Getting started with vim

Md Arif Shaikh

December 19, 2020

Contents

1 Installing Plugins

2 Python in vim

One of the most important task of my job is to write python code. That is why, any editor that I choose must be capable of being a good python editor. Bellow are the things that an editor should be able do provide:

1. **Autocompletion:** For autocompletion of python code one need the `vim-jedi` plugin. To bring up the auto completion one has to use `C-space`. Also the autocompletion automatically works after dot.

   Launch mvim from the terminal:
   Thing to keep in mind is that the jedi-vim uses the python installed in the system if the vim application is launched from say the spotlight in Mac. In that case it will not use the conda environment. Therefor it is best to launch it from the command line where you are already in a conda env. In that way it will have access and will do autocompletion for all the modules and packages installed in the conda env.

2. **flake8:** flake8 highlights any deviation from pep8 style guideline as well as error in the python code. This could be done in vim in the following steps.

   - First install flake8 using pip: `pip install flake8`.
   - Install flake8 plugin in .vimrc: Plugin `'nvie/vim-flake8'`
   - If vim cannot find flake8 then we need to provide the flake8 plugin the absolute path of the flake8 installation. For example in the path could be obtained in terminal using `which flake8`. Now in the `.vim` folder go the folder for `vim-flake8` plugin and then to `autoload` directory where you will find the `flake8.vim` file. In that file we need to change the line which calls the flake8 package in the following way:

```
call s:DeclareOption('flake8_cmd', '','"/absolute/path/to/flake8"')
```

- Now that flake8 is currently working, the last thing we want is to run the flake8 command each time we save the python file. This is done easily by putting the following line in the `.vimrc`: `autocmd BufWritePost *.py call Flake8()`.
- Often it is nice to have a marker in the file itself and also at the begining of the line showing the error or the warning. This could be accomplished using the following settings in the `.vimrc`

```
let g:flake8_show_in_file = 1
let g:flake8_show_in_gutter = 1
```

3. **Correct indentation:** Sometimes when inside a bracket a python line becomes too long we have make a line break but this might mess the indentation. To rectify the indentation put the cursor on one of the brackets and in normal mode and use `=%`.
4. **Commentting and uncommenting:** It's always handy to know how to comment out multiple line of code. In vim this can be done in the following way.
   (a) start by selecting a block using `Control+v` and using `j` or `k` to go down or up.
   (b) After the selecting the lines use `Shift+I` to insert the desired commenting character. For example in python this would be `#` .
   (c) Enter `ESC` and it will do the job.
   (d) For uncommenting, select the lines using the first step and use `x` to uncommnet.

3 Latex in vim

Apart from programming, an integral part of my academic life is to write documents for publications of my research and for that I have to write a lot of latex files. For the same reason, I have to figure out how to vim efficiently to write `tex` files and compile. Luckily there exist a good Plugin named `vimtex` which fulfill almost all of my requirements for writing latex files.

Bellow are some commands useful when writing latex files:

Start compiling on save	`\ll` or :VimtexCompile
Show errors	`\le`
after writing `\begin{env}` complete it	`]]`
set spell checking	`:set spell`
correct spell on point	`z=`

How to use `\ll` in `Normal` mode to view the line on cursor in the pdf file opened in Skim?: This could be accomplished just by adding the following lines in the `.vimrc` file.

```
let g:vimtex_view_general_viewer = '/Applications/Skim.app/Contents/SharedSupport/displayline'
let g:vimtex_view_general_options = '-r @line @pdf @tex'
let g:vimtex_view_general_options_latexmk = '-r 1'
```

4 Snippets in vim

To make one's workflow one should use Snippets. In vim this nicely accomplished using `UltiSnips`. To create and use snippets one should follow the steps described below.

1. First install the plugin using in `.vimrc`.

```
Plugin 'SirVer/ultisnips'
```

2. Next make a directory named `UltiSnips` inside `.vim` or symlink it there.
3. Now let's say we want to create snippets for `.tex` files. We create a file named `tex.snippets`.
4. We are now ready to create snippets for latex insdie that file. template of snippets is the following

```
    snippet eq "equation" b
    \begin{equation}
\label{eq:$1}
$0
    \end{equation}
    endsnippet
```

5 Diectories in vim

Simply use `:e` to explore file systems. Once in side the nerd tree view, it's easy to see all the commands using the help me by pressing `?`. Here are few of the Shortcuts one needs

Key	Action
`:u`	up a directory
`:C-j`	move to next child
`:C-k`	move to previous child
`m`	open the menu for adding/deleting sud directories.

6 Shortcuts

Buffers	
Open a new buffer midrule Open a terminal inside vim	:new :term
Switch between windows	C-w C-w
close/unload/delete the current buffer	:bd
Navigation	
Beginning of the buffer	1G
End of buffer	G
Go to nth line	nG
Beginning of line	0
End of line	$
Open new line after cursor	o
Open new line before cursor	O
Delete the current line	dd

Move down by one screen	`C-f`
Move up by one screen	`C-b`
In the given screen move to top	`H`
In the given screen move to midlle	`M`
In the given screen move to the last line	`L`
Copy & paste	
Copy the current line	`yy`
Paste the copied line below the cursor	`p`
Paste the copied line above the cursor	`P`
Visual marking	
start marking	v
select the region	using arrows
cut the marked region	d
copy the marked region	y
paste the copied/cut text	p