source or dot operator Man Page

Archived: 2026-04-05 17:52:52 UTC

Read and execute commands from the *filename* argument in the current shell context.

```
Syntax
      . filename [arguments]

      source filename [arguments]
```

source is a synonym for dot/period '.' in bash, but not in POSIX sh, so for maximum compatibility use the period.

When a script is run using source it runs within the existing shell, any variables created or modified by the script will remain available after the script completes. In contrast if the script is run just as *filename*, then a separate subshell (with a completely separate set of variables) would be spawned to run the script.

There is a subtle difference between executing a script by running .ss64script (dot ss64script) and . ss64script (dot space ss64script)
the first is running a file thats been hidden from the 'ls' command, (although ls -a will show hidden files) the second option will execute ss64script even if it has not been set as an executable with chmod.

Unless you provide an exact path to *filename* then bash will look first via the `PATH` variable and then in the current directory (only if *filename* is not found in `$PATH`.) If any *arguments* are supplied, they become the positional parameters when *filename* is executed. Otherwise the positional parameters are unchanged.

When a script is run using 'source' it runs within the existing shell, any variables created or modified by the script will remain available after the script completes.

If a script is run just as ss64script, then a separate subshell (with a separate set of variables) will be spawned to run the script.

Other ways that the bash shell can interpret a dot/period:
A dot can represent the current directory ("./filename")
In a regular expression, "." will match any single character, (not zero or more characters.)

This is a BASH shell builtin, to display your local syntax from the bash prompt type: help [s]ource

The return status is the exit status of the last command executed, or zero if no commands are executed. If *filename* is not found, or cannot be read, the return status is non-zero.

This is a BASH shell builtin, to display your local syntax from the bash prompt type: help source

Examples

```
$ source ~/demoscript
```

“Although the world is full of suffering, it is also full of the overcoming of it” ~ Helen Keller

Related Linux commands

command - Run a command - ignoring shell functions.
builtin - Run a shell builtin.
crontab - Schedule a command to run at a later time.
chroot - Run a command with a different root directory.
exec - Execute a command.
if - Conditionally perform a command.
nohup - Run a command immune to hangups.
su - Run a command with substitute user and group id.
type - Describe a command.
watch - Execute/display a program periodically.
Equivalent Windows command: SETLOCAL - Set options to control the visibility of variables.
