The Stata Journal (*yyyy*) *ii*, Number *nn*, pp. 1–3

Stata tip 124: Passing temporary variables to subprograms

Maarten L. Buis
University of Konstanz
Department of History and Sociology
Konstanz, Germany
maarten.buis@uni-konstanz.de

A useful tool when programming in Stata is the temporary variable, which can be created using the `tempvar` command, see [P] **macro**. when it is convenient to store intermediate steps in a temporary variable, the `tempvar` will reserves a variable name for that temporary variable that is guaranteed not to exist in your current dataset. That way your program will not accidentally overwrite an already existing variable. The `tempvar` command will also ensure that that temporary variable will be removed once the program that created it is finished. That way your program will not clutter the user's dataset with unwanted intermediate results. Similarly, one can create temporary scalars and matrices with the `tempname` command (also see: [P] **macro**). Another useful tool when programming in Stata is to break up larger programs in a number of smaller sub-routines. This is a way to keep longer programs organized, making it easier to write, debug, certify and maintain them. Sometimes the creation of temporary results in a temporary variable is a good candidate for such a sub-routine. If we use `tempvar` or `tempname` in that sub-routine, the temporary variable, scalar or matrix will be deleted as soon as the sub-routine is finished. So that is, in this case, not what we want.

To use the temporary objects created or changed in subroutines in the main program we need to use `tempvar` or `tempname` in the main program and pass that name to the sub-routine. Consider the example below:

```
. clear all
. set seed 1234567
. program define mainprog
  1.         tempvar random
  2.         qui gen `random´ = .
  3.         tempname mean
  4.         scalar `mean´ = 2
  5.         subprog, random(`random´) mean(`mean´)
  6.         sum `random´
  7. end
.
. program define subprog
  1.         syntax, random(name) mean(name)
  2.         qui replace `random´ = rnormal(`mean´)
  3. end
.
. sysuse auto, clear
(1978 Automobile Data)
. mainprog
    Variable |       Obs        Mean    Std. Dev.       Min        Max
```



```
    __000000 |         74    2.150455    .9593332  -.5362754   4.171665
```

In line 1 of `mainprog` a variable name was chosen that does not exist in the current data, and this variable name is stored in the local macro `‘random’`. In line 2 this name was used to create a variable. In lines 3 and 4 a temporary scalar `‘mean’` is created. In line 5 the names of the temporary variable and the temporary scalar are passed on to `subprog` in the options `random()` and `mean()`. Notice, that when `subprog` runs `mainprog` is not yet finished, so variables created with `tempvar` and matrices and scalars created with `tempname` still exist. Line 1 of `subprog` means that `subprog` expects two options containing a name, and that that name will be stored in the local macros `‘random’` and `‘mean’`. Line 2 of `subprog` than changes the temporary variable using the temporary scalar. Now we go back to line 6 of `mainprog`, which uses that changed temporary variable. After that `mainprog` ends and the temporary variable `‘random’` and temporary scalar `‘mean’` are deleted.

The same logic can also be used to pass temporary variables, matrices and scalars to Mata functions: As long as the program that created them has not finished the objects exist. All you need to do to pass them on, is to pass their names to the Mata function. For example, the program below does the same thing as the example above, except that it uses Mata for the subroutine.

```
. clear all
. mata
------------------------------------------------- mata (type end to exit) -----
: void mata_subprog(
>     string scalar randomname,
>     string scalar meanname) {
>
>     st_view(random=., ., randomname)
>     mean = st_numscalar(meanname)
>
>     random[.,.] = rnormal(st_nobs(),1,mean,1)
> }
: end
--------------------------------------------------------------------------------

.
. program define mainprog
  1.         tempvar random
  2.         qui gen `random´ = .
  3.         tempname mean
  4.         scalar `mean´ = 2
  5.         mata: mata_subprog("`random´", "`mean´")
  6.         sum `random´
  7. end
.
. sysuse auto, clear
(1978 Automobile Data)
. mainprog
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
```

```
__000000 |        74    2.261774    1.051417  -.1316733   4.615112
```