BASIC VI TUTORIAL

Introduction

The VI editor is a screen-based editor used by many Unix users. The VI editor has powerful features to aid programmers, but many beginning users avoid using VI because the different features overwhelm them. This tutorial is written to help beginning users get accustomed to using the VI editor, but also contains sections relevant to regular users of VI as well. Examples are provided, and the best way to learn is to try these examples, and think of your own examples as well... There's no better way than to experience things yourself.

Starting the VI editor

The VI editor lets a user create new files or edit existing files. The command to start the VI editor is vi, followed by the filename.

For example to edit a file called *example1*, you would type **vi example1** and then hit *return* key.

You can start VI without a filename, but when you want to save your work, you will have to tell VI which filename to save it into later. When you start VI for the first time, you will see a screen filled with tildes (A tilde looks like this: ~) on the left side of the screen. Any blank lines beyond the end of the file are shown this way. At the bottom of your screen, the filename should be shown, if you specified an existing file, and the size of the file will be shown as well, like this:

"filename" 21 lines, 385 characters

If you started VI without a filename, the bottom line of the screen will just be blank when VI starts.

VI Editor Modes

The first thing most users learn about the VI editor is that it has two modes: *command* and *insert*. The command mode allows the entry of commands to manipulate text. These commands are usually one or two characters long, and can be entered with few keystrokes. The insert mode puts anything typed on the keyboard into the current file.

VI starts out in command mode. There are several commands that put the VI editor into insert mode. The most commonly used commands to get into insert mode are *a* and *i*.

For example, hit *i* key and type "This is EASY.", then hit the *escape* key.

Once you are in insert mode, you get out of it by hitting the *escape* key.

You can hit escape two times in a row and VI would definitely be in command mode. Hitting escape while you are already in command mode doesn't take the editor out of command mode. It may beep to tell you that you are already in that mode.

Saving and Quitting a file

The command to quit out of VI is *:q*. Once in command mode, type colon, and 'q', followed by return. If your file has been modified in any way, the editor will warn you of this, and not let you quit. To ignore this message, the command to quit out of VI without saving is *:q!*. This lets you exit VI without saving any of the changes.

Of course, normally in an editor, you would want to save the changes you have made. The command to save the contents of the editor is :w. You can combine the above command with the quit command, or *:wq*. You can specify a different file name to save to by specifying the name after the *:w*.

For example, if you wanted to save the file you were working as another filename called **example2**, you would type **:w example2** and hit *return*.

At the bottom of your screen, the filename and the size of the file will be shown like this:

```
"example2" [New file] 1 line, 14 characters
```

Basic Text and Command Modes

In this section, we will learn and practice the basic text and command modes in order to be able to use VI editor more efficiently and effectively.

Editing with o option: Hit *o* key to enter insert mode in a new line below the current cursor position and type "This is very very easy." and get out of the text mode by hitting the *escape* key.

The letter *O* inserts mode in a new line above the current cursor position.

Cutting and/or yanking and pasting a line: Hit *dd* key to delete or cut the line from current cursor position and then hit *p* key to paste the deleted line after the cursor position, in this case as second line of the file and the cursor will move to the beginning of the pasted line as well.

The commands to paste are *p* and *P*. *P* pastes the specified or general buffer before the cursor position unlike *p* key.

Hit *$* key to move the cursor to the end of the current line and press *a* to append to the current line and type " Now, we are going to try to yank and paste this line." then hit *escape* key and type *:w example3* to save this file as example3 file. At the bottom of your screen, the filename and the size of the file will be shown like this:

"example3" [New file] 2 lines, 91 characters

There is also the *y* command which operates similarly to the dd command which take text from the file without deleting the text. For example, hit *yy* key to yank the current line rather than deleting the line, then hit *p* to paste the yanked line below the line of the current cursor position. Now, you should have the example3 file as shown below with cursor located at the beginning of last (3rd) line:

This is EASY.

This is very very easy.Now, we are going to try to yank and paste this line.
This is very very easy.Now, we are going to try to yank and paste this line.
Type *:w example4* to save this file as example4 file.

Yanking and pasting multiple lines: Before we yank and paste multiple lines, let's move the cursor to the beginning of the first line. Hitting *k* key k will move the cursor up one line, thus, please hit the *k* key twice, which will bring the cursor to the first line since the cursor was on the third line of the example4 file.

Other basic cursor movement commands are:

h moves the cursor to the left one character position.
j moves the cursor down one line.
k moves the cursor up one line.
l moves the cursor to the right one character position.
w moves the cursor forward one word. If the cursor is in the middle of a word, move the cursor to the first character of the next word.
To yank three lines from current cursor position downwards, hit the keys *3yy*, then hit *p* key to paste the 3 lines below the line of the current cursor position and save the file as example5 via executing *:w example5* command. However, since we wanted to yank the three lines, to the bottom of the file, hit *u* key to undo the pasting then hit *G* key to move the cursor to the last line of the file if no line number specified before executing the command, i.e. *4G* will move the cursor to the forth line of the file, and paste the yanked 3 lines by hitting *p* key and save the file as example6. Example6 file should be as shown below;

This is EASY.

This is very very easy.Now, we are going to try to yank and paste this line.
This is very very easy.Now, we are going to try to yank and paste this line.
This is EASY.
This is very very easy.Now, we are going to try to yank and paste this line.
This is very very easy.Now, we are going to try to yank and paste this line.

Searching for text of characters: */* key searches the file downwards for the string specified after the /, whereas *?* key searches the file upwards for the string specified after the ?. For example, type **/very** and hit *enter*, the cursor should move to the first letter of the first 'very' word found after the first cursor position. You can execute the same command by simply hitting *n* key again. *n* repeats last search given by '/' or '?'. If the search goes till the end of the file, at the bottom of the screen, you will receive a warning "Search wrapped around BOTTOM of buffer" and the search will restart searching from the beginning of the file. In this case, the cursor should be on the first letter of the first 'very' word in the file.

Replacing strings of text: A word that occurs in a file more than several times can be replaced by using the command *:1,$ s/word/word to replace/g*. For example, executing *:1,$ s/EASY/easy/g* will replace words EASY to easy from 1st till the end of the line, which can be changed to any line number by changing "1" and/or "$" right after the colon. Then, please save the file as example7 and quit the VI editor by executing *:wq example7*.

If you want to edit a read-only copy of a file, you can use the command *view*. For example, type *view example7* and hit *enter* key. You will view the example7 file with a "read only" notification at the bottom of your screen '"example8" [Read only] 6 lines, 336 characters'. The view invocation is the same as VI except that the readonly flag is set. For more information about the vi and view editors, you can execute *man vi* or *view* from the unix prompt to enter the VI editor manual.

Summary of VI commands

This list is a summary of VI commands, categorized by function. There may be other commands available, so check the on-line manual on VI.

Cutting and Pasting/Deleting text

"
: Specify a buffer to be used any of the commands using buffers. Follow the " with a letter or a number, which corresponds to a buffer.

D
: Delete to the end of the line from the current cursor position.

P
: Paste the specified buffer before the current cursor position or line. If no buffer is specified (with the " command.) then 'P' uses the general buffer.

X
: Delete the character before the cursor.

Y
: Yank the current line into the specified buffer. If no buffer is specified, then the general buffer is used.

d
: Delete until *where*. "dd" deletes the current line. A count deletes that many lines. Whatever is deleted is placed into the buffer specified with the " command. If no buffer is specified, then the general buffer is used.

p

Paste the specified buffer after the current cursor position or line. If no buffer is specified (with the " command.) then 'p' uses the general buffer.

x

Delete character under the cursor. A count tells how many characters to delete. The characters will be deleted after the cursor.

y

Yank until , putting the result into a buffer. "yy" yanks the current line. a count yanks that many lines. The buffer can be specified with the " command. If no buffer is specified, then the general buffer is used.

Inserting New Text

A

Append at the end of the current line.

I

Insert from the beginning of a line.

O

(letter oh) Enter *insert* mode in a new line above the current cursor position.

a

Enter *insert* mode, the characters typed in will be inserted after the current cursor position. A count inserts all the text that had been inserted that many times.

i

Enter *insert* mode, the characters typed in will be inserted before the current cursor position. A count inserts all the text that had been inserted that many times.

o

Enter *insert* mode in a new line below the current cursor position.

Moving the Cursor Within the File

^B

Scroll backwards one page. A count scrolls that many pages.

^D

Scroll forwards half a window. A count scrolls that many lines.

^F

Scroll forwards one page. A count scrolls that many pages.

^H

Move the cursor one space to the left. A count moves that many spaces.

^J

Move the cursor down one line in the same column. A count moves that many lines down.

^M

Move to the first character on the next line.

^N

Move the cursor down one line in the same column. A count moves that many lines down.

^P

Move the cursor up one line in the same column. A count moves that many lines up.

^U

Scroll backwards half a window. A count scrolls that many lines.

$

Move the cursor to the end of the current line. A count moves to the end of the following lines.

%

Move the cursor to the matching parenthesis or brace.

^

Move the cursor to the first non-whitespace character.

(

Move the cursor to the beginning of a sentence.

)

Move the cursor to the beginning of the next sentence.

{

Move the cursor to the preceding paragraph.

}

Move the cursor to the next paragraph.

|

Move the cursor to the column specified by the count.

+

Move the cursor to the first non-whitespace character in the next line.

-

Move the cursor to the first non-whitespace character in the previous line.

_

Move the cursor to the first non-whitespace character in the current line.

0

(Zero) Move the cursor to the first column of the current line.

B

Move the cursor back one word, skipping over punctuation.

E

Move forward to the end of a word, skipping over punctuation.

G

Go to the line number specified as the count. If no count is given, then go to the end of the file.

H

Move the cursor to the first non-whitespace character on the top of the screen.

L

Move the cursor to the first non-whitespace character on the bottom of the screen.

M

Move the cursor to the first non-whitespace character on the middle of the screen.

W

Move forward to the beginning of a word, skipping over punctuation.

b

Move the cursor back one word. If the cursor is in the middle of a word, move the cursor to the first character of that word.

e

Move the cursor forward one word. If the cursor is in the middle of a word, move the cursor to the last character of that word.

h

Move the cursor to the left one character position.

j

Move the cursor down one line.

k

Move the cursor up one line.

l

Move the cursor to the right one character position.

w

Move the cursor forward one word. If the cursor is in the middle of a word, move the cursor to the first character of the next word.

Moving the Cursor Around the Screen

^E

Scroll forwards one line. A count scrolls that many lines.

^Y

Scroll backwards one line. A count scrolls that many lines.

z

Redraw the screen with the following options. "z<return>" puts the current line on the top of the screen; "z." puts the current line on the center of the screen; and "z-" puts the current line on the bottom of the screen. If you specify a count before the 'z' command, it changes the current line to the line specified. For example, "16z." puts line 16 on the center of the screen.

Replacing Text

C

Change to the end of the line from the current cursor position.

R

Replace characters on the screen with a set of characters entered, ending with the Escape key.

S

Change an entire line.

c

Change until . "cc" changes the current line. A count changes that many lines.

r

Replace one character under the cursor. Specify a count to replace a number of characters.

s

Substitute one character under the cursor, and go into insert mode. Specify a count to substitute a number of characters. A dollar sign ($) will be put at the last character to be substituted.

Searching for Text or Characters

,
: Repeat the last f, F, t or T command in the reverse direction.

/
: Search the file downwards for the string specified after the /.

;
: Repeat the last f, F, t or T command.

?
: Search the file upwards for the string specified after the ?.

F
: Search the current line backwards for the character specified after the 'F' command. If found, move the cursor to the position.

N
: Repeat the last search given by '/' or '?', except in the reverse direction.

T
: Search the current line backwards for the character specified after the 'T' command, and move to the column after the if it's found.

f
: Search the current line for the character specified after the 'f' command. If found, move the cursor to the position.

n
: Repeat last search given by '/' or '?'.

t
: Search the current line for the character specified after the 't' command, and move to the column before the character if it's found.

Manipulating Character/Line Formatting

~
: Switch the case of the character under the cursor.

<
: Shift the lines up to *where* to the left by one shiftwidth. "<<" shifts the current line to the left, and can be specified with a count.

>
: Shift the lines up to *where* to the right by one shiftwidth. ">>" shifts the current line to the right, and can be specified with a count.

J
: Join the current line with the next one. A count joins that many lines.

Saving and Quitting

^\
: Quit out of "VI" mode and go into "EX" mode. The EX editor is the line editor VI is build upon. The EX command to get back into VI is ":vi".

Q
: Quit out of "VI" mode and go into "EX" mode. The ex editor is a line-by-line editor. The EX command to get back into VI is ":vi".

ZZ
: Exit the editor, saving if any changes were made.

Miscellany

^G
: Show the current filename and the status.

^L
: Clear and redraw the screen.

^R
: Redraw the screen removing false lines.

^[
: Escape key. Cancels partially formed command.

^^
: Go back to the last file edited.

!
: Execute a shell. If a is specified, the program which is executed using ! uses the specified line(s) as standard input, and will replace those lines with the standard output of the program executed. "!!" executes a program using the current line as input. For example, "!4jsort" will take five lines from the current cursor position and execute sort. After typing the command, there will be a single exclamation point where you can type the command in.

&
: Repeat the previous ":s" command.

.
: Repeat the last command that modified the file.

:
: Begin typing an EX editor command. The command is executed once the user types return. (See section below.)

@
: Type the command stored in the specified buffer.

U
: Restore the current line to the state it was in before the cursor entered the line.

m
: Mark the current position with the character specified after the 'm' command.

u
: Undo the last change to the file. Typing 'u' again will re-do the change.

EX Commands

The VI editor is built upon another editor, called EX. The EX editor only edits by line. From the VI editor you use the : command to start entering an EX command. This list given here is not complete, but the commands given are the more commonly used. If more than one line is to be modified by certain commands (such as ":s" and ":w") the range must be specified before the command. For example, to substitute lines 3 through 15, the command is ":3,15s/from/this/g".

:ab string strings
: Abbreviation. If a word is typed in VI corresponding to string1, the editor automatically inserts the corresponding words. For example, the abbreviation ":ab usa United States of America" would insert the words, "United States of America" whenever the word "usa" is typed in.

:map keys new_seq
: Mapping. This lets you map a key or a sequence of keys to another key or a sequence of keys.

:q
: Quit VI. If there have been changes made, the editor will issue a warning message.

:q!
: Quit VI without saving changes.

:s/*pattern*/*to_pattern*/*options*
: Substitute. This substitutes the specified pattern with the string in the to_pattern. Without options, it only substitutes the first occurence of the pattern. If a 'g' is specified, then all occurences are substituted. For example, the command ":1,$s/Dwayne/Dwight/g" substitutes all occurences of "Dwayne" to "Dwight".

:set [all]
: Sets some customizing options to VI and EX. The ":set all" command gives all the possible options. (See the section on customizing VI for some options.)

:una string
: Removes the abbreviation previously defined by ":ab".

:unm keys
: Removes the remove mapping defined by ":map".

:vi filename
: Starts editing a new file. If changes have not been saved, the editor will give you a warning.

:w
: Write out the current file.

:w filename
: Write the buffer to the filename specified.

:w >> filename
: Append the contents of the buffer to the filename.

:wq
: Write the buffer and quit.