Rocky Enterprise Linux 9.2 Manual Pages on command 'script.1'

$ man script.1

SCRIPT(1) User Commands SCRIPT(1)

NAME

script - make typescript of terminal session

SYNOPSIS

script [options] [file]

DESCRIPTION

script makes a typescript of everything on your terminal session. The terminal data are stored in raw form to the log file and information about timing to another (optional) structured log file. The timing log file is necessary to replay the session later by scriptreplay(1) and to store additional information about the session.

Since version 2.35, script supports multiple streams and allows the logging of input and output to separate files or all the one file. This version also supports new timing file which records additional information. The command scriptreplay --summary then provides all the information.

If the argument file or option --log-out file is given, script saves the dialogue in this file. If no filename is given, the dialogue is

saved in the file typescript.

Note that logging input using --log-in or --log-io may record security-sensitive information as the log file contains all terminal session input (e.g., passwords) independently of the terminal echo flag setting.

OPTIONS

Below, the size argument may be followed by the multiplicative suffixes KiB (=1024), MiB (=1024*1024), and so on for GiB, TiB, PiB, EiB, ZiB and YiB (the "iB" is optional, e.g., "K" has the same meaning as "KiB"), or the suffixes KB (=1000), MB (=1000*1000), and so on for GB, TB, PB, EB, ZB and YB.

-a, --append

Append the output to file or to typescript, retaining the prior contents.

-c, --command command

Run the command rather than an interactive shell. This makes it easy for a script to capture the output of a program that behaves differently when its stdout is not a tty.

-E, --echo when

This option controls the ECHO flag for the slave end of the session?s pseudoterminal. The supported modes are always, never, or auto.

The default is auto ? in this case, ECHO enabled for the pseudoterminal slave; if the current standard input is a terminal, ECHO is disabled for it to prevent double echo; if the current standard input is not a terminal (for example pipe: echo date | script) then keeping ECHO enabled for the pseudoterminal slave enables the standard input data to be viewed on screen while being recorded to session log simultaneously.

Note that 'never' mode affects content of the session output log, because users input is not repeated on output.

-e, --return

Return the exit status of the child process. Uses the same format

as bash termination on signal termination (i.e., exit status is 128 + the signal number). The exit status of the child process is always stored in the type script file too.

-f, --flush

Flush output after each write. This is nice for telecooperation: one person does mkfifo foo; script -f foo, and another can supervise in real-time what is being done using cat foo. Note that flush has an impact on performance; it?s possible to use SIGUSR1 to flush logs on demand.

--force

Allow the default output file typescript to be a hard or symbolic link. The command will follow a symbolic link.

-B, --log-io file

Log input and output to the same file. Note, this option makes sense only if --log-timing is also specified, otherwise it?s impossible to separate output and input streams from the log file.

-I, --log-in file

Log input to the file. The log output is disabled if only --log-in specified.

Use this logging functionality carefully as it logs all input, including input when terminal has disabled echo flag (for example, password inputs).

-O, --log-out file

Log output to the file. The default is to log output to the file with name typescript if the option --log-out or --log-in is not given. The log output is disabled if only --log-in specified.

-T, --log-timing file

Log timing information to the file. Two timing file formats are supported now. The classic format is used when only one stream (input or output) logging is enabled. The multi-stream format is used on --log-io or when --log-in and --log-out are used together. See also --logging-format.

-m, --logging-format format

Force use of advanced or classic format. The default is the classic format to log only output and the advanced format when input as well as output logging is requested.

Classic format

The log contains two fields, separated by a space. The first field indicates how much time elapsed since the previous output. The second field indicates how many characters were output this time.

Advanced (multi-stream) format

The first field is an entry type identifier ('I?nput, 'O?utput, 'H?eader, 'S?ignal). The socond field is how much time elapsed since the previous entry, and the rest of the entry is type-specific data.

-o, --output-limit size

Limit the size of the typescript and timing files to size and stop the child process after this size is exceeded. The calculated file size does not include the start and done messages that the script command prepends and appends to the child process output. Due to buffering, the resulting output file might be larger than the specified value.

-q, --quiet

Be quiet (do not write start and done messages to standard output).

-t[file], --timing[=file]

Output timing data to standard error, or to file when given. This option is deprecated in favour of --log-timing where the file argument is not optional.

-V, --version

Display version information and exit.

-h, --help

Display help text and exit.

SIGNALS

Upon receiving SIGUSR1, script immediately flushes the output files.

ENVIRONMENT

The following environment variable is utilized by script:

SHELL

If the variable SHELL exists, the shell forked by script will be that shell. If SHELL is not set, the Bourne shell is assumed. (Most shells set this variable automatically).

NOTES

The script ends when the forked shell exits (a control-D for the Bourne shell (sh(1p)), and exit, logout or control-d (if ignoreeof is not set) for the C-shell, csh(1)).

Certain interactive commands, such as vi(1), create garbage in the typescript file. script works best with commands that do not manipulate the screen, the results are meant to emulate a hardcopy terminal.

It is not recommended to run script in non-interactive shells. The inner shell of script is always interactive, and this could lead to unexpected results. If you use script in the shell initialization file, you have to avoid entering an infinite loop. You can use for example the .profile file, which is read by login shells only:

```
if test -t 0 ; then
    script
    exit
fi
```

You should also avoid use of script in command pipes, as script can read more input than you would expect.

HISTORY

The script command appeared in 3.0BSD.

BUGS

script places everything in the log file, including linefeeds and backspaces. This is not what the naive user expects.

script is primarily designed for interactive terminal sessions. When stdin is not a terminal (for example: echo foo | script), then the session can hang, because the interactive shell within the script session misses EOF and script has no clue when to close the session. See the NOTES section for more information.

SEE ALSO

csh(1) (for the history mechanism), scriptreplay(1), scriptlive(1)

REPORTING BUGS

For bug reports, use the issue tracker at

https://github.com/karelzak/util-linux/issues.

AVAILABILITY

The script command is part of the util-linux package which can be

downloaded from Linux Kernel Archive

<https://www.kernel.org/pub/linux/utils/util-linux/>.