \newhookcommand@series{Xnotefontsize}
4315
4316 \newhookcommand@series{Xboxlinenum}%
4317 \newhookcommand@series{Xboxlinenumalign}%
4318
4319 \newhookcommand@series{Xboxstartlinenum}%
4320 \newhookcommand@series{Xboxendlinenum}%
4321
4322 \newhookcommand@series{Xafternote}%
4323 \newhookcommand@series{Xparafootsep}
4324
4325 \fi
4326 %

```

XXI.3.5 Options for familiar notes

Before generating the optional commands for familiar notes, we check the `\nofamiliar` option.

```

4327 \unless\ifnofamiliar@
4328 \newhooktoggle@series{parindentX}
4329 \newhookcommand@series{hangindentX}
4330 \newhookcommand@series{raggedX}
4331 \newhookcommand@series{hsizetwocolX}
4332 \newhookcommand@series{hsizethreecolX}
4333 \newhookcommand@series{colalignX}%
4334 \newhookcommand@series{notenumfontX}
4335 \newhookcommand@series{bhooknoteX}
4336 \newhookcommand@series@reload{beforenotesX}
4337 \newhookcommand@series@reload{maxhnotesX}
4338 \newhooktoggle@series@reload{noteswidthliketwocolumnsX}%
4339 \newhookcommand@series@reload{afterruleX}
4340 \newhookcommand@series{notefontsizeX}
4341 \newhookcommand@series{afternoteX}
4342 \newhookcommand@series{parafootsepX}
4343 \fi
4344 %

```

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.

```

4345 \unless\ifnoend@
4346   \newhookcommand@series{Xendtwolines}[apprefwithpage]
4347   \newhookcommand@series{Xendmorethantwolines}[apprefwithpage]
4348   \newhooktoggle@series{Xendtwolinesbutnotmore}[apprefwithpage]
4349   \newhooktoggle@series{Xendtwolinesonlyinsamepage}[apprefwithpage]
4350   \newhookcommand@series{Xendnotenumfont}
4351   \newhookcommand@series{Xendbhooknote}
4352
4353   \newhookcommand@series{Xendboxlinenum}%
4354   \newhookcommand@series{Xendboxlinenumalign}%
4355
4356   \newhookcommand@series{Xendboxstartlinenum}%
4357   \newhookcommand@series{Xendboxendlinenum}%
4358
4359   \newhookcommand@series{Xendnotefontsize}
4360   \newhooktoggle@series{Xendlemmadisablefontselection}
4361   \newhookcommand@series{Xendlemmaseparator}
4362   \newhookcommand@series{Xendbeforelemmaseparator}
4363   \newhookcommand@series{Xendafterlemmaseparator}
4364   \newhookcommand@series{Xendinplaceoflemmaseparator}
4365
4366   \newhooktoggle@series{Xendparagraph}
4367   \newhookcommand@series{Xendafternote}
4368   \newhookcommand@series{Xendsep}
4369
4370   \newhookcommand@series{Xendinplaceofnumber}%
4371   \newhooktoggle@series{Xendnonumber}%
4372 \fi
4373 %

```

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`.

```

4374 \newtoggle{fulllines@}%
4375 %

```

`\nonum@` `\nonum@` toggle is used to disable line number printing in a particular footnote.

```

4376 \newtoggle{nonum@}
4377 %

```

`\nosep@` `\nonum@` toggle is used to disable the lemma separator in a particular footnote.

```
4378 \newtoggle{nosep@}
4379 %
```

\nomk@ \nomk@ toggle is used by reledpar to remove the footnote mark in the text when using \footnoteXmk. Read reledpar handbook.

```
4380 \newtoggle{nomk@}%
4381 %
```

XXI.5 Alias

\Xnolemmaseparator \Xnolemmaseparator[*<series>*] is just an alias for \Xlemmaseparator[*<series>*]{ }.

```
4382 \newcommand*{\Xnolemmaseparator}[1][1]{\Xlemmaseparator[#1]{}}
4383 %
```

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

```
4384 \countdef\pageno=0 \pageno=1
4385 \newcommand*{\advancepageno}{\ifnum\pageno<\z@ \global\advance\pageno\m@ne
4386 \else\global\advance\pageno\@ne\fi}
4387
4388 %
```

XXII.0.2 Extra footnotes output

With luck we might only have to change \@makecol and \@reinserts of the \TeX 's kernel. Since reledmac, we use etoolbox's patching commands instead of overriding. It should provides better compatibility with other package which modify these commands

\doxtrafeet \doxtrafeet is the code extending \@makecol to cater for the extra reledmac feet. We have two categories of extra footnotes. By default, we order the footnote inserts so that the regular footnotes of \TeX are first, then familiar familiar footnotes and finally the critical footnotes.

```
4389 \newcommand*{\l@ddoxtrafeet}{%
4390 \IfStrEq{familiar-critical}{\@fnpos}
4391 {\do@feetX\Xdo@feet}%
4392 %
4393 \IfStrEq{critical-familiar}{\@fnpos}%
4394 {\Xdo@feet\do@feetX}%
```

```

4395     {\do@feetX\Xdo@feet}%
4396   }%
4397 }%
4398
4399 %

```

\Xdo@feet \Xdo@feet is the code extending \@makecol to cater for the extra critical feet.

```

4400 \newcommand*\Xdo@feet{%
4401   \setbox\@outputbox \vbox{%
4402     \unvbox\@outputbox
4403     \@opXfeet}}
4404 %

```

\@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.

```

4405 \newcommand\print@Xnotes[1]{%
4406   \csuse{#1footstart}{#1}%
4407   \csuse{#1footgroup}{#1}%
4408 }%
4409 %

```

We print all series of notes by looping on them. We check before printing them that they are not voided.

```

4410 \newcommand*\@opXfeet{%
4411   \unless\ifnocritical@%
4412     \gdef\firstXseries@{%
4413       \def\do##1{%
4414         \ifvoid\csuse{##1footins}\else%
4415           \global\skip\csuse{##1footins}=\csuse{Xbeforenotes@##1}%
4416           \global\advance\skip\csuse{##1footins} by\csuse{Xafterrule@##1}%
4417           \print@Xnotes{##1}%
4418         \fi%
4419       }%
4420       \dolistloop{\@series}%
4421     \fi%
4422   }%
4423 %

```

\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.

```

4424 \newcommand*\l@ddodoreinextrafeet{%
4425   \IfStrEq{familiar-critical}{\@fnpos}
4426   {\@doreinfeetX\X@doreinfeet}%
4427   {%
4428     \IfStrEq{critical-familiar}{\@fnpos}%
4429     {\X@doreinfeet\@doreinfeetX}%
4430     {\@doreinfeetX\X@doreinfeet}%

```



```

4431 }%
4432 }
4433
4434 %

```

\X@doreinfeet \X@doreinfeet is the code for catering for the extra critical footnotes within \@reinserts.

```

4435 \newcommand*{\X@doreinfeet}{%
4436   \unless\ifnocritical@%
4437   \def\do##1{%
4438     \ifvoid\csuse{##1footins}\else%
4439     \insert\csuse{##1footins}{\unvbox\csuse{##1footins}}%
4440     \fi}%
4441   \dolistloop{\@series}
4442   \fi%
4443 }
4444
4445 %

```

\print@notesX We have to add all the new kinds of familiar footnotes to the output routine. The normal way to add one series. **\do@feetX** \print@Xnotes is replaced by reledpar when using \Pages.

\@doreinfeetX

```

4446 \newcommand\print@notesX[1]{%
4447   \csuse{footstart#1}{#1}%
4448   \csuse{footgroup#1}{#1}%
4449 }%
4450 %

```

We print all the series of notes by looping on them. We check before printing them that they are not voided.

```

4451 \newcommand*{\do@feetX}{%
4452   \unless\ifnofamiliar@%
4453   \gdef\firstseriesX@{}%
4454   \setbox\@outputbox \vbox{%
4455     \unvbox\@outputbox%
4456     \def\do##1{%
4457       \ifvoid\csuse{footins##1}\else%
4458       \global\skip\csuse{footins##1}=\csuse{beforenotesX@##1}%
4459       \global\advance\skip\csuse{footins##1} by\csuse{afterruleX@##1}%
4460       \print@notesX{##1}%
4461       \fi%
4462     }%
4463     \dolistloop{\@series}}%
4464   \fi%
4465 }%
4466
4467 \newcommand{\@doreinfeetX}{%
4468   \unless\ifnofamiliar@%
4469   \def\do##1{%
4470     \ifvoid\csuse{footins##1}\else

```

```

4471     \insert%
4472         \csuse{footins##1}
4473         {\unvbox\csuse{footins##1}}%
4474     \fi%
4475 }%
4476 \dolistloop{\@series}%
4477 \fi%
4478 }%
4479
4480 %

```

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.)

```

4481 \@ifclassloaded{memoir}{%
4482 %

```

memoir is loaded so we use memoir's built in hooks.

```

4483 \g@addto@macro{\m@mdoextrafeet}{\l@ddoxtrafeet}%
4484 \g@addto@macro{\m@mdodoreinextrafeet}{\l@ddodoreinextrafeet}%
4485 }{%
4486 %

```

memoir has not been loaded, so patch `\@makecol` and `\@reinserts`.

```

4487 \@ifpackageloaded{fancyhdr}{%
4488 \patchcmd%
4489     {\latex@makecol}%
4490     {\xdef\@freelist{\@freelist\@midlist}}%
4491     {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
4492     {}%
4493     {\led@error@fail@patch@@@makecol}}%
4494 }{%
4495 \patchcmd%
4496     {\@makecol}%
4497     {\xdef\@freelist{\@freelist\@midlist}}%
4498     {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
4499     {}%
4500     {\led@error@fail@patch@@@makecol}}%
4501 }%
4502
4503 \patchcmd%
4504     {\@reinserts}%
4505     {\ifvbox}%
4506     {\l@ddodoreinextrafeet\ifvbox}%
4507     {}%

```

```

4508     {\led@error@fail@patch@@reinserts}%
4509 }
4510
4511 %

```

It turns out that `\@doclearpage` also needs modifying.

`\if@led@nofoot` We have to check if there are any leftover feet.

```

4512 \newif\if@led@nofoot
4513
4514 %

```

```

4515 \@ifclassloaded{memoir}{%
4516 %

```

If the memoir class is loaded we hook into its modified `\@doclearpage`.

```

\@mem@extranofeet 17 \g@addto@macro{\@mem@extranofeet}{%%
4518 \def\do#1{%
4519 \unless\ifnocritical@%
4520 \ifvoid\csuse{#1footins}\else\@mem@nofootfalse\fi%
4521 \fi%
4522 \unless\ifnofamiliar@%
4523 \ifvoid\csuse{footins#1}\else\@mem@nofootfalse\fi%
4524 \fi%
4525 }
4526 \dolistloop{\@series}%
4527 }%
4528 }{%
4529 %

```

As memoir is not loaded we have patch `\@doclearpage`.

```

\@led@testifnofoot 30 \newcommand*{\@led@testifnofoot}{%
\@doclearpage 31 \@led@nofoottrue%
4532 \ifvoid\footins\else%
4533 \@led@nofootfalse%
4534 \fi%
4535 \def\do##1{%
4536 \unless\ifnocritical@%
4537 \ifvoid\csuse{##1footins}\else%
4538 \@led@nofootfalse%
4539 \fi%
4540 \fi%
4541 \unless\ifnofamiliar@%
4542 \ifvoid\csuse{footins##1}\else%
4543 \@led@nofootfalse%
4544 \fi%
4545 \fi%

```

```

4546 }%
4547 \dolistloop{\@series}%
4548 }%
4549
4550 \pretocmd%
4551 {\@docclearpage}%
4552 {\@led@testifnofoot}%
4553 {}%
4554 {\led@error@fail@patch@@docclearpage}%
4555
4556 \patchcmd%
4557 {\@docclearpage}%
4558 {\ifvoid\footins}%
4559 {\if@led@nofoot}%
4560 {}%
4561 {\led@error@fail@patch@@docclearpage}%
4562
4563 }
4564
4565 %

```

XXIII Cross referencing

You can mark a place in the text using a command of the form `\edlabel{<foo>}`, and later refer to it using the label `<foo>` by saying `\edpageref{<foo>}`, or `\lineref{<foo>}` or `\sublineref{<foo>}` or `\pstartref`. These reference commands will produce, respectively, the page, line sub-line and pstart on which the `\edlabel{<foo>}` command occurred.

The reference macros warn you if a reference is made to an undefined label. If `{<foo>}` has been used as a label before, the `\edlabel{<foo>}` command will issue a complaint; subsequent `\edpageref` and `\edlineref` commands will refer to the latest occurrence of `\edlabel{<foo>}`.

\labelref@list Set up a new list, `\labelref@list`, to hold the page, line and sub-line numbers for each label.

```

4566 \list@create{\labelref@list}
4567 %

```

\zz@@@ A convenience macro to zero two labeling counters in one go.

```

4568 \newcommand*{\zz@@@}{000|000} % set two counters to zero in one go
4569
4570 %

```

\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.³¹

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 Billett³² found that the original code could be off by several pages. This version, hopefully cures that, and also allows for non-arabic page numbering.

```

4571 \newcommand*{\edlabel}[1]{%
4572   \ifl@dpairing\ifautopar%
4573     \strut%
4574     \fi\fi%
4575     \@bsphack%
4576     \ifledRcol%
4577       \write\linenum@out{\string\@lab}%
4578       \ifx\labelref@listR\empty%
4579         \xdef\label@refs{\zz@@@}%
4580       \else%
4581         \gl@p\labelref@listR\to\label@refs%
4582       \fi%
4583       \ifvmode%
4584         \advancelabel@refs%
4585       \fi%
4586   }%

```

Use code from the kernel `\label` command to write the correct page number. Also define an `hypertarget` if `hyperref` package is loaded.

```

4587   \protected@write\@auxout{%
4588     {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstartR
4589     |{#1}}}%
4590   \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1}}}{}%
4591   \else%
4592     \write\linenum@out{\string\@lab}%
4593     \ifx\labelref@listR\empty%
4594       \xdef\label@refs{\zz@@@}%
4595     \else%
4596       \gl@p\labelref@listR\to\label@refs%
4597     \fi%
4598     \ifvmode%
4599       \advancelabel@refs%
4600     \fi%
4601   \protected@write\@auxout{%
4602     {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstartR|{#1}}}%
4603     \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1}}}{}%
4604   \fi%
4605   \@esphack}%

```

³¹The remaining macros in this section were kindly revised by Wayne Sullivan, who substantially improved their efficiency and flexibility.

³²(jdb43@cam.ac.uk) via the `ctt` thread ‘`ledmac cross referencing`’, 25 August 2003.

```

4605
4606 %

\advancelabel@refs In cases where \edlabel is the first element in a paragraph, we have a problem with
\labelrefsparseline line counts, because line counts change only at the first horizontal box of the paragraph.
\labelrefsparsesubline Hence, we need to test \edlabel if it occurs at the start of a paragraph. To do so, we
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.

4607 \newcounter{line}%
4608 \newcounter{subline}%
4609 \newcommand{\advancelabel@refs}{%
4610     \setcounter{line}{\expandafter\labelrefsparseline\label@refs}%
4611     \stepcounter{line}%
4612     \ifsublines%
4613         \setcounter{subline}{\expandafter\labelrefsparsesubline\label@refs}%
4614     %
4615     \stepcounter{subline}{1}%
4616     \def\label@refs{\theline|\thesubline}%
4617 \else%
4618     \def\label@refs{\theline|0}%
4619 \fi%
4620 }
4621 \def\labelrefsparseline#1|#2{#1}
4622 \def\labelrefsparsesubline#1|#2{#2}
4623 %

```

\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.

```

4623 \newcommand*{\l@dmake@labels}{%
4624 \def\l@dmake@labels#1|#2|#3|#4|#5{%
4625     \expandafter\ifx\csname the@label#5\endcsname \relax\else
4626     \led@warn@DuplicateLabel{#5}%
4627 \fi
4628     \expandafter\gdef\csname the@label#5\endcsname{#1|#2|#3|#4}%
4629 \ignorespaces}
4630
4631 %

```

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.

```

4632 \AtBeginDocument{%
4633   \def\l@make@labels#1|#2|#3|#4|#5{%
4634 }
4635
4636 %

```

\@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.

```

4637
4638 \newcommand*\@lab{%
4639   \ifledRcol
4640     \xright@appenditem{\linenumr@p{\line@numR}}|{%
4641       \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
4642     \to\labelref@listR
4643   \else
4644     \xright@appenditem{\linenumr@p{\line@num}}|{%
4645       \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
4646     \to\labelref@list
4647   \fi}
4648 %

```

\applabel \applabel, if called in \edtext will insert automatically both a start and an end label for the current edtext lines.

```

4649 \newcommand*\applabel}[1]{%
4650   \ifnum\@edtext@level>0%
4651   %

```

Label should not be already defined.

```

4652     \ifcsundef{the@label#1}{%
4653       \csdef{the@label#1}{\applabel}%
4654     }%
4655     {%
4656       \led@warn@DuplicateLabel{#1 (\applabel)}%
4657     }%
4658 %

```

Parse the \edtext line numbers.

```

4659   \expandafter\l@dp@rsefootspec\l@d@nums|{%
4660   %

```

Use the TeX standard hack for label.

```

4661 \@bsphack%
4662 %

```

And now, write the data in the auxiliary file.

```

4663 \ifledRcol%
4664 \protected@write\@auxout{}%
4665 {\string\l@dmake@labelsR\space\l@dparsedstartpage|\
4666 \l@dparsedstartline|\l@dparsedstartsub|\the\c@pstartR|{#1:start}}%
4667 \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1:start}}}%
4668 \protected@write\@auxout{}%
4669 {\string\l@dmake@labelsR\space\l@dparsedendpage|\l@dparsedendline
4670 |\l@dparsedendsub|\the\c@pstartR|{#1:end}}%
4671 \else%
4672 \protected@write\@auxout{}%
4673 {\string\l@dmake@labelsR\space\l@dparsedstartpage|\
4674 \l@dparsedstartline|\l@dparsedstartsub|\the\c@pstartR|{#1:start}}%
4675 \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1:start}}}%
4676 \protected@write\@auxout{}%
4677 {\string\l@dmake@labelsR\space\l@dparsedendpage|\l@dparsedendline
4678 |\l@dparsedendsub|\the\c@pstartR|{#1:end}}%
4679 \fi%
4680 %

```

Use the \LaTeX standard hack for label.

```

4677 \@esphack%
4678 %

```

Warning if `\applabel` is called outside of `\edtext`.

```

4679 \else%
4680 \led@warn@AppLabelOutEdtext{#1}%
4681 \fi%
4682 %

```

End of `\applabel`

```

4683 }%
4684 %

```

`\wrap@edcrossref` `\wrap@edcrossref` is called around all `reledmac` crossref commands, except those which start with `x`. It adds the hyperlink.

```

4685 \newrobustcmd{\wrap@edcrossref}[2]{%
4686 \ifdef{\hyperlink}%
4687 {\hyperlink{#1}{#2}}%
4688 {#2}%
4689 }
4690 %

```

`\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`.

```
4691 \newcommand*\edpageref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{1}{#1}}}%
4692 \newcommand*\xpageref}[1]{\l@dgetref@num{1}{#1}}%
4693
4694 %
```

`\edlineref` If the specified label exists, `\lineref` gives its line number.

```
\xlineref
4695 \newcommand*\edlineref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{2}{#1}}}%
4696 \newcommand*\xlineref}[1]{\l@dgetref@num{2}{#1}}%
4697
4698 %
```

`\sublineref` If the specified label exists, `\sublineref` gives its sub-line number.

```
\xsublineref
4699 \newcommand*\sublineref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{3}{#1}}}%
4700 \newcommand*\xsublineref}[1]{\l@dgetref@num{3}{#1}}%
4701
4702 %
```

`\pstarteref` If the specified label exists, `\pstartref` gives its pstart number.

```
\xpstartref
4703 \newcommand*\pstartref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{4}{#1}}}%
4704 \newcommand*\xpstartref}[1]{\l@dgetref@num{4}{#1}}%
4705
4706 %
```

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.

```
4707 \newcommand*\l@dref@undefined}[1]{%
4708 \expandafter\ifx\csname the@label#1\endcsname\relax
4709 \led@warn@RefUndefined{#1}%
```

```

4710 \fi}
4711
4712 %

```

\l@getref@num Next, `\l@getref@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@dlab@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`.

```

4713 \newcommand*{\l@getref@num}[2]{%
4714   \expandafter
4715   \ifx\csname the@label#2\endcsname \relax
4716     000%
4717   \else
4718     \expandafter\expandafter\expandafter
4719     \l@dlab@parse\csname the@label#2\endcsname|#1%
4720   \fi}
4721
4722 %

```

\l@dlab@parse Notice that we slipped another `|` delimiter into the penultimate line of `\l@getref@num`, to keep the ‘switch-number’ separate from the reference numbers. This `|` is used as another parameter delimiter by `\l@dlab@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@getref@num`.)

```

4723 \newcommand*{\l@dlab@parse}{%
4724   \def\l@dlab@parse#1|#2|#3|#4|#5{%
4725     \ifcase #5%
4726       \or #1%
4727       \or #2%
4728       \or #3%
4729       \or #4%
4730     \fi}
4731 %

```

\xxref The `\xxref` command takes two arguments, both of which are labels, e.g., `\xxref{mouse}{elephant}`. It first does some checking to make sure that the labels do exist (if one does not, those numbers are set to zero). Then it calls `\linenum` and sets the beginning page, line, and sub-line numbers to those of the place where `\label{mouse}` was placed, and the ending numbers to those at `{elephant}`. The point of this is to be able to manufacture footnote line references to passages which can not be specified in the normal way as the first argument to `\edtext` for one reason or another. Using `\xxref` in the second argument of `\edtext` lets you set things up at least semi-automatically.

```

4732 \newcommand*{\xxref}[2]{%
4733   {%
4734     \expandafter\ifx\csname the@label#1\endcsname \relax%
4735     \expandafter\let\csname the@@label#1\endcsname\zz@@@%
4736     \else%
4737       \expandafter\def\csname the@@label#1\endcsname{\l@dgetref@num
4738         {1}{#1}|\l@dgetref@num{2}{#1}|\l@dgetref@num{3}{#1}}%
4739       \fi%
4740       \expandafter\ifx\csname the@label#2\endcsname \relax%
4741       \expandafter\let\csname the@@label#2\endcsname\zz@@@%
4742       \else%
4743         \expandafter\def\csname the@@label#2\endcsname{\l@dgetref@num
4744           {1}{#2}|\l@dgetref@num{2}{#2}|\l@dgetref@num{3}{#2}}%
4745         \fi%
4746         \ifdefined\@Rlineflag%
4747         \StrDel{\csuse{the@@label#1}}{\@Rlineflag}[\@tempa]%
4748         \StrDel{\csuse{the@@label#2}}{\@Rlineflag}[\@tempb]%
4749         \else%
4750         \letcs{\@tempa}{the@@label#1}%
4751         \letcs{\@tempb}{the@@label#2}%
4752         \fi%
4753         \linenum{\@tempa}%
4754         \@tempb}}}%
4755 %

```

\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.

```

4755 \xdef\Xtwolines@appref{}%
4756 \xdef\Xmorethantwolines@appref{}%
4757 \newtoggle{Xtwolinesbutnotmore@appref}%
4758 \newtoggle{Xtwolinesonlyinsamepage@appref}%
4759
4760 \xdef\Xendtwolines@apprefwithpage{}%
4761 \xdef\Xendmorethantwolines@apprefwithpage{}%
4762 \newtoggle{Xendtwolinesbutnotmore@apprefwithpage}%
4763 \newtoggle{Xendtwolinesonlyinsamepage@apprefwithpage}%
4764
4765 %

```

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.

```

4766
4767 \xdef\Xboxstartlinenum@appref{Opt}
4768 \xdef\Xboxendlinenum@appref{Opt}
4769
4770 \xdef\Xendboxstartlinenum@apprefwithpage{Opt}
4771 \xdef\Xendboxendlinenum@apprefwithpage{Opt}
4772
4773 %

```

Now, declare the default value of `\@apprefprefixsingle` and `\@apprefprefixmore`, and the commands which defines them

```

4774 \newcommand\@apprefprefixsingle{}%
4775 \newcommand\@apprefprefixmore{}%
4776
4777 \newcommand{\apprefprefixsingle}[1]{%
4778   \gdef\@apprefprefixsingle{#1}%
4779 }
4780
4781 \newcommand{\setapprefprefixmore}[1]{%
4782   \gdef\@apprefprefixmore{#1}%
4783 }
4784
4785 %

```

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.

```

4786 \newcommandx{\appref}[2][1,usedefault]{%
4787   \IfStrEq{#1}{fulllines}%
4788     {\toggletrue{fulllines@}}%
4789     {}%
4790   \xdef\@currentseries{appref}%
4791   \ifdefempty{\@apprefprefixmore}%
4792     {\@apprefprefixsingle}%
4793     {%
4794       \IfEq{\xlineref{#2:start}}{\xlineref{#2:end}}%
4795       {\@apprefprefixsingle}%
4796       {\@apprefprefixmore}%
4797     }%
4798   \printlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:start}
4799   }|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}||%
4800   \togglefalse{fulllines@}%
4801 }%
4802
4803 % \changes{v1.23.0}{2015/05/18}{Debug \protect\cs{Xendtwolines}, \protect\
cs{Xendmorethantwolines}, \protect\cs{Xendtwolinesbutnotmore} and \protect\

```

```

cs{Xendtwolinesonlyinsamepage} when using \protect\cs{apprefwithpage}.}
4803 \newcommand{\apprefwithpage}[2][1,usedefault]{%
4804   \IfStrEq{#1}{fulllines}%
4805     {\toggletrue{fulllines@}}%
4806     {}%
4807   \xdef\@currentseries{apprefwithpage}%
4808   \printendlines\pageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:
start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}||%
4809   \togglefalse{fulllines@}%
4810 }%
4811 %

```

\edmakelabel Sometimes the `\edlabel` command cannot be used to specify exactly the page and line desired; you can use the `\edmakelabel` macro make your own label. For example, if you say `\edmakelabel{elephant}{10|25|0}` you will have created a new label, and a later call to `\edpageref{elephant}` would print ‘10’ and `\lineref{elephant}` would print ‘25’. The sub-line number here is zero. `\edmakelabel` takes a label, followed by a page and a line number(s) as arguments. \TeX defines a `\makelabel` macro which is used in lists. Peter Wilson has changed the name to `\edmakelabel`.

```

4812 \newcommand*{\edmakelabel}[2]{\expandafter\xdef\csname the@label#1\
endcsname{#2}}
4813
4814 %

```

(If you are only going to refer to such a label using `\xxref`, then you can omit entries in the same way as with `\linenum` (see VI.3 p. 110 and V.9 p. 81), since `\xxref` makes a call to `\linenum` in order to do its work.)

XXIV Side notes

Regular `\marginpars` do not work inside numbered text — they do not produce any note but do put an extra unnumbered blank line into the text.

\@xympar Changing `\@xympar` a little at least ensures that `\marginpars` in numbered text do not disturb the flow.

```

4815 \pretocmd{\@xympar}%
4816   {\ifnumberedpar@
4817     \led@warn@NoMarginpars
4818     \@esphack
4819   \else}%
4820 {}%
4821 {}%
4822
4823 \apptocmd{\@xympar}%
4824   {\fi}%
4825 {}
4826 {}

```

```
4827
4828 %
```

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

```
4829 \newcount\sidenote@margin
4830 \newcommand*{\sidenotemargin}[1]{\%
4831   \l@dgetsidenote@margin{#1}%
4832   \ifnum\@l@tempcntb>\m@ne
4833     \ifl@Rcol
4834       \global\sidenote@marginR=\@l@tempcntb
4835     \else
4836       \global\sidenote@margin=\@l@tempcntb
4837     \fi
4838   \fi}}
4839 \newcommand*{\l@dgetsidenote@margin}[1]{\%
4840   \def\@tempa{#1}\def\@tempb{left}%
4841   \ifx\@tempa\@tempb
4842     \@l@tempcntb \z@
4843   \else
4844     \def\@tempb{right}%
4845     \ifx\@tempa\@tempb
4846       \@l@tempcntb \@ne
4847     \else
4848       \def\@tempb{outer}%
4849       \ifx\@tempa\@tempb
4850         \@l@tempcntb \tw@
4851       \else
4852         \def\@tempb{inner}%
4853         \ifx\@tempa\@tempb
4854           \@l@tempcntb \thr@@
4855         \else
4856           \led@warn@BadSidenotemargin
4857           \@l@tempcntb \m@ne
4858         \fi
4859       \fi
4860     \fi
```

```

4861 \fi}
4862 \sidenotemargin{right}
4863
4864 %

```

`\l@dlp@rbox` We need two boxes to store sidenote texts.

```

\l@drp@rbox
4865 \newbox\l@dlp@rbox
4866 \newbox\l@drp@rbox
4867
4868 %

```

`\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
4869 \newdimen\ledlsnotewidth \ledlsnotewidth=\marginparwidth
\ledrsnotesep
4870 \newdimen\ledrsnotewidth \ledrsnotewidth=\marginparwidth
\ledlsnotefontsetup
4871 \newdimen\ledlsnotesep \ledlsnotesep=\linenumsep
\ledrsnotefontsetup
4872 \newdimen\ledrsnotesep \ledrsnotesep=\linenumsep
4873 \newcommand*\ledlsnotefontsetup{\raggedleft\footnotesize}
4874 \newcommand*\ledrsnotefontsetup{\raggedright\footnotesize}
4875
4876 %

```

`\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
4877 \newcommand*\ledleftnote[1]{\edtext{}\l@dlsnote{#1}}
4878 \newcommand*\ledrightnote[1]{\edtext{}\l@drsnote{#1}}
4879
4880 \newcommand*\ledinnernote[1]{%
4881 \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
4882 \ledleftnote{#1}%
4883 \else%
4884 \ledrightnote{#1}%
4885 \fi%
4886 }
4887
4888 \newcommand*\ledouternote[1]{%
4889 \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
4890 \ledrightnote{#1}%
4891 \else%
4892 \ledleftnote{#1}%
4893 \fi%
4894 }
4895

```

```

4896 \newcommand*{\ledsidenote}[1]{\edtext{}{\l@dcnote{#1}}}
4897 %

```

`\l@dlsnote` . The ‘footnotes’ for left, right, and moveable sidenotes. The whole scheme is reminiscent of the critical footnotes code.

```

\l@dcnote
4898 \newif\ifrighnoteup
4899 \righnoteuptrue
4900
4901 \newcommand*{\l@dlsnote}[1]{%
4902 \begingroup%
4903 \newcommand{\content}{#1}%
4904 \ifnumberedpar@
4905 \ifledRcol%
4906 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}{%
4907 \to\inserts@listR
4908 \global\advance\insert@countR \@ne%
4909 \else%
4910 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}{%
4911 \to\inserts@list
4912 \global\advance\insert@count \@ne%
4913 \fi
4914 \fi\ignorespaces\endgroup}
4915
4916 \newcommand*{\l@drsnote}[1]{%
4917 \begingroup%
4918 \newcommand{\content}{#1}%
4919 \ifnumberedpar@
4920 \ifledRcol%
4921 \xright@appenditem{\noexpand\l@drsnote{\expandonce\content}}{%
4922 \to\inserts@listR
4923 \global\advance\insert@countR \@ne%
4924 \else%
4925 \xright@appenditem{\noexpand\l@drsnote{\expandonce\content}}{%
4926 \to\inserts@list
4927 \global\advance\insert@count \@ne%
4928 \fi
4929 \fi\ignorespaces\endgroup}
4930
4931 \newcommand*{\l@dcnote}[1]{%
4932 \begingroup%
4933 \newcommand{\content}{#1}%
4934 \ifnumberedpar@
4935 \ifledRcol%
4936 \xright@appenditem{\noexpand\l@dcnote{\expandonce\content}}{%
4937 \to\inserts@listR
4938 \global\advance\insert@countR \@ne%
4939 \else%
4940 \xright@appenditem{\noexpand\l@dcnote{\expandonce\content}}{%
4941 \to\inserts@list

```



```

4942 \global\advance\insert@count \@ne%
4943 \fi
4944 \fi\ignorespaces\endgroup}
4945
4946 %

```

\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.

```

4947 \newcommand*{\vl@dlsnote}[1]{%
4948 \ifledRcol{%
4949 \@l@tempcntb=\sidenote@marginR%
4950 \ifnum\@l@tempcntb>\@ne%
4951 \advance\@l@tempcntb by\page@numR%
4952 \fi%
4953 \else%
4954 \@l@tempcntb=\sidenote@margin%
4955 \ifnum\@l@tempcntb>\@ne%
4956 \advance\@l@tempcntb by\page@num%
4957 \fi%
4958 \fi%
4959 \ifodd\@l@tempcntb%
4960 \listgadd{\l@dcsnotetext@l}{#1}%
4961 \else%
4962 \listgadd{\l@dcsnotetext}{#1}%
4963 \fi
4964 }
4965 \newcommand*{\vl@drsnote}[1]{%
4966 \ifledRcol{%
4967 \@l@tempcntb=\sidenote@marginR%
4968 \ifnum\@l@tempcntb>\@ne%
4969 \advance\@l@tempcntb by\page@numR%
4970 \fi%
4971 \else%

```

```

4972 \l@dttempcntb=\sidenote@margin%
4973 \ifnum\l@dttempcntb>\@ne%
4974 \advance\l@dttempcntb by\page@num%
4975 \fi%
4976 \fi%
4977 \ifodd\l@dttempcntb%
4978 \listgadd{\l@dcsnotetext}{#1}%
4979 \else%
4980 \listgadd{\l@dcsnotetext@r}{#1}%
4981 \fi%
4982 }
4983 \newcommand*{\vl@dcsnote}[1]{\listgadd{\l@dcsnotetext}{#1}}
4984
4985 %

```

\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.

```

4986 \newcommand*{\setl@dlp@rbox}[1]{%
4987 {\parindent\z@\hsize=\ledlsnotewidth\ledlsnotefontsetup
4988 \global\setbox\l@dlp@rbox
4989 \ifleftnoteup
4990 =\vbox to\z@{\vss #1}%
4991 \else
4992 =\vbox to 0.70\baselineskip{\strut#1\vss}%
4993 \fi}}
4994 \newcommand*{\setl@drp@rbox}[1]{%
4995 {\parindent\z@\hsize=\ledrsnotewidth\ledrsnotefontsetup
4996 \global\setbox\l@drp@rbox
4997 \ifrightrightnoteup
4998 =\vbox to\z@{\vss#1}%
4999 \else
5000 =\vbox to0.7\baselineskip{\strut#1\vss}%
5001 \fi}}
5002 \newif\ifleftnoteup
5003 \leftnoteuptrue
5004 %

```

\@sidenotesep This macro is used to separate sidenotes of the same line.

```

5005 \newcommand{\setsidenotesep}[1]{\gdef\@sidenotesep{#1}}
5006 \newcommand{\@sidenotesep}{, }
5007 %

```

\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.

```

5008 \newcommand*{\affixside@note}{%
5009   \def\sidenotecontent@{%
5010     \numgdef{\itemcount@}{0}%
5011     \def\do##1{%
5012       \ifnumequal{\itemcount@}{0}%
5013         {%
5014           \appto\sidenotecontent@{##1}}% Not print not separator before
the 1st note
5015         {\appto\sidenotecontent@{\@sidenotesep ##1}%
5016         }%
5017       \numgdef{\itemcount@}{\itemcount@+1}%
5018     }%
5019     \dolistloop{\l@dcsnotetext}%
5020     \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
5021   }%

```

And we do the same for left and right notes (not movable).

```

5022 \gdef\@templ@d{%
5023 \gdef\@templ@n{\l@dcsnotetext\l@dcsnotetext@l\l@dcsnotetext@r}%
5024 \ifx\@templ@d\@templ@n \else%
5025   \if@twocolumn%
5026     \if@firstcolumn%
5027       \setl@dlp@rbox{##1}{\sidenotecontent@}%
5028     \else%
5029       \setl@drp@rbox{\sidenotecontent@}%
5030     \fi%
5031   \else%
5032     \l@l@tempcntb=\sidenote@margin%
5033     \ifnum\l@l@tempcntb>\@ne%
5034       \advance\l@l@tempcntb by\page@num%
5035     \fi%
5036     \ifodd\l@l@tempcntb%
5037       \setl@drp@rbox{\sidenotecontent@}%
5038       \gdef\sidenotecontent@{%
5039         \numgdef{\itemcount@}{0}%
5040         \dolistloop{\l@dcsnotetext@l}%
5041         \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
5042         \setl@dlp@rbox{\sidenotecontent@}%
5043       \else%
5044         \setl@dlp@rbox{\sidenotecontent@}%
5045         \gdef\sidenotecontent@{%
5046           \numgdef{\itemcount@}{0}%
5047           \dolistloop{\l@dcsnotetext@r}%
5048           \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
5049           \setl@drp@rbox{\sidenotecontent@}%
5050         \fi%
5051       \fi%
5052     \fi%
5053   }
5054 }%

```

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.

```

5055 \ifnoledgroup@else%
5056 \newcommand*{\l@dfeetbeginmini}{\l@dedbeginmini\l@dfambeginmini}
5057 \newcommand*{\l@dfeetendmini}{%
5058   \IfStrEq{critical-familiar}{\@mpfnpos}%
5059   {\l@dedendmini\l@dfamendmini}%
5060   {%
5061     \IfStrEq{familiar-critical}{\@mpfnpos}%
5062     {\l@dfamendmini\l@dedendmini}%
5063     {\l@dedendmini\l@dfamendmini}%
5064   }%
5065 }%
5066 %

```

`\l@dedbeginmini` These handle the initiation and closure of critical footnotes in a minipage environment.

```

\l@dedendmini
5067 \newcommand*{\l@dedbeginmini}{%
5068   \unless\ifnocritical@%
5069   \def\do##1{\csletcs{v##1footnote}{mpv##1footnote}}%
5070   \dolistloop{\@series}%
5071   \fi%
5072 }
5073 \newcommand*{\l@dedendmini}{%
5074   \unless\ifnocritical@%
5075   \ifl@dpairing%
5076     \ifledRcol%
5077       \flush@notesR%
5078     \else%
5079       \flush@notes%
5080     \fi%
5081   \fi
5082   \def\do##1{%
5083     \ifvoid\csuse{mp##1footins}\else%
5084     \ifl@dpairing\ifparledgroup%
5085       \ifledRcol%
5086         \dingdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+
skip\@nameuse{mp##1footins}}%
5087       \else%
5088         \dingdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL
+ \skip\@nameuse{mp##1footins}}%
5089       \fi%

```

```

5090     \fi\fi%
5091     \csuse{mp##1footgroup}{##1}%
5092     \fi}%
5093     \dolistloop{\@series}%
5094     \fi%
5095 }%
5096 %
5097 %

```

`\l@dfambeginmini` These handle the initiation and closure of familiar footnotes in a minipage environment.

`\l@dfamendmini`

```

5098 \newcommand*\l@dfambeginmini{%
5099   \unless\ifnofamiliar%
5100     \def\do##1{\csletcs{vfootnote##1}{mpvfootnote##1}}%
5101     \dolistloop{\@series}%
5102     \fi%
5103 }%
5104
5105 \newcommand*\l@dfamendmini{%
5106   \unless\ifnofamiliar%
5107     \def\do##1{%
5108       \ifvoid\csuse{mpfootins##1}\else%
5109         \csuse{mpfootgroup##1}{##1}%
5110       \fi}%
5111     \dolistloop{\@series}%
5112     \fi%
5113 }%
5114 %

```

`\@iiiminipage` This is our extended form of the kernel `\@iiiminipage` defined in `ltboxes.dtx`.

```

5115 \patchcmd%
5116   {\@iiiminipage}%
5117   {\let\@footnotetext\@mpfootnotetext}%
5118   {\let\@footnotetext\@mpfootnotetext\l@dfeetbeginmini}%
5119   {}%
5120   {\led@error@fail@patch@\@iiiminipage}%
5121   %

```

`\endminipage` This is our extended form of the kernel `\endminipage` defined in `ltboxes.dtx`.

```

5122 \patchcmd%
5123   {\endminipage}%
5124   {\footnoterule}%
5125   {\footnoterule\l@advance@parledgroup@beforenormalnotes}%
5126   {}%
5127   {\led@error@fail@patch@\endminipage}%
5128
5129 \patchcmd%
5130   {\endminipage}%

```

```

5131 {\@minipagefalse}%
5132 {\l@dfeetendmini\@minipagefalse}%
5133 {}%
5134 {\led@error@fail@patch@endminipage}
5135
5136 %

```

`\l@dunboxmpfoot` `\@ldunboxmpfoot` insert normal footnotes for ledgroup.
`\advance@parledgroup@beforenormalnotes`

```

5137 \newcommand*{\l@dunboxmpfoot}{%
5138   \vskip\skip\@mpfootins
5139   \normalcolor
5140   \footnoterule
5141   \l@advance@parledgroup@beforenormalnotes
5142   \unvbox\@mpfootins%
5143 }
5144 %

```

When using parallel ledgroup, we need to store the vertical space added before footnote, in order to compensate them between left and right pages.

```

5145 \newcommand{\l@advance@parledgroup@beforenormalnotes}{%
5146   \ifparledgroup
5147     \ifl@dpairing
5148       \ifledRcol
5149         \dimgdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+\
skip\@mpfootins}
5150       \else
5151         \dimgdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL+\
skip\@mpfootins}
5152       \fi
5153     \fi
5154   \fi
5155 }
5156 %

```

`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.

```

5157 \newenvironment{ledgroup}{%
5158   \resetprevpage@num%
5159   \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@%
5160   \let\@footnotetext\@mpfootnotetext
5161   \l@dfeetbeginmini%
5162 }{%
5163   \par
5164   \unskip
5165   \ifvoid\@mpfootins\else
5166     \l@dunboxmpfoot

```

```

5167 \fi
5168 \l@dfeetendmini%
5169 }
5170
5171 %

```

```

\ledgroupsize \begin{ledgroupsize}[\langle pos \rangle]{\langle width \rangle}

```

This environment puts footnotes at the end, even if that happens to be in the middle of a page, or crossing a page boundary. It is a sort of unboxed, variable $\langle width \rangle$ minipage. The optional $\langle pos \rangle$ controls the sideways position of numbered text.

```

5172 \newenvironment{ledgroupsize}[2][l]{%
5173 %

```

Set the various text measures.

```

5174 \hsize #2\relax
5175 %% \textwidth #2\relax
5176 %% \columnwidth #2\relax
5177 %

```

Initialize fills for centering.

```

5178 \let\ledllfill\hfil
5179 \let\ledrlfill\hfil
5180 \def\@tempa{#1}\def\@tempb{1}%
5181 %

```

Left adjusted numbered lines

```

5182 \ifx\@tempa\@tempb
5183 \let\ledllfill\relax
5184 \else
5185 \def\@tempb{r}%
5186 \ifx\@tempa\@tempb
5187 %

```

Right adjusted numbered lines

```

5188 \let\ledrlfill\relax
5189 \fi
5190 \fi
5191 %

```

Set up the footnoting.

```

5192 \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
5193 \let\@footnotetext\@mpfootnotetext
5194 \l@dfeetbeginmini%
5195 }{
5196 \par
5197 \unskip
5198 \ifvoid\@mpfootins\else
5199 \l@dunboxmpfoot
5200 \fi

```

```

5201 \l@dfetendmini%
5202 }
5203
5204 %

```

Close the \ifnoledgroup@else.

```

5205 \fi%
5206 %

```

`\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`.

```

5207 \newif\ifledgroupnotesL@
5208 \newif\ifledgroupnotesR@
5209 %

```

XXVI Indexing

Here is some code for indexing using page and line numbers.

First, ensure that `imakeidx` or `indextools` is loaded *before* `eledmac`.

```

5210 \AtBeginDocument{%
5211   \unless\ifl@imakeidx%
5212     \@ifpackageloaded{imakeidx}{\led@error@ImakeidxAfterEledmac}{}%
5213   \fi%
5214   \unless\ifl@indextools%
5215     \@ifpackageloaded{indextools}{\led@error@indextoolsAfterEledmac}{}%
5216   \fi%
5217 }
5218 %

```

`\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
5219 \newcommand{\pagelinesep}{-}
5220 \newcommand{\edindexlab}{${}}
5221 \newcounter{labidx}
5222 \setcounter{labidx}{0}
5223
5224 %

```

`\doedindexlabel` This macro sets an `\edlabel`.

```

5225 \newcommand{\doedindexlabel}{%
5226   \stepcounter{labidx}%
5227   \edlabel{\edindexlab\thelabidx}%
5228 }
5229
5230 %

```


`\thepageline` This macro makes up the page/line number combo from the label/ref.

```

5231 \newcommand{\thepageline}{%
5232   \thepage%
5233   \pagelinesep%
5234   \xlineref{\edindexlab\thelabidx}%
5235 }
5236 %

```

`\thestartpageline` These macros make up the page/line start/end number when the `\edindex` command is called in critical notes.
`\theendpageline`

```

5237 \newcommand{\thestartpageline}{%
5238   \l@dparsedstartpage%
5239   \pagelinesep%
5240   \l@dparsedstartline%
5241 }
5242 \newcommand{\theendpageline}{%
5243   \l@dparsedendpage%
5244   \pagelinesep%
5245   \l@dparsedendline%
5246 }
5247 %

```

`\if@edindex@fornote@true` This boolean test is switching at the beginning of each critical note, to allow index referring to this note.

```

5248 \newif\if@edindex@fornote@
5249 %

```

`\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 |.

```

5250 \newcommand{\prepare@edindex@fornote}[1]{%
5251   \l@dp@rsefootspec#1|%
5252   \@edindex@fornote@true%
5253 }
5254 %

```

`\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.

```

5255 \newcommand{\get@edindex@ledinnote@command}{%
5256   \ifxindy@%
5257     \gdef\@ledinnote@command{%
5258       ledinnote\thelabidx%

```

```

5259 }%
5260 \ifxindyhyperref%
5261   \immediate\write\eledmac@xindy@out{%
5262     (define-attributes ("ledinnote\thelabidx"))^^J
5263     \space\space(markup-locref^^J
5264     \eledmacmarkuplocdepth^^J
5265     :open "\string\ledinnote[\edindexlab\thelabidx]{\@index@command
5266   }{"^^J
5267     :close "}"^^J
5268     :attr "ledinnote\thelabidx"^^J
5269   )
5270 }%
5271 \else%
5272   \immediate\write\eledmac@xindy@out{%
5273     (define-attributes ("ledinnote\thelabidx"))^^J
5274     \space\space(markup-locref^^J
5275     \eledmacmarkuplocdepth^^J
5276     :open "\string\ledinnote{\@index@command}{"}^^J
5277     :close "}"^^J
5278     :attr "ledinnote\thelabidx"^^J
5279   )
5280 }%
5281 \fi%
5282 %

```

If we do not use xindy option, \@ledinnote@command will produce something like ledinnote{formattingcommand}.

```

5282 \else%
5283   \gdef\@ledinnote@command{%
5284     ledinnote[\edindexlab\thelabidx]{\@index@command}%
5285   }%
5286 \fi%
5287 }
5288 %

```

\get@index@command This macro is used to analyse if a text to be indexed has a command after a |.

```

5289 \def\get@index@command#1|#2+{%
5290   \gdef\@index@txt{#1}%
5291   \gdef\@index@command{#2}%
5292   \xdef\@index@parenthesis{}%
5293   \IfBeginWith{\@index@command}{(}{%
5294     \StrGobbleLeft{\@index@command}{1}[\@index@command@]%
5295     \global\let\@index@command\@index@command@%
5296     \xdef\@index@parenthesis{(%
5297   }{}%
5298   \IfBeginWith{\@index@command}{)}{%
5299     \StrGobbleLeft{\@index@command}{1}[\@index@command@]%
5300     \global\let\@index@command\@index@command@%
5301     \xdef\@index@parenthesis{)%

```

```

5302 }{}%
5303 }
5304 %

```

\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

```

5305 \newcommand{\ledinnote}[3][1,usedefault]{%
5306   \ifboolexpr{%
5307     test{\ifdefequal{\iftrue}{\ifHy@hyperindex}}%
5308     or%
5309     bool {xindyhyperref@}%
5310   }%
5311   {%
5312     \csuse{#2}{\hyperlink{#1}{\ledinnotemark{#3}}}%
5313   }%
5314   {%
5315     \csuse{#2}{\ledinnotemark{#3}}%
5316   }%
5317 }%
5318 \newcommand{\ledinnothyperpage}[2]{\csuse{#1}{\ledinnotemark{\hyperpage
5319   {#2}}}%
5320 }%

```

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

```

5321 \newcommand{\@wredindex}[2][1=\expandonce\jobname,usedefault]{%#1 = the
5322   index name, #2 = the text
5323   \global\let\old@Rlineflag\@Rlineflag%
5324   \gdef\@Rlineflag{}%
5325   \ifl@imakeidx%
5326     \if@edindex@fornote@%
5327       \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
5328       \get@edindex@ledinnote@command%
5329       \expandafter\imki@wrindexentry{#1}{\@index@txt|(\@ledinnote@command
5330 }{\thestartpageline}%
5331       \expandafter\imki@wrindexentry{#1}{\@index@txt|)\@ledinnote@command
5332 }{\theendpageline}%

```

```

5330 \else%
5331 \get@edindex@hyperref{#2}%
5332 \imki@wrindexentry{#1}{\@index@txt\@edindex@hyperref}{\thepageline}%
5333 \fi%
5334 \else%
5335 \if@edindex@fornote@%
5336 \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
5337 \get@edindex@ledinnote@command%
5338 \expandafter\protected@write\@indexfile{%
5339 {\string\indexentry{\@index@txt|(\@ledinnote@command){\thestartpageline}
5340 }%
5341 \expandafter\protected@write\@indexfile{%
5342 {\string\indexentry{\@index@txt|)\@ledinnote@command}{\theendpageline}
5343 }%
5344 \else%
5345 \protected@write\@indexfile{%
5346 {\string\indexentry{#2}{\thepageline}
5347 }%
5348 \fi%
5349 \fi%
5350 \endgroup
5351 \global\let\@Rlineflag\old@Rlineflag%
5352 \@esphack%
5353 }
5354 %

```

Need to add the definition of `\edindex` to `\makeindex`, and initialise `\edindex` to do nothing.

```

5355 \pretocmd{\makeindex}{%
5356 \def\edindex{\@bsphack
5357 \doedindexlabel
5358 \begingroup
5359 \@sanitize
5360 \@wredindex}}{}{}
5361 \newcommand{\edindex}[1]{\@bsphack\@esphack}
5362 %

```

`\hyperlinkformat` `\hyperlinkformat` command is to be used to have both a internal hyperlink and a format, when indexing.

```

5363 \newcommand{\hyperlinkformat}[3]{%
5364 \ifstrempy{#1}%
5365 {\hyperlink{#2}{#3}}%
5366 {\csuse{#1}{\hyperlink{#2}{#3}}%
5367 }}
5368 %

```

`\hyperlinkR` `\hyperlinkR` command is to be used to create a internal hyperlink and `\ledRflag`, when indexing.

```

5369 \newcommand{\hyperlinkR}[2]{%
5370 \hyperlink{#1}{#2\@Rlineflag}%
5371 }%
5372
5373 %

```

\hyperlinkformatR \hyperlinkformatR command is to be used to create a internal hyperlink, a format and a \@Rlineflag, when indexing.

```

5374 \newcommand{\hyperlinkformatR}[3]{%
5375 \hyperlinkformat{#1}{#2}{#3\@Rlineflag}%
5376 }%
5377
5378 %

```

\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.

```

5379 \newcommand{\get@edindex@hyperref}[1]{%
5380 %

```

We have to disable temporary spaces to work through a xstring bug (or feature?)

```

5381 \edef\temp@{%
5382 \catcode`\ =9 %space need for catcode
5383 #1%
5384 \catcode`\ =10 % space need for catcode
5385 }%
5386 %

```

Now, we define \@edindex@hyperref if the hyperindex of hyperref is enabled.

```

5387 \ifdefequal{\iftrue}{\ifHy@hyperindex}{%
5388 \IfSubStr{\temp@}{|}%
5389 {\get@index@command#1+%
5390 \ifledRcol%
5391 \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
5392 hyperlinkformatR{\@index@command}%
5393 {\edindexlab\thelabidx}}%
5394 \else%
5395 \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
5396 hyperlinkformat{\@index@command}%
5397 {\edindexlab\thelabidx}}%
5398 \fi%
5399 }%
5400 {\get@index@command#1|+%
5401 \ifledRcol%
5402 \gdef\@edindex@hyperref{|\hyperlinkR{\edindexlab\thelabidx}}%
5403 \else%
5404 \gdef\@edindex@hyperref{|\hyperlink{\edindexlab\thelabidx}}%
5405 \fi%
5406 }%

```

```

5407 }%
5408 %
5409 % If we use both xindy and hyperref, first get the \protect\cs{
index@command} command.
5410 % Then define \protect\cs{@edindex@hyperref} in the form \verb+eledmacXXX+
5411 % \begin{macrocode}
5412 {\ifxindyhyperref%
5413 \IfSubStr{\temp@}{|}%
5414 {\get@index@command#1+}%
5415 {\get@index@command#1|+}%
5416 \gdef\@edindex@hyperref{|eledmac\thelabidx}%
5417 %

```

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.

```

5418 \IfStrEq{\@index@parenthesis}{(}%
5419 {%
5420 \csxdef{xindyparenthesis@\@index@txt}{\thelabidx}%
5421 \gdef\@edindex@hyperref{|(eledmac\thelabidx}%
5422 }%
5423 {}%
5424 %

```

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.

```

5425 \IfStrEq{\@index@parenthesis}{)}%
5426 {%
5427 \xdef\@edindex@hyperref{|)eledmac\csuse{xindyparenthesis@\@index@txt}}%
5428 \global\csundef{xindyparenthesis@\@index@txt}%
5429 }%
5430 %

```

Write in the .xdy file the attributes of the location.

```

5431 {%
5432 \immediate\write\eledmac@xindy@out{%
5433 (define-attributes ("eledmac\thelabidx"))^^J
5434 \space\space(markup-locref^^J
5435 \eledmacmarkuplocrefdepth^^J
5436 :open "\string\hyperlink%
5437 \ifledRcol R\fi%
5438 {\edindexlab\thelabidx}%
5439 {\ifdefempty{\@index@command}%
5440 }%
5441 {\@backslashchar\@index@command}%
5442 {"^^J

```

```

5443         :close "}}"^^J
5444         :attr "eledmac\thelabidx"^^J
5445     )
5446 }%
5447 }%
5448 %

```

And now, in any other case.

```

5449 \else%
5450     \gdef\@index@txt{#1}%
5451     \gdef\@edindex@hyperref{}%
5452 \fi%
5453 }%
5454 }
5455 %

```

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.

```

5456 \def\@hangingsymbol{}
5457 \newcommand*{\sethangingsymbol}[1]{%
5458     \gdef\@hangingsymbol{#1}%
5459 }%
5460 \newif\ifinstanza
5461 %

```

`\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.

```

5462 \newif\ifinserthangingsymbol
5463 \newcommand{\inserthangingsymbol}{%
5464     \ifinserthangingsymbol%
5465         \ifinstanza%

```

```

5466 \hangingsymbol%
5467 \fi%
5468 \fi%
5469 }
5470 %

```

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.

```

5471 \newcommand*{\ampersand}{\char`\&}
5472
5473 %

```

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.

```

5474 \chardef\body=\catcode`\@
5475 \catcode`\@=11
5476 \chardef\next=\catcode`\&
5477 \catcode`\&=\active
5478
5479 %

```

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.

```

5480 \newcount\stanza@count
5481 \newlength{\stanzaindentbase}
5482 \setlength{\stanzaindentbase}{20pt}
5483
5484 %

```

\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.


```

5485 \def\strip@szacnt#1,#2|{\def\@tempb{#1}\def\@tempa{#2|}}
5486 \newcommand*{\setstanzavalues}[2]{\def\@tempa{#2,,|}%
5487   \stanza@count\z@
5488   \def\next{\expandafter\strip@szacnt\@tempa
5489     \ifx\@tempb\empty\let\next\relax\else
5490     \expandafter\mathchardef\csname #1@number\stanza@count
5491     \@endcsname\@tempb\relax
5492     \advance\stanza@count\@ne\fi\next}%
5493   \next}
5494
5495 %

```

\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).

```

5496 \newcommand*{\setstanzaindents}[1]{\setstanzavalues{sza}{#1}}
5497 \newcommand*{\setstanzapenalties}[1]{\setstanzavalues{szp}{#1}}
5498 %
5499 %

```

\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.

```

5500 \newcounter{stanzaindentsrepetition}
5501 \newcount\stanza@modulo
5502
5503 \newcommand*{\managestanza@modulo}[0]{
5504   \advance\stanza@modulo\@ne
5505   \ifnum\stanza@modulo>\value{stanzaindentsrepetition}
5506     \stanza@modulo\@ne
5507   \fi
5508 }
5509 %

```

\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.

```

5510 \newcommand{\stanzaindent}[1]{%
5511   \hspace{\dimexpr#1\stanzaindentbase-\parindent\relax}%
5512   \ignorespaces%
5513 }%
5514 \WithSuffix\newcommand\stanzaindent*[1]{%
5515   \stanzaindent{#1}%

```

```

5516 \global\advance\stanza@modulo-\@ne%
5517 \ifnum\stanza@modulo=0%
5518 \global\stanza@modulo=\value{stanzaindentrepetition}%
5519 \fi%
5520 \ignorespaces%
5521 }%
5522 %

```

XXVII.5 Numbering stanza

Here, macro for numbering stanza. First, the stanza counter.

```

\thestanza \newcounter{stanza}
5524 \renewcommand{\thestanza}{%
5525 \textbf{\arabic{stanza}}%
5526 }
5527 %

```

`\ifnumberstanza` Then, macro to activate automatically numbering of stanza.

```

5528 \newif\ifnumberstanza%
5529 %

```

`\@insertstanzanumber` Now, macro called at the first line of verse of a stanza.

```

5530 \newcommand{\@insertstanzanumber}[0]{%
5531 \ifnumberstanza%
5532 \ifl@pairing%
5533 \ifledRcol%
5534 \stanzanumwrapper{\thestanzaR}%
5535 \else%
5536 \stanzanumwrapper{\thestanzaL}%
5537 \fi%
5538 \else%
5539 \stanzanumwrapper{\thestanza}%
5540 \fi%
5541 \setline{1}%
5542 \fi%
5543 }%
5544 %

```

`\@advancestanzanumber` Also a command to advance the counter of stanza.

```

5545 \newcommand{\@advancestanzanumber}[0]{%
5546 \ifnumberstanza%
5547 \ifl@pairing%
5548 \ifledRcol%
5549 \addtocounter{stanzaR}{1}%
5550 \else%

```

```

5551 \addtocounter{stanzaL}{1}%
5552 \fi%
5553 \else%
5554 \addtocounter{stanza}{1}%
5555 \fi%
5556 \fi%
5557 }%
5558 %

```

`\stanzanumwrapper` And finally, the wrapper for stanza number

```

5559 \newcommand{\stanzanumwrapper}[1]{%
5560 \flagstanza{#1}%
5561 }%
5562 %

```

XXVII.6 Stanza number in note

Here, the command called when printing stanza number in notes.

```

5563 \newcommand{\printstanza}[0]{%
5564 \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
5565 l@dprintingcolumns}}{%
5566 \ifledRcol{%
5567 \thestanzaR%
5568 \else%
5569 \thestanzaL%
5570 \fi%
5571 }{%
5572 \thestanza%
5573 }%
5574 }%

```

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.

```

5575 \newcommandx{\stanza@line}[1][1]{
5576 \ifnum\value{stanzaindentrepetition}=0
5577 \parindent=\csname sza@number\stanza@count
5578 @\endcsname\stanzaindentbase

```

```

5579 \else
5580 \parindent=\csname sza@\number\stanza@modulo
5581 @\endcsname\stanzaindentbase
5582 \managestanza@modulo
5583 \fi
5584 \pstart[#1]\stanza@hang\ignorespaces}
5585 \xdef\stanza@hang{\noexpand\leavevmode\noexpand\startlock
5586 \hangindent\expandafter
5587 \noexpand\csname sza@0@\endcsname\stanzaindentbase
5588 \hangafter\@ne}
5589 \def\sza@penalty{\count@\csname szp@\number\stanza@count @\endcsname
5590 \ifnum\count@>\@M\advance\count@-\@M\penalty-\else
5591 \penalty\fi\count@}
5592 %

```

`\@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 `\&`.

```

5593 \xdef\@startstanza[#1]{%
5594 \noexpand\instanzatrue\expandafter
5595 \begingroup%
5596 \catcode`\noexpand\&\active%
5597 \global\stanza@count\@ne\stanza@modulo\@ne
5598 \noexpand\ifnum\expandafter\noexpand
5599 \csname sza@0@\endcsname=\z@\let\noexpand\stanza@hang\relax
5600 \let\noexpand\endlock\relax\noexpand\else\interlinepenalty
5601 \@M\rightskip\z@ plus 1fil\relax\noexpand\fi\noexpand\ifnum
5602 \expandafter\noexpand\csname szp@0@\endcsname=\z@
5603 \let\noexpand\sza@penalty\relax\noexpand\fi%
5604 \def\noexpand&{%
5605 \noexpand\newverse[] []}%
5606 \def\noexpand\&{\noexpand\@stopstanza}%
5607 \noexpand\@advancestanza@number%
5608 \noexpand\stanza@line[#1]\noexpand\@insertstanza@number}
5609
5610 \newcommandx{\stanza}[1][1,usedefault]{\@startstanza[#1]}
5611
5612 \newcommandx{\@stopstanza}[1][1,usedefault]{%
5613 \unskip%
5614 \endlock%
5615 \pend[#1]%
5616 \endgroup%
5617 \instanzafalse%

```

```

5618 }
5619
5620 \newcommand*{\newverse}[2][1,2,usedefault]{%
5621   \unskip%
5622   \endlock\pend[#1]\sza@penalty\global%
5623   \advance\stanza@count\@ne\stanza@line[#2]%
5624 }
5625
5626 %

```

\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`.

```

5627 \newcommand*{\flagstanza}[2][\stanzaindentbase]{%
5628   \hskip -#1\llap{#2}\hskip #1\ignorespaces}
5629
5630 %

```

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.

```

5631 \catcode`\&=\next
5632 \catcode`\@=\body
5633 \setstanzavalues{szp}{0}
5634
5635 %

```

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.

```

5636 \newtoks\@emptytoks
5637
5638 %

```

The rest is from amsmath.

\l@denbody A token register to contain the body.

```
5639 \newtoks\l@denbody
5640
5641 %
```

\addtol@denbody \addtol@denbody{arg} adds arg to the token register \l@denbody.

```
5642 \newcommand{\addtol@denbody}[1]{%
5643   \global\l@denbody\expandafter{\the\l@denbody#1}}
5644
5645 %
```

\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.

```
5646 \newcommand{\l@dcollect@body}[1]{%
5647   \l@denbody{\expandafter#1\expandafter{\the\l@denbody}}%
5648   \edef\processl@denbody{\the\l@denbody\noexpand\end{\@currenvir}}%
5649   \l@denbody\emptytoks \def\l@dbegin@stack{b}%
5650   \begingroup
5651     \expandafter\let\csname\@currenvir\endcsname\l@dcollect@@body
5652     \edef\processl@denbody{\expandafter\noexpand\csname\@currenvir\endcsname}%
5653     \processl@denbody%
5654   }%
5655
5656 %
```

\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.

```
5657 \def\l@dpush@begins#1\begin#2{%
5658   \ifx\end#2\else b\expandafter\l@dpush@begins\fi}
5659
5660 %
```

\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.

```

5661 \def\l@dcollect@@body#1\end#2{%
5662   \edef\l@dbegin@stack{\l@dpush@begins#1\begin\end
5663     \expandafter\@gobble\l@dbegin@stack}%
5664   \ifx\@empty\l@dbegin@stack
5665     \endgroup
5666     \@checkend{#2}%
5667     \addtol@denvbody{#1}%
5668   \else
5669     \addtol@denvbody{#1\end{#2}}%
5670   \fi
5671   \processl@denvbody % A little tricky! Note the grouping
5672 }
5673
5674 %

```

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.

```

5675 \newcommand*{\l@dtabnoexpands}{%
5676   \let\rtab=0%
5677   \let\ctab=0%
5678   \let\ltab=0%
5679   \let\rtabtext=0%
5680   \let\ltabtext=0%
5681   \let\ctabtext=0%
5682   \let\edbeforetab=0%
5683   \let\edaftertab=0%
5684   \let\edatleft=0%
5685   \let\edatright=0%
5686   \let\edvertline=0%
5687   \let\edvertdots=0%
5688   \let\edrowfill=0%
5689 }
5690
5691 %

```

`\disable@familiarnotes` Macros to disable and restore familiar notes, to prevent them from printing multiple times in `edtabularx` and `edarrayx` environments.

```

5692 \newcommand{\disable@familiarnotes}{%
5693   \unless\ifnofamiliar%
5694     \def\do##1{%
5695       \csletcs{footnote@##1}{footnote##1}%

```

```

5696 \expandafter\renewcommand \csname footnote##1\endcsname[1]{%
5697 \protected@csxdef{@thefnmark##1}{\csuse{thefootnote##1}}%
5698 \csuse{@footnotemark##1}%
5699 }%
5700 }%
5701 \dolistloop{\@series}%
5702 \fi%
5703 }%
5704 \newcommand{\restore@familiarnotes}{%
5705 \unless\ifnofamiliar%
5706 \def\do##1{%
5707 \csletcs{footnote##1}{footnote@##1}%
5708 }%
5709 \dolistloop{\@series}%
5710 \fi%
5711 }%
5712 %
5713 %

```

`\disable@sidenotes` The same, for side notes.

```

\restore@sidenotes
5714 \newcommand{\disable@sidenotes}{%
5715 \let\@@ledrightnote\ledrightnote%
5716 \let\@@ledleftnote\ledleftnote%
5717 \let\@@ledsidenote\ledsidenote%
5718 \let\ledrightnote@gobble%
5719 \let\ledleftnote@gobble%
5720 \let\ledsidenote@gobble%
5721 }%
5722 \newcommand{\restore@sidenotes}{%
5723 \let\ledrightnote\@@ledrightnote%
5724 \let\ledleftnote\@@ledleftnote%
5725 \let\ledsidenote\@@ledsidenote%
5726 }%
5727 %

```

`\disable@notes` Disable/restore side and familiar notes.

```

\restore@notes
5728 \newcommand{\disable@notes}{%
5729 \disable@sidenotes%
5730 \disable@familiarnotes%
5731 }%
5732 \newcommand{\restore@notes}{%
5733 \restore@sidenotes%
5734 \restore@familiarnotes%
5735 }%
5736 %

```

`\EDTEXT` We need to be able to modify the `\edtext` macros and also restore their original definitions.
`\xedtext`

```

5737 \let\EDTEXT=\edtext
5738 \newcommand{\xedtext}[2]{\EDTEXT{#1}{#2}}
5739 %

```

\EDLABEL We need to be able to modify and restore the `\edlabel` macro.

```

\xedlabel
5740 \let\EDLABEL=\edlabel
5741 \newcommand*\xedlabel[1]{\EDLABEL{#1}}
5742 %

```

\EDINDEX Macros supporting modification and restoration of `\edindex`.

```

\xedindex
\nulledindex
5743 \let\EDINDEX=\edindex
5744 \newcommand{\xedindex}{\@bsphack%
5745   \ifnextchar [{\l@d@index}{\l@d@index[\jobname]}}
5746 \newcommand{\nulledindex}[2][\jobname]{\@bsphack\@esphack}
5747
5748 %

```

\@line@@num Macro supporting restoration of `\linenum`.

```

5749 \let\@line@@num=\linenum
5750 %

```

\l@dgobblearg `\l@dgobbleoptarg[⟨arg⟩]{⟨arg⟩}` replaces these two arguments (first is optional) by `\relax`.

```

5751 \newcommand*\l@dgobbleoptarg[2][ ]{\relax}%
5752
5753 %

```

\Relax `\let\Relax=\relax`

\NEXT `\let\NEXT=\next`

```

5756
5757 %

```

\l@dmodforedtext Modify and restore various macros for when `\edtext` is used.

```

\l@drestoreforedtext
5758 \newcommand{\l@dmodforedtext}{%
5759   \let\edtext\relax
5760   \def\do##1{\global\csletcs{##1footnote}{\l@dgobbleoptarg}}%
5761   \dolistloop{\@series}%
5762   \let\edindex\nulledindex
5763   \let\linenum\@gobble}
5764 \newcommand{\l@drestoreforedtext}{%
5765   \def\do##1{\global\csletcs{##1footnote}{##1@footnote}}
5766   \dolistloop{\@series}%
5767   \let\edindex\xedindex}
5768 %

```

`\l@dnnullfills` Nullify and restore some column fillers, etc.

```
\l@drestorefills
5769 \newcommand{\l@dnnullfills}{%
5770   \def\edlabel##1{%
5771     \def\edrowfill##1##2##3{%
5772     }
5773   \newcommand{\l@drestorefills}{%
5774     \def\edrowfill##1##2##3{\@EDROWFILL@{##1}{##2}{##3}}%
5775   }
5776
5777 %
```

`\letsforverteilen` Gathers some lets and other code that is common to the **verteilen** macros.

```
5778 \newcommand{\letsforverteilen}{%
5779   \let\edtext\xedtext
5780   \let\edindex\xedindex
5781   \def\do##1{\global\csletcs{##1footnote}{##1@@footnote}}
5782   \dolistloop{\@series}%
5783   \let\linenum\@line@num
5784   \hilfsskip=\l@dcwidth%
5785   \advance\hilfsskip by -\wd\hilfsbox
5786   \def\edlabel##1{\xedlabel{##1}}
5787
5788 %
```

`\disablel@dtabfeet` Declarations for using or using `\edtext` inside tabulars. The default at this point is for `\enablel@dtabfeet` `\edtext`.

```
5789 \newcommand\disablel@dtabfeet{\l@dmodforedtext}%
5790 \newcommand\enablel@dtabfeet{\l@drestoreforedtext}%
5791 %
```

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.

```
5792 \newcount\l@dampcount
5793 \l@dampcount=1\relax
5794 \newcount\l@dcolcount
5795 \l@dcolcount=0\relax
5796
5797 %
```

`\hilfsbox` Some (temporary) helper items.

```
\hilfsskip
5798 \newbox\hilfsbox
\Hilfsbox
5799 \newskip\hilfsskip
\hilfscount
5800 \newbox\Hilfsbox
```

```

5801 \newcount\hilfscount
5802
5803 %

```

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).

```

5804 \newdimen\dcoli
5805 \newdimen\dcolii
5806 \newdimen\dcoliii
5807 \newdimen\dcoliv
5808 \newdimen\dcolv
5809 \newdimen\dcolvi
5810 \newdimen\dcolvii
5811 \newdimen\dcolviii
5812 \newdimen\dcolix
5813 \newdimen\dcolx
5814 \newdimen\dcolxi
5815 \newdimen\dcolxii
5816 \newdimen\dcolxiii
5817 \newdimen\dcolxiv
5818 \newdimen\dcolxv
5819 \newdimen\dcolxvi
5820 \newdimen\dcolxvii
5821 \newdimen\dcolxviii
5822 \newdimen\dcolxix
5823 \newdimen\dcolxx
5824 \newdimen\dcolxxi
5825 \newdimen\dcolxxii
5826 \newdimen\dcolxxiii
5827 \newdimen\dcolxxiv
5828 \newdimen\dcolxxv
5829 \newdimen\dcolxxvi
5830 \newdimen\dcolxxvii
5831 \newdimen\dcolxxviii
5832 \newdimen\dcolxxix
5833 \newdimen\dcolxxx
5834 \newdimen\dcolerr % added for error handling
5835
5836 %

```

`\l@dcolwidth` This is a cunning way of storing the columnwidths indexed by the column number `\l@dcolcount`, like an array. (was `\Dimenzuordnung`)

```

5837 \newcommand{\l@dcolwidth}{\ifcase \the\l@dcolcount \dcoli %???
5838 \or \dcoli \or \dcolii \or \dcoliii
5839 \or \dcoliv \or \dcolv \or \dcolvi
5840 \or \dcolvii \or \dcolviii \or \dcolix \or \dcolx
5841 \or \dcolxi \or \dcolxii \or \dcolxiii
5842 \or \dcolxiv \or \dcolxv \or \dcolxvi

```

```

5843 \or \dcolxvii \or \dcolxviii \or \dcolxix \or \dcolxx
5844 \or \dcolxxi \or \dcolxxii \or \dcolxxiii
5845 \or \dcolxxiv \or \dcolxxv \or \dcolxxvi
5846 \or \dcolxxvii \or \dcolxxviii \or \dcolxxix \or \dcolxxx
5847 \else \dcolerr \fi}
5848
5849 %

```

\step1@dcolcount This increments the column counter, and issues an error message if it is too large.

```

5850 \newcommand*{\step1@dcolcount}{\advance\l@dcolcount\@ne
5851 \ifnum\l@dcolcount>30\relax
5852 \led@err@TooManyColumns
5853 \fi}
5854
5855 %

```

\l@dsetmaxcolwidth Sets the column width to the maximum value seen so far.

```

5856 \newcommand{\l@dsetmaxcolwidth}{%
5857 \ifdim\l@dcolwidth < \wd\hilfsbox
5858 \l@dcolwidth = \wd\hilfsbox
5859 \else \relax \fi}
5860
5861 %

```

\measuremcell Measure (recursively) the width required for a math cell.

```

5862 \def\measuremcell #1{%
5863 \ifx #1\ \ifnum\l@dcolcount=0\let\NEXT\relax%
5864 \else\l@dcheckcols%
5865 \l@dcolcount=0%
5866 \let\NEXT\measuremcell%
5867 \fi%
5868 \else\setbox\hilfsbox=\hbox{${\displaystyle{#1}}$}%
5869 \step1@dcolcount%
5870 \l@dsetmaxcolwidth%
5871 \let\NEXT\measuremcell%
5872 \fi\NEXT}
5873
5874 %

```

\measuretcell Measure (recursively) the width required for a text cell.

```

5875 \def\measuretcell #1{%
5876 \ifx #1\ \ifnum\l@dcolcount=0\let\NEXT\relax%
5877 \else\l@dcheckcols%
5878 \l@dcolcount=0%
5879 \let\NEXT\measuretcell%
5880 \fi%

```

```

5881 \else\setbox\hilfsbox=\hbox{#1}%
5882 \step1@dcolcount%
5883 \l@dsetmaxcolwidth%
5884 \let\NEXT\measuretcell%
5885 \fi\NEXT}
5886
5887 %

```

\measuremrow Measure (recursively) the width required for a math row.

```

5888 \def\measuremrow #1\{%
5889 \ifx #1&\let\NEXT\relax%
5890 \else\measuremcell #1&\&\&%
5891 \let\NEXT\measuremrow%
5892 \fi\NEXT}
5893 %

```

\measuretrow Measure (recursively) the width required for a text row.

```

5894 \def\measuretrow #1\{%
5895 \ifx #1&\let\NEXT\relax%
5896 \else\measuretcell #1&\&\&%
5897 \let\NEXT\measuretrow%
5898 \fi\NEXT}
5899
5900 %

```

\edtabcolsep The length `\edtabcolsep` controls the distance between columns.

```

5901 \newskip\edtabcolsep
5902 \global\edtabcolsep=10pt
5903
5904 %

```

\variab₀₅ \newcommand{\variab}{\relax}

```

5906
5907 %

```

\l@dcheckcols Check that the number of columns is consistent.

```

5908 \newcommand*{\l@dcheckcols}{%
5909 \ifnum\l@dcolcount=1\relax
5910 \else
5911 \ifnum\l@dampcount=1\relax
5912 \else
5913 \ifnum\l@dcolcount=\l@dampcount\relax
5914 \else
5915 \l@d@err@UnequalColumns

```

```

5916     \fi
5917   \fi
5918   \l@dampcount=\l@dcolcount
5919 \fi}
5920
5921 %

```

\edfilldimen A length.

```

5922 \newdimen\edfilldimen
5923 \edfilldimen=0pt
5924
5925 %

```

\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

```

5926 \newcounter{addcolcount}
5927 \renewcommand{\theadcolcount}{\roman{addcolcount}}
5928 %

```

XXVIII.2.3 Tabular typesetting

\setmcellright Typeset (recursively) cells of display math right justified.

```

5929 \def\setmcellright #1&{\def\edlabel##1{}}%
5930   \let\edindex\nulledindex
5931   \ifx #1\\ \ifnum\l@dcolcount=0%\removelastskip
5932     \let\Next\relax%
5933   \else\l@dcolcount=0%
5934     \let\Next=\setmcellright%
5935   \fi%
5936 \else%
5937   \disablel@dtabfeet%
5938   \stepl@dcolcount%
5939   \disable@notes%
5940   \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
5941   \restore@notes%
5942   \letsforverteilen%
5943   \hskip\hilfsskip$\displaystyle{#1}$%
5944   \hskip\edtabcolsep%
5945   \let\Next=\setmcellright%
5946 \fi\Next}
5947
5948 %

```

\settcellright Typeset (recursively) cells of text right justified.


```

5949 \def\settccllright #1{\def\edlabel##1{}}%
5950 \let\edindex\nulledindex
5951 \ifx #1\\ \ifnum\l@dcclcount=0%\removelastskip
5952 \let\Next\relax%
5953 \else\l@dcclcount=0%
5954 \let\Next=\settccllright%
5955 \fi%
5956 \else%
5957 \disablel@dtabfeet%
5958 \step1@dcclcount%
5959 \disable@notes%
5960 \setbox\hilfsbox=\hbox{#1}%
5961 \restore@notes%
5962 \letsforverteilen%
5963 \hskip\hilfsskip#1%
5964 \hskip\edtabcolsep%
5965 \let\Next=\settccllright%
5966 \fi\Next}
5967 %

```

\setmcellleft Typeset (recursively) cells of display math left justified.

```

5968 \def\setmcellleft #1{\def\edlabel##1{}}%
5969 \let\edindex\nulledindex
5970 \ifx #1\\ \ifnum\l@dcclcount=0 \let\Next\relax%
5971 \else\l@dcclcount=0%
5972 \let\Next=\setmcellleft%
5973 \fi%
5974 \else \disablel@dtabfeet%
5975 \step1@dcclcount%
5976 \disable@notes%
5977 \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
5978 \restore@notes%
5979 \letsforverteilen%
5980 $\displaystyle{#1}$\hskip\hilfsskip\hskip\edtabcolsep%
5981 \let\Next=\setmcellleft%
5982 \fi\Next}
5983 %
5984 %

```

\settcclleft Typeset (recursively) cells of text left justified.

```

5985 \def\settcclleft #1{\def\edlabel##1{}}%
5986 \let\edindex\nulledindex
5987 \ifx #1\\ \ifnum\l@dcclcount=0 \let\Next\relax%
5988 \else\l@dcclcount=0%
5989 \let\Next=\settcclleft%
5990 \fi%
5991 \else \disablel@dtabfeet%
5992 \step1@dcclcount%

```

```

5993         \disable@notes%
5994         \setbox\hilfsbox=\hbox{#1}%
5995         \restore@notes%
5996         \letsforverteilen%
5997         #1\hskip\hilfsskip\hskip\edtabcolsep%
5998         \let\Next=\settcclleft%
5999     \fi\Next}
6000 %

```

\setmcellcenter Typeset (recursively) cells of display math centered.

```

6001 \def\setmcellcenter #1{\def\edlabel##1{%
6002     \let\edindex\nulledindex
6003     \ifx #1\ \ifnum\l@dcolcount=0\let\Next\relax%
6004         \else\l@dcolcount=0%
6005             \let\Next=\setmcellcenter%
6006         \fi%
6007     \else \disablel@dtabfeet%
6008         \stepl@dcolcount%
6009         \disable@notes%
6010         \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6011         \restore@notes%
6012         \letsforverteilen%
6013         \hskip 0.5\hilfsskip$\displaystyle{#1}$\hskip0.5\hilfsskip%
6014         \hskip\edtabcolsep%
6015         \let\Next=\setmcellcenter%
6016     \fi\Next}
6017
6018 %

```

\settcclcenter Typeset (recursively) cells of text centered.

```

6019 \def\settcclcenter #1{\def\edlabel##1{%
6020     \let\edindex\nulledindex
6021     \ifx #1\ \ifnum\l@dcolcount=0 \let\Next\relax%
6022         \else\l@dcolcount=0%
6023             \let\Next=\settcclcenter%
6024         \fi%
6025     \else \disablel@dtabfeet%
6026         \stepl@dcolcount%
6027         \disable@notes%
6028         \setbox\hilfsbox=\hbox{#1}%
6029         \restore@notes%
6030         \letsforverteilen%
6031         \hskip 0.5\hilfsskip #1\hskip 0.5\hilfsskip%
6032         \hskip\edtabcolsep%
6033         \let\Next=\settcclcenter%
6034     \fi\Next}
6035
6036 %

```

```
\NEXT6037 \let\NEXT=\relax
```

```
6038  
6039 %
```

\setmrowright Typeset (recursively) rows of right justified math.

```
6040 \def\setmrowright #1\{\%  
6041   \ifx #1& \let\NEXT\relax  
6042   \else \centerline{\setmcellright #1&\\&\\&}  
6043       \let\NEXT=\setmrowright  
6044   \fi\NEXT}  
6045 %
```

\settroright Typeset (recursively) rows of right justified text.

```
6046 \def\settroright #1\{\%  
6047   \ifx #1& \let\NEXT\relax  
6048   \else \centerline{\settcclright #1&\\&\\&}  
6049       \let\NEXT=\settroright  
6050   \fi\NEXT}  
6051  
6052 %
```

\setmrowleft Typeset (recursively) rows of left justified math.

```
6053 \def\setmrowleft #1\{\%  
6054   \ifx #1& \let\NEXT\relax  
6055   \else \centerline{\setmcclleft #1&\\&\\&}  
6056       \let\NEXT=\setmrowleft  
6057   \fi\NEXT}  
6058 %
```

\settrorleft Typeset (recursively) rows of left justified text.

```
6059 \def\settrorleft #1\{\%  
6060   \ifx #1& \let\NEXT\relax  
6061   \else \centerline{\settcclleft #1&\\&\\&}  
6062       \let\NEXT=\settrorleft  
6063   \fi\NEXT}  
6064  
6065 %
```

\setmrowcenter Typeset (recursively) rows of centered math.

```
6066 \def\setmrowcenter #1\{\%  
6067   \ifx #1& \let\NEXT\relax%  
6068   \else \centerline{\setmcellcenter #1&\\&\\&}  
6069       \let\NEXT=\setmrowcenter  
6070   \fi\NEXT}  
6071 %
```

`\settrowcenter` Typeset (recursively) rows of centered text.

```

6072 \def\settrowcenter #1\\{%
6073   \ifx #1& \let\NEXT\relax
6074   \else \centerline{\settcellcenter #1&\\&\\&}
6075         \let\NEXT=\settrowcenter
6076   \fi\NEXT}
6077
6078 %

```

```

\newcommand{\nullsetzen}{%
6080   \step1@dcolcount%
6081   \l@dcolwidth=0pt%
6082   \ifnum\l@dcolcount=30\let\NEXT\relax%
6083     \l@dcolcount=0\relax
6084   \else\let\NEXT\nullsetzen%
6085   \fi\NEXT}
6086
6087 %

```

`\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.

```

6088 \newcommand{\edatleft}[3][\@empty]{%
6089   \ifx#1\@empty
6090     \vbox to 10pt{\vss\hbox{$\left#2\vrule width0pt height #3
6091                               depth 0pt \right. $\hss}\vfil}
6092   \else
6093     \vbox to 4pt{\vss\hbox{$#1\left#2\vrule width0pt height #3
6094                               depth 0pt \right. $\}\vfil}
6095   \fi}
6096 %

```

`\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.

```

6097 \newcommand{\edatright}[3][\@empty]{%
6098   \ifx#1\@empty
6099     \vbox to 10pt{\vss\hbox{$\left.\vrule width0pt height #3
6100                               depth 0pt \right#2 $\hss}\vfil}
6101   \else
6102     \vbox to 4pt{\vss\hbox{$\left.\vrule width0pt height #3
6103                               depth 0pt \right#2 #1 $\}\vfil}
6104   \fi}
6105
6106 %

```

`\edvertline` `\edvertline{ $\langle len \rangle$ }` vertical line $\langle len \rangle$ high.

```

6107 \newcommand{\edvertline}[1]{\vbox to 8pt{\vss\hbox{\vrule height #1\vfil}}
6108
6109 %

```

\edvertdots `\edvertdots{<len>}` vertical dotted line <len> high.

```

6110 \newcommand{\edvertdots}[1]{\vbox to 1pt{\vss\vbox to #1%
6111     {\cleaders\hbox{$\mathhbox{.}\vbox to 0.5em{ }$}\vfil}}}
6112
6113 %

```

\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@`.

```

6114 \newcommand{\l@dtabaddcols}[2]{%
6115     \l@dcheckstartend{#1}{#2}%
6116     \ifl@dstartendok
6117     \setcounter{addcolcount}{#1}%
6118     \@whilenum \value{addcolcount}<#2\relax \do
6119     {\advance\edfilldimen by \the \csname dcol\theaddcolcount\endcsname
6120     \advance\edfilldimen by \edtabcolsep
6121     \stepcounter{addcolcount}}%
6122     \advance\edfilldimen by \the \csname dcol\theaddcolcount\endcsname
6123     \fi
6124 }
6125
6126 %

```

\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.

```

6127 \newif\ifl@dstartendok
6128 \newcommand{\l@dcheckstartend}[2]{%
6129     \l@dstartendoktrue
6130     \ifnum #1<\@ne
6131         \l@dstartendokfalse
6132         \led@err@LowStartColumn
6133     \fi
6134     \ifnum #2>30\relax
6135         \l@dstartendokfalse
6136         \led@err@HighEndColumn
6137     \fi
6138     \ifnum #1>#2\relax
6139         \l@dstartendokfalse
6140         \led@err@ReverseColumns
6141     \fi
6142 }
6143
6144 %

```

`\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.

```
6145 \newcommand*{\edrowfill}[3]{%
6146   \l@dtabaddcols{#1}{#2}%
6147   \hb@xt@ \the\l@dcolwidth{\hb@xt@ \the\edfilldimen{#3}\hss}}
6148 \let\@edrowfill=\edrowfill
6149 \def\@EDROWFILL@#1#2#3{\@edrowfill@{#1}{#2}{#3}}
6150
6151 %
```

`\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`.

```
6152 \newcommand{\leftltab}[1]{%
6153   \hb@xt@ \z@{\vbox{\edtabindent%
6154     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
6155
6156 %
```

`\leftrtab` `\leftrtab{<text>}{<math>}` for `\edbeforetab` in `\rtab`.

```
6157 \newcommand{\leftrtab}[2]{%
6158   #2\hb@xt@ \z@{\vbox{\edtabindent%
6159     \advance\Hilfsskip by\dcoli%
6160     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
6161
6162 %
```

`\leftctab` `\leftctab{<text>}{<math>}` for `\edbeforetab` in `\ctab`.

```
6163 \newcommand{\leftctab}[2]{%
6164   \hb@xt@ \z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6165     \advance\Hilfsskip by 0.5\dcoli%
6166     \setbox\hilfsbox=\hbox{\def\edlabel##1{}%
6167       \disablel@dtabfeet$\displaystyle{#2}$}%
6168     \advance\Hilfsskip by -0.5\wd\hilfsbox%
6169     \moveleft\Hilfsskip\hbox{\ #1}}\hss}%
6170   #2}
6171
6172 %
```

`\rightctab` `\rightctab{<math>}{<text>}` for `\edaftertab` in `\ctab`.

```

6173 \newcommand{\rightctab}[2]{%
6174     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6175         \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
6176         #1\hb@xt@{\z@\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6177         \advance\Hilfsskip by 0.5\l@dcolwidth%
6178         \advance\Hilfsskip by -\wd\hilfsbox%
6179         \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6180             \disablel@dtabfeet$\displaystyle{#1}$}%
6181             \advance\Hilfsskip by -0.5\wd\hilfsbox%
6182             \advance\Hilfsskip by \edtabcolsep%
6183             \moveright\Hilfsskip\hbox{ #2}}\hss}%
6184         }
6185     }
6186     %

```

\rightltab `\rightltab{<math><math>}{<text>}` for `\edaftertab` in `\ltab`.

```

6187 \newcommand{\rightltab}[2]{%
6188     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6189         \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
6190         #1\hb@xt@{\z@\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6191         \advance\Hilfsskip by \l@dcolwidth%
6192         \advance\Hilfsskip by -\wd\hilfsbox%
6193         \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6194             \disablel@dtabfeet$\displaystyle{#1}$}%
6195             \advance\Hilfsskip by -\wd\hilfsbox%
6196             \advance\Hilfsskip by \edtabcolsep%
6197             \moveright\Hilfsskip\hbox{ #2}}\hss}%
6198         }
6199     }
6200     %

```

\rightrtab `\rightrtab{<math><math>}{<text>}` for `\edaftertab` in `\rtab`.

```

6201 \newcommand{\rightrtab}[2]{%
6202     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6203         \disablel@dtabfeet#2}%
6204         #1\hb@xt@{\z@\vbox{\edtabindent%
6205         \advance\Hilfsskip by -\wd\hilfsbox%
6206         \advance\Hilfsskip by \edtabcolsep%
6207         \moveright\Hilfsskip\hbox{ #2}}\hss}%
6208     }
6209     %
6210     %

```

\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.

```

6211 \newcommand{\rtab}[1]{%
6212   \l@dnnullfills
6213   \def\edbeforetab##1##2{\lefttrtab{##1}{##2}}%
6214   \def\edaftertab##1##2{\righttrtab{##1}{##2}}%
6215   \measurebody{#1}%
6216   \l@drestorefills
6217   \variab
6218   \setmrowright #1\\&\\%
6219   \enablel@dtabfeet}
6220
6221 %

```

\measurebody `\measurebody{<body>}` measures the array `<body>`.

```

6222 \newcommand{\measurebody}[1]{%
6223   \disablel@dtabfeet%
6224   \l@dcolcount=0%
6225   \nullsetzen%
6226   \l@dcolcount=0
6227   \measuremrow #1\\&\\%
6228   \global\l@dampcount=1}
6229
6230 %

```

\rtabtext `\rtabtext{<body>}` typesets `<body>` as a tabular with the entries right justified.

```

6231 \newcommand{\rtabtext}[1]{%
6232   \l@dnnullfills
6233   \measuretbody{#1}%
6234   \l@drestorefills
6235   \variab
6236   \settroright #1\\&\\%
6237   \enablel@dtabfeet}
6238
6239 %

```

\measuretbody `\measuretbody{<body>}` measures the tabular `<body>`.

```

6240 \newcommand{\measuretbody}[1]{%
6241   \disable@notes%
6242   \disablel@dtabfeet%
6243   \l@dcolcount=0%
6244   \nullsetzen%
6245   \l@dcolcount=0
6246   \measuretror #1\\&\\%
6247   \restore@notes%
6248   \global\l@dampcount=1}
6249
6250 %

```


\ltab Array with entries left justified.

```

\edbeforetab
\edaftertab
6251 \newcommand{\ltab}[1]{%
6252   \l@dnnullfills
6253   \def\edbeforetab##1##2{\leftltab{##1}{##2}}%
6254   \def\edaftertab##1##2{\rightltab{##1}{##2}}%
6255   \measurebody{#1}%
6256   \l@drestorefills
6257   \variab
6258   \setmrowleft #1\\&\\%
6259   \enablel@dtabfeet}
6260
6261 %

```

\ltabtext Tabular with entries left justified.

```

6262 \newcommand{\ltabtext}[1]{%
6263   \l@dnnullfills
6264   \measuretbody{#1}%
6265   \l@drestorefills
6266   \variab
6267   \settrrowleft #1\\&\\%
6268   \enablel@dtabfeet}
6269
6270 %

```

\ctab Array with centered entries.

```

\edbeforetab
\edaftertab
6271 \newcommand{\ctab}[1]{%
6272   \l@dnnullfills
6273   \def\edbeforetab##1##2{\leftctab{##1}{##2}}%
6274   \def\edaftertab##1##2{\rightctab{##1}{##2}}%
6275   \measurebody{#1}%
6276   \l@drestorefills
6277   \variab
6278   \setmrowcenter #1\\&\\%
6279   \enablel@dtabfeet}
6280
6281 %

```

\ctabtext Tabular with entries centered.

```

6282 \newcommand{\ctabtext}[1]{%
6283   \l@dnnullfills
6284   \measuretbody{#1}%
6285   \l@drestorefills
6286   \variab
6287   \settrrowcenter #1\\&\\%
6288   \enablel@dtabfeet}
6289
6290 %

```

```
\spreadtext \newcommand{\spreadtext}[1]{%\l@dcolcount=\l@dampcount%
6292 \hb@xt@ \the\l@dcolwidth{\hbox{#1}\hss}}
6293 %
```

```
\spreadmath \newcommand{\spreadmath}[1]{%
6295 \hb@xt@ \the\l@dcolwidth{\hbox{$\displaystyle{#1}$}\hss}}
6296
6297 %
```

\HILFSskip More helpers.

```
\Hilfsskip
6298 \newskip\HILFSskip
6299 \newskip\Hilfsskip
6300
6301 %
```

```
\EDTABINDENT \newcommand{\EDTABINDENT}{%
6303 \ifnum\l@dcolcount=30\let\NEXT\relax\l@dcolcount=0%
6304 \else\step\l@dcolcount%
6305 \advance\Hilfsskip by\l@dcolwidth%
6306 \ifdim\l@dcolwidth=0pt\advance\hilfscount\@ne
6307 \else\advance\Hilfsskip by \the\hilfscount\edtabcolsep%
6308 \hilfscount=1\fi%
6309 \let\NEXT=\EDTABINDENT%
6310 \fi\NEXT}%
6311 %
```

\edtabindent (was \tabindent)

```
6312 \newcommand{\edtabindent}{%
6313 \l@dcolcount=0\relax
6314 \Hilfsskip=0pt%
6315 \hilfscount=1\relax
6316 \EDTABINDENT%
6317 \hilfsskip=\hsize%
6318 \advance\hilfsskip -\Hilfsskip%
6319 \Hilfsskip=0.5\hilfsskip%
6320 }%
6321
6322 %
```

\EDTAB (was \TAB)

```
6323 \def\EDTAB #1|#2|{%
6324 \setbox\tabhilfbox=\hbox{$\displaystyle{#1}$}%
6325 \setbox\tabHilfbox=\hbox{$\displaystyle{#2}$}%
6326 \advance\tabelskip -\wd\tabhilfbox%
```

```

6332 \def\EDTABtext #1|#2|{%
6333   \setbox\tabhilfbox=\hbox{#1}%
6334   \setbox\tabHilfbox=\hbox{#2}%
6335   \advance\tabelskip -\wd\tabhilfbox%
6336   \advance\tabelskip -\wd\tabHilfbox%
6337   \unhbox\tabhilfbox\hskip\tabelskip%
6338   \unhbox\tabHilfbox}%
6339 %

```

```

\begin{document}
\begin{tikzpicture}
\begin{tikzstyle}
\tabHilfbox = [draw=black, fill=white, text=black, font=\ttfamily]
\end{tikzstyle}
\begin{tikzstyle}
\tabHilfbox = [draw=black, fill=white, text=black, font=\ttfamily]
\end{tikzstyle}
\end{tikzpicture}
\end{document}

```

edarrayl edarrayc edarrayr The ‘environment’ forms for \ltab, \ctab and \rtab.

```
6344 \newenvironment{edarrayl}{\l@dcollect@body\ltab}{}
6345 \newenvironment{edarrayc}{\l@dcollect@body\ctab}{}
6346 \newenvironment{edarrayr}{\l@dcollect@body\rtab}{}
6347
6348 %
```

edtabularl edtabularc edtabularr The ‘environment’ forms for `\ltabtext`, `\ctabtext` and `\rtabtext`.

```

6349 \newenvironment{edtabularl}{\l@ddcollect@body\ltabtext}{\}
6350 \newenvironment{edtabularc}{\l@ddcollect@body\ctabtext}{\}
6351 \newenvironment{edtabularr}{\l@ddcollect@body\rtabtext}{\}
6352
6353 %

```

`\initnumbering@quote` This macro, called at the begin of any numbered section, redefine locally the quotation and quote environment, in order to allow to use it inside numbered section.

```
\quotation \initnumbering@quote defines quotation environment.
\endquotation
\quote
\endquote
```

```

6354 \newcommand{\initnumbering@quote}{
6355   \ifnoquotation@
6356   \renewcommand{\quotation}{\par\leavevmode%
6357     \parindent=1.5em%
6358     \skipnumbering%
6359     \ifautopar%
6360       \vskip-\parskip%
6361     \else%
6362       \vskip\topsep%
6363     \fi%
6364     \global\leftskip=\leftmargin%
6365     \global\rightskip=\leftmargin%
6366   }
6367   \renewcommand{\endquotation}{\par%
6368     \global\leftskip=0pt%
6369     \global\rightskip=0pt%
6370     \leavevmode%
6371     \skipnumbering%
6372     \ifautopar%
6373       \vskip-\parskip%
6374     \else%
6375       \vskip\topsep%
6376     \fi%
6377   }
6378   \renewcommand{\quote}{\par\leavevmode%
6379     \parindent=0pt%
6380     \skipnumbering%
6381     \ifautopar%
6382       \vskip-\parskip%
6383     \else%
6384       \vskip\topsep%
6385     \fi%
6386     \global\leftskip=\leftmargin%
6387     \global\rightskip=\leftmargin%
6388   }
6389   \renewcommand{\endquote}{\par%
6390     \global\leftskip=0pt%
6391     \global\rightskip=0pt%
6392     \leavevmode%
6393     \skipnumbering%
6394     \ifautopar%
6395       \vskip-\parskip%
6396     \else%
6397       \vskip\topsep%
6398     \fi%
6399   }
6400   \fi
6401 }
6402 %

```

XXX Section's title commands

XXX.1 Commands to disable some feature

`\ledsectnotoc` The `\ledsectnotoc` only disables the `\addcontentsline` macro.

```
6403 \newcommand{\ledsectnotoc}{\let\addcontentsline@gobblethree}
6404 %
```

`\ledsectnomark` The `\ledsectnomark` only disables the `\chaptermark`, `\sectionmark` and `\subsectionmark` macros.

```
6405 \newcommand{\ledsectnomark}{%
6406   \let\chaptermark@gobble%
6407   \let\sectionmark@gobble%
6408   \let\subsectionmark@gobble%
6409 }
6410 %
```

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 \LaTeX is run a other time, this file is read. That:
 - Adds the pstart number to a list of pstarts where a sectioning command is used.
 - Defines a command, the name of which contains the pstart number, and which calls the normal \LaTeX sectioning command.
4. This last command is called when the pstart is effectively printed.

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).

```
6411 \catcode`\#=12
6412 \notbool{@noeled@sec}{%
6413 %
```

\beforeeledchapter For technical reasons, not yet solved, page-breaking before chapters can't be made automatically by eledmac. Users have to use \beforeeledchapter.

```
6414 \ifl@dmemoir
6415 \newcommand\beforeeledchapter{%
6416 \clearforchapter%
6417 }
6418 \else
6419 \newcommand\beforeeledchapter{%
6420 \if@openright%
6421 \cleardoublepage%
6422 \else%
6423 \clearpage%
6424 \fi%
6425 }
6426 \fi
6427 %
```

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.

```
6428 \newif\if@eled@sectioning
6429 %
```

\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.

```
6430 \def\print@rightmargin@eledsection{%
6431 \if@eled@sectioning%
6432 \begingroup%
6433 \if@RTL%
6434 \let\llap\rlap%
6435 \let\leftlinenum\rightlinenum%
```

```

6436         \let\leftlinenum\rightlinenumR%
6437         \let\l@drd@ta\l@dld@ta%
6438         \let\l@drsn@te\l@dlsn@te%
6439     \fi%
6440     \hfill\l@drd@ta \csuse{LR}{\l@drsn@te}%
6441     \endgroup%
6442 \fi%
6443 }%
6444
6445 \def\print@leftmargin@eledsection{%
6446     \if@eled@sectioning%
6447         \leavevmode%
6448         \begingroup%
6449         \if@RTL%
6450             \let\rlap\llap%
6451             \let\rightlinenum\leftlinenum%
6452             \let\rightlinenumR\leftlinenumR%
6453             \let\l@dld@ta\l@drd@ta%
6454             \let\l@dlsn@te\l@drsn@te%
6455         \fi%
6456         \l@dld@ta\csuse{LR}{\l@dlsn@te}%
6457         \endgroup%
6458     \fi%
6459 }%
6460
6461 %

```

XXX.5 Patching standard commands

`\chapter` We have to patch \LaTeX , book and memoir sectioning commands in order to:

- `\M@ssect` • Disable `\edtext` inside.
- `\@mem@old@ssect` • Disable page breaking (for `\chapter`).
- `\@makechapterhead` • Add line numbers and sidenotes.
- `\@makechapterhead`
- `\@makeschapterhead`

`\@ssect`
`\@sssect`

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.

```

6462 \AtBeginDocument{%
6463     \patchcmd{\chapter}{\clearforchapter}{%
6464         \if@eled@sectioning\else%
6465         \ifl@dprintingpages\else%
6466             \clearforchapter%
6467         \fi%
6468     \fi%
6469 }
6470 {}
6471 {}

```

```

6472
6473
6474 \pretocmd{\M@sect}
6475   {\let\old@edtext=\edtext%
6476   \let\edtext=\dummy@edtext@showlemma%
6477   }
6478   {}
6479   {}
6480
6481 \apptocmd{\M@sect}
6482   {\let\edtext=\old@edtext}
6483   {}
6484   {}
6485
6486 \patchcmd{\M@sect}
6487   { #9}
6488   { #9%
6489   \print@rightmargin@eledsection%
6490   }
6491   {}
6492   {}
6493
6494 \patchcmd{\M@sect}
6495   {\hskip #3\relax}
6496   {\hskip #3\relax%
6497   \print@leftmargin@eledsection%
6498   }
6499   {}
6500   {}
6501
6502 \patchcmd{\@mem@old@ssect}
6503   {#5}
6504   {#5%
6505   \print@leftmargin@eledsection%
6506   }
6507   {}
6508   {}
6509
6510 \patchcmd{\@mem@old@ssect}
6511   {\hskip #1}
6512   {\hskip #1%
6513   \print@rightmargin@eledsection%
6514   }
6515   {}
6516   {}
6517
6518 \patchcmd{\chapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
6519   \if@eled@sectioning\else%
6520   \ifl@dprintingpages\else%
6521     \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%
6522 \fi%
6523 \fi%
6524 }%
6525 {}%
6526 {}%
6527
6528 \patchcmd{\scr@startchapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
\if@eled@sectioning\else%
6529 \ifl@dprintingpages\else%
6530 \if@openright\cleardoublepage\else\clearpage\fi}%No clearpage inside a
6531 \Pages: will keep critical notes from printing on the title page. Here for
scrbook.
6532 \fi%
6533 \fi%
6534 }
6535 {}
6536 {}
6537
6538 \patchcmd{\@makechapterhead}
6539 {#1}
6540 {\print@leftmargin@eledsection%
6541 #1%
6542 \print@rightmargin@eledsection%
6543 }
6544 {}
6545 {}
6546
6547 \patchcmd{\@makechapterhead}% For BIDI
6548 {\if@RTL\raggedleft\else\raggedright\fi}%
6549 {\if@eled@sectioning\else%
6550 \if@RTL\raggedleft\else\raggedright\fi%
6551 \fi%
6552 }%
6553 {}%
6554 {}%
6555
6556 \patchcmd{\@makeschapterhead}
6557 {#1}
6558 {\print@leftmargin@eledsection%
6559 #1%
6560 \print@rightmargin@eledsection%
6561 }
6562 {}
6563 {}
6564
6565 \pretocmd{\@sect}
6566 {\let\old@edtext=\edtext

```

```

6567 \let\edtext=\dummy@edtext@showlemma%
6568 }
6569 {}
6570 {}
6571
6572 \apptocmd{\@sect}
6573 {\let\edtext=\old@edtext}
6574 {}
6575 {}
6576
6577 \pretocmd{\@ssect}
6578 {\let\old@edtext=\edtext%
6579 \let\edtext=\dummy@edtext@showlemma%
6580 }
6581 {}
6582 {}
6583
6584 \apptocmd{\@ssect}
6585 {\let\edtext=\old@edtext}
6586 {}
6587 {}
6588
6589 %

```

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.

```

6590 \@ifpackageloaded{nameref}{
6591
6592   \patchcmd{\NR@sect}
6593     {#8}
6594     {#8%
6595     \print@rightmargin@eledsection%
6596     }
6597     {}
6598     {}
6599
6600   \patchcmd{\NR@sect}
6601     {\hskip #3\relax}
6602     {\hskip #3\relax%
6603     \print@leftmargin@eledsection%
6604     }
6605     {}
6606     {}
6607
6608   \patchcmd{\NR@ssect}
6609     {#5}
6610     {#5%
6611     \print@rightmargin@eledsection%
6612     }
6613     {}

```

```

6614 {}
6615
6616 \patchcmd{\NR@ssect}
6617 {\hskip #1}
6618 {\hskip #1%
6619 \print@leftmargin@eledsection%
6620 }
6621 {}
6622 {}
6623 }%
6624 {
6625 \patchcmd{\@sect}
6626 {#8}
6627 {#8%
6628 \print@rightmargin@eledsection%
6629 }
6630 {}
6631 {}
6632
6633 \patchcmd{\@sect}
6634 {\hskip #3\relax}
6635 {\hskip #3\relax%
6636 \print@leftmargin@eledsection%
6637 }
6638 {}
6639 {}
6640
6641 \patchcmd{\@ssect}
6642 {#5}
6643 {#5%
6644 \print@rightmargin@eledsection%
6645 }
6646 {}
6647 {}
6648
6649 \patchcmd{\@ssect}
6650 {\hskip #1}
6651 {\hskip #1%
6652 \print@leftmargin@eledsection%
6653 }
6654 {}
6655 {}
6656 }%
6657 }
6658 %

```

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.

```

6659 {}}%

```

```

6660 \protect\catcode`\#=6 %Space NEEDS by \catcode
6661 \notbool{@noeled@sec}{%
6662 %

```

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.

```

6663 \newwrite\eled@sectioning@out
6664 %

```

`\eledchapter` And now, the user sectioning commands, which write to the file, and also add content as a “normal” line.

```

\eledsubsection
\eledsubsubsection
\eledchapter*
\eledsection*
\eledsubsection*
\eledsubsubsection*
6665 \newcommand{\eledchapter}[2] [] {%
6666   #2%
6667   \ifledRcol%
6668     \immediate\write\eled@sectioningR@out{%
6669     \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
6670     }%
6671   \else%
6672     \immediate\write\eled@sectioning@out{%
6673     \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{L}
6674     }%
6675   \fi%
6676 }
6677
6678 \newcommand{\eledsection}[2] [] {%
6679   #2%
6680   \ifledRcol%
6681     \immediate\write\eled@sectioningR@out{%
6682     \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
6683     }%
6684   \else%
6685     \immediate\write\eled@sectioning@out{%
6686     \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{L}
6687     }%
6688   \fi%
6689 }
6690
6691 \newcommand{\eledsubsection}[2] [] {%
6692   #2%
6693   \ifledRcol%
6694     \immediate\write\eled@sectioningR@out{%
6695     \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
6696     }%
6697   \else%

```

```

6698 \immediate\write\eled@sectioning@out{%
6699 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{-}{-}
6700 }%
6701 \fi%
6702 }
6703 \newcommand{\eledsubsubsection}[2][{}]{%
6704 #2%
6705 \ifledRcol%
6706 \immediate\write\eled@sectioningR@out{%
6707 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR
6708 }{-}{R}
6709 }%
6710 \else%
6711 \immediate\write\eled@sectioning@out{%
6712 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL
6713 }{-}{-}
6714 }%
6715 \fi%
6716 }
6717 \WithSuffix\newcommand\eledchapter*[2][{}]{%
6718 #2%
6719 \ifledRcol%
6720 \immediate\write\eled@sectioningR@out{%
6721 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
6722 }%
6723 \else%
6724 \immediate\write\eled@sectioning@out{%
6725 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{-}
6726 }%
6727 \fi%
6728 }
6729
6730 \WithSuffix\newcommand\eledsection*[2][{}]{%
6731 #2%
6732 \ifledRcol%
6733 \immediate\write\eled@sectioningR@out{%
6734 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
6735 }%
6736 \else%
6737 \immediate\write\eled@sectioning@out{%
6738 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{-}
6739 }%
6740 \fi%
6741 }
6742
6743 \WithSuffix\newcommand\eledsubsection*[2][{}]{%
6744 #2%
6745 \ifledRcol%
```

```

6746 \immediate\write\eled@sectioningR@out{%
6747 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsR}{*}{
R}
6748 }%
6749 \else%
6750 \immediate\write\eled@sectioning@out{%
6751 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL
}{*}{
6752 }%
6753 \fi%
6754 }
6755
6756 \WithSuffix\newcommand\eledsubsubsection*[2] [] {%
6757 #2%
6758 \ifledRcol%
6759 \immediate\write\eled@sectioningR@out{%
6760 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsR
}{*}{R}
6761 }%
6762 \else%
6763 \immediate\write\eled@sectioning@out{%
6764 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL
}{*}{
6765 }%
6766 \fi%
6767 }
6768 %

```

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.

```

6769 \def\eled@chapter#1#2#3#4#5{%
6770 \ifstrempy{#4}%
6771 {%
6772 \ifstrempy{#1}%
6773 {%
6774 \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter{#2}}%

```

```

6775 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
chaptermark{#2}}%
6776 }%Need for \pairs, because of using parbox.
6777 {%
6778 \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter{#1}{#2}}%
6779 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
chaptermark{#2}}%Need for \pairs, because of using parbox.
6780 }%
6781 }%
6782 {%
6783 \ifstrempy{#1}%
6784 {\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#2}}}%
6785 {\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#1}{#2}}}%Bug in LaTeX!
6786 }%
6787 \listcsadd{eled@sections#5@@}{#3}%
6788 }
6789 \def\eled@section#1#2#3#4#5{%
6790 \ifstrempy{#4}%
6791 {\ifstrempy{#1}%
6792 {%
6793 \global\csdef{eled@sectioning@#3#5}{\section{#2}}%
6794 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
sectionmark{#2}}%Need for \pairs, because of using parbox.
6795 }%
6796 {%
6797 \global\csdef{eled@sectioning@#3#5}{\section[#1]{#2}}%
6798 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
sectionmark{#1}}%Need for \pairs, because of using parbox.
6799 }%
6800 }%
6801 {\ifstrempy{#1}%
6802 {\global\csdef{eled@sectioning@#3#5}{\section*{#2}}}%
6803 {\global\csdef{eled@sectioning@#3#5}{\section*{#1}{#2}}}%Bug in
LaTeX!
6804 }
6805 \listcsadd{eled@sections#5@@}{#3}%
6806 }
6807 \def\eled@subsection#1#2#3#4#5{%
6808 \ifstrempy{#4}%
6809 {\ifstrempy{#1}%
6810 {%
6811 \global\csdef{eled@sectioning@#3#5}{\subsection{#2}}%
6812 \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)
6813 }%
6814 {%

```

```

6815 \global\csdef{eled@sectioning@#3#5}{\subsection[#1]{#2}}%
6816 \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)
6817 }%
6818 }%
6819 {\ifstrempy{#1}%
6820 {\global\csdef{eled@sectioning@#3#5}{\subsection*{#2}}}%
6821 {\global\csdef{eled@sectioning@#3#5}{\subsection*{#1}{#2}}}%Bug in
LaTeX!
6822 }
6823 \listcsgadd{eled@sections#5@@}{#3}%
6824 }
6825 \def\eled@subsubsection#1#2#3#4#5{%
6826 \ifstrempy{#4}%
6827 {\ifstrempy{#1}%
6828 {\global\csdef{eled@sectioning@#3#5}{\subsubsection{#2}}}%
6829 {\global\csdef{eled@sectioning@#3#5}{\subsubsection[#1]{#2}}}%
6830 }%
6831 {\ifstrempy{#1}%
6832 {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#2}}}%
6833 {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#1}{#2}}}%Bug
in LaTeX!
6834 }
6835 \listcsgadd{eled@sections#5@@}{#3}%
6836 }
6837
6838 %

End of the conditional test about noeledsec option.

6839 }{}
6840 %

```

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.

```

6841 \def\normal@page@break{}
6842 %

```


\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.

```
6843 \def\l@prev@pb{}
6844 \def\l@prev@nopb{}
6845 %
```

\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.

```
6846 \newcommand{\ledpb}{\write\linenum@out{\string\led@pb}}
6847 \newcommand{\ledpbnum}[1]{\write\linenum@out{\string\led@pbnum{#1}}}
6848 \newcommand{\lednopb}{\write\linenum@out{\string\led@nopb}}
6849 \newcommand{\lednopbnum}[1]{\write\linenum@out{\string\led@nopbnum{#1}}}
6850 %
```

\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.

```
\led@pbnum
6851 \newcommand{\led@pb}{\listxadd{\l@prev@pb}{\the\absline@num}}
6852 \newcommand{\led@pbnum}[1]{\listxadd{\l@prev@pb}{#1}}
6853 \newcommand{\led@nopb}{\listxadd{\l@prev@nopb}{\the\absline@num}}
6854 \newcommand{\led@nopbnum}[1]{\listxadd{\l@prev@nopb}{#1}}
6855 %
```

\ledpbsetting The \ledpbsetting macro only changes the value of \led@pb@macro, for which the default value is before.

```
\led@pb@setting
6856 \def\led@pb@setting{before}
6857 \newcommand{\ledpbsetting}[1]{\gdef\led@pb@setting{#1}}
6858 %
```

\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.

```
6859 \newcommand{\led@check@pb}{\xifinlist{\the\absline@num}{\l@prev@pb}{\pagebreak[4]}}
6860 \newcommand{\led@check@nopb}{%
6861   \IfStrEq{\led@pb@setting}{before}{%
6862     \xifinlist{\the\absline@num}{\l@prev@nopb}%
6863     {\numdef{\abs@prevline}{\the\absline@num-1}}%
6864     \xifinlist{\abs@prevline}{\normal@page@break}%
6865     {\nopagebreak[4]\enlargethispage{\baselineskip}}%
6866     {}}%
6867   {}}%
```

```

6868     {}%
6869   {}%
6870   \IfStrEq{\led@pb@setting}{after}{%
6871     \xifinlist{\the\absline@num}{\l@prev@nopb}{%
6872       \xifinlist{\the\absline@num}{\normal@page@break}%
6873       {\nopagebreak[4]\enlargethispage{\baselineskip}}%
6874     }%
6875 }%
6876   {}}%
6877   {}%
6878   {}%
6879 }
6880 %

```

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.

```

6881 \newcommand{\check@pb@in@verse}{%
6882   \ifinstanza\iflednopbinverse\ifinserthangingsymbol% Using stanzas and
6883   enabling page breaks in verse control, while on a hanging verse.
6884   \ifnum\page@num=\last@page@num\else%If we have change page
6885   \IfStrEq{\led@pb@setting}{before}{%
6886     \numgdef{\abs@line@verse}{\the\absline@num-1}%
6887     \ledpbnum{\abs@line@verse}%
6888     }{}%
6889   \IfStrEq{\led@pb@setting}{after}{%
6890     \numgdef{\abs@line@verse}{\the\absline@num-1}%
6891     \lednopbnum{\abs@line@verse}%
6892     }{}%
6893   \fi%
6894 }
6895 %

```

XXXIII Compatibility with eledmac

Here, we define some command for the eledmac-compat option.

```

6896 \ifeledmaccompat@%
6897
6898 \newcommand{\footnormalX}[1]{\arrangementX[#1]{normal}}%
6899 \newcommand{\footparagraphX}[1]{\arrangementX[#1]{paragraph}}%
6900 \newcommand{\foottwocolX}[1]{\arrangementX[#1]{twocol}}%
6901 \newcommand{\footthreecolX}[1]{\arrangementX[#1]{threecol}}%
6902
6903 \unless\ifnocritical@
6904 \newcommand{\footnormal}[1]{\Xarrangement[#1]{normal}}%
6905 \newcommand{\footparagraph}[1]{\Xarrangement[#1]{paragraph}}%
6906 \newcommand{\foottwocol}[1]{\Xarrangement[#1]{twocol}}%
6907 \newcommand{\footthreecol}[1]{\Xarrangement[#1]{threecol}}%
6908 \let\hsizetwocol\Xhsizetwocol
6909 \let\hsizethreecol\Xhsizethreecol
6910 \let\bhookXnote\Xbhooknote
6911 \let\boxsymlinenum\Xboxsymlinenum
6912 \let\symlinenum\Xsymlinenum
6913 \let\beforenumberinfootnote\Xbeforenumber
6914 \let\afternumberinfootnote\Xafternumber
6915 \let\beforeXsymlinenum\XbeforeXsymlinenum
6916 \let\afterXsymlinenum\XafterXsymlinenum
6917 \let\inplaceofnumber\Xinplaceofnumber
6918 \let\Xlemmaseparator\lemmaseparator
6919 \let\afterlemmaseparator\Xafterlemmaseparator
6920 \let\beforelemmaseparator\Xbeforelemmaseparator
6921 \let\inplaceoflemmaseparator\Xinplaceoflemmaseparator
6922 \let\txbeforeXnotes\Xtxbeforenotes
6923 \let\afterXrule\Xafterrule
6924 \let\numberonlyfirstinline\Xnumberonlyfirstinline
6925 \let\numberonlyfirstintwolines\Xnumberonlyfirstintwolines
6926 \let\nonumberinfootnote\Xnonumberinfootnote
6927 \let\pstartinfootnote\Xpstart
6928 \let\pstartinfootnoteeverytime\Xpstarteverytime
6929 \let\onlyXpstart\Xonlypstart
6930 \let\Xnonumberinfootnote\Xnonumber
6931 \let\nonbreakableafternumber\Xnonbreakableafternumber
6932 \let\maxhXnotes\Xmaxhnotes
6933 \let\beforeXnotes\Xbeforenotes
6934 \let\boxlinenum\Xboxlinenum
6935 \let\boxlinenumalign\Xboxlinenumalign
6936 \let\boxstartlinenum\Xboxstartlinenum
6937 \let\boxendlinenum\Xboxendlinenum
6938 \let\twolines\Xtwolines
6939 \let\morethantwolines\Xmorethantwolines
6940 \let\twolinesbutnotmore\Xtwolinesbutnotmore
6941 \let\twolinesonlyinsamepage\Xtwolinesonlyinsamepage

```

```

6942 \fi
6943
6944 \unless\ifnofamiliar@
6945 \let\notesXwidthliketwocolumns\noteswidthliketwocolumnsX
6946 \fi
6947 \newcommandx{\parafootsep}[2][1,usedefault]{%
6948   \Xparafootsep[#1]{#2}%
6949   \parafootsepX[#1]{#2}
6950 }%
6951
6952 \newcommandx{\afternote}[2][1,usedefault]{%
6953   \Xafternote[#1]{#2}%
6954   \afternoteX[#1]{#2}%
6955 }%
6956
6957 \unless\ifnoend@
6958 \let\XendXtwolines\Xendtwolines
6959 \let\XendXmoreethantwolines\Xendmoreethantwolines
6960 \let\bhookXendnote\Xendbhooknote
6961 \let\boxXendlinenum\Xendboxlinenum%
6962 \let\boxXendlinenumalign\Xendboxlinenumalign%
6963 \let\boxXendstartlinenum\Xendboxstartlinenum%
6964 \let\boxXendendlinenum\Xendboxendlinenum%
6965 \let\XendXlemmaseparator\Xendlemmaseparator
6966 \let\XendXbeforelemmaseparator\Xendbeforelemmaseparator
6967 \let\XendXafterlemmaseparator\Xendafterlemmaseparator
6968 \let\XendXinplaceoflemmaseparator\Xendinplaceoflemmaseparator
6969 \fi
6970
6971 \AtBeginDocument{%
6972   \ifdef\lineref{}\let\lineref\edlineref}%
6973 }%
6974
6975
6976 \fi%
6977 %

```

</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 changed³³ to `\critext`; use the `\edtext` macro instead. However, if you do use `\critext` (the new name for `\text`), the following is a reminder.

`\critext` Within numbered paragraphs, footnotes and endnotes are generated by forms of the `\critext` macro:

```
\critext{⟨lemma⟩}⟨commands⟩/
```

The `⟨lemma⟩` argument is the lemma in the main text: `\critext` both prints this as part of the text, and makes it available to the `⟨commands⟩` you specify to generate notes. The `/` at the end terminates the command; it is part of the macro's definition so that spaces after the macro will be treated as significant.

For example:

<pre>I saw my friend \critext{Smith} \Afootnote{Jones C, D.}/ on Tuesday.</pre>	<pre>1 I saw my friend 2 Smith on Tuesday. 2 Smith] Jones C, D.</pre>
---	---

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

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

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

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

The second argument of the `\critext` macro, `⟨commands⟩`, is the same as the second argument to the `\edtext` macro.

It is possible to define aliases for `\critext`, which can be easier to type. You can make a single character substitute for `\critext` by saying this:

³³A 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. 102 to find out why.) Aliases of the second kind may be nested without any problem.

If you really have to use `\critext` in any of the tabular or array environments, then `\edtext` must not be used in the same environment. If you use `\critext` in one of these environments then you have to issue the declaration `\usingcritext` beforehand. The declaration `\usingedtext` must be issued to revert to the default assumption that `\edtext` will be used.

Appendix A.2 Migration from ledmac to eledmac

In `eledmac`, some changes were made in the code to allow easy customization. This may cause problems for people who have already made their own. The next sections explain how to handle this.

If you have created your own series using `\addfootins` and `\addfootinsX`, you must use instead the `\newseries` command (see 5.5.1 p. 28), and remove any `\Xfootnote` command.

If you have customized the `\XXXXXfmt` command, please check whether you can achieve the same by the commands documented for display options (6 p. 28) or `\Xfootnote` options (5.2.2 p. 22). Otherwise please add a new ticket on Github to request a new function for doing this.³⁴

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]
```

³⁴<https://github.com/maieul/ledmac/issues>

If you do not make that, you will get a spurious [X], where X is series letter.

If you used a `\protect` command inside a `\footnote` command inside a numbered section, you must change the `\protect` to `\noexpand`. Otherwise the command after the `\protect` will be discarded.

Appendix A.3 Migration to eledmac 1.5.1

The version 1.5.1 corrects a bug in `stanzaindentsrepetition` (cf. 8.3 p. 39). This bug had two consequences:

1. `stanzaindentsrepetition` did not work when its value was greater than 2.
2. `stanzaindentsrepetition` worked wrong when its value was equal to 2.

So, if you used `stanzaindentsrepetition` with a value equal to 2, you had to change your `\setstanzaindents`. Explanation:

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

This code, in versions prior to 1.5.1, made the first line have an indentation of 0, the second line of 1, the third verse of 0, the fourth verse of 1 and so forth.

But this code should have instead achieved quite the contrary: the first line would have an indentation of 1, the second line of 0, the third line of 1, the fourth line of 0 and so forth.

So version 1.5.1 corrected this bug. If you want to keep the former presentation, you must change:

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

to:

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

Appendix A.4 Migration to eledmac 1.12.0

The migration to eledmac 1.12.0 is easy:

- You must first delete all the auxiliary files, then compile your document three times as usual.
- If you have modified `\l@reg`, which is not advisable, you must rename it to `\@nl@reg`.

There is an additional problem. If you have put text into brackets just after `\pstart` or `\pend`, this text will be considered to be an optional argument of `\pstart` or `\pend` (see 4.2.3 p. 16). If so, add a `\relax` between `\pstart`/`\pend` and the first bracket.

The version 1.12.0 also introduce a better way to handle sectional divisions inside numbered text. Please read 14.2 p. 52.

Appendix A.5 Migration to eledmac 17.1

This version changes the default setting of `\Xpstart`. Henceforth, pstart numbers will be printed in footnotes within the section of text where you have called `\numberpstarttrue`.

We do not see any reason to print them in the other sections. However, if you want to print the pstart numbers in all of the footnotes, whatever the section, without having to use `\numberpstarttrue`, you can use `\Xpstarteverytime`.

Appendix A.6 Migration to eledmac 1.21.0

Appendix A.6.1 `\Xledsetnormalparstuff` and `\ledsetnormalparstuffX`

The `\ledsetnormalparstuff` has been split into two different commands:

- `\Xledsetnormalparstuff` for critical notes;
- `\ledsetnormalparstuffX` for familiar notes.

Both commands can take an optional argument which is the series letter. If you have redefined `\ledsetnormalparstuff` or any of the commands which call them, you must change them accordingly.

Appendix A.6.2 Endnotes

In any case, delete the `.end` file before the next run.

The previous version of Eledmac had a bug: there were two spaces between the starting page number and the starting line number, but only one space between the ending page number and the ending line number.

As a matter of fact, a spurious space was added after the first `\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 bug when using `\Xbhooknote` and `\bhooknoteX` not in order to execute code at the beginning of each notes, but to insert content of at the beginning of each notes.

People which use these commands to do it, which is not the original idea, must change to things:

1. Horizontal space is no more automatically added after the content of `\Xbhooknote`/`\bhooknoteX` argument. You must include it manually. So instead of `\Xbhooknote{content}`, use `\Xbhooknote{content }`.
2. Indent is no more automatically added before the content of `\Xbhooknote`/`\bhooknoteX` argument. If you want to keep it, add `\indent` in the argument of `\Xbhooknote`/`\bhooknoteX`.

References

- [Bre96] Herbert Breger. `tabmac`. October 1996. (Available from CTAN in `macros/plain/contrib/tabmac`)
- [Bur01] John Burt. ‘Typesetting critical editions of poetry’. *TUGboat*, **22**, 4, pp 353–361, December 2001. (Code available from CTAN in `macros/latex/contrib/poemscol`)
- [Eck03] Matthias Eckermann. *The Parallel-Package*. April 2003. (Available from CTAN in `macros/latex/contrib/parallel`)
- [Fai03] Robin Fairbairns. *footmisc — a portmanteau package for customising footnotes in L^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>	39
<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

<code>\@emptytoks</code>	1
<code>\@fnpos</code>	1
<code>\@footnotemark</code>	1
<code>\@footnotetext</code>	1
<code>\@getfirstseries</code>	1
<code>\@gobblefive</code>	1
<code>\@gobblefour</code>	1
<code>\@gobblethree</code>	1
<code>\@h</code>	1
<code>\@hangingsymbol</code>	1
<code>\@iiiminipage</code>	1
<code>\@insertstanza</code>	1
<code>\@k</code>	1
<code>\@l@dttempcnta</code>	1
<code>\@l@dttempcntb</code>	1
<code>\@lab</code>	1
<code>\@led@testifnofoot</code>	1
<code>\@lemma</code>	1
<code>\@line@#num</code>	1
<code>\@lock</code>	1
<code>\@lopL</code>	1
<code>\@lopR</code>	1
<code>\@makechapterhead</code>	1
<code>\@makeschapterhead</code>	1
<code>\@mem@extranofeet</code>	1
<code>\@mem@old@ssect</code>	1
<code>\@mpfnpos</code>	1
<code>\@nl</code>	1
<code>\@nl@reg</code>	1
<code>\@opXfeet</code>	1
<code>\@pend</code>	1
<code>\@pendR</code>	1
<code>\@ref</code>	1
<code>\@ref@reg</code>	1
<code>\@sect</code>	1
<code>\@series</code>	1
<code>\@set</code>	1
<code>\@sidenotesep</code>	1
<code>\@ssect</code>	1
<code>\@startstanza</code>	1
<code>\@stopstanza</code>	1
<code>\@sw</code>	1
<code>\@tag</code>	1
<code>\@wredindex</code>	1
<code>\@xloop</code>	1
<code>\@xympar</code>	1
<code>CLASSbook</code>	271
<code>CLASSmemoir</code>	166, 210, 211, 235, 271, 289, 339, 343
<code>CLASSscrbook</code>	343
<code>COMMAND*footnote</code>	56

COMMAND\...\@footnotemark...	168
COMMAND\...d@ta	124
COMMAND\<hook	
@<series	201
COMMAND\<hookname	
<pseudoserries	203
COMMAND\<type	
footfmt	158
COMMAND\@@line	151
COMMAND\@MM	139, 340
COMMAND\@Rlineflag	237, 340
COMMAND\@add@	261
COMMAND\@adv	89
COMMAND\@apprefprefixmore	219, 220
COMMAND\@apprefprefixsingle	219, 220
COMMAND\@bspback	213
COMMAND\@doclearpage	211, 333, 343
COMMAND\@doreinfeetX	343
COMMAND\@dprintingcolumns	340
COMMAND\@edindex@hyperref	237, 238
COMMAND\@edtext@	105
COMMAND\@esphack	213
COMMAND\@fnpos	183
COMMAND\@footnotemark	166, 333, 343
COMMAND\@footnotetext	166, 167, 333
COMMAND\@gobble	104
COMMAND\@gobblefive	198, 341
COMMAND\@gobblefour	339
COMMAND\@gobblethree	332
COMMAND\@h	153
COMMAND\@hangingsymbol	239
COMMAND\@iiiminipage	228, 229, 332, 343
COMMAND\@iiiminipage	228
COMMAND\@l	338
COMMAND\@l@tempcnta	127, 128, 135
COMMAND\@l@tempcntb	128
COMMAND\@l@reg	338
COMMAND\@lab	86, 212, 215, 218, 332
COMMAND\@ldunboxmpfoot	230
COMMAND\@led@extranofeet	290
COMMAND\@ledinnote@command	233, 234
COMMAND\@lemma	108, 109
COMMAND\@lock	80, 239
COMMAND\@lopL	333
COMMAND\@lopR	333
COMMAND\@makecol	207, 208, 210, 343
COMMAND\@mpfnpos	183
COMMAND\@nl	86–90, 98, 215, 332, 333
COMMAND\@nl@reg	87, 287, 333, 338
COMMAND\@opXfeet	333

COMMAND\@opfeetX	343
COMMAND\@opxtrafeeti	343
COMMAND\@page	88, 215
COMMAND\@pend	333
COMMAND\@pendR	333
COMMAND\@ref	86, 95, 96, 99, 103
COMMAND\@ref@reg	95, 333
COMMAND\@reinserts	207–210, 343
COMMAND\@secondoftwo	56
COMMAND\@sect	274
COMMAND\@series	199
COMMAND\@set	90
COMMAND\@sidenotesepp	226
COMMAND\@sw	96, 111, 114
COMMAND\@tag	105, 106, 109
COMMAND\@tempcnta	67
COMMAND\@tempcntb	67
COMMAND\@toksa	72
COMMAND\@toksb	72
COMMAND\@xloop	136
COMMAND\@xympar	221, 343
COMMAND\Aendnote	14, 22
COMMAND\Afootfmt	139
COMMAND\Afootgroup	139
COMMAND\Afootnote	7, 13, 21, 22, 24, 106, 146, 165, 184, 194, 342
COMMAND\Afootstart	139
COMMAND\AtEveryPend	16, 120, 339, 341, 342
COMMAND\AtEveryPstart	16, 339, 341, 342
COMMAND\Bendnote	14, 21
COMMAND\Bfootnote	7, 13, 165, 184, 194
COMMAND\Centering	35
COMMAND\Cfootnote	165
COMMAND\Columns	68, 143
COMMAND\Dfootnote	165
COMMAND\Efootnote	165
COMMAND\NR	274
COMMAND\Pages	68, 208, 209
COMMAND\ProcessOptionsX	60
COMMAND\RaggedLeft	35
COMMAND\RaggedRight	35
COMMAND\Stanza	338
COMMAND\Waklam	262
COMMAND\X@doreinfeet	209, 343
COMMAND\XXXXXXfmt	286
COMMAND\XXXXXXfmt	286
COMMAND\Xafterlemmaseparator	33, 290
COMMAND\Xafternote	36, 289, 290
COMMAND\Xafternumber	31, 32, 290
COMMAND\Xafterrule	37, 184, 290, 338, 341
COMMAND\Xaftersymlinenum	32, 290

COMMAND\Xarrangement	29, 140, 201, 290
COMMAND\Xarrangement@footparagraph	144
COMMAND\Xarrangement@normal	140
COMMAND\Xarrangement@paragraph	144
COMMAND\Xbeforelemmaseparator	33, 290
COMMAND\Xbeforenotes	36, 184, 290, 338, 341
COMMAND\Xbeforenumber	31, 32, 290
COMMAND\Xbeforesymlinenum	32, 290
COMMAND\Xbhookendnote	291
COMMAND\Xbhooknote	35, 291, 292, 343
COMMAND\Xboxendlinum	32, 33, 291, 342
COMMAND\Xboxlinenum	32, 33, 291
COMMAND\Xboxlinenumalign	32, 33, 291, 342
COMMAND\Xboxstartlinenum	32, 33, 291, 342
COMMAND\Xboxsymlinenum	32, 291
COMMAND\Xcolalign	35, 341
COMMAND\Xdo@feet	208, 333, 343
COMMAND\Xend	198
COMMAND\XendXafterlemmaseparator	291
COMMAND\XendXbeforelemmaseparator	291
COMMAND\XendXinplaceoflemmaseparator	291
COMMAND\XendXlemmaseparator	291
COMMAND\XendXmorethantwolines	291
COMMAND\XendXtwolines	291
COMMAND\Xendafterlemmaseparator	33, 291
COMMAND\Xendafternote	37
COMMAND\Xendbeforelemmaseparator	33, 291
COMMAND\Xendbhooknote	35
COMMAND\Xendboxendlinum	33, 342
COMMAND\Xendboxlinenum	33, 291, 340
COMMAND\Xendboxlinenumalign	33, 291, 342
COMMAND\Xendboxstartlinenum	33, 342
COMMAND\Xendinplaceoflemmaseparator	33, 291
COMMAND\Xendinplaceofnumber	32, 343
COMMAND\Xendinsertsep@	191
COMMAND\Xendlemmadisablefontselection	34
COMMAND\Xendlemmaseparator	23, 33, 291
COMMAND\Xendletboxendlinum	291
COMMAND\Xendmorethantwolines	22, 30, 45, 291, 341
COMMAND\Xendmorethantwolinesappref	45
COMMAND\Xendnonumber	31, 343
COMMAND\Xendnote	187, 197, 198, 341
COMMAND\Xendnotefontsize	34, 290
COMMAND\Xendnotenumfont	34, 290
COMMAND\Xendparagraph	37, 338
COMMAND\Xendsep	37
COMMAND\Xendtwolines	22, 30, 45, 291, 341
COMMAND\Xendtwolinesappref	45
COMMAND\Xendtwolinesbutnotmore	30, 45, 341
COMMAND\Xendtwolinesbutnotmoreappref	45

COMMAND\Xendtwolinesonlyinsamepage	30, 45, 341
COMMAND\Xendtwolinesonlyinsamepageappref	45
COMMAND\Xfootfint	292
COMMAND\Xfootgroup	144
COMMAND\Xfootins	142
COMMAND\Xfootnote	43, 105, 286, 335, 339–341
COMMAND\Xfootstarts	144
COMMAND\Xhangindent	34
COMMAND\Xhsizethreecol	35, 291
COMMAND\Xhsizetwocol	35, 203, 291
COMMAND\Xinplaceoflemmaseparator	33, 291
COMMAND\Xinplaceofnumber	32, 291, 340, 342
COMMAND\Xinsertparafootsep	149, 150
COMMAND\Xledsetnormalparstuff	288, 289, 341
COMMAND\Xlemmadisablefontselection	34
COMMAND\Xlemmaseparator	33, 204, 206, 207, 291
COMMAND\Xmaxhnotes	37, 290, 291, 338, 340
COMMAND\Xmorethantwolines	22, 30, 45, 206, 219, 291, 340
COMMAND\Xmorethantwolinesappref	45, 219
COMMAND\Xnolemmaseparator	33, 207, 291
COMMAND\Xnonbreakableafternumber	31, 291, 336
COMMAND\Xnonumber	30, 31, 291
COMMAND\Xnonumberinfootnote	291
COMMAND\Xnotefontsize	34, 290
COMMAND\Xnotefontsize@(<s>	149, 153
COMMAND\Xnotenunfont	34, 290
COMMAND\Xnoteswidthliketwocolumns	37, 339
COMMAND\Xnumberonlyfirstinline	29, 30, 83, 203, 204, 206, 291, 335, 340
COMMAND\Xnumberonlyfirstintwolines	30, 291, 335
COMMAND\Xonlypstart	31, 291, 335, 340
COMMAND\Xparafootsep	36, 83, 290, 291
COMMAND\Xparafootsep@series	149
COMMAND\Xparindent	34, 341
COMMAND\Xpstart	31, 288, 291, 335, 340
COMMAND\Xpstarteverytime	31, 288, 291, 340
COMMAND\Xragged	36
COMMAND\Xstanza	31, 41
COMMAND\Xstanzaseparator	31
COMMAND\Xsymlinenum	30, 36, 290, 291, 342
COMMAND\Xtwolines	22, 30, 45, 162, 163, 203, 206, 219, 291, 340
COMMAND\Xtwolinesappref	45, 203, 219
COMMAND\Xtwolinesbutnotmore	30, 45, 291, 341
COMMAND\Xtwolinesbutnotmoreappref	45, 203
COMMAND\Xtwolinesonlyinsamepage	30, 45, 291, 341
COMMAND\Xtwolinesonlyinsamepageappref	45
COMMAND\Xtxtbeforenotes	36, 291
COMMAND\Xunvxh	147, 292
COMMAND\&	290
COMMAND\absline@num	80, 126
COMMAND\accent	104

COMMAND\actionlines@list	81, 127
COMMAND\actions@list	81
COMMAND\add@inserts	81, 134
COMMAND\add@inserts@next	134
COMMAND\add@penalties	126, 135
COMMAND\addcontentsline	269
COMMAND\addfootins	286, 289
COMMAND\addfootinsX	286, 289
COMMAND\advancelabel@refs	214
COMMAND\advanceline	19, 20, 82, 89, 100, 343
COMMAND\advancepageno	207
COMMAND\affixlin@num	226
COMMAND\affixline@num	128, 131, 132, 333
COMMAND\affixpstart@num	132
COMMAND\afterXrule	290
COMMAND\afterXsymlinenum	290
COMMAND\aftergroup	103, 107
COMMAND\afterlemmaseparator	290
COMMAND\afternote	290
COMMAND\afternoteX	36, 289, 290
COMMAND\afternumberinfootnote	290
COMMAND\afterruleX	338, 341
COMMAND\applabel	45, 215, 216, 219, 341
COMMAND\appref	45, 219, 220
COMMAND\apprefprefixmore	290
COMMAND\apprefprefixsingle	290
COMMAND\apprefwithpage	45, 219, 220
COMMAND\arrangementX	29, 168, 201, 290
COMMAND\arrangementX@normal	172
COMMAND\at@every@pend	120
COMMAND\autopar	15, 16, 117, 120, 121, 181, 334, 336, 337, 341
COMMAND\ballast	55
COMMAND\ballast@count	126, 135
COMMAND\baselineskip	29, 145, 149
COMMAND\before	17
COMMAND\beforeXnotes	290
COMMAND\beforeXsymlinenum	290
COMMAND\beforeeledchapter	8, 53, 270
COMMAND\beforelemmaseparator	290
COMMAND\beforenotesX	36, 37, 337, 338, 341
COMMAND\beforenumberinfootnote	290
COMMAND\begin	246
COMMAND\beginnumbering	14, 16, 17, 68, 69, 71, 79, 84, 98, 120, 186, 335, 338, 342, 343
COMMAND\bf	335
COMMAND\bfseries	34, 335
COMMAND\bhookXnote	291
COMMAND\bhooknoteX	35, 292, 343
COMMAND\body	240
COMMAND\bodyfootmarkA	27
COMMAND\boxXendlinenum	291

COMMAND\boxXendlinenumalign	291
COMMAND\boxXendstartlinenum	291
COMMAND\boxendlinenum	291
COMMAND\boxlinefootnote	160
COMMAND\boxlinenum	291
COMMAND\boxlinenumalign	291
COMMAND\boxstartlinenum	291
COMMAND\boxsymlinenum	291
COMMAND\break	147
COMMAND\brokenpenalty	135
COMMAND\centering	35
COMMAND\ch@ck@l@ck	334
COMMAND\ch@cksub@l@ck	131, 334
COMMAND\chapter	52, 271, 338, 341, 343
COMMAND\chaptermark	269
COMMAND\check@pb@in@verse	282
COMMAND\colalignX	35, 341
COMMAND\collect@body	247
COMMAND\colorbox	56
COMMAND\columns	37
COMMAND\columnwidth	145, 339
COMMAND\command names	203
COMMAND\copyright	104
COMMAND\correct@Xfootins@box	340
COMMAND\correct@footinsX@box	340
COMMAND\count	152
COMMAND\critex	334
COMMAND\critext	110, 285, 286, 289
COMMAND\csname	60, 113
COMMAND\ctab	262, 267
COMMAND\ctabtext	267
COMMAND\dcoll	256
COMMAND\def	58
COMMAND\detokenize	113
COMMAND\dimen	152
COMMAND\discretionary	146
COMMAND\displaywidowpenalty	135
COMMAND\do@actions	126, 127, 334
COMMAND\do@actions@fixedcode	333
COMMAND\do@actions@next	126, 127
COMMAND\do@ballast	126, 135
COMMAND\do@feetX	343
COMMAND\do@insidelinehook	336
COMMAND\do@line	81, 103, 119, 122, 124, 125, 134, 135, 239, 334, 336, 338
COMMAND\do@linehook	334
COMMAND\do@lockoff	82
COMMAND\do@lockon	82
COMMAND\dodoreinextrafeet	332
COMMAND\doendnotes	22, 191, 192, 292, 341
COMMAND\doendnotesbysection	23, 192, 199, 342

COMMAND\doinsidelinehook	20, 339
COMMAND\dolinehook	20, 339
COMMAND\doreinextrafeeti	343
COMMAND\doreinextrafeetii	343
COMMAND\doxtrafeet	207, 332
COMMAND\doxtrafeeti	343
COMMAND\doxtrafeetii	343
COMMAND\dummy@ref	103
COMMAND\edaftertab	51, 262, 263
COMMAND\edatleft	51, 260
COMMAND\edatright	51, 260
COMMAND\edbeforetab	51, 262
COMMAND\edfilldimen	261
COMMAND\edfont@info	109
COMMAND\edindex	48, 233, 236, 238, 251, 336, 339, 340, 343
COMMAND\edindexlab	48
COMMAND\edlabel	43–45, 104, 212, 214, 215, 217, 221, 232, 251, 332, 335–337, 340
COMMAND\edlineref	43, 212, 291, 340, 342
COMMAND\edmakelabel	44, 221
COMMAND\edpageref	43, 212, 216, 217, 221
COMMAND\edrowfill	262
COMMAND\edtabcolsep	255
COMMAND\edtext	6, 21–26, 38, 43–45, 49, 56, 81, 95, 96, 99, 102–110, 112, 114, 115, 215, 216, 218, 250, 252, 271, 285, 286, 289, 333, 334, 336, 338–342
COMMAND\edtext@level	342
COMMAND\edvertdots	52, 261
COMMAND\edvertline	51, 52, 260
COMMAND\elechapter	53
COMMAND\eled@sectioning@out	276
COMMAND\eledchapter	52, 289, 339, 343
COMMAND\eledchapter*	52
COMMAND\eledmac@error	332
COMMAND\eledsection	6, 13, 52, 103, 124, 270, 289, 340
COMMAND\eledsection*	52
COMMAND\eledsubsection	52, 289
COMMAND\eledsubsection*	52
COMMAND\eledsubsubsection	52, 289
COMMAND\eledsubsubsection*	52
COMMAND\eledxxx	8, 53, 270, 276, 338
COMMAND\eledxxxx	269
COMMAND\else	232, 270
COMMAND\empty	67, 129, 130, 213
COMMAND\end	246
COMMAND\end@lemmas	103
COMMAND\endashchar	38, 157
COMMAND\endgraf	119, 149, 181
COMMAND\endlock	19, 82, 101, 244
COMMAND\endminipage	228, 229, 332, 343
COMMAND\endnotes	341
COMMAND\endnumbering	14, 17, 68, 70, 71, 334, 342

COMMAND\endprint	187, 188, 198, 288
COMMAND\endstanzaextra	290
COMMAND\endsub	19, 82, 100
COMMAND\everypar	121
COMMAND\extensionchars	54, 68
COMMAND\f@x@l@cks	334
COMMAND>falseverse	289, 336, 338
COMMAND\fi	270
COMMAND\firstlinenum	18, 128, 334
COMMAND\firstsublinenum	18, 334
COMMAND\fix@page	87, 88, 333
COMMAND\flag@end	99, 108, 338
COMMAND\flag@start	99, 108, 338, 339
COMMAND\flagstanza	42
COMMAND\floatingpenalty	139, 340
COMMAND\flush@notes	135, 136
COMMAND\fnpos	183, 337
COMMAND\footfmt	138, 141
COMMAND\footfmt...	169
COMMAND\footfootmarkA	27
COMMAND\footfudgefactor	147
COMMAND\footfudgefiddle	56, 145, 332
COMMAND\footgroup	139
COMMAND\footins	142
COMMAND\footnormal	202, 290, 333
COMMAND\footnormalX	290
COMMAND\footnote	27, 55, 166, 167, 287, 333
COMMAND\footnote@lang	157
COMMAND\footnoteA	14, 27
COMMAND\footnoteB	14
COMMAND\footnoteC	21
COMMAND\footnoteE	27
COMMAND\footnoteX	7, 197
COMMAND\footnoteXmk	207
COMMAND\footnotelang@lua	137
COMMAND\footnotelang@poly	138
COMMAND\footnoteoption@	137
COMMAND\footnoterule	152
COMMAND\footnotesize	34
COMMAND\footparagraph	29, 145, 202, 290, 338
COMMAND\footparagraphX	177, 290, 338
COMMAND\footsplitskips	334, 340
COMMAND\footstart	139, 142, 152
COMMAND\footstrut	149
COMMAND\footthreecol	290
COMMAND\footthreecolX	290, 341
COMMAND\foottwocol	290
COMMAND\foottwocolX	290, 341
COMMAND\fulllines@	206
COMMAND\fullstop	38

COMMAND\get@edindex@hyperref	237
COMMAND\get@edindex@ledinnote@command	233
COMMAND\get@index@command	337
COMMAND\get@linelistfile	334
COMMAND\getline@num	125, 127
COMMAND\gl@p	73
COMMAND\global	86
COMMAND\globaldefs	86
COMMAND\hangindentX	341
COMMAND\hangingsymbol	41, 290, 334
COMMAND\hbox	146, 147
COMMAND\hfill	337
COMMAND\hidenumbering	20, 94, 341
COMMAND\hline	49
COMMAND\hrulefill	262
COMMAND\hspace	29, 143, 145–147, 153, 155, 182, 333, 339
COMMAND\hsizethreecol	291
COMMAND\hsizethreecolX	35
COMMAND\hsizetwocol	35, 291
COMMAND\hyperlinkR	236
COMMAND\hyperlinkformat	236
COMMAND\hyperlinkformatR	237
COMMAND\if@RTL	61
COMMAND\if@edtext@	339, 342
COMMAND\if@eled@sectioning	270
COMMAND\if@noneed@Footnote	99
COMMAND\ifbypage@	73
COMMAND\ifbypage@R	73
COMMAND\ifbypstart@	73
COMMAND\ifbypstart@R	73
COMMAND\iffirst@linenum@out@	97, 98
COMMAND\ifinserthangingsymbol	239
COMMAND\ifinstanza	239
COMMAND\ifstwofollowinglines	163, 164
COMMAND\ifl@d@Xmorethantwolines	161, 341
COMMAND\ifl@d@Xtwolines	161
COMMAND\ifl@d@dash	161
COMMAND\ifl@d@elin	161
COMMAND\ifl@d@esl	161
COMMAND\ifl@d@pnum	160
COMMAND\ifl@d@ssub	160
COMMAND\ifl@dend@X	198
COMMAND\ifl@dmemoir	332
COMMAND\ifl@dpaging	339
COMMAND\ifl@dpairing	67, 334
COMMAND\ifl@dprintingpages	340
COMMAND\ifl@dskipnumber	129
COMMAND\ifl@dstartendok	261
COMMAND\ifl@imakeidx	61
COMMAND\ifledRcol	68, 335

COMMAND\ifledRcol@	68, 338
COMMAND\iflemmacommand@	340
COMMAND\ifnoledgroup@	232
COMMAND\ifnoteschanged@	83
COMMAND\ifnumberedpar@	117
COMMAND\ifnumbering	68, 71
COMMAND\ifnumberingR	68, 335
COMMAND\ifnumberline	108, 129
COMMAND\ifpst@rted	334
COMMAND\ifpst@rtedL	69
COMMAND\ifseriesbefore	200
COMMAND\ifsublines@	80, 92
COMMAND\iftrue	342
COMMAND\ifvmode	214
COMMAND\ifxxx	270
COMMAND\ignorespaces	106, 107
COMMAND\imki@wrindexentry	61
COMMAND\immediate	97, 98, 186
COMMAND\indent	16, 120, 292
COMMAND\indtl@wrindexentry	61
COMMAND\initnumbering@quote	267, 343
COMMAND\initnumbering@reg	334
COMMAND\initnumbering@sectcmd	343
COMMAND\inplaceoflemmaseparator	291
COMMAND\inplaceofnumber	291
COMMAND\insert	133, 138, 141, 169
COMMAND\insert@count	95, 99, 106
COMMAND\insert@countR	106
COMMAND\insertthangingsymbol	337
COMMAND\insertlines@list	81, 95
COMMAND\insertparafootsepX	181
COMMAND\inserts@list	103, 117, 134, 146
COMMAND\interAfootnotelinepenalty	333
COMMAND\interfootnotelinepenalty	333
COMMAND\interlinepenalty	139
COMMAND\interparanoteglue	289
COMMAND\justifying	35
COMMAND\l@advance@parledegroupp@beforenormalnotes	343
COMMAND\l@d@@wrindexhyp	339
COMMAND\l@d@add	110
COMMAND\l@d@end	186, 198
COMMAND\l@d@nums	106, 108, 110, 160, 161
COMMAND\l@d@section	187
COMMAND\l@d@set	90, 101
COMMAND\l@dampcount	252
COMMAND\l@dbfnote	167, 333
COMMAND\l@dcheckstartend	261
COMMAND\l@dchset@num	90
COMMAND\l@dcolcount	252, 253
COMMAND\l@dcollect@@body	246

COMMAND\l@dcollect@body	246
COMMAND\l@dcsnote	338
COMMAND\l@dcsnotetext	124, 225
COMMAND\l@dcsnotetext@l	124, 225
COMMAND\l@dcsnotetext@r	124, 225
COMMAND\l@ddodoreinxtrafeet	208, 332
COMMAND\l@ddoxtrafeet	208, 332
COMMAND\l@demptyd@ta	334
COMMAND\l@dend@close	186
COMMAND\l@dend@open	186
COMMAND\l@dend@stuff	186
COMMAND\l@denvbody	246
COMMAND\l@dfeetbeginmini	333
COMMAND\l@dfeetendmini	333
COMMAND\l@dgetline@margin	334
COMMAND\l@dgetlock@disp	334
COMMAND\l@dgetref@num	218
COMMAND\l@dgetsidenote@margin	222, 334
COMMAND\l@dgobbeloptarg	339
COMMAND\l@dgobblearg	339
COMMAND\l@dgobbleoptarg	251
COMMAND\l@dlabel@parse	218
COMMAND\l@dld@ta	128, 130
COMMAND\l@dlp@rbox	226
COMMAND\l@dlsn@te	334
COMMAND\l@dlsnote	338
COMMAND\l@dmake@labels	214
COMMAND\l@dnumpsstartL	69, 334
COMMAND\l@dp@rsefootspec	161
COMMAND\l@dparsefootspec	161
COMMAND\l@dpush@begins	246
COMMAND\l@drd@ta	128, 130
COMMAND\l@dref@undefined	217
COMMAND\l@drsn@te	334
COMMAND\l@drsnote	338
COMMAND\l@dtabaddcols	261
COMMAND\l@dtabnoexpands	332
COMMAND\l@dumboxmpfoot	343
COMMAND\l@dunboxmpfoot	334
COMMAND\l@dzeropenalties	334, 339
COMMAND\l@pb	281
COMMAND\l@prev@nopb	281
COMMAND\l@prev@pb	281
COMMAND\l@reg	287
COMMAND\label	16, 44, 48, 213, 218
COMMAND\label@refs	213
COMMAND\labelstarttrue	16, 335
COMMAND\labelref@list	212, 215
COMMAND\language	146
COMMAND\last@page@num	333

COMMAND\lastbox	121
COMMAND\lastskip	100
COMMAND\leavevmode	16, 121
COMMAND\led@check@nopb	281
COMMAND\led@check@pb	281
COMMAND\led@nopb	281, 282
COMMAND\led@nopbnum	281
COMMAND\led@pb	281, 282
COMMAND\led@pb@macro	281
COMMAND\led@pbnum	281
COMMAND\ledRflag	236
COMMAND\ledchapter	289, 336
COMMAND\ledfootinsdim	290
COMMAND\ledinnernote	46, 223, 338
COMMAND\ledinnote	235, 288, 342
COMMAND\ledinnotemark	43, 288, 341
COMMAND\ledleftnote	46, 223
COMMAND\ledlinenum	78, 334
COMMAND\ledllfill	125
COMMAND\ledlsnotesep	46
COMMAND\ledlsnotewidth	46
COMMAND\lednopb	53, 54, 281
COMMAND\lednopbinverse	282
COMMAND\lednopbinversetrue	41, 54
COMMAND\lednopbnum	281
COMMAND\ledouternote	46, 223, 338
COMMAND\ledpb	53, 54, 281, 289
COMMAND\ledpbnum	281
COMMAND\ledpbsetting	281, 343
COMMAND\ledrightnote	46, 223
COMMAND\ledrsnotesep	46
COMMAND\ledrsnotewidth	46
COMMAND\ledsection	289
COMMAND\ledsectnomark	269
COMMAND\ledsectnotoc	269
COMMAND\ledsetnormalparstuff	288, 289, 341
COMMAND\ledsetnormalparstuff@common	181
COMMAND\ledsetnormalparstuffX	288, 289, 341
COMMAND\ledsidenote	46, 223, 225
COMMAND\ledsubsection	289
COMMAND\ledsubsubsection	289
COMMAND\ledxxx	338
COMMAND\left	51
COMMAND\leftctab	262
COMMAND\leftheadline	78
COMMAND\leftlinenum	19, 78, 332, 334
COMMAND\leftltab	262
COMMAND\leftnoteupfalse	46
COMMAND\leftpstartnum	132
COMMAND\leftrtab	262

COMMAND\leftsidenote	225
COMMAND\leftskip	142, 143, 146, 147
COMMAND\lemma	2, 22–26, 102, 105–107, 109, 110, 112, 285, 334, 335, 342, 343
COMMAND\lemmaseparator	291
COMMAND\let	24, 39, 244, 332
COMMAND\letboxXendendlinum	291
COMMAND\line	151, 154
COMMAND\line@list	81, 96, 108
COMMAND\line@list@stuff	69, 84, 98, 332, 334
COMMAND\line@list@version	86
COMMAND\line@margin	75, 130, 222
COMMAND\line@num	79, 80, 82, 128, 332
COMMAND\line@set	110
COMMAND\lineation	18, 74
COMMAND\linenum	22–24, 44, 45, 102, 110, 217, 218, 221, 285
COMMAND\linenum@out	97, 212, 215
COMMAND\linenumberlist	18, 67, 129, 130, 332
COMMAND\linenumberstyle	20, 78, 332
COMMAND\linenumincrement	18, 334
COMMAND\linenummargin	18, 75, 222
COMMAND\linenumr@p	78, 332, 334
COMMAND\linenumrep	78, 334
COMMAND\linenumsep	19, 46, 78, 223
COMMAND\lineref	212, 217, 221, 291, 340
COMMAND\list@clear	72
COMMAND\list@clearing@reg	334
COMMAND\list@create	72
COMMAND\lock@disp	77
COMMAND\lock@off	93
COMMAND\lock@on	92
COMMAND\lockdisp	19, 77
COMMAND\loop	136, 240
COMMAND\ltab	262, 263, 267
COMMAND\ltabtext	267
COMMAND\m@mmf@prepare	166
COMMAND\makeatletter	124
COMMAND\makehboxofhboxes	148, 149
COMMAND\makeindex	47, 236
COMMAND\makelabel	221
COMMAND\managestanza@modulo	241
COMMAND\marginpar	46, 55, 221, 222, 333
COMMAND\marginparwidth	46, 223
COMMAND\markboth	124
COMMAND\mathchardef	240
COMMAND\maxhXnotes	291
COMMAND\maxhnotesX	37, 290, 337, 338, 340, 341
COMMAND\maxlinesinpar@list	84
COMMAND\measurembdy	264
COMMAND\measuretbody	264
COMMAND\memorybreak	17

COMMAND\morenoexpands	56, 102, 104
COMMAND\morethantwolines	291
COMMAND\mpfnpos	183, 337
COMMAND\mpnormalfootgroup	333
COMMAND\mpnormalvfootnote	333
COMMAND\multfootsep	27, 166
COMMAND\multiplefootnotemark	166
COMMAND\musixtex	338
COMMAND\n@num	334, 341
COMMAND\n@num@ref	341
COMMAND\new@line	98, 333
COMMAND\newcommand	24, 58, 166, 214
COMMAND\newcommandx	24
COMMAND\newhookcommand@series	203, 204, 341
COMMAND\newhookcommand@series@reload	204
COMMAND\newhookcommand@toggle@reload	203, 339
COMMAND\newhooktoggle@series	203, 341
COMMAND\newif	341
COMMAND\newlinechar	198
COMMAND\newseries	28, 286, 289, 290
COMMAND\newseries@	192, 193, 201
COMMAND\newverse	41, 289, 338
COMMAND\next	240
COMMAND\next@action	84
COMMAND\next@actionline	84
COMMAND\next@insert	134
COMMAND\nl@regR	87
COMMAND\no@expands	56, 109, 332
COMMAND\noXlemmaseparator	291
COMMAND\nobreak	160
COMMAND\nocritical	193
COMMAND\noeledsec	53, 289
COMMAND\noendnotes	289
COMMAND\noexpand	287
COMMAND\nofamiliar	205
COMMAND\noindent	16, 120
COMMAND\nomk@	207
COMMAND\nonbreakableafternumber	291
COMMAND\nonum@	206
COMMAND\nonumberinfootnote	291
COMMAND\normal@footnotemarkX	169
COMMAND\normal@page@break	280
COMMAND\normal@pars	181
COMMAND\normalbfnoteX	334
COMMAND\normalbodyfootmarkX	169
COMMAND\normalfootfmt	39, 142, 149, 157, 187
COMMAND\normalfootfmtX	170
COMMAND\normalfootfootmarkX	170
COMMAND\normalfootgroup	143
COMMAND\normalfootgroupX	171

COMMAND\normalfootnoterule	139
COMMAND\normalfootstart	142, 146
COMMAND\normalfootstartX	170
COMMAND\normalvfootnote	141
COMMAND\normalvfootnoteX	169
COMMAND\notbool	270
COMMAND\notfontsetup	290
COMMAND\notfontsizeX	34, 290
COMMAND\notenumfont	290
COMMAND\notenumfontX	34, 290
COMMAND\notesXwidthliketwocolumns	291
COMMAND\noteswidthliketwocolumnsX	37, 291, 339, 341
COMMAND\num@lines	117, 135
COMMAND\numberlinefalse	17
COMMAND\numberlinetrue	17
COMMAND\numberonlyfirstinline	201, 291
COMMAND\numberonlyfirstintwolines	291
COMMAND\numberpstartfalse	16
COMMAND\numberpstarttrue	16, 31, 288, 334, 343
COMMAND\numberstanza	31
COMMAND\numberstanzafalse	41
COMMAND\numberstanzatrue	41
COMMAND\numlabfont	19, 38, 78, 79
COMMAND\none@line	117
COMMAND\nonlyXpstart	291
COMMAND\npage@action	82, 91
COMMAND\npage@start	82, 334
COMMAND\npagecontents	82
COMMAND\npagelinesep	47
COMMAND\npageno	207
COMMAND\npageparbreak	289
COMMAND\npageref	44, 217
COMMAND\npar	120, 121, 181
COMMAND\npar@line	117, 135
COMMAND\npara@footgroup	146
COMMAND\npara@footgroupX	180
COMMAND\npara@footsetup	145, 332
COMMAND\npara@footsetupX	178, 332, 339
COMMAND\npara@vfootnote	149
COMMAND\npara@vfootnoteX	179
COMMAND\nparafootfmt	148, 149
COMMAND\nparafootfmtX	179
COMMAND\nparafootftm	150
COMMAND\nparafootftmX	181
COMMAND\nparafootftmsep	290
COMMAND\nparafootsep	36, 291, 337, 342
COMMAND\nparafootsepX	36, 83, 290, 291
COMMAND\nparafootstart	146
COMMAND\nparafootstartX	178
COMMAND\nparavfootnote	146

COMMAND\parfillskip	148
COMMAND\parindentX	34
COMMAND\parshape	55
COMMAND\parskip	121
COMMAND\pausenumbering	17, 71, 84, 86, 121, 337, 339
COMMAND\penalty	148
COMMAND\pend	2, 6, 15–18, 20, 52, 53, 100, 103, 105, 111, 117, 119–122, 132, 133, 287, 337, 338
COMMAND\preXnotes	36, 184, 341
COMMAND\preXnotes@	142, 184, 335
COMMAND\prenotesX	36, 37, 185, 341
COMMAND\prepare@preXnotes	184
COMMAND\prev@nopb	281
COMMAND\prev@pb	281
COMMAND\prevlineX	83
COMMAND\prevpageX@num	83
COMMAND\print@Xfootnoterule	341
COMMAND\print@Xnotes	208, 209
COMMAND\print@Xnotes@forpages	340
COMMAND\print@eledsection	124
COMMAND\print@footnoteXrule	341
COMMAND\print@leftmargin@eledsection	270
COMMAND\print@line	122
COMMAND\print@notesX@forpages	340
COMMAND\print@rightmargin@eledsection	270
COMMAND\printendlines	188, 220, 332, 334
COMMAND\printlinefootnote	158, 159, 340
COMMAND\printlinefootnotearea	159, 160, 340
COMMAND\printlinefootnotenumbers	158
COMMAND\printlines	142, 157, 160, 162, 188, 220, 332, 334, 341
COMMAND\printnpnum	23, 288
COMMAND\printpstart	157
COMMAND\protect	104, 287
COMMAND\providecommand	166, 332
COMMAND\pstart	2, 6, 15–18, 20, 52, 53, 90, 100, 101, 105, 111, 117, 119–121, 124, 133, 287, 334, 335, 337–339, 341–343
COMMAND\pstartinfootnote	291
COMMAND\pstartinfootnoteeverytime	291
COMMAND\pstartnum	132
COMMAND\pstartref	43, 212, 217, 337
COMMAND\pstarts	335
COMMAND\raggedX	36
COMMAND\raggedleft	35
COMMAND\raggedright	35
COMMAND\raw@text	117
COMMAND\rbracket	33, 38
COMMAND\read@linelist	84, 85
COMMAND\ref	44, 48
COMMAND\relax	16, 90, 126, 134, 244, 251, 287
COMMAND\renewcommand	56, 290
COMMAND\resetprevline@	83

COMMAND\resetprevpage@	83
COMMAND\resumenumbers	17, 68, 71, 84, 86, 121, 334, 338, 339
COMMAND\right	51
COMMAND\rightctab	262
COMMAND\rightlinenum	19, 78, 332, 334
COMMAND\rightltab	263
COMMAND\rightnoteupfalse	46
COMMAND\rightrtab	263
COMMAND\rightrightnote	225
COMMAND\rightrightskip	142, 143, 146–148
COMMAND\rightstartnum	132
COMMAND\rigidbalance	151–154
COMMAND\robustify	30
COMMAND\rtab	262, 263, 267
COMMAND\rtabtext	264, 267
COMMAND\sameword	25, 26, 111–113, 115, 340, 342
COMMAND\sameword@inedtext	112
COMMAND\saweword	112
COMMAND\scriptsize	79
COMMAND\section	52, 334
COMMAND\section@num	68
COMMAND\sectionmark	269
COMMAND\select@lemmafont	38, 137
COMMAND\series	192
COMMAND\series@	192
COMMAND\seriesatbegin	28, 200, 341
COMMAND\seriesatend	28, 200, 342
COMMAND\set@line	108
COMMAND\set@line@action	82, 91
COMMAND\setapprefprefixmore	45, 290
COMMAND\setapprefprefixsingle	45, 290
COMMAND\setcommand@series	202
COMMAND\sethangingsymbol	41, 239, 290
COMMAND\sethanginsymbol	39
COMMAND\setistwoofollowinglines	163
COMMAND\setl@dlprbox	226
COMMAND\setline	19, 20, 82, 86, 90, 100, 104, 119, 343
COMMAND\setlinenum	20, 86, 90, 101, 332
COMMAND\setprintendlines	188, 190, 334
COMMAND\setprintlines	162, 163, 189, 334
COMMAND\setsidenotessep	47
COMMAND\setsidenotsep	290
COMMAND\setstanzaindent	241
COMMAND\setstanzaindents	40, 241, 287
COMMAND\setstanzapenalties	241
COMMAND\setstanzavalues	241
COMMAND\settoggle@series	201, 335, 339
COMMAND\showlemma	103, 333
COMMAND\showwordrank	27, 113
COMMAND\sidenote@margin	333

COMMAND\sidenotemargin	46, 333, 338
COMMAND\sidenoteseq	290
COMMAND\sidepstartnumtrue	16
COMMAND\skip	142
COMMAND\skipnumbering	20, 94, 101, 334, 342
COMMAND\skipnumbering@reg	342
COMMAND\small	34
COMMAND\special	11
COMMAND\splitmaxdepth	139, 153
COMMAND\splitoff	151
COMMAND\splittopskip	139, 153
COMMAND\stanza	20, 41, 244, 290
COMMAND\stanza@hang	243
COMMAND\stanza@line	243
COMMAND\stanzaindent	40, 241, 340
COMMAND\stanzaindent*	40
COMMAND\stanzaindentbase	240
COMMAND\stanzanumwrapper	42
COMMAND\start	53
COMMAND\startlock	19, 82, 101, 244
COMMAND\startstanzahook	290
COMMAND\startsub	19, 82, 100
COMMAND\strip@pt	146
COMMAND\strutbox	153
COMMAND\sub@action	82, 92
COMMAND\sub@lock	80
COMMAND\sub@off	89, 215
COMMAND\sub@on	89, 215
COMMAND\subline@num	79, 80, 82
COMMAND\sublinenum@rep	332
COMMAND\sublinenumberstyle	20, 78, 332
COMMAND\sublinenumincrement	18
COMMAND\sublinenumr@p	78, 332, 334
COMMAND\sublinenumrep	78, 334
COMMAND\sublineref	43, 212, 217
COMMAND\subsectionmark	269
COMMAND\sw@inthisedtext	105
COMMAND\sw@list@inedtext	109, 115
COMMAND\symlinenum	291
COMMAND\symplinenum	290
COMMAND\sza@penalty	243
COMMAND>tag	340
COMMAND\text	285
COMMAND\textcolor	56
COMMAND\the	332
COMMAND\thefootnoteA	27
COMMAND\thefootnoteX	336
COMMAND\thelabidx	238
COMMAND\thepage	87
COMMAND\thepstart	16

COMMAND\thepstartL	335
COMMAND\thepstartR	335
COMMAND\thestanza	41
COMMAND\this@line@list@version	97
COMMAND\threecolfootfmt	153
COMMAND\threecolfootfmtX	176
COMMAND\threecolfootgroup	152
COMMAND\threecolfootgroupX	176
COMMAND\threecolfootsetup	152
COMMAND\threecolfootsetupX	176
COMMAND\threecolvfootnote	153
COMMAND\threecolvfootnoteX	176
COMMAND\twocolfootfmtX	174
COMMAND\twocolfootgroupX	175
COMMAND\twocolfootsetupX	174
COMMAND\twocolvfootnoteX	174
COMMAND\twolines	201, 291
COMMAND\twolines@A	201
COMMAND\twolines@B	201
COMMAND\twolines@C	201
COMMAND\twolinesbutnotmore	291
COMMAND\twolinesonlyinsamepage	291
COMMAND\txtbeforeXnotes	291
COMMAND\unhbox	146
COMMAND\unpenalty	148, 149
COMMAND\unskip	148
COMMAND\unvxh	148, 292
COMMAND\unvxhX	292
COMMAND\upbracefill	262
COMMAND\usingcritext	286, 289
COMMAND\usingedtext	286, 289
COMMAND\vAfootnote	139
COMMAND\variant	24
COMMAND\vbox	119, 120, 147, 151, 183
COMMAND\vfootnote	138, 142, 143, 146, 153
COMMAND\vl@dbfnote	167, 333
COMMAND\vnumfootnoteX	334
COMMAND\vsplit	135
COMMAND\waklam	262
COMMAND\waklamec	262
COMMAND\wapunktel	262
COMMAND\wastricht	262
COMMAND\wrap@edcrossref	216, 339
COMMAND\x...	44
COMMAND\xdef	72, 244
COMMAND\xleft@appenditem	73, 103
COMMAND\xlineref	44
COMMAND\xpageref	44
COMMAND\xpstartref	44, 337
COMMAND\xright@appenditem	72, 73

COMMAND\sublineref	44
COMMAND\xxref	44, 218, 221, 337, 340, 341
COMMAND\zz@@@	332
ENVIRONMENTedarrayc	267
ENVIRONMENTedarrayl	267
ENVIRONMENTedarrayr	267
ENVIRONMENTedtabularc	267
ENVIRONMENTedtabularl	267
ENVIRONMENTedtabularr	267
ENVIRONMENTledgroup	230
ENVIRONMENTledgroupsize	231
PACKAGE(r)(e)ledmac	28
PACKAGEEledmac	9, 10, 27, 30, 58, 82, 235, 288, 289, 340–342
PACKAGEEledpar	341, 342
PACKAGEEtoolbox	60
PACKAGEParallel	293
PACKAGEReledmac	292
PACKAGEamsgen	245
PACKAGEamsmath	245, 246
PACKAGEbabel	57
PACKAGEbiblatex	55
PACKAGEbidi	61
PACKAGEccaption	67
PACKAGEcolor	56
PACKAGEedmac	1, 5, 9–12, 58, 160, 165, 213, 241, 285, 293, 332
PACKAGEedstanza	1, 12, 239
PACKAGEeledmac	1, 9, 12–14, 22, 25, 26, 35–37, 47, 48, 111, 166, 232, 235, 249, 271, 283, 286, 288–290, 336, 338, 340
PACKAGEeledpar	16, 37, 54, 67, 139, 269, 293, 335, 338–341
PACKAGEetoolbox	72, 111, 192, 201, 207, 225, 270, 280, 281
PACKAGEfootmisc	27, 57, 166, 293
PACKAGEhandout	339
PACKAGEhyperref	44, 213, 237, 274, 337–339
PACKAGEifluatex	60
PACKAGEifxetex	60
PACKAGEimakeidx	47, 55, 61, 232, 235, 289, 336–338, 340
PACKAGEindextool	289
PACKAGEindextools	47, 55, 61, 232, 235, 289, 340
PACKAGEinputenc	113
PACKAGEledarab	57
PACKAGEledmac	1, 9, 12, 57, 72, 235, 285, 286, 289
PACKAGEledpar	57
PACKAGEMemoir	61, 235, 289, 293, 339
PACKAGEMorewrites	55
PACKAGEMusixtex	338
PACKAGEpolyglossia	38, 57, 107, 138, 157
PACKAGERagged2e	60
PACKAGEreledmac	1, 2, 9, 10, 12–14, 17, 18, 20, 21, 24, 25, 28, 32, 34, 38, 41, 44, 45, 47–49, 53, 55–59, 73, 76, 81, 82, 85, 86, 89, 97, 104, 133, 140, 143, 147, 165, 166, 187, 193, 196, 201, 207, 216, 219, 235, 249, 269, 270, 282, 289, 290, 292, 343

PACKAGEeledpar	1,
3, 5, 7, 13, 37, 53–55, 57, 59, 67, 73, 84, 89, 106, 141, 143, 182, 193, 207–209, 232, 239	
PACKAGESuffix	60
PACKAGEtabmac	1, 12, 293
PACKAGEuninormalize	25
PACKAGExargs	24, 60
PACKAGExkeyval	59
PACKAGExstring	60, 237

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@denbody	1
Adelard II	11
\advancelabel@refs	1
\advanceline	1, 19
\advancepageno	1
\Aendnote	22
\affixline@num	1
\affixpstart@num	1
\affixside@note	1
\Afootnote	22
\afternoteX	36
\afterruleX	37
\ampersand	1, 42
\applabel	1, 45
\appref	1, 45
\apprefwithpage	1, 45
\arrangementX	1, 29
\arrangementX@normal	1
\arrangementX@threecol	1
\arrangementX@twocol	1
\at@every@pend	1
\AtEveryPend	1, 16
\AtEveryPstart	1, 16
\autopar	1, 15

B

\ballast	55
\ballast@count	1
Beeton, Barbara Ann Neuhaus Friend	16
\beforeeledchapter	1
\beforenotesX	36
\beginnumbering	1, 14
\Bendnote	22

<code>\Bfootnote</code>	22
<code>\bhooknoteX</code>	35
<code>\bodyfootmarkA</code>	27
<code>\boxfootnotenumbers</code>	1
Bredon, Simon	11
Breger, Herbert	11, 12, 249
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
Copernicus, Nicolaus	11
<code>\critext</code>	285
<code>\ctab</code>	1
<code>\ctabtext</code>	1

D

Dekker, Dirk-Jan	56
<code>\Dendnote</code>	22
<code>\Dfootnote</code>	22
<code>\disable@familiarnotes</code>	1
<code>\disable@notes</code>	1
<code>\disable@sidenotes</code>	1
<code>\disablel@dtabfeet</code>	1
<code>\do@actions</code>	1
<code>\do@actions@fixedcode</code>	1
<code>\do@actions@next</code>	1
<code>\do@ballast</code>	1
<code>\do@feetX</code>	1
<code>\do@insidelinehook</code>	1
<code>\do@line</code>	1
<code>\do@linehook</code>	1

<code>\do@lockoff</code>	1
<code>\do@lockoffL</code>	1
<code>\do@lockon</code>	1
<code>\do@lockonL</code>	1
<code>\doedindexlabel</code>	1
<code>\doendnotes</code>	1
<code>\doendnotesbysection</code>	1, 22
<code>\doinsidelinehook</code>	1, 20
<code>\dolinehook</code>	1, 20
<code>\dosplits</code>	1
Downes, Michael	56, 146, 148
<code>\doxtrafeet</code>	1
<code>\dummy@edtext</code>	1
<code>\dummy@edtext@showlemma</code>	1
<code>\dummy@ref</code>	1

E

<code>\edaftertab</code>	1, 51, 262
<code>edarrayc</code> (environment)	49
<code>edarrayl</code> (environment)	49
<code>edarrayr</code> (environment)	49
<code>\edatleft</code>	1, 51
<code>\edatright</code>	1, 51
<code>\edbeforetab</code>	1, 51, 262
<code>\edfilldimen</code>	1
<code>\edfont@info</code>	1
<code>\EDINDEX</code>	1
<code>\edindex</code>	1, 47
<code>\edindexlab</code>	1, 48
<code>\EDLABEL</code>	1
<code>\edlabel</code>	1, 43
<code>\edlineref</code>	1, 43
<code>\edmakelabel</code>	1, 44
<code>\edpageref</code>	1, 43
<code>\edrowfill</code>	1, 50
<code>\EDTAB</code>	1
<code>\edtabcolsep</code>	1, 49
<code>\EDTABINDENT</code>	1
<code>\edtabindent</code>	1
<code>\EDTABtext</code>	1
<code>edtabularc</code> (environment)	49
<code>edtabularl</code> (environment)	49
<code>edtabularr</code> (environment)	49
<code>\EDTEXT</code>	1
<code>\edtext</code>	1, 21
<code>\edvertdots</code>	1, 51
<code>\edvertline</code>	1, 51
<code>\Eendnote</code>	22
<code>\Efootnote</code>	22
<code>\eled@chapter</code>	1

<code>\eled@section</code>	1
<code>\eled@sectioning@out</code>	1
<code>\eled@subsection</code>	1
<code>\eled@subsubsection</code>	1
<code>\eledchapter</code>	1
<code>\eledchapter*</code>	1
<code>\eledsection</code>	1
<code>\eledsection*</code>	1
<code>\eledsubsection</code>	1
<code>\eledsubsection*</code>	1
<code>\eledsubsubsection</code>	1
<code>\eledsubsubsection*</code>	1
<code>\enablel@dtabfeet</code>	1
<code>\end@lemmas</code>	1
<code>\endashchar</code>	1, 38
<code>\endline@num</code>	1
<code>\endlock</code>	1, 19
<code>\endminipage</code>	1
<code>\endnumbering</code>	1, 14
<code>\endpage@num</code>	1
<code>\endprint</code>	1
<code>\endquotation</code>	1
<code>\endquote</code>	1
<code>\endsub</code>	1, 19
<code>\endsubline@num</code>	1
environments:	
<code>edarrayc</code>	49
<code>edarrayl</code>	49
<code>edarrayr</code>	49
<code>edtabularc</code>	49
<code>edtabularl</code>	49
<code>edtabularr</code>	49
<code>ledgroup</code>	42
<code>ledgroupsize</code>	42
<code>minipage</code>	42
<code>Euclid</code>	11
<code>\extensionchars</code>	1, 54

F

<code>\f@x@l@cks</code>	1
Fairbairns, Robin	27
<code>\first@linenum@out@false</code>	1
<code>\first@linenum@out@true</code>	1
<code>\firstlinenum</code>	1, 18
<code>\firstseriesX@</code>	1
<code>\firstsublinenum</code>	1, 18
<code>\firstXseries@</code>	1
<code>\fix@page</code>	1
<code>\flag@end</code>	1
<code>\flag@start</code>	1

\flagstanza	1, 42
\flush@notes	1
\fnpos	1, 28
Folkerts, Menso	11
\footfootmarkA	27
\footfudgefiddle	1, 56
\footnoteA	27
\footnoteB	27
\footnoteC	27
\footnoteD	27
\footnoteE	27
\footnotelang@lua	1
\footnotelang@poly	1
\footnoteoptions@	1
\footsplitskips	1
\fulllines@	1
\fullstop	1, 38

G

Gädeke, Nora	11
\get@edindex@hyperref	1
\get@edindex@ledinnote@command	1
\get@index@command	1
\get@linelistfile	1
\get@sw@txt	1
\getline@num	1
\gl@p	1

H

\h@num	1
\hangindentX	34
\hidenumbering	1, 20
\Hilfsbox	1
\hilfsbox	1
\hilfscount	1
\HILFSskip	1
\Hilfsskip	1
\hilfsskip	1
\hsizethreecolX	35
\hsizetwocolX	35
\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

<code>\if@noeled@sec</code>	1
<code>\if@noneed@Footnote</code>	1
<code>\if@RTL</code>	1
<code>\ifautopar@pause</code>	1
<code>\ifbypage@</code>	1
<code>\ifbypage@R</code>	1
<code>\ifbypstart@</code>	1
<code>\ifbypstart@R</code>	1
<code>\ifeledmaccompat@</code>	1
<code>\iffirst@linenum@out@</code>	1
<code>\ifinserthangingsymbol</code>	1
<code>\ifinstanza</code>	1
<code>\ifl@d@dash</code>	1
<code>\ifl@d@elin</code>	1
<code>\ifl@d@esl</code>	1
<code>\ifl@d@pnun</code>	1
<code>\ifl@d@ssub</code>	1
<code>\ifl@d@Xmorethantwolines</code>	1
<code>\ifl@d@Xtwolines</code>	1
<code>\ifl@dend@X</code>	1
<code>\ifl@dhidenumber</code>	1
<code>\ifl@dmemoir</code>	1
<code>\ifl@dpaging</code>	1
<code>\ifl@dpairing</code>	1
<code>\ifl@dprintingcolumns</code>	1
<code>\ifl@dprintingpages</code>	1
<code>\ifl@dskipnumber</code>	1
<code>\ifl@dskipversenumber</code>	1
<code>\ifl@dstartendok</code>	1
<code>\ifl@imakeidx</code>	1
<code>\ifl@indextools</code>	1
<code>\ifledfinal</code>	1, 54
<code>\ifledgroupnotesL@</code>	1
<code>\ifledgroupnotesR@</code>	1
<code>\iflednopbinverse</code>	1
<code>\ifledRcol</code>	1
<code>\ifledRcol@</code>	1
<code>\ifnocritical@</code>	1
<code>\ifnoend@</code>	1
<code>\ifnofamiliar@</code>	1
<code>\ifnoledgroup@</code>	1
<code>\ifnoquotation@</code>	1
<code>\ifnoteschanged@</code>	1
<code>\ifnumberedpar@</code>	1
<code>\ifnumbering</code>	1
<code>\ifnumberingR</code>	1
<code>\ifnumberline</code>	1
<code>\ifnumberstanza</code>	1
<code>\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>\insertthangingsymbol</code>	1
<code>\insertlines@list</code>	1
<code>\inserts@list</code>	1

J

Jayaditya	11
-----------------	----

K

Kabelschacht, Alois	136
---------------------------	-----

L

<code>\l@advance@parledgroup@beforenormalnotes</code>	1
<code>\l@d@add</code>	1
<code>\l@d@nums</code>	1
<code>\l@d@section</code>	1
<code>\l@d@set</code>	1
<code>\l@d@Xend</code>	1
<code>\l@dampcount</code>	1
<code>\l@dbfnote</code>	1
<code>\l@dcheckcols</code>	1
<code>\l@dcheckstartend</code>	1
<code>\l@dchset@num</code>	1
<code>\l@dcolcount</code>	1
<code>\l@dcollect@@body</code>	1
<code>\l@dcollect@body</code>	1
<code>\l@dcolwidth</code>	1
<code>\l@dcsnote</code>	1
<code>\l@dcsnotetext</code>	1
<code>\l@dcsnotetext@l</code>	1
<code>\l@dcsnotetext@r</code>	1
<code>\l@ddodoreinextrafeet</code>	1
<code>\l@dedbeginmini</code>	1
<code>\l@dedendmini</code>	1
<code>\l@emptyd@ta</code>	1
<code>\l@dend@close</code>	1
<code>\l@dend@open</code>	1
<code>\l@dend@stuff</code>	1
<code>\l@dend@Xfalse</code>	1
<code>\l@dend@Xtrue</code>	1

<code>\@denvbody</code>	1
<code>\@dfambeginmini</code>	1
<code>\@dfamendmini</code>	1
<code>\@dfeetbeginmini</code>	1
<code>\@dfeetendmini</code>	1
<code>\@dgetline@margin</code>	1
<code>\@dgetlock@disp</code>	1
<code>\@dgetref@num</code>	1
<code>\@dgetsidenote@margin</code>	1
<code>\@dgobblearg</code>	1
<code>\@dlabel@parse</code>	1
<code>\@dld@ta</code>	1
<code>\@dlp@rbox</code>	1
<code>\@dlsn@te</code>	1
<code>\@dlsnote</code>	1
<code>\@dmake@labels</code>	1
<code>\@dmodforedtext</code>	1
<code>\@dnullfills</code>	1
<code>\@dnumpstartsL</code>	1
<code>\@doldold@footnotetext</code>	1
<code>\@dp@rsefootspec</code>	1
<code>\@dpagingfalse</code>	1
<code>\@dpagingtrue</code>	1
<code>\@dpairingfalse</code>	1
<code>\@dpairingtrue</code>	1
<code>\@dparsedendline</code>	1
<code>\@dparsedendpage</code>	1
<code>\@dparsedendsub</code>	1
<code>\@dparsedstartline</code>	1
<code>\@dparsedstartpage</code>	1
<code>\@dparsedstartsub</code>	1
<code>\@dparsefootspec</code>	1
<code>\@dprintingcolumnfalse</code>	1
<code>\@dprintingcolumntrue</code>	1
<code>\@dprintingpagesfalse</code>	1
<code>\@dprintingpagetrue</code>	1
<code>\@dpush@begins</code>	1
<code>\@drd@ta</code>	1
<code>\@dref@undefined</code>	1
<code>\@drestorefills</code>	1
<code>\@drestoreforedtext</code>	1
<code>\@drp@rbox</code>	1
<code>\@drsn@te</code>	1
<code>\@drsnote</code>	1
<code>\@dsetmaxcolwidth</code>	1
<code>\@dskipnumberfalse</code>	1
<code>\@dskipnumbertrue</code>	1
<code>\@dtabaddcols</code>	1
<code>\@dtabnoexpands</code>	1
<code>\@dunboxmpfoot</code>	1

<code>\l@dunhbox@line</code>	1
<code>\l@dzeropenalties</code>	1
<code>\label</code>	44
<code>\labelpstartfalse</code>	1
<code>\labelpstarttrue</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@@doclearpage</code>	1
<code>\led@error@fail@patch@@iiiminipage</code>	1
<code>\led@error@fail@patch@@makecol</code>	1
<code>\led@error@fail@patch@@reinserts</code>	1
<code>\led@error@fail@patch@endminipage</code>	1
<code>\led@error@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>	42
<code>ledgroupsized (environment)</code>	42
<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, 46
<code>\ledlsnotesep</code>	1, 46
<code>\ledlsnotewidth</code>	1, 46
<code>\lednopb</code>	1, 53
<code>\lednopbinversetrue</code>	54
<code>\lednopbnum</code>	1
<code>\ledouternote</code>	46
<code>\ledouterote</code>	1
<code>\ledpb</code>	1, 53
<code>\ledpbnum</code>	1
<code>\ledpbsetting</code>	1, 54
<code>\ledrightnote</code>	1, 46
<code>\ledrlfill</code>	1
<code>\ledrsnotefontsetup</code>	1, 46
<code>\ledrsnotesep</code>	1, 46
<code>\ledrsnotewidth</code>	1, 46
<code>\ledsectnomark</code>	1
<code>\ledsectnotoc</code>	1
<code>\ledsetnormalparstuff@common</code>	1
<code>\ledsetnormalparstuffX</code>	1
<code>\ledsidenote</code>	1, 46
<code>\leftctab</code>	1
<code>\leftlinenum</code>	1, 19
<code>\leftltab</code>	1

<code>\leftnoteupfalse</code>	46
<code>\leftstartnum</code>	1
<code>\lefttab</code>	1
Leibniz	11
<code>\lemma</code>	1, 23
<code>\letsforverteilen</code>	1
<code>\line@list</code>	1
<code>\line@list@stuff</code>	1
<code>\line@list@version</code>	1
<code>\line@margin</code>	1
<code>\line@num</code>	1
<code>\line@set</code>	1
<code>\lineation</code>	1, 18
<code>\linenum</code>	1, 23
<code>\linenum@out</code>	1
<code>\linenumberlist</code>	1, 18
<code>\linenumberstyle</code>	1, 20
<code>\linenumincrement</code>	1, 18
<code>\linenummargin</code>	1, 18
<code>\linenumr@p</code>	1
<code>\linenumrep</code>	1
<code>\linenumsep</code>	1, 19
<code>\list@clear</code>	1
<code>\list@clearing@reg</code>	1
<code>\list@create</code>	1
<code>\lock@disp</code>	1
<code>\lock@off</code>	1
<code>\lock@on</code>	1
<code>\lockdisp</code>	1, 19
Lorch, Richard	11
<code>\ltab</code>	1
<code>\ltabtext</code>	1
Luecking, Dan	60

M

<code>\m@mmf@check</code>	1
<code>\m@mmf@prepare</code>	1
<code>\M@sect</code>	1
<code>\makehboxofhboxes</code>	1
<code>\managestanza@modulo</code>	1
<code>\maxhnotesX</code>	37
Mayer, Gyula	11
<code>\measuretbody</code>	1
<code>\measuremcell</code>	1
<code>\measuremrow</code>	1
<code>\measuretbody</code>	1
<code>\measuretcell</code>	1
<code>\measuretrow</code>	1
Middleton, Thomas	11, 80
<code>minipage</code> (environment)	42

Mittelbach, Frank	10, 11
\morenoexpands	1, 56
\mpfnpos	1, 28
\mpnormalfootgroup	1
\mpnormalfootgroupX	1
\mpnormalvfootnote	1
\mpnormalvfootnoteX	1
\mppara@footgroupX	1
\mppara@vfootnoteX	1
\mpparafootgroup	1
\mpparavfootnote	1
\mpthreecolfootgroup	1
\mpthreecolfootgroupX	1
\mpthreecolfootsetup	1
\mpthreecolfootsetupX	1
\mptwocolfootgroup	1
\mptwocolfootgroupX	1
\mptwocolfootsetup	1
\mptwocolfootsetupX	1
\multfootsep	1, 27
\multiplefootnotemarker	1

N

\n@num	1
\n@num@stanza	1
\new@line	1
\newhookcommand@series	1
\newhookcommand@series@reload	1
\newhooktoggle@series	1
\newhooktoggle@series@reload	1
\newseries@	1
\newverse	1
\NEXT	1
\no@expands	1
\noeledsec	53
\nomk@	1
\nonum@	1
\normal@footnotemarkX	1
\normal@page@break	1
\normal@pars	1
\normalbfnoteX	1
\normalbodyfootmarkX	1
\normalfootfmt	1
\normalfootfmtX	1
\normalfootfootmarkX	1
\normalfootgroup	1
\normalfootgroupX	1
\normalfootnoterule	1
\normalfootnoteruleX	1
\normalfootstart	1

<code>\normalfootstartX</code>	1
<code>\normalvfootnote</code>	1
<code>\normalvfootnoteX</code>	1
<code>\nosep@</code>	1
<code>\notefontsizeX</code>	34
<code>\notenumfontX</code>	34
<code>\noteschanged@false</code>	1
<code>\noteschanged@true</code>	1
<code>\noteswidthliketwocolumnsX</code>	37
<code>\nulledindex</code>	1
<code>\nullsetzen</code>	1
<code>\num@lines</code>	1
<code>\numberedpar@false</code>	1
<code>\numberedpar@true</code>	1
<code>\numberingfalse</code>	1
<code>\numberingtrue</code>	1
<code>\numberlinefalse</code>	17
<code>\numberlinetrue</code>	17
<code>\numberpstartfalse</code>	1, 16
<code>\numberpstarttrue</code>	1, 16
<code>\numberstanzafalse</code>	41
<code>\numberstanzatrue</code>	41
<code>\numlabfont</code>	1, 38

O

<code>\old@hsize</code>	1
<code>\one@line</code>	1

P

<code>\page@action</code>	1
<code>\page@num</code>	1
<code>\pagelinesep</code>	1, 47
<code>\pageno</code>	1
<code>\pageref</code>	44
<code>\par@line</code>	1
<code>\para@footgroupX</code>	1
<code>\para@footsetup</code>	1
<code>\para@footsetupX</code>	1
<code>\para@vfootnoteX</code>	1
<code>\parafootfmt</code>	1
<code>\parafootfmtX</code>	1
<code>\parafootgroup</code>	1
<code>\parafootsepX</code>	36
<code>\parafootstart</code>	1
<code>\parafootstartX</code>	1
<code>\paravfootnote</code>	1
<code>\parindentX</code>	34
<code>\pausenumbering</code>	1, 17
<code>\pend</code>	1, 15
Plato of Tivoli	11

<code>\postbodyfootmark</code>	1
<code>\prebodyfootmark</code>	1
<code>\prenotesX</code>	36, 37
<code>\prepare@edindex@fornote</code>	1
<code>\prepare@prenotesX</code>	1
<code>\prepare@preXnotes</code>	1
<code>\prev@nopb</code>	1
<code>\prev@pb</code>	1
<code>\prevpage@num</code>	1
<code>\preXnotes</code>	1, 36
<code>\preXnotes@</code>	1
<code>\print@eledsection</code>	1
<code>\print@footnoteXrule</code>	1
<code>\print@leftmargin@eledsection</code>	1
<code>\print@line</code>	1
<code>\print@notesX</code>	1
<code>\print@rightmargin@eledsection</code>	1
<code>\print@Xfootnoterule</code>	1
<code>\print@Xnotes</code>	1
<code>\printendlines</code>	1
<code>\printlineendnotearea</code>	1
<code>\printlinefootnote</code>	1
<code>\printlinefootnotearea</code>	1
<code>\printlinefootnotenumbers</code>	1
<code>\printlines</code>	1
<code>\printnpnum</code>	1
<code>\printpstart</code>	1
<code>\printXafternumber</code>	1
<code>\printXbeforenumber</code>	1
<code>\pst@rtedLfalse</code>	1
<code>\pst@rtedLtrue</code>	1
<code>\pstart</code>	1, 15
<code>\pstarteref</code>	1
<code>\pstartnum</code>	1
<code>\pstartref</code>	43

Q

<code>\quotation</code>	1
<code>\quote</code>	1

R

<code>\raggedX</code>	36
<code>\raw@text</code>	1
<code>\rbracket</code>	1, 38
<code>\read@linelist</code>	1
<code>\ref</code>	44
<code>\Relax</code>	1
<code>\reledmac@error</code>	1
<code>\reledmac@warning</code>	1
<code>\removehboxes</code>	1

\resetprevline@	1, 83
\resetprevpage@	1
\resetprevpage@num	83
\restore@familiarnotes	1
\restore@notes	1
\restore@sidenotes	1
\resumenumbering	1, 17
\rightctab	1
\rightlinenum	1, 19
\rightltab	1
\rightnoteupfalse	46
\rightrtab	1
\rightstartnum	1
\rigidbalance	1
\rtab	1
\rtabtext	1

S

Sacrobosco	11
\sameword	1, 25
\sameword@inedtext	1
Schöpf, Rainer	11
\section@num	1
\select@lemmafонт	1
\select@lemmafонт	1, 38
\series	1
\seriesatbegin	1, 28
\seriesatend	1, 28
\set@line	1
\set@line@action	1
\setapprefprefixmore	1, 45
\setapprefprefixsingle	1, 45
\setcommand@series	1
\sethangingsymbol	1, 41
\setistwofollowinglines	1
\setl@dlp@rbox	1
\setl@drpr@box	1
\setline	1, 19
\setlinenum	1, 20
\setmcellcenter	1
\setmcellleft	1
\setmcellright	1
\setmrowcenter	1
\setmrowleft	1
\setmrowright	1
\setnoteswidthliketwocolumnsX@	1
\setnotesXpositionliketwocolumns@	1
\setprintendlines	1
\setprintlines	1
\setsidenotessep	47

<code>\setstanzaindents</code>	1, 39
<code>\setstanzapenalties</code>	1, 40
<code>\setstanzavalues</code>	1
<code>\settccllcenter</code>	1
<code>\settccllleft</code>	1
<code>\settccllright</code>	1
<code>\settoggle@series</code>	1
<code>\setthrowcenter</code>	1
<code>\setthrowleft</code>	1
<code>\setthrowright</code>	1
<code>\setXnotespositionliketwocolumns@</code>	1
<code>\setXnoteswidthliketwocolumns@</code>	1
<code>\showlemma</code>	1, 54
<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, 49
<code>\spreadtext</code>	1, 49
<code>\stanza</code>	1, 39
<code>\stanza@count</code>	1
<code>\stanza@chang</code>	1
<code>\stanza@line</code>	1
<code>\stanzaindent</code>	1, 40
<code>\stanzaindent*</code>	1, 40
<code>\stanzaindentbase</code>	1, 39
<code>\stanzanumwrapper</code>	1, 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, 43
<code>\sublines@false</code>	1
<code>\sublines@true</code>	1
<code>\sublock@disp</code>	1
<code>\sublockdisp</code>	1
Sullivan, Wayne	11, 12, 55, 67, 71, 146, 148, 213, 239
<code>\sza@penalty</code>	1

T

<code>\tabHilfbox</code>	1
<code>\tabhilfbox</code>	1
<code>\theadcolcount</code>	1
<code>\theendpageline</code>	1
<code>\thefootnoteA</code>	27
Theodosius	11
<code>\thepageline</code>	1
<code>\thepstart</code>	1, 16
<code>\thestanza</code>	1, 41
<code>\thestartpageline</code>	1
<code>\this@line@list@version</code>	1
<code>\threecolfootfmt</code>	1
<code>\threecolfootfmtX</code>	1
<code>\threecolfootgroup</code>	1
<code>\threecolfootgroupX</code>	1
<code>\threecolfootsetup</code>	1
<code>\threecolfootsetupX</code>	1
<code>\threecolvfootnote</code>	1
<code>\threecolvfootnoteX</code>	1
<code>\twocolfootfmt</code>	1
<code>\twocolfootfmtX</code>	1
<code>\twocolfootgroup</code>	1
<code>\twocolfootgroupX</code>	1
<code>\twocolfootsetup</code>	1
<code>\twocolfootsetupX</code>	1
<code>\twocolvfootnote</code>	1
<code>\twocolvfootnoteX</code>	1

U

<code>\unvxhX</code>	1
----------------------------	---

V

Vamana	11
<code>\variab</code>	1
<code>\vbfnoteX</code>	1
<code>\vl@dbfnote</code>	1
<code>\vl@dcnote</code>	1
<code>\vl@dlsnote</code>	1
<code>\vl@drsnote</code>	1
<code>\vnumfootnoteX</code>	1

W

Whitney, Ron	11
<code>\wrap@edcrossref</code>	1
Wujastyk, Dominik	10

X

<code>\X@doreinfeet</code>	1
<code>\Xafterlemmaseparator</code>	33

\Xafternote	36
\Xafternumber	31
\Xafterrule	37
\Xaftersymlinenum	32
\Xarrangement	1, 29
\Xarrangement@normal	1
\Xarrangement@paragraph	1
\Xarrangement@threecol	1
\Xarrangement@twocol	1
\Xbeforelemmaseparator	33
\Xbeforenotes	36
\Xbeforenumber	31
\Xbeforesymlinenum	32
\Xhooknote	35
\Xboxlinenum	32
\Xboxlinenumalign	32
\Xboxsymlinenum	32
\Xdo@feet	1
\xedindex	1
\xedlabel	1
\xedtext	1
\Xendafterlemmaseparator	33
\Xendafternote	37
\Xendbeforelemmaseparator	33
\Xendbhooknote	35
\Xendboxendlinenumalign	33
\Xendboxlinenum	33
\Xendboxlinenumalign	33
\Xendboxstartlinenumalign	33
\Xendinplaceoflemmaseparator	33
\Xendinplaceofnumber	32
\Xendlemmadisablefontselection	34
\Xendlemmaseparator	33
\Xendmorethantwolines	30
\Xendmorethantwolinesapprefwithpage	45
\Xendnonumber	31
\Xendnotefontsize	34
\Xendnotenumfont	34
\Xendparagraph	37
\Xendsep	37
\Xendtwolines	30
\Xendtwolinesapprefwithpage	45
\Xendtwolinesbutnotmore	30
\Xendtwolinesbutnotmoreapprefwithpage	45
\Xendtwolinesonlyinsamepageapprefwithpage	45
\Xhangindent	34
\Xhsizethreecol	35
\Xhsizetwocol	35
\Xinplaceoflemmaseparator	33
\Xinplaceofnumber	32

\Xinsertparafootsep	1
\Xledsetnormalparstuff	1
\xleft@appenditem	1
\Xlemmadisablefontselection	34
\Xlemmaseparator	33
\xlineref	1, 44
\Xmaxhnotes	37
\Xmorethantwolines	30
\Xmorethantwolinesappref	45
\Xnolemmaseparator	1, 33
\Xnonbreakableafternumber	31
\Xnonumber	30
\Xnotefontsize	34
\Xnotenumfont	34
\Xnoteswidthliketwocolumns	37
\Xnumberonlyfirstinline	29
\Xnumberonlyfirstintwolines	30
\Xonlypstart	31
\xpageref	1, 44
\Xparafootsep	36
\Xparindent	34
\Xpstart	31
\Xpstarteverytime	31
\xpstartref	1, 44
\Xragged	36
\xright@appenditem	1
\Xstanza	31
\Xstanzaseparator	31
\xsublineref	1, 44
\Xsymlinenum	30
\Xtwolines	30
\Xtwolinesappref	45
\Xtwolinesbutnotmoreappref	45
\Xtwolinesonlyinsamepage	30, 45
\Xtxtbeforenotes	36
\Xunvxh	1
\xxref	1, 44

Z

\zz@@@	1
--------------	---

Change History

v0.1.0.	
General: First public release	1
v0.2.0.	
General: Added tabmac code, and extended indexing	1
\ifl@dmemoir: Added \ifl@dmemoir for memoir class having been used	61
\morenoexpands: Added \l@dtabnoexpands to \no@expands	104
\reledmac@error: Added \eledmac@error and replaced error messages	62
v0.2.1.	
\@lab: Removed page setting from \@lab	215
General: Added text about normal labeling	44
Bug fixes and match with mempatch v1.8	1
Major changes to insert code when memoir is loaded	210
\doxtrafeet: Renamed \doxtrafeet to \l@ddoxtrafeet	207
\edlabel: Tweaked \edlabel to get correct page numbers	213
\l@ddodoreinxtrafeet: Renamed \dodoreinxtrafeet to \l@ddodoreinxtrafeet	208
\morenoexpands: Removed some \lets from \no@expands. These were in edmac but Peter Wilson feels that they should not have been as they disabled page/line refs in a footnotes	104
\zz@@@: Minor change to \zz@@@	212
v0.2.2.	
General: Improved paragraph footnotes	1
New Dekker example	1
Used \providecommand for \@gobblethree to avoid clash with the amsfonts package	67
\footfudgefiddle: Added \footfudgefiddle	145
\line@list@stuff: Added initial write of page number in \line@list@stuff	98
\para@footsetup: Added \footfudgefiddle to \para@footsetup	145
\para@footsetupX: Added \footfudgefiddle to \para@footsetupX	178
v0.3.0.	
\@lab: Replaced \the\line@num by \linenumr@p\line@num in \@lab, and similar for sub-lines	215
\@nl@reg: Added a bunch of code to \@nl for handling \setlinenum	87
General: Includes edstanza and more	1
\ledlinenum: Added \linenumr@p and \sublinenum@rep to \leftlinenum and \rightlinenum	79
\linenumberlist: Added \linenumberlist mechanism	67
\printendlines: Added \linenumr@p and \sublinenumr@p to \printendlines	190
\printlines: Added \linenumr@p and \sublinenumr@p to \printlines	165
\sublinenumr@p: Added \linenumberstyle and \sublinenumberstyle	78
v0.3.1.	
General: Not released. Added remarks about the parallel package	1
v0.4.0.	
\@iiiminipage: Modified kernel \@iiiminipage and \endminipage to cater for critical footnotes	229
General: Added final/draft options	59
Added ledgroup environment	230
Added ledgroupsize environment	231
Added minipage, etc., support	1

\edtext: Added \showlemma to \edtext	105
\l@dfeetendmini: Added \l@dfeetbeginmini, \l@dfeetendmini and all their supporting code	228
\mpnormalfootgroup: Added \mpnormalfootgroup	144
\mpnormalvfootnote: Added \mpnormalvfootnote	142
\showlemma: Added \showlemma	67
\Xarrangement@normal: Added minpage footnote setup to \footnormal	141
v0.4.1.	
General: Added code for changing \docclearpage	211
Not released. Minor editorial improvements and code tweaks	1
Only change \@footnotetext and \@footnotemark if memoir not used	166
\edindex: Let eledmac take advantage of memoir's indexing	235
\print@Xnotes: Added \@opXfeet	208
\Xdo@feet: Changed \Xdo@feet code for easier extensions	208
v0.5.0.	
\@footnotetext: Enabled regular \footnote in numbered text	167
\@xympar: Eliminated \marginpar disturbance	221
General: Added left and right side notes	222
Added sidenotes, familiar footnotes in numbered text	1
v0.5.1.	
General: Added moveable side note	222
Fixed right line numbers killed in v0.5	1
Only change \hsize in ledgroupsized environment otherwise page number can be in wrong place	231
\affixline@num: Changed \affixline@num to cater for sidenotes	129
\l@getsidenote@margin: Added \sidenotemargin and \sidenote@margin	222
v0.6.0.	
\@lopR: Added \@pend, \@pendR, \@lopL and \@lopR in anticipation of parallel processing	89
\@nl@reg: Added \fix@page to \@nl	87
Extended \@nl to include the page number	87
General: Fixed long paragraphs looping	1
Fixed minor typos	1
Prepared for eledpar package	1
\fix@page: Added \last@page@num and \fix@page	88
\new@line: Extended \new@line to output page numbers	98
\vl@dbfnote: Changed \l@dbfnote and \vl@dbfnote as originals could give incorrect markers in the footnotes	167
v0.7.0.	
\@nl@reg: Added \@nl@reg	87
\@ref@reg: Added \@ref@reg	95
General: eledmac having been available for 2 years, deleted the commented out original edmac texts	1
Maïeul Rouquette new maintainer	1
Made macros of all messages	61
Replaced all \interAfootnotelinepenalty, etc., by just \interfootnotelinepenalty	1
Tidying up for eledpar and ledarab packages	1
\affixline@num: Added skipnumering to \affixline@num	129
\do@actions@fixedcode: Added \do@actions@fixedcode	127

\do@actions@next: Added number skipping to \do@actions	126
\do@insidelinehook: Added \do@linehook for use in \do@line	124
\endnumbering: Changed \endnumbering foreledpar	70
\fx@l@cks: Added \ch@cksub@l@ck, \ch@ck@l@ck and \fx@l@cks	131
\footsplitskips: Added \footsplitskips for use in many footnote styles	139
\get@linelistfile: Added \get@linelistfile	85
\ifledRcol@: Added \l@dnumpstartsL, \ifl@dpairing and \ifpst@rted for/from eledpar	67
\initnumbering@reg: Added \initnumbering@reg	69
\l@advance@parledgroup@beforenormalnotes: Added \l@dunboxmpfoot contain- ing some common code	230
\l@dcsnotetext@r: Added \l@emptyd@ta	124
\l@dgetline@margin: Added \l@dgetline@margin	75
\l@dgetlock@disp: Added \l@dgetlock@disp	77
\l@dgetsidenote@margin: Added \l@dgetsidenote@margin	222
\l@drsn@te: Added \l@dlsn@te and \l@drsn@te for use in \do@line	125
\l@dzeropenalties: Added \l@dzeropenalties	120
\ledlinenum: Added \ledlinenum for use by \leftlinenum and \rightlinenum	79
\line@list@stuff: Deleted \page@start from \line@list@stuff	98
\list@clearing@reg: Added \list@clearing@reg	85
\n@num: Added \n@num	94
\normalbfnoteX: Removed extraneous space from \normalbfnoteX	171
\resumenumbering: Changed \resumenumbering foreledpar	71
\setprintendlines: Added \setprintendlines for use by \printendlines	189
\setprintlines: Added \setprintlines for use by \printlines	162
\skipnumbering: Added \skipnumbering and supports	101
\sublinenumincrement: Added \firstlinenum, \linenumincrement, \firstsublinenum and \linenumincrement	76
\sublinenumr@p: Using \linenumrep instead of \linenumr@p	78
Using \sublinenumrep instead of \sublinenumr@p	78
\vnumfootnoteX: Removed extraneous space from \vnumfootnoteX	172
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 com- mands	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 typog- raphy. Redefines the command \hangingsymbol to define the character.	1
v0.12.0.	
General: For compatibility witheledpar, possibility to use \autopar on the right side.	1
Possibility to number \pstart.	16
Possibility to number the pstart with the commands \numberpstarttrue.	1

\ifledRcol@: Added \ifledRcol and \ifnumberingR for/from eledpar	67
v0.12.1.	
General: Don't number \pstarts of stanza.	1
The numbering of \pstarts restarts on each \beginnumbering.	1
v0.13.0.	
General: New stanzaindentsrepetition counter to repeat stanza indents every n verses.	39
New stanzaindentsrepetition counter: to repeat stanza indents every n verses.	1
\managestanza@modulo: New stanzaindentsrepetition counter to repeat stanza indents every n verses.	241
v0.13.1.	
General: \thepstartL and \thepstartR use now \bfseries and not \bf, which is deprecated and makes conflicts with memoir class.	1
v0.14.0.	
General: Tweaked \edlabel to get correct line number if the command is first element of a paragraph.	1
\edlabel: Tweaked \edlabel to get correct line number if the command is first element of a paragraph.	213
v0.15.0.	
General: Line numbering can be reset at each pstart.	73
Possibility to print \pstart number inside.	16
\affixline@num: Line numbering can be disabled.	129
\ifinserthangingsymbol: New management of hangingsymbol insertion, preventing undesirable insertions.	239
\printlines: Line numbering can be reset at each pstart.	164
v0.17.0.	
\ifinserthangingsymbol: New new management of hangingsymbol insertion, preventing undesirable insertions.	239
v1.0.0.	
General: \lemma can contain commands.	23
Debug in lineation command	18
New generic commands to customize footnote display.	28, 201
Options nonum and nosepl in \Xfootnote.	22
Options of \Xfootnotes.	137
Possibility to have commands in sidenotes.	46
Some compatibility break with eledmac. Change of name: eledmac.	1
\morenoexpands: Change to be compatible with new features	104
v1.0.1.	
General: Correction on \Xnumberonlyfirstinline with lineation by pstart or by page.	29
v1.1.0.	
General: Add \labelpstarttrue.	16
Add \Xnumberonlyfirstintwolines	30
Add \Xpstart and \Xonlypstart	31
New hook to add arbitrary code at the beginning of the notes	35
New options for block of notes.	36
New package option: parapparatus.	1
New tools to change order of series	200
Sectioning commands.	52
\preXnotes: New skip \preXnotes@	184
\settoggle@series: \settoggle@series switch the global value of the toggle, not only the local value.	201

v1.2.0.	
<code>\endquote</code> :	Compatibility of <code>\ledchapter</code> with the <i>memoir</i> class. 267
<code>\preXnotes</code> :	Debug in familiar footnotes (bug introduced by v1.1). 184
v1.3.0.	
<code>\endquote</code> :	<i>Quotation</i> and quote environment inside numbered sections. 267
v1.4.0.	
General:	Compatibility with LuaTeX of RTL notes. 1
<code>\edtext</code> :	Compatibility of <code>\edtext</code> with the right-to-left direction (with Polyglossia). 105
<code>\ledsetnormalparstuffX</code> :	Direction of footnotes with polyglossia. 181
<code>\newseries@</code> :	Remembers the language of the lemma, in order to create a correct direc- tion for the footnote separator. 194
<code>\rbracket</code> :	Switch the right bracket to a left bracket when the lemma is RTL (needs polyglossia or LuaTeX). 157
v1.4.1.	
<code>\affixside@note</code> :	Remove spurious spaces. 226
<code>\endquote</code> :	New option <i>noquotation</i> 267
<code>\labelrefsparsesubline</code> :	Fix bug with <code>\edlabel</code> 214
<code>\vl@dbfnote</code> :	Compatibility of standard footnotes with <code>eledmac</code> when these footnotes contain any commands. 167
v1.4.2.	
General:	Debug with some special classes. 1
v1.4.3.	
General:	Add <code>\Xnonbreakableafternumber</code> 31
Spurious space after familiar footnotes. 1
v1.4.4.	
General:	Label inside familiar footnotes. 1
v1.4.5.	
General:	Bug with <code>komasscript</code> + <code>eledpar</code> + <code>chapter</code> 1
v1.4.6.	
General:	Bug with <i>memoir</i> class introduced by 1.4.5. 1
v1.4.7.	
<code>\endquote</code> :	Compatibility of sectioning commands with <code>\autopar</code> 267
v1.4.8.	
General:	Corrects a bug with parallel texts introduced by 1.1. 1
v1.4.9.	
<code>\normalbfnoteX</code> :	Allow to redefine <code>\thefootnoteX</code> with <code>alph</code> when some packages are loaded. 171
v1.5.0.	
General:	Correct indexing when the call is made in critical notes. 232
<code>\do@insidelinehook</code> :	Added <code>\do@insidelinehook</code> for use in <code>\do@line</code> 124
<code>\edindex</code> :	Compatibility with <code>imakeidx</code> package, and possibility to use multiple index with <code>\edindex</code> 235
v1.5.1.	
<code>\managestanza@modulo</code> :	Correct <code>stanzaindentsrepetition</code> counter 241
<code>\normalvfootnoteX</code> :	Fix bug with normal familiar footnotes when mixing RTL and LTR text. 169
v1.6.0.	
<code>\newverse</code> :	Add <code>\falseverse</code> macro. 244
v1.6.1.	
General:	Corrects a false hanging verse when a verse is exactly the length of a line. 1

\AtEveryPstart: Spurious space in \pstart.	117
\ifinserthangingsymbol: Hang verse is now not automatically flush right.	239
\l@dunhbox@line: Move the call to \inserthangingsymbol to allow use \hfill in-side.	122
\pend: Spurious space in \pend.	119
v1.7.0.	
General: New features for managing page breaks.	53
v1.8.0.	
General: Compatibility with parledgroup option of eledpar package.	1
If imakeidx and hyperref are loaded, adds hyperref in the index.	232
\endquote: Correction of sectioning commands in parallel texts.	267
\get@index@command: Debug \get@index@command and compatibility with hyperref package.	234
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work.	204
\prevpage@num: Correct \parafootsep when using with ledgroup.	151
v1.8.1.	
General: Debug endnotes when more than one series is used (change the position where tools for endnotes are defined).	186
v1.8.2.	
General: Debug compatibility problem with hebrew option of babel package.	1
v1.8.3.	
General: Fixes spurious spaces added by v1.7.0.	1
v1.8.5.	
General: Debug indexing in right column, with eledpar.	232
v1.9.0.	
\doxtrafeet: Add \fnpos to choice the order of footnotes.	207
\l@dfeetendmini: Add \mpfnpos to choice the order of footnotes in minipage / led-group.	228
v1.10.0.	
General: Add \pstartref and \xpstartref to refer to a pstart number (extension of \edlabel).	1
\endquote: Correction of sectioning commands in parallel texts.	267
v1.10.1.	
General: Compatibility with cleveref.	1
v1.10.2.	
General: Compatibility of stanza with v1.8a of babel-greek.	1
v1.10.3.	
General: Debug of cross-referencing.	1
v1.10.4.	
General: Debug of critical notes in edtabular environment.	1
v1.10.5.	
General: Debug of \pausenumbering.	1
Debug of \xxref.	1
v1.10.6.	
General: Debug of interaction between \autopar and \pausenumbering.	1
v1.11.0.	
General: Add hooks to disable the font selection for lemma in footnote.	34
v1.11.1.	
General: Correct a bug when a critical note starts with plus or minus.	1

v1.12.0.

\@nl@reg: To ensure compatibility with \musixtex, \@l becomes \@l. Consequently, \@l@reg becomes \@nl@reg.	86
General: Add \ledinnernote and \ledouternote commands.	46
Add \Xendparagraph and related settings.	37
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	232
New hooks: \Xafterrule and \afterruleX	37
New options for ragged-paragraph notes	36
New sectioning commands.	52
Optional arguments for \pstart and \pend.	16
\AtEveryPstart: New optional argument for \pstart, to execute code before it. . .	117
\edindex: Use correctly default index when imakeidx is loaded.	235
\endquote: \ledxxx sectioning commands are deprecated and replaced by \eledxxx commands.	267
\ifledRcol@: Add \ifledRcol@ for eledpar	67
\initnumbering@reg: \beginnumbering is defined only on eledmac, not on eledpar. .	69
\l@dcnote: \l@dlsnote, \l@drnote and \l@dcnote defined only one time, in eledmac, including needs for eledpar case.	224
\l@dgetsidenote@margin: \sidenotemargin is now directly defined in eledmac to be able to manage eledpar.	222
\l@dunhbox@line: \do@line is split in more little commands.	122
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work when called after \footparagraphX.	204
Debug \Xbeforenotes and \Xmaxhnotes which did not work when called after \footparagraph.	204
\pend: New optional argument for \pend, to execute code after it.	119
\stanza: &can have an optional argument: content to be printed after.	244
\Stanza can have an optional argument: content to be printed before.	244
Add \newverse macro, \falseverse deprecated.	244

v1.12.1.

\wrap@edcrossref: Fix spurious spaces.	216
--	-----

v1.12.2.

\l@dunhbox@line: Fix a bug with critical notes at the tops of pages (added by v12.0.0)	122
--	-----

v1.12.3.

General: Add macros for new messages since v0.7	61
Correct bug with side and familiar notes in tabular environments.	1
Debug \eledxxx with some paper size	1
Debug \ledinnernote and \ledouternote commands in the top of pages.	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	99
\flag@start send a error message when a \edtext is done without insert (note) .	99
\reledmac@error: Replaced error messages	62

v1.12.4.

General: Debug spurious page breaks before \chapter (bug added in 1.12.0)	1
---	---

v1.12.5.	
\edindex@hyperref: Debug \edindex when hyperref is not loaded	237
\sssect: Debug \eledchapter in parallel with memoir	271
\doinsidelinehook: Added \dolinehook and \doinsidelinehook	124
\endnumbering: Allow to mix parallel columns and normal text when using \pausenumbering	70
\l@dgobblearg: \l@dgobblearg becomes \l@dgobbeloptarg	251
\l@drestoreforedtext: Debug optional arguments of \Xfootnote in tabular context	251
\resumenumbering: Debug \resumenumbering	71
v1.12.7.	
\wrap@edcrossref: \wrap@edcrossref is now robust	216
v1.12.8.	
\flag@end: \flag@start do not send a error message when a \edtext is done without insert (note) but have a endnote	99
v1.13.0.	
General: Add \Xnoteswidthliketwocolumns and \noteswidthliketwocolumnsX	37
Added widthliketwocolumns option	59
\newhooktoggle@series@reload: Add \newhookcommand@toggle@reload	203
\para@footsetupX: In \para@footsetupX, use \columnwidth instead of \hsize	178
\settoggle@series: \settoggle@series can take an optional arguments to reload series setup.	201
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@dzero penalties in \pstart	117
v1.13.2.	
General: Fix bug with normal footnotes, added by v1.13.0.	1
\ifledRcol@: Add \ifl@dpadding for \eledpar	67
v1.13.3.	
General: Fix extra spaces with paragraphed footnotes, added by v1.13.0.	1
v1.13.4.	
General: Fix bug with index when memoir class is used without hyperref	1
v1.14.0.	
General: Debug spurious characters before endnotes.	186
Delete previous override of \l@d@wrindexhyp at the beginning of a document when hyperref is not loaded.	239
Move gobbling command	67
Provide \@gobblefour	67
\edindex: Let \eledmac take advantage of \makeidx even when memoir class is used	235
v1.14.1.	
\sssect: Debug sectioning commands when using both handout and hyperref package.	274
v1.14.2.	
\sssect: Debug \edtext after starred sectioning commands when using memoir class.	271
v1.15.0.	
\edtext@level: New boolean \if@edtext@.	105
General: Fix bug with footnotes layout when using some options of the geometry package (bug add by v1.13.0).	1
New commands \AtEveryPstart and \AtEveryPend.	16

New tools to prevent ambiguous references in lemma	24
\arrangementX@threecol: Correct bug with paragraphed familiar footnotes setting.	177
\endsub: Restore subline feature (disabled by mistake in v1.8.0).	100
\if@lemmacommand@: New boolean \iflemmacommand@.	110
v1.15.1.	
\line@list@stuff: Revert modification of 1.5.2 which makes bug with numbering.	
Leave vertical mode to solve spurious space before minipage.	98
v1.16.0.	
General: \edtext is now defined only in eledmac, not in eledpar. Debug wrong numbering when using \sameword + eledpar + \tag command.	105
Compatibility of standard footnotes with some biblatex styles.	1
New \stanzaindent command.	1
v1.16.1.	
\edlineref: \lineref is not defined if defined by some other package, like lineno.	
Eledmac provides \edlineref instead.	217
v1.17.0.	
\edtext: Error message when calling \edtext outside of a numbered paragraph.	105
v1.18.0.	
\@edindex@hyperref: Fix spurious space with \edindex when using imakeidx/indextools + hyperref.	237
General: Add \Xpstarteverytime	31
Compatibility with Lua \TeX RTL languages.	1
Debug \Xonlypstart when using \Xnumberonlyfirstinline and the current line number differs from the previous.	31
\edlabel: \edlabel is now defined only one time for both eledmac and eledpar	213
\ifledRcol@: Add \ifl@dprintingpages and \@dprintingcolumns for eledpar	67
\l@d@section: Option parapparus works for endnotes.	187
\print@line: Compatibility with Lua \TeX RTL languages.	122
\printlinefootnote: Code refactoring in \printlinefootnote: the printing of the numbers are factorized in \printlinefootnotearea	158
\printpstart: Debug \Xpstart with parallel pages and columns (eledpar)	157
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
\footsplitskips: \footsplitskips doesn't set \floatingpenalty to \@MM when processing parallel pages.	139
\xxref: \xxref works also with right side numbers, when \@Rlineflag is not empty.	218
v1.19.1.	
General: Call \correct@footinsX@box and \correct@Xfootins@box directly in \print@notesX@forpages and \print@Xnotes@forpages, that is in eledpar.	1
v1.20.0.	
General: Add \Xendboxlinenum	33
Add \Xtwolines and \Xmorethantwolines hooks	30
Add series option.	1
Correct \Xinplaceofnumber hook.	1
Explicit error message when calling \Xfootnote outside of \edtext.	1
Fix bug with line number typesetting direction when using \eledsection and similar commands for RTL texts with Lua \TeX .	1
Fix issues with RTL text in notes when using Lua \TeX .	1

Options fulllines in <code>\Xfootnote</code> .	22
The <code>\newifs</code> are not followed by boolean values set to false, because it is the \TeX default setting.	1
<code>\printlines</code> : Added <code>\ifl@d@Xmorethantwolines</code> and <code>\ifl@d@Xmorethantwolines</code> to <code>\printlines</code>	165
<code>\stanza</code> : <code>&</code> and <code>\&</code> can be preceded by spaces.	244
<code>\xxref</code> : Debug <code>\xxref</code> when not loading <code>eledpar</code> (fix bug added in 1.19.0).	218
v1.21.0.	
<code>\@edindex@hyperref</code> : Look at the hyperindex option of <code>hyperref</code> before inserting <code>hyperref</code>	237
General: <code>\AtEveryPstart</code> and <code>\AtEveryPend</code> are now compatible with <code>\autopar</code>	1
<code>\Xafterrule</code> and <code>\afterruleX</code> features no longer create problems of overflowing at the bottom of the page.	1
<code>\chapter</code> inside optional argument of <code>\pstart</code> works when typesetting parallel pages	1
<code>\preXnotes</code> and <code>\prenotesX</code> features no longer create problems of overflowing at the bottom of the page.	1
<code>\seriesatbegin</code> and <code>\seriesatbegin</code> more efficient	200
Add <code>\applabel</code> and related	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	35
Add <code>\Xendtwolines</code> , <code>\Xendmorethantwolines</code> , <code>\Xendtwolinesbutnotmore</code> and <code>\Xendtwolinesonlyinsamepage</code> .	30
Add <code>\Xparindent</code> and <code>\hangindentX</code>	34
Add <code>\Xtwolinesbutnotmore</code> and <code>\Xtwolinesonlyinsamepage</code> .	1
Add <code>nocritical</code> , <code>noend</code> , <code>nofamiliar</code> and <code>noledgroup</code> options.	1
Add <code>noeledsec</code> package option	1
Debug <code>\beforenotesX</code> <code>\maxhnotesX</code> <code>\noteswidthliketwocolumnsX</code> and <code>\afterruleX</code> with footnotes set in two and three columns.	1
Fix bug when a <code>\Xfootnote</code> follows a <code>\Xendnote</code> in the second argument of <code>\edtext</code> (bug added in <code>eledmac</code> 1.0.0).	1
Fix bug with <code>\maxhnotesX</code> when using <code>\foottwocolX</code> or <code>\footthreecolX</code> .	1
Fix bug with space between columns with notes in two columns (bug added in v1.13.0).	1
Fix spurious space after first page number in <code>\doendnotes</code> . <code>oldprintnpnumspace</code> option allows to come back to previous setting	1
<code>parapparatus</code> option works now with familiar footnotes.	1
Provide <code>\@gobblefive</code>	67
<code>\l@d@section</code> : <code>\endnotes</code> take five arguments.	187
<code>\ledinnotemark</code> : Add <code>\ledinnotemark</code> .	235
<code>\ledsetnormalparstuffX</code> : <code>\ledsetnormalparstuff</code> is deprecated and becomes <code>\ledsetnormalparstuffX</code> and <code>\Xledsetnormalparstuff</code> .	181
<code>\n@num</code> : <code>\n@num@ref</code> deleted	94
<code>\n@num</code> defined only one time for both <code>Eledmac</code> and <code>Eledpar</code>	94
<code>\newhookcommand@series</code> : <code>\newhookcommand@series</code> can take an optional argument.	203
<code>\newhooktoggle@series</code> : <code>\newhooktoggle@series</code> can take an optional argument.	203
<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>	183

\seriesatend: Fix spurious space in \seriesatend	200
\skipnumbering: \skipnumbering defined only one time for both Eledmac and Eledpar.	
Correct \skipnumbering for stanza.	101
Delete \skipnumbering@reg.	101
v1.22.0.	
General: Add \doendnotesbysection command.	22
Add option for lemma separator inside endnotes	33
Adds hyperlink for references to notes in indices.	1
Fix conflict between noend package option and edtabularx environments	1
Provides support for xindy.	1
Standardize endnotes handbook.	22
When using hyperref package, internal links in index or with \edlineref are now targeted to the top and not longer to the bottom of the lines they refer to.	1
\ledinnote: \ledinnote takes a first optional argument, which is the label for hyperlinks.	235
v1.22.1.	
General: Fix bug (added on v1.22.0) with \Xinplaceofnumber hook.	1
\prevpage@num: Correct double symbol when using both \parafootsep and \Xsymlinenum.	151
v1.23.0.	
\@edtext@level: The boolean \if@edtext@ becomes the counter \edtext@level.	105
General: Add \Xboxlinenumalign and \Xendboxlinenumalign.	32
Add \Xboxstartlinenum, \Xendboxstartlinenum, \Xboxendlinenum, \Xendboxendlinenum.	32
Allow use of \sameword with inputenc managing of UTF-8.	1
Compatibility between nofamiliar/nocriticals option and minipage/ledgroup.	1
Error message when using \beginnumbering... \endnumbering without \pstart.	1
Fix bug with \sameword when the lemma overlaps multiple line.	24
Fix bug with \sameword when the same lemma is used for multiple notes or for nested \edtexts.	24
Fix bug with \skipnumbering called immediately after a \pstart.	1
Fix error of \iftrue not closed.	1
Fix spurious space with \skipnumbering (bug added on v1.21.0).	1
New tools to ensure the line-list file uses the right version of commands when upgrading the eledmac version.	1
Optional argument of \sameword can be a comma separated list of \edtext depth.	24
\lemma: Fix spurious space after \lemma command	109
\newseries@: Prevent spurious spaces when \Afootnote and similar commands are followed by spaces (bug added on 1.0.0).	194
\sameword: In order to allow use of \sameword with inputenc, we detokenize its mandatory argument before using it in control sequence names.	113
v1.23.1.	
General: Fix bug with \lemma command in the right side.	1
v1.23.2.	
General: Compatibility with L ^A T _E X's release 2015.	1
v1.24.0.	
General: We can reinitialize \AtEveryPstart and \AtEveryPend providing to it an empty argument.	1

v1.24.1.	
General: <code>\lemma</code> is disabled when using ‘nocritical’ option.	1
v1.24.2.	
General: Fix incompatibility between ‘nofamiliar’ option and ‘memoir’ package.	1
v1.24.3.	
General: Restore marginal numbers and notes with sectioning command (bug introduced in v1.21.0)	1
v1.24.4.	
General: Fix spurious space with <code>\edindex</code> when using <code>xindy+hyperref</code> option.	1
v1.24.5.	
General: Fix bug of indent, when a added in 1.1.0, when a <code>\beginnumbering</code> immediately follow a sectioning command.	1
v2.0.0.	
<code>\@iiiminipage</code> : Patch <code>\@iiiminipage</code> instead of redefining it.	229
<code>\@xympar</code> : Patching <code>\@xympar</code> instead of redefining it	221
General: <code>\@makecol</code> , <code>\@reinserts</code> and <code>\@doclearpaper</code> are patched instead of begin redefined	210
<code>\doxtrafeeti</code> becomes <code>\do@feetX</code> ; <code>\doxtrafeetii</code> becomes <code>\Xdo@feet</code> ; <code>\@opxtrafeeti</code> becomes <code>\@opfeetX</code> ; <code>\doreinxtrafeetii</code> becomes <code>\X@doreinfeet</code> ; <code>\doreinxtrafeeti</code> becomes <code>\@doreinfeetX</code>	210
Add <code>\Xendinplaceofnumber</code> hook.	1
Add <code>\Xendnonumber</code> hook.	1
Add <code>nonum</code> option for endnotes.	1
Fix bug when printing only one series of endnotes, but wanted to keep endnotes for other series.	1
In order to have a more consistent name’s convention, many names has been changed.	1
Many \TeX ’s output macros are now patched and not override.	1
Package’s name becomes <code>reledmac</code>	1
Patch <code>\@footnotemark</code> instead of redefine it	166
Suppress indexing command specific to <code>memoir</code>	235
<code>\endminipage</code> : Patch <code>\endminipage</code> instead of redefining it.	229
<code>\initnumbering@quote</code> : <code>\initnumbering@sectcmd</code> becomes <code>\initnumbering@quote</code>	267
<code>\l@advance@parledgroup@beforenormalnotes</code> : Some conde of <code>\l@dumboxmpfoot</code> moved to <code>\l@advance@parledegroupp@beforenormalnotes</code>	230
<code>\newsseries@</code> : One endnotes file by series.	197
v2.0.1.	
General: Fix bug in <code>eledmac-compat</code> option	1
Fix incompatibility between optional argument of <code>\pstart</code> and <code>\numberpstarttrue</code>	1
v2.1.0.	
General: Fix bug with <code>\advanceline</code> at the begin of a <code>\pstart</code>	1
Fix bug with <code>\chapter</code> in optional argument of <code>\pstart</code> in parallel typesetting with <code>scrbook</code>	1
Fix bug with <code>\eledchapter</code> in parallel typesetting with <code>scrbook</code>	1
Fix bug with <code>\setline</code> at the begin of a <code>\pstart</code>	1
Fix space bug with <code>\Xhooknote</code> and <code>\hooknoteX</code> when using to insert text and not to execute code.	1
New tools to number stanza	1
v2.1.1.	
General: Fix bug with <code>\ledpbsetting{before}</code>	1