The Stata Journal (2008)
8, Number 4, pp. 586–587

Stata tip 70: Beware the evaluating equal sign

Nicholas J. Cox
Department of Geography
Durham University
Durham, UK
n.j.cox@durham.ac.uk

When you assign to a local or global macro the result of evaluating an expression, you can lose content because of limits on the length of string expression that Stata will handle. This tip explains this pitfall and the way around it in detail. The problem is described in [U] **18.3.4 Macros and expressions**, but it occurs often enough that another warning may be helpful. Illustrations will be in terms of local macros, but the warning applies equally to global macros.

What is the difference between the following two statements?

```
. local foo1 "just 23 characters long"
. local foo2 = "just 23 characters long"
```

Let's look at the results:

```
. display "`foo1´"
just 23 characters long
. display "`foo2´"
just 23 characters long
```

The answer is nothing, in terms of results. Do not let that deceive you. Two quite different processes are involved.

The first statement defines local macro `foo1` by copying the string `just 23 characters long` into it. The second statement does more. It first evaluates the string expression to the right of the equal sign and then defines the local macro by assigning the result of that evaluation to it.

The mantra to repeat is "the equal sign implies evaluation". Here the evaluation makes no difference to the result, but that will be not true in other cases. A more treacherous pitfall lies ahead.

It should be clear that we often need an evaluation. If the statements had been

```
. local bar1 lower("FROG")
. local bar2 = lower("FROG")
```

the results would have been quite different:

```
. display "`bar1´"
lower("FROG")
. display "`bar2´"
frog
```


The first just copies what is on the right to what is named on the left, the local macro `bar1`. The second does the evaluation, and then the result is put in the local macro.

Whenever you need an evaluation, you should be aware of a potential limit. `help limits` gives you information on limits applying to your Stata. As seen in the list, there are many different limits, and they can vary between Stata version and Stata flavor; there is no point in echoing that list here. But at the time of writing, the limit for Stata 10.1 on the length of a string in a string expression in all flavors is 244 characters, which is quite a modest number.

The implication is that whenever you have a choice between copying and evaluation, always use copying. And if you must use evaluation, make sure that no single evaluation is affected by the limit of 244 characters. Further, be aware of work-arounds. Thus the `length()` function cannot report string lengths above 245 (yes, 245) characters, but the extended function `: length local` can measure much longer strings. See `help extended_fcn` for more details.

If you do not pay attention to the limit, you may run into problems. First, your local macro will be shorter than you want. Second, it may not even make sense to whatever you feed it to. Any error message you see will then not refer to the real underlying error, a truncated string. So your bug may be elusive.

For example, users sometimes collect variable names (or other names) in a list by using a loop centered on something like this:

```
. local mylist = "`mylist´ `newitem´"
```

This may seem like self-evidently clear and correct code. But as `mylist` grows beyond 244 characters, the result of the evaluation entailed by `=` can only be to truncate the list, with possibly mysterious consequences.