

Learn Vim: Step by Step

Teach yourself Vim

Javier Tiá / October 13, 2015

javier.tia@gmail.com or javier.tia@hpe.com

Level 1 - Survive

Only five commands

- **i** → Insert mode. Type ESC to return to Normal mode
- **x** → Delete the char under the cursor
- **:wq** → Save and Quit (**:w** **save**, **:q** quit)
- **dd** → Delete (and copy) the current line
- **p** → Paste

Cursor move

- **hjk^l** (highly recommended but not mandatory), basic cursor move (← ↓ ↑ →). Hint: **j** looks like a down arrow
- **:help <command>** → Show help about . You can use **:help** without a to get general help

Level 2 - Feel comfortable

Insert mode variations

- **a** → insert after the cursor
- **o** → insert a new line after the current one
- **O** → insert a new line before the current one
- **cw** → replace from the cursor to the end of the word

Basic moves

- **0** → go to the first column
- **^** → go to the first non-blank character of the line
- **\$** → go to the end of line
- **g_** → go to the last non-blank character of line
- **/pattern** → search for pattern

Copy/Paste and Undo/Redo

- **P** → paste before, remember **p** is paste after current position
- **yy** → copy the current line, easier but equivalent to **ddP**
- **u** → undo
- **<Ctrl-r>** → redo

Load/Save/Quit/Change File

- **:e** → open
- **:w** → save
- **:saveas** → save to
- **:wq** → save and quit
- **:q!** → quit without saving, also **:qa!** to quit even if there are modified hidden buffers
- **:bn** (resp. **:bp**) → show next (resp. previous) file (buffer)

**3rd Level – Better &
Stronger & Faster**

Better

- `.` → will repeat the last command
- `N<command>` → will repeat the command *N* times

Better example

- **2dd** → will delete 2 lines
- **3p** → will paste the text 3 times
- **5idesu [ESC]** → will write "desu desu desu desu desu"

Stronger

- **NG** → go to line *N*
- **gg** → shortcut for **1G** - go to the start of the file
- **G** → go to last line

Stronger - Word moves

- **w** → go to the start of the following word
- **e** → go to the end of this word
- **W** → go to the start of the following WORD
- **E** → go to the end of this WORD

```
          ew          E W
          ||          ||
x = (name_1,vision_3); # this is a comment.
```

Stronger - More efficient moves

- % → go to the corresponding (, {, [characters
- * → go to next occurrence of the word under the cursor
- # → go to previous occurrence of the word under the cursor
- **n** and **p** → same as * and #

Faster

<start position><command><end position>

0y\$ means:

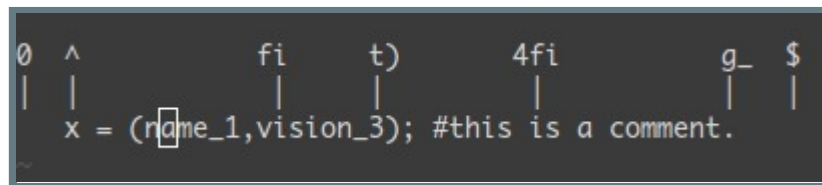
- **0** → go to the beginning of this line
- **y** → yank from here
- **\$** → up to the end of this line

4th Level – Vim

Superpowers

Move on current line

- **0** → go to column **0**
- **^** → go to first character on the line
- **\$** → go to the last column
- **g_** → go to the last character on the line
- **fa** → go to next occurrence of the letter **a** on the line
- **;** → will find the next occurrence
- **t,** → go to just before the character **,**
- **3fa** → find the 3rd occurrence of **,** on this line
- **F** and **T** → like **f** and **t** but backward



The diagram illustrates the movement of a cursor on the line of code `x = (name_1, vision_3); #this is a comment.` using various navigation commands. Vertical lines with labels above them indicate the cursor's position at different points:

- 0**: At the start of the line (column 0).
- ^**: At the first character, `x`.
- fi**: At the first occurrence of the letter `i` in the word `vision`.
- t)**: Just before the closing parenthesis `)` of the function call.
- 4fi**: At the fourth occurrence of the letter `i` in the word `comment`.
- g_**: At the last character on the line, the period `.`.
- \$**: At the end of the line.

Zone selection

<action>a<object> and **<action>i<object>**

Suppose the cursor is on the first o of *(map (+) ("foo"))*

- **vi"** → will select *foo*
- **va"** → will select *"foo"*
- **vi)** → will select *"foo"*
- **va)** → will select *("foo")*
- **v2i)** → will select *map (+) ("foo")*
- **v2a)** → will select *(map (+) ("foo"))*

Select rectangular blocks

- **^** → go to the first non-blank character of the line
- **<Ctrl-v>** → Start block selection
- **<Ctrl-d>** → move down; could also be **j****j****j** or % or ...
- **I - -** and [ESC] → write - - to comment each line

Macros

- **qa** → record your actions in the register **a**
- **@a** → will replay the macro saved into the register **a** as if you typed it
- **@@** → is a shortcut to replay the last executed macro

A Macro to Count Until 10

qaYp<C-a>q

- **qa** → start recording
- **Yp** → duplicate this line
- **<C-a>** → increment the number
- **q** → stop recording
- **@a** → write 2 under the 1
- **@@** → write 3 under the 2
- **10@@** → will create a list of increasing numbers until 13

Visual selection

- **v** → select character by character
- **V** → select a whole line
- **J** → join all the lines together
- **<** and **>** → indent to the left and right
- **=** → auto indent

Level 5th - Vim as IDE

Featured plugins

spf13-vim - A Vim Distribution

- It has an excellent configuration
- `.vimrc` file is suited for programming
- It fixes many of the inconveniences of vanilla Vim config
- For personal customizations use: `~/ .vimrc.local` and `~/ .vimrc.bundles.local` (additional plugins)

spf13-vim Install and Update

Install

```
$ curl http://j.mp/spf13-vim3 -L -o - | sh
```

Update

```
$ alias update-spf13-vim="( \
  cd $HOME/.spf13-vim-3 \
  && git pull \
  && vim +BundleInstall! +BundleClean +q )"
```

Vundle - A Plugin Manager

- spf13-vim uses the Vundle plugin management system to have a well organized vim directory
- It ensures that the latest versions of your plugins are installed
- It make easy to keep the plugins up to date

NERDTree - File Navigation Plugin

- NERDTree is a file explorer plugin
- **<Leader>e** → quickstart launch

Ctrlp - Fast File Finder

- It provides an intuitive and fast mechanism to load files (with regex and fuzzy find)

Surround - Managing All the ""[]{}

- A plugin for dealing with pairs of "surroundings": single/double quotes, parentheses, etc...
- Closely related to Vim text-objects

Old text	Command	New text
"Hello *world!"	ds"	Hello world!
[123+4*56]/2	cs])	(123+456)/2
"Look ma, I'm *HTML!"	cs"	<i>Look ma, I'm HTML!</i>
if *x>3 {	ysW(if (x>3) {
my \$str = *whee!;	vlllS'	my \$str = 'whee!';

NERDCommenter - Comment++

- Allow to comment code regardless of filetype
- ***N*<Leader>cc** → comment out the current line or *N* text selected in visual mode
- ***N*<Leader>cu** → uncomments the selected *N* line(s)
- **:help NERDCommenter** → for more details

Syntastic - Integrated Syntax Checking

- Plugin to check syntax with external syntax checkers as the files are saved or/and opened
- If syntax errors are detected, the user is notified
- No need to compile the code to find error
- **:help syntastic** → for more details

Numbers - Better Line Numbers

- It alternate between relative numbering and absolute numbering
- It allows you to easily move code around

YouCompleteMe - A Code-Completion Engine

- A fast, as-you-type, fuzzy-search code completion engine for Vim
- An identifier-based engine that works with every programming language
- A Clang-based engine that provides native semantic code completion for The C-family languages
- Integrated with UltiSnips (A snippet solution for Vim)

Tagbar - Tag Generation and Navigation

- It provides a panel to navigate easily via tags
- It requires exuberant-ctags and will automatically generate tags for your open files
- **<Ctrl-]>** → while cursor is on keyword, such as function name, to jump to its definition
- **<Ctrl-T>** → jump back up to origin
- **<Leader>tt** → to toggle the tagbar panel

EasyMotion - Jump Anywhere

- It provides an interactive way to use motions in Vim
- It quickly maps each possible jump destination to a key allowing very fast and straightforward movement
- **<Leader><Leader>w** → quickstart EasyMotion

Some final words

- Learn no more than one or two new commands per day
- You can see progress after two to three weeks
- The learning curve is steep, but the reward is high
- Practice and practice. Use `vimtutor` and `:help usr_02.txt`
- Vim has an excellent documentation
- As a programmer handling text is important. Vim is a powerful tool for it

References

- This presentation: <https://github.com/jetm/learn-vim-steps/> or pdf file.
- Presentation made with: <http://lab.hakim.se/reveal-js/>
- Presentation based in <http://yannesposito.com/Scratch/en/blog/Learn-Vim-Progressively/>
- My `.vimrc.local` file
- My `.vimrc.bundles.local` file

Contact information

- javier.tia@gmail.com / javier.tia@hpe.com

Questions?