

footmisc —
a portmanteau package
for customizing footnotes in L^AT_EX*

Robin Fairbairns[†] Frank Mittelbach[‡]

March 8, 2022

Copyright statement

Program: `footmisc.dtx`

Copyright 1995–2011 Robin Fairbairns

Copyright 2018–2022 Robin Fairbairns, Frank Mittelbach

This program is offered under the terms of the L^AT_EX Project Public License, version 1.3c of this license or (at your option) any later version. The latest version of this license is in <https://www.latex-project.org/lppl.txt>, and version 1.3c or later is part of all distributions of L^AT_EX version 2008 or later.

This work has the LPPL maintenance status ‘maintained’.

Contents

1	User interface — package options	3
1.1	Option <code>perpage</code>	3
1.2	Option <code>para</code>	3
1.3	Option <code>side</code>	4
1.4	Option <code>ragged</code> and <code>\footnotelayout</code>	4
1.5	Option <code>symbol</code>	4
1.6	Option <code>symbol*</code>	5
1.7	The <code>\setfnsymbol</code> and <code>\DefineFNSymbols</code> commands	5
1.8	Options altering the footnotes/floats relationship	5
1.8.1	Option <code>bottom</code>	6
1.8.2	Option <code>bottomfloats</code>	6
1.8.3	Options <code>abovefloats</code> and <code>belowfloats</code>	6
1.8.4	Combining the four options	6
1.8.5	Fixing a strange behavior of L ^A T _E X	7
1.9	Option <code>marginal</code>	7
1.10	Option <code>flushmargin</code>	7
1.11	Option <code>hang</code>	7

*This file has version number v6.0d, last revised 2022/03/08

[†]Formerly: University of Cambridge Computer Laboratory, Cambridge, UK

[‡]Responsible maintainer since 2018

1.12	Option <code>norule</code>	8
1.13	Option <code>splitrule</code>	8
1.14	The <code>stable</code> option	8
1.15	The <code>multiple</code> option	8
1.16	User interface — miscellaneous commands	9
2	User interface — interactions with other packages	9
3	Code: Preliminaries	10
4	Package options	10
4.1	The <code>symbol</code> option	10
4.2	The <code>symbol*</code> option	11
4.3	The <code>para</code> option	11
4.4	The <code>side</code> option	11
4.5	The <code>ragged</code> option	12
4.6	The <code>perpage</code> option	12
4.7	The <code>PPdebug</code> option	12
4.8	Fixing the L ^A T _E X misbehavior with respect to spacing	12
4.9	The footnote/float placement options	12
4.9.1	The <code>abovefloats</code> , <code>belowfloats</code> options	13
4.9.2	The <code>bottom</code> option	13
4.9.3	The <code>bottomfloats</code> option	13
4.10	The <code>marginal</code> option	13
4.11	The <code>flushmargin</code> option	13
4.12	The <code>hang</code> option	14
4.13	The <code>norule</code> option	14
4.14	The <code>splitrule</code> option	14
4.15	The <code>stable</code> option	15
4.16	The <code>multiple</code> option	15
4.17	The start of the endgame	15
5	Hacking kernel commands	15
5.1	The output routine part	15
5.2	The output routine configuration components	18
5.3	The <code>\@makecol</code> configuration based on options	20
5.4	The requirements of <code>\@footnotetext</code>	22
5.5	Support code for paragraph footnotes	24
5.6	The other footnote commands	26
6	Remaining requirements	27
6.1	The code that executes the <code>multiple</code> option	28
6.2	The code that executes the <code>stable</code> option	29
7	Symbol option variants	30
8	Other miscellaneous commands	34
8.1	Minipage <code>\footnotemarks</code>	34

History

This package originated as support of a personal project, which I (Robin) was switching to L^AT_EX 2_ε over the Christmas holiday period of 1993, using the first β release.

In its first form, it was known as the “footnote” package, but by the time I had released it to CTAN, that name had already been used by a package written by Mark Wooding. So the package is now known (as you can see) as “footmisc”.

Frank took over maintenance in 2018 but due to other commitments never got around finishing the changes he started to make in 2018.

In 2022 a few new options (`abovefloats`, `belowfloats`, and `bottomfloats`) got introduced and the package now works with `hyperref` regardless of loading order. There are however, still a few restrictions when using both packages together, in particular the `multiple` option does not fully work.

1 User interface — package options

The `footmisc` package provides several different customizations of the way footnotes are represented in L^AT_EX 2_ε documents (the sources of the code in this package are various, but all of it has been massaged by the author; where the code comes from elsewhere, there are attributions given below, somewhere or other).

The interface to the package’s options is mostly rather simple — each one is presented as an option in the `\usepackage` command, and for most, nothing else needs to be done. For example, to use a useful and consistent set, the author invokes the package with the command `\usepackage[perpage,para,symbol*]{footmisc}`.

For a small number of options, there are additional parameters available; these are described in the subsections below.

1.1 Option `perpage`

This option resets footnote numbering for each page of the document. It needs at least two passes to do this correctly (though it comes as close as possible on the first pass). You generally have to make two passes with L^AT_EX anyway, to get the cross-references right, so an additional pass for this purpose shouldn’t cause any additional problem. The option includes code to report that ‘*Label(s) may have changed*’, which will help the poor user to realize that (yet) another run is in order.

1.2 Option `para`

This option (derived from code by Dominik Wujastyk and Chris Rowley) causes footnotes to be typeset as a single paragraph at the bottom of the page on which they occur. In the case that there is only one footnote on the page, no effect will be observed. However, if there are several footnotes on the page, they will be run together in the page foot, each introduced by its footnote mark. The original demand for the option came from the needs of those preparing critical editions; such documents typically have large numbers of small footnotes, which look ridiculous if each is typeset in a paragraph of its own; in most other disciplines, such multiplicities of footnotes represent mere self-indulgence: the author of this package is disgracefully guilty of this.

Please note that “old” L^AT_EX installations may have problems with the algorithm for `para` footnotes on very wide pages (for example, those used by the `a0poster` class). Recent L^AT_EX installations use an improved technique that is believed not to be susceptible to this problem.

1.3 Option `side`

This option (suggested by Frank Mittelbach) causes footnotes to be typeset using the `\marginpar` command: this has the advantage that the note appears close to its “call-up”, but has all the disadvantages associated with the `\marginpar` command (which consumes ‘float’ slots, and doesn’t always place itself correctly at the top of pages in two-sided documents). Since the measure in which the footnote is to be typeset is likely to be pretty narrow, users of the `side` option are recommended also to use the `ragged` option, to avoid ugly spacing and line breaks.

There is a further problem (apart from the occasional failure to place the marginal note on the correct side of the page) in two-sided documents: one would like ‘raggedness’ to appear differently in different margins (setting the left, rather than the right, side ragged in the left margin). (The author would welcome suggestions on means of addressing the problem.)

1.4 Option `ragged` and `\footnotelayout`

The package provides facilities for ragged right setting of footnotes (so long as the `para` option isn’t in effect). The change is effected by use of the command `\footnotelayout`; the package inserts this command into the start of the argument of `\footnotetext` (in effect: `\footnote` works, roughly, by calling the guts of `\footnotetext` at its end).

If you want to use some special effect other than ragged right, feel free to change `\footnotelayout` yourself: some intriguing (and completely undesirable) results are no doubt available. Change the setting simply by use of `\renewcommand\footnotelayout...`. The `ragged` option simply sets `\footnotelayout` to set `\raggedright` or `\RaggedRight` as appropriate. (If you intend to use the `ragged2e` package, load it before `footmisc` — if `footmisc` finds `\RaggedRight` is available, it automatically uses it in place of `\raggedright`.)

1.5 Option `symbol`

This option simply establishes that footnotes are ‘labelled’ by a symbol sequence. The command used is equivalent to that suggested in L^AT_EX manuals such as Lamport’s (the job performed by the option is very simple, and doesn’t really need a package).

Using symbols to ‘number’ your footnotes can be problematic: there is a limited number of symbols, and L^AT_EX will report an error if your footnotes exceed that limit. To avoid such problems, consider the `symbol*` option, or the `\setfnsymbol` command (see the next two sections), or number your footnotes by the page (see section 1.1).

1.6 Option `symbol*`

This is the `symbol` option, but with protection against the tedium that arises because of the instability of the `perpage` option. When executing the `perpage` option, the package often allocates footnotes to the wrong pages, only to correct itself on a later run (having warned the user of the need for the later run with a ‘*Label(s) may have changed*’ message). In these circumstances the `symbol` option is prone to producing L^AT_EX errors, which stop processing, and confound automatic generation procedures. In the same situation, the `symbol*` option produces information messages and a warning message at end document, and the user may scan the log for those messages *after* processing has stabilized. The option produces numbers (17 and higher, in the case of the default symbol set) in place of symbols, when the footnote number is too large.

1.7 The `\setfnsymbol` and `\DefineFnsymbols` commands

NOTE: *At some point in the past this interface got extended, but the documentation lags behind so this needs updating.*

These commands permit the definition and use of alternative (ordered) sets of symbols for numbering footnotes. L^AT_EX of course comes with such a set ready-defined, but the choice of symbols isn’t universally loved.

You may define a set of symbols with the `\DefineFnsymbols` command. L^AT_EX’s default set would be defined by the command:

```
\DefineFnsymbols*{lamport}{*\dagger\ddagger\S\P\|/%
                    **}{\dagger\dagger}{\ddagger\ddagger}}
```

Defined this way, the symbol set produces a “counter too large” error; a robust version of the set (cf. the `symbol*` option (see 1.6) using the `\DefineFnsymbols` command without the optional `*`. You may select a set of symbols by use of the `\setfnsymbol` command; so to restore use of the default set, you would type:

```
\setfnsymbol{lamport}
```

This package defines a small selection of alternative sets of symbols, using `\DefineFnsymbol`:

```
bringhurst  * † ‡ § || ¶
chicago    * † ‡ § || #
wiley       * ** † ‡ § ¶ ||
```

together with a version of Lamport’s original set that, with doubled versions of § and ¶, and tripled versions of everything but the vertical bars, provides a symbol range to cover counters up to 16.

This last set, known as `lamport*` is selected as the default symbol set by the package.

1.8 Options altering the footnotes/floats relationship

In L^AT_EX the default order on a page is “page text” followed by “footnotes” (if any) followed by “bottom floats” (if any). The spacing between the three components depends on whether pages are always stretched to the same height (`\flushbottom`

as used by the book class) or if they can run short (`\raggedbottom` as used by the article or report class). If `\raggedbottom` is in force, then \LaTeX would normally set the footnotes a mere `\skip\footins` distant from the bottom of the text and bottom floats follow separated by `\textfloatsep`. Both spaces might get stretched if `\flushbottom` is in force.

New: 2022-02

If you want to diverge from this default placement, then there are a number of alterations that can be made:

- the order of footnotes and floats can be swapped; and
- both footnotes and floats can be forced to the bottom (i.e., `\raggedbottom` then only applies to pages with neither), or
- only one of them is forced to the bottom, the other stays close to the text.

These can be achieved by applying one or more of the options discussed below.

1.8.1 Option `bottom`

This option forces footnotes (but not the floats) to the bottom of the page and therefore by default also implies `belowfloats`. If `\raggedbottom` is in force then the excess space goes above the footnotes if any are present. If `\flushbottom` is in force there is no visible difference to just specifying `belowfloats`.

1.8.2 Option `bottomfloats`

New: 2022-02

If you want force only floats to the bottom while the footnotes stay close to the text use the option `bottomfloats`. If not overwritten this implies `abovefloats`.

1.8.3 Options `abovefloats` and `belowfloats`

New: 2022-02

These two options describe the footnote placement with respect to floats on the page. \LaTeX 's default is `abovefloats`, but it can still be useful to specify it because it fixes the bug discussed in section 1.8.5.

1.8.4 Combining the four options

New: 2022-02

By default, `bottom` and `bottomfloats` options put any excess space (i.e., when `\raggedbottom` is in force) between floats and footnotes if both are present on a given page. If only one of them is present, the excess space goes below the text. If you prefer both footnotes and floats at the bottom instead, you can achieve this as follows:

`bottom,abovefloats` This puts the footnotes above any floats and both at the bottom when present.

`bottomfloats,belowfloats` This puts floats and footnotes at the bottom but footnotes last if both are present. If there are only footnotes they are still placed at the very bottom (think of them as being placed below the float “area” even if that has no floats inside).

The other combinations are duplicates, e.g., `bottom,belowfloats` is the same as just specifying `bottom`.

New: 2022-02

1.8.5 Fixing a strange behavior of L^AT_EX

In the default case (if `footmisc` is not loaded) L^AT_EX shows a somewhat strange discrepancy: on most pages the footnotes are attached a distance of `\skip\footins` from the main text, even if that page is run short (i.e., with `\raggedbottom` in force). However, whenever there is some infinite stretch at the end of the page, e.g., from using `\newpage` or `\clearpage` the footnotes are pushed to the bottom (in particular on the last page of a document or chapter).

This is automatically corrected if `footmisc` is loaded with any of the options that deal with footnote placement, i.e., `abovefloats`, `belowfloats`, `bottom`, or `bottomfloats`. In particular, if you want to have the standard L^AT_EX placement (but with this strange behavior fixed, apply `abovefloats` (which is the normal order but with the bug fixed).

1.9 Option `marginal`

This option adjusts the position of footnote mark relative to the start of the line in which they appear (the option is incompatible with option `para`, for obvious reasons).

When this option is in effect, the footnote is set `\footnotemargin` relative to the left margin of the page; the default setting for `\footnotemargin` is `-0.8em`, which means that the footnote mark will be set jutting `0.8em` into the margin. If `\footnotemargin` is a positive length, the footnote mark will be set with its right edge `\footnotemargin` from the margin. (In the absence of the option, `\footnotemargin` is set to `1.8em`; you may change that value with a `\setlength` command.)

1.10 Option `flushmargin`

This option is as option `marginal`, but sets the footnote marker flush with, but just inside the margin from, the text of the footnote.

1.11 Option `hang`

This option sets the footnote mark flush with the margin, and makes the body of the footnote hang at an indentation of `\footnotemargin` (if that is a positive distance), or the width of the marker (if `\footnotemargin ≤ 0`). The option code itself leaves `\footnotemargin` at its default value of `1.8em`.

The footnote itself may of course be longer than one paragraph; if so, the paragraphs will be separated by the vertical space specified by `\hangfootparskip`, and the second and subsequent paragraphs are indented by `\hangfootparindent`. Default values are:

```
\hangfootparskip    0.5\baselineskip
\hangfootparindent  0em
```

The user may redefine these values (using `\renewcommand`): it is best to use the font-size-dependent measures (multiples of `\baselineskip` for the skip, multiples of `em` for the indent). Note that the default has only one of the two values non-zero; both zero may result in easily-missed paragraph breaks, and both non-zero is not generally thought to be a good-looking option.

1.12 Option `norule`

This option suppresses the ‘normal’ footnote rule, and advances `\skip\footins` a bit to compensate

1.13 Option `splitrule`

This option makes puts a full-width rule above the split-off part of a split footnote. (Remember that split footnotes don’t happen if you’re doing paragraph footnotes.)

The option provides three different `\footnoterule` commands:

<code>\mpfootnoterule</code>	for use in minipages
<code>\pagefootnoterule</code>	for normal footnotes on regular pages
<code>\splitfootnoterule</code>	for the tail of a split footnote

By default, `\mpfootnoterule` and `\pagefootnoterule` retain the original definition of `\footnoterule` (which may have been modified by a `norule` option), while `\splitfootnoterule` becomes a full-width rule.

1.14 The stable option

This option deals with the problem of placing footnotes in section titles (and so on). While there is (sometimes, just) justification for putting footnotes in titles, \LaTeX ’s treatment of the content of titles militates against them. Of course, the title argument is ordinarily a moving one, and `\footnote` is a fragile command, but the real problem comes from the way the argument actually moves — which is to two places. The argument moves to the table of contents, where the footnote will (at least) look odd. But the argument also moves to the marks that make up page headers, etc., and *there* it creates havoc, since page headers are executed in page make-up, and page make-up *must not* create footnotes.

If you use the `stable` option, the footnote won’t move to the table of contents or the page headers, but it will be typeset correctly within the title itself.

The situation with `\footnotemark` is less dire (it could in principle appear in page headers, for example); footnote marks appearing on pages other than where their text appears are none the less confusing, and the `stable` option treats `\footnotemark` in the same way that it treats `\footnote`.

1.15 The multiple option

This option deals with the case where the author needs to type things like

```
mumble\footnote{blah}\footnote{grumble}
```

Without special treatment, \LaTeX would output something like

```
mumble1314
```

What the `multiple` option makes of the above is

```
mumble13,14
```

which is what most people would expect. The comma separator actually derives from the definition of `\multfootsep`, which may be changed by `\renewcommand` if the option is in effect.

The option also treats `\footnotemark` in the same way.

1.16 User interface — miscellaneous commands

The package also defines some miscellaneous footnote-related commands. The present group provides alternative means of producing footnote marks: `\footref` and `\mpfootnotemark`.

When you're in a minipage, `\footnote` numbers run according to the minipage's own footnote counter, and the marks are set in italic letters. However, the numbers used by `\footnotemark` make reference to the 'main' footnote counter, and are set in whatever is the current style for that: this behavior often surprises, and there's no obvious way in standard L^AT_EX to "get around" it. The command `\mpfootnotemark` gets around this problem in a minipage, by generating footnote marks in the same way as those used by `\footnote`.

In fact, making reference to footnotes in general can be problematic: it can be done by noting down the value of the footnote marker in a counter (or the like) and then using the value in a subsequent `\footnotemark` or `\mpfootnotemark`. This is a tedious way of going about things, and doesn't allow representation of all possible forms of footnote mark; `\footref` is a form of reference command that sets the reference as if it were a footnote.¹ The label should be set *within* the argument of the footnote command that is being labelled:

```
... \footnote{Note text\label{fnlabel}}
...
... potato head\footref{fnlabel}
```

2 User interface — interactions with other packages

The `footmisc` package modifies several parts of the L^AT_EX kernel; what gets modified depends on the options you select. This behavior can cause problems with other packages, particularly those that also modify the kernel.

Known interactions are:

`setspace` The `setspace` package modifies the way line spacing is calculated in footnotes. `Footmisc` knows about this, and preserves the change. However, you *must* load `setspace` *before* `footmisc`.

`memoir class` The class emulates `setspace`, and we detect that emulation and deal with it in the same way as `setspace`.

`manyfoot` The `manyfoot` package permits several independent sequences of footnotes. Some preliminary work towards interworking with `footmisc` has been completed, but more remains to be done at the time of writing.

`hyperref` The `hyperref` package works together with `footmisc` (as proved by this documentation), but at this point in time not all options of `footmisc` can be used — this will change over time.

¹This command is already provided by the L^AT_EX format.

3 Code: Preliminaries

Well — here we go: let’s make the package file:

```
1 (*package)
   Now declare what environment we need: version 6 needs a fairly recent LATEX.
2 \NeedsTeXFormat{LaTeX2e}[2020/10/01]
   We need a token register in case we have to patch \@makecol:
3 \newtoks\FN@temptoken
```

```
\protected@writeaux This command is defined for future compatibility with Matt Swift’s newclude pack-
age (still, after all this time, not out of beta status).
4 \providecommand\protected@writeaux{%
5 \protected@write\auxout
6 }
```

```
\l@advance@macro We make the following (\@@dvance@macro) generalizable as follows (the global
\@@dvance@macro form isn’t used in this package ... yet):
\advance@macro
7 \def\l@advance@macro{\@@dvance@macro\edef}
8 \def\@@dvance@macro#1#2#3{\expandafter\@tempcnta#2\relax
9 \advance\@tempcnta#3\relax
10 #1#2{\the\@tempcnta}%
11 }
```

Now we define a jolly little macro to advance a macro count (#1) by a given amount (#2).

```
12 \let\@advance@macro\l@advance@macro
```

```
\footnotemargin Finally, we define the length used by the marginal option, and initialize it as if
we’ve not had the option.
```

```
13 \newdimen\footnotemargin
14 \footnotemargin1.8em\relax
```

4 Package options

Most of the code of the package is contained within the option processing, one way or another (that which isn’t, is executed after `\ProcessOptions` as a result of flags set in the option processing).

4.1 The `symbol` option

This is a declaration that appears in the original L^AT_EX book. Since it appeared in the old `pagefoots.sty` (presumably since it goes so naturally with the `perpage` option), I’ve added this trivial piece of customization to the package.

```
15 \DeclareOption{symbol}{\renewcommand\thefootnote{\fnsymbol{footnote}}}
```

4.2 The `symbol*` option

The robust version of the `symbol` option: if the current ‘`symbol`’ option doesn’t provide enough variants, use arabic footnote number. We use a robust version of the “extended ordinary” symbol set, described later (in section 1.7).

```
16 \newif\ifFN@robust \FN@robustfalse
17 \DeclareOption{symbol*}{%
18   \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
19   \FN@robusttrue
20 \AtEndOfPackage{\setfnsymbol{lamport*-robust}}%
21 }
```

4.3 The `para` option

The basis of the code for this option comes from `TEXbook`, p.398 ff. (“Dirty Tricks”), though it does (of course) avoid redefining `\` which has some other (somewhat significant) uses in `LATEX`! The user should be aware of Knuth’s note on the limitations of this method of doing the job: the `TEX` stack is used four times per footnote, and the stack is limited (see the `TEXbook`, p.300 ff.). If you have very large numbers of footnotes (in the hundreds), and encounter the error “! TeX capacity exceeded, sorry (... save size ...)”, you may need to break your text into smaller sections and compile the separately. Fortunately (say the comments on the original `fnpara.sty`) this is very easy to do with `LATEX`, provided that you reset the footnote counter to make the joins seamless.

`\ifFN@para` Define the `para` option: now simply sets a marker for use later when defining the option’s auxiliary code and when patching the output routine and so on.

```
22 \newif\ifFN@para \FN@parafalse
23 \DeclareOption{para}{\ifFN@sidefn
24   \PackageError{footmisc}{Option "\CurrentOption" incompatible with
25     option "side"%
26     {I shall ignore "\CurrentOption"}}%
27   \else
28     \FN@paratrue
29   \fi
30 }
```

4.4 The `side` option

`\ifFN@sidefn` Simply changes the behavior of `\@footnotetext`; incompatible with paragraph footnotes.

```
31 \newif\ifFN@sidefn \FN@sidefnfalse
32 \DeclareOption{side}{\ifFN@para
33   \PackageError{footmisc}{Option "\CurrentOption" incompatible with
34     option "para"%
35     {I shall ignore "\CurrentOption"}}%
36   \else
37     \FN@sidefntrue
38   \fi
39 }
```

4.5 The ragged option

`\footnotelayout` A very simple option that merely changes the definition of one macro. Note detection of the presence of the `ragged2e` package.

```
40 \let\footnotelayout\@empty
41 \DeclareOption{ragged}{%
42   \@ifundefined{RaggedRight}{%
43     {\renewcommand\footnotelayout{\linepenalty50 \raggedright}}%
44     {\renewcommand\footnotelayout{\linepenalty50 \RaggedRight}}%
45 }
```

4.6 The perpage option

`\ifFN@perpage` A footnote-numbering modification: a new algorithm replacing one from Brian T. Schellenberger, which has proved to be flawed. We simply set a marker here, and define code later depending on the state of the marker (see section 5.6).

```
46 \newif\ifFN@perpage
47 \FN@perpagefalse
48 \DeclareOption{perpage}{%
49   \FN@perpagetrue
50 }
```

4.7 The PPdebug option

`\ifFN@pp@debug` Sets a flag; the messages are generated in various places throughout the code. The option is not available in the package as distributed: modify the `.ins` file to generate a version of the package that includes the option, if you feel you need it.

```
51 (*PPdebug)
52 \newif\ifFN@pp@debug \FN@pp@debugfalse
53 \DeclareOption{PPdebug}{\FN@pp@debugtrue}
54 \PPdebug)
```

4.8 Fixing the L^AT_EX misbehavior with respect to spacing

`\ifFN@fixskip` We maintain a boolean to decide if we want to fix that, by default we don't but if any placement option is given we apply the fix.

```
55 \newif\ifFN@fixskip \FN@fixskipfalse
```

4.9 The footnote/float placement options

We have up to three blocks on a page (four if you count top-floats but they don't matter here). If there is any excess space that needs to be added the question is where that goes:

1. above footnotes and floats;
2. between footnotes and floats;
3. after footnotes and floats;
4. nowhere in particular (everything is equally spaced out if `\flushbottom` is in force and close together otherwise).

We handle that with a 3-way switch differentiating the different bottom cases: `bottom`, `bottomfloats` or neither of the two options. Within those with split the coding based on whether or not `abovefloats` was given (explicitly or implicitly).

```
\FN@bottomcases We record in which case we want to be in \FN@bottomcases. The default is case
3 (no option).
56 \let\FN@bottomcases\thr@@
```

4.9.1 The `abovefloats`, `belowfloats` options

```
\ifFN@abovefloats All this needs to do is to set a flag to say that it should happen.
57 \newif\ifFN@abovefloats \FN@abovefloatstrue
```

4.9.2 The `bottom` option

```
\ds@bottom The bottom option implements case 1 and puts the footnotes by default below the
floats.
58 \DeclareOption{bottom}{%
59 \let\FN@bottomcases@ne
60 \FN@abovefloatsfalse
We also state that we want to fix LATEX space issue (as we do in all other options.
61 \FN@fixskiptrue
62 }
```

4.9.3 The `bottomfloats` option

```
ds@bottomfloats This option is for case 2. By default the footnotes are above (close to the text).
63 \DeclareOption{bottomfloats}{%
64 \let\FN@bottomcases@tw@
65 \FN@abovefloatstrue \FN@fixskiptrue
66 }
```

```
ds@abovefloats,ds@belowfloats These options change the order and that's it. The important aspect is that they
are declared after the last two, otherwise they can't overwrite them.
67 \DeclareOption{abovefloats}{\FN@abovefloatstrue \FN@fixskiptrue}
68 \DeclareOption{belowfloats}{\FN@abovefloatsfalse \FN@fixskiptrue}
```

4.10 The `marginal` option

Again, the processing of the option is pretty trivial:

```
69 \DeclareOption{marginal}{%
70 \footnotemargin-0.8em\relax
71 }
```

4.11 The `flushmargin` option

Again, the processing of the option is pretty trivial:

```
72 \DeclareOption{flushmargin}{%
73 \footnotemarginOpt\relax
74 }
```

4.12 The hang option

`\ifFN@hangfoot` We need a switch, since `\@makefn` needs to be patched.

```
75 \newif\ifFN@hangfoot \FN@hangfootfalse
76 \DeclareOption{hang}{%
77   \FN@hangfoottrue
78 }
```

`\hangfootparskip` Layout parameters for hanging footnotes; `\hangfootparskip` and `\hangfootparindent`
`\hangfootparindent` are (respectively) values to use for `\parskip` and `\parindent` when in hanging footnotes.

```
79 \newcommand*{\hangfootparskip}{0.5\baselineskip}
80 \newcommand*{\hangfootparindent}{0em}
```

4.13 The norule option

Pretty simple too...

```
81 \DeclareOption{norule}{%
82   \renewcommand\footnoterule{}%
83   \advance\skip\footins 4\p@\@plus2\p@\relax
84 }
```

4.14 The splitrule option

`\split@prev` This is from a posting by Donald Arseneau dated 13 November 1996. The code relies on the fact that L^AT_EX only uses inserts for footnotes, so that if any insert is going to be split, it's going to be a footnote.

```
85 \DeclareOption{splitrule}{%
86   \gdef\split@prev{0}
```

`\pagefootnoterule` Define defaults for the three footnote rules: note, we inherit the current state of
`\mpfootnoterule` `\footnoterule` for the two 'regular' footnote defaults, and if we've been preceded
`\splitfootnoterule` by option `norule`, they will both become null...

```
87 \let\pagefootnoterule\footnoterule
88 \let\mpfootnoterule\footnoterule
89 \def\splitfootnoterule{\kern-3\p@ \hrule \kern2.6\p@}
```

Now redefine `\footnoterule` to distinguish the three situations.

```
90 \def\footnoterule{\relax
91   \ifx \@listdepth\@mplistdepth
```

In a minipage

```
92   \mpfootnoterule
93   \else
94     \ifnum\split@prev=\z@
```

Normal footnote on a regular page

```
95     \pagefootnoterule
96     \else
```

Second part of a split footnote

```
97     \splitfootnoterule
98     \fi
```

Remember a split for next page

```
99     \xdef\split@prev{\the\insertpenalties}%
100   \fi
101 }%
102 }
```

`\ifFN@stablefootnote` 4.15 The stable option

Simply set a flag: the code of this gets executed at the very end of the package.

```
103 \newif\ifFN@stablefootnote \FN@stablefootnotefalse
104 \DeclareOption{stable}{\FN@stablefootnotetrue}
```

4.16 The multiple option

`\ifFN@multiplefootnote` Again, simply set a flag, for code that gets executed at the very very very end of the package.

```
105 \newif\ifFN@multiplefootnote \FN@multiplefootnotefalse
106 \DeclareOption{multiple}{\FN@multiplefootnotetrue}
```

4.17 The start of the endgame

Exercise the options that the user has requested...

```
107 \ProcessOptions
```

5 Hacking kernel commands

Various standard commands (some of them internal ones) need to be hacked to achieve our effects, and we do all of this now, according to flags set in option processing.

5.1 The output routine part

We interface with `\@makecol` from the kernel. Eventually this should move directly into the kernel.

In order for other packages to prepend or append code to `\@makecol`, they can use the generic command hooks `cmd/@makecol/before` and `cmd/@makecol/after`, so there is nothing we need to do here.

`\@makecol` `\@makecol` is shortened a lot, basically all the hardwired code in the middle has moved into a configuration point.

```
108 \def \@makecol {%
109   \@kernel@before@cclv
110   \setbox\@outputbox \box\@cclv
```

The only real addition is the next command which either does nothing or removes an infinite glue from the bottom of the `\@outputbox`.

```
111   \@outputbox@removebskip
```

Any “here” floats in the `\@outputbox` are now handled so we recycle their registers and put them back to the `\@freelist`.

```
112 \let\@elt\relax
113 \xdef\@freelist{\@freelist\@midlist}%
114 \global \let \@midlist \@empty
```

Here we have the configurable part.

NOTE: *Interface to configuration points will change in the future*

```
115 \@makecol@appendblocks
```

The we deal with any `\enlargethispage` or run the normal code to build a column.

```
116 \ifvbox\@kludgeins
117   \@makespecialcolbox
118 \else
119   \@makenormalcolbox
120 \fi
121 \global \maxdepth \@maxdepth
122 }
```

`\@outputbox@depth` We need to know the depth of `\@outputbox` once in a while. Rather than using a temp dimen (as it was done in the past), we give it a proper register.

```
123 \newdimen\@outputbox@depth
```

`\@makenormalcolbox` Taken out of `\@makecol` for readability.

```
124 \def \@makenormalcolbox {%
125   \setbox\@outputbox \vbox to\@colht {%
126     \@texttop
127     \@outputbox@depth \dp\@outputbox
128     \unvbox \@outputbox
129     \vskip -\@outputbox@depth
130     \@textbottom
131   }%
132 }
```

`\@makespecialcolbox` Make the colbox when `\enlargethispage` was used.

```
133 \def \@makespecialcolbox {%
134   \@outputbox@append {\vskip-\@outputbox@depth}%
135   \@tempdima \@colht
136   \ifdim \wd\@kludgeins>\z@
137     \advance \@tempdima -\ht\@outputbox
138     \advance \@tempdima \pageshrink
139     \setbox\@outputbox \vbox to \@colht {%
140       \unvbox\@outputbox
141       \vskip \@tempdima
142       \@textbottom
143     }%
144   \else
145     \advance \@tempdima -\ht\@kludgeins
146     \setbox \@outputbox \vbox to \@colht {%
147       \vbox to \@tempdima {%
148         \unvbox\@outputbox
149         \@textbottom}%
150     }
```



```

150     \vss}%
151   \fi
152   {\setbox \@tempboxa \box \@kludgeins}%
153 }

```

`\@outputbox@removebskip` This is really a bug fix for the kernel, but perhaps one has to make it optional because it is in there since day one). If `\raggedbottom` is in force, footnotes get attached to the main galley at a distance of `\footskip` on all pages except on those that are ended by `\newpage` or `\clearpage` where the `\vfil` from `\newpage` pushes the footnotes to the very bottom.

This is kind of a weird difference to a page ending with `\pagebreak`—in that case the page is also run short, but the footnotes are not pushed to the bottom.

This is fixed by `\@outputbox@removebskip` but only if the switch `FN@fixskip` is set to true (which is done whenever `footmisc` is called with with an option specifying the footnote placement, i.e., not in the default case).

```

154 \ifFN@fixskip
155   \def\@outputbox@removebskip{%

```

We first test if we are in a `\raggedbottom` layout. If not we do nothing, but we don't disable the code because `\raggedbottom` may get used only for some parts of the document.

```

156   \ifx\@textbottom\relax \else

```

We then append some negative glue at the end of `\@outputbox` provided it has a glue stretch order of 1 or more (i.e., contains a `fil` or `fill` part).

```

157     \@outputbox@append{%
158       \@tempskipa\lastskip
159       \ifnum \gluestretchorder\@tempskipa>\z@
160         \vskip-\@tempskipa

```

`\@outputbox@reinsertbskip` We also record the value so that it can be reinserted elsewhere. As we have to do this globally, we also need to explicitly reset it if we don't find any such glue.

```

161       \xdef\@outputbox@reinsertbskip
162         {\noexpand\@outputbox@append{\vskip\the\@tempskipa}}%
163     \else
164       \global\let\@outputbox@reinsertbskip\relax
165     \fi
166   }%
167 \fi
168 }

```

We need a trivial top-level definition for `\@outputbox@reinsertbskip` in case the first page has no bottom glue and the command gets called.

```

169 \let\@outputbox@reinsertbskip\relax

```

If nothing should get fixed we set both commands to `\relax`.

```

170 \else
171   \let\@outputbox@removebskip \relax
172   \let\@outputbox@reinsertbskip\relax
173 \fi

```

`\@kernel@before@cclv` These two commands are internal kernel hooks intended for tagging support in case that is active. By default they do nothing (and may have been defined already by `\DocumentMetadata`).

```

174 \providecommand\@kernel@before@cclv{}
175 \providecommand\@kernel@before@footins{}

```

5.2 The output routine configuration components

Here we provide the components that are used to define `\@makecol@appendblocks`.

`@outputbox@append` This general purpose command alters the `\@outputbox` box by appending material to it. As this is a box typesetting operation we make sure that the last line of the box reflects the true depth of the last line (in case that is needed later). We also expose the current depth of `\@outputbox` as `\@outputbox@depth` before unboxing so that its value can be used by `#1` if wanted.

```

176 \def\@outputbox@append #1{%
177     \setbox\@outputbox \vbox {%
178         \boxmaxdepth \@maxdepth
179         \@outputbox@depth\dp\@outputbox      % if needed in #1
180         \unvbox \@outputbox
181         #1%
182     }%
183 }

```

`\@outputbox@appendfootnotes` This command appends the footnotes to the `\@outputbox` (if there are any). If not then it does nothing.

```

184 \def\@outputbox@appendfootnotes {%
185     \ifvoid\footins \else

```

First come two configuration points: what to do if we are in a split footnote situation and a second one that does some manipulation of the `\footins` box before it gets appended.

NOTE: *this code will get revised as part of CP handling in the future*

```

186     \@makecol@handlesplitfootnotes
187     \@makecol@preparefootinshook

```

Then the footnotes are appended:

```

188     \@outputbox@append{%
189         \vskip \skip\footins
190         \@kernel@before@footins
191         \color@begingroup
192         \normalcolor
193         \footnoterule

```

Support for `pdfcolfoot`, eventually this can go once color is properly supported.

```

194         \csname pdfcolfoot@current\endcsname
195         \unvbox \footins
196         \color@endgroup
197     }%
198 \fi
199 }

```

`\@outputbox@attachfloats` Attaching top and bottom floats can usually be done in one go, but for special layouts we might want more control so we provide also separate commands.

```

\@outputbox@attachbottomfloats 200 \let \@outputbox@attachfloats \@combinefloats

```

```

201 \def \@outputbox@attachtopfloats {%
202   \ifx \@toplist\@empty \else \@cflt \fi
203 }
204 \def \@outputbox@attachbottomfloats {%
205   \ifx \@botlist\@empty \else \@cflb \fi
206 }

```

`\@makecol@handlesplitfootnotes` This is only an early draft and doesn't do much. Contains incomplete preparation for tagging commented out.

NOTE: *Interfaces and code will change in the future*

```

207 \def \@makecol@handlesplitfootnotes {%
208 %   \ifx\splitfootnote@continuation\@empty \else
209 %     \setbox\footins\vbox{\splitfootnote@continuation\unvbox\footins}%
210 %     \global\let\splitfootnote@continuation\@empty
211 %   \fi
212   \ifnum\insertpenalties>\z@
213     \@makecol@splitfootnotemessagehook
214 %     \setbox\footins\vbox{\unvbox\footins --- END at split}%
215 %     \gdef\splitfootnote@continuation {--- START after split}%
216   \fi
217 }
218 %\def\splitfootnote@continuation{}

```

This could issue warning if split footnotes are encountered.

```

219 \let \@makecol@splitfootnotemessagehook \@empty

```

`\@makecol@preparefootinshook` Configuration point to support manipulation of footins box (result needs to be moved back in there). Used by the `para` option.

NOTE: *Interface will change in the future*

```

220 \let \@makecol@preparefootinshook \@empty

```

Footnote box layout for `para` footnotes; this would also be the hook to support `dblfootnotes` (from the `dblfnote` package if we integrate that).

```

221 \ifFN@para
222   \def \@makecol@preparefootinshook {%
223     \global\setbox\footins\vbox{\FN@makefootnoteparagraph}%
224   }
225 \fi

```

NOTE: *Some temp interfaces until configuration points are available.*

`\@if@flushbottom@TF` Test for `\flushbottom` (currently not used).

```

226 \def \@if@flushbottom@TF{%
227   \ifx\@textbottom\relax
228     \expandafter\@firstoftwo
229   \else
230     \expandafter\@secondoftwo
231   \fi
232 }

```

`\@if@footnotes@TF` Test if footnotes are present on the current page.

```
233 \def\@if@footnotes@TF{%
234   \ifvoid\footins
235     \expandafter\@secondoftwo
236   \else
237     \expandafter\@firstoftwo
238   \fi
239 }
```

`\@if@bfloats@TF` Test if bottom floats are around.

```
240 \def\@if@bfloats@TF{%
241   \ifx \@botlist\@empty
242     \expandafter\@secondoftwo
243   \else
244     \expandafter\@firstoftwo
245   \fi
246 }
```

5.3 The `\@makecol` configuration based on options

Placement of footnotes in relation to main galley and floats is covered by the value of `\FN@bottomcases` (type of bottom option) and the status of the switch `@abovefloats`.

```
247 \ifcase \FN@bottomcases\relax
248 %-----
249 % 0 = undefined
250 %-----
251 \ERROR
252 \or
253 %-----
254 % 1 = bottom option given
255 %-----
```

All excess space are above the footnote and bottom float blocks. The order of the blocks depend on `@abovefloats`:

```
256 \ifFN@abovefloats
257 %-----
```

If footnotes above floats floats both are at the bottom:

```
258   \def\@makecol@appendblocks {%
259     \@if@footnotes@TF
260       {\@outputbox@append{\vfill}}%
261       {\@if@bfloats@TF{\@outputbox@append{\vfill}}%
262         {\@outputbox@reinsertbskip}}%
263     \@outputbox@appendfootnotes
264     \@outputbox@attachfloats
265   }
266 %-----
```

```
267 \else
```

Otherwise only the footnotes are at the very bottom and floats stay close to the text:

```
268   \def\@makecol@appendblocks {%
269     \@outputbox@attachfloats
```

```

270     \@if@footnotes@TF
271         {\@outputbox@append{\vfill}}%
272         {\@outputbox@reinsertbskip}%
273     \@outputbox@appendfootnotes
274 }
275 \fi
276 \or
277 %-----
278 % 2 = bottomfloats option given
279 %-----
280     \ifFN@abovefloats
281 %-----

```

Footnotes first then space then floats at bottom:

```

282     \def\@makecol@appendblocks {%
283         \@outputbox@appendfootnotes
284         \@if@bfloats@TF
285             {\@outputbox@append{\vfill}}%
286             {\@outputbox@reinsertbskip}%
287         \@outputbox@attachfloats
288     }
289 %-----
290 \else

```

If belowfloats was given too, then the excess space ends up directly below the text

```

291     \def\@makecol@appendblocks {%
292         \@if@footnotes@TF
293             {\@outputbox@append{\vfill}}%
294             {\@if@bfloats@TF{\@outputbox@append{\vfill}}%
295                 {\@outputbox@reinsertbskip}}%
296         \@outputbox@attachfloats
297         \@outputbox@appendfootnotes
298     }
299 %-----
300 \fi
301 \or
302 %-----
303 % 3 = neither bottom nor bottomfloats given
304 %-----

```

In this case any excess space distribution is handled by `\raggedbottom` or `\flushbottom` settings. In case of `\raggedbottom` it goes to the bottom but we don't append `\vfill` there. Instead we make use of the fact that `\raggedbottom` already puts a stretchable space there, and if we are in a `\flushbottom` scenario then any excess space is supposed to be distributed across the whole page.

```

305     \ifFN@abovefloats
306 %-----
307     \def\@makecol@appendblocks {%
308         \@outputbox@appendfootnotes
309         \@outputbox@attachfloats

```

We do, however, reinsert the bottom skip from `\newpage` if it was taken out earlier. This is, strictly speaking, not necessary in most cases, but it is a `\vfil` while `\raggedbottom` is only generating `\space{0pt plus .0001fil}`, so if you

have several `\vfil` on the page before the `\newpage` you would alter the space distribution if one is taken out.

```

310     \outputbox@reinsertbskip
311   }
312   \else
313 %-----
    Same thing but with blocks swapped.
314   \def\@makecol@appendblocks {%
315     \outputbox@attachfloats
316     \outputbox@appendfootnotes
317     \outputbox@reinsertbskip
318 }
319 %-----
320   \fi
321 \else
322 %-----
323 % 3 > undefined
324 %-----

```

The `\ERROR` here and above should never execute, like “This can’t happen” in the `TEX` program code. If they execute then code is badly broken.

```

325 \ERROR
326 \fi
327

```

5.4 The requirements of `\@footnotetext`

Instead of (re)defining `\@footnotetext` we define `\FN@footnotetext` and at the end we check what we do with it, depending on whether or not `hyperref` was loaded.

```

\ifFN@baselinestretch \FN@singlespace

```

Whatever we do, we are going to patch `\@footnotetext`; so first of all, we’ll check it’s not been hacked by anyone other than `setspace.sty` (while we’re at it we also record whether `setspace` is loaded). so we do this here:

```

328 \newif\ifFN@setspace
329 \@ifpackageloaded{setspace}{%
330   \FN@setspacetrue
331   \@ifclassloaded{memoir}{%
    we’re seeing memoir’s emulation of setspace
332     \let\FN@baselinestretch\m@m@singlespace
333   }{%
    we’re seeing setspace in its own right
334     \let\FN@baselinestretch\setspace@singlespace
335   }%
336 }{%
337   \FN@setspacefalse
338 }

```

There’s substantial patching to be done if we’re doing paragraph footnotes:

```

339 \ifFN@para
340   \long\def\FN@footnotetext#1{%
341     \insert\footins{%

```

insert compatibility code with `setspace.sty` if necessary

```
342     \ifFN@setspace
343         \let\baselinestretch\FN@baselinestretch
344     \fi
345     \reset@font\footnotesize
346     \interlinepenalty\interfootnotelinepenalty
347     \splittopskip\footnotesep
348     \splitmaxdepth \dp\strutbox
349     \floatingpenalty\@MM
350     \hsize\columnwidth
351     \@parboxrestore
352     \protected@edef\@currentlabel{\csname p@footnote\endcsname\@thefnmark}%
353     \color@begingroup
```

We set the paragraph in an `\hbox` and apply the fudge factor here (these days done with e \TeX methods):

```
354     \setbox\FN@tempboxa\hbox{%
```

This needs a parameter; the rule should be moved to the beginning of the footnote paragraph, but the `\ignorespaces` should be left here.

```
355         \@makefntext{\ignorespaces#1\strut
```

We insert a penalty here to help line breaking in the footnote paragraph; the value is taken from the \TeX book.

```
356             \penalty-10\relax
357             \hskip\footglue
358         }% end of \@makefntext parameter
359     }% end of \hbox
360     \dp\FN@tempboxa\z@
361     \ht\FN@tempboxa\dimexpr\wd\FN@tempboxa *%
362             \footnotebaselineskip / \columnwidth\relax
363     \box\FN@tempboxa
364     \color@endgroup
365 }%
366 \FN@mf@prepare
367 }
```

If we're not doing paragraph footnotes, we now simply tag a `\FN@mf@prepare` command on the end of the definition; of course, there are different definitions according as whether we're using side footnotes...

```
368 \else
369     \ifFN@sidefn
370         \long\def\FN@footnotetext#1{%
371             \marginpar{%
```

insert compatibility code with `setspace.sty` if necessary

```
372         \ifFN@setspace
373             \let\baselinestretch\FN@baselinestretch
374         \fi
375         \reset@font\footnotesize
376         \protected@edef\@currentlabel{%
377             \csname p@footnote\endcsname\@thefnmark
378         }%
379         \color@begingroup
```

```

380     \makefntext{%
381         \ignorespaces#1%
382     }%
383     \color@endgroup
384 }%
385 \FN@mf@prepare
386 }%
387 \else
388     \long\def\FN@footnotetext#1{%
389         \insert\footins{%
insert compatibility code with setspace if necessary
390         \ifFN@setspace
391             \let\baselinestretch\FN@baselinestretch
392         \fi
393         \reset@font\footnotesize
394         \interlinepenalty\interfootnotelinepenalty
395         \splittopskip\footnotesep
396         \splitmaxdepth \dp\strutbox
397         \floatingpenalty\@MM
398         \hsize\columnwidth
399         \@parboxrestore
400         \protected@edef\@currentlabel{%
401             \csname p@footnote\endcsname\@thefnmark
402         }%
403         \color@begingroup
404         \makefntext{%
405             \rule\z@\footnotesep
406             \ignorespaces#1\@finalstrut\strutbox
407         }%
408         \color@endgroup
409     }%
410     \FN@mf@prepare
411 }%
412 \fi
413 \fi

```

5.5 Support code for paragraph footnotes

This code used (most inefficiently) to be in the argument of the `\DeclareOption`; this no doubt comes of that code having been written over Christmas 1993...

Now all executed under the `para` conditional set in the option declaration.

```
414 \ifFN@para
```

```

\FN@tempboxa We need some temporary boxes, and LATEX only defines one
\FN@tempboxb 415 \let\FN@tempboxa\@tempboxa
\FN@tempboxb 416 \newbox\FN@tempboxb
417 \newbox\FN@tempboxc

```

```
\footglue A direct crib from the TEXbook:
```

```
418 \newskip\footglue \footglue=1em plus.3em minus.3em
```

```
\makefntext The standard classes set the footnote mark flush with the text of the footnote, but that's not appropriate for paragraph footnotes, we find.
```


There's not much point in patching this code from the original, since the only things it has in common with the original are the footnote mark and the footnote text (which last is the argument). Note that the `\leavevmode` isn't necessary except in the case of footnotes in minipages, which otherwise end up with the `\makefnmark` being executed in restricted vertical mode, which results in its `\hbox` ending up in a line of its own.

```
419 \long\def\@makefntext#1{\leavevmode
420   \@makefnmark\nobreak
421   \hskip.5em\relax#1%
422 }
```

`\footnotebaselineskip` We need to record a value for the baseline skip when in footnotes:

```
423 \newdimen\footnotebaselineskip
424 {%
425   \footnotesize
426   \global
427   \footnotebaselineskip=\normalbaselineskip
428 }
```

`\FN@makefootnoteparagraph` For use in the output routine

```
429 \long\def\FN@makefootnoteparagraph{\unvbox\footins \FN@makehboxoffboxes
430   \setbox\FN@tempboxa=\hbox{\unhbox\FN@tempboxa \FN@removehboxes}%
```

Now we are ready to set the paragraph:

```
431   \FN@setfootnoteparawidth
432   \@parboxrestore
433   \baselineskip=\footnotebaselineskip
434   \noindent
435   \rule{\z@}{\footnotesep}%
436   \unhbox\FN@tempboxa\par
437 }
```

`\FN@makehboxoffboxes` Support code for `\FN@makefootnoteparagraph`

```
\FN@removehboxes 438 \def\FN@makehboxoffboxes{\setbox\FN@tempboxa=\hbox{}}%
439   \loop
440     \setbox\FN@tempboxb=\lastbox
441     \ifhbox\FN@tempboxb
442       \setbox\FN@tempboxa=\hbox{\box\FN@tempboxb\unhbox\FN@tempboxa}%
443     \repeat
444   }
445 \def\FN@removehboxes{\setbox\FN@tempboxa=\lastbox
446   \ifhbox
447     \FN@tempboxa{\FN@removehboxes}%
448   \unhbox\FN@tempboxa
449   \fi
450 }
451 \fi
```

`\FN@setfootnoteparawidth` What we have to use as the width for the footnote paragraph depends on whether or not we typeset in several columns. If single column or normal two-column is used then the right value is `\columnwidth`. However, inside a `multicols` environment we need to use `\textwidth` as the footnotes there will span across all columns.

To detect if we are inside such an environment we look at `\doublecolnumber` which is only positive if inside such an environment.

```

452 \@ifpackageloaded{multicol}
453   {\def\FN@setfootnoteparawidth
454     {\hspace\ifnum\doublecolnumber>\@ne
455       \textwidth
456       \else \columnwidth \fi}}
457   {\def\FN@setfootnoteparawidth{\hspace\columnwidth}}

```

5.6 The other footnote commands

We delegate the `perpage` option to a different package ...

```

458 \ifFN@perpage
459   \RequirePackage{perpage}
460   \MakePerPage{footnote}

```

Unfortunately `perpage` has a bug and doesn't handle founters correctly which are part of a reset list of another counter, e.g., it doesn't work correctly if you use the `report` class which resets footnotes at each chapter start. As a result the first footnote on the first page of a chapter starts with 2. We therefore alter one L^AT_EX internal if `perpage` is in use:

```

461   \def\@stpelt#1{\global\csname c@#1\endcsname \m@ne
462     \stepcounter{#1}%
463     \pp@fix@MakePerPage{#1}%
464   }
465   \def\pp@fix@MakePerPage#1{%
466     \ifnum \value{#1}>\z@
467       \addtocounter{#1}\m@ne\fi
468   }

```

The above code may look a bit odd: the `\stepcounter` sets the counter to zero and then we alter it if it is not zero. The reason is that `\stepcounter` resets other counters and when `perpage` is loaded this results in updating counters on the reset list to 1 (or to a higher starting value if `\MakePerPage` is used with an optional argument, which is precisely the problem here. By subtracting 1 in that case we set it back to 1 lower than the starting value.

But to make this fully work we also need to update a support command in `perpage`:

```

469   \def\pp@cl@end@iii\stepcounter#1\pp@fix@MakePerPage#2{
470 \fi

```

Finally, if we're not doing paragraph footnotes, we redefine `\@makefntext` to take account of the value of `\footnotemargin`, to impose `\footnotelayout`, and to make the footnote body text hang, if appropriate.

```

471 \ifFN@para
472 \else
    hanging footnote version:
473   \long\def\@makefntext#1{%
474     \ifFN@hangfoot
475       \bgroup

```

get the marker so we can measure it:

```
476     \setbox\@tempboxa\hbox{%
477       \ifdim\footnotemargin>0pt
478         \hb@xt@\footnotemargin{\@makefnmark\hss}%
479       \else
480         \@makefnmark
481       \fi
482     }%
```

use the width of the box to set up hanging (potentially for more than one paragraph); note that the hanging `\parskip` and `\parindent` are set *after* we've executed `\leavevmode(!)`

```
483     \leftmargin\wd\@tempboxa
484     \rightmargin\z@
485     \linewidth \columnwidth
486     \advance \linewidth -\leftmargin
487     \parshape \@ne \leftmargin \linewidth
488     \footnotesize
```

stop the `\parshape` being overwritten:

```
489     \@setpar{\@@par}}%
```

and finally put the marker in its chosen place:

```
490     \leavevmode
491     \llap{\box\@tempboxa}%
492     \parskip\hangfootparskip\relax
493     \parindent\hangfootparindent\relax
494     \else
```

ordinary (non-hanging) footnote version:

```
495     \parindent1em
496     \noindent
497     \ifdim\footnotemargin>\z@
498       \hb@xt@ \footnotemargin{\hss\@makefnmark}%
499     \else
500       \ifdim\footnotemargin=\z@
501         \llap{\@makefnmark}%
502       \else
503         \llap{\hb@xt@ -\footnotemargin{\@makefnmark\hss}}%
504       \fi
505     \fi
506     \fi
507     \footnotelayout#1%
```

if we're hanging, close the hang group

```
508     \ifFN@hangfoot
509       \par\egroup
510     \fi
511   }
512 \fi
```

6 Remaining requirements

We have to insert the code that executes the `stable` and `multiple` options. Since `stable` may suppress the setting of a footnote altogether, we put the `multiple`

option first, as otherwise we might get isolated superscripted commas that separate footnotes that have otherwise been suppressed.

6.1 The code that executes the multiple option

```

\multiplefootnotemarker This (revised) code derives from a suggestion by Alexander Rozhenko (the
  \multfootsep author of the manyfoot package): the intention is that footmisc and many-
\FN@footnotemark foot should be able to ‘interwork’, in the sense that each would recognize
\FN@mf@prepare the other’s footnote marks and behave appropriately. The trick is that both
\FN@mf@check \footnote and \footnotemark insert a marker (a cancelling pair of kerns of
\multiplefootnotemarker (of opposite signs), which is detected in following
\footnote or \footnotemark commands. Note we have to take special precau-
tions to ensure that the kerns are the last things added to the horizontal list by
the commands.

513 \ifFN@multiplefootnote
514 \providecommand*\multiplefootnotemarker}{3sp}
515 \providecommand*\multfootsep}{,}
516 %
517 % FMi: not checking, more harm than gain
518 % \CheckCommand*\@footnotemark{%
519 % \leavevmode
520 % \ifhmode\edef\@x@sf{\the\spacefactor}\nobreak\fi
521 % \@makefnmark
522 % \ifhmode\spacefactor\@x@sf\fi
523 % \relax
524 % }
525 %
526 \newcommand*\FN@footnotemark{%
527 \leavevmode
528 \ifhmode
529 \edef\@x@sf{\the\spacefactor}%
530 \FN@mf@check
531 \nobreak
532 \fi
533 \@makefnmark
534 \FN@mf@prepare
535 \ifhmode\spacefactor\@x@sf\fi
536 \relax
537 }
538 \def\FN@mf@prepare{%
539 \kern-\multiplefootnotemarker
540 \kern\multiplefootnotemarker\relax
541 }
542 \def\FN@mf@check{%
543 \ifdim\lastkern=\multiplefootnotemarker\relax
544 \edef\@x@sf{\the\spacefactor}%
545 \unkern
546 \textsuperscript{\multfootsep}%
547 \spacefactor\@x@sf\relax
548 \fi
549 }

```

If we’re not doing multiple, just create an empty \FN@mf@prepare

```

550 \else
551   \let\FN@mf@prepare\relax
    Need to provide a definition for \FN@footnotemark in that case.
552   \let\FN@footnotemark\@footnotemark
553 \fi

```

6.2 The code that executes the stable option

`\ifFN@stablefootnote` The basic idea is to use the ‘original’ code of `\footnote` (which this package may have hacked around something chronic) only if we’re in typesetting mode (as determined by the state of the `\protect` command. Otherwise, the command becomes an elaborate multistage ‘gobble’.

```

554 \ifFN@stablefootnote
555 \let\FN@sf@@footnote\footnote
556 \def\footnote{\ifx\protect\@typeset@protect
557   \expandafter\FN@sf@@footnote
558   \else
559   \expandafter\FN@sf@gobble@opt
560   \fi
561 }

```

`\FN@sf@gobble@opt` Define `\FN@sf@gobble@opt` as a robust command that gobbles either an optional and a mandatory argument, or just a mandatory one.

```

562 \edef\FN@sf@gobble@opt{\noexpand\protect
563   \expandafter\noexpand\csname FN@sf@gobble@opt \endcsname}
564 \expandafter\def\csname FN@sf@gobble@opt \endcsname{%
565   \@ifnextchar [%]
566     \FN@sf@gobble@twobrace
567     \@gobble
568 }
569 \def\FN@sf@gobble@twobrace[#1]#2{}

```

`\FN@sf@@footnotemark` Now the same for `\footnotemark`

```

\FN@sf@gobble@optonly 570 \let\FN@sf@@footnotemark\footnotemark
\FN@sf@gobble@bracket 571 \def\footnotemark{\ifx\protect\@typeset@protect
572   \expandafter\FN@sf@@footnotemark
573   \else
574   \expandafter\FN@sf@gobble@optonly
575   \fi
576 }
577 \edef\FN@sf@gobble@optonly{\noexpand\protect
578   \expandafter\noexpand\csname FN@sf@gobble@optonly \endcsname}
579 \expandafter\def\csname FN@sf@gobble@optonly \endcsname{%
580   \@ifnextchar [%]
581     \FN@sf@gobble@bracket
582     {}%
583 }
584 \def\FN@sf@gobble@bracket[#1]{}
585 \fi

```

7 Symbol option variants

`\setfnsymbol` Lamport’s choice of symbols for `\fnsymbol` wasn’t entirely “traditional”, so we
`\FN@fnsymbol@lamport` (now) provide alternatives. The `\setfnsymbol` command offers a small number
of choices, and the user may define more still, using the `\DefineFnsymbols` or
`\DefineFnsymbolsTM` commands, defined below.

```
586 \newcommand\setfnsymbol[1]{%
587   \@bsphack
588   \@ifundefined{FN@fnsymbol@#1}%
589   {%
590     \PackageError{footmisc}{Symbol style "#1" not known}%
591     \@eha
592   }{%
593     \expandafter\let\expandafter\@fnsymbol\csname
594       FN@fnsymbol@#1\endcsname
595   }%
596   \@esphack
597 }
```

The default selection is Lamport’s original, as represented in current L^AT_EX — we preserve it in case we need to “get back” to it.

```
598 \let\FN@fnsymbol@lamport\@fnsymbol
599 \</package>
```

```
\if@tempwb We need another temp conditional
\@tempwbfalse 600 \newif\if@tempwb
\@tempwbtrue
\DefineFnsymbols The macro \DefineFnsymbols allows the user to define a set of footnote symbols,
\@DefineFnsymbols to be used with the \setfnsymbol command. Syntax:
\@DefineFnsymbols@ \DefineFnsymbols[*]{<set name>}[<style>]{<symbol list>}
\FN@build@symboldef If the optional asterisk is present, the set defined will produce an error if the
symbol number is too large; otherwise it will quietly change to numbering in place
of symbol use (a warning is produced at the end of the document). The set name
is the future argument of \setfnsymbol). The style (default text) gives the style
the symbols are typeset (this is the correct method, but unfortunately not all
symbols, even for Lamport’s original set for LATEX \fnsymbol may be expressed
this way in a sufficiently old LATEX distribution). The symbol list is a set of objects
to be used when the set is selected.
```

Example of use:

define a direct replacement for Lamport’s original `\fnsymbol` command —

```
\DefineFnsymbols*{lamport}[math]{*\dagger\ddagger\mathsection
\mathparagraph\|{**}{\dagger\dagger}{\ddagger\ddagger}}%
}
```

Note that doubled-up (and worse — see below) symbols need braces around them.

```
601 \DeclareDocumentCommand\DefineFnsymbols {sm0{text}m}{%
602   \expandafter\ifx\csname FN@fnsymbol@#2\endcsname\relax
603     \PackageInfo{footmisc}{Declaring symbol style #2}%
604   \else
605     \PackageWarning{footmisc}{Redefining symbol style #2}%
606   \fi
607   \toks@{}}%
```

```

608 \def\@tempb{\end}%
609 \FN@build@symboldef#4\end
610 \def\@tempc{math}%
611 \def\@tempd{#3}%
612 \expandafter\xdef\csname FN@fnsymbol@#2\endcsname##1{%
613   \ifx\@tempc\@tempd
614     \noexpand\ensuremath
615   \else
616     \noexpand\nfss@text
617   \fi
618   {%
619     \noexpand\ifcase##1%
620     \the\toks@
621   \noexpand\else
622     \IfBooleanTF#1{\noexpand\@ctrerr}%
623     {\noexpand\FN@orange##1}%
624     \noexpand\fi
625   }%
626 }%
627 }

628 \def\FN@build@symboldef#1{%
629   \def\@tempa{#1}%
630   \ifx\@tempa\@tempb
631   \else
632     \toks@\expandafter{\the\toks@\or#1}%
633     \expandafter\FN@build@symboldef
634   \fi
635 }

```

\DefineFNsymbolsTM Now do the same job for the “modern” way of having both text and maths variants
\@DefineFNsymbolsTM of everything.
\FN@build@symboldefTM

```

636 \DeclareDocumentCommand\DefineFNsymbolsTM {smm}{%
637   \expandafter\ifx\csname FN@fnsymbol@#2\endcsname\relax
638     \PackageInfo{footmisc}{Declaring symbol style #2}%
639   \else
640     \PackageWarning{footmisc}{Redeclaring symbol style #2}%
641   \fi
642   \toks@{}%
643   \def\@tempb{\end}%
644   \FN@build@symboldefTM#3\end\@null
645   \expandafter\xdef\csname FN@fnsymbol@#2\endcsname##1{%
646     \noexpand\ifcase##1%
647     \the\toks@
648   \noexpand\else
649     \IfBooleanTF#1{\noexpand\@ctrerr}%
650     {\noexpand\FN@orange##1}%
651     \noexpand\fi
652   }%
653 }

```

Note that this version has two variants of every definition, so needs two stopper codes above.

```

654 \def\FN@build@symboldefTM#1#2{%
655   \def\@tempa{#1}%

```

```

656 \ifx\@tempa\@tempb
657 \else
658 \toks@\expandafter{\the\toks@\or\TextOrMath{#1}{#2}}%
659 \expandafter\FN@build@symboldefTM
660 \fi
661 }

```

\FN@orange Macros to deal with footnote symbols going out of range (when they're allowed

\@fnsymbol@orange to—e.g., in the symbol* option).

```

\@diagnose@fnsymbol@orange 662 \def\FN@orange#1{%
663 \ifFN@robust
664 \@arabic#1%
665 \@bsphack
666 \PackageInfo{footmisc}{Footnote number \number#1 out of range}%
667 \protect\@fnsymbol@orange
668 \@esphack
669 \else \@ctrerrr \fi
670 }
671 \global\let\@diagnose@fnsymbol@orange\relax
672 \AtEndDocument{\@diagnose@fnsymbol@orange}
673 \def\@fnsymbol@orange{%
674 \gdef\@diagnose@fnsymbol@orange{%
675 \PackageWarningNoLine{footmisc}{Some footnote number(s)
676 were out of range
677 \MessageBreak
678 see log for details%
679 }%
680 }%
681 }

```

\FN@fnsymbol@bringhurst These macros provide replacement orderings (and symbol sets) for footnote symbols, plus a robust version of the original Lamport set, and an extended version of Lamport's original

\FN@fnsymbol@chicago

\FN@fnsymbol@wiley

```

\FN@fnsymbol@lamport-robust 682 \DefineFNsymbolsTM{bringhurst}{%
\FN@fnsymbol@lamport 683 \textasteriskcentered *%
684 \textdagger \dagger
685 \textdaggerdbl \ddagger
686 \textsection \mathsection
687 \textbardbl \||%
688 \textparagraph \mathparagraph
689 }%
690 \DefineFNsymbolsTM{chicago}{%
691 \textasteriskcentered *%
692 \textdagger \dagger
693 \textdaggerdbl \ddagger
694 \textsection \mathsection
695 \textbardbl \||%
696 \#\#%
697 }%
698 \DefineFNsymbolsTM{wiley}{%
699 \textasteriskcentered *%
700 {\textasteriskcentered\textasteriskcentered}{**}%
701 \textdagger \dagger
702 \textdaggerdbl \ddagger

```



```

703 \textsection \mathsection
704 \textparagraph \mathparagraph
705 \textbardbl \l%
706 }%
707 \DefineFNsymbolsTM{lampport-robust}{%
708 \textasteriskcentered *%
709 \textdagger \dagger
710 \textdaggerdbl \ddagger
711 \textsection \mathsection
712 \textparagraph \mathparagraph
713 \textbardbl \l%
714 {\textasteriskcentered\textasteriskcentered}{**}%
715 {\textdagger\textdagger}{\dagger\dagger}%
716 {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
717 }
718 \DefineFNsymbolsTM{*{lampport*}}{%
719 \textasteriskcentered *%
720 \textdagger \dagger
721 \textdaggerdbl \ddagger
722 \textsection \mathsection
723 \textparagraph \mathparagraph
724 \textbardbl \l%
725 {\textasteriskcentered\textasteriskcentered}{**}%
726 {\textdagger\textdagger}{\dagger\dagger}%
727 {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
728 {\textsection\textsection}{\mathsection\mathsection}%
729 {\textparagraph\textparagraph}{\mathparagraph\mathparagraph}%
730 {\textasteriskcentered\textasteriskcentered\textasteriskcentered}{***}%
731 {\textdagger\textdagger\textdagger}{\dagger\dagger\dagger}%
732 {\textdaggerdbl\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger\ddagger}%
733 {\textsection\textsection\textsection}%%
734 {\mathsection\mathsection\mathsection}%
735 {\textparagraph\textparagraph\textparagraph}%%
736 {\mathparagraph\mathparagraph\mathparagraph}%
737 }
738 \setfnsymbol{lampport*}
739 \DefineFNsymbolsTM{lampport*-robust}{%
740 \textasteriskcentered *%
741 \textdagger \dagger
742 \textdaggerdbl \ddagger
743 \textsection \mathsection
744 \textparagraph \mathparagraph
745 \textbardbl \l%
746 {\textasteriskcentered\textasteriskcentered}{**}%
747 {\textdagger\textdagger}{\dagger\dagger}%
748 {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
749 {\textsection\textsection}{\mathsection\mathsection}%
750 {\textparagraph\textparagraph}{\mathparagraph\mathparagraph}%
751 {\textasteriskcentered\textasteriskcentered\textasteriskcentered}{***}%
752 {\textdagger\textdagger\textdagger}{\dagger\dagger\dagger}%
753 {\textdaggerdbl\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger\ddagger}%
754 {\textsection\textsection\textsection}%%
755 {\mathsection\mathsection\mathsection}%
756 {\textparagraph\textparagraph\textparagraph}%%

```

```

757     {\mathparagraph\mathparagraph\mathparagraph}%
758 }

```

8 Other miscellaneous commands

8.1 Minipage \footnotemarks

`\mpfootnotemark` Syntax: `\mpfootnotemark[number]`
 Here we define `\mpfootnotemark`, which has the same syntax as `\footnotemark`, and which applies the semantics of `\footnotemark` to the minipage footnote series.

```

759 \newcommand\mpfootnotemark{%
760   \@ifnextchar [%
761     \@xmpfootnotemark
762     {%
763       \stepcounter\@mpfn
764       \protected@xdef\@thefnmark{\thempfn}%
765       \@footnotemark
766     }%
767 }
768 \def\@xmpfootnotemark[#1]{%
769   \begingroup
770   \csname c@\@mpfn\endcsname #1\relax
771   \unrestored@protected@xdef\@thefnmark{\thempfn}%
772   \endgroup
773   \@footnotemark
774 }

```

If `hyperref` was loaded first, it has saved `\@footnotetext` and `\@footnotemark` away and then redefined them. The saved versions are now wrong, so we reassign them.

```

775 \@ifpackageloaded{hyperref}{%
776   \let\H@@footnotetext\FN@footnotetext
777   \let\H@@footnotemark\FN@footnotemark
778 }{%

```

If `hyperref` wasn't loaded we copy our new definitions to `\@footnotetext` and `\@footnotemark` for actual use. If `hyperref` is loaded later it will do its magic and save our definitions.

```

779   \let \@footnotetext \FN@footnotetext
780   \let \@footnotemark \FN@footnotemark
781 }
782 \endinput
783 </package>

```