Title

stata.com

exit — Exit from a program or do-file

Description Syntax Options Remarks and examples Also see

Description

`exit`, when typed from the keyboard, causes Stata to terminate processing and returns control to the operating system. If the dataset in memory has changed since the last `save` command, you must specify the `clear` option before Stata will let you leave. Use of the command in this way is discussed in [R] **exit**.

More generally, `exit` causes Stata to terminate the current process and returns control to the calling process. The return code is set to the value of the expression or to zero if no expression is specified. Thus `exit` can be used to exit a program or do-file and return control to Stata. With an option, `exit` can even be used to exit Stata from a program or do-file. Such use of `exit` is the subject of this entry.

Syntax

`exit` [[`=`]*exp*] [`, clear STATA`]

Options

`clear` permits you to exit, even if the current dataset has not been saved.

`STATA` exits Stata and returns control to the operating system, even when given from a do-file or program. The `STATA` option is implied when `exit` is issued from the keyboard.

Remarks and examples

stata.com

`exit` can be used at the terminal, from do-files, or from programs. From the terminal, it allows you to leave Stata. Given from a do-file or program without the `STATA` option, it causes the do-file or program to terminate and return control to the calling process, which might be the keyboard or another do-file or program.

Caution should be used if `exit` is included to break execution within a loop. A more suitable command is `continue` or `continue, break`; see [P] **continue**. `continue` is used to explicitly break execution of the current loop iteration with execution resuming at the top of the loop unless the `break` option is specified, in which case execution resumes with the command following the looping command.

When using `exit` to force termination of a program or do-file, you may specify an expression following the `exit`, and the resulting value of that expression will be used to set the return code. Not specifying an expression is equivalent to specifying `exit 0`.

▷ Example 1

Here is a useless program that will tell you whether a variable exists:

```
. program check
  1. capture confirm variable `1'
  2. if _rc!=0 {
  3.     display "`1' not found"
  4.     exit
  5. }
  6. display "The variable `1' exists."
  7. end
. check median_age
The variable median_age exists.
. check age
age not found
```

`exit` did not close Stata and cause a return to the operating system; it instead terminated the program.

◁

▷ Example 2

You type `exit` from the keyboard to leave Stata and return to the operating system. If the dataset in memory has changed since the last time it was saved, however, Stata will refuse. At that point, you can either `save` the data and then `exit` or type `exit, clear`:

```
. exit
no; data in memory would be lost
r(4);
. exit, clear
```

(*Operating system prompts you for next command*)

◁

❑ Technical note

You can also exit Stata and return to the operating system from a do-file or program by including the line `exit, STATA` in your do-file or program. To return to the operating system regardless of whether the dataset in memory has changed, you include the line `exit, STATA clear`.

❑

Also see

[P] **capture** — Capture return code

[P] **class exit** — Exit class-member program and return result

[P] **continue** — Break out of loops

[P] **error** — Display generic error message and exit

[R] **error messages** — Error messages and return codes

[R] **exit** — Exit Stata