Standard Document Classes for LaTeX version 2e*

1997/10/10

Contents

*This file has version number v1.3x, last revised 1997/10/10.

1 The DOCSTRIP modules

The following modules are used in the implementation to direct DOCSTRIP in generating the external files:

article	produce the documentclass article
report	produce the documentclass report
size10	produce the class option for 10pt
size11	produce the class option for 11pt
size12	produce the class option for 12pt
book	produce the documentclass book
bk10	produce the book class option for 10pt
bk11	produce the book class option for 11pt
bk12	produce the book class option for 12pt
driver	produce a documentation driver file

2 Initial Code

In this part we define a few commands that are used later on.

\@ptsize This control sequence is used to store the second digit of the pointsize we are typesetting in. So, normally, it's value is one of 0, 1 or 2.

```
⟨*article | report | book⟩
\newcommand\@ptsize{}
```

\if@restonecol When the document has to printed in two columns, we sometimes have to temporarily switch to one column. This switch is used to remember to switch back.

```
\newif\if@restonecol
```

\if@titlepage A switch to indicate if a titlepage has to be produced. For the article document class the default is not to make a separate titlepage.

```
\newif\if@titlepage
⟨article⟩\@titlepagefalse
⟨!article⟩\@titlepagetrue
```

\if@openright A switch to indicate if chapters must start on a right-hand page. The default for the report class is no; for the book class it's yes.

```
⟨!article⟩\newif\if@openright
```

`\if@mainmatter` The switch `\if@mainmatter`, only available in the document class book, indicates whether we are processing the main material in the book.

```
⟨book⟩\newif\if@mainmatter \@mainmattertrue
```

3 Declaration of Options

3.1 Setting Paper Sizes

The variables `\paperwidth` and `\paperheight` should reflect the physical paper size after trimming. For desk printer output this is usually the real paper size since there is no post-processing. Classes for real book production will probably add other paper sizes and additionally the production of crop marks for trimming. In compatibility mode, these (and some of the subsequent) options are disabled, as they were not present in LaTeX2.09.

```
\if@compatibility\else
\DeclareOption{a4paper}
   {\setlength\paperheight {297mm}%
    \setlength\paperwidth  {210mm}}
\DeclareOption{a5paper}
   {\setlength\paperheight {210mm}%
    \setlength\paperwidth  {148mm}}
\DeclareOption{b5paper}
   {\setlength\paperheight {250mm}%
    \setlength\paperwidth  {176mm}}
\DeclareOption{letterpaper}
   {\setlength\paperheight {11in}%
    \setlength\paperwidth  {8.5in}}
\DeclareOption{legalpaper}
   {\setlength\paperheight {14in}%
    \setlength\paperwidth  {8.5in}}
\DeclareOption{executivepaper}
   {\setlength\paperheight {10.5in}%
    \setlength\paperwidth  {7.25in}}
```

The option landscape switches the values of `\paperheight` and `\paperwidth`, assuming the dimensions were given for portrait paper.

```
\DeclareOption{landscape}
   {\setlength\@tempdima   {\paperheight}%
    \setlength\paperheight {\paperwidth}%
    \setlength\paperwidth  {\@tempdima}}
\fi
```

3.2 Choosing the type size

The type size options are handled by defining `\@ptsize` to contain the last digit of the size in question and branching on `\ifcase` statements. This is done for historical reasons to stay compatible with other packages that use the `\@ptsize` variable to select special actions. It makes the declarations of size options less than 10pt difficult, although one can probably use 9 and 8 assuming that a class wont define both 8pt and 18pt options.

```
\if@compatibility
  \renewcommand\@ptsize{0}
\else
\DeclareOption{10pt}{\renewcommand\@ptsize{0}}
\fi
\DeclareOption{11pt}{\renewcommand\@ptsize{1}}
\DeclareOption{12pt}{\renewcommand\@ptsize{2}}
```

3.3 Two-side or one-side printing

For two-sided printing we use the switch `\if@twoside`. In addition we have to set the `\if@mparswitch` to get any margin paragraphs into the outside margin.

```
\if@compatibility\else
\DeclareOption{oneside}{\@twosidefalse \@mparswitchfalse}
\fi
\DeclareOption{twoside}{\@twosidetrue  \@mparswitchtrue}
```

3.4 Draft option

If the user requests draft we show any overfull boxes. We could probably add some more interesting stuff to this option.

```
\DeclareOption{draft}{\setlength\overfullrule{5pt}}
\if@compatibility\else
\DeclareOption{final}{\setlength\overfullrule{0pt}}
\fi
```

3.5 Titlepage option

An article usually has no separate titlepage, but the user can request one.

```
\DeclareOption{titlepage}{\@titlepagetrue}
\if@compatibility\else
\DeclareOption{notitlepage}{\@titlepagefalse}
\fi
```

3.6 openright option

This option determines whether or not a chapter must start on a right-hand page request one.

```
⟨!article⟩\if@compatibility
⟨book⟩\@openrighttrue
⟨!article⟩\else
⟨!article⟩\DeclareOption{openright}{\@openrighttrue}
⟨!article⟩\DeclareOption{openany}{\@openrightfalse}
⟨!article⟩\fi
```

3.7 Twocolumn printing

Two-column and one-column printing is again realized via a switch.

```
\if@compatibility\else
\DeclareOption{onecolumn}{\@twocolumnfalse}
\fi
\DeclareOption{twocolumn}{\@twocolumntrue}
```

3.8 Equation numbering on the left

The option leqno can be used to get the equation numbers on the left side of the equation. It loads code which is generated automatically from the kernel files when the format is built. If the equation number does get a special formatting then instead of using the kernel file the class would need to provide the code explicitly.

```
\DeclareOption{leqno}{\input{leqno.clo}}
```

3.9 Flush left displays

The option fleqn redefines the displayed math environments in such a way that they come out flush left, with an indentation of `\mathindent` from the prevailing left margin. It loads code which is generated automatically from the kernel files when the format is built.

```
\DeclareOption{fleqn}{\input{fleqn.clo}}
```

3.10 Open bibliography

The option openbib produces the "open" bibliography style, in which each block starts on a new line, and succeeding lines in a block are indented by `\bibindent`.

```
\DeclareOption{openbib}{%
```

First some hook into the bibliography environment is filled.

```
  \AtEndOfPackage{%
   \renewcommand\@openbib@code{%
      \advance\leftmargin\bibindent
      \itemindent -\bibindent
      \listparindent \itemindent
      \parsep \z@
      }%
```

In addition the definition of `\newblock` is overwritten.

```
   \renewcommand\newblock{\par}}%
}
```

4 Executing Options

Here we execute the default options to initialize certain variables. Note that the document class ‘book’ always uses two sided printing.

```
⟨*article⟩
\ExecuteOptions{letterpaper,10pt,oneside,onecolumn,final}
⟨/article⟩
⟨*report⟩
\ExecuteOptions{letterpaper,10pt,oneside,onecolumn,final,openany}
⟨/report⟩
⟨*book⟩
\ExecuteOptions{letterpaper,10pt,twoside,onecolumn,final,openright}
⟨/book⟩
```

The `\ProcessOptions` command causes the execution of the code for every option FOO which is declared and for which the user typed the FOO option in his `\documentclass` command. For every option BAR he typed, which is not declared, the option is assumed to be a global option. All options will be passed as document options to any `\usepackage` command in the document preamble.

```
\ProcessOptions
```

Now that all the options have been executed we can load the chosen class option file that contains all size dependent code.

```
⟨!book⟩\input{size1\@ptsize.clo}
⟨book⟩\input{bk1\@ptsize.clo}
⟨/article | report | book⟩
```

5 Loading Packages

The standard class files do not load additional packages.

6 Document Layout

In this section we are finally dealing with the nasty typographical details.

6.1 Fonts

LaTeX offers the user commands to change the size of the font, relative to the ‘main’ size. Each relative size changing command `\size` executes the command `\@setfontsize\size`⟨*font-size*⟩⟨*baselineskip*⟩ where:

⟨***font-size***⟩ The absolute size of the font to use from now on.

⟨***baselineskip***⟩ The normal value of `\baselineskip` for the size of the font selected. (The actual value will be `\baselinestretch` * ⟨*baselineskip*⟩.)

A number of commands, defined in the LaTeX kernel, shorten the following definitions and are used throughout. They are:

`\@vpt`	5	`\@vipt`	6	`\@viipt`	7
`\@viiipt`	8	`\@ixpt`	9	`\@xpt`	10
`\@xipt`	10.95	`\@xiipt`	12	`\@xivpt`	14.4
...					

`\normalsize`
`\@normalsize`

The user level command for the main size is `\normalsize`. Internally LaTeX uses `\@normalsize` when it refers to the main size. `\@normalsize` will be defined to work like `\normalsize` if the latter is redefined from its default definition (that just issues an error message). Otherwise `\@normalsize` simply selects a 10pt/12pt size.

The `\normalsize` macro also sets new values for `\abovedisplayskip`, `\abovedisplayshortskip` and `\belowdisplayshortskip`.

```
⟨*10pt | 11pt | 12pt⟩
\renewcommand\normalsize{%
⟨*10pt⟩
   \@setfontsize\normalsize\@xpt\@xiipt
   \abovedisplayskip 10\p@ \@plus2\p@ \@minus5\p@
   \abovedisplayshortskip \z@ \@plus3\p@
   \belowdisplayshortskip 6\p@ \@plus3\p@ \@minus3\p@
⟨/10pt⟩
⟨*11pt⟩
   \@setfontsize\normalsize\@xipt{13.6}%
   \abovedisplayskip 11\p@ \@plus3\p@ \@minus6\p@
   \abovedisplayshortskip \z@ \@plus3\p@
   \belowdisplayshortskip 6.5\p@ \@plus3.5\p@ \@minus3\p@
⟨/11pt⟩
⟨*12pt⟩
   \@setfontsize\normalsize\@xiipt{14.5}%
   \abovedisplayskip 12\p@ \@plus3\p@ \@minus7\p@
   \abovedisplayshortskip \z@ \@plus3\p@
   \belowdisplayshortskip 6.5\p@ \@plus3.5\p@ \@minus3\p@
⟨/12pt⟩
```

The `\belowdisplayskip` is always equal to the `\abovedisplayskip`. The parameters of the first level list are always given by `\@listI`.

```
   \belowdisplayskip \abovedisplayskip
   \let\@listi\@listI}
```

We initially choose the normalsize font.

```
\normalsize
```

`\small`

This is similar to `\normalsize`.

```
\newcommand\small{%
⟨*10pt⟩
   \@setfontsize\small\@ixpt{11}%
   \abovedisplayskip 8.5\p@ \@plus3\p@ \@minus4\p@
   \abovedisplayshortskip \z@ \@plus2\p@
   \belowdisplayshortskip 4\p@ \@plus2\p@ \@minus2\p@
   \def\@listi{\leftmargin\leftmargini
```

```
               \topsep 4\p@ \@plus2\p@ \@minus2\p@
               \parsep 2\p@ \@plus\p@ \@minus\p@
               \itemsep \parsep}%
⟨/10pt⟩
⟨*11pt⟩
   \@setfontsize\small\@xpt\@xiipt
   \abovedisplayskip 10\p@ \@plus2\p@ \@minus5\p@
   \abovedisplayshortskip \z@ \@plus3\p@
   \belowdisplayshortskip 6\p@ \@plus3\p@ \@minus3\p@
   \def\@listi{\leftmargin\leftmargini
               \topsep 6\p@ \@plus2\p@ \@minus2\p@
               \parsep 3\p@ \@plus2\p@ \@minus\p@
               \itemsep \parsep}%
⟨/11pt⟩
⟨*12pt⟩
   \@setfontsize\small\@xipt{13.6}%
   \abovedisplayskip 11\p@ \@plus3\p@ \@minus6\p@
   \abovedisplayshortskip \z@ \@plus3\p@
   \belowdisplayshortskip 6.5\p@ \@plus3.5\p@ \@minus3\p@
   \def\@listi{\leftmargin\leftmargini
               \topsep 9\p@ \@plus3\p@ \@minus5\p@
               \parsep 4.5\p@ \@plus2\p@ \@minus\p@
               \itemsep \parsep}%
⟨/12pt⟩
   \belowdisplayskip \abovedisplayskip
}
```

\footnotesize This is similar to \normalsize.

```
\newcommand\footnotesize{%
⟨*10pt⟩
   \@setfontsize\footnotesize\@viiipt{9.5}%
   \abovedisplayskip 6\p@ \@plus2\p@ \@minus4\p@
   \abovedisplayshortskip \z@ \@plus\p@
   \belowdisplayshortskip 3\p@ \@plus\p@ \@minus2\p@
   \def\@listi{\leftmargin\leftmargini
               \topsep 3\p@ \@plus\p@ \@minus\p@
               \parsep 2\p@ \@plus\p@ \@minus\p@
               \itemsep \parsep}%
⟨/10pt⟩
⟨*11pt⟩
   \@setfontsize\footnotesize\@ixpt{11}%
   \abovedisplayskip 8\p@ \@plus2\p@ \@minus4\p@
   \abovedisplayshortskip \z@ \@plus\p@
   \belowdisplayshortskip 4\p@ \@plus2\p@ \@minus2\p@
   \def\@listi{\leftmargin\leftmargini
               \topsep 4\p@ \@plus2\p@ \@minus2\p@
               \parsep 2\p@ \@plus\p@ \@minus\p@
               \itemsep \parsep}%
⟨/11pt⟩
⟨*12pt⟩
```

```
    \@setfontsize\footnotesize\@xpt\@xiipt
    \abovedisplayskip 10\p@ \@plus2\p@ \@minus5\p@
    \abovedisplayshortskip \z@ \@plus3\p@
    \belowdisplayshortskip 6\p@ \@plus3\p@ \@minus3\p@
    \def\@listi{\leftmargin\leftmargini
                \topsep 6\p@ \@plus2\p@ \@minus2\p@
                \parsep 3\p@ \@plus2\p@ \@minus\p@
                \itemsep \parsep}%
⟨/12pt⟩
    \belowdisplayskip \abovedisplayskip
}
⟨/10pt | 11pt | 12pt⟩
```

\scriptsize \tiny \large \Large \LARGE \huge \Huge

These are all much simpler than the previous macros, they just select a new fontsize, but leave the parameters for displays and lists alone.

```
⟨*10pt⟩
\newcommand\scriptsize{\@setfontsize\scriptsize\@viipt\@viiipt}
\newcommand\tiny{\@setfontsize\tiny\@vpt\@vipt}
\newcommand\large{\@setfontsize\large\@xiipt{14}}
\newcommand\Large{\@setfontsize\Large\@xivpt{18}}
\newcommand\LARGE{\@setfontsize\LARGE\@xviipt{22}}
\newcommand\huge{\@setfontsize\huge\@xxpt{25}}
\newcommand\Huge{\@setfontsize\Huge\@xxvpt{30}}
⟨/10pt⟩
⟨*11pt⟩
\newcommand\scriptsize{\@setfontsize\scriptsize\@viiipt{9.5}}
\newcommand\tiny{\@setfontsize\tiny\@vipt\@viipt}
\newcommand\large{\@setfontsize\large\@xiipt{14}}
\newcommand\Large{\@setfontsize\Large\@xivpt{18}}
\newcommand\LARGE{\@setfontsize\LARGE\@xviipt{22}}
\newcommand\huge{\@setfontsize\huge\@xxpt{25}}
\newcommand\Huge{\@setfontsize\Huge\@xxvpt{30}}
⟨/11pt⟩
⟨*12pt⟩
\newcommand\scriptsize{\@setfontsize\scriptsize\@viiipt{9.5}}
\newcommand\tiny{\@setfontsize\tiny\@vipt\@viipt}
\newcommand\large{\@setfontsize\large\@xivpt{18}}
\newcommand\Large{\@setfontsize\Large\@xviipt{22}}
\newcommand\LARGE{\@setfontsize\LARGE\@xxpt{25}}
\newcommand\huge{\@setfontsize\huge\@xxvpt{30}}
\let\Huge=\huge
⟨/12pt⟩
```

6.2 Paragraphing

\lineskip \normallineskip

These parameters control TEX's behaviour when two lines tend to come too close together.

```
⟨*article | report | book⟩
\setlength\lineskip{1\p@}
```

```
\setlength\normallineskip{1\p@}
```

`\baselinestretch`

This is used as a multiplier for `\baselineskip`. The default is to *not* stretch the baselines. Note that if this command doesn't resolve to "empty" any `plus` or `minus` part in the specification of `\baselineskip` is ignored.

```
\renewcommand\baselinestretch{}
```

`\parskip`
`\parindent`

`\parskip` gives extra vertical space between paragraphs and `\parindent` is the width of the paragraph indentation. The value of `\parindent` depends on whether we are in two column mode.

```
\setlength\parskip{0\p@ \@plus \p@}
⟨/article | report | book⟩
⟨*10pt | 11pt | 12pt⟩
\if@twocolumn
  \setlength\parindent{1em}
\else
⟨10pt⟩  \setlength\parindent{15\p@}
⟨11pt⟩  \setlength\parindent{17\p@}
⟨12pt⟩  \setlength\parindent{1.5em}
\fi
⟨/10pt | 11pt | 12pt⟩
```

`\smallskipamount`
`\medskipamount`
`\bigskipamount`

The values for these three parameters are set in the LaTeX kernel. They should perhaps vary, according to the size option specified. But as they have always had the same value regardless of the size option we do not change them to stay compatible with both LaTeX 2.09 and older releases of LaTeX 2_ε.

```
⟨*10pt | 11pt | 12pt⟩
\setlength\smallskipamount{3\p@ \@plus 1\p@ \@minus 1\p@}
\setlength\medskipamount{6\p@ \@plus 2\p@ \@minus 2\p@}
\setlength\bigskipamount{12\p@ \@plus 4\p@ \@minus 4\p@}
⟨/10pt | 11pt | 12pt⟩
```

`\@lowpenalty`
`\@medpenalty`
`\@highpenalty`

The commands `\nopagebreak` and `\nolinebreak` put in penalties to discourage these breaks at the point they are put in. They use `\@lowpenalty`, `\@medpenalty` or `\@highpenalty`, dependent on their argument.

```
⟨*article | report | book⟩
\@lowpenalty   51
\@medpenalty  151
\@highpenalty 301
```

`\clubpenalty`
`\widowpenalty`

These penalties are use to discourage club and widow lines. Because we use their default values we only show them here, commented out.

```
% \clubpenalty  150
% \widowpenalty 150
```

`\displaywidowpenalty`
`\predisplaypenalty`
`\postdisplaypenalty`

Discourage (but not so much) widows in front of a math display and forbid breaking directly in front of a display. Allow break after a display without a penalty. Again the default values are used, therefore we only show them here.

```
% \displaywidowpenalty 50
% \predisplaypenalty   10000
% \postdisplaypenalty  0
```

`\interlinepenalty` Allow the breaking of a page in the middle of a paragraph.

```
% \interlinepenalty 0
```

`\brokenpenalty` We allow the breaking of a page after a hyphenated line.

```
% \brokenpenalty 100
⟨/article | report | book⟩
```

6.3 Page Layout

All margin dimensions are measured from a point one inch from the top and lefthand side of the page.

6.3.1 Vertical spacing

`\headheight` `\headsep` `\topskip` The `\headheight` is the height of the box that will contain the running head. The `\headsep` is the distance between the bottom of the running head and the top of the text. The `\topskip` is the `\baselineskip` for the first line on a page; LaTeX's output routine will not work properly if it has the value 0pt, so do not do that!

```
⟨*10pt | 11pt | 12pt⟩
\setlength\headheight{12\p@}
⟨!bk⟩ \setlength\headsep    {25\p@}
⟨10pt & bk⟩ \setlength\headsep    {.25in}
⟨11pt & bk⟩ \setlength\headsep    {.275in}
⟨12pt & bk⟩ \setlength\headsep    {.275in}
⟨10pt⟩ \setlength\topskip    {10\p@}
⟨11pt⟩ \setlength\topskip    {11\p@}
⟨12pt⟩ \setlength\topskip    {12\p@}
```

`\footskip` The distance from the baseline of the box which contains the running footer to the baseline of last line of text is controlled by the `\footskip`.

```
⟨!bk⟩ \setlength\footskip{30\p@}
⟨10pt & bk⟩ \setlength\footskip{.35in}
⟨11pt & bk⟩ \setlength\footskip{.38in}
⟨12pt & bk⟩ \setlength\footskip{30\p@}
```

`\maxdepth` The TeX primitive register `\maxdepth` has a function that is similar to that of `\topskip`. The register `\@maxdepth` should always contain a copy of `\maxdepth`. This is achieved by setting it internally at `\begin{document}`. In both plain TeX and LaTeX 2.09 `\maxdepth` had a fixed value of `4pt`; in native LaTeX2e mode we let the value depend on the typesize. We set it so that `\maxdepth` + `\topskip` = typesize ×1.5. As it happens, in these classes `\topskip` is equal to the typesize, therefore we set `\maxdepth` to half the value of `\topskip`.

```
\if@compatibility \setlength\maxdepth{4\p@} \else
\setlength\maxdepth{.5\topskip} \fi
```

6.3.2 The dimension of text

\textwidth When we are in compatibility mode we have to make sure that the dimensions of the printed area are not different from what the user was used to see.

```
\if@compatibility
  \if@twocolumn
    \setlength\textwidth{410\p@}
  \else
⟨10pt&!bk⟩      \setlength\textwidth{345\p@}
⟨11pt&!bk⟩      \setlength\textwidth{360\p@}
⟨12pt&!bk⟩      \setlength\textwidth{390\p@}
⟨10pt & bk⟩      \setlength\textwidth{4.5in}
⟨11pt & bk⟩      \setlength\textwidth{5in}
⟨12pt & bk⟩      \setlength\textwidth{5in}
  \fi
```

When we are not in compatibility mode we can set some of the dimensions differently, taking into account the paper size for instance.

```
\else
```

First, we calculate the maximum `\textwidth`, which we will allow on the selected paper and store it in `\@tempdima`. Then we store the length of a line with approximately 60–70 characters in `\@tempdimb`. The values given are more or less suitable when Computer Modern fonts are used.

```
  \setlength\@tempdima{\paperwidth}
  \addtolength\@tempdima{-2in}
⟨10pt⟩  \setlength\@tempdimb{345\p@}
⟨11pt⟩  \setlength\@tempdimb{360\p@}
⟨12pt⟩  \setlength\@tempdimb{390\p@}
```

Now we can set the `\textwidth`, depending on whether we will be setting one or two columns.

In two column mode each *column* shouldn't be wider than `\@tempdimb` (which could happen on A3 paper for instance).

```
  \if@twocolumn
    \ifdim\@tempdima>2\@tempdimb\relax
      \setlength\textwidth{2\@tempdimb}
    \else
      \setlength\textwidth{\@tempdima}
    \fi
```

In one column mode the text should not be wider than the minimum of the paperwidth (minus 2 inches for the margins) and the maximum length of a line as defined by the number of characters.

```
  \else
    \ifdim\@tempdima>\@tempdimb\relax
      \setlength\textwidth{\@tempdimb}
    \else
      \setlength\textwidth{\@tempdima}
    \fi
  \fi
\fi
```

Here we modify the width of the text a little to be a whole number of points.

```
\if@compatibility\else
  \@settopoint\textwidth
\fi
```

`\textheight` Now that we have computed the width of the text, we have to take care of the height. The `\textheight` is the height of text (including footnotes and figures, excluding running head and foot).

First make sure that the compatibility mode gets the same dimensions as we had with LaTeX2.09. The number of lines was calculated as the floor of the old `\textheight` minus `\topskip`, divided by `\baselineskip` for `\normalsize`. The old value of `\textheight` was 528pt.

```
\if@compatibility
⟨10pt&!bk⟩   \setlength\textheight{43\baselineskip}
⟨10pt & bk⟩  \setlength\textheight{41\baselineskip}
⟨11pt⟩  \setlength\textheight{38\baselineskip}
⟨12pt⟩  \setlength\textheight{36\baselineskip}
```

Again we compute this, depending on the papersize and depending on the baselineskip that is used, in order to have a whole number of lines on the page.

```
\else
  \setlength\@tempdima{\paperheight}
```

We leave at least a 1 inch margin on the top and the bottom of the page.

```
  \addtolength\@tempdima{-2in}
```

We also have to leave room for the running headers and footers.

```
  \addtolength\@tempdima{-1.5in}
```

Then we divide the result by the current `\baselineskip` and store this in the count register `\@tempcnta`, which then contains the number of lines that fit on this page.

```
  \divide\@tempdima\baselineskip
  \@tempcnta=\@tempdima
```

From this we can calculate the height of the text.

```
  \setlength\textheight{\@tempcnta\baselineskip}
\fi
```

The first line on the page has a height of `\topskip`.

```
\addtolength\textheight{\topskip}
```

6.3.3 Margins

Most of the values of these parameters are now calculated, based on the papersize in use. In the calcluations the `\marginparsep` needs to be taken into account so we give it its value first.

\marginparsep
\marginparpush

The horizontal space between the main text and marginal notes is determined by \marginparsep, the minimum vertical separation between two marginal notes is controlled by \marginparpush.

```
\if@twocolumn
  \setlength\marginparsep {10\p@}
\else
⟨10pt&!bk⟩   \setlength\marginparsep{11\p@}
⟨11pt&!bk⟩   \setlength\marginparsep{10\p@}
⟨12pt&!bk⟩   \setlength\marginparsep{10\p@}
⟨bk⟩   \setlength\marginparsep{7\p@}
\fi
⟨10pt | 11pt⟩\setlength\marginparpush{5\p@}
⟨12pt⟩\setlength\marginparpush{7\p@}
```

Now we can give the values for the other margin parameters. For native $\LaTeX\,2_\varepsilon$, these are calculated.

\oddsidemargin
\evensidemargin
\marginparwidth

First we give the values for the compatibility mode.

Values for two-sided printing:

```
\if@compatibility
⟨*bk⟩
⟨10pt⟩    \setlength\oddsidemargin   {.5in}
⟨11pt⟩    \setlength\oddsidemargin   {.25in}
⟨12pt⟩    \setlength\oddsidemargin   {.25in}
⟨10pt⟩    \setlength\evensidemargin  {1.5in}
⟨11pt⟩    \setlength\evensidemargin  {1.25in}
⟨12pt⟩    \setlength\evensidemargin  {1.25in}
⟨10pt⟩    \setlength\marginparwidth {.75in}
⟨11pt⟩    \setlength\marginparwidth {1in}
⟨12pt⟩    \setlength\marginparwidth {1in}
⟨/bk⟩
⟨*!bk⟩
  \if@twoside
⟨10pt⟩      \setlength\oddsidemargin   {44\p@}
⟨11pt⟩      \setlength\oddsidemargin   {36\p@}
⟨12pt⟩      \setlength\oddsidemargin   {21\p@}
⟨10pt⟩      \setlength\evensidemargin  {82\p@}
⟨11pt⟩      \setlength\evensidemargin  {74\p@}
⟨12pt⟩      \setlength\evensidemargin  {59\p@}
⟨10pt⟩      \setlength\marginparwidth {107\p@}
⟨11pt⟩      \setlength\marginparwidth {100\p@}
⟨12pt⟩      \setlength\marginparwidth {85\p@}
```

Values for one-sided printing:

```
  \else
⟨10pt⟩      \setlength\oddsidemargin   {63\p@}
⟨11pt⟩      \setlength\oddsidemargin   {54\p@}
⟨12pt⟩      \setlength\oddsidemargin   {39.5\p@}
⟨10pt⟩      \setlength\evensidemargin  {63\p@}
⟨11pt⟩      \setlength\evensidemargin  {54\p@}
```

```
⟨12pt⟩      \setlength\evensidemargin  {39.5\p@}
⟨10pt⟩      \setlength\marginparwidth  {90\p@}
⟨11pt⟩      \setlength\marginparwidth  {83\p@}
⟨12pt⟩      \setlength\marginparwidth  {68\p@}
  \fi
⟨/!bk⟩
```

And values for two column mode:

```
  \if@twocolumn
     \setlength\oddsidemargin  {30\p@}
     \setlength\evensidemargin {30\p@}
     \setlength\marginparwidth {48\p@}
  \fi
```

When we are not in compatibility mode we can take the dimensions of the selected paper into account.

The values for `\oddsidemargin` and `\marginparwidth` will be set depending on the status of the `\if@twoside`.

If `@twoside` is true (which is always the case for book) we make the inner margin smaller than the outer one.

```
\else
  \if@twoside
    \setlength\@tempdima        {\paperwidth}
    \addtolength\@tempdima      {-\textwidth}
    \setlength\oddsidemargin    {.4\@tempdima}
    \addtolength\oddsidemargin  {-1in}
```

The width of the margin for text is set to the remainder of the width except for a 'real margin' of white space of width 0.4in. A check should perhaps be built in to ensure that the (text) margin width does not get too small!

```
    \setlength\marginparwidth   {.6\@tempdima}
    \addtolength\marginparwidth {-\marginparsep}
    \addtolength\marginparwidth {-0.4in}
```

For one-sided printing we center the text on the page, by calculating the difference between `\textwidth` and `\paperwidth`. Half of that difference is than used for the margin (thus `\oddsidemargin` is `1in` less).

```
  \else
    \setlength\@tempdima        {\paperwidth}
    \addtolength\@tempdima      {-\textwidth}
    \setlength\oddsidemargin    {.5\@tempdima}
    \addtolength\oddsidemargin  {-1in}
    \setlength\marginparwidth   {.5\@tempdima}
    \addtolength\marginparwidth {-\marginparsep}
    \addtolength\marginparwidth {-0.4in}
    \addtolength\marginparwidth {-.4in}
  \fi
```

With the above algorithm the `\marginparwidth` can come out quite large which we may not want.

```
  \ifdim \marginparwidth >2in
     \setlength\marginparwidth{2in}
  \fi
```

Having done these calculations we make them pt values.

```
  \@settopoint\oddsidemargin
  \@settopoint\marginparwidth
```

The \evensidemargin can now be computed from the values set above.

```
  \setlength\evensidemargin  {\paperwidth}
  \addtolength\evensidemargin{-2in}
  \addtolength\evensidemargin{-\textwidth}
  \addtolength\evensidemargin{-\oddsidemargin}
```

Setting \evensidemargin to a full point value may produce a small error. However it will lie within the error range a doublesided printer of today's technology can accurately print.

```
  \@settopoint\evensidemargin
\fi
```

\topmargin The \topmargin is the distance between the top of 'the printable area'—which is 1 inch below the top of the paper–and the top of the box which contains the running head.

It can now be computed from the values set above.

```
\if@compatibility
⟨!bk⟩   \setlength\topmargin{27pt}
⟨10pt & bk⟩   \setlength\topmargin{.75in}
⟨11pt & bk⟩   \setlength\topmargin{.73in}
⟨12pt & bk⟩   \setlength\topmargin{.73in}
\else
  \setlength\topmargin{\paperheight}
  \addtolength\topmargin{-2in}
  \addtolength\topmargin{-\headheight}
  \addtolength\topmargin{-\headsep}
  \addtolength\topmargin{-\textheight}
  \addtolength\topmargin{-\footskip}     % this might be wrong!
```

By changing the factor in the next line the complete page can be shifted vertically.

```
  \addtolength\topmargin{-.5\topmargin}
  \@settopoint\topmargin
\fi
```

6.3.4 Footnotes

\footnotesep \footnotesep is the height of the strut placed at the beginning of every footnote. It equals the height of a normal \footnotesize strut in this class, thus no extra space occurs between footnotes.

```
⟨10pt⟩\setlength\footnotesep{6.65\p@}
⟨11pt⟩\setlength\footnotesep{7.7\p@}
⟨12pt⟩\setlength\footnotesep{8.4\p@}
```

`\footins` `\skip\footins` is the space between the last line of the main text and the top of the first footnote.

```
⟨10pt⟩\setlength{\skip\footins}{9\p@ \@plus 4\p@ \@minus 2\p@}
⟨11pt⟩\setlength{\skip\footins}{10\p@ \@plus 4\p@ \@minus 2\p@}
⟨12pt⟩\setlength{\skip\footins}{10.8\p@ \@plus 4\p@ \@minus 2\p@}
⟨/10pt | 11pt | 12pt⟩
```

6.3.5 Float placement parameters

All float parameters are given default values in the LaTeX 2_ε kernel. For this reason parameters that are not counters need to be set with `\renewcommand`.

Limits for the placement of floating objects

`\c@topnumber` The *topnumber* counter holds the maximum number of floats that can appear on the top of a text page.

```
⟨*article | report | book⟩
\setcounter{topnumber}{2}
```

`\topfraction` This indicates the maximum part of a text page that can be occupied by floats at the top.

```
\renewcommand\topfraction{.7}
```

`\c@bottomnumber` The *bottomnumber* counter holds the maximum number of floats that can appear on the bottom of a text page.

```
\setcounter{bottomnumber}{1}
```

`\bottomfraction` This indicates the maximum part of a text page that can be occupied by floats at the bottom.

```
\renewcommand\bottomfraction{.3}
```

`\c@totalnumber` This indicates the maximum number of floats that can appear on any text page.

```
\setcounter{totalnumber}{3}
```

`\textfraction` This indicates the minimum part of a text page that has to be occupied by text.

```
\renewcommand\textfraction{.2}
```

`\floatpagefraction` This indicates the minimum part of a page that has to be occupied by floating objects before a ‘float page’ is produced.

```
\renewcommand\floatpagefraction{.5}
```

`\c@dbltopnumber` The *dbltopnumber* counter holds the maximum number of two column floats that can appear on the top of a two column text page.

```
\setcounter{dbltopnumber}{2}
```

`\dbltopfraction` This indicates the maximum part of a two column text page that can be occupied by two column floats at the top.

```
\renewcommand\dbltopfraction{.7}
```

`\dblfloatpagefraction` This indicates the minimum part of a page that has to be occupied by two column wide floating objects before a ‘float page’ is produced.

```
\renewcommand\dblfloatpagefraction{.5}
⟨/article | report | book⟩
```

Floats on a text page

`\floatsep`
`\textfloatsep`
`\intextsep`

When a floating object is placed on a page with text, these parameters control the separation between the float and the other objects on the page. These parameters are used for both one-column mode and single-column floats in two-column mode.

`\floatsep` is the space between adjacent floats that are moved to the top or bottom of the text page.

`\textfloatsep` is the space between the main text and floats at the top or bottom of the page.

`\intextsep` is the space between in-text floats and the text.

```
⟨*10pt⟩
\setlength\floatsep    {12\p@ \@plus 2\p@ \@minus 2\p@}
\setlength\textfloatsep{20\p@ \@plus 2\p@ \@minus 4\p@}
\setlength\intextsep   {12\p@ \@plus 2\p@ \@minus 2\p@}
⟨/10pt⟩
⟨*11pt⟩
\setlength\floatsep    {12\p@ \@plus 2\p@ \@minus 2\p@}
\setlength\textfloatsep{20\p@ \@plus 2\p@ \@minus 4\p@}
\setlength\intextsep   {12\p@ \@plus 2\p@ \@minus 2\p@}
⟨/11pt⟩
⟨*12pt⟩
\setlength\floatsep    {12\p@ \@plus 2\p@ \@minus 4\p@}
\setlength\textfloatsep{20\p@ \@plus 2\p@ \@minus 4\p@}
\setlength\intextsep   {14\p@ \@plus 4\p@ \@minus 4\p@}
⟨/12pt⟩
```

`\dblfloatsep`
`\dbltextfloatsep`

When floating objects that span the whole `\textwidth` are placed on a text page when we are in twocolumn mode the separation between the float and the text is controlled by `\dblfloatsep` and `\dbltextfloatsep`.

`\dblfloatsep` is the space between adjacent floats that are moved to the top or bottom of the text page.

`\dbltextfloatsep` is the space between the main text and floats at the top or bottom of the page.

```
⟨*10pt⟩
\setlength\dblfloatsep    {12\p@ \@plus 2\p@ \@minus 2\p@}
\setlength\dbltextfloatsep{20\p@ \@plus 2\p@ \@minus 4\p@}
⟨/10pt⟩
⟨*11pt⟩
\setlength\dblfloatsep    {12\p@ \@plus 2\p@ \@minus 2\p@}
\setlength\dbltextfloatsep{20\p@ \@plus 2\p@ \@minus 4\p@}
⟨/11pt⟩
⟨*12pt⟩
\setlength\dblfloatsep    {14\p@ \@plus 2\p@ \@minus 4\p@}
```

```
\setlength\dbltextfloatsep{20\p@ \@plus 2\p@ \@minus 4\p@}
⟨/12pt⟩
```

Floats on their own page or column

\@fptop
\@fpsep
\@fpbot

When floating objects are placed on separate pages the layout of such pages is controlled by these parameters. At the top of the page `\@fptop` amount of stretchable whitespace is inserted, at the bottom of the page we get an `\@fpbot` amount of stretchable whitespace. Between adjacent floats the `\@fpsep` is inserted.

These parameters are used for the placement of floating objects in one column mode, or in single column floats in two column mode.

Note that at least one of the two parameters `\@fptop` and `\@fpbot` should contain a `plus ...fil` to allow filling the remaining empty space.

```
⟨*10pt⟩
\setlength\@fptop{0\p@ \@plus 1fil}
\setlength\@fpsep{8\p@ \@plus 2fil}
\setlength\@fpbot{0\p@ \@plus 1fil}
⟨/10pt⟩
⟨*11pt⟩
\setlength\@fptop{0\p@ \@plus 1fil}
\setlength\@fpsep{8\p@ \@plus 2fil}
\setlength\@fpbot{0\p@ \@plus 1fil}
⟨/11pt⟩
⟨*12pt⟩
\setlength\@fptop{0\p@ \@plus 1fil}
\setlength\@fpsep{10\p@ \@plus 2fil}
\setlength\@fpbot{0\p@ \@plus 1fil}
⟨/12pt⟩
```

\@dblfptop
\@dblfpsep
\@dblfpbot

Double column floats in two column mode are handled with similar parameters.

```
⟨*10pt⟩
\setlength\@dblfptop{0\p@ \@plus 1fil}
\setlength\@dblfpsep{8\p@ \@plus 2fil}
\setlength\@dblfpbot{0\p@ \@plus 1fil}
⟨/10pt⟩
⟨*11pt⟩
\setlength\@dblfptop{0\p@ \@plus 1fil}
\setlength\@dblfpsep{8\p@ \@plus 2fil}
\setlength\@dblfpbot{0\p@ \@plus 1fil}
⟨/11pt⟩
⟨*12pt⟩
\setlength\@dblfptop{0\p@ \@plus 1fil}
\setlength\@dblfpsep{10\p@ \@plus 2fil}
\setlength\@dblfpbot{0\p@ \@plus 1fil}
⟨/12pt⟩
⟨*article | report | book⟩
```

6.4 Page Styles

The page style *foo* is defined by defining the command `\ps@foo`. This command should make only local definitions. There should be no stray spaces in the definition, since they could lead to mysterious extra spaces in the output (well, that's something that should be always avoided).

`\@evenhead` `\@oddhead` `\@evenfoot` `\@oddfoot`

The `\ps@...` command defines the macros `\@oddhead`, `\@oddfoot`, `\@evenhead`, and `\@evenfoot` to define the running heads and feet—e.g., `\@oddhead` is the macro to produce the contents of the heading box for odd-numbered pages. It is called inside an `\hbox` of width `\textwidth`.

6.4.1 Marking conventions

To make headings determined by the sectioning commands, the page style defines the commands `\chaptermark`, `\sectionmark`, . . . ,
where `\chaptermark{`⟨*TEXT*⟩`}` is called by `\chapter` to set a mark, and so on.

The `\...mark` commands and the `\...head` macros are defined with the help of the following macros. (All the `\...mark` commands should be initialized to no-ops.)

LaTeX extends TeX's `\mark` facility by producing two kinds of marks, a 'left' and a 'right' mark, using the following commands:

`\markboth{`⟨*LEFT*⟩`}{`⟨*RIGHT*⟩`}`: Adds both marks.
`\markright{`⟨*RIGHT*⟩`}`: Adds a 'right' mark.
`\leftmark`: Used in the `\@oddhead`, `\@oddfoot`, `\@evenhead` or `\@evenfoot` macros, it gets the current 'left' mark. `\leftmark` works like TeX's `\botmark` command.
`\rightmark`: Used in the `\@oddhead`, `\@oddfoot`, `\@evenhead` or `\@evenfoot` macros, it gets the current 'right' mark. `\rightmark` works like TeX's `\firstmark` command.

The marking commands work reasonably well for right marks 'numbered within' left marks—e.g., the left mark is changed by a `\chapter` command and the right mark is changed by a `\section` command. However, it does produce somewhat anomalous results if two `\markboth`'s occur on the same page.

Commands like `\tableofcontents` that should set the marks in some page styles use a `\@mkboth` command, which is `\let` by the pagestyle command (`\ps@...`) to `\markboth` for setting the heading or to `\@gobbletwo` to do nothing.

6.4.2 Defining the page styles

The pagestyles *empty* and *plain* are defined in `latex.dtx`.

`\ps@headings`

The definition of the page style *headings* has to be different for two sided printing than it is for one sided printing.

```
\if@twoside
  \def\ps@headings{%
```

The running feet are empty in this page style, the running head contains the page number and one of the marks.

```
    \let\@oddfoot\@empty\let\@evenfoot\@empty
    \def\@evenhead{\thepage\hfil\slshape\leftmark}%
    \def\@oddhead{{\slshape\rightmark}\hfil\thepage}%
```

When using this page style, the contents of the running head is determined by the chapter and section titles. So we `\let \@mkboth` to `\markboth`.

```
    \let\@mkboth\markboth
```

For the article document class we define `\sectionmark` to clear the right mark and put the number of the section (when it is numbered) and its title in the left mark. The rightmark is set by `\subsectionmark` to contain the subsection titles.

Note the use of `##1` for the parameter of the `\sectionmark` command, which will be defined when `\ps@headings` is executed.

```
⟨*article⟩
    \def\sectionmark##1{%
      \markboth {\MakeUppercase{%
        \ifnum \c@secnumdepth >\z@
          \thesection\quad
        \fi
        ##1}}{}}%
    \def\subsectionmark##1{%
      \markright {%
        \ifnum \c@secnumdepth >\@ne
          \thesubsection\quad
        \fi
        ##1}}}
⟨/article⟩
```

In the report and book document classes we use the `\chaptermark` and `\sectionmark` macros to fill the running heads.

Note the use of `##1` for the parameter of the `\chaptermark` command, which will be defined when `\ps@headings` is executed.

```
⟨*report | book⟩
    \def\chaptermark##1{%
      \markboth {\MakeUppercase{%
        \ifnum \c@secnumdepth >\m@ne
⟨book⟩          \if@mainmatter
            \@chapapp\ \thechapter. \ %
⟨book⟩          \fi
        \fi
        ##1}}{}}%
    \def\sectionmark##1{%
      \markright {\MakeUppercase{%
        \ifnum \c@secnumdepth >\z@
          \thesection. \ %
        \fi
        ##1}}}}
⟨/report | book⟩
```

The definition of `\ps@headings` for one sided printing can be much simpler, because we treat even and odd pages the same. Therefore we don't need to define `\@even....`

```
\else
  \def\ps@headings{%
    \let\@oddfoot\@empty
    \def\@oddhead{{\slshape\rightmark}\hfil\thepage}%
    \let\@mkboth\markboth
```

We use `\markright` now instead of `\markboth` as we did for two sided printing.

```
⟨*article⟩
    \def\sectionmark##1{%
      \markright {\MakeUppercase{%
        \ifnum \c@secnumdepth >\m@ne
          \thesection\quad
        \fi
        ##1}}}}
⟨/article⟩
```

```
⟨*report | book⟩
    \def\chaptermark##1{%
      \markright {\MakeUppercase{%
        \ifnum \c@secnumdepth >\m@ne
⟨book⟩           \if@mainmatter
            \@chapapp\ \thechapter. \ %
⟨book⟩          \fi
        \fi
        ##1}}}}
⟨/report | book⟩
\fi
```

`\ps@myheadings`

The definition of the page style *myheadings* is fairly simple because the user determines the contents of the running head himself by using the `\markboth` and `\markright` commands.

```
\def\ps@myheadings{%
    \let\@oddfoot\@empty\let\@evenfoot\@empty
    \def\@evenhead{\thepage\hfil\slshape\leftmark}%
    \def\@oddhead{{\slshape\rightmark}\hfil\thepage}%
```

We have to make sure that the marking commands that are used by the chapter and section headings are disabled. We do this `\let`ting them to a macro that gobbles its argument(s).

```
    \let\@mkboth\@gobbletwo
⟨!article⟩      \let\chaptermark\@gobble
    \let\sectionmark\@gobble
⟨article⟩      \let\subsectionmark\@gobble
    }
```

7 Document Markup

7.1 The title

\title \author \date These three macros are provided by latex.dtx to provide information about the title, author(s) and date of the document. The information is stored away in internal control sequences. It is the task of the \maketitle command to use the information provided. The definitions of these macros are shown here for information.

```
% \newcommand*{\title}[1]{\gdef\@title{#1}}
% \newcommand*{\author}[1]{\gdef\@author{#1}}
% \newcommand*{\date}[1]{\gdef\@date{#1}}
```

The \date macro gets today's date by default.

```
% \date{\today}
```

\maketitle The definition of \maketitle depends on whether a separate title page is made. This is the default for the report and book document classes, but for the article class it is optional.

When we are making a title page, we locally redefine \footnotesize and footnoterule to change the appearance of the footnotes that are produced by the \thanks command; these changes affect all footnotes.

```
\if@titlepage
\newcommand\maketitle{\begin{titlepage}%
\let\footnotesize\small
\let\footnoterule\relax
\let \footnote \thanks
```

We center the entire title vertically; the centering is set off a little by adding a \vskip. (In compatibility mode the pagenumber is set to 0 by the titlepage environment to keep the behaviour of LaTeX 2.09 style files.)

```
\null\vfil
\vskip 60\p@
```

Then we set the title, in a \LARGE font; leave a little space and set the author(s) in a \large font. We do this inside a tabular environment to get them in a single column. Before the date we leave a little whitespace again.

```
\begin{center}%
  {\LARGE \@title \par}%
  \vskip 3em%
  {\large
   \lineskip .75em%
    \begin{tabular}[t]{c}%
      \@author
    \end{tabular}\par}%
    \vskip 1.5em%
  {\large \@date \par}%       % Set date in \large size.
\end{center}\par
```

Then we call `\@thanks` to print the information that goes into the footnote and finish the page.

```
  \@thanks
  \vfil\null
  \end{titlepage}%
```

We reset the *footnote* counter, disable `\thanks` and `\maketitle` and save some storage space by emptying the internal information macros.

```
  \setcounter{footnote}{0}%
  \global\let\thanks\relax
  \global\let\maketitle\relax
  \global\let\@thanks\@empty
  \global\let\@author\@empty
  \global\let\@date\@empty
  \global\let\@title\@empty
```

After the title is set the declaration commands `\title`, etc. can vanish. The definition of `\and` makes only sense within the argument of `\author` so this can go as well.

```
  \global\let\title\relax
  \global\let\author\relax
  \global\let\date\relax
  \global\let\and\relax
}
```

When the title is not on a page of its own, the layout of the title is a little different. We use symbols to mark the footnotes and we have to deal with two column documents.

Therefore we first start a new group to keep changes local. Then we redefine `\thefootnote` to use `\fnsymbol`; and change `\@makefnmark` so that footnotemarks have zero width (to make the centering of the author names look better).

```
\else
\newcommand\maketitle{\par
  \begingroup
    \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
    \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
    \long\def\@makefntext##1{\parindent 1em\noindent
            \hb@xt@1.8em{%
                \hss\@textsuperscript{\normalfont\@thefnmark}}##1}%
```

If this is a twocolumn document we start a new page in twocolumn mode, with the title set to the full width of the text. The actual printing of the title information is left to `\@maketitle`.

```
    \if@twocolumn
      \ifnum \col@number=\@ne
        \@maketitle
      \else
        \twocolumn[\@maketitle]%
      \fi
    \else
```

When this is not a twocolumn document we just start a new page, prevent floating objects from appearing on the top of this page and print the title information.

```
      \newpage
      \global\@topnum\z@   % Prevents figures from going at top of page.
      \@maketitle
    \fi
```

This page gets a *plain* layout. We call `\@thanks` to produce the footnotes.

```
    \thispagestyle{plain}\@thanks
```

Now we can close the group, reset the *footnote* counter, disable `\thanks`, `\maketitle` and `\@maketitle` and save some storage space by emptying the internal information macros.

```
  \endgroup
  \setcounter{footnote}{0}%
  \global\let\thanks\relax
  \global\let\maketitle\relax
  \global\let\@maketitle\relax
  \global\let\@thanks\@empty
  \global\let\@author\@empty
  \global\let\@date\@empty
  \global\let\@title\@empty
  \global\let\title\relax
  \global\let\author\relax
  \global\let\date\relax
  \global\let\and\relax
}
```

`\@maketitle` This macro takes care of formatting the title information when we have no separate title page.

We always start a new page, leave some white space and center the information. The title is set in a `\LARGE` font, the author names and the in a `\large` font.

```
\def\@maketitle{%
  \newpage
  \null
  \vskip 2em%
  \begin{center}%
  \let \footnote \thanks
    {\LARGE \@title \par}%
    \vskip 1.5em%
    {\large
      \lineskip .5em%
      \begin{tabular}[t]{c}%
        \@author
      \end{tabular}\par}%
    \vskip 1em%
    {\large \@date}%
  \end{center}%
  \par
  \vskip 1.5em}
```

```
\fi
```

7.2 Chapters and Sections

7.2.1 Building blocks

The definitions in this part of the class file make use of two internal macros, `\@startsection` and `\secdef`. To understand what is going on here, we describe their syntax.

The macro `\@startsection` has 6 required arguments, optionally followed by a $*$, an optional argument and a required argument:

`\@startsection`⟨*name*⟩⟨*level*⟩⟨*indent*⟩⟨*beforeskip*⟩⟨*afterskip*⟩⟨*style*⟩ optional *
`[`⟨*altheading*⟩`]`⟨*heading*⟩

It is a generic command to start a section, the arguments have the following meaning:

⟨***name***⟩ The name of the user level command, e.g., 'section'.

⟨***level***⟩ A number, denoting the depth of the section – e.g., chapter=1, section = 2, etc. A section number will be printed if and only if ⟨*level*⟩ <= the value of the *secnumdepth* counter.

⟨***indent***⟩ The indentation of the heading from the left margin

⟨***beforeskip***⟩ The absolute value of this argument gives the skip to leave above the heading. If it is negative, then the paragraph indent of the text following the heading is suppressed.

⟨***afterskip***⟩ If positive, this gives the skip to leave below the heading, else it gives the skip to leave to the right of a run-in heading.

⟨***style***⟩ Commands to set the style of the heading.

$*$ When this is missing the heading is numbered and the corresponding counter is incremented.

⟨***altheading***⟩ Gives an alternative heading to use in the table of contents and in the running heads. This should be present when the $*$ form is used.

⟨***heading***⟩ The heading of the new section.

A sectioning command is normally defined to `\@startsection` and its first six arguments.

The macro `\secdef` can be used when a sectioning command is defined without using `\@startsection`. It has two arguments:

`\secdef`⟨*unstarcmds*⟩⟨*starcmds*⟩

⟨***unstarcmds***⟩ Used for the normal form of the sectioning command.

⟨***starcmds***⟩ Used for the $*$-form of the sectioning command.

You can use `\secdef` as follows:

```
\def\chapter { ... \secdef \CMDA \CMDB }
\def\CMDA    [#1]#2{ ... }  % Command to define
                            % \chapter[...]{...}
\def\CMDB    #1{ ... }      % Command to define
                            % \chapter*{...}
```

7.2.2 Mark commands

`\chaptermark`
`\sectionmark`
`\subsectionmark`
`\subsubsectionmark`
`\paragraphmark`
`\subparagraphmark`

Default initializations of `\...mark` commands. These commands are used in the definition of the page styles (see section 6.4.2) Most of them are already defined by `latex.dtx`, so they are only shown here.

```
⟨!article⟩\newcommand*\chaptermark[1]{}
% \newcommand*\sectionmark[1]{}
% \newcommand*\subsectionmark[1]{}
% \newcommand*\subsubsectionmark[1]{}
% \newcommand*\paragraphmark[1]{}
% \newcommand*\subparagraphmark[1]{}
```

7.2.3 Define Counters

`\c@secnumdepth`

The value of the counter *secnumdepth* gives the depth of the highest-level sectioning command that is to produce section numbers.

```
⟨article⟩\setcounter{secnumdepth}{3}
⟨!article⟩\setcounter{secnumdepth}{2}
```

`\c@part`
`\c@chapter`
`\c@section`
`\c@subsection`
`\c@subsubsection`
`\c@paragraph`
`\c@subparagraph`

These counters are used for the section numbers. The macro
`\newcounter{`⟨*newctr*⟩`}[`⟨*oldctr*⟩`]`
defines⟨*newctr*⟩ to be a counter, which is reset to zero when counter ⟨*oldctr*⟩ is stepped. Counter ⟨*oldctr*⟩ must already be defined.

```
\newcounter {part}
⟨article⟩\newcounter {section}
⟨*report | book⟩
\newcounter {chapter}
\newcounter {section}[chapter]
⟨/report | book⟩
\newcounter {subsection}[section]
\newcounter {subsubsection}[subsection]
\newcounter {paragraph}[subsubsection]
\newcounter {subparagraph}[paragraph]
```

`\thepart`
`\thechapter`
`\thesection`
`\thesubsection`
`\thesubsubsection`
`\theparagraph`
`\thesubparagraph`

For any counter *CTR*, `\theCTR` is a macro that defines the printed version of counter *CTR*. It is defined in terms of the following macros:

`\arabic{`*COUNTER*`}` prints the value of *COUNTER* as an arabic numeral.

`\roman{`*COUNTER*`}` prints the value of *COUNTER* as a lowercase roman numeral.

`\Roman{`*COUNTER*`}` prints the value of *COUNTER* as an uppercase roman numberal.

`\alph{`*COUNTER*`}` prints the value of *COUNTER* as a lowercase letter: 1 = a, 2 = b, etc.

`\Alph{`*COUNTER*`}` prints the value of *COUNTER* as an uppercase letter: 1 = A, 2 = B, etc.

Actually to save space the internal counter repesentations and the commands operating on those are used.

```
\renewcommand \thepart {\@Roman\c@part}
⟨article⟩\renewcommand \thesection {\@arabic\c@section}
⟨*report | book⟩
\renewcommand \thechapter {\@arabic\c@chapter}
\renewcommand \thesection {\thechapter.\@arabic\c@section}
⟨/report | book⟩
\renewcommand\thesubsection    {\thesection.\@arabic\c@subsection}
\renewcommand\thesubsubsection{\thesubsection .\@arabic\c@subsubsection}
\renewcommand\theparagraph     {\thesubsubsection.\@arabic\c@paragraph}
\renewcommand\thesubparagraph  {\theparagraph.\@arabic\c@subparagraph}
```

`\@chapapp` `\@chapapp` is initially defined to be '`\chaptername`'. The `\appendix` command redefines it to be '`\appendixname`'.

```
⟨report | book⟩\newcommand\@chapapp{\chaptername}
```

7.2.4 Front Matter, Main Matter, and Back Matter

A book contains these three (logical) sections. The switch `\@mainmatter` is true iff we are processing Main Matter. When this switch is false, the `\chapter` command does not print chapter numbers.

Here we define the commands that start these sections.

`\frontmatter` This command starts Roman page numbering and turns off chapter numbering.

```
⟨*book⟩
\newcommand\frontmatter{%
  \if@openright
    \cleardoublepage
  \else
    \clearpage
  \fi
  \@mainmatterfalse
  \pagenumbering{roman}}
```

`\mainmatter` This command clears the page, starts arabic page numbering and turns on chapter numbering.

```
\newcommand\mainmatter{%
  \if@openright
    \cleardoublepage
  \else
    \clearpage
  \fi
  \@mainmattertrue
  \pagenumbering{arabic}}
```

\backmatter This clears the page, turns off chapter numbering and leaves page numbering unchanged.

```
\newcommand\backmatter{%
  \if@openright
    \cleardoublepage
  \else
    \clearpage
  \fi
  \@mainmatterfalse}
⟨/book⟩
```

7.2.5 Parts

\part The command to start a new part of our document.

In the article class the definition of `\part` is rather simple; we start a new paragraph, add a little white space, suppress the indentation of the first paragraph and make use of `\secdef`.

```
⟨*article⟩
\newcommand\part{\par
   \addvspace{4ex}%
   \@afterindentfalse
   \secdef\@part\@spart}
⟨/article⟩
```

For the report and book classes we things a bit different.

We start a new (righthand) page and use the *plain* pagestyle.

```
⟨*report | book⟩
\newcommand\part{%
  \if@openright
    \cleardoublepage
  \else
    \clearpage
  \fi
  \thispagestyle{plain}%
```

When we are making a two column document, this will be a one column page. We use `@tempswa` to remember to switch back to two columns.

```
  \if@twocolumn
    \onecolumn
    \@tempswatrue
  \else
    \@tempswafalse
  \fi
```

We need an empty box to prevent the fil glue from disappearing.

```
  \null\vfil
```

Here we use \secdef to indicate which commands to use to make the actual heading.

```
  \secdef\@part\@spart}
⟨/report | book⟩
```

\@part This macro does the actual formatting of the title of the part. Again the macro is differently defined for the article document class than for the document classes report and book.

When *secnumdepth* is larger than -1 for the document class article, we have a numbered part, otherwise it is unnumbered.

```
⟨*article⟩
\def\@part[#1]#2{%
    \ifnum \c@secnumdepth >\m@ne
      \refstepcounter{part}%
      \addcontentsline{toc}{part}{\thepart\hspace{1em}#1}%
    \else
      \addcontentsline{toc}{part}{#1}%
    \fi
```

We print the title flush left in the article class. Also we prevent breaking between lines and reset the font.

```
    {\parindent \z@ \raggedright
     \interlinepenalty \@M
     \normalfont
```

When this is a numbered part we have to print the number and the title. The \nobreak should prevent a page break here.

```
     \ifnum \c@secnumdepth >\m@ne
       \Large\bfseries \partname~\thepart
       \par\nobreak
     \fi
     \huge \bfseries #2%
```

Now we empty the mark registers, leave some white space and let \@afterheading take care of suppressing the indentation.

```
     \markboth{}{}\par}%
    \nobreak
    \vskip 3ex
    \@afterheading}
⟨/article⟩
```

When *secnumdepth* is larger than -2 for the document class report and book, we have a numbered part, otherwise it is unnumbered.

```
⟨*report | book⟩
\def\@part[#1]#2{%
    \ifnum \c@secnumdepth >-2\relax
      \refstepcounter{part}%
      \addcontentsline{toc}{part}{\thepart\hspace{1em}#1}%
    \else
      \addcontentsline{toc}{part}{#1}%
    \fi
```

We empty the mark registers and center the title on the page in the report and book document classes. Also we prevent breaking between lines and reset the font.

```
    \markboth{}{}%
    {\centering
     \interlinepenalty \@M
     \normalfont
```

When this is a numbered part we have to print the number.

```
     \ifnum \c@secnumdepth >-2\relax
       \huge\bfseries \partname~\thepart
       \par
```

We leave some space before we print the title and leave the finishing up to `\@endpart`.

```
       \vskip 20\p@
     \fi
     \Huge \bfseries #2\par}%
    \@endpart}
⟨/report | book⟩
```

`\@spart` This macro does the actual formatting of the title of the part when the star form of the user command was used. In this case we *never* print a number. Otherwise the formatting is the same.

The differences between the definition of this macro in the article document class and in the report and book document classes are similar as they were for `\@part`.

```
⟨*article⟩
\def\@spart#1{%
    {\parindent \z@ \raggedright
     \interlinepenalty \@M
     \normalfont
     \huge \bfseries #1\par}%
     \nobreak
     \vskip 3ex
     \@afterheading}
⟨/article⟩
⟨*report | book⟩
\def\@spart#1{%
    {\centering
     \interlinepenalty \@M
     \normalfont
     \Huge \bfseries #1\par}%
    \@endpart}
⟨/report | book⟩
```

`\@endpart` This macro finishes the part page, for both `\@part` and `\@spart`.

First we fill the current page.

```
⟨*report | book⟩
\def\@endpart{\vfil\newpage
```

Then, when we are in twosided mode, we produce a completely blank page.

```
             \if@twoside
               \null
               \thispagestyle{empty}%
               \newpage
             \fi
```

When this was a two column document we have to switch back to two column mode.

```
             \if@tempswa
               \twocolumn
             \fi}
⟨/report | book⟩
```

7.2.6 Chapters

\chapter A chapter should always start on a new page therefore we start by calling `\clearpage` and setting the pagestyle for this page to *plain*.

```
⟨*report | book⟩
\newcommand\chapter{\if@openright\cleardoublepage\else\clearpage\fi
                    \thispagestyle{plain}%
```

Then we prevent floats from appearing at the top of this page because it looks weird to see a floating object above a chapter title.

```
                    \global\@topnum\z@
```

Then we suppress the indentation of the first paragraph by setting the switch `\@afterindent` to `false`. We use `\secdef` to specify the macros to use for actually setting the chapter title.

```
                    \@afterindentfalse
                    \secdef\@chapter\@schapter}
```

\@chapter This macro is called when we have a numbered chapter. When *secnumdepth* is larger than -1 and, in the book class, `\@mainmatter` is true, we display the chapter number. We also inform the user that a new chapter is about to be typeset by writing a message to the terminal.

```
\def\@chapter[#1]#2{\ifnum \c@secnumdepth >\m@ne
⟨book⟩                       \if@mainmatter
                         \refstepcounter{chapter}%
                         \typeout{\@chapapp\space\thechapter.}%
                         \addcontentsline{toc}{chapter}%
                                   {\protect\numberline{\thechapter}#1}%
⟨*book⟩
                       \else
                         \addcontentsline{toc}{chapter}{#1}%
                       \fi
⟨/book⟩
                    \else
                      \addcontentsline{toc}{chapter}{#1}%
                    \fi
```

After having written an entry to the table of contents we store the (alternative) title of this chapter with \chaptermark and add some white space to the lists of figures and tables.

```
                    \chaptermark{#1}%
                    \addtocontents{lof}{\protect\addvspace{10\p@}}%
                    \addtocontents{lot}{\protect\addvspace{10\p@}}%
```

Then we call upon \@makechapterhead to format the actual chapter title. We have to do this in a special way when we are in twocolumn mode in order to have the chapter title use the entire \textwidth. In one column mode we call \@afterheading which takes care of suppressing the indentation.

```
                    \if@twocolumn
                      \@topnewpage[\@makechapterhead{#2}]%
                    \else
                      \@makechapterhead{#2}%
                      \@afterheading
                    \fi}
```

\@makechapterhead The macro above uses \@makechapterhead⟨*text*⟩ to format the heading of the chapter.

We begin by leaving some white space. The we open a group in which we have a paragraph indent of 0pt, and in which we have the text set ragged right. We also reset the font.

```
\def\@makechapterhead#1{%
  \vspace*{50\p@}%
  {\parindent \z@ \raggedright \normalfont
```

Then we check whether the number of the chapter has to be printed. If so we leave some whitespace between the chapternumber and its title.

```
    \ifnum \c@secnumdepth >\m@ne
⟨book⟩      \if@mainmatter
        \huge\bfseries \@chapapp\space \thechapter
        \par\nobreak
        \vskip 20\p@
⟨book⟩      \fi
    \fi
```

Now we set the title in a large bold font. We prevent a pagebreak from occuring in the middle of or after the title. Finally we leave some whitespace before the text begins.

```
    \interlinepenalty\@M
    \Huge \bfseries #1\par\nobreak
    \vskip 40\p@
  }}
```

\@schapter This macro is called when we have an unnumbered chapter. It is much simpler than \@chapter because it only needs to typeset the chapter title.

```
\def\@schapter#1{\if@twocolumn
                   \@topnewpage[\@makeschapterhead{#1}]%
```

```
                  \else
                    \@makeschapterhead{#1}%
                    \@afterheading
                  \fi}
```

\@makeschapterhead The macro above uses \@makeschapterhead⟨*text*⟩to format the heading of the chapter. It is similar to \@makechapterhead except that it never has to print a chapter number.

```
\def\@makeschapterhead#1{%
  \vspace*{50\p@}%
  {\parindent \z@ \raggedright
    \normalfont
    \interlinepenalty\@M
    \Huge \bfseries  #1\par\nobreak
    \vskip 40\p@
  }}
⟨/report | book⟩
```

7.2.7 Lower level headings

These commands all make use of \@startsection.

\section This gives a normal heading with white space above and below the heading, the title set in \Large\bfseries, and no indentation on the first paragraph.

```
\newcommand\section{\@startsection {section}{1}{\z@}%
                                   {-3.5ex \@plus -1ex \@minus -.2ex}%
                                   {2.3ex \@plus.2ex}%
                                   {\normalfont\Large\bfseries}}
```

\subsection This gives a normal heading with white space above and below the heading, the title set in \large\bfseries, and no indentation on the first paragraph.

```
\newcommand\subsection{\@startsection{subsection}{2}{\z@}%
                                     {-3.25ex\@plus -1ex \@minus -.2ex}%
                                     {1.5ex \@plus .2ex}%
                                     {\normalfont\large\bfseries}}
```

\subsubsection This gives a normal heading with white space above and below the heading, the title set in \normalsize\bfseries, and no indentation on the first paragraph.

```
\newcommand\subsubsection{\@startsection{subsubsection}{3}{\z@}%
                                     {-3.25ex\@plus -1ex \@minus -.2ex}%
                                     {1.5ex \@plus .2ex}%
                                     {\normalfont\normalsize\bfseries}}
```

\paragraph This gives a run-in heading with white space above and to the right of the heading, the title set in \normalsize\bfseries.

```
\newcommand\paragraph{\@startsection{paragraph}{4}{\z@}%
                                    {3.25ex \@plus1ex \@minus.2ex}%
                                    {-1em}%
                                    {\normalfont\normalsize\bfseries}}
```

`\subparagraph` This gives an indented run-in heading with white space above and to the right of the heading, the title set in `\normalsize\bfseries`.

```
\newcommand\subparagraph{\@startsection{subparagraph}{5}{\parindent}%
                                       {3.25ex \@plus1ex \@minus .2ex}%
                                       {-1em}%
                                      {\normalfont\normalsize\bfseries}}
```

7.3 Lists

7.3.1 General List Parameters

The following commands are used to set the default values for the list environment's parameters. See the LaTeX manual for an explanation of the meanings of the parameters. Defaults for the list environment are set as follows. First, `\rightmargin`, `\listparindent` and `\itemindent` are set to 0pt. Then, for a Kth level list, the command `\@listK` is called, where ‘K’ denotes ‘i’, ’‘i’, ... , ‘vi’. (I.e., `\@listiii` is called for a third-level list.) By convention, `\@listK` should set `\leftmargin` to `\leftmarginK`.

`\leftmargin` `\leftmargini` `\leftmarginii` `\leftmarginiii` `\leftmarginiv` `\leftmarginv` `\leftmarginvi`

When we are in two column mode some of the margins are set somewhat smaller.

```
\if@twocolumn
  \setlength\leftmargini  {2em}
\else
  \setlength\leftmargini  {2.5em}
\fi
```

Until the whole of the parameter setting in these files is rationalised, we need to set the value of `\leftmargin` at this outer level.

```
\leftmargin  \leftmargini
```

The following three are calculated so that they are larger than the sum of `\labelsep` and the width of the default labels (which are ‘(m)’, ‘vii.’ and ‘M.’).

```
\setlength\leftmarginii  {2.2em}
\setlength\leftmarginiii {1.87em}
\setlength\leftmarginiv  {1.7em}
\if@twocolumn
  \setlength\leftmarginv  {.5em}
  \setlength\leftmarginvi {.5em}
\else
  \setlength\leftmarginv  {1em}
  \setlength\leftmarginvi {1em}
\fi
```

`\labelsep` `\labelwidth`

`\labelsep` is the distance between the label and the text of an item; `\labelwidth` is the width of the label.

```
\setlength  \labelsep  {.5em}
\setlength  \labelwidth{\leftmargini}
\addtolength\labelwidth{-\labelsep}
```

`\partopsep` When the user leaves a blank line before the environment an extra vertical space of `\partopsep` is inserted, in addition to `\parskip` and `\topsep`.

```
⟨/article | report | book⟩
⟨10pt⟩\setlength\partopsep{2\p@ \@plus 1\p@ \@minus 1\p@}
⟨11pt⟩\setlength\partopsep{3\p@ \@plus 1\p@ \@minus 1\p@}
⟨12pt⟩\setlength\partopsep{3\p@ \@plus 2\p@ \@minus 2\p@}
```

`\@beginparpenalty` `\@endparpenalty` These penalties are inserted before and after a list or paragraph environment. They are set to a bonus value to encourage page breaking at these points.

`\@itempenalty` This penalty is inserted between list items.

```
⟨*article | report | book⟩
\@beginparpenalty -\@lowpenalty
\@endparpenalty   -\@lowpenalty
\@itempenalty     -\@lowpenalty
⟨/article | report | book⟩
```

`\@listi` `\@listI` `\@listi` defines the values of `\leftmargin`, `\parsep`, `\topsep`, `\itemsep`, etc. for the lists that appear on top-level. Its definition is modified by the font-size commands (eg within `\small` the list parameters get "smaller" values).

For this reason listI is defined to hold a saved copy of listi so that `\normalsize` can switch all parameters back.

```
⟨*10pt | 11pt | 12pt⟩
\def\@listi{\leftmargin\leftmargini
⟨*10pt⟩
            \parsep 4\p@ \@plus2\p@ \@minus\p@
            \topsep 8\p@ \@plus2\p@ \@minus4\p@
            \itemsep4\p@ \@plus2\p@ \@minus\p@}
⟨/10pt⟩
⟨*11pt⟩
            \parsep 4.5\p@ \@plus2\p@ \@minus\p@
            \topsep 9\p@   \@plus3\p@ \@minus5\p@
            \itemsep4.5\p@ \@plus2\p@ \@minus\p@}
⟨/11pt⟩
⟨*12pt⟩
            \parsep 5\p@  \@plus2.5\p@ \@minus\p@
            \topsep 10\p@ \@plus4\p@   \@minus6\p@
            \itemsep5\p@  \@plus2.5\p@ \@minus\p@}
⟨/12pt⟩
\let\@listI\@listi
```

We initialise the parameters although strictly speaking that is not necessary.

```
\@listi
```

`\@listii` `\@listiii` `\@listiv` `\@listv` `\@listvi` Here are the same macros for the higher level lists. Note that they don't have saved versions and are not modified by the font size commands. In other words this class assumes that nested lists only appear in `\normalsize`, i.e. the main document size.

```
\def\@listii {\leftmargin\leftmarginii
```

```
              \labelwidth\leftmarginii
              \advance\labelwidth-\labelsep
⟨*10pt⟩
              \topsep    4\p@ \@plus2\p@ \@minus\p@
              \parsep    2\p@ \@plus\p@  \@minus\p@
⟨/10pt⟩
⟨*11pt⟩
              \topsep    4.5\p@ \@plus2\p@ \@minus\p@
              \parsep    2\p@   \@plus\p@  \@minus\p@
⟨/11pt⟩
⟨*12pt⟩
              \topsep    5\p@   \@plus2.5\p@ \@minus\p@
              \parsep    2.5\p@ \@plus\p@    \@minus\p@
⟨/12pt⟩
              \itemsep   \parsep}
\def\@listiii{\leftmargin\leftmarginiii
              \labelwidth\leftmarginiii
              \advance\labelwidth-\labelsep
⟨10pt⟩             \topsep    2\p@ \@plus\p@\@minus\p@
⟨11pt⟩             \topsep    2\p@ \@plus\p@\@minus\p@
⟨12pt⟩             \topsep    2.5\p@\@plus\p@\@minus\p@
              \parsep    \z@
              \partopsep \p@ \@plus\z@ \@minus\p@
              \itemsep   \topsep}
\def\@listiv {\leftmargin\leftmarginiv
              \labelwidth\leftmarginiv
              \advance\labelwidth-\labelsep}
\def\@listv  {\leftmargin\leftmarginv
              \labelwidth\leftmarginv
              \advance\labelwidth-\labelsep}
\def\@listvi {\leftmargin\leftmarginvi
              \labelwidth\leftmarginvi
              \advance\labelwidth-\labelsep}
⟨/10pt | 11pt | 12pt⟩
```

7.3.2 Enumerate

The enumerate environment uses four counters: *enumi*, *enumii*, *enumiii* and *enumiv*, where *enumN* controls the numbering of the Nth level enumeration.

`\theenumi`
`\theenumii`
`\theenumiii`
`\theenumiv`

The counters are already defined in `latex.dtx`, but their representation is changed here.

```
⟨*article | report | book⟩
\renewcommand\theenumi{\@arabic\c@enumi}
\renewcommand\theenumii{\@alph\c@enumii}
\renewcommand\theenumiii{\@roman\c@enumiii}
\renewcommand\theenumiv{\@Alph\c@enumiv}
```

`\labelenumi`
`\labelenumii`
`\labelenumiii`
`\labelenumiv`

The label for each item is generated by the commands `\labelenumi` … `\labelenumiv`.

```
\newcommand\labelenumi{\theenumi.}
\newcommand\labelenumii{(\theenumii)}
\newcommand\labelenumiii{\theenumiii.}
\newcommand\labelenumiv{\theenumiv.}
```

`\p@enumii` `\p@enumiii` `\p@enumiv`

The expansion of `\p@enumN\theenumN` defines the output of a `\ref` command when referencing an item of the Nth level of an enumerated list.

```
\renewcommand\p@enumii{\theenumi}
\renewcommand\p@enumiii{\theenumi(\theenumii)}
\renewcommand\p@enumiv{\p@enumiii\theenumiii}
```

7.3.3 Itemize

`\labelitemi` `\labelitemii` `\labelitemiii` `\labelitemiv`

Itemization is controlled by four commands: `\labelitemi`, `\labelitemii`, `\labelitemiii`, and `\labelitemiv`, which define the labels of thevarious itemization levels: the symbols used are bullet, bold en-dash, centered asterisk and centred dot.

```
\newcommand\labelitemi{\textbullet}
\newcommand\labelitemii{\normalfont\bfseries \textendash}
\newcommand\labelitemiii{\textasteriskcentered}
\newcommand\labelitemiv{\textperiodcentered}
```

7.3.4 Description

`description`

The description environment is defined here – while the itemize and enumerate environments are defined in `latex.dtx`.

```
\newenvironment{description}
               {\list{}{\labelwidth\z@ \itemindent-\leftmargin
                        \let\makelabel\descriptionlabel}}
               {\endlist}
```

`\descriptionlabel`

To change the formatting of the label, you must redefine `\descriptionlabel`.

```
\newcommand*\descriptionlabel[1]{\hspace\labelsep
                                \normalfont\bfseries #1}
```

7.4 Defining new environments

7.4.1 Abstract

`abstract`

When we are producing a separate titlepage we also put the abstract on a page of its own. It will be centred vertically on the page.

Note that this environment is not defined for books.

```
% \changes{v1.3m}{1995/10/23}{Added setting of \cs{beginparpenalty} to
%    discourage page break before abstract heading.}
⟨*article | report⟩
\if@titlepage
  \newenvironment{abstract}{%
      \titlepage
      \null\vfil
      \@beginparpenalty\@lowpenalty
      \begin{center}%
        \bfseries \abstractname
        \@endparpenalty\@M
      \end{center}}%
     {\par\vfil\null\endtitlepage}
```

When we are not making a separate titlepage –the default for the article document class– we have to check if we are in twocolumn mode. In that case the abstract is as a `\section*`, otherwise the quotation environment is used to typeset the abstract.

```
\else
  \newenvironment{abstract}{%
      \if@twocolumn
        \section*{\abstractname}%
      \else
        \small
        \begin{center}%
          {\bfseries \abstractname\vspace{-.5em}\vspace{\z@}}%
        \end{center}%
        \quotation
      \fi}
      {\if@twocolumn\else\endquotation\fi}
\fi
⟨/article | report⟩
```

7.4.2 Verse

`verse` The verse environment is defined by making clever use of the list environment's parameters. The user types `\\` to end a line. This is implemented by `\let`'ing `\\` equal `\@centercr`.

```
\newenvironment{verse}
               {\let\\\@centercr
                \list{}{\itemsep      \z@
                        \itemindent   -1.5em%
                        \listparindent\itemindent
                        \rightmargin  \leftmargin
                        \advance\leftmargin 1.5em}%
                \item\relax}
               {\endlist}
```

7.4.3 Quotation

`quotation` The quotation environment is also defined by making clever use of the list environment's parameters. The lines in the environment are set smaller than `\textwidth`. The first line of a paragraph inside this environment is indented.

```
\newenvironment{quotation}
               {\list{}{\listparindent 1.5em%
```

```
                       \itemindent    \listparindent
                       \rightmargin   \leftmargin
                       \parsep        \z@ \@plus\p@}%
                \item\relax}
               {\endlist}
```

7.4.4 Quote

quote The quote environment is like the quotation environment except that paragraphs are not indented.

```
\newenvironment{quote}
               {\list{}{\rightmargin\leftmargin}%
                \item\relax}
               {\endlist}
```

7.4.5 Theorem

This document class does not define it's own theorem environments, the defaults, supplied by `latex.dtx` are available.

7.4.6 Titlepage

titlepage In the normal environments, the titlepage environment does nothing but start and end a page, and inhibit page numbers. In the report style, it also resets the page number to one, and then sets it back to one at the end. In compatibility mode, it sets the page number to zero. This is incorrect since it results in using the page parameters for a right-hand page but it is the way it was. In two-column style, it still makes a one-column page.

First we do give the definition for compatibility mode.

```
\if@compatibility
\newenvironment{titlepage}
    {%
⟨book⟩      \cleardoublepage
      \if@twocolumn
        \@restonecoltrue\onecolumn
      \else
        \@restonecolfalse\newpage
      \fi
      \thispagestyle{empty}%
      \setcounter{page}\z@
    }%
    {\if@restonecol\twocolumn \else \newpage \fi
    }
```

And here is the one for native $\LaTeX\,2_{\varepsilon}$.

```
\else
\newenvironment{titlepage}
    {%
⟨book⟩      \cleardoublepage
```

```
      \if@twocolumn
        \@restonecoltrue\onecolumn
      \else
        \@restonecolfalse\newpage
      \fi
      \thispagestyle{empty}%
      \setcounter{page}\@ne
    }%
    {\if@restonecol\twocolumn \else \newpage \fi
```

If we are not in two-side mode the first page after the title page should also get page number 1.

```
     \if@twoside\else
        \setcounter{page}\@ne
     \fi
    }
\fi
```

7.4.7 Appendix

\appendix The \appendix command is not really an environment, it is a macro that makes some changes in the way things are done.

In the article document class the \appendix command must do the following:

- reset the section and subsection counters to zero,
- redefine \thesection to produce alphabetic appendix numbers.

```
⟨*article⟩
\newcommand\appendix{\par
  \setcounter{section}{0}%
  \setcounter{subsection}{0}%
  \renewcommand\thesection{\@Alph\c@section}}
⟨/article⟩
```

In the report and book document classes the \appendix command must do the following:

- reset the chapter and section counters to zero,
- set \@chapapp to \appendixname (for messages),
- redefine the chapter counter to produce appendix numbers,
- possibly redefine the \chapter command if appendix titles and headings are to look different from chapter titles and headings.

```
⟨*report | book⟩
\newcommand\appendix{\par
  \setcounter{chapter}{0}%
  \setcounter{section}{0}%
  \renewcommand\@chapapp{\appendixname}%
  \renewcommand\thechapter{\@Alph\c@chapter}}
⟨/report | book⟩
```

7.5 Setting parameters for existing environments

7.5.1 Array and tabular

`\arraycolsep` The columns in an array environment are separated by 2`\arraycolsep`.

```
\setlength\arraycolsep{5\p@}
```

`\tabcolsep` The columns in an tabular environment are separated by 2`\tabcolsep`.

```
\setlength\tabcolsep{6\p@}
```

`\arrayrulewidth` The width of rules in the array and tabular environments is given by `\arrayrulewidth`.

```
\setlength\arrayrulewidth{.4\p@}
```

`\doublerulesep` The space between adjacent rules in the array and tabular environments is given by `\doublerulesep`.

```
\setlength\doublerulesep{2\p@}
```

7.5.2 Tabbing

`\tabbingsep` This controls the space that the `\'` command puts in. (See LaTeX manual for an explanation.)

```
\setlength\tabbingsep{\labelsep}
```

7.5.3 Minipage

`\@minipagerestore` The macro `\@minipagerestore` is called upon entry to a minipage environment to set up things that are to be handled differently inside a minipage environment. In the current styles, it does nothing.

`\@mpfootins` Minipages have their own footnotes; `\skip\@mpfootins` plays same rôle for footnotes in a minipage as `\skip\footins` does for ordinary footnotes.

```
\skip\@mpfootins = \skip\footins
```

7.5.4 Framed boxes

`\fboxsep` The space left by `\fbox` and `\framebox` between the box and the text in it.

`\fboxrule` The width of the rules in the box made by `\fbox` and `\framebox`.

```
\setlength\fboxsep{3\p@}
\setlength\fboxrule{.4\p@}
```

7.5.5 Equation and eqnarray

`\theequation` When within chapters, the equation counter will be reset at the beginning of a new chapter and the equation number will be prefixed by the chapter number.

This code must follow the `\chapter` definition or, more exactly, the definition of the chapter counter.

```
⟨article⟩\renewcommand \theequation {\@arabic\c@equation}
⟨*report | book⟩
\@addtoreset {equation}{chapter}
\renewcommand\theequation
  {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@equation}
⟨/report | book⟩
```

`\jot` `\jot` is the extra space added between lines of an eqnarray environment. The default value is used.

```
% \setlength\jot{3pt}
```

`\@eqnnum` The macro `\@eqnnum` defines how equation numbers are to appear in equations. Again the default is used.

```
% \def\@eqnnum{(\theequation)}
```

7.6 Floating objects

The file `latex.dtx` only defines a number of tools with which floating objects can be defined. This is done in the document class. It needs to define the following macros for each floating object of type `TYPE` (e.g., `TYPE` = figure).

`\fps@TYPE` The default placement specifier for floats of type `TYPE`.

`\ftype@TYPE` The type number for floats of type `TYPE`. Each `TYPE` has associated a unique positive `TYPE` number, which is a power of two. E.g., figures might have type number 1, tables type number 2, programs type number 4, etc.

`\ext@TYPE` The file extension indicating the file on which the contents list for float type `TYPE` is stored. For example, `\ext@figure` = 'lof'.

`\fnum@TYPE` A macro to generate the figure number for a caption. For example, `\fnum@TYPE` == 'Figure `\thefigure`'.

`\@makecaption`⟨***num***⟩⟨***text***⟩ A macro to make a caption, with ⟨*num*⟩ the value produced by `\fnum@...` and ⟨*text*⟩ the text of the caption. It can assume it's in a `\parbox` of the appropriate width. This will be used for *all* floating objects.

The actual environment that implements a floating object such as a figure is defined using the macros `\@float` and `\end@float`, which are defined in `latex.dtx`.

An environment that implements a single column floating object is started with `\@float{TYPE}[`⟨*placement*⟩`]` of type `TYPE` with ⟨*placement*⟩ as the placement specifier. The default value of ⟨*PLACEMENT*⟩ is defined by `\fps@TYPE`.

The environment is ended by \end@float. E.g., \figure == \@floatfigure, \endfigure == \end@float.

7.6.1 Figure

Here is the implementation of the figure environment.

\c@figure First we have to allocate a counter to number the figures.

In the report and book document classes figures within chapters are numbered per chapter.

```
⟨*article⟩
\newcounter{figure}
\renewcommand \thefigure {\@arabic\c@figure}
⟨/article⟩
⟨*report | book⟩
\newcounter{figure}[chapter]
\renewcommand \thefigure
     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@figure}
⟨/report | book⟩
```

\fps@figure Here are the parameters for the floating objects of type 'figure'.

\ftype@figure
\ext@figure
\num@figure

```
\def\fps@figure{tbp}
\def\ftype@figure{1}
\def\ext@figure{lof}
\def\fnum@figure{\figurename~\thefigure}
```

figure
figure* And the definition of the actual environment. The form with the * is used for double column figures.

```
\newenvironment{figure}
               {\@float{figure}}
               {\end@float}
\newenvironment{figure*}
               {\@dblfloat{figure}}
               {\end@dblfloat}
```

7.6.2 Table

Here is the implementation of the table environment. It is very much the same as the figure environment.

\c@table First we have to allocate a counter to number the tables.

In the report and book document classes tables within chapters are numbered per chapter.

```
⟨*article⟩
\newcounter{table}
\renewcommand\thetable{\@arabic\c@table}
⟨/article⟩
⟨*report | book⟩
```

```
\newcounter{table}[chapter]
\renewcommand \thetable
     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@table}
⟨/report | book⟩
```

\fps@table
\ftype@table
\ext@table
\num@table

Here are the parameters for the floating objects of type 'table'.

```
\def\fps@table{tbp}
\def\ftype@table{2}
\def\ext@table{lot}
\def\fnum@table{\tablename~\thetable}
```

table
table*

And the definition of the actual environment. The form with the * is used for double column tables.

```
\newenvironment{table}
               {\@float{table}}
               {\end@float}
\newenvironment{table*}
               {\@dblfloat{table}}
               {\end@dblfloat}
```

7.6.3 Captions

\@makecaption

The `\caption` command calls `\@makecaption` to format the caption of floating objects. It gets two arguments, ⟨*number*⟩, the number of the floating object and ⟨*text*⟩, the text of the caption. Usually ⟨*number*⟩ contains a string such as 'Figure 3.2'. The macro can assume it is called inside a `\parbox` of right width, with `\normalsize`.

\abovecaptionskip
\belowcaptionskip

These lengths contain the amount of white space to leave above and below the caption.

```
\newlength\abovecaptionskip
\newlength\belowcaptionskip
\setlength\abovecaptionskip{10\p@}
\setlength\belowcaptionskip{0\p@}
```

The definition of this macro is `\long` in order to allow more then one paragraph in a caption.

```
\long\def\@makecaption#1#2{%
  \vskip\abovecaptionskip
```

We want to see if the caption fits on one line on the page, therefore we first typeset it in a temporary box.

```
  \sbox\@tempboxa{#1: #2}%
```

We can the measure its width. It that is larger than the current `\hsize` we typeset the caption as an ordinary paragraph.

```
  \ifdim \wd\@tempboxa >\hsize
    #1: #2\par
```

If the caption fits, we center it. Because this uses an `\hbox` directly in vertical mode, it does not execute the `\everypar` tokens; the only thing that could be needed here is resetting the 'minipage flag' so we do this explicitly.

```
  \else
    \global \@minipagefalse
    \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
  \fi
  \vskip\belowcaptionskip}
```

7.7 Font changing

Here we supply the declarative font changing commands that were common in LaTeX version 2.09 and earlier. These commands work in text mode *and* in math mode. They are provided for compatibility, but one should start using the `\text...` and `\math...` commands instead. These commands are defined using `\DeclareTextFontCommand`, a command with three arguments: the user command to be defined; LaTeX commands to execute in text mode and LaTeX commands to execute in math mode.

`\rm` `\tt` `\sf` The commands to change the family. When in compatibility mode we select the 'default' font first, to get LaTeX2.09 behviour.

```
\DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
\DeclareOldFontCommand{\tt}{\normalfont\ttfamily}{\mathtt}
```

`\bf` The command to change to the bold series. One should use `\mdseries` to explicitly switch back to medium series.

```
\DeclareOldFontCommand{\bf}{\normalfont\bfseries}{\mathbf}
```

`\sl` `\it` `\sc` And the commands to change the shape of the font. The slanted and small caps shapes are not available by default as math alphabets, so those changes do nothing in math mode. However, we do warn the user that the selection will not have any effect.One should use `\upshape` to explicitly change back to the upright shape.

```
\DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
\DeclareOldFontCommand{\sl}{\normalfont\slshape}{\@nomath\sl}
\DeclareOldFontCommand{\sc}{\normalfont\scshape}{\@nomath\sc}
```

`\cal` `\mit` The commands `\cal` and `\mit` should only be used in math mode, outside math mode they have no effect. Currently the New Font Selection Scheme defines these commands to generate warning messages. Therefore we have to define them 'by hand'.

```
\DeclareRobustCommand*\cal{\@fontswitch\relax\mathcal}
\DeclareRobustCommand*\mit{\@fontswitch\relax\mathnormal}
```

8 Cross Referencing

8.1 Table of Contents, etc.

A `\section` command writes a `\contentsline{section}{`⟨*title*⟩`}{`⟨*page*⟩`}` command on the `.toc` file, where ⟨*title*⟩ contains the contents of the entry and ⟨*page*⟩ is the page number. If sections are being numbered, then ⟨*title*⟩ will be of the form `\numberline{`⟨*num*⟩`}{`⟨*heading*⟩`}` where ⟨*num*⟩ is the number produced by `\thesection`. Other sectioning commands work similarly.

A `\caption` command in a 'figure' environment writes
`\contentsline{figure}{\numberline{`⟨*num*⟩`}{ `⟨*caption*⟩`}}{`⟨*page*⟩`}`
on the .lof file, where ⟨*num*⟩ is the number produced by `\thefigure` and ⟨*caption*⟩ is the figure caption. It works similarly for a 'table' environment.

The command `\contentsline{`⟨*name*⟩`}` expands to `\l@`⟨*name*⟩. So, to specify the table of contents, we must define `\l@chapter`, `\l@section`, `\l@subsection`, ... ; to specify the list of figures, we must define `\l@figure`; and so on. Most of these can be defined with the `\@dottedtocline` command, which works as follows.
`\@dottedtocline{`⟨*level*⟩`}{`⟨*indent*⟩`}{`⟨*numwidth*⟩`}{`⟨*title*⟩`}{`⟨*page*⟩`}`

⟨***level***⟩ An entry is produced only if⟨ *level*⟩ <= value of the *tocdepth* counter. Note, `\chapter` is level 0, `\section` is level 1, etc.

⟨***indent***⟩ The indentation from the outer left margin of the start of the contents line.

⟨***numwidth***⟩ The width of a box in which the section number is to go, if ⟨*title*⟩ includes a `\numberline` command.

`\@pnumwidth`
`\@tocrmarg`
`\@dotsep`

This command uses the following three parameters, which are set with a `\newcommand` (so em's can be used to make them depend upon the font).

`\@pnumwidth` The width of a box in which the page number is put.

`\@tocrmarg` The right margin for multiple line entries. One wants `\@tocrmarg` ≥ `\@pnumwidth`

`\@dotsep` Separation between dots, in mu units. Should be defined as a number like 2 or 1.7

```
\newcommand\@pnumwidth{1.55em}
\newcommand\@tocrmarg{2.55em}
\newcommand\@dotsep{4.5}
⟨article⟩\setcounter{tocdepth}{3}
⟨!article⟩\setcounter{tocdepth}{2}
```

8.1.1 Table of Contents

`\tableofcontents`

This macro is used to request that LaTeX produces a table of contents. In the report and book document classes the tables of contents, figures etc. are always set in single-column style.

```
\newcommand\tableofcontents{%
⟨*report | book⟩
    \if@twocolumn
      \@restonecoltrue\onecolumn
    \else
      \@restonecolfalse
    \fi
```

The title is set using the `\chapter*` command, making sure that the running head –if one is required– contains the right information.

```
    \chapter*{\contentsname
⟨/report | book⟩
⟨article⟩    \section*{\contentsname
        \@mkboth{%
           \MakeUppercase\contentsname}{\MakeUppercase\contentsname}}%
```

The the actual table of contents is made by calling `\@starttoc{toc}`. After that we restore twocolumn mode if necessary.

```
    \@starttoc{toc}%
⟨!article⟩    \if@restonecol\twocolumn\fi
    }
```

`\l@part` Each sectioning command needs an additional macro to format its entry in the table of contents, as described above. The macro for the entry for parts is defined in a special way.

First we make sure that if a pagebreak should occur, it occurs *before* this entry. Also a little whitespace is added and a group begun to keep changes local.

```
\newcommand*\l@part[2]{%
  \ifnum \c@tocdepth >-2\relax
⟨article⟩    \addpenalty\@secpenalty
⟨!article⟩    \addpenalty{-\@highpenalty}%
    \addvspace{2.25em \@plus\p@}%
    \begingroup
```

We set `\parindent` to 0pt and use `\rightskip` to leave enough room for the pagenumbers. To prevent overfull box messages the `\parfillskip` is set to a negative value.

```
      \parindent \z@ \rightskip \@pnumwidth
      \parfillskip -\@pnumwidth
```

Now we can set the entry, in a large bold font. We make sure to leave vertical mode, set the part title and add the pagenumber, set flush right.

```
      {\leavevmode
       \large \bfseries #1\hfil \hb@xt@\@pnumwidth{\hss #2}}\par
```

Prevent a pagebreak immediately after this entry, but use `\everypar` to reset the `\if@nobreak` switch. Finally we close the group.

```
       \nobreak
⟨article⟩      \if@compatibility
         \global\@nobreaktrue
```

```
          \everypar{\global\@nobreakfalse\everypar{}}%
⟨article⟩        \fi
    \endgroup
  \fi}
```

\l@chapter This macro formats the entries in the table of contents for chapters. It is very similar to \l@part

First we make sure that if a pagebreak should occur, it occurs *before* this entry. Also a little whitespace is added and a group begun to keep changes local.

```
⟨*report | book⟩
\newcommand*\l@chapter[2]{%
  \ifnum \c@tocdepth >\m@ne
    \addpenalty{-\@highpenalty}%
    \vskip 1.0em \@plus\p@
```

The macro \numberline requires that the width of the box that holds the part number is stored in LaTeX's scratch register \@tempdima. Therefore we put it there. We begin a group, and change some of the paragraph parameters.

```
    \setlength\@tempdima{1.5em}%
    \begingroup
      \parindent \z@ \rightskip \@pnumwidth
      \parfillskip -\@pnumwidth
```

Then we leave vertical mode and switch to a bold font.

```
      \leavevmode \bfseries
```

Because we do not use \numberline here, we have do some fine tuning 'by hand', before we can set the entry. We discourage but not disallow a pagebreak immediately after a chapter entry.

```
      \advance\leftskip\@tempdima
      \hskip -\leftskip
      #1\nobreak\hfil \nobreak\hb@xt@\@pnumwidth{\hss #2}\par
      \penalty\@highpenalty
    \endgroup
  \fi}
⟨/report | book⟩
```

\l@section In the article document class the entry in the table of contents for sections looks much like the chapter entries for the report and book document classes.

First we make sure that if a pagebreak should occur, it occurs *before* this entry. Also a little whitespace is added and a group begun to keep changes local.

```
⟨*article⟩
\newcommand*\l@section[2]{%
  \ifnum \c@tocdepth >\z@
    \addpenalty\@secpenalty
    \addvspace{1.0em \@plus\p@}%
```

The macro \numberline requires that the width of the box that holds the part number is stored in LaTeX's scratch register \@tempdima. Therefore we put it there. We begin a group, and change some of the paragraph parameters.

```
    \setlength\@tempdima{1.5em}%
    \begingroup
      \parindent \z@ \rightskip \@pnumwidth
      \parfillskip -\@pnumwidth
```

Then we leave vertical mode and switch to a bold font.

```
      \leavevmode \bfseries
```

Because we do not use `\numberline` here, we have do some fine tuning 'by hand', before we can set the entry. We discourage but not disallow a pagebreak immediately after a chapter entry.

```
      \advance\leftskip\@tempdima
      \hskip -\leftskip
      #1\nobreak\hfil \nobreak\hb@xt@\@pnumwidth{\hss #2}\par
    \endgroup
  \fi}
⟨/article⟩
```

In the report and book document classes the definition for `\l@section` is much simpler.

```
⟨*report | book⟩
\newcommand*\l@section{\@dottedtocline{1}{1.5em}{2.3em}}
⟨/report | book⟩
```

`\l@subsection` `\l@subsubsection` `\l@paragraph` `\l@subparagraph`

All lower level entries are defined using the macro `\@dottedtocline` (see above).

```
⟨*article⟩
\newcommand*\l@subsection{\@dottedtocline{2}{1.5em}{2.3em}}
\newcommand*\l@subsubsection{\@dottedtocline{3}{3.8em}{3.2em}}
\newcommand*\l@paragraph{\@dottedtocline{4}{7.0em}{4.1em}}
\newcommand*\l@subparagraph{\@dottedtocline{5}{10em}{5em}}
⟨/article⟩
⟨*report | book⟩
\newcommand*\l@subsection{\@dottedtocline{2}{3.8em}{3.2em}}
\newcommand*\l@subsubsection{\@dottedtocline{3}{7.0em}{4.1em}}
\newcommand*\l@paragraph{\@dottedtocline{4}{10em}{5em}}
\newcommand*\l@subparagraph{\@dottedtocline{5}{12em}{6em}}
⟨/report | book⟩
```

8.1.2 List of figures

`\listoffigures`

This macro is used to request that LaTeX produces a list of figures. It is very similar to `\tableofcontents`.

```
\newcommand\listoffigures{%
⟨*report | book⟩
    \if@twocolumn
      \@restonecoltrue\onecolumn
    \else
      \@restonecolfalse
    \fi
    \chapter*{\listfigurename
⟨/report | book⟩
⟨article⟩     \section*{\listfigurename
      \@mkboth{\MakeUppercase\listfigurename}%
              {\MakeUppercase\listfigurename}}%
    \@starttoc{lof}%
⟨report | book⟩     \if@restonecol\twocolumn\fi
    }
```

\l@figure This macro produces an entry in the list of figures.

```
\newcommand*\l@figure{\@dottedtocline{1}{1.5em}{2.3em}}
```

8.1.3 List of tables

\listoftables This macro is used to request that LaTeX produces a list of tables. It is very similar to \tableofcontents.

```
\newcommand\listoftables{%
⟨*report | book⟩
    \if@twocolumn
      \@restonecoltrue\onecolumn
    \else
      \@restonecolfalse
    \fi
    \chapter*{\listtablename
⟨/report | book⟩
⟨article⟩     \section*{\listtablename
      \@mkboth{%
          \MakeUppercase\listtablename}{\MakeUppercase\listtablename}}%
    \@starttoc{lot}%
⟨report | book⟩     \if@restonecol\twocolumn\fi
    }
```

\l@table This macro produces an entry in the list of tables.

```
\let\l@table\l@figure
```

8.2 Bibliography

\bibindent The “open” bibliography format uses an indentation of \bibindent.

```
\newdimen\bibindent
\setlength\bibindent{1.5em}
```

thebibliography The ‘thebibliography’ environment executes the following commands:

\renewcommand{\newblock}{\hskip.11em \@plus.33em \@minus.07em}
— Defines the “closed” format, where the blocks (major units of information) of an entry run together.

\sloppy — Used because it’s rather hard to do line breaks in bibliographies,

\sfcode‘\.=1000\relax — Causes a ‘.’ (period) not to produce an end-of-sentence space.

The implementation of this environment is based on the generic list environment. It uses the *enumiv* counter internally to generate the labels of the list.

When an empty 'thebibliography' environment is found, a warning is issued.

```
\newenvironment{thebibliography}[1]
⟨*article⟩
      {\section*{\refname
         \@mkboth{\MakeUppercase\refname}{\MakeUppercase\refname}}%
⟨/article⟩
⟨*!article⟩
      {\chapter*{\bibname
         \@mkboth{\MakeUppercase\bibname}{\MakeUppercase\bibname}}%
⟨/!article⟩
      \list{\@biblabel{\@arabic\c@enumiv}}%
           {\settowidth\labelwidth{\@biblabel{#1}}%
            \leftmargin\labelwidth
            \advance\leftmargin\labelsep
            \@openbib@code
            \usecounter{enumiv}%
            \let\p@enumiv\@empty
            \renewcommand\theenumiv{\@arabic\c@enumiv}}%
      \sloppy
```

This is setting the normal (non-infinite) value of `\clubpenalty` for the whole of this environment, so we must reset its stored value also. (Why is there a `%` after the second 4000 below?)

```
      \clubpenalty4000
      \@clubpenalty \clubpenalty
      \widowpenalty4000%
      \sfcode`\.\@m}
     {\def\@noitemerr
       {\@latex@warning{Empty `thebibliography' environment}}%
      \endlist}
```

`\newblock` The default definition for `\newblock` is to produce a small space.

```
\newcommand\newblock{\hskip .11em\@plus.33em\@minus.07em}
```

`\@openbib@code` The default definition for `\@openbib@code` is to do nothing. It will be changed by the openbib option.

```
\let\@openbib@code\@empty
```

`\@biblabel` The label for a `\bibitem[...]` command is produced by this macro. The default from `latex.dtx` is used.

```
% \renewcommand*{\@biblabel}[1]{[#1]\hfill}
```

`\@cite` The output of the `\cite` command is produced by this macro. The default from `latex.dtx` is used.

```
% \renewcommand*{\@cite}[1]{[#1]}
```

8.3 The index

theindex

The environment 'theindex' can be used for indices. It makes an index with two columns, with each entry a separate paragraph. At the user level the commands \item, \subitem and \subsubitem are used to produce index entries of various levels. When a new letter of the alphabet is encountered an amount of \indexspace white space can be added.

```
\newenvironment{theindex}
               {\if@twocolumn
                  \@restonecolfalse
                \else
                  \@restonecoltrue
                \fi
                \columnseprule \z@
                \columnsep 35\p@
⟨article⟩                 \twocolumn[\section*{\indexname}]%
⟨!article⟩                 \twocolumn[\@makeschapterhead{\indexname}]%
                \@mkboth{\MakeUppercase\indexname}%
                        {\MakeUppercase\indexname}%
                \thispagestyle{plain}\parindent\z@
                \parskip\z@ \@plus .3\p@\relax
                \let\item\@idxitem}
```

When the document continues after the index and it was a one column document we have to switch back to one column after the index.

```
               {\if@restonecol\onecolumn\else\clearpage\fi}
```

\@idxitem
\subitem
\subsubitem

These macros are used to format the entries in the index.

```
\newcommand\@idxitem{\par\hangindent 40\p@}
\newcommand\subitem{\@idxitem \hspace*{20\p@}}
\newcommand\subsubitem{\@idxitem \hspace*{30\p@}}
```

\indexspace

The amount of white space that is inserted between 'letter blocks' in the index.

```
\newcommand\indexspace{\par \vskip 10\p@ \@plus5\p@ \@minus3\p@\relax}
```

8.4 Footnotes

\footnoterule

Usually, footnotes are separated from the main body of the text by a small rule. This rule is drawn by the macro \footnoterule. We have to make sure that the rule takes no vertical space (see plain.tex) so we compensate for the natural heigth of the rule of 0.4pt by adding the right amount of vertical skip.

To prevent the rule from colliding with the footnote we first add a little negative vertical skip, then we put the rule and make sure we end up at the same point where we begun this operation.

```
\renewcommand\footnoterule{%
  \kern-3\p@
  \hrule\@width.4\columnwidth
  \kern2.6\p@}
```

\c@footnote Footnotes are numbered within chapters in the report and book document styles.

```
⟨!article⟩\@addtoreset{footnote}{chapter}
```

\@makefntext The footnote mechanism of LaTeX calls the macro `\@makefntext` to produce the actual footnote. The macro gets the text of the footnote as its argument and should use `\@thefnmark` as the mark of the footnote. The macro `\@makefntext`is called when effectively inside a `\parbox` of width `\columnwidth` (i.e., with `\hsize` = `\columnwidth`).

An example of what can be achieved is given by the following piece of TeX code.

```
\newcommand\@makefntext[1]{%
   \@setpar{\@@par
            \@tempdima = \hsize
            \advance\@tempdima-10pt
            \parshape \@ne 10pt \@tempdima}%
   \par
   \parindent 1em\noindent
   \hbox to \z@{\hss\@makefnmark}#1}
```

The effect of this definition is that all lines of the footnote are indented by 10pt, while the first line of a new paragraph is indented by 1em. To change these dimensions, just substitute the desired value for '10pt' (in both places) or '1em'. The mark is flushright against the footnote.

In these document classes we use a simpler macro, in which the footnote text is set like an ordinary text paragraph, with no indentation except on the first line of a paragraph, and the first line of the footnote. Thus, all the macro must do is set `\parindent` to the appropriate value for succeeding paragraphs and put the proper indentation before the mark.

```
\newcommand\@makefntext[1]{%
    \parindent 1em%
    \noindent
    \hb@xt@1.8em{\hss\@makefnmark}#1}
```

\@makefnmark The footnote markers that are printed in the text to point to the footnotes should be produced by the macro `\@makefnmark`. We use the default definition for it.

```
%\renewcommand\@makefnmark{\hbox{\@textsuperscript
%                               {\normalfont\@thefnmark}}}
```

9 Initialization

9.1 Words

This document class is for documents prepared in the English language. To prepare a version for another language, various English words must be replaced. All the English words that require replacement are defined below in command names.

These commands may be redefined in any class or package that is customising LaTeX for use with non-English languages.

\contentsname
\listfigurename
\listtablename

```
\newcommand\contentsname{Contents}
\newcommand\listfigurename{List of Figures}
\newcommand\listtablename{List of Tables}
```

\refname
\bibname
\indexname

```
⟨article⟩ \newcommand\refname{References}
⟨report | book⟩ \newcommand\bibname{Bibliography}
\newcommand\indexname{Index}
```

\figurename
\tablename

```
\newcommand\figurename{Figure}
\newcommand\tablename{Table}
```

\partname
\chaptername
\appendixname
\abstractname

```
\newcommand\partname{Part}
⟨report | book⟩ \newcommand\chaptername{Chapter}
\newcommand\appendixname{Appendix}
⟨!book⟩ \newcommand\abstractname{Abstract}
```

9.2 Date

\today This macro uses the TeX primitives `\month`, `\day` and `\year` to provide the date of the LaTeX-run.

At `\begin{document}` this definition will be optimised so that the names of all the 'wrong' months are not stored. This optimisation is not done here as that would 'freeze' `\today` in any special purpose format made by loading the class file into the format file.

```
\def\today{\ifcase\month\or
  January\or February\or March\or April\or May\or June\or
  July\or August\or September\or October\or November\or December\fi
  \space\number\day, \number\year}
```

9.3 Two column mode

\columnsep This gives the distance between two columns in two column mode.

```
\setlength\columnsep{10\p@}
```

\columnseprule This gives the width of the rule between two columns in two column mode. We have no visible rule.

```
\setlength\columnseprule{0\p@}
```

9.4 The page style

We have *plain* pages in the document classes article and report unless the user specified otherwise. In the 'book' document class we use the page style *headings* by default. We use arabic pagenumbers.

```
1325 ⟨!book⟩\pagestyle{plain}
1326 ⟨book⟩\pagestyle{headings}
1327 \pagenumbering{arabic}
```

9.5 Single or double sided printing

When the twoside option wasn't specified, we don't try to make each page as long as all the others.

```
1328 \if@twoside
1329 \else
1330   \raggedbottom
1331 \fi
```

When the twocolumn option was specified we call `\twocolumn` to activate this mode. We try to make each column as long as the others, but call `sloppy` to make our life easier.

```
1332 \if@twocolumn
1333   \twocolumn
1334   \sloppy
1335   \flushbottom
```

Normally we call `\onecolumn` to initiate typesetting in one column.

```
1336 \else
1337   \onecolumn
1338 \fi
1339 ⟨/article | report | book⟩
```

Index

Numbers written in italic refer to the page where the corresponding entry is described, the ones underlined to the code line of the definition, the rest to the code lines where the entry is used.

Change History

v1.0d

- General: Added use of \NeedsTeXFormat 1
  - Made eqaution and eqnarray environments in the fleqn option up to date with latex.dtx 1
  - remove \@in, made option makeindex a synonym for option makeidx 1
  - removed \@minus, \@plus, \@settopoint, \@setfontsize; they are now in the kernel 1
  - Replaced \bf with \bfseries; \rm with \rmfamily 1

v1.0e

- \@normalsize: \normalsize doesn't exist, so use \newcommand 8

v1.0f

- General: Made all lines shorter than 72 characters 1
- \bf: Macro added 47
- \it: Macro added 47
- \rm: Macro added 47
- \sc: Macro added 47
- \sf: Macro added 47
- \sl: Macro added 47
- \tt: Macro added 47

v1.0g

- General: Distinguished between compatibility and 'normal' mode for the font changing commands. 47
  - Made change in eqnarray for the fleqn option, as suggested by Rainer. 1
  - Removed typo, A4 is not 279 mm high 4
- \cal: Macro added 47
- theindex: Moved the setting of \@restonecoltrue 54
- titlepage: Moved the setting of \@restonecolfalse 41
- \listoffigures: Moved the setting of \@restonecolfalse 51
- \listoftables: Moved the setting of \@restonecolfalse 52
- \maketitle: Removed the setting of the page number, when not in compatibility mode 24
- \mit: Macro added 47
- \tableofcontents: Moved the setting of \@restonecolfalse 48

v1.0h

- \@normalsize: \normalsize is now defined in the kernel, so use \renewcommand. ASAJ. 8
- General: Corrected some typos. ASAJ. 6
  - Made the definitions of the font- and size-changing commands use \renew rather than \new. Defined the float parameters with \renewcommand rather than \newcommand. Corrected some typos in the fleqn option. Replaced two occurrences of -\@secpenalty by \@secpenalty. ASAJ. 1
  - These are now defined in the kernel, so use \@renewfontswitch. Compatibility mode defines \@renewfontswitch to do nothing, so we don't need to check for compatibility mode any more. 47
- \bottomfraction: Replaced \newcommand with \renewcommand. ASAJ. 18
- \dblfloatpagefraction: Replaced \newcommand with \renewcommand. ASAJ. 19
- \dbltopfraction: Replaced \newcommand with \renewcommand. ASAJ. 18
- \floatpagefraction: Replaced \newcommand with \renewcommand. ASAJ. 18
- \footnotesize: \footnotesize is now defined in the kernel, so use \renewcommand. ASAJ. 9

necessary; made a slight implementation modification to the openright and openany options. 1

titlepage: Moved \cleardoublepage inside definition of titlepage environment 41

v1.2e

General: \@renewfontswitch has become \DeclareOldFontCommand 47

\footnotesize: use \newcommand again 9

\Huge: use \newcommand again . . 10

\small: \small is no longer defined in the kernel; use \newcommand . 8

v1.2g

\sc: Added warning if used in math mode 47

\sl: Added warning if used in math mode 47

v1.2i

General: Use LaTeX instead of LaTeX2e in messages 1

\l@chapter: Don't print a toc line when the tocdepth counter is less then 0 50

\l@part: Don't print a toc line when the tocdepth counter is less then -1 49

\l@section: Don't print a toc line when the tocdepth counter is less then 1 50

v1.2j

General: Removed the use of \fileversion c.s. 1

v1.2k

\descriptionlabel: Inserted \normalfont 39

\labelitemii: Inserted \normalfont 39

\maketitle: Added check on number of columns in use locally . 25

\maxdepth: Added setting of \maxdepth and \@maxdepth . . 12

v1.2l

General: changed some \changes entries 1

v1.2m

General: Forgot a few entries 1

v1.2o

General: Changed file information 1

v1.2p

General: Moved identification and driver to the front of the file . . 1

v1.2q

\@makecaption: Use \sbox\@tempboxa instead of \setbox\@tempboxa\hbox to make this colour safe 46

v1.2s

\maketitle: Reset \@makefntext 25

v1.2t

General: Refrased a few sentences to prevent overfull hboxes 1

v1.2u

\topmargin: Moved rounding of \topmargin to native mode . . 17

v1.2v

\@dotsep: Changed documentation from $>$ or $=$ to $\geq$ 48

\@makechapterhead: Added \interlinepenalty\@M to prevent a pagebreak in the middle of a title 34

Added a \nobreak to prevent a pagebreak between the chapternumber and the chaptertitle . 34

\@makeschapterhead: Added \interlinepenalty\@M to prevent a pagebreak in the middle of a title 35

General: Made the oneside option work for the book class 1

removed typo 57

\if@mainmatter: Moved the allocation of \if@mainmatter here . . 4

v1.2w

General: Use \newcommand* for commands with arguments . . . 1

\mit: Now define \cal and \mit using \DeclareRobustCommand* 47

v1.2x

\@makecaption: Due to a change in the way floats are handled we need to set the \if@minipage switch to false 47

v1.2y

\descriptionlabel: made command short 39