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

C:\>man tset.1

tset(1) General Commands Manual tset(1)

NAME

tset, reset - terminal initialization

SYNOPSIS

tset [-IQVcqrsw] [-] [-e ch] [-i ch] [-k ch] [-m mapping] [terminal]

reset [-IQVcqrsw] [-] [-e ch] [-i ch] [-k ch] [-m mapping] [terminal]

DESCRIPTION

tset - initialization

This program initializes terminals.

First, tset retrieves the current terminal mode settings for your terminal. It does this by successively testing

? the standard error,

? standard output,

? standard input and

? ultimately ?/dev/tty?

to obtain terminal settings. Having retrieved these settings, tset remembers which file descriptor to use when updating settings.

Next, tset determines the type of terminal that you are using. This determination is done as follows, using the first terminal type found.

1. The terminal argument specified on the command line.

2. The value of the TERM environmental variable.

3. (BSD systems only.) The terminal type associated with the standard error output

device in the /etc/ttys file. (On System-V-like UNIXes and systems using that con? vention, getty does this job by setting TERM according to the type passed to it by /etc/inittab.)

4. The default terminal type, ?unknown?.

If the terminal type was not specified on the command-line, the -m option mappings are then applied (see the section TERMINAL TYPE MAPPING for more information). Then, if the terminal type begins with a question mark (???), the user is prompted for confirmation of the terminal type. An empty response confirms the type, or, another type can be entered to specify a new type. Once the terminal type has been determined, the terminal description for the terminal is retrieved. If no terminal description is found for the type, the user is prompted for another terminal type.

Once the terminal description is retrieved,

? if the ?-w? option is enabled, tset may update the terminal's window size. If the window size cannot be obtained from the operating system, but the termi? nal description (or environment, e.g., LINES and COLUMNS variables specify this), use this to set the operating system's notion of the window size.

? if the ?-c? option is enabled, the backspace, interrupt and line kill charac? ters (among many other things) are set

? unless the ?-I? option is enabled, the terminal and tab initialization strings are sent to the standard error output, and tset waits one second (in case a hardware reset was issued).

? Finally, if the erase, interrupt and line kill characters have changed, or are not set to their default values, their values are displayed to the standard er? ror output.

reset - reinitialization

When invoked as reset, tset sets the terminal modes to ?sane? values:

? sets cooked and echo modes,

? turns off cbreak and raw modes,

? turns on newline translation and

? resets any unset special characters to their default values

before doing the terminal initialization described above. Also, rather than using the terminal initialization strings, it uses the terminal reset strings.

The reset command is useful after a program dies leaving a terminal in an abnormal

state:

? you may have to type

<LF>reset<LF>

(the line-feed character is normally control-J) to get the terminal to work, as carriage-return may no longer work in the abnormal state.

? Also, the terminal will often not echo the command.

OPTIONS

The options are as follows:

-c Set control characters and modes.

-e Set the erase character to ch.

-I Do not send the terminal or tab initialization strings to the terminal.

-i Set the interrupt character to ch.

-k Set the line kill character to ch.

-m Specify a mapping from a port type to a terminal. See the section TERMINAL TYPE MAPPING for more information.

-Q Do not display any values for the erase, interrupt and line kill characters. Normally tset displays the values for control characters which differ from the system's default values.

-q The terminal type is displayed to the standard output, and the terminal is not initialized in any way. The option ?-? by itself is equivalent but archaic.

-r Print the terminal type to the standard error output.

-s Print the sequence of shell commands to initialize the environment variable TERM to the standard output. See the section SETTING THE ENVIRONMENT for de? tails.

-V reports the version of ncurses which was used in this program, and exits.

-w Resize the window to match the size deduced via setupterm(3X). Normally this has no effect, unless setupterm is not able to detect the window size.

The arguments for the -e, -i, and -k options may either be entered as actual char? acters or by using the ?hat? notation, i.e., control-h may be specified as ?^H? or ?^h?.

If neither -c or -w is given, both options are assumed.

SETTING THE ENVIRONMENT

It is often desirable to enter the terminal type and information about the termi?

nal's capabilities into the shell's environment. This is done using the -s option.

When the -s option is specified, the commands to enter the information into the shell's environment are written to the standard output. If the SHELL environmental variable ends in ?csh?, the commands are for csh, otherwise, they are for sh. Note, the csh commands set and unset the shell variable noglob, leaving it unset. The following line in the .login or .profile files will initialize the environment correctly:

eval `tset -s options ... `

TERMINAL TYPE MAPPING

When the terminal is not hardwired into the system (or the current system informa? tion is incorrect) the terminal type derived from the /etc/ttys file or the TERM environmental variable is often something generic like network, dialup, or unknown. When tset is used in a startup script it is often desirable to provide information about the type of terminal used on such ports.

The -m options maps from some set of conditions to a terminal type, that is, to tell tset ?If I'm on this port at a particular speed, guess that I'm on that kind of terminal?.

The argument to the -m option consists of an optional port type, an optional opera? tor, an optional baud rate specification, an optional colon (?:?) character and a terminal type. The port type is a string (delimited by either the operator or the colon character). The operator may be any combination of ?>?, ?<?, ?@?, and ?!?; ?>? means greater than, ?<? means less than, ?@? means equal to and ?!? inverts the sense of the test. The baud rate is specified as a number and is compared with the speed of the standard error output (which should be the control terminal). The terminal type is a string.

If the terminal type is not specified on the command line, the -m mappings are ap? plied to the terminal type. If the port type and baud rate match the mapping, the terminal type specified in the mapping replaces the current type. If more than one mapping is specified, the first applicable mapping is used.

For example, consider the following mapping: dialup>9600:vt100. The port type is dialup , the operator is >, the baud rate specification is 9600, and the terminal type is vt100. The result of this mapping is to specify that if the terminal type is dialup, and the baud rate is greater than 9600 baud, a terminal type of vt100

will be used.

If no baud rate is specified, the terminal type will match any baud rate. If no port type is specified, the terminal type will match any port type. For example, -m dialup:vt100 -m :?xterm will cause any dialup port, regardless of baud rate, to match the terminal type vt100, and any non-dialup port type to match the terminal type ?xterm. Note, because of the leading question mark, the user will be queried on a default port as to whether they are actually using an xterm terminal.

No whitespace characters are permitted in the -m option argument. Also, to avoid problems with meta-characters, it is suggested that the entire -m option argument be placed within single quote characters, and that csh users insert a backslash character (?\?) before any exclamation marks (?!?).

HISTORY

A reset command appeared in 2BSD (April 1979), written by Kurt Shoens. This pro? gram set the erase and kill characters to ^H (backspace) and @ respectively. Mark Horton improved that in 3BSD (October 1979), adding intr, quit, start/stop and eof characters as well as changing the program to avoid modifying any user settings.

Later in 4.1BSD (December 1980), Mark Horton added a call to the tset program using the -I and -Q options, i.e., using that to improve the terminal modes. With those options, that version of reset did not use the termcap database.

A separate tset command was provided in 2BSD by Eric Allman. While the oldest pub? lished source (from 1979) provides both tset and reset, Allman's comments in the 2BSD source code indicate that he began work in October 1977, continuing develop? ment over the next few years.

In September 1980, Eric Allman modified tset, adding the code from the existing ?reset? feature when tset was invoked as reset. Rather than simply copying the ex? isting program, in this merged version, tset used the termcap database to do addi? tional (re)initialization of the terminal. This version appeared in 4.1cBSD, late in 1982.

Other developers (e.g., Keith Bostic and Jim Bloom) continued to modify tset until 4.4BSD was released in 1993.

The ncurses implementation was lightly adapted from the 4.4BSD sources for a ter? minfo environment by Eric S. Raymond <esr@snark.thyrsus.com>.

COMPATIBILITY

Neither IEEE Std 1003.1/The Open Group Base Specifications Issue 7 (POSIX.1-2008) nor X/Open Curses Issue 7 documents tset or reset.

The AT&T tput utility (AIX, HPUX, Solaris) incorporated the terminal-mode manipula? tion as well as termcap-based features such as resetting tabstops from tset in BSD (4.1c), presumably with the intention of making tset obsolete. However, each of those systems still provides tset. In fact, the commonly-used reset utility is al? ways an alias for tset.

The tset utility provides for backward-compatibility with BSD environments (under most modern UNIXes, /etc/inittab and getty(8) can set TERM appropriately for each dial-up line; this obviates what was tset's most important use). This implementa? tion behaves like 4.4BSD tset, with a few exceptions specified here.

A few options are different because the TERMCAP variable is no longer supported un? der terminfo-based ncurses:

? The -S option of BSD tset no longer works; it prints an error message to the standard error and dies.

? The -s option only sets TERM, not TERMCAP.

There was an undocumented 4.4BSD feature that invoking tset via a link named ?TSET? (or via any other name beginning with an upper-case letter) set the terminal to use upper-case only. This feature has been omitted.

The -A, -E, -h, -u and -v options were deleted from the tset utility in 4.4BSD. None of them were documented in 4.3BSD and all are of limited utility at best. The -a, -d, and -p options are similarly not documented or useful, but were retained as they appear to be in widespread use. It is strongly recommended that any usage of these three options be changed to use the -m option instead. The -a, -d, and -p options are therefore omitted from the usage summary above.

Very old systems, e.g., 3BSD, used a different terminal driver which was replaced in 4BSD in the early 1980s. To accommodate these older systems, the 4BSD tset pro? vided a -n option to specify that the new terminal driver should be used. This im? plementation does not provide that choice.

It is still permissible to specify the -e, -i, and -k options without arguments, although it is strongly recommended that such usage be fixed to explicitly specify the character.

As of 4.4BSD, executing tset as reset no longer implies the -Q option. Also, the

interaction between the - option and the terminal argument in some historic imple? mentations of tset has been removed.

The -c and -w options are not found in earlier implementations. However, a differ? ent window size-change feature was provided in 4.4BSD.

? In 4.4BSD, tset uses the window size from the termcap description to set the window size if tset is not able to obtain the window size from the operating system.

? In ncurses, tset obtains the window size using setupterm, which may be from the operating system, the LINES and COLUMNS environment variables or the terminal description.

Obtaining the window size from the terminal description is common to both implemen? tations, but considered obsolescent. Its only practical use is for hardware termi? nals. Generally speaking, a window size would be unset only if there were some problem obtaining the value from the operating system (and setupterm would still fail). For that reason, the LINES and COLUMNS environment variables may be useful for working around window-size problems. Those have the drawback that if the win? dow is resized, those variables must be recomputed and reassigned. To do this more easily, use the resize(1) program.

ENVIRONMENT

The tset command uses these environment variables:

SHELL

tells tset whether to initialize TERM using sh or csh syntax.

TERM Denotes your terminal type. Each terminal type is distinct, though many are similar.

TERMCAP

may denote the location of a termcap database. If it is not an absolute path? name, e.g., begins with a ?/?, tset removes the variable from the environment before looking for the terminal description.

FILES

/etc/ttys

system port name to terminal type mapping database (BSD versions only).

/etc/terminfo

terminal capability database

SEE ALSO

csh(1), sh(1), stty(1), curs_terminfo(3X), tty(4), terminfo(5), ttys(5), environ(7)

This describes ncurses version 6.2 (patch 20200212).