A style file for printing sheets of labels

Sebastian Rahtz

Contents

Abstract

A LaTeX style to print a regular grid of labels on a page, suitable for sheets of labels which can be fed through a laser printer. Macros are provided to allow easy input of names and addresses in a form free of TeX markup.

1 Usage

This style file was written to print labels from the shop around the corner from me. These have 8 rows and 3 columns on a sheet of A4 paper. Your labels will very likely be different. So first you have to tailor this file to your particular type of label. Edit the lines below which look like this:

```
\num@labelcols=3
\num@labelrows=8
```

to reflect *your* grid (maybe you have only two columns of ten labels each, for instance). Now make sure that your printer driver prints the page *exactly* as it should in vanilla TeX, i.e. with the origin of the page down 1in and right 1in from the top left hand corner of the paper. If it doesn't, adjust your driver parameters, or edit the settings below where I take 1in off the margins. The most likely problem with these macros is that you will have contents which are quite wide, and which therefore need to use the very edges of the paper, on which your printer may not write correctly. Little one can do about this — use a small point size.

The simplest form of input is very easy, as in the following example:

```
\documentstyle{labels}
\begin{document}
\begin{labels}
\input names.dat
\end{labels}
\end{document}
```

where `names.dat` contains names and address in plain format with simply a blank line between entries. You can, of course, just have the names and addresses in the main file, rather than using `\input` to include them. If the file ends in blank lines, expect problems — sorry! Use your editor...

But there are also other ways off accessing the same system:

1. by having entries like this:

   ```
   \addresslabel{me\\
   here and there\\
   england\\
   }
   ```

   without the `labels` environment.

2. if you have labels in the simple format in a file, just write a `.tex` file like this:

   ```
   \documentstyle{labels}
   \begin{document}
   \labelfile{filename}
   \end{document}
   ```

   and all will be done for you.

3. if you want to *duplicate* the label, there is a counter called `\numberoflabels` which you can set, so

   ```
   \numberoflabels=4
   \addresslabel{Me \\my street
           \\ mytown \\ England}
   ```

   will print the address 4 times in a row

4. For more sophisticated users, there is a macro `\genericlabel` which you can call, with an argument of whatever you want to appear on the label (e.g. for disk labels, etc.). Thus you could have

   ```
   \genericlabel{%
   \begin{tabular}{c}
   \hline
   My Amazing Program\\
   \hline
   Disk 1 of 1
   \hline
   \em We aim to serve\\
   \end{tabular}
   }
   ```

   to produce a label like this:

   ```
   |------------------|
   |My Amazing Program|
   |------------------|
   |Disk 1 of 1       |
   |------------------|
   |We aim to serve   |
   ```

In all modes, you can opt for a frame around each label by setting a Boolean variable called 'framedlabels', e.g.

```
\framedlabelstrue
```

By default you get no frames — I am not sure when you *would* want frames, but who knows.

2 The utility macros

First of all, identify what is happening.

```
\def\fileversion{v4}
\def\filedate{92/1/1}
\immediate\write\sixt@@n{File: 'labels.sty'
\fileversion\space <\filedate> (SPQR)}
```

Now take a copy all of 'article' style to start with, just in case any of it is needed (probably not, but you never know).

```
\input article.sty
```

We will be recording the size of a label, and the dimensions of the grid, so set up variables accordingly.

```
\newdimen\label@width
\newdimen\label@height
\newcount\num@labelcols
\newcount\num@labelrows
\newdimen\left@border
\newdimen\top@border
\newdimen\half@label
\newdimen\area@width
\newsavebox{\this@label}
\newcount\label@number
\newcount\numberoflabels
\newcount\l@so@far
\newif\ifframedlabels
\newif\iffirst@label
\first@labeltrue
\framedlabelsfalse
```

The user will probably need to change the following values to reflect the style of labels in use.

```
\num@labelcols=3
\num@labelrows=8
```

Variables are provided to allow you to force a border on the left edge of labels, in case you do not want to print right to the edge, and at the top; these values will affect every label, of course, so you may need to experiment to get pleasing results. 8mm is the amount my LaserJetIII seems to ignore on the left.

```
\left@border=8mm
\top@border=4mm
```

We need to reset all the dimensions appropriately for an A4 page of labels, and the printer will need to know about A4 as well. Obviously if you use a different page size, you will need to alter things here. Some of these changes may be printer dependent. This should all mean we are actually dealing with the whole bit of paper.

```
\textwidth=210mm
\textheight=297mm
\topmargin=-1in
\headheight=0em
```

```
\headsep=0em
\topskip=0em
\footskip=0em
\footheight=0em
\oddsidemargin=-1in
\evensidemargin=-1in
\pagestyle{empty}
\parindent=0em
\parskip=0pt
```

Now calculate the size of labels simply as a proportion of the page size (if you haven't got that right, this won't work, will it?).

```
\label@width\textwidth\divide\label@width by\num@labelcols
\label@height\textheight\divide\label@height by\num@labelrows
\typeout{Creating labels sized \the\label@width\space by \the\label@height}
\label@number=1
```

It is not usually advisable to make the label printing go right to the edge of the available area, so 'area@width' gives the area that will actually be used for printing; the width is cut down by whatever we gave as 'left@border'. It can always be set to 0 if you have a design that uses the whole label.

```
\area@width=\label@width%
\advance\area@width by -\left@border%
\half@label=\label@height\divide\half@label by 2
\advance\half@label by -\top@border
```

We might want to print the same label several times, so `\sticky@label` will repeat `\make@label` a specified number of times (`\numberoflabels`)

```
\numberoflabels=1%

\def\sticky@label{\l@so@far=0%
\loop\ifnum\l@so@far<\numberoflabels\advance\l@so@far by 1\make@label%
\repeat}
```

The real label-making macro, which assumes the actual text is in a box called `\this@label`. It is vital to make sure spaces are not included at the end of lines in these macros, or all hell breaks loose.

```
\def\make@label{%
\ifframedlabels%
\let\boxing@type\framebox%
\else%
\let\boxing@type\makebox%
\fi%
\boxing@type[\label@width][c]{%
\rule{0pt}{\label@height}%
```

We set a position to half-way up a strut of the height of the label, so forcing text to be the right height and vertically centred.

```
\raisebox{\half@label}[0pt][0pt]{%
\rule{\left@border}{0pt}\usebox{\this@label}}}%
```

We only start a new line if we have printed a row of `\num@labelcols` labels

```
\ifnum\label@number=\num@labelcols%
\endgraf\nointerlineskip%
\label@number=1\else\advance\label@number by 1\fi%
}%
```

Now some macros to allow 'verbatim' names and addresses separated by blank lines. First we need some hackery from Phil Taylor to redefine end of line; define carriage-return to check what the next token is; if its another `^M` then we have a blank line.

```
\catcode `\^^M = \active
```

```
\def ^^M{\futurelet\nexttoken\isitapar}%
\def\isitapar{\ifx^^M\nexttoken\let\action=\new@label%
\else\let\action\start@newline\fi\action}%
```

If we have met a blank line, finish current label and start a new one. swallow pending `^M`, or we will have a blank line at the start of each label

```
\def\new@label{\message{+}\end@@label\start@@label\@gobble}%
```

Otherwise just start a new line

```
\def\start@newline{\expandafter\newline}%
\def\startingtoken{\ifx^^M\firsttoken\let\action=\@gobble\else%
\let\action=\relax\fi\action}%
```

Re-instate the original catcode for carriage-return

```
\catcode '\^^M = 5\relax%
```

Define macros to call at beginning and end of labels, to set things up properly.

```
\def\start@@label{%
\savebox{\this@label}\bgroup\raggedright%
\begin{minipage}{\area@width}%
\catcode '\^^M =\active}%
\def\end@@label{%
\end{minipage}\egroup\sticky@label}%
```

3 User macros

The basic case is a generic macro which takes its argument and puts it out on a label.

```
\def\genericlabel#1{%
\iffirst@label\ifframedlabels%
\advance\label@height by-2\fboxsep%
\advance\label@height by-2\fboxrule%
\half@label\label@height\divide\half@label by 2
\advance\half@label by -\top@border%
\first@labelfalse%
\fi\fi%
\savebox{\this@label}{#1}\sticky@label}
```

For compatibility with an old label style, lines ending in // and marked with `\addresslabel{....}`

```
\def\addresslabel#1{\genericlabel{%
\begin{tabular}{l}#1\end{tabular}}}
```

Now easier environments for verbatim labels. If we want framed labels, we need to adjust the width available to use to allow for the rule width and the gap between box and rule, in both axes. This is doubled up, as it happens on both sides / bottoms. We have to check in case the first `\begin{labels}` has a `^^M` after it or (preferably) is terminated by a `%`

```
\newenvironment{labels}%
{%
\iffirst@label\ifframedlabels%
\advance\area@width by-2\fboxsep%
\advance\area@width by-2\fboxrule%
\advance\label@height by-2\fboxsep%
\advance\label@height by-2\fboxrule%
\half@label\label@height\divide\half@label by 2
\advance\half@label by -\top@border%
\first@labelfalse%
\fi\fi%
\start@@label\futurelet\firsttoken\startingtoken}%
{\end@@label}
```

Even more foolproof: simply take a parameter of file name

```
\def\labelfile#1{\begin{labels}\input#1\end{labels}}
```

or prompt for it:

```
\def\promptlabels{\typein[\labelfilename]{What is the name of the
label file?}
\labelfile{\labelfilename}}
```

4 History and acknowledgements

- v.1 May 9th 1989 simply allowed for `\addresslabel{... \\ ...\\...}`
- v.2 July 15th permitted verbatim style with no explicit end of lines
- v.3 March 1991 made more generic
- v.4 January 1992 checked and made to work with emtex drivers to my satisfaction, and documented to bare-bones level with 'doc' system.

The crucial macros which make the system bearable for mailing lists by redefining end of line came from Phil Taylor; apologies to him for using them in a LaTeX style file!

◇ Sebastian Rahtz
ArchaeoInformatica
12 Cygnet Street
York Y02 1AG
spqr@uk.ac.york.minster

Abstracts

Les Cahiers GUTenberg Contents of Recent Issues

Numéro 13 – juin 1992

Bernard GAULLE, Éditorial : morosité francophone ou récession économique ? [French moodiness or economic recession?]; pp. 1–4

GUTenberg's president provides ample evidence of a strong and dynamic (LA)TeX community, in spite of the low interest shown for the 1991 GUTenberg meeting and other reasons which caused the board to decide to skip GUTenberg'92. While the two meetings seemed to have all the best possible organisation and components, participation didn't seem to match the efforts made. And yet the intensity of work being done on various fronts (e.g., LaTeX3), the increasing number of articles on (LA)TeX in various national magazines and journals, the increase in requests to GUTenberg for information, courses and subscriptions to the *Cahiers*, all show that things are humming. It would therefore seem that the economy, not a downturn in interest (along with perhaps too many TeX meetings), is the most likely cause for reduced participation at recent meetings.

Hans Ed. MEIER, Régles fondamentales de mise en page [Basic rules for page layout]; pp. 5–38

Originally published in 1991 in German, this article as translated has been augmented with comments on French typographic style. Meier, described in the editorial as a retired typographer, discusses in his article design elements not only of