The sepfootnotes package,* or a footnote to Plato

Eduardo C. Lourenço de Lima
elourenco@phi.pro.br

March 6, 2012

Abstract

This package allows the content of footnotes to be defined before their marks are inserted in a document. Footnotes and endnotes can, thus, be easily grouped together in a separate file, so that the main body is less cluttered.

Introduction

Footnotes and endnotes can be a problem when reading LaTeX code. If there are many notes to a document, or if some of them are too long, as is often the case with philosophers and people from the humanities in general, reading and editing the main body may be quite difficult.

They can be a problem for the simple reason that standard `\footnote` and `\endnote` both take the text of a note as a mandatory argument. For one thing, a verb and its complement may be[1] severed by a dozen lines of code.

This package aims to get around this inconvenience by means of macros that take as arguments mere *identifiers,* instead of text, thus allowing the content of footnotes and endnotes to be defined *before* their marks are inserted in a document. Notes can, thus, be easily grouped together in a *separate file,* so that the main body is less cluttered.

The point is not to prevent writers from indulging in footnotes. Quite the contrary.

*This document corresponds to sepfootnotes v0.1, dated 2012/03/06.

[1]Notice that this footnote does precisely that to the main body of this document. There are ten or more lines of code between 'may be' and 'severed' in such an otherwise short paragraph. In a posting to `texhax.tug.org` in April 2010, someone complained that using footnotes "tends to disrupt the flow of the document on the screen and makes editing the text itself harder", and asked about the possibility of "putting all of the footnotes in a separate file" [1]. Another user, back in 2004, remarked that the "only thing I missed about word processors was the ability to keep footnote text at the bottom of the page, or in a separate window. I often have extensive footnotes, and don't want them cluttering up my main body text when I'm composing" [2]. Some people circumvent that issue by other artful maneuvering, such as indenting or folding footnote code lines.

Synopsis

`\newfootnotes {`⟨*prefix*⟩`}`
`\newendnotes {`⟨*prefix*⟩`}`
`\newsymbolfootnotes [`⟨*master counter*⟩`] {`⟨*prefix*⟩`}`

`\`⟨*prefix*⟩`note {`⟨*id*⟩`}`
`\`⟨*prefix*⟩`notemark {`⟨*id*⟩`}`
`\`⟨*prefix*⟩`notetext {`⟨*id*⟩`}`
`\`⟨*prefix*⟩`notecontent {`⟨*id*⟩`} {`⟨*text*⟩`}`

`\`⟨*prefix*⟩`notes {`⟨*id*⟩`}`

Contents

1 Basic Usage

1.1 Footnotes

`\newfootnotes` The first step is to create a new footnote apparatus.

`\newfootnotes {`⟨*prefix*⟩`}`

The argument ⟨*prefix*⟩ can be any sequence of letters, such as 'a' or 'dog', except 'foot' and other names that have already been taken. For example, executing `\newfootnotes{a}` creates a number of new commands, the names of which all begin with 'a-': `\anotemark`, `\anotetext`, `\anote`, and `\anotecontent`.

`\<prefix>notemark` `\<prefix>notetext` The first two correspond to LaTeX `\footnotemark`, and `\footnotetext`. They also produce a footnote, or rather, a mark without a footnote, and a footnote without a mark. But the important difference to their analogues is that they do not have the text of the footnote as arguments. They take a *identifier* instead.

`\`⟨*prefix*⟩`notemark {`⟨*id*⟩`}`
`\`⟨*prefix*⟩`notetext {`⟨*id*⟩`}`

`\<prefix>notecontent` The identifier ⟨*id*⟩ identifies the text of a footnote. It can be any arbitrary sequence of letters, numbers, space and punctuation marks. But it is not itself the text of the footnote; it is more like a tag, or a name. This naming is brought about by:

`\`⟨*prefix*⟩`notecontent {`⟨*id*⟩`} {`⟨*text*⟩`}`

`\<prefix>note` And once ⟨*text*⟩ bears a name it can be identified by `\`⟨*prefix*⟩`note` to produce a footnote.

`\`⟨*prefix*⟩`note {`⟨*id*⟩`}`

Example. How to typeset 'Aristotle's master' as a footnote.

```
\newfootnotes{a}
\anotecontent{Plato}{Aristotle's master.}

This was first brought up by
the great Plato\anote{Plato}
in the \textit{Sophist}.
```

This was first brought up by the great Plato[1] in the *Sophist*.

[1] Aristotle's master.

That sentence will take three lines of code however long the footnote to Plato is. No pun intended.

1.2 Endnotes

\newendnotes
\<prefix>notes

Endnote and footnote commands are essentially the same but for two differences. To use endnotes first substitute `\newendnotes` for `\newfootnotes`, and then insert `\`⟨*prefix*⟩`notes`, plural, at the end of the document.

`\newendnotes {`⟨*prefix*⟩`}`
`\`⟨*prefix*⟩`notes`

Example. How to effortlessly convert all footnotes into endnotes.

```
\newendnotes{a}
\anotecontent{Plato}{Aristotle's master.}

This was first brought up by
the great Plato\anote{Plato}
in the \textit{Sophist}.

\section*{Notes}
\anotes
```

1.3 Symbol footnotes

\newsymbolfootnotes

To get footnotes with symbols, such as *, †, and ‡, instead of numbers, just declare `\newsymbolfootnotes` instead of `\newfootnotes`. (See section 2.3 for notes with Roman numerals or letters.)

`\newsymbolfootnotes [`⟨*master counter*⟩`] {`⟨*prefix*⟩`}`

The optional argument ⟨*master counter*⟩ can be anything like `page`, `section`, `chapter`, etc. The default master counter is `page`. The series of symbols is reset each new page, so that * marks the first symbol footnote on any page, whereas † marks the second, ‡ the third, and so on.

```
\newsymbolfootnotes[page]{a}
\anotecontent{Plato}{Aristotle's master.}

This was first brought up by
the great Plato\anote{Plato}
in the \textit{Sophist}.
```

No need to modify the main body. Symbol footnotes are just footnotes using the same commands as before.

1.4 Footnotes in separate file

It is now easy to see how footnotes can be stored in a separate file. It is just a matter of moving all \⟨*prefix*⟩`notecontent` declarations to a new file, say, `notes.tex`, and then loading it in the preamble with `\input`.

Example. How to move all footnotes to a separate file.

This is `notes.tex`

```
\anotecontent{Homer}{Everybody's master.}
\anotecontent{Plato}{Aristotle's master.}
```

And here is `document.tex`

```
\documentclass{article}
\usepackage{sepfootnotes}
\newfootnotes{a}
\input{notes.tex}
\begin{document}
 This was first brought up by
 the great Plato\anote{Plato}
 in the \textit{Sophist}. But
 nobody could suppose that an
 antecedent is to be found in
 Homer.\anote{Homer}
\end{document}
```

Notice that `Homer` is the first entry in `notes.tex` though it will be typeset as note 2, whereas `Plato` is the first footnote here but the second entry there.

> This was first brought up by the great Plato[1] in the *Sophist*. But nobody could suppose that an antecedent is to be found in Homer.[2]
>
> [1] Aristotle's master.
> [2] Everybody's master.

Fortunately, the order in which content declarations appear in `notes.tex` is **not** particularly important when it comes to numbering footnotes. Numbering depends solely upon the order in which \⟨*prefix*⟩`note` commands appear in the document.

Content declarations can be either randomly inserted into `notes.tex`, or more interestingly perhaps, grouped together by similarity of subject or size.

2 Advanced Usage

2.1 Footnotes and endnotes together

If a writer wants to insert footnotes and endnotes in the same document, all he has to do is declare `\newfootnotes {`⟨*prefix*$_1$⟩`}` and `\newendnotes {`⟨*prefix*$_2$⟩`}` in the preamble. Each `\new...notes` creates an independent note apparatus with its own counter, `\`⟨*prefix*⟩`note`, `\`⟨*prefix*⟩`notecontent`, etc.

The trick though is to give each apparatus a different prefix, say, 'a' and 'b':

```
\newfootnotes{a}
\newendnotes{b}
```

and then substitute prefix 'b' for 'a' whenever a note is to be identified by the `b` apparatus. In our example if `Plato` is a footnote, but `Homer` is to be made into an endnote, just replace `\anotecontent{Homer}` with `\bnotecontent{Homer}` in `notes.tex`

```
\bnotecontent{Homer}{Everybody's master.}
\anotecontent{Plato}{Aristotle's master.}
```

And do the same to `\anote{Homer}` and `\bnote{Homer}` in `document.tex`. But don't forget the `\bnotes`, plural, at the end.

```
\newfootnotes{a}
\newendnotes{b}
\input{notes.tex}
\begin{document}
 This was first brought up by
 the great Plato\anote{Plato}
 in the \textit{Sophist}. But
 nobody could suppose that an
 antecedent is to be found in
 Homer.\bnote{Homer}
\section*{Notes}
\bnotes
\end{document}
```

Here `Plato` will be typeset as a footnote, whereas `Homer` as an endnote.

Now imagine that, for whatever reason, a writer or editor needs footnotes, endnotes, and symbol footnotes all together in the same document. He just has to declare three independent note apparatus taking three distinct prefixes:

```
\newfootnotes{a}
\newendnotes{b}
\newsymbolfootnotes{c}
```

and then use `\anote` and `\anotecontent` for footnotes, but reserve `\bnote` and `\bnotecontent` for endnotes, as well as `\cnote` and `\cnotecontent` for symbol footnotes.

2.2 Cross-references

`\<prefix>noteref` Footnotes and endnotes can be referenced to with `\`⟨*prefix*⟩`noteref`.

`\`⟨*prefix*⟩`noteref {`⟨*id*⟩`}`.

Example. How to produce the number of that footnote to Plato.

```
This was first brought up by
the great Plato\anote{Plato}
in the \textit{Sophist}.

See note~\anoteref{Plato}.
```

It is okay to use LaTeX `\label` and `\ref` to refer the reader to notes created by `sepfootnotes`. However, because both `\label`, and `\`⟨*prefix*⟩`note`, assign a unique identifier—or label, tag, marker, or name—to a particular note number, it may in most cases be just superfluous to label it twice. Compare:

```
This was first brought up by
the great Plato\anote{Plato}\label{Plato}
in the \textit{Sophist}.

See note~\ref{Plato}.
```

But imagine our writer would rather stick to `\ref`. Since the labels `sepfootnotes` internally assigns to note numbers all have names like `Notes@refs@`⟨*prefix*⟩`@`⟨*id*⟩, the footnote number to `Plato` in our example may also be typeset, though somewhat clumsily, by `\ref{Notes@refs@a@Plato}`.

2.3 Roman numerals and letters

`\the<prefix>` To change the style of footnote or endnote marks, simply redefine `\the`⟨*prefix*⟩ to use another style: `roman`, `Roman`, `alph`, `Alph`, or `fnsymbol`.

`\renewcommand\the`⟨*prefix*⟩ `{\`⟨*style*⟩ `{`⟨*prefix*⟩`}}`

Example. How to typeset lowercase Roman numerals in footnotes.

```
\renewcommand\thea{\roman{a}}

This was first brought up by
the great Plato\anote{Plato}
in the \textit{Sophist}.
```

This was first brought up by the great Plato[i] in the *Sophist*.

[i] Aristotle's master.

2.4 Endnotes customization

To customize how the text of endnotes is typeset at the end of the document, you may redefine \⟨*prefix*⟩`endnotemark` and the ⟨*prefix*⟩`endnotes` environment.

`\<prefix>endnotemark` Numbers right before the text of endnotes are typeset by \⟨*prefix*⟩`endnotemark`, which is not to be confused with `\the`⟨*prefix*⟩. See section 2.3.

\⟨*prefix*⟩`endnotemark {`⟨*number*⟩`}`

`<prefix>endnotes` The text of all endnotes is typeset within the ⟨*prefix*⟩`endnotes` environment.

Example. How to typeset endnotes as a list of numbers, instead of numbered paragraphs, and in boldface.

```
\newendnotes{a}
\anotecontent{Homer}{Everybody's master.}
\anotecontent{Plato}{Aristotle's master.}

\renewcommand     \aendnotemark [1] {\item[\textbf{#1.}]}
\renewenvironment {aendnotes}       {\begin{list}{}{}}
                                    {\end{list}}
This was first brought up by
the great Plato\anote{Plato}
in the \textit{Sophist}. But
nobody could suppose that an
antecedent is to be found in
Homer.\anote{Homer}
\section*{Notes}
\anotes
```

This was first brought up by the great Plato[1] in the *Sophist*. But nobody could suppose that an antecedent is to be found in Homer.[2]

Notes

1. Aristotle's master.
2. Everybody's master.

References

[1] At `http://tug.org/mailman/htdig/texhax/2010-April/014558.html`

[2] At `http://www.44342.com/tex-f809-t9440-p1.htm`.

[3] Frank Mittelbach, and Michel Goossens, with Johannes Braams, David Carlisle, and Chris Rowley. *The LaTeX Companion*. 2nd Addison Wesley, 2004. ISBN 0-201-36299-6.

3 Implementation

Namespace

First we define a namespace for labels and identifiers.

```
\newcommand\@SFnNamespace{Notes@}
```

3.1 Public macros

This section comments on public macros dynamically named, and defined, using a user provided prefix. They fall into four categories: footnote macros, endnote macros, symbol footnote macros, and macros that are common to all of them.

3.1.1 Common macros

`\@SFnNew` The public `\newfootnotes`, `\newsymbolfootnotes`, and `\newendnotes` commented upon in the next sections first call the private `\@SFnNew {`⟨*prefix*⟩`}`, which dynamically defines a counter and three public macros common to both footnotes and endnotes. They are all named with ⟨*prefix*⟩: if 'bar' is provided as the prefix then `\barnotecontent`, `\barnoteref`, `\barnmark`, and the counter `bar` are defined.

```
\newcommand\@SFnNew [1]{%
```

The counter ⟨*prefix*⟩ will only track notes typeset by macros named with ⟨*prefix*⟩.

```
  \newcounter{#1}%
```

`\<prefix>notecontent` The macro `\`⟨*prefix*⟩`notecontent {`⟨*id*⟩`} {`⟨*text*⟩`}` stores ⟨*text*⟩ as ⟨*id*⟩, which is in the ⟨*prefix*⟩ namespace itself.

```
  \expandafter
  \newcommand\csname #1notecontent\endcsname [2]
  {\@SFnStoreText {#1} {##1} {##2}}%
```

`\<prefix>noteref` The macro `\`⟨*prefix*⟩`noteref {`⟨*id*⟩`}` typesets the reference to the label associated with ⟨*id*⟩. See `\`⟨*prefix*⟩`notemark` in 3.1.2 below.

```
  \expandafter
  \newcommand\csname #1noteref\endcsname [1]
  {\@SFnGetReference {#1} {##1}}%
```

`\<prefix>nmark` Finally `\`⟨*prefix*⟩`nmark` typesets the default note mark.

```
  \expandafter
  \newcommand\csname #1nmark\endcsname
  {\csname the#1\endcsname}}%
```

3.1.2 Footnotes

`\newfootnotes` The macro `\newfootnotes {`⟨*prefix*⟩`}` dynamically defines three public macros. They also are all named with ⟨*prefix*⟩.

```
\newcommand\newfootnotes [1]
  {\@SFnNew{#1}%
```

\<prefix>notemark The macro \⟨*prefix*⟩notemark {⟨*id*⟩} steps the counter, assigns it a label, links that label up with ⟨*id*⟩, and typesets a footnote mark. See \@SFnInsertMark.

```
\expandafter
\newcommand\csname #1notemark\endcsname [1]
{\@SFnInsertMark   {#1} {##1}}%
```

\<prefix>notetext The macro \⟨*prefix*⟩notetext ⟨*id*⟩ typesets the text that was previously associated with ⟨*id*⟩ by \⟨*prefix*⟩notecontent.

```
\expandafter
\newcommand\csname #1notetext\endcsname [1]
{\@SFnSetMark     {#1}%
 \@SFnTypesetText {#1} {##1}}
```

\<prefix>note The macro \⟨*prefix*⟩note ⟨*id*⟩ typesets the mark and text of a footnote.

```
\expandafter
\newcommand\csname #1note\endcsname [1]
{\csname #1notemark\endcsname{##1}%
 \csname #1notetext\endcsname{##1}}
```

3.1.3 Symbol footnotes

\newsymbolfootnotes The macro \newsymbolfootnotes [⟨*master counter*⟩] {⟨*prefix*⟩} dynamically define all of the footnote macros commented upon in 3.1.2, but it also redefines \⟨*prefix*⟩nmark so as to render symbols instead of numbers. Further, it creates a counter to be reset when ⟨*master counter*⟩ is stepped. The default master counter is page.

```
\newcommand\newsymbolfootnotes [2][page]
{\newfootnotes {#2}%
 \@addtoreset  {#2} {#1}%
 \expandafter
 \renewcommand\csname #2nmark\endcsname
 {\fnsymbol {#2}}}
```

3.1.4 Endnotes

Endnote identifiers, but not endnote number and text, need to be stored in a file for later processing. The list of endnote identifiers may then be converted into endnotes at the end of the document within a suitable environment.

The following is similar to 3.1.2 on footnotes but for the fact that now we have to deal with files, and delayed typesetting of endnote text. Another difference to the footnote set is that \⟨*prefix*⟩notemark and \⟨*prefix*⟩note are the same here.

\newendnotes The macro \newendnotes {⟨*prefix*⟩} dynamically defines a corresponding environment ⟨*prefix*⟩endnotes, as well as five public macros available to endnotes only. They also are all named with ⟨*prefix*⟩.

```
\newcommand\newendnotes [1]
{\@SFnNew         {#1}%
 \@SFnOpenFileOut {#1}%
```

`\<prefix>note`
`\<prefix>notemark`

Both `\`⟨*prefix*⟩`note {`⟨*id*⟩`}` and `\`⟨*prefix*⟩`notemark {`⟨*id*⟩`}` typeset an endnote mark, and write down its ⟨*id*⟩ for later processing.

```
\expandafter
\newcommand\csname #1notemark\endcsname [1]
{\@SFnInsertMark    {#1} {##1}%
 \@SFnWriteToFile   {#1} {##1}}%
\expandafter
\newcommand\csname #1note\endcsname [1]
{\csname #1notemark\endcsname {##1}}%
```

`\<prefix>notetext`

Though public, `\`⟨*prefix*⟩`notetext {`⟨*id*⟩`}` is not meant to be directly used when applied to endnotes. It typesets a single endnote mark and text, preferably within the ⟨*prefix*⟩`endnotes` environment.

```
\expandafter
\newcommand\csname #1notetext\endcsname [1]
{\csname #1endnotemark\endcsname
 {\@SFnGetReference {#1} {##1}}%
 \@SFnRetrieveText  {#1} {##1}\par}%
```

`\<prefix>notes`

The `\`⟨*prefix*⟩`notes` immediately closes the auxiliary file, and typesets all endnotes within a ⟨*prefix*⟩`endnotes` environment.

```
\expandafter
\newcommand\csname #1notes\endcsname
{\@SFnCloseFile {#1}%
 \begin{#1endnotes}
 \input\@SFnFileName{#1}%
 \end{#1endnotes}}%
```

`\<prefix>endnotemark`

Now, the macro `\`⟨*prefix*⟩`endnotemark {`⟨*number*⟩`}` typesets marks preferably within the ⟨*prefix*⟩`endnotes` environment.[2]

```
\expandafter
\newcommand\csname #1endnotemark\endcsname [1]
{\noindent\makebox[0pt][r]{\mbox{{\normalfont ##1.\,}}}}
```

`<prefix>endnotes`

And here is the environment where to typeset all endnotes.

```
\newenvironment{#1endnotes}
 {\footnotesize\setlength\parskip\footnotesep}
 {}%
```

[2] From [3, p.114].

3.2 Private macros

This section comments on private macros.

3.2.1 Files

The following apply to endnotes only. See 3.1.4.

`\@SFnFileName` The macro `\@SFnFileName {`⟨*prefix*⟩`}` returns a filename based on, surprise, ⟨*prefix*⟩. `Todo:` 3 letter extension: `sep`.

```
\newcommand\@SFnFileName [1]
 {\jobname.notes-#1}
```

`\@SFnOpenFileOut` `\@SFnWriteToFile` `\@SFnCloseFile` Here are three macros that operate on file descriptors:

The first, `\@SFnOpenFileOut {`⟨*prefix*⟩`}`, opens for writing a dynamically named file `\@SFnFileOut`⟨*prefix*⟩.

```
\newcommand\@SFnOpenFileOut [1]
 {\expandafter\newwrite\csname @SFnFileOut#1\endcsname
  \immediate\expandafter\openout
  \csname @SFnFileOut#1\endcsname=\@SFnFileName#1\relax}
```

The macro `\@SFnWriteToFile {`⟨*prefix*⟩`} {`⟨*id*⟩`}` writes down a note identifier, ⟨*id*⟩, for later processing. See `\`⟨*prefix*⟩`notes` in section 3.1.4.

```
\newcommand\@SFnWriteToFile [2]
 {\immediate\write\csname @SFnFileOut#1\endcsname
  {\expandafter\string\csname #1notetext\endcsname
  {#2}}}%
```

And `\@SFnCloseFile {`⟨*prefix*⟩`}` closes the file containing note identifiers related to ⟨*prefix*⟩.

```
\newcommand\@SFnCloseFile [1]
 {\immediate\expandafter
  \closeout\csname @SFnFileOut#1\endcsname\relax}
```

3.2.2 Text

These are private macros to store, retrieve, and typeset the text of notes. They apply to both footnotes and endnotes.

`\@SFnStoreText` First `\@SFnStoreText {`⟨*prefix*⟩`} {`⟨*id*⟩`} {`⟨*text*⟩`}` stores the text of a note as ⟨*id*⟩ in the ⟨*prefix*⟩ `text` namespace. The package aborts if ⟨*id*⟩ is already in use in that namespace.

```
\newcommand\@SFnStoreText [3]
 {\@ifundefined{\@SFnNamespace text@#1@#2}
  {\@SFnNameDef {#1} {#2} {#3}}
  {\PackageError
   {\@SFnPackageName}
   {`#1' is already in use.}
   {The note identifier `#1' already identifies a piece of
    text.\MessageBreak Solution: Use another identifier.}}}
```

`\@SFnRetrieveText` And `\@SFnRetrieveText {`⟨*prefix*⟩`} {`⟨*id*⟩`}` retrieves a previously stored piece of text identified by ⟨*id*⟩ in the ⟨*prefix*⟩ `text` namespace. It silently ignores undefined content.

```
\newcommand\@SFnRetrieveText [2]
 {\@ifundefined{\@SFnNamespace text@#1@#2}
  {}
  {\@SFnNameUse {#1} {#2}}}
```

`\@SFnTypesetText` Finally, `\@SFnTypesetText {`⟨*prefix*⟩`} {`⟨*id*⟩`}` not only retrieves but also typesets a note text identified by ⟨*id*⟩ in the ⟨*prefix*⟩ `text` namespace.

```
\newcommand\@SFnTypesetText [2]
 {\@footnotetext{\@SFnRetrieveText {#1} {#2}}}
```

3.2.3 Marks

These are private macros to typeset note marks. They apply to both footnotes and endnotes.

`\@SFnInsertMark` The macro `\@SFnInsertMark {`⟨*prefix*⟩`} {`⟨*id*⟩`}` steps the counter, assigns it a label, links that label up with ⟨*id*⟩, and then typesets a note mark. It does the real work behind `\`⟨*prefix*⟩`notemark` in sections 3.1.2 and 3.1.4.

```
\newcommand\@SFnInsertMark [2]
 {\@SFnStepCounter  {#1}%
  \@SFnSetReference {#1} {#2}%
  \@SFnSetMark      {#1}%
  \@SFnTypesetMark}
```

`\@SFnSetMark` Now, `\@SFnSetMark {`⟨*prefix*⟩`}` sets[3] LaTeX internal note engine to typeset a mark as publicly defined by `\`⟨*prefix*⟩`nmark`.

```
\newcommand\@SFnSetMark [1]
 {\protected@xdef\@thefnmark{\csname #1nmark\endcsname}}
```

`\@SFnTypesetMark` And `\@SFnTypesetMark` is just a more descriptive wrapper to the internal macro that typesets the note mark.

```
\newcommand\@SFnTypesetMark
 {\@footnotemark}
```

3.2.4 Labels and cross-references

This package relies heavily on labels and cross-references to keep track of endnote numbers. But every note number, be it footnote or endnote, is assigned a label in the ⟨*prefix*⟩ `label` namespace. Thus note numbers can be directly referenced to with `\ref{Notes@refs@`⟨*prefix*⟩`@`⟨*id*⟩`}`, or indirectly, but more simply, with `\`⟨*prefix*⟩`noteref` ⟨*id*⟩.

On the one hand, labels require running `latex` at least twice to get cross-references right. But on the other they allow us not to bother with note numbers.

[3] From LaTeX Kernel (Floats), 2002/10/01 v1.1v.: `base/ltfloat.dtx`.

As long as note text and labels are implicitly connected by a unique identifier ⟨*id*⟩ in ⟨*prefix*⟩ namespace, the explicit connection between note content and number does not need to be hard coded into the endnote auxiliary file, as it is in the `endnotes` package.

But the motivation for the identifier approach—notes defined before notes inserted—is quite independent of such a feature, if it is a feature.

`Todo:` However it be, the identifier approach might prove itself useful in solving LaTeX misnumbering footnotes when two or more `\footnotemark` precedes two or more `\footnotetext`. Since the `sepfootnotes` package requires note text to be defined before note marks are inserted, even in a separate file, it no longer matters how many times `\`⟨*prefix*⟩`notemark` ⟨*id*⟩ is invoked before `\`⟨*prefix*⟩`notetext` ⟨*id*⟩. That is precisely what goes on with endnotes.

`\@SFnSetReference` The macro `\@SFnSetReference {`⟨*prefix*⟩`} {`⟨*id*⟩`}` assigns a note number a label in the ⟨*prefix*⟩ `refs` namespace.

```
\newcommand\@SFnSetReference [2]
 {\label{\@SFnNamespace refs@#1@#2}}
```

`\@SFnGetReference` And `\@SFnGetReference {`⟨*prefix*⟩`} {`⟨*id*⟩`}` simply returns that number.

```
\newcommand\@SFnGetReference [2]
 {\ref{\@SFnNamespace   refs@#1@#2}}
```

3.2.5 Counters

`\@SFnStepCounter` The macro `\@SFnStepCounter {`⟨*prefix*⟩`}` steps the counter `prefix`, and makes sure it can be properly labeled afterwards.

```
\newcommand\@SFnStepCounter [1]
 {\refstepcounter{#1}}
```

3.2.6 Names

`\@SFnNameDef` The macro `\@SFnNameDef {`⟨*prefix*⟩`} {`⟨*id*⟩`} {`⟨*text*⟩`}` stores ⟨*text*⟩ as ⟨*id*⟩ in the ⟨*prefix*⟩ `text` namespace.[4]

```
\newcommand\@SFnNameDef [3]
 {\@namedef{\@SFnNamespace text@#1@#2}{#3}}
```

`\@SFnNameUse` And `\@SFnNameUse {`⟨*prefix*⟩`} {`⟨*id*⟩`}` retrieves a previously stored piece of text identified by ⟨*id*⟩ in the ⟨*prefix*⟩ `text` namespace. Undefined content is silently ignored.

```
\newcommand\@SFnNameUse [2]
 {\@nameuse{\@SFnNamespace text@#1@#2}}
```

[4] I owe Dan Luecking the idea of using LaTeX internal table to store note text. [2]