Vim lectures

The beginning

Michał Syposz

July 3, 2020

My idea behind the lectures

- Not all lectures will have corresponding presentations.
- I will try to write some sort of note to every lecture.
- Lectures are not going to be long. I don't want to show you every command and aspect of Vim I know since there's no way you're going to remember and use all of it after one meeting.
- I want to encourage terminal usage, I will put most of the lecture notes on github.[1]

[1] If someone doesn't remember how to properly use github it's good opportunity for them. I can send some good materials explaining the usage of git

Vim vs NeoVim

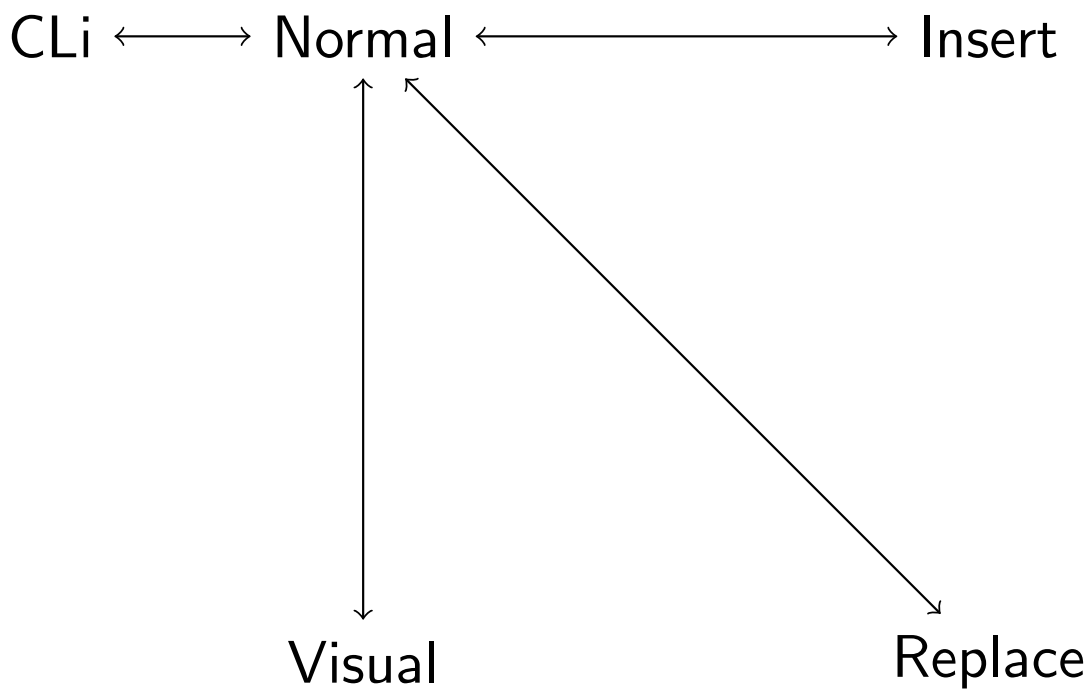
So let's get one thing straight. When we're talking about Vim we don't really mean Vim. We mean NeoVim.
Basically Vim creator was close-minded on many features he could add to it. Some developers decided to fork Vim and write extensions themselves. That's how NeoVim was created. It was much faster than normal Vim because for example it was asynchronous. After Vim creator found out about it he **copied all the features he was refusing to write for years**. As smart people we don't want to support anyone like that. Therefore we use NeoVim instead of Vim.
Quick disclaimer: everything I'm going to show you works in Vim as well.

Motivation behind learning Vim

1. Programming is not really about writing code, it's mostly about modyfing it. Vim possibilities makes that super fast.
2. Vim is fast AF. I think like only nano is faster to open big files.
3. It's almost everywhere. If I SSHed into a server and there would be no Vi/Vim/NeoVim installed I would be surprised.
4. You don't need to use mouse anymore. Using mouse a lot is bad for your back apparently.
5. It makes programming smooth, fast and exciting. You'll see.

Modes

Vim has modal editing. That means:
Vim has 4 real modes: Normal, Insert, Visual and Replace.
There's a fifth called command line mode.



Modes

The modes' names are self explanatory.

- Normal - this mode is the reason why Vim is so awesome. You can use almost all commands that come with Vim in it.
- Insert - it's for inserting text into the file.
- Replace - it's like insert but it's replacing the text you're writing on.
- Visual - highlighting
- Command line - you can launch Vim commands from it. Even bash or ZSH commands!

Modes

There's lots of ways you can change modes. I will show you some easy ones for now.
First of all to change any mode to Normal you press `Escape`.
Since you use it a lot it might be a good idea to remap it. [2]
As you can see changing modes is only possible from Normal mode. These are **basic** commands to achieve that:
Entering insert mode: `i`
Entering replace mode: `r,R`
Entering visual mode: `v,V,ctrl+v`
Entering command line mode: `:`

[2] Clue: probably as me you don't use CapsLock a lot.

Fundamental Movements

Better arrow keys in vim

$\leftarrow$ `h`
$\downarrow$ `j`
$\uparrow$ `k`
$\rightarrow$ `l`

If you use touch typing [3]. As you probably as self respecting programmer should those letters are under your right hand.

[3] I can provide some tutorial link's for that as well

Fundamental Movements

Moving over by h/l might be really annoying. Luckily there are easy commands to save us.

`w` - jump to the beginning of next word

`e` - jump to the end of next word (not used every often)

`b` - jump to the beginning of previous word

Basic editing commands

`y` - stands for yank $\Leftrightarrow$ copy
`yy` - copies current line to Default register [4]
`d` - stands for delete. Deleting some texts puts it in the default register as well
`dd` - deletes current line
`p` - pastes what's in Default register
`u` - undoes When you want to edit something more than a line you can use visual mode. In visual mode movement and editing commands work as in normal.

[4] I'm planing to discuss registers later in the lectures

Combining what we know

Now I'll quickly demonstrate what you can do with commands I've already showed you. Visual + movement + editing commands.

Most important command you'll learn this lecture

You save and exit files in Vim from command line mode. All you need to write is:
`:w` to save the file
`:q` to exit Vim
You can combine these commands to save file and exit vim `:wq` in short `:x`. Exiting Vim is not always what you want to do. I'll talk about some alternatives in later lectures.

I think these commands are enough to start your adventure with vim.
For now it's probably best if you just use a plug in for your favorite editor, since we didn't modify vim at all. We'll get to the point of using vim as your primary editing tool in couple of meetings. If you feel like learning more vim by yourself I highly recommend ThePrimeagen on YouTube. He recently made a vim intro series, which the beginning of these lectures is based upon.

See you in the next one