

The fancyhdr and extramarks packages

version v4.0.

Piet van Oostrum*
Dept. of Computer Science[†]
Utrecht University

March 27, 2019

Abstract

This document describes how to customize the page layout of your LaTeX documents, i.e how to change page margins and sizes, headers and footers, and the proper placement of figures and tables (collectively called floats) on the page.

This documentation describes version 4.0 or later of the `fancyhdr` and `extramarks` packages. The user documentation is valid for the versions 3.8 or later of the `fancyhdr` package (except for the changes mentioned in section 32.1), and version 2.1 or later of the `extramarks` package.

Contents

I	Introduction	2
1	Installation	3
2	Using <code>fancyhdr</code>	3
3	Using <code>extramarks</code>	4
II	Page Layout in L^AT_EX	6
4	Introduction	6
5	Page headers and footers	6
6	What is <code>fancyhdr</code>	8
7	Simple use of <code>fancyhdr</code>	9
8	A simple example	9
9	Fancy Centering	10
10	An example of two-sided printing	11
11	Redefining page style <code>plain</code>	12

*A considerable part of this documentation was written by George Grätzer (University of Manitoba) in *Notices Amer. Math. Soc.* Thanks, George!

[†]This was my employer at the time I developed this package. I am now retired.

12	Defining other page styles	13
13	Switching page styles	15
14	Package options	15
15	The default layout	17
16	The scoop on L ^A T _E X's marks	18
17	Dictionary style headers	21
18	Fancy layouts	21
19	Two book examples	24
20	Special page layout for float pages	26
21	Those blank pages	27
22	N of M style page numbers	28
23	Chapter or section related page numbers	28
24	When to change the headers and footers?	29
25	Headers and footers induced by the text	33
26	A movie	36
27	Thumb-indexes	36
28	Float placement	37
29	Multipage Floats	40
30	Deprecated commands	42
31	Contact information	43
32	Version information	43
 III Questions & Answers		45
33	Large chapter/section titles	45
34	I lost my chapter/section titles	48
 IV Implementation		49
35	fancyhdr.sty	49
36	extramarks.sty	64
37	fancyheadings.sty	65

Part I

Introduction

This document contains four parts:

Part I is a short documentation on the user commands of the fancyhdr and extramarks packages.

Part II contains elaborate documentation on page layout in L^AT_EX. This used to be the complete documentation of fancyhdr and extramarks for several years.

Part III contains Questions and Answers.

Part IV contains the annotated implementation.

1 Installation

The preferred way to install this package is with a package installer. If you want to install it by hand, then first run the command `tex fancyhdr.ins` and then move the files `fancyhdr.sty`, `extramarks.sty` and `fancyheadings.sty` to a place where L^AT_EX can find it, preferably in a directory similar to `.../texmf/tex/latex/fancyhdr/` in your T_EX directory tree.

2 Using fancyhdr

The package `fancyhdr` gives you several commands to define headers and footers of the pages in a L^AT_EX document. You load the package with the following command in the preamble:

```
\usepackage[options]{fancyhdr}
```

(Options are available since version 4.0) The following options are supported:

Option	Meaning
<code>nocheck</code>	do not check the heights of the header and footer (see section 18 on page 22)
<code>compatV3</code>	keep some behaviour (now considered undesirable) as in version 3 (see section 14 and section 18 on page 22)
<code>headings</code>	redefine the <code>headings</code> page style to be fancy-based
<code>myheadings</code>	redefine the <code>myheadings</code> page style to be fancy-based

```
\fancyhead
\fancyfoot
\fancyhf
```

```
\fancyhead[places]{field}
\fancyfoot[places]{field}
\fancyhf[places]{field}
```

Here `places` is a comma-separated list of places where `field` will be placed. There are 12 places defined: Left, Center and Right Headers and Footers, and both can be on Even or Odd pages. Each place therefore has 3 coordinates which are the initial letters of the above description: (1) E or O, (2) L, C or R, (3) H or F. So a place is given with 3 letters, like EOH. A missing coordinate means: all possibilities, except for `\fancyhead` where H is implied and `\fancyfoot` where F is implied.

```
\fancyheadoffset
\fancyfootoffset
\fancyhfoffset
```

```
\fancyheadoffset[places]{field}
\fancyfootoffset[places]{field}
\fancyhfoffset[places]{field}
```

These define offsets to let the headers stick into the margin (or to the inside if negative). Places cannot contain the C specifier. See section 19 for more details.

```
\headrulewidth
\headrulewidth
\headruleskip
\footruleskip
\headrule
\footrule
\headwidth
```

`\headrulewidth` and `\headrulewidth` are macros to define the thickness of a line under the header and above the footer. `\headruleskip` and `\footruleskip` are macros that define the distance between the lines and the header and footer text, respectively. (But `\headruleskip` is only available since version 4.0.) `\headrule` and `\footrule` are macros to completely redefine these lines. And `\headwidth` is a length parameter that defines the total width of the headers and footers. See sections 8, 18 and 19 for more details.

`\fancyheadinit`
`\fancyfootinit`
`\fancyhfini` `\fancyheadinit` and `\fancyfootinit` can be used to define initialisation code for the header and footer, respectively, and `\fancyhfini` defines both of these. These commands are only available in fancyhdr version 4.0 and later. See section 24.

`\fancycenter` (Only in version 4.0 and later.) The command `\fancycenter` packs 3 header fields into a full-width header. See section 9.

`\iftopfloat`
`\ifbotfloat`
`\iffloatpage`
`\iffootnote` The macros `\iftopfloat`, `\ifbotfloat`, `\iffloatpage`, `\iffloatpage`, `\iffootnote` are used to detect if there is a float on the top or the bottom of the page, or the page is a float page, or if there is a footnote at the bottom of the page. These can be used to choose different headers and/or footers if these conditions are met. See section 20 for more details.

`\fancypagestyle` `\fancypagestyle{<style-name>}[<base-style>]{<definitions>}`

This command lets you (re)define page styles for use in special situations. See section 12 for more details.

3 Using extramarks

The `extramarks` gives you some extra marks in L^AT_EX, besides the normal `\leftmark` and `\rightmark`, that are defined by the `\markboth` and `\markright` commands.

`\firstleftmark`
`\firstleftmark`
`\firstrightmark`
`\lastleftmark`
`\firstleftmark`
`\lastrightmark`
`\firstrightmark`
`\lastleftmark`

Standard L^AT_EX has two marks: a left and a right one. The standard command `\leftmark` gives you the last left mark on a page, and `\rightmark` gives you the first right one. These macros give you also the other combinations, where `\firstrightmark = \rightmark` and `\lastleftmark = \leftmark`. As with the standard marks, these are meant to be used in headers and footers. In other places they will not work properly.

`\extramarks{aa}{bb}`
`\firstleftxmark`

```
\firstrightxmark  
\topleftxmark  
\toprightxmark  
\lastleftxmark  
\lastrightxmark  
\firstxmark  
\lastxmark  
\topxmark
```

The command `\extramarks{aa}{bb}` defines two extra marks, similar to the standard ones by L^AT_EX, where `aa` is the left one and `bb` is the right one. The other commands are to extract these in the headers and footers, similar to the ones without the `x`. See sections 16 and 25 for more details.

Part II

Page Layout in L^AT_EX

4 Introduction

A page in a L^AT_EX document is built from various elements as shown in figure 1.

The body contains the main text of the document together with the so called floats (tables and figures).

The pages are constructed by L^AT_EX's output routine, which is quite complicated and should therefore not be modified. Some of the packages described in this paper contains small modifications to the output routine to accomplish things that cannot be done in another way. You should use these packages to get the desired result rather than fiddling with the output routine yourself.

There are a number of things that you must be aware of:

1. The margins on the left are not called `\leftmargin`, but `\evensidemargin` (on even-numbered pages) and `\oddsidemargin` (on odd-numbered pages). In one-sided documents `\oddsidemargin` is used for either. `\leftmargin` is also a valid L^AT_EX parameter but it has a different use (namely the indentation of lists).
2. Most of the parameters should not be changed in the middle of a document. Some changes might work at a pagebreak. If you want to change the height of a single page, you can use the `\enlargethispage` command.

The margin notes area contains small pieces of information created by the `\marginpar` command. On twosided documents the margin notes appear on the left and right alternatively. The margin notes are not on fixed places with respect to the paper but at approximately the same height as the paragraph in which they appear. Due to the algorithm used to decide the placement of margin notes, in a twosided document unfortunately they may appear on the wrong side if they are close to a page break. If you want to put information on fixed places in the margins you may use the technique described in sections 26 and 27.

The first part of this paper describes how to change the header and footer areas. The last part describes how to get your floats at the desired place.

5 Page headers and footers

The page headers and footers in L^AT_EX are defined by the `\pagestyle` and `\pagenumbering` commands. `\pagestyle` defines the general contents of the headers and footers (e.g. where the page number will be printed), while `\pagenumbering` defines the format of the page number. L^AT_EX has four standard page styles:

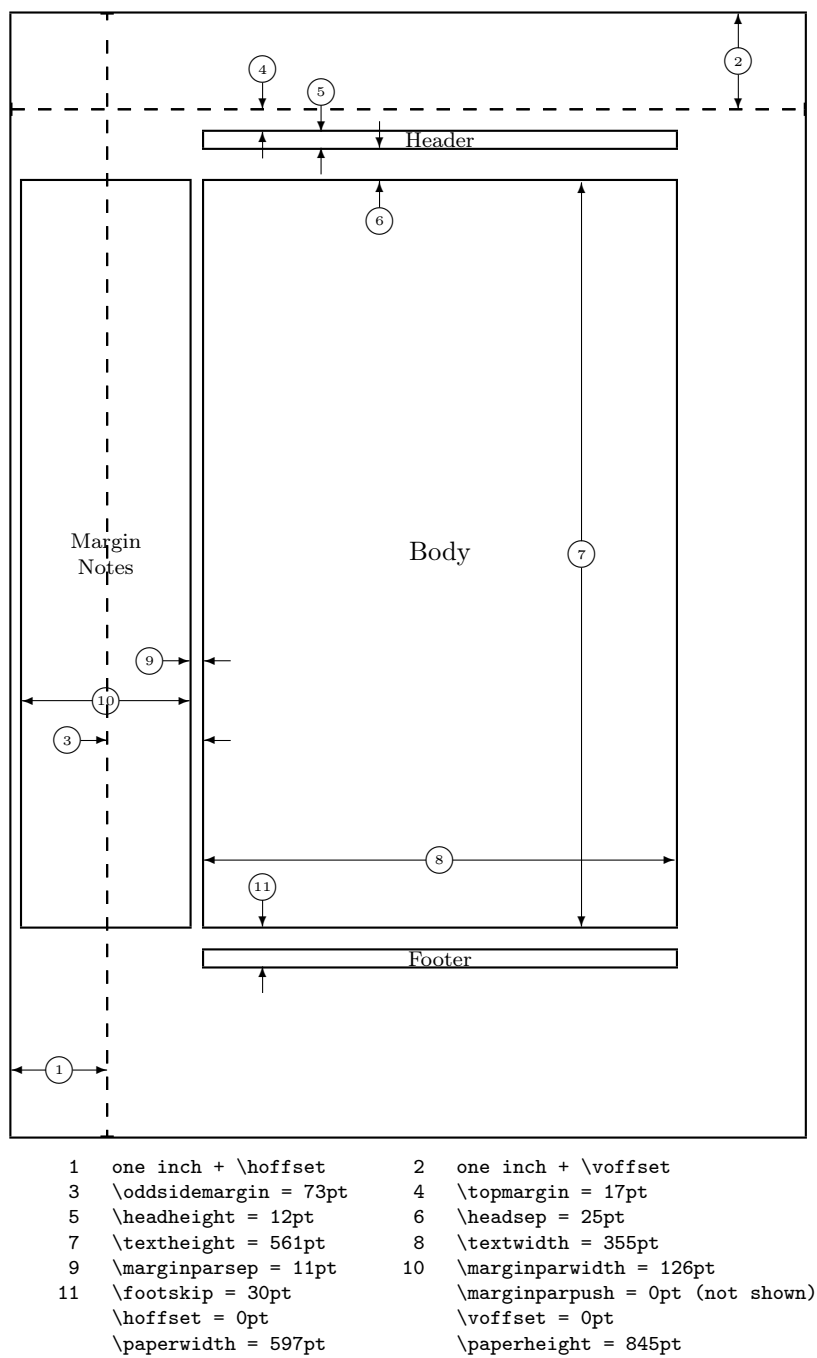


Figure 1: Page elements. The values shown are those in effect in the current document, not the defaults.

<code>empty</code>	no headers or footers
<code>plain</code>	no header, footer contains page number centered
<code>headings</code>	no footer, header contains name of chapter/section and/or subsection and page number
<code>myheadings</code>	no footer, header contains page number and user supplied information

Although these are useful styles, they are quite limited. Additional page styles can be defined by defining commands of the form `\ps@xxx`. This command is executed when a `\pagestyle{xxx}` is given in the document. The `\ps@xxx` command should define the following commands for the contents of the headers and footers:

<code>\@oddhead</code>	header on odd numbered pages in two-sided documents (on all pages in one-sided)
<code>\@evenhead</code>	header on even numbered pages in two-sided documents
<code>\@oddfoot</code>	footer on odd numbered pages in two-sided documents (on all pages in one-sided)
<code>\@evenfoot</code>	footer on even numbered pages in two-sided documents

These are not user commands, but rather “variables” that are used by L^AT_EX’s output routine. As the command names contain the character ‘@’, they should be defined in a package file, or otherwise be sandwiched between the commands `\makeatletter` and `\makeatother`.

The `\pagenumbering` command defines the layout of the page number. It has a parameter from the following list:

<code>arabic</code>	arabic numerals
<code>roman</code>	lower case roman numerals
<code>Roman</code>	upper case roman numerals
<code>alph</code>	lower case letter
<code>Alph</code>	upper case letter

The `\pagenumbering{xxx}` defines the command `\thepage` to be the expansion of the page number in the given notation `xxx`. The `\pagestyle` command then would include `\thepage` in the appropriate place. Additionally the `\pagenumbering` command resets the page number to 1. The `\pagestyle` and `\pagenumbering` apply to the page that is being constructed, so they should be used at a location where it is clear to what page they apply (see section 24).

6 What is fancyhdr

The `fancyhdr` macro package allows you to customize in L^AT_EX your page headers and footers in an easy way. You can define:

- three-part headers and footers
- decorative lines in headers and footers

- headers and footers wider than the width of the text
- multi-line headers and footers
- separate headers and footers for even and odd pages
- different headers and footers for chapter pages
- different headers and footer on pages with floats

Of course, you also have complete control over fonts, uppercase and lowercase displays, etc.

7 Simple use of fancyhdr

To use this package install it in a place where L^AT_EX can find it (see section 1)¹, and include in the preamble of your document the commands:

```
\usepackage{fancyhdr}
\pagestyle{fancy}
```

We can visualize the page layout we can create with fancyhdr as follows:

LeftHeader	CenteredHeader	RightHeader
page body		
LeftFooter	CenteredFooter	RightFooter

The LeftHeader and LeftFooter are left justified; the CenteredHeader and CenteredFooter are centered; the RightHeader and RightFooter are right justified.

We define each of the six “fields” and the two decorative lines separately.

8 A simple example

K. Grant is writing a report to Dean A. Smith, on “The performance of new graduates” with the following page layout:

The performance of new graduates		
page body		
From: K. Grant	To: Dean A. Smith	3

¹In most modern T_EX installation the package is already included.

where “3” is the page number. The title: “The performance of new graduates” is bold. The rule above the footer is a bit thicker (2pt).

This is accomplished by these commands following `\pagestyle{fancy}`²:

```
\fancyhead[L,C]{}
\fancyhead[R]{\textbf{The performance of new graduates}}
\fancyfoot[L]{From: K. Grant}
\fancyfoot[C]{To: Dean A. Smith}
\fancyfoot[R]{\thepage}
\renewcommand{\headrulewidth}{0.4pt}
\renewcommand{\footrulewidth}{2pt}
```

(The `\thepage` macro displays the current page number. `\textbf` puts it in bold face.)

This is now fine, except that the first page does not need all these headers and footers. To eliminate all but the centered page number, issue the command

```
\thispagestyle{plain}
```

after the `\begin{document}` and the `\maketitle` commands.

Alternatively, issue

```
\thispagestyle{empty}
```

if you do not want any headers or footers.

In fact the standard L^AT_EX classes have the command `\maketitle` defined in such a way that a `\thispagestyle{plain}` is automatically issued. So if you *do* want the fancy layout on a page containing `\maketitle` you must issue a `\thispagestyle{fancy}` after the `\maketitle`.

9 Fancy Centering

Note: This section only applies to `fancyhdr` version 4.0 and later³.

The marks in a fancy header and footer are prepared using `\parbox` command. So, you can use multiline marks. In the header, they are aligned to the bottom line, but, in the footer, they are aligned to the top line. The maximum width of every mark is equal to the `\headwidth`. This can lead to overlapping of neighbouring marks.

`\fancycenter` If you want to prepare marks in more traditional way in a line not exceeding the `\headwidth`, you can use the following command in any mark command:

```
\fancycenter[⟨distance⟩][⟨stretch⟩]
               {⟨left-mark⟩}{⟨center-mark⟩}{⟨right-mark⟩}
```

²Note that version 1 of fancyheadings used the `\setlength` command to change the `\...rulewidth` parameters.

³This comes from the `nccfancyhdr` package by Alexander I. Rozhenko.

This command works like

```
\hbox to\linewidth{{\langle left-mark\rangle}\hfil{\langle center-mark\rangle}\hfil{\langle right-mark\rangle}}
```

but does this more carefully trying to exactly center the central part of the text if possible. The solution for exact centering is applied if the width of $\langle center-mark \rangle$ is less than

$$\text{\linewidth} - 2 * (\text{\stretch} * \text{\langle distance\rangle} + \max(\text{width}(\text{\langle left-mark\rangle}), \text{width}(\text{\langle right-mark\rangle}))).$$

Otherwise the $\langle center-mark \rangle$ will slightly migrate to a shorter item ($\langle left-mark \rangle$ or $\langle right-mark \rangle$), but at least $\langle distance \rangle$ space between all parts of line is provided. The default values of $\langle distance \rangle$ and $\langle stretch \rangle$ are 1em and 3.

If the $\langle center-mark \rangle$ is empty, the `\fancycenter` is equivalent to the following command:

```
\hbox to\linewidth {{\langle left-mark\rangle}\hfil {\langle right-mark\rangle}}
```

You would use this in a header for example with

```
\fancyhead[C]{\fancycenter[\langle distance\rangle][\langle stretch\rangle]
{\langle left-mark\rangle}{\langle center-mark\rangle}{\langle right-mark\rangle}}
```

and leave the [L,R] parts empty.

Note 1: If the whole of the `\fancycenter` is wider than `\headheight` it will stick out on the right. See section 33 for possible solutions.

Note 2: The usage of `\fancycenter` command is not limited to the argument of header/footer marks. You can use it anywhere in your document.

10 An example of two-sided printing

Some document classes, such as `book.cls`, print two-sided by default: the even pages and the odd pages have different layouts; other document classes use the `twoside` option to print two-sided.

Now let us print the report two-sided. Let the above page layout be used for the odd (right-side) pages, and the following for the even (left-side) pages:

The performance of new graduates		
page body		
4	From: K. Grant	To: Dean A. Smith

where “4” is the page number.

Here are the commands:

```
\fancyhead{} % clear all header fields
\fancyhead[RO,LE]{\textbf{The performance of new graduates}}
```

```

\fancyfoot{} % clear all footer fields
\fancyfoot[LE,RO]{\thepage}
\fancyfoot[LO,CE]{From: K. Grant}
\fancyfoot[CO,RE]{To: Dean A. Smith}
\renewcommand{\headrulewidth}{0.4pt}
\renewcommand{\footrulewidth}{0.4pt}

```

The commands `\fancyhead` and `\fancyfoot` have an additional parameter between square brackets that specifies for which pages and/or parts of the header/footer they apply. The first `\fancyhead` command above omits this parameter, and thus applies to all header fields. In general this is only useful to get rid of the defaults or a previous definition, as is done here. Similar the `\fancyfoot` command without square brackets clears all footer fields. In this particular example it could be omitted as all footer fields have a value specified. The selectors that can be used between the square brackets are given in figure 2. Selectors can be combined so `\fancyhead[LE,RO]{text}` will define the field for both the left header on even pages and the right header on odd pages. If you don't give an E or O the definition applies to both. Similar for LRC. The selectors may be given as uppercase or lowercase letters.

E	Even page
O	Odd page
L	Left field
C	Center field
R	Right field
H	Header
F	Footer

Figure 2: Selectors

There is also a more general command `\fancyhf` that you can use to combine the specifications for headers and footers. This allows additional selectors H (header) and F (footer). In fact `\fancyhead` and `\fancyfoot` are just `\fancyhf` with H and F pre-specified, respectively.

Again, you may use `\thispagestyle{plain}` for a simple page layout for page 1.

11 Redefining page style plain

Some L^AT_EX commands, like `\chapter`, use the `\thispagestyle` command to automatically switch to the `plain` page style, thus ignoring the page style currently in effect.

They do this by issuing a `\thispagestyle{plain}` command. The most well-known places where this could happen are:

- The first pages of chapters in the `book` and `report` class

- The first page of a document in the `article` class when `\maketitle` is used
- The first page of an index

but it could happen at other places depending on the class and the packages used.

To customize even such pages you must redefine the `plain` page style. As we indicated before you could do this by defining the `\ps@plain` command, but `fancyhdr` gives you an easier way with the `\fancypagestyle` command. This command can be used to redefine existing page styles (like `plain`) or to define new ones, e.g. if part of your document needs a different page style. This command has two mandatory parameters: the first one is the name of the page style to be defined, the second consists of commands that change the headers and/or footers, i.e. `fancyhead` etc. Also allowed are changes to `\headrulewidth` and `\footrulewidth` or even `\headrule` and `\footrule`. The (re)defined page style uses the standard `fancy` definitions, amended by the definitions in the second parameter. In other words, those parts that are not redefined in the second parameter get their value from the `fancy` definition that is current. In particular, if the second parameter is empty, i.e. given as `{}`, then the new page style is equal to page style `fancy`.

As an example, let us redefine the `plain` style so that it will be the same as page style `fancy`:

```
\fancypagestyle{plain}{}
```

Now when these special pages use the `plain` page style, they use your redefined version.

As another example, let us redefine the `plain` style for the report in Section 10 by making the page number bold and enclosing it in en-dashes without any rules.

```
\fancypagestyle{plain}{%
  \fancyhf{}% clear all header and footer fields
  \fancyfoot[C]{\textbf{--\thepage~--}} % except the center
  \renewcommand{\headrulewidth}{0pt}%
  \renewcommand{\footrulewidth}{0pt}%
}
```

12 Defining other page styles

Just like redefining the `plain` page style in the previous section, you can define or redefine other page styles based on page style `fancy`. This is also done with the `\fancypagestyle` command. The general form of this command is:

```
\fancypagestyle{<style-name>}[<base-style>]{<definitions>}
```

As you see, there is an optional argument between the two mandatory arguments. For example:

```

\fancypagestyle{toc}{%
  \fancyhf{}%
  \fancyhead[R0]{\thepage}%
  \fancyhead[R0]{\textsl{TABLE OF CONTENTS}}%
  \fancyfoot[C]{\thepage}
}

```

This defines a special page style `toc` for use in the table of contents with `\pagestyle{toc}`. Inside the definition you can define the headers and/or footers, change the header and footer rules, and redefine commands like `\chaptermark` (see section 14 for an example). The headers and footers and marks that are not redefined inside the `\fancypagestyle` definition, are taken from the global page style `fancy` values.

You can also give an optional base page style to the `\fancypagestyle` command. Then the new page style will be based on the base style. This base style must be a `fancyhdr`-defined style. Also you should take care not to create circular dependencies. In this case the order of picking up the definitions (headers, footers, marks) is:

1. The definitions from the base style are taken.
2. The definitions given in the `\fancypagestyle` command override and/or augment these.
3. Any definitions that are not given by the two above, are taken from the environment at the time the new page style is used.

Only the first two parts are embedded in the page style. When no base style is given, part 1 is null.

The optional base style argument is only available since version 4.0. In this version it is also possible to redefine page style `fancy` in this way. In version 3.x and earlier this was not possible.

If you want to restore the original default definitions from page style `fancy` as described in section 15, you can use

```

\fancypagestyle{myfancy}[fancydefault]{
  . . . override some here
}

```

Page style `fancydefault` is the version of page style `fancy` that has all the initialisation embedded. Contrary to this, page style `fancy` as defined in the package uses the same defaults, but doesn't have them embedded. It picks them up from the environment. So if the environment changes, because you redefine headers, footers, mark commands, etc, the functioning of page style `fancy` changes with it. The page style `fancydefault` does not change, however. However, `fancydefault` is only available since `fancyhdr` version 4.0.

13 Switching page styles

Page style `fancy`, if not redefined, does not have the definitions of the headers and footers built-in, but they are defined in the document, globally, or locally in a group. This also applies to the definitions of the `\chaptermark` and/or `\[sub]sectionmark` commands. So if you want to switch from another page style to the `fancy` page style later in the document, and that other page style has changed for example the `\chaptermark` and/or `\[sub]sectionmark` commands, you will have to redefine these yourself and maybe also the definitions of the headers and footers, at that point. For example

```
\pagestyle{fancy}
\renewcommand{\chaptermark}[1]{\markboth{Chapter \thechapter. ##1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection\ ##1}}
```

Please note that you have to double the # signs, because the definitions are inside a macro.

If the previous page style was one of the standard L^AT_EX page styles, or some page style that is not based on `fancyhdr`, then the definitions of `\fancyhead` or `\fancyfoot` are not affected. So strictly you don't have to include them. But if it was based on `fancyhdr` and had different definitions, you will get the wrong headers and/or footers when you switch back to page style `fancy`. So it is safer to include them anyway.

A better possibility is to define your own page style, and include these definitions in that page style:

```
\fancypagestyle{myfancy}{
  \renewcommand{\chaptermark}[1]{\markboth{Chapter \thechapter. ##1}{}}
  \renewcommand{\sectionmark}[1]{\markright{\thesection\ ##1}}
  \fancyhead{...}
}
...
\pagestyle{myfancy}
```

In general, when you use only one page style `fancy` in your document, with the occasional `\thispagestyle` excursion to page style `plain` or `empty`, you can just keep the definitions globally in your document, but as soon as you use more than one page style, and switch between them, it is highly advisable to define them (including page style `fancy`) with `\fancypagestyle` and put all the relevant definitions inside them.

14 Package options

NOTE: This section applies to `fancyhdr` version 4.0 and later.

You can supply options to the `\usepackage` command:

```
\usepackage[<options>]{fancyhdr}
```

The following options are supported:

Option	Meaning
<code>nocheck</code>	do not check the heights of the header and footer
<code>compatV3</code>	keep some behaviour (now considered undesirable) as in version 3
<code>headings</code>	redefine the <code>headings</code> page style to be fancy-based
<code>myheadings</code>	redefine the <code>myheadings</code> page style to be fancy-based

- Option `nocheck` is described in section 18 on page 22.
- Option `compatV3` keeps two `fancyhdr` version 3.x (or earlier) features that are now considered undesirable.
 1. The automatic adjustment of `\headheight` or `\footskip` when these are too small. This causes the page layout to become inconsistent. See section 18 on page 22.
 2. In these previous versions the changes to the `fancyhdr` headers and footers (including those by `\fancyhead`, `\fancyheadoffset` and similar commands) are made globally, except within a page style defined by `\fancypagestyle`. That is, when these commands are given inside a \LaTeX group, they affect the whole document, not only the group. If your document depends on this behaviour, you can give the `compatV3` package option. However, this is only considered a short-time solution. You should change your document as soon as possible to work around this problem. In version 4.0 and later, without this option, the changes are always local.

The option is scheduled to disappear in version 5 of `fancyhdr`.

- The options `headings` and `myheadings` redefine the corresponding page style with `fancyhdr` commands (including a decorative line under the header), so that you can later select this page style as the page style for (part of) the document⁴.

The page style `headings` is in some aspects similar to the default page style `fancy` settings. In the `fancy` page style, the page number is in the footer, but in the `headings` page style it is in the header. The header fields look similar, however.

Please note that these page styles redefine the `\chaptermark` and/or `\[sub]sectionmark` commands (see section 16), as do the standard \LaTeX page styles. The consequence is, that if you select e.g. `\pagestyle{headings}`, the definitions of `\pagestyle{fancy}` are overridden. Also when you change the headers and/or footers while such a page style is in effect, and you then switch back to this page style, for example with `\pagestyle{headings}`, they revert to the built-in settings. Therefore it is not advisable to change the headers or footers in this way, but instead define your own page style, as explained in section 12.

⁴These options were copied from the `nccfancyhdr` package, but contrary to that package, they are not automatically selected.

15 The default layout

Let us use the `book.cls` documentclass and the default settings for `fancyhdr`; so we don't use any of the page style options in the `\usepackage{fancyhdr}` command, and we don't redefine any headers or footers. So just:

```
\usepackage{fancyhdr}
\pagestyle{fancy}
```

and let `fancyhdr` take care of everything. As mentioned before, we get a layout that is very similar to the page style `headings`.

On the pages where new chapters start, we get a centered page number in the footer; there is no header, and there are no decorative lines.

On an even page, we get the layout:

<i>1.2 EVALUATION</i>	<i>CHAPTER 1. INTRODUCTION</i>
page body	
2	

On an odd page, we get the layout:

<i>CHAPTER 1. INTRODUCTION</i>	<i>1.2 EVALUATION</i>
page body	
3	

where the header text is slanted uppercase.

This default layout is produced by the following commands:

```
\fancyhead[LE,R0]{\textsl{\rightmark}}
\fancyhead[LO,RE]{\textsl{\leftmark}}
\fancyfoot[C]{\thepage}
```

The following settings are used for the decorative lines:

```
\headrulewidth      0.4pt
\footrulewidth      0 pt
```

The header text is turned into all uppercase by the standard L^AT_EX code in `book.cls`.

16 The scoop on L^AT_EX's marks

Usually, for documents of class `book` and `report`, you may want to use chapter and section information in the headings (chapter only for one-sided printing), and for documents of class `article`, section and subsection information (section only for one-sided printing). L^AT_EX uses a marker mechanism to remember the chapter and section (section and subsection) information for a page; this is discussed in detail in *The L^AT_EX Companion*, Section 4.3.1, or *The L^AT_EX Companion, Second Edition*, Section 4.3.4.

There are two ways you can use and change the higher- and lower-level sectioning information available to you. The macros: `\leftmark` (higher-level) and `\rightmark` (lower-level) contain the information processed by L^AT_EX, and you can use them directly as shown in section 15.

These marks are set by the commands `\markboth{leftmark}{rightmark}` and `\markright{rightmark}`. These commands are usually used inside commands like `\chaptermark` and `\sectionmark` but they can be also be given directly in your document, although this not very usual.

The `\leftmark` contains the **L**eft argument of the *La*st `\markboth` on the page, the `\rightmark` contains the **R**ight argument of the *fi*Rst `\markboth` or the only argument of the *fi*Rst `\markright` on the page. If no marks are present on a page they are “inherited” from the previous page.

You can influence how chapter, section, and subsection information (only two of them!) is displayed by redefining the `\chaptermark`, `\sectionmark`, and `\subsectionmark` commands⁵. You must put the redefinition after the first call of `\pagestyle{fancy}` as this sets up the defaults.

Let us illustrate this with chapter info. It is made up of three parts:

- the number (say, 2), displayed by the macro `\thechapter`
- the name (in English, Chapter), displayed by the macro `\chaptername`
- the title, contained in the argument of `\chapter`.

We combine these below with `\markboth` in `\chaptermark`.

For the lower-level sectioning information, we do the same with `\markright` in `\sectionmark`.

So if “2. Implementation” is the current chapter and “2.1. First steps” is the current section, then

```
\renewcommand{\chaptermark}[1]{%
  \markboth{\chaptername\ \thechapter.\ #1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection.\ #1}}
```

will give “Chapter 2. Implementation” and “2.1. First steps”

Redefining the `\chaptermark` and `\sectionmark` commands may not eliminate all uppercaseness. E.g. the bibliography will have a title of BIBLIOGRAPHY in the header, as the `\MakeUppercase` is explicitly given in the definition of

⁵There are similar commands for `paragraph` and `subparagraph` but they are seldom used.

Code:	Prints:
<pre>\renewcommand{\chaptermark}[1]{% \markboth{\chaptername \thechapter.\ #1}{}}</pre>	Chapter 2. Implementation
<pre>\renewcommand{\chaptermark}[1]{% \markboth{\MakeUppercase{% \chaptername}\thechapter.% \ #1}{}}</pre>	CHAPTER 2. Implementation
<pre>\renewcommand{\chaptermark}[1]{% \markboth{\MakeUppercase{% \chaptername\thechapter.% \ #1}}{}}</pre>	CHAPTER 2. IMPLEMENTATION
<pre>\renewcommand{\chaptermark}[1]{% \markboth{\ #1}{}}</pre>	Implementation
<pre>\renewcommand{\chaptermark}[1]{% \markboth{\thechapter.\ #1}{}}</pre>	2. Implementation
<pre>\renewcommand{\chaptermark}[1]{% \markboth{\thechapter.% \chaptername.\ #1}{}}</pre>	2. Chapter. Implementation

Figure 3: Marker variants

`\thebibliography`. Similar for INDEX etc. If you don't want to redefine these commands, you can use the `\nouppercase` command that `fancyhdr` makes available in the header and footer fields. Note that this may screw other things, like uppercase roman numerals in your headers, so it should be used with care. Essentially this command typesets its argument in an environment where `\MakeUppercase` and `\uppercase` are changed into do-nothing operations.

```
\fancyhead[L]{\nouppercase{\rightmark}}
\fancyhead[R]{\nouppercase{\leftmark}}
```

Figure 3 shows some variants for “Chapter 2. Implementation” (the last example is appropriate in some non-English languages). The % signs at the end of the lines are to prevent unwanted space. Normally you would continue the lines and remove these % signs⁶.

It should be noted that the L^AT_EX marking mechanism works fine with chapters (which always start on a new page) and sections (which are reasonably long). It

⁶The `\MakeUppercase` command is used in L^AT_EX to generate uppercase text, while `\uppercase` is the plain T_EX command for this. The difference is that `\MakeUppercase` also deals with non-ASCII letters.

does not work quite as well with short sections and subsections. This is a problem with L^AT_EX, not with `fancyhdr`.

As an example let's take a page layout where the leftmarks are generated by the sections and the rightmarks by the subsections (as is default in the `article` class). Take a page with some short sections, e.g.

Section 1.

subsection 1.1

subsection 1.2

Section 2.

As the leftmark contains the *last* mark of the page it will be "Section 2.", and the rightmark will be "subsection 1.1" as it will be the *first* mark of the page. So the page header info will combine section 2 with subsection 1.1 which isn't very nice. One thing you can do in these cases is use only the `\rightmarks` and redefine `\sectionmark` accordingly.

However, the `extramarks` package described in section 25 contains a command `\firstleftmark` that can be used to get the first of the leftmarks on the page in the header. This might be the best solution in this situation. Now the header will contain "Section 1." in the situation described above.

```
\usepackage{extramarks}
. . .
\fancyhead[R]{\firstleftmark}
```

Another problem with the marks in the standard L^AT_EX classes is that the higher level sectioning commands (e.g. `\chapter`) call `\markboth` with an empty right argument. This means that on the first page of a chapter (or a section in article style) the `\rightmark` will be empty. The underlying problem is that the T_EX machinery has only one `\mark`. All the marks must be packed together in this one. So there are no independent left or right marks. That also applies to the extra marks as described in section 25. If this is a problem you must manually insert extra `\markright` commands or redefine the `\chaptermark` (`\sectionmark`) commands to issue a `\markboth` command with two decent parameters.

As a final remark you should also note that the `*` forms of the `\chapter` etc. commands do *not* call the mark commands. So if you want your preface to set the header info but not be numbered nor be put in the table of contents, you must issue the `\markboth` command yourself, e.g.

```
\chapter*{Preface}
\markboth{Preface}{}

Or in a documentclass without chapters:
```

```
\section*{Preface}
\markboth{Preface}{}


```

17 Dictionary style headers

Dictionaries and concordances usually have a header containing the first word defined on the page or both the first and the last words. This can easily be accomplished with `fancyhdr` and L^AT_EX's `mark` mechanism. Of course if you use the marks for dictionary style headers, you cannot use them for chapter and section information, so if there are also chapters and sections present, you must redefine the `\chaptermark` and `\sectionmark` to make them harmless:

```
\renewcommand{\chaptermark}[1]{}
\renewcommand{\sectionmark}[1]{}

```

Now you do a `\markboth{#1}{#1}` for each dictionary or concordance entry #1 and use `\rightmark` for the first entry defined on the page and `\leftmark` for the last one.

If you want to use a header entry of the form `firstword–lastword` it would be nice if this would be reduced to just the form `firstword` if both are the same. This could happen if there is just one entry on the page. In this case a test must be made to check if the marks are the same. However, T_EX's marks are strange beasts, which cannot be compared out of the box with the plain T_EX `\if` commands. Fortunately the `ifthen` package works well:

```
\newcommand{\mymarks}{
  \ifthenelse{equal{\leftmark}{\rightmark}}
    {\rightmark} % if equal
    {\rightmark--\leftmark} % if not equal
\fancyhead[LE,RO]{\mymarks}
\fancyhead[LO,RE]{\thepage}

```

18 Fancy layouts

You can make a multi-line field with the `\` command. It is also possible to put extra space in a field with the `\vspace` command. Note that if you do this you will probably have to increase the height of the header (`\headheight`) and/or of the footer (`\footskip`), otherwise you may get error messages “Overfull \vbox ... has occurred while \output is active”⁷. See the warning below. See also Section 4.1 of the L^AT_EX *Companion* for detail.

For instance, the following code will place the section title and the subsection title of an article in two lines in the upper right hand corner:

```
\documentclass{article}
\usepackage{fancyhdr}
\pagestyle{fancy}
\addtolength{\headheight}{\baselineskip}

```

⁷If you use 11pt or 12pt you will probably also have to do this, because L^AT_EX's defaults are quite small

```
\renewcommand{\sectionmark}[1]{\markboth{#1}{}}
\renewcommand{\subsectionmark}[1]{\markright{#1}}
\fancyhead[R]{\leftmark\\\rightmark}
```

You can also customize the decorative lines. You can make the decorative line in the header quite thick with

```
\renewcommand{\headrulewidth}{0.6pt}
```

or you can make the decorative line in the footer disappear with

```
\renewcommand{\footrulewidth}{0pt}
```

The decorative lines, themselves, are defined in the two macros `\headrule` and `\footrule`. For instance, if you want a dotted line rather than a solid line in the header, redefine the command `\headrule`:

```
\renewcommand{\headrule}{\vbox to 0pt{\hbox
  to\headwidth{\dotfill}\vss}}
```

The redefined `\headrule` should preferably take up no vertical space, as in the example above, and as in the standard definition. If it does take vertical space, the header may come too close to the text, or even intrude in the text. In that case `fancyhdr` will give you a warning that `\headheight` is too small. Like

```
Package fancyhdr Warning: \headheight is too small (12.0pt):
(fancyhdr)             Make it at least 14.0pt, for example:
(fancyhdr)             \setlength{\headheight}{14.0pt}.
(fancyhdr)             You might also make \topmargin smaller to compensate:
(fancyhdr)             \addtolength{\topmargin}{-2.0pt}.
```

You will probably get this warning on every page. **Note:** Before version 4.0, `fancyhdr` would change the `\headheight` itself, causing the text on the following pages to come out lower than on this page. This appeared to be confusing, so since version 4.0 this is no longer done (except when you give the `compatV3` package option. You should not give this as a permanent solution, however, but solve the problem). Therefore you are strongly advised to redefine `\headheight` in the preamble, like this:

```
\setlength{\headheight}{14pt}
```

This would cause the main text to be put 2pt lower on the page, which might be undesirable. You can compensate this by making `\topmargin` correspondingly smaller, for example

```
\addtolength{\topmargin}{-2pt}
```

A similar change would be necessary for `\footskip` if the footer comes out too tall.

You can also eliminate this check completely by using the `nocheck` option of the package. But this may risk unwanted run-ins of the header or footer with other text. So this is generally discouraged. It is better to change `\headheight`, `\footskip`, and/or `\topmargin`. But in cases where you generate the L^AT_EX code automatically, and the software does not know how tall the header or footer will be, this may be handy.

As an alternative to changing `\headrulewidth` to 0 to have the rule disappear, you can also make it empty with

```
\renewcommand{\headrule}{}{}
```

Visually this makes no difference, but it is more difficult to restore it later to its default value.

Finally, let us make a real ‘decorative’ line⁸.

```
\usepackage{fourier-orns}
...
\renewcommand\headrule{%
  \hrulefill
  \raisebox{-2.1pt}
    {\quad\decofourleft\decotwo\decofourright\quad}%
  \hrulefill}
```

This gives us the following headrule:



Note that we haven’t taken care to make this decorative line occupy zero vertical space. The consequence is that it will extend towards the text and that we will get the warning about `\headheight` too small. So we should change `\headheight` as given above. Another problem is that the distance between the line and the header text is quite big. We can reduce this by putting a negative `\vspace` above it, like

```
\renewcommand\headrule{%
  \vspace{-6pt}
  \hrulefill
  \raisebox{-2.1pt}
    {\quad\decofourleft\decotwo\decofourright\quad}%
  \hrulefill}
```

We can use the same code for the `\footrule`, but we wouldn’t need the `\vspace`. If you want to change the distance between that decorative line and the footer text you need to adjust the parameter `\footruleskip`. It defines the distance between the decorative line in the footer and the top of the footer text

⁸Based upon an idea by Wayne Chan.

line. By default it is set to 30% of the normal line distance. You may want to adjust it if you use unusually large or small fonts in the footer. Change it with `\renewcommand`.

`\headruleskip` You can also change the distance between the baseline of the header text and the decorative line in the header. Normally this distance is determined by the maximum depth of possible descenders in the text, which is 30% of the normal line distance. You can increase or decrease this distance by defining the macro `\headruleskip`, similar to `\footruleskip`⁹. This defines the extra distance. The default value is 0pt, and positive values make the distance larger, and negative values make the distance shorter. Please note that this does not change the position of the decorative line with respect to the page, but it shifts the header text. If you want to keep the header text fixed, but move the decorative line, then you must also change the parameter `\headsep` (see figure 1). The header and footer in this page show the *strut* (the amount of space in the text area above and below the baseline), and the `\headruleskip` and `\footruleskip`. For this page `\headruleskip` is 4pt.

19 Two book examples

The following definitions give an approximation of the style used in L. Lamport's \LaTeX book.

Lamport's header overhangs the outside margin. This is done as follows.

The width of headers and footers is `\headwidth`, which by default equals the width of the text: `\textwidth`. You can make the width wider (or narrower) by redefining `\headwidth` with the `\setlength` and `\addtolength` commands. To overhang the outside margin where the marginal notes are printed, add both `\marginparsep` and `\marginparwidth` to `\headwidth` with the commands:

```
\addtolength{\headwidth}{\marginparsep}
\addtolength{\headwidth}{\marginparwidth}
```

It is safest to issue these commands *after* the first `\pagestyle{fancy}` command.

And now a complete definition of Lamport's book style. The header has the width of the text plus the marginpar area. The header on even pages has the page number on the left, and the chapter title on the right. On odd pages it has the section title preceded by the section number on the left and the page number on the right. All in boldface. There is no footer. The `plain` style is redefined to have no header and no footer. (In the \LaTeX book this makes sense because each chapter begins with a page that contains only a drawing. In most other cases you probably would want a page number on the page.)

```
\documentclass{book}
\usepackage{fancyhdr}
```

⁹(But `\headruleskip` is only available since version 4.0.)


```

\pagestyle{fancy}
\addtolength{\headwidth}{\marginparsep}
\addtolength{\headwidth}{\marginparwidth}
\renewcommand{\chaptermark}[1]{\markboth{#1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection\ #1}}
\fancyhf{}
\fancyhead[LE,R0]{\textbf{\thepage}}
\fancyhead[LO]{\textbf{\rightmark}}
\fancyhead[RE]{\textbf{\leftmark}}
\fancypagestyle{plain}{%
  \fancyhead{} % get rid of headers
  \renewcommand{\headrulewidth}{0pt} % and the line
}

```

Notice that the `\chaptermark` and `\sectionmark` commands have been redefined to eliminate the chapter numbers and the uppercaseness.

For more control about the horizontal position of the headers and/or footers, `fancyhdr` has additional commands to specify the offset of the header and/or footer elements. Use `\fancyhfoffset[place]{length}` to offset one or more elements. The `place` parameter is like the optional parameter of `\fancyhf`, like `L R E O`, except that `C` cannot be used. It specifies for which elements the offset should be applied. The `length` parameter specifies the actual offset. Positive values move the element outward (into the margin), negative values inward. There are also specialised commands `\fancyheadoffset` and `\fancyfootoffset`, which have the `H` and `F` parameter pre-applied, respectively.

When you use these commands, \LaTeX will recalculate `\headwidth`, based on the given parameters.

So the above example could also have been done with (N.B. You can only use such an expression as a length parameter if the `calc` package is used):

```

\fancyheadoffset[LE,R0]{\marginparsep+\marginparwidth}

```

For the second example, we take the $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\text{\LaTeX}$ book¹⁰.

Chapter pages have no headers or footers. So we declare

```

\thispagestyle{empty}

```

for every chapter page, and we do not need to redefine `plain`.

Chapter and section titles appear in the form: 2. IMPLEMENTATION, so we have to redefine `\chaptermark` and `\sectionmark` as follows (see Section 16):

```

\renewcommand{\chaptermark}[1]%
  {\markboth{\MakeUppercase{\thechapter.\ #1}}{}}
\renewcommand{\sectionmark}[1]%
  {\markright{\MakeUppercase{\thesection.\ #1}}}

```

¹⁰George Gratzer, *Math into LaTeX, An Introduction to LaTeX and $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\text{\LaTeX}$* , Birkhauser, Boston.

On an even page, the page number is printed as the left header and the chapter info as the right header; on an odd page, the section info is printed as the left header and the page number as the right header. The center headers are empty. There are no footers.

There is a decorative line in the header. It is 0.5pt wide, so we need the commands:

```
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}
```

The font used in the headers is 9 pt bold Helvetica. The PSNFSS system by Sebastian Rahtz uses the short (Karl Berry) name `phv` for Helvetica, so this font is selected with the commands:

```
\fontfamily{phv}\fontseries{b}\fontsize{9}{11}\selectfont
```

(See Sections 7.6.1 and 11.9.1 of the *L^AT_EX Companion* or Sections 7.6 and 7.9.1 of *The L^AT_EX Companion, Second Edition*.) Let us define a shorthand for this:

```
\newcommand{\helv}{%
  \fontfamily{phv}\fontseries{b}\fontsize{9}{11}\selectfont}
```

Now we are ready for the page layout:

```
\documentclass{book}
\usepackage{fancyhdr}
\pagestyle{fancy}
\renewcommand{\chaptermark}[1]{%
  {\markboth{\MakeUppercase{\thechapter.\ #1}}{}}}
\renewcommand{\sectionmark}[1]{%
  {\markright{\MakeUppercase{\thesection.\ #1}}}}
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}
\newcommand{\helv}{%
  \fontfamily{phv}\fontseries{b}\fontsize{9}{11}\selectfont}
\fancyhf{}
\fancyhead[LE,R0]{\helv \thepage}
\fancyhead[LO]{\helv \rightmark}
\fancyhead[RE]{\helv \leftmark}
```

20 Special page layout for float pages

Some people want to have a special layout for float pages (pages only containing floats). As these pages are generated autonomically by L^AT_EX, the user doesn't have any control over them. There is no `\thispagestyle` for float pages and any change of the page style will at least also affect the page before the float page. With `fancyhdr`, however, you can specify in each of the header- or footer fields

```
\iffloatpage{<value for float page>}{<value for other pages>}
```

You can even use this to get rid of the decorative line on float pages only by defining:

```
\renewcommand{\headrulewidth}{\iffloatpage{0pt}{0.4pt}}
```

Sometimes you may want to change the layout also for pages that contain a float on the top of the page, a float on the bottom of the page or a footnote on the bottom of the page.

`fancyhdr` gives you the commands `\iftopfloat`, `\ifbotfloat` and `\iffootnote` similar to `\iffloatpage`. For example:

```
\fancyhead[R]{\iftopfloat{This page has a topfloat}
               {There is no topfloat here}}
```

Note: Marks in floats will not be visible in L^AT_EX's output routine, so it is not useful to put marks in floats. So there is currently no way to let a float (e.g. a figure caption) influence the page header or footer.

21 Those blank pages

In the `book` class when the `openany` option is not given or in the `report` class when the `openright` option is given, chapters start at odd-numbered pages, half of the time causing a blank page to be inserted. Some people prefer this page to be completely empty, i.e. without headers and footers. This cannot be done with `\thispagestyle` as this command would have to be issued on the *previous* page. There is, however, no magic necessary to get this done:

```
\clearpage\begin{group}\pagestyle{empty}\cleardoublepage\end{group}
```

As the `\pagestyle{empty}` is enclosed in a group it only affects the page that may be generated by the `\cleardoublepage`. You can of course put the above in a private command. If you want to have this done automatically at each chapter start or when you want some other text on the page then you must redefine the `\cleardoublepage` command.

```
\makeatletter
\def\cleardoublepage{\clearpage\if@twoside \ifodd\c@page\else
\begin{group}
\mbox{}
\vspace*{\fill}
\begin{center}
This page intentionally contains only this sentence.
\end{center}
\vspace{\fill}
\thispagestyle{empty}
```

```

\newpage
\if@twocolumn\mbox{}\newpage\fi
\endgroup\fi\fi}
\makeatother

```

22 N of M style page numbers

Some document writers prefer the pages to be numbered as n of m where m is the number of pages in the document. There is a package `lastpage` available which you can use with `fancyhdr` as follows:

```

\usepackage{lastpage}
...
\fancyfoot[C]{\thepage\ of \pageref{LastPage}}

```

Because you want the pages with `pagestyle plain` to contain the same style of page numbers, you will have to redefine this `pagestyle` too.

```

\fancypagestyle{plain}{\fancyhead{}\renewcommand{\headrule}{}}

```

We clear all the headers including its rule. The footer will be “inherited” from the `pagestyle fancy`.

The value of the `LastPage` label can be used to make different headers or footers on the last page of a document. E.g. if you want the footer of every odd page, except if it is the last one, to contain the text “please turn over”, this can be done as follows:

```

\usepackage{lastpage}
\usepackage{ifthen}
...
\fancyfoot[R]{\ifthenelse{\isodd{\value{page}} \and \not
  \(\value{page}=\pageref{LastPage}\)}{please turn over}{}}

```

In order to get the number of pages correctly used, you usually have to do one additional \LaTeX run.

23 Chapter or section related page numbers

In technical documentation very often page numbers are used of the form 2-10 where the first number is the chapter number and the second is the pagenumber relative to the chapter. Sometimes section is used rather than chapter. The package `chappg` can be used to get this format.

Basically this package redefines `\thepage` as `\thechapter\chappgsep\arabic{page}`, where `\chappgsep` by default is ‘-’. If you want to use a different separator, you must redefine `\chappgsep`, for example to use an en-dash:

```
\renewcommand{\chappgsep}{--}
```

To use a different prefix, for example the section number, use the `\pagenumbering{bychapter}` command with an optional argument specifying the prefix.

```
\pagenumbering[\thesection]{bychapter}
```

What the package also does is reset the page number to 1 at the beginning of each chapter.

In the frontmatter of your document (for example the Table of Contents) there will be no chapter numbers. Therefore a simple page number will be used there. This may be confusing, so you might prefer to use roman pagenumbers in the front matter. Do this by using `\pagenumbering{roman}` in the beginning of the document and `\pagenumbering{bychapter}` after the first `\chapter` command. If you want to do it before the `\chapter` command you must precede it by a `\newpage` command (see the next section).

```
\pagenumbering{roman}
\tableofcontents
\newpage
\pagenumbering{bychapter}
\chapter{Introduction}
```

There is a caveat when you have appendices in your document. Before the `\appendix` command you should give a `\clearpage` or `\cleardoublepage`. See the `chappg` documentation for details.

There is a fundamental difference between the page numbering of the style “*m* of *n*” as described in the previous section and the current one. The *m* of *n* style is only used in the page header or footer, but not in the table of contents, index, or references like “*See page xx*”. Therefore it does not change the command `\thepage`. The page numbering style “2-10”, however should be used in all references to the page number, therefore it must be done by redefining `\thepage`.

24 When to change the headers and footers?

Sometimes you want to change the header or footer layout in the course of a document. Some of these changes can be accomplished by using the mark mechanism as may be seen in section 16 and 25. However, sometimes we want a more drastic change, e.g. to change the page numbering from roman to arabic (with `\pagenumbering`), to change one of the `fancyhdr` fields or to change to another page style. Sometimes you may be surprised to find the change to occur too early. In general the above mentioned changes take effect immediately, i.e. on the page that is currently being built. If you want the change to take effect at the next page you must make sure that the current page is finished. In most cases

this can be done by issuing a `\newpage` or `\clearpage` command before any of the above mentioned changes. If this is not possible you can use the `afterpage` package with:

`\afterpage{\fancyhead[L]{new value}}` or `\afterpage{\pagenumbering{roman}}`. You cannot use `\afterpage` to change the `\pagestyle` as the commands issued by `\afterpage` are local in a group, and the `\pagestyle` command makes only local changes. The `\pagenumbering` and the `fancyhdr` commands make global changes so they will work, as will the `\thispagestyle` command.

It should be noted that although the `fancyhdr` commands like `\fancyhead` take effect immediately, this does not mean that any “variables” used in these commands get the value they have at the place where these commands are given. E.g. if `\fancyfoot[C]{\thepage}` is given the page number that will be inserted in the footer is not the page number of the page where this command is given, but rather the page number of the actual page where the footer is constructed. Of course for the page number this is what you expect, but it is also true for other commands.

So if you have a book where each chapter is written by a different author and you want the name of the author in the header opposite the chapter title, you can use the following commands:

```
\newcommand{\TheAuthor}{}
\newcommand{\Author}[1]{\renewcommand{\TheAuthor}{#1}}
\fancyhead[LE,R0]{\TheAuthor}
```

and start each chapter with the command `\Author{Real Name}`. If however, the author name would be changed before a page is completed the wrong author could come in the footer. This would be the case if you gave the above command *before* the `\chapter` command rather than after it. Another source of problems is the fact that \TeX 's output routine processes commands ahead, so it may already have processed some commands that produce text that will appear on the next page.

So now we are giving an example how to change the headers and footers only on one specific page. In some cases this can be done by the `\thispagestyle` command. This changes the page style for the “current” page only. But then we may be hit by the problem mentioned in the previous paragraph. \LaTeX may have a different idea about the “current” page than you. The use of `\thispagestyle` is OK if you can be sure that the text where the command `\thispagestyle` is executed is the same page as where the surrounding text appears. So for example directly after a `\chapter` command, or after a `\newpage`. However, when the command is given near the end of a page, \LaTeX may execute the command, and then decide that the page is full and move the text that contains the command to the next page. So now the page style is changed on one page earlier than was intended.

The solution to this problem is to put a label, like `\label{otherpagestyle}` in the text where you want the different page style, and then in the header and/or footer definitions compare the page number with the label page number and choose the proper value. For example, if we want to replace the section title on the special

page with “MYFANCY SECTION”, like in

```
\fancypagestyle{myfancy}{
  \fancyhead[LE,RO]{MYFANCY SECTION}
}
```

we define a new pagestyle that makes the choice:

```
\usepackage{ifthen}
. . .
\fancypagestyle{switch}{
  \fancyhead[LE,RO]{%
    \ifthenelse{\value{page}=\pageref{otherpagestyle}}
      {MYFANCY SECTION}
      {\textsl{\rightmark}}}
}
```

where `\textsl{\rightmark}` is the normal value of the header field from `\pagestyle{fancy}`. Now we choose `\pagestyle{switch}` before our text, or even for the whole document.

There can still be some ambiguity on which page gets the different header. For example, if the text says:

This page gets a different header than the surrounding pages.

where do you put the `\label`? L^AT_EX could break the page between “This” and “page”, and then would you want the special heading on the page where “This” appears, or on the page where “page” appears. It depends on the positioning of the `\label` command. Probably it is safer to make sure the sentence isn’t broken. This can be done by putting the text in a `\parbox` or `minipage` environment.

```
\noindent
\begin{minipage}{\textwidth}
  This page should have a different header than the surrounding pages.
  \label{otherpagestyle}
  It is done with the \verb|\pagestyle{switch}| command, that
  has tests in the header field definitions. This chooses the actual
  header depending on the page number.
\end{minipage}
```

The `\noindent` is necessary, otherwise the whole `minipage` will be shifted right by the paragraph indentation.

Note that you cannot reset the page style immediately after this code, as this may still influence the current page. If you want to reset it, for example to `\pagestyle{fancy}`, you must be sure that it happens on a following page. But in this case it isn’t even necessary, as the special page style acts as the default on all pages except the special page.

The special header and footer in page 24, which show the struts are done in a similar way, although the header and footer are a bit more elaborated there. Also

there is another complication there, as we also want to make both `\headruleskip` and `\footrulewidth` dependent on the page number. Unfortunately, this cannot be done with a simple `\ifthenelse` command. Both `\headruleskip` and `\footrulewidth` are eventually used as length parameters, and this requires that they are *expandable*. However, the `\ifthenelse` construct is not expandable, so you will get strange error messages if you use something like

```
%%% NOTE: This does not work %%%
\renewcommand{\footrulewidth}{%
  \ifthenelse{\value{page}=\pageref{otherpagestyle}}{0.4pt}{0pt}%
}
```

For cases like this `fancyhdr` version 4.0 and later has some new commands `\fancyheadinit`, `\fancyfootinit` and `\fancyhfini`. With `\fancyheadinit{<code>}` you can define some code that will be executed just before the construction of the header. As it is executed in the header, it can test the correct page number, because the counter `page` is guaranteed to have the correct value in the headers and footers. Similarly, the code in `\fancyfootinit{<code>}` is executed in the footer. And `\fancyhfini{<code>}` sets its code for both the header and the footer. Now we can set for example `\headruleskip` or `\footrulewidth` depending on the page number. So instead of putting the test inside the definition of `\headruleskip`, we can put it outside, and then we can use the command `\ifthenelse`. So we put the following in `\pagestyle{switch}`:

```
\fancyheadinit{%
  \ifthenelse{\value{page}=\pageref{otherpagestyle}}
    {\renewcommand{\headruleskip}{6pt}}
    {\renewcommand{\headruleskip}{0pt}}
}
\fancyfootinit{%
  \ifthenelse{\value{page}=\pageref{otherpagestyle}}
    {\renewcommand{\footrulewidth}{4pt}}
    {\renewcommand{\footrulewidth}{0pt}}
}
```

Now here is the definition of the page style used for page 24.

```
\fancypagestyle{showstruts}{%
  \fancyhead[L]{%
    \ifthenelse{\value{page}=\pageref{showstruts}}%
      {\strutheader}%
      {\rightmark}%
  }
  \fancyfoot[L]{%
    \ifthenelse{\value{page}=\pageref{showstruts}}%
      {\strutfooter}%
      {}%
  }
}
```



```

\fancyheadinit{%
  \ifthenelse{\value{page}=\pageref{showstruts}}{%
    {\renewcommand{\headruleskip}{6pt}}%
    {\renewcommand{\headruleskip}{0pt}}%
  }
\fancyfootinit{%
  \ifthenelse{\value{page}=\pageref{showstruts}}{%
    {\renewcommand{\footrulewidth}{0.4pt}}%
    {\renewcommand{\footrulewidth}{0pt}}%
  }
}

```

The label used on that page is `showstruts`. `\strutheader` and `\strutfooter` are macros that contain the code to draw these pictures. In this example the values for `\headruleskip` and `\footrulewidth` in the *else* case are the same as the global values. So we could have left these *else* parts empty. Then they would keep the global values. However, often explicit is better than implicit.

These initialisation commands cannot be used to make global changes to the page, for example to `\headheight`. Neither can you use them to change `\fancyhead` or `\fancyfoot`, because these have already been set up. But you can use it to set the color and font of the header and/or footer, for example to get large, red text in the headers and footers on this specific page:

```

\fancyhfini{%
  \ifthenelse{\value{page}=\pageref{otherpagestyle}}
    {\color{red}\Large}
    {}
}

```

25 Headers and footers induced by the text

We have seen how we can use L^AT_EX's marks to get information from the document contents to the headers and footers. The marks mechanism is the only reliable mechanism that you can use to get changing information to the headers or footers. This is because L^AT_EX may be processing your document ahead before deciding to break the page.

Sometimes the two marks that L^AT_EX offers are not enough. An example is the following:

If a solution to an exercise goes across a page break, then I would like to have “(Continued on next page...)” at the bottom of the first page and “(Continued...)” at the top in the margin of the next page.

You cannot use L^AT_EX's mark mechanisms for this if you also want to use chapter and section information.

The `extramarks` package gives you two extra marks that can be used in this situation. Here is a way to use this package:

```

\usepackage{extramarks}
...
\pagestyle{fancy}
\fancyhead[L]{\firstxmark}
\fancyfoot[R]{\lastxmark}
...
\extramarks{}{Continued on next page\ldots} %1
Some text that may or may not cross a page boundary...
\extramarks{Continued\ldots}{} %2
\extramarks{}{} %3

```

Note that the `\extramarks` command must be close to the text, i.e no empty lines (paragraph boundaries) should intervene. Otherwise the page may be broken at that boundary and the extramarks would come on the wrong page. The final `\extramarks{}{}` is to prevent the ‘Continued...’ header to appear on the following pages.

Explanation: There are two new marks that can be used in the page layout with this package: If commands of the form `\extramarks{ m_1 }{ m_2 }` are given `\firstxmark` gives you the first m_1 value and `\lastxmark` gives you the last m_2 value of the current page. In the above example, when the complete block falls on the same page, the `\firstxmark` will be the empty parameter of the first `\extramarks` command (indicated by %1), and the `\lastxmark` will be the empty parameter from the last `\extramarks` command (indicated by %3).

However, when the page break falls in the block, the mark generated by %1 will be the last one on the first page. Therefore on that page `\lastxmark` will be ‘Continued on next page...’. On the second page, the first mark will be %2, therefore `\firstxmark` will be ‘Continued...’. On all the following pages the values of %3 will be used, i.e. empty ones. Of course in real life you would leave out the % indicators.

In case you want the last m_1 value or the first m_2 value, you can use the `\lastleftxmark` or `\firstrightxmark`, respectively. For symmetry reasons there are also commands `\firstleftxmark` ($=\text{\firstxmark}$), `\lastrightxmark` ($=\text{\lastxmark}$), `\topleftxmark` ($=\text{\topxmark}$) and `\toprightxmark`. The top-marks are basically the last-marks of the previous page.

The package also gives you the `\firstleftmark` and `\lastrightmark` commands that complement the standard L^AT_EX marks.

To stress the point that marks are the correct way to do this, let me give you a “solution” that will not work¹¹:

```

\fancyhead[L]{Continued}
\fancyfoot[R]{Continued on next page\ldots}
Some text that may or may not cross a page boundary...
\fancyhead[L]{}
\fancyfoot[R]{}

```

¹¹ Actually there is another way but it requires two L^AT_EX passes: you can put `\label` commands before and after the text and compare the `\pagerefs`.

You may be tempted to think that the first `\fancyhead` and `\fancyfoot` will be in effect when \TeX breaks the page in the middle of the text, and the last ones when the page breaks after the text. This is not true as the whole paragraph (including the last definitions) will be processed before \TeX considers the page break, so at the time of the page break the last definitions are effective, whether the page break occurs inside the text or outside of it. Putting a paragraph boundary between the text and the last definitions will not work either, because you don't want the first definitions to be in effect when \TeX decides to break the page exactly at this boundary. Actually the marks mechanism was invented to get rid of these kinds of problems.

In the above example the text “Continued” appears in the page header. It may be nicer to put it in the margin. This can be easily accomplished by positioning it at a fixed place relative to the page header. In plain \TeX you would use a concoction of `\hbox to 0pt`, `\vbox to 0pt`, `\hskip`, `\vskip`, `\hss` and `\vss` but fortunately \LaTeX 's `picture` environment gives a much cleaner way to do this. In order not to disturb the normal header layout we put the text in a zero-sized `picture`. Generally this is the best way to position things on fixed places on the page. You can then also use the normal headings. See also section 27 for another example of this technique.

```
\fancyhead[L]{\setlength{\unitlength}{\baselineskip}%
\begin{picture}(0,0)
  \put(-2,-3){\makebox(0,0)[r]{\firstxmark}}
\end{picture}\leftmark}
```

This solution can of course also be used for the footer. Make sure you put the `picture` as the first thing in left-handside entries and last in right-handside ones.

Finally you may want to put “(Continued...)” in the *text* rather than in the header or the margin. Then you have to use the `afterpage` package. We also decide to make a separate environment for it.

```
\newenvironment{continued}{\par
  \extramarks{}{Continued on next page\ldots}
  \afterpage{\noindent\firstxmark\vspace{1ex}}
}{\extramarks{(Continued\ldots)}{}\par}
```

It is a bit dangerous to use `\firstxmark` outside the page layout routine, but apparently with `\afterpage` this works. If you would need the information further on in the page you must remember the state of the marks in your own variable. You can set this in one of the `fancyhdr` fields. For example if you want to add something *after* the broken piece of text you can use the following:

```
\newcommand{\mysaved}{}

\newenvironment{continued}{\par
  \extramarks{}{Continued on next page\ldots}
}{\extramarks{(Continued\ldots)}{}\par\vspace{1ex}\mysaved}
```

```

\fancyhead[L]{\leftmark}
\ifthenelse{\equal{\lastxmark}{}}
{\gdef\mysaved{}}
{\gdef\mysaved{\noindent[Continued from previous page]}}

```

If you want to include one of the marks or other varying information in the saved text, you must use `\xdef` rather than `\gdef`.

26 A movie

If you put at each page on the same place a picture that slightly changes from page to page you can get a movie-like effect by flipping through the pages. You can create such a movie easily with `fancyhdr`. For simplicity we assume that each picture is in a postscript (EPS) file called `pic⟨n⟩.ps` where `⟨n⟩` is the page number and that we use the `graphics` or `graphicx` package¹². To put the movie in the righthandside bottom corner the following will work:

```

\fancyfoot[R]{\setlength{\unitlength}{1mm}
\begin{picture}(0,0)
\put(5,0){\includegraphics{pic\thepage.ps}}
\end{picture}}

```

Notice that the `\unitlength` parameter should be set locally in the `fancyhdr` field in order to avoid unwanted interference with its value in the text.

27 Thumb-indexes

Some railroad guides and expensive bibles have so called *thumb-indexes*, i.e. there are marks on the sides of the pages that indicate where the chapters are. You can create these by printing black blobs in the margin of the pages. The vertical position should be determined by the chapter number or some other counter. As the position is independent of the contents of the page, we print these blobs as part of the header in a zero-sized `picture` as described in the previous section.

Of course we have to take care of two-sided printing, and we may want to have an index page with all the blobs in the correct position. The solution requires some hand-tuning to get the blobs nicely spaced out vertically. For the application that I had there were 12 sections, so I made the blobs 18 mm apart, i.e. 9 mm blob separated by 9 mm whitespace. In order to avoid calculations they are set in a `picture` environment with the `\unitlength` set to 18 mm. Page numbers are set in the headers at the outer sides, and the blobs are attached to these. In this example the section numbers are used to position the blobs, but you can replace this with any numeric value. See figure 4 for the resulting overview page and figure 5 for the code.

¹²If you use an older version of L^AT_EX you could use the `epsf` or `epsfig` package.

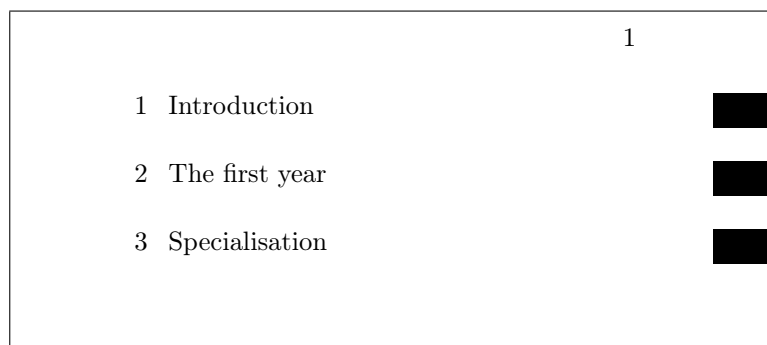


Figure 4: Thumb-index overview page

28 Float placement

Floats are page elements that float with respect to the rest of the document. Standard floats are tables and figures, but with the `float` package you can easily make new ones, like algorithms. Most of the time floats work satisfactory, but sometimes \LaTeX seems too stubborn to do what you want. This section describes how you can influence \LaTeX so that it will do most of the time what you want. There might, however be some pathological cases where it is impossible to convince \LaTeX to do things your way. In the following we will use figures as an example but everything applies to other floats as well.

The most encountered problems with floats are:

1. You want a float at a certain position in the text, but \LaTeX moves it, usually to the next page.
2. From a certain point, \LaTeX moves all your floats to the end of the document or the end of a chapter.
3. \LaTeX complains about “Too many floats”.

In the first two cases you must first check if you have given the correct “placement” parameter to your float, e.g. `\begin{figure}[htp]` specifies that your figure may be placed either: Here (i.e. in the text position where the command is given), on the Top of a page (which may be the page where you put the command), or on a separate Page of floats. You could also have specified “b” for Bottom of the page. The order of the letters is insignificant, you cannot force \LaTeX to try Bottom first and then Top by specifying `[bt]`.

If \LaTeX doesn’t put the float at the place where you expected it, it is usually caused by the following:

1. The float didn’t fit on the page. In this case it has to move to the next page or even further. If you didn’t specify either `[t]` or `[b]` in the position parameter, \LaTeX must save it until it has enough for a page of floats. So

```

\setlength{\unitlength}{18mm}
\newcommand{\blob}{\rule[-.2\unitlength]{2\unitlength}{.5\unitlength}}

\newcommand{\rblob}{\thepage
  \begin{picture}(0,0)
    \put(1,-\value{section}){\blob}
  \end{picture}}

\newcommand{\lblob}{^A
  \begin{picture}(0,0)
    \put(-3,-\value{section}){\blob}
  \end{picture}^A
\thepage}

\pagestyle{fancy}
\fancyfoot[C]{}

\newcounter{line}
\newcommand{\secname}[1]{\addtocounter{line}{1}^A
  \put(1,-\value{line}){\blob}
  \put(-7.5,-\value{line}){\Large \arabic{line}}
  \put(-7,-\value{line}){\Large #1}}

\newcommand{\overview}{\thepage
  \begin{picture}(0,0)
    \secname{Introduction}
    \secname{The first year}
    \secname{Specialisation}
    ...etc...
  \end{picture}}

\begin{document}
\fancyhead[R]{\overview}\mbox{}\newpage % This produces the overview page
\fancyhead[R]{} % Front matter may follow here
\clearpage
\fancyhead[RE]{\rightmark}
\fancyhead[RO]{\rblob}
\fancyhead[LE]{\lblob}
\fancyhead[LO]{\leftmark}
...

```

Figure 5: Thumb-index code

don't specify only [h]. If you want to give L^AT_EX a chance to put the float on a page of floats, you must also specify "p".

2. The placement would violate the constraints imposed by L^AT_EX's float placement parameters. This is one of the most occurring causes and it can easily be corrected by changing the parameters. Here is a list of them:

Counters – change with <code>\setcounter</code>		
<code>topnumber</code>	max. number of floats at top of page	2
<code>bottomnumber</code>	max. number of floats at bottom of page	1
<code>totalnumber</code>	max. number of floats on a page	3
Other – change with <code>\renewcommand</code>		
<code>\topfraction</code>	max fraction of page for floats at top	0.7
<code>\bottomfraction</code>	max fraction of page for floats at bottom	0.3
<code>\textfraction</code>	min fraction of page for text	0.2
<code>\floatpagefraction</code>	min fraction of floatpage that should have floats	0.5

There are also some others for double column floats in two-column documents.

The values in the righthand column are the defaults for the standard L^AT_EX classes. Other classes could use different defaults. As you see with the default values a float will not be put in the bottom of a page if its height is more than 30% of the page height. So if you specify [hb] for a float which is taller it has to move to a float page. But if it is less than 50% of the page height it will have to wait until some more floats are given before a float page can be filled to satisfy the `\floatpagefraction` parameter. If you have this kind of behaviour you can easily adapt the parameters, e.g. with:

```
\renewcommand{\textfraction}{0.05}
\renewcommand{\topfraction}{0.95}
\renewcommand{\bottomfraction}{0.95}
\renewcommand{\floatpagefraction}{0.35}
\setcounter{totalnumber}{5}
```

You may want to be careful not to make `\floatpagefraction` too small, otherwise you may get too many small floatpages.

You can force L^AT_EX to ignore most of the parameters for one specific float occurrence by including an exclamation mark (!) in the placement parameters, e.g.

```
\begin{figure}[!htb]
```

Floats which contain a "t" in the position parameter could be placed before the place where they are referenced (but on the same page). This is normal behaviour for L^AT_EX but some people just don't like it. There are a number of ways to prevent this:

1. Of course deleting the “`t`” will help, but in general this is undesirable, as you may want the float to be placed at the top of the next page.
2. use the `flafter` package which causes floats never to be placed “backwards”.
3. use the command `\suppressfloats[t]`. This command will cause floats for the top position *on this page* to be moved to the next page. This can also be done with `[b]` or without parameter for all floats on this page.

If in spite of all your attempts \LaTeX still moves your floats to the end of the document or the end of a chapter, you can insert a `\clearpage` command. This will start a new page and insert all pending floats before continueing. If it is undesirable to have a pagebreak you can use the `afterpage` package and the following command:

```
\afterpage{clearpage}
```

This will wait until the current page is finished and then flush all outstanding floats. In some pathological circumstances `afterpage` may give strange results, however.

Finally, if you want a float only at the place where you define it, without \LaTeX moving it whatsoever, you can use the `float` package and give the command:

```
\restylefloat{figure}
```

in the preamble. Now you will be able to specify `[H]` as the position parameter, which will mean “HERE and only HERE”. This may cause an unwanted page break however. If you want to avoid the unwanted pagebreak, i.e. let \LaTeX move the float only if it doesn’t fit on the page, then use the `afterpage` package with:

```
\afterpage{\clearpage \begin{figure}[H] ... \end{figure}}
```

Complaints from \LaTeX about “Too many floats” are usually caused by one of the above problems: floats not being able to be placed and \LaTeX collecting too many of them. The solutions given above, especially those with `\clearpage` in them will usually help. In some cases there really are too many floats, as \LaTeX has a limited number of “boxes” to store the floats. The package `morefloats` can be used to increase this number. If you need still more then you must edit a private copy of this file, but even then there will be some limit that you cannot pass. Then your only resort will be to change your document.

29 Multipage Floats

\LaTeX ’s floats cannot be split across pages. Sometimes, however, you want to have a table or figure that doesn’t fit on one page. The easiest way is to split these into multiple table or figure environments, but this has a number of undesirable effects:

- Where do you split it? This is in general more difficult for tables than for figures.
- How do you keep them together?
- You don't want more than one entry in the list of figures/tables.

Although these problems are not fully solvable in all cases, here are a couple of suggestions:

29.1 Tables

For tables longer than a page you can use the `longtable` package. This package defines a `longtable` environment that is a kind of amalgamation of `table` and `tabular`. It has approximately the same syntax as the `tabular` environment, but it adds some features of `table`, like captions. Longtables will be automatically split when they don't fit on the page. And they will be entered in the list of tables when a caption is given. They will not float, however, and cannot be used inside a float environment. This could mean that another `table` environment, which was defined before the `longtable`, will float past it, and therefore the numbers may get out of order. Another problem could be that the `longtable` starts rather far down the page, which isn't a pleasant sight. If you want the `longtable` to start at the top of the page, the best thing to do is to include it in an `\afterpage` command (using the `afterpage` package). As a `longtable` is by definition large, it is best to put it in a separate file, and `\input` it in the `\afterpage` command:

```
\afterpage{\input{mytable}}
```

```
\afterpage{\clearpage\input{mytable}}
```

The last form has the additional advantage that most of the outstanding floats will be printed first.

29.2 Figures

There isn't an equivalent `longfigure` solution, so for figures you will usually have to split it yourself. In general this is less of a problem. However, the problem you get now is how to keep them together, i.e. how to get the parts on subsequent pages, and how to get a single entry in the list of figures.

You will have to split the figure into pieces and put each part in a separate `figure` environment. To keep them together it is best to use only the `[p]` placement, so that they will be put on floatpages. As they are bigger than a page this is appropriate. The first part would then get a `\caption`, the subsequent parts would be used without a caption. If you want to add a caption-like text, enter it as normal text rather than a `\caption`, so that it will not be entered in the list of figures. It may also be desirable to issue a `\clearpage` first, just like we did for the `longtable`, and to encapsulate this in the `\afterpage` command. E.g.

```
\afterpage{\clearpage\input{myfigure}}
```

where `myfigure.tex` contains:

```
\begin{figure}[p]
\includegraphics{myfig1.eps}
\caption{This is a multipage figure}
\label{fig:xxx}
\end{figure}
\begin{figure}[p]
\includegraphics{myfig2.eps}
\begin{center}
Figure~\ref{fig:xxx} (continued)
\end{center}
\end{figure}
```

You have to make sure that the last part is big enough, otherwise L^AT_EX could decide to postpone it until it has collected some more floats. This can be done either by making the figure big enough (e.g. by adding some `\vspace`), or by tweaking the `\floatpagefraction` parameter.

If you want your multipage figure to start at a lefthand-side (even-numbered) page you can use a test in the `\afterpage` command (using the `ifthen` package):

```
\afterpage{\clearpage
\ifthenelse{\isodd{\value{page}}}{\afterpage{\input{myfigure}}} % odd page
{\input{myfigure}}}} % even page
```

If there are too many floats on the skipped page, this may still fail to start your multipage figure on an even page, however.

30 Deprecated commands

This section contains the description of deprecated commands. These were parts of the original implementation of `fancyheadings`. They continue to work for compatibility reasons, but it is recommended not to use them anymore. This description is given so that you know what they mean and how to convert them to the standard commands. To be honest, I use these sometimes myself in quick examples, because `\lhead` is less typing than `\fancyhead[L]`.

These commands for specifying the header or footer fields and their translation to the modern commands are given in table 1.

As you see, if there is an optional parameter, this one applies to the even pages, whereas the required parameter applies to the odd pages. Of course this only works if the `twoside` option is given in the documentclass. If there is no optional parameter, the required parameter applies to both even and odd pages.

There was also a special pagestyle `fancyplain` that could be used to define both the pagestyle `fancy` and to redefine the pagestyle `plain` at the same time. In order to use that you say

<code>\lhead{xx}</code>	<code>\fancyhead[L]{xx}</code>
<code>\lhead[xx]{yy}</code>	<code>\fancyhead[LE]{xx} \fancyhead[LO]{yy}</code>
<code>\chead{xx}</code>	<code>\fancyhead[C]{xx}</code>
<code>\chead[xx]{yy}</code>	<code>\fancyhead[CE]{xx} \fancyhead[CO]{yy}</code>
<code>\rhead{xx}</code>	<code>\fancyhead[R]{xx}</code>
<code>\rhead[xx]{yy}</code>	<code>\fancyhead[RE]{xx} \fancyhead[RO]{yy}</code>
<code>\lfoot{xx}</code>	<code>\fancyfoot[L]{xx}</code>
<code>\lfoot[xx]{yy}</code>	<code>\fancyfoot[LE]{xx} \fancyfoot[LO]{yy}</code>
<code>\cfoot{xx}</code>	<code>\fancyfoot[C]{xx}</code>
<code>\cfoot[xx]{yy}</code>	<code>\fancyfoot[CE]{xx} \fancyfoot[CO]{yy}</code>
<code>\rfoot{xx}</code>	<code>\fancyfoot[R]{xx}</code>
<code>\rfoot[xx]{yy}</code>	<code>\fancyfoot[RE]{xx} \fancyfoot[RO]{yy}</code>

Table 1: Deprecated commands and their translation

```
\pagestyle{fancyplain}}
```

and then in the headers/footers you use for example:

```
\fancyhead[L]{\fancyplain{value for ‘plain’ page}{value for other pages}}}
```

The `\fancyplain` command is only useful within the `pagestyle fancyplain`. Nowadays you would just redefine `pagestyle plain` with the `\fancypagestyle{plain}{xxxx}` command (see section 11).

There are also `\plainheadrulewidth` and `\plainfootrulewidth` commands to define the values of `\headrulewidth` and `\footrulewidth` to be used on ‘plain’ pages. This also only works with the `pagestyle fancyplain`, not when you redefine `pagestyle plain` with the `\fancypagestyle` command.

31 Contact information

Piet van Oostrum

E-mail: piet@vanoostrum.org

WWW: <http://piet.vanoostrum.org>

32 Version information

- Version 1.0. March 11, 2003. This is the version that was distributed for a long time on CTAN. Version history before this has been lost.
- Version 2.0. August 27, 2016:
 - Removed references to `fixmarks.sty` as that is no longer used.
 - References to older L^AT_EX versions removed.

- Removed obsolete source code of `extramarks.sty`
 - Changed font commands to `\textbf` and `\textsl`.
 - Added description of the `\fancy...offset` commands.
 - Added various `\...xmark` commands from `extramarks.sty`.
 - Various corrections applied.
 - Updated contact information.
 - Added Version information. :)
- Version 2.1. August 28, 2016
 - Explain what the top-marks are.
- Version 2.1. Sept. 6, 2016
 - Add `\string` to special indexing commands to get a neater index file.
 - Add a decorative headrule example.
- Version 3.9, October 13, 2016.
 - Documentation integrated in `fancyhdr.dtx`.
 - Version number unified with `fancyhdr.sty`.
 - All deprecated commands moved to a separate section (30).
 - Documentation expanded.
- Version 3.9a, June 30, 2017.
 - Updated contact information.
 - Restore `\newtoks\@temptokenb`
- Version 3.10, Januari 25, 2019
 - Distribution based on `fancydhr.dtx`.
 - Use `\f@nch@ifundefined` instead of `\ifx` or `\@ifundefined`.
 - Replace `\def` with `\newcommand` in several places.
 - Don't use `\global\setlength`.
 - Put `\footrule` in a `\vbox` to accommodate for flexible footrules, and then `\unvbox` that. Move the `\footruleskip` vertical space outside of the definition of `\footrule`.

32.1 Changes in version 4

Version 4 is a significant rewrite of the package. It also introduces a number of new features.

- Version 4.0, March 15-22, 2019
 - Options introduced on the `\usepackage` command.
 - The check whether the header or footer fits in `\headheight` and `\footskip`, respectively, no longer adjusts these values for the following pages. This appeared to be too confusing. However, when the package option `compatV3` is given, the old behaviour is kept.
The `nocheck` option now eliminates these checks completely, on your own risk. (See section 18 on page 22.)
 - Eliminated global definitions. All definitions are now local. The `\global` case was originally so that you could do definitions in a group and they would be applied globally. This was a mistake. If you make them locally they should stay local. And it caused problems with switching page styles, because then the global style would be changed, which you generally don't want.
However, when the package option `compatV3` is given, the old behaviour is kept. (See section 14.)
 - The page style `fancydefault`.
 - The `\headruleskip` parameter.
 - The `\fancyheadinit`, `\fancyfootinit`, and `\fancyhfini` commands.
Note: The following changes were mostly copied from the `nccfancyhdr` package by Alexander I. Rozhenko.
 - The `\fancycenter` command (section 9).
 - The `headings` and `myheadings` package options (see section 14).
 - The `\fancypagestyle` command has an optional parameter [*base-style*].

Part III

Questions & Answers

This part contains answers to questions that have been emailed to me, or have been asked at various internet forums, and don't have a logical place in the other documentation. It is expected to grow gradually.

33 Large chapter/section titles

Sometimes a chapter or section title is too large to fit in the header or footer. It may take more than one line in the header/footer, or it may overwrite other parts.

How can we shorten these titles in the header/footer without changing the actual title?

Here is an example:

```
\fancyhead[LE,R0]{\nouppercase{\rightmark}} % Section title
\fancyhead[LO,RE]{\nouppercase{\leftmark}} % Chapter title
\fancyfoot[C]{\thepage}
. . .
\chapter{This is a very long chapter title}
. . .
\section{This is a very long section title that will not fit in the header}
. . .
```

With these settings the header will come out as:

Chapter 1. This is a very long chapter title This is a very long section title that will not fit in the header

which isn't very nice. There are basically three options to solve this problem.

33.1 Using optional arguments

As we have seen in section 16, the header info comes from the marks. So if we want the text in the header to be smaller we have to supply smaller marks. This can be done by giving these as optional arguments in the `\chapter` and `\section` commands.¹³

```
\chapter[Short chapter title]{This is a very long chapter title}
. . .
\section[Short section title]
{This is a very long section title that will not fit in the header}
```

The short titles will now appear in the header. However, these will also appear in the table of contents. If that is what you want than you are ready. But if you want to use the long titles in the table of contents, you have to use some trickery. In particular you have to supply the marks yourself.

33.2 Using explicit marks

First we show how you can supply a different value for the chapter title in the heading, because this is the easiest. Remember from section 16 that this mark is defined by calling `\chaptermark`. Also, because it is used as `\leftmark`, the last value of this mark on the page is used. So we can easily overrule the value that is supplied by the `\chapter` command, by supplying an additional `\chaptermark` command after the `\chapter` command, like this:

```
\chapter{This is a very long chapter title that does not fit in the header}
\chaptermark{This is a not so long chapter title}
```

¹³At least in the `book` and `report` documentclasses. In the `article` class this would be the `\section` and `\subsection` commands.

For the section titles the situation is more complicated. Here we use the `\rightmark`, which uses the first mark of its kind on the page. So you might think putting a `\sectionmark` before the `\section` command would be the solution. Unfortunately, it is not that simple. In many cases, this will work, but not when there is a page break just before the section title, because in that case the `\sectionmark` will stay behind on the previous page. However, we can put the `\sectionmark` inside the argument of the `\section` command. Because L^AT_EX first typesets the title (which will execute the included `\sectionmark` command), and after that executes its own `\sectionmark`, our `\sectionmark` will be the first. But there is one case in which this fails: if the next page does not have any `\sectionmark` commands, it will inherit the **last** mark from the page before it, which will be the long title. To correct this we must also give an additional `\sectionmark` with the short title **after** the `\section` command.

As if this isn't enough, there is still a problem with this setup. Our section title is not only used to typeset the title in the text, but it is also included in the table of contents. But the table of contents does not accept a `\sectionmark` in its title. It will generate an ugly error message. To prevent this we must give the long title (that we want to appear in the table of contents) also as the optional argument to the `\section` command. Of course this will also generate a mark for the header, but this will be overruled by our included `\sectionmark` commands

So the complete code would be:

```
\section[Long title]{Long title\sectionmark{Short title}}
\sectionmark{Short title}
```

To avoid all the repetitions, it is better to make a macro:

```
\newcommand{\Section}[2]{\section[#1]{#1\sectionmark{#2}}\sectionmark{#2}}
...
\Section{Text title}{Header title}
```

And if you want to use yet a different text in the table of contents, you can make a macro with three parameters. The third parameter is the text to be put in the table of contents. We use this parameter as the optional argument for the `\section` command.

```
\newcommand{\Sectioniii}[3]{\section[#3]{#1\sectionmark{#2}}\sectionmark{#2}}
...
\Sectioniii{Text title}{Header title}{TOC title}
```

Please note that if you use the `article` class, instead of `\chaptermark` and `\sectionmark`, you would probably use `\sectionmark\subsectionmark`.

33.3 Using automatic truncation

For this solution we use the `truncate` package by Donald Arseneau. This has a `\truncate` command that truncates a text to a maximum size, when it exceeds

that size. We put both headers in `\truncate` to limit it to half the `\headwidth`. Of course it is also possible to make asymmetric arrangements.

```
\usepackage[fit]{truncate}
\fancyhead[LE,R0]{\nouppercase{\truncate{0.5\headwidth}{\rightmark}}}\fancyhead[LO,RE]{\nouppercase{\truncate{0.5\headwidth}{\leftmark}}}
```

We don't have to make any changes to the chapter and section titles because `\truncate` will take care of this. This arrangement gives the following header when both titles are too big, like in the example above:

Chapter 1. This is a very long chapter ... 1.2. This is a very long section title ...

Note that we have used the `[fit]` option of the `truncate` package. Otherwise the right header will not be right aligned, but it will start at halfway the header. Note also that, as each part can occupy half of the available width, they could theoretically touch each other. This can be prevented by making the widths slightly smaller. And when there is only one title in the header, you can make the width equal to or slightly smaller than `\headwidth`.

34 I lost my chapter/section titles

Some time ago I got a question like this (edited to get the essentials):

"I redefined the `\pagestyle{fancy}` to get my own kind of headings. Also, I redefined the `\chaptermark`. I need the `fancy` style from chapter 1 and on (mainmatter part), but, until the Introduction chapter (that I included into the frontmatter part) I need the `myheadings` style.

When I set the `myheadings` style into the frontmatter the `fancy` style doesn't show the chapter title any more.

What can I do in order to reestablish the right behavior of the `fancy` style?"

The solution to this problem is actually very simple. The `pagestyle myheadings` (as well as `headings`) redefines the `\chaptermark` and `\sectionmark`, so when you return to `pagestyle fancy`, the definitions you had given before (or the ones that `fancyhdr` provided) are lost. You just have to repeat them at the point where you switch back to `pagestyle fancy`.

```
\begin{document}
\frontmatter
\pagestyle{myheadings}
. . .
\mainmatter
\pagestyle{fancy}
\renewcommand{\chaptermark}[1]{...}
```


Part IV

Implementation

35 fancyhdr.sty

<*fancyhdr>

`\if@nch@empty` This macro tests if its argument is empty.

```
1 \newcommand\if@nch@empty[1]{\def\temp@a{#1}\ifx\temp@a\@empty}
```

`\iff@nch@check` Boolean for the `nocheck` option.

```
2 \newif\iff@nch@check
3 \f@nch@checktrue
4 \DeclareOption{nocheck}{%
5   \f@nch@checkfalse
6 }
```

`\iff@nch@compatViii` Define `\iff@nch@compatViii` to track the `compatV3` option.

```
7 \newif\iff@nch@compatViii
```

`\f@nch@gbl` Initialise `\f@nch@gbl` to do nothing (except with the `compatV3` option).

```
8 \let\f@nch@gbl\relax
9 \DeclareOption{compatV3}{%
10   \let\f@nch@gbl\global
11   \f@nch@compatViiitrue
12 }
```

`\f@nch@def` This macro defines another macro (usually a header or footer field). Depending on the value of `\f@nch@gbl` the definition will be global or local. Default it is always local. But with the `compatV3` option it is `\global` in the normal definitions, and local in `\fancypagestyle`. The `\global` case is now considered a bug (or at least undesirable).

If the value (argument 2) is empty, a `\leavevmode` will be substituted. If it is not empty, a `\strut` will be added.

```
13 \newcommand\f@nch@def[2]{\if@nch@empty{#2}\f@nch@gbl\def#1{\leavevmode}\else
14   \f@nch@gbl\def#1{#2\strut}\fi}
```

`\f@nch@ifundefined` This macro tests if a command is undefined. Older versions of `fancyhdr` used `\@ifundefined`, but this had an undesired side effect in the original `LATEX` (the command was made equal to `\relax` if it was undefined). Another way was `\ifx\thecommand\undefined ...` or `\ifx\thecommand\@undefined ...` but that could conflict with packages that use the `\@ifundefined` method. `LATEX` versions later than 2018 have a definition of `\@ifundefined` that avoids these problems, but not everybody may have such a version installed. Therefore we define our own version `\f@nch@ifundefined`. This definition is copied from the `tocloft` package by Peter Wilson and Will Robertson.

```

15 \newcommand{\f@nch@ifundefined}[1]{%
16   \begingroup\expandafter\expandafter\expandafter\endgroup
17   \expandafter\ifx\csname #1\endcsname\relax
18     \expandafter\@firstoftwo
19   \else
20     \expandafter\@secondoftwo
21   \fi}
22 % \end{macrocode}
23 % \end{macro}
24 %
25 % \changes{fancyhdr v4.0}{2019/03/17}{Added \texttt{headings} and
26 % \texttt{myheadings} options.}
27 % Standard styles are redefined optionally. These definitions are
28 % borrowed from the \Package{nccfancyhdr} package by Alexander I. Rozhenko.
29 %
30 % \begin{macro}{\ps@myheadings}
31 % The redefinition of the \texttt{myheadings} style is conditional.
32 % We test the existence of the \cs{chapter} command and redefine the
33 % style accordingly.
34 %
35 % \OPTindex{myheadings}\PSindex{myheadings}
36 %   \begin{macrocode}
37 \DeclareOption{myheadings}{%
38   \f@nch@ifundefined{chapter}{%
An article-like class without chapters:
39     \def\ps@myheadings{\ps@f@nch@fancyproto \let\@mkboth\@gobbletwo
40       \fancyhf{}}
41     \fancyhead[LE,RO]{\thepage}%
42     \fancyhead[RE]{\slshape\leftmark}%
43     \fancyhead[LO]{\slshape\rightmark}%
44     \let\sectionmark\@gobble
45     \let\subsectionmark\@gobble
46   }%
47 }%
A book/report-like class with chapters:
48 {\def\ps@myheadings{\ps@f@nch@fancyproto \let\@mkboth\@gobbletwo
49   \fancyhf{}}
50   \fancyhead[LE,RO]{\thepage}%
51   \fancyhead[RE]{\slshape\leftmark}%
52   \fancyhead[LO]{\slshape\rightmark}%
53   \let\chaptermark\@gobble
54   \let\sectionmark\@gobble
55 }%
56 }%
57 }

```

`\ps@headings` The redefinition of the `headings` style also differs for book-like and article-like classes. It also differs for one-side and two-side modes.

```
58 \DeclareOption{headings}{%
```

```

59 \f@nch@ifundefined{chapter}{%
60 \if@twoside

```

An article in two-side mode:

```

61 \def\ps@headings{\ps@f@nch@fancyproto \let\@mkboth\markboth
62 \fancyhf{}
63 \fancyhead[LE,RO]{\thepage}%
64 \fancyhead[RE]{\slshape\leftmark}%
65 \fancyhead[LO]{\slshape\rightmark}%
66 \def\sectionmark##1{%
67 \markboth{\MakeUppercase{%
68 \ifnum \c@secnumdepth >\z@ \thesection\quad \fi##1}}{}}%
69 \def\subsectionmark##1{%
70 \markright{%
71 \ifnum \c@secnumdepth >\@ne \thesubsection\quad \fi##1}}%
72 }%
73 \else

```

An article in one-side mode:

```

74 \def\ps@headings{\ps@f@nch@fancyproto \let\@mkboth\markboth
75 \fancyhf{}
76 \fancyhead[LE,RO]{\thepage}%
77 \fancyhead[RE]{\slshape\leftmark}%
78 \fancyhead[LO]{\slshape\rightmark}%
79 \def\sectionmark##1{%
80 \markright {\MakeUppercase{%
81 \ifnum \c@secnumdepth >\z@ \thesection\quad \fi##1}}}%
82 \let\subsectionmark\@gobble % Not needed but inserted for safety
83 }%
84 \fi
85 }\if@twoside

```

A book in two-side mode:

```

86 \def\ps@headings{\ps@f@nch@fancyproto \let\@mkboth\markboth
87 \fancyhf{}
88 \fancyhead[LE,RO]{\thepage}%
89 \fancyhead[RE]{\slshape\leftmark}%
90 \fancyhead[LO]{\slshape\rightmark}%
91 \def\chaptermark##1{%
92 \markboth{\MakeUppercase{%
93 \ifnum \c@secnumdepth >\m@ne \if@mainmatter
94 \@chapapp\ \thechapter. \ \fi\fi##1}}{}}%
95 \def\sectionmark##1{%
96 \markright {\MakeUppercase{%
97 \ifnum \c@secnumdepth >\z@ \thesection. \ \fi##1}}}%
98 }%
99 \else

```

A book in one-side mode:

```

100 \def\ps@headings{\ps@f@nch@fancyproto \let\@mkboth\markboth
101 \fancyhf{}
102 \fancyhead[LE,RO]{\thepage}%

```

```

103     \fancyhead[RE]{\slshape\leftmark}%
104     \fancyhead[LO]{\slshape\rightmark}%
105     \def\chaptermark##1{%
106         \markright{\MakeUppercase{%
107             \ifnum \c@secnumdepth >\m@ne \if@mainmatter
108                 \chapapp\ thechapter. \ \fi\fi##1}}}%
109     \let\sectionmark\@gobble % Not needed but inserted for safety
110     }%
111     \fi
112 }%
113 }

```

Process the options.

```
114 \ProcessOptions*
```

`\f@nch@errmsg` This macro generates an error message.

```

115 \newcommand\f@nch@errmsg[1]{%
116     \f@nch@ifundefined{PackageError}{\errmessage{#1}}{\PackageError{fancyhdr}{#1}{}}}

```

`\f@nch@warning` This macro generates a warning.

```

117 \newcommand\f@nch@warning[1]{%
118     \f@nch@ifundefined{PackageWarning}{\errmessage{#1}}{\PackageWarning{fancyhdr}{#1}{}}}

```

`\f@nch@forc` Usage: `\f@nch@forc \var {charstring}{body}`.

Execute the body for each character in `charstring` bound to `\var`. This is similar to L^AT_EX's `\@tfor`, but it expands the `charstring`.

```

119 % \changes{fancyhdr v3.10}{2019/01/25}{Use \cs{newcommand} instead of \cs{def}.}
120 \newcommand{\f@nch@forc}[3]{\expandafter\f@nch@rc\expandafter#1\expandafter{#2}{#3}}
121 \newcommand{\f@nch@rc}[3]{\def\temp@ty{#2}\ifx\@empty\temp@ty\else
122     \f@nch@rc#1#2\f@nch@rc{#3}\fi}
123 \def\f@nch@rc#1#2#3\f@nch@rc#4{\def#1{#2}#4\f@nch@rc#1{#3}{#4}}

```

`\f@nch@for` Usage: `\f@nch@for\var{list}{body}`

Execute the body for each element of the list, bound to `\var`. List elements are separated by commas. This is like L^AT_EX's `\@for` but an empty list is treated as a list with an empty element.

```

124 \newcommand{\f@nch@for}[3]{\edef\@fortmp{#2}%
125     \expandafter\@forloop#2,\@nil,\@nil\@@#1{#3}}

```

`\f@nch@default` Usage: `\f@nch@default \var{defaults}{argument}`

Sets `\var` to the characters from `defaults` appearing in `argument`, or to `defaults` if it would be empty. All characters are lowercased first.

```

126 \newcommand\f@nch@default[3]{%
127     \edef\temp@a{\lowercase{\edef\noexpand\temp@a{#3}}}\temp@a \def#1{%
128     \f@nch@forc\tempf@ra{#2}%
129     {\expandafter\f@nch@ifin\tempf@ra\temp@a{\edef#1{#1\tempf@ra}}{}}%
130     \ifx\@empty#1\def#1{#2}\fi}

```

`\f@nch@ifin` Usage: `\f@nch@ifin <char> <set> <truecase> <falsecase>`
If `<char>` is in `<set>`, then `<truecase>` else `<falsecase>`.

```
131 \newcommand{\f@nch@ifin}[4]{%
132   \edef\temp@a{#2}\def\temp@b##1#1##2\temp@b{\def\temp@b{##1}}%
133   \expandafter\temp@b#2#1\temp@b\ifx\temp@a\temp@b #4\else #3\fi}
```

`\fancyhead` These are the principal user macros. Pick up the parameters, and supply an 'h'

`\fancyfoot` (`\fancyhead`) or 'f' (`\fancyfoot`).

```
\fancyhf 134 \newcommand{\fancyhead}[2][\f@nch@fancyhf]{\fancyhead h[#1]{#2}}%
135 \newcommand{\fancyfoot}[2][\f@nch@fancyhf]{\fancyfoot f[#1]{#2}}%
136 \newcommand{\fancyhf}[2][\f@nch@fancyhf]{\fancyhf {}[#1]{#2}}%
```

`\fancyheadoffset` The commands for offsets. Pick up the parameters, and supply an 'h'

`\fancyfootoffset` (`\fancyheadoffset`) or 'f' (`\fancyfootoffset`).

```
\fancyhfoffset 137 \newcommand{\fancyheadoffset}[2][\f@nch@fancyhfoffs]{\fancyheadoffset h[#1]{#2}}%
138 \newcommand{\fancyfootoffset}[2][\f@nch@fancyhfoffs]{\fancyfootoffset f[#1]{#2}}%
139 \newcommand{\fancyhfoffset}[2][\f@nch@fancyhfoffs]{\fancyhfoffset {}[#1]{#2}}%
```

`\f@nch@fancyhf` This macro interprets the parameters for the headers and footers.

Parameters:

(1) The user command that was used (like `\fancyhead`). This is used for errors/warnings.

(2) h (for `\fancyhead`), f (for `\fancyfoot`), or {} (for `\fancyhf`).

(3) The optional parameter that was given to these commands (default []).

(4) The required parameter that was given to these commands.

The header and footer fields are stored in command sequences with names of the form: `\f@nch@<x><y><z>` with `<x>` from [eo], `<y>` from [lcr] and `<z>` from [hf].

```
140 \def\f@nch@fancyhf#1#2[#3]#4{%
141   \def\temp@c{}%
142   \f@nch@forc\tempf@ra{#3}%
143   {\expandafter\f@nch@ifin\tempf@ra{eolcrhf,EOLCRHF}%
144     {\edef\temp@c{\temp@c\tempf@ra}}}%
145   \ifx\empty\temp@c\else \f@nch@errmsg{Illegal char '\temp@c' in
146     \string#1 argument: [#3]}%
147   \fi \f@nch@for\temp@c{#3}%
148   {\f@nch@default\f@nch@eo{eo}\temp@c \if@twoside\else \if\f@nch@eo
149     e\f@nch@warning {\string#1's 'E' option without twoside option is
150       useless}\fi\fi \f@nch@default\f@nch@lcr{lcr}\temp@c
151     \f@nch@default\f@nch@hf{hf}{#2\temp@c}%
152     \f@nch@forc\f@nch@eo\f@nch@eo
153       {\f@nch@forc\f@nch@lcr\f@nch@lcr
154         {\f@nch@forc\f@nch@hf\f@nch@hf
155           {\expandafter\f@nch@def\csname
156             f@nch@\f@nch@eo\f@nch@lcr\f@nch@hf\endcsname {#4}}}}}
```

`\f@nch@fancyhfoffs` This macro interprets the parameters for the header and footer offsets.

Parameters:

(1) The user command that was used (like `\fancyheadoffset`). This is used for errors/warnings.

(2) `h` (for `\fancyheadoffset`), `f` (for `\fancyfootoffset`), or `{}` (for `\fancyhfoffset`).

(3) The optional parameter that was given to these commands (default `[]`).

(4) The required parameter that was given to these commands.

The header and footer offsets are stored in command sequences with names of the form: `\f@nch@0@<x><y><z>` with `<x>` from `[eo]`, `<y>` from `[lr]` and `<z>` from `[hf]`.

```

157 \def\f@nch@fancyhfoffs#1#2[#3]#4{%
158   \def\temp@c{}%
159   \f@nch@forc\tmpf@ra{#3}%
160   {\expandafter\f@nch@ifin\tmpf@ra{eolrhf,EOLRHF}%
161    {}{\edef\temp@c{\temp@c\tmpf@ra}}}%
162   \ifx\@empty\temp@c\else \f@nch@errmsg{Illegal char '\temp@c' in
163     \string#1 argument: [#3]}%
164   \fi \f@nch@for\temp@c{#3}%
165   {\f@nch@default\f@nch@eo{eo}\temp@c \if@twoside\else \if\f@nch@eo
166     e\f@nch@warning {\string#1's 'E' option without twoside option is
167       useless}\fi\fi \f@nch@default\f@nch@lcr{lr}\temp@c
168     \f@nch@default\f@nch@hf{hf}{#2\temp@c}%
169     \f@nch@forc\f@nch@eo\f@nch@eo
170       {\f@nch@forc\f@nch@lcr\f@nch@lcr
171        {\f@nch@forc\f@nch@hf\f@nch@hf
172         {\expandafter\setlength\csname
173           f@nch@0@\f@nch@eo\f@nch@lcr\f@nch@hf\endcsname {#4}}}}}%
174   \f@nch@setoffs}

```

`\lhead` Fancyheadings version 1 commands. These are deprecated, but they continue to work for compatibility reasons. They have an optional parameter that is used as the value for even pages in a two-sided document. If this is not given (or if the document is not two-sided) the required parameter is used for both even and odd pages. Therefore the default value for the optional parameter is the required parameter. It is not possible to express this directly in the definition. Therefore we use a trick. Both parameters are store in a macro. For example for `\lhead` the parameter for even pages is stored in `\f@nch@elh`, and the one for odd pages in `\f@nch@olh`. For the others it is similar, just replace the `l` with `c` or `r`, and the `h` with `f`. In the body of the macro we first store the required parameter in `\f@nch@olh`, and we use this macro as default for the optional parameter. The optional parameter is then stored in `\f@nch@elh`. The order of the assignments is therefore important.

```

175 \newcommand{\lhead}[2][\f@nch@olh]%
176   {\f@nch@def\f@nch@olh{#2}\f@nch@def\f@nch@elh{#1}}
177 \newcommand{\chead}[2][\f@nch@och]%
178   {\f@nch@def\f@nch@och{#2}\f@nch@def\f@nch@ech{#1}}
179 \newcommand{\rhead}[2][\f@nch@orh]%
180   {\f@nch@def\f@nch@orh{#2}\f@nch@def\f@nch@erh{#1}}
181 \newcommand{\lfoot}[2][\f@nch@olf]%
182   {\f@nch@def\f@nch@olf{#2}\f@nch@def\f@nch@elf{#1}}
183 \newcommand{\cfoot}[2][\f@nch@ocf]%
184   {\f@nch@def\f@nch@ocf{#2}\f@nch@def\f@nch@ecf{#1}}
185 \newcommand{\rfoot}[2][\f@nch@orf]%

```

```

186 { \f@nch@def \f@nch@orf{#2} \f@nch@def \f@nch@erf{#1}}

\f@nch@headwidth Length parameter to be used for \headwidth. We use this rather than defining
\headwidth as a length paramater directly to protect ourself to someone saying:
\let\headwidth\textwidth.
187 \newlength{\f@nch@headwidth} \let\headwidth\f@nch@headwidth

\f@nch@O@elh Length parameters for the offsets.
\f@nch@O@erh 188 \newlength{\f@nch@O@elh}
\f@nch@O@olh 189 \newlength{\f@nch@O@olh}
\f@nch@O@orh 190 \newlength{\f@nch@O@orh}
\f@nch@O@elf 191 \newlength{\f@nch@O@orh}
\f@nch@O@erf 192 \newlength{\f@nch@O@elf}
\f@nch@O@olf 193 \newlength{\f@nch@O@erf}
\f@nch@O@orf 194 \newlength{\f@nch@O@olf}
195 \newlength{\f@nch@O@orf}

\headrulewidth
\footrulewidth 196 \newcommand{\headrulewidth}{0.4pt}
197 \newcommand{\footrulewidth}{0pt}

\headruleskip Don't define \headruleskip if it is already defined.
198 \f@nch@ifundefined{headruleskip}%
199 { \newcommand{\headruleskip}{0pt}}{}

\footruleskip Memoir also defines \footruleskip. Don't define \footruleskip if it is already
defined.
200 \f@nch@ifundefined{footruleskip}%
201 { \newcommand{\footruleskip}{.3\normalbaselineskip}}{}

\plainheadrulewidth Fancyplain stuff shouldn't be used anymore (rather \fancypagestyle{plain}
\plainfootrulewidth should be used), but we keep it for compatibility reasons.
202 \newcommand{\plainheadrulewidth}{0pt}
203 \newcommand{\plainfootrulewidth}{0pt}

\if@fancyplain Boolean for the implementation of \fancyplain
204 \newif\if@fancyplain \@fancyplainfalse

\fancyplain Deprecated macro
205 \def\fancyplain#1#2{\if@fancyplain#1\else#2\fi}

\headwidth Initialise \headwidth with a magic constant.
206 \headwidth=-123456789sp

\f@nch@raggedleft Save the standard definitions of \raggedleft, \raggedright, \centering and
\f@nch@raggedright \everypar so that we can reset them when we are typesetting the headers and
\f@nch@centering footers. Some packages change these to incompatible values.
\f@nch@everypar 207 \let\f@nch@raggedleft\raggedleft
208 \let\f@nch@raggedright\raggedright
209 \let\f@nch@centering\centering
210 \let\f@nch@everypar\everypar

```

`\f@nch@reset` Command to reset various things in the headers: a.o. single spacing (taken from `setspace.sty`) and the catcode of `\endlinechar` (so that `epsf` files in the header work if a verbatim crosses a page boundary) It also defines a `\nouppercase` command that disables `\uppercase` and `\Makeuppercase`. It can only be used in the headers and footers. Set `\hsize` to `\headwidth` (this helps for multicol); reset `\raggedleft`, `\raggedright` and `\centering` to their default values (for `tabu`), and `\everypar` to empty.

The font is reset to `\normalfont`.

```
211 \def\f@nch@reset{\f@nch@everypar{}\restorecr\endlinechar=13
212 \let\\\@normalcr \let\raggedleft\f@nch@raggedleft
213 \let\raggedright\f@nch@raggedright \let\centering\f@nch@centering
214 \def\baselinestretch{1}%
215 \hsize=\headwidth
216 \def\nouppercase##1{\let\uppercase\relax\let\MakeUppercase\relax
217 \expandafter\let\csname MakeUppercase \endcsname\relax##1}%
218 \f@nch@ifundefined{@newbaseline} % NFSS not present; 2.09 or 2e
219 {\f@nch@ifundefined{@normalsize} {\normalsize} % for ucthesi.cls
220 {\@normalsize}}%
221 {\@newbaseline}% NFSS (2.09) present
222 }
```

`\fancycenter` `\fancycenter[$\langle dist \rangle$][$\langle stretch \rangle$]{ $\langle left-mark \rangle$ }{ $\langle center-mark \rangle$ }{ $\langle right-mark \rangle$ }`

```
223 \newcommand*\fancycenter[1][1em]{%
224 \ifnextchar[\f@nch@center{#1}]{\f@nch@center{#1}[3]}%
225 }
226 \def\f@nch@center#1[#2]#3#4#5{%
```

At first, we execute the case when the $\langle center-mark \rangle$ is empty:

```
227 \def\@tempa{#4}\ifx\@tempa\@empty
228 \hbox to\linewidth{\color@begingroup{#3}\hfil {#5}\color@endgroup}%
229 \else
```

All we need to do is to calculate skips inserted before and after $\langle center-mark \rangle$. We will calculate them in the `|tempkipa|` and `|tempkipb|`. At first:

```
\@tempdima:= $\langle dist \rangle$ ;
\@tempdimb:= $\langle dist \rangle * \langle stretch \rangle$ ;
\@tempdimc:= $\langle dist \rangle * \langle stretch \rangle - \langle dist \rangle$ ;
\@tempkipa:=\@tempkipb:=\@tempdimb + 1fil - \@tempdimc;

230 \setlength\@tempdima{#1}%
231 \setlength{\@tempdimb}{#2\@tempdima}%
232 \@tempdimc \@tempdimb \advance\@tempdimc -\@tempdima
233 \setlength\@tempkipa{\@tempdimb \@plus 1fil \@minus \@tempdimc}%
234 \@tempkipb\@tempkipa
```

At this point, the `\@tempkipa` and `\@tempkipb` registers have the natural size $\langle dist \rangle * \langle stretch \rangle$, unlimited stretchability, and the minimum size $\langle dist \rangle$. Now we decrease the minimum size of `\@tempkipa` to zero if the $\langle left-mark \rangle$ is empty:

```
235 \def\@tempa{#3}\ifx\@tempa\@empty
236 \addtolength\@tempkipa{\z@ \@minus \@tempdima}%
237 \fi
```


Do the same things with the `\@tempskipb` register if the *right-mark* is empty:

```
238 \def\@tempa{#5}\ifx\@tempa\@empty % empty right
239 \addtolength\@tempskipb{\z@ \@minus \@tempdima}%
240 \fi
```

Finally, we correct the left and right glues taking into account the difference between lengths of *left-mark* and *right-mark*. We calculate which mark is shorter and increase the natural size of the corresponding register by the difference between their lengths.

```
241 \settowidth{\@tempdimb}{#3}%
242 \settowidth{\@tempdimc}{#5}%
243 \ifdim\@tempdimb>\@tempdimc
244 \advance\@tempdimb -\@tempdimc
245 \addtolength\@tempskipb{\@tempdimb \@minus \@tempdimb}%
246 \else
247 \advance\@tempdimc -\@tempdimb
248 \addtolength\@tempkipa{\@tempdimc \@minus \@tempdimc}%
249 \fi
```

The `\@tempkipa` and `\@tempskipb` have been calculated. Put everything in the box.

```
250 \hbox to\linewidth{\color@begingroup{#3}\hskip \@tempkipa
251 \hskip \@tempkipb {#5}\color@endgroup}%
252 \fi
253 }
```

`\fancyheadinit` This macro can be used to define initialisation code that will be run before the construction of the header. It can for example set the color or the font, or change `\headrulewidth` or `\headruleskip`. It cannot make global changes, just changes for the header.

`\f@nch@headinit` Storage for the header initialisation code.

```
254 \newcommand{\f@nch@headinit}{}
255 \newcommand{\fancyheadinit}[1]{%
256 \def\f@nch@headinit{#1}%
257 }
```

`\fancyfootinit` This macro can be used to define initialisation code that will be run before the construction of the footer. It can for example set the color or the font, or change `\footrulewidth` or `\footruleskip`. It cannot make global changes, just changes for the footer.

`\f@nch@footinit` Storage for the footer initialisation code.

```
258 \newcommand{\f@nch@footinit}{}
259 \newcommand{\fancyfootinit}[1]{%
260 \def\f@nch@footinit{#1}%
261 }
```

`\fancyhfini` This macro sets both the header and the footer initialisation codes to the same value.

```
262 \newcommand{\fancyhfini}[1]{%
```

```

263 \def\f@nch@headinit{#1}%
264 \def\f@nch@footinit{#1}%
265 }

```

\f@nch@vbox Make a \vbox with the header or footer. Check whether there is enough space and give a warning if not. Use box 0 as a temp box and dimen 0 as temp dimen. This can be done, because this code will always be used inside another box, and therefore the changes are local.

Parameter 1 is \headheight or \footskip, respectively.

Parameter 2 is the contents of the box.

```

266 \newcommand\f@nch@vbox[2]{%
267   \setbox0\vbox{#2}%
268   \ifdim\ht0>#1\relax
269     \iff@nch@check
270       \dimen0=#1\advance\dimen0-\ht0
271       \f@nch@warning{%
272         \string#1 is too small (\the#1): \MessageBreak
273         Make it at least \the\ht0, for example:\MessageBreak
274         \string\setlength{\string#1}{\the\ht0}%
275         \iff@nch@compatViii .\MessageBreak
276         We now make it that large for the rest of the document.\MessageBreak
277         This may cause the page layout to be inconsistent, however
278         \fi
279         \ifx#1\headheight .\MessageBreak
280           You might also make \topmargin smaller to compensate:\MessageBreak
281           \string\addtolength{\string\topmargin}{\the\dimen0}%
282         \fi
283         \@gobble
284       }%
285     \fi
286     \iff@nch@compatViii
287       \dimen0=#1\relax
288       \global#1=\ht0\relax
289       \ht0=\dimen0 %
290     \else
291       \ht0=#1%
292     \fi
293   \fi
294   \box0}

```

\f@nch@head Put together a header or footer given the left, center and right text, fillers at left and right and a rule. The \xlap commands put the text into an hbox of zero size, so overlapping text does not generate an error message.

These macros have 5 parameters:

1. LEFTSIDE BEARING. This determines at which side the header will stick out. When \fancyhfoffset is used this calculates \headwidth, otherwise it is \hss or \relax (after expansion).
2. \f@nch@olh, \f@nch@elh, \f@nch@olf or \f@nch@elf. This is the left component.

3. `\f@nch@och`, `\f@nch@ech`, `\f@nch@ocf` or `\f@nch@ecf`. This is the center component.
4. `\f@nch@orh`, `\f@nch@erh`, `\f@nch@orf` or `\f@nch@erf`. This is the right component.
5. **RIGHTSIDE BEARING**. This is always `\relax` or `\hss` (after expansion). Before constructing the header or footer, the environment is reset to a known state, and then the corresponding initialisation code as given in `\fancyheadinit` or `\fancyfootinit`, respectively, is run.

```

295 \newcommand\f@nch@head[5]{%
296   \f@nch@reset
297   \f@nch@headinit\relax
298   #1%
299   \hbox to\headwidth{%
300     \f@nch@vbox\headheight{%
301       \hbox{%
302         \rlap{\parbox[b]{\headwidth}{\raggedright#2}}}%
303         \hfill
304         \parbox[b]{\headwidth}{\centering#3}%
305         \hfill
306         \llap{\parbox[b]{\headwidth}{\raggedleft#4}}}%
307       }%
308       \vskip\headruleskip\relax
309       \headrule
310     }%
311   }%
312   #5%
313 }
```

`\f@nch@foot` We put the `\footrule` in a `\vbox` to accommodate for flexible footrules (e.g. using `\hrulefill`), so that the `\headwidth` will be used as the line width. But to preserve the vertical spacing we then `\unvbox` this box.

```

314 \newcommand\f@nch@foot[5]{%
315   \f@nch@reset
316   \f@nch@footinit\relax
317   #1%
318   \hbox to\headwidth{%
319     \f@nch@vbox\footskip{%
320       \setbox0=\vbox{\footrule}\unvbox0
321       \vskip\footruleskip
322       \hbox{%
323         \rlap{\parbox[t]{\headwidth}{\raggedright#2}}}%
324         \hfill
325         \parbox[t]{\headwidth}{\centering#3}%
326         \hfill
327         \llap{\parbox[t]{\headwidth}{\raggedleft#4}}}%
328       }%
329     }%
330   }%
331   #5%
332 }
```

```

\MakeUppercase Define For old LATEXen. Note: we used \def rather than \let, so that
\let\uppercase\relax (from the version 1 documentation) will still work.
333 \f@nch@ifundefined{MakeUppercase}{\def\MakeUppercase{\uppercase}}{}%

\@chapapp Define \@chapapp for classes that don't have it, e.g. amsbook
334 \f@nch@ifundefined{@chapapp}{\let\@chapapp\chaptername}{}%

\f@nch@initialise This macro initialises the headers and footers and \chaptermark and/or
\sectionmark for pagestyle fancy
335 \def\f@nch@initialise{%

\chaptermark Standard definitions for \chaptermark, \sectionmark and \subsectionmark.
\sectionmark
\subsectionmark 336 \f@nch@ifundefined{chapter}%
337 {\def\sectionmark##1{\markboth{\MakeUppercase{\ifnum \c@secnumdepth>\z@
338 \thesection\hskip 1em\relax
339 \fi ##1}}{}}%
340 \def\subsectionmark##1{\markright {\ifnum \c@secnumdepth >\@ne
341 \thesubsection\hskip 1em\relax \fi ##1}}}%
342 {\def\chaptermark##1{\markboth {\MakeUppercase{\ifnum
343 \c@secnumdepth>\m@ne \@chapapp\ thechapter. \ \fi ##1}}{}}%
344 \def\sectionmark##1{\markright{\MakeUppercase{\ifnum \c@secnumdepth >\z@
345 \thesection. \ \fi ##1}}}%
346 }%

\headrule
347 \def\headrule{\if@fancyplain\let\headrulewidth\plainheadrulewidth\fi
348 \hrule\@height\headrulewidth\@width\headwidth
349 \vskip-\headrulewidth}}%

\footrule
350 \def\footrule{\if@fancyplain\let\footrulewidth\plainfootrulewidth\fi
351 \hrule\@width\headwidth\@height\footrulewidth}}%

Default values for \headrulewidth, \footrulewidth, \headruleskip and
\footruleskip.
352 \def\headrulewidth{0.4pt}%
353 \def\footrulewidth{0pt}%
354 \def\headruleskip{0pt}%
355 \def\footruleskip{0.3\normalbaselineskip}%

Initialisation of the head and foot text.
The default values still contain \fancyplain for compatibility: lefthead empty
on “plain” pages, \rightmark on even, \leftmark on odd pages; evenhead empty
on “plain” pages, \leftmark on even, \rightmark on odd pages.
356 \fancyhf{}%
357 \if@twoside
358 \fancyhead[el,or]{\fancyplain{}\slshape\rightmark}}%
359 \fancyhead[er,ol]{\fancyplain{}\slshape\leftmark}}%
360 \else

```

```

361 \fancyhead[l]{\fancyplain{}{\slshape\rightmark}}%
362 \fancyhead[r]{\fancyplain{}{\slshape\leftmark}}%
363 \fi
364 \fancyfoot[c]{\rmfamily\thepage}% page number
365 }

```

Call the initialisation

```
366 \f@nch@initialise
```

\ps@f@nch@fancyproto **\ps@f@nch@fancyproto** is the initial value for pagestyle **fancy**. The real page style is **\f@nch@fancycore**. This definition is for the first use. It also initialises **\headwidth**, and then resets itself to **\f@nch@fancycore**. For backwards compatibility it still contains **\@fancyplainfalse**. The reason we have **\ps@f@nch@fancyproto** is so that page style **fancy** can be redefined.

```
367 \def\ps@f@nch@fancyproto{%
```

Initialise **\headwidth** if the user didn't. If **\headwidth < 0**, then the user did not initialise it, or they just added something to it in the expectation that it was initialised to **\textwidth**. We compensate this now. This loses if the user intended to multiply it by a factor. But that case is more likely done by saying something like **\setlength{\headwidth}{1.2\textwidth}**. The documentation advises to change **\headwidth** after the first call to **\pagestyle{fancy}**. This code is just to catch the most common cases where that is not the case.

```

368 \ifdim\headwidth<0sp
369 \global\advance\headwidth123456789sp\global\advance\headwidth\textwidth
370 \fi

```

Now we reset **\ps@f@nch@fancyproto** to **\f@nch@fancycore** with **\@fancyplainfalse** and call that version.

```

371 \gdef\ps@f@nch@fancyproto{\@fancyplainfalse\f@nch@fancycore}%
372 \@fancyplainfalse\f@nch@fancycore
373 }%

```

Let the system know this is a **fancyhdr** pagestyle.

```
374 \@namedef{f@nch@ps@f@nch@fancyproto-is-fancyhdr}{}
```

ps@fancy Define **\ps@fancy** just to call **\ps@f@nch@fancyproto**.

```

375 \def\ps@fancy{\ps@f@nch@fancyproto}
376 \@namedef{f@nch@ps@fancy-is-fancyhdr}{ }

```

\ps@fancyplain The pagestyle **fancyplain** (deprecated).

```

377 \def\ps@fancyplain{\ps@fancy \let\ps@plain\ps@plain@fancy}
378 \def\ps@plain@fancy{\@fancyplaintrue\f@nch@fancycore}

```

\f@nch@ps@empty Save the definition of **\ps@empty** (pagestyle **empty**).

```
379 \let\f@nch@ps@empty\ps@empty
```

\f@nch@fancycore The actual implementation of pagestyle **fancy**. For **amsbook/amsart**, which do strange things with **\topskip**, we start with **\f@nch@ps@empty**. We construct the even and odd headers and footers from all the parts that we have collected.

```

380 \def\f@nch@fancycore{%
381   \f@nch@ps@empty
382   \def\@mkboth{\protect\markboth}%
383   \def\@oddhead{\f@nch@head\f@nch@Oolh\f@nch@olh\f@nch@och\f@nch@orh\f@nch@Oorh}%
384   \def\@oddfoot{\f@nch@foot\f@nch@Oolf\f@nch@olf\f@nch@ocf\f@nch@orf\f@nch@Oorf}%
385   \def\@evenhead{\f@nch@head\f@nch@Oelh\f@nch@elh\f@nch@ech\f@nch@erh\f@nch@Oerh}%
386   \def\@evenfoot{\f@nch@foot\f@nch@Oelf\f@nch@elf\f@nch@ecf\f@nch@erf\f@nch@Oerf}%
387 }

```

ps@fancydefault This is page style `fancydefault`. It is in fact page style `fancy` with all the defaults embedded. In contrast with page style `fancy` that gets all its settings from the environment. It reruns all initialisations and then calls `\ps@f@nch@fancyproto`.

```

388 \def\ps@fancydefault{%
389   \f@nch@initialise
390   \ps@f@nch@fancyproto
391 }
392 \@namedef{f@nch@ps@fancydefault-is-fancyhdr}{ }

```

\f@nch@Oolh Default definitions for compatibility mode: These cause the header/footer to take
\f@nch@Oorh the defined `\headwidth` as its width and if required to shift it in the direction of
\f@nch@Oelh the marginpar area.

```

\f@nch@Oerh 393 \def\f@nch@Oolh{\if@reversemargin\hss\else\relax\fi}
\f@nch@Oolf 394 \def\f@nch@Oorh{\if@reversemargin\relax\else\hss\fi}
\f@nch@Oorf 395 \let\f@nch@Oelh\f@nch@Oorh
\f@nch@Oelf 396 \let\f@nch@Oerh\f@nch@Oolh
\f@nch@Oerf 397 \let\f@nch@Oolf\f@nch@Oolh
              398 \let\f@nch@Oorf\f@nch@Oorh
              399 \let\f@nch@Oelf\f@nch@Oelh
              400 \let\f@nch@Oerf\f@nch@Oerh

```

\f@nch@offsolh New definitions for the use of `\fancyhfoffset`, `\fancyheadoffset`,
\f@nch@offselh `\fancyfootoffset`. These calculate the `\headwidth` from `\textwidth`
 and the specified offsets.

First for the header.

```

401 \def\f@nch@offsolh{\headwidth=\textwidth\advance\headwidth\f@nch@Oolh
402   \advance\headwidth\f@nch@Oorh\hskip-\f@nch@Oolh}
403 \def\f@nch@offselh{\headwidth=\textwidth\advance\headwidth\f@nch@Oelh
404   \advance\headwidth\f@nch@Oerh\hskip-\f@nch@Oelh}

```

\f@nch@offsolh The same for the footer.

```

\f@nch@offselh 405 \def\f@nch@offsolf{\headwidth=\textwidth\advance\headwidth\f@nch@Oolf
406   \advance\headwidth\f@nch@Oorf\hskip-\f@nch@Oolf}
407 \def\f@nch@offself{\headwidth=\textwidth\advance\headwidth\f@nch@Oelf
408   \advance\headwidth\f@nch@Oerf\hskip-\f@nch@Oelf}

```

\f@nch@setoffs Set the offset parts to be used in the construction of the headers and footers. Depending on `\f@nch@gbl` it will be done globally (for pagestyle `fancy`) or locally (for `\fancypagestyle`). Just in case `\let\headwidth\textwidth` was used, we reset `\headwidth` to the length parameter that it should be.

```

409 \def\fnch@setoffs{%
410   \fnch@gb\let\headwidth\fnch@headwidth
411   \fnch@gb\let\fnch@0olh\fnch@offsolh
412   \fnch@gb\let\fnch@0elh\fnch@offselh \fnch@gb\let\fnch@0orh\hss
413   \fnch@gb\let\fnch@0erh\hss \fnch@gb\let\fnch@0olf\fnch@offsof
414   \fnch@gb\let\fnch@0elf\fnch@offself \fnch@gb\let\fnch@0orf\hss
415   \fnch@gb\let\fnch@0erf\hss
416 }

\iffnch@footnote Redefine \@makecol so that we can capture if there are top/bottom floats, foot-
  \@makecol      notes or if we are on a float page. Because of a clash with the footmisc package
                  we do this at \begin{document}.
                  We need a boolean \iffnch@footnote to capture if there was a footnote.

417 \newif\iffnch@footnote
418 \AtBeginDocument{%
419   \let\latex@makecol\@makecol
420   \def\@makecol{\ifvoid\footins\fnch@footnotefalse\else\fnch@footnotetrue\fi
421     \let\topfloat\@toplist\let\botfloat\@botlist\latex@makecol}%
422 }

\iftopfloat These can be used in a header/footer field to make them conditional on the pres-
\ifbotfloat  ence of floats and/or footnotes.
\iffloatpage 423 \newcommand\iftopfloat[2]{\ifx\topfloat\empty #2\else #1\fi}%
\iffootnote 424 \newcommand\ifbotfloat[2]{\ifx\botfloat\empty #2\else #1\fi}%
              425 \newcommand\iffloatpage[2]{\if@colmade #1\else #2\fi}%
              426 \newcommand\iffootnote[2]{\iffnch@footnote #1\else #2\fi}%

\fancypagestyle Define a new page style. The optional second argument is the base page style. It
                 defaults to \ps@fnch@fancyproto.

427 \newcommand{\fancypagestyle}[1]{%
428   \@ifnextchar[{\fnch@pagestyle{#1}}{\fnch@pagestyle{#1}[fnch@fancyproto]]}%
429 }

\fnch@pagestyle The actual code for \fancypagestyle. Build the page style body. Declare it as a
                 fancyhdr page style.

430 \def\fnch@pagestyle#1[#2]#3{%
431   \fnch@ifundefined{ps@#2}{%
432     \fnch@errmsg{\string\fancypagestyle: Unknown base page style ‘#2’}%
433   }{%
434     \fnch@ifundefined{fnch@ps@#2-is-fancyhdr}{%
435       \fnch@errmsg{\string\fancypagestyle: Base page style ‘#2’ is not fancyhdr-based}%
436     }%
437     {%
438       \@namedef{ps@#1}{\let\fnch@gb\relax\@nameuse{ps@#2}#3\relax}%
439       \@namedef{fnch@ps@#1-is-fancyhdr}{}%
440     }%
441   }%
442 }%

</fancyhdr>

```

36 extramarks.sty

<*extramarks>

\@temptokenb A token register to store some marks information

443 \newtoks\@temptokenb

\unrestored@protected@xdef Define this macro just in case it isn't defined (should be part of L^AT_EX).

444 \providecommand\unrestored@protected@xdef{%

445 \let\protect\@unexpandable@protect \xdef}

\markboth Our own definition of \markboth, mainly because \@markboth gets more parameters.

446 \def\markboth#1#2{%

447 \begingroup

448 \let\label\relax \let\index\relax \let\glossary\relax

449 \expandafter\@markboth\@themark{#1}{#2}%

450 \@temptokena \expandafter{\@themark}%

451 \mark{\the\@temptokena}%

452 \endgroup

453 \if@nobreak\ifvmode\nobreak\fi\fi}

\markright We use the standard definition of \markright. No use to duplicate here.

\@markboth Note: put #3#4 in toks register.

454 \def\@markboth#1#2#3#4#5#6{\@temptokena{{#3}{#4}}%

455 \unrestored@protected@xdef\@themark{{#5}{#6}}\the\@temptokena}}

\@markright Note: put #1 and #3#4 in toks registers. Maybe I can get rid of the extra \@temptokenb by doing the expansion of #5 to a temp separately. But then, nowadays registers are plenty.

456 \def\@markright#1#2#3#4#5{\@temptokena{#1}\@temptokenb{{#3}{#4}}%

457 \unrestored@protected@xdef\@themark{{\the\@temptokena}{#5}}\the\@temptokenb}}

\@leftmark Internal macros to get the standard marks.

\@rightmark 458 \def\@leftmark#1#2#3#4{#1}

459 \def\@rightmark#1#2#3#4{#2}

\leftmark The standard marks + the new ones (based on the standard marks info).

\rightmark 460 \def\leftmark{\expandafter\@leftmark

\firstleftmark 461 \botmark\@empty\@empty\@empty\@empty}

\lastrightmark 462 \def\rightmark{\expandafter\@rightmark

\firstrightmark 463 \firstmark\@empty\@empty\@empty\@empty}

\lastleftmark 464 \def\firstleftmark{\expandafter\@leftmark

465 \firstmark\@empty\@empty\@empty\@empty}

466 \def\lastrightmark{\expandafter\@rightmark

467 \botmark\@empty\@empty\@empty\@empty}

468 \let\firstrightmark \rightmark

469 \let\lastleftmark \leftmark


```

\@themark This is where the marks information is stored.
470 \def\@themark{{}{}}{}{}

\extramarks This command is used to define the extra marks.
471 \newcommand\extramarks[2]{%
472   \begingroup
473   \let\label\relax \let\index\relax \let\glossary\relax
474   \expandafter\@markextra\@themark{#1}{#2}%
475   \@temptokena \expandafter{\@themark}%
476   \mark{\the\@temptokena}%
477   \endgroup
478   \if@nobreak\ifvmode\nobreak\fi\fi}

\@markextra Internal macro to store the extra marks in the marks storage.
Note: Put #1#2 in toks register.
479 \def\@markextra#1#2#3#4#5#6{\@temptokena {{#1}{#2}}%
480   \unrestored@protected@xdef\@themark{\the\@temptokena{#5}{#6}}}}

\firstleftxmark The new extra marks.
\firstrightxmark 481 \def\firstleftxmark{\expandafter\@leftxmark
\topleftxmark 482   \firstmark\@empty\@empty\@empty\@empty}
\toprightxmark 483 \def\firstrightxmark{\expandafter\@rightxmark
\lastleftxmark 484   \firstmark\@empty\@empty\@empty\@empty}
\lastrightxmark 485 \def\topleftxmark{\expandafter\@leftxmark
\firstxmark 486   \topmark\@empty\@empty\@empty\@empty}
\lastxmark 487 \def\toprightxmark{\expandafter\@rightxmark
\topxmark 488   \topmark\@empty\@empty\@empty\@empty}
489 \def\lastleftxmark{\expandafter\@leftxmark
490   \botmark\@empty\@empty\@empty\@empty}
491 \def\lastrightxmark{\expandafter\@rightxmark
492   \botmark\@empty\@empty\@empty\@empty}
493 \let\firstxmark\firstleftxmark
494 \let\lastxmark\lastrightxmark
495 \let\topxmark\topleftxmark

\@tleftxmark Internal macros to extract the extra marks out of the marks storage.
\@rightxmark 496 \def\@leftxmark#1#2#3#4{#3}
497 \def\@rightxmark#1#2#3#4{#4}

```

</extramarks>

37 fancyheadings.sty

Fancyheadings.sty was the original style file (as they were called then) to implement fancy headers and footers in L^AT_EX. This was in the time when MSDOS was still quite a dominant “Operating System”. It had a nasty property (amongst others): filenames consisted of at most 8 characters + a 3 character extension. This meant that the name ‘fancyheadings.sty’ was internally truncated in MSDOS to ‘fancyhea.sty’, although it was perfectly OK to say ‘fancyheadings’ in L^AT_EX.

However, some people started to write also 'fancyhea' in L^AT_EX documents, which made them unportable to for example Unix systems, unless there a copy or link was made to 'fancyhea.sty'. I found this so annoying that I decided to rename the package to 'fancyhdr.sty'. This package has evolved to a version that is incompatible with the original 'fancyheadings'. Fancyheadings should no longer be used, therefore this package is provided that issues a clear warning and then switches to fancyhdr.

```
<*fancyheadings>
498 \PackageWarningNoLine{fancyheadings}{%
499   Please stop using fancyheadings!\MessageBreak
500   Use fancyhdr instead.\MessageBreak
501   We will call fancyhdr with the very same!\MessageBreak
502   options you passed to fancyheadings.\MessageBreak
503   \MessageBreak
504   fancyhdr is 99 percent compatible with!\MessageBreak
505   fancyheadings. The only incompatibility is!\MessageBreak
506   that \protect\headrulewidth\space and \protect\footrulewidth\space
507   and!\MessageBreak
508   their \protect\plain... versions are no longer length!\MessageBreak
509   parameters, but normal macros (to be changed!\MessageBreak
510   with \protect\renewcommand\space rather than \protect\setlength).}
511 \RequirePackage{fancyhdr}
</fancyheadings>
```

Change History

extramarks v1.99e	extramarks v2.1
General: Added a few % marks to get rid of unwanted spaces, and \endinput.	General: Added a \ProvidesPackage line.
Added LPPL license clause. . . 64	Updated contact information. 64
extramarks v2.0beta	extramarks v3.9
General: Adapted for the new implementation of marks in L ^A T _E X to solve bug latex/3203.	General: Unify version number with fancyhdr.sty. 64
Added symmetric commands \firstrightmark, \lastleftmark, \firstleftxmark, \firstrightxmark, \lastrightxmark, \lastleftxmark, \topleftxmark and \toprightxmark. 64	extramarks v3.9a
	General: Restore \newtoks\@temptokenb 64
fancyhdr v 2.0	fancyhdr v1.4
General: version 2.0 Release. . . . 64	General: Correction for use with \reversemarginpar 49
	fancyhdr v1.5
	General: Added the \iftopfloat, \ifbotfloat and \iffloatpage commands . . . 49
	fancyhdr v1.6
	General: Reset single spacing in headers/footers for use with setspace.sty or doublespace.sty 49

- fancyhdr v1.7
 General: Changed
`\let\@mkboth\markboth` to
`\def\@mkboth{\protect\markboth}`
 to make it more robust. 49
- fancyhdr v1.8
 General: corrections for
 amsbook/amsart: define
`\@chapapp` and (more
 importantly) use the
`\chapter/sectionmark`
 definitions from `\ps@headings` if
 they exist (which should be
 true for all standard classes). . . 49
- fancyhdr v1.9
 General: The proposed
`\renewcommand{\headrulewidth}`
`{\iffloatpage...`
 construction in the doc did not
 work properly with the
`fancyplain` style. 49
- fancyhdr v1.91
 General: The definition of
`\@mkboth` wasn't restored on
 subsequent
`\pagestyle{fancy}`'s. 49
- fancyhdr v1.92
 General: The sequence
`\pagestyle{fancyplain}`
`\pagestyle{plain}`
`\pagestyle{fancy}` would
 erroneously select the plain
 version. 49
- fancyhdr v1.93
 General: `\fancypagestyle`
 command added. 49
- fancyhdr v1.94
 General: (suggested by Conrad
 Hughes
`<chughes@maths.tcd.ie>`):
 added `\footruleskip` to allow
 control over footrule position
 (old hardcoded value of
`.3\normalbaselineskip` is far
 too high when used with very
 small footer fonts). 49
- fancyhdr v1.95
 General: call `\@normalsize` in the
 reset code if that is defined,
 otherwise `\normalsize`. This is
 to solve a problem with
`ucthesis.cls`, as this doesn't
 define `\@currszize`.
 Unfortunately for latex209
 calling `\normalsize` doesn't
 work as this is optimized to do
 very little, so there
`\@normalsize` should be called.
 Hopefully this code works for
 all versions of LaTeX known to
 mankind. 49
- fancyhdr v1.96
 General: Initialise `\headwidth` to a
 magic (negative) value to catch
 most common cases that
 people change it before calling
`\pagestyle{fancy}`. Note it
 can't be initialised when
 reading in this file, because
`\textwidth` could be changed
 afterwards. This is quite
 probable. We also switch to
`\MakeUppercase` rather than
`\uppercase` and introduce a
`\nouppercase` command for
 use in headers. and footers. . . 49
- fancyhdr v1.97
 General: Two changes:
 1. Undo the change in version
 1.8 (using the
`\pagestyle{headings}`
 defaults for the chapter and
 section marks). The current
 version of amsbook and amsart
 classes don't seem to need
 them anymore. Moreover the
 standard L^AT_EX classes don't
 use `\markboth` if twoside isn't
 selected, and this is confusing
 as `\leftmark` doesn't work as
 expected.
 2. Include a call to `\ps@empty`
 in `\ps@@fancy`. This is to solve
 a problem in the amsbook and
 amsart classes, that make
 global changes to `\topskip`,
 which are reset in `\ps@empty`.
 Hopefully this doesn't break
 other things. 49
- fancyhdr v1.98
 General: Added % after the line
`\def\nouppercase` 49

fancyhdr v1.99

General: This is the alpha version of fancyhdr 2.0

Introduced the new commands

`\fancyhead`, `\fancyfoot`, and `\fancyhf`. Changed

`\headrulewidth`,

`\footrulewidth`,

`\footruleskip` to macros

rather than length parameters,

In this way they can be

conditionalized and they don't consume length registers.

There is no need to have them as length registers unless you

want to do calculations with

them, which is unlikely. Note

that this may make some uses

of them incompatible (i.e. if

you have a file that uses

`\setlength` or `\xxxx=`) 49

fancyhdr v1.99a

General: Added a few more %

signs. 49

fancyhdr v1.99b

General: Changed the syntax of

`\f@nch@for` to be resistant to catcode changes of `:=`.

Removed the [1] from the defs

of `\lhead` etc. because the

parameter is consumed by the

`\@[xy]lhead` etc. macros. . . . 49

fancyhdr v1.99c

General: Corrected `\nouppercase`

to also include the protected form of `\MakeUppercase`.

`\global` added to

manipulation of `\headwidth`.

`\iffootnote` command added.

Some comments added about

`\f@nch@head` and

`\f@nch@foot`. 49

fancyhdr v1.99d

General: Changed the default

`\ps@empty` to `\ps@@empty` in

order to allow

`\fancypagestyle{empty}`

redefinition. 49

fancyhdr v2.0

General: Added LPPL license clause.

A check for `\headheight` is

added. An error message is

given (once) if the header is

too large. Empty headers don't

generate the error even if

`\headheight` is very small or

even 0pt.

Warning added for the use of

'E' option when twoside option

is not used. In this case the 'E'

fields will never be used. 49

fancyhdr v2.1beta

General: New command:

`\fancyhfoffset[place]{length}`

defines offsets to be applied to

the header/footer to let it stick

into the margins (if length >

0). `place` is like in

`\fancyhead`, except that only

E,O,L,R can be used. This

replaces the old calculation

based on `\headwidth` and the

marginpar area. `\headwidth`

will be dynamically calculated

in the headers/footers when

this is used. 49

fancyhdr v2.1beta2

General: `\fancyhfoffset` now also

takes H,F as possible letters in

the argument to allow the

header and footer widths to be

different.

New commands

`\fancyheadoffset` and

`\fancyfootoffset` added

comparable to `\fancyhead` and

`\fancyfoot`.

Error messages and warnings

have been made more

informative. 49

fancyhdr v2.1

General: The defaults for

`\footrulewidth`,

`\plainheadrulewidth` and

`\plainfootrulewidth` are

changed from `\z@skip` to 0pt.

In this way when someone

inadvertantly uses `\setlength`

to change any of these, the

value of `\z@skip` will not be

changed, rather an

errormessage will be given. . .	49	<code>\relax</code> , which may be undesirable in some cases. . . .	49
fancyhdr v3.0		fancyhdr v3.4	
General: Release of version 3.0. . .	49	General: Replace <code>\rm</code> by <code>\normalfont\rmfamily</code> and <code>\sl</code> by <code>\normalfont\slshape</code> . . .	49
fancyhdr v3.1		fancyhdr v3.5	
General: Added <code>'\endlinechar=13'</code> to <code>\f@nch@reset</code> to prevent problems with <code>\includegraphics</code> in header/footer when <code>verbatiminput</code> is active.	49	General: Don't define <code>\footruleskip</code> if it is already defined.	49
fancyhdr v3.10		fancyhdr v3.6	
<code>\f@nch@errmsg</code> : Use <code>\f@nch@ifundefined</code> instead of <code>\ifx</code>	52	General: Added a <code>\ProvidesPackage</code> line. Updated contact information. . .	49
Use <code>\newcommand</code> instead of <code>\def</code>	52	fancyhdr v3.7	
<code>\f@nch@foot</code> : Move <code>\footruleskip</code> outside of the <code>\footrule</code> definition.	59	General: Removed <code>\normalfont</code> from default values, as every field is already initialised with <code>\normalfont</code> . Set <code>\hspace</code> to <code>\headwidth</code> in header/footer.	49
Put <code>\footrule</code> in a <code>\vbox</code> to accommodate for flexible footrules.	59	fancyhdr v3.8	
Use <code>\unvbox</code> on the <code>footrule</code> <code>\vbox</code> to preserve vertical spacing.	59	General: Reset <code>\,</code> , <code>\raggedleft</code> , <code>\raggedright</code> and <code>\centering</code> to their default values to avoid a clash with the <code>tabu</code> package. Move the redefinition of <code>\@makecol</code> to <code>\begin{document}</code> to avoid a clash with the <code>footmisc</code> package (and maybe others). Define a working <code>\iffootnote</code> command.	49
<code>\f@nch@vbox</code> : Don't use <code>\global\setlength</code>	58	fancyhdr v3.9	
Use <code>\newcommand</code> instead of <code>\def</code>	58	<code>\cfoot</code> : Let <code>\newcommand</code> do the handling of the optional parameter.	54
<code>\f@nch@warning</code> : Use <code>\f@nch@ifundefined</code> instead of <code>\ifx</code>	52	<code>\chead</code> : Let <code>\newcommand</code> do the handling of the optional parameter.	54
Use <code>\newcommand</code> instead of <code>\def</code>	52	<code>\fancyfoot</code> : Let <code>\newcommand</code> do the handling of the optional parameter.	53
<code>\footrule</code> : Move <code>\footruleskip</code> outside of the <code>\footrule</code> definition and remove useless <code>\vskip</code> at the top.	60	<code>\fancyfootoffset</code> : Let <code>\newcommand</code> do the handling of the optional parameter. . . .	53
fancyhdr v3.2		<code>\fancyhead</code> : Let <code>\newcommand</code> do the handling of the optional parameter.	53
General: Reset <code>\everypar</code> (the real one) in <code>\f@nch@reset</code> because <code>spanish.ldf</code> does strange things with <code>\everypar</code> between « and »	49	<code>\fancyheadoffset</code> : Let <code>\newcommand</code> do the handling	
fancyhdr v3.3			
General: Replace <code>'\@ifundefined{chapter}'</code> with <code>'\ifx\chapter\@undefined'</code> because the former subtly makes <code>\chapter</code> equal to			

of the optional parameter.	53	<code>\f@nch@gb1</code> : Implement the	
<code>\fancyhf</code> : Let <code>\newcommand</code> do the		<code>compatV3</code> option	49
handling of the optional		Remove the <code>\global</code> in	
parameter.	53	definitions	49
<code>\fancyhfoffset</code> : Let <code>\newcommand</code>		<code>\f@nch@head</code> : <code>\fancyheadinit</code>	
do the handling of the optional		initialisation code added and	
parameter.	53	<code>\f@nch@reset</code> moved up.	58
<code>\lfoot</code> : Let <code>\newcommand</code> do the		Parameter <code>\headruleskip</code>	58
handling of the optional		<code>\f@nch@initialise</code> : Put all the	
parameter.	54	initialisation code in	
<code>\lhead</code> : Let <code>\newcommand</code> do the		<code>\f@nch@initialise</code>	60
handling of the optional		<code>\f@nch@ps@empty</code> : Rename	
parameter.	54	<code>\ps@empty</code> to	
<code>\rfoot</code> : Let <code>\newcommand</code> do the		<code>\f@nch@ps@empty</code>	61
handling of the optional		<code>\f@nch@vbox</code> : Don't adjust the	
parameter.	54	<code>\headheight</code> / <code>\footskip</code> ,	
<code>\rhead</code> : Let <code>\newcommand</code> do the		except when the <code>compatV3</code>	
handling of the optional		option is given.	58
parameter.	54	Don't check when the <code>nocheck</code>	
General: Put everything in a <code>.dtx</code>		option is given.	58
file.	49	<code>\fancycenter</code> : Macro	
Rename some macros to have		<code>\fancycenter</code> added	56
' <code>f@nch@</code> ' in their names, to get		<code>\fancypagestyle</code> : Added optional	
a more uniform naming scheme		base style argument.	63
for internal macros.	49	<code>\headruleskip</code> : Parameter	
<code>fancyhdr</code> v4.0		<code>\headruleskip</code>	55
<code>\f@nch@fancycore</code> : Rename		<code>\iff@nch@check</code> : Implement the	
<code>\ps@fancy</code> to		<code>nocheck</code> option	49
<code>\f@nch@fancycore</code>	61	<code>\ps@f@nch@fancyproto</code> :	
<code>\f@nch@foot</code> : <code>\fancyfootinit</code>		Reorganise <code>\ps@fancy</code>	61
initialisation code added and		General: Process package options.	52
<code>\f@nch@reset</code> moved up.	59	<code>ps@fancydefault</code> : Added	
		<code>\ps@fancydefault</code>	62

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols	<code>\@temptokenb</code> <u>443</u>	BIBLIOGRAPHY 18
<code>\@chapapp</code> <u>334</u>	<code>\@themark</code> <u>470</u>	blank page 27
<code>\@leftmark</code> <u>458</u>	<code>\@tleftxmark</code> <u>496</u>	<code>\bottomfraction</code> 39
<code>\@makecol</code> <u>417</u>		<code>bottomnumber</code> 39
<code>\@markboth</code> <u>454</u>	A	
<code>\@markextra</code> <u>479</u>	<code>\afterpage</code> 41	C
<code>\@markright</code> <u>456</u>	<code>afterpage.sty</code> 30, 35	<code>\cfoot</code> 43
<code>\@rightmark</code> <u>458</u>	B	<code>\cfoot</code> <u>175</u>
<code>\@rightxmark</code> <u>496</u>	bible 36	<code>\chaptermark</code> 18, 19, 46
		<code>\chaptermark</code> <u>336</u>

- \chaptername 18
 \thead 43
 \thead 175
 \cleardoublepage .. 27
 \clearpage 27, 29
 concordance 21
 Continued... 33
- D**
- dictionary 21
- E**
- epsf 36
 epsfig 36
 \extramarks 34
 \extramarks 471
- F**
- \f@nch@centering .. 207
 \f@nch@def 13
 \f@nch@default 126
 \f@nch@errmsg 115
 \f@nch@everypar ... 207
 \f@nch@fancycore .. 380
 \f@nch@fancyhf 140
 \f@nch@fancyhfoffs . 157
 \f@nch@foot 314
 \f@nch@footinit ... 258
 \f@nch@for 124
 \f@nch@forc 119
 \f@nch@gb1 8
 \f@nch@head 295
 \f@nch@headinit ... 254
 \f@nch@headwidth .. 187
 \f@nch@ifin 131
 \f@nch@ifundefined . 15
 \f@nch@initialise . 335
 \f@nch@O@elf 188
 \f@nch@O@elh 188
 \f@nch@O@erf 188
 \f@nch@O@erh 188
 \f@nch@O@olf 188
 \f@nch@O@olh 188
 \f@nch@O@orf 188
 \f@nch@O@orh 188
 \f@nch@O@elf 393
 \f@nch@O@elh 393
 \f@nch@O@erf 393
 \f@nch@O@erh 393
 \f@nch@offselh 401, 405
 \f@nch@offsolh 401, 405
- \f@nch@O@olf 393
 \f@nch@O@olh 393
 \f@nch@O@orf 393
 \f@nch@O@orh 393
 \f@nch@pagestyle .. 430
 \f@nch@ps@empty ... 379
 \f@nch@raggedleft . 207
 \f@nch@raggedright . 207
 \f@nch@reset 211
 \f@nch@setoffs 409
 \f@nch@vbox 266
 \f@nch@warning 117
 \fancycenter . 4, 10, 223
 \fancyfoot 10, 12
 \fancyfoot 3, 134
 \fancyfootinit 4, 32, 258
 \fancyfootoffset .. 25
 \fancyfootoffset 3, 137
 \fancyhead 10, 12
 \fancyhead 3, 134
 \fancyheadinit 4, 32, 254
 \fancyheadoffset .. 25
 \fancyheadoffset 3, 137
 \fancyhf 12
 \fancyhf 3, 134
 \fancyhfinit . 4, 32, 262
 \fancyhfoffset 25
 \fancyhfoffset .. 3, 137
 \fancypagestyle ... 13
 \fancypagestyle . 4, 427
 \fancyplain 42
 \fancyplain 205
 \firstleftmark .. 20, 34
 \firstleftmark .. 4, 460
 \firstleftxmark ... 34
 \firstleftxmark ... 481
 \firstrightmark . 4, 460
 \firstrightxmark .. 34
 \firstrightxmark .. 481
 \firstxmark 34
 \firstxmark 481
 float page 26
 \floatpagefraction .
 39, 42
 \footrule 3, 350
 \footruleskip 23
 \footruleskip 3, 23, 200
 \footrulewidth 10, 17, 22
 \footrulewidth 196
 \footskip 21
- G**
- graphics 36
 graphicx 36
- H**
- \headheight 21
 \headrule 3, 347
 \headruleskip 24
 \headruleskip 3, 24, 198
 \headrulewidth 10, 17, 22
 \headrulewidth .. 3, 196
 \headwidth 24
 \headwidth 3, 206
- I**
- \if@fancyplain 204
 \if@nch@empty 1
 \ifbotfloat 4, 423
 \iff@nch@check 2
 \iff@nch@compatViii . 7
 \iff@nch@footnote . 417
 \iffloatpage 4, 423
 \iffootnote 4, 423
 ifthen 21
 \ifthenelse 31, 32
 \iftopfloat 4, 423
 INDEX 19
- L**
- \lastleftmark ... 4, 460
 \lastleftxmark 34
 \lastleftxmark 481
 \lastrightmark 34
 \lastrightmark 460
 \lastrightxmark ... 34
 \lastrightxmark ... 481
 \lastxmark 34
 \lastxmark 481
 \leftmark 17, 18
 \leftmark 460
 \lfoot 43
 \lfoot 175
 \lhead 43
 \lhead 175
 long titles 45
 longtable 41
- M**
- \maketitle 10
 \MakeUppercase 19
 \MakeUppercase 333

-
- margin 35
 - \marginparsep 24
 - \marginparwidth ... 24
 - \markboth 446
 - \markright 454
 - movie 36
 - multi-line 21
 - O**
 - option
 - compatV3 .. 3, 16, 22
 - headings 3, 16
 - myheadings 3, 16
 - nocheck 3, 16, 23, 49
 - Overfull \vbox 21
 - P**
 - page break 35
 - page style
 - changes 29
 - empty 27
 - fancy 9
 - fancydefault ... 14
 - fancyplain 42
 - headings 16, 48
 - myheadings ... 16, 48
 - plain 13, 25
 - \pagenumbering .. 29, 30
 - picture 35, 36
 - \plainfootrulewidth 43
 - \plainfootrulewidth 202
 - \plainheadrulewidth 43
 - \plainheadrulewidth 202
 - \ProcessOptions ... 52
 - \ps@f@nch@fancyproto 367
 - \ps@fancy 375
 - \ps@fancydefault .. 388
 - \ps@fancyplain 377
 - \ps@headings 58
 - R**
 - \rfoot 43
 - \rfoot 175
 - \rhead 43
 - \rhead 175
 - \rightmark 17, 18
 - \rightmark 460
 - S**
 - \sectionmark 18, 47
 - \sectionmark 336
 - \subsectionmark ... 18
 - \subsectionmark ... 336
 - T**
 - \textfraction 39
 - \thechapter 18
 - \thispagestyle .. 10, 30
 - thumb-index 36
 - \topfraction 39
 - \topleftxmark 34
 - \topleftxmark 481
 - topnumber 39
 - \toprightxmark 34
 - \toprightxmark 481
 - \topxmark 481
 - totalnumber 39
 - \truncate 47
 - twoside 11
 - U**
 - \unrestored@protected@xdef 444
 - \uppercase 18, 19
 - W**
 - Warning 22