Vi Text Editor! (Text editing, the right way!)

Table of Contents

Vi Text Editor! (Text editing, the right way!)
- Introduction
- A Command Line Editor
- Saving and Exiting
- Navigating a file in Vi
- Deleting content
- Undoing
- Taking it Further
- Summary
- Exercise

VI Cheat Sheet

Cursor Movement
- Move by single character
- Jump over spaces

Navigation
- On a line
- Navigation commands

Global commands

Editing
- INSERT mode
- REPLACE mode
- Deletion
- Miscellaneous
  - Change

Modifying files

Introduction

Ref: ryanstutorials.net

Master the Vi text editor and learn how to make complex edits on your files with less time and effort.

Vi is a text editor that is most likely very different to any editor you have used before. It will take a while to get your head around but once you do you will realize it is actually quite powerful. It's kinda like touch typing, initially learning is awkward and you wonder why you're bothering but once you get the hang of it you will not want to go back.

Even if you don't use Vi all the time you will definitely find that work patterns you develop in learning the editor can be transferred easily to other programs and to great effect.

Vi is a very powerful tool. In this section my aim is not to cover everything that Vi can do but to get you up and running with the basics. At the end of the section I'll provide some links to resources where you can learn Vi further. I highly recommend you look into a few of them.

A Command Line Editor

Vi is a command line text editor. As you would be quite aware now, the command line is quite a different environment to your GUI. It's a single window with text input and output only. Vi has been designed to work within these limitations and many would argue, is actually quite powerful as a result. Vi is intended as a plain text editor (similar to Notepad on Windows, or Textedit on Mac) as opposed to a word processing suite such as Word or Pages. It does, however have a lot more power compared to Notepad or Textedit.

As a result you have to ditch the mouse. Everything in Vi is done via the keyboard.

There are two modes in Vi.

- **Insert (or Input) mode**
- **Edit mode.**

  In input mode you may input or enter content into the file. In edit mode you can move around the file, perform actions such as deleting, copying, search and replace, saving etc.

A common mistake is to start entering commands without first going back into edit mode or to start typing input without first going into insert mode. If you do either of these it is generally easy to recover so don't worry too much.

When we run vi we normally issue it with a single command line argument which is the file you would like to edit.

```
vi <filename>
```

If you forget to specify a file then there is a way to open a file within vi but it is easiest to just quit vi and have another go. Also remember that when we specify the file it can be with either an absolute or relative path.

First off let's move into your directory you created in the section on file manipulation. We're going to create a few files and this will keep them out of the way of your normal stuff.

Now we'll edit our first file.

```
user@bash:~$ vi firsfile
```

When you run this command it opens up the file. If the file does not exist then it will create it for you then open it up. (no need to touch files before editing them) Once you enter vi it will look something like this (though depending on what system you are on it may look slightly different).

```
~
~
~
~
~
"firstfile" [New File]
```

You always start off in edit mode so the first thing we are going to do is switch to insert mode by pressing **i**. You can tell when you are in insert mode as the bottom left corner will tell you.

```
~
~
~
~
~
-- INSERT --
```

Now type in a few lines of text and press **Esc** which will take you back to edit mode.

Saving and Exiting

There are a few ways to go about doing this. They all do essentially the same thing so pick whichever way you prefer. For all of these, make sure you are in edit mode first.

> **Tip**
>
> If you are unsure if you are in edit mode or not you can look at the bottom left corner. As long as it doesn't say INSERT you are fine. Alternatively you can just press Esc to be sure. If you are already in edit mode, pressing Esc does nothing so you won't do any harm.

- **ZZ** (Note: capitals) - Save and exit
- **:q!** - discard all changes, since the last save, and exit
- **:w** - save file but don't exit
- **:wq** - again, save and exit

Most commands within vi are executed as soon as you press a sequence of keys. Any command beginning with a colon **(:)** requires you to hit to complete the command.

Navigating a file in Vi

Below are some of the many commands you may enter to move around the file. Have a play with them and see how they work.

- **Arrow keys** - move the cursor around
- **j, k, h, l** - move the cursor down, up, left and right (similar to the arrow keys)
- **^ (caret)** - move cursor to beginning of current line
- **$** - move cursor to end of the current line
- **nG** - move to the nth line (eg 5G moves to 5th line)
- **G** - move to the last line
- **w** - move to the beginning of the next word
- **nw** - move forward n word (eg 2w moves two words forwards)
- **b** - move to the beginning of the previous word
- **nb** - move back n word
- **{** - move backward one paragraph
- **}** - move forward one paragraph

If you type **:set nu** in edit mode within vi it will enable line numbers. I find that turning line numbers on makes working with files a lot easier.

Deleting content

Below are some of the many ways in which we may delete content within vi. Have a play with them now. (also check out the section below on undoing so that you can undo your deletes.)

- **x** - delete a single character
- **nx** - delete n characters (eg 5x deletes five characters)
- **dd** - delete the current line
- **dn** - d followed by a movement command. Delete to where the movement command would have taken you. (eg **d5w** means delete 5 words)

Undoing

Undoing changes in vi is fairly easy. It is the character **u**.

- **u** - Undo the last action (you may keep pressing u to keep undoing)
- **U** (Note: capital) - Undo all changes to the current line

Taking it Further

We can now insert content into a file, move around the file, delete content and undo it then save and exit. You can now do basic editing in vi. This is just touching the surface of what vi can do however. There are many vi cheat sheets out there too which list all the commands available to you.

- copy and paste
- search and replace
- buffers
- markers
- ranges
- settings
- Have fun and remember to keep at it. vi will be painful at first but with practice it will soon become your friend.

Summary

> **Note**
>
> **vi** Edit a file.
> **cat** View a file.
> **less** Convenient for viewing large files.

> **Important**
>
> **No mouse**
> vi is a text editor where everything is done on the keyboard. If you can touch type then this is great. If not then maybe you should think about learning.

> **Edit commands**
> There are many of them. Practice is the key to remember the most commonly used and useful ones.

Exercise

1. Start by creating a file and putting some content into it.
2. Save the file and view it in both cat and less
3. Go back into the file in vi and enter some more content.
4. Move around the content using at least 6 different movement commands.
5. Play about with several of the delete commands, especially the ones that incorporate a movement command. Remember you may undo your changes so you don't have to keep putting new content in.

VI Cheat Sheet

The following document is a minimal reference for the **vi** text editor [Ref: gist.github.com/Tarang7/vi-cheat-sheet.md](gist.github.com/Tarang7/vi-cheat-sheet.md).

To start **vi**:

```
vi [filename]
```

There are two basic modes in vi:

1. **Visual mode (default)** - view the contents of a file
2. **Insert Mode** - modify the contents of a file

Additionally, commands can be executed using either vi or ex shortcuts.

vi commands are registered immediately based on sequential keystrokes, whereas ex commands begin with a colon `:[command]` and only execute when the user presses `Enter`.

The following sequence generally defines the structure of a vi command. When in VISUAL mode, commands are immediately executed.

```
[quantifier?][command][argument?]
```

Quantifiers are typically numeric values that specify the number of times to repeat the adjacent command.

Commands perform text processing.

Arguments are commonly characters, words, or lines that can be passed immediately after a command.

Cursor Movement

Move by single character

`h` / `←` move **LEFT**

`j` / `↓` move **DOWN**

`k` / `↑` move **UP**

`l` / `→` move **RIGHT**

`Space` and `Backspace` can also be used to navigate left and right in command mode,

Jump over spaces

*The following shortcuts allow the user to jump to the closest **chunk** of characters, where a chunk is any string that does not contain a **space** .*

*Here **word** considers punctuation to be it's own chunk of characters, whereas **WORD** treats punctuation as a normal character in the chunk.*

`w` jump to **start** of next word

`W` jump to **start** of next WORD

`e` jump to **end** of next word

`E` jump to **end** of next WORD

`b` jump **back** to start of previous word

`B` jump **back** to start of previous WORD

Navigation

On a line

`0` go to **first** character

`$` go to **last** character

`^` go to **first non-blank** character

Navigation commands

`G` go to **last** line

`n` `G` go to line *n*

Global commands

`Esc` **Escape** to command mode

`u` undo previous command

`.` repeat last command

Editing

`Insert` toggle INSERT/REPLACE mode

INSERT mode

INSERT mode adds characters after the cursor.

`i` enter INSERT mode **at cursor**

`I` enter INSERT mode **at start of line**

`a` enter INSERT mode **after cursor**

`A` enter INSERT mode **at end of line**

`s` delete character at cursor and enter INSERT mode

`S` delete line and enter INSERT mode at cursor

REPLACE mode

REPLACE mode replaces characters at the cursor.

`r` enter REPLACE mode for a *single* character

`R` enter REPLACE mode at cursor

Deletion

`Delete` or `x` delete character at cursor

`Backspace` or `X` delete character before cursor

`d` `motion` delete character

`d` `d` delete all characters in line

`D` delete characters from cursor to end of line

Miscellaneous

`j` **join** next line to current line

Change

`c` `motion` delete characters based on movement commands, then enter INSERT mode

`C` delete characters from cursor to end of line and enter INSERT mode

Modifying files

`:e` reload current file