

**timer** — Time sections of code by recording and reporting time spent

[Description](#)[Syntax](#)[Remarks and examples](#)[Stored results](#)[Also see](#)

## Description

`timer` starts, stops, and reports up to 100 interval timers. Results are reported in seconds.

`timer clear` resets timers to zero.

`timer on` begins a timing. `timer off` stops a timing. A timing may be turned on and off repeatedly without clearing, which causes the timer to accumulate.

`timer list` lists the timings. If `#` is not specified, timers that contain zero are not listed.

## Syntax

*Reset timers to zero*

```
timer clear [#]
```

*Turn a timer on*

```
timer on #
```

*Turn a timer off*

```
timer off #
```

*List the timings*

```
timer list [#]
```

where `#` is an integer, 1–100.

## Remarks and examples

stata.com

`timer` can be used to time sections of code. For instance,

```
program tester
  version ...
  timer clear 1
  forvalues repeat=1(1)100 {
    timer on 1
    mycmd ...
    timer off 1
  }
  timer list 1
end
```

## Stored results

`timer list` stores the following in `r()`:

Scalars

<code>r(t1)</code>	value of first timer
<code>r(nt1)</code>	# of times turned on and off
<code>r(t2)</code>	value of second timer
<code>r(nt2)</code>	# of times turned on and off
.	
.	
.	
<code>r(t100)</code>	value of 100th timer
<code>r(nt100)</code>	# of times turned on and off

Only values for which `r(nt#)`  $\neq 0$  are stored.

`r()` results produced by other commands are not cleared.

## Also see

[P] [rmsg](#) — Return messages