

```

* * * * *
  Warnings & Limitations
* * * * *

```

\relax and Watch the Numbers

Thanks to the T_EXhax participants and Mike Spivak for these ideas.

When T_EX encounters a ⟨number⟩, it will keep parsing until it recognizes a non-number. This can lead to some unexpected situations.

Mr. Ogawa (Ogawa%SLACVM.BitNet@Berkeley) thought he had a problem with the @ character:

```

% UNLOCK TEST - test TeX unlock feature
% this character can now be used
%                               in macro names

```

```

\def\unlock{%
  \catcode\@=11%
}%
\unlock\rOggedbottomtrue\bye

```

yielded

```

This is TeX, VM/CMS Version 1.0 [SLAC]
(preloaded format=plain 84.2.17)
      19 APR 1984 21:07

```

```

**&PLAIN unlock.test
(unlock.test
! Undefined control sequence.
1.5 \unlock\r
      Oggedbottomtrue\bye

```

?

[1]

```

Output written on unlock.dvi
      (1 page, 224 bytes).

```

A slight modification gave the result he originally expected:

```

\def\unlock{%
  \catcode\@=11%
  \relax
}%
\unlock\rOggedbottomtrue\bye

```

```

This is TeX, VM/CMS Version 1.0 [SLAC]
(preloaded format=plain 84.2.17)
      19 APR 1984 21:13

```

```

**&PLAIN unlock.test
(unlock.test
No pages of output.

```

A different variation on the second line would also have done the job:

```

\catcode\@=11 %

```

(Note that the problem surfaces with the expansion, not with the definition, of \unlock, so one could also

avoid the problem by modifying the last line to say \unlock\relax\rOggedbottomtrue...)

Arthur Keller offers the following explanation: “When T_EX sees the “\catcode\@=11%” it is still parsing a number. For example, you could have said

```

\def\unlock{%
  \catcode\@=1}%
\unlock1\rOggedbottomtrue\bye

```

This would mean \catcode\@=11 (the two ones are put together). The \relax causes the mouth to realize that it has completed parsing a number and that allows it to pass the mess to the stomach, and the \catcode changes. In general, you should use \relax to terminate such parameters when it isn't self delimiting (such as 5pt).”

A related situation has been pointed out by Mike Spivak:

```

\if ... \advance\foo by \the\fam \fi

```

Since \the\fam expands to a number, T_EX will absorb the \fi looking for the end of the number, and if digits follow \fi they will be included in the value by which \foo is \advanced.

For the official explanation, see *The T_EXbook*, pages 208 and 269–270. There it is recommended that “for best results, *always put a blank space after a numeric constant*; this blank space tells T_EX that the constant is complete, and such a space will never ‘get through’ to the output.” But only one space, and note the comment on aesthetics vs. efficiency.

A final word from Arthur Keller: “You should be very careful to avoid extra spaces in macros lest they creep into the output when in horizontal mode. Using \relax to delimit parameters is consequently much safer.”

Barbara Beeton