

# The latex-lab-footnotes code\*

Frank Mittelbach, L<sup>A</sup>T<sub>E</sub>X Project

February 22, 2025

## Abstract

*to be written*

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Configuration methods . . . . .	3
<b>2</b>	<b>Sockets and hooks</b>	<b>3</b>
2.1	Formatting the mark in the main text . . . . .	3
2.1.1	Sockets . . . . .	3
2.1.2	Hooks for formatting the footnote mark in text . . . . .	3
2.1.3	Additional configuration possibilities . . . . .	4
2.2	Formatting the footnote text . . . . .	4
2.2.1	Sockets . . . . .	4
2.2.2	Hooks for formatting the footnote text . . . . .	6
2.2.3	Additional configuration possibilities . . . . .	7
2.3	Debugging sockets and hooks . . . . .	7
<b>3</b>	<b>Tagging and hyperlinking support</b>	<b>7</b>
3.1	Technical details for the tagging . . . . .	7
3.2	Requirements for links . . . . .	9
3.3	The algorithmus to connect marks and notes . . . . .	9
3.3.1	\footref . . . . .	10
3.3.2	\footnotemark after \footnotetext . . . . .	10
3.4	Links . . . . .	10
3.5	Implementation details regarding tagging . . . . .	10
3.6	Handling the mark . . . . .	10
3.7	Handling the footnotetext . . . . .	11
3.8	Footnotes in minipages . . . . .	11
<b>4</b>	<b>TODOs</b>	<b>11</b>

---

\*

<b>5</b>	<b>The Implementation</b>	<b>11</b>
5.1	File declaration	11
5.2	code not fully handled yet	11
5.3	Temporary variables	13
5.4	Public variables	13
5.5	Internal variables	13
5.6	Variants	14
5.7	Updating <code>\@thefnmark</code>	14
5.8	Hooks	15
5.9	Debugging code	15
5.10	The new <code>\@footnotemark</code> command	16
5.11	The new <code>\@footnotetext</code> command	17
5.12	The new <code>\@makefnmark</code> command	20
5.12.1	Making documents use the new <code>\@makefnmark</code>	21
5.13	Document-level commands	24
5.14	Firstaid for packages and classes	25
5.15	Kernel patches	25
5.15.1	memoir	25
5.15.2	setspace	25
5.15.3	hyperref	25
5.16	Tagging and hyperlink code	26
5.16.1	Rolemap for structure tags	26
5.16.2	Extending the label system	26
5.16.3	Storing and retrieving reference data	26
5.16.4	Enabling tagging and links for the mark command	28
5.16.5	The footnote text	29
<b>6</b>	<b>Reimplementing the <code>footmisc</code> package</b>	<b>31</b>
	<b>Index</b>	<b>43</b>

## 1 Introduction

This code reimplements the footnote interfaces for  $\text{\LaTeX}$  offering configurable methods for layout and functionality adjustments that avoid overwriting each other when used in classes as well as in packages (as far as possible — obviously some adjustments are mutually exclusive). This is achieved by providing a larger number of hooks (for areas where different packages/classes can easily coexist with their adjustments) and a number of sockets to which only one class or package can write to successfully (in case of multiple changes the last one wins). The latter are for special functionality, e.g., if footnote text is typeset as a single paragraph, it can't be configured the same time to be typeset vertically with one footnote below each other.

The interfaces are set up to support tagged PDF, but in order for this to work, all packages altering the footnote setup should use the interfaces provided here and not do it through the legacy methods (though there is some support for the latter as well, but it will not work in all cases).

## 1.1 Configuration methods

Historically, the footnote setup in L<sup>A</sup>T<sub>E</sub>X was done by providing definitions for `\@makefnmark` (format the footnote mark in running text and in front of the footnote text) and `\@makefntext` (formatting the footnote text and placing a mark in front of it).

There was a default definition for `\@makefnmark` in the format that was used by most document classes, but `\@makefntext` had to be defined in the class itself because the format didn't provide a default. As a result you will find definitions for the latter in all document classes and definitions for `\@makefnmark` only in very few.

Furthermore, to enable special footnote layouts or provide additional functionality a few packages (and a few classes) overwrote other internal commands of L<sup>A</sup>T<sub>E</sub>X's footnote mechanism. The commands affected in this way are mainly `\@footnotemark` and `\@footnotetext`. These overwrites could not be used in combination, so either the packages/classes had to be aware of being loaded together (which they sometimes did or tried to) or they would fail by overwriting each other unconditionally.

The present rewrite is an attempt to improve this situation, but of course, it will only work if all packages/classes make use of the new interfaces. Fortunately, the number of problematical packages altering these internal commands are fairly small so arranging for updates is a realistic goal — to achieve properly tagged PDF it is a requirement.

## 2 Sockets and hooks

We use sockets for those parts that can be controlled only by one package or by the kernel and hooks for places where it may be possible that several packages or the document class adds code (typically declarations such as font changes, etc.).

Note that sockets are of interest only to very few specialized packages, mainly `footmisc`, and packages providing similar functionality—the current documentation is therefore fairly sketchy.

In contrast the hooks are of interest to many classes to provide their layout alterations in a way that it works smoothly with other packages handling aspects of footnote formatting.

### 2.1 Formatting the mark in the main text

This implements formatting the mark<sup>1</sup> and its relation to surrounding text, e.g., if several marks appear in the same place, etc.

#### 2.1.1 Sockets

None: everything is implemented through a single definition for `\@footnotemark` that offers a number of hooks that can be used by packages to implement handling of multiple marks and the formatting of marks.

#### 2.1.2 Hooks for formatting the footnote mark in text

The hooks to customize the marks in the text are the following:

---

<sup>1</sup>Like this one.

### **fnmark/before**

Executed at the very beginning of `\footnotemark`. Currently there are two packages (`bibarts` and `chextras`) that prepend material at this point (not necessarily correctly, e.g., they do not all check that they are in horizontal mode).

This hook is paired with hook `fnmark/after`.

### **fnmark**

Executed in horizontal mode and after the current space factor has been saved away for reuse. This is where currently code for multiple marks does its preparation (as done by `footmisc` and others).

The hook is only executed in hmode, i.e., not if the mark is generated in math — maybe that means the multiple handling should happen later?

After the hook a `\nobreak` is executed, so any “material” added in the hook is tied to the following mark unless it contains its own permissible penalty.

### **fnmark/begin**

This hook is executed directly in front of the typeset mark. This is the place where `hyperref` would have added part of its code, i.e., after the `\nobreak` mentioned above. With the integration of hyperlinks in the tagging code this hook may not be necessary at all.

### **fnmark/end**

This hook is executed directly after the typeset mark. It is used by `memhfixc`, `sctrltr2`, and `footmisc`. Used, for example, to implement support for multiple marks in succession.

It is *not* a reversed hook.

### **fnmark/after**

This hook is executed at the very end of the `\footnotemark` command.

It is a reversed hook to pair with `fnmark/before`

## **2.1.3 Additional configuration possibilities**

The actual formatting is done through `\@makefnmark` — no special customization support for now.

## **2.2 Formatting the footnote text**

This implements the formatting of the footnote text the way it appears at the bottom of the page (default case), or possibly elsewhere, e.g. in the margin.

### **2.2.1 Sockets**

To cater for different layout configurations there are four sockets that can be set by a package or class but there should be only one per document setting them, i.e., if two packages/classes set them they are mutually incompatible (or rather the last one wins most likely). These are:

**fntext/process (1 argument)**

This socket receives all material that is to be processed (or stored) including color protection code and what have you. The `default` executes `\insert\footins`.

Available plugs are `default`, `side` (side notes), and `mp` (minipage).

**fntext/make (1 argument)**

This socket receives the  $\langle text \rangle$  as given in `\footnote` or `\footnotetext` in the document and adds formatting instructions to it.

The `default` plug runs `\@makefntext` which contains various hooks for customization. For most scenarios this is sufficient. However, when running all footnotes as a single paragraph at the bottom, then each footnote needs to be prepared prior to storing it with `\insert` and this socket allows running extra code to do that.

Available plugs are `default` and `para`.

**fntext/begin (no argument)**

The socket is executed near the start of the argument for the `fntext/make` socket. By `default` it adds a strut to the footnote material so that consecutive footnotes are properly spaced vertically. In some use cases this is not appropriate (e.g., when running all footnotes as a single paragraph) and so with this socket one can cancel the action or do something else instead.

Available plugs are `default` and `noop`.

**fntext/end (no argument)**

This socket is executed at the very end of the argument passed to socket `fntext/make`. By `default` it adds a final strut as long as we are still in horizontal mode (i.e., processing the footnote text paragraph). When running several footnotes in one paragraph some additional material (some horizontal glue) needs adding at this point which is done with the plug `para`.

Available plugs are `default`, `para`, and `noop`.

All standard plugs for the socket `fntext/make` run `\@makefntext` and this command contains two further sockets (unless it is overwritten by a legacy class):

**fntext/mark (0 arguments)**

This socket has no input arguments but uses `\@makefnmark` to typeset the mark in front of the footnote text. Its `default` uses code that examines the value of `\footnotemargin` and based on its setting typeset the mark in different ways:

- positive: typeset the mark in a box of that size
- zero: use `\llap` around the mark
- negative: use `\llap` but with a box of the given size negated inside
- `-\maxdimen`: just use `\@makefnmark`

For most cases this would be flexible enough, but if not then a class can define its own plug to specify the placement of the mark.

Available plugs are `default` and `noop` (no mark is produced).

### **fn`text`/text (1 argument)**

This socket manages the formatting of the footnote text (presented as an argument) once the mark has been typeset. In all cases we can think of this formatting is better configured via the available hooks described below, so the `default` just grabs the argument and processes it without any other action. It is really only there to allow for some fancy stuff that some design comes up with.

Available plugs are `identity` (default) and `noop`.

The above configuration points are sufficient to implement all commonly used footnote layouts assuming L-R typesetting. For R-L typesetting they or may or may not need some extension (though that is not clear right now).

## **2.2.2 Hooks for formatting the footnote text**

### **fn`text`/before**

Executed at the very beginning of `\footnotetext`. Currently there is one package (`linguex`) that prepends material at this point.

This hook is paired with hook `fnmark/after`.

### **fn`text`**

Executed at the beginning of the material passed to the first configuration point. Typically used to set any baseline stretch for the footnote text, e.g., by `setspace`, `footmisc`, `uathesis.cls` and others. Could be done in a later hook but is a bit more efficient here.

After the hook has run, the font is established, i.e., it can't be used to set a different font size.

### **fn`text`/para**

After the font is set (after the previous hook), some default paragraph parameters are set up including `\interlinepenalty`, `\hsize`, `\parindent` and a number of others, as some of them depend on the font size. Then the `fntext/para` is run which can overwrite the default. If one wants to change the font size, it is probably necessary to reset these other parameters too, e.g., `\parindent`, which can be done here.

Note: the socket `fntext/make` normally runs the command `\@makefntext` or some code that eventually runs this command, and this then produces the footnote mark in front of the formatted footnote text. In front of both the mark and the footnote text some classes have placed paragraph parameter adjustments in their redefinition of `\@makefntext`. However, there is no need to place it there it could equally well go into the `fntext/para` hook. We therefore do not provide another hook at this other point.

### **fn`text`/begin & fn`text`/end**

The footnote text itself is surrounded by the hooks `fntext/begin` and `fntext/end`. The two hooks are not paired as they are typically used independently.

### **fn`text`/after**

At the very end of `\footnotetext` we execute the hook `fntext/after` which is a reversed hook paired with `fntext/before`. Some packages, e.g., `linguex`, have code in that position.

### 2.2.3 Additional configuration possibilities

The formatting of the footnote mark in front of the footnote text is influenced by the setting of the `dimen` parameter `\footnotemargin`. By default its value is 1.8em in the current text font (or `-\maxdimen` when the `para` option is chosen). The following rules apply:

- If it has the value `-\maxdimen` then the mark is generated by `\@makefnmark`.
- Otherwise, if the value is negative then the mark is placed into an `\llap` left aligned in a box of size `-\footnotemargin`.
- If the value is zero an `\llap` is used without an inner box.
- If the value is greater zero (but less than `\maxdimen`) the mark is placed right aligned into a box of size `\footnotemargin`.
- The value `\maxdimen` is used as a marker to indicate that no value was given and that the default should be used, i.e. 1.8em or `-\maxdimen` depending on the chosen option.

## 2.3 Debugging sockets and hooks

For some rudimentary debugging we currently have `\DebugFNotesOn` (and `\DebugFNotesOff`). At the moment `\DebugFNotesOn` only shows the current settings for hooks and sockets related to the footnote code and then automatically turns itself off again.

## 3 Tagging and hyperlinking support

*TODO: this section needs work (and probably csnames changes)*

Footnotes consist of a *footnotemark* (short: mark) that is typically placed in the text as a superscript number like this<sup>1</sup>, and a *footnotetext* (short: note) that is placed at the bottom of the page. The *footnotetext* normally repeats at the begin the mark as a visual clue.

Tagging (and hyperlinking) has to connect the mark with the note. For the tagging code, we assume that every mark has exactly one associated note, and that every note is associated to at least one mark and can have more associated marks.

The mark doesn't need to be visible, e.g. the typesetted mark<sup>1-3</sup> denotes three marks, where the second is invisible. Tagging should produce here probably three `Lb1` structures (one without content), and an artifact for the range marker. If such a range is used, links can only point to the notes 1 and 3 and one has to suppress the linking for the second mark. This means that links and tagging are also related to the actual formatting of the footnote mark. In the following this problem is mostly ignored for now, but should not be forgotten and handled later.

### 3.1 Technical details for the tagging

The following sockets are set up for kernel use, when doing tagging: Their name and/or function will probably change as they currently mix tagging with the link support.

*TODO: review this sockets*

### **tagsupport/fnmark (1 argument)**

The socket is used in `\@footnotemark/\fnote_footnotemark:` and takes `\@makefnmark` as argument. It prints the mark in the text and surrounds it with a tagging structure and a link. As such it is not solely for tagging and so should not be used with `\UseTaggingSocket` as this would swallow the argument and lose the link support.

TODO: *describe and decide on names*

### **tagsupport/fntext/begin (no argument)**

This socket is used before the main processing socket (so before the `\insert` command). It opens the FEnote structure. As it sets also the `tl-var` for the current structure and this is used in destinations it should not use as tagging socket.

### **tagsupport/fntext/end (no argument)**

This socket is used after the main processing socket (so after the `\insert` command). It closes the FEnote structure.

### **tagsupport/fntext/mark (1 argument)**

This socket is used around the mark in the footnote text. It adds tagging support but also link support, so like the other tagsupport sockets it should be always active.

### **tagsupport/fntext/text (1 argument)**

This socket handles mc-chunks around the text of the footnote. As it takes an argument (the text) it should not be used as tagging socket either.

The *footnotemark* should create a `/Lb1` structure<sup>2</sup> that should contain a `/Ref` entry pointing to the structure of the *footnotetext*.

The *footnotetext* should create a `/FENote`<sup>3</sup> structure with a `/Ref` entry pointing to the structures of *all* marks related to the note. The mark at the begin of the note is in a `/Lb1`<sup>4</sup> structure but has to fulfil no special requirements.

Structure objects and the underlying properties used by the tagging code are initialized when the structure is opened. This means that one can not directly add data to a future structure but as structure objects are written at the end of the document it is possible to update `/Ref` entries in an end document hook.

So tagging has to solve two problems:

- the mark and the footnote text must be surrounded by the correct structure and marked content commands. This is not trivial as there are various layouts (bottom, marginpar, minipage) and the tagging from the automatic paratagging must be taken into account if one wants to avoid faulty nesting.
- It must detect which marks are related to which notes so that it can setup the `/Ref` cross-references.

---

<sup>2</sup>to make it easier to identify the role we use `/footnotemark` which we rolemap to `/Lb1`

<sup>3</sup>We tag it as `/footnote` and role map it.

<sup>4</sup>We tag it as `/footnotelabel`.



## 3.2 Requirements for links

Links should go from the mark to the note. Sometimes it has been requested that links go back too, but as there can be more than one mark connected to a note it is not clear how to decide to which mark it should go. Using the keys from the PDF viewer to go back is normally better.

Links are closely related to the references stored in the `/Ref` entry of a mark and so are handled in the code together with them. But there are subtle technical differences to take care of as links and destinations are whatsits and so must be created at the correct time.

It should be possible to suppress the links both globally and locally.<sup>5</sup>

## 3.3 The algorithmus to connect marks and notes

The connection is made by comparing the value of `\@thefnmark`.

The standard mark commands (`\footnotemark` and `\footnote`) store the current value of `\@thefnmark` with their own structure number as a key in a property.

A following `\footnotetext` compares its own `\@thefnmark` with the values in the prop. If there is one or more match it stores the structure numbers and removes the entries from the property (so in a normal document the property will never contain more than a few entries).

This works well as long as the `\footnotemark` commands are issued before the `\footnotetext` and as long as nothing unusual is done to `\@thefnmark`. It also works if a document uses more than one footnote series as long as they have distinct numbering systems, but in case a distinction is needed it is possible to define a new class with its own data structure and to switch locally to use this class. The following three commands are used for this.

The default class uses the name `default`

---

`\fnote_class_new:nn` `\fnote_class_new:nn{<name>}{<key/value option>}`

This declaration sets up the needed data structure. Currently this only consists of a property list which is used to store and manage the mark values. There are no options yet.

---

`\fnote_mark_gput:nn` `\fnote_mark_gput:nn{<mark>}{<class name>}`  
`\fnote_mark_gput:(no|oo)`

This command stores the current structure number as key and the `<mark>` as value in the property list associated with the `<class name>`.

---

`\fnote_mark_gpop_all:nnN` `\fnote_mark_gpop_all:nnN{<mark>}{<class name>}{<sequence>}`

This command stores all the keys/structure numbers whose value in the property list for `<class name>` are equal to `<mark>` into the sequence `<sequence>` and then removes them from the property list. The content of the sequence can then be used to create link targets and references.

---

<sup>5</sup>Currently `hyperref` only offers the option to suppress the footnote links globally with the option `hyperfootnotes=false`. To suppress them locally only the `NoHyper` environment is provided.

### 3.3.1 `\footref`

`\footref` uses internally the same command to set the mark as `\footnotemark`, it only defines `\@thefnmark` differently. This `\@thefnmark` is not suitable for the method described above: as it contains a reference command it can't be used to match a note, also `\footref` can be used after the note has already been set. `\footref` disables therefore the automatic detection.

Instead the `\label` command is extended in the `\footnotetext` command to also store the structure number and `\footref` retrieves this number to setup the reference and the link.

The structure related to the `\footref` is added to the end of the `/Ref` array of the note and so the `/Ref` array doesn't necessarily reflect the order of the marks in the document. It would probably be possible to change this, but it is not clear if it actually matters and so it worth the additional coding and processing.

### 3.3.2 `\footnotemark` after `\footnotetext`

The automatic detection doesn't work if a `\footnotemark` is issued after the `\footnotetext` it refers to. There will be no error, but neither the link nor the `/Ref` will connect both.

The simple way to handle this is to use a label and `\footref`:

```
\footnotetext{\label{fn:a}text} ... \footref{fn:a}
```

An alternative would be to extend the syntax of `\footnotemark` and `\footnotetext` to allow to add a label which can then be used. For example

```
\footnotetext[label=fn:a]{text} ... \footnotemark[label=fn:a]
```

As both have already an optional argument, that requires the optional argument extension.

## 3.4 Links

The structure numbers detected for the `/Ref` are also used for links: even if tagging is not activated the tagging commands are defined through the `tagpdf-base` package and the structure commands increase the structure counter and this info can be used.

A `\footnotetext` creates a bunch of destinations (in most cases this sums up to two destinations): one for every structure number in the `/Ref` (used as target by the mark commands) and one for the structure number of the `footnotetext` itself (used as target by `\footrefs` commands).

## 3.5 Implementation details regarding tagging

### 3.6 Handling the mark

The mark in the text is handled by assigning an appropriate plug to the socket `tagsupport/fnmark`. It takes one argument, `\@makefnmark`, the command which formats the mark, and surrounds it by link and tagging commands. At the point where the socket is executed, `\@thefnmark` has already been defined and can be used to setup the reference detections.

### 3.7 Handling the `footnotetext`

The main part is done by assigning a different plug to socket `tagsupport/fntext/begin` and `tagsupport/fntext/end` surrounding the footnote text. These sockets are used to start and end the structure and attempt to detect to which mark the note is related.

The actual typesetting of the note text is done by `\fnote_makefntext:n` (or its L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> name `\@makefntext`). In the new implementation this contains two further kernel sockets for tagging: `tagsupport/fntext/mark` and `tagsupport/fntext/text`. They get plugs assigned that add the tagging commands around note mark and note text.

### 3.8 Footnotes in minipages

In minipages the `\footnote` command uses a special marker (small italic letters by default) and puts the footnote text at the bottom of the box. The `\footnotemark` command uses the standard footnote counter and marker (and so typically creates a superscript number). It is meant to be used with a `\footnotetext` *outside* the minipage to create a footnote mark which refers to a footnote text at the bottom of the page. This means to repeat a footnote marker in a minipage you should use the `\footref` command.

Tagging works quite similar to normal footnotes if the new definition is used and if the minipage code is changed to use the new configuration point. The main problem here is currently the tagging of the minipage itself.

## 4 TODOs

- Special formatting of footnote marks in the text, e.g. if ranges or commas are used require special care as they should normally mark up such text as artifacts and perhaps have to insert empty structures to represent an invisible mark. This must be coordinated with the relevant packages and classes.
- manyfoot doesn't work correctly and must be analyzed.
- memoir is not supported at all and errors when the code tries to patch `\@makefntext`.

*To be documented*

## 5 The Implementation

All this is very rough and misses a lot of documentation.

```
1 \*kernel)
2 \@@=fnote)
```

### 5.1 File declaration

```
3 \ProvidesFile{latex-lab-footnotes.ltx}
4 [\l{labfootnotedate}\space v\l{labfootnoteversion}\space
5 changes to the footnote interfaces]
```

### 5.2 code not fully handled yet

```
6 %
```

```

7 % latex.ltx
8 % not looked at yet
9 % \@mpfootnotetext is probably no longer needed, or only to support other
10 % classes and package. See below about the minipage code.
11 %
12 % \long\def\@mpfootnotetext#1{%
13 %   \global\setbox\@mpfootins\vbox{%
14 %     \unvbox\@mpfootins
15 %     \reset@font\footnotesize
16 %     \hsize\columnwidth
17 %     \@parboxrestore
18 %     \def\@currentcounter{mpfootnote}%
19 %     \protected@edef\@currentlabel
20 %       {\csname p@mpfootnote\endcsname\@thefnmark}%
21 %     \color@begingroup
22 %       \@makefnmark{%
23 %         \rule\z@\footnotesep\ignorespaces#1\@finalstrut\strutbox}%
24 %     \par
25 %     \color@endgroup}}
26 % =====
27 % used by the minipage footnote code.
28 %
29 % \def\@mpfn{footnote}
30 % \def\thempfn{\thefootnote}
31 % =====
32 % this perhaps need some configuration options.
33 %
34 %\def\@makefnmark{\hbox{\@textsuperscript{\normalfont\@thefnmark}}}%
35 %
36 % =====
37 %% alterations not covered:
38 %
39 % ./arabtex/afoot.sty --- too different (and probably too old)
40 %
41 % =====
42 % alterations of footnotetext not covered:
43 %
44 % ./revtex4-1/revtex4-1.cls ./revtex/ltxutil.sty ./revtex/revtex4-2.cls ... (need analysis)
45 % ./bigfoot/bigfoot.sty
46 %
47 % memoir needs checking too
48 %
49 % =====
50 %
51 % use of kerns to mark h-mode positions (unit sp)
52 %
53 % 1 = CJK
54 % 2 = CJK
55 % 3 = multiple footnotes (footmisc, koma, eledmac, tufte, memoir,
56 %   parnotes, sidenotes)
57 % 3 = outer kern in letter spacing (letterspace)
58 % 3 = beginning of list (examdesign.cls)
59 % 4 = CJK pigin
60 % 5 = CJK ruby

```

```

61
62 % 1-4 = polyglossia for korean
63 %

```

---

```

64 \ExplSyntaxOn

```

### 5.3 Temporary variables

```

65 \prop_new:N \l__fnote_tmpa_prop
66 \tl_new:N \l__fnote_tmpa_tl

```

### 5.4 Public variables

A footnote mark will store its structure number (key) and the expanded `\@thefnmark` in this prop so that a following note can retrieve this info if needed. It is possible to use more than one footnote series (type) if needed (if different footnotes/note use the same numbering system). If this command is changed an accompanying property must be created

`\l_fnote_type_tl`

```

67 \tl_new:N \l_fnote_type_tl
68 \tl_set:Nn \l_fnote_type_tl {default}

```

*(End of definition for `\l_fnote_type_tl`. This function is documented on page ??.)*

It must be possible to suppress the hyperlinking, both locally and globally. `hyperref`'s `hyperfootnotes` option should set the boolean.

`\l_fnote_link_bool`

```

69 \bool_new:N \l_fnote_link_bool
70 \bool_set_true:N \l_fnote_link_bool

```

*(End of definition for `\l_fnote_link_bool`. This function is documented on page ??.)*

A hyperlink should have an changeable link type. This can be e.g. used to change the color or the border.

`\l_fnote_link_type_tl`

```

71 \tl_new:N \l_fnote_link_type_tl
72 \tl_set:Nn \l_fnote_link_type_tl {link}

```

*(End of definition for `\l_fnote_link_type_tl`. This function is documented on page ??.)*

### 5.5 Internal variables

`\l__fnote_linktarget_tl` This command stores the name of a linktarget/destination when needed

```

73 \tl_new:N \l__fnote_linktarget_tl

```

*(End of definition for `\l__fnote_linktarget_tl`.)*

`\l__fnote_currentlabel_tl` This command is used to pass a label name around.

```

74 \tl_new:N \l__fnote_currentlabel_tl

```

*(End of definition for `\l__fnote_currentlabel_tl`.)*

`\l__fnote_currentrefs_seq` This sequence stores the list of reference of a note

```

75 \seq_new:N \l__fnote_currentrefs_seq

```

(End of definition for `\l__fnote_currentrefs_seq`.)

The connection between the mark(s) in the text and the note is either deduced automatically or done through an label. The default is automatic, but we must be able to suppress it. For this we use a boolean.

`\l__fnote_autodetect_bool`

```
76 \bool_new:N      \l__fnote_autodetect_bool
77 \bool_set_true:N \l__fnote_autodetect_bool
```

(End of definition for `\l__fnote_autodetect_bool`.)

This is used to pass the structure number of the note around, e.g. to a label inside the note.

```
78 \tl_new:N      \l__fnote_currentstruct_tl
79 \tl_set:Nn     \l__fnote_currentstruct_tl {2}
```

## 5.6 Variants

```
80 \cs_generate_variant:Nn \hook_gput_code:nnn{nne}
81 \cs_generate_variant:Nn \tag_struct_use:n {e}
```

## 5.7 Updating `\@thefnmark`

`\fnote_step_fnmark:nn` This command updates `\@thefnmark`. The first argument is an optional integer expression, the second a counter name. If the optional argument is not given it steps the counter.

```
82 \cs_new_protected:Npn \fnote_step_fnmark:nn #1#2 {
83   \tl_if_novalue:nTF {#1}
84     {
85       \stepcounter {#2}
86       \protected@xdef \@thefnmark { \use:c { the#2 } }
87     }
88     {
89       \group_begin:
```

Note that this is a local assignment even though L<sup>A</sup>T<sub>E</sub>X counters are normally globally changed. This is the way it was in 2e and so far we haven't changed it. The alternative would be to store the current value and restore it after `\@thefnmark` is altered.

```
90       \int_set:cn { c@#2 }{ #1 }
91       \unrestored@protected@xdef \@thefnmark { \use:c { the#2 } }
92     \group_end:
93   }
94 }
```

(End of definition for `\fnote_step_fnmark:nn`. This function is documented on page ??.)

`\fnote_set_fnmark:nn` This is similar to the previous command, but it doesn't step the counter but use the current value.

```
95 \cs_new_protected:Npn \fnote_set_fnmark:nn #1#2 {
96   \tl_if_novalue:nTF {#1}
97     {
98       \protected@xdef \@thefnmark { \use:c { the#2 } }
99     }
100    {
101    \group_begin:
```

```

102         \int_set:cn { c@#2 }{ #1 }
103         \unrestored@protected@xdef \@thefnmark { \use:c { the#2 } }
104     \group_end:
105 }
106 }

```

(End of definition for `\fnote_set_fnmark:nn`. This function is documented on page ??.)

## 5.8 Hooks

`fnmark/before` (*hook*) Hooks in the `footnotemark` command.

```

fnmark/after (hook) 107 \NewMirroredHookPair{fnmark/before}{fnmark/after}
  fnmark (hook)    108 \NewHook{fnmark}
fnmark/begin (hook) 109 \NewHook{fnmark/begin}
  fnmark/end (hook) 110 \NewHook{fnmark/end}

```

`fnntext/before` (*hook*) Hooks in the `footnotetext` command:

```

fnntext/after (hook) 111 \NewMirroredHookPair{fnntext/before}{fnntext/after}
  fnntext (hook)    112 \NewHook{fnntext}
fnntext/begin (hook) 113 \NewHook{fnntext/para}
  fnntext/end (hook) 114 \NewHook{fnntext/begin}
  fnntext/para (hook) 115 \NewHook{fnntext/end}

```

## 5.9 Debugging code

The debugging code is just temporary

```

116 \bool_new:N          \g_fnote_debug_bool

\DebugFNotesOn
\DebugFNotesOff 117 \cs_new_protected:Npn \DebugFNotesOn { \bool_gset_true:N \g_fnote_debug_bool }
118 \cs_new_protected:Npn \DebugFNotesOff { \bool_gset_false:N \g_fnote_debug_bool }

```

(End of definition for `\DebugFNotesOn` and `\DebugFNotesOff`. These functions are documented on page ??.)

We log the hooks in the footnote mark command, but only once

```

119 \cs_new_protected:Npn \__fnote_debug_footnotemark:
120 {
121     \bool_if:NT \g_fnote_debug_bool
122     {
123         \hook_log:n {fnmark/before}
124         \hook_log:n {fnmark}
125         \hook_log:n {fnmark/begin}
126         \hook_log:n {fnmark/end}
127         \hook_log:n {fnmark/after}
128         \cs_gset_eq:NN \__fnote_debug_footnotemark: \prg_do_nothing:
129     }
130 }

```

Similar for the `footnotetext`

```

131 \cs_new_protected:Npn \__fnote_debug_footnotetext:
132 {
133     \bool_if:NT \g_fnote_debug_bool
134     {
135         \socket_log:n {fnntext/process}

```

```

136     \socket_log:n {fntext/make}
137     \socket_log:n {fntext/begin}
138     \socket_log:n {fntext/end}

139     \socket_log:n {fntext/mark}
140     \socket_log:n {fntext/text}

141     \socket_log:n {tagsupport/fnmark}
142     \socket_log:n {tagsupport/fntext/begin}
143     \socket_log:n {tagsupport/fntext/end}
144     \socket_log:n {tagsupport/fntext/mark}
145     \socket_log:n {tagsupport/fntext/text}

146     \hook_log:n {fntext/before}
147     \hook_log:n {fntext}
148     \hook_log:n {fntext/para}
149     \hook_log:n {fntext/begin}
150     \hook_log:n {fntext/end}
151     \hook_log:n {fntext/after}

```

Show the info only once (if at all).

```

152     \cs_gset_eq:NN \_fnote_debug_footnotetext: \prg_do_nothing:
153     }
154 }

```

## 5.10 The new \@footnotemark command

`\fnote_footnotemark:` This is the main command which will replace `\@footnotemark`.

```

155 \cs_new_protected:Npn \fnote_footnotemark: {
156   \_fnote_debug_footnotemark:
157   %-----
158   % bibarts
159   % chextras --- actually in the wrong place does an \unskip
160   \hook_use:n {fnmark/before}
161   %-----
162   \leavevmode
163   \ifhmode
164     \edef\@x@sf{\the\spacefactor}
165   %-----
166   % bxjsja-minimal.def --- what they do could be done at ‘‘bibarts’’
167   % (a bit less efficient)
168   % memhfixc.sty
169   % footmisc.sty
170   \hook_use:n {fnmark}
171   %-----
172   \nobreak
173   \fi
174   %-----
175   % hyperref.sty
176   \hook_use:n {fnmark/begin}
177   %-----

```

The kernel socket for tagging. It picks up `\@makefnmark` as its argument and if tagging is not active it contains the identity plug.

```

178   \socket_use:nn {tagsupport/fnmark} \@makefnmark
179   %-----

```



If a footnote mark is placed by its own then it should finish by executing the hook `fnmark/end`, resetting the space factor, and finishing with the hook `fnmark/after`. However, in a complete footnote these actions have to happen only after we have handled the footnote text (e.g., by placing it into an `\insert`). In such a situation `\__fnote_footmark_finish:` below does nothing and the action is carried out later.

```
180 \__fnote_footnotemark_finish:
181 }
```

*(End of definition for \fnote\_footnotemark:. This function is documented on page ??.)*

`\__fnote_footnotemark_default_finish:` The default definition for `\__fnote_footnotemark_finish:` is called `\__fnote_footnotemark_default_finish:`

```
182 \cs_new_protected:Npn \__fnote_footnotemark_default_finish: {
183 % hyperref.sty
184 % memhfixc.sty --- could move fnmark/after
185 % scrلتtr2.cls --- could vanish if footmisc uses a hook
186 % footmisc.sty
187 \UseHook{fnmark/end}
188 %-----
189 \ifhmode
190 \spacefactor \@x@sf \relax
191 \fi
192 %
193 %-----
194 \UseHook{fnmark/after}
195 %-----
196 }
197 \cs_new_eq:NN \__fnote_footnotemark_finish: \__fnote_footnotemark_default_finish:
```

*(End of definition for \\_\_fnote\_footnotemark\_default\_finish: and \\_\_fnote\_footnotemark\_finish:.)*

`tagsupport/fnmark (socket)` Not a public socket but reserved for tagging. By default it contains `identity` and is reassigned if tagging is active.

```
198 \NewSocket{tagsupport/fnmark}{1}
```

`\@footnotemark` Here we provide the traditional L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> name in case it is directly used in some legacy class.

```
199 \cs_set_eq:NN \@footnotemark \fnote_footnotemark:
```

*(End of definition for \@footnotemark. This function is documented on page ??.)*

## 5.11 The new `\@footnotetext` command

`\fnote_footnotetext:n` We temporarily test for the tagging socket until it is in the next release:

```
200 \socket_if_exist:nF {tagsupport/para/restore}
201 {
202 \NewSocket{tagsupport/para/restore}{0}
203 \NewSocketPlug{tagsupport/para/restore}{default}
204 {
205 \tl_set:Nn \l__tag_para_main_tag_tl {text-unit}
206 \tl_set_eq:NN \l__tag_para_tag_tl \l__tag_para_tag_default_tl
207 \bool_set_false:N \l__tag_para_flattened_bool
208 }
```

```

209     \AssignSocketPlug{tagsupport/para/restore}{default}
210   }
211   \cs_new_protected:Npn \fnote_footnotetext:n #1 {
212     \__fnote_debug_footnotetext:
213     %-----
214     % ./linguex/linguex.sty
215     \hook_use:n {fntext/before}
216     %-----

```

Execute a kernel socket for tagging.

```

217   \socket_use:n {tagsupport/fntext/begin}
218   \socket_use:mn {fntext/process}
219   {
220   %-----
221   % resetting baselinestretch ... (could be done further down)
222   % ./uafthesis/uafthesis.cls
223   % ./setspace/setspace.sty
224   % ./footmisc/footmisc.sty (normal)
225     \hook_use:n {fntext}
226   %-----
227     \reset@font
228     \footnotesize
229   %-----
230   % some classes use a different font size, e.g.,
231   % ./nrc/nrc1.cls ./nrc/nrc2.cls
232   % but those could be done in fntext/para instead
233   %-----

```

In case of sidenotes the next settings are pointless, but as they do not hurt (except for the `\hsize` setting) and are needed for all other cases we make them here and overwrite them for side notes

```

234     \interlinepenalty\interfootnotelinepenalty
235     \splittopskip\footnotesep
236     \splitmaxdepth \dp\strutbox
237     \floatingpenalty \@MM
238     \hsize\columnwidth
239     \@parboxrestore
240     \UseTaggingSocket{para/restore}
241     \parindent 1em % typical default used in \@makefntext moved up here
242     \def\@currentcounter{footnote}
243     \protected@edef \@currentlabel { \p@footnote \@thefnmark }
244   %-----
245   % for altering para parameters ...
246   % code for resphilosophica came earlier but it could go here.
247   % Has the advantage that one can also overwrite \cs{@currentcounter}
248   % and \cs{@currentlabel} is that is necessary.
249   %
250   % ./resphilosophica/resphilosophica.cls
251     \hook_use:n {fntext/para}
252   %-----
253     \color@begingroup
254   %-----
255   % fnpara wants to replace \@makefntext{...} and para and side

```

```

256 % option of footmisc etc too ...
257 % so we make this a socket, because only one action can be active:
258 %-----
259     \socket_use:nn {fntext/make}
260     {
261 %-----
262 % ./resphilosophica/resphilosophica.cls
263 %-----
264     \socket_use:n {fntext/begin}%
265 %-----
266 % bibarts
267 % fnbreak.sty
268     \hook_use:n {fntext/begin}
269 %-----
270     \ignorespaces
271     #1
272 %-----
273 % bibarts
274 % fnbreak.sty
275     \hook_use:n {fntext/end}
276 %-----

```

The socket code (by default adding a strut) has to come *after* everything added into the hook above.

```

277     \socket_use:n {fntext/end}
278     }
279     \par
280     \color@endgroup
281     }
282 %-----

```

The corresponding kernel hook that ends the tagging structure if tagging is active.

```

283     \socket_use:n{tagsupport/fntext/end}
284 %-----
285 % ./linguex/linguex.sty
286     \hook_use:n {fntext/after}
287 %-----
288     }

```

(End of definition for `\fnote_footnotetext:n`. This function is documented on page ??.)

`fntext/process` (*socket*)

```

289 \NewSocket      {fntext/process}{1}
290 \NewSocketPlug{fntext/process}{default}{ \insert\footins {#1} }
291 \NewSocketPlug{fntext/process}{side}   { \marginpar {#1} }
292 \AssignSocketPlug{fntext/process}{default}

```

`fntext/make` (*socket*) This socket receives the  $\langle text \rangle$  from the `\footnote` or `\footnotetext` and formats it.

```

293 \NewSocket      {fntext/make}{1}
294 \NewSocketPlug{fntext/make}{default}{ \@makefntext {#1} }

```

When running several footnotes together as a paragraph some additional work is necessary to unbox the individual footnotes recursively (see T<sub>E</sub>Xbook algorithm in appendix D).

```

295 \NewSocketPlug{fntext/make}{para}

```

```

296 {
297   \setbox\FN@tempboxa\hbox{\@makefntext{#1}}%
298   \dp\FN@tempboxa\z@
299   \ht\FN@tempboxa
300   \dimexpr\wd\FN@tempboxa *%
301           \footnotelineskip /\columnwidth\relax
302   \box\FN@tempboxa
303 }
304 \AssignSocketPlug{fntext/make}{default}

```

`fntext/begin (socket)` By default adds a strut at the start of the footnote text.

```

305 \NewSocket {fntext/begin}{0}
306 \NewSocketPlug{fntext/begin}{default}{ \rule\z@\footnotesep }
307 \AssignSocketPlug{fntext/begin}{default}

```

`fntext/end (socket)` By default adds a strut at the end of the footnote text unless we are no longer in hmode.

```

308 \NewSocket {fntext/end}{0}
309 \NewSocketPlug{fntext/end}{default}{ \@finalstrut\strutbox }

```

When running several footnotes together as a paragraph some additional glue has to be added between them.

```

310 \NewSocketPlug{fntext/end}{para}
311 {%
312     \strut
313     \penalty-10\relax
314     \hskip\footglue
315 }
316 \AssignSocketPlug{fntext/end}{default}

```

`tagsupport/fntext/begin (socket)` Kernel sockets for tagging.

```

tagsupport/fntext/end (socket) 317 \NewSocket{tagsupport/fntext/begin}{0}
318 \NewSocket{tagsupport/fntext/end}{0}

```

Provide the name  $\text{\LaTeX} 2_{\epsilon}$  is used to and do this unconditionally (no patching of class code if any). This means that if a class provides it own definition that gets lost and if necessary needs to be handled with firstaid (or updating of the class).

```

319 \AddToHook{begindocument}
320 {
321   \cs_set_eq:NN \@footnotetext \fnote_footnotetext:n
322 }

```

## 5.12 The new `\@makefntext` command

`\footnotemargin` is the logic implemented by footmisc. Perhaps we don't want to do this like that in the kernel but for now I have used this interface unchanged.

```

323 \newdimen\footnotemargin
324 \footnotemargin\maxdimen % no value given
325
326 \AtBeginDocument
327 {
328   \ifdim \footnotemargin=\maxdimen
329     \setlength\footnotemargin{1.8em}
330   \fi
331 }

```

`\fnote_makefntext:n`

```
332 \cs_new_protected:Npn \fnote_makefntext:n #1 {
Some classes in their redefinition for \@makefntext have placed some paragraph parameters at this point, but those can equally well go into the hook fntext/para. We therefore do not provide a further hook at this point.
333 \noindent
334 \socket_use:nn {tagssupport/fntext/mark} { \socket_use:n {fntext/mark} }
335 \socket_use:nn {tagssupport/fntext/text} { \socket_use:nn {fntext/text}{#1} }
336 }
```

*(End of definition for `\fnote_makefntext:n`. This function is documented on page ??.)*

`fntext/mark` (*socket*) A socket to typeset the mark at the start of a footnote.

```
337 \NewSocket {fntext/mark}{0}
```

The default plug implements the logic introduced with the `footmisc` package.

```
338 \NewSocketPlug{fntext/mark}{default}{
339 \ifdim\footnotemargin>\z@
340 \hb@xt@ \footnotemargin{\hss\@makefnmark}
341 \else
342 \ifdim\footnotemargin=\z@
343 \llap{\@makefnmark}
344 \else
345 \ifdim\footnotemargin=-\maxdimen
346 \@makefnmark
347 \else
348 \llap{\hb@xt@ -\footnotemargin{\@makefnmark\hss}}
349 \fi
350 \fi
351 \fi
352 }
353 \AssignSocketPlug{fntext/mark}{default}
```

`fntext/text` (*socket*) By default this socket does nothing special and simply processes its argument as provided.

```
354 \NewSocket {fntext/text}{1}
```

`tagssupport/fntext/mark` (*socket*) Not a public socket but reserved for tagging. By default it contains `identity` and is `tagssupport/fntext/text` (*socket*) reassigned if tagging is active.

```
355 \NewSocket{tagssupport/fntext/mark}{1}
356 \NewSocket{tagssupport/fntext/text}{1}
```

### 5.12.1 Making documents use the new `\@makefntext`

If the definition for `\@makefntext` is that of the standard classes then replace it with `\fnote_makefntext:n`, otherwise try to patch the definition used in the class.

Here is the definition the way it is in `classes.dtx`. Notice that (for saving space) there is no space after `em` to terminate the assignment. We need to mimic that, otherwise a test would return false even if the definition has not been modified.

`\old@std@class@makefntext`

```
357 \newcommand\old@std@class@makefntext[1]{%
358   \parindent 1em%
359   \noindent
360   \hb@xt@1.8em{\hss\@makefnmark}#1}
```

(End of definition for `\old@std@class@makefntext`. This function is documented on page ??.)

Here is the messy code for patching. Note that this is only there to help classes along that aren't updated yet so it does some minimal patching to hopefully add kernel configuration hooks in the right place while otherwise leaving the legacy code alone. An updated class would not redefine `\@makefntext` but simply add appropriate code to the provided hooks.

What it does is roughly the following: It looks for a definition of `\@makefntext` of the form

```
{AAA \hbox BBB { CCC } DDD #1 EEE }
```

where "BBB" is something like `to 1em` or similar. It then replaces that with

```
{AAA \UseSocket{tagsupport/fntext/mark}{\hbox BBB { CCC }} DDD
  \UseSocket{tagsupport/fntext/text}{#1} EEE }
```

The patching is not very careful, i.e., it assumes there is only one `#1` in the replacement text and that the first `\hbox` found is the right one to patch. But that is enough to cater for all definitions of `\@makefntext` out there in the TL distribution.

If `\hbox` is not found it tries the same looking for `\hb@xt@` which is what some classes use and if that is not found either it assume that this is a version that uses `\@makefnmark` without surrounding it in a box and if that fails it gives up with an `\ERROR` (which needs to get a proper definition).

```
361 \tl_new:N \l__fnote_patch_tl
362 \cs_new_eq:NN \__fnote_tmp:w \ERROR
363
364 \cs_new_protected:Npn \__fnote_patch:
365   {
366     \tl_set:Nn \l__fnote_patch_tl { \@makefntext { \UseSocket{tagsupport/fntext/text}{##1} } }
367     \tl_if_in:NnTF \l__fnote_patch_tl { \hbox }
368       { \cs_set_eq:NN \__fnote_tmp:w \__fnote_patch_hbox:w }
369       {
370         \tl_if_in:NnTF \l__fnote_patch_tl { \hb@xt@ }
371           { \cs_set_eq:NN \__fnote_tmp:w \__fnote_patch_hb@xt:w }
372           {
```

Some styles/classes use `\makebox[...][...]` instead of `\hb@xt@` so try to patch those too.

```
373         \tl_if_in:NnTF \l__fnote_patch_tl { \makebox }
374           { \cs_set_eq:NN \__fnote_tmp:w \__fnote_patch_makebox:w }
375           {
376             \tl_if_in:NnTF \l__fnote_patch_tl { \@makefnmark }
377               { \cs_set_eq:NN \__fnote_tmp:w \__fnote_patch_@makefnmark:w }
378               { \ERROR
379                 \cs_set_eq:NN \__fnote_tmp:w \exp_stop_f: }
380             }
381         }
382     }
```

```

383 \tl_set:Nf \l__fnote_patch_tl
384 { \exp_after:wN \__fnote_tmp:w \l__fnote_patch_tl }
385 \cs_set:Npn \__fnote_tmp:w { \long \def \@makefntext ###1 }
386 \exp_after:wN \__fnote_tmp:w \exp_after:wN { \l__fnote_patch_tl }
387 }

```

If \@makefntext contains \hbox then grab “AAA” as #1 and “BBB” (up to the open {) and return it as

```
AAA \@makefntext@processX { \hbox BBB }
```

```

388 \cs_new:Npn \__fnote_patch_hbox:w #1 \hbox #2 #
389 { \exp_stop_f: #1 \@makefntext@processX { \hbox #2 } }

```

Same for the other cases.

```

390 \cs_new:Npn \__fnote_patch_hb@xt@:w #1 \hb@xt@ #2 #
391 { \exp_stop_f: #1 \@makefntext@processX { \hb@xt@ #2 } }
392 \cs_new:Npn \__fnote_patch_makebox:w #1 \makebox #2 #
393 { \exp_stop_f: #1 \@makefntext@processX { \makebox #2 } }

```

If the definition contains neither \hbox, \hb@xt@ nor \makebox, we see if it contains \@makefnmark and if so put the socket before that.

```

394 \cs_new:Npn \__fnote_patch_@makefnmark:w #1 \@makefnmark
395 { \exp_stop_f: #1 \@makefntext@processX { \use:n } { \@makefnmark } }

```

The code provided by Bruno above expects 2 arguments but we need a different structure so this is a simple reshuffling. Would be better if we can patch the right structure in directly, but I’m not a patch person, so this is the simple way out for now:

```
396 \cs_new:Npn \@makefntext@processX #1#2{\UseSocket{tagsupport/fntext/mark}{#1{#2}}}
```

At \begin{document} check if the current definition is that of the standard classes and if so replace it by \fnote\_makefntext:n otherwise try and patch the definition made by some class or package using the approach above.

```

397
398 \AddToHook{begindocument}
399 {
400   \cs_if_eq:NNTF \@makefntext \old@std@class@makefntext
401   {
402     \cs_set_eq:NN \@makefntext \fnote_makefntext:n
403   }
404   {

```

If \@makefntext contains the definition from footmisc we do nothing, otherwise we try to patch.

```

405     \cs_if_eq:NNTF \@makefntext \footmisc@chang@makefntext
406     { \__fnote_patch: }
407   }
408 }

```

```

409
410
411 % possibly add the following to check for multiple \hbox in
412 % the definition:
413 %
414 % \seq_set_split:NnV \l__fnote_patch_seq { \hbox } \l__fnote_patch_tl
415 % \int_compare:nT { \seq_count:N \l__fnote_patch_seq } > 2 \ERROR
416 %

```

## 5.13 Document-level commands

`\footnotetext`

```
417 \DeclareDocumentCommand\footnotetext {o+m}
418 {
419   \fnote_set_fnmark:nn {#1} \@mpfn
420   \@footnotetext {#2}
421 }
```

*(End of definition for \footnotetext. This function is documented on page ??.)*

`\footnote`

```
422 \DeclareDocumentCommand\footnote {o+m}
423 {
424   \fnote_step_fnmark:nn {#1} \@mpfn
425   \cs_set_eq:NN \__fnote_footnotemark_finish: \prg_do_nothing:
426   \@footnotemark
427   \cs_set_eq:NN \__fnote_footnotemark_finish: \__fnote_footnotemark_default_finish:
428   \@footnotetext {#2}
429   \__fnote_footnotemark_finish:
430 }
```

*(End of definition for \footnote. This function is documented on page ??.)*

`\footnotemark`

```
431 \DeclareDocumentCommand\footnotemark {o}
432 {
433   \fnote_step_fnmark:nn {#1} { footnote }
434   \@footnotemark
435 }
```

*(End of definition for \footnotemark. This function is documented on page ??.)*

`\footref` `\footref` used the starred `\ref` in `\@thefnmark` as the linking is handled by the tagging code inside the `\@footnotemark`. `\footref` should not try to link to its related note automatically but should instead use the label. This is passed to `\@footnotemark` through `\l__fnote_currentlabel_tl`.

```
436 \DeclareDocumentCommand\footref {m}
437 {
438   \begingroup
439     \unrestored@protected@xdef\@thefnmark{\ref*{#1}}%
440   \endgroup
441   \bool_set_false:N \l__fnote_autodetect_bool
442   \tl_set:Nn \l__fnote_currentlabel_tl {#1}
443   \@footnotemark
444   \bool_set_true:N \l__fnote_autodetect_bool
445 }
```

*(End of definition for \footref. This function is documented on page ??.)*



## 5.14 Firstaid for packages and classes

### 5.15 Kernel patches

Tagging of footnotes in minipages require a change in the minipage commands We define at first a local configuration command for minipage footnotes.

```
446 \NewSocketPlug{fntext/process}{mp}  
447 {  
448   \global\setbox\@mpfootins\vbox{%  
449     \unvbox\@mpfootins  
450     #1  
451   }  
452 }
```

#### 5.15.1 memoir

The memoir class redefines various internal commands to inject its hooks and additional code. The following reinstates the kernel command and so probably breaks various options of memoir, but without the changes it errors anyway. The footmisc package should be used to change for example to para footnotes.

```
453 \AddToHook{class/memoir/before}  
454 { \let\new@std@class@makecol\@makecol }  
455 \AddToHook{class/memoir/after}  
456 {  
457   \cs_set_eq:NN \@footnotemark \fnote_footnotemark:  
458   \cs_set_eq:NN \@makefntext\old@std@class@makefntext  
459   \cs_set_eq:NN \@makecol\new@std@class@makecol  
460 }
```

#### 5.15.2 setspace

It should not overwrite it any longer but use a hook, so for now we do just that here.

```
461 \AddToHook{package/setspace/after}  
462 {\let \@footnotetext \fnote_footnotetext:n  
463   \AddToHook{fntext}[setspace]{\let\baselinestretch\setspace@singlespace}}
```

#### 5.15.3 hyperref

hyperref has a hook which allows to disable its footnote related patches. As we will handle links directly in the code this is used.

```
464 \def\hyper@nopatch@footnote{}
```

We use the hyperref commands for now for links. To avoid to have to test for hyperref we provide dummies. TODO consider to use specials to get similar spacing.

```
465 \AtBeginDocument  
466 {  
467   \providecommand\hyper@linkstart{\@gobbletwo}  
468   \providecommand\hyper@linkend{\@empty}  
469 }
```

It must be possible to suppress the hyperlinking, both locally and globally. `hyperref` should set the boolean `\l_fnote_link_bool`. For now we test for the `hyperref` boolean (so it can be suppressed only globally).

```

470 \AtBeginDocument
471 {
472   \ifpackageloaded{hyperref}
473   {
474     \legacy_if:nF{Hy@hyperfootnotes}{\bool_set_false:N \l_fnote_link_bool}
475   }
476   {
477     \bool_set_false:N \l_fnote_link_bool
478   }
479 }

```

## 5.16 Tagging and hyperlink code

### 5.16.1 Rolemap for structure tags

We use role-mapping to get more speaking names in the PDF and so ease debugging. These names are already provided by `tagpdf` directly.

### 5.16.2 Extending the label system

For `\footref` and (perhaps later for labeled footnotes) we must extend the label system. Beside the normal values we also need the structure number of the note. We use the inbuilt label hook `At` first we define a suitable attribute, it uses as value the structure number of the note as stored in `\l__fnote_currentstruct_tl`

```

480 \property_new:nnnn {fnote/struct}{now}{1}{\l__fnote_currentstruct_tl}

```

We add a hook to the label hook. By default it does nothing

```
\__fnote_label_hook:e
```

```

481 \cs_new_protected:Npn \__fnote_label_hook:e #1 {}
482 \AddToHookWithArguments{label}{ \__fnote_label_hook:e{#1}}

```

(End of definition for `\__fnote_label_hook:e`.)

Inside a `footnotetext` we change the hook to store the structure number too. The name of label is provided as argument in the label hook.

```

483 \AddToHook{fntext/begin}
484 {
485   \cs_set_protected:Npn \__fnote_label_hook:e #1
486   {
487     \property_record:ee {\__fnote/#1} {fnote/struct}
488   }
489 }

```

### 5.16.3 Storing and retrieving reference data

To establish the connection between a mark and a note the mark has to store its representation, and the note has to analyse the stored representations to get the structure numbers of its mark. This is done with the public function to allow similar systems (e.g. tabular notes, other footnote series) to make use of this.

`\fnote_class_new:nn` This sets up a new footnote type, the first argument is the name, the second is meant for options. Currently it does nothing at all. It is not necessary to setup every footnote like command as its own type!

```

490 \cs_new_protected:Npn \fnote_class_new:nn #1 #2 % #1 name, #2 options
491 {
492   \prop_new:c { g__fnote_currentmarks_ #1 _prop }
493 }
494
495 \fnote_class_new:nn {default}{}

```

*(End of definition for \fnote\_class\_new:nn. This function is documented on page 9.)*

`\fnote_mark_gput:nn` This commands takes as argument the representation of the mark, e.g., `\@thefnmark` and the type (typically default should work).

```

496 \cs_new_protected:Npn \fnote_mark_gput:nn #1 #2 % #1 the representation of the mark, #2 type
497 {
498   \prop_gput:cen { g__fnote_currentmarks_ #2 _prop }
499     { \tag_get:n{struct_num} }
500     { #1 }
501 }
502
503 \cs_generate_variant:Nn \fnote_mark_gput:nn {no,oo}

```

*(End of definition for \fnote\_mark\_gput:nn. This function is documented on page 9.)*

`\fnote_mark_gpop_all:nnN` This commands takes as argument the representation of the mark (e.g. the content of `\@thefnmark`), the class (typically default should work) and a sequence into which every structure number in the property is stored that has the same value as the mark. The sequence is cleared first.

```

504 \cs_new_protected:Npn \fnote_mark_gpop_all:nnN #1 #2 #3
505 {
506   \seq_clear:N #3
507   \prop_set_eq:Nc \l__fnote_tmpa_prop { g__fnote_currentmarks_ #2 _prop }
508   \prop_map_inline:Nn \l__fnote_tmpa_prop
509     {
510       \tl_if_eq:nnT {#1} { ##2 }
511       {

```

store the key (the structure number) in the seq

```

512         \seq_put_right:Nn #3 { ##1 }

```

remove entry as used from the global prop

```

513         \prop_gremove:cn { g__fnote_currentmarks_ #2 _prop } {##1}
514       }
515     }
516   }
517 \cs_generate_variant:Nn \fnote_mark_gpop_all:nnN {ooN}

```

*(End of definition for \fnote\_mark\_gpop\_all:nnN. This function is documented on page 9.)*

#### 5.16.4 Enabling tagging and links for the mark command

**FEMark** (*plug*) To handle the mark in the text, we define a special plug for the socket `tagsupport/fnmark` that receives `\@makefntext` as its argument. At this time `\@thefnmark` is already set.

```
518 \NewSocketPlug{tagsupport/fnmark}{FEMark}
519 {
```

End an open mc and start the structure.

```
520   \tag_mc_end_push:
521   \tag_struct_begin:n { tag=footnotemark }
```

The associated note is either auto detected or given by the user.

```
522   \bool_if:NTF \l__fnote_autodetect_bool
523   {
```

For the auto detecting we store the structure number and `\@thefnmark` inside a prop and set the target name of the link to the current structure number. TODO: this should be usable for other footnote types which means the name of the prop shouldn't be fix.

```
524     \fnote_mark_gput:oo {\@thefnmark}{\l_fnote_type_tl}
525     \tl_set:Nc \l__fnote_linktarget_tl {footnote*.\tag_get:n{struct_num}}
526   }
```

If there is no autodetecting we need some id, currently it is called `\l__fnote_currentlabel_tl`. the Ref is set by looking at the label value. We must also add the current structure number to the ???f the FEnote. Both must be delayed as we don't know if the objects of the FEnote and the mark have already been created.

```
527   {
528     \hook_gput_code:nne {tagpdf/finish/before} {tagpdf/footnote}
529     {
530       \exp_not:N\fnote_gput_refs:ee
531       { \tag_get:n{struct_num} }
532       { \property_ref:ee{ __fnote/\l__fnote_currentlabel_tl } {fnote/struct} }
533     }
```

in this case we set the name of the linktarget in the note to the structure number of the text mark.

```
534     \tl_set:Nc \l__fnote_linktarget_tl {footnote*.\property_ref:ee {__fnote/\l__fnote_cu
535   }
```

And now the actual content

```
536   \tag_mc_begin:n{tag=Lbl}
537   %
538   \bool_if:NTF \l_fnote_link_bool
539   {
540     \exp_args:No
541     \hyper@linkstart
542     { \l_fnote_link_type_tl }
543     { \l__fnote_linktarget_tl }
544     #1
545     \hyper@linkend
546   }
547   { #1 }
548   \tag_mc_end:
549   \tag_struct_end:
550   \tag_mc_begin_pop:n{ }
551 }
```

At last assign the plug:

```
552 \AssignSocketPlug{tagssupport/fnmark}{FEMark}
```

### 5.16.5 The footnote text

We need a public command to append values to the Ref keys

```
553 \cs_new_protected:Npn \__fnote_gput_ref:nn #1 #2 % #1 the structure number receiving the ref
554 {
555   \tag_struct_gput:nnn {#1}{ref_num}{#2}
556 }
557 \cs_new_protected:Npn \fnote_gput_refs:nn #1 #2 % pair of numbers
558 {
559   \__fnote_gput_ref:nn {#1}{#2}
560   \__fnote_gput_ref:nn {#2}{#1}
561 }
562 \cs_generate_variant:Nn \fnote_gput_refs:nn {ee}
```

*(End of definition for \\_\_fnote\_gput\_ref:nn and \fnote\_gput\_refs:nn. This function is documented on page ??.)*

kernel hooks for tagging this sets the structure around the whole text

**FENote** (*plug*)

```
563 \NewSocketPlug{tagssupport/fntext/begin}{FENote}
564 {
565   \tag_mc_end_push:
```

test if a footnote is allowed, if not move up to the next sect or the document structure.

```
566   \tag_check_child:nnTF {FENote}{pdf2}
567   {
568     \tag_struct_begin:n { tag=footnote }
569   }
570   {
571     \tag_struct_begin:n
572     {
573       tag=footnote,
```

We add 0 for now to ensure to get a number even if the sec code is not loaded or if the seq is empty (which it shouldn't unless there is an coding error)

```
574       parent=\int_max:nn{2}{\tag_get:n{current_Sect}+0}
575     }
576   }
```

Store the current structure number for labels.

```
577   \tl_set:Ne \l__fnote_currentstruct_tl { \tag_get:n{struct_num} }
```

after we have opened the structure we can use the structure number to try to detect the connected marks. As with the marks we assume that sometimes no auto detection is done.

```
578   \bool_if:NTF \l__fnote_autodetect_bool
579   {
```

find open marks with identical \@thefnmark

```
580     \fnote_mark_gpop_all:ooN { \@thefnmark }{ \l_fnote_type_tl } \l__fnote_currentrefs_s
```

Then we store the object numbers of the marks in the /Ref of the FENote structure: and the number of the FENote into the marks structure:

```

581     \seq_map_inline:Nn \l__fnote_currentrefs_seq
582     {
583         \fnote_gput_refs:ee {##1}{ \l__fnote_currentstruct_tl }
584     }
585 }
```

If no auto detection is done

```

586     {%no auto
587
588     }
589 }
```

This finish the setup of the tagging structure.

Now we process the text. The destinations for the links are set with the label so that we can be sure that we are in hmode.

```

590 \NewSocketPlug{taggsupport/fntext/end}{FENote}
591 {
592     \tag_struct_end:
593     \tag_mc_begin_pop:n{ }
594 }
```

At last assign the plugs:

```

595 \AssignSocketPlug{taggsupport/fntext/begin}{FENote}
596 \AssignSocketPlug{taggsupport/fntext/end}{FENote}
```

The kernel socket taggsupport/fntext/mark is responsible for tagging the mark in the note. We use it to surround the mark with the needed tagging commands.

TODO check if additional kernel configuration points are needed. If yes, what about the paragraph start and the paratagging??

**FENoteLbl** (*plug*) This plug creates the label in the note on the bottom. It also adds link targets for the hyperlinking.

```

597 \NewSocketPlug{taggsupport/fntext/mark}{FENoteLbl}
598 {
599     \tag_mc_end_push:
```

We create a link target for every related mark. The name is footnote\* (*structure number of the mark*). We also add a link target for the current structure (for \footref).

```

600     %\seq_show:N\l__fnote_currentrefs_seq
601     \seq_map_inline:Nn\l__fnote_currentrefs_seq {\MakeLinkTarget*{footnote*.*##1}}
602     \MakeLinkTarget*{footnote*.*\l__fnote_currentstruct_tl}
```

Now we add the tagging commands. We move the structure of the label to the begin of the footnote structure.

```

603     \tag_struct_begin:n { tag=footnotelabel,parent=\l__fnote_currentstruct_tl,firstkid }
604     \tag_mc_begin:n { tag=Lbl }
605     #1
606     \tag_mc_end:
607     \tag_struct_end:
608     \tag_mc_begin_pop:n{ }
609 }
```

```
610 \AssignSocketPlug{tagsupport/fntext/mark}{FENoteLb1}
```

FENotetext (*plug*) This plug is for the kernel socket `tagsupport/fntext/text` around the actual note text when doing tagging. Currently it only adds an MC chunk.

TODO Should it set a mc or could it rely on the content?

```
611 \NewSocketPlug{tagsupport/fntext/text}{FENotetext}
612 {
613   \tag_mc_end_push:
614   \tag_mc_begin:n{ }
615   #1
616   \tag_mc_end:
617   \tag_mc_begin_pop:n{ }
618 }
619 \AssignSocketPlug{tagsupport/fntext/text}{FENotetext}
```

---

```
620 \ExplSyntaxOff
621 </kernel>
622 <@@=>
```

## 6 Reimplementing the `footmisc` package

```
623 <*footmisc>
624 %%
625 %% Copyright (c) 1995-2011 Robin Fairbairns
626 %% Copyright (c) 2018-2023 Robin Fairbairns, Frank Mittelbach
627 %%
628 %% This file is part of the 'latex-lab Bundle'.
629 %% -----
630 %%
631 %% It may be distributed and/or modified under the
632 %% conditions of the LaTeX Project Public License, either version 1.3c
633 %% of this license or (at your option) any later version.
634 %% The latest version of this license is in
635 %%   https://www.latex-project.org/lppl.txt
636 %% and version 1.3c or later is part of all distributions of LaTeX
637 %% version 2008 or later.
638 %%
639 %% This work has the LPPL maintenance status 'maintained'.
640 %%
641 \NeedsTeXFormat{LaTeX2e}
642 \providecommand\DeclareRelease[3]{ }
643 \providecommand\DeclareCurrentRelease[2]{ }
644
645 \DeclareRelease{v5}{2011-06-06}{footmisc-2011-06-06.sty}
646 \DeclareCurrentRelease{}{2022-02-14}
647 \ProvidesPackage{latex-lab-footmisc}%
648   [2025/02/20 v6.0e
649     a miscellany of footnote facilities -- latex-lab version%
650     ]
651
652 \NeedsTeXFormat{LaTeX2e}[2020/10/01]
```

```

653 \newtoks\FN@temptoken
654 \providecommand\protected@writeaux{%
655   \protected@write\@auxout
656 }
657 \def\l@advance@macro{\@dvance@macro\edef}
658 \def\@dvance@macro#1#2#3{\expandafter\@tempcnta#2\relax
659   \advance\@tempcnta#3\relax
660   #1#2{\the\@tempcnta}%
661 }
662 \let\@advance@macro\l@advance@macro
663 \DeclareOption{symbol}{\renewcommand\thefootnote{\fnsymbol{footnote}}}
664 \newif\ifFN@robust \FN@robustfalse
665 \DeclareOption{symbol*}{%
666   \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
667   \FN@robusttrue
668   \AtEndOfPackage{\setfnsymbol{lamport*-robust}}}%
669 }
670 \newif\ifFN@para \FN@parafalse
671 \DeclareOption{para}{%

```

Options are executed in the order of declaration, thus no point in checking for side option as footmisc did in the past

```

672 %   \PackageError{footmisc}{Option "\CurrentOption" incompatible with
673 %                                     option "side"}%
674 %                                     {I shall ignore "\CurrentOption"}%
675 \FN@paratrue
676 \setlength\footnotemargin{-\maxdimen} % default when para is used
677 }
678 \DeclareOption{side}{\ifFN@para
679   \PackageError{footmisc}{Option "\CurrentOption" incompatible with
680   option "para"}%
681   {I shall ignore "\CurrentOption"}%
682 \else
683   \AddToHook{fntext/para}{%
684     \hsize\marginparwidth % correct the default \hsize
685     \footnotesep\z@ % don't add a default separation
686   }
687   \AssignSocketPlug{fntext/process}{side}
688 % \AssignSocketPlug{fntext/make}{default}
689 \AssignSocketPlug{fntext/begin}{noop}
690 \AssignSocketPlug{fntext/end}{noop}
691 \fi
692 }
693 \let\footnotelayout\@empty
694 \DeclareOption{ragged}{%
695   \@ifundefined{RaggedRight}{%
696     {\renewcommand\footnotelayout{\linepenalty50 \raggedright}}%
697     {\renewcommand\footnotelayout{\linepenalty50 \RaggedRight}}%
698 }
699 \newif\ifFN@perpage
700 \FN@perpagefalse
701 \DeclareOption{perpage}{%
702   \FN@perpagetrue
703 }

```



```

704 \newif\ifFN@fixskip      \FN@fixskipfalse
705
706 \let\FN@bottomcases\thr@@
707 \newif\ifFN@abovefloats \FN@abovefloatstrue
708 \DeclareOption{bottom}{%
709   \let\FN@bottomcases\@ne
710   \FN@abovefloatsfalse
711   \FN@fixskiptrue
712 }
713 \DeclareOption{bottomfloats}{%
714   \let\FN@bottomcases\tw@
715   \FN@abovefloatstrue \FN@fixskiptrue
716 }
717 \DeclareOption{abovefloats}{\FN@abovefloatstrue \FN@fixskiptrue}
718 \DeclareOption{belowfloats}{\FN@abovefloatsfalse \FN@fixskiptrue}
719 \DeclareOption{marginal}{%
720   \footnotemargin-0.8em\relax
721 }
722 \DeclareOption{flushmargin}{%
723   \footnotemarginOpt\relax
724 }
725 \newif\ifFN@hangfoot    \FN@hangfootfalse
726 \DeclareOption{hang}{%
727   \FN@hangfoottrue
728 }
729 \newcommand*{\hangfootparskip}{0.5\baselineskip}
730 \newcommand*{\hangfootparindent}{0em}%
731 \DeclareOption{norule}{%
732   \renewcommand\footnoterule{}%
733   \advance\skip\footins 4\p@\@plus2\p@\relax
734 }
735 \DeclareOption{splitrule}{%
736   \gdef\split@prev{0}
737   \let\pagefootnoterule\footnoterule
738   \let\mpfootnoterule\footnoterule
739   \def\splitfootnoterule{\kern-3\p@ \hrule \kern2.6\p@}
740   \def\footnoterule{\relax
741     \ifx \listdepth\@implistdepth
742       \mpfootnoterule
743     \else
744       \ifnum\split@prev=\z@
745         \pagefootnoterule
746       \else
747         \splitfootnoterule
748       \fi
749     \xdef\split@prev{\the\insertpenalties}%
750     \fi
751   }%
752 }
753 \newif\ifFN@stablefootnote \FN@stablefootnotefalse
754 \DeclareOption{stable}{\FN@stablefootnotetrue}
755 \newif\ifFN@multiplefootnote \FN@multiplefootnotefalse
756 \DeclareOption{multiple}{\FN@multiplefootnotetrue}
757 \ProcessOptions

```

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

```

758 \ifFN@para
759   \NewSocketPlug{build/column/footnotes}{para}{%
760     \global\setbox\footins\vbox{\FN@makefootnoteparagraph}%
761   }
762   \AssignSocketPlug{build/column/footnotes}{para}
763 \fi

764 \ifFN@fixskip
765   \def\@outputbox@removebskip{%
766     \ifx\@textbottom\relax \else
767       \@outputbox@append{%
768         \@tempskipa\lastskip
769         \ifnum \gluestretchorder\@tempskipa>\z@
770           \vskip-\@tempskipa
771           \xdef\@outputbox@reinsertbskip
772             {\noexpand\@outputbox@append{\vskip\the\@tempskipa}}%
773         \else
774           \global\let\@outputbox@reinsertbskip\relax
775         \fi
776       }%
777     \fi
778   }
779   \let\@outputbox@reinsertbskip\relax
780 \else
781   \let\@outputbox@removebskip \relax
782   \let\@outputbox@reinsertbskip\relax
783 \fi

784 \ifcase \FN@bottomcases\relax
785   \ERROR
786 \or %1 bottom option
787   \ifFN@abovefloats
788     \AssignSocketPlug {build/column/outputbox}{space-footnotes-floats}
789   \else
790     \AssignSocketPlug {build/column/outputbox}{floats-space-footnotes}
791   \fi
792 \or %2 bottomfloats
793   \ifFN@abovefloats
794     \AssignSocketPlug {build/column/outputbox}{footnotes-space-floats}
795   \else
796     \AssignSocketPlug {build/column/outputbox}{space-floats-footnotes}
797   \fi
798 \or %3 neither bottom nor bottomfloats
799   \ifFN@abovefloats
800     \AssignSocketPlug {build/column/outputbox}{footnotes-floats}
801   \else
802     \AssignSocketPlug {build/column/outputbox}{floats-footnotes}
803   \fi
804 \else
805   \ERROR
806 \fi
807
808 % next can be dropped when cleaned up

```

```

809 \newif\ifFN@setspace
810 \@ifpackageloaded{setspace}%
811 {%
812   \FN@setspacetrue
813   \@ifclassloaded{memoir}%
814   {%
815     \AddToHook{fntext}{\let\baselinestretch\m@m@singlespace}%
816     \let\FN@baselinestretch\m@m@singlespace
817   }%
818   {%
819     \AddToHook{fntext}{\let\baselinestretch\setspace@singlespace}%
820     \let\FN@baselinestretch\setspace@singlespace
821   }%
822 }%
823 {%
824   \FN@setspacefalse
825 }
826
827
828
829 \ifFN@para
830
831 % \AssignSocketPlug{fntext/process}{default}
832 \AssignSocketPlug{fntext/make}{para}
833 \AssignSocketPlug{fntext/begin}{noop}
834 \AssignSocketPlug{fntext/end}{para}
835
836 \fi
837
838
839
840 \ifFN@para
841 \let\FN@tempboxa\@tempboxa
842 \newbox\FN@tempboxb
843 \newbox\FN@tempboxc
844 \newskip\footglue \footglue=1em plus.3em minus.3em
845
846 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
847 \newdimen\footnotebaselineskip
848
849 % establish late:
850
851 \AddToHook{begindocument/before} {%
852   {%
853     \footnotesize
854     \global\footnotebaselineskip=\normalbaselineskip
855   }%
856 }

```

The coding is based on David Kastrup's improvement to Don Knuth's original implementation. You find in the  $\TeX$ book if you own the latest edition.

```

857
858 \long\def\FN@makefootnoteparagraph{%
859   \FN@setfootnoteparawidth
860   \@parboxrestore

```

```

861     \baselineskip=\footnotebaselineskip
862     \unvbox\footins \FN@removehboxes
863     \RawParEnd
864 }
865 \def\FN@removehboxes{\setbox\FN@tempboxa\lastbox
866 \ifhbox\FN@tempboxa{\FN@removehboxes}%
867 \unhbox\FN@tempboxa
868 \else
869 \RawNoindent
870 \rule{z@\footnotesep
871 \fi
872 }
873 \fi
874
875
876 \@ifpackageloaded{multicol}
877 {\def\FN@setfootnoteparawidth
878 {\hsize\ifnum\doublecol@number>\@ne
879 \textwidth
880 \else \columnwidth \fi}}
881 {\def\FN@setfootnoteparawidth{\hsize\columnwidth}}
882
883 \ifFN@perpage
884 \RequirePackage{perpage}
885 \MakePerPage{footnote}

```

Fix a bug in perpage ...

```

886 \def\@stpelt#1{\global\csname c@#1\endcsname \m@ne
887 \stepcounter{#1}%
888 \pp@fix@MakePerPage{#1}%
889 }
890 \def\pp@fix@MakePerPage#1{%
891 \ifnum \value{#1}>\z@
892 \addtocounter{#1}\m@ne\fi
893 }

```

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`:

```

894 \def\pp@c1@end@iii\stepcounter#1\pp@fix@MakePerPage#2{}
895 \fi
896
897
898 \ifFN@para
899
900 % This can use the default interface, except that a negative value for
901 % \footnotemargin makes little sense, so we test for this and warn if
902 % necessary. But -\maxdimen is ok again, so would need to be a little bit more elaborate.
903 %
904
905 %\AddToHook{fntext/para}{

```

```

906 % \ifdim \footnotemargin >\z@ \else
907 %   \PackageWarningNoline{footmisc}{Option 'para' needs positive \noexpand\footnotemargin}%
908 %   \footnotemargin 1.8em\relax
909 % \fi
910 %}
911
912
913 \AddToHook{fntext/begin}{\nobreak \hspace{.2em}}
914
915 \else
916
917 \ifFN@hangfoot
918   \long\def\footmisc@hang@makefntext#1{%
919     \bgroup
920     \SuspendTagging{footmisc}%
921     \setbox\@tempboxa\hbox{%
922       \ifdim\footnotemargin>\z@
923         \hb@xt@\footnotemargin{\@makefnmark\hss}%
924       \else
925         \@makefnmark
926       \fi
927     }%
928     \leftmargin\wd\@tempboxa
929     \rightmargin\z@
930     \linewidth \columnwidth
931     \advance \linewidth -\leftmargin
932     \parshape \@ne \leftmargin \linewidth
933     \footnotesize
934     \@setpar{\@par}%
935     \ResumeTagging{footmisc}%
936     \leavevmode

```

Typesetting the mark twice means that one can't have any material inside that gets unhappy in that case. That shouldn't be a problem, but perhaps we have to come up with a more elaborate solution in the end.

```

937   \UseSocket{tagssupport/fntext/mark}%
938   {\llap{%
939     \ifdim\footnotemargin>\z@
940       \hb@xt@\footnotemargin{\@makefnmark\hss}%
941     \else
942       \@makefnmark
943     \fi
944   }}%
945   \parskip\hangfootparskip\relax
946   \parindent\hangfootparindent\relax
947   \footnotelayout#1%
948   \par
949 \egroup
950 }

```

Defined in a roundabout way so that we can test for it when patching classes that are not updated.

```

951   \let \@makefntext \footmisc@hang@makefntext
952
953 \else

```

```

954
955 % This is now using the default interface:
956 %
957 % \long\def\@makefntext#1{%
958 %     \parindent1em
959 %     \noindent
960 %     \ifdim\footnotemargin>\z@
961 %         \hb@xt@ \footnotemargin{\hss\@makefnmark}%
962 %     \else
963 %         \ifdim\footnotemargin=\z@
964 %             \llap{\@makefnmark}%
965 %         \else
966 %             \llap{\hb@xt@ -\footnotemargin{\@makefnmark\hss}}%
967 %         \fi
968 %     \fi
969 % \footnotelayout#1%
970 % }
971
972 \fi
973 \fi
974
975
976
977
978 \ifFN@multiplefootnote
979 \providecommand*\multiplefootnotemarker}{3sp}
    We tag the separator as artifact
    TODO: why is this done with \providecommand?
980 \ExplSyntaxOn
981 \providecommand*\multfootsep}{\tag_mc_end_push:\tag_mc_begin:n{artifact},\tag_mc_end:\tag
982 \ExplSyntaxOff
983 \AddToHook{fnmark}      {\FN@mf@check}
984 \AddToHook{fnmark/end}  {\FN@mf@prepare}
985 %
986 \def\FN@mf@prepare{%
987     \kern-\multiplefootnotemarker
988     \kern\multiplefootnotemarker\relax
989 }
990 \def\FN@mf@check{%
991     \ifdim\lastkern=\multiplefootnotemarker\relax
992 %?? is that necessary or even correct ??
993     \edef\@x@sf{\the\spacefactor}%
994 %?? shouldn't that be 2 unkerns ?? (none would also be ok)
995     \unkern % new
996     \unkern
997     \textsuperscript{\multfootsep}%
998     \spacefactor\@x@sf\relax
999     \fi
1000 }
1001 \else
1002 \let\FN@mf@prepare\relax
1003 \fi
1004 \ifFN@stablefootnote

```

```

1005 \let\FN@sf@@footnote\footnote
1006 \def\footnote{\ifx\protect\@typeset@protect
1007   \expandafter\FN@sf@@footnote
1008   \else
1009     \expandafter\FN@sf@gobble@opt
1010   \fi
1011 }
1012 \edef\FN@sf@gobble@opt{\noexpand\protect
1013   \expandafter\noexpand\csname FN@sf@gobble@opt \endcsname}
1014 \expandafter\def\csname FN@sf@gobble@opt \endcsname{%
1015   \@ifnextchar[%]
1016     \FN@sf@gobble@twobrace
1017     \@gobble
1018 }
1019 \def\FN@sf@gobble@twobrace[#1]#2{}
1020 \let\FN@sf@@footnotemark\footnotemark
1021 \def\footnotemark{\ifx\protect\@typeset@protect
1022   \expandafter\FN@sf@@footnotemark
1023   \else
1024     \expandafter\FN@sf@gobble@optonly
1025   \fi
1026 }
1027 \edef\FN@sf@gobble@optonly{\noexpand\protect
1028   \expandafter\noexpand\csname FN@sf@gobble@optonly \endcsname}
1029 \expandafter\def\csname FN@sf@gobble@optonly \endcsname{%
1030   \@ifnextchar[%]
1031     \FN@sf@gobble@bracket
1032     {}%
1033 }
1034 \def\FN@sf@gobble@bracket[#1]{}
1035 \fi
1036 \newcommand\setfnsymbol[1]{%
1037   \@bsphack
1038   \@ifundefined{FN@fnsymbol@#1}%
1039   {%
1040     \PackageError{footmisc}{Symbol style "#1" not known}%
1041     \@eha
1042   }{%
1043     \expandafter\let\expandafter\@fnsymbol\csname
1044       FN@fnsymbol@#1\endcsname
1045   }%
1046   \@esphack
1047 }
1048 \let\FN@fnsymbol@lampoort\@fnsymbol
1049 \newif\if@tempswb
1050 \DeclareDocumentCommand\DefineFNSymbols {sm0{text}m}{%
1051   \expandafter\ifx\csname FN@fnsymbol@#2\endcsname\relax
1052     \PackageInfo{footmisc}{Declaring symbol style #2}%
1053   \else
1054     \PackageWarning{footmisc}{Redeclaring symbol style #2}%
1055   \fi
1056   \toks@{}%
1057   \def\@tempb{\end}%
1058   \FN@build@symboldef#4\end

```

```

1059 \def\@tempc{math}%
1060 \def\@tempd{#3}%
1061 \expandafter\xdef\csname FN@fnsymbol@#2\endcsname##1{%
1062   \ifx\@tempc\@tempd
1063     \noexpand\ensuremath
1064   \else
1065     \noexpand\nfss@text
1066   \fi
1067   {%
1068     \noexpand\ifcase##1%
1069     \the\toks@
1070     \noexpand\else
1071     \IfBooleanTF#1{\noexpand\@ctrerr}%
1072     {\noexpand\FN@orange##1}%
1073     \noexpand\fi
1074   }%
1075 }%
1076 }
1077 \def\FN@build@symboldef#1{%
1078   \def\@tempa{#1}%
1079   \ifx\@tempa\@tempb
1080   \else
1081     \toks@\expandafter{\the\toks@\or#1}%
1082     \expandafter\FN@build@symboldef
1083   \fi
1084 }
1085 \DeclareDocumentCommand\DefineFNsymbolsTM {smm}{%
1086   \expandafter\ifx\csname FN@fnsymbol@#2\endcsname\relax
1087     \PackageInfo{footmisc}{Declaring symbol style #2}%
1088   \else
1089     \PackageWarning{footmisc}{Redeclaring symbol style #2}%
1090   \fi
1091   \toks@{}%
1092   \def\@tempb{\end}%
1093   \FN@build@symboldefTM#3\end\@null
1094   \expandafter\xdef\csname FN@fnsymbol@#2\endcsname##1{%
1095     \noexpand\ifcase##1%
1096     \the\toks@
1097     \noexpand\else
1098     \IfBooleanTF#1{\noexpand\@ctrerr}%
1099     {\noexpand\FN@orange##1}%
1100     \noexpand\fi
1101   }%
1102 }
1103 \def\FN@build@symboldefTM#1#2{%
1104   \def\@tempa{#1}%
1105   \ifx\@tempa\@tempb
1106   \else
1107     \toks@\expandafter{\the\toks@\or\TextOrMath{#1}{#2}}%
1108     \expandafter\FN@build@symboldefTM
1109   \fi
1110 }
1111 \def\FN@orange#1{%
1112   \ifFN@robust

```



```

1113     \@arabic#1%
1114     \@bsphack
1115     \PackageInfo{footmisc}{Footnote number \number#1 out of range}%
1116     \protect\@fnsymbol@orange
1117     \@esphack
1118     \else \@ctrerr \fi
1119 }
1120 \global\let\@diagnose@fnsymbol@orange\relax
1121 \AtEndDocument{\@diagnose@fnsymbol@orange}
1122 \def\@fnsymbol@orange{%
1123     \gdef\@diagnose@fnsymbol@orange{%
1124         \PackageWarningNoLine{footmisc}{Some footnote number(s)
1125             were out of range
1126             \MessageBreak
1127             see log for details%
1128         }%
1129     }%
1130 }
1131 \DefineFNSymbolsTM{bringhurst}{%
1132     \textasteriskcentered *%
1133     \textdagger \dagger
1134     \textdaggerdbl \ddagger
1135     \textsection \mathsection
1136     \textbardbl \||%
1137     \textparagraph \mathparagraph
1138 }%
1139 \DefineFNSymbolsTM{chicago}{%
1140     \textasteriskcentered *%
1141     \textdagger \dagger
1142     \textdaggerdbl \ddagger
1143     \textsection \mathsection
1144     \textbardbl \||%
1145     \#\#%
1146 }%
1147 \DefineFNSymbolsTM{wiley}{%
1148     \textasteriskcentered *%
1149     {\textasteriskcentered\textasteriskcentered}{**}%
1150     \textdagger \dagger
1151     \textdaggerdbl \ddagger
1152     \textsection \mathsection
1153     \textparagraph \mathparagraph
1154     \textbardbl \||%
1155 }%
1156 \DefineFNSymbolsTM{lampport-robust}{%
1157     \textasteriskcentered *%
1158     \textdagger \dagger
1159     \textdaggerdbl \ddagger
1160     \textsection \mathsection
1161     \textparagraph \mathparagraph
1162     \textbardbl \||%
1163     {\textasteriskcentered\textasteriskcentered}{**}%
1164     {\textdagger\textdagger}{\dagger\dagger}%
1165     {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
1166 }

```

```

1167 \DefineFNsymbolsTM*{\lamport*}{%
1168   \textasteriskcentered *%
1169   \textdagger    \dagger
1170   \textdaggerdbl \ddagger
1171   \textsection   \mathsection
1172   \textparagraph \mathparagraph
1173   \textbardbl    \|%
1174   {\textasteriskcentered\textasteriskcentered}{**}%
1175   {\textdagger\textdagger}{\dagger\dagger}%
1176   {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
1177   {\textsection\textsection}{\mathsection\mathsection}%
1178   {\textparagraph\textparagraph}{\mathparagraph\mathparagraph}%
1179   {\textasteriskcentered\textasteriskcentered\textasteriskcentered}{***}%
1180   {\textdagger\textdagger\textdagger}{\dagger\dagger\dagger}%
1181   {\textdaggerdbl\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger\ddagger}%
1182   {\textsection\textsection\textsection}%%
1183   {\mathsection\mathsection\mathsection}%
1184   {\textparagraph\textparagraph\textparagraph}%%
1185   {\mathparagraph\mathparagraph\mathparagraph}%
1186 }
1187 \setfnsymbol{\lamport*}
1188 \DefineFNsymbolsTM{\lamport*-robust}{%
1189   \textasteriskcentered *%
1190   \textdagger    \dagger
1191   \textdaggerdbl \ddagger
1192   \textsection   \mathsection
1193   \textparagraph \mathparagraph
1194   \textbardbl    \|%
1195   {\textasteriskcentered\textasteriskcentered}{**}%
1196   {\textdagger\textdagger}{\dagger\dagger}%
1197   {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
1198   {\textsection\textsection}{\mathsection\mathsection}%
1199   {\textparagraph\textparagraph}{\mathparagraph\mathparagraph}%
1200   {\textasteriskcentered\textasteriskcentered\textasteriskcentered}{***}%
1201   {\textdagger\textdagger\textdagger}{\dagger\dagger\dagger}%
1202   {\textdaggerdbl\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger\ddagger}%
1203   {\textsection\textsection\textsection}%%
1204   {\mathsection\mathsection\mathsection}%
1205   {\textparagraph\textparagraph\textparagraph}%%
1206   {\mathparagraph\mathparagraph\mathparagraph}%
1207 }
1208 \newcommand\mpfootnotemark{%
1209   \@ifnextchar [%]
1210     \@xmpfootnotemark
1211     {%
1212       \stepcounter\@mpfn
1213       \protected@xdef\@thefnmark{\thempfn}%
1214       \@footnotemark
1215     }%
1216 }
1217 \def\@xmpfootnotemark[#1]{%
1218   \begingroup
1219     \csname c@\@mpfn\endcsname #1\relax
1220     \unrestored@protected@xdef\@thefnmark{\thempfn}%

```

```

1221 \endgroup
1222 \@footnotemark
1223 }
1224 \endinput
1225 </footmisc>

```

## Index

The *italic* numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

<b>Symbols</b>	
\#	1145
\	1136, 1144, 1154, 1162, 1173, 1194
<b>A</b>	
\addtocounter	892
\AddToHook	319, 398, 453, 455, 461, 463, 483, 683, 815, 819, 851, 905, 913, 983, 984
\AddToHookWithArguments	482
\advance	659, 733, 931
\AssignSocketPlug	209, 292, 304, 307, 316, 353, 552, 595, 596, 610, 619, 687, 688, 689, 690, 762, 788, 790, 794, 796, 800, 802, 831, 832, 833, 834
\AtBeginDocument	326, 465, 470
\AtEndDocument	1121
\AtEndOfPackage	668
<b>B</b>	
\baselineskip	729, 861
\baselinestretch	463, 815, 819
\begingroup	438, 1218
\bgroup	919
bool commands:	
\bool_gset_false:N	118
\bool_gset_true:N	117
\bool_if:NTF	121, 133, 522, 538, 578
\bool_new:N	69, 76, 116
\bool_set_false:N	207, 441, 474, 477
\bool_set_true:N	70, 77, 444
\box	302
<b>C</b>	
\columnwidth	16, 238, 301, 880, 881, 930
\cs	247, 248
cs commands:	
\cs_generate_variant:Nn	80, 81, 503, 517, 562
\cs_gset_eq:NN	128, 152
\cs_if_eq:NNTF	400, 405
\cs_new:Npn	388, 390, 392, 394, 396
\cs_new_eq:NN	197, 362
\cs_new_protected:Npn	82, 95, 117, 118, 119, 131, 155, 182, 211, 332, 364, 481, 490, 496, 504, 553, 557
\cs_set:Npn	385
\cs_set_eq:NN	199, 321, 368, 371, 374, 377, 379, 402, 425, 427, 457, 458, 459
\cs_set_protected:Npn	485
\csname	20, 886, 1013, 1014, 1028, 1029, 1043, 1051, 1061, 1086, 1094, 1219
\CurrentOption	672, 674, 679, 681
<b>D</b>	
\dagger	1133, 1141, 1150, 1158, 1164, 1169, 1175, 1180, 1190, 1196, 1201
\ddagger	1134, 1142, 1151, 1159, 1165, 1170, 1176, 1181, 1191, 1197, 1202
\DebugFNotesOff	7, 117
\DebugFNotesOn	7, 117
\DeclareCurrentRelease	643, 646
\DeclareDocumentCommand	417, 422, 431, 436, 1050, 1085
\DeclareOption	663, 665, 671, 678, 694, 701, 708, 713, 717, 718, 719, 722, 726, 731, 735, 754, 756
\DeclareRelease	642, 645
\def	12, 18, 29, 30, 34, 242, 385, 464, 657, 658, 739, 740, 765, 858, 865, 877, 881, 886, 890, 894, 918, 957, 986, 990, 1006, 1014, 1019, 1021, 1029, 1034, 1057, 1059, 1060, 1077, 1078, 1092, 1103, 1104, 1111, 1122, 1217
default (plug)	5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 21
\DefineFNsymbols	1050
\DefineFNsymbolsTM	1085, 1131, 1139, 1147, 1156, 1167, 1188
\dimexpr	300
\dp	236, 298

<b>E</b>	
<code>\edef</code> .....	164, 657, 993, 1012, 1027
<code>\egroup</code> .....	949
<code>\else</code> .	341, 344, 347, 682, 743, 746, 766, 773, 780, 789, 795, 801, 804, 868, 880, 906, 915, 924, 941, 953, 962, 965, 1001, 1008, 1023, 1053, 1064, 1070, 1080, 1088, 1097, 1106, 1118
<code>\end</code> .....	1057, 1058, 1092, 1093
<code>\endcsname</code> .....	.. 20, 886, 1013, 1014, 1028, 1029, 1044, 1051, 1061, 1086, 1094, 1219
<code>\endgroup</code> .....	440, 1221
<code>\endinput</code> .....	1224
<code>\ensuremath</code> .....	1063
<code>\ERROR</code> .....	22, 362, 378, 415, 785, 805
exp commands:	
<code>\exp_after:wN</code> .....	384, 386
<code>\exp_args:No</code> .....	540
<code>\exp_not:N</code> .....	530
<code>\exp_stop_f:</code> ..	379, 389, 391, 393, 395
<code>\expandafter</code> .....	658, 1007, 1009, 1013, 1014, 1022, 1024, 1028, 1029, 1043, 1051, 1061, 1081, 1082, 1086, 1094, 1107, 1108
<code>\ExplSyntaxOff</code> .....	620, 982
<code>\ExplSyntaxOn</code> .....	64, 980
<b>F</b>	
<code>FEMark</code> (plug) .....	518
<code>FENote</code> (plug) .....	563
<code>FENoteLbl</code> (plug) .....	597
<code>FENotetext</code> (plug) .....	611
<code>\fi</code> .....	173, 191, 330, 349, 350, 351, 691, 748, 750, 763, 775, 777, 783, 791, 797, 803, 806, 836, 871, 873, 880, 892, 895, 909, 926, 943, 967, 968, 972, 973, 999, 1003, 1010, 1025, 1035, 1055, 1066, 1073, 1083, 1090, 1100, 1109, 1118
<code>\floatingpenalty</code> .....	237
<code>fnmark</code> (hook) .....	4, 4, 107
<code>fnmark/after</code> (hook) ...	4, 4, 4, 6, 17, 107
<code>fnmark/before</code> (hook) .....	4, 4, 4, 107
<code>fnmark/begin</code> (hook) .....	4, 4, 107
<code>fnmark/end</code> (hook) .....	4, 4, 17, 107
fnote commands:	
<code>\fnote_class_new:nm</code> .	9, 490, 490, 495
<code>\g_fnote_debug_bool</code> .....	116, 117, 118, 121, 133
<code>\fnote_footnotemark:</code> .....	8, 155, 155, 199, 457
<code>\fnote_footnotetext:n</code> .....	200, 211, 321, 462
<code>\fnote_gput_refs:nn</code> .....	530, 553, 557, 562, 583
<code>\l_fnote_link_bool</code> 26, 69, 474, 477, 538	
<code>\l_fnote_link_type_tl</code> .....	71, 542
<code>\fnote_makefncontext:n</code> .....	11, 21, 23, 332, 332, 402
<code>\fnote_mark_gpop_all:nnN</code> .....	9, 504, 504, 517, 580
<code>\fnote_mark_gput:nn</code> .....	9, 496, 496, 503, 524
<code>\fnote_set_fnmark:nn</code> ...	95, 95, 419
<code>\fnote_step_fnmark:nn</code> 82, 82, 424, 433	
<code>\l_fnote_type_tl</code> .....	67, 524, 580
fnote internal commands:	
<code>\l__fnote_autodetect_bool</code> .....	76, 441, 444, 522, 578
<code>\l__fnote_currentlabel_tl</code> .....	24, 28, 74, 442, 532, 534
<code>\l__fnote_currentrefs_seq</code> .....	75, 580, 581, 600, 601
<code>\l__fnote_currentstruct_tl</code> .....	26, 78, 79, 480, 577, 583, 602, 603
<code>\__fnote_debug_footnotemark:</code> ...	119, 128, 156
<code>\__fnote_debug_footnotetext:</code> ...	131, 152, 212
<code>\__fnote_footmark_finish:</code> .....	17
<code>\__fnote_footnotemark_default_-</code> finish: .....	17, 182, 182, 197, 427
<code>\__fnote_footnotemark_finish:</code> ...	17, 180, 182, 197, 425, 427, 429
<code>\__fnote_gput_ref:nn</code> 553, 553, 559, 560	
<code>\__fnote_label_hook:n</code> .....	481, 481, 482, 485
<code>\l__fnote_linktarget_tl</code> .....	73, 525, 534, 543
<code>\__fnote_patch:</code> .....	364, 406
<code>\__fnote_patch_@makefnmark:w</code> 377, 394	
<code>\__fnote_patch_hb@xt@:w</code> ...	371, 390
<code>\__fnote_patch_hbox:w</code> .....	368, 388
<code>\__fnote_patch_makebox:w</code> ...	374, 392
<code>\l__fnote_patch_seq</code> .....	414, 415
<code>\l__fnote_patch_tl</code> .....	361, 366, 367, 370, 373, 376, 383, 384, 386, 414
<code>\__fnote_tmp:w</code> .....	362, 368, 371, 374, 377, 379, 384, 385, 386
<code>\l__fnote_tmpa_prop</code> ...	65, 507, 508
<code>\l__fnote_tmpa_tl</code> .....	66
<code>\fnsymbol</code> .....	663
<code>fncontext</code> (hook) .....	6, 6, 111
<code>fncontext/after</code> (hook) .....	6, 6, 6, 111
<code>fncontext/before</code> (hook) .....	6, 6, 6, 111
<code>fncontext/begin</code> (hook) .....	6, 6, 6, 111
<code>fncontext/socket</code> .....	5, 5, 305







<code>\@null</code>	1093	<code>\FN@setspacetrue</code>	812
<code>\@outputbox@append</code>	767, 772	<code>\FN@sf@@footnote</code>	1005, 1007
<code>\@outputbox@reinsertbskip</code>	771, 774, 779, 782	<code>\FN@sf@@footnotemark</code>	1020, 1022
<code>\@outputbox@removebskip</code>	765, 781	<code>\FN@sf@gobble@bracket</code>	1031, 1034
<code>\@parboxrestore</code>	17, 239, 860	<code>\FN@sf@gobble@opt</code>	1009, 1012
<code>\@plus</code>	733	<code>\FN@sf@gobble@optonly</code>	1024, 1027
<code>\@setpar</code>	934	<code>\FN@sf@gobble@twobracket</code>	1016, 1019
<code>\@stpelt</code>	886	<code>\FN@stablefootnotefalse</code>	753
<code>\@tempa</code>	1078, 1079, 1104, 1105	<code>\FN@stablefootnotetrue</code>	754
<code>\@tempb</code>	1057, 1079, 1092, 1105	<code>\FN@tempboxa</code>	297, 298, 299, 300, 302, 841, 865, 866, 867
<code>\@tempboxa</code>	841, 921, 928	<code>\FN@tempboxb</code>	842
<code>\@tempc</code>	1059, 1062	<code>\FN@tempboxc</code>	843
<code>\@tempcnta</code>	658, 659, 660	<code>\FN@temptoken</code>	653
<code>\@tempd</code>	1060, 1062	<code>\footmisc@hang@makefntext</code>	405, 918, 951
<code>\@tempskipa</code>	768, 769, 770, 772	<code>\hb@xt@</code>	22, 23, 340, 348, 360, 370, 390, 391, 923, 940, 961, 966
<code>\@textbottom</code>	766	<code>\hyper@linkend</code>	468, 545
<code>\@textsuperscript</code>	34	<code>\hyper@linkstart</code>	467, 541
<code>\@thefnmark</code>	9, 10, 13, 14, 24, 27–29, 20, 34, 86, 91, 98, 103, 243, 439, 524, 580, 1213, 1220	<code>\hyper@nopatch@footnote</code>	464
<code>\@typeset@protect</code>	1006, 1021	<code>\if@tempswb</code>	1049
<code>\@x@sf</code>	164, 190, 993, 998	<code>\ifFN@abovefloats</code>	707, 787, 793, 799
<code>\@xmpfootnotemark</code>	1210, 1217	<code>\ifFN@fixskip</code>	704, 764
<code>\c@footnote</code>	666	<code>\ifFN@hangfoot</code>	725, 917
<code>\color@begingroup</code>	21, 253	<code>\ifFN@multiplefootnote</code>	755, 978
<code>\color@endgroup</code>	25, 280	<code>\ifFN@para</code>	670, 678, 758, 829, 840, 898
<code>\doublecol@number</code>	878	<code>\ifFN@perpage</code>	699, 883
<code>\FN@abovefloatsfalse</code>	710, 718	<code>\ifFN@robust</code>	664, 1112
<code>\FN@abovefloatstrue</code>	707, 715, 717	<code>\ifFN@setspace</code>	809
<code>\FN@baselinestretch</code>	816, 820	<code>\ifFN@stablefootnote</code>	753, 1004
<code>\FN@bottomcases</code>	706, 709, 714, 784	<code>\l@advance@macro</code>	657, 662
<code>\FN@build@symboldef</code>	1058, 1077, 1082	<code>\m@m@singlespace</code>	815, 816
<code>\FN@build@symboldefTM</code>	1093, 1103, 1108	<code>\m@ne</code>	886, 892
<code>\FN@fixskipfalse</code>	704	<code>\new@std@class@makecol</code>	454, 459
<code>\FN@fixskiptrue</code>	711, 715, 717, 718	<code>\nfss@text</code>	1065
<code>\FN@fnsymbol@lamport</code>	1048	<code>\old@std@class@makefntext</code>	357, 400, 458
<code>\FN@hangfootfalse</code>	725	<code>\p@</code>	733, 739
<code>\FN@hangfoottrue</code>	727	<code>\p@footnote</code>	243
<code>\FN@makefootnoteparagraph</code>	760, 858	<code>\pp@cl@end@iii</code>	894
<code>\FN@mf@check</code>	983, 990	<code>\pp@fix@MakePerPage</code>	888, 890, 894
<code>\FN@mf@prepare</code>	984, 986, 1002	<code>\protected@edef</code>	19, 243
<code>\FN@multiplefootnotefalse</code>	755	<code>\protected@write</code>	655
<code>\FN@multiplefootnotetrue</code>	756	<code>\protected@writeaux</code>	654
<code>\FN@orange</code>	1072, 1099, 1111	<code>\protected@xdef</code>	86, 98, 1213
<code>\FN@parafalse</code>	670	<code>\reset@font</code>	15, 227
<code>\FN@paratrue</code>	675	<code>\setspace@singlespace</code>	463, 819, 820
<code>\FN@perpagefalse</code>	700	<code>\split@prev</code>	736, 744, 749
<code>\FN@perpagetrue</code>	702	<code>\thr@@</code>	706
<code>\FN@removeboxes</code>	862, 865, 866	<code>\toks@</code>	1056, 1069, 1081, 1091, 1096, 1107
<code>\FN@robustfalse</code>	664	<code>\tw@</code>	714
<code>\FN@robusttrue</code>	667	<code>\unrestored@protected@xdef</code>	91, 103, 439, 1220
<code>\FN@setfootnoteparawidth</code>	859, 877, 881		
<code>\FN@setspacefalse</code>	824		



