**Configure iTerm2 and Vim like a Pro**

[Yong Su](https://medium.com/@jeantimex?source=post_page-----e303d25d5b5c-----------------------------------)

[Yong Su](https://medium.com/@jeantimex?source=post_page-----e303d25d5b5c-----------------------------------)

[Dec 2, 2019·7 min read](https://medium.com/@jeantimex/how-to-configure-iterm2-and-vim-like-a-pro-on-macos-e303d25d5b5c?source=post_page-----e303d25d5b5c-----------------------------------)

A screenshot of a computer

Description automatically generated with medium confidence

Introduction

Every time when I got a new computer or reinstalled the macOS, the first thing I would like to do is to set up my terminal and make it look cool. There are several tools I always like to gear up for the terminal including the following:

* iTerm2
* Oh My Zsh
* Powerlevel10k (The coolest theme for Zsh)
* Vim Airline (Vim status bar)
* NERDTree (A file system tree for Vim)
* FZF (fuzzy finder)

There are already a lot of great articles that teach people how to install and configure the above tools, however, in order to set them up correctly, usually I need to refer to at least 5 separate articles and it took me a while to figure out the necessary steps, and most importantly I don’t remember what I did last time. So I decided to put all I have learned together and write them down, this article will show you how to install and configure all the above tools with necessary and concrete steps, and hope it helps.

Ok let’s get started.

Instructions

***Step 1. Install iTerm2***

Make sure you have [Homebrew](https://brew.sh/) installed, if not, simply paste the following in your Terminal:

$ /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"

During the above installation, you may be asked to install the command line developer tools for xcode-select command, go ahead and install it and then finish the Homebrew installation, once it’s completed, you will be able to use brew command to install iTerm2.

$ brew cask install iterm2

***Step 2. Install Oh My Zsh***

[Oh My Zsh](https://ohmyz.sh/) is a framework for managing Zsh configuration and it comes with many helpful plugins, themes etc to make you enjoy shell scripting. Install Oh My Zsh is simple:

$ sh -c "$(curl -fsSL https://raw.github.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"

***Step 3. Install and Configure Powerlevel10k Theme***

[Powerlevel10k](https://github.com/romkatv/powerlevel10k) is a fast reimplementation of [Powerlevel9k](https://github.com/Powerlevel9k/powerlevel9k) which is probably the most popular Powerline theme for Zsh.





Run the following commands to install Powerlevel10k:

$ git clone --depth=1 <https://github.com/romkatv/powerlevel10k.git> ~/powerlevel10k  
$ echo 'source ~/powerlevel10k/powerlevel10k.zsh-theme' >>! ~/.zshrc

Now launch iTerm2 and let’s configure Powerlevel10k theme using the configuration wizard.



A screenshot of a computer

Description automatically generated with medium confidence



Text

Description automatically generated

First install the Meslo Nerd Font, type y. Once the font is installed, restart iTerm2 and continue with the configuration. The wizard will then verify if the font is installed properly by asking you whether you can see some symbols correctly including a diamond, a lock, a Debian logo and some other icons, type y if you see them.

Next choose a prompt style:



A screenshot of a computer

Description automatically generated with medium confidence

The Rainbow style is very popular, I usually go with that. After choosing the prompt style, there are several styles for you to choose, including:

* Show current time?
* Display icons?
* Enable transient prompt?
* Prompt separators, heads, tails, height, spacing, and flow

Once it’s done, apply the changes to ~/.zshrc.

Now the Powerlevel10k theme is installed and your screen may look like this:



Text

Description automatically generated

I don’t like the color, let’s fix it, open iTerm2 preferences dialog, go to Profiles > Colors, choose Solarized Dark color preset:



Graphical user interface, application

Description automatically generated

Now your screen should look like the following, the color looks much better, isn’t it?



Text

Description automatically generated



Graphical user interface, text

Description automatically generated

Well done, we just completed the basic setup for our iTerm2. Let’s move on to configure Vim.

***Step 4. Install and Configure Vim Airline***

[Vim Airline](https://github.com/vim-airline/vim-airline) is an awesome Vim plugin that polishes the status bar, it provides a lot of styles.



Graphical user interface, text, website

Description automatically generated

In order to install Vim Airline, let’s ask a Vim plugin manager to help us, I use [Vim Plug](https://github.com/junegunn/vim-plug), run the following command to install it.

$ curl -fLo ~/.vim/autoload/plug.vim --create-dirs \  
 https://raw.githubusercontent.com/junegunn/vim-plug/master/plug.vim

Once Vim Plug is installed, edit ~/.vimrc:

call plug#begin('~/.vim/plugged')Plug 'vim-airline/vim-airline'  
Plug 'vim-airline/vim-airline-themes'call plug#end()

Save the changes and exit Vim. Now reopen Vim and type in :PlugInstall and hit Enter.



Text

Description automatically generated



Text

Description automatically generated

Now both Vim Airline and Vim Airline Themes are installed. If you open a file, you should be able to see the status bar like the following:



Text

Description automatically generated

The status bar doesn’t look like the one from Vim Airline official website, that’s because we have to configure the fonts, run the following commands to install the missing fonts for powerline theme:

$ git clone <https://github.com/powerline/fonts.git> --depth=1  
$ cd fonts  
$ ./install.sh  
$ cd ..  
$ rm -rf fonts

Finally, edit ~/.vimrc and add the following line:

let g:airline\_powerline\_fonts = 1

This will automatically populate the g:airline\_symbols dictionary with the powerline symbols.

Now our Vim editor should have a better look:



Text

Description automatically generated

***Step 5. Install and Configure FZF***

[FZF](https://github.com/junegunn/fzf) allows you to do fuzzy search for files and folders in the Terminal and Vim, we can use brew to install it.

$ brew install fzf# To install useful key bindings and fuzzy completion:  
$ $(brew --prefix)/opt/fzf/install

Follow the prompts and answer y(yes), once it’s done, the following has been added to ~/.zshrc:

[ -f ~/.fzf.zsh ] && source ~/.fzf.zsh

Add some useful FZF options in the ~/.zshrc:

[ -f ~/.fzf.zsh ] && source ~/.fzf.zshexport FZF\_DEFAULT\_OPTS=‘—height=40% —preview=“cat {}” —preview-window=right:60%:wrap’

Basically it tells FZF to show the preview window when performing the search.

Now when you type infzf and Enter in iTerm2, FZF will search in the current directory:



Graphical user interface, text

Description automatically generated

Use arrow keys to navigate the files, and you will see a preview window on the right.

Another way to use FZF is to type in some command and then \*\*<TAB>, for example, vim \*\*<TAB>:



Text

Description automatically generated

To use FZF in Vim, add the following line to ~/.vimrc:

set rtp+=/usr/local/opt/fzf

Save it and relaunch Vim, type in :FZF command in Vim, you will now be able to search for the file to edit:



Graphical user interface, text, chat or text message

Description automatically generated

Great, we now completed the setup of FZF!

***Step 6. Install and Configure NERDTree***

[NERDTree](https://github.com/scrooloose/nerdtree) allows us to navigate the directory inside Vim editor, to install it, let’s use Vim Plug, add the following in bold in ~/.vimrc:

call plug#begin('~/.vim/plugged')Plug 'vim-airline/vim-airline'  
Plug 'vim-airline/vim-airline-themes'  
**Plug 'scrooloose/nerdtree'**call plug#end()let g:airline\_powerline\_fonts = 1  
**let g:NERDTreeDirArrowExpandable = '▸'  
let g:NERDTreeDirArrowCollapsible = '▾'autocmd StdinReadPre \* let s:std\_in=1  
autocmd VimEnter \* if argc() == 0 && !exists("s:std\_in") | NERDTree | endif**

Run :PlugInstall command in Vim to complete the plugin installation:



Text

Description automatically generated

Now NERDTree plugin is installed, to see the directory tree, in Vim editor, type in command :NERDTree:



Graphical user interface, text

Description automatically generated

Well done, now you can easily navigate the directory inside Vim!

***Step 7. Set color scheme for Vim***

To make our Vim look more professional, I choose [OneDark](https://github.com/joshdick/onedark.vim) theme.



Text

Description automatically generated

To install the theme for Vim, first run the following command in iTerm2:

$ git clone <https://github.com/joshdick/onedark.vim.git> ~/onedark  
$ cd ~/onedark  
$ mkdir ~/.vim/colors  
$ cp colors/onedark.vim ~/.vim/colors/  
$ cp autoload/onedark.vim ~/.vim/autoload/

Then edit ~/.vimrcand add the following:

colorscheme onedark  
syntax on  
set number  
highlight Normal ctermbg=None  
highlight LineNr ctermfg=DarkGrey

Now our Vim should look much prettier :)

Conclusion

As you can see, the whole process involves a lot of referencing, I hope this article can help you set up your own development environment around iTerm2 and Vim.

Happy coding!

References

1. How to Jazz Up Your Bash Terminal — A Step By Step Guide With Pictures [[link](https://www.freecodecamp.org/news/jazz-up-your-bash-terminal-a-step-by-step-guide-with-pictures-80267554cb22/)]
2. macOS ohmyzsh + tmux + vim + iTerm2 + Powerlevel9k = Badass terminal [[link](https://blogs.technet.microsoft.com/jessicadeen/linux/macos-ohmyzsh-tmux-vim-iterm2-powerlevel9k-badass-terminal/)]
3. Beautifying your terminal with Zsh, Prezto, & Powerlevel9k [[link](https://medium.com/@oldwestaction/beautifying-your-terminal-with-zsh-prezto-powerlevel9k-9e8de2023046)]
4. How to Customize your Terminal with ZSH [[link](https://hackernoon.com/how-to-trick-out-terminal-287c0e93fce0)]
5. Solarized [[link](https://ethanschoonover.com/solarized/)]

<https://medium.com/@jeantimex/how-to-configure-iterm2-and-vim-like-a-pro-on-macos-e303d25d5b5c>