A style option to adapt the standard LaTeX document styles to A4 paper*

Nico Poppelier
TeXnique
Washingtondreef 153
3564 KD Utrecht
Poppelier@elsevier.nl

Johannes Braams
PTT Research
P.O. Box 421
2260 AK Leidschendam
J.L.Braams@cistron.nl

Printed August 21, 1995

Abstract

This article describes a new style option that can be used with the document styles that are distributed with the LaTeX distributions. It modifies the page layout to conform to the paper format most commonly used in Europe, portrait A4.

1 Introduction

This file is based on the document style options `A4.sty` and `A4wide.sty`, which can be found in the Rochester style archive. The original style option `A4.sty` we started from was written by John Pavel, and is dated May 1987. This option only changes the vertical size of the text somewhat, by increasing the number of lines on a page. The style option `A4wide.sty` was written by Jean-Francois Lamy, and is dated July 1986. This option only increases the width of the text.

2 Goals and design decisions

As many people before us, we found the page layout as implemented in the standard LaTeX document styles too much geared towards the American-sized paper, which is somewhat wider than A4 paper, but also noticeably less high.

Our goal was to get a page layout that was suitable for A4 paper, and produced legible texts. There are a number of layout parameters that influence the legibility of a text. A parameter of major importance is the number of words (or characters) on a line. The maximum number of words per line is ten to twelve for optimal legibility, a rule-of-thumb that can be found in typographic literature (we used [?]). This results in a number of characters per line which lies somewhere between sixty and seventy.

Another important parameter is the amount of white space surrounding the text. Here we have to distinguish between texts that are printed one-sided and

*This file has version number ?– last revision ?.

texts that are printed two-sided (back to back). In the first case the margins on odd and even pages should be equal; in the latter case care should be taken that the texts on both sides of the paper overlap. Also a printed document is likely to be bound some way or another, so there should be enough white space in the 'inner' margin[1] of the text to allow this.

There is yet one more thing to take into account when designing a page layout. LaTeX offers the possibility of using marginal notes and if someone wants to use marginal notes, they should of course fit on the paper.

So, we have the following goals:

1. Choose the text width such that there will be sixty to seventy characters on a line;

2. See to it that in documents that are printed two-sided, the texts which end up on two sides of one sheet of paper overlap;

3. Leave enough white space in the 'inner' margin to allow for the binding of the document;

4. Leave enough white space in the 'outer' margin for marginal notes if they are going to be used.

3 Update for LaTeX 2ε

With the new document classes this package may not be needed much longer, because A4 paper is now supported through the standard option a4paper. Yet this package additional functionality, not available in the standard document classes. With LaTeX 2ε, you can now use this package with the command:

```
\usepacke[widemargins]{a4}
```

The option widemargins executes the `\WideMargins` command.

4 The implementation

4.1 The starting point

Thus we set out to modify some of the design decisions in the standard document styles. Because we knew that we were not the first to tackle the problem, we started by having a look at what was already available. We came up with the two options mentioned earlier, which are publicly available. Undoubtedly there will exist many more such files, some of them maybe modifications of those two files.

We had a look at the layout produced by both options and were not satisfied with it. For one thing, both of the original options a4 and A4wide modify only one aspect of the page layout. The first thing to do was to put these two files together. This resulted in a layout which was still unsatisfactory, since for the 10-point and 11-point options lines in the text contained on the average eighty characters or more.

[1] For two-sided printing, this is the left margin on odd-numbered pages and the right margin on even-numbered ones; for one-sided printing, this is always the left margin.

4.2 What else?

\textwidth
\marginparwidth

Because the result so far gave us lines that contained too many characters, we decreased the `\textwidth` to get lines that contain about sixty to seventy characters for all three size options. Still more work had to be done. As it turned out, using our new `A4.sty` together with the option `twosided` had a drawback: when the document was printed two-sided the texts on both side of one piece of paper overlapped only partly, which does not look good. We solved this by modifying the width of the margins for two-sided printing. At the same time we modified the `\marginparwidth` so that if someone uses a marginal note it would completely fit on the paper instead of falling off the page, which obviously would render the note unreadable.

\WideMargins

The decisions described above allow for marginal notes to be printed along with the normal text, but if someone makes heavy use of marginal notes, the resultant layout will not be very satisfactory, because if the full width of the marginal notes is used, they will take up too much space in the 'outer' margin. For this case we provide the macro `\WideMargins`. This macro modifies the page-layout parameters in such a way that the width reserved for marginal notes becomes 1.5 inches. To achieve this the width of the main body of the text is decreased. This macro is meant to be used only in the preamble of the document.

4.3 The docstrip modules

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

driver	produce a documentation driver file
package	produce a package file

4.4 Producing the documentation

A short driver is provided that can be extracted if necessary by the DOCSTRIP program provided with LaTeX 2_ε.

```
⟨*driver⟩
\NeedsTeXFormat{LaTeX2e}
\documentclass{ltxdoc}
\DisableCrossrefs

\CodelineIndex

\MakeShortVerb{\|}
\newcommand\Lopt[1]{\textsf{#1}}
\newcommand\file[1]{\texttt{#1}}
\begin{document}

\DocInput{a4.dtx}

\end{document}
⟨/driver⟩
```

4.5 The code

Declare the option widemargins.

```
⟨*package⟩
\ifx\documentclass\undefined
\else
  \DeclareOption{widemargins}{\WideMargins}
\fi
```

\topmargin First, we redefine the `\textheight` and `\topmargin`. The `\topmargin` is the distance from the reference point on the page to the top of the page of text. In most cases extra white space is not necessary since one inch of white space at the top of the page suffices.

```
\topmargin 0pt
```

\textheight The dimension parameter `\textheight` gives the total height of the text, including footnotes and figures, excluding the running head and foot. This height is given as an integral number times the `\baselineskip`, which results in an integral number of lines on a page.

We have to include definitions of all relevant dimension parameters for each of the cases 10-point, 11-point and 12-point. We do this with a case statement:

```
\ifcase \@ptsize
    \textheight 53\baselineskip
```

which modifies the height of the text for texts to be produced with the ten-point typeface:

```
\or
    \textheight 46\baselineskip
```

the same for eleven point:

```
\or
    \textheight 42\baselineskip
```

and for twelve point. Finally we close the `\ifcase` statement:

```
\fi
```

The only thing left to be done is to add the `\topskip` to the `\textheight`. The value of `\topskip` appears always to be 10pt.

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

\textwidth \oddsidemargin \evensidemargin That was the 'vertical part' of the work. Now we have some work to do to get things right horizontally. Again we have to distinguish between the various character sizes because sixty eleven-point characters take up more space than sixty ten-point characters. But there's more to take into account. If documents are printed two-sided, the texts on both sides of the paper should overlap completely. This can be done be assigning appropriate values to `\oddsidemargin` and `\evensidemargin`, the parameters that define the left margins on odd and even pages respectively.

First we start a case statement to distinguish between the various typeface sizes.

```
\ifcase \@ptsize
```

Then we specify the width of the text.

```
    \textwidth      5.00in
```

Also specify the width of marginal notes. They must have a reasonable width to be of any use, and this should be the same for either one-sided or two-sided printing.

```
    \marginparwidth 1.00in
```

Here we need an if statement to test whether the option `twosided` has been specified.

```
  \if@twoside
```

If it was, assign appropriate values to the margin parameters

```
    \oddsidemargin  0.55in
    \evensidemargin 0.75in
  \else
```

If the option `twosided` was not used, both margin parameters must have the same value, for texts on consecutive pages have to be put in the same place on the paper.

```
    \oddsidemargin  0.55in
    \evensidemargin 0.55in
```

Now we close the if statement.

```
  \fi
```

We are ready with the modifications for the ten-point typeface size, so now we do something similar for the eleven-point typeface.

```
\or
    \textwidth      5.20in
    \marginparwidth 1.00in
  \if@twoside
    \oddsidemargin  0.45in
    \evensidemargin 0.65in
  \else
    \oddsidemargin  0.45in
    \evensidemargin 0.45in
  \fi
```

One more time, now for the twelve-point typeface.

```
\or
    \textwidth      5.70in
    \marginparwidth 0.80in
  \if@twoside
    \oddsidemargin  0.20in
    \evensidemargin 0.40in
  \else
    \oddsidemargin  0.20in
    \evensidemargin 0.20in
  \fi
```

Finally we close the case statement.

```
\fi
```

`\WideMargins` This macro is somewhat tricky: it has to find out which typeface size is used, whether the document should be printed two-sided, and whether the `\reversemarginpar` is in effect. `\reversemarginpar` makes the marginal notes appear in the margin on the opposite side of the normal placement.

```
\def\WideMargins{%
```

Because for each typeface size the changes to the parameters that need to be made are similar, the macro \WideMargins uses an internal macro \@widemargins.

\ExtraWidth In order to store the amount of extra width needed for the marginal notes an extra dimension parameter is defined.

```
\newdimen\ExtraWidth
```

First find out about the point size, then call \@widemargins to modify the margin widths by the amount given in \ExtraWidth.

```
\ifcase \@ptsize
```

For both 10-point and 11-point texts the width for marginal notes is already 1 inch, so we increase it by half an inch. We subtract half an inch from the text width and modify the margins appropriately.

```
  \ExtraWidth = 0.5in
  \@widemargins
\or
  \ExtraWidth = 0.5in
  \@widemargins
\or
```

For 12-point texts the marginal notes are only 0.8 inch wide, so now we have to add 0.7 inch to get them 1.5 inch wide.

```
  \ExtraWidth = 0.7in
  \@widemargins
```

This macro should only be called once, during the preamble of a document, so we \let it be equal to \relax as soon as the work is done.

```
\fi\let\WideMargins\relax\let\@widemargins\relax}
```

\@preamblecmds We add \WideMargins to \@preamblecmds, which is a list of commands to be used only in the preamble of a document.

```
{\def\do{\noexpand\do\noexpand}
 \xdef\@preamblecmds{\@preamblecmds \do\WideMargins}
}
```

\@widemargins This macro modifies the margin parameters. To do this it uses the dimension variable \ExtraWidth, which was defined by \WideMargins.
First the \ExtraWidth is subtracted from the \textwidth and added to the \marginparwidth.

```
\def\@widemargins{%
   \global\advance\textwidth by -\ExtraWidth
   \global\advance\marginparwidth by \ExtraWidth
```

Then we modify the margins, but the value of the switch \if@twoside has to be taken into account. Because we have to test another switch (\if@reversemargin) we add another level of macros to modify the margin parameters

```
   \if@twoside
     \tw@sidedwidemargins
   \else
     \@nesidedwidemargins
   \fi}
```

\tw@sidedwidemargins

Normally the marginal notes are printed in the 'outer' margins, so we have to increase the `\evensidemargin` to keep the text balanced on both sides of the paper, but if `\reversemarginpar` is in effect we have to increase the `\oddsidemargin` and decrease the `\evensidemargin` accordingly.

```
\def\tw@sidedwidemargins{%
   \if@reversemargin
```

Notice that for documents printed two-sided, the `\evensidemargin` is wider than the `\oddsidemargin`; this difference in width is transferred to the other margin.

```
     \@tempdima=\evensidemargin
     \advance\@tempdima by -\oddsidemargin
     \advance\oddsidemargin by \ExtraWidth
     \advance\oddsidemargin by \@tempdima
     \advance\evensidemargin by -\@tempdima
   \else
```

If the marginal notes go on the normal side of the paper, only the `\evensidemargin` has to be increased.

```
     \advance\evensidemargin by \ExtraWidth
   \fi}
```

\@nesidedwidemargins

For documents that are printed one-sided, both margins have the same width. The default placement for the marginal notes is in the right margin, so if `\reversemarginpar` is *not* in effect the margin parameters need not be modified. If it is in effect, both the `\oddsidemargin` and the `\evensidemargin` need to be increased.

```
\def\@nesidedwidemargins{%
   \if@reversemargin
     \advance\oddsidemargin by \ExtraWidth
     \advance\evensidemargin by \ExtraWidth
   \fi}
```

The command `\ProcessOptions` can only be executed *after* `\WideMargins` has been defined. Defering the execution of `\WideMargins` with `\AtBeginDocument` doesn't work, the changing of `\textwidth` then comes too late because of the time when the `\@begindocumenthook` gets executed by `\begin{document`.

```
\ifx\documentclass\undefined
\else
  \ProcessOptions
\fi
⟨/package⟩
```

5 Conclusion

We have presented a new approach to adapt the page layout of the document styles that are part of the standard LaTeX distributions to the dimensions of A4 paper. The width of marginal notes has been taken into account and a means to get wider marginal notes at the cost of shorter lines in the main body of the text has been provided.

References

[1] K.F. Treebus. *Tekstwijzer, een gids voor het grafisch verwerken van tekst.* SDU Uitgeverij ('s-Gravenhage, 1988). A Dutch book on layout design and typography.