

# GIT: Branching and Merging

## GIT Default Editor Activity: Setting our default editor to VSCode

Brian Gorman, Author/Instructor/Trainer

©2017 - MajorGuidanceSolutions

# Introduction

---

Although BASH allows us edit files via a call to VIM as the default editor, sometimes we might want to take a different approach. I remember the first time VIM opened up for a message and all I could think was “How the heck do I get out of this thing!” Long story short, unless VIM is your thing, you probably would also like another, more common/modern option. Please don’t get me wrong here, I’m not “Bash”ing VIM, as it is quite a capable tool. Most of us these days are just used to a little more user-friendly.

Let’s gets started!

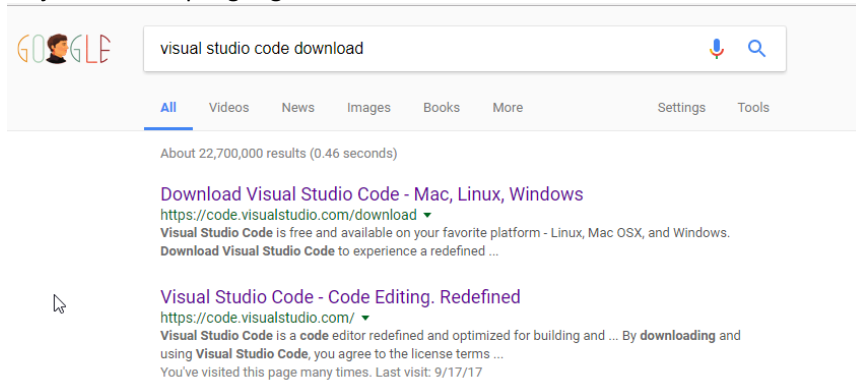
## GFBTF: Git Default Editor Activity

## Step 1: Get Visual Studio Code Setup [if you don't already have it]

- a) Download and install Visual Studio Code onto our machine.

Go To: <https://code.visualstudio.com/download>

Or just do a simple google search for Visual Studio Code:



Install the application, start it up and make sure it works.

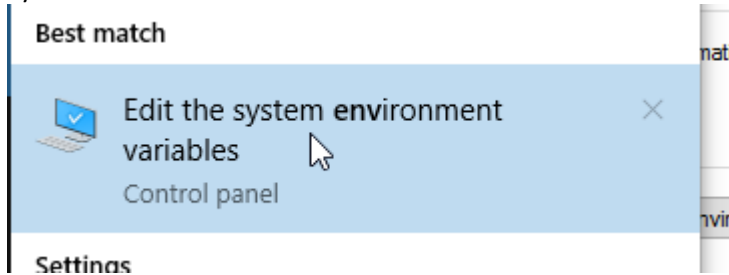
More information about the application can be found here:

<https://code.visualstudio.com/docs>

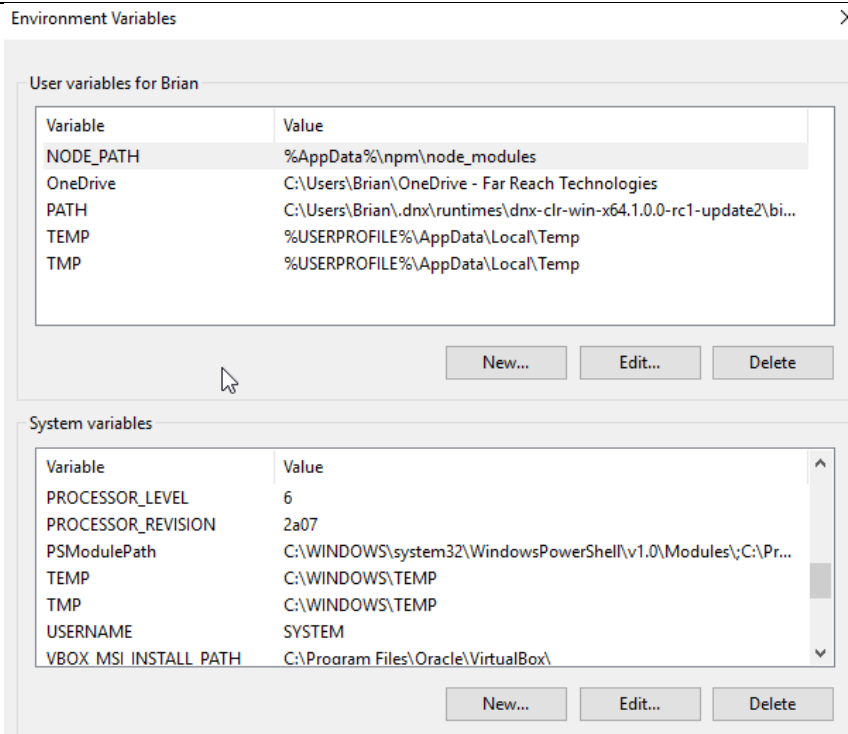
- b) If on a windows machine, make sure code is in your PATH variable.

We don't have to do this, but it will make things a lot easier. We'll be able to reference the executable directly, rather than having to code the entire path to the executable.

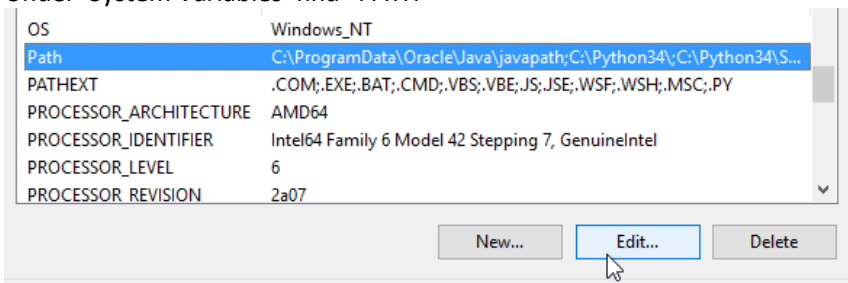
Go to the start menu and type “Environment Variables” Then select “Edit System Environment variables”



## Notes



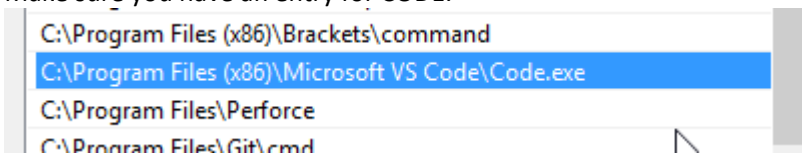
Under 'System Variables' find "PATH"



Select Edit.

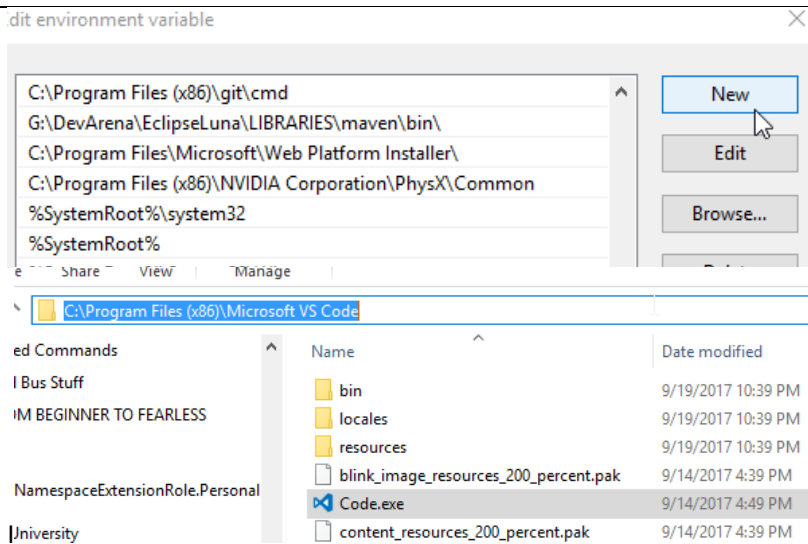
If you are on an older version of windows, you may need to parse the string in a text editor. It's much easier now in Windows 10:

Make sure you have an entry for CODE:



If you do not, then you will want to add one using the "NEW..." button:

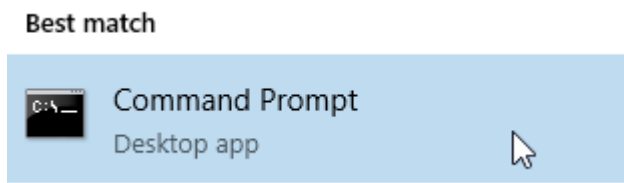
First find the path to your executable:



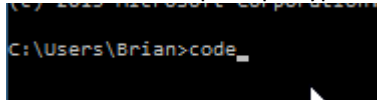
Then place that path into your Environment variables.

#### c) Test that it works:

Go to the start menu and type 'cmd'  
Then select "Command Prompt"



In the command prompt, type "code":



If VS Code launches, you are all set. If not, you can either try again or just use the full path from above when setting values in the git config for difftool.

## Step 2: Go to any repository

- a) Make sure you are on any working repository. It doesn't even have to be up to date. We are not going to be affecting anything.

Our goal is to get our global config to have one entry for a default editor:

```
core.editor='C:\Program Files (x86)\Microsoft VS Code\code.exe' -w
core.excludesfile=C:/Users/Brian/.gitignore
```

For code, the critical item is the "-w" flag. Without it, BASH won't wait until we are done editing to complete our action. This would cause a big problem for some items later on. Also, it might be sufficient to just enter "code -w" here, when code is in our environment variables. However, I'm going to leave the full path so that I have it referenced and so that we have one that shows what it

GBAM:

takes to setup the value with a full path in case environment variables are not working.

b) [Unset a global config value.](#)

Since my global config already has an entry for the core.editor, now is a great time to review how to unset a global value. You won't need to run this command now, but even if you did it wouldn't hurt anything:

```
[git config --global --unset <value>]
```

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/AmendDemo (master)
$ git config --global --unset core.editor
```

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/AmendDemo (master)
$ git config --global --get core.editor
```

c) [Set the core editor to be VSCode](#)

To make this work, it is easiest to just modify the file in VIM.

```
[git config --global -e]
```

```
[hit <i> to insert]
```

```
[enter the value core.editor]
```

```
[core]
```


```
editor='C:/Program Files (x86)/Microsoft VS Code/code.exe' -w
```

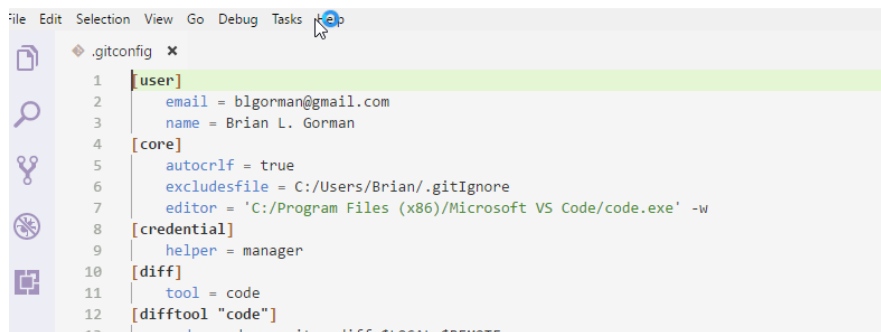
### Step 3: *Verify editor is working*

a) [To see that the editor is working, all we need to do is just try to edit the global config file:](#)

```
[git config --global -e]
```

//we know it's working if the editor pops up in code

 .gitconfig — Visual Studio Code



This concludes our GIT Default Editor Activity.

--	--

## Closing Thoughts

Setting VSCode [or another tool] to use for the default editor gives us a nice way to start working with a more user-friendly option [rather than VIM]. Again, VIM is perfectly fine if you like that tool, and so using a tool like VSCode is entirely optional.

Take a few minutes to make some notes about the various commands we've learned about in this activity, and practice using them.

## Notes

[illegible]