Git Tutorial

Tuesday, October 1, 2019

WHat is GIT?

> A **Verison Control System** (VCS) for tracking changes in computer files

- Distributed version control
- Coordinates work between multiple developers
- Who made what changes and when
- Revert back at any time
- Local & remote repositories

Concepts of GIT

- Keeps track of code history
- Take "snapshot" of your files
- You decide when to take a snapshot by making a "commit"
- You can visit any snapshot at any time
- You can stage files before comitting

Download & Install GIT

- Download
  - MacOS
  - Windows
- Install
  - `git --version`: check git version
  - `git config --global user.name __NAME__`: set the user name so GIT knows who is making changes
  - `git config --global user.email __EMAIL@ADDRESS__`: set the user email
  - `git congfig user.name / user.email`: check if name and email are set correctly

Basics

- repository
  - A container for a project you want to track with GIT
  - As many repository as you want for different projects on computer
- basic commands:

  Note: All commands are in terminals (Mac) / cmd (win)

  `git init`: Initialize local Git repository

`git add <file>`: add file to staging area to Index

`git status`: check status of working tree

`git commit`: take everything in the staging area, put them into the local repository

`git push`: take local repository, push to remote repostory (Github)

`git pull`: pull latest verison from remote repository

`git clone`: clone repository into a new directory, download others' to local

Step by Step

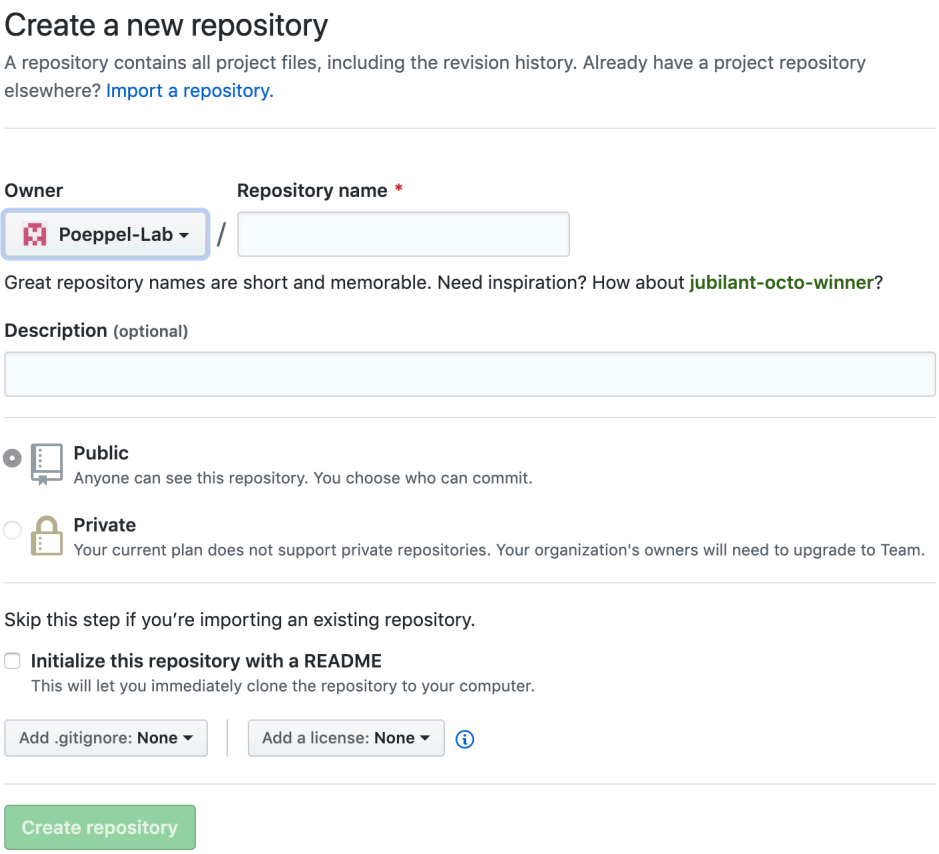
- creating a repository
  - `cd Desktop/`: change directory to Folder Desktop/
  - `mkdir mypro`: make a new folder under the current directory
  - `cd mypro/`: change directory to the folder just created
  - `git init`
- Add file to staging area and commit
  - `touch index.html`: create a html file (touch works only in mac)
  - `git status`
  - `git add index.html`
  - `git status`
  - `git rm --cached index.html`: remove file from staging area
  - `git status`
  - `git add .`: add all files to staging area
  - `git status`
  - `git commit -m '__TEXT__'`
  - `git log --oneline`: check all commits you made
- Undo changes
  - checkout commit: safe, just review
    - `git checkout <logID>`: checkout a specific commit
    - `git checkout master`: switch back to the master branch
  - revert commit: not very safe, could change file
    - `git revert <logID>`: revert a specific commit, undo anything in that commit
  - reset commit: danger, use with caution
    - `git reset <logID> --hard`: delete anything after a specific commit, no way back
- Branches
  - Create a copy of master, leave master alone
  - Good for test new feature
  - `git branch <__BRANCH__>`: create a new branch
  - `git branch -a`: check all branches
  - `git checkout <__BRANCH__>`: switch between branches
  - `git branch -D <__BRANCH__>`: delete branch
  - `git checkout -b <__BRANCH__>:` create and switch to a new branch

- Merge
    - merge branch to master
    - need to stay at master
    - `git merge <__BRANCH__>`: merge a specific branch to master
    - Note: don't edit master before merge branches

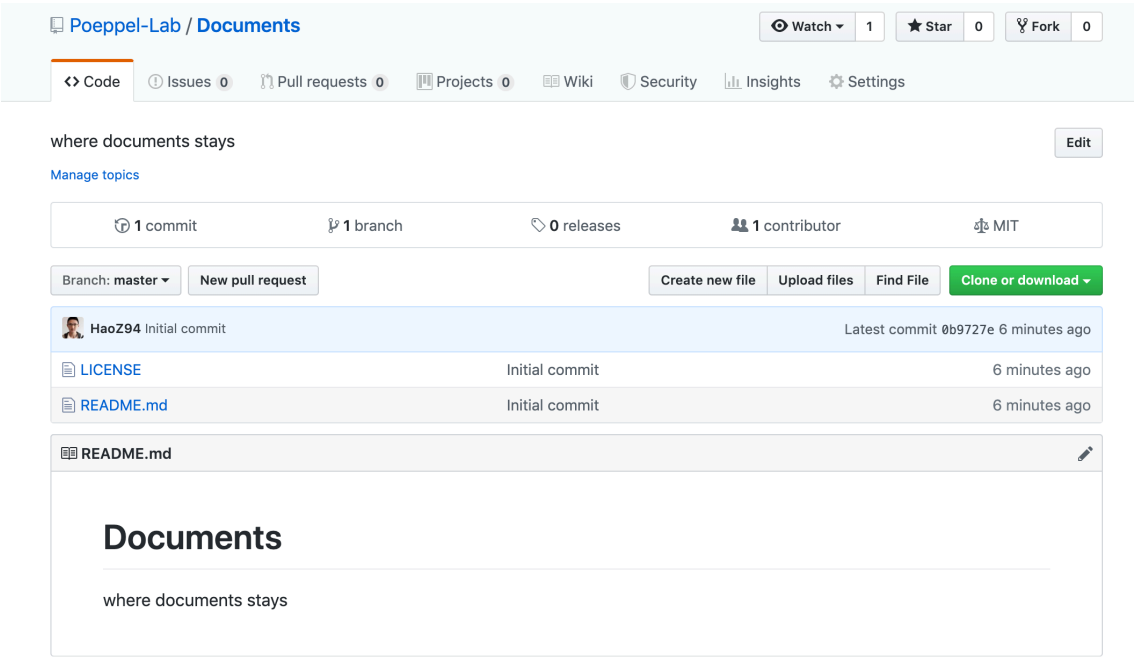
GitHub

- Online service that hosts projects
- Share code with other developers
- Developers can download the projects and work on them
- They can re-upload their edits and merge them with the main codebase
- CommandLine based
    - `git remote add origin <__LINK__>`: associate remote repository to local
    - `git push origin master`: push local to remote
    - `git pull origin master`: get latest update
    - `git push origin <__BRANCH__>`: push a new branch to the remote repository
- Web based

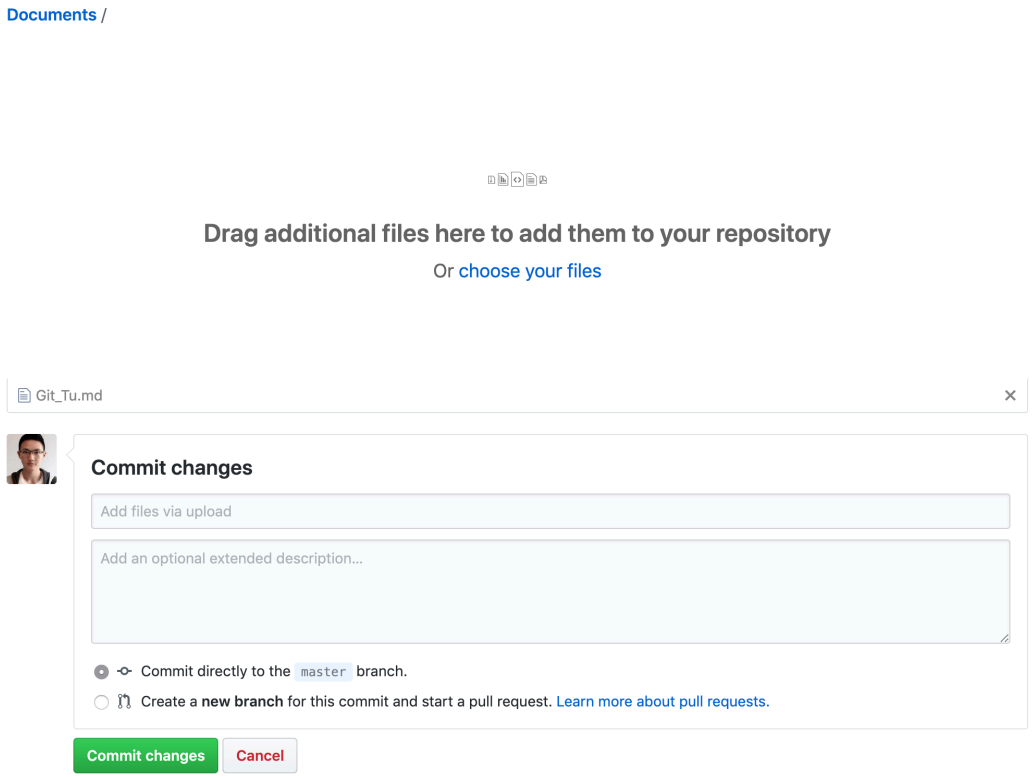
        - Create new repository



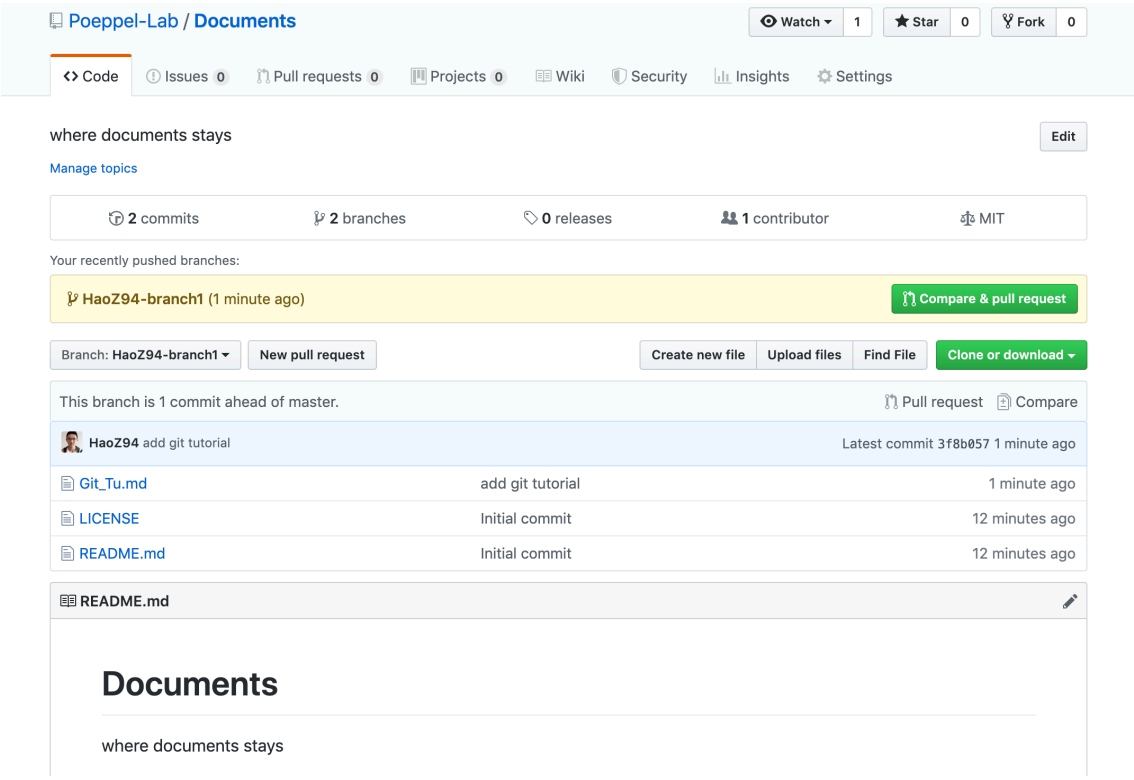
- Repository name: define a name for repository just created
- Description: Brief sentences to abstract the repository
- Initialize this repository with a README: **recommanded**, allows people to know more about the repository. It's **Markdown** based.
- .gitignore: choose affliation (coding language)
- license: Normally **MIT**



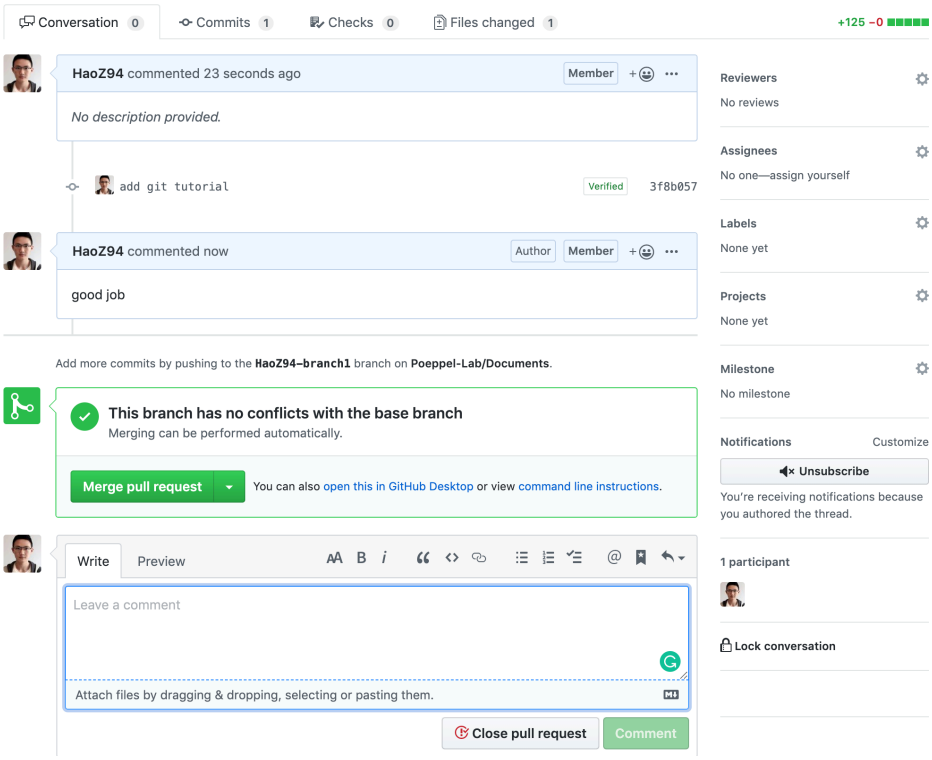
- drag any files you want to commit to the repository. Or use the *Upload files* tab.



- you can add more files in this interface
- commit to the master. Or create a new branch (**recommanded** if you not sure about the change you just made)
- *Don't forget to label your change in order to make others to understand*



- Compare & pull request: request to merge the branch



- anyone can review the changes in the branch and make comments.
- Once everyone agrees, **Merge pull request**.