Git and Github for beginners

Link: <https://www.youtube.com/watch?v=RGOj5yH7evk>

Gwendolyn Faraday

* Boson University / Mechanical and computer engineering
* Master of computer science

**What is Git?** // A: Free and open source version control system

**What is Version Control?** // A: The management of changes to documents, computer programs, large websites, and other collections of information.

**Terms:**

* Dictionary -> Folders
* Terminal or Command Line -> Interface for Text Commands
* CLI -> Command Line Interface
* cd -> Change Directory
* Code Editor -> Word Processor for Writing Code
* Repository -> Project, or the folder/ place where your project is kept
* Github -> A website to host your git repositories online

**Git Commands**

* Clone -> Bring a repository that is hosted somewhere like Github into a folder on your local machine
* add -> Track your files and changes in Git
* commit -> Save your files in git
* push -> upload Git commits to a remote repo, like Github
* pull -> download changes from remote repo to your local machine, the opposite of push

**Steps:**

1. Profile page / Dashboard page
2. Create a new repository (a new project incl. all files, codes, files, etc)

* Can create your files and folders for this repository locally on your machine
* Or can create them straight in the online editor on the GitHub website.
* Name it: demo-repo-GF
* Description: Git and GitHub for Beginners - Crash Course by G. Faraday from Youtube

1. Create a new file

* README.md

Md: markdown is basically an easy way to format your text in these sort of files.

***Mark down short cuts***

Placeholder text – auto generated /

1. **Git install**

choose the Git Bash option <https://www.atlassian.com/git/tutorials/install-git>

1. **Getting a code editor**

visual studio code <https://code.visualstudio.com/> free code editor by Microsoft

* Files - Open a folder – so all coding docs will be saved in the folder
* Clone git: work locally
* Go to github – code – ssh – copy

Use Git as version control

1. Using Visual Studio Editor

* Open a folder – it’s a folder in visual studio code that I can add files / add folders to / pull the repo I’ve created from the GitHub e.g. README.md – through git
* Top bar -> View -> Terminal
* In VSC terminal: type git clone SSH link

In GitHub: Dashboard /Clone / SSH / Copy

Back to VSC: paste & click enter on keyboard

* In VSC: type cd “the github repository folder name” => to change directory into the folder

Dot get directory – a special command in my terminal that I will use to show you that folder.

la: short for ls -la => List everything in the directory including hidden files and folders

Windows: ls- Force

A screenshot of a computer error

Description automatically generated

* Terminal: clear
* Edit the files e.g. README in the windows (above terminal)

Or create a new file html

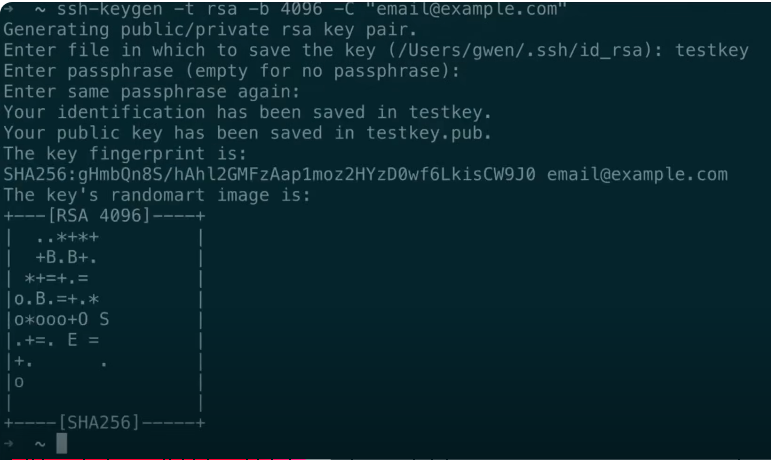
* Terminal: check status

1. SSH Key – connect your local machine to your Github

* Start generating a key locally

Ssh-keygen -t rsa -b 4096 – C “email”

The default file for your SSH key is inside your user directory in the dot SSH directory, and it will be called ID underscore RSA



* Search for the key generated.



* public key – other people can see
* The other one is private key. (only the private key can generate the public key)
* Highlight the key – it auto copies it
* Terminal command allows you to copy: Pbcopy < the directory you have your key in
* Github – settings – SSH and GPG keys

1. Make sure that your local Git command line interface knows about the key you just generated.

Open git bash

* Start the SSH Agent / open Git Bash

eval $(ssh-agent -s)

* Add the SSH key to the SSH Agent

ssh-add ~/.ssh/id\_rsa

* Verify the SSH key is added / This will list the currently loaded keys in the SSH agent.

ssh-add -l

A computer screen with text

Description automatically generated

1. Gitpush

## SSH Keys

Connect local machine to github

### Git Bash

1. Generating a key locally

*Ssh-keygen -t rsa -b 4096 -C “onechen23@gmail.com”*

Name the key *proj1Key*

1. Find the key

A black screen with white text

Description automatically generated

Public key connect to github

Use private key – to show github – only the private key can generate

A screenshot of a computer

Description automatically generated

### Github

1. Add the key to github

Profile photo – setting – SSH and GPG keys – Add SSH pub key

Git bash – SSH agent

1. Local git command line interface knows about the key

A black screen with white text

Description automatically generated

## Visual Studio Code

1. Open a folder – any folder you want to store
2. View – terminal

*git clone* copy paste SSH

*cd demo-repo* (change the repository

*ls – la*: list everything in the directory, including hidden files and folders

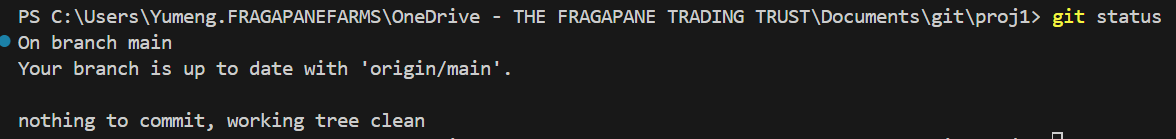
.git directory is a hidden folder // it’s blue – means it’s a folder

A close up of a black screen

Description automatically generated

*git status*: the command shows me all of the files that were updated or created or deleted but haven’t saved in a commit yet

check add a html file -> check again

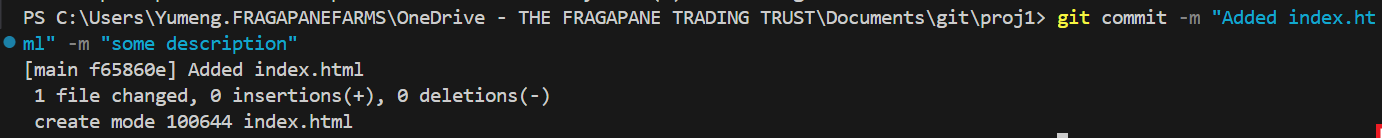


A screen shot of a computer

Description automatically generated

*git add .* :track all the files listed here

*git commit -m “Added index.html” – m “some description”*  : -m means message // “explain”



*git push origin main* // main or master is the branch we want to push to .

## proj2

1. Create a new folder called proj2 in VSC
2. In terminal: move to the folder – cd ../proj2

So if I create this proj2 locally. Git won’t know where to push it to as it’s not connected to anything.

1. Create a new repository in Github
2. Copy SSH
3. Terminal, VSC – git remote add origin paste SSH
4. Git remote -v (check)

Go to git bash

1. Open the proj2 folder

cd "C:\Users\Yumeng.FRAGAPANEFARMS\OneDrive - THE FRAGAPANE TRADING TRUST\Documents\git\proj2"

$ git init

Reinitialized existing Git repository in C:/Users/Yumeng.FRAGAPANEFARMS/OneDrive

- THE FRAGAPANE TRADING TRUST/Documents/git/proj2/.git/

# Error

Code in terminal: ls -la

For: List everything in the directory incl. files and folders

Error: Get-ChildItem : A parameter cannot be found that matches parameter name 'la'.

Reason: It seems that you're using **PowerShell**, and the command ls -la is specific to **Unix-based** shells (like Git Bash or Linux/macOS). In **PowerShell**, the equivalent command is slightly different.

Fix: PowerShell Equivalent of ls -la: Get-ChildItem -Force / ls -Force

A screen shot of a computer

Description automatically generated

proj2> git push origin main

error: src refspec main does not match any

error: failed to push some refs to 'github.com:ChloeChen003/proj2.git'

Bash

Git branch

$ git add .

fatal: Unable to create ‘C:/.git/index.lock': File exists.

Another git process seems to be running in this repository, e.g.

an editor opened by 'git commit'. Please make sure all processes

are terminated then try again. If it still fails, a git process

may have crashed in this repository earlier:

remove the file manually to continue.

# Proj3

## Git commands / VSC Terminal

git clone -> bring a repository on github e.g. proj3 into a folder on your local machine

git add -> track file and changes in git

git commit -> save your file in git

git push -> upload to a remote repo, e.g. github

git pull -> download changes from github to your local machine

**Github**

1. Github – create a new repository – **proj3**
2. Create a new file “**README.md**” – marked down file

Check if git installed: windows - **command prompt** – type: **git --version**

A screen shot of a computer program

Description automatically generated

**VSC**

1. Bring the repository proj3 from github to my local machine

VSC – open a folder – **git**

VSC – view – **terminal**

1. git clone git@github.com:ChloeChen003/proj3.git
2. move from folder git to proj3

**cd proj3**

1. Show all files in the proj3 folder including hidden ones

**ls -Force**

.git file: is a hidden file // saves all the changes recorded to the repository

1. Check all files // e.g. create a README.md file from VSC

**git status**

README.md file is untracked

1. Ask git to track all the files

**git add .** // all files

**git add README.md** // individual fileor folder

1. See status again if the README.md file been tracked by git

**git status**

README.md file is tracked

1. Commit the file locally

**git commit -m “Title of the message” -m “Description of the message”**

1. Push the live to online repository in github

**git push**

**git bash**

**Set up SSH Keys:** connect local machine to github

1. Generate the ssh key

**Ssh-keygen -t rsa -b 4096 -C “onechen23@gmail.com”**

Name the key **proj1key**

1. Find the key just generated

**ls | grep testkey**

1. Print the public key

**Cat proj1key.pub**

1. Copy the public key

**Highlight the key in the terminal / command line – auto copy**

**Pbcopy < /.ssh/proj1key**

**Github**

1. Past the SSH key to github

Profile – SSH and GPG keys – paste

**Git bash**

**Make sure your local git command line interface knows about the key too.**

1. Start the SSH Agent

eval $(ssh-agent -s)

A black background with colorful text

Description automatically generated

1. Add the SSH key to the SSH Agent

ssh-add ~/.ssh/id\_rsa



1. Verify the SSH key is added / This will list the currently loaded keys in the SSH agent.

ssh-add -l

A computer screen with white text

Description automatically generated

**VSC – Terminal**

1. Do step 11 here // push

git push origin main / git push origin master

origin: argument 1 – Origin is an option set for us here, and is basically a word that stands for the location of our Git repository.

master / main: argument 2 – master / main is the branch that we want to push to.

# Proj4

**VSC**

1. Create a folder in the folder git – called proj4
2. Termnimal – move into proj4 folder

**cd ../proj4**

1. Add a file in proj4

README.md

1. Turn the file into a git repository

**git init**



1. Clear above text

**Clear**

1. Check files

**git status**

**Untracked files: README.md**

1. Add the file and check status

**git add README.md**

**git status**

1. Commit the file into git

**git commit -m “add README.md file” -m “Some descriptions”**

1. Push the commit to git hub

Create a repository in github, name it proj4

Copy the SSH link

1. Back to VSC Terminal / clone the repository

**git remote add origin git@github.com:ChloeChen003/proj4.git**

1. Check the repository

**git remote -v**

1. Push the origin

**git push origin master**

or I can set up “up stream” = default

**git push -u origin master**

# Workflow

* Github workflow: write code -> commit changes -> make a pull request
* Local git workflow: write code -> stage changes (

A diagram of a process

Description automatically generated

A diagram of a diagram

Description automatically generated

All changes to the branch will only be able to be saved in it’s own branch.

Main branch cannot see changes made in feature branch

1. Moving from proj4 folder back to git folder

cd ..

Move to folder proj3

cd ../git/proj3

1. Check how many branches are there

git branch

1. Create a new branch / switch branches

git checkout -b feature-1-readme-instructions

1. Create a new file and push through the feature branch

Create a file index.html // <div> Hello </div>

git status

git add .

git status

git push origin feature-1-readme-instructions

git push -u origin feature-1-readme-instructions

git commit -m “add index.html file and update README.md file” -m “two updates”

1. Pull Request or PR

A request to have your code pulled from feature branch into master branch.

Once PR merged. Generally delete the feature branch and switch back to the master branch.

Then next time you want to create a new branch. Repeat the cycle.

Github

1. Compare a pull request

Feature branch is merged into main branch.

Conversation – make notes

Commitments – add comments

VSC Terminal

1. If there’s anything updated in github, how to update them to git too?

git pull origin main

1. Delete the feature branch

git branch

git branch -d feature-1-readme-instructions

1. Create a new branch

**Git checkout -b quick-test**

Quickly update index.html file

**Git status** // check the change

1. Check all the changes since last commit

git diff main

A black screen with white text

Description automatically generated

1. Undoing in git

Git reset readme.md

Git reset

1. Undo a commit // go back to a commit further – skip the last commit

Git reset HEAD~1

1. See a list of all your commit

Git log

A computer screen with colorful text

Description automatically generated

1. You can undo specific commit

Git reset 319157bc7be68991e8eea81eeb3c1a18a5e9b0fd

**Forking**

1. Fork – duplicate