GITHUB

* WHAT?

Git and GitHub exist for Version control.

Git🡪Version control software

GitHub🡪Web Service for better UX

* Repository

Vault for all the stuff/Project of that repo

* Readme

A file with markdown denoting the essence of the repo.

* Commit🡪 Kind of like save button

While Committing a new note/ name can be attached for future ref.

Each commit can be viewed in History of the repo.

Changes denoted by : ‘+’ and highlighted by Green

A new Commit Hash is assigned to each commit. Can be referenced in the URL as well.

* Branches

To create a separate division to work on the project.

Changes to branch is not referenced/ committed to master directly.

A new branch can be created using git: git branch <branch name>

git checkout <branch name>

* PUSH

When trying to send a new file to a repo.

* Fork

A way to work on the repo at your local machine and not have to get permission/access from the author.

After changes in order to contribute to main author’s repo, a pull request to the author’s repo is made. So after fork 🡪 changes 🡪 create new pull request 🡪 Add comments explaining changes. Wait for acceptance.

At the authors end, a pull request can be seen made by other user. Can be merged

* Github Issues

To report any kind of abnormalities in the repo.

Can range from technical to grammatical. Cn have ss/pics attachable. Supports markdown. Issues have specific id numbers.

Issue is open until resolved! Can be resolved by author.

Author can do changes in the repo according to the issue stated and commit that issue with id number of that issue for reference.

Issue can be closed manually or certain keywords in commit can close it automatically. Like “fixes”

Issues resolved have a particular id/hash code. Can be used to reference this issue for future communications. Acts as a link to that issue/ commit.

* GIT BASH

Emulates Unix interface to work on github.

File System: cd- change directory;

Pwd- print Working Directory

Ls- List all files inside a directory

Ls –all :Will show complete file structure with hidden files as well.

Clear- clear the shell

Open . –Will open the current directory

Open <filename>- will open that particular file.

.. –Will change directory back to previous directory

* Clone

Taking a repo from a github server and pulling/downloading it locally to your computer.

**git clone <url>**

**git status** : tells a lot of info about the current directory/ repo like what branch you’re on and what are the statuses of various files, i.e., modified, new or unchanged.

**git commit**: Once the files are changed user can commit

**git commit –a – m “Message to be put”** : -a (means all changes)

Error/ Warnings:

At this point if an identity error comes up, its cuz you might have changed any username/ password etc.

So **git - -config global user.name “<Your username>”** and similary password/email to be entered to tell github about yout identity.

**git config - -list** : will display all the configurations done to that local system.

**git log -2** : Will show last two commits from the log.

GIT VIM: Kind of a text editor that one can get stuck in. To get out of it:

:q 🡨Will get you out.

C:\Program Files (x86)\Microsoft Office\MEDIA\CAGCAT10\j0205582.wmf

C:\Program Files (x86)\Microsoft Office\MEDIA\CAGCAT10\j0212957.wmf

PULL

PUSH

GITHUB USER

* PUSH

When wanting to add the changes to your repo on github from your local machine.

Remote: A duplicate copy of repo on a server, eg. repo on github.

**git remote** 🡪 Lists all the remote branches

**git remote –v** 🡪 (-v stands for verbose). Denotes the url of the remote

Origin: That’s the original repo cloned.

To push the modified repo**: git push origin master [**git push <root directory> <branch to be pushed>**]** (Pushing into master branch)

Error: While pushing if the branch/ remote name is different, i.e., can be main or anything else use:

Git push origin main 🡨[The master to be replaced with the remote name].

* INIT

Any folder locally created is not a git repository. This command adds an empty git repo to that folder.

* ADD

For the intermediator before commit (a staging area of sort). We add the desired files from the file system to local repo and then commit.

On local system:

After Making changes to a file 🡪 Hit Save 🡪 Will save to the local file system 🡪 Add 🡪Commit (Will commit to the repo).

[Add 🡪 Before commit its necessary if there is a new file/ changed file.]

For adding a particular file:

**git add <file name>**

For everything: **git add .**

To remove added file **: git rm - - cached**

**Untracked Files:** Files that are modified but not been added for the commit.

After adding the file: **git commit –m “<Message for commit>”**

If while pushing next **: git push origin master An Error comes:** Origin doesn’t appear to be git repo. Git remote 🡪 Doesn’t display anything

This is because when locally created there is no remote associated with that local repo. Hence solution is to create a new repo on GitHub and DO NOT create a readme as you’d like to add files first to a blank repo.

**Git remote add origin <Url>**

[origin is a variable name. Can be changed].

* PULL

For working on a new branch, merging it into the master branch a pull request is made.

After pushing , if some changes have been made to github, it can be reflected in the local directory using pull command.

**git pull origin master**

* GITHUB PAGES

If master branch is gh-pages 🡪 Can be hosted as a github page and accessed through a URL. To do that:

1. Make a branch called gh-pages for that repo.
2. In settings 🡪 branch 🡪 Change default branch to master branch.
3. In branches, delete the master branch (Just to be sure not to work on it and lose your work as default branch is now gh-pages).
4. In the settings of the repo, the URL is generated for accessing the web version of it.

* MERGE CONFLICT

When multiple versions of a same file/ repo exist, and changes are made to them, after pushing them to github, when the author finally starts to accept the pull requests, github shows merge conflict error. And asks to resolve the conflicts.

As the author, we can detect which files are conflicting 🡪 make improvements to them to resolve conflict 🡪 Merge non-conflicted files 🡪 Accept pull request.

* REMOTES

A remote is a virtual location of the repo.

A new remote can be added or pulled etc.

Once you want to pull a new remote, you can create a branch for safekeeping instead of pulling into the master.

git branch graphics

git checkout graphics

git pull keeghan

Error: sometimes pulling a different repo or having conflicted/ multiple branches the bash can go to VIM Editor.

:i 🡪to insert your comments into the VIM editor.

Esc 🡪 to move out of insert

:wq 🡪 Write the inserted changes from the VIM.

After this it should pull and author can accept merge request on github or directly commit from bash.

git commit –m “Merging Branch Keeghan”

[ Now change branch to master]

git checkout master

git merge graphics

git push origin master 🡨 Finally merge all changes to master