UNIX VI EDITOR

Cheat Sheet

NOV 12, 2016

TECHGOEASY.COM

INTRODUCTION TO UNIX VI EDITOR

vi editor is a visual text editor. It is pronounced as V and I editor

vi shows you part of your file and allows you to enter commands that change something (add new stuff, delete a char or line, etc.).

vi has a couple of modes:

command mode: move the cursor around, move to a different part of the file, issue editing commands, switch to insert mode. This mode enables you to perform administrative tasks such as saving files, executing commands, move the cursor around, move to a different part of the file, issue editing commands, switch to insert mode. In this mode, whatever you type is interpreted as a command.

insert mode: whatever you type is put in the file (not interpreted as commands).

when you first start vi you will be in command mode.

CHEAT SHEET FOR OPENING AND ENDING VI EDITOR

When Starting VI

vi filename	Edits filename or open up a new file example vi x.sh
vi -r filename	Edits last save version of filename after a crash
vi + n filename	Edits filename and places curser at line n example vi +2 x.txt
vi + filename	Edits filename and places curser on last line example vi + x.txt
vi +/string filename	Edits filename and places curser on first occurrence of string
vi filename file2 ...	Edits filename, then edits file2 ... After the save, use :n

When Ending VI

ZZ or :wq or	Saves and exits VI

:x	
:w	Saves current file but doesn't exit
:w!	Saves current file overriding normal checks but doesn't exit
:w!	file Saves to file overriding normal checks but doesn't exit
:n, mw file	Saves lines n through m to file example :2,10w j.txt
:n,mw >>file	Saves lines n through m to the end of file
:q	Quits VI and may prompt if you need to save
:q!	Quits VI and without saving. If you have done some mistake and want to quit without making any changes
:e!	Edits file discarding any unsaved changes (starts over)
:we!	Saves and continues to edit current file

CHEAT SHEET FOR TEXT CHANGES IN VI EDITOR

Inserting Text

i	Insert before cursor
I	Insert before line
a	Append after cursor
A	Append after line
o	Open a new line after current line
O	Open a new line before current line

r	Replace one character
R	Replace many characters
:r	file Reads file and inserts it after current line
:nr	file Reads file and inserts it after line n

Cursor Movement

h	Move left
j	Move down
k	Move up
l	Move right
w	Move to next word
W	Move to next blank delimited word
E	Move to the end of Blank delimited word
0 or |	Move to the beginning of the line
n|	Moves to the column n in the current line
$	Move to the end of the line
1G	Move to the first line of the file
G	Move to the last line of the file
nG	Move to nth line of the file. Suppose you want to move to 100th line then 100G
:n	Move to nth line of the file

H	Move to top of screen
nH	Moves to nth line from the top of the screen
M	Move to middle of screen

Deleting Text

D	Delete to the end of the line
d$	Deletes from the cursor to the end of the line
dd or :d	Delete current line
ndw	Deletes the next n words starting with current
ndb	Deletes the previous n words starting with current
ndd	Deletes n lines beginning with the current line. Suppose we want to delete 100 lines then 100dd
:n,md	Deletes lines n through m
"np	Retrieves the last nth delete (last 9 deletes are kept in a buffer)

Yanking Text

yy	Yank the current line
:y	Yank the current line
nyy or nY	Places n lines in the buffer-copies

Putting text

p	Put after the position or after the line
P	Put before the position or before the line

CHEAT SHEET FOR SEARCHING AND SUBSTITUTION IN VI EDITOR

Search for strings (vi cheat sheet)

/string	Search forward for string like error
?string	Search back for string
n	Search for next instance of string
N	Search for previous instance of string
%	Searches to beginning of balancing ( ) [ ] or { }
?str	Finds in reverse for str
:set ic	Ignores case when searching. Suppose you want to search for failure, now we don't the case it will happen, so setting this will help in finding all the occurrence
:set noic	Pays attention to case when searching
:n,ms/str1/str2/opt	Searches from n to m for str1; replaces str1 to str2; using opt-opt can be g for global change, c to confirm change (y to acknowledge, to suppress), and p to print changed lines
&	Repeats last :s command
:g/str/cmd	Runs cmd on all lines that contain str

:g/str1/s/str2/str3/	Finds the line containing str1, replaces str2 with str3
:v/str/cmd E	xecutes cmd on all lines that do not match str

Examples of Search Strings

A file x.txt is given, we do below search strings on it

/fmw_home	search forward for fmw_home in the file
?fmw_home	search backward for fmw_home in the file
n	repeat previous search
N	repeat search in opposite direction
/, ?	repeat search forward or backward

Replace or Substitution

It is a very function which is used in vi. Many times you have to replace certain field with other, that time it is very useful

The search and replace function is accomplished with the :s command. It is commonly used in combination with ranges or the :g command (below).
:s/pattern/string/flags Replace pattern with string according to flags.
g Flag - Replace all occurrences of pattern
c Flag - Confirm replaces.
& Repeat last :s command

It can be used like this also
:s, pattern, string, flags
:s^pattern^string^flags

Examples of Replace

:s^/oracle^/applmgr^g	This will replace /oracle with /applmgr everywhere in the file
s,^a,^b, g	This will replace ^a with ^b everywhere in the file
:2,8/this/s//that	replace first occurrence of `this' with `that' in lines 2 through 8
:2,$/this/s//that	replace first occurrence of `this' with `that' in lines 2 through the end

:g/\/home\/me/s//\/r2\/you/g	replace /home/me with /r2/you throughout the file
: 33,224s/^/hh/	insert the string 'hh' at the beginning of lines
:%s/{TAB}*$//	Strip blanks at end of line:

Ranges

Ranges may precede most "colon" commands and cause them to be executed on a line or lines. For example :3,7d would delete lines 3-7. Ranges are commonly combined with the :s command to perform a replacement on several lines, as with :.,$s/pattern/string/g to make a replacement from the current line to the end of the file.

:n,m Range - Lines n-m
:. Range - Current line
:$ Range - Last line
:% Range - All lines in file
:g/pattern/ Range - All lines that contain pattern

VI CHEAT SHEET FOR SHELL FUNCTIONS

:! cmd	Executes shell command cmd; you can add these special characters to indicate:% name of current file# name of last file edited
:!!	Executes last shell command
:r! cmd	Reads and inserts output from cmd
:f file	Renames current file to file
:w !cmd	Sends currently edited file to cmd as standard input and execute cmd
:cd dir	Changes current working directory to dir

:sh	Starts a sub-shell (CTRL-d returns to editor)

let us take the examples of Executing Unix commands in vi:
Any UNIX command can be executed from the vi command line by typing an "!" before the UNIX command.
Examples:
":!pwd" - shows your current working directory.
":r !date" - reads the results from the date command into a new line following the cursor.
":r !ls -1" - Place after the cursor, the current directory listing displayed as a single column.

CHEAT SHEET FOR VI EDITOR FOR FILES

:w file	Write to file
:r file	Read file in after line
:n	Go to next file
:p	Go to previous file
:e file	Edit file
!!program	Replace line with output from program

Editing multiple files:
vi file1 file2 file3
:n Edit next file (file2)
:n Edit next file (file3)
:rew Rewind to the first file (file1)

CHEAT SHEET FOR VI SETTINGS

Note: Options given are default.
To change them, enter type :set option to turn them on or :set nooptioni to turn them off.
To make them execute every time you open VI, create a file in your HOME directory called .exrc and type the options without the colon (:) preceding the option :set ai Turns on auto indentation

:set all	Prints all options to the screen
:set dir=tmp	Sets tmp to directory or buffer file
:set ic	Ignores case when searching
:set list	Shows tabs (^I) and end of line ($)
:set nu	Shows line numbers
:set ro	Changes file type to "read only"
: set showmode	Indicates input or replace mode at bottom\

SOME OTHER COMMANDS

u - Undo the latest change.
U - Undo all changes on a line, while not having moved off it (unfortunately).
:u - Undo last substitution on line (only one)
^G - Give file name, status, current line number and relative position.
:[x,y]s/

//g - Substitute (on lines x through y) the pattern

with ,`g' for `global'
:nnpu paste after line nn