`footmisc` — a portmanteau package for customizing footnotes in LaTeX*

Robin Fairbairns† Frank Mittelbach‡

December 24, 2024

Copyright statement

Contents

*This file has version number v6.0g, last revised 2024/12/24

†Formerly: University of Cambridge Computer Laboratory, Cambridge, UK
‡Responsible maintainer since 2018

History

This package originated as support of a personal project, which I (Robin) was switching to LaTeX 2e 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 LaTeX 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 LaTeX 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" LaTeX installations may have problems with the algorithm for `para` footnotes on very wide pages (for example, those used by the a0poster class). Recent LaTeX 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 LaTeX 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 LaTeX 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 LaTeX 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. LaTeX 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. LaTeX'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)) is established by 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 `\DefineFNsymbols`:

`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 LaTeX 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 of 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`.

1.8.5 Fixing a strange behavior of LaTeX

New: 2022-02

In the default case (if footmisc is not loaded) LaTeX 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 the 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 LaTeX 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:

`\mpfootnoterule`	for use in minipages
`\pagefootnoterule`	for normal footnotes on regular pages
`\splitfootnoterule`	for the tail of a split footnote

By default, `\mpfootnoterule` and `\pagefootnoterule` retain the original definition of `\footnoterule` (which nay 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

mumble[1314]

What the `multiple` option makes of the above is

mumble[13,14]

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 LaTeX 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 LaTeX 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.

[1] This command is already provided by the LaTeX 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 package (still, after all this time, not out of beta status).

```
4 \providecommand\protected@writeaux{%
5   \protected@write\@auxout
6 }
```

`\l@advance@macro` `\@@dvance@macro` `\@advance@macro` We make the following (`\@@dvance@macro`) generalizable as follows (the global form isn't used in this package ... yet):

```
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 LaTeX 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).

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

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.

```
\newif\ifFN@para  \FN@parafalse
\DeclareOption{para}{\ifFN@sidefn
    \PackageError{footmisc}{Option "\CurrentOption" incompatible with
      option "side"}%
      {I shall ignore "\CurrentOption"}%
  \else
    \FN@paratrue
  \fi
}
```

4.4 The `side` option

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

```
\newif\ifFN@sidefn  \FN@sidefnfalse
\DeclareOption{side}{\ifFN@para
    \PackageError{footmisc}{Option "\CurrentOption" incompatible with
      option "para"}%
      {I shall ignore "\CurrentOption"}%
  \else
    \FN@sidefntrue
  \fi
}
```

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.

```
\let\footnotelayout\@empty
\DeclareOption{ragged}{%
  \@ifundefined{RaggedRight}%
    {\renewcommand\footnotelayout{\linepenalty50 \raggedright}}%
    {\renewcommand\footnotelayout{\linepenalty50 \RaggedRight}}%
}
```

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).

```
\newif\ifFN@perpage
\FN@perpagefalse
\DeclareOption{perpage}{%
  \FN@perpagetrue
}
```

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.

```
⟨*PPdebug⟩
\newif\ifFN@pp@debug     \FN@pp@debugfalse
\DeclareOption{PPdebug}{\FN@pp@debugtrue}
⟨/PPdebug⟩
```

4.8 Fixing the LaTeX 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.

```
\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 differenciating 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).

```
\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.

```
\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.

```
\DeclareOption{bottom}{%
  \let\FN@bottomcases\@ne
  \FN@abovefloatsfalse
```

We also state that we want to fix LaTeX space issue (as we do in all other options.

```
  \FN@fixskiptrue
}
```

4.9.3 The `bottomfloats` option

`ds@bottomfloats` This option is for case 2. By default the footnotes are above (close to the text).

```
\DeclareOption{bottomfloats}{%
  \let\FN@bottomcases\tw@
  \FN@abovefloatstrue \FN@fixskiptrue
}
```

`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.

```
\DeclareOption{abovefloats}{\FN@abovefloatstrue  \FN@fixskiptrue}
\DeclareOption{belowfloats}{\FN@abovefloatsfalse \FN@fixskiptrue}
```

4.10 The `marginal` option

Again, the processing of the option is pretty trivial:

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

4.11 The `flushmargin` option

Again, the processing of the option is pretty trivial:

```
\DeclareOption{flushmargin}{%
  \footnotemargin0pt\relax
}
```

4.12 The hang option

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

```
\newif\ifFN@hangfoot  \FN@hangfootfalse
\DeclareOption{hang}{%
  \FN@hangfoottrue
}
```

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

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

4.13 The norule option

Pretty simple too...

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

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 LaTeX only uses inserts for footnotes, so that if any insert is going to be split, it's going to be a footnote.

```
\DeclareOption{splitrule}{%
  \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...

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

Now redefine \footnoterule to distinguish the three situations.

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

In a minipage

```
      \mpfootnoterule
    \else
      \ifnum\split@prev=\z@
```

Normal footnote on a regular page

```
        \pagefootnoterule
      \else
```

Second part of a split footnote

```
        \splitfootnoterule
      \fi
```

Remember a split for next page

```
      \xdef\split@prev{\the\insertpenalties}%
    \fi
  }%
}
```

\ifFN@stablefootnote

4.15 The `stable` option

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

```
\newif\ifFN@stablefootnote  \FN@stablefootnotefalse
\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.

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

4.17 The start of the endgame

Exercise the options that the user has requested...

```
\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.

```
\def \@makecol {%
  \@kernel@before@cclv
  \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`.

```
  \@outputbox@removebskip
```

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

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

Here we have the configurable part.

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

```
\@makecol@appendblocks
```

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

```
\ifvbox\@kludgeins
   \@makespecialcolbox
\else
   \@makenormalcolbox
\fi
\global \maxdepth \@maxdepth
}
```

\@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.

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

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

```
\def \@makenormalcolbox {%
   \setbox\@outputbox \vbox to\@colht {%
       \@texttop
       \@outputbox@depth \dp\@outputbox
       \unvbox \@outputbox
       \vskip -\@outputbox@depth
       \@textbottom
      }%
}
```

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

```
\def \@makespecialcolbox {%
   \@outputbox@append {\vskip-\@outputbox@depth}%
   \@tempdima \@colht
   \ifdim \wd\@kludgeins>\z@
     \advance \@tempdima -\ht\@outputbox
     \advance \@tempdima \pageshrink
     \setbox\@outputbox \vbox to \@colht {%
       \unvbox\@outputbox
       \vskip \@tempdima
       \@textbottom
       }%
   \else
     \advance \@tempdima -\ht\@kludgeins
     \setbox \@outputbox \vbox to \@colht {%
       \vbox to \@tempdima {%
         \unvbox\@outputbox
         \@textbottom}%
```

```
      \vss}%
  \fi
  {\setbox \@tempboxa \box \@kludgeins}%
}
```

`\@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).

```
\ifFN@fixskip
  \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.

```
    \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).

```
      \@outputbox@append{%
        \@tempskipa\lastskip
        \ifnum \gluestretchorder\@tempskipa>\z@
          \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.

```
          \xdef\@outputbox@reinsertbskip
              {\noexpand\@outputbox@append{\vskip\the\@tempskipa}}%
        \else
          \global\let\@outputbox@reinsertbskip\relax
        \fi
      }%
  \fi
}
```

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

```
\let\@outputbox@reinsertbskip\relax
```

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

```
\else
  \let\@outputbox@removebskip \relax
  \let\@outputbox@reinsertbskip\relax
\fi
```

`\@kernel@before@cclv` `\@kernel@before@footins` 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`).

```
\providecommand\@kernel@before@cclv{}
\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.

```
\def\@outputbox@append #1{%
     \setbox\@outputbox \vbox {%
       \boxmaxdepth \@maxdepth
       \@outputbox@depth\dp\@outputbox      % if needed in #1
       \unvbox \@outputbox
       #1%
     }%
}
```

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

```
\def\@outputbox@appendfootnotes {%
  \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*

```
    \@makecol@handlesplitfootnotes
    \@makecol@preparefootinshook
```

Then the footnotes are appended:

```
    \@outputbox@append{%
      \vskip \skip\footins
      \@kernel@before@footins
      \color@begingroup
        \normalcolor
        \footnoterule
```

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

```
        \csname pdfcolfoot@current\endcsname
        \unvbox \footins
      \color@endgroup
     }%
  \fi
}
```

`\@outputbox@attachfloats` `\@outputbox@attachtopfloats` `\@outputbox@attachbottomfloats` 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. There are packages out there that patch `\@combinefloats` so we are careful to call it rather than give it a new name.

```
\def \@outputbox@attachfloats {\@combinefloats}
```

```
\def \@outputbox@attachtopfloats {%
  \ifx \@toplist\@empty \else \@cflt \fi
}
\def \@outputbox@attachbottomfloats {%
    \ifx \@botlist\@empty \else \@cflb \fi
}
```

`\@makecol@handlesplitfootnotes` `\@makecol@splitfootnotemessagehook` 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*

```
\def\@makecol@handlesplitfootnotes {%
%  \ifx\splitfootnote@continuation\@empty \else
%    \setbox\footins\vbox{\splitfootnote@continuation\unvbox\footins}%
%    \global\let\splitfootnote@continuation\@empty
%  \fi
  \ifnum\insertpenalties>\z@
    \@makecol@splitfootnotemessagehook
%    \setbox\footins\vbox{\unvbox\footins --- END at split}%
%    \gdef\splitfootnote@continuation    {--- START after split}%
  \fi
}
%\def\splitfootnote@continuation{}
```

This could issue warning if split footnotes are encountered.

```
\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*

```
\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).

```
\ifFN@para
  \def\@makecol@preparefootinshook {%
     \global\setbox\footins\vbox{\FN@makefootnoteparagraph}%
    }
\fi
```

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

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

```
\def\@if@flushbottom@TF{%
  \ifx\@textbottom\relax
    \expandafter\@firstoftwo
  \else
    \expandafter\@secondoftwo
  \fi
}
```

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

```
\def\@if@footnotes@TF{%
  \ifvoid\footins
    \expandafter\@secondoftwo
  \else
    \expandafter\@firstoftwo
  \fi
}
```

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

```
\def\@if@bfloats@TF{%
  \ifx \@botlist\@empty
    \expandafter\@secondoftwo
  \else
    \expandafter\@firstoftwo
  \fi
}
```

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`.

```
\ifcase \FN@bottomcases\relax
%------------------------------------------------------------
% 0 = undefined
%------------------------------------------------------------
\ERROR
\or
%------------------------------------------------------------
% 1 = bottom option given
%------------------------------------------------------------
```

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

```
  \ifFN@abovefloats
%------------------------------------------------------------
```

If footnotes above floats floats both are at the bottom:

```
    \def\@makecol@appendblocks {%
       \@if@footnotes@TF
          {\@outputbox@append{\vfill}}%
          {\@if@bfloats@TF{\@outputbox@append{\vfill}}%
                          {\@outputbox@reinsertbskip}}%
       \@outputbox@appendfootnotes
       \@outputbox@attachfloats
      }
%------------------------------------------------------------
  \else
```

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

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

```
      \@if@footnotes@TF
         {\@outputbox@append{\vfill}}%
         {\@outputbox@reinsertbskip}%
      \@outputbox@appendfootnotes
    }
  \fi
\or
%-------------------------------------------------------
% 2 = bottomfloats option given
%-------------------------------------------------------
  \ifFN@abovefloats
%-------------------------------------------------------
```

Footnotes first then space then floats at bottom:

```
    \def\@makecol@appendblocks {%
       \@outputbox@appendfootnotes
       \@if@bfloats@TF
           {\@outputbox@append{\vfill}}%
           {\@outputbox@reinsertbskip}%
       \@outputbox@attachfloats
    }
%-------------------------------------------------------
  \else
```

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

```
    \def\@makecol@appendblocks {%
      \@if@footnotes@TF
         {\@outputbox@append{\vfill}}%
         {\@if@bfloats@TF{\@outputbox@append{\vfill}}%
                         {\@outputbox@reinsertbskip}}%
       \@outputbox@attachfloats
       \@outputbox@appendfootnotes
    }
%-------------------------------------------------------
  \fi
\or
%-------------------------------------------------------
% 3 = neither bottom nor bottomfloats given
%-------------------------------------------------------
```

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.

```
  \ifFN@abovefloats
%-------------------------------------------------------
    \def\@makecol@appendblocks {%
       \@outputbox@appendfootnotes
       \@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 `\vspace{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.

```
      \@outputbox@reinsertbskip
    }
  \else
%-------------------------------------------------------------
```

Same thing but with blocks swapped.

```
    \def\@makecol@appendblocks {%
      \@outputbox@attachfloats
      \@outputbox@appendfootnotes
      \@outputbox@reinsertbskip
}
%-------------------------------------------------------------
  \fi
\else
%-------------------------------------------------------------
% 3 > undefined
%-------------------------------------------------------------
```

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.

```
\ERROR
\fi

```

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:

```
\newif\ifFN@setspace
\@ifpackageloaded{setspace}{%
  \FN@setspacetrue
  \@ifclassloaded{memoir}{%
```

we're seeing memoir's emulation of setspace

```
    \let\FN@baselinestretch\m@m@singlespace
  }{%
```

we're seeing setspace in its own right

```
    \let\FN@baselinestretch\setspace@singlespace
  }%
}{%
  \FN@setspacefalse
}
```

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

```
\ifFN@para
  \long\def\FN@footnotetext#1{%
    \insert\footins{%
```

insert compatibility code with `setspace.sty` if necessary

```
      \ifFN@setspace
        \let\baselinestretch\FN@baselinestretch
      \fi
      \reset@font\footnotesize
      \interlinepenalty\interfootnotelinepenalty
      \splittopskip\footnotesep
      \splitmaxdepth \dp\strutbox
      \floatingpenalty\@MM
      \hsize\columnwidth
      \@parboxrestore
      \def\@currentcounter{footnote}%
      \protected@edef\@currentlabel{\csname p@footnote\endcsname\@thefnmark}%
      \color@begingroup
```

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

```
        \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.

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

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

```
            \penalty-10\relax
            \hskip\footglue
          }% end of \@makefntext parameter
        }% end of \hbox
        \dp\FN@tempboxa\z@
        \ht\FN@tempboxa\dimexpr\wd\FN@tempboxa *%
                        \footnotebaselineskip / \columnwidth\relax
        \box\FN@tempboxa
      \color@endgroup
    }%
    \FN@mf@prepare
  }
```

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...

```
\else
  \ifFN@sidefn
    \long\def\FN@footnotetext#1{%
      \marginpar{%
```

insert compatibility code with `setspace.sty` if necessary

```
        \ifFN@setspace
          \let\baselinestretch\FN@baselinestretch
        \fi
        \reset@font\footnotesize
        \def\@currentcounter{footnote}%
        \protected@edef\@currentlabel{%
```

```
          \csname p@footnote\endcsname\@thefnmark
        }%
        \color@begingroup
          \@makefntext{%
            \ignorespaces#1%
          }%
        \color@endgroup
      }%
      \FN@mf@prepare
    }%
  \else
    \long\def\FN@footnotetext#1{%
      \insert\footins{%
```

insert compatibility code with setspace if necessary

```
        \ifFN@setspace
          \let\baselinestretch\FN@baselinestretch
        \fi
        \reset@font\footnotesize
        \interlinepenalty\interfootnotelinepenalty
        \splittopskip\footnotesep
        \splitmaxdepth \dp\strutbox
        \floatingpenalty\@MM
        \hsize\columnwidth
        \@parboxrestore
        \def\@currentcounter{footnote}%
        \protected@edef\@currentlabel{%
          \csname p@footnote\endcsname\@thefnmark
        }%
        \color@begingroup
          \@makefntext{%
            \rule\z@\footnotesep
            \ignorespaces#1\@finalstrut\strutbox
          }%
        \color@endgroup
      }%
      \FN@mf@prepare
    }%
  \fi
\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.

```
\ifFN@para
```

`\FN@tempboxa` We need some temporary boxes, and LaTeX only defines one

`\FN@tempboxb`

`\FN@tempboxb`

```
  \let\FN@tempboxa\@tempboxa
  \newbox\FN@tempboxb
  \newbox\FN@tempboxc
```

\footglue A direct crib from the TEXbook:

```
\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.

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

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

```
\newdimen\footnotebaselineskip
{%
  \footnotesize
  \global
    \footnotebaselineskip=\normalbaselineskip
}
```

\FN@makefootnoteparagraph For use in the output routine

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

Now we are ready to set the paragraph:

```
  \FN@setfootnoteparawidth
  \@parboxrestore
  \baselineskip=\footnotebaselineskip
  \noindent
  \rule{\z@}{\footnotesep}%
  \unhbox\FN@tempboxa\par
}
```

\FN@makehboxofhboxes Support code for \FN@makefootnoteparagraph

\FN@removehboxes

```
\def\FN@makehboxofhboxes{\setbox\FN@tempboxa=\hbox{}%
  \loop
    \setbox\FN@tempboxb=\lastbox
    \ifhbox\FN@tempboxb
    \setbox\FN@tempboxa=\hbox{\box\FN@tempboxb\unhbox\FN@tempboxa}%
  \repeat
}
\def\FN@removehboxes{\setbox\FN@tempboxa=\lastbox
  \ifhbox
    \FN@tempboxa{\FN@removehboxes}%
    \unhbox\FN@tempboxa
  \fi
}
\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.

```
\@ifpackageloaded{multicol}
  {\def\FN@setfootnoteparawidth
    {\hsize\ifnum\doublecol@number>\@ne
                  \textwidth
            \else \columnwidth \fi}}
  {\def\FN@setfootnoteparawidth{\hsize\columnwidth}}
```

5.6 The other footnote commands

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

```
\ifFN@perpage
  \RequirePackage{perpage}
  \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 LaTeX internal if perpage is in use:

```
  \def\@stpelt#1{\global\csname c@#1\endcsname \m@ne
    \stepcounter{#1}%
    \pp@fix@MakePerPage{#1}%
  }
  \def\pp@fix@MakePerPage#1{%
      \ifnum \value{#1}>\z@
        \addtocounter{#1}\m@ne\fi
  }
```

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:

```
  \def\pp@cl@end@iii\stepcounter#1\pp@fix@MakePerPage#2{}
\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.

```
\ifFN@para
\else
```

hanging footnote version:

```
  \long\def\@makefntext#1{%
```

```
    \ifFN@hangfoot
      \bgroup
```

get the marker so we can measure it:

```
      \setbox\@tempboxa\hbox{%
        \ifdim\footnotemargin>0pt
          \hb@xt@\footnotemargin{\@makefnmark\hss}%
        \else
          \@makefnmark
        \fi
      }%
```

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`(!)

```
      \leftmargin\wd\@tempboxa
      \rightmargin\z@
      \linewidth \columnwidth
      \advance \linewidth -\leftmargin
      \parshape \@ne \leftmargin \linewidth
```

We also update `\@totalleftmargin` so that display environments, such as `quote` if used inside the footnote know about the hanging indentation (otherwise something like `quote` isn't centered in the available space):

```
      \@totalleftmargin \leftmargin
      \footnotesize
```

stop the `\parshape` being overwritten:

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

and finally put the marker in its chosen place:

```
      \leavevmode
      \llap{\box\@tempboxa}%
      \parskip\hangfootparskip\relax
      \parindent\hangfootparindent\relax
    \else
```

ordinary (non-hanging) footnote version:

```
      \parindent1em
      \noindent
      \ifdim\footnotemargin>\z@
        \hb@xt@ \footnotemargin{\hss\@makefnmark}%
      \else
        \ifdim\footnotemargin=\z@
          \llap{\@makefnmark}%
        \else
          \llap{\hb@xt@ -\footnotemargin{\@makefnmark\hss}}%
        \fi
      \fi
    \fi
    \footnotelayout#1%
```

if we're hanging, close the hang group

```
    \ifFN@hangfoot
      \par\egroup
    \fi
```

```
  }
\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` `\multfootsep` `\FN@footnotemark` `\FN@mf@prepare` `\FN@mf@check`

This (revised) code derives from a suggestion by Alexander Rozhenko (the author of the *manyfoot* package): the intention is that *footmisc* and *manyfoot* should be able to 'interwork', in the sense that each would recognize the other's footnote marks and behave appropriately. The trick is that both `\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 precautions to ensure that the kerns are the last things added to the horizontal list by the commands.

```
\ifFN@multiplefootnote
  \providecommand*{\multiplefootnotemarker}{3sp}
  \providecommand*{\multfootsep}{,}
%
% FMi: not checking, more harm than gain
%  \CheckCommand*\@footnotemark{%
%    \leavevmode
%    \ifhmode\edef\@x@sf{\the\spacefactor}\nobreak\fi
%    \@makefnmark
%    \ifhmode\spacefactor\@x@sf\fi
%    \relax
%  }
%
  \newcommand*\FN@footnotemark{%
    \leavevmode
    \ifhmode
      \edef\@x@sf{\the\spacefactor}%
      \FN@mf@check
      \nobreak
    \fi
    \@makefnmark
    \FN@mf@prepare
    \ifhmode\spacefactor\@x@sf\fi
    \relax
  }
  \def\FN@mf@prepare{%
    \kern-\multiplefootnotemarker
    \kern\multiplefootnotemarker\relax
  }
  \def\FN@mf@check{%
    \ifdim\lastkern=\multiplefootnotemarker\relax
```

```
      \edef\@x@sf{\the\spacefactor}%
      \unkern
      \textsuperscript{\multfootsep}%
      \spacefactor\@x@sf\relax
    \fi
  }
```

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

```
\else
  \let\FN@mf@prepare\relax
```

Need to provide a definition for `\FN@footnotemark` in that case.

```
  \let\FN@footnotemark\@footnotemark
\fi
```

6.2 The code that executes the `stable` option

`\ifFN@stablefootnote` `\FN@sf@@footnote` 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'.

```
\ifFN@stablefootnote
\let\FN@sf@@footnote\footnote
\def\footnote{\ifx\protect\@typeset@protect
    \expandafter\FN@sf@@footnote
  \else
    \expandafter\FN@sf@gobble@opt
  \fi
}
```

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

```
\edef\FN@sf@gobble@opt{\noexpand\protect
  \expandafter\noexpand\csname FN@sf@gobble@opt \endcsname}
\expandafter\def\csname FN@sf@gobble@opt \endcsname{%
  \@ifnextchar[%]
    \FN@sf@gobble@twobracket
    \@gobble
}
\def\FN@sf@gobble@twobracket[#1]#2{}
```

`\FN@sf@@footnotemark` `\FN@sf@gobble@optonly` `\FN@sf@gobble@bracket` Now the same for `\footnotemark`

```
\let\FN@sf@@footnotemark\footnotemark
\def\footnotemark{\ifx\protect\@typeset@protect
    \expandafter\FN@sf@@footnotemark
  \else
    \expandafter\FN@sf@gobble@optonly
  \fi
}
\edef\FN@sf@gobble@optonly{\noexpand\protect
  \expandafter\noexpand\csname FN@sf@gobble@optonly \endcsname}
\expandafter\def\csname FN@sf@gobble@optonly \endcsname{%
  \@ifnextchar[%]
    \FN@sf@gobble@bracket
```

```
    {}%
}
\def\FN@sf@gobble@bracket[#1]{}
\fi
```

7 Symbol option variants

`\setfnsymbol` `\FN@fnsymbol@lamport`

Lamport's choice of symbols for `\fnsymbol` wasn't entirely "traditional", so we (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.

```
\newcommand\setfnsymbol[1]{%
  \@bsphack
  \@ifundefined{FN@fnsymbol@#1}%
  {%
    \PackageError{footmisc}{Symbol style "#1" not known}%
    \@eha
  }{%
    \expandafter\let\expandafter\@fnsymbol\csname
                          FN@fnsymbol@#1\endcsname
  }%
  \@esphack
}
```

The default selection is Lamport's original, as represented in current LaTeX — we preserve it in case we need to "get back" to it.

```
\let\FN@fnsymbol@lamport\@fnsymbol
⟨/package⟩
```

`\if@tempswb` `\@tempswbfalse` `\@tempswbtrue`

We need another temp conditional

```
\newif\if@tempswb
```

`\DefineFNsymbols` `\@DefineFNsymbols` `\@DefineFNsymbols@` `\FN@build@symboldef`

The macro `\DefineFNsymbols` allows the user to define a set of footnote symbols, to be used with the `\setfnsymbol` command. Syntax:

`\DefineFNsymbols[*]{`⟨*set name*⟩`}[`⟨*style*⟩`]{`⟨*symbol list*⟩`}`

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.

```
\DeclareDocumentCommand\DefineFNsymbols {smO{text}m}{%
```

```
  \expandafter\ifx\csname FN@fnsymbol@#2\endcsname\relax
    \PackageInfo{footmisc}{Declaring symbol style #2}%
  \else
    \PackageWarning{footmisc}{Redeclaring symbol style #2}%
  \fi
  \toks@{}%
  \def\@tempb{\end}%
  \FN@build@symboldef#4\end
  \def\@tempc{math}%
  \def\@tempd{#3}%
  \expandafter\xdef\csname FN@fnsymbol@#2\endcsname##1{%
    \ifx\@tempc\@tempd
      \noexpand\ensuremath
    \else
      \noexpand\nfss@text
    \fi
    {%
      \noexpand\ifcase##1%
      \the\toks@
      \noexpand\else
      \IfBooleanTF#1{\noexpand\@ctrerr}%
        {\noexpand\FN@orange##1}%
      \noexpand\fi
    }%
  }%
}
\def\FN@build@symboldef#1{%
  \def\@tempa{#1}%
  \ifx\@tempa\@tempb
  \else
    \toks@\expandafter{\the\toks@\or#1}%
    \expandafter\FN@build@symboldef
  \fi
}
```

`\DefineFNsymbolsTM` `\@DefineFNsymbolsTM` `\FN@build@symboldefTM`

Now do the same job for the "modern" way of having both text and maths variants of everything.

```
\DeclareDocumentCommand\DefineFNsymbolsTM {smm}{%
  \expandafter\ifx\csname FN@fnsymbol@#2\endcsname\relax
    \PackageInfo{footmisc}{Declaring symbol style #2}%
  \else
    \PackageWarning{footmisc}{Redeclaring symbol style #2}%
  \fi
  \toks@{}%
  \def\@tempb{\end}%
  \FN@build@symboldefTM#3\end\@null
  \expandafter\xdef\csname FN@fnsymbol@#2\endcsname##1{%
    \noexpand\ifcase##1%
      \the\toks@
    \noexpand\else
      \IfBooleanTF#1{\noexpand\@ctrerr}%
        {\noexpand\FN@orange##1}%
      \noexpand\fi
  }%
```

```
}
```

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

```
\def\FN@build@symboldefTM#1#2{%
  \def\@tempa{#1}%
  \ifx\@tempa\@tempb
  \else
    \toks@\expandafter{\the\toks@\or\TextOrMath{#1}{#2}}%
    \expandafter\FN@build@symboldefTM
  \fi
}
```

`\FN@orange` `\@fnsymbol@orange` `\@diagnose@fnsymbol@orange`

Macros to deal with footnote symbols going out of range (when they're allowed to – e.g., in the `symbol*` option).

```
\def\FN@orange#1{%
  \ifFN@robust
    \@arabic#1%
    \@bsphack
    \PackageInfo{footmisc}{Footnote number \number#1 out of range}%
    \protect\@fnsymbol@orange
    \@esphack
  \else \@ctrerr \fi
}
\global\let\@diagnose@fnsymbol@orange\relax
\AtEndDocument{\@diagnose@fnsymbol@orange}
\def\@fnsymbol@orange{%
  \gdef\@diagnose@fnsymbol@orange{%
    \PackageWarningNoLine{footmisc}{Some footnote number(s)
      were out of range
      \MessageBreak
      see log for details%
    }%
  }%
}
```

`\FN@fnsymbol@bringhurst` `\FN@fnsymbol@chicago` `\FN@fnsymbol@wiley` `\FN@fnsymbol@lamport-robust` `\FN@fnsymbol@lamport`

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

```
\DefineFNsymbolsTM{bringhurst}{%
  \textasteriskcentered *%
  \textdagger    \dagger
  \textdaggerdbl \ddagger
  \textsection   \mathsection
  \textbardbl    \|%
  \textparagraph \mathparagraph
}%
\DefineFNsymbolsTM{chicago}{%
  \textasteriskcentered *%
  \textdagger    \dagger
  \textdaggerdbl \ddagger
  \textsection   \mathsection
  \textbardbl    \|%
  \#\#%
```

```
}%
\DefineFNsymbolsTM{wiley}{%
  \textasteriskcentered *%
  {\textasteriskcentered\textasteriskcentered}{**}%
  \textdagger    \dagger
  \textdaggerdbl \ddagger
  \textsection   \mathsection
  \textparagraph \mathparagraph
  \textbardbl    \|%
}%
\DefineFNsymbolsTM{lamport-robust}{%
  \textasteriskcentered *%
  \textdagger    \dagger
  \textdaggerdbl \ddagger
  \textsection   \mathsection
  \textparagraph \mathparagraph
  \textbardbl    \|%
  {\textasteriskcentered\textasteriskcentered}{**}%
  {\textdagger\textdagger}{\dagger\dagger}%
  {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
}
\DefineFNsymbolsTM*{lamport*}{%
  \textasteriskcentered *%
  \textdagger    \dagger
  \textdaggerdbl \ddagger
  \textsection   \mathsection
  \textparagraph \mathparagraph
  \textbardbl    \|%
  {\textasteriskcentered\textasteriskcentered}{**}%
  {\textdagger\textdagger}{\dagger\dagger}%
  {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
  {\textsection\textsection}{\mathsection\mathsection}%
  {\textparagraph\textparagraph}{\mathparagraph\mathparagraph}%
  {\textasteriskcentered\textasteriskcentered\textasteriskcentered}{***}%
  {\textdagger\textdagger\textdagger}{\dagger\dagger\dagger}%
  {\textdaggerdbl\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger\ddagger}%
  {\textsection\textsection\textsection}%%
    {\mathsection\mathsection\mathsection}%
  {\textparagraph\textparagraph\textparagraph}%%
    {\mathparagraph\mathparagraph\mathparagraph}%
}
\setfnsymbol{lamport*}
\DefineFNsymbolsTM{lamport*-robust}{%
  \textasteriskcentered *%
  \textdagger    \dagger
  \textdaggerdbl \ddagger
  \textsection   \mathsection
  \textparagraph \mathparagraph
  \textbardbl    \|%
  {\textasteriskcentered\textasteriskcentered}{**}%
  {\textdagger\textdagger}{\dagger\dagger}%
  {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
  {\textsection\textsection}{\mathsection\mathsection}%
  {\textparagraph\textparagraph}{\mathparagraph\mathparagraph}%
```

```
  {\textasteriskcentered\textasteriskcentered\textasteriskcentered}{***}%
  {\textdagger\textdagger\textdagger}{\dagger\dagger\dagger}%
  {\textdaggerdbl\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger\ddagger}%
  {\textsection\textsection\textsection}%%
    {\mathsection\mathsection\mathsection}%
  {\textparagraph\textparagraph\textparagraph}%%
    {\mathparagraph\mathparagraph\mathparagraph}%
}
```

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.

```
\newcommand\mpfootnotemark{%
  \@ifnextchar[%
    \@xmpfootnotemark
    {%
      \stepcounter\@mpfn
      \protected@xdef\@thefnmark{\thempfn}%
      \@footnotemark
    }%
}
\def\@xmpfootnotemark[#1]{%
  \begingroup
    \csname c@\@mpfn\endcsname #1\relax
    \unrestored@protected@xdef\@thefnmark{\thempfn}%
  \endgroup
  \@footnotemark
}
```

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.

```
\@ifpackageloaded{hyperref}{%
  \let\H@@footnotetext\FN@footnotetext
  \let\H@@footnotemark\FN@footnotemark
}{%
```

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.

```
  \let \@footnotetext \FN@footnotetext
  \let\@footnotemark  \FN@footnotemark
}
```

```
\endinput
```

⟨/package⟩