TEXT EDITOR

A *text editor* is a tool that allows a user to create and revise *documents* in a computer. Though this task can be carried out in other modes, the word text editor commonly refers to the tool that does this interactively. Earlier computer documents used to be primarily plain text documents, but nowadays due to improved input-output mechanisms and file formats, a document frequently contains pictures along with texts whose appearance (script, size, colour and style) can be varied within the document. Apart from producing output of such wide variety, text editors today provide many advanced features of interactiveness and output.

Some common Unix Text editors are : vim, nano, GUI editors , emacs, mousepad , xedit , Pico, Emacs , Xemacs - a GUI-based version of Emacs., vi - a mostly terminal-based text editor.

VIM

What Is Vim?

Vim is a highly configurable text editor built to enable efficient text editing. It is an improved version of the vi editor distributed with most UNIX systems.

Vim is often called a "programmer's editor," and so useful for programming that many consider it an entire IDE. It's not just for programmers, though. Vim is perfect for all kinds of text editing, from composing email to editing configuration files.

Despite what the above comic suggests, Vim can be configured to work in a very simple (Notepad-like) way, called evim or Easy Vim.

Features of Vim

This section discusses some of the important features of Vim −

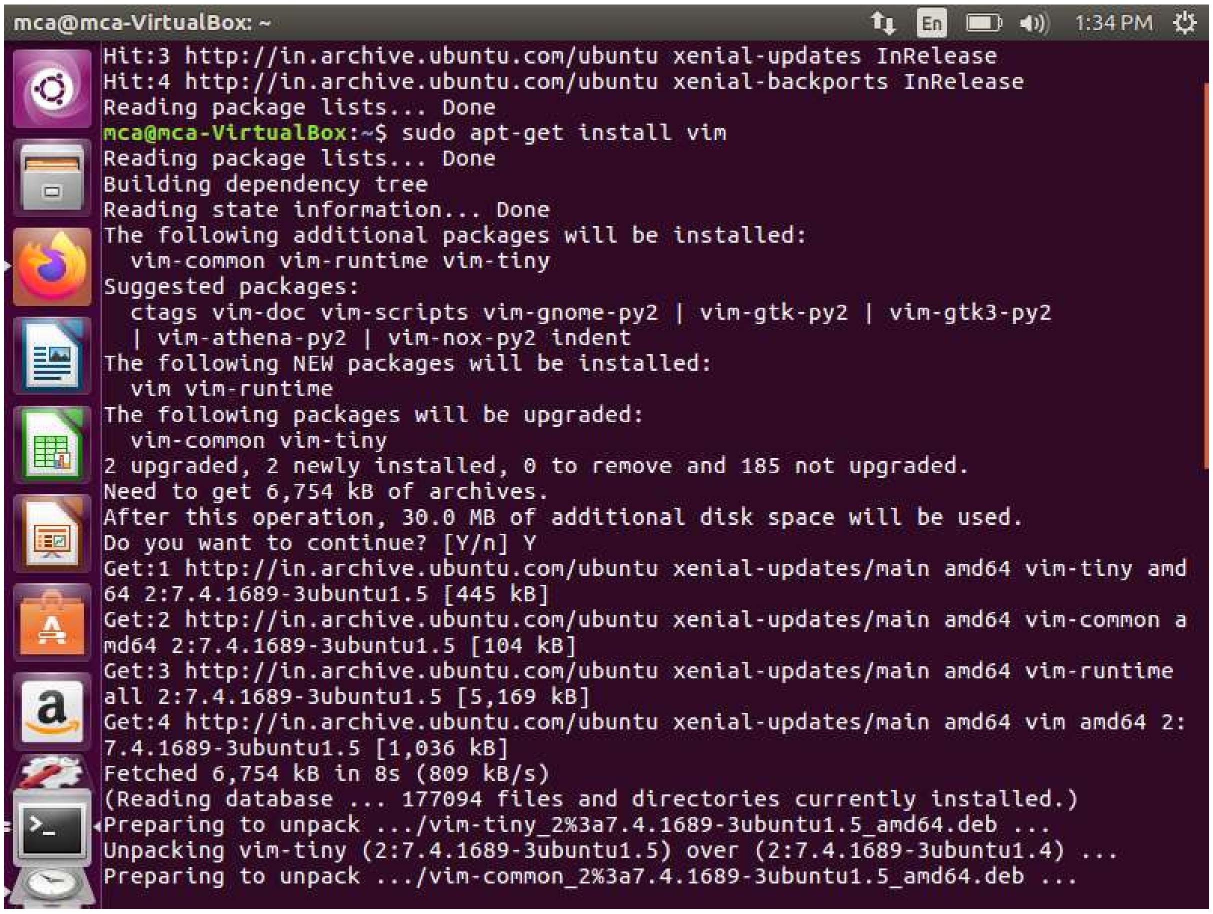
- Its memory footprint is very low
- It is command centric. You can perform complex text related task with few commands
- It is highly configurable and uses simple text file to store its configuration
- There are many plug-in available for Vim. Its functionality can be extended in great manner using these plug-in
- It supports multiple windows. Using this feature screen can be split into multiple windows

- Same as multiple windows, it also supports multiple buffers
- It supports multiple tabs which allows to work on multiple files
- It supports recording features which allows to record and play Vim commands in repeated manner

VIM INSTALLATION

To install vim on Debian based Linux like ubuntu run the command:

sudo apt-get install vim



```
Hit:3 http://in.archive.ubuntu.com/ubuntu xenial-updates InRelease
Hit:4 http://in.archive.ubuntu.com/ubuntu xenial-backports InRelease
Reading package lists... Done
mca@mca-VirtualBox:~$ sudo apt-get install vim
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  vim-common vim-runtime vim-tiny
Suggested packages:
  ctags vim-doc vim-scripts vim-gnome-py2 | vim-gtk-py2 | vim-gtk3-py2
  | vim-athena-py2 | vim-nox-py2 indent
The following NEW packages will be installed:
  vim vim-runtime
The following packages will be upgraded:
  vim-common vim-tiny
2 upgraded, 2 newly installed, 0 to remove and 185 not upgraded.
Need to get 6,754 kB of archives.
After this operation, 30.0 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://in.archive.ubuntu.com/ubuntu xenial-updates/main amd64 vim-tiny amd
64 2:7.4.1689-3ubuntu1.5 [445 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu xenial-updates/main amd64 vim-common a
md64 2:7.4.1689-3ubuntu1.5 [104 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu xenial-updates/main amd64 vim-runtime
all 2:7.4.1689-3ubuntu1.5 [5,169 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu xenial-updates/main amd64 vim amd64 2:
7.4.1689-3ubuntu1.5 [1,036 kB]
Fetched 6,754 kB in 8s (809 kB/s)
(Reading database ... 177094 files and directories currently installed.)
Preparing to unpack .../vim-tiny_2%3a7.4.1689-3ubuntu1.5_amd64.deb ...
Unpacking vim-tiny (2:7.4.1689-3ubuntu1.5) over (2:7.4.1689-3ubuntu1.4) ...
Preparing to unpack .../vim-common_2%3a7.4.1689-3ubuntu1.5_amd64.deb ...
```

Modes in VIM editor

There are three modes in vim editor.

o Command Mode

o Insert Mode

o Last Line Mode

Command mode

When we first start editing a file using the Vim editor, the editor will be opened in a command mode.We can issue many commands that allow us to insert, append, delete text or search and navigate within our file.When using command mode we cannot insert text immediately.We first need to issue an insert(i), append(a), or open(o) command to insert the text in the file.

Insert mode

When we issue an insert,append,or open command, we will be in insert mode.Once in an insert mode we can type text into our file or navigate within the file.We can toggle between the command mode and the insert mode by pressing the ESC key.

Last Line Mode

The last line mode normally is used to perform operations like quitting the Vim session or savig a file.To go to the last line mode we first need to be in the command mode.From command mode we can go to last line mode by pressing the colon(:) key.After pressing this key,we will see a colon character at the beginning of the last line of our editor window with a cursor blinking near it. This indicates that the editor is ready for accepting a ‘last line command’.

It is possible to toggle back to the command mode from the last line mode by pressing the ESC key twice or by pressing the backspace key until the initial ‘ : ‘ character is gone along with all the characters that we had typed or by simply pressing the ENTER key.

COMMANDS

2. Command mode (Where you give commands to the editor to get things done . Press ESC for command mode)

Most of them below are in command mode

x – to delete the unwanted character

u – to undo the last the command and U to undo the whole line

CTRL-R to redo

A – to append text at the end

:wq – to save and exit

:q! – to trash all changes

dw – move the cursor to the beginning of the word to delete that word

2w – to move the cursor two words forward.

3e – to move the cursor to the end of the third word forward.

0 (zero) to move to the start of the line.

d2w – which deletes 2 words .. number can be changed for deleting the number of consecutive words like d3w

dd to delete the line and 2dd to delete to line .number can be changed for deleting the number of consecutive words

The format for a change command is: operator [number] motion
-operator – is what to do, such as d for delete
– [number] – is an optional count to repeat the motion
– motion – moves over the text to operate on, such as w (word), $ (to the end of line), etc.

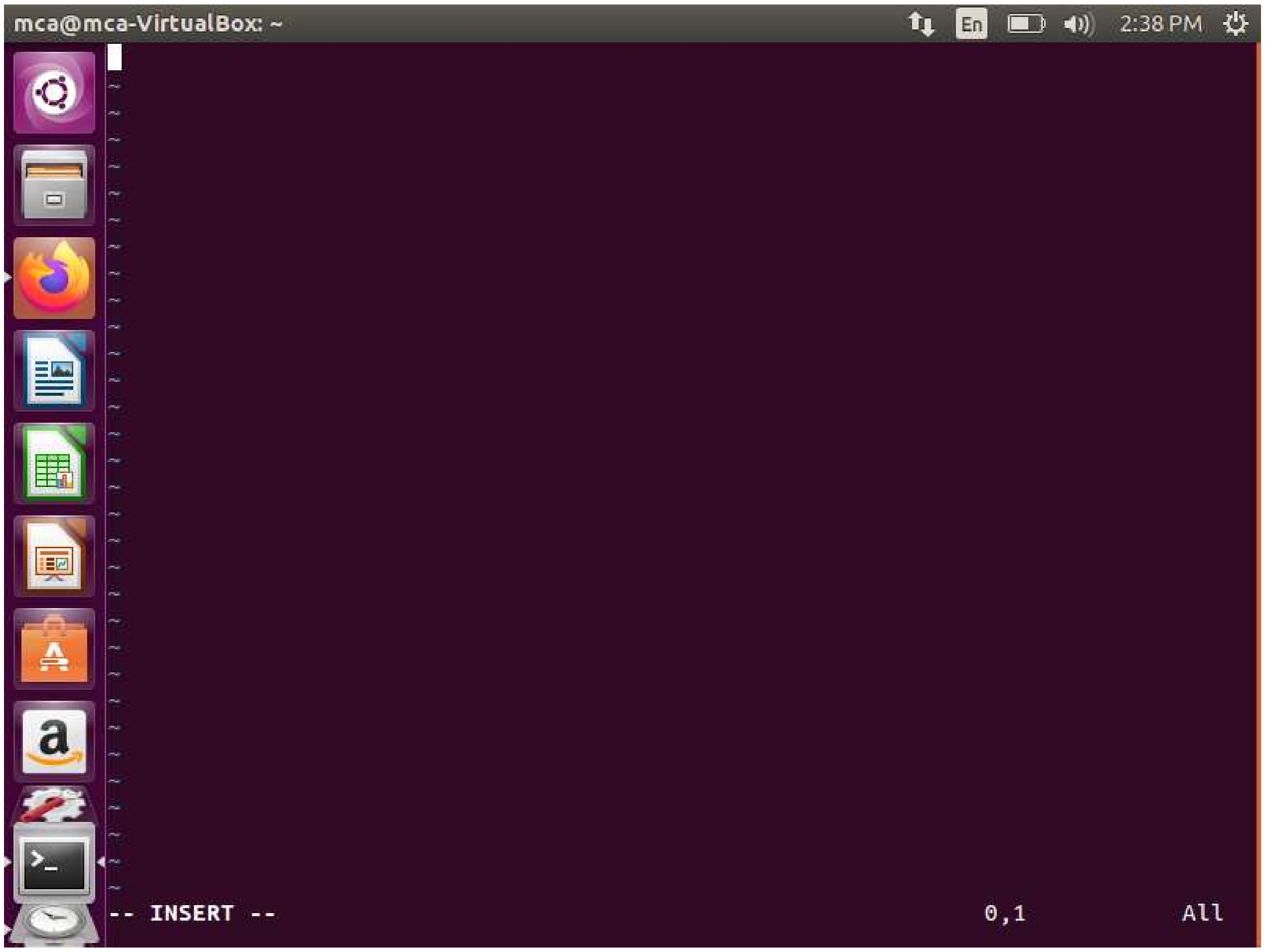
- p – puts the previously deleted text after the cursor(Type dd to delete the line and store it in a Vim register. and p to put the line)

- r – to replace the letter e.g press re to replace the letter with e
- ce – to change until the end of a word (place the cursor on the u in lubw it will delete ubw)
- ce – deletes the word and places you in Insert mode
- G – to move you to the bottom of the file.
- gg – to move you to the start of the file.
  Type the number of the line you were on and then G
- % to find a matching),], or }
- :s/old/new/g to substitute 'new' for 'old' where g is globally
- / backward search n to find the next occurrence and N to search in opposite direction
- ? forward search
- :! to run the shell commands like :!dir, :!ls
- :w – TEST (where TEST is the filename you chose.) . Save the file
- v – starts visual mode for selecting the lines and you can perform operation on that like d delete
- :r – Filename will insert the content into the current file
- R – to replace more than one character
- y – operator to copy text using v visual mode and p to paste it
- yw – (copy)yanks one word
- o – opens a line below the cursor and start Insert mode.
- O – opens a line above the cursor.
- a – inserts text after the cursor.
- A – inserts text after the end of the line.
- e – command moves to the end of a word.
- y – operator yanks (copies) text, p puts (pastes) it.
- R – enters Replace mode until <ESC> is pressed.

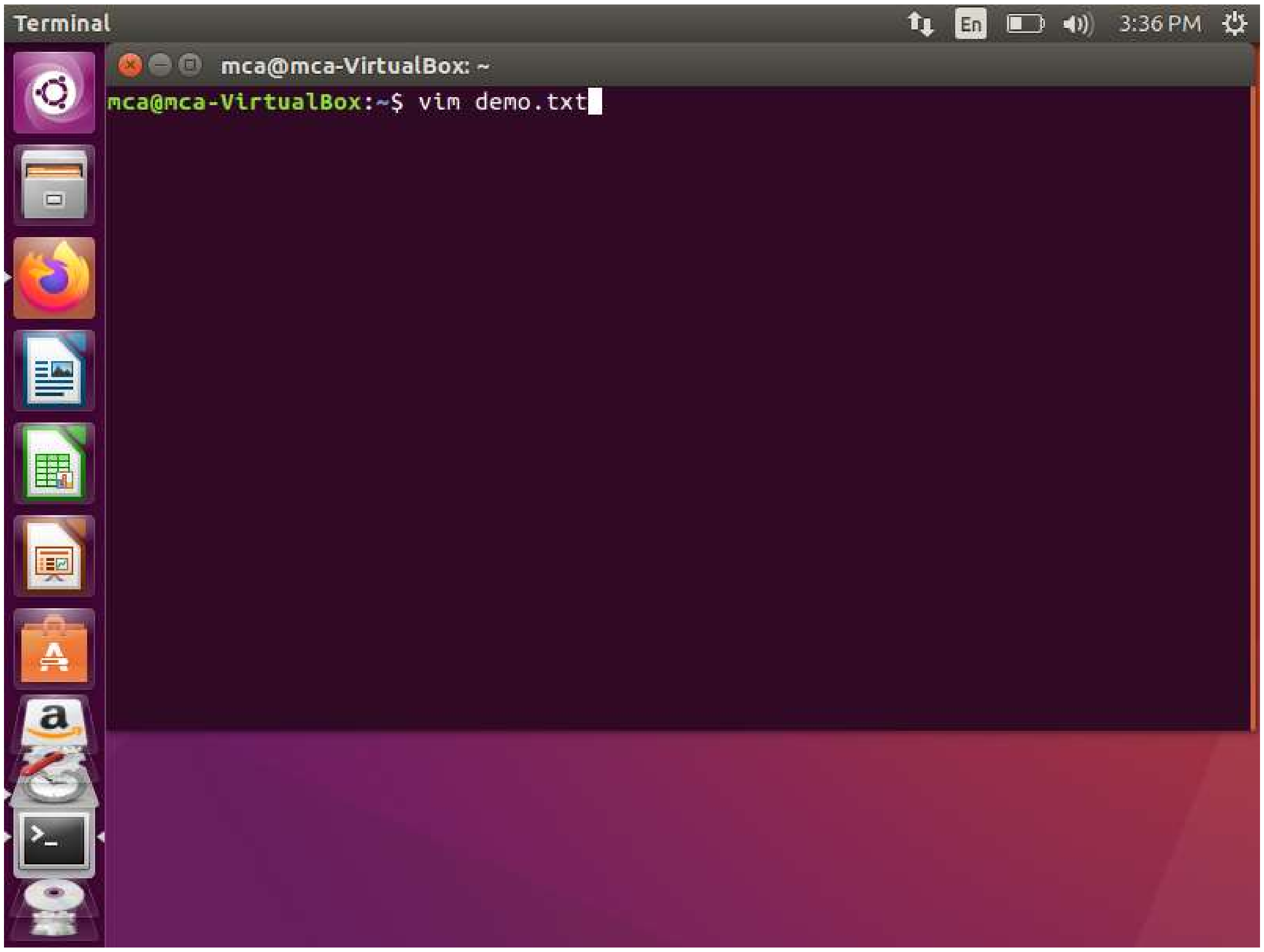
- ctrl-w to jump from one window to another

type a command :e and press ctrl+D to list all the command name starts with :e and press tab to complete the command

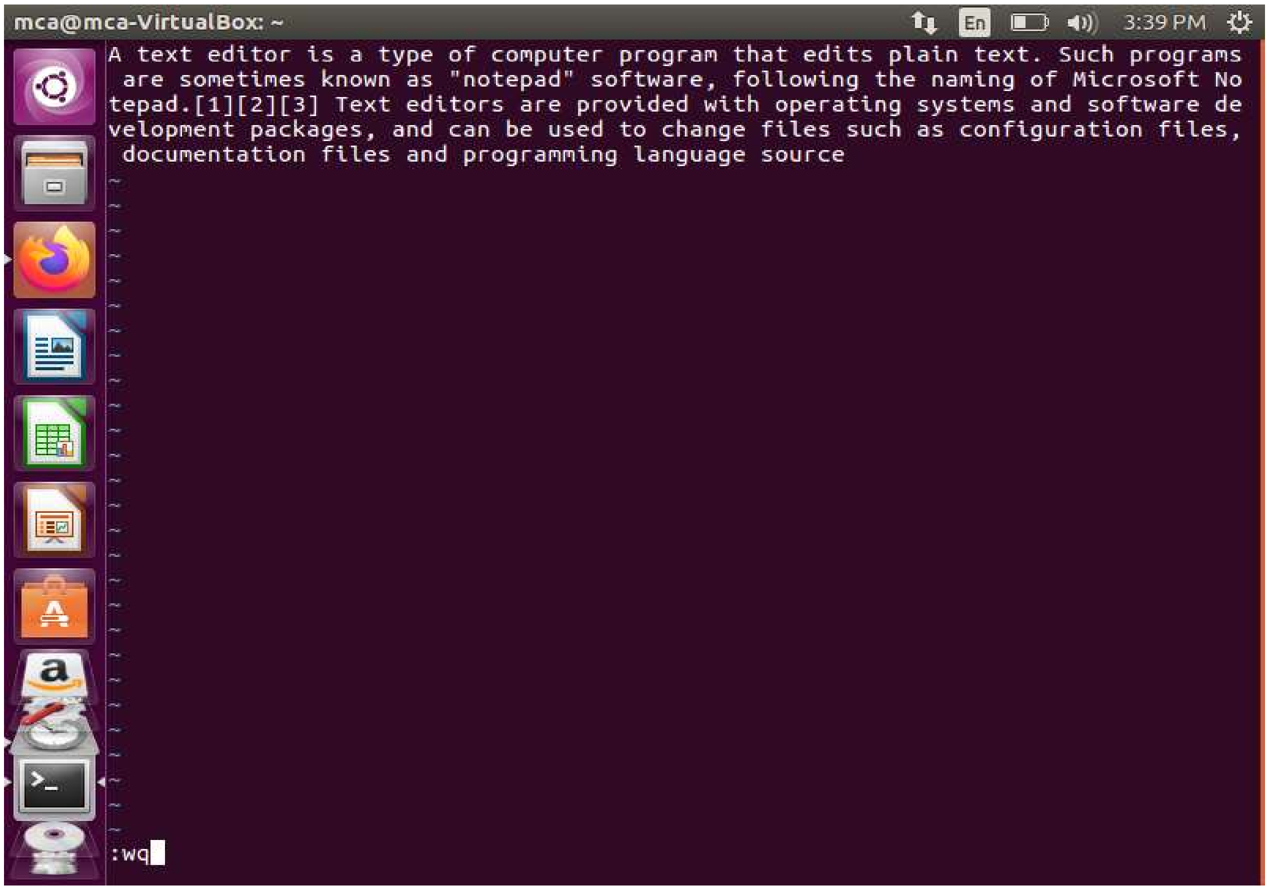
Insert Mode



```
~
~
~
-- INSERT --                                        0,1           All
```



```
mca@mca-VirtualBox:~$ vim demo.txt
```



```
A text editor is a type of computer program that edits plain text. Such programs
 are sometimes known as "notepad" software, following the naming of Microsoft No
tepad.[1][2][3] Text editors are provided with operating systems and software de
velopment packages, and can be used to change files such as configuration files,
 documentation files and programming language source
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```