Rolling your own Document Class: Using LaTeX to keep away from the Dark Side

Peter Flynn
Silmaril Consultants
peter (at) silmaril dot ie
http://silmaril.ie/cgi-bin/blog

Abstract

Document classes in LaTeX provide automation to improve consistency, productivity, and accuracy in creating and maintaining documents, thereby avoiding the inefficiencies of wordprocessors. However, users who want to package their macros or applications as a document class are often put off by the apparent complexity of the sample classes in the standard distribution. This paper describes what the code in the article document class file does and suggests solutions to some of the popular requirements for changes.

1 Know thine enemy

One of the key features of TeX systems is the extensibility offered by re-usable pieces of programming called macros. Rudimentary macros exist in many text-handling packages (in fact they were at the heart of the first editors for markup applications), and some wordprocessors make use of general-purpose programming languages such as *Visual Basic* or *Java*; but only typesetters have dedicated languages to doing typesetting, and TeX's is by far the most accessible.

This has led to several large and well-known macro packages (LaTeX, ConTeXt, Texinfo, Eplain, etc) which have all but taken the place of Knuth's original language as the end-user's primary interfaces. Most users now only have to use the macro commands of their chosen interface instead of having to write their own macros afresh or maintain a large private collection of personal macros.

This is not to say that there is no place for homebrew macros in plain TeX: some people have perfectly valid reasons for avoiding the aforementioned packages and continuing to use TeX in the raw. Using one of the above 'standards' does not always mean that you avoid raw TeX in your own code, because you may need some advanced operations which operate at a lower level than normal. It nevertheless remains true that the use of macros to perform groups of frequently-used functions provides a level of automation not found in most wordprocessing systems, and is a major factor in helping users become and remain more productive.

1.1 Standard document classes

The standard document classes installed with LaTeX (article, report, book, and letter) were written in a hybrid of LaTeX and plain TeX code. Sometimes this was because the function Lamport wanted was not worth writing a single-use LaTeX macro for; sometimes it is because (as Knuth describes in another context) "TeX is only 'half obedient' while these definitions are half finished" [4, p. 352]; and sometimes because of the need mentioned above to perform lower-level functions. While the LaTeX 2_ε developers and maintainers have replaced much of the earlier plain TeX code with updated LaTeX equivalents, the code remains fairly dense and is not immediately obvious to the beginner; and the mix of syntax variants can be confusing to the user accustomed to the fairly small set of commands used for common LaTeX documents. Plain TeX itself has some 900 'control sequences' (commands) of which about 350 are 'primitives' (indivisible low-level operations), whereas many regular LaTeX users get by with some 20–30 commands, if even that.

Users who have started to write their own macros, or who have encountered the need to modify LaTeX's defaults for whatever reason, sometimes find the need to encapsulate their favourite format as a document class, governing the entire document, rather than just a package (style file) handling one or two specific features. In this paper we will dissect one of the common document classes and examine what the features and functions are.

1.2 Caveats

This paper uses the article class as the example. The book and report classes are structured very similarly

and the user who has examined the following sections should have no difficulty in identifying the differences.

The letter class, however, is a very different animal. It implements a vertically-centered format once common in typewritten letters but rarely seen nowadays, and has no provision for many of the features users expect to be able to find in a letter template. For this reason I do not refer any further to this format.

The ConTEXt system implements a different and extensible set of document classes — including letters — in a radically different manner to LaTeX and has been discussed and presented extensively in recent years. The Eplain macros implement many of the features of the LaTeX internal mechanisms, but without imposing any document format at all, leaving the plain TeX user free to write those herself.

1.3 More background

The essential documentation to read before you start writing your own classes is *LaTeX 2ε for class and package writers* [8] (available on all modern TeX installations by typing `texdoc clsguide`, and *The LaTeX Companion* [6, App : A.4]. These describe in detail the additional commands available to class and package authors. There are also some special declarations explained in *Companion* [6, p. 847]. The article by Hefferon [3] which I refer to later is a good example of how to build on an existing class. If you have to deal with an obsolete LaTeX 2.09 style file, there is an older paper in *TUGboat* [1].

2 Dissection of `article.cls`

In this example, we use the file from the TeX Live 2005 distribution (so the line numbers refer to that version only). Lines 1–53 are comments and are omitted here for brevity: they explain where the file came from and how it can be used. This is autogenerated because the document class and package files in the standard distributions of LaTeX are derived from master copies maintained in docTeX (`.dtx`) format [7], which combines documentation and LaTeX code in a single file, much in the same way that Knuth's *WEB* system does for many programming languages [9].[1] A short explanation of the sources of the class files is in the *TeX FAQ* [2, label : ltxcmds].

[1] If you intend making your document class available to the rest of the LaTeX community (eg via CTAN), you should make it a docTeX document so that you can combine documentation with your code. Actually, you should probably be doing this anyway...

2.1 Version and identification

The first thing a document class or package must do is identify itself by name, and specify the oldest version of LaTeX with which it will work (it is assumed that it will therefore work with all later versions).

article.cls

```
54 \NeedsTeXFormat{LaTeX2e}[1995/12/01]
55 \ProvidesClass{article}
56               [2004/02/16 v1.4f
57  Standard LaTeX document class]
```

In your new document class file you should set the date and version to the earliest version you have tested your code with (probably your current version). The name of the document class being provided gets checked against the name requested in the `\documentclass` declaration, and LaTeX will give a warning if there is a discrepancy. You may provide a label for the class as well: this will appear in the log file. The linebreaks and indentation are for human readability only.

```
\NeedsTeXFormat{LaTeX2e}[1995/12/01]
\ProvidesClass{ladingbill} [2006/07/01 v0.01 Bill of Lading
                            specialist LaTeX document class]
```

2.2 Initial code and compatibility

On a number of occasions, classes define values as null or a default for later use, so that subsequent code won't trip up as it would if they were undefined. In most cases you will probably need to keep the internal definitions (such as `\@ptsize` here) for use later on (see section 2.4.1 on p. 113).[2]

article.cls

```
58 \newcommand\@ptsize{}
59 \newif\if@restonecol
60 \newif\if@titlepage
61 \@titlepagefalse
```

The conditionals `\if@restonecol` (which flags the restoration of one-column layout after using LaTeX's built-in two-column usage, as distinct from using the multicol package) and `\if@titlepage` (which flags use of the separate title-page layout — set to false in the following line) are used in the default `\maketitle` command in section 2.4.4 on

[2] The use of the `@` sign may be unfamiliar to newcomers: in normal LaTeX it is classified as an 'other' character [4, p. 37]. This means it cannot be used as part of a control sequence (command) in your document. But in class and package files, LaTeX reclassifies it as a 'letter', and uses it in command definitions which are intended to be inaccessible to the normal user. Its use here indicates that the `\@ptsize` command is going to be given a value that the end-user should not be able to interfere with, or even know exists.

p. 116. If you're planning to rewrite `\maketitle` to your own design you may need to take these conditionals into account.[3]

If you are going to invoke additional packages to provide facilities needed by your options, use the `\RequirePackage` command here, before the options section. If the additional packages are unconnected with your option definitions, use the `\RequirePackage` command after the options are executed (see section 2.3.4 on p. 113).

2.3 Options

In an ideal world we wouldn't have to support obsolete versions of software, but the LaTeX defaults still allow v2.09-type `\documentstyle` declarations to be processed, with a warning. However, for a modern class file this is not necessary, so in your own class you can omit all the tests for `\@ifcompatibility` and their `\else` and terminating `\fi` commands, here and throughout, leaving just the code that was in the `\else` blocks.

2.3.1 Paper sizes

How many paper size options you want to support in your class is entirely up to you. You should allow at least A4 and Letter for normal office work.

article.cls

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

[3] How much to cater for and how much to ignore will depend on how much your class deviates from the default. Many LaTeX users will expect to be able to use options like `twocolumn` and `titlepage` simply because they are available in the default classes. But if you are writing a much more prescriptive format, you may want to remove these options entirely, which means removing all references to conditional flags which depend on them.

In some cases you may be writing for a highly specific environment such as your own office or employer, where only one size is required. If so, just omit all the other declarations and add the one option to the `\ExecuteOptions` command (see section 2.3.4 on p. 113).

2.3.2 Type sizes and layout options

As mentioned above, the compatibility settings in this block can be removed in your own class, because modern class files use default option settings via the `\DeclareOption` command instead.

article.cls

```
\if@compatibility
  \renewcommand\@ptsize{0}
\else
\DeclareOption{10pt}{\renewcommand\@ptsize{0}}
\fi
\DeclareOption{11pt}{\renewcommand\@ptsize{1}}
\DeclareOption{12pt}{\renewcommand\@ptsize{2}}
\if@compatibility\else
\DeclareOption{oneside}{\@twosidefalse \@mparswitchfalse}
\fi
\DeclareOption{twoside}{\@twosidetrue  \@mparswitchtrue}
\DeclareOption{draft}{\setlength\overfullrule{5pt}}
\if@compatibility\else
\DeclareOption{final}{\setlength\overfullrule{0pt}}
\fi
\DeclareOption{titlepage}{\@titlepagetrue}
\if@compatibility\else
\DeclareOption{notitlepage}{\@titlepagefalse}
\fi
\if@compatibility\else
\DeclareOption{onecolumn}{\@twocolumnfalse}
\fi
\DeclareOption{twocolumn}{\@twocolumntrue}
\DeclareOption{leqno}{\input{leqno.clo}}
\DeclareOption{fleqn}{\input{fleqn.clo}}
\DeclareOption{openbib}{%
  \AtEndOfPackage{%
   \renewcommand\@openbib@code{%
      \advance\leftmargin\bibindent
      \itemindent -\bibindent
      \listparindent \itemindent
      \parsep \z@
      }%
   \renewcommand\newblock{\par}}%
}
```

The other options should probably be retained, as users may expect them to work, bearing in mind the comment about two-column and title-page settings above. Note that the `openbib` declaration is 10 lines long, and defers itself to end of the package

as a \renewcommand so that it doesn't conflict with the command being declared later.

As with paper sizes above, if your class only needs one specific size setup, just invoke it in \ExecuteOptions.

2.3.3 Your own options

Now is the time to add your own option declarations, if any. Note that option names have no backslash; otherwise the \DeclareOption command works the same way as the \newcommand command (but with no parameters).

Details of how to preserve the options of an existing class you are 'borrowing' via the \LoadClass command are discussed in section 3.1 on p. 122.

2.3.4 Applying options

Two commands control when the options are applied:

article.cls

```
\ExecuteOptions{letterpaper,10pt,oneside,onecolumn,final}
\ProcessOptions
```

\ExecuteOptions applies all the options you specify in the argument, in order, as your selected defaults. The \ProcessOptions command then applies any options the user has selected in their \documentclass declaration.

2.4 Layout

A large number of size and shape settings depend on the selected point size (default 10pt, otherwise as selected in your options). The exact sizes of type chosen for all the different type-size commands are kept in three Class Option files, size10.clo, size11.clo, and size12.clo. There are some others available from CTAN, such as James Kilfiger's size14.clo for readers needing larger type editions, but the three mentioned above cover the vast majority of normal text setting.

If you are going to invoke additional packages that are unconnected with your option definitions, put the \RequirePackage commands here (see section 3.2 on p. 122). Be aware that some packages expect certain variables or definitions already to be present, so their invocation may need to be deferred until after everything else. In this case, enclose the \RequirePackage command in a \AtEndOfPackage or \AtBeginDocument command. This will invoke the package[s] at the specified point in processing, and thus avoid error messages or interference with code in the class file that has not yet been executed.

2.4.1 Type size

To invoke the right settings, the \@ptsize command is embedded in the argument to an \input command:

article.cls

```
\input{size1\@ptsize.clo}
\setlength\lineskip{1\p@}
\setlength\normallineskip{1\p@}
\renewcommand\baselinestretch{}
\setlength\parskip{0\p@ \@plus \p@}
```

A number of basic settings are then made using the internal definition of a point (\p@). The class option files contain a lot of other size-specific settings as well as the font size specifications for the chosen body size.

2.4.1.1 Identity and basic sizes

The class option files (we show size10.clo here) identify themselves in the same way as class files, but using the \ProvidesFile instead of \ProvidesClass.

size10.clo

```
\ProvidesFile{size10.clo}
              [2004/02/16 v1.4f
      Standard LaTeX file (size option)]
\renewcommand\normalsize{%
   \@setfontsize\normalsize\@xpt\@xiipt
   \abovedisplayskip 10\p@ \@plus2\p@ \@minus5\p@
   \abovedisplayshortskip \z@ \@plus3\p@
   \belowdisplayshortskip 6\p@ \@plus3\p@ \@minus3\p@
   \belowdisplayskip \abovedisplayskip
   \let\@listi\@listI}
\normalsize
\newcommand\small{%
   \@setfontsize\small\@ixpt{11}%
   \abovedisplayskip 8.5\p@ \@plus3\p@ \@minus4\p@
   \abovedisplayshortskip \z@ \@plus2\p@
   \belowdisplayshortskip 4\p@ \@plus2\p@ \@minus2\p@
   \def\@listi{\leftmargin\leftmargini
               \topsep 4\p@ \@plus2\p@ \@minus2\p@
               \parsep 2\p@ \@plus\p@ \@minus\p@
               \itemsep \parsep}%
   \belowdisplayskip \abovedisplayskip
}
\newcommand\footnotesize{%
   \@setfontsize\footnotesize\@viiipt{9.5}%
   \abovedisplayskip 6\p@ \@plus2\p@ \@minus4\p@
   \abovedisplayshortskip \z@ \@plus\p@
   \belowdisplayshortskip 3\p@ \@plus\p@ \@minus2\p@
   \def\@listi{\leftmargin\leftmargini
               \topsep 3\p@ \@plus\p@ \@minus\p@
               \parsep 2\p@ \@plus\p@ \@minus\p@
               \itemsep \parsep}%
   \belowdisplayskip \abovedisplayskip
}
\newcommand\scriptsize{\@setfontsize\scriptsize\@viipt\@viiipt}
\newcommand\tiny{\@setfontsize\tiny\@vpt\@vipt}
\newcommand\large{\@setfontsize\large\@xiipt{14}}
```

```
\newcommand\Large{\@setfontsize\Large\@xivpt{18}}
\newcommand\LARGE{\@setfontsize\LARGE\@xviipt{22}}
\newcommand\huge{\@setfontsize\huge\@xxpt{25}}
\newcommand\Huge{\@setfontsize\Huge\@xxvpt{30}}
```

The first block defines the standard LaTeX sizes. These are named using roman numerals (eg `\@xiipt` for 12pt) because digits are not allowed in control sequence names. The more frequently-used sizes also define the display math spacing and the spacing for top-level lists (list definition names also use roman numerals like `\@listi`).

2.4.1.2 Spacing This section controls paragraph indentation (differing between one-column and two-column setting); the dimensions of the three 'shortcut' spacing commands (small, med, and big) but not the actual commands themselves, which are defined in LaTeX itself; and some top-of-page and bottom-of-page spacing settings (normally reset using the geometry package).

size10.clo

```
\if@twocolumn
  \setlength\parindent{1em}
\else
  \setlength\parindent{15\p@}
\fi
\setlength\smallskipamount{3\p@ \@plus 1\p@ \@minus 1\p@}
\setlength\medskipamount{6\p@ \@plus 2\p@ \@minus 2\p@}
\setlength\bigskipamount{12\p@ \@plus 4\p@ \@minus 4\p@}
\setlength\headheight{12\p@}
\setlength\headsep   {25\p@}
\setlength\topskip   {10\p@}
\setlength\footskip{30\p@}
\if@compatibility \setlength\maxdepth{4\p@} \else
\setlength\maxdepth{.5\topskip} \fi
```

2.4.1.3 Text area Text width and text height are set to depend on the columnar setting and a multiple of line-heights respectively.

size10.clo

```
\if@compatibility
  \if@twocolumn
    \setlength\textwidth{410\p@}
  \else
    \setlength\textwidth{345\p@}
  \fi
\else
  \setlength\@tempdima{\paperwidth}
  \addtolength\@tempdima{-2in}
  \setlength\@tempdimb{345\p@}
  \if@twocolumn
    \ifdim\@tempdima>2\@tempdimb\relax
      \setlength\textwidth{2\@tempdimb}
    \else
      \setlength\textwidth{\@tempdima}
    \fi
  \else
    \ifdim\@tempdima>\@tempdimb\relax
      \setlength\textwidth{\@tempdimb}
    \else
      \setlength\textwidth{\@tempdima}
    \fi
  \fi
\fi
\if@compatibility\else
  \@settopoint\textwidth
\fi
\if@compatibility
  \setlength\textheight{43\baselineskip}
\else
  \setlength\@tempdima{\paperheight}
  \addtolength\@tempdima{-2in}
  \addtolength\@tempdima{-1.5in}
  \divide\@tempdima\baselineskip
  \@tempcnta=\@tempdima
  \setlength\textheight{\@tempcnta\baselineskip}
\fi
\addtolength\textheight{\topskip}
```

(The compatibility-mode settings were absolute values in points.) As with paper size and type size, you can preselect exact values for the text area and margins (see next section) using the geometry package.

2.4.1.4 Page margins Margins also depend on the column settings, and on the one-side/two-side setting.

size10.clo

```
\if@twocolumn
 \setlength\marginparsep {10\p@}
\else
  \setlength\marginparsep{11\p@}
\fi
\setlength\marginparpush{5\p@}
\if@compatibility
  \if@twoside
    \setlength\oddsidemargin   {44\p@}
    \setlength\evensidemargin  {82\p@}
    \setlength\marginparwidth {107\p@}
  \else
    \setlength\oddsidemargin   {63\p@}
    \setlength\evensidemargin  {63\p@}
    \setlength\marginparwidth  {90\p@}
  \fi
  \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}
    \setlength\marginparwidth   {.6\@tempdima}
    \addtolength\marginparwidth {-\marginparsep}
    \addtolength\marginparwidth {-0.4in}
  \else
    \setlength\@tempdima        {\paperwidth}
    \addtolength\@tempdima      {-\textwidth}
    \setlength\oddsidemargin    {.5\@tempdima}
    \addtolength\oddsidemargin  {-1in}
    \setlength\marginparwidth   {.5\@tempdima}
    \addtolength\marginparwidth {-\marginparsep}
    \addtolength\marginparwidth {-0.4in}
    \addtolength\marginparwidth {-.4in}
  \fi
  \ifdim \marginparwidth >2in
     \setlength\marginparwidth{2in}
  \fi
  \@settopoint\oddsidemargin
  \@settopoint\marginparwidth
  \setlength\evensidemargin  {\paperwidth}
  \addtolength\evensidemargin{-2in}
  \addtolength\evensidemargin{-\textwidth}
  \addtolength\evensidemargin{-\oddsidemargin}
  \@settopoint\evensidemargin
\fi
\if@compatibility
  \setlength\topmargin{27pt}
\else
  \setlength\topmargin{\paperheight}
  \addtolength\topmargin{-2in}
  \addtolength\topmargin{-\headheight}
  \addtolength\topmargin{-\headsep}
  \addtolength\topmargin{-\textheight}
  \addtolength\topmargin{-\footskip}% this might be wrong
  \addtolength\topmargin{-.5\topmargin}
  \@settopoint\topmargin
\fi
```

Again, the compatibility-mode settings are absolute, whereas the modern defaults are computed.

2.4.1.5 Footnote space Spacing for footnotes and floats is flexible (plus and minus a certain amount) so that the page-breaking routine doesn't become too rigid.

size10.clo

```
\setlength\footnotesep{6.65\p@}
\setlength{\skip\footins}{9\p@ \@plus 4\p@ \@minus 2\p@}
\setlength\floatsep    {12\p@ \@plus 2\p@ \@minus 2\p@}
\setlength\textfloatsep{20\p@ \@plus 2\p@ \@minus 4\p@}
\setlength\intextsep   {12\p@ \@plus 2\p@ \@minus 2\p@}
\setlength\dblfloatsep    {12\p@ \@plus 2\p@ \@minus 2\p@}
\setlength\dbltextfloatsep{20\p@ \@plus 2\p@ \@minus 4\p@}
\setlength\@fptop{0\p@ \@plus 1fil}
\setlength\@fpsep{8\p@ \@plus 2fil}
\setlength\@fpbot{0\p@ \@plus 1fil}
\setlength\@dblfptop{0\p@ \@plus 1fil}
\setlength\@dblfpsep{8\p@ \@plus 2fil}
\setlength\@dblfpbot{0\p@ \@plus 1fil}
\setlength\partopsep{2\p@ \@plus 1\p@ \@minus 1\p@}
```

2.4.1.6 Lists Finally, for the values dependent on type size, the dimensions of lists are set. As mentioned above, names are fabricated using roman numerals (i to vi).

size10.clo

```
\def\@listi{\leftmargin\leftmargini
            \parsep 4\p@ \@plus2\p@ \@minus\p@
            \topsep 8\p@ \@plus2\p@ \@minus4\p@
            \itemsep4\p@ \@plus2\p@ \@minus\p@}
\let\@listI\@listi
\@listi
\def\@listii {\leftmargin\leftmarginii
              \labelwidth\leftmarginii
              \advance\labelwidth-\labelsep
              \topsep    4\p@ \@plus2\p@ \@minus\p@
              \parsep    2\p@ \@plus\p@  \@minus\p@
              \itemsep   \parsep}
\def\@listiii{\leftmargin\leftmarginiii
              \labelwidth\leftmarginiii
              \advance\labelwidth-\labelsep
              \topsep    2\p@ \@plus\p@\@minus\p@
              \parsep    \z@
              \partopsep \p@ \@plus\z@ \@minus\p@
              \itemsep   \topsep}
\def\@listiv {\leftmargin\leftmarginiv
              \labelwidth\leftmarginiv
              \advance\labelwidth-\labelsep}
\def\@listv  {\leftmargin\leftmarginv
              \labelwidth\leftmarginv
              \advance\labelwidth-\labelsep}
\def\@listvi {\leftmargin\leftmarginvi
              \labelwidth\leftmarginvi
              \advance\labelwidth-\labelsep}
\endinput
```

2.4.2 Spacing penalties

Three penalties are set which get invoked in various decisions on paragraph-breaking. You probably don't want to change these unless you are doing deep surgery.

article.cls

```
\@lowpenalty   51
\@medpenalty  151
\@highpenalty 301
\setcounter{topnumber}{2}
\renewcommand\topfraction{.7}
\setcounter{bottomnumber}{1}
\renewcommand\bottomfraction{.3}
\setcounter{totalnumber}{3}
\renewcommand\textfraction{.2}
\renewcommand\floatpagefraction{.5}
\setcounter{dbltopnumber}{2}
\renewcommand\dbltopfraction{.7}
\renewcommand\dblfloatpagefraction{.5}
```

The fractions and numbers refer to the proportions of the page that can be taken up by figures and tables, and the number of floats allowed, when calculating the location of floats.

2.4.3 Running heads

Depending on the imposition (one-sided or two-sided), the default running heads are specified as in the original LaTeX manual [5].

article.cls

```
\if@twoside
  \def\ps@headings{%
      \let\@oddfoot\@empty\let\@evenfoot\@empty
      \def\@evenhead{\thepage\hfil\slshape\leftmark}%
      \def\@oddhead{{\slshape\rightmark}\hfil\thepage}%
      \let\@mkboth\markboth
    \def\sectionmark##1{%
      \markboth {\MakeUppercase{%
        \ifnum \c@secnumdepth >\z@
          \thesection\quad
        \fi
        ##1}}{}}%
    \def\subsectionmark##1{%
      \markright {%
        \ifnum \c@secnumdepth >\@ne
          \thesubsection\quad
        \fi
        ##1}}}
\else
  \def\ps@headings{%
    \let\@oddfoot\@empty
    \def\@oddhead{{\slshape\rightmark}\hfil\thepage}%
    \let\@mkboth\markboth
    \def\sectionmark##1{%
      \markright {\MakeUppercase{%
        \ifnum \c@secnumdepth >\m@ne
          \thesection\quad
        \fi
        ##1}}}}
\fi
\def\ps@myheadings{%
    \let\@oddfoot\@empty\let\@evenfoot\@empty
    \def\@evenhead{\thepage\hfil\slshape\leftmark}%
    \def\@oddhead{{\slshape\rightmark}\hfil\thepage}%
    \let\@mkboth\@gobbletwo
    \let\sectionmark\@gobble
    \let\subsectionmark\@gobble
    }
```

In many cases it may be preferable to use the fancyhdr package instead. This lets you specify a very wide range of header and footer layouts, with left/right switching for double-sided work.

2.4.4 Titling

This is possibly the first big change you'll need to make. There are two `\maketitle` commands defined, one for use on a separate title page (without facilities for attribution), and one for normal use on the starting page (with attributions, and allowing for two columns, using the `\@maketitle` command as well). Both are controlled by the `\if@titlepage` switch.

article.cls

```
  \if@titlepage
  \newcommand\maketitle{\begin{titlepage}%
  \let\footnotesize\small
  \let\footnoterule\relax
  \let \footnote \thanks
  \null\vfil
  \vskip 60\p@
  \begin{center}%
    {\LARGE \@title \par}%
    \vskip 3em%
    {\large
     \lineskip .75em%
      \begin{tabular}[t]{c}%
        \@author
      \end{tabular}\par}%
      \vskip 1.5em%
    {\large \@date \par}%       % Set date in \large size.
  \end{center}\par
  \@thanks
  \vfil\null
  \end{titlepage}%
  \setcounter{footnote}{0}%
  \global\let\thanks\relax
  \global\let\maketitle\relax
  \global\let\@thanks\@empty
  \global\let\@author\@empty
  \global\let\@date\@empty
  \global\let\@title\@empty
  \global\let\title\relax
  \global\let\author\relax
  \global\let\date\relax
```

```
  \global\let\and\relax
}
\else
\newcommand\maketitle{\par
  \begingroup
    \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
    \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
    \long\def\@makefntext##1{\parindent 1em\noindent
            \hb@xt@1.8em{%
                \hss\@textsuperscript{\normalfont\@thefnmark}}##1}%
    \if@twocolumn
      \ifnum \col@number=\@ne
        \@maketitle
      \else
        \twocolumn[\@maketitle]%
      \fi
    \else
      \newpage
      \global\@topnum\z@   % Prevents figures from going at top of page.
      \@maketitle
    \fi
    \thispagestyle{plain}\@thanks
  \endgroup
  \setcounter{footnote}{0}%
  \global\let\thanks\relax
  \global\let\maketitle\relax
  \global\let\@maketitle\relax
  \global\let\@thanks\@empty
  \global\let\@author\@empty
  \global\let\@date\@empty
  \global\let\@title\@empty
  \global\let\title\relax
  \global\let\author\relax
  \global\let\date\relax
  \global\let\and\relax
}
\def\@maketitle{%
  \newpage
  \null
  \vskip 2em%
  \begin{center}%
  \let \footnote \thanks
    {\LARGE \@title \par}%
    \vskip 1.5em%
    {\large
      \lineskip .5em%
      \begin{tabular}[t]{c}%
        \@author
      \end{tabular}\par}%
    \vskip 1em%
    {\large \@date}%
  \end{center}%
  \par
  \vskip 1.5em}
\fi
```

In all of these you can redefine the size, location, and spacing of the three basic titling elements, `\@title`, `\@author`, and `\@date`. (`\author` itself is defined as part of the LaTeX core.) If you are not using two-column setting, or a title-page option, you could replace the whole lot with a single `\renewcommand{\maketitle}{...}` of your own design.

You can also make up your own additional elements, for example an optional subtitle:

```
\def\@subtitle{\relax}
\newcommand{\subtitle}[1]{\gdef\@subtitle{#1}}
\renewcommand{\maketitle}{
   \begin{titlepage}
     \huge\@author\par
     \Large\@title\par
     \if\@subtitle\relax\else\large\@subtitle\par\fi
     \normalsize\@date\par
   \end{titlepage}
}
```

This lets the phantom `\@subtitle` exist unused, set to `\relax` unless an author explicitly uses the `\subtitle` command, because the titling routine can test whether it is still set to `\relax`, and if not, format it accordingly. This technique can be used to add many of the items of metadata used by publishers, such as author affiliations, email and web addresses, and dates of submission.

2.5 Structure

Unless you are doing a very rigid class for data-handling, you probably want to keep the basic sectional structures for normal continuous text as they are, and only change the formatting.

article.cls

```
\setcounter{secnumdepth}{3}
\newcounter {part}
\newcounter {section}
\newcounter {subsection}[section]
\newcounter {subsubsection}[subsection]
\newcounter {paragraph}[subsubsection]
\newcounter {subparagraph}[paragraph]
\renewcommand \thepart {\@Roman\c@part}
\renewcommand \thesection {\@arabic\c@section}
\renewcommand\thesubsection   {\thesection.\@arabic\c@subsection}
\renewcommand\thesubsubsection{\thesubsection .\@arabic\c@subsubsection}
\renewcommand\theparagraph    {\thesubsubsection.\@arabic\c@paragraph}
\renewcommand\thesubparagraph {\theparagraph.\@arabic\c@subparagraph}
\newcommand\part{%
   \if@noskipsec \leavevmode \fi
   \par
   \addvspace{4ex}%
   \@afterindentfalse
   \secdef\@part\@spart}
```

The `\part` command is defined separately, as it operates like `\chapter` in other classes, with more space and a prefix (the book and report classes define a separate `\chapter` command).

article.cls

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

```
      \addcontentsline{toc}{part}{#1}%
    \fi
    {\parindent \z@ \raggedright
     \interlinepenalty \@M
     \normalfont
     \ifnum \c@secnumdepth >\m@ne
       \Large\bfseries \partname\nobreakspace\thepart
       \par\nobreak
     \fi
     \huge \bfseries #2%
     \markboth{}{}\par}%
    \nobreak
    \vskip 3ex
    \@afterheading}
\def\@spart#1{%
    {\parindent \z@ \raggedright
     \interlinepenalty \@M
     \normalfont
     \huge \bfseries #1\par}%
     \nobreak
     \vskip 3ex
     \@afterheading}
```

The sectional formatting is one of the most common features of a document class that need to change. Details of the operation of the `\@startsection` command are in the LaTeX manual [5] if you want to do a complete rewrite, but in many cases one of the packages like sectsty can be used to change fonts or spacing without you having to redo everything from scratch.

article.cls

```
\newcommand\section{\@startsection {section}{1}{\z@}%
                                   {-3.5ex \@plus -1ex \@minus -.2ex}%
                                   {2.3ex \@plus.2ex}%
                                   {\normalfont\Large\bfseries}}
\newcommand\subsection{\@startsection{subsection}{2}{\z@}%
                                     {-3.25ex\@plus -1ex \@minus -.2ex}%
                                     {1.5ex \@plus .2ex}%
                                     {\normalfont\large\bfseries}}
\newcommand\subsubsection{\@startsection{subsubsection}{3}{\z@}%
                                     {-3.25ex\@plus -1ex \@minus -.2ex}%
                                     {1.5ex \@plus .2ex}%
                                     {\normalfont\normalsize\bfseries}}
\newcommand\paragraph{\@startsection{paragraph}{4}{\z@}%
                                    {3.25ex \@plus1ex \@minus.2ex}%
                                    {-1em}%
                                    {\normalfont\normalsize\bfseries}}
\newcommand\subparagraph{\@startsection{subparagraph}{5}{\parindent}%
                                      {3.25ex \@plus1ex \@minus .2ex}%
                                      {-1em}%
                                      {\normalfont\normalsize\bfseries}}
```

2.6 Indents and margins

In this section the class file defines the internal margins set around block elements like lists. For controlling lists, LaTeX provides four levels of indentation. As explained earlier, because digits are not permitted in command names, all these parameters end in the Roman-numeral equivalents.

article.cls

```
\if@twocolumn
  \setlength\leftmargini  {2em}
\else
  \setlength\leftmargini  {2.5em}
\fi
\leftmargin  \leftmargini
\setlength\leftmarginii  {2.2em}
\setlength\leftmarginiii {1.87em}
\setlength\leftmarginiv  {1.7em}
\if@twocolumn
  \setlength\leftmarginv  {.5em}
  \setlength\leftmarginvi {.5em}
\else
  \setlength\leftmarginv  {1em}
  \setlength\leftmarginvi {1em}
\fi
\setlength  \labelsep  {.5em}
\setlength  \labelwidth{\leftmargini}
\addtolength\labelwidth{-\labelsep}
\@beginparpenalty -\@lowpenalty
\@endparpenalty   -\@lowpenalty
\@itempenalty     -\@lowpenalty
\renewcommand\theenumi{\@arabic\c@enumi}
\renewcommand\theenumii{\@alph\c@enumii}
\renewcommand\theenumiii{\@roman\c@enumiii}
\renewcommand\theenumiv{\@Alph\c@enumiv}
\newcommand\labelenumi{\theenumi.}
\newcommand\labelenumii{(\theenumii)}
\newcommand\labelenumiii{\theenumiii.}
\newcommand\labelenumiv{\theenumiv.}
\renewcommand\p@enumii{\theenumi}
\renewcommand\p@enumiii{\theenumi(\theenumii)}
\renewcommand\p@enumiv{\p@enumiii\theenumiii}
\newcommand\labelitemi{\textbullet}
\newcommand\labelitemii{\normalfont\bfseries \textendash}
\newcommand\labelitemiii{\textasteriskcentered}
\newcommand\labelitemiv{\textperiodcentered}
\newenvironment{description}
         {\list{}{\labelwidth\z@ \itemindent-\leftmargin
                  \let\makelabel\descriptionlabel}}
         {\endlist}
\newcommand*\descriptionlabel[1]{\hspace\labelsep
                                 \normalfont\bfseries #1}
```

The variables and their meaning are described in more detail in the LaTeX manual [5] and the *Companion* [6], but essentially:

`\leftmargin`*rr* are the list level indentations from outer page margin to the start of the text;

`\labelsep` is the space between the number or bullet and the start of the text;

`\labelwidth` is how much space to allow for the numbering or bulleting;

`\theenum`*rr* controls the style of numbering;

`\labelenum`*rr* controls the style of bulleting.

In all these cases, you can remove the conditional code surrounding the variants for two-column work, and have just one setting, if you are not going to provide for two-column setting.

The description environment works slightly differently: the `\makelabel` command is equated to a `\descriptionlabel` command to indent and format the item label. This is easily redefined, for example to make the labels use the sans-serif font instead of the default roman typeface, and add an automatic em-rule afterwards:

```
\renewcommand*\descriptionlabel[1]{
  \hspace\labelsep
  \relax\sffamily{\bfseries #1}~---\space
                  \ignorespaces}
```

2.7 Abstract

The default abstract is formatted differently according to where it appears: on the first page or on a page by itself after a separate title page.

article.cls

```
\if@titlepage
  \newenvironment{abstract}{%
      \titlepage
      \null\vfil
      \@beginparpenalty\@lowpenalty
      \begin{center}%
        \bfseries \abstractname
        \@endparpenalty\@M
      \end{center}}%
     {\par\vfil\null\endtitlepage}
\else
  \newenvironment{abstract}{%
    \if@twocolumn
      \section*{\abstractname}%
    \else
      \small
      \begin{center}%
       {\bfseries \abstractname\vspace{-.5em}\vspace{\z@}}%
      \end{center}%
      \quotation
    \fi}
    {\if@twocolumn\else\endquotation\fi}
\fi
```

One common requirement is for the Abstract formatting to follow the pattern of a subsection when it appears on a separate page, eg

```
\newenvironment{abstract}{%
    \titlepage
    \subsection*{\abstractname}}%
   {\par\vfil\null\endtitlepage}
```

Some styles require turning off the initial indentation when the abstract is on the first page, for consistency with the default Anglo-American style used in sections:

```
\newenvironment{abstract}{%
  \if@twocolumn
    \section*{\abstractname}%
  \else
    \small
    \begin{center}%
      {\bfseries \abstractname\vspace{-.5em}
                 \vspace{\z@}}%
    \end{center}%
    \quotation\noindent\ignorespaces
  \fi}
  {\if@twocolumn\else\endquotation\fi}
```

Note that if you will be adding to an existing class in the manner described in section 3.1 on p. 122, these last two examples will use the `\renewenvironment` command instead.

2.8 Structural elements

The default classes contain some rudimentary environments for verse and quotations, and a compatibility setting for LaTeX 2.09 users, which can be omitted from new classes (make sure you keep one definition of the titlepage environment, though!

article.cls

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

```
                {\endlist}
\if@compatibility
\newenvironment{titlepage}
    {%
      \if@twocolumn
        \@restonecoltrue\onecolumn
      \else
        \@restonecolfalse\newpage
      \fi
      \thispagestyle{empty}%
      \setcounter{page}\z@
    }%
    {\if@restonecol\twocolumn \else \newpage \fi
    }
\else
\newenvironment{titlepage}
    {%
      \if@twocolumn
        \@restonecoltrue\onecolumn
      \else
        \@restonecolfalse\newpage
      \fi
      \thispagestyle{empty}%
      \setcounter{page}\@ne
    }%
    {\if@restonecol\twocolumn \else \newpage \fi
     \if@twoside\else
        \setcounter{page}\@ne
     \fi
    }
\fi
\newcommand\appendix{\par
  \setcounter{section}{0}%
  \setcounter{subsection}{0}%
  \gdef\thesection{\@Alph\c@section}}
```

The quotation environment is another which benefits from the removal of the initial indentation:

```
\newenvironment{quotation}
       {\list{}{\listparindent 1.5em%
        \itemindent    \z@
        \rightmargin   \leftmargin
        \parsep        \z@ \@plus\p@}%
        \item\relax}
       {\endlist}
```

For the reasons noted in section 2.7 on p. 119, this may need to be a `\renewcommand`.

This section ends with a definition for `\appendix` which switches the `\section` settings to produce labels with A, B, C, etc instead of 1, 2, 3.

2.9 Figures and tables

These are controlled by a number of dimensions which you may already be familiar with, such as `\tabcolsep` for the gap between table columns. The `\fboxsep` and `\fboxrule` dimensions control the gap and rule thickness around boxed text.

article.cls

```
\setlength\arraycolsep{5\p@}
\setlength\tabcolsep{6\p@}
\setlength\arrayrulewidth{.4\p@}
\setlength\doublerulesep{2\p@}
\setlength\tabbingsep{\labelsep}
\skip\@mpfootins = \skip\footins
\setlength\fboxsep{3\p@}
\setlength\fboxrule{.4\p@}
\renewcommand \theequation {\@arabic\c@equation}
\newcounter{figure}
\renewcommand \thefigure {\@arabic\c@figure}
\def\fps@figure{tbp}
\def\ftype@figure{1}
\def\ext@figure{lof}
\def\fnum@figure{\figurename\nobreakspace\thefigure}
\newenvironment{figure}
               {\@float{figure}}
               {\end@float}
\newenvironment{figure*}
               {\@dblfloat{figure}}
               {\end@dblfloat}
\newcounter{table}
\renewcommand\thetable{\@arabic\c@table}
\def\fps@table{tbp}
\def\ftype@table{2}
\def\ext@table{lot}
\def\fnum@table{\tablename\nobreakspace\thetable}
\newenvironment{table}
               {\@float{table}}
               {\end@float}
\newenvironment{table*}
               {\@dblfloat{table}}
               {\end@dblfloat}
\newlength\abovecaptionskip
\newlength\belowcaptionskip
\setlength\abovecaptionskip{10\p@}
\setlength\belowcaptionskip{0\p@}
\long\def\@makecaption#1#2{%
  \vskip\abovecaptionskip
  \sbox\@tempboxa{#1: #2}%
  \ifdim \wd\@tempboxa >\hsize
    #1: #2\par
  \else
    \global \@minipagefalse
    \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
```

```
  \fi
  \vskip\belowcaptionskip}
```

At the end of this section is the `\@makecaption` command, another popular candidate for redesign, but consider also using the ccaption package.

2.10 Legacy support

The obsolescent commands `\rm`, `\it`, `\bf`, etc are declared here to function as their modern equivalents.

article.cls

```
\DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
\DeclareOldFontCommand{\tt}{\normalfont\ttfamily}{\mathtt}
\DeclareOldFontCommand{\bf}{\normalfont\bfseries}{\mathbf}
\DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
\DeclareOldFontCommand{\sl}{\normalfont\slshape}{\@nomath\sl}
\DeclareOldFontCommand{\sc}{\normalfont\scshape}{\@nomath\sc}
\DeclareRobustCommand*\cal{\@fontswitch\relax\mathcal}
\DeclareRobustCommand*\mit{\@fontswitch\relax\mathnormal}
```

2.11 Table of contents

The Table of Contents section starts with some commands which evaluate to dimensions, plus the `\tableofcontents` command itself.

article.cls

```
\newcommand\@pnumwidth{1.55em}
\newcommand\@tocrmarg{2.55em}
\newcommand\@dotsep{4.5}
\setcounter{tocdepth}{3}
\newcommand\tableofcontents{%
    \section*{\contentsname
        \@mkboth{%
           \MakeUppercase\contentsname}{\MakeUppercase\contentsname}}%
    \@starttoc{toc}%
    }
\newcommand*\l@part[2]{%
  \ifnum \c@tocdepth >-2\relax
    \addpenalty\@secpenalty
    \addvspace{2.25em \@plus\p@}%
    \setlength\@tempdima{3em}%
    \begingroup
      \parindent \z@ \rightskip \@pnumwidth
      \parfillskip -\@pnumwidth
      {\leavevmode
       \large \bfseries #1\hfil \hb@xt@\@pnumwidth{\hss #2}}\par
       \nobreak
       \if@compatibility
         \global\@nobreaktrue
         \everypar{\global\@nobreakfalse\everypar{}}%
       \fi
    \endgroup
  \fi}
\newcommand*\l@section[2]{%
  \ifnum \c@tocdepth >\z@
    \addpenalty\@secpenalty
    \addvspace{1.0em \@plus\p@}%
    \setlength\@tempdima{1.5em}%
    \begingroup
      \parindent \z@ \rightskip \@pnumwidth
      \parfillskip -\@pnumwidth
      \leavevmode \bfseries
      \advance\leftskip\@tempdima
      \hskip -\leftskip
      #1\nobreak\hfil \nobreak\hb@xt@\@pnumwidth{\hss #2}\par
    \endgroup
  \fi}
\newcommand*\l@subsection{\@dottedtocline{2}{1.5em}{2.3em}}
\newcommand*\l@subsubsection{\@dottedtocline{3}{3.8em}{3.2em}}
\newcommand*\l@paragraph{\@dottedtocline{4}{7.0em}{4.1em}}
\newcommand*\l@subparagraph{\@dottedtocline{5}{10em}{5em}}
\newcommand\listoffigures{%
    \section*{\listfigurename}%
      \@mkboth{\MakeUppercase\listfigurename}%
              {\MakeUppercase\listfigurename}%
    \@starttoc{lof}%
    }
\newcommand*\l@figure{\@dottedtocline{1}{1.5em}{2.3em}}
\newcommand\listoftables{%
    \section*{\listtablename}%
      \@mkboth{%
          \MakeUppercase\listtablename}%
         {\MakeUppercase\listtablename}%
    \@starttoc{lot}%
    }
\let\l@table\l@figure
```

There are `\l@`*ttt* commands (`\l@part`, `\l@section`, etc) which produce the ToC lines from the `.aux` file. The List of Tables and List of Figures are implemented in the same way as the ToC. As with other features, consider the tocloft package for common modifications.

2.12 Bibliography and index

Bibliography styles themselves are implemented in `.bst` files, but the style of the section can be changed here, including indentation and spacing.

article.cls

```
\newdimen\bibindent
\setlength\bibindent{1.5em}
\newenvironment{thebibliography}[1]
     {\section*{\refname}%
      \@mkboth{\MakeUppercase\refname}{\MakeUppercase\refname}%
      \list{\@biblabel{\@arabic\c@enumiv}}%
           {\settowidth\labelwidth{\@biblabel{#1}}%
            \leftmargin\labelwidth
            \advance\leftmargin\labelsep
            \@openbib@code
            \usecounter{enumiv}%
            \let\p@enumiv\@empty
            \renewcommand\theenumiv{\@arabic\c@enumiv}}%
      \sloppy
      \clubpenalty4000
      \@clubpenalty \clubpenalty
      \widowpenalty4000%
      \sfcode`\.\@m}
     {\def\@noitemerr
       {\@latex@warning{Empty `thebibliography' environment}}%
      \endlist}
\newcommand\newblock{\hskip .11em\@plus.33em\@minus.07em}
\let\@openbib@code\@empty
\newenvironment{theindex}
               {\if@twocolumn
                  \@restonecolfalse
```

```
                \else
                  \@restonecoltrue
                \fi
                \twocolumn[\section*{\indexname}]%
                \@mkboth{\MakeUppercase\indexname}%
                        {\MakeUppercase\indexname}%
                \thispagestyle{plain}\parindent\z@
                \parskip\z@ \@plus .3\p@\relax
                \columnseprule \z@
                \columnsep 35\p@
                \let\item\@idxitem}
               {\if@restonecol\onecolumn\else\clearpage\fi}
\newcommand\@idxitem{\par\hangindent 40\p@}
\newcommand\subitem{\@idxitem \hspace*{20\p@}}
\newcommand\subsubitem{\@idxitem \hspace*{30\p@}}
\newcommand\indexspace{\par \vskip 10\p@ \@plus5\p@ \@minus3\p@\relax}
```

2.13 Odds 'n' ends

The final section starts with the footnote 'fence' and the footnote alignment. There is also a list of the section names, which are the ones which get customised for other languages when you use the babel multilingual/multicultural package.

article.cls

```
\renewcommand\footnoterule{%
  \kern-3\p@
  \hrule\@width.4\columnwidth
  \kern2.6\p@}
\newcommand\@makefntext[1]{%
    \parindent 1em%
    \noindent
    \hb@xt@1.8em{\hss\@makefnmark}#1}
\newcommand\contentsname{Contents}
\newcommand\listfigurename{List of Figures}
\newcommand\listtablename{List of Tables}
\newcommand\refname{References}
\newcommand\indexname{Index}
\newcommand\figurename{Figure}
\newcommand\tablename{Table}
\newcommand\partname{Part}
\newcommand\appendixname{Appendix}
\newcommand\abstractname{Abstract}
\def\today{\ifcase\month\or
  January\or February\or March\or April\or May\or June\or
  July\or August\or September\or October\or
  November \or December\fi \space\number\day, \number\year}
\setlength\columnsep{10\p@}
\setlength\columnseprule{0\p@}
\pagestyle{plain}
\pagenumbering{arabic}
\if@twoside
\else
  \raggedbottom
\fi
\if@twocolumn
  \twocolumn
  \sloppy
  \flushbottom
\else
  \onecolumn
\fi
\endinput
```

To end with, there is the `\today` date, which non-Americans can recode as:

```
\def\today{\number\day\space\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\year}
```

The last few lines include the column spacing, page style, and page numbering setups. Single-sided work is allowed to have a slightly variable text height (the `\raggedbottom` command), and two-column setting has a strict height but slightly greater tolerance on justification.

3 Rolling your own

Having seen what the article class does and how it works, you have a choice: create your new class file from scratch, or build onto an existing class.

Writing a wholly new class requires a significant knowledge of LaTeX and TeX internals, but will have the advantage of being dedicated to the specific task on hand, and may offer more scope for automation, particularly if the process of generating the output is to be embedded within a larger application.

3.1 Re-using an existing class

Building on the work of other classes is more common, and has been described for a specific application (Minutes of meetings) in [3]. This involves loading the existing class file, handling any existing or new options, and then adding or modifying the commands and environments it provides.

We have already seen the use of `\renewcommand` (section 2.4.4 on p. 116) and its counterpart for environments, `\renewenvironment` (section 2.7 on p. 119), but you need to ensure the command and environments you are replacing are correctly preloaded. Hefferon [3] describes in detail the use of the `\LoadClass` and `\DeclareOption*` commands to specify the class on which you want to base yours, how to preserve existing options, and how to add your own.

3.2 Packages

As well as rewriting or modifying the code of an existing class, you can also invoke extra packages. In most cases this is faster, more reliable, and easier to do than rewriting the code of the existing class.

We have mentioned several useful packages:

geometry for the text area and page margins;

multicol for multiple columns of text;

fancyhdr for running headers and footers;

sectsty for changes to section and title styles;

ccaption for changes to the layout of Table and Figure captions;

tocloft for changes to the layout of the Table of Contents and Lists of Figures and Tables;

babel for working in multiple languages.

In your new class file, use the `\RequirePackage` command after the options (see section 2.3.4 on p. 113). If an option needs to refer to a specific package, put the `\RequirePackage` after the version and identification section but before your options (see section 2.2 on p. 111).

3.3 Four last things

The *Companion* [6, p. 888] specifies that 'every class file *must* contain four things':

1. a definition of `\normalsize`;
2. a value for `\textwidth`;
3. a value for `\textheight`;
4. a specification for `\pagenumbering`.

Beyond that, it's up to you! If you have been documenting your class file in docTeX format as you go along, as explained in the first paragraph in section 2, you should now consider releasing it for general use by submitting it to the CTAN maintainers so that others can use it.

Acknowledgments

This article originally appeared in the *PracTeX Journal* (2006:4) where it was set full out. The challenge in a two-column layout of fitting wide lines of verbatim code from files whose line-numbers are needed for reference was met by a suggestion from Karl Berry to use the Latin Modern Typewriter Light Condensed font (`lmttlc`) in the `\VerbatimInput` command of the fancyvrb package.

References

[1] Johannes Braams. Document classes and packages in LaTeX 2_ε. *TUGboat*, 15(3), Sep 1994. `http://www.tug.org/TUGboat/Contents/contents15-3.html`.

[2] Robin Fairbairns (Ed). Frequently Asked Questions about TeX. Technical report, UK TeX Users Group, Cambridge, UK, Nov 2005. `http://www.tex.ac.uk/cgi-bin/texfaq2html?label=ltxcmds`.

[3] Jim Hefferon. Minutes in less than hours: Using LaTeX resources. *The PracTeX Journal*, 2005(4), Oct 2005. `http://www.tug.org/pracjourn/2005-4/hefferon/`.

[4] Donald Knuth. *The TeXbook*. Addison-Wesley, Reading, MA, Jun 1986.

[5] Leslie Lamport. *LaTeX: A document preparation system*. Addison-Wesley, Reading, MA, 2nd edition, 1994.

[6] Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, Chris Rowley, Christine Detig, and Joachim Schrod. *The LaTeX Companion: Tools and Techniques for Computer Typesetting*. Addison-Wesley, Reading, MA, 2nd edition, May 2004.

[7] Scott Pakin. How to package your LaTeX package. *CTAN*, Nov 2005. `http://www.ctan.org/tex-archive/info/dtxtut/dtxtut.pdf`.

[8] The LaTeX3 Project. LaTeX 2_ε for class and package writers, Dec 2003. `$TEXMFMAIN/texmf-dist/doc/latex/base/clsguide.{dvi|pdf}`.

[9] Wayne Sewell. *Literate Programming in WEB*. Van Nostrand Reinhold, New York, NY, 1989.