NAME

sed - a Stream EDitor

SYNOPSIS

sed [-n] [-V] [--quiet] [--silent] [--version] [--help]
[-e script] [--expression=script]
[-f script-file] [--file=script-file]
[script-if-no-other-script]
[file...]

DESCRIPTION

Sed is a stream editor. A stream editor is used to perform basic text transformations on an input stream (a file or input from a pipeline). While in some ways similar to an editor which permits scripted edits (such as ed), sed works by making only one pass over the input(s), and is consequently more efficient. But it is sed's ability to filter text in a pipeline which particularly distinguishes it from other types of editors.

OPTIONS

Sed may be invoked with the following command-line options:

-V

--version
Print out the version of sed that is being run and a copyright notice, then exit.

-h

--help
Print a usage message briefly summarizing these command-line options and the bug-reporting address, then exit.

-n

--quiet

--silent
By default, sed will print out the pattern space at the end of each cycle through the script. These options disable this automatic printing, and sed will only produce output when explicitly told to via the p command.

-e script

--expression=script
Add the commands in script to the set of commands to be run while processing the input.

-f _s_c_r_i_p_t_-_f_i_l_e

--file=_s_c_r_i_p_t_-_f_i_l_e
Add the commands contained in the file _s_c_r_i_p_t_-_f_i_l_e to the set of commands to be run while processing the input.

If no -e,-f,--expression, or --file options are given on the command-line, then the first non-option argument on the command line is taken to be the _s_c_r_i_p_t to be executed.

If any command-line parameters remain after processing the above, these parameters are interpreted as the names of input files to be processed. A file name of - refers to the standard input stream. The standard input will processed if no file names are specified.

Command Synopsis This is just a brief synopsis of _s_e_d commands to serve as a reminder to those who already know sed; other documentation (such as the texinfo document) must be consulted for fuller descriptions.

Zero-address ‘‘commands’’

: _l_a_b_e_l
Label for b and t commands.

#_c_o_m_m_e_n_t
The comment extends until the next newline (or the end of a -e script fragment).

} The closing bracket of a { } block.

Zero- or One- address commands

= Print the current line number.

a
_t_e_x_t Append _t_e_x_t, which has each embedded newline preceeded by a backslash.

i
_t_e_x_t Insert _t_e_x_t, which has each embedded newline preceeded by a backslash.

q Immediately quit the _s_e_d script without processing any

more input, except that if auto-print is not diabled the current pattern space will be printed.

r *filename*
Append text read from *filename*.

Commands which accept address ranges

{ Begin a block of commands (end with a }).

b *label*
Branch to *label*; if *label* is omitted, branch to end of script.

t *label*
If a s/// has done a successful substitution since the last input line was read and since the last t command, then branch to *label*; if *label* is omitted, branch to end of script.

c
text Replace the selected lines with *text*, which has each embedded newline preceeded by a backslash.

d Delete pattern space. Start next cycle.

D Delete up to the first embedded newline in the pattern space. Start next cycle, but skip reading from the input if there is still data in the pattern space.

h H Copy/append pattern space to hold space.

g G Copy/append hold space to pattern space.

x Exchange the contents of the hold and pattern spaces.

l List out the current line in a ‘‘visually unambiguous’’ form.

n N Read/append the next line of input into the pattern space.

p Print the current pattern space.

P Print up to the first embedded newline of the current pattern space.

s/*regexp*/*replacement*/
Attempt to match *regexp* against the pattern space. If successful, replace that portion matched with

replacement. The replacement may contain the special character & to refer to that portion of the pattern space which matched, and the special escapes 1 through 9 to refer to the corresponding matching sub-expressions in the regexp.

w filename Write the current pattern space to filename.

y/source/dest/
Transliterate the characters in the pattern space which appear in source to the corresponding character in dest.

Addresses Sed commands can be given with no addresses, in which case the command will be executed for all input lines; with one address, in which case the command will only be executed for input lines which match that address; or with two addresses, in which case the command will be executed for all input lines which match the inclusive range of lines starting from the first address and continuing to the second address. Three things to note about address ranges: the syntax is addr1,addr2 (i.e., the addresses are separated by a comma); the line which addr1 matched will always be accepted, even if addr2 selects an earlier line; and if addr2 is a regexp, it will not be tested against the line that addr1 matched.

After the address (or address-range), and before the command, a ! may be inserted, which specifies that the command shall only be executed if the address (or address-range) does not match.

The following address types are supported:

number
Match only the specified line number.

first~step
Match every step'th line starting with line first. For example, ''sed -n 1~2p'' will print all the odd-numbered lines in the input stream, and the address 2~5 will match every fifth line, starting with the second. (This is a GNU extension.)

$ Match the last line.

/regexp/
Match lines matching the regular expression regexp.

Match lines matching the regular expression regexp.

The c may be any character.

Regular expressions POSIX.2 BREs should be supported, but they aren't completely yet. The 0equence in a regular expression matches the newline character. There are also some GNU extensions. [XXX FIXME: more needs to be said. At the very least, a reference to another document which describes what is supported should be given.]

Miscellaneous notes This version of sed supports a <newline> sequence in all regular expressions, the replacement part of a substitute (s) command, and in the source and dest parts of a transliterate (y) command. The is stripped, and the newline is kept.

SEE ALSO awk(1), ed(1), expr(1), emacs(1), perl(1), tr(1), vi(1), regex(5) [well, one ought to be written... XXX], sed.info, any of various books on sed, the sed FAQ (http://www.wollery.demon.co.uk/sedtut10.txt, http://www.ptug.org/sed/sedfaq.htm).

BUGS

E-mail bug reports to bug-gnu-utils@gnu.org. Be sure to include the word ‘‘sed’’ somewhere in the ‘‘Subject:’’ field.