NAME

lesskey - specify key bindings for less

SYNOPSIS

lesskey [-o output] [--] [input]
lesskey [--output=output] [--] [input]
lesskey -V
lesskey --version

DESCRIPTION

Lesskey is used to specify a set of key bindings to be used by *less.* The input file is a text file which describes the key bindings. If the input file is -, standard input is read. If no input file is specified, a standard filename is used as the name of the input file, which depends on the system being used: On Unix systems, $HOME/.lesskey is used; on MS-DOS systems, $HOME/_lesskey is used; and on OS/2 systems $HOME/lesskey.ini is used, or $INIT/lesskey.ini if $HOME is undefined. The output file is a binary file which is used by *less.* If no output file is specified, and the environment variable LESSKEY is set, the value of LESSKEY is used as the name of the output file. Otherwise, a standard filename is used as the name of the output file, which depends on the system being used: On Unix and OS-9 systems, $HOME/.less is used; on MS-DOS systems, $HOME/_less is used; and on OS/2 systems, $HOME/less.ini is used, or $INIT/less.ini if $HOME is undefined. If the output file already exists, *lesskey* will overwrite it.

The -V or --version option causes *lesskey* to print its version number and immediately exit. If -V or --version is present, other options and arguments are ignored.

The input file consists of one or more *sections.* Each section starts with a line that identifies the type of section. Possible sections are:

#command
: Defines new command keys.

#line-edit
: Defines new line-editing keys.

#env
: Defines environment variables.

Blank lines and lines which start with a pound sign (#) are ignored, except for the special section header lines.

COMMAND SECTION

The command section begins with the line

#command

If the command section is the first section in the file, this line may be omitted. The command section consists of lines of the form:

string <whitespace> *action* [extra-string] <newline>

Whitespace is any sequence of one or more spaces and/or tabs. The *string* is the command key(s) which invoke the action. The *string* may be a single command key, or a sequence of up to 15 keys. The*action* is the name of the less action, from the list belo w. The characters in the *string* may appear literally, or be prefixed by a caret to indicate a control key. A backslash followed by one to three octal digits may be used to specify a character by its octal value. A backslash followed by certain characters specifies input characters as follows:

b	BACKSPACE
e	ESCAPE
n	NEWLINE
r	RETURN

t	TAB
ku	UP ARROW
kd	DOWN ARROW
kr	RIGHT ARROW
kl	LEFT ARROW
kU	PAGE UP
kD	PAGE DOWN
kh	HOME
ke	END
kx	DELETE

A backslash followed by any other character indicates that character is to be taken literally. Characters which must be preceded by backslash include caret, space, tab and the backslash itself.

An action may be followed by an extra string. When such a command is entered while running *less,* the action is performed, and then the extra string is parsed, just as if it were typed in to *less.* This feature can be used in certain cases to extend the functionality of a command. For example, see the { and :t commands in the example below. The extra string has a special meaning for the quit action: when *less* quits, first character of the extra string is used as its exit status.

EXAMPLE

The following input file describes the set of default command keys used by less:

```
#command
r forw-line
n forw-line
e forw-line
j forw-line
kd forw-line
^E forw-line
^N forw-line
k back-line
y back-line
^Y back-line
^K back-line
^P back-line
J forw-line-force
K back-line-force
Y back-line-force
d forw-scroll
^D forw-scroll
u back-scroll
^U back-scroll
40 forw-screen
f forw-screen
^F forw-screen
^V forw-screen
kD forw-screen
b back-screen
^B back-screen
ev back-screen
kU back-screen
z forw-window
```

```
w back-window
e40 forw-screen-force
F forw-forever
eF forw-until-hilite
R repaint-flush
r repaint
ˆR repaint
ˆL repaint
eu undo-hilite
g goto-line
kh goto-line
< goto-line
e< goto-line
p percent
% percent
e[ left-scroll
e] right-scroll
e( left-scroll
e) right-scroll
{ forw-bracket {}
} back-bracket {}
( forw-bracket ()
) back-bracket ()
[ forw-bracket []
] back-bracket []
eˆF forw-bracket
eˆB back-bracket
G goto-end
e> goto-end
> goto-end
ke goto-end
= status
ˆG status
:f status
/ forw-search
? back-search
e/ forw-search *
e? back-search *
n repeat-search
en repeat-search-all
N reverse-search
eN reverse-search-all
& filter
m set-mark
ˆXˆX goto-mark
E examine
:e examine
ˆXˆV examine
:n next-file
:p prev-file
t next-tag
T prev-tag
:x index-file
:d remove-file
```

```
- toggle-option
:t toggle-option t
s toggle-option o
_ display-option
| pipe
v visual
! shell
+ firstcmd
H help
h help
V version
0 digit
1 digit
2 digit
3 digit
4 digit
5 digit
6 digit
7 digit
8 digit
9 digit
q quit
Q quit
:q quit
:Q quit
ZZ quit
```

PRECEDENCE

Commands specified by *lesskey* take precedence over the default commands. A default command key may be disabled by including it in the input file with the action invalid. Alternatively, a key may be defined to do nothing by using the action noaction. noaction is similar to invalid, but *less* will give an error beep for an invalid command, but not for a noaction command. In addition, ALL default commands may be disabled by adding this control line to the input file:

```
#stop
```

This will cause all default commands to be ignored. The #stop line should be the last line in that section of the file.

Be aware that #stop can be dangerous. Since all default commands are disabled, you must provide sufficient commands before the #stop line to enable all necessary actions. For example, failure to provide a quit command can lead to frustration.

LINE EDITING SECTION

The line-editing section begins with the line:

```
#line-edit
```

This section specifies new key bindings for the line editing commands, in a manner similar to the way key bindings for ordinary commands are specified in the #command section. The line-editing section consists of a list of keys and actions, one per line as in the example below.

EXAMPLE

The following input file describes the set of default line-editing keys used by less:

```
#line-edit
t forw-complete
17 back-complete
```

et back-complete
ˆL expand
ˆV literal
ˆA literal
el right
kr right
eh left
kl left
eb word-left
ekl word-left
ew word-right
ekr word-right
ei insert
ex delete
kx delete
eX word-delete
ekx word-delete
eb word-backspace
e0 home
kh home
e$ end
ke end
ek up
ku up
ej down
ˆG abort

LESS ENVIRONMENT VARIABLES

The environment variable section begins with the line

#env

Following this line is a list of environment variable assignments. Each line consists of an environment variable name, an equals sign (=) and the value to be assigned to the environment variable. White space before and after the equals sign is ignored. Variables assigned in this way are visible only to *less.* If a variable is specified in the system environment and also in a lesskey file, the value in the lesskey file takes precedence. Although the lesskey file can be used to override variables set in the environment, the main purpose of assigning variables in the lesskey file is simply to have all *less* configuration information stored in one file.

EXAMPLE

The following input file sets the -i option whenever *less* is run, and specifies the character set to be latin1:

```
#env
LESS = -i
LESSCHARSET = latin1
```

SEE ALSO

less(1)

WARNINGS

On MS-DOS and OS/2 systems, certain keys send a sequence of characters which start with a NUL character (0). This NUL character should be represented as 340 in a lesskey file.

COPYRIGHT

AUTHOR

Mark Nudelman <bug-less@gnu.org>
Send bug reports or comments to bug-less@gnu.org.