Contents

1 Introduction 3

2 The docstrip modules 3

3 Initial Code 3

4 Declaration of Options 4
  4.1 Setting Paper Sizes 4
  4.2 Choosing the type size 5
  4.3 Two-side or one-side printing 5
  4.4 Draft option 6
  4.5 Titlepage option 6
  4.6 openright option 6
  4.7 Table of contents formatting 6
  4.8 Formatting of the title 6
  4.9 Twocolumn printing 7
  4.10 Equation numbering on the left 7
  4.11 Flush left displays 7
  4.12 Open bibliography 7

5 Executing Options 7

6 Loading Packages 8

7 Document Layout 8
  7.1 Fonts 8
  7.2 Paragrapging 11
  7.3 Page Layout 13
    7.3.1 Vertical spacing 14

*This file has version number v2.1c, last revised 2020/02/22.


1 Introduction

This file contains the set of document classes that were made available by Working Group 13 of the NTG (Nederlandstalige \TeX\ Gebruikersgroep). They are compatible with the standard \LaTeX\2e document classes, but implement different layouts.

2 The docstrip modules

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

- artikel: produce the documentclasses artikel\?
- rapport: produce the documentclasses rapport\?
- 10pt: produce the class option for 10pt
- 11pt: produce the class option for 11pt
- 12pt: produce the class option for 12pt
- boek: produce the documentclasses boek\?
- type1: produce the ‘1’ variants of the classes
- type2: produce the ‘2’ variants of the classes
- type3: produce the ‘3’ variants of the classes
- driver: produce a documentation driver file

3 Initial Code

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

\@ptsize This control sequence is used to store the second digit of the pointsize we are typesetting in. So, normally, it’s value is one of 0, 1 or 2.
\if@restonecol When the document has to printed in two columns, we sometimes have to temporarily switch to one column. This switch is used to remember to switch back.
\newif\if@restonecol
\iftitlepage A switch to indicate if a titlepage has to be produced. For the artikel document class the default is not to make a separate titlepage.
\newif\if@titlepage
\ifname\@titlepagefalse
\else\@titlepagetrue\fi
\ifopenright A switch to indicate if chapters must start on a right-hand page. The default for the report class is no; for the book class it’s yes.
\newif\if@openright
\ifmainmatter The switch \if@mainmatter, only available in the document class book, indicates whether we are processing the main material in the book.
\newif\if@mainmatter \@mainmattertrue
\ifoldtoc A switch to indicate if ‘old’ layout of the table of contents should be produced. These document classes normally produce a table of contents that looks quite different from what the standard classes produce.
\newif\if@oldtoc \@oldtocfalse
\ifallcaps By default the text on the titlepage is set in capital letters. This can be disabled by the option mctitle, which sets the switch \if@allcaps to false.
\newif\if@allcaps
\iftitlecentered In the document classes artikel3 and rapport3 the default placement of the title that is produced by \maketitle is flushleft. This can be changed by the switch \if@titlecentered.
\newif\if@titlecentered \@titlecenteredfalse
\ifrevlabel These document classes need to be able to change the positioning of the label in labeled lists. This switch is used for that purpose.
\newif\if@revlabel

4 Declaration of Options

4.1 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.

\begin{verbatim}
\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}}
\DeclareOption{landscape}
{\setlength\@tempdima \{\paperheight\%
 \setlength\paperheight \{\paperwidth\%
 \setlength\paperwidth \{\@tempdima\}}
\end{verbatim}

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

\begin{verbatim}
\DeclareOption{oneside}{\@twosidefalse \@mparswitchfalse}
\DeclareOption{twoside}{\@twosidetrue \@mparswitchtrue}
\end{verbatim}

\subsection*{4.2 Choosing the type size}

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

\begin{verbatim}
\DeclareOption{10pt}{\renewcommand\@ptsize{0}}
\DeclareOption{11pt}{\renewcommand\@ptsize{1}}
\DeclareOption{12pt}{\renewcommand\@ptsize{2}}
\end{verbatim}

\subsection*{4.3 Two-side or one-side printing}

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

\begin{verbatim}
\DeclareOption{oneside}{\@twosidefalse \@mparswitchfalse}
\DeclareOption{twoside}{\@twosidetrue \@mparswitchtrue}
\end{verbatim}
4.4 Draft option

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

\DeclareOption{draft}{\setlength\overfullrule{5pt}}
\DeclareOption{final}{\setlength\overfullrule{0pt}}

4.5 Titlepage option

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

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

4.6 openright option

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

\DeclareOption{openright}{\@openrighttrue}
\DeclareOption{openany}{\@openrightfalse}

For these document classes there used to be a file voorwerk.sty which was a replacement for titlepag.sty. Therefore we also have the option voorwerk.

\DeclareOption{voorwerk}{\@titlepagetrue}
\DeclareOption{geenvoorwerk}{\@titlepagefalse}

4.7 Table of contents formatting

This document class uses a new layout for the table of contents, but in order to maintain compatibility with the standard \LaTeX document classes we supply an extra option: oldtoc. If this option is specified the switch \if@oldtoc will be set true.

\DeclareOption{oldtoc}{\@oldtoctrue}

4.8 Formatting of the title

The option titlecentered changes the behaviour of the \maketitle command. It then produces a title like it does for the artikel1 document class.

\DeclareOption{titlecentered}{\@titlecenteredtrue}

In the rapport and boek document styles the titlepage uses all capital letters. The option mctitle (for ‘mixed case’) prevents this.

\DeclareOption{mctitle}{\@allcapsfalse}
\DeclareOption{uctitle}{\@allcapstrue}
4.9 Twocolumn printing

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

\[\text{\DeclareOption{onecolumn}{\@twocolumnfalse}}\]
\[\text{\DeclareOption{twocolumn}{\@twocolumntrue}}\]

4.10 Equation numbering on the left

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

\[\text{\DeclareOption{leqno}{\input{leqno.clo}}}\]

4.11 Flush left displays

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

\[\text{\DeclareOption{fleqn}{\input{fleqn.clo}}}\]

4.12 Open bibliography

The option \texttt{openbib} produces the “open” bibliography style, in which each block starts on a new line, and succeeding lines in a block are indented by $\texttt{\bibindent}$.

\[\text{\DeclareOption{openbib}{%}\}
\text{\AtEndOfPackage{\%}\}
\text{\renewcommand}\texttt{\openbib@code{%\}
\text{\advance}\texttt{\leftmargin}\texttt{\bibindent}\}
\text{\itemindent} -\texttt{\bibindent}\}
\text{\listparindent} \texttt{\itemindent}\}
\text{\parsep} \z@\}
\text{\%}\}
\text{\In addition the definition of} \texttt{\newblock} \text{is overwritten.}\]
\[\text{\renewcommand}\texttt{\newblock}(\texttt{\par})\%\]
\[}\]

5 Executing Options

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

\[\text{\texttt{\(*artikel\)}}\]

7
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.

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

6 Loading Packages

These class files do not load additional packages.

7 Document Layout

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

7.1 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:

\begin{verbatim}
\@vpt  5 \@vipt  6 \@viipt  7
\@viiipt 8 \@ixpt  9 \@xpt 10
\@xipt 10.95 \@xipt 12 \@xivpt 14.4
\end{verbatim}
The user-level command for the main size is \texttt{\normalsize}. Internally \LaTeX{} uses \texttt{\normalsize} when it refers to the main size. \texttt{\@normalsize} will be defined to work like \texttt{\normalsize} if the latter is redefined from its default definition (that just issues an error message). Otherwise \texttt{\@normalsize} simply selects a 10pt/12pt size.

The \texttt{\normalsize} macro also sets new values for \texttt{\abovedisplayskip}, \texttt{\abovedisplayshortskip} and

\verb|\abovedisplayskip| 10pt \plus6pt \minus5pt
\verb|\abovedisplayshortskip| \z@ \plus3pt
\verb|\belowdisplayshortskip| 6.5pt \plus3.5pt \minus3pt

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

\texttt{\belowdisplayskip \abovedisplayskip}
\let\@listi\@listI}

Make \texttt{\@normalsize} a synonymm for \texttt{\normalsize}.

We initially choose the \texttt{\normalsize} font.

\texttt{\normalsize}

We use \texttt{\MakeRobust} instead of \texttt{\DeclareRobustCommand} above to avoid a log entry for the redefinition. But if we are running in a rollback situation (prior to 2015) we don’t touch it.

\texttt{\ifx\MakeRobust\@undefined \else
\MakeRobust\normalsize
\fi}

\texttt{\small} This is similar to \texttt{\normalsize}.

\texttt{\DeclareRobustCommand:\small{\%\@setfontsize\small\@ixpt11\}}
This is similar to \footnotesize.
\footnotesize{
\DeclareRobustCommand{\footnotesize}{\@setfontsize{\footnotesize}{\@viiipt}{\@viiipt}}
}\scriptsize{This is similar to \normalsize.}
\scriptsize{
\DeclareRobustCommand{\scriptsize}{\@setfontsize{\scriptsize}{\@viiipt}{\@viiipt}}
}\tiny{This is similar to \scriptsize.}
\tiny{
\DeclareRobustCommand{\tiny}{\@setfontsize{\tiny}{\@vpt}{\@vipt}}
}\large{This is similar to \tiny.}
\large{
\DeclareRobustCommand{\large}{\@setfontsize{\large}{\@xiipt}{14}}
}\Large{This is similar to \large.}
\Large{
\DeclareRobustCommand{\Large}{\@setfontsize{\Large}{\@xivpt}{18}}
}\LARGE{This is similar to \Large.}
\LARGE{
\DeclareRobustCommand{\LARGE}{\@setfontsize{\LARGE}{\@xviipt}{22}}
}\Huge{These are all much simpler than the previous macros, they just select a new
font size, but leave the parameters for displays and lists alone.}
\Huge{
\DeclareRobustCommand{\Huge}{\@setfontsize{\Huge}{\@xviipt}{22}}
}
\lineskip \normallineskip

These parameters control \TeX^\textregistered\'s behaviour when two lines tend to come too close together.

\baselineskip

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

\unitindent

These document classes all use a single dimension for a number of layout parameters:

\begin{itemize}
  \item the label width in section heading,
  \item the \parindent
  \item the footnote label indent (= half \unitindent)
  \item listindent on the first level
\end{itemize}
The default setting accommodates three levels of single digit section numbering.

\typemargin

Other indentations are maximal label width plus white space.

\if@needwriteindent

If this is not enough, a new width is calculated, set, and the file.aux file contains an instruction that will set \unitindent on the next run.

For this we need a switch

\@indentset

And a command that sets the various parameters.

\@writeindent

The \end{document} command will call \@writeindent to write the final width of \unitindent on the .aux file. Also a command is written to set \unitindent. To be compatible with other document classes a check is written to the .aux file for the existence of \unitindent. This prevents nasty errors when another document class is used.

\AtEndDocument{%
\if@filesw
\if@needwriteindent
\@writeindent{\the\unitindent}
\fi
\fi}

In the document class artikel2 the width of \unitindent is fixed and related to \othermargin.

\begin{verbatim}
  \@indentset{\parindent=\unitindent\leftmargini=\unitindent\@needwriteindenttrue}
  \@writeindent{\immediate\write\@mainaux{\string\@ifundefined{unitindent}{\string\newdimen\string\unitindent\let\string\@indentset\relax}{}\immediate\write\@mainaux{\global\string\unitindent=#1\string\relax\string\@indentset \string\relax}}}
  \immediate\write\@mainaux{
    \if@filesw
      \if@needwriteindent
        \@writeindent{\the\unitindent}
      \fi
      \fi}
\end{verbatim}
\parskip gives extra vertical space between paragraphs and \parindent is the width of the paragraph indentation. The value of \parindent depends on whether we are in two column mode.

\setlength{\parindent}{\unitindent}

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.

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

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

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

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

7.3 Page Layout

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

\texttt{\headheight}  The \texttt{\headheight} is the height of the box that will contain the running head. The \texttt{\headsep} is the distance between the bottom of the running head and the top of the text. \texttt{\topskip} is the \texttt{\baselineskip} for the first line on a page.

228 \texttt{(+10pt | 11pt | 12pt)}
229 \texttt{\setlength{\headheight}{12\p@}}
230 \texttt{\setlength{\headsep}{25\p@}}
231 \texttt{(10pt | \setlength{\topskip}{10\p@})}
232 \texttt{(11pt | \setlength{\topskip}{11\p@})}
233 \texttt{(12pt | \setlength{\topskip}{12\p@})}

\texttt{\footskip}  The distance from the baseline of the box which contains the running footer to the baseline of last line of text is controlled by the \texttt{\footskip}. Bottom of page:

234 \texttt{\setlength{\footskip}{30\p@}} \%

\texttt{\maxdepth}  The \TeX{} primitive register \texttt{\maxdepth} has a function that is similar to that of \texttt{\topskip}. The register \texttt{\@maxdepth} should always contain a copy of \texttt{\maxdepth}. In both plain \TeX{} and \LaTeX{} 2.09 \texttt{\maxdepth} had a fixed value of \texttt{4pt}; in native \LaTeX{}2e mode we let the value depend on the typesize. We set it so that \texttt{\maxdepth + \topskip = typesize \times 1.5}. As it happens, in these classes \texttt{\topskip} is equal to the typesize, therefore we set \texttt{\maxdepth} to half the value of \texttt{\topskip}.

235 \texttt{\if@compatibility}
236 \texttt{\setlength{\maxdepth}{4\p@}}
237 \texttt{\else}
238 \texttt{\setlength{\maxdepth}{.5\topskip}}
239 \texttt{\fi}

7.3.2 The dimension of text

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

240 \texttt{\if@compatibility}
241 \texttt{\if@twocolumn}
242 \texttt{\setlength{\textwidth}{410\p@}}
243 \texttt{\else}
244 \texttt{(10pt \setlength{\textwidth}{345\p@})}
245 \texttt{(11pt \setlength{\textwidth}{360\p@})}
246 \texttt{(12pt \setlength{\textwidth}{390\p@})}
247 \texttt{\fi}

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

248 \texttt{\else}

First, we calculate the maximum textwidth, which will we will allow on the selected paper and store it in \texttt{\@tempdimxa}. Then we store the length of a line with approximately 60 – 70 characters in \texttt{\@tempdimxb}. The values given are taken from...
the file `a4.sty` by Johannes Braams and Nico Poppelier and are more or less suitable when Computer Modern fonts are used.

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

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

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

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

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

First make sure that the compatibility mode gets the same dimensions as we had with \LaTeX\ 2.09. The number of lines was calculated as the floor of the old `\textheight` minus `\topskip`, divided by `\baselineskip` for `\normalsize`. The old value of `\textheight` was 528pt.
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.

\[\text{We leave at least a 1 inch margin on the top and the bottom of the page.}\]

\[\text{We also have to leave room for the running headers and footers.}\]

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

\[\text{From this we can calculate the height of the text.}\]

\[\text{The first line on the page has a height of } \texttt{topskip}.\]

\section*{Margins}

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

\begin{verbatim}
\texttt{\setlength\marginparsep}{10\p@}
\else
\setlength\marginparsep{11\p@}
\setlength\marginparsep{10\p@}
\fi
\setlength\marginparpush{5\p@}
\setlength\marginparpush{7\p@}
\end{verbatim}

Now we can give the values for the other margin parameters. For native \TeX, these are calculated.

\begin{verbatim}
\texttt{\if@compatibility}
\texttt{\if@twoside}
\texttt{\setlength\oddsidemargin}{44\p@}
\else
\setlength\evensidemargin{44\p@}
\fi
\fi
\end{verbatim}
When we are not in compatibility mode we can take the dimensions of the selected paper into account.

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

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

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

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

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

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

For one-sided printing we center the text on the page, by calculating the difference between textwidth and \paperwidth. Half of that difference is then used for the
With the above algorithm the \marginparwidth can come out quite large which we may not want.
\ifdim \marginparwidth >2in
\setlength\marginparwidth{2in}
\fi

Having done these calculations we make them pt values.
\@settopoint\oddsidemargin
\@settopoint\marginparwidth

The \evensidemargin can now be computed from the values set above.
\setlength\evensidemargin \{\paperwidth\}
\addtolength\evensidemargin{-2in}
\addtolength\evensidemargin{-\textwidth}
\addtolength\evensidemargin{-\oddsidemargin}

Setting \evensidemargin to a full point value may produce a small error. However it will lie within the error range a doublesided printer of todays technology can accurately print.
\@settopoint\evensidemargin
\fi

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

It can now be computed from the values set above.
\if@compatibility
\setlength\topmargin{27pt}
\else
\setlength\topmargin\{\paperheight\}
\addtolength\topmargin{-2in}
\addtolength\topmargin{-\headheight}
\addtolength\topmargin{-\headsep}
\addtolength\topmargin{-\textheight} % this might be wrong!
\addtolength\topmargin{-\footskip}
\fi

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

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

\footnotepsep is the space between the last line of the main text and the top of the first footnote.

7.3.5 Float placement parameters

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

Limits for the placement of floating objects

\c@topnumber The topnumber counter holds the maximum number of floats that can appear on the top of a text page.
\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.
\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.
\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}
The \texttt{dbltopnumber} counter holds the maximum number of two column floats that can appear on the top of a two column text page.

\begin{verbatim}
377 \setcounter{dbltopnumber}{2}
\end{verbatim}

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

\begin{verbatim}
378 \renewcommand{dbltopfraction}{.7}
\end{verbatim}

\texttt{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.

\begin{verbatim}
379 \renewcommand{dblfloatpagefraction}{.5}
\end{verbatim}

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

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

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

\texttt{\intextsep} is the space between in-text floats and the text.

\begin{verbatim}
381 (+10pt)
382 \setlength{\floatsep}{12\p@ \plus 2\p@ \minus 2\p@}
383 \setlength{\textfloatsep}{20\p@ \plus 2\p@ \minus 4\p@}
384 \setlength{\intextsep}{12\p@ \plus 2\p@ \minus 2\p@}
385 (/10pt)
386 (+11pt)
387 \setlength{\floatsep}{12\p@ \plus 2\p@ \minus 2\p@}
388 \setlength{\textfloatsep}{20\p@ \plus 2\p@ \minus 4\p@}
389 \setlength{\intextsep}{12\p@ \plus 2\p@ \minus 2\p@}
390 (/11pt)
391 (+12pt)
392 \setlength{\floatsep}{12\p@ \plus 2\p@ \minus 4\p@}
393 \setlength{\textfloatsep}{20\p@ \plus 2\p@ \minus 4\p@}
394 \setlength{\intextsep}{14\p@ \plus 4\p@ \minus 4\p@}
395 (/12pt)
\end{verbatim}

\texttt{\dblfloatsep} \texttt{\dbltextfloatsep} When floating objects that span the whole \texttt{textwidth} are placed on a text page when we are in two-column mode the separation between the float and the text is controlled by \texttt{\dblfloatsep} and \texttt{\dbltextfloatsep}.

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

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

\begin{verbatim}
396 (+10pt)
\end{verbatim}
Floats on their own page or column

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

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

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

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

\@dblfpbot
7.4 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).

\def\oddhead{\hbox{\textwidth}}
\def\evenhead{\hbox{\textwidth}}
\def\oddfoot{\hbox{\textwidth}}
\def\evenfoot{\hbox{\textwidth}}

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.

Several commands (\index, \maketitle) give a \thispagestyle{plain} command, which will overrule a \pagestyle{empty} command. This situation is almost always unwanted. Therefore we provide a more careful definition.

First save the original definition.
\let\Thispagestyle\thispagestyle
Then we provide the new definition, for which we must also adapt \pagestyle a little.

7.4.1 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.)

\TeX 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.

7.4.2 Defining the page styles

The pagestyle empty is defined in latex.dtx, but the pagestyle plain is slightly altered here. The difference is that the page numbers are set flush right in onesided and flush left and right in the twosided style.

\ps@plain
\renewcommand*{\ps@plain}{%
  The running heads are empty in this pagestyle, the page number appears in the running foot.
  \let\@oddhead\@empty\let\@evenhead\@empty
  \def\@oddfoot{\hfil\PageFont\thepage}\%
  \if@twoside
    \def\@evenfoot{\PageFont\thepage\hfil}\%
  \else
    \let\@evenfoot\@oddfoot
  \fi
  Because the running heads should be empty we let \@mkboth to \@gobbletwo, thus disabling the mark commands.
  \let\@mkboth\@gobbletwo}

\ps@headings
  The definition of the page style headings has to be different for two sided printing than it is for one sided printing.
  \if@twoside
  \def\ps@headings{%
    The running feet are empty in this page style, the running head contains the page number and one of the marks.
    \let\@oddfoot\@empty\let\@evenfoot\@empty
    \def\@evenhead{\{\PageFont\thepage}\hfil\MarkFont\leftmark}\%
    \def\@oddhead{\{\MarkFont\rightmark\}\hfil\PageFont\thepage}\%
  \else
    \def\ps@headings{%
  \fi
}
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`.

For the artikel document classes 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.

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

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

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

\sectionmark##1{}
\chaptermark##1{}
\subsectionmark##1{}
\let\@mkboth\@gobbletwo
\let\chaptermark\@gobble
\let\sectionmark\@gobble
\let\subsectionmark\@gobble

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.

\ps@myheadings

The macros are use to store the fonts that are used to typeset the pagenumber (\PageFont) and the marks (\MarkFont) in the running head and feet.

\RunningFonts

Use this macro to change the fonts that are used in the running heads.
8 Document Markup

8.1 The title

\title
\author
\date

These three macros are provided by \texttt{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 \texttt{\maketitle} command to use the information provided. The definitions of these macros are shown here for information.

\begin{verbatim}
527 \newcommand*{\title}[1]{\gdef\@title{#1}}
528 \newcommand*{\author}[1]{\gdef\@author{#1}}
529 \newcommand*{\date}[1]{\gdef\@date{#1}}
\end{verbatim}

The \texttt{\date} macro gets today’s date by default.

\TitleFont

This selects the font to use in the title of the document.

\begin{verbatim}
\newcommand*{\TitleFont}{\bfseries}
\end{verbatim}

\maketitle

The definition of \texttt{\maketitle} depends on whether a separate title page is made. This is the default for the rapport and boek document classes, but for the artikel classes it is optional. Note that the title, author and date information is printed in capital letters by default. This can be changed by the option \texttt{mctitle}.

When we are making a title page, we locally redefine \texttt{\footnotesize} and \texttt{\footnoterule} to change the appearance of the footnotes that are produced by the \texttt{\thanks} command.

\begin{verbatim}
\if@titlepage
\renewcommand*{\TitleFont}{\rmfamily}
\newcommand*{\maketitle}{% 
\begin{titlepage}%
\let\footnotesize\small
\let\footnoterule\relax
\let\footnote\thanks
\end{titlepage}%
\long\def\@makefntext##1{\parindent\z@
\def\labelitemi{\textendash}\@revlabeltrue
\leavevmode\@textsuperscript{\@thefnmark}\kern1em\relax ##1}
\renewcommand*{\thefootnote}{\@fnsymbol\c@footnote}%
\end{verbatim}

Footnotes on the titlepage, generated by the use of \texttt{\thanks}, use symbols in these document classes.

\begin{verbatim}
\long\def\makefntext#1{\parindent\z@
\def\labelitemi{\textendash}\@revlabeltrue
\leavevmode@textsuperscript{\@thefnmark}\kern1em\relax #1}
\renewcommand*\thefootnote{\@fnsymbol\c@footnote}%
\end{verbatim}

We center the entire title vertically; the centering is set off a little by adding a \texttt{\vskip}. In compatibility mode the pagernumber is set to 0 to keep the behaviour of \LaTeX{} 2.09 style files.

\begin{verbatim}
\if@compatibility\setcounter{page}{0}\fi
\null\vfil
\vskip 60\p@
\end{verbatim}
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}
\TitleFont
\begin{tabular}{c}
\if@allcaps\expandafter\uc@nothanks\@title\thanks\relax
\else\@title\fi\par
\end{tabular}
\vskip 1.5em
\{\large
\if@allcaps\uppercase\expandafter\uc@author\and\relax
\else\@author\fi
\fi
\end{tabular}\par
\end{center}

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}\relax
\global\let\thanks\relax
\global\let\maketitle\relax
\global\let\@thanks\@empty
\global\let\@title\@empty
\global\let\@date\@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
We want to have the title, author and date information in uppercase, but we have to be very careful not to put too much text in uppercase. The macros that perform the filtering of texts that shouldn’t be in uppercase were developed with the help of Howard Trickey.

\uc@nothanks

This macro takes all the text up to the first use of \thanks and passes it to \uppercase. The use of \futurelet will store the token after the \thanks in @tempa. The macro \u@tx uses that information to determine what to do next.

\uc@authornothanks

A document can have more than one author. Usually they are separated with \and. For each author a footnote –using \thanks can be present. Therefore this macro takes all the text up to the first use of \and, thus picking up all the information for one author. This is then passed to \uc@nothanks, which checks for the presence of \thanks. For this to work the argument of \uc@nothanks has to be delimited by \thanks\relax.

\uc@ax

When @tempa contains a \relax token nothing needs to be done, when it doesn’t we put in a linebreak \ the word ‘and’ (stored in \andname so that this control sequence can be redefined for other languages), another linebreak and we call \uc@authornothanks to continue processing. The \expandafter lets \TeX see the \fi first.

\uc@tx

This macro simply checks whether @tempa contains a \relax token. When it doesn’t further processing is performed by \u@ty.

\uc@ty

The macro \uc@ty gets executed when the \thanks that delimited text earlier on in the processing had a real argument. In that case it was a \thanks put in by
the user, not by these macros. Therefore the argument is now passed to \thanks and processing continues by calling \uc@nothanks.

\def\uc@ty#1{\thanks{#1}\uc@nothanks}

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 \@makefntext so that footnotemarks have zero width (to make the centering of the author names look better). We also want raised footnotemarks in the footnotes here.

\if\boek
\else
\newcommand\maketitle{\par
\begingroup
\renewcommand\thefootnote{\@fnsymbol\c@footnote}\
\@makefntext\@makefntext
\long\def\@makefntext##1{\parindent\z@ \labelitemi{\textendash} \leavevmode\hb@xt@.5\unitindent{\@textsuperscript{\normalfont\@thefnmark}\hfil}##1}\fi

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

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

\global\@topnum\z@
\@maketitle
\fi

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

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

\begin{verbatim}
\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\relax
\global\let\author\relax
\global\let\date\relax
\global\let\and\relax
\end{verbatim}

\@maketitle

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

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

\begin{verbatim}
\def\@maketitle{\newpage\null\vskip 2em\langle\type3\rangle
\if@titlecentered
\begin{center}
\let\footnote\thanks
\LARGE \TitleFont \@title \par\vskip 1.5em\large \TitleFont\lineskip .5em\begin{tabular}{c}\@author\end{tabular}\par\vskip 1em\large \TitleFont \@date\end{center}
\else
\LARGE \TitleFont \head@style \@title \par\vskip 1.5em\large \TitleFont\lineskip .5em\begin{tabular}{l}\@author\end{tabular}\par\vskip 1em\large \TitleFont \@date\fi}
\end{verbatim}
8.2 Chapters and Sections

8.2.1 Building blocks

The definitions in this part of the class file make use of two macros, \@startsection and \secedef, which are defined by latex.dtx. To understand what is going on here, we describe their syntax.

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

\@startsection\langle name\rangle\langle level\rangle\langle indent\rangle\langle beforeskip\rangle\langle afterskip\rangle\langle style\rangle \ optional *

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

\langle name\rangle The name of the user level command, e.g., ‘section’.

\langle level\rangle A number, denoting the depth of the section – e.g., chapter=1, section = 2, etc. A section number will be printed if and only if \langle level\rangle < the value of the seconumdepth counter.

\langle indent\rangle The indentation of the heading from the left margin

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

\langle afterskip\rangle If positive, this gives the skip to leave below the heading, else it gives the skip to leave to the right of a run-in heading.

\langle style\rangle Commands to set the style of the heading. Since the June 1996 release of \TeX\ the last command in this argument may be a command such as \MakeUppercase or \fbox that takes an argument. The section heading will be supplied as the argument to this command. So setting \#6 to, say, \bfseries\MakeUppercase would produce bold, uppercase headings.

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

\langle altheading\rangle Gives an alternative heading to use in the table of contents and in the running heads. This should be not present when the * form is used.

\langle heading\rangle The heading of the new section.
A sectioning command is normally defined to \texttt{\@startsection} and its first six arguments. The macro \texttt{\secdef} can be used when a sectioning command is defined without using \texttt{\@startsection}. It has two arguments:
\texttt{\secdef(unstarcmds)(starcmds)}

(unstarcmds) Used for the normal form of the sectioning command.
(starcmds) Used for the *-form of the sectioning command.

You can use \texttt{\secdef} as follows:
\begin{verbatim}
def\chapter { ... \secdef \CMDA \CMDB }
def\CMDA [#1]#2{ ... } % Command to define % \chapter[#1]{#2}
def\CMDB #1{ ... } % Command to define % \chapter*{#1}
\end{verbatim}

\head@style In the definition of chapter and section commands a number of settings frequently occur. Therefore we store them in a control sequence. Section headings are to be set extremely ragged right, with no hyphenations, not even at explicit hyphens.

\begin{verbatim}
\newcommand*{\head@style}{% interlinepenalty \@M hyphenpenalty=\@M \exhyphenpenalty=\@M rightskip=0cm plus .7\hsize\relax}
\end{verbatim}

\@sect The definition of this macro from \texttt{latex.dtx} needs to be repeated here because we want to modify its behaviour with respect to:
1. the width of the number, which is fixed;
2. checking the value of \texttt{\unitindent};
3. formatting the section title ragged right;
4. changing the argument of \texttt{\contentsline}.

\begin{verbatim}
def\@sect#1#2#3#4#5#6[#7]{% ifnum #2>c@secnumdepth \let\@svsec@empty \else \refstepcounter{#1}% \texttt{\@sect\#1\#2\#3\#4\#5\#6\#7}\% \let\@svsec \empty \else \reset@tocounter\#1\% \texttt{\@sect\#1\#2\#3\#4\#5\#6\#7}\% \fi \fi \let\@svsec \@empty \ifnum #2>c@secnumdepth \else \ifnum #2=0 \else \ifcase #6 \or \@dblarg \or \@dblarg \or \@dblarg \or \@dblarg \fi \fi \fi \ifcase #6 \or \@dblarg \or \@dblarg \or \@dblarg \or \@dblarg \fi \fi \fi \fi \fi \ifnum #2>c@secnumdepth \else \ifnum #2=0 \else \ifcase #6 \or \@dblarg \or \@dblarg \or \@dblarg \or \@dblarg \fi \fi \fi \fi \fi \let\@svsec \@empty \fi \fi \fi \fi \fi \fi
\end{verbatim}

The following code (within the group) checks the value of \texttt{\unitindent}. If the sectionnumber is wider than \texttt{\unitindent} its value is adapted and a flag is set to rememeber to store the new value in the .aux-file.

\begin{verbatim}(+type1|type3) \begingroup \setbox\@tempboxa=\hbox{#6}\relax
\end{verbatim}
\csname the#1\endcsname
\hskip.5em}
\ifdim\wd\@tempboxa>\unitindent
\global\unitindent=\wd\@tempboxa
\@indentset
\fi
\endgroup
⟨
/\textsc{type1}\break
/\textsc{type3}⟩
Since \@seccntformat might end with an improper \hskip which is scanning forward for plus or minus we end the definition of \@svsec with \relax as a precaution.
\protected@edef\@svsec{\@seccntformat{#1}\relax}
\fi
\@tempskipa #5\relax
\ifdim\@tempskipa>\z@
\begingroup
This \{ used to be after the argument to \textbackslash hangfrom but was moved here to allow commands such as \textbackslash MakeUppercase to be used at the end of \#6.
\#6{%
(+\textsc{type1}\textsc{type3})
\textbackslash hangfrom{\hskip \#3\relax\@svsec}\textbackslash head@style \#8\textbackslash endgraf}%
\endgroup
⟨\textbackslash type2⟩
\@hangfrom{\hskip \#3}
\textbackslash head@style \@svsec \hskip.3em\relax \#8\textbackslash endgraf}
\endgroup
\csname #1mark\endcsname{#7}%
\addcontentsline{toc}{#1}{%
\ifnum#2>\c@secnumdepth
\else
\protect\numberline{\csname the#1\endcsname}%
\fi
#7}%
\else
\def\@svsechd{\#6\hskip \#3\relax
\@svsec \#8\csname #1mark\endcsname{#7}%
\addcontentsline{toc}{#1}{%
\ifnum#2>\c@secnumdepth
\else
\protect\numberline{\csname the#1\endcsname}%
\fi
#7ipop}=\c@secnumdepth
\else
\protect\numberline{\csname the#1\endcsname}%
\fi
\def\@seccntformat{#1}{}
This macro was introduced in \LaTeX\ 2\epsilon, its definition is changed here to get the fixed with of the section number.
\def\@seccntformat{#1}{%
Similar changes need to be made to the definition of \@ssect, which is used in ‘starred’ sections.

This \ used to be after the argument to \hangfrom but was moved here to allow commands such as \MakeUppercase to be used at the end of #6.

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

Default initializations of \...mark commands. These commands are used in the definition of the page styles (see section 7.4.2) Most of them are already defined by latex.tex, so they are only shown here.

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

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

The value of the counter secnumdepth gives the depth of the highest-level sectioning command that is to produce section numbers.
For any counter \texttt{CTR}, \texttt{theCTR} is a macro that defines the printed version of counter \texttt{CTR}. It is defined in terms of the following macros:

- \texttt{\arabic{COUNTER}} prints the value of \texttt{COUNTER} as an arabic numeral.
- \texttt{\roman{COUNTER}} prints the value of \texttt{COUNTER} as a lowercase roman numeral.
- \texttt{\Roman{COUNTER}} prints the value of \texttt{COUNTER} as an uppercase roman numeral.
- \texttt{\alph{COUNTER}} prints the value of \texttt{COUNTER} as a lowercase letter: 1 = a, 2 = b, etc.
- \texttt{\Alph{COUNTER}} prints the value of \texttt{COUNTER} as an uppercase letter: 1 = A, 2 = B, etc.

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

\renewcommand*{\thepart}{\@Roman\c@part}
\renewcommand{\thesection}{\@arabic\c@section}
\renewcommand*{\thechapter}{\@arabic\c@chapter}
\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\texttt{@chapapp} is initially defined to be `\texttt{chaptername}`. The \texttt{appendix} command redefines it to be `\texttt{appendixname}`.

8.2.4 Front Matter, Main Matter, and Back Matter

A \texttt{boek} contains these three sections. First, we define the switch \texttt{@mainmatter} that is true iff we are processing Main Matter. When this switch is false, the \texttt{chapter} command does not print chapter numbers.

Here we define the commands that start these sections.

\frontmatter\texttt{frontmatter} This command starts Roman page numbering and turns off chapter numbering.
\mainmatter This command clears the page, starts arabic page numbering and turns on chapter numbering.
\newcommand*{\mainmatter}{\cleardoublepage \@mainmattertrue \pagenumbering{arabic}}

\backmatter This clears the page, turns off chapter numbering and leaves page numbering unchanged.
\newcommand*{\backmatter}{\if@openright\cleardoublepage\else\clearpage\fi \@mainmatterfalse}

8.2.5 Parts
\part The command to start a new part of our document.
In the artikel classes the definition of \part is rather simple; we start a new paragraph, add a little white space, suppress the indentation of the first paragraph (not for the artikel2 document class) and make use of \@secdef.
\newcommand*{\part}{\if@noskipsec \leavevmode \fi \par \addvspace{4ex} \@afterindentfalse \@afterindenttrue \secdef\@part\@spart}

For the rapport and boek classes we things a bit different.
We start a new (righthand) page and use the empty pagestyle.
\newcommand*{\part}{\cleardoublepage \thispagestyle{empty} \if@twocolumn \onecolumn \@tempswatrue \else \@tempswafalse \fi

When we are making a two column document, this will be a one column page. We use \@tempswa to remember to switch back to two columns.
\if@twocolumn \onecolumn \@tempswatrue \else \@tempswafalse \fi

We need an empty box to prevent the fil glue from disappearing.
\null\vfil
Here we use `\secdef` to indicate which commands to use to make the actual heading.

```latex
\secdef\@part\@spart}
```

\@part

This macro does the actual formatting of the title of the part. Again the macro is differently defined for the artikel document classes than for the document classes rapport and boek.

\PartFont

The font used to typeset the part is stored in this macro.

```latex
\newcommand*\PartFont{\bfseries}
```

When `secnumdepth` is larger than \(-1\) for the artikel document classes, we have a numbered part, otherwise it is unnumbered.

```latex
⟨∗artikel⟩\def\@part[#1]{%\ifnum\c@secnumdepth >\m@ne\refstepcounter{part}\addcontentsline{toc}{part}{\protect\numberline{\thepart}#1}\else\addcontentsline{toc}{part}{#1}\fi\head@style
\parindent\unitindent\normalfont
\ifnum\c@secnumdepth >\m@ne\Large\PartFont\noindent\partname\nobreakspace\thepart\fi\Large\PartFont\noindent#2\%\Large\PartFont#2\%
\markboth{}{}
\nobreak
\vskip 3ex\@afterheading}
```

When `secnumdepth` is larger than \(-2\) for the document class rapport and boek, we have a numbered part, otherwise it is unnumbered.

```latex
⟨∗rapport|boek⟩\markboth{}{\par}
\nobreak
\vskip 3ex\@afterheading}
```

When `secnumdepth` is larger than \(-2\) for the document class rapport and boek, we have a numbered part, otherwise it is unnumbered.
We empty the mark registers and center the title on the page in the rapport and boek document classes. Also we prevent breaking between lines and reset the font.

When this is a numbered part we have to print the number. We have to expand \partname before \uppercase is called, therefore we use a temporary control sequence that, when called will execute \MakeUppercase on the contents of \partname.

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

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

The differences between the definition of this macro in the artikel document classes and in the rapport and boek document classes are similar as they were for \@part.
858 \def\@part#1{% 859 \centering 860 \interlinepenalty \@M 861 \normalfont 862 \Large \PartFont #1\par} 863 \@endpart} 864 \@endpart

\@endpart This macro finishes the part page, for both \part and \spart.
First we fill the current page.

865 \def\@endpart{\vfil\newpage
Then, when we are in twosided mode and chapters are supposed to be on right
hand sides, we produce a completely blank page.

866 ⟨!boek⟩\if@twoside
867 \null\thispagestyle{empty}\newpage
868 ⟨!boek⟩\fi
When this was a two column document we have to switch back to two column
mode.
873 \if@tempswa
874 \twocolumn
875 \fi}
876 \@chapter This macro is called when we have a numbered chapter. When secnumdepth is
larger than −1 and, in the boek class, \@mainmatter is true, we display the chapter

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

877 \def\@chapter{% 878 \newcommand\chapter{%\if@openright\cleardoublepage\else\clearpage\fi
879 \thispagestyle{plain}\global\@topnum\z@
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.
880 \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.
881 \@afterindentfalse
882 \secdef\@chapter\@schapter}
\@chapter This macro is called when we have a numbered chapter. When secnumdepth is
larger than −1 and, in the boek class, \@mainmatter is true, we display the chapter
number. We also inform the user that a new chapter is about to be typeset by writing a message to the terminal.

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

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

\ChapFont The font used to typeset the chapters is stored in this macro.

\@makechapterhead The macro above uses \@makechapterhead{text} to format the heading of the chapter.

We begin by leaving some white space. The we open a group in which we have a paragraph indent of 0pt, and in which we have the text set ragged right. We also reset the font.
Then we check whether the number of the chapter has to be printed. If so we leave some whitespace between the chapter number and its title.

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

\@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.

\@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.

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

\section This gives a normal heading with white space above the heading (the whitespace below the heading will be generated by the \parskip that is inserted at the start
of the first paragraph), the title set in $\texttt{large}\texttt{bfseries}$, and no indentation on
the first paragraph.

943 $\texttt{\newcommand*{\section}{%}$
944 $\langle \ast \text{type1} \mid \text{type3} \rangle$
945 $\texttt{\@startsection {section}{1}{\z@} \{ -2 \text{baselineskip} \plus -1 \text{baselineskip} \ominus .5 \text{baselineskip} \} \{/\text{type1} | \text{type3}\}$
946 $\langle \ast \text{type2} \rangle$
947 $\texttt{\@startsection {section}{1}{\unitindent} \{ 2 \text{baselineskip} \plus \text{baselineskip} \ominus .5 \text{baselineskip} \} \{/\text{type2}\}$
948 $\langle \ast \text{type2} \rangle$
949 $\langle \text{type1} \rangle$
950 $\langle \text{type2} \mid \text{type3} \rangle$
951 $\{ .5 \text{baselineskip} \}$
952 $\langle \text{type2} \mid \text{type3} \rangle$
953 $\{ .01 \text{baselineskip} \}$
954 $\langle \{ \text{normalfont} \text{large} \texttt{\SectFont}\} \rangle$

$\texttt{\SectFont}$ The font used to typeset the sections is stored in this maro.

955 $\texttt{\newcommand*{\SectFont}{\bfseries}}$

$\texttt{\subsection}$ This gives a normal heading with white space above the heading, the title set in
$\texttt{\normalsize}\texttt{bfseries}$, and no indentation on the first paragraph.

956 $\texttt{\newcommand*{\subsection}{%}$
957 $\langle \ast \text{type1} \mid \text{type3} \rangle$
958 $\texttt{\@startsection {subsection}{2}{\z@} \{ -1 \text{baselineskip} \ominus .5 \text{baselineskip} \} \{/\text{type1} | \text{type3}\}$
959 $\langle \ast \text{type2} \rangle$
960 $\texttt{\@startsection {subsection}{2}{\unitindent} \{ 1 \text{baselineskip} \ominus .5 \text{baselineskip} \} \{/\text{type2}\}$
961 $\langle \ast \text{type2} \rangle$
962 $\langle \text{type1} \rangle$
963 $\langle \text{type2} \mid \text{type3} \rangle$
964 $\{ .25 \text{baselineskip} \}$
965 $\langle \text{type2} \mid \text{type3} \rangle$
966 $\{ .01 \text{baselineskip} \}$
967 $\langle \{ \text{normalfont} \text{normalsize} \texttt{\SSectFont}\} \rangle$

$\texttt{\SSectFont}$ The font used to typeset the subsections is stored in this maro.

968 $\texttt{\newcommand*{\SSectFont}{\bfseries}}$

$\texttt{\subsubsection}$ This gives a normal heading with white space above the heading, the title set in
$\texttt{\normalsize}\texttt{tm}$, and no indentation on the first paragraph.

969 $\texttt{\newcommand*{\subsubsection}{%}$
970 $\langle \ast \text{type1} \mid \text{type3} \rangle$
971 $\texttt{\@startsection {subsubsection}{3}{\z@} \{ -1 \text{baselineskip} \plus -.5 \text{baselineskip} \min -.25 \text{baselineskip} \} \{/\text{type1} | \text{type3}\}$
972 $\langle \ast \text{type2} \rangle$
973 $\texttt{\@startsection {subsubsection}{3}{\unitindent} \{ 1 \text{baselineskip} \plus .5 \text{baselineskip} \min -.25 \text{baselineskip} \} \{/\text{type2}\}$
974 $\langle \ast \text{type2} \rangle$
975 $\langle \text{type1} \rangle$
976 $\langle \text{type2} \mid \text{type3} \rangle$
977 $\{ .25 \text{baselineskip} \}$
978 $\langle \text{type2} \mid \text{type3} \rangle$
979 $\{ .01 \text{baselineskip} \}$
980 $\langle \{ \text{normalfont} \text{normalsize} \texttt{\SSSectFont}\} \rangle$

42
The font used to typeset the subsubsections is stored in this macro.
\newcommand*{\SSSectFont}{\rmfamily}
\newcommand*{\SectFont}{\sffamily}
\newcommand*{\SSectFont}{\slshape}

This gives a run-in heading with white space above and to the right of the heading, the title set in \normalsize\slshape.
\newcommand{\paragraph}{\@startsection{paragraph}{4}{0}{3.25ex plus 1ex minus .2ex}{-1em}{\normalfont\normalsize\ParaFont}}
\newcommand{\ParaFont}{\slshape}
\newcommand{\ParaFont}{\scshape}

This gives an indented run-in heading with white space above and to the right of the heading, the title set in \normalsize\slshape.
\newcommand{\subparagraph}{\@startsection{subparagraph}{5}{0}{3.25ex plus 1ex minus .2ex}{-1em}{\normalfont\normalsize\SParaFont}}
\newcommand{\SParaFont}{\slshape}

To change the fonts that are used to typeset the title, part, chapter and section headings this macro can be used.
\newcommand*{\HeadingFonts}[7]{\renewcommand*{\TitleFont}{#1}\renewcommand*{\PartFont}{#2}\renewcommand*{\ChapFont}{#3}\renewcommand*{\SectFont}{#4}\renewcommand*{\SSectFont}{#5}\renewcommand*{\SSSectFont}{#6}\renewcommand*{\ParaFont}{#7}}
8.3 Lists

8.3.1 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 \textit{Kth} level list, the command \texttt{@listK} is called, where ‘K’ denotes ‘i’, ‘ii’, ... , ‘vi’. (I.e., \texttt{@listiii} is called for a third-level list.) By convention, \texttt{@listK} should set \texttt{leftmargin} to \texttt{leftmarginK}.

\begin{verbatim}
\leftmargin \leftmargini \leftmarginii \leftmarginiii \leftmarginiv \leftmarginv \leftmarginvi
\end{verbatim}

For efficiency, level-one list’s values are defined at top level, and \texttt{@listi} is defined to set only \texttt{leftmargin}.

\begin{verbatim}
\leftmargini \setlength{\leftmargini}{\unitindent}
\leftmarginii \setlength{\leftmarginii}{\othermargin}
\leftmarginiii \setlength{\leftmarginiii}{\othermargin}
\leftmarginiv \setlength{\leftmarginiv}{\othermargin}
\leftmarginv \setlength{\leftmarginv}{\othermargin}
\leftmarginvi \setlength{\leftmarginvi}{1em}
\end{verbatim}

Here we set the top level \texttt{leftmargin}.

\begin{verbatim}
\labelsep \labelwidth
\end{verbatim}

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

\begin{verbatim}
\labelsep \setlength{\labelsep}{5\p@}
\labelwidth \setlength{\labelwidth}{\leftmargini}
\end{verbatim}

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

\begin{verbatim}
\partopsep \setlength{\partopsep}{\z@}
\end{verbatim}

\texttt{topsep} Extra vertical space, in addition to \texttt{parskip}, added above and below list and paragraphing environments.

\begin{verbatim}
\topsep \setlength{\topsep}{\z@}
\end{verbatim}

\texttt{@beginparpenalty} These penalties are inserted before and after a list or paragraph environment. \texttt{@endparpenalty} They are set to a bonus value to encourage page breaking at these points.
\@itempenalty This penalty is inserted between list items.
\@beginparpenalty -\@lowpenalty
\@endparpenalty -\@lowpenalty
\@itempenalty -\@lowpenalty
\@listi \@listi defines values of \leftmargin, \parsep, \topsep, and \itemsep, etc. 
\@listI for the lists that appear on top-level. Its definition is modified by the font-size
commands (eg within \texttt{small} the list parameters get “smaller” values).
For this reason listI is defined to hold a saved copy of listi so that \texttt{\normalsize}
can switch all parameters back.
\def\@listi{%
\langle type2 \rangle \leftmargin \unitindent
\langle type2 \rangle \leftmargin \leftmargini
\langle type2 \rangle \labelsep .5em
\langle type2 \rangle \labelsep .45em
\labelwidth \leftmargin
\advance \labelwidth \- \labelsep
\langle type3 \rangle \topsep \z@ \@plus \p@ 
\langle type3 \rangle \topsep \- .5 \parskip \@plus \p@ 
\parsep \z@ 
\itemsep \z@ \@plus \p@}
\let\@listI\@listi
We initialise these parameters although strictly speaking that is not necessary.
\def\@listii{%
\langle type2 \rangle \leftmargini \leftmarginii
\langle type2 \rangle \labelsep .5em
\langle type2 \rangle \labelsep .3em
\labelwidth \leftmarginii
\advance \labelwidth \- \labelsep
\langle type3 \rangle \topsep \z@ \@plus \p@ 
\langle type3 \rangle \topsep \- .5 \parskip \@plus \p@ 
\parindent \z@ 
\partopsep \z@ \@plus \p@ 
\itemsep \z@ \@plus \p@}
\def\@listiii{%
\langle type2 \rangle \labelsep .5em
\langle type2 \rangle \labelsep .3em
\labelwidth \leftmarginiii
\advance \labelwidth \- \labelsep
\langle type3 \rangle \topsep \z@ \@plus \p@ 
\langle type3 \rangle \topsep \- .5 \parskip \@plus \p@ 
\parindent \z@ 
\partopsep \z@ \@plus \p@ 
\itemsep \z@ \@plus \p@}
\def\@listiv{%
\langle type2 \rangle \labelsep .5em
\langle type2 \rangle \labelsep .3em
\labelwidth \leftmarginiv
\advance \labelwidth \- \labelsep
\langle type3 \rangle \topsep \z@ \@plus \p@ 
\langle type3 \rangle \topsep \- .5 \parskip \@plus \p@ 
\parindent \z@ 
\partopsep \z@ \@plus \p@ 
\itemsep \z@ \@plus \p@}
\def\@listv{%
\langle type2 \rangle \labelsep .5em
\langle type2 \rangle \labelsep .3em
\labelwidth \leftmarginv
\advance \labelwidth \- \labelsep
\langle type3 \rangle \topsep \z@ \@plus \p@ 
\langle type3 \rangle \topsep \- .5 \parskip \@plus \p@ 
\parindent \z@ 
\partopsep \z@ \@plus \p@ 
\itemsep \z@ \@plus \p@}
\def\@listvi{%
\langle type2 \rangle \labelsep .5em
\langle type2 \rangle \labelsep .3em
\labelwidth \leftmarginvi
\advance \labelwidth \- \labelsep
\langle type3 \rangle \topsep \z@ \@plus \p@ 
\langle type3 \rangle \topsep \- .5 \parskip \@plus \p@ 
\parindent \z@ 
\partopsep \z@ \@plus \p@ 
\itemsep \z@ \@plus \p@}
8.3.2 Enumerate

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

\begin{verbatim}
\renewcommand*{\theenumi}{\@arabic{\c@enumi}}
\renewcommand*{\theenumii}{\@alph{\c@enumii}}
\renewcommand*{\theenumiii}{\@roman{\c@enumiii}}
\renewcommand*{\theenumiv}{\@Alph{\c@enumiv}}
\end{verbatim}

The label for each item is generated by the commands

\begin{verbatim}
\labelenumi The label for each item is generated by the commands
\labelenumii \labelenumiii \labelenumiv.
\end{verbatim}

\begin{verbatim}
\renewcommand*{\theenumi}{\@arabic{\c@enumi}}
\renewcommand*{\theenumii}{\@alph{\c@enumii}}
\renewcommand*{\theenumiii}{\@roman{\c@enumiii}}
\renewcommand*{\theenumiv}{\@Alph{\c@enumiv}}
\end{verbatim}

\begin{verbatim}
\p@enumi The expansion of \texttt{\p@enum\theenum} defines the output of a \texttt{\ref} command
\p@enumii when referencing an item of the Nth level of an enumerated list.
\p@enumiv
\end{verbatim}
We want to have different label positioning on different levels of list. To achieve this we have to redefine the `enumerate` environment.

We try to suppress spaces after these list constructs.

### 8.3.3 Itemize

Itemization is controlled by four commands: \textbullet, \textbf{\textendash}, \textasteriskcentered, and \textperiodcentered, which define the labels of the various itemization levels: the symbols used are bullet, bold en-dash, asterisk and centred dot.

The default definition for \textbullet is to reset the font to \textnormal so that always the same symbol is produced regardless of surrounding conditions.

A possible alternative would be
\renewcommand\labelitemfont{%
\fontseries\seriesdefault
\fontshape\shapedefault\selectfont}

which resets series and shape doesn’t touch the family.

\newcommand\labelitemfont\{\normalfont\}

itemize  We want to have different label positioning on different levels of list. To achieve this we have to redefine the itemize environment.

\renewenvironment\{itemize\}{%\ifnum\@itemdepth >\thr@@
\@toodeep
\else
\advance\@itemdepth\@ne
\edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%
\expandafter\list\csname\@itemitem\endcsname{\langle\parbox{\textwidth}{\textwidth}}
\ifnum\@listdepth=1\relax\langle∗\parbox{\textwidth}{\textwidth}\langle\textwidth}
\if@revlabel\def\makelabel\{\hskip .5\unitindent\hfil\textwidth}\else\langle\parbox{\textwidth}{\textwidth}\langle\parbox{\textwidth}{\textwidth}\fi
\langle\parbox{\textwidth}{\textwidth}\fi\fi}
\fi}

We try to suppress spaces after these list constructs.

\newenvironment\{description\}{\list{}{\labelwidth\z@ \itemindent-\leftmargin\let\makelabel\descriptionlabel}}\endlist
\descriptionlabel To change the formatting of the label, you must redefine \descriptionlabel.

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

8.3.4 Description

description The description environment is defined here – while the itemize and enumerate environments are defined in \texttt{latex.dtx}.

\renewenvironment\{description\}{\{\list{\{\labelwidth\z@ \itemindent-\leftmargin\let\makelabel\descriptionlabel\}}\endlist}
\descriptionlabel To change the formatting of the label, you must redefine \descriptionlabel.
8.4 Adapting existing environments

Because we globally set \topsep to zero, we need to modify the definitions of a number of environments slightly to get a little whitespace around them in the document classes artikel1 and rapport1.

\texttt{center} Add a little surrounding whitespace.
\begin{verbatim}
1166 (+type1)
1167 \def\center{
1168 \topsep=.25\baselineskip \@plus .1\baselineskip
1169 \@minus .1\baselineskip
1170 \trivlist \centering\item[]}
1171 \let\endcenter=\endtrivlist
\end{verbatim}

\texttt{flushleft} Add a little surrounding whitespace.
\begin{verbatim}
1172 \def\flushleft{
1173 \topsep=.25\baselineskip \@plus .1\baselineskip
1174 \@minus .1\baselineskip
1175 \trivlist \raggedright\item[]}
1176 \let\endflushleft=\endtrivlist
\end{verbatim}

\texttt{flushright} Add a little surrounding whitespace.
\begin{verbatim}
1177 \def\flushright{
1178 \topsep=.25\baselineskip \@plus .1\baselineskip
1179 \@minus .1\baselineskip
1180 \trivlist \raggedleft\item[]}
1181 \let\endflushright=\endtrivlist
1182 (/type1)
\end{verbatim}

\texttt{verbatim} In verbatim we add a little surrounding whitespace, –which for artikel3 and rapport3 is negative to compensate for the positive \parskip– but also an indent for the artikel1 and rapport1 document classes.
\begin{verbatim}
1183 \def\verbatim{%
1184 (+type1 | type2)
1185 \topsep=.25\baselineskip \@plus .1\baselineskip
1186 \@minus .1\baselineskip
1187 \@verbatim
1188 (/type1 | type2)
1189 (+type1) \leftskip\unitindent
1190 (+type2) \leftskip\z@
1191 (+type3)
1192 \topsep=-.5\parskip
1193 @verbatim
1194 (/type3)
1195 \frenchspacing\@obeyspaces @verbatim}
1196 (type1)\def\endverbatim{\if@newlist \leavevmode \fi\endtrivlist}
\end{verbatim}
8.5 Defining new environments

8.5.1 Abstract

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

Note that this environment is not defined for books.

When we are not making a separate titlepage—the default for the artikel document classes—we have to check if we are in twocolumn mode. In that case the abstract is set as a \section*, otherwise the abstract is typeset flushleft, an amount \unitindent smaller as the normal text.

As always, the artikel2 document class has a different implementation.

Which implies that the definition of \end{abstract} is also different.
8.5.2 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`\` equal `\@centercr`.

\newenvironment{verse}
\begin{list}{}
\itemsep\z@ \itemindent-1.5em\%
\itemindent\listparindent\rightmargin\leftmargin
\advance\leftmargin1.5em\%
\item\relax
\endlist

8.5.3 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}
\begin{list}{}
\itemindent\listparindent\rightmargin\leftmargin
\parsep\z@ \@plus\p@\%
\item\relax
\endlist

8.5.4 Quote

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

\newenvironment{quote}
\begin{list}{}
\item\relax
\endlist

8.5.5 Theorem

These document classes have a slightly modified theorem environment style. Surrounding whitespace is added and an initialisation of `\labelsep` is added. Finally a slanted font instead of an italic font is used.

\begin{verbatim}
def\begintheorem#1#2{\vskip\baselineskip \labelsep=.5em\%
\trivlist
def\@opargbegintheorem{\itemindent\listparindent\rightmargin\leftmargin
\parsep\z@ \@plus\p@\%
\item\relax
\endlist
\end{verbatim}
8.5.6 Titlepage

The titlepage environment does nothing but start and end a page, and inhibit page numbers. It also resets the page number to zero. This is incorrect since it results in using the page parameters for a right-hand page but it is the way it was. In two-column style, it still makes a one-column page.

8.5.7 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 artikel document classes the \appendix command must do the following:

- reset the section and subsection counters to zero,
- redefine \thesec tion to produce alphabetic appendix numbers.
In the rapport and boek document classes the \appendix command must do the following:

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

1289 (*rapport | boek)  
1290 \newcommand*\appendix{\par} \setcounter{chapter}{0}\%  
1291 \setcounter{section}{0}\%  
1292 \gdef\@chapapp{\appendixname}\%  
1293 \gdef\thechapter{\@Alph\c@chapter}\}  
1295 (/rapport | boek)

8.6 Setting parameters for existing environments

8.6.1 Array and tabular

\arraycolsep The columns in an array environment are separated by 2\arraycolsep.
1296 \setlength\arraycolsep(5\p@)

\tabcolsep The columns in tabular environment are separated by 2\tabcolsep.
1297 \setlength\tabcolsep(6\p@)

\arrayrulewidth The width of rules in the array and tabular environments is given \arrayrulewidth.
1298 \setlength\arrayrulewidth(0.4\p@)

\doublerulesep The space between adjacent rules in the array and tabular environments is given by \doublerulesep.
1299 \setlength\doublerulesep(2\p@)

8.6.2 Tabbing

\tabbingsep This controls the space that the \' command puts in. (See \TeX manual for an explanation.)
1300 \setlength\tabbingsep(\labelsep)

8.6.3 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.
1301 (\texttt{type1}) \texttt{\def\@minipagerestore{\parindent\unitindent}}
1302 (\texttt{type3})
\def\@minipagerestore{%
 \parskip=.5\baselineskip \@plus .1\baselineskip
 \@minus .1\baselineskip}
\langle \type3 \rangle
\@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

8.6.4 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@}

8.6.5 Equation and eqnarray
\theequation When within chapters, 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.
\renewcommand*{\theequation}{\@arabic\c@equation}
\@addtoreset{equation}{chapter}
\renewcommand*{\theequation}{\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@equation}
\langle \rapport | boek \rangle
\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{\langle \theequation \rangle}

8.7 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.
The file extension indicating the file on which the contents list for float type \texttt{TYPE} is stored. For example, \texttt{\textbackslash ext\@TYPE figure} = ‘lof’.

\texttt{\textbackslash fnum\@TYPE} A macro to generate the figure number for a caption. For example, \texttt{\textbackslash fnum\@TYPE} == ‘Figure \texttt{\textbackslash thefigure}’.

\texttt{\textbackslash @makecaption\langle num\rangle\langle text\rangle} A macro to make a caption, with \langle\texttt{num}\rangle the value produced by \texttt{\textbackslash fnum\@...} and \langle\texttt{text}\rangle the text of the caption. It can assume it’s in a \texttt{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 \texttt{\textbackslash @float} and \texttt{\textbackslash end\@float}, which are defined in \texttt{latex.dtx}.

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

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

8.7.1 Figure

Here is the implementation of the figure environment.

\texttt{\textbackslash c@figure} First we have to allocate a counter to number the figures. In the rapport and boek document classes the figures are numbered per chapter.

\begin{verbatim}
\c@figure
\newcounter{figure}
\renewcommand*{\thefigure}{\@arabic{\thefigure}}
\end{verbatim}

\begin{verbatim}
\fps@figure
\ftype@figure
\ext@figure
\num@figure
\end{verbatim}

Here are the parameters for the floating objects of type ‘figure’.

\begin{verbatim}
\fps\@figure
\ftype\@figure
\ext\@figure
\num\@figure
\end{verbatim}

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

\begin{verbatim}
\newenvironment{figure}
{\@float{figure}}
{\end@float}
\end{verbatim}

\begin{verbatim}
\newenvironment{figure*}
{\@dblfloat{figure}}
{\end@dblfloat}
\end{verbatim}
8.7.2 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 rapport and boek document classes the tables are numbered per chapter.

\newcounter{table}
\renewcommand*{\thetable}{\@arabic{\c@table}}
\newcounter{table}[chapter]
\renewcommand{\thetable}{% 
\ifnum\c@chapter>\z@ \thethechapter. \fi\@arabic{\c@table}}

\fps@table  Here are the parameters for the floating objects of type ‘table’.
\ftype@table{2}
\ext@table{lot}
\num@table{\tablename\nobreakspace\thetable}

table  And the definition of the actual environment. The form with the * is used for double column tables.
\newenvironment{table}{\@float{table}}{\end@float}
\newenvironment{table*}{\@dblfloat{table}}{\end@dblfloat}

8.7.3 Captions

\@makecaption  The \caption command calls \@makecaption to format the caption of floating objects. It gets two arguments, \texttt{\langle number\rangle}, the number of the floating object and \texttt{\langle text\rangle}, the text of the caption. Usually \texttt{\langle number\rangle} contains a string such as ‘Figure 3.2’. The macro can assume it is called inside a \parbox of right width, with \normalshape.

\abovecaptionskip  These lengths contain the amount of white space to leave above and below the caption.
\belowcaptionskip
The definition of this macro is \long in order to allow more than 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{{\CaptionLabelFont#1:} \CaptionTextFont#2}\
We can measure its width. If that is larger than the current \hsize we typeset the caption as an ordinary paragraph.
\ifdim \wd\@tempboxa >\hsize
\CaptionLabelFont#1: \CaptionTextFont#2\par
If the caption fits, we center it. Because this uses an \hbox directly in vertical mode, it does not execute the \everypar tokens; the only thing that could be needed here is resetting the ‘minipage flag’ so we do this explicitly.
\else
\global \@minipagefalse
\hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
\fi
\vskip\belowcaptionskip}

\CaptionLabelFont These macros can contain the fonts used for typesetting captions. By default they do nothing.
\newcommand*\CaptionLabelFont{\relax}
\newcommand*\CaptionTextFont{\relax}

\CaptionFonts To change the fonts that are used to typeset captions this macro can be used.
\newcommand*\CaptionFonts[2]{%
\renewcommand*\CaptionLabelFont{#1}%
\renewcommand*\CaptionTextFont{#2}%
}

8.8 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 \DeclareOldFontCommand, 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 The commands to change the family. When in compatibility mode we select the ‘default’ font first, to get \LaTeXX2.09 behviour.
\tt
\DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\DeclareOldFontCommand{\sf}{\normalfont\sfamily}{\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.

\s  And the commands to change the shape of the font. The slanted and small caps shapes are not available by default as math alphabets, so those changes do nothing in math mode. One should use \upshape to explicitly change back to the upright shape.

\cal 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’.

\em The definition of \em is changed here to have slanted instead of italic fonts.

9 Cross Referencing

9.1 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 either the \@dottedtocline or the \@regtocline command, which work as follows.
An entry is produced only if \( \text{level} \leq \text{value of the tocdepth counter} \).

Note, \texttt{\textbackslash chapter} is level 0, \texttt{\textbackslash section} is level 1, etc.

The indentation from the outer left margin of the start of the contents line.

The width of a box in which the section number is to go, if \texttt{\textbackslash title} includes a \texttt{\textbackslash numberline} command.

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

The width of a box in which the page number is put.

The right margin for multiple line entries. One wants \( \texttt{\@tocrmarg} \geq \texttt{\@pnumwidth} \).

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

\begin{verbatim}
\newcommand*{\@pnumwidth}{1.55em}
\newcommand*{\@tocrmarg}{2.55em}
\newcommand*{\@dotsep}{4.5}
\setcounter{tocdepth}{3}
\setcounter{tocdepth}{2}
\end{verbatim}

\subsection{Table of Contents}

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

\begin{verbatim}
\newcommand*{\tableofcontents}{\if@twocolumn\@restonecoltrue\onecolumn\else\@restonecolfalse\fi
\chapter*{\contentsname}\
\ifartikel\setcounter{tocdepth}{3}\else\setcounter{tocdepth}{2}\fi
}
\end{verbatim}

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

\begin{verbatim}
\chapter*{\contentsname}
\end{verbatim}

\section{\contentsname}
The code for \@mkboth is placed inside the heading to avoid any influence on vertical spacing after the heading (in some cases). For other commands, such as \listoffigures below this has been changed from the \LaTeX2.09 version as it will produce a serious bug if used in two-column mode (see, \LaTeX{} pt/3285). However \tableofcontents is always typeset in one-column mode in these classes, therefore the somewhat inconsistent setting has been retained for compatibility reasons.

\begin{verbatim}
\@mkboth{\MakeUppercase{\contentsname}}{\MakeUppercase{\contentsname}}
\end{verbatim}

The actual table of contents is made by calling \@starttoc{toc}. After that we restore two-column mode if necessary.

\begin{verbatim}
\@starttoc{toc}
\end{verbatim}

\@starttoc The internal \LaTeX{} 2\epsilon macro \@starttoc needs to be adapted for the \artikel and \rapport document classes, in order to deal with the fact that for these document classes the \parskip is normally non-zero. We don’t want that in the table of contents.

\begin{verbatim}
\@starttoc#1\{\begingroup
\makeatletter
\parskip\z@\@input{\jobname.#1}
\if@filesw
\expandafter\newwrite\csname tf@#1\endcsname
\immediate\openout\csname tf@#1\endcsname\jobname.#1\relax
\fi\@nobreakfalse\endgroup
\end{verbatim}

\@regtocline These document classes use a different format for the table of contents than the standard classes from which they were developped. In order to achieve this different format we defined the macro \@regtocline.

\begin{verbatim}
\newcommand\@regtocline[3]{
\ifnum#1>\c@tocdepth
\else
\vskip\z@\@plus.2\p@
\hangindent\z@ \@afterindenttrue \interlinepenalty\@M
\leftskip\unitindent \rightskip\unitindent\@plus 1fil
\parfillskip\z@
\leavevmode
\hbox{\hskip -\leftskip}
\ifnum#1<0 \textbf{#2}\else
60
\fi\nobreakfalse \endgroup
\end{verbatim}
\numberline This internal macro is redefined for the \texttt{article2} document class.
\toc@font The changed definition of \texttt{@sect} that we use, selects a different font for the table of contents for the various header levels. It does this using \texttt{\toc@font}.
\toc@fontsel

A line of the table of contents contains \texttt{\numberline} and the section number as its first two elements. We don't want to set the section number using \texttt{\toc@font}, therefor we give it two additional arguments and pass them on first, \textit{before} changing the font. Note that we need to re-insert the braces around the second argument.

\begin{verbatim}
\numberline
\texttt{\\numberline}
\end{verbatim}

When the user wants to produce a hyper-document using \texttt{hyperref} we need to take special precautions to make it work for the table of contents. We check for the existence of \texttt{\hyper@linkstart} to detect this situation at \texttt{\begin{document}}.

\texttt{Hyperref} injects extra tokens (\texttt{\hyper@linkstart\{link\}\texttt{Hy@tocdestname}}) into the stream in front of the real contentsline. The command \texttt{\hyper@linkstart} and its arguments need to be protected from expanding too early or being “uppercased” themselves.

\begin{verbatim}
\AtBeginDocument{%
\end{verbatim}

In the contentslines for chapters, sections etc., the command selection of the appropriate font needs to come after the code that \texttt{hyperref} injects. we do this with some argument shuffling.
\let\ORG@hyper@linkstart\hyper@linkstart
\protected\def\hyper@linkstart#1#2{%\lowercase{\ORG@hyper@linkstart{#1}{#2}}}%
\fi}
\if\fi
\toc@case

In the \texttt{rapport} and \texttt{boek} document classes, the entries for parts are typeset in capital letters in the new style of the table of contents. In the old style this isn’t done. The macro \texttt{\toc@case} is used to switch this.

\if\if@oldtoc
\newcommand*{\toc@case}{\relax}
\else
\newcommand*{\toc@case}{\MakeUppercase}
\fi
\fi
\l@part

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

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

First we have the definition from the standard classes.

\if\if@oldtoc
\newcommand*{\l@part}[2]{%\ifnum\c@tocdepth>-2\relax
\langle artikel \rangle \addpenalty\@secpenalty
\langle !artikel \rangle \addpenalty{-\@highpenalty}\
\addvspace{2.25em \@plus\p@}\
\begingroup
The macro \texttt{\numberline} requires that the width of the box that holds the part number is stored in LaTeX’s scratch register \texttt{\@tempdima}. Therefore we put it there.

\setlength{\@tempdima}{3em}%

The we set \texttt{\parindent} to 0pt and use \texttt{\rightskip} to leave enough room for the pagenumbers. To prevent overfull box messages the \texttt{\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 pagernumber, set flush right.

{\leavevmode
\large \bfseries #1\hfil \bb@xt\@pnumwidth{\hss #2}\
\kern-\p@\kern\p@}\par

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

\nobreak

62
Then we can introduce our new definition.

\l@chapter

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

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

Again we first present the ‘standard’ definition

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.

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

Because we do not use \numberline here, we have do some fine tuning ‘by hand’, before we can set the entry. We discourage but not disallow a pagebreak immediately after a chapter entry.
Then we present our new definition.
\begin{verbatim}
\newcommand*\l@chapter{\@regtocline{0}}
\end{verbatim}
\newcommand*\l@section{\@dottedtocline{1}{1.5em}{2.3em}}
\newcommand*\l@article{\@regtocline{1}}
\if@oldtoc
\newcommand*\l@article{\@regtocline{1}}
\fi
\newcommand*\l@article{\@dottedtocline{1}{1.5em}{2.3em}}

\l@section In the artikel document classes the entry in the table of contents for sections looks much like the chapter entries for the rapport and boek document classes.

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

\begin{verbatim}
\setlength{\@tempdima}{1.5em}
\begingroup
\parindent \z@ \rightskip \@pnumwidth
\parfillskip -\@pnumwidth
\leavevmode \bfseries
\end{verbatim}

The new definition:
\begin{verbatim}
\newcommand*\l@article{\@regtocline{1}}
\end{verbatim}
\newcommand*\l@article{\@dottedtocline{1}{1.5em}{2.3em}}

In the rapport and boek document classes the definition for \l@section is much simpler.
All lower level entries are defined using the macro \dottedtocline or \regtocline (see above).

\subsection{List of figures}

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

\listoffigures

This macro produces an entry in the list of figures.

\figure
9.1.3 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}{% 
\if@twocolumn
   \@restonecoltrue\onecolumn
\else
   \@restonecolfalse
\fi
\chapter*{\listtablename}\
\section*{\listtablename}
\@mkboth{\MakeUppercase{\listtablename}}{\MakeUppercase{\listtablename}}
\@starttoc{lot}

\let\l@table\l@figure

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

9.2 Bibliography

\bibindent  The “open” bibliography format uses an indentation of \bibindent.
\newdimen\bibindent
\setlength{\bibindent}{1.5em}
\newblock This is a dummy definition for this macro which is used in the \thebibliography environment.

\newenvironment{thebibliography}[1][1]{% 
\section*{\refname}
\@mkboth{\MakeUppercase\refname}{\MakeUppercase\refname}
\let\makelabel\bfseries
\let\l@table\l@figure

\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`\relax=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 \emph{enumiv} counter internally to generate the labels of the list.

When an empty ‘thebibliography’ environment is found, a warning is issued.
The default definition for \newblock is to produce a small space.

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

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

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

9.3 The index

The environment ‘theindex’ can be used for indices. It makes an index with two columns, with each entry a separate paragraph. At the user level the commands \item, \subitem and \subsubitem are used to produce index entries of various levels. When a new letter of the alphabet is encountered an amount of \indexspace white space can be added.
Parameter changes to \texttt{\textbackslash columnseprule} and \texttt{\textbackslash columnsep} have to be done after \texttt{\textbackslash twocolumn} has acted. Otherwise they can affect the last page before the index.

\texttt{\textbackslash columnseprule \textbackslash Z@}
\texttt{\textbackslash columnsep 35\textbackslash p@}
\texttt{\textbackslash parskip\textbackslash Z@ \textbackslash plus .3\textbackslash p@\relax}
\texttt{\textbackslash let\textbackslash item\textbackslash \textbackslash id	extbackslash xitem}
\texttt{}%  
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.
\texttt{\textbackslash if\textbackslash@restonecol\textbackslash onecolumn\textbackslash else\textbackslash cleardouble\textbackslash fi}

\texttt{\textbackslash \textbackslash id	extbackslash xitem}
\texttt{\textbackslash subitem}
\texttt{\textbackslash subsubitem}
\texttt{\textbackslash newcommand\textbackslash \textbackslash id	extbackslash xitem \{\textbackslash par\textbackslash hangindent 40\textbackslash p@}\}}
\texttt{\textbackslash newcommand\textbackslash \textbackslash subitem \{\textbackslash \textbackslash id	extbackslash xitem\textbackslash hspace\{20\textbackslash p@\}\}}
\texttt{\textbackslash newcommand\textbackslash \textbackslash subsubitem\textbackslash \textbackslash id	extbackslash xitem\textbackslash hspace\{30\textbackslash p@\}\}}

\texttt{\textbackslash indexspace}
The amount of white space that is inserted between ‘letter blocks’ in the index.
\texttt{\textbackslash newcommand\textbackslash \textbackslash indexspace\{\textbackslash par\textbackslash vskip10\textbackslash p@\textbackslash plus5\textbackslash p@\textbackslash minus3\textbackslash p@\textbackslash relax\}}

\section*{9.4 Footnotes}

\texttt{\textbackslash footnoterule}

Usually, footnotes are separated from the main body of the text by a small rule. This rule is drawn by the macro \texttt{\textbackslash footnoterule}. The standard \LaTeX{} document classes make sure that the rule takes no vertical space (see \texttt{plain.tex}) and compensate for the natural height of the rule of 0.4pt by adding the right amount of vertical skip. For the \texttt{artikel2} document class this is still true, but for the others the amount of whitespace between the last line of the text and the start of the footnotes is increased by giving \texttt{\textbackslash footnoterule} a positive height\footnote{This should perhaps have been done by increasing the value of \texttt{\textbackslash skip\footins}, but changing that now would mean changing the formatting of existing documents. (JLB, 08/09/1997)}.

To prevent the rule from colliding with the footnote we first add a little negative vertical skip, then we put the rule and add some positive vertical skip.

\texttt{\textbackslash renewcommand\textbackslash \textbackslash footnoterule\%}
\texttt{\textbackslash kern\textbackslash -3\textbackslash p@\}}
\texttt{(\textbackslash type1|\textbackslash type3)}
\texttt{\textbackslash kern\textbackslash .5\textbackslash baseline\space\textbackslash skip}
\texttt{\textbackslash hrule\textbackslash @width\textbackslash unit\textbackslash indent}
\texttt{\textbackslash kern\textbackslash .4\textbackslash baseline\space\textbackslash skip}
\texttt{(/\textbackslash type1|\textbackslash type3)}
\texttt{\textbackslash (\textbackslash type2)}
\texttt{\textbackslash hrule\textbackslash @width 3\textbackslash unit\textbackslash indent}
\texttt{\textbackslash kern\textbackslash 2.6\textbackslash p@\}}
Footnotes are numbered within chapters in the rapport and boek document styles.

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.

\begin{verbatim}
\long\def\@xmakefntext#1#2{\parindent=.5\unitindent
%<!type3> \parindent=.5\unitindent
%<type3> \parindent=.5\baselineskip\labelitemi{--}\@revlabeltrue
\setbox0=\hbox {#1\hskip.5em plus 1fil}%
\dimen0=2\wd0
\ifdim\dimen0>\unitindent
\global\unitindent=\dimen0
\indentset
\fi%
\@setpar\@@par
\@tempdima \hsize
\advance\@tempdima-.5\unitindent
\parshape \one .5\unitindent \@tempdima)%
\par
\noindent\llap{\hb@xt@.5\unitindent(#1\hfil)}#2}
\end{verbatim}

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 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. We change the label of itemized lists inside footnotes and need to check that the \unitindent is large enough for our purposes.

For most of the document classes produced from this file we need a slightly modified \makefntext on the title page, so we introduce an extra macro, \xmakefntext.

\begin{verbatim}
\newcommand*\xmakefntext[1]{\normalfont\thefnmark.}
\end{verbatim}
For the \texttt{article2} document class we have a simpler definition of \texttt{@makefntext}.

\[
\begin{align*}
\texttt{\@makefnmark} & \quad \text{The footnote markers that are printed in the text to point to the footnotes should be produced by the macro \texttt{@makefnmark}. We use the default definition for it.} \\
\texttt{\renewcommand\@makefnmark{\hbox{\@textsuperscript{\normalfont\@thefnmark}}}} 
\end{align*}
\]

\section{Initialization}

\subsection{Words}

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

\begin{align*}
\texttt{\newcommand*{\contentsname}{Contents}} \\
\texttt{\newcommand*{\listfigurename}{List of Figures}} \\
\texttt{\newcommand*{\listtablename}{List of Tables}} \\
\texttt{\newcommand*{\refname}{References}} \\
\texttt{\newcommand*{\bibname}{Bibliography}} \\
\texttt{\newcommand*{\indexname}{Index}} \\
\texttt{\newcommand*{\figurename}{Figure}} \\
\texttt{\newcommand*{\tablename}{Table}} \\
\texttt{\newcommand*{\partname}{Part}} \\
\texttt{\newcommand*{\chaptername}{Chapter}} \\
\texttt{\newcommand*{\appendixname}{Appendix}} \\
\texttt{\newcommand*{\abstractname}{Abstract}} \\
\texttt{\newcommand*{\seename}{see}} \\
\texttt{\newcommand*{\andname}{and}} \\
\end{align*}
\section{Date}

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

\begin{verbatim}
\newcommand*{\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}
\end{verbatim}

\section{Two column mode}
\cmd{\columnsep} This gives the distance between two columns in two column mode.

\begin{verbatim}
\setlength{\columnsep}{10\p@}
\end{verbatim}

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

\begin{verbatim}
\setlength{\columnseprule}{0\p@}
\end{verbatim}

\section{The page style}

We have \emph{plain} pages in the document classes \texttt{artikel} and \texttt{rapport} unless the user specified otherwise. In the \texttt{boek} document class we use the page style \emph{headings} by default. We use arabic pagenumbers.

\begin{verbatim}
⟨\texttt{boek}⟩ \pagestyle{plain}
⟨\texttt{boek}⟩ \pagestyle{headings}
\pagenumbering{arabic} % Arabic page numbers
\end{verbatim}

\section{Single or double sided printing}

When the \texttt{twoside} option wasn’t specified, we don’t try to make each page as long as all the others.

\begin{verbatim}
⟨∗\texttt{artikel}⟩
\if@twoside
\else
\raggedbottom
\fi
⟨/\texttt{artikel}⟩
\end{verbatim}

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

\begin{verbatim}
\if@twocolumn
\end{verbatim}
Normally we call \texttt{onecolumn} to initiate typesetting in one column.

\frenchspacing Controls the amount of space after a punctuation mark.

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

\begin{longtable}{|l|c|c|}
\hline
\texttt{Symbols} & \texttt{\@dblfttop} & \texttt{\@idxitem} \\
\hline
\texttt{\@Roman} & \ldots \ldots & 755 \\
\texttt{\@dotsep} & \ldots & 1392 \\
\texttt{\@dottedtocline} & \ldots & 1544, 1551--1554, 1557--1560, 1584 \\
\hline
\texttt{\@author} & \ldots & 528, 559, 561, 579, 631, 651, 663 \\
\texttt{\@chapter} & \ldots & 882, 883 \\
\texttt{\@cite} & \ldots & 882, 883 \\
\texttt{\@chapter} & \ldots & 882, 883 \\
\texttt{\@cite} & \ldots & 882, 883 \\
\hline
\texttt{\@listI} & \ldots & 102, 1036 \\
\texttt{\@listdepth} & \ldots & 1116, 1146 \\
\texttt{\@listii} & \ldots & 1049 \\
\texttt{\@listiii} & \ldots & 1049 \\
\texttt{\@listiv} & \ldots & 1049 \\
\texttt{\@listv} & \ldots & 1049 \\
\texttt{\@listvi} & \ldots & 1049 \\
\texttt{\@makecaption} & \ldots & 1616, 1617, 1633 \\
\texttt{\@makechapterhead} & \ldots & 1634 \\
\texttt{\@makefnmark} & \ldots & 1687 \\
\texttt{\@makefntext} & \ldots & 1687 \\
\texttt{\@makeindex} & \ldots & 1687 \\
\texttt{\@makecaption} & \ldots & 1687 \\
\texttt{\@makecaption} & \ldots & 1687 \\
\texttt{\@makecaption} & \ldots & 1687 \\
\texttt{\@makecaption} & \ldots & 1687 \\
\hline
\end{longtable}