Vi Editor

What is vi

The vi editor is elaborated as **vi**sual editor. It is installed in every Unix system. In other words, it is available in all Linux distros. It is user-friendly and works same on different distros and platforms. It is a very powerful application. An improved version of vi editor is **vim**.

The vi editor has two modes:

❍**Command Mode:** In command mode, actions are taken on the file. The vi editor starts in command mode. Here, the typed words will act as commands in vi editor. To pass a command, you need to be in command mode.

❍**Insert Mode:** In insert mode, entered text will be inserted into the file. The **Esc** key will take you to the command mode from insert mode.

By default, the vi editor starts in command mode. To enter text, you have to be in insert mode, just type **'i'** and you'll be in insert mode. Although, after typing **i** nothing will appear on the screen but you'll be in insert mode. Now you can type anything.

To exit from insert mode press **Esc** key, you'll be directed to command mode.

If you are not sure which mode you are in, press Esc key twice and you'll be in command mode.

Using vi

The vi editor tool is an interactive tool as it displays changes made in the file on the screen while you edit the file.

In vi editor you can insert, edit or remove a word as cursor moves throughout the file.

Commands are specified for each function like to delete it's x or dd.

The vi editor is case-sensitive. For example, **p** allows you to paste after the current line while **P** allows you to paste before the current line.

vi syntax:

1. vi **<fileName>**
   In the terminal when you'll type vi command with a file name, the terminal will get clear and content of the file will be displayed. If there is no such file, then a

new file will be created and once completed file will be saved with the mentioned file name.

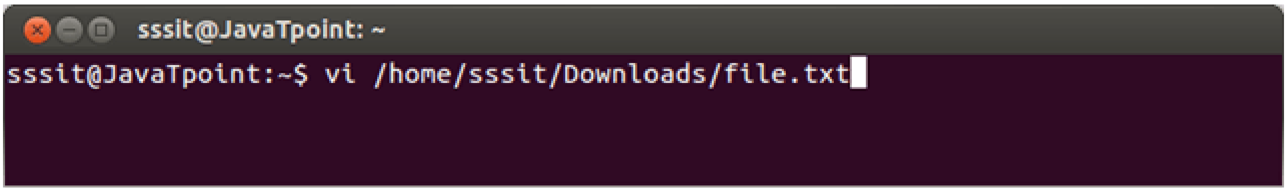
Linux vi example

Let's understand vi through an example:

To start vi open your terminal and type vi command followed by file name. If your file is in some other directory, you can specify the file path. And if in case, your file doesn't exist, it will create a new file with the specified name at the given location.

Example:

1. vi /home/sssit/Downloads/file.txt

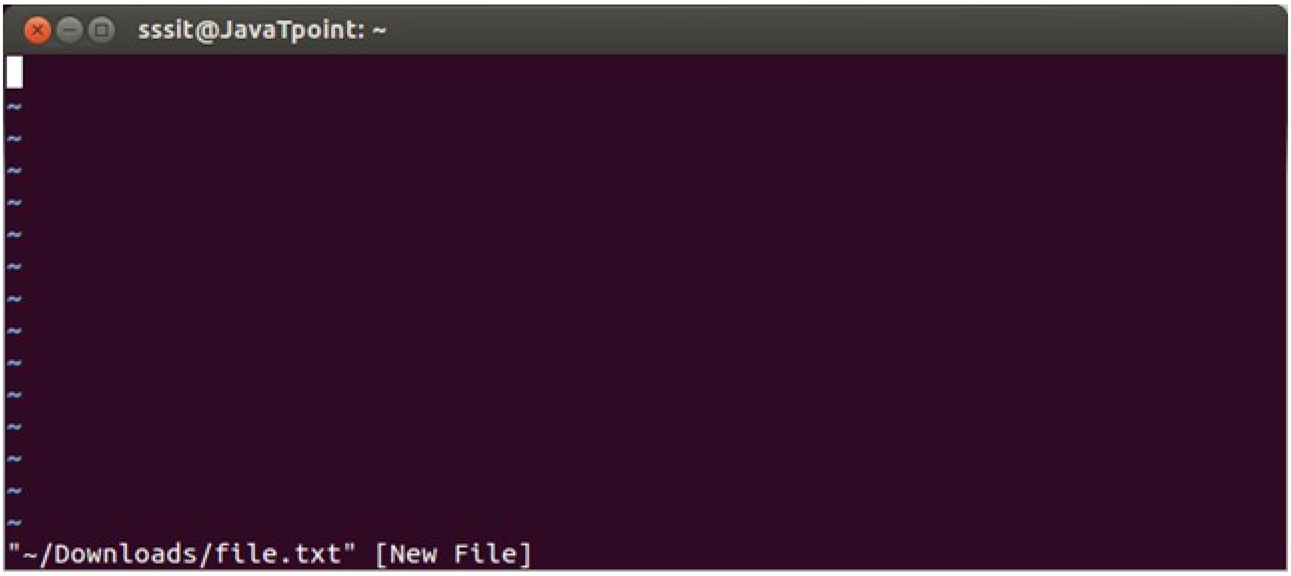


```
sssit@JavaTpoint:~$ vi /home/sssit/Downloads/file.txt
```

Look at the above snapshot, we are creating a new file **file.txt** (as this file doesn't exist) and have entered the full path for the directory **Downloads.**

Command mode

This is what you'll see when you'll press enter after the above command. If you'll start typing, nothing will appear as you are in command mode. By default vi opens in command mode.



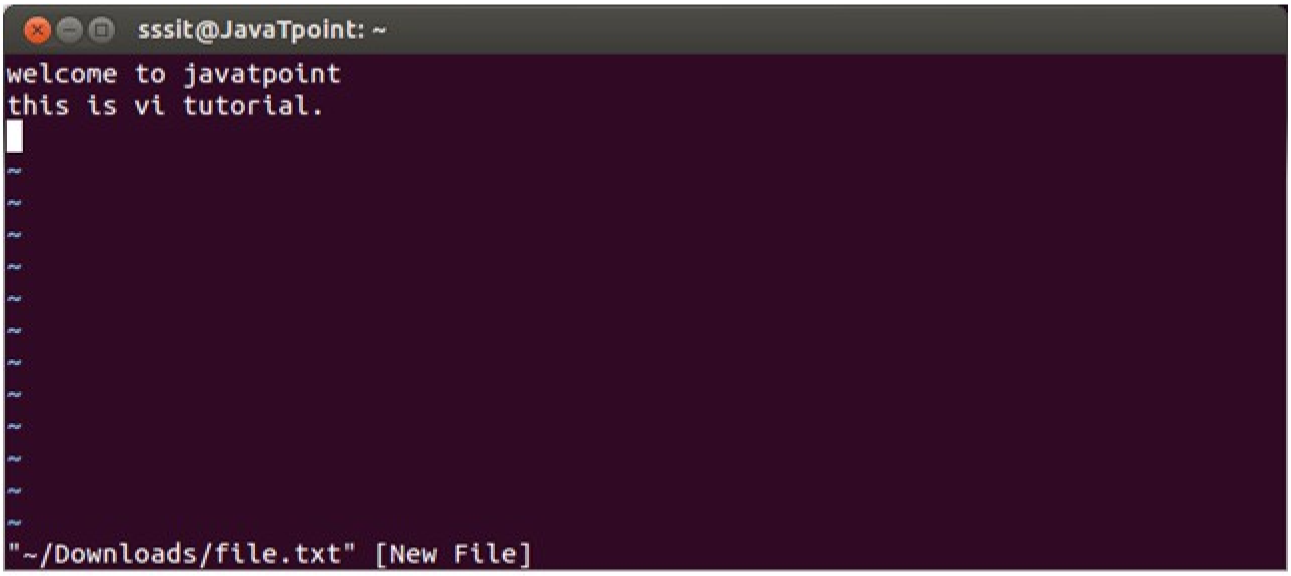
```

~
~
~
~
~
~
~
~
~
~
~
~
~
~
"~/Downloads/file.txt" [New File]
```

Look at the above snapshot, it is blank as it is a new file. To start typing, you have to move to the insert mode. At the end of the terminal window, directory name and file name are displayed.

Insert mode

To move to the insert mode press **i.** Although, there are other commands also to move to insert mode which we'll study in next page.



```
welcome to javatpoint
this is vi tutorial.

~
~
~
~
~
~
~
~
~
~
~
~
"~/Downloads/file.txt" [New File]
```

Look at the above snapshot, after pressing **i** we have entered into insert mode. Now we can write anything. To move to the next line press enter.

Once you have done with your typing, press **esc** key to return to the command mode.

To save and quit

You can save and quit vi editor from command mode. Before writing save or quit command you have to press colon **(:).** Colon allows you to give instructions to vi.

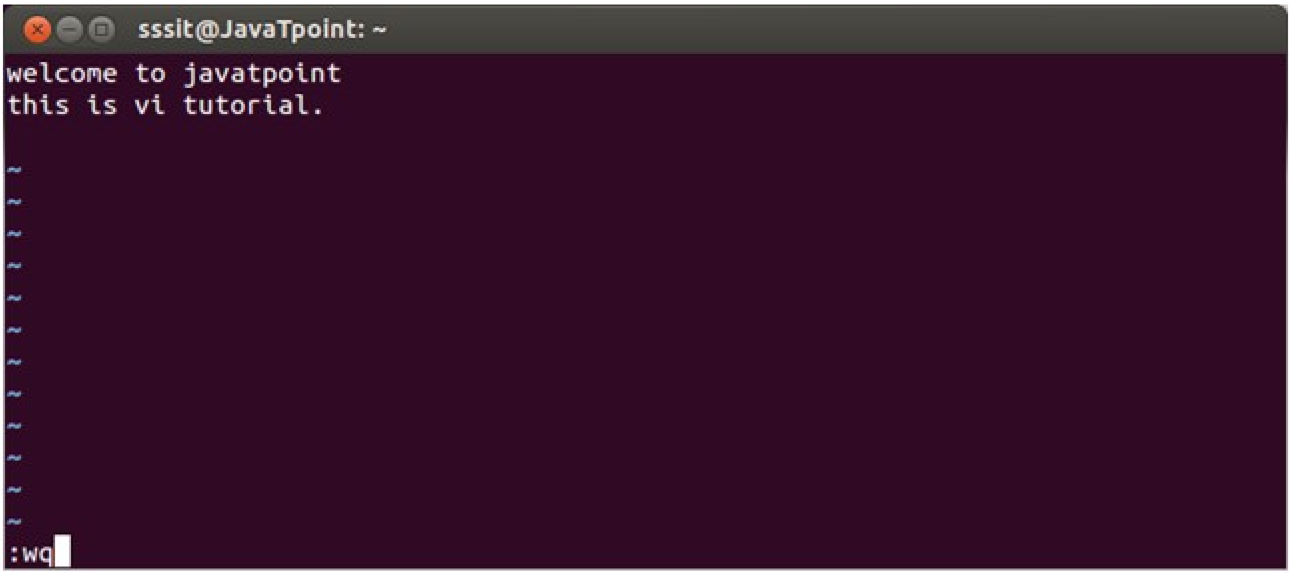
exit vi table:

Commands	Action
:wq	Save and quit
:w	Save
:q	Quit
:w fname	Save as fname
ZZ	Save and quit
:q!	Quit discarding changes made

:w!	Save (and write to non-writable file)

To exit from vi, first ensure that you are in command mode. Now, type :wq and press enter. It will save and quit vi.

Type **:wq** to save and exit the file.

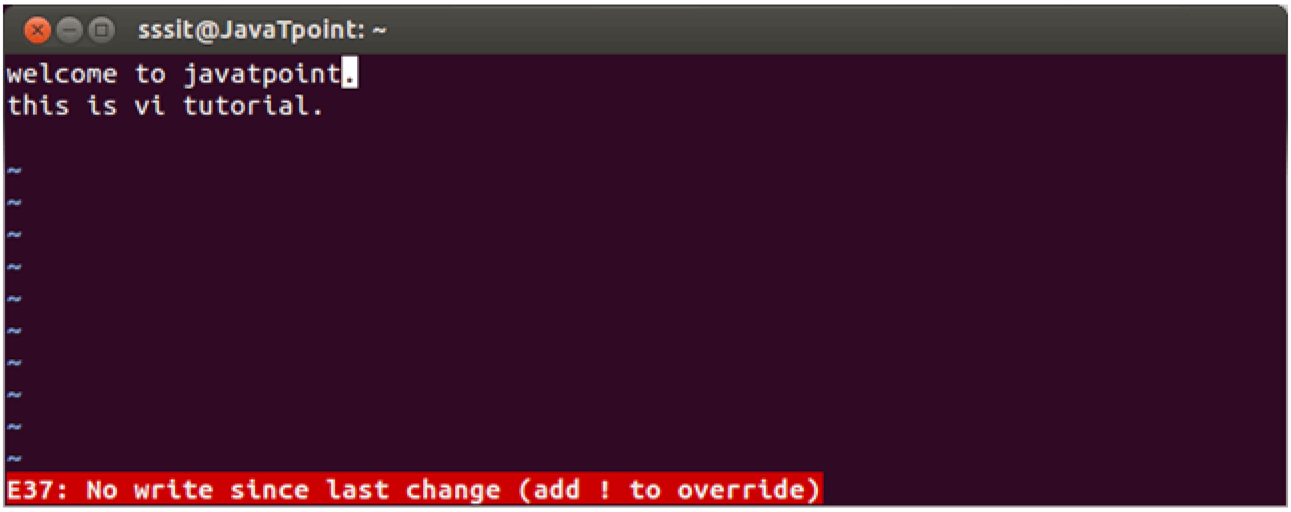


```
welcome to javatpoint
this is vi tutorial.

~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

Look at the above snapshot, command :wq will save and quit the vi editor. When you'll type it in command mode, it will automatically come at bottom left corner.

If you want to quit without saving the file, use **:q.** This command will only work when you have not made any changes in the file.

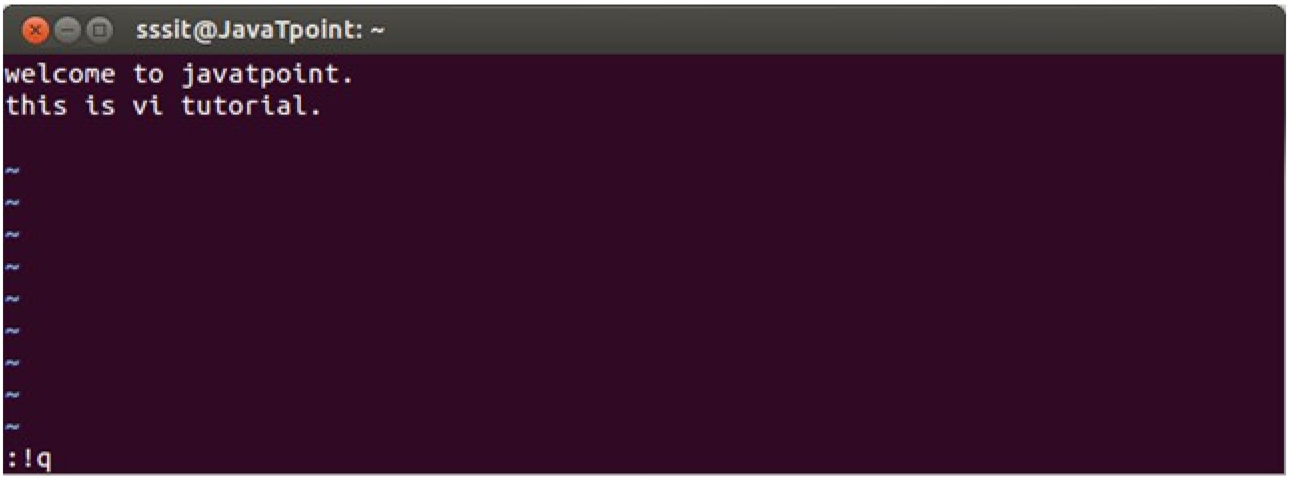


```
welcome to javatpoint.
this is vi tutorial.

~
~
~
~
~
~
~
~
~
~
E37: No write since last change (add ! to override)
```

Look at the above snapshot, this file is modified and hence on typing **:q** it displays this message at bottom left corner.

The above file can be saved with the command **:!q.** It discards the changes made in the file and save it.



```
welcome to javatpoint.
this is vi tutorial.

~
~
~
~
~
~
~
~
~
:!q
```

Look at the above snapshot, we have typed **:!q,** it will save our file by discarding the changes made.

Vi Commands

Linux vi editor is different from other editors. You have to use different keys to use different functions. Although, it's quite easy and interesting to use vi editor.

The vi editor commands are case sensitive.

Have a look at the vi commands in the following table.

To switch from command to insert mode:

Command	Action
i	Start typing before the current character
I	Start typing at the start of current line
a	Start typing after the current character
A	Start typing at the end of current line
o	Start typing on a new line after the current line
O	Start typing on a new line before the current line

To move around a file:

Commands	Action
j	To move down
k	To move up
h	To move left
l	To move right

To jump lines:

Commands	Action
G	Will direct you at the last line of the file
``	Will direct you to your last position in the file

To delete:

Commands	Action
x	Delete the current character
X	Delete the character before the cursor
r	Replace the current character
xp	Switch two characters
dd	Delete the current line
D	Delete the current line from current character to the end of the line
dG	delete from the current line to the end of the file

To repeat and undo:

Commands	Action
u	Undo the last command
.	Repeat the last command

Command to cut, copy and paste:

Commands	Action
dd	Delete a line
yy	(yank yank) copy a line
p	Paste after the current line
P	Paste before the current line

Command to cut, copy and paste in blocks:

Commands	Action
<n>dd	Delete the specified n number of lines
<n>yy	Copy the specified n number of lines

Start and end of line:

Commands	Action
θ	Bring at the start of the current line
^	Bring at the start of the current line
$	Bring at the end of the current line

dθ	Delete till start of a line
d$	Delete till end of a line

Joining lines:

Commands	Action
J	Join two lines
yyp	Repeat the current line
ddp	Swap two lines

Move forward or backward:

Commands	Action
w	Move one word forward
b	Move one word backward
<n>w	Move specified number of words forward
dw	Delete one word
yw	Copy one word
<n>dw	Delete specified number of words

Search a string:

Commands	Action
/string	Forward search for given string
?string	Backward search for given string
/^string	Forward search string at beginning of a line

/string$	Forward search string at end of a line
n	Go to next occurrence of searched string
/\<he\>	Search for the word he (and not for there, here, etc.)
/pl[abc]ce	Search for place, plbce, and plcce

Replace all

Syntax:

1. :**<startLine**,endLine**>** s/**<oldString>**/**<newString>**/g

Example:

Commands	Action
:1,$ s/readable/changed/	Replace forward with backward from first line to the last line
:3,6 s/letters/neww/g	Replace forward with backward from third line to the ninth line

Text buffers:

Commands	Action
"add	Delete current line and put text in buffer a
"ap	Paste the line from buffer a

Abbreviation

Syntax:

1. :ab **<abbreviation> <abbreviatedWord>**

Example:

Commands	Action
:ab au abbrevition and	Abbreviate au to be 'abbrevition and

unabbreviation	unabbreviation'
:una au	Un - abbreviate au