Document Class `refman` for LaTeX version 2e[*]

Copyright (C) 1988 by Hubert Partl
Copyright (C) 1994-2006 by Axel Kielhorn

2006/11/13

Contents

1 The DOCSTRIP modules

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

refart	produce the documentclass refart
refrep	produce the documentclass refrep
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, its value is one of 0, 1 or 2.

```
⟨*refart | refrep⟩
\newcommand\@ptsize{}
```

`\if@restonecol` Only the index is printed in two-column layout.

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

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

```
\newif\if@titlepage
⟨+refart⟩\@titlepagefalse
⟨+refrep⟩\@titlepagetrue
```

`\if@openright` A switch to indicate if chapters must start on a right-hand page. The default for the `refrep` class is no. There are no chapters in the `refart` class.

```
⟨+refrep⟩\newif\if@openright
```

[*]This file has version number v2.0e, last revised 2006/11/13.

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. Since `repbook` does not exist you may change the papersizes in your document if needed.

```
\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}}
```

A new lenght `\papermarginwidth` is allocated and set to 1 in. The option smallborder reduces the margin around the document to 0.25 in. This is suitable for documents that are ment to be read on the screen, since it wastes less screen area. It is probably no good idea to use this option for a printed document.

If you want to change this margin use `\setlength\papermarginwidth` in your document and issue a `\settowidth{0.7}` command to recalculate the page layouot.

```
\newlength\papermarginwidth
\setlength\papermarginwidth  {1in}
\DeclareOption{smallborder}
   {\setlength\papermarginwidth  {0.25in}}
```

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}}
```

The option square assigns the values of `\paperwidth` to `\paperheight`, which will result in a square layout. If you use landscape first you will get a square layout which uses the height of your original paper.

```
\DeclareOption{square}
   {\setlength\paperheight {\paperwidth}}
```

3.2 Telling the DVI-driver what we know

We know the actual paper size, let's tell the dvi-driver about it. You have to specify the dvi-driver you are using. These three options are compatible to the options of the KOMA-script class, thanks Markus.

`dvips`

```
\DeclareOption{dvips}{\AtBeginDocument{\AtBeginDvi{%
      \special{papersize=\the\paperwidth,\the\paperheight}}}}
```

`pdftex`

```
\DeclareOption{pdftex}{\AtBeginDocument{%
    \pdfpagewidth=\paperwidth \pdfpageheight=\paperheight}}
```

Since we need to know whether we are generating DVI or PDF we can make this information available to the user as well.

`\ifpdfoutput`

```
\newcommand{\ref@ifpdfoutput}[2]{%
  \begingroup\@ifundefined{pdfoutput}{\endgroup #2}{\endgroup%
    \ifnum\pdfoutput>0\relax #1\else #2\fi}}%
\@ifundefined{ifpdfoutput}{%
  \let\ifpdfoutput\ref@ifpdfoutput}{%
⟨*refart⟩
\PackageInfo{refart}
⟨/refart⟩
⟨*refrep⟩
\PackageInfo{refrep}
⟨/refrep⟩
    {\string\ifpdfoutput\space already defined.\MessageBreak
    If \string\ifpdfoutput\space does not behave like\MessageBreak
    is is described at the Refman manual, try find out\MessageBreak
    at which package \string\ifpdfoutput\space was defined.}}
```

The option pagesize will export the correct definition for the dvi-driver used. (Unless magic fails, then you need to specify the driver manually.)

`pagesize`

```
\DeclareOption{pagesize}{\AtBeginDocument{%
    \ref@ifpdfoutput
    {\pdfpagewidth=\paperwidth\pdfpageheight=\paperheight}
    {\AtBeginDvi{\special{papersize=\the\paperwidth,\the\paperheight}}}}}
```

3.3 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 8 assuming that a class won't define both 8pt and 18pt options.

```
\DeclareOption{10pt}{\renewcommand\@ptsize{0}}
\DeclareOption{11pt}{\renewcommand\@ptsize{1}}
\DeclareOption{12pt}{\renewcommand\@ptsize{2}}
```

3.4 Two-side or one-side printing

For two-sided printing we use the switch `\if@twoside`. We set `\if@mparswitch` which does nothing now but is kept for compatibility reasons.

```
\DeclareOption{oneside}{\@twosidefalse \@mparswitchfalse}
\DeclareOption{twoside}{\@twosidetrue  \@mparswitchtrue}
```

3.5 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}}
\DeclareOption{final}{\setlength\overfullrule{0pt}}
```

3.6 Titlepage option

The `refart` usually has no separate titlepage, but the user can request one.

```
\DeclareOption{titlepage}  {\@titlepagetrue}
\DeclareOption{notitlepage}{\@titlepagefalse}
```

3.7 openright option

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

```
⟨+refrep⟩\DeclareOption{openright}{\@openrighttrue}
⟨+refrep⟩\DeclareOption{openany}  {\@openrightfalse}
```

3.8 Twocolumn printing

Two-column is used in the index. There is no user command or option to request two-column printing. Therefore twocolumn will lead to an error message.

```
\DeclareOption{onecolumn}{\@twocolumnfalse}
\DeclareOption{twocolumn}{%
⟨+refart⟩      \ClassError{Refart}
⟨+refrep⟩      \ClassError{Refrep}
    {There is no twocolumn layout in this class!}
    {Can you imagine how twocolumn layout will look\MessageBreak
     in this class? That's why!}
    \@twocolumnfalse}
```

3.9 Equation numbering on the left

The option leqno can be used to get the equation numbers on the left side of the equation.

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

3.10 Flush left displays

The option fleqn redefines the displayed math environmens in such a way that they come out flush left, with an indentation of `\mathindent` from the prevailing left margin.

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

3.11 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.

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

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

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

3.12 User flags

There are some flags the user may change to control the behaviour of some commands:

`\ifdescriptioncolon` This switch controls whether there is a colon in the description item or not. The default is to include a colon.

```
\newif\ifdescriptioncolon \descriptioncolontrue
```

`\ifdescriptionleft` This switch controls whether the description items are set left bound or right bound. The default is right bound.

```
\newif\ifdescriptionleft  \descriptionleftfalse
```

`\ifmaxipagerule` This switch controls whether there is a rule at the beginning and end of a maxipage. This flag may later be used to select rules at other places (like part or chapter) as well.

```
\newif\ifmaxipagerule     \maxipageruletrue
```

4 Executing Options

Here we execute the default options to initialize certain variables.

```
⟨*refart⟩
\ExecuteOptions{letterpaper,10pt,oneside,onecolumn,final}
⟨/refart⟩
⟨*refrep⟩
\ExecuteOptions{letterpaper,10pt,oneside,onecolumn,final,openany}
⟨/refrep⟩
```

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. We are using the `sizexx.clo` Files from classes.dtx now and do the page layout caculation inside the class-file.

```
⟨*refart | refrep⟩
\input{size1\@ptsize.clo}
⟨/refart | refrep⟩
```

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.

See `classes.dtx` for documentaion on `sizexx.clo`

6.2 Paragraphing

`\lineskip` `\normallineskip` These parameters control TeX's behaviour when two lines tend to come too close together.

```
⟨*refart | refrep⟩
\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.

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

`\parskip` `\parindent` `\parskip` gives extra vertical space between paragraphs and `\parindent` is the width of the paragraph indentation. The value of `\parindent` is set to 0.

```
\setlength\parskip   {0.5\baselineskip \@plus 2\p@}
\setlength\parindent {\z@}
```

`\@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.

```
\@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`

Discourrage (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
```

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. `\topskip` is the `\baselineskip` for the first line on a page. Only the definition of `\headsep` differs from `sizexx` and has to be changed.

```
\setlength\headsep   {\baselineskip}
```

6.3.2 The dimension of text

`\fullwidth` `\textwidth` `\leftmarginwidth`

There is no need to supply a compatibility mode since the independent `refman.sty` was never released to the public.

We will set the dimensions differently, taking into account the paper size for instance.

First, we calculate the maximum text width, which will fit on the selected paper and store it in `\@tempdima`.

```
\newdimen\leftmarginwidth
\newdimen\fullwidth
```

`\emptyfoottopmargin` `\emptyheadtopmargin`

Your document uses either *footings* or *headings*. Depending on this, the whole page is shifted up or down by one line.

```
\newdimen\emptyfoottopmargin
\newdimen\emptyheadtopmargin
```

`\settextfraction`

You can specify how much of the `\fullwidth` will be used for the text by using the `\settextfraction` command. The argument should be between 0 and 1. The remaining width is used for the left margin. This command recalculates the page layout. You have to call `\settextfraction` with the default value of 0.7 to adjust the layout when you want to change the `\papermarginwidth`.

```
\newcommand\settextfraction[1]%
{
  \setlength\fullwidth{\paperwidth}
  \addtolength\fullwidth{-2\papermarginwidth}
  \@settopoint\fullwidth
```

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

```
  \if@twocolumn
      \setlength\textwidth{\fullwidth}
  \else
      \setlength\textwidth{#1\fullwidth}
  \fi
```

Here we modify the width of the text a little to be a whole number of points and calculate the remaining margin.

```
  \@settopoint\textwidth
  \setlength\leftmarginwidth{\fullwidth}
  \addtolength\leftmarginwidth{-\textwidth}
```

6.3.3 Horizontal margins

`\oddsidemargin` `\evensidemargin` `\marginparwidth`

The values for `\oddsidemargin` and `\marginparwidth` will be set independing on the status of the `\if@twoside`. (We have the same layout on odd and even pages.)

The `\oddsidemargin` is simply the `\papermarginwidth` plus the `\leftmarginwidth` calculated earlier, adjusted by 1 in for the printer driver offset and rounded to full points.

```
  \setlength\oddsidemargin    {-1in}
  \addtolength\oddsidemargin  {\papermarginwidth}
  \addtolength\oddsidemargin  {\leftmarginwidth}
  \@settopoint\oddsidemargin
```

Then `\evensidemargin` and `\marginparwidth` are set to `\oddsidemargin`. `\marginparwidth` will be modified later.

```
  \setlength\evensidemargin  {\oddsidemargin}
  \setlength\marginparwidth  {\leftmarginwidth
  }
```

`\marginparsep` `\marginparpush`

The horizontal space between the main text and marginal notes is determined by `\marginparsep` (defined in `sizexx`), the minimum vertical separation between two marginal notes is controlled by `\marginparpush` which is set to 0 because we will have lots of margin notes. The width of the `marginpar` is reduced by `marginparsep` to produce flush left pages.

```
  \addtolength\marginparwidth {-\marginparsep}
  \setlength\marginparpush    {0\p@}
}
```

Now we call `\settextfraction` with the default value of 0.7

```
\@onlypreamble\settextfraction
\settextfraction {0.7}
```

\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).

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.

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

We leave at least a \papermarginwidth margin on the top and the bottom of the page.

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

The running headers and footers extend partly into the top and bottom margins.

```
\addtolength\@tempdima{-.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}
```

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

```
\advance\textheight by \topskip
```

6.3.4 Vertical margins

\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 and rounded to full points.

```
\setlength\topmargin  {\paperheight}
\addtolength\topmargin{-\headheight}
\addtolength\topmargin{-\headsep}
\addtolength\topmargin{-\textheight}
\addtolength\topmargin{-\footskip}     % this might be wrong!
\addtolength\topmargin{-.5\topmargin}
\addtolength\topmargin{-1in}
\@settopoint\topmargin
```

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

The contents of the page is shifted up or down by one \baselineskip depending on the pagestyle. Do not combine headings and footings in one document!

```
\setlength\emptyfoottopmargin  {\topmargin}
\addtolength\emptyfoottopmargin{\baselineskip}
\setlength\emptyheadtopmargin  {\topmargin}
\addtolength\emptyheadtopmargin{-\baselineskip}
```

6.3.5 Float placement parameters

All float parameters are given default values in the $\LaTeX\,2_\varepsilon$ kernel. For this reason counters only need to be set with `\setcounter` and other parameters are set using `\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.

```
\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}
⟨/refart | refrep⟩
```

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.

```
179 % %%%\mark{{}{}}   % Initializes TeX's marks   <--- can vanish
```

6.4.2 Defining the page styles

The pagestyles *empty* is defined in `latex.dtx`.

`\ps@plain` We have to redefine *plain* to support twoside layout.

```
180 ⟨*refart | refrep⟩
```

```
\if@twoside
  \def\ps@plain{%
    \let\@mkboth\@gobbletwo
    \let\@oddhead\@empty
    \let\@evenhead\@empty
    \def\@oddfoot{\normalfont\hfil\thepage}
    \def\@evenfoot{\normalfont\thepage\hfil}}
\else
  \def\ps@plain{%
    \let\@mkboth\@gobbletwo
    \let\@oddhead\@empty
    \let\@evenhead\@empty
    \def\@oddfoot{\normalfont\hfil\thepage}
    \let\@evenfoot\@oddfoot}
\fi
%  \end{macrocode}
% \end{macro}
%
% \begin{macro}{\ps@headings}
%    The definition of the page style \pstyle{headings} has to be
%    different for two sided printing than it is for one sided
%    printing.
%
%    \begin{macrocode}
\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{\hss\vbox to \z@{\vss\hsize=\fullwidth
      \hb@xt@\fullwidth{\thepage\hfil\slshape\leftmark}
      \vskip 3\p@ \hrule}}%
      \def\@oddhead{\hss\vbox to \z@{\vss\hsize=\fullwidth
      \hb@xt@\fullwidth{{\slshape\rightmark}\hfil\thepage}
      \vskip 3\p@ \hrule}}%
```

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
```

We shift the page one `\baselineskip` to the bottom to compensate for the headings.

```
      \topmargin\emptyfoottopmargin
⟨/refart | refrep⟩
```

For the refart 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.

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

In the refrep document class 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.

```
⟨*refrep⟩
      \def\chaptermark##1{%
        \markboth {\ifnum \c@secnumdepth >\m@ne
            \@chapapp\ \thechapter \ \fi
            ##1}{}}%
      \def\sectionmark##1{%
        \markright {\ifnum \c@secnumdepth >\z@
            \thesection \ \fi
            ##1}}}
⟨/refrep⟩
```

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...`.

```
⟨*refart | refrep⟩
\else
    \def\ps@headings{%
      \let\@oddfoot\@empty
      \def\@oddhead{\hss\vbox to \z@{\vss\hsize=\fullwidth
      \hb@xt@\fullwidth{{\slshape\rightmark}\hfil\thepage}
      \vskip 3\p@ \hrule}}%
      \let\@mkboth\markboth
```

We shift the page one `\baselineskip` to the bottom to compensate for the headings.

```
      \topmargin\emptyfoottopmargin
⟨/refart | refrep⟩
```

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

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

```
⟨*refrep⟩
      \def\chaptermark##1{%
        \markright {\ifnum \c@secnumdepth >\m@ne
            \@chapapp\ \thechapter \ \fi
            ##1}}}
⟨/refrep⟩
⟨*refart | refrep⟩
\fi
```

`\ps@footings` The definition of the page style *footings* has to be different for two sided printing than it is for one sided printing.

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

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

```
      \let\@oddhead\@empty\let\@evenhead\@empty
      \def\@evenfoot{\hss\vbox to \z@{\vss\hsize=\fullwidth
      \hrule \vskip 3\p@
      \hb@xt@\fullwidth{\thepage\hfil\slshape\leftmark}}}%
      \def\@oddfoot{\hss\vbox to \z@{\vss\hsize=\fullwidth
      \hrule \vskip 3\p@
      \hb@xt@\fullwidth{{\slshape\rightmark}\hfil\thepage}}}%
```

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

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

We shift the page one `\baselineskip` to the top to compensate for the footings.

```
      \topmargin\emptyheadtopmargin
⟨/refart | refrep⟩
```

For the refart 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.

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

In the refrep document class 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@footings` is executed.

```
⟨*refrep⟩
      \def\chaptermark##1{%
        \markboth {\ifnum \c@secnumdepth >\m@ne
            \@chapapp\ \thechapter \ \fi
            ##1}{}}%
      \def\sectionmark##1{%
        \markright {\ifnum \c@secnumdepth >\z@
            \thesection \ \fi
            ##1}}}
⟨/refrep⟩
```

The definition of `\ps@footings` 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...`.

```
⟨*refart | refrep⟩
\else
    \def\ps@footings{%
      \let\@oddhead\@empty
      \def\@oddfoot{\hss\vbox to \z@{\vss\hsize=\fullwidth
      \hrule \vskip 3\p@
      \hb@xt@\fullwidth{{\slshape\rightmark}\hfil\thepage}}}%
      \let\@mkboth\markboth
```

We shift the page one `\baselineskip` to the top to compensate for the footings.

```
      \topmargin\emptyheadtopmargin
⟨/refart | refrep⟩
```

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

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

```
⟨*refrep⟩
      \def\chaptermark##1{%
        \markright {\ifnum \c@secnumdepth >\m@ne
            \@chapapp\ \thechapter \ \fi
            ##1}}}
⟨/refrep⟩
⟨*refart | refrep⟩
\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{\hss\vbox to \z@{\vss\hsize=\fullwidth
    \hb@xt@\fullwidth{\thepage\hfil\slshape\leftmark}
    \vskip 3\p@ \hrule}}%
    \def\@oddhead{\hss\vbox to \z@{\vss\hsize=\fullwidth
    \hb@xt@\fullwidth{{\slshape\rightmark}\hfil\thepage}
    \vskip 3\p@ \hrule}}%
```

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
⟨+refrep⟩    \let\chaptermark\@gobble
    \let\sectionmark\@gobble
⟨+refart⟩    \let\subsectionmark\@gobble
```

We shift the page one `\baselineskip` to the bottom to compensate for the headings.

```
    \topmargin\emptyfoottopmargin
    }
```

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

```
\def\ps@myfootings{%
      \let\@oddhead\@empty\let\@evenhead\@empty
      \def\@evenfoot{\hss\vbox to \z@{\vss\hsize=\fullwidth
      \hrule \vskip 3\p@
      \hb@xt@\fullwidth{\thepage\hfil\slshape\leftmark}}}%
      \def\@oddfoot{\hss\vbox to \z@{\vss\hsize=\fullwidth
      \hrule \vskip 3\p@
      \hb@xt@\fullwidth{{\slshape\rightmark}\hfil\thepage}}}%
```

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

```
    \let\@mkboth\@gobbletwo
⟨+refrep⟩    \let\chaptermark\@gobble
    \let\sectionmark\@gobble
⟨+refart⟩    \let\subsectionmark\@gobble
```

We shift the page one `\baselineskip` to the top to compensate for the footings.

```
    \topmargin\emptyheadtopmargin
    }
```

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.

```
% \gdef\@date{\today}
```

\maketitle The definition of `\maketitle` depends on whether a separate title page is made. This is the default for `refrep`. If you want a titlepage with `refart` you can enable it using the `titlepage` option.

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.

```
\if@titlepage
  \newcommand\maketitle{\begin{titlepage}%
  \let\footnotesize\small
  \let\footnoterule\relax
  \let\footnote\thanks
  \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
  \def\@makefnmark%
      {\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
  \long\def\@makefntext##1{%
       \@setpar{\@@par
          \@tempdima = \hsize
          \advance\@tempdima -1em
          \parshape \@ne 1em \@tempdima}%
       \par\parindent 1em \noindent
       \hb@xt@\z@{\hss\@textsuperscript{\normalfont\@thefnmark}\,}##1}
```

We center the entire title vertically; the centering is set off a little by adding a `\vskip`. In compatibility mode the page number is set to 0 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{%
         \@setpar{\@@par
            \@tempdima = \hsize
            \advance\@tempdima -1em
            \parshape \@ne 1em \@tempdima}%
         \par\parindent 1em \noindent
         \hb@xt@\z@{\hss\@textsuperscript{\normalfont\@thefnmark}\,}##1}
```

If this is a twocolumn document we start a new page in two-column 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 two-column 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 *empty* layout. We call `\@thanks` to produce the footnotes.

```
      \thispagestyle{empty}\@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 and put the title flush left using a `\Large` bold font with thick rules above and below. Then we put the autor information flush right in slanted type. This title will allways show the date unless it is set to nothing, using the `\date{}` command.

```
  \def\@maketitle{%
```

```
    \newpage
    \null
    \longthickrule\vskip1.5em%
    \let \footnote \thanks
    {\secshape \parskip\z@ \parindent\z@
    \Large\bfseries \@title \par}%
    \vskip1.5em\longthickrule\vskip1.5em%
    {\normalsize
      \lineskip .5em%
      \begin{flushright}%
        {\slshape\@author\par}
        \vskip 1em%
        {\@date}%
      \end{flushright}\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 macros, `\@startsection` and `\secdef`, which are defined by `latex.dtx`. They are not described here, see the `classes.dtx` for more information.

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 ??) Most of them are already defined by `latex.dtx`, so they are only shown here.

```
⟨+refrep⟩\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.

```
⟨+refart⟩\setcounter{secnumdepth}{3}
⟨+refrep⟩\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}
```

```
⟨+refart⟩\newcounter {section}
⟨*refrep⟩
\newcounter {chapter}
\newcounter {section}[chapter]
⟨/refrep⟩
\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 numberal.

`\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.

```
\renewcommand\thepart          {\@Roman\c@part}
⟨+refart⟩\renewcommand\thesection       {\@arabic\c@section}
⟨*refrep⟩
\renewcommand\thechapter       {\@arabic\c@chapter}
\renewcommand\thesection       {\thechapter.\@arabic\c@section}
⟨/refrep⟩
\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 empty. The `\appendix` command redefines it to be '`\appendixname`'.

```
⟨+refrep⟩\newcommand\@chapapp{}
```

7.2.4 Parts

`\part`

The command to start a new part of our document.

In the refart class the definition of `\part` is rather simple; we start a new paragraph, add a little whitespace, suppress the indentation of the first paragraph and make use of `\@secdef`. As in other sectioning commands (cf. `\@startsection` in the LaTeX 2_ε kernel), we need to check the `@noskipsec` switch and force horizontal mode if it is set.

```
⟨*refart⟩
\newcommand\part{%
  \if@noskipsec \leavevmode \fi
```

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

For the refrep class things are a bit different.

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

```
⟨*refrep⟩
\newcommand\part{%
  \if@openright
    \cleardoublepage
  \else
    \clearpage
  \fi
  \thispagestyle{empty}%
```

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 fill glue from disappearing.

```
                 \null\vfil
```

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

```
                 \secdef\@part\@spart}
⟨/refrep⟩
```

`\@part` This macro does the actual formatting of the title of the part. Again the macro is differently defined for the refart document class than for the document class refrep.

When *secnumdepth* is larger than -1 for the document class refart or -2 for the document class refrep, we have a numbered part, otherwise it is unnumbered.

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

We print the title flush left, we also prevent breaking between lines and reset the font.

```
    \longrule\medskip
   {\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.

```
    \Large
⟨+refart⟩      \ifnum \c@secnumdepth >\m@ne
⟨+refrep⟩      \ifnum \c@secnumdepth >-2 \relax
    \thepart.\quad
    \fi
    #2\par \medskip
    \longrule\bigskip%
```

Then we empty the mark registers, leave some whitespace and call `\@afterheading` to takes care of suppressing the indentation.

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

`\@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.

```
⟨*refart | refrep⟩
\def\@spart#1{%
   \longrule\medskip
   {\parindent \z@ \raggedright
    \interlinepenalty \@M
    \normalfont
    \Large #1\par}%
   \medskip\longrule
    \nobreak
    \vskip 3ex
    \@afterheading}
⟨/refart | refrep⟩
```

7.2.5 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*.

```
⟨*refrep⟩
\newcommand\chapter{\if@openright\cleardoublepage\else\clearpage\fi
                             \if@pageperchapter\setcounter{page}{1}\fi
                      \thispagestyle{empty}%
```

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 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
                      \refstepcounter{chapter}%
                      \typeout{\@chapapp\space\thechapter.}%
                      \addcontentsline{toc}{chapter}%
                                {\protect\numberline{\thechapter}#1}%
                    \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 whitespace 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 two-column 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 whitespace. 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{%
  \longthickrule\bigskip%
  {\parindent \z@ \secshape \normalfont \Large\bfseries
```

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.

```
  \@hangfrom{\ifnum \c@secnumdepth >\m@ne
     \@chapapp\space \thechapter\quad
  \fi}%
```

Now we set the title in a large bold font. We prevent a page break at this point and leave some whitespace before the text begins.

```
  #1\par}
  \bigskip\longthickrule\bigskip
  }
```

`\@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{%
    \longthickrule\bigskip%
    {\parindent \z@ \secshape \normalfont
    \Large \bfseries  #1\par}
    \bigskip\longthickrule\bigskip
    }
⟨/refrep⟩
```

7.2.6 Lower level headings

`\secshape`

```
\newcommand\secshape{\leftskip=-\leftmarginwidth%
                     \rightskip=\@flushglue%
                     \hyphenpenalty=2000}
```

These commands all make use of `\@startsection`.

`\section` This gives a normal heading with whitespace above and below the heading, the title set in `\large\bfseries`, and no indentation on the first paragraph.

```
\newcommand\section{\@startsection {section}{1}{\z@}%
                                   {-2ex \@plus -1ex \@minus -.2ex}%
                                   {0.5ex \@plus .2ex}%
                                   {\secshape\normalfont\large\bfseries}}
```

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

```
\newcommand\subsection{\@startsection{subsection}{2}{\z@}%
                                     {-1.5ex\@plus -.5ex \@minus -.2ex}%
                                     {0.5ex \@plus .2ex}%
                                     {\secshape\normalfont\normalsize\bfseries}}
```

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

```
\newcommand\subsubsection{\@startsection{subsubsection}{3}{\z@}%
                                    {-1.5ex\@plus -.5ex \@minus -.2ex}%
                                    {0.5ex \@plus .2ex}%
                                    {\secshape\normalfont\normalsize\mdseries}}
```

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

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

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

```
\newcommand\subparagraph{\@startsection{subparagraph}{5}{\parindent}%
                                    {2ex \@plus 1ex \@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`

For efficiency, level-one list's values are defined at top level, and `\@listi` is defined to set only `\leftmargin`.

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
```

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
```

Here we set the top level leftmargin.

```
\setlength\leftmargin    {\leftmargini}
```

`\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}
```

`\@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.

```
⟨*refart | refrep⟩
\@beginparpenalty -\@lowpenalty
\@endparpenalty   -\@lowpenalty
\@itempenalty     -\@lowpenalty
⟨/refart | refrep⟩
```

`\@listI` `\@listi` `\@listI` defines top level and `\@listi` values of `\leftmargin`, `\parsep`, `\topsep`, and `\itemsep`

```
⟨*refart | refrep⟩
\def\@listI{\leftmargin\leftmargini
            \parsep \parskip
            \topsep \z@
            \itemsep\z@}
\let\@listi\@listI
```

We have to initialise these parameters.

```
\@listi
```

`\@listii` `\@listiii` `\@listiv` `\@listv` `\@listvi` Here are the same macros for the higher level lists.

```
\def\@listii {\leftmargin\leftmarginii
              \labelwidth\leftmarginii
              \advance\labelwidth-\labelsep
              \topsep    \z@
              \parsep    \parskip
              \itemsep   \z@}
\def\@listiii{\leftmargin\leftmarginiii
              \labelwidth\leftmarginiii
              \advance\labelwidth-\labelsep
              \topsep    \z@
              \parsep    \parskip
              \partopsep \z@
              \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}
⟨*refart | refrep⟩
```

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.

```
⟨*refart | refrep⟩
\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 the various itemization levels: the symbols used are bullet, bold en-dash, 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{}{%
                    \labelsep\marginparsep
                    \labelwidth\leftmarginwidth
                    \advance\labelwidth by \leftmargin
                    \advance\labelwidth by -\labelsep
                    \let\makelabel\descriptionlabel}}
               {\endlist}
```

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

```
\newcommand*\descriptionlabel[1]{%
           \ifdescriptionleft\else \hfil\fi
           \normalfont #1 \ifdescriptioncolon :\fi
           \ifdescriptionleft \hfil \fi}
```

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.

```
\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 refart document class– we have to check if we are in twocolumn mode. In that case the abstract is as a `\section*`, otherwise the quote 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}%
        \quote
      \fi}
      {\if@twocolumn\else\endquote\fi}
\fi
```

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 Example

\example The example environment is a verse environment with tt font which tries to avoid page brakes at the `\begin{example}`.

```
\newenvironment{example}
             {\@beginparpenalty=\@highpenalty
```

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

7.4.6 Theorem

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

7.4.7 Titlepage

`titlepage` In the normal environments, the titlepage environment does nothing but start and end a page, and inhibit page numbers. It also resets the page number to zero. In two-column style, it still makes a one-column page.

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

7.4.8 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.

```
⟨*refart⟩
\newcommand\appendix{\par
  \setcounter{section}{0}%
  \setcounter{subsection}{0}%
  \gdef\thesection{\@Alph\c@section}}
```

```
⟨/refart⟩
```

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

- issue a `\newpage` if pageperchapter is defined, otherwise the page number would come out wrong.
- 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.

```
⟨*refrep⟩
\newcommand\appendix{\par
  \if@pageperchapter\newpage\fi
  \setcounter{chapter}{0}%
  \setcounter{section}{0}%
  \gdef\@chapapp{\appendixname}%
  \gdef\thechapter{\@Alph\c@chapter}}
⟨/refrep⟩
```

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` The equation counter will be reset at 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.

```
⟨+refart⟩\renewcommand\theequation{\@arabic\c@equation}
⟨*refrep⟩
\@addtoreset{equation}{chapter}
\renewcommand\theequation
{\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@equation}
⟨/refrep⟩
```

`\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` == `\@float`figure, `\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 the figures are numbered per chapter.

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

`\fps@figure` `\ftype@figure` `\ext@figure` `\num@figure` Here are the parameters for the floating objects of type 'figure'.

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

`figure` `figure*` Here is 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 the tables are numbered per chapter.

```
⟨*refart⟩
\newcounter{table}
\renewcommand\thetable{\@arabic\c@table}
⟨/refart⟩
⟨*refrep⟩
\newcounter{table}[chapter]
\renewcommand\thetable%
     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@table}
⟨/refrep⟩
```

`\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*` Here is 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 whitespace 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. If 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.

```
  \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 `\@newfontswitch`, 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` Here are 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. 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` $\geq$ `\@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}
⟨+refart⟩\setcounter{tocdepth}{3}
⟨+refrep⟩\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{%
⟨*refrep⟩
    \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
⟨/refrep⟩
⟨+refart⟩    \section*{\contentsname
        \@mkboth{\contentsname}{\contentsname}}%
```

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

```
    \@starttoc{toc}%
    \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 page break 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
⟨+refart⟩    \addpenalty\@secpenalty
⟨+refrep⟩    \addpenalty{-\@highpenalty}%
    \addvspace{2.25em \@plus\p@}%
    \begingroup
```

The we set `\parindent` to 0pt and use `\rightskip` to leave enough room for the page numbers. 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 page number, set flush right.

```
      {\leavevmode
       \large \bfseries #1\hfil \hbox to\@pnumwidth{\hss #2}}\par
```

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

```
       \nobreak
         \global\@nobreaktrue
         \everypar{\global\@nobreakfalse\everypar{}}%
      \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 page break should occur, it occurs *before* this entry. Also a little whitespace is added and a group begun to keep changes local.

```
⟨*refrep⟩
\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 page break immediately after a chapter entry.

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

`\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 page break should occur, it occurs *before* this entry. Also a little whitespace is added and a group begun to keep changes local.

```
⟨*refart⟩
```

```
\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 page break immediately after a chapter entry.

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

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

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

`\l@subsection` `\l@subsubsection` `\l@paragraph` `\l@subparagraph` All lower level entries are defined using the macro `\@dottedtocline` (see above).

```
⟨*refart⟩
\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}}
⟨/refart⟩
⟨*refrep⟩
\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}}
⟨/refrep⟩
```

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{%
⟨*refrep⟩
    \if@twocolumn
      \@restonecoltrue\onecolumn
    \else
      \@restonecolfalse
    \fi
    \chapter*{\listfigurename
⟨/refrep⟩
⟨+refart⟩      \section*{\listfigurename
      \@mkboth{\listfigurename}%
              {\listfigurename}}%
    \@starttoc{lof}%
⟨+refrep⟩     \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{%
⟨*refrep⟩
    \if@twocolumn
      \@restonecoltrue\onecolumn
    \else
      \@restonecolfalse
    \fi
    \chapter*{\listtablename
⟨/refrep⟩
⟨+refart⟩      \section*{\listtablename
      \@mkboth{\listtablename}{\listtablename}}%
    \@starttoc{lot}%
⟨+refrep⟩     \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
\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]
⟨+refart⟩      {\section*{\refname
⟨+refart⟩          \@mkboth{\refname}{\refname}}%
⟨+refrep⟩      {\chapter*{\bibname
⟨+refrep⟩          \@mkboth{\bibname}{\bibname}}%
      \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
            \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}
%(    \end{macrocode}
% \end{macro}
%
% \begin{macro}{\@cite}
%    The output of the |\cite| command is produced by this macro. The
%    default from \file{latex.dtx} is used.
%    \begin{macrocode}
% \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` whitespace can be added.

```
\newenvironment{theindex}
               {\if@twocolumn
                  \@restonecolfalse
                \else
                  \@restonecoltrue
                \fi
                \begin{fullpage}
                \let\twocolumn\REF@twocolumn
                \columnseprule \z@
                \columnsep 35\p@
⟨+refart⟩                \twocolumn[\section*{\indexname}]%
⟨*refrep⟩
                \if@pageperchapter
                   \setcounter{page}{1}
                   \ifnum \c@secnumdepth >\m@ne
                      \refstepcounter{chapter}%
                      \typeout{\@chapapp\space\thechapter.}%
                      \addcontentsline{toc}{chapter}
                         {\protect\numberline{\thechapter}\indexname}%
                   \else
                      \addcontentsline{toc}{chapter}{\indexname}%
                   \fi
                   \addtocontents{lof}{\protect\addvspace{10\p@}}%
                   \addtocontents{lot}{\protect\addvspace{10\p@}}%
                   \twocolumn[\@makechapterhead{\indexname}]%
                \else
                   \twocolumn[\@makeschapterhead{\indexname}]%
                \fi
⟨/refrep⟩
                \@mkboth{\indexname}%
                        {\indexname}%
                \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.

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

\@idxitem
\subitem
\subsubitem

Thsee 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 whitespace 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
  \kern 2.6\p@}
```

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

```
% \newcounter{footnote}
⟨+refrep⟩\@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.

```
\long\def\@makefntext#1{%
   \@setpar{\@@par
            \@tempdima = \hsize
            \advance\@tempdima-10\p@
            \parshape \@ne 10\p@ \@tempdima}%
   \par
   \parindent 1em\noindent
   \hb@xt@\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.

```
\long\def\@makefntext#1{%
   \@setpar{\@@par
      \@tempdima = \hsize
      \advance\@tempdima -1em
      \parshape \@ne 1em \@tempdima}%
    \par\parindent 1em \noindent
    \hb@xt@\z@{\hss\@textsuperscript{\normalfont\@thefnmark}\,}#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.

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

9 New commands

`\@addmarginpar` Redefine the `\@addmarginpar` command to only use the left margin.

```
\def\@addmarginpar{\@next\@marbox\@currlist{\@cons\@freelist\@marbox
    \@cons\@freelist\@currbox}\@latexbug\@tempcnta\@ne
    \if@twocolumn
       \if@firstcolumn \@tempcnta\m@ne \fi
    \else
       \@tempcnta\m@ne
    \fi
    \ifnum\@tempcnta <\z@  \global\setbox\@marbox\box\@currbox \fi
    \@tempdima\@mparbottom
    \advance\@tempdima -\@pageht
    \advance\@tempdima\ht\@marbox
    \ifdim\@tempdima >\z@
       \@@warning{Marginpar on page \thepage\space moved}%
    \else
       \@tempdima\z@
    \fi
    \global\@mparbottom\@pageht
    \global\advance\@mparbottom\@tempdima
    \global\advance\@mparbottom\dp\@marbox
    \global\advance\@mparbottom\marginparpush
    \advance\@tempdima -\ht\@marbox
    \global\setbox \@marbox
    \vbox {\vskip \@tempdima \box \@marbox}%
    \global \ht\@marbox \z@
    \global \dp\@marbox \z@
    \kern -\@pagedp
    \nointerlineskip
    \hb@xt@\columnwidth
      {\ifnum \@tempcnta >\z@
          \hskip\columnwidth \hskip\marginparsep
       \else
          \hskip -\marginparsep \hskip -\marginparwidth
       \fi
      \box\@marbox \hss}%
    \nointerlineskip
    \hbox{\vrule \@height\z@ \@width\z@ \@depth\@pagedp}}
```

9.1 Margin commands

\marginlable
\seealso
\attention
\attentionsymbol

This defines three commands to put information in the margin: \marginlabel buts the argument into a flush right marginpar, \attention puts \attentionsymbol to the left of the text to mark an important piece of text and \seealso puts a → to the left of the margin to mark a reference within the text. \attentionsymbol is defined as **! → but can be changed with a \renewcommand{\attentionsymbol}{:-)} command.**

```
\newcommand*{\marginlabel}[1]
{\mbox{}\marginpar{\raggedleft #1}\ignorespaces}
\newcommand*{\seealso}[1]
    {\mbox{}\marginpar{\small $\rightarrow$ #1}\ignorespaces}
\newcommand*{\attention}[1][\attentionsymbol]
    {\mbox{}\marginpar{\raggedleft #1}}
\newcommand*{\attentionsymbol}{\large\bfseries ! $\rightarrow$}
```

9.2 Rules

\longrule
\longthickrule

These rules are used in several places, like the title, new parts and chapters and for maxi- and fullpages.

```
\def\longrule{\par\hb@xt@\linewidth{\hss
                \vrule width \fullwidth height 0.4\p@ depth \z@}\par}
\def\longthickrule{\par\hb@xt@\linewidth{\hss
                \vrule width \fullwidth height 1.0\p@ depth \z@}\par}
```

9.3 Pages

maxipage
fullpage

The \maxipage is a minipage which uses the full width of the page with optional rules on the top and bottom. A maxipage can not split over pages. You can use it for wide tables, long math equations and the like. It can be used in floats.

The \fullpage changes the page layout such that normal text and all environments use the full width of the page. Inside the \fullpage-environment, the \leftmarginwidth is reset to 0, thus it is possible to start a new chapter inside a \fullpage. This will be used in the index.

```
\newenvironment{maxipage}{\par
                \mbox{}\kern-\leftmarginwidth %\kern-\@totalleftmargin
                \begin{minipage}{\fullwidth}
                  \medskip \ifmaxipagerule \hrule\medskip \fi
                  \parskip = 0.5\baselineskip
                  \def\marginpar{%
⟨+refart⟩                 \ClassError{Refart}
⟨+refrep⟩                 \ClassError{Refrep}
```

```
                  {Marginpar not allowed within Maxipage.}
                  {Where should I put them?\MessageBreak
                   I'm using the full pagewidth.}}}
                {\par \vskip\parskip
                   \medskip \ifmaxipagerule \hrule\medskip \fi
                 \end{minipage}\par}
\newenvironment{fullpage}{%
                \clearpage
                \textwidth=\fullwidth
                \addtolength\oddsidemargin  {-\leftmarginwidth}
                \setlength\evensidemargin{\oddsidemargin}
                \leftmarginwidth=\z@
                \hsize=\fullwidth
                \linewidth=\fullwidth
                \columnwidth=\fullwidth
                \def\marginpar{%
⟨+refart⟩                        \ClassError{Refart}
⟨+refrep⟩                        \ClassError{Refrep}
                  {Marginpar not allowed within Fullpage.}
                  {Where should I put them? I'm already\MessageBreak
                   using the whole page for text.}}}
                {\clearpage}
```

9.4 Miscellaneous

\condbreak
\noparskip

The \condbreak{length} controls page breaks: If less then length is left on this page it will be moved to the next page. Thus it will remain together, either on this page or on the next.

\noparskip removes the vertical parskip like \noindent removes the parindent.

```
\def\condbreak#1{\vskip \z@ plus #1\pagebreak[3]\vskip \z@ plus -#1\relax}
\def\noparskip{\vskip-\parskip}
```

\REF@twocolumn
\twocolumn

Since this layout does not support \twocolumn the command is disabled but saved in \REF@twocolumn. The saved version will be used in the index. This is still experimental! Don't rely on it in future releases.

```
\let\REF@twocolumn\twocolumn
\def\twocolumn{%
⟨+refart⟩\ClassError{Refart}
⟨+refrep⟩\ClassError{Refrep}
{Sorry, there is no twocolumn layout in this class}
{Can you imagine how twocolumn layout will look?\MessageBreak
 That's why!}}
```

9.5 Obsolete commands

Well, these comands are not really obsolete, but they are not implemented in this version and will not be implemented later unless there is popular demand.

`\makeauthor`: The author is printed when `\maketitle` is executed thus there is no need for this command.

`\setleftmarginwidth` has been used in version 1.1 to change the horizontal layout. I would prefer to set the `leftmarginfraction` instead but I'm still open to suggestions from users.

9.6 Future commands

The following commands are not yet implemented but sound like a good idea.

`\ppc` `\pageperchapter` This gives you a page count per chapter like 1-1, 1-2, 2-1. Since this is often requested and would be usefull in a reference manual style. `\pageperchapter` is only supported in refrep.cls.

This version redefines the LaTeX `\@wrindex` command which writes the indexentry. This hack is needed to keep MakeIndex happy when processing the index-file. The `\ppc` command is responsible to extract the chapter number from the index-entry and reformat it. The chapter number can be a Roman or Alpha number but the page has to be arabic.

```
⟨+refrep⟩\newif\if@pageperchapter \@pageperchapterfalse
⟨+refrep⟩\newcommand{\pageperchapter}
⟨+refrep⟩    {\@pageperchaptertrue
⟨+refrep⟩     \let\ppthepage=\thepage
⟨+refrep⟩     \renewcommand\@pnumwidth{2.55em}
⟨+refrep⟩     \@openrighttrue
⟨+refrep⟩     \renewcommand\thepage{%
⟨+refrep⟩       \ifnum \c@chapter = \z@
⟨+refrep⟩          \ppthepage
⟨+refrep⟩       \else
⟨+refrep⟩          \thechapter\ -- \arabic{page}
⟨+refrep⟩       \fi
⟨+refrep⟩       }
⟨+refrep⟩    \def\@wrindex##1{%
⟨+refrep⟩       \ifnum \c@chapter = \z@
⟨+refrep⟩         \protected@write\@indexfile{}%
⟨+refrep⟩         {\string\indexentry{##1}{\arabic{page}}}%
⟨+refrep⟩       \else
⟨+refrep⟩         \protected@write\@indexfile{}%
⟨+refrep⟩         {\string\indexentry{##1|ppc{\thechapter}}%
⟨+refrep⟩         {\arabic{page}}}%
⟨+refrep⟩       \fi
⟨+refrep⟩       \endgroup
⟨+refrep⟩       \@esphack
⟨+refrep⟩       }
⟨+refrep⟩       \def\ppc##1##2{##1 -- ##2}
⟨+refrep⟩     }
```

`\leftmarginfraction` This provides an interface to change the horizontal layout. In this version, the margin is set to 0.3 `fullwidth`, this may change in future versions.

10 Initialization

10.1 Words

`\contentsname` `\listfigurename` `\listtablename` `\refname` `\bibname` `\indexname` `\figurename` `\tablename` `\partname` `\chaptername` `\appendixname` `\abstractname`

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.

```
\newcommand\contentsname{Contents}
\newcommand\listfigurename{List of Figures}
\newcommand\listtablename{List of Tables}
⟨+refart⟩\newcommand\refname{References}
⟨+refrep⟩\newcommand\bibname{Bibliography}
\newcommand\indexname{Index}
\newcommand\figurename{Figure}
\newcommand\tablename{Table}
\newcommand\partname{Part}
⟨+refrep⟩\newcommand\chaptername{Chapter}
\newcommand\appendixname{Appendix}
\newcommand\abstractname{Abstract}
```

10.2 Date

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

```
\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}
```

10.3 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@}
```

10.4 The page style

We have *plain* pages in the document classes refart and refrep unless the user specified otherwise. We use arabic page numbers.

```
\pagestyle{plain}
\pagenumbering{arabic}      % Arabic page numbers
```

10.5 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.

```
\if@twoside
\else
  \raggedbottom
\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.

```
\if@twocolumn
  \twocolumn
  \sloppy
  \flushbottom
```

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

```
\else
  \onecolumn
\fi
⟨/refart | refrep⟩
```