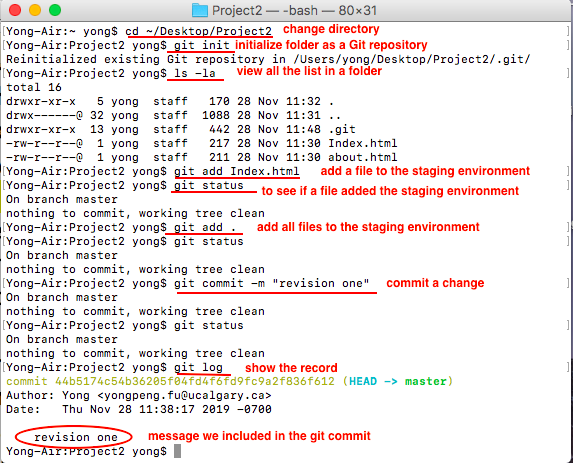
Learning Git and GitHub

**Github account: fyp20081866**

**fyp@FYP900410**

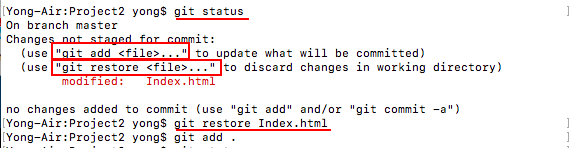
**1, Initializing, adding, committing, and status.**

* first step is to create a folder on the desktop.
* second in the folder, create 2 html files (using Sublime text)
* third open the Terminal and travel to this folder directory (“cd ~/Desktop/Project2”), and use “git init” command to initialize this folder as a Git repository
* You can tell that this folder is being tracked by Git because there is a hidden folder (.git) within this directory (run command “ls –la” to view the hidden folder)
* Git has a couple of different places to where you can store the state of files.
* One of them is called the staging environment. And you can add anything to the staging environment by using the command “git add”.
* use command “git status” to tell if the file has been added into the staging environment (if it is the case, the line will say” Changes to be committed”). And the file not added will be listed as an untracked document (meaning git does not know anything about it)
* For most of the time, you want to work on a number of files, or just add the files that you have made changes to by using the command “git add .”
* Commit: to make a record of where our environment is at this time. A commit is a record of the state of the folder. Whatever was in the staging environment will be locked as a record in the commit log.
* command of “git commit –m “notification””: you always do a –m to allow you to create a message for this commit.
* Git log: show the record in the log that shows us where we are at.
* command of “git log”



**2, Working with the staging environment** (get used to manage files in and out of the staging environment)

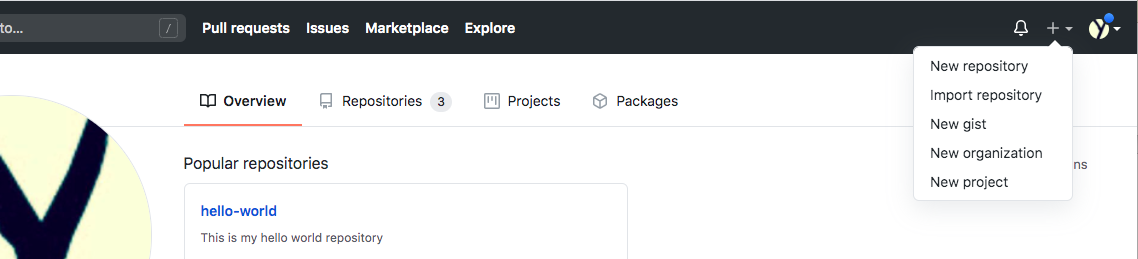
* Make a change to the Index.html file and go back to the terminus with “git status” to show what is in there:
* First, it says it’s not staged for committing, meaning the modified file is not in the staging environment yet.
* You can easily add the file using command “git add .”.
* Or you can discard the change in the working directory using command “git rectore <file>...” (meaning it will completely return back to the version before any modification were made)



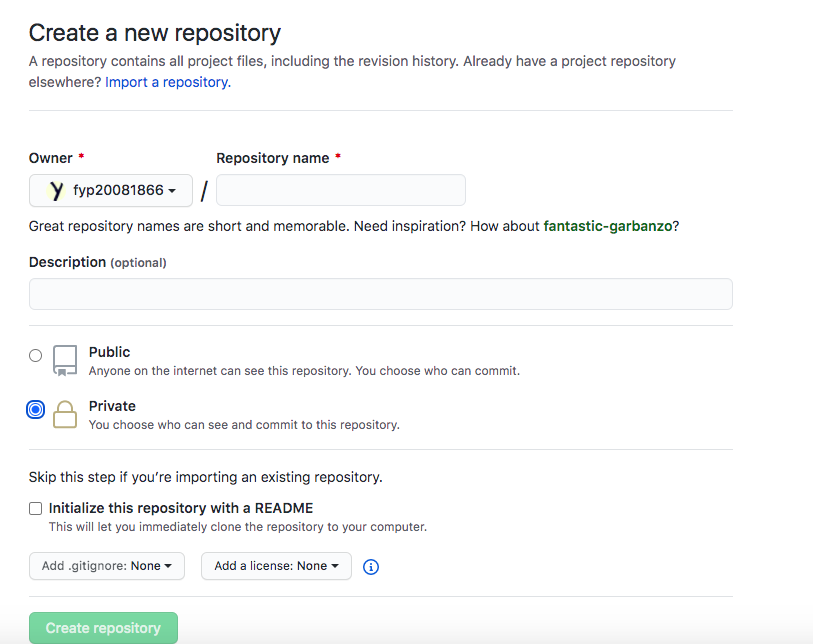
* Unstage: if something is already in the stage environment, we can take it out by using “git restore –staged <file>...”. It’s taking whatever was in our staging environment and replacing what’s in our working files environment with whatever is in staging.
* Make a change to the file, and run “git status”, it will tell you 2 things, 1) is the file you made a change on

**3, A step-by-step guide to git.** <https://opensource.com/article/18/1/step-step-guide-git>

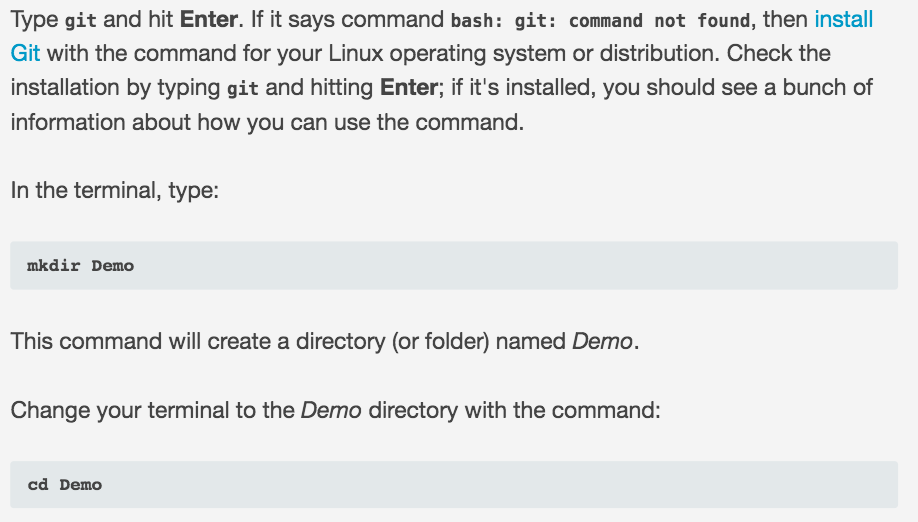
* **Git** isn't the same thing as GitHub. Git is a version-control system (i.e., a piece of software) that helps you keep track of your computer programs and files and the changes that are made to them over time. It also allows you to collaborate with your peers on a program, code, or file. **GitHub** and similar services (including GitLab and BitBucket) are websites that host a Git server program to hold your code.
* Step 1: create a GitHub account

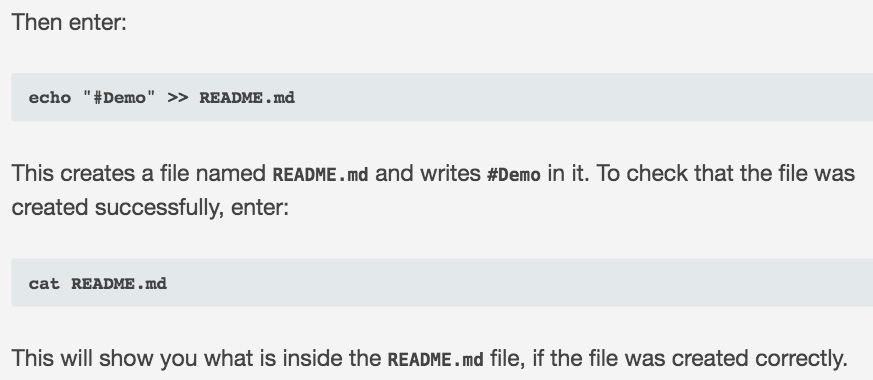


* step 2: Create a new repository
* A repository is like a place or a container where something is stored; in this case we're creating a Git repository to store code. To create a new repository, select New Repository from the + sign dropdown menu (you can see I've selected it in the upper-right corner in the image above). Enter a name for your repository (e.g, "Demo") and click **Create Repository**. Don't worry about changing any other options on this page.

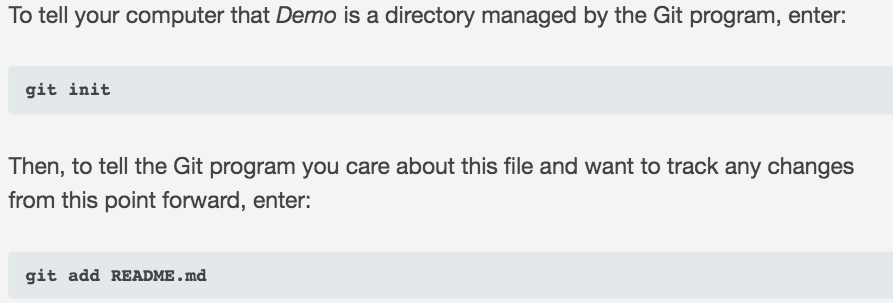


* Step 3: Create a file.

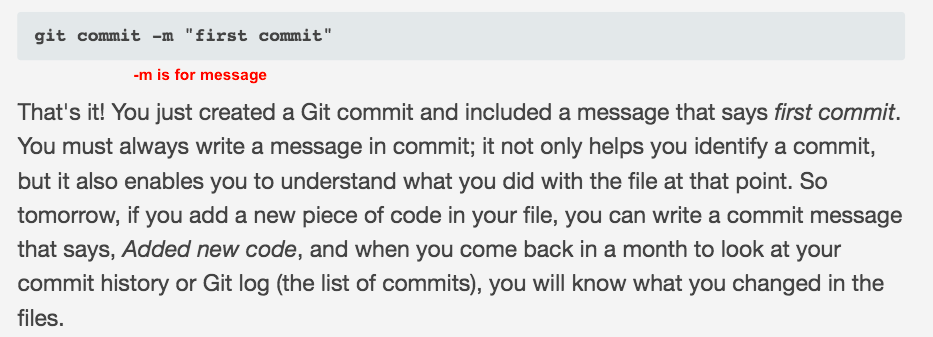




* To tell your computer that Demo is a directory **managed by the Git program** using **git init**
* to tell the Git program you care about this file and want to track any changes from this point forward, enter: git add README.md



* Step 4: Make a commit.
* So far you've created a file and told Git about it, and now it's time to create a *commit*. Commit can be thought of as a milestone. Every time you accomplish some work, you can write a Git commit to store that version of your file, so you can go back later and see what it looked like at that point in time. Whenever you make a change to your file, you create a new version of that file, different from the previous one.



* Step 5: Connect your GitHub repo with your computer.