Variable width boxes in LaTeX

Simon Law

Seasoned LaTeX users are familiar with the default box commands: `\makebox`, `\framebox` and `\parbox`. They are the building blocks for page layout, and are commonly used. After all, being able to create boxes allows a typesetter great flexibility in positioning objects on a page. Figure 1 illustrates a simple use of `\parbox`.

Hello brave world Goodbye cruel world

Figure 1: Using `\parbox` to position text

1 Traditional `\parbox`

As you can see, I was able to align the two boxes so that each would be aligned. Looking at the source code in Figure 2, you'll see that I had to manually specify the box widths.

```
\parbox[t]{1cm}{Hello\\brave\\world}
\parbox[b]{1.5cm}{Goodbye\\cruel\\world}
```

Figure 2: `\parbox` source code

Of course, guessing the width of the longest line gets tedious. You can try using `\settolength` on the longest line, but that might change as your text changes.

2 Using `\pbox`

In order to automatically determine the width of the box, we will use the pbox[1] package. It provides the `\pbox` command, which is analogous to the `\mbox` command. In Figure 3, I typeset the same text using `\pbox` instead.

```
\pbox[t]{\textwidth}{Hello\\brave\\world}
\hspace{0.1cm}
\pbox[b]{\textwidth}{Goodbye\\cruel\\world}
```

Figure 3: `\pbox` source code

The syntax for `\pbox` is quite similar to that of `\parbox`. You must provide the maximum width of the box (*max-width*) and the contents (*text*):

`\pbox[`*pos*`][`*height*`][`*inner-pos*`]{`*max-width*`}{`*text*`}`

By default, the centre of each box will be vertically aligned. However, the three optional arguments allow you to align the `\pbox` as necessary. These options work exactly like their `\parbox` counterparts.

3 Now with `minipage`

This works well for simple paragraphs, where environments need not be embedded. However, once you start needing the features of the `minipage` environment, you begin to run into the same problems. David Arseneau has solved this problem with his varwidth[2] package.

An example use would be to centre a `verbatim` environment. This is normally done in a `minipage` because the `verbatim` environment left-flushes all its text against the left margin. In order to use the `minipage`, you still have to figure out the width of its contents and specify it manually.

```
#include <stdio.h>
int main()
{
  printf ("Hello world!\n");
  return 0;
}
```

Figure 4: Centered source code example

Figure 4 shows a snippet of source code that is representative of a sample in an article or a textbook. The code in Figure 5 illustrates how to typeset this without manually determining the width.

```
\centering
\begin{varwidth}{\columnwidth}
\begin{verbatim}
#include␣<stdio.h>
int␣main()
{
␣␣printf␣("Hello␣world!\n");
␣␣return␣0;
}
\end{verbatim}
\end{varwidth}
```

Figure 5: `varwidth` source code

4 Conclusion

Both the pbox and varwidth packages are useful extensions to standard LaTeX 2ε. They allow typesetters to place boxes and minipages throughout their documents without the need for guessing widths.

◇ Simon Law
sfllaw@law.yi.org
http://www.law.yi.org/~sfllaw/

[1] http://www.ctan.org/tex-archive/macros/latex/contrib/pbox/

[2] http://www.ctan.org/tex-archive/macros/latex/contrib/misc/varwidth.sty