NAME

vi, view, vedit – screen-oriented (visual) display editor based on ex

SYNOPSIS

/usr/bin/vi [– | –**s**] [–**l**] [–**L**] [–**R**] [–**r** [*filename*]] [–**t** *tag*] [–**v**] [–**V**] [–**x**] [–**w***n*] [–**C**] [+*command* | –**c** *command*] *filename*…

/usr/bin/view [– | –**s**] [–**l**] [–**L**] [–**R**] [–**r** [*filename*]] [–**t** *tag*] [–**v**] [–**V**] [–**x**] [–**w***n*] [–**C**] [+*command* | –**c** *command*] *filename*…

/usr/bin/vedit [– | –**s**] [–**l**] [–**L**] [–**R**] [–**r** [*filename*]] [–**t** *tag*] [–**v**] [–**V**] [–**x**] [–**w***n*] [–**C**] [+*command* | –**c** *command*] *filename*…

/usr/xpg4/bin/vi [– | –**s**] [–**l**] [–**L**] [–**R**] [–**r** [*filename*]] [–**t** *tag*] [–**v**] [–**V**] [–**x**] [–**w***n*] [–**C**] [+*command* | –**c** *command*] *filename*…

/usr/xpg4/bin/view [– | –**s**] [–**l**] [–**L**] [–**R**] [–**r** [*filename*]] [–**t** *tag*] [–**v**] [–**V**] [–**x**] [–**w***n*] [–**C**] [+*command* | –**c** *command*] *filename*…

/usr/xpg4/bin/vedit [– | –**s**] [–**l**] [–**L**] [–**R**] [–**r** [*filename*]] [–**t** *tag*] [–**v**] [–**V**] [–**x**] [–**w***n*] [–**C**] [+*command* | –**c** *command*] *filename*…

AVAILABILITY

/usr/bin/vi
/usr/bin/view
/usr/bin/vedit

SUNWcsu

/usr/xpg4/bin/vi
/usr/xpg4/bin/view
/usr/xpg4/bin/vedit

SUNWxcu4

DESCRIPTION

vi (visual) is a display-oriented text editor based on an underlying line editor **ex**. It is possible to use the command mode of **ex** from within **vi** and to use the command mode of **vi** from within **ex**. The visual commands are described on this manual page; how to set options (like automatically numbering lines and automatically starting a new output line when you type carriage return) and all **ex** line editor commands are described on the **ex**(1) manual page.

When using **vi**, changes you make to the file are reflected in what you see on your terminal screen. The position of the cursor on the screen indicates the position within the file.

The **view** invocation is the same as **vi** except that the **readonly** flag is set.

The **vedit** invocation is intended for beginners. It is the same as **vi** except that the **report** flag is set to 1, the **showmode** and **novice** flags are set, and **magic** is turned off. These defaults make it easier to learn how to use **vi**.

OPTIONS

Invocation Options

The following invocation options are interpreted by **vi** (previously documented options are discussed in the **NOTES** section of this manual page):

– | –**s**	Suppress all interactive user feedback. This is useful when processing editor scripts.
–**l**	Set up for editing LISP programs.

–L List the name of all files saved as the result of an editor or system crash.

–R **Readonly** mode; the **readonly** flag is set, preventing accidental overwriting of the file.

–r *filename* Edit *filename* after an editor or system crash. (Recovers the version of *filename* that was in the buffer when the crash occurred.)

–t *tag* Edit the file containing the *tag* and position the editor at its definition.

–v Start up in display editing state using **vi**. You can achieve the same effect by simply typing the –**vi** command itself.

–V Verbose. Any non-tty input will be echoed on standard error. This may be useful when processing editor commands within shell scripts.

–x Encryption option; when used, **vi** simulates the **X** command of **ex** and prompts the user for a key. This key is used to encrypt and decrypt text using the algorithm of the **crypt** command. The **X** command makes an educated guess to determine whether text read in is encrypted or not. The temporary buffer file is encrypted also, using a transformed version of the key typed in for the –**x** option.

–w*n* Set the default window size to *n*. This is useful when using the editor over a slow speed line.

–C Encryption option; same as the –**x** option, except that **vi** simulates the **C** command of **ex**. The **C** command is like the **X** command of **ex**, except that all text read in is assumed to have been encrypted.

+*command* | **–c** *command*
Begin editing by executing the specified editor *command* (usually a search or positioning command).

/usr/xpg4/bin/vi

If both the –**t** *tag* and the –**c** *command* options are given, the –**t** *tag* will be processed first. That is, the file containing the tag is selected by –**t** and then the command is executed.

OPERANDS

The following operands are supported:

filename A file to be edited.

COMMAND SUMMARY

vi Modes

Command Normal and initial mode. Other modes return to command mode upon completion. **ESC** (escape) is used to cancel a partial command.

Input Entered by setting any of the following options: **a A i I o O c C s S R**. Arbitrary text may then be entered. Input mode is normally terminated with **ESC** character, or, abnormally, with an interrupt.

Last line Reading input for **: / ?** or **!**; terminate by typing a carriage return; an interrupt cancels termination.

Sample commands

In the descriptions, **CR** stands for carriage return and **ESC** stands for the escape key.

← ↓ ↑ →	arrow keys move the cursor
h j k l	same as arrow keys
i*text***ESC**	insert *text*
cw*new***ESC**	change word to *new*
ea*s***ESC**	pluralize word (end of word; append **s**; escape from input state)
x	delete a character
dw	delete a word
dd	delete a line
3dd	delete 3 lines
u	undo previous change
ZZ	exit **vi**, saving changes
:q!CR	quit, discarding changes
/*text***CR**	search for *text*
^U ^D	scroll up or down
:*cmd***CR**	any **ex** or **ed** command

Counts before vi commands

Numbers may be typed as a prefix to some commands. They are interpreted in one of these ways.

line/column number	**z G |**
scroll amount	**^D ^U**
repeat effect	most of the rest

Interrupting, canceling

ESC	end insert or incomplete cmd
DEL	(delete or rubout) interrupts

File manipulation

ZZ	if file modified, write and exit; otherwise, exit
:wCR	write back changes
:w ! CR	forced write, if permission originally not valid
:qCR	quit
:q ! CR	quit, discard changes
:e *name***CR**	edit file *name*
:e ! CR	reedit, discard changes
:e + *name***CR**	edit, starting at end
:e +*n***CR**	edit starting at line *n*
:e #CR	edit alternate file
:e ! #CR	edit alternate file, discard changes
:w *name***CR**	write file *name*
:w ! *name***CR**	overwrite file *name*
:shCR	run shell, then return
: ! *cmd***CR**	run *cmd*, then return
:nCR	edit next file in arglist
:n *args***CR**	specify new arglist
^G	show current file and line

:ta *tag***CR**	position cursor to *tag*

In general, any **ex** or **ed** command (such as *substitute* or *global*) may be typed, preceded by a colon and followed by a carriage return.

Positioning within file

^F	forward screen
^B	backward screen
^D	scroll down half screen
^U	scroll up half screen
*n***G**	go to the beginning of the specified line (end default), where *n* is a line number
/*pat*	next line matching *pat*
?*pat*	previous line matching *pat*
n	repeat last **/** or **?** command
N	reverse last **/** or **?** command
/*pat*/+*n*	nth line after *pat*
?*pat***?**–*n*	nth line before *pat*
]]	next section/function
[[	previous section/function
(	beginning of sentence
)	end of sentence
{	beginning of paragraph
}	end of paragraph
%	find matching **( )** { or }

Adjusting the screen

^L	clear and redraw window
^R	clear and redraw window if **^L** is → key
zCR	redraw screen with current line at top of window
z–CR	redraw screen with current line at bottom of window
z .CR	redraw screen with current line at center of window
/*pat*/**z–CR**	move *pat* line to bottom of window
z*n* **.CR**	use *n*-line window
^E	scroll window down 1 line
^Y	scroll window up 1 line

Marking and returning

``	move cursor to previous context
´´	move cursor to first non-white space in line
m*x*	mark current position with the ASCII lower-case letter *x*
`*x*	move cursor to mark *x*
´*x*	move cursor to first non-white space in line marked by *x*

Line positioning

H	top line on screen
L	last line on screen
M	middle line on screen
+	next line, at first non-white
–	previous line, at first non-white
CR	return, same as +
↓ or **j**	next line, same column
↑ or **k**	previous line, same column

Character positioning

ˆ	first non white-space character
0	beginning of line
$	end of line
l or →	forward
h or ←	backward
ˆH	same as ← (backspace)
space	same as → (space bar)
f*x*	find next *x*
F*x*	find previous **x**
t*x*	move to character prior to next *x*
T*x*	move to character following previous *x*
;	repeat last **f**, **F**, **t**, or **T**
,	repeat inverse of last **f**, **F**, **t**, or **T**
n|	move to column *n*
%	find matching **({** **)** or **}**

Words, sentences, paragraphs

w	forward a word
b	back a word
e	end of word
)	to next sentence
}	to next paragraph
(	back a sentence
{	back a paragraph
W	forward a blank-delimited word
B	back a blank-delimited word
E	end of a blank-delimited word

Corrections during insert

^H	erase last character (backspace)
^W	erase last word
erase	your erase character, same as **^H** (backspace)
kill	your kill character, erase this line of input
\	quotes your erase and kill characters
ESC	ends insertion, back to command mode
CTRL-C	interrupt, suspends insert mode
^D	backtab one character; reset left margin of *autoindent*
^^D	caret (^) followed by control-d (^D); backtab to beginning of line; do not reset left margin of *autoindent*
0^D	backtab to beginning of line; reset left margin of *autoindent*
^V	quote non-printable character

Insert and replace

a	append after cursor
A	append at end of line
i	insert before cursor
I	insert before first non-blank
o	open line below
O	open above
r*x*	replace single char with *x*
R*text***ESC**	replace characters

Operators

Operators are followed by a cursor motion, and affect all text that would have been moved over. For example, since **w** moves over a word, **dw** deletes the word that would be moved over. Double the operator, for example, **dd** to affect whole lines.

d	delete
c	change
y	yank lines to buffer
<	left shift
>	right shift
!	filter through command

Miscellaneous Operations

C	change rest of line (**c$**)
D	delete rest of line (**d$**)
s	substitute chars (**cl**)
S	substitute lines (**cc**)
J	join lines
x	delete characters (**dl**)
X	delete characters before cursor (**dh**)
Y	yank lines (**yy**)

Yank and Put

Put inserts the text most recently deleted or yanked; however, if a buffer is named (using the ASCII lower-case letters **a** - **z**), the text in that buffer is put instead.

3yy	yank 3 lines
3yl	yank 3 characters
p	put back text after cursor
P	put back text before cursor
"*x***p**	put from buffer *x*
"*x***y**	yank to buffer *x*
"*x***d**	delete into buffer *x*

Undo, Redo, Retrieve

u	undo last change
U	restore current line
.	repeat last change
"*d* **p**	retrieve *d*'th last delete

AUTHOR

vi and **ex** were developed by The University of California, Berkeley California, Computer Science Division, Department of Electrical Engineering and Computer Science.

ENVIRONMENT

If any of the **LC_*** variables (**LC_CTYPE, LC_MESSAGES, LC_TIME, LC_COLLATE, LC_NUMERIC,** and **LC_MONETARY**) (see **environ**(5)) are not set in the environment, the operational behavior of **vi** for each corresponding locale category is determined by the value of the **LANG** environment variable. If **LC_ALL** is set, its contents are used to override both the **LANG** and the other **LC_*** variables. If none of the above variables is set in the environment, the "C" (U.S. style) locale determines how **vi** behaves.

LC_CTYPE

Determines how **vi** handles characters. When **LC_CTYPE** is set to a valid value, **vi** can display and handle text and filenames containing valid characters for that locale. **vi** can display and handle Extended Unix code (EUC) characters where any individual character can be 1, 2, or 3 bytes wide. **vi** can also handle EUC characters of 1, 2, or more column widths. In the "C" locale, only characters from ISO 8859-1 are valid.

LC_TIME

Determines how **vi** handles date and time formats. In the "C" locale, date and time handling follows the U.S. rules.

FILES

/var/tmp	default directory where temporary work files are placed; it can be changed using the **directory** option (see the **ex**(1) **set** command)
/usr/share/lib/terminfo/?/*	compiled terminal description database
/usr/lib/.COREterm/?/*	subset of compiled terminal description database

SEE ALSO

intro(1), **ed**(1), **edit**(1), **ex**(1), **environ**(5)

Solaris Advanced User's Guide

NOTES

Two options, although they continue to be supported, have been replaced in the documentation by options that follow the Command Syntax Standard (see **intro**(1)). A −**r** option that is not followed with an option-argument has been replaced by −**L** and +*command* has been replaced by −**c** *command*.

The message **file too large to recover with** −**r option ,** which is seen when a file is loaded, indicates that the file can be edited and saved successfully, but if the editing session is lost, recovery of the file with the −**r** option will not be possible.

The editing environment defaults to certain configuration options. When an editing session is initiated, **vi** attempts to read the **EXINIT** environment variable. If it exists, the editor uses the values defined in **EXINIT**, otherwise the values set in **$HOME/.exrc** are used. If **$HOME/.exrc** does not exist, the default values are used.

To use a copy of **.exrc** located in the current directory other than **$HOME**, set the *exrc* option in **EXINIT** or **$HOME/.exrc**. Options set in **EXINIT** can be turned off in a local **.exrc** only if *exrc* is set in **EXINIT** or **$HOME/.exrc**.

Tampering with entries in **/usr/share/lib/terminfo/?/**∗ or **/usr/share/lib/terminfo/?/**∗ (for example, changing or removing an entry) can affect programs such as **vi** that expect the entry to be present and correct. In particular, removing the "dumb" terminal may cause unexpected problems.

Software tabs using **^T** work only immediately after the *autoindent.*

Left and right shifts on intelligent terminals do not make use of insert and delete character operations in the terminal.

The standard Solaris version of **vi** will be replaced by the POSIX.2 conformant version in the future. Scripts which use the **ex** family of addressing and features should use the /usr/xpg4/bin version of these utilities.