**UNIX/LINUX OPERATING SYSTEMS**

**(Vi Editor, Shell Script)**

**Vi Editor:**

* An editor: a program for creating and modifying files containing source code, text, data, memo, reports and other information. Vi is an editor Popular choice among UNIX/LINUX users. Called vi because it is visual; immediately displays the changes you make to text.
* It is modal, works in three modes:

Insert mode: lets you enter text, accessed by typing the letter i after the editor starts.

Command mode: started by pressing ESC, lets you enter commands to perform editing tasks.

EX mode: employs and extended set of commands used in an early UNIX editor called ex. Can access this mode by pressing ESC to enter the command mode and then typing a colon (:) to enter extended commands at the bottom of the screen.

**Vi Command List:**

To start the vi editor, type vi [file-name] at the linux$ prompt. When vi starts you are in the COMMAND mode. To add text, you must first get into INPUT mode. To delete, change, or move text, you must return to COMMAND mode. There are several ways to get into INPUT mode, as described below. To return to the COMMAND mode, press the ESC (escape) key. If you get stuck or confused while learning the vi editor, press ESC.

**Opening, Saving and Closing Files Using vi:**

vim [file] [RETURN] Open a file using the vim version of vi

ZZ Exit vi, saving changes

**Input mode:**

Note: To get into insert mode, the character i from COMMAND mode:

Press ESC (escape) to return to COMMAND mode.

i [text] Insert text before cursor

I [text] Insert text at start of line

a [text] Append text after cursor

A [text] Append text at end of line

o [text] Open a new line below the cursor

O [text] Open a new line above the cursor

r [char] Replace a single character with [char>

r [RETURN] Split a line at the cursor

R [text] Replace old text

s [text] Substitute one or more characters for a char.

S [text] Substitute an entire line

cw [text] Change a word to [text]

cb [text] Change from start of word to cursor

ce [text] Change from cursor to end of word

C [text] Change from cursor to end of line

**Command Mode:**

To repeat the last modification, type: . (period)

To undo the last modification, type: u

**Moving the Cursor:**

Note: The arrow keys work but the mouse does NOT move the cursor.

h Move left one space

j Move down one line

k Move up one line

l Move right one space

0 Move to start of line (zero)

$ Move to end of line

[n]G or :[n] Move to the n-th line in the file

G Move to the last line in the file

H Move to first line on screen

L Move to last line on screen

M Move to middle line on screen

w Move to beginning of next word

( Move to beginning of sentence

) Move to end of sentence

{ Move to beginning of paragraph

} Move to end of paragraph

**Deleting Text:**

x Delete a single character at the cursor

dd Delete an entire line and put into buffer

D Delete from cursor to end of line

dw Delete from cursor to start of next word

df[char] Delete from cursor through [char]

dt[char] Delete from cursor to, but not including [char]

dH Delete from top of screen to current line

dL Delete from current line to bottom of screen

**Putting Text:**

yy or Y Copy a line into buffer without deleting

yw Copy a word into buffer without deleting

p Put last yank or delete after the cursor

P Put last yank or delete before the cursor

xp Reverse two characters

ddp Reverse two lines

J Join the next line to the current line

**Pattern Searching:**

/[pattern] [RETURN] Search forward for [pattern]

n or / [RETURN] Repeat the last search forward

?[pattern] [RETURN] Search backward for [pattern]

N or ? [RETURN] Repeat the last search backward

'' (2 single quotes) Return to previous cursor position

:1,$ s/[old]/[new]/g Globally search for and replace [old] with [new]

**Screen by Screen Scrolling:**

[CTRL] U Scroll up half a screen

[CTRL] D Scroll down half a screen

[CTRL] F Scroll forward one screen

[CTRL] B Scroll backward one screen

z [RETURN] Redraw the screen, cursor line at top

**ex Editor commands:**

Note: These commands (issued from COMMAND mode) belong to ex, a line editor.

:wq [RETURN] Write (save) the file to disk and quit vi

:w [RETURN] Write (save) the file to disk without quitting

:q! [RETURN] Quit without saving the last changes

:f [RETURN] Show filename and current line

:r [file] [RETURN] Read [file] into buffer at cursor position

:! [command] [RETURN] Execute a UNIX command without quitting vi

**Common UNIX/LINUX commands:**

=====================================================================

Note: These commands are not part of the vi editor.

Issue these commands from the unix $ prompt:

ls, ls -l, ls -la [dir] List contents of a directory

cp [file1] [file2] Copy [file1] to [file2]

mv [file2] [file2] Move [file1] to [file2]

rm [file] Remove (delete) a file

mkdir [dir] Create a new directory

cd [dir] Change the current directory

rmdir [dir] Remove (delete) a directory

[CTRL] z Stop the current process

bg Resume a stopped process in the background

ps -a List all processes and characteristics

kill -9 [job number] Kill a stopped job (process, or program)

man [command] Display manual pages about a command, such as vi

!! Repeat the last command

**Introduction to Shell Scripts:**

* Created usually with the vi editor. Contains instructions that do not need to be written from scratch but can be selectively chosen from the operating system’s inventory of executable commands. Do not need to be converted to machine language: SHELL interprets the lines in the shell script.

##### Variables

* When a script starts all environment variables are turned into shell variables. New variables can be instantiated like this:

Name = value

* You must do it exactly like that, with no spaces either side of the equals sign, the name must only be made up of alphabetic characters, numeric characters and underscores, it cannot begin with a numeric character. You should avoid using keywords like for or anything like that, the interpreter will let you use them but doing so can lead to obfuscated code ;)
* Variables are referenced like this: **$***name*, here is an example:

|  |
| --- |
| #!/bin/sh  msg1=Hello  msg2=There!  echo $msg1 $msg2 |

|  |
| --- |
| #!/bin/sh  msg1="one"  msg2="$msg1 two"  msg3="$msg2 three"  echo $msg3 |

|  |
| --- |
| #!/bin/sh  echo 'msg="Hello World!"' > hello  echo 'echo $msg' >> hello  chmod 700 hello  ./hello |