

The latex-lab-footnotes code*

Frank Mittelbach, L^AT_EX Project

August 22, 2024

Abstract

to be written

Contents

1	Introduction	2
1.1	Configuration methods	3
2	Sockets and hooks	3
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
3	Tagging and hyperlinking support	7
3.1	Technical details for the tagging	7
3.2	Requirements for links	8
3.3	The algorithmus to connect marks and notes	8
3.3.1	\footref	9
3.3.2	\footnotemark after \footnotetext	9
3.4	Links	10
3.5	Implementation details regarding tagging	10
3.6	Handling the mark	10
3.7	Handling the footnotetext	10
3.8	Footnotes in minipages	10
4	TODOs	11

*

5	The Implementation	11
5.1	File declaration	11
5.2	code not fully handled yet	11
5.3	Temporary variables	12
5.4	Public variables	12
5.5	Internal variables	13
5.6	Variants	14
5.7	Updating <code>\@thefnmark</code>	14
5.8	Hooks	14
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
6	Reimplementing the <code>footmisc</code> package	31
	Index	43

1 Introduction

This code reimplements the footnote interfaces for \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^AT_EX 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^AT_EX'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 mark¹ 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:

¹Like 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`, `scrlttr2`, 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 csname changes)

Footnotes consist of a *footnotemark* (short: mark) that is typically placed in the text as a superscript number like this¹, 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 mark¹⁻³ 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:

tagsupport/fnmark (1 argument)

TODO: describe and decide on names

`tagsupport/fntext/begin` (no argument)

`tagsupport/fntext/end` (no argument)

`tagsupport/fntext/mark` (1 argument)

`tagsupport/fntext/text` (1 argument)

The *footnotemark* should create a `/Lb1` structure² that should contain a `/Ref` entry pointing to the structure of the *footnotetext*.

The *footnotetext* should create a `/FENote`³ 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`⁴ 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 want to avoid faulty nesting.
- It must detect which marks are related to which notes so that it can setup the `/Ref` cross-references.

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.⁵

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 a match it stores the structure numbers and removes the entries from the properties (so in a normal document the property will never contain more than a few entries).

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

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

⁴We tag it as `/footnotelabel`.

⁵Currently `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.

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.

3.3.1 `\footref`

`\footref` use 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^AT_EX_{2 ϵ} 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 `\@makefn`text.

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           [\ltxlabfootnotedate\space v\ltxlabfootnoteversion\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 %       \@makefn
```

```
text{%
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 {1}
```

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^AT_EX 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}

```

```

fntext/before (hook) Hooks in the footnotetext command:
fntext/after (hook) 111 \NewMirroredHookPair{fntext/before}{fntext/after}
    fntext (hook) 112 \NewHook{fntext}
fntext/begin (hook) 113 \NewHook{fntext/para}
    fntext/end (hook) 114 \NewHook{fntext/begin}
fntext/para (hook) 115 \NewHook{fntext/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 {fntext/process}
136       \socket_log:n {fntext/make}
137       \socket_log:n {fntext/begin}
138       \socket_log:n {fntext/end}
139
140       \socket_log:n {fntext/mark}
141       \socket_log:n {fntext/text}
142
143       \socket_log:n {tagssupport/fnmark}
144       \socket_log:n {tagssupport/fntext/begin}
145       \socket_log:n {tagssupport/fntext/end}
146       \socket_log:n {tagssupport/fntext/mark}
147       \socket_log:n {tagssupport/fntext/text}
148
149       \hook_log:n {fntext/before}
150       \hook_log:n {fntext}
151       \hook_log:n {fntext/para}
152       \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 {taggsupport/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 % scr1ttr2.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^AT_EX 2_ε 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 \str_if_exist:cF { l__socket_tagsupport/para/restore_plug_str }
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:nn {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 % fnpa 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 % fntext.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 `<text>` from the `\footnote` or `\footnotetext` and formats it.

```

293 \NewSocket    {fntext/make}{1}
294 \NewSocketPlug{fntext/make}{default}{ \@makefntext {#1} }
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         \footnotebaselineskip /\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 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$ 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 {tagsupport/fntext/mark} { \socket_use:n {fntext/mark} }
335 \socket_use:nn {tagsupport/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}

```

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

```

355 \NewSocket{tagsupport/fntext/mark}{1}
356 \NewSocket{tagsupport/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 `\@makefnntext` 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:Nx \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:Nx \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{tagsupport/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_if_active:T
556   {
557     \tag_struct_gput:nne {#1}{ref}{\tag_struct_object_ref:e { #2 }}
558   }
559 }
560 \cs_new_protected:Npn \fnote_gput_refs:nn #1 #2 % pair of numbers
561 {
562   \__fnote_gput_ref:nn {#1}{#2}
563   \__fnote_gput_ref:nn {#2}{#1}
564 }
565 \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 ??.)

```
566 \tl_new:N \l__fnote_dflt_struct_tl
567 \tl_set:Nn \l__fnote_dflt_struct_tl {1}
```

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

FENote (*plug*)

```
568 \NewSocketPlug{tagsupport/fntext/begin}{FENote}
569 {
570   \tag_mc_end_push:
test if a footnote is allowed, if not move up to the document structure.
571   \tag_check_child:nnTF {FENote}{pdf2}
572   {
573     \tag_struct_begin:n { tag=footnote }
574   }
575   {
576     \tag_struct_begin:n { tag=footnote,parent=\l__fnote_dflt_struct_tl }
577   }
```

Store the current structure number for labels.

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

We want to move the label structure here. So we provide a container It number is \l__fnote_currentstruct_tl plus 1.

```
579   \tag_struct_begin:n { tag=footnotelabel }\tag_struct_end:
```

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.

```
580   \bool_if:NTF \l__fnote_autodetect_bool
581   {
```

find open marks with identical \@thefnmark

```
582     \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:

```
583     \seq_map_inline:Nn \l__fnote_currentrefs_seq
584     {
585         \fnote_gput_refs:ee {##1}{ \l__fnote_currentstruct_tl }
586     }
587 }
```

If no auto detection is done

```
588     {%no auto
589
590     }
591 }
```

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.

```
592 \NewSocketPlug{taggsupport/fntext/end}{FENote}
593 {
594     \tag_struct_end:
595     \tag_mc_begin_pop:n{f}
596 }
```

At last assign the plugs:

```
597 \AssignSocketPlug{taggsupport/fntext/begin}{FENote}
598 \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.

```
599 \NewSocketPlug{taggsupport/fntext/mark}{FENoteLbl}
600 {
601     \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`).

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

Now we add the tagging commands. We move the structure of the label to to the container at the begin of the note.

```
605     \tag_struct_begin:n { tag=NonStruct,parent=\l__fnote_currentstruct_tl +1 }
606     \tag_mc_begin:n { tag=Lbl }
607     #1
608     \tag_mc_end:
609     \tag_struct_end:
```

```

610     \tag_mc_begin_pop:n{
611   }
612 \AssignSocketPlug{tagsupport/fntext/mark}{FENoteLbl}

```

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?

```

613 \NewSocketPlug{tagsupport/fntext/text}{FENotetext}
614 {
615   \tag_mc_end_push:
616   \tag_mc_begin:n{
617     #1
618     \tag_mc_end:
619     \tag_mc_begin_pop:n{
620   }
621 \AssignSocketPlug{tagsupport/fntext/text}{FENotetext}

```

```

622 \ExplSyntaxOff
623 </kernel>
624 <@@=)

```

6 Reimplementing the `footmisc` package

```

625 <*footmisc>
626 %%
627 %% Copyright (c) 1995–2011 Robin Fairbairns
628 %% Copyright (c) 2018–2023 Robin Fairbairns, Frank Mittelbach
629 %%
630 %% This file is part of the ‘latex-lab Bundle’.
631 %% -----
632 %%
633 %% It may be distributed and/or modified under the
634 %% conditions of the LaTeX Project Public License, either version 1.3c
635 %% of this license or (at your option) any later version.
636 %% The latest version of this license is in
637 %%   https://www.latex-project.org/lppl.txt
638 %% and version 1.3c or later is part of all distributions of LaTeX
639 %% version 2008 or later.
640 %%
641 %% This work has the LPPL maintenance status ‘maintained’.
642 %%
643 \NeedsTeXFormat{LaTeX2e}
644 \providecommand\DeclareRelease[3]{}
645 \providecommand\DeclareCurrentRelease[2]{}
646
647 \DeclareRelease{v5}{2011-06-06}{footmisc-2011-06-06.sty}
648 \DeclareCurrentRelease{}{2022-02-14}
649 \ProvidesPackage{latex-lab-footmisc}%
650   [2022/03/08 v6.0d
651   a miscellany of footnote facilities -- latex-lab version%

```

```

652 ]
653
654 \NeedsTeXFormat{LaTeX2e}[2020/10/01]
655 \newtoks\FN@temptoken
656 \providecommand\protected@writeaux{%
657   \protected@write\@auxout
658 }
659 \def\l@advance@macro{\@dvance@macro\edef}
660 \def\@dvance@macro#1#2#3{\expandafter\@tempcnta#2\relax
661   \advance\@tempcnta#3\relax
662   #1#2{\the\@tempcnta}%
663 }
664 \let\@advance@macro\l@advance@macro
665 \DeclareOption{symbol}{\renewcommand\thefootnote{\fnsymbol{footnote}}}
666 \newif\ifFN@robust \FN@robustfalse
667 \DeclareOption{symbol*}{%
668   \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
669   \FN@robusttrue
670   \AtEndOfPackage{\setfnsymbol{lampost*-robust}}%
671 }
672 \newif\ifFN@para \FN@parafalse
673 \DeclareOption{para}{%

```

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

```

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

```



```

703 \DeclareOption{perpage}{%
704   \FN@perpagetrue
705 }
706 \newif\ifFN@fixskip      \FN@fixskipfalse
707
708 \let\FN@bottomcases\thr@@
709 \newif\ifFN@abovefloats \FN@abovefloatstrue
710 \DeclareOption{bottom}{%
711   \let\FN@bottomcases\@ne
712   \FN@abovefloatsfalse
713   \FN@fixskiptrue
714 }
715 \DeclareOption{bottomfloats}{%
716   \let\FN@bottomcases\tw@
717   \FN@abovefloatstrue \FN@fixskiptrue
718 }
719 \DeclareOption{abovefloats}{\FN@abovefloatstrue \FN@fixskiptrue}
720 \DeclareOption{belowfloats}{\FN@abovefloatsfalse \FN@fixskiptrue}
721 \DeclareOption{marginal}{%
722   \footnotemargin-0.8em\relax
723 }
724 \DeclareOption{flushmargin}{%
725   \footnotemargin0pt\relax
726 }
727 \newif\ifFN@hangfoot    \FN@hangfootfalse
728 \DeclareOption{hang}{%
729   \FN@hangfoottrue
730 }
731 \newcommand*\hangfootparskip{0.5\baselineskip}
732 \newcommand*\hangfootparindent{0em}%
733 \DeclareOption{norule}{%
734   \renewcommand\footnoterule{}%
735   \advance\skip\footins 4\p@\@plus2\p@\relax
736 }
737 \DeclareOption{splitrule}{%
738   \gdef\split@prev{0}
739   \let\pagefootnoterule\footnoterule
740   \let\mpfootnoterule\footnoterule
741   \def\splitfootnoterule{\kern-3\p@ \hrule \kern2.6\p@}
742   \def\footnoterule{\relax
743     \ifx \@listdepth\@mplistdepth
744       \mpfootnoterule
745     \else
746       \ifnum\split@prev=\z@
747         \pagefootnoterule
748       \else
749         \splitfootnoterule
750       \fi
751       \xdef\split@prev{\the\insertpenalties}%
752     \fi
753   }%
754 }
755 \newif\ifFN@stablefootnote \FN@stablefootnotefalse
756 \DeclareOption{stable}{\FN@stablefootnotetrue}

```

```

757 \newif\ifFN@multiplefootnote \FN@multiplefootnotefalse
758 \DeclareOption{multiple}{\FN@multiplefootnotetrue}
759 \ProcessOptions

```

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

```

760 \ifFN@para
761 \NewSocketPlug{@makecol/footnotes}{para}{%
762   \global\setbox\footins\vbox{\FN@makefootnoteparagraph}%
763 }
764 \AssignSocketPlug{@makecol/footnotes}{para}
765 \fi
766 \ifFN@fixskip
767 \def\@outputbox@removebskip{%
768   \ifx\@textbottom\relax \else
769     \@outputbox@append{%
770       \@tempskipa\lastskip
771       \ifnum \gluestretchorder\@tempskipa>\z@
772         \vskip-\@tempskipa
773       \xdef\@outputbox@reinsertbskip
774         {noexpand\@outputbox@append{\vskip\the\@tempskipa}}%
775     \else
776       \global\let\@outputbox@reinsertbskip\relax
777     \fi
778   }%
779 \fi
780 }
781 \let\@outputbox@reinsertbskip\relax
782 \else
783 \let\@outputbox@removebskip \relax
784 \let\@outputbox@reinsertbskip\relax
785 \fi
786 \ifcase \FN@bottomcases\relax
787 \ERROR
788 \or %1
789 \ifFN@abovefloats
790 \AssignSocketPlug {@makecol/outputbox}{space-footnotes-floats}
791 \else
792 \AssignSocketPlug {@makecol/outputbox}{floats-footnotes-space}
793 \fi
794 \or %2
795 \ifFN@abovefloats
796 \AssignSocketPlug {@makecol/outputbox}{footnotes-space-floats}
797 \else
798 \AssignSocketPlug {@makecol/outputbox}{space-floats-footnotes}
799 \fi
800 \or %3
801 \ifFN@abovefloats
802 \AssignSocketPlug {@makecol/outputbox}{footnotes-floats}
803 \else
804 \AssignSocketPlug {@makecol/outputbox}{floats-footnotes}
805 \fi
806 \else
807 \ERROR

```

```

808 \fi
809
810 % next can be dropped when cleaned up
811 \newif\ifFN@setspace
812 \@ifpackageloaded{setspace}%
813 {%
814   \FN@setspace>true
815   \@ifclassloaded{memoir}%
816   {%
817     \AddToHook{fntext}{\let\baselinestretch\m@m@singlespace}%
818     \let\FN@baselinestretch\m@m@singlespace
819   }%
820   {%
821     % \AddToHook{fntext}{\let\baselinestretch\setspace@singlespace}%
822     \let\FN@baselinestretch\setspace@singlespace
823   }%
824 }%
825 {%
826   \FN@setspace>false
827 }
828
829
830
831 \ifFN@para
832
833 % \AssignSocketPlug{fntext/process}{default}
834 \AssignSocketPlug{fntext/make}{para}
835 \AssignSocketPlug{fntext/begin}{noop}
836 \AssignSocketPlug{fntext/end}{para}
837
838 \fi
839
840
841
842 \ifFN@para
843   \let\FN@tempboxa\@tempboxa
844   \newbox\FN@tempboxb
845   \newbox\FN@tempboxc
846   \newskip\footglue \footglue=1em plus.3em minus.3em
847
848   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
849   \newdimen\footnotebaselineskip
850
851   % establish late:
852
853   \AddToHook{begindocument/before} {%
854     {%
855       \footnotesize
856       \global\footnotebaselineskip=\normalbaselineskip
857     }%
858   }

```

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.

```

859
860 \long\def\FN@makefootnoteparagraph{%
861   \FN@setfootnoteparawidth
862   \@parboxrestore
863   \baselineskip=\footnotebaselineskip
864   \unvbox\footins \FN@removeboxes
865   \RawParEnd
866 }
867 \def\FN@removeboxes{\setbox\FN@tempboxa\lastbox
868   \ifhbox\FN@tempboxa{\FN@removeboxes}%
869   \unhbox\FN@tempboxa
870   \else
871     \RawNoindent
872     \rule\z@\footnotesep
873   \fi
874 }
875 \fi
876
877
878 \@ifpackageloaded{multicol}
879   {\def\FN@setfootnoteparawidth
880     {\hsize\ifnum\doublecol@number>\@ne
881       \textwidth
882       \else \columnwidth \fi}}
883   {\def\FN@setfootnoteparawidth{\hsize\columnwidth}}
884
885 \ifFN@perpage
886   \RequirePackage{perpage}
887   \MakePerPage{footnote}

```

Fix a bug in perpage ...

```

888 \def\@stpelt#1{\global\csname c@#1\endcsname \m@ne
889   \stepcounter{#1}%
890   \pp@fix@MakePerPage{#1}%
891 }
892 \def\pp@fix@MakePerPage#1{%
893   \ifnum \value{#1}>\z@
894     \addtocounter{#1}\m@ne\fi
895 }

```

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`:

```

896 \def\pp@c1@end@iii\stepcounter#1\pp@fix@MakePerPage#2{}
897 \fi
898
899
900 \ifFN@para
901
902 % This can use the default interface, except that a negative value for
903 % \footnotemargin makes little sense, so we test for this and warn if

```

```

904 % necessary. But -\maxdimen is ok again, so would need to be a little bit more elaborate.
905 %
906
907 %\AddToHook{fntext/para}{
908 % \ifdim \footnotemargin >\z@ \else
909 % \PackageWarningNoline{footmisc}{Option 'para' needs positive \noexpand\footnotemargin}%
910 % \footnotemargin 1.8em\relax
911 % \fi
912 %}
913
914
915 \AddToHook{fntext/begin}{\nobreak \hspace{.2em}}
916
917 \else
918
919 \ifFN@hangfoot
920 \long\def\footmisc@hang@makefntext#1{%
921 \bgroup
922 \SuspendTagging{footmisc}%
923 \setbox\@tempboxa\hbox{%
924 \ifdim\footnotemargin>\z@
925 \hb@xt@\footnotemargin{\@makefnmark\hss}%
926 \else
927 \@makefnmark
928 \fi
929 }%
930 \leftmargin\wd\@tempboxa
931 \rightmargin\z@
932 \linewidth \columnwidth
933 \advance \linewidth -\leftmargin
934 \parshape \@ne \leftmargin \linewidth
935 \footnotesize
936 \@setpar{\@@par}}%
937 \ResumeTagging{footmisc}%
938 \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.

```

939 \UseSocket{tagsupport/fntext/mark}%
940 {\llap{%
941 \ifdim\footnotemargin>\z@
942 \hb@xt@\footnotemargin{\@makefnmark\hss}%
943 \else
944 \@makefnmark
945 \fi
946 }}%
947 \parskip\hangfootparskip\relax
948 \parindent\hangfootparindent\relax
949 \footnotelayout#1%
950 \par
951 \egroup
952 }

```

Defined in a roundabout way so that we can test for it when patching classes that are

not updated.

```
953 \let \@makefntext \footmisc@hang@makefntext
954
955 \else
956
957 % This is now using the default interface:
958 %
959 % \long\def\@makefntext#1{%
960 % \parindent1em
961 % \noindent
962 % \ifdim\footnotemargin>\z@
963 % \hb@xt@ \footnotemargin{\hss\@makefnmark}%
964 % \else
965 % \ifdim\footnotemargin=\z@
966 % \llap{\@makefnmark}%
967 % \else
968 % \llap{\hb@xt@ -\footnotemargin{\@makefnmark\hss}}%
969 % \fi
970 % \fi
971 % \footnotelayout#1%
972 % }
973
974 \fi
975 \fi
976
977
978
979
980 \ifFN@multiplefootnote
981 \providecommand*\multiplefootnotemarker}{3sp}
982
983 We tag the separator as artifact
984 TODO: why is this done with \providecommand?
985
986 \ExplSyntaxOn
987 \providecommand*\multfootsep}{\tag_mc_end_push:\tag_mc_begin:n{artifact},\tag_mc_end:\tag
988 \ExplSyntaxOff
989 \AddToHook{fnmark} {\FN@mf@check}
990 \AddToHook{fnmark/end} {\FN@mf@prepare}
991 %
992 \def\FN@mf@prepare{%
993 \kern-\multiplefootnotemarker
994 \kern\multiplefootnotemarker\relax
995 }
996 \def\FN@mf@check{%
997 \ifdim\lastkern=\multiplefootnotemarker\relax
998 %?? is that necessary or even correct ??
999 \edef\x@sf{\the\spacefactor}%
1000 %?? shouldn't that be 2 unkerns ?? (none would also be ok)
1001 \unkern % new
1002 \unkern
1003 \textsuperscript{\multfootsep}%
1004 \spacefactor\x@sf\relax
1005 \fi
1006 }
```

```

1003 \else
1004   \let\FN@mf@prepare\relax
1005 \fi
1006 \ifFN@stablefootnote
1007 \let\FN@sf@@footnote\footnote
1008 \def\footnote{\ifx\protect\@typeset@protect
1009   \expandafter\FN@sf@@footnote
1010   \else
1011     \expandafter\FN@sf@gobble@opt
1012   \fi
1013 }
1014 \edef\FN@sf@gobble@opt{\noexpand\protect
1015   \expandafter\noexpand\csname FN@sf@gobble@opt \endcsname}
1016 \expandafter\def\csname FN@sf@gobble@opt \endcsname{%
1017   \@ifnextchar [%]
1018     \FN@sf@gobble@twobracket
1019     \@gobble
1020 }
1021 \def\FN@sf@gobble@twobracket[#1]#2{}
1022 \let\FN@sf@@footnotemark\footnotemark
1023 \def\footnotemark{\ifx\protect\@typeset@protect
1024   \expandafter\FN@sf@@footnotemark
1025   \else
1026     \expandafter\FN@sf@gobble@optonly
1027   \fi
1028 }
1029 \edef\FN@sf@gobble@optonly{\noexpand\protect
1030   \expandafter\noexpand\csname FN@sf@gobble@optonly \endcsname}
1031 \expandafter\def\csname FN@sf@gobble@optonly \endcsname{%
1032   \@ifnextchar [%]
1033     \FN@sf@gobble@bracket
1034     {}%
1035 }
1036 \def\FN@sf@gobble@bracket[#1]{}
1037 \fi
1038 \newcommand\setfnsymbol[1]{%
1039   \@bspack
1040   \@ifundefined{FN@fnsymbol@#1}%
1041   {%
1042     \PackageError{footmisc}{Symbol style "#1" not known}%
1043     \@eha
1044   }{%
1045     \expandafter\let\expandafter\@fnsymbol\csname
1046       FN@fnsymbol@#1\endcsname
1047   }%
1048   \@espack
1049 }
1050 \let\FN@fnsymbol@lampoort\@fnsymbol
1051 \newif\if@tempwb
1052 \DeclareDocumentCommand\DefineFNSymbols {smO{text}m}{%
1053   \expandafter\ifx\csname FN@fnsymbol@#2\endcsname\relax
1054     \PackageInfo{footmisc}{Declaring symbol style #2}%
1055   \else
1056     \PackageWarning{footmisc}{Redeclaring symbol style #2}%

```

```

1057 \fi
1058 \toks@{}%
1059 \def\@tempb{\end}%
1060 \FN@build@symboldef#4\end
1061 \def\@tempc{math}%
1062 \def\@tempd{#3}%
1063 \expandafter\xdef\csname FN@fnsymbol@#2\endcsname##1{%
1064   \ifx\@tempc\@tempd
1065     \noexpand\ensuremath
1066   \else
1067     \noexpand\nfss@text
1068   \fi
1069   {%
1070     \noexpand\ifcase##1%
1071     \the\toks@
1072     \noexpand\else
1073     \IfBooleanTF#1{\noexpand\@ctrerr}%
1074     {\noexpand\FN@orange##1}%
1075     \noexpand\fi
1076   }%
1077 }%
1078 }
1079 \def\FN@build@symboldef#1{%
1080   \def\@tempa{#1}%
1081   \ifx\@tempa\@tempb
1082     \else
1083     \toks@\expandafter{\the\toks@\or#1}%
1084     \expandafter\FN@build@symboldef
1085   \fi
1086 }
1087 \DeclareDocumentCommand\DefineFNsymbolsTM {smm}{%
1088   \expandafter\ifx\csname FN@fnsymbol@#2\endcsname\relax
1089   \PackageInfo{footmisc}{Declaring symbol style #2}%
1090   \else
1091   \PackageWarning{footmisc}{Redeclaring symbol style #2}%
1092   \fi
1093   \toks@{}%
1094   \def\@tempb{\end}%
1095   \FN@build@symboldefTM#3\end\@null
1096   \expandafter\xdef\csname FN@fnsymbol@#2\endcsname##1{%
1097     \noexpand\ifcase##1%
1098     \the\toks@
1099     \noexpand\else
1100     \IfBooleanTF#1{\noexpand\@ctrerr}%
1101     {\noexpand\FN@orange##1}%
1102     \noexpand\fi
1103   }%
1104 }
1105 \def\FN@build@symboldefTM#1#2{%
1106   \def\@tempa{#1}%
1107   \ifx\@tempa\@tempb
1108     \else
1109     \toks@\expandafter{\the\toks@\or\TextOrMath{#1}{#2}}%
1110     \expandafter\FN@build@symboldefTM

```



```

1111 \fi
1112 }
1113 \def\FN@orange#1{%
1114 \ifFN@robust
1115 \@arabic#1%
1116 \@bsphack
1117 \PackageInfo{footmisc}{Footnote number \number#1 out of range}%
1118 \protect\@fnsymbol@orange
1119 \@esphack
1120 \else \@ctrerr \fi
1121 }
1122 \global\let\@diagnose@fnsymbol@orange\relax
1123 \AtEndDocument{\@diagnose@fnsymbol@orange}
1124 \def\@fnsymbol@orange{%
1125 \gdef\@diagnose@fnsymbol@orange{%
1126 \PackageWarningNoLine{footmisc}{Some footnote number(s)
1127 were out of range
1128 \MessageBreak
1129 see log for details%
1130 }%
1131 }%
1132 }
1133 \DefineFNSymbolsTM{bringhurst}{%
1134 \textasteriskcentered *%
1135 \textdagger \dagger
1136 \textdaggerdbl \ddagger
1137 \textsection \mathsection
1138 \textbardbl \||%
1139 \textparagraph \mathparagraph
1140 }%
1141 \DefineFNSymbolsTM{chicago}{%
1142 \textasteriskcentered *%
1143 \textdagger \dagger
1144 \textdaggerdbl \ddagger
1145 \textsection \mathsection
1146 \textbardbl \||%
1147 \#\#%
1148 }%
1149 \DefineFNSymbolsTM{wiley}{%
1150 \textasteriskcentered *%
1151 {\textasteriskcentered\textasteriskcentered}{**}%
1152 \textdagger \dagger
1153 \textdaggerdbl \ddagger
1154 \textsection \mathsection
1155 \textparagraph \mathparagraph
1156 \textbardbl \||%
1157 }%
1158 \DefineFNSymbolsTM{lampport-robust}{%
1159 \textasteriskcentered *%
1160 \textdagger \dagger
1161 \textdaggerdbl \ddagger
1162 \textsection \mathsection
1163 \textparagraph \mathparagraph
1164 \textbardbl \||%

```

```

1165   {\textasteriskcentered\textasteriskcentered}{**}%
1166   {\textdagger\textdagger}{\dagger\dagger}%
1167   {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
1168 }
1169 \DefineFNsymbolsTM*{\lamport*}{%
1170   \textasteriskcentered *%
1171   \textdagger    \dagger
1172   \textdaggerdbl \ddagger
1173   \textsection   \mathsection
1174   \textparagraph \mathparagraph
1175   \textbardbl    \|%
1176   {\textasteriskcentered\textasteriskcentered}{**}%
1177   {\textdagger\textdagger}{\dagger\dagger}%
1178   {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
1179   {\textsection\textsection}{\mathsection\mathsection}%
1180   {\textparagraph\textparagraph}{\mathparagraph\mathparagraph}%
1181   {\textasteriskcentered\textasteriskcentered\textasteriskcentered}{***}%
1182   {\textdagger\textdagger\textdagger}{\dagger\dagger\dagger}%
1183   {\textdaggerdbl\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger\ddagger}%
1184   {\textsection\textsection\textsection}%%
1185   {\mathsection\mathsection\mathsection}%
1186   {\textparagraph\textparagraph\textparagraph}%%
1187   {\mathparagraph\mathparagraph\mathparagraph}%
1188 }
1189 \setfnsymbol{\lamport*}
1190 \DefineFNsymbolsTM{\lamport*-robust}{%
1191   \textasteriskcentered *%
1192   \textdagger    \dagger
1193   \textdaggerdbl \ddagger
1194   \textsection   \mathsection
1195   \textparagraph \mathparagraph
1196   \textbardbl    \|%
1197   {\textasteriskcentered\textasteriskcentered}{**}%
1198   {\textdagger\textdagger}{\dagger\dagger}%
1199   {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
1200   {\textsection\textsection}{\mathsection\mathsection}%
1201   {\textparagraph\textparagraph}{\mathparagraph\mathparagraph}%
1202   {\textasteriskcentered\textasteriskcentered\textasteriskcentered}{***}%
1203   {\textdagger\textdagger\textdagger}{\dagger\dagger\dagger}%
1204   {\textdaggerdbl\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger\ddagger}%
1205   {\textsection\textsection\textsection}%%
1206   {\mathsection\mathsection\mathsection}%
1207   {\textparagraph\textparagraph\textparagraph}%%
1208   {\mathparagraph\mathparagraph\mathparagraph}%
1209 }
1210 \newcommand\mpfootnotemark{%
1211   \@ifnextchar [%]
1212     \@xmpfootnotemark
1213     {%
1214       \stepcounter\@mpfn
1215       \protected@xdef\@thefnmark{\thempfn}%
1216       \@footnotemark
1217     }%
1218 }

```

```

1219 \def\@xmpfootnotemark[#1]{%
1220 \begingroup
1221   \csname c@\@mpfn\endcsname #1\relax
1222   \unrestored@protected@xdef\@thefnmark{\thempfn}%
1223 \endgroup
1224 \@footnotemark
1225 }

1226 \endinput
1227 </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.

Symbols		cs commands:	
\#	1147	\cs_generate_variant:Nn	80, 81, 503, 517, 565
\	1138, 1146, 1156, 1164, 1175, 1196	\cs_gset_eq:NN	128, 152
A			
\addtocounter	894	\cs_if_eq:NNTF	400, 405
\AddToHook	319, 398, 453, 455, 461, 463, 483, 685, 817, 821, 853, 907, 915, 985, 986	\cs_new:Npn	388, 390, 392, 394, 396
\AddToHookWithArguments	482	\cs_new_eq:NN	197, 362
\advance	661, 735, 933	\cs_new_protected:Npn	82, 95, 117, 118, 119, 131, 155, 182, 211, 332, 364, 481, 490, 496, 504, 553, 560
\AssignSocketPlug	209, 292, 304, 307, 316, 353, 552, 597, 598, 612, 621, 689, 690, 691, 692, 764, 790, 792, 796, 798, 802, 804, 833, 834, 835, 836	\cs_set:Npn	385
\AtBeginDocument	326, 465, 470	\cs_set_eq:NN	199, 321, 368, 371, 374, 377, 379, 402, 425, 427, 457, 458, 459
\AtEndDocument	1123	\cs_set_protected:Npn	485
\AtEndOfPackage	670	\csname	20, 888, 1015, 1016, 1030, 1031, 1045, 1053, 1063, 1088, 1096, 1221
B			
\baselineskip	731, 863	\CurrentOption	674, 676, 681, 683
\baselinestretch	463, 817, 821	D	
\begingroup	438, 1220	\dagger	1135, 1143, 1152, 1160, 1166, 1171, 1177, 1182, 1192, 1198, 1203
\bgroup	921	\ddagger	1136, 1144, 1153, 1161, 1167, 1172, 1178, 1183, 1193, 1199, 1204
bool commands:		\DebugFNotesOff	7, <u>117</u>
\bool_gset_false:N	118	\DebugFNotesOn	7, <u>117</u>
\bool_gset_true:N	117	\DeclareCurrentRelease	645, 648
\bool_if:NTF	121, 133, 522, 538, 580	\DeclareDocumentCommand	417, 422, 431, 436, 1052, 1087
\bool_new:N	69, 76, 116	\DeclareOption	665, 667, 673, 680, 696, 703, 710, 715, 719, 720, 721, 724, 728, 733, 737, 756, 758
\bool_set_false:N	207, 441, 474, 477	\DeclareRelease	644, 647
\bool_set_true:N	70, 77, 444	\def	12, 18, 29, 30, 34, 242, 385, 464, 659, 660, 741, 742, 767, 860, 867, 879, 883, 888, 892, 896, 920, 959, 988, 992, 1008, 1016, 1021, 1023, 1031,
\box	302		
C			
\columnwidth	16, 238, 301, 882, 883, 932		
\cs	247, 248		

<code>\linewidth</code>	932, 933, 934		
<code>\llap</code>	5, 7, 343, 348, 940, 966, 968		
<code>\long</code>	12, 385, 860, 920, 959		
<code>\ltxlabfootnotedate</code>	4		
<code>\ltxlabfootnoteversion</code>	4		
			M
<code>\makebox</code>	23, 373, 392, 393		
<code>\MakeLinkTarget</code>	603, 604		
<code>\MakePerPage</code>	36, 887		
<code>\marginpar</code>	291		
<code>\marginparwidth</code>	686		
<code>\mathparagraph</code>	1139, 1155, 1163, 1174, 1180, 1187, 1195, 1201, 1208		
<code>\mathsection</code> ..	1137, 1145, 1154, 1162, 1173, 1179, 1185, 1194, 1200, 1206		
<code>\maxdimen</code> ...	5, 7, 324, 328, 345, 678, 904		
<code>\MessageBreak</code>	1128		
<code>mp (plug)</code>	5		
<code>\mpfootnotemark</code>	1210		
<code>\mpfootnoterule</code>	740, 744		
<code>\multfootsep</code>	983, 999		
<code>\multiplefootnotemark</code>	981, 989, 990, 993		
			N
<code>\NeedsTeXFormat</code>	643, 654		
<code>\newbox</code>	844, 845		
<code>\newcommand</code>	357, 731, 732, 1038, 1210		
<code>\newdimen</code>	323, 849		
<code>\NewHook</code> .	108, 109, 110, 112, 113, 114, 115		
<code>\newif</code>	666, 672, 701, 706, 709, 727, 755, 757, 811, 1051		
<code>\NewMirroredHookPair</code>	107, 111		
<code>\newskip</code>	846		
<code>\NewSocket</code>	198, 202, 289, 293, 305, 308, 317, 318, 337, 354, 355, 356		
<code>\NewSocketPlug</code>	203, 290, 291, 294, 295, 306, 309, 310, 338, 446, 518, 568, 592, 599, 613, 761		
<code>\newtoks</code>	655		
<code>\nobreak</code>	4, 172, 915		
<code>\noexpand</code>	774, 909, 1014, 1015, 1029, 1030, 1065, 1067, 1070, 1072, 1073, 1074, 1075, 1097, 1099, 1100, 1101, 1102		
<code>\noindent</code>	333, 359, 961		
<code>noop (plug)</code>	5, 5, 5, 6		
<code>\normalbaselineskip</code>	856		
<code>\normalfont</code>	34		
<code>\number</code>	1117		
			O
<code>\or</code>	788, 794, 800, 1083, 1109		
			P
<code>\PackageError</code>	674, 681, 1042		
<code>\PackageInfo</code>	1054, 1089, 1117		
<code>\PackageWarning</code>	1056, 1091		
<code>\PackageWarningNoLine</code>	1126		
<code>\PackageWarningNoline</code>	909		
<code>\pagefootnoterule</code>	739, 747		
<code>\par</code>	24, 279, 950		
<code>para (plug)</code>	5, 5, 5		
<code>\parindent</code>	6, 241, 358, 948, 960		
<code>\parshape</code>	934		
<code>\parskip</code>	947		
<code>\penalty</code>	313		
Plugs:			
<code>default</code>	5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 21		
<code>FEMark</code>	518		
<code>FENote</code>	568		
<code>FENoteLbl</code>	599		
<code>FENotetext</code>	613		
<code>identity</code>	6, 16, 17, 21		
<code>mp</code>	5		
<code>noop</code>	5, 5, 5, 6		
<code>para</code>	5, 5, 5		
<code>side</code>	5		
prg commands:			
<code>\prg_do_nothing:</code>	128, 152, 425		
<code>\ProcessOptions</code>	759		
prop commands:			
<code>\prop_gput:Nnn</code>	498		
<code>\prop_gremove:Nn</code>	513		
<code>\prop_map_inline:Nn</code>	508		
<code>\prop_new:N</code>	65, 492		
<code>\prop_set_eq:NN</code>	507		
property commands:			
<code>\property_new:nnnn</code>	480		
<code>\property_record:nn</code>	487		
<code>\property_ref:nn</code>	532, 534		
<code>\protect</code>	1008, 1014, 1023, 1029, 1118		
<code>\providecommand</code>	38, 467, 468, 644, 645, 656, 981, 983		
<code>\ProvidesFile</code>	3		
<code>\ProvidesPackage</code>	649		
			R
<code>\RaggedRight</code>	699		
<code>\raggedright</code>	698		
<code>\RawNoindent</code>	871		
<code>\RawParEnd</code>	865		
<code>\ref</code>	24, 439		
<code>\relax</code>	190, 301, 313, 660, 661, 722, 725, 735, 742, 768, 776, 781, 783, 784, 786, 910, 947, 948, 990, 993, 1000, 1004, 1053, 1088, 1122, 1221		
<code>\renewcommand</code>	665, 668, 698, 699, 734		

<code>\@makefnmark</code>	<code>\FN@mf@check</code>	985, 992
... 3-5, 7, 10, 16, 22, 23, 34, 178,	<code>\FN@mf@prepare</code>	986, 988, 1004
340, 343, 346, 348, 360, 376, 394,	<code>\FN@multiplefootnotefalse</code>	757
395, 925, 927, 942, 944, 963, 966, 968	<code>\FN@multiplefootnotetrue</code>	758
<code>\@makefntext</code>	<code>\FN@orange</code>	1074, 1101, 1113
3, 5, 6, 10, 11,	<code>\FN@parafalse</code>	672
20-23, 28, 22, 241, 255, 294, 297,	<code>\FN@paratrue</code>	677
366, 385, 400, 402, 405, 458, 953, 959	<code>\FN@perpagefalse</code>	702
<code>\@makefntext@processX</code>	<code>\FN@perpagetrue</code>	704
..... 389, 391, 393, 395, 396	<code>\FN@removeboxes</code>	864, 867, 868
<code>\@mpfn</code>	<code>\FN@robustfalse</code>	666
29, 419, 424, 1214, 1221	<code>\FN@robusttrue</code>	669
<code>\@mpfootins</code>	<code>\FN@setfootnoteparawidth</code> 861, 879, 883	
13, 14, 448, 449	<code>\FN@setspacefalse</code>	826
<code>\@mpfootnotetext</code>	<code>\FN@setspacetrue</code>	814
9, 12	<code>\FN@sf@@footnote</code>	1007, 1009
<code>\@mplistdepth</code>	<code>\FN@sf@@footnotemark</code> ...	1022, 1024
743	<code>\FN@sf@gobble@bracket</code> ...	1033, 1036
<code>\@ne</code>	<code>\FN@sf@gobble@opt</code>	1011, 1014
711, 880, 934	<code>\FN@sf@gobble@optonly</code> ...	1026, 1029
<code>\@null</code>	<code>\FN@sf@gobble@twobracket</code> .	1018, 1021
1095	<code>\FN@stablefootnotefalse</code>	755
<code>\@outputbox@append</code>	<code>\FN@stablefootnotetrue</code>	756
769, 774	<code>\FN@tempboxa</code>	297,
<code>\@outputbox@reinsertskip</code>	298, 299, 300, 302, 843, 867, 868, 869	
..... 773, 776, 781, 784	<code>\FN@tempboxb</code>	844
<code>\@outputbox@removebskip</code> ...	<code>\FN@tempboxc</code>	845
767, 783	<code>\FN@temptoken</code>	655
<code>\@parboxrestore</code>	<code>\footmisc@hang@makefntext</code>	
17, 239, 862 405, 920, 953	
<code>\@plus</code>	<code>\hb@xt@</code>	22, 23, 340, 348,
735	360, 370, 390, 391, 925, 942, 963, 968	
<code>\@setpar</code>	<code>\hyper@linkend</code>	468, 545
936	<code>\hyper@linkstart</code>	467, 541
<code>\@stpelt</code>	<code>\hyper@nopatch@footnote</code>	464
888	<code>\if@tempswb</code>	1051
<code>\@tempa</code>	<code>\ifFN@abovefloats</code> ..	709, 789, 795, 801
1080, 1081, 1106, 1107	<code>\ifFN@fixskip</code>	706, 766
<code>\@tempb</code>	<code>\ifFN@hangfoot</code>	727, 919
1059, 1081, 1094, 1107	<code>\ifFN@multiplefootnote</code>	757, 980
<code>\@tempboxa</code>	<code>\ifFN@para</code> .	672, 680, 760, 831, 842, 900
843, 923, 930	<code>\ifFN@perpage</code>	701, 885
<code>\@tempc</code>	<code>\ifFN@robust</code>	666, 1114
1061, 1064	<code>\ifFN@setspace</code>	811
<code>\@tempcnta</code>	<code>\ifFN@stablefootnote</code>	755, 1006
660, 661, 662	<code>\l@advance@macro</code>	659, 664
<code>\@tempd</code>	<code>\m@m@singlespace</code>	817, 818
1062, 1064	<code>\m@ne</code>	888, 894
<code>\@tempskipa</code>	<code>\new@std@class@makecol</code>	454, 459
770, 771, 772, 774	<code>\nfss@text</code>	1067
<code>\@textbottom</code>	<code>\old@std@class@makefntext</code>	
768 357, 400, 458	
<code>\@textsuperscript</code>	<code>\p@</code>	735, 741
34	<code>\p@footnote</code>	243
<code>\@thefnmark</code>		
8-10, 12,		
14, 24, 27, 28, 30, 20, 34, 86, 91,		
98, 103, 243, 439, 524, 582, 1215, 1222		
<code>\@typeset@protect</code>		
1008, 1023		
<code>\@x@sf</code>		
164, 190, 995, 1000		
<code>\@xmpfootnotemark</code>		
1212, 1219		
<code>\c@footnote</code>		
668		
<code>\color@begingroup</code>		
21, 253		
<code>\color@endgroup</code>		
25, 280		
<code>\doublecol@number</code>		
880		
<code>\FN@abovefloatsfalse</code>		
712, 720		
<code>\FN@abovefloatstrue</code> ...		
709, 717, 719		
<code>\FN@baselinestretch</code>		
818, 822		
<code>\FN@bottomcases</code> ...		
708, 711, 716, 786		
<code>\FN@build@symboldef</code> .		
1060, 1079, 1084		
<code>\FN@build@symboldefTM</code>		
1095, 1105, 1110		
<code>\FN@fixskipfalse</code>		
706		
<code>\FN@fixskiptrue</code> ...		
713, 717, 719, 720		
<code>\FN@fnsymbol@lampport</code>		
1050		
<code>\FN@hangfootfalse</code>		
727		
<code>\FN@hangfoottrue</code>		
729		
<code>\FN@makefootnoteparagraph</code> ..		
762, 860		

