Macros

The bag of tricks

Victor Eijkhout

G'day, my friends in TEX. Today I want to address two slightly TEXnical points. The first is the practical issue of how to format macros, the second is in response to comments from readers.

1 How to lay out a macro

If you write macros that will sooner or later be given to other people, you may want to be particularly careful about your style of writing. The way you write a macro should bring out its structure. If you ever wonder 'now, where is the closing brace for this box' in your own macro, consider it an indication that other people will have trouble reading that macro too!

Here are some guidelines, distilled from my experience, and from looking at the style of formatting in *TUGboat*. The examples are all taken from *TUGboat* with spacing and line breaking intact. Although all illustrate the general principle, their style still differs in detail.

1.1 Align for readability

Recognize when things are 'on the same level' and align them vertically. For example,

```
\def\myloop #1 {
  \ifx #1\end\else
    \myoperation #1
    \myloop
  \fi
}
```

or

```
\csname
  \ifx\a\b mid%
  \else
    \ifx\ad\d top\fi
  \fi insert%
\endcsname
```

1.2 Indent for readability

Look for 'levels' in your input, and give the next level more indentation. For instance, indent nested definitions:

```
\gdef\mathsraise
{%
    \begingroup
    \def \@ctivateall
    {%
        \count 0 = 0
        \loop
...
```

or (a particularly neat effect by using four spaces):

```
\def\pullet#1\of#2\to#3{%
    \def#3{\outofrange\}%
...
```

Indent the material in a box:

```
\def\EnvMakeBox#1#2{
 \setbox#1\vbox{
  \parindent0pt
...
```

Indent statements that have to be broken because they are too long:

```
\expandafter\expandafter
   \expandafter#1%
      \expandafter\expandafter
         \expandafter#3\expandafter#3#4
```

Do you wonder about the percent sign after #1 and the lack of one on the last line? The first one is necessary to prevent an unwanted space in the middle of a statement. There is no percent sign necessary at the end of the whole statement because the author apparently knows that this statement will be executed in a place where spaces are ignored. See the next section.

1.3 But what about all those spaces?

I have met people whose experiences with unwanted spaces were so traumatic that they didn't indent their macros any more, and didn't put spaces in them either. This is unnecessary, and sometimes even dangerous. Here are some rules.[1]

- Spaces at the start of a line are irrelevant, and
- a space or a line end after a control sequence is irrelevant. This point and the previous are true unless a macro such as `\obeyspaces` appeared earlier. On the other hand,
- A space or a line end after a control symbol, that is a backslash followed by anything but a letter, for instance `\}`, *is* relevant; this is not true for control space: `\␣`.
- Multiple spaces are almost always equivalent to just one space.
- Spaces are a good idea after a number, but be careful, not all numbers are numbers. For instance, you want a space (or a line end) after

[1] This is no attempt at explanation, and these rules are not 100% accurate either.

`\pageno=13`, `\box37` or `\ifodd42`, but not after `\multispan5`, `\magstep2` (`\multispan` and `\magstep` are macros with one parameter), or `#1` (neither in the definition parameter text or the body of a macro).

There is more to this number business; for instance, after control sequences that expand to numbers you sometimes want to place `\relax` analogous to the space after a genuine number. But this would be taking us too far.

1.4 When is a space not a space?

Sometimes a macro can contain spaces in the input that are relevant, but that you don't care about anyway. This happens if that macro or some part of it will occur in vertical mode. If TEX is in vertical mode it ignores spaces. Thus, if you write

```
\hbox{
 \Title
}
```

you get an unwanted space after the opening brace, but not in

```
\vbox{
 \noindent\Title
}
```

because TEX starts vertical boxes in vertical mode. (Makes sense, doesn't it?) Similarly, often you know that a part of a macro will be in vertical mode:

```
\def\Heading#1{\par
  \hbox{\bf #1}
  \nobreak}
```

After the `\par` TEX is in vertical mode, so the space after the `\hbox` is irrelevant.

In general it is advisable to put comment signs at line ends only if a space there can find its way into the output. If the space is guaranteed to disappear, as in the above examples, an extra comment sign would only confuse the readers who understand this point, whereas the readers who are (blissfully!) unaware of it will not be bothered by its absence.

(A bonus remark for advanced TEX users: most spaces in output routines are also irrelevant.)

2 Skip that question!

In a previous issue of *TUGboat* I wrote about conditionals such as

```
\def\ifUndefinedCS#1{%
  \expandafter\ifx
    \csname#1\endcsname\relax}
```

There is a problem with such home-made conditionals: TEX treats them as ordinary macros, so if you try to nest them in another conditional as

```
\ifnum ...
  \ifUndefinedCS{...}
  \else ...
  \fi
\fi
```

TEX will get confused if the outer conditional is false: it matches that conditional with the first `\else` or `\fi` that it finds.

Here's a way out, proposed by the big K himself, and used extensively in Stephan von Bechtolsheim's forthcoming book *TEX in Practice*. Define

```
\def\UndefinedCS#1{00\fi
  \expandafter\ifx
    \csname#1\endcsname\relax}
```

and use it as

```
\if\UndefinedCS{...}
```

Now TEX sees a conditional, and this conditional is matched up correctly if you nest it.

And that's it for this time. `\relax` and hang on to your towel!

◇ Victor Eijkhout
Department of Computer Science
University of Tennessee at
Knoxville
Knoxville TN 37996-1301
Internet: `eijkhout@cs.utk.edu`

Too Many Errors

Jonathan Fine

One of the attractions of TEX the program is that it is substantially without error. Sadly, the same is not always true for macros written for TEX. I would like to see a regular column in *TUGboat* where errors discovered in previous issues are reported. Perhaps a small prize could be given each year to the person who finds the most. Here is my contribution for the June 1991 issue (vol. 12, no. 2).

Some tools etc.: Part I—Lincoln Durst (p.248–252) Here, the error is slightly complicated to explain. The macro `\Pageno` (p.252) is intended to help write zero-padded page numbers (001, ..., 009, 010, ..., 999) to an index file. Its use is within a context equivalent to

```
\edef\next{\write\inx{\Pageno}}
```