

The `latex-lab-footnotes` code^{*}

Frank Mittelbach, L^AT_EX Project

January 29, 2026

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	9
3.3	The algorithmus to connect marks and notes	9
3.3.1	<code>\footref</code>	10
3.3.2	<code>\footnotemark</code> after <code>\footnotetext</code>	10
3.4	Links	10
3.5	Implementation details regarding tagging	11
3.6	Handling the mark	11
3.7	Handling the <code>footnotetext</code>	11
3.8	Footnotes in minipages	11
4	TODOs	11

*

5	The Implementation	11
5.1	File declaration	12
5.2	code not fully handled yet	12
5.3	Temporary variables	13
5.4	Public variables	13
5.5	Internal variables	14
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	18
5.12	The new <code>\@makefnmark</code> command	21
5.12.1	Making documents use the new <code>\@makefnmark</code>	22
5.13	Document-level commands	24
5.14	Firstaid for packages and classes	25
5.15	Kernel patches	25
5.15.1	<code>memoir</code>	25
5.15.2	<code>setspace</code>	25
5.15.3	<code>hyperref</code>	26
5.16	Tagging and hyperlink code	26
5.16.1	Rolemap for structure tags	26
5.16.2	Extending the label system	27
5.16.3	Storing and retrieving reference data	27
5.16.4	Enabling tagging and links for the mark command	28
5.16.5	The footnote text	30
6	Reimplementing the <code>footmisc</code> package	32
	Index	44

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`, `sclttr2`, 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).

fntext/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

fntext/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**.

fntext

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.

fntext/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.

fntext/begin & fntext/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.

fntext/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 cname 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: Their name and/or function will probably change as they currently mix tagging with the link support.

TODO: review this sockets

tagsupport/fnmark (2 argument)

The socket is used in `\@footnotemark/\fnote_footnotemark:` and takes `\@makefnmark` or — if link support is wanted — `\socket_use:n{fnmark/link}{\@makefnmark}` as second argument. It prints the mark in the text and surrounds it with a tagging structure.

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.

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 (2 argument)

This socket is used around the mark in the footnote text. It adds tagging support.

tagsupport/fntext/text (2 argument)

This socket handles mc-chunks around the text of the footnote. It takes as second argument the text.

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.

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`.

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

Footnote links should work also if tagging is disabled, either locally (e.g. in an artifact) or if tagging is deactivated globally. They should therefore use their own system to store the relations between mark and note, even if this duplicates some code.

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 a combination of their own structure number (for the tagging) and a unique id (for the link) 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 keys 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`

<code>\g_fnote_id_int</code>	This integer is used as unique identifier to store data for a footnote. It should be increase when a footnote mark or a footnote text is created.
------------------------------	---

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

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.

<code>\fnote_mark_struct_gput:nnn</code>	<code>\fnote_mark_struct_gput:nnn{<mark>}{<class name>}</code>
<code>\fnote_mark_id_gput:nnn</code>	<code>\fnote_mark_id_gput:nnn{<mark>}{<class name>}</code>

These commands store either the current structure number or the current id as key and the `<mark>` as value in the property list associated with the `<class name>`.

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.

<code>\fnote_mark_gpop_all:nnnN</code>	<code>\fnote_mark_gpop_all:nnN{⟨id or struct⟩}{⟨mark⟩}{⟨class name⟩}{⟨sequence⟩}</code>
--	---

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` 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 keys containing the structure number/id combination detected for the `/Ref` are also used for links. For links the second part is used as this id is increased also if tagging is deactivated.

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 `\@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      [\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 %       \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.)

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.)

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.)

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:
```

```

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

```

<code>\fnote_set_fnmark:nn</code>	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  }

```

5.8 Hooks

<code>fnmark/before</code>	(hook)		Hooks in the footnote mark command.
<code>fnmark/after</code>	(hook)	107	<code>\NewMirroredHookPair{fnmark/before}{fnmark/after}</code>
<code>fnmark</code>	(hook)	108	<code>\NewHook{fnmark}</code>
<code>fnmark/begin</code>	(hook)	109	<code>\NewHook{fnmark/begin}</code>
<code>fnmark/end</code>	(hook)	110	<code>\NewHook{fnmark/end}</code>
<code>fntext/before</code>	(hook)		Hooks in the footnotetext command:
<code>fntext/after</code>	(hook)	111	<code>\NewMirroredHookPair{fntext/before}{fntext/after}</code>
<code>fntext</code>	(hook)	112	<code>\NewHook{fntext}</code>
<code>fntext/begin</code>	(hook)	113	<code>\NewHook{fntext/para}</code>
<code>fntext/end</code>	(hook)	114	<code>\NewHook{fntext/begin}</code>
<code>fntext/para</code>	(hook)	115	<code>\NewHook{fntext/end}</code>

5.9 Debugging code

```

116 \bool_new:N          \g_fnote_debug_bool

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.)

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 {tagsupport/fnmark}
144     \socket_log:n {tagsupport/fntext/begin}
145     \socket_log:n {tagsupport/fntext/end}
146     \socket_log:n {tagsupport/fntext/mark}
147     \socket_log:n {tagsupport/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}
153     \hook_log:n {fntext/end}
154     \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   % chextris --- 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. For link support it should surround the argument with the link socket.

```

178   \tag_socket_use:nnn {fnmark}{\{ \socket_use:n{fnmark/link}{\@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:.`)

`__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 (without tagging) it uses the transparent plug.

```

198 \NewTaggingSocket{fnmark}{2}

```

`\@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.)
```

5.11 The new `\@footnotetext` command

`\fnote_footnotetext:n`

```
200 \cs_new_protected:Npn \fnote_footnotetext:n #1 {
201   \__fnote_debug_footnotetext:
202   %-----
203   % ./linguex/linguex.sty
204   \hook_use:n {fntext/before}
205   %-----
```

Execute a kernel socket for tagging.

```
206   \tag_socket_use:n {fntext/begin}
207   \socket_use:nn {fntext/process}
208   {
209   %-----
210   % resetting baselinestretch ... (could be done further down)
211   % ./uafthesis/uafthesis.cls
212   % ./setspace/setspace.sty
213   % ./footmisc/footmisc.sty (normal)
214   \hook_use:n {fntext}
215   %-----
216   \reset@font
217   \footnotesize
218   %-----
219   % some classes use a different font size, e.g.,
220   % ./nrc/nrc1.cls ./nrc/nrc2.cls
221   % but those could be done in fntext/para instead
222   %-----
```

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

```
223   \interlinepenalty\interfootnotelinepenalty
224   \splittopskip\footnotesep
225   \splitmaxdepth \dp\strutbox
226   \floatingpenalty \@MM
227   \hsize\columnwidth
228   \@parboxrestore
229   \UseTaggingSocket{para/restore}
230   \parindent 1em % typical default used in \@makefntext moved up here
231   \def\@currentcounter{footnote}
232   \protected@edef \@currentlabel { \p@footnote \@thefnmark }
233   %-----
234   % for altering para parameters ...
235   % code for resphilosophica came earlier but it could go here.
236   % Has the advantage that one can also overwrite \cs{@currentcounter}
237   % and \cs{@currentlabel} is that is necessary.
238   %
```

```

239 % ./resphilosophica/resphilosophica.cls
240 \hook_use:n {fntext/para}
241 %-----
242 \color@begingroup
243 %-----
244 % fnpara wants to replace \@makefntext{...} and para and side
245 % option of footmisc etc too ...
246 % so we make this a socket, because only one action can be active:
247 %-----
248 \socket_use:nn {fntext/make}
249 {
250 %-----
251 % ./resphilosophica/resphilosophica.cls
252 %-----
253 \socket_use:n {fntext/begin}%
254 %-----
255 % bibarts
256 % fnbreak.sty
257 \hook_use:n {fntext/begin}
258 %-----
259 \ignorespaces
260 #1
261 %-----
262 % bibarts
263 % fnbreak.sty
264 \hook_use:n {fntext/end}
265 %-----

```

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

```

266 \socket_use:n {fntext/end}
267 }
268 \par
269 \color@endgroup
270 }
271 %-----

```

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

```

272 \tag_socket_use:n{fntext/end}
273 %-----
274 % ./linguex/linguex.sty
275 \hook_use:n {fntext/after}
276 %-----
277 }

```

(End of definition for \fnote_footnotetext:n.)

fntext/process (socket)

```

278 \NewSocket {fntext/process}{1}
279 \NewSocketPlug{fntext/process}{default}{ \insert\footins {#1} }
280 \NewSocketPlug{fntext/process}{side} { \marginpar {#1} }
281 \AssignSocketPlug{fntext/process}{default}

```

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

```

282 \NewSocket {fntext/make}{1}
283 \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 \TeX book algorithm in appendix D).

```

284 \NewSocketPlug{fntext/make}{para}
285 {
286   \setbox\FN@tempboxa\hbox{\@makefntext{#1}}%
287   \dp\FN@tempboxa\z@
288   \ht\FN@tempboxa
289   \dimexpr\wd\FN@tempboxa *%
290           \footnotebaselineskip /\columnwidth\relax
291   \box\FN@tempboxa
292 }
293 \AssignSocketPlug{fntext/make}{default}

```

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

```

294 \NewSocket      {fntext/begin}{0}
295 \NewSocketPlug{fntext/begin}{default}{ \rule\z@\footnotesep }
296 \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.

```

297 \NewSocket      {fntext/end}{0}
298 \NewSocketPlug{fntext/end}{default}{ \@finalstrut\strutbox }

```

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

```

299 \NewSocketPlug{fntext/end}{para}
300 {%
301     \strut
302     \penalty-10\relax
303     \hskip\footglue
304 }
305 \AssignSocketPlug{fntext/end}{default}

```

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

`tagssupport/fntext/end (socket)`

```

306 \NewTaggingSocket{fntext/begin}{0}
307 \NewTaggingSocket{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).

```

308 \AddToHook{begindocument}
309 {
310   \cs_set_eq:NN \@footnotetext \fnote_footnotetext:n
311 }

```

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.

```

312 \newdimen\footnotemargin
313 \footnotemargin\maxdimen          % no value given
314
315 \AtBeginDocument
316 {
317   \ifdim \footnotemargin=\maxdimen
318     \setlength\footnotemargin{1.8em}
319   \fi
320 }
```

\fnote_makefntext:n

```

321 \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.

```

322   \noindent
```

after leaving the vmode we can set the link targets.

```

323   \socket_use:n{fntext/mark/link}
324   \tag_socket_use:nnn {fntext/mark} {} { \socket_use:n {fntext/mark} }
325   \tag_socket_use:nnn {fntext/text} {} { \socket_use:nn {fntext/text}{#1} }
326 }
```

(End of definition for \fnote_makefntext:n.)

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

```

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

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

```

328 \NewSocketPlug{fntext/mark}{default}{
329   \ifdim\footnotemargin>\z@
330     \hb@xt@ \footnotemargin{\hss\@makefnmark}
331   \else
332     \ifdim\footnotemargin=\z@
333       \llap{\@makefnmark}
334     \else
335       \ifdim\footnotemargin=-\maxdimen
336         \@makefnmark
337       \else
338         \llap{\hb@xt@ -\footnotemargin{\@makefnmark\hss}}
339       \fi
340     \fi
341   \fi
342 }
343 \AssignSocketPlug{fntext/mark}{default}
```

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

```

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

`tagssupport/fntext/mark` (*socket*) Not a public socket but reserved for tagging. By default they contain the `transparent`
`tagssupport/fntext/text` (*socket*) and are reassigned if tagging is active.

```
345 \NewTaggingSocket{fntext/mark}{2}
346 \NewTaggingSocket{fntext/text}{2}
```

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
```

```
347 \newcommand\old@std@class@makefntext[1]{%
348   \parindent 1em%
349   \noindent
350   \hb@xt@1.8em{\hss\@makefnmark}\#1}
```

(End of definition for `\old@std@class@makefntext`.)

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 \UseTaggingSocket{fntext/mark}{}\{\hbox BBB { CCC }\} DDD
\UseTaggingSocket{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).

```
351 \tl_new:N \l__fnote_patch_tl
352 \cs_new_eq:NN \__fnote_tmp:w \ERROR
353
354 \cs_new_protected:Npn \__fnote_patch:
355 {
356   \tl_set:Nn \l__fnote_patch_tl { \@makefntext { \UseTaggingSocket{fntext/text}{}\{##1\} } }
357   \tl_if_in:NnTF \l__fnote_patch_tl { \hbox }
358   { \cs_set_eq:NN \__fnote_tmp:w \__fnote_patch_hbox:w }
359   {
360     \tl_if_in:NnTF \l__fnote_patch_tl { \hb@xt@ }
361     { \cs_set_eq:NN \__fnote_tmp:w \__fnote_patch_hb@xt:w }
362     {
```

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

```

363         \tl_if_in:NnTF \l__fnote_patch_tl { \makebox }
364         { \cs_set_eq:NN \__fnote_tmp:w \__fnote_patch_makebox:w }
365         {
366             \tl_if_in:NnTF \l__fnote_patch_tl { \@makefnmark }
367             { \cs_set_eq:NN \__fnote_tmp:w \__fnote_patch_@makefnmark:w }
368             { \ERROR
369               \cs_set_eq:NN \__fnote_tmp:w \exp_stop_f: }
370         }
371     }
372 }
373 \tl_set:Nf \l__fnote_patch_tl
374 { \exp_after:wN \__fnote_tmp:w \l__fnote_patch_tl }
375 \cs_set:Npn \__fnote_tmp:w { \long \def \@makefntext ###1 }
376 \exp_after:wN \__fnote_tmp:w \exp_after:wN { \l__fnote_patch_tl }
377 }

```

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

```

AAA \@makefntext@processX { \hbox BBB }

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

```

Same for the other cases.

```

380 \cs_new:Npn \__fnote_patch_hb@xt@:w #1 \hb@xt@ #2 #
381 { \exp_stop_f: #1 \@makefntext@processX { \hb@xt@ #2 } }

382 \cs_new:Npn \__fnote_patch_makebox:w #1 \makebox #2 #
383 { \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.

```

384 \cs_new:Npn \__fnote_patch_@makefnmark:w #1 \@makefnmark
385 { \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:

```

386 \cs_new:Npn \@makefntext@processX #1#2
387 {
388     \socket_use:n{fntext/mark/link}
389     \UseTaggingSocket{fntext/mark}{#1}{#2}
390 }

```

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.

```

391 \AddToHook{begindocument}
392 {
393     \cs_if_eq:NNF \@makefntext \fnote_makefntext:n
394     {
395         \cs_if_eq:NNTF \@makefntext \old@std@class@makefntext
396         {

```

```

397         \cs_set_eq:NN \@makefntext \fnote_makefntext:n
398     }
399     {

```

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

```

400         \cs_if_eq:NNF \@makefntext \footmisc@hang@makefntext
401             { \__fnote_patch: }
402     }
403 }
404 }
405
406
407 % possibly add the following to check for multiple \hbox in
408 % the definition:
409 %
410 % \seq_set_split:NnV \l__fnote_patch_seq { \hbox } \l__fnote_patch_tl
411 % \int_compare:nT { \seq_count:N \l__fnote_patch_seq } > 2 \ERROR
412 %

```

5.13 Document-level commands

`\footnotetext`

```

413 \DeclareDocumentCommand\footnotetext {o+m}
414 {
415     \fnote_set_fnmark:nn {#1} \@mpfn
416     \@footnotetext {#2}
417 }

```

(End of definition for \footnotetext.)

`\footnote`

```

418 \DeclareDocumentCommand\footnote {o+m}
419 {
420     \fnote_step_fnmark:nn {#1} \@mpfn
421     \cs_set_eq:NN \__fnote_footnotemark_finish: \prg_do_nothing:
422     \@footnotemark
423     \cs_set_eq:NN \__fnote_footnotemark_finish: \__fnote_footnotemark_default_finish:
424     \@footnotetext {#2}
425     \__fnote_footnotemark_finish:
426 }

```

(End of definition for \footnote.)

`\footnotemark`

```

427 \DeclareDocumentCommand\footnotemark {o}
428 {
429     \fnote_step_fnmark:nn {#1} { footnote }
430     \@footnotemark
431 }

```

(End of definition for \footnotemark.)

`\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`.

```

432 \DeclareDocumentCommand\footref {m}
433 {
434   \beginngroup
435     \unrestored@protected@xdef\@thefnmark{\ref*{#1}}%
436   \endgroup
437   \bool_set_false:N \l__fnote_autodetect_bool
438   \tl_set:Nn \l__fnote_currentlabel_tl {#1}
439   \@footnotemark
440   \bool_set_true:N \l__fnote_autodetect_bool
441 }

```

(End of definition for `\footref`.)

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.

```

442 \NewSocketPlug{fntext/process}{mp}
443 {
444   \global\setbox\@mpfootins\vbox{%
445     \unvbox\@mpfootins
446     #1
447   }
448 }

```

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.

```

449 \AddToHook{class/memoir/before}
450 { \let\new@std@class@makecol\@makecol }
451 \AddToHook{class/memoir/after}
452 {
453   \cs_set_eq:NN \@footnotemark \fnote_footnotemark:
454   \cs_set_eq:NN \@makefntext\old@std@class@makefntext
455   \cs_set_eq:NN \@makecol\new@std@class@makecol
456 }

```

5.15.2 setspace

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

```

457 \AddToHook{package/setspace/after}
458 {\let \@footnotetext \fnote_footnotetext:n
459   \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.

```
460 \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.

```
461 \AtBeginDocument
462 {
463   \providecommand\hyper@linkstart{\@gobbletwo}
464   \providecommand\hyper@linkend{\@empty}
465 }
```

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).

```
466 \AtBeginDocument
467 {
468   \ifpackageloaded{hyperref}
469   {
470     \AssignSocketPlug{fnmark/link}{link}
471     \AssignSocketPlug{fntext/mark/link}{link}
472     \legacy_if:nF{Hy@hyperfootnotes}{\bool_set_false:N \l_fnote_link_bool}
473   }
474   {
475     \bool_set_false:N \l_fnote_link_bool
476   }
477 }
```

5.16 Tagging and hyperlink code

`\g_fnote_id_int`

For links we need an unique id to identify footnote marks and footnotes. It is increased both for the mark in the text and the note below (similar to the structure numbers). We make it public to allow the implementation of other footnote commands, see the test-note-setup test. The int is increased in the fntext/before hook, so that the following tagging socket can pick it up. TODO: check if the hook is the right place or if this should be static code.

```
478 \int_new:N \g_fnote_id_int
479 \AddToHook{fntext/before}{\int_gincr:N \g_fnote_id_int}
```

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 and id of the note. We use the in-built label hook to record properties. We define two suitable properties: the one stores the structure number as stored in `\l__fnote_currentstruct_tl`, the other the id.

```
480 \property_new:nnnn {fnote/struct}{now}{1}{\l__fnote_currentstruct_tl}
481 \property_new:nnnn {fnote/id}{now}{1}{\int_use:N\g_fnote_id_int}
```

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

```
\__fnote_label_hook:e
```

```
482 \cs_new_protected:Npn \__fnote_label_hook:e #1 {}
483 \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 and id too. The name of label is provided as argument in the label hook.

```
484 \AddToHook{fntext/begin}
485 {
486   \cs_set_protected:Npn \__fnote_label_hook:e #1
487   {
488     \property_record:ee {\__fnote/#1} {fnote/struct,fnote/id}
489   }
490 }
```

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 set up every footnote command as its own type!

```
491 \cs_new_protected:Npn \fnote_class_new:nn #1 #2 % #1 name, #2 options
492 {
493   \prop_new:c { g__fnote_currentmarks_struct_#1_prop }
494   \prop_new:c { g__fnote_currentmarks_id_#1_prop }
495 }
496
497 \fnote_class_new:nn {default}{}

```

(End of definition for `\fnote_class_new:nn`. This function is documented on page 9.)

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

```
498 \cs_new_protected:Npn \fnote_mark_struct_gput:nn #1 #2 % #1 the representation of the mark,
499 {
500   \prop_gput:cen { g__fnote_currentmarks_struct_#2_prop }
501   { \tag_get:n{struct_num} }
502   { #1 }

```

```

503 }
504 \cs_new_protected:Npn \fnote_mark_id_gput:nn #1 #2 % #1 the representation of the mark, #2 t
505 {
506   \prop_gput:cen { g__fnote_currentmarks_id_#2 _prop }
507   { \int_use:N\g_fnote_id_int }
508   { #1 }
509 }
510 \cs_generate_variant:Nn \fnote_mark_struct_gput:nn {no,oo}
511 \cs_generate_variant:Nn \fnote_mark_id_gput:nn {no,oo}

```

(End of definition for \fnote_mark_id_gput:nn and \fnote_mark_struct_gput:nn. These functions are documented on page ??.)

\fnote_mark_gpop_all:nnnN This command takes as first argument either id or struct, then representation of the mark (e.g. the content of \@thefnmark), the class (typically `default` should work) and a sequence into which every key in the property is stored that has the same value as the mark. The sequence is cleared first.

```

512 \cs_new_protected:Npn \fnote_mark_gpop_all:nnnN #1 #2 #3 #4
513 {
514   \seq_clear:N #4
515   \prop_set_eq:Nc \l__fnote_tmpa_prop { g__fnote_currentmarks_#1_#3_prop }
516   \prop_map_inline:Nn \l__fnote_tmpa_prop
517   {
518     \tl_if_eq:nnT {#2} { ##2 }
519     {

```

store the key (struct num or id) in the seq

```

520       \seq_put_right:Nn #4 { ##1 }

```

remove entry as used from the global prop

```

521       \prop_gremove:cn { g__fnote_currentmarks_#1_#3_prop } {##1}
522     }
523   }
524 }
525 \cs_generate_variant:Nn \fnote_mark_gpop_all:nnnN {nooN}

```

(End of definition for \fnote_mark_gpop_all:nnnN. This function is documented on page 10.)

5.16.4 Enabling tagging and links for the mark command

fnmark/link (*socket*) A socket that contains the code to surround the mark in the text with a link. It should be used in the second argument of the tagging socket as we normally want the link to be inside the Lbl structure.

```

526 \NewSocket{fnmark/link}{1}

```

link (*plug*) To handle the link around the mark we define a plug for the socket fnmark/link

```

527 \NewSocketPlug{fnmark/link}{link}
528 {
529   \int_gincr:N\g_fnote_id_int
530   \bool_if:NTF \l__fnote_autodetect_bool
531   {
532     \fnote_mark_id_gput:oo {\@thefnmark}{\l_fnote_type_tl}
533     \tl_set:Ne \l__fnote_linktarget_tl {footnote*.\int_use:N\g_fnote_id_int}
534   }
535   {

```

```

536         \tl_set:Nc \l__fnote_linktarget_tl
537         {footnote*.\property_ref:ee {__fnote/\l__fnote_currentlabel_tl} {fnote/id}}
538     }
539     \bool_if:NTF \l_fnote_link_bool
540     {
541         \exp_args:No
542         \hyper@linkstart
543         { \l_fnote_link_type_tl }
544         { \l__fnote_linktarget_tl }
545         #1
546         \hyper@linkend
547     }
548     { #1 }
549 }

```

FEMark (*plug*)

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

```

550 \NewTaggingSocketPlug{fnmark}{FEMark}
551 {

```

End an open mc and start the structure.

```

552     \tag_mc_end_push:
553     \tag_struct_begin:n { tag=footnotemark }

```

The associated note is either auto detected or given by the user. If there is no autode-
tecting 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
/Ref of the FEnote. Both must be delayed as we don't know if the objects of the FEnote
and the mark have already been created.

```

554     \bool_if:NF \l__fnote_autodetect_bool
555     {
556         \hook_gput_code:nne {tagpdf/finish/before} {tagpdf/footnote}
557         {
558             \exp_not:N\fnote_gput_refs:ee
559             { \tag_get:n{struct_num} }
560             { \property_ref:ee{ __fnote/\l__fnote_currentlabel_tl } {fnote/struct} }
561         }
562     }

```

And now the actual content

```

563     \tag_mc_begin:n{tag=Lbl}

```

For the auto detecting we store the struct num and `\@thefnmark` inside a prop TODO:
this should be usable for other footnote types which means the name of the prop shouldn't
be fix.

```

564     \bool_if:NT \l__fnote_autodetect_bool
565     {
566         \fnote_mark_struct_gput:oo {\@thefnmark}{\l_fnote_type_tl}
567     }

```

The target in the footnote will use the current id. This part is needed even if tagging is
deactivated.

```

568     #2
569     \tag_mc_end:

```

```

570     \tag_struct_end:
571     \tag_mc_begin_pop:n{ }
572 }

```

At last assign the plug:

```

573 \AssignTaggingSocketPlug{fnmark}{FEMark}

```

5.16.5 The footnote text

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

```

\__fnote_gput_ref:nn
\fnote_gput_refs:nn 574 \cs_new_protected:Npn \__fnote_gput_ref:nn #1 #2 % #1 the structure number receiving the ref
\fnote_gput_refs:ee 575 {
576     \tag_struct_gput:nnn {#1}{ref_num}{#2}
577 }
578 \cs_new_protected:Npn \fnote_gput_refs:nn #1 #2 % pair of structure numbers
579 {
580     \__fnote_gput_ref:nn {#1}{#2}
581     \__fnote_gput_ref:nn {#2}{#1}
582 }
583 \cs_generate_variant:Nn \fnote_gput_refs:nn {ee}
(End of definition for \__fnote_gput_ref:nn and \fnote_gput_refs:nn.)

```

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

FENote (*plug*)

```

584 \NewTaggingSocketPlug{fntext/begin}{FENote}
585 {

```

The id has been increased in the previous fntext/before hook.

```

586     \tag_mc_end_push:

```

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

```

587     \tag_check_child:nnTF {FENote}{pdf2}
588     {
589         \tag_struct_begin:n { tag=footnote }
590     }
591     {
592         \tag_struct_begin:n
593         {
594             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)

```

595         parent=\int_max:nn{2}{\tag_get:n{current_Sect}+0}
596     }
597 }

```

Store the current structure number for labels.

```

598     \tl_set:Nx \l__fnote_currentstruct_tl { \tag_get:n{struct_num} }

```

After we have opened the structure we can use the `struct num` to try to detect the connected marks. As with the marks we assume that sometimes no auto detection is done.

```

599     \bool_if:NTF \l__fnote_autodetect_bool
600     {

```

Find open marks with identical \@thefnmark:

```
601 \fnote_mark_gpop_all:noon {struct}{ \@thefnmark }{ \l_fnote_type_t1 } \l__fnote_curr
```

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:

```
602 \seq_map_inline:Nn \l__fnote_currentrefs_seq
603 {
604 \fnote_gput_refs:ee {##1}{ \l__fnote_currentstruct_t1 }
605 }
606 }
```

If no auto detection is done

```
607 {%no auto
608
609 }
610 }
```

This finish the setup of the tagging structure.

FENote (*plug*) This is the plug for the socket at the end of the structure.

```
611 \NewTaggingSocketPlug{fntext/end}{FENote}
612 {
613 \tag_struct_end:
614 \tag_mc_begin_pop:n{ }
615 }
```

At last assign the plugs:

```
616 \AssignTaggingSocketPlug{fntext/begin}{FENote}
617 \AssignTaggingSocketPlug{fntext/end}{FENote}
```

fntext/mark/link (*socket*) This socket adds targets for links before the mark of the footnote in the note text. It should be used in hmode.

```
618 \NewSocket{fntext/mark/link}{0}
```

link (*plug*) We create a link target for every related mark. The name is footnote*.\<id>. We also add a link target for the current id (for \footref). TODO: perhaps it should be always active.

```
619 \NewSocketPlug{fntext/mark/link}{link}
620 {
621 \fnote_mark_gpop_all:noon {id}{ \@thefnmark }{ \l_fnote_type_t1 } \l__fnote_currentrefs_
622 \seq_map_inline:Nn\l__fnote_currentrefs_seq {\MakeLinkTarget*{footnote*.\<##1>}}
623 \MakeLinkTarget*{footnote*.\int_use:N\g_fnote_id_int}
624 }
```

The kernel socket `tagsupport/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.

```
625 \NewTaggingSocketPlug{fntext/mark}{FENoteLbl}
626 {
627 \tag_mc_end_push:
```

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

```

628     \tag_struct_begin:n { tag=footnotelabel,parent=\l__fnote_currentstruct_tl,firstkid }
629     \tag_mc_begin:n { tag=Lbl }
630     #2
631     \tag_mc_end:
632     \tag_struct_end:
633     \tag_mc_begin_pop:n{
634   }
635 \AssignSocketPlug{tagssupport/fntext/mark}{FENoteLbl}

```

FENotetext (*plug*)

This plug is for the kernel socket `tagssupport/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?

```

636 \NewTaggingSocketPlug{fntext/text}{FENotetext}
637 {
638   \tag_mc_end_push:
639   \tag_mc_begin:n{
640     #2
641     \tag_mc_end:
642     \tag_mc_begin_pop:n{
643   }
644 \AssignTaggingSocketPlug{fntext/text}{FENotetext}

```

```

645 \ExplSyntaxOff
646 </kernel>
647 <@@=)

```

6 Reimplementing the footmisc package

```

648 <*footmisc>
649 %%
650 %% Copyright (c) 1995-2011 Robin Fairbairns
651 %% Copyright (c) 2018-2023 Robin Fairbairns, Frank Mittelbach
652 %%
653 %% This file is part of the 'latex-lab Bundle'.
654 %% -----
655 %%
656 %% It may be distributed and/or modified under the
657 %% conditions of the LaTeX Project Public License, either version 1.3c
658 %% of this license or (at your option) any later version.
659 %% The latest version of this license is in
660 %%   https://www.latex-project.org/lppl.txt
661 %% and version 1.3c or later is part of all distributions of LaTeX
662 %% version 2008 or later.
663 %%
664 %% This work has the LPPL maintenance status 'maintained'.
665 %%
666 \NeedsTeXFormat{LaTeX2e}

```



```

667 \providecommand\DeclareRelease[3]{}
668 \providecommand\DeclareCurrentRelease[2]{}
669
670 \DeclareRelease{v5}{2011-06-06}{footmisc-2011-06-06.sty}
671 \DeclareCurrentRelease{}{2022-02-14}
672 \ProvidesPackage{latex-lab-footmisc}%
673     [2025/02/20 v6.0e
674     a miscellany of footnote facilities -- latex-lab version%
675     ]
676
677 \NeedsTeXFormat{LaTeX2e}[2020/10/01]
678 \newtoks\FN@temptoken
679 \providecommand\protected@writeaux{%
680     \protected@write\@auxout
681 }
682 \def\l@advance@macro{\@@d@vance@macro\edef}
683 \def\@@d@vance@macro#1#2#3{\expandafter\@tempcnta#2\relax
684     \advance\@tempcnta#3\relax
685     #1#2{\the\@tempcnta}%
686 }
687 \let\@advance@macro\l@advance@macro
688 \DeclareOption{symbol}{\renewcommand\thefootnote{\fnsymbol{footnote}}}
689 \newif\ifFN@robust \FN@robustfalse
690 \DeclareOption{symbol*}{%
691     \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
692     \FN@robusttrue
693     \AtEndOfPackage{\setfnsymbol{lamport*-robust}}}%
694 }
695 \newif\ifFN@para \FN@parafalse
696 \DeclareOption{para}{%

```

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

```

697 % \PackageError{footmisc}{Option "\CurrentOption" incompatible with
698 % option "side"}%
699 % {I shall ignore "\CurrentOption"}%
700 \FN@paratrue
701 \setlength\footnotemargin{-\maxdimen} % default when para is used
702 }
703 \DeclareOption{side}{\ifFN@para
704     \PackageError{footmisc}{Option "\CurrentOption" incompatible with
705     option "para"}%
706     {I shall ignore "\CurrentOption"}%
707 }
708 \AddToHook{fntext/para}{%
709     \hspace\marginparwidth % correct the default \hspace
710     \footnotesep\z@ % don't add a default separation
711 }
712 \AssignSocketPlug{fntext/process}{side}
713 % \AssignSocketPlug{fntext/make}{default}
714 \AssignSocketPlug{fntext/begin}{noop}
715 \AssignSocketPlug{fntext/end}{noop}
716 \fi
717 }

```

```

718 \let\footnotelayout\@empty
719 \DeclareOption{ragged}{%
720   \@ifundefined{RaggedRight}%
721     {\renewcommand\footnotelayout{\linepenalty50 \raggedright}}%
722     {\renewcommand\footnotelayout{\linepenalty50 \RaggedRight}}%
723 }
724 \newif\ifFN@perpage
725 \FN@perpagefalse
726 \DeclareOption{perpage}{%
727   \FN@perpagetrue
728 }
729 \newif\ifFN@fixskip      \FN@fixskipfalse
730
731 \let\FN@bottomcases\thr@@
732 \newif\ifFN@abovefloats  \FN@abovefloatstrue
733 \DeclareOption{bottom}{%
734   \let\FN@bottomcases\@ne
735   \FN@abovefloatsfalse
736   \FN@fixskiptrue
737 }
738 \DeclareOption{bottomfloats}{%
739   \let\FN@bottomcases\tw@
740   \FN@abovefloatstrue \FN@fixskiptrue
741 }
742 \DeclareOption{abovefloats}{\FN@abovefloatstrue \FN@fixskiptrue}
743 \DeclareOption{belowfloats}{\FN@abovefloatsfalse \FN@fixskiptrue}
744 \DeclareOption{marginal}{%
745   \footnotemargin-0.8em\relax
746 }
747 \DeclareOption{flushmargin}{%
748   \footnotemargin0pt\relax
749 }
750 \newif\ifFN@hangfoot    \FN@hangfootfalse
751 \DeclareOption{hang}{%
752   \FN@hangfoottrue
753 }
754 \newcommand*\hangfootparskip{0.5\baselineskip}
755 \newcommand*\hangfootparindent{0em}%
756 \DeclareOption{norule}{%
757   \renewcommand\footnoterule{}%
758   \advance\skip\footins 4\p@\@plus2\p@\relax
759 }
760 \DeclareOption{splitrule}{%
761   \gdef\split@prev{0}
762   \let\pagefootnoterule\footnoterule
763   \let\mpfootnoterule\footnoterule
764   \def\splitfootnoterule{\kern-3\p@ \hrule \kern2.6\p@}
765   \def\footnoterule{\relax
766     \ifx \@listdepth\@mplistdepth
767       \mpfootnoterule
768     \else
769       \ifnum\split@prev=\z@
770         \pagefootnoterule
771       \else

```

```

772         \splitfootnoterule
773     \fi
774     \xdef\split@prev{\the\insertpenalties}%
775 \fi
776 }%
777 }
778 \newif\ifFN@stablefootnote \FN@stablefootnotefalse
779 \DeclareOption{stable}{\FN@stablefootnotetrue}
780 \newif\ifFN@multiplefootnote \FN@multiplefootnotefalse
781 \DeclareOption{multiple}{\FN@multiplefootnotetrue}
782 \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).

783 \ifFN@para
784     \NewSocketPlug{build/column/footnotes}{para}{%
785         \global\setbox\footins\vbox{\FN@makefootnoteparagraph}%
786     }
787     \AssignSocketPlug{build/column/footnotes}{para}
788 \fi

789 \ifFN@fixskip
790     \def\@outputbox@removebskip{%
791         \ifx\@textbottom\relax \else
792             \@outputbox@append{%
793                 \@tempskipa\lastskip
794                 \ifnum \gluestretchorder\@tempskipa>\z@
795                     \vskip-\@tempskipa
796                     \xdef\@outputbox@reinsertbskip
797                         {\noexpand\@outputbox@append{\vskip\the\@tempskipa}}%
798                 \else
799                     \global\let\@outputbox@reinsertbskip\relax
800                 \fi
801             }%
802         \fi
803     }
804     \let\@outputbox@reinsertbskip\relax
805 \else
806     \let\@outputbox@removebskip \relax
807     \let\@outputbox@reinsertbskip\relax
808 \fi

809 \ifcase \FN@bottomcases\relax
810     \ERROR
811 \or %1 bottom option
812     \ifFN@abovefloats
813         \AssignSocketPlug {build/column/outputbox}{space-footnotes-floats}
814     \else
815         \AssignSocketPlug {build/column/outputbox}{floats-space-footnotes}
816     \fi
817 \or %2 bottomfloats
818     \ifFN@abovefloats
819         \AssignSocketPlug {build/column/outputbox}{footnotes-space-floats}
820     \else
821         \AssignSocketPlug {build/column/outputbox}{space-floats-footnotes}

```

```

822 \fi
823 \or %3 neither bottom nor bottomfloats
824 \ifFN@abovefloats
825 \AssignSocketPlug {build/column/outputbox}{footnotes-floats}
826 \else
827 \AssignSocketPlug {build/column/outputbox}{floats-footnotes}
828 \fi
829 \else
830 \ERROR
831 \fi
832
833 % next can be dropped when cleaned up
834 \newif\ifFN@setspace
835 \@ifpackageloaded{setspace}%
836 {%
837 \FN@setspace>true
838 \@ifclassloaded{memoir}%
839 {%
840 \AddToHook{fntext}{\let\baselinestretch\m@m@singlespace}%
841 \let\FN@baselinestretch\m@m@singlespace
842 }%
843 {%
844 % \AddToHook{fntext}{\let\baselinestretch\setspace@singlespace}%
845 \let\FN@baselinestretch\setspace@singlespace
846 }%
847 }%
848 {%
849 \FN@setspace=false
850 }
851
852
853
854 \ifFN@para
855
856 % \AssignSocketPlug{fntext/process}{default}
857 \AssignSocketPlug{fntext/make}{para}
858 \AssignSocketPlug{fntext/begin}{noop}
859 \AssignSocketPlug{fntext/end}{para}
860
861 \fi
862
863
864
865 \ifFN@para
866 \let\FN@tempboxa\@tempboxa
867 \newbox\FN@tempboxb
868 \newbox\FN@tempboxc
869 \newskip\footglue \footglue=1em plus.3em minus.3em
870
871 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
872 \newdimen\footnotebaselineskip
873
874 % establish late:
875

```

```

876 \AddToHook{begindocument/before} {%
877   {%
878     \footnotesize
879     \global\footnotebaselineskip=\normalbaselineskip
880   }%
881 }

```

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.

```

882 \long\def\FN@makefootnoteparagraph{%
883   \FN@setfootnoteparawidth
884   \@parboxrestore
885   \baselineskip=\footnotebaselineskip
886   \unvbox\footins \FN@removehboxes
887   \RawParEnd
888 }
889 \def\FN@removehboxes{\setbox\FN@tempboxa\lastbox
890   \ifhbox\FN@tempboxa{\FN@removehboxes}%
891   \unhbox\FN@tempboxa
892   \else
893     \RawNoindent
894     \rule{z@}{footnotesep}
895   \fi
896 }
897 \fi
898
899
900 \@ifpackageloaded{multicol}
901   {\def\FN@setfootnoteparawidth
902     {\hsize\ifnum\doublecol@number>\@ne
903       \textwidth
904       \else \columnwidth \fi}}
905   {\def\FN@setfootnoteparawidth{\hsize\columnwidth}}
906
907 \ifFN@perpage
908   \RequirePackage{perpage}
909   \MakePerPage{footnote}

```

Fix a bug in perpage ...

```

910 \def\@stpelt#1{\global\csname c@#1\endcsname \m@ne
911   \stepcounter{#1}%
912   \pp@fix@MakePerPage{#1}%
913 }
914 \def\pp@fix@MakePerPage#1{%
915   \ifnum \value{#1}>z@
916     \addtocounter{#1}\m@ne\fi
917 }

```

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:

```

918 \def\pp@ccl@end@iii\stepcounter#1\pp@fix@MakePerPage#2{}
919 \fi
920
921
922 \ifFN@para
923
924 % This can use the default interface, except that a negative value for
925 % \footnotemargin makes little sense, so we test for this and warn if
926 % necessary. But -\maxdimen is ok again, so would need to be a little bit more elaborate.
927 %
928
929 %\AddToHook{fntext/para}{
930 % \ifdim \footnotemargin >\z@ \else
931 % \PackageWarningNoline{footmisc}{Option 'para' needs positive \noexpand\footnotemargin}%
932 % \footnotemargin 1.8em\relax
933 % \fi
934 %}
935
936
937 \AddToHook{fntext/begin}{\nobreak \hspace{.2em}}
938
939 \else
940
941 \ifFN@hangfoot
942 \long\def\footmisc@hang@makefntext#1{%
943 \bgroup
944 \SuspendTagging{footmisc}%
945 \setbox\@tempboxa\hbox{%
946 \ifdim\footnotemargin>\z@
947 \hb@xt@\footnotemargin{\@makefnmark\hss}%
948 \else
949 \@makefnmark
950 \fi
951 }%
952 \leftmargin\wd\@tempboxa
953 \rightmargin\z@
954 \linewidth \columnwidth
955 \advance \linewidth -\leftmargin
956 \parshape \@ne \leftmargin \linewidth
957 \footnotesize
958 \parskip\hangfootparskip\relax
959 \parindent\hangfootparindent\relax
960 \@setpar{\@@par}}%
961 \ResumeTagging{footmisc}%
962 \leavevmode

```

The setting of \parindent has to happen prior to calling \leavevmode (which happens below to support tagging). Originally, this was done later so was moved here where we haven't started hmode yet.

after leaving the vmode we can set the link targets. They should be in the footnote structure.

```

963 \UseSocket{fntext/mark/link}%

```

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.

```

964         \UseTaggingSocket{fntext/mark}{}%
965         f\llap{%
966             \ifdim\footnotemargin>\z@
967             \hb@xt@\footnotemargin{\@makefnmark\hss}%
968             \else
969             \@makefnmark
970             \fi
971         }%
972         \footnotelayout#1%
973         \par
974     \egroup
975 }

```

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

```

976     \let \@makefntext \footmisc@hang@makefntext
977
978     \else
979
980     % This is now using the default interface:
981     %
982     % \long\def\@makefntext#1{%
983     %     \parindent1em
984     %     \noindent
985     %     \ifdim\footnotemargin>\z@
986     %         \hb@xt@ \footnotemargin{\hss\@makefnmark}%
987     %     \else
988     %         \ifdim\footnotemargin=\z@
989     %             \llap{\@makefnmark}%
990     %         \else
991     %             \llap{\hb@xt@ -\footnotemargin{\@makefnmark\hss}}%
992     %         \fi
993     %     \fi
994     %     \footnotelayout#1%
995     % }
996
997     \fi
998     \fi
999
1000
1001
1002
1003 \iffN@multiplefootnote
1004     \providecommand*\multiplefootnotemarker}{3sp}

```

We tag the separator as artifact

TODO: *why is this done with \providecommand?*

```

1005     \ExplSyntaxOn
1006     \providecommand*\multfootsep}{\tag_mc_end_push:\tag_mc_begin:n{artifact},\tag_mc_end:\tag
1007     \ExplSyntaxOff

```

```

1008 \AddToHook{fnmark}      {\FN@mf@check}
1009 \AddToHook{fnmark/end}  {\FN@mf@prepare}
1010 %
1011 \def\FN@mf@prepare{%
1012   \kern-\multiplefootnotemarker
1013   \kern\multiplefootnotemarker\relax
1014 }
1015 \def\FN@mf@check{%
1016   \ifdim\lastkern=\multiplefootnotemarker\relax
1017   ??? is that necessary or even correct ??
1018   \edef\x@sf{\the\spacefactor}%
1019   ??? shouldn't that be 2 unkerns ?? (none would also be ok)
1020   \unkern % new
1021   \unkern
1022   \textsuperscript{\multfootsep}%
1023   \spacefactor\x@sf\relax
1024   \fi
1025 }
1026 \else
1027   \let\FN@mf@prepare\relax
1028 \fi
1029 \ifFN@stablefootnote
1030 \let\FN@sff@footnote\footnote
1031 \def\footnote{\ifx\protect\@typeset@protect
1032   \expandafter\FN@sff@footnote
1033   \else
1034     \expandafter\FN@sff@gobble@opt
1035   \fi
1036 }
1037 \edef\FN@sff@gobble@opt{\noexpand\protect
1038   \expandafter\noexpand\csname FN@sff@gobble@opt \endcsname}
1039 \expandafter\def\csname FN@sff@gobble@opt \endcsname{%
1040   \@ifnextchar[%]
1041     \FN@sff@gobble@twobracket
1042     \@gobble
1043 }
1044 \def\FN@sff@gobble@twobracket[#1]#2{}
1045 \let\FN@sff@footnotemark\footnotemark
1046 \def\footnotemark{\ifx\protect\@typeset@protect
1047   \expandafter\FN@sff@footnotemark
1048   \else
1049     \expandafter\FN@sff@gobble@optonly
1050   \fi
1051 }
1052 \edef\FN@sff@gobble@optonly{\noexpand\protect
1053   \expandafter\noexpand\csname FN@sff@gobble@optonly \endcsname}
1054 \expandafter\def\csname FN@sff@gobble@optonly \endcsname{%
1055   \@ifnextchar[%]
1056     \FN@sff@gobble@bracket
1057     {}%
1058 }
1059 \def\FN@sff@gobble@bracket[#1]{}
1060 \fi
1061 \newcommand\setfnsymbol[1]{%

```



```

1062 \@bsphack
1063 \@ifundefined{FN@fnsymbol@#1}%
1064 {%
1065   \PackageError{footmisc}{Symbol style "#1" not known}%
1066   \@eha
1067 }{%
1068   \expandafter\let\expandafter\@fnsymbol\csname
1069     FN@fnsymbol@#1\endcsname
1070 }%
1071 \@esphack
1072 }
1073 \let\FN@fnsymbol@lamport\@fnsymbol
1074 \newif\if@tempswb
1075 \DeclareDocumentCommand\DefineFNSymbols {sm0{text}m}{%
1076   \expandafter\ifx\csname FN@fnsymbol@#2\endcsname\relax
1077     \PackageInfo{footmisc}{Declaring symbol style #2}%
1078   \else
1079     \PackageWarning{footmisc}{Redeclaring symbol style #2}%
1080   \fi
1081   \toks@{}%
1082   \def\@tempb{\end}%
1083   \FN@build@symboldef#4\end
1084   \def\@tempc{math}%
1085   \def\@tempd{#3}%
1086   \expandafter\xdef\csname FN@fnsymbol@#2\endcsname##1{%
1087     \ifx\@tempc\@tempd
1088       \noexpand\ensuremath
1089     \else
1090       \noexpand\nfss@text
1091     \fi
1092     {%
1093       \noexpand\ifcase##1%
1094       \the\toks@
1095       \noexpand\else
1096       \IfBooleanTF#1{\noexpand\@ctrerr}%
1097       {\noexpand\FN@orange##1}%
1098       \noexpand\fi
1099     }%
1100   }%
1101 }
1102 \def\FN@build@symboldef#1{%
1103   \def\@tempa{#1}%
1104   \ifx\@tempa\@tempb
1105   \else
1106     \toks@\expandafter{\the\toks@\or#1}%
1107     \expandafter\FN@build@symboldef
1108   \fi
1109 }
1110 \DeclareDocumentCommand\DefineFNSymbolsTM {smm}{%
1111   \expandafter\ifx\csname FN@fnsymbol@#2\endcsname\relax
1112     \PackageInfo{footmisc}{Declaring symbol style #2}%
1113   \else
1114     \PackageWarning{footmisc}{Redeclaring symbol style #2}%
1115   \fi

```

```

1116 \toks@{ }%
1117 \def\@tempb{\end}%
1118 \FN@build@symboldefTM#3\end\@null
1119 \expandafter\xdef\csname FN@fnsymbol@#2\endcsname##1{%
1120   \noexpand\ifcase##1%
1121     \the\toks@
1122   \noexpand\else
1123     \IfBooleanTF#1{\noexpand\@ctrerr}%
1124     {\noexpand\FN@orange##1}%
1125   \noexpand\fi
1126 }%
1127 }
1128 \def\FN@build@symboldefTM#1#2{%
1129   \def\@tempa{#1}%
1130   \ifx\@tempa\@tempb
1131   \else
1132     \toks@\expandafter{\the\toks@\or\TextOrMath{#1}{#2}}%
1133     \expandafter\FN@build@symboldefTM
1134   \fi
1135 }
1136 \def\FN@orange#1{%
1137   \ifFN@robust
1138     \@arabic#1%
1139     \@bsphack
1140     \PackageInfo{footmisc}{Footnote number \number#1 out of range}%
1141     \protect\@fnsymbol@orange
1142     \@esphack
1143   \else \@ctrerr \fi
1144 }
1145 \global\let\@diagnose@fnsymbol@orange\relax
1146 \AtEndDocument{\@diagnose@fnsymbol@orange}
1147 \def\@fnsymbol@orange{%
1148   \gdef\@diagnose@fnsymbol@orange{%
1149     \PackageWarningNoLine{footmisc}{Some footnote number(s)
1150       were out of range
1151       \MessageBreak
1152       see log for details%
1153     }%
1154   }%
1155 }
1156 \DefineFNSymbolsTM{bringhurst}{%
1157   \textasteriskcentered *%
1158   \textdagger \dagger
1159   \textdaggerdbl \ddagger
1160   \textsection \mathsection
1161   \textbardbl \||%
1162   \textparagraph \mathparagraph
1163 }%
1164 \DefineFNSymbolsTM{chicago}{%
1165   \textasteriskcentered *%
1166   \textdagger \dagger
1167   \textdaggerdbl \ddagger
1168   \textsection \mathsection
1169   \textbardbl \||%

```

```

1170 \#\#%
1171 }%
1172 \DefineFNSymbolsTM{wiley}{%
1173 \textasteriskcentered *%
1174 {\textasteriskcentered\textasteriskcentered}{**}%
1175 \textdagger \dagger
1176 \textdaggerdbl \ddagger
1177 \textsection \mathsection
1178 \textparagraph \mathparagraph
1179 \textbardbl \||%
1180 }%
1181 \DefineFNSymbolsTM{lamport-robust}{%
1182 \textasteriskcentered *%
1183 \textdagger \dagger
1184 \textdaggerdbl \ddagger
1185 \textsection \mathsection
1186 \textparagraph \mathparagraph
1187 \textbardbl \||%
1188 {\textasteriskcentered\textasteriskcentered}{**}%
1189 {\textdagger\textdagger}{\dagger\dagger}%
1190 {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
1191 }
1192 \DefineFNSymbolsTM*{lamport*}{%
1193 \textasteriskcentered *%
1194 \textdagger \dagger
1195 \textdaggerdbl \ddagger
1196 \textsection \mathsection
1197 \textparagraph \mathparagraph
1198 \textbardbl \||%
1199 {\textasteriskcentered\textasteriskcentered}{**}%
1200 {\textdagger\textdagger}{\dagger\dagger}%
1201 {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
1202 {\textsection\textsection}{\mathsection\mathsection}%
1203 {\textparagraph\textparagraph}{\mathparagraph\mathparagraph}%
1204 {\textasteriskcentered\textasteriskcentered\textasteriskcentered}{***}%
1205 {\textdagger\textdagger\textdagger}{\dagger\dagger\dagger}%
1206 {\textdaggerdbl\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger\ddagger}%
1207 {\textsection\textsection\textsection}%%
1208 {\mathsection\mathsection\mathsection}%
1209 {\textparagraph\textparagraph\textparagraph}%%
1210 {\mathparagraph\mathparagraph\mathparagraph}%
1211 }
1212 \setfnsymbol{lamport*}
1213 \DefineFNSymbolsTM{lamport*-robust}{%
1214 \textasteriskcentered *%
1215 \textdagger \dagger
1216 \textdaggerdbl \ddagger
1217 \textsection \mathsection
1218 \textparagraph \mathparagraph
1219 \textbardbl \||%
1220 {\textasteriskcentered\textasteriskcentered}{**}%
1221 {\textdagger\textdagger}{\dagger\dagger}%
1222 {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}%
1223 {\textsection\textsection}{\mathsection\mathsection}%

```

```

1224 {\textparagraph\textparagraph}{\mathparagraph\mathparagraph}%
1225 {\textasteriskcentered\textasteriskcentered\textasteriskcentered}{***}%
1226 {\textdagger\textdagger\textdagger}{\dagger\dagger\dagger}%
1227 {\textdaggerdbl\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger\ddagger}%
1228 {\textsection\textsection\textsection}%%
1229 {\mathsection\mathsection\mathsection}%
1230 {\textparagraph\textparagraph\textparagraph}%%
1231 {\mathparagraph\mathparagraph\mathparagraph}%
1232 }
1233 \newcommand\mpfootnotemark{%
1234 \ifnextchar[%]
1235 \xmpfootnotemark
1236 {%
1237 \stepcounter\@mpfn
1238 \protected@xdef\@thefnmark{\thempfn}%
1239 \@footnotemark
1240 }%
1241 }
1242 \def\xmpfootnotemark[#1]{%
1243 \begingroup
1244 \csname c@\@mpfn\endcsname #1\relax
1245 \unrestored@protected@xdef\@thefnmark{\thempfn}%
1246 \endgroup
1247 \@footnotemark
1248 }
1249 \endinput
1250 </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			
<code>\#</code>	1170	<code>\AtEndDocument</code> 1146
<code>\ </code>	1161, 1169, 1179, 1187, 1198, 1219	<code>\AtEndOfPackage</code> 693
A		B	
<code>\addtocounter</code>	916	<code>\baselineskip</code> 754, 885
<code>\AddToHook</code>	308,	<code>\baselinestretch</code> 459, 840, 844
		391, 449, 451, 457, 459, 479, 484,	<code>\begingroup</code> 434, 1243
		708, 840, 844, 876, 929, 937, 1008, 1009	<code>\bgroup</code> 943
<code>\AddToHookWithArguments</code>	483	bool commands:
<code>\advance</code>	684, 758, 955	<code>\bool_gset_false:N</code> 118
<code>\AssignSocketPlug</code>	281,	<code>\bool_gset_true:N</code> 117
		293, 296, 305, 343, 470, 471, 635,	<code>\bool_if:NTF</code>
		712, 713, 714, 715, 787, 813, 815, 121, 133, 530, 539, 554, 564, 599
		819, 821, 825, 827, 856, 857, 858, 859	<code>\bool_new:N</code> 69, 76, 116
<code>\AssignTaggingSocketPlug</code>		<code>\bool_set_false:N</code> 437, 472, 475
		573, 616, 617, 644	<code>\bool_set_true:N</code> 70, 77, 440
<code>\AtBeginDocument</code>	315, 461, 466	<code>\box</code> 291

C	
<code>\columnwidth</code> ..	16, 227, 290, 904, 905, 954
<code>\cs</code>	236, 237
cs commands:	
<code>\cs_generate_variant:Nn</code>	80, 81, 510, 511, 525, 583
<code>\cs_gset_eq:NN</code>	128, 152
<code>\cs_if_eq:NNTF</code>	393, 395, 400
<code>\cs_new:Npn</code>	378, 380, 382, 384, 386
<code>\cs_new_eq:NN</code>	197, 352
<code>\cs_new_protected:Npn</code>	82, 95, 117, 118, 119, 131, 155, 182, 200, 321, 354, 482, 491, 498, 504, 512, 574, 578
<code>\cs_set:Npn</code>	375
<code>\cs_set_eq:NN</code>	199, 310, 358, 361, 364, 367, 369, 397, 421, 423, 453, 454, 455
<code>\cs_set_protected:Npn</code>	486
<code>\csname</code>	20, 910, 1038, 1039, 1053, 1054, 1068, 1076, 1086, 1111, 1119, 1244
<code>\CurrentOption</code>	697, 699, 704, 706
990, 1026, 1033, 1048, 1078, 1089, 1095, 1105, 1113, 1122, 1131, 1143	
<code>\end</code>	1082, 1083, 1117, 1118
<code>\endcsname</code>	
..	20, 910, 1038, 1039, 1053, 1054, 1069, 1076, 1086, 1111, 1119, 1244
<code>\endgroup</code>	436, 1246
<code>\endinput</code>	1249
<code>\ensuremath</code>	1088
<code>\ERROR</code>	22, 352, 368, 411, 810, 830
exp commands:	
<code>\exp_after:wN</code>	374, 376
<code>\exp_args:No</code>	541
<code>\exp_not:N</code>	558
<code>\exp_stop_f:</code> ...	369, 379, 381, 383, 385
<code>\expandafter</code>	
..	683, 1032, 1034, 1038, 1039, 1047, 1049, 1053, 1054, 1068, 1076, 1086, 1106, 1107, 1111, 1119, 1132, 1133
<code>\ExplSyntaxOff</code>	645, 1007
<code>\ExplSyntaxOn</code>	64, 1005
D	
<code>\dagger</code> ..	1158, 1166, 1175, 1183, 1189, 1194, 1200, 1205, 1215, 1221, 1226
<code>\ddagger</code> ..	1159, 1167, 1176, 1184, 1190, 1195, 1201, 1206, 1216, 1222, 1227
<code>\DebugFNotesOff</code>	7, <u>117</u>
<code>\DebugFNotesOn</code>	7, <u>117</u>
<code>\DeclareCurrentRelease</code>	668, 671
<code>\DeclareDocumentCommand</code>	413, 418, 427, 432, 1075, 1110
<code>\DeclareOption</code>	688, 690, 696, 703, 719, 726, 733, 738, 742, 743, 744, 747, 751, 756, 760, 779, 781
<code>\DeclareRelease</code>	667, 670
<code>\def</code>	12, 18, 29, 30, 34, 231, 375, 460, 682, 683, 764, 765, 790, 882, 889, 901, 905, 910, 914, 918, 942, 982, 1011, 1015, 1031, 1039, 1044, 1046, 1054, 1059, 1082, 1084, 1085, 1102, 1103, 1117, 1128, 1129, 1136, 1147, 1242
<code>default (plug)</code> ..	5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 21
<code>\DefineFNSymbols</code>	1075
<code>\DefineFNSymbolsTM</code>	1110, 1156, 1164, 1172, 1181, 1192, 1213
<code>\dimexpr</code>	289
<code>\dp</code>	225, 287
E	
<code>\edef</code>	164, 682, 1018, 1037, 1052
<code>\egroup</code>	974
<code>\else</code> ..	331, 334, 337, 707, 768, 771, 791, 798, 805, 814, 820, 826, 829, 892, 904, 930, 939, 948, 968, 978, 987,
F	
<code>FEMark (plug)</code>	550
<code>FENote (plug)</code>	<u>584</u> , <u>611</u>
<code>FENoteLbl (plug)</code>	<u>625</u>
<code>FENotetext (plug)</code>	<u>636</u>
<code>\fi</code>	173, 191, 319, 339, 340, 341, 716, 773, 775, 788, 800, 802, 808, 816, 822, 828, 831, 861, 895, 897, 904, 916, 919, 933, 950, 970, 992, 993, 997, 998, 1024, 1028, 1035, 1050, 1060, 1080, 1091, 1098, 1108, 1115, 1125, 1134, 1143
<code>\floatingpenalty</code>	226
<code>fnmark (hook)</code>	4, 4, <u>107</u>
<code>fnmark/after (hook)</code>	4, 4, 4, 6, 17, <u>107</u>
<code>fnmark/before (hook)</code>	4, 4, 4, <u>107</u>
<code>fnmark/begin (hook)</code>	4, 4, <u>107</u>
<code>fnmark/end (hook)</code>	4, 4, 17, <u>107</u>
<code>fnmark/link (socket)</code>	<u>526</u>
fnote commands:	
<code>\fnote_class_new:nn</code> ..	9, <u>491</u> , 491, 497
<code>\g_fnote_debug_bool</code>	116, 117, 118, 121, 133
<code>\fnote_footnotemark:</code>	8, <u>155</u> , 155, 199, 453
<code>\fnote_footnotetext:n</code>	<u>200</u> , 200, 310, 458
<code>\fnote_gput_refs:nn</code>	558, <u>574</u> , 578, 583, 604
<code>\g_fnote_id_int</code>	9, 26, 478, 479, 481, 507, 529, 533, 623
<code>\l_fnote_link_bool</code> ..	<u>69</u> , 472, 475, 539

\l_fnote_link_type_tl	71 , 543
\fnote_makefntext:n	11 , 22 , 23 , 321 , 321 , 393 , 397
\fnote_mark_gpop_all:nnN	10
\fnote_mark_gpop_all:nnnN	10 , 512 , 512 , 525 , 601 , 621
\fnote_mark_id_gput:nn	498 , 504 , 511 , 532
\fnote_mark_id_gput:nnn	9
\fnote_mark_struct_gput:nn	498 , 498 , 510 , 566
\fnote_mark_struct_gput:nnn	9
\fnote_set_fnmark:nn	95 , 95 , 415
\fnote_step_fnmark:nn	82 , 82 , 420 , 429	
\l_fnote_type_tl	67 , 532 , 566 , 601 , 621	
fnote internal commands:		
\l__fnote_autodetect_bool	76 , 437 , 440 , 530 , 554 , 564 , 599
\l__fnote_currentlabel_tl	25 , 29 , 74 , 438 , 537 , 560
\l__fnote_currentrefs_seq	75 , 601 , 602 , 621 , 622
\l__fnote_currentstruct_tl	27 , 78 , 79 , 480 , 598 , 604 , 628
__fnote_debug_footnotemark:	...	119 , 128 , 156
__fnote_debug_footnotetext:	...	131 , 152 , 201
__fnote_footmark_finish:	17
__fnote_footnotemark_default_	finish:	17 , 182 , 182 , 197 , 423
__fnote_footnotemark_finish:	..	17 , 180 , 182 , 197 , 421 , 423 , 425
__fnote_gput_ref:nn	574 , 574 , 580 , 581	
__fnote_label_hook:n	482 , 482 , 483 , 486
\l__fnote_linktarget_tl	73 , 533 , 536 , 544
__fnote_patch:	354 , 401
__fnote_patch_@makefnmark:w	367 , 384	
__fnote_patch_hb@xt:w	361 , 380
__fnote_patch_hbox:w	358 , 378
__fnote_patch_makebox:w	..	364 , 382
\l__fnote_patch_seq	410 , 411
\l__fnote_patch_tl	351 , 356 , 357 , 360 , 363 , 366 , 373 , 374 , 376 , 410
__fnote_tmp:w	352 , 358 , 361 , 364 , 367 , 369 , 374 , 375 , 376
\l__fnote_tmpa_prop	65 , 515 , 516
\l__fnote_tmpa_tl	66
\fnsymbol	688
fntext (hook)	6 , 6 , 111
fntext/after (hook)	6 , 6 , 6 , 111
fntext/before (hook)	6 , 6 , 6 , 111
fntext/begin (hook)	6 , 6 , 6 , 111
fntext/end (hook)	6 , 6 , 6 , 111
fntext/end (socket)	5 , 5 , 297
fntext/make (socket)	5 , 5 , 5 , 5 , 5 , 6 , 282
fntext/mark (socket)	5 , 5 , 327
fntext/mark/link (socket)	618
fntext/para (hook)	6 , 6 , 6 , 6 , 21 , 111
fntext/process (socket)	5 , 5 , 278
fntext/text (socket)	6 , 6 , 344
\footglue	303 , 869
\footins	5 , 279 , 758 , 785 , 886
\footnote	5 , 9 , 11 , 19 , 418 , 1030 , 1031
\footnotebaselineskip	.	290 , 872 , 879 , 885
\footnotelayout	...	718 , 721 , 722 , 972 , 994
\footnotemargin	5 , 7 , 21 , 312 , 313 , 317 , 318 , 329 , 330 , 332 , 335 , 338 , 701 , 745 , 748 , 925 , 930 , 931 , 932 , 946 , 947 , 966 , 967 , 985 , 986 , 988 , 991
\footnotemark	4 , 9–11 , 427 , 1045 , 1046
\footnoterule	757 , 762 , 763 , 765
\footnotesep	23 , 224 , 295 , 710 , 894
\footnotesize	15 , 217 , 878 , 957
\footnotetext	5 , 6 , 9–11 , 19 , 413
\footref	10 , 11 , 25 , 27 , 31 , 432
G		
\gdef	761 , 1148
\global	..	13 , 444 , 785 , 799 , 879 , 910 , 1145

P	
\PackageError	697, 704, 1065
\PackageInfo	1077, 1112, 1140
\PackageWarning	1079, 1114
\PackageWarningNoLine	1149
\PackageWarningNoline	931
\pagefootnoterule	762, 770
\par	24, 268, 973
para (plug)	5, 5, 5
\parindent	6, 38, 230, 348, 959, 983
\parshape	956
\parskip	958
\penalty	302
Plugs:	
default	5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 21
FEMark	550
FENote	584, 611
FENoteLbl	625
FENotetext	636
identity	6
link	527, 619
mp	5
noop	5, 5, 5, 6
para	5, 5, 5
side	5
transparent	22
prg commands:	
\prg_do_nothing:	128, 152, 421
\ProcessOptions	782
prop commands:	
\prop_gput:Nnn	500, 506
\prop_gremove:Nn	521
\prop_map_inline:Nn	516
\prop_new:N	65, 493, 494
\prop_set_eq:NN	515
property commands:	
\property_new:nnnn	480, 481
\property_record:nn	488
\property_ref:nn	537, 560
\protect	1031, 1037, 1046, 1052, 1141
\providecommand	39, 463, 464, 667, 668, 679, 1004, 1006
\ProvidesFile	3
\ProvidesPackage	672
R	
\RaggedRight	722
\raggedright	721
\RawNoindent	893
\RawParEnd	887
\ref	25, 435
\relax	190, 290, 302, 683, 684, 745, 748, 758, 765, 791, 799, 804, 806, 807, 809, 932, 958, 959, 1013, 1016, 1023, 1027, 1076, 1111, 1145, 1244
\renewcommand	688, 691, 721, 722, 757
\RequirePackage	908
\ResumeTagging	961
\rightmargin	953
\rule	23, 295, 894
S	
seq commands:	
\seq_clear:N	514
\seq_count:N	411
\seq_map_inline:Nn	602, 622
\seq_new:N	75
\seq_put_right:Nn	520
\seq_set_split:Nnn	410
\setbox	13, 286, 444, 785, 889, 945
\setfnsymbol	693, 1061, 1212
\setlength	318, 701
side (plug)	5
\skip	758
socket commands:	
\socket_log:n	135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145
\socket_use:n	178, 253, 266, 323, 324, 388
\socket_use:nn	207, 248, 325
Sockets:	
fnmark/link	526
fntext/begin	5, 5, 294
fntext/end	5, 5, 297
fntext/make	5, 5, 5, 5, 5, 6, 282
fntext/mark	5, 5, 327
fntext/mark/link	618
fntext/process	5, 5, 278
fntext/text	6, 6, 344
tagsupport/fnmark	8, 8, 11, 29, 198
tagsupport/fntext/begin	8, 8, 11, 306
tagsupport/fntext/end	8, 8, 11, 306
tagsupport/fntext/mark	8, 8, 11, 31, 345
tagsupport/fntext/text	8, 8, 11, 32, 345
\space	4
\spacefactor	164, 190, 1018, 1023
\splitfootnoterule	764, 772
\splitmaxdepth	225
\splittopskip	224
\stepcounter	37, 85, 911, 918, 1237
\strut	301
\strutbox	23, 225, 298
\SuspendTagging	944
T	
tag commands:	
\tag_check_child:nnTF	587

<code>\tag_get:n</code>	501, 559, 595, 598	
<code>\tag_mc_begin:n</code> . .	563, 629, 639, 1006	
<code>\tag_mc_begin_pop:n</code>		
.	571, 614, 633, 642, 1006	
<code>\tag_mc_end:</code>	569, 631, 641, 1006	
<code>\tag_mc_end_push:</code>		
.	552, 586, 627, 638, 1006	
<code>\tag_socket_use:n</code>	206, 272	
<code>\tag_socket_use:nnn</code>	178, 324, 325	
<code>\tag_struct_begin:n</code>	553, 589, 592, 628	
<code>\tag_struct_end:</code>	570, 613, 632	
<code>\tag_struct_gput:nnn</code>	576	
<code>\tag_struct_use:n</code>	81	
<code>tagsupport/fnmark (socket)</code> . .	8, 8, 11, 29, 198	
<code>tagsupport/fntext/begin (socket)</code>	8, 8, 11, 306	
<code>tagsupport/fntext/end (socket)</code> . .	8, 8, 11, 306	
<code>tagsupport/fntext/mark (socket)</code>		
.	8, 8, 11, 31, 345	
<code>tagsupport/fntext/text (socket)</code>		
.	8, 8, 11, 32, 345	
TeX and L ^A T _E X 2 _ε commands:		
<code>\@@dvance@macro</code>	682, 683	
<code>\@@par</code>	960	
<code>\@MM</code>	226	
<code>\@advance@macro</code>	687	
<code>\@arabic</code>	1138	
<code>\@auxout</code>	680	
<code>\@bsphack</code>	1062, 1139	
<code>\@ctrerr</code>	1096, 1123, 1143	
<code>\@currentcounter</code>	18, 231	
<code>\@currentlabel</code>	19, 232	
<code>\@diagnose@fnsymbol@orange</code>		
.	1145, 1146, 1148	
<code>\@eha</code>	1066	
<code>\@empty</code>	464, 718	
<code>\@esphack</code>	1071, 1142	
<code>\@finalstrut</code>	23, 298	
<code>\@fnsymbol</code>	691, 1068, 1073	
<code>\@fnsymbol@orange</code>	1141, 1147	
<code>\@footnotemark</code>	3, 8, 16,	
25, 199, 422, 430, 439, 453, 1239, 1247		
<code>\@footnotetext</code>	3, 18, 310, 416, 424, 458	
<code>\@gobble</code>	1042	
<code>\@gobbletwo</code>	463	
<code>\@ifclassloaded</code>	838	
<code>\@ifnextchar</code>	1040, 1055, 1234	
<code>\@ifpackageloaded</code>	468, 835, 900	
<code>\@ifundefined</code>	720, 1063	
<code>\@listdepth</code>	766	
<code>\@makecol</code>	450, 455	
<code>\@makefnmark</code>		
.	3-5, 7, 8, 11, 17, 22, 23, 34, 178,	
330, 333, 336, 338, 350, 366, 384,		
385, 947, 949, 967, 969, 986, 989, 991		
<code>\@makefntext</code>	3, 5, 6, 11, 21-24,	
29, 22, 230, 244, 283, 286, 356,		
375, 393, 395, 397, 400, 454, 976, 982		
<code>\@makefntext@processX</code>		
.	379, 381, 383, 385, 386	
<code>\@mpfn</code>	29, 415, 420, 1237, 1244	
<code>\@mpfootins</code>	13, 14, 444, 445	
<code>\@mpfootnotetext</code>	9, 12	
<code>\@mplistdepth</code>	766	
<code>\@ne</code>	734, 902, 956	
<code>\@null</code>	1118	
<code>\@outputbox@append</code>	792, 797	
<code>\@outputbox@reinsertbskip</code>		
.	796, 799, 804, 807	
<code>\@outputbox@removebskip</code>	790, 806	
<code>\@parboxrestore</code>	17, 228, 884	
<code>\@plus</code>	758	
<code>\@setpar</code>	960	
<code>\@stpelt</code>	910	
<code>\@tempa</code>	1103, 1104, 1129, 1130	
<code>\@tempb</code>	1082, 1104, 1117, 1130	
<code>\@tempboxa</code>	866, 945, 952	
<code>\@tempc</code>	1084, 1087	
<code>\@tempcnta</code>	683, 684, 685	
<code>\@tempd</code>	1085, 1087	
<code>\@tempskipa</code>	793, 794, 795, 797	
<code>\@textbottom</code>	791	
<code>\@textsuperscript</code>	34	
<code>\@thefnmark</code>	9-11, 13-15, 25,	
27-29, 31, 20, 34, 86, 91, 98, 103,		
232, 435, 532, 566, 601, 621, 1238, 1245		
<code>\@typeset@protect</code>	1031, 1046	
<code>\@x@sf</code>	164, 190, 1018, 1023	
<code>\@xmpfootnotemark</code>	1235, 1242	
<code>\c@footnote</code>	691	
<code>\color@begingroup</code>	21, 242	
<code>\color@endgroup</code>	25, 269	
<code>\doublecol@number</code>	902	
<code>\FN@abovefloatsfalse</code>	735, 743	
<code>\FN@abovefloatstrue</code>	732, 740, 742	
<code>\FN@baselinestretch</code>	841, 845	
<code>\FN@bottomcases</code>	731, 734, 739, 809	
<code>\FN@build@symboldef</code>	1083, 1102, 1107	
<code>\FN@build@symboldefTM</code>	1118, 1128, 1133	
<code>\FN@fixskipfalse</code>	729	
<code>\FN@fixskiptrue</code>	736, 740, 742, 743	
<code>\FN@fnsymbol@lampoort</code>	1073	
<code>\FN@hangfootfalse</code>	750	
<code>\FN@hangfoottrue</code>	752	
<code>\FN@makefootnoteparagraph</code>	785, 882	
<code>\FN@mfc@check</code>	1008, 1015	
<code>\FN@mfc@prepare</code>	1009, 1011, 1027	

<code>\UseSocket</code>	963		W
<code>\UseTaggingSocket</code>	229, 356, 389, 964	<code>\wd</code>	289, 952
V			
<code>\value</code>	915		
<code>\vbox</code>	13, 444, 785		X
<code>\vskip</code>	795, 797	<code>\xdef</code>	774, 796, 1086, 1119