Git and Git Hub

*\*The lines that are underlined and bold are actually codes*

Both are two different things. Git is open source control tool and Git Hub is cloud server allow us to version control of our code. Git is a UNIX based command line.

Recommendation

1. Use Git Bash at the beginning.
2. Create a folder and right click to open git bash.
3. **git status** (Will tell if the file you are in, is a git repository or not.)

*Note: Every version has different style of installation. To check your git is installed properly type on command line (CMD)* **git –version** *\*You have to add in path though\**

1. Make a git repo in Local comp. (**git init**) \*i.e. make the current folder into git folder\* init -> initialize.

*Note: This will make .git hidden folder in that same folder and it will make the simple folder into git repo for version control.*

Flow chart of git’s source control.

Working directory Staging Area Git directory

Stage files

Commit

Check-out the project

1. Now consider there are same codes in local git repo to add them into staging area (Still unsaved at the state) use **git add .**

Imp note: Before committing its necessary to give a commit message for which we use **–m “\*\*\*”**

To commit (**git commit –m “\*Add your msg. here\*”**)

Q. How to check who have committed what, if so many people are working in the same project?

Ans: **git log**

Q. How to add user name and user email to git?

Ans: **git config --global user.name “\*Enter your name\*”**

**git config --global user.email “\*Enter your email\*”**

**How to use Git**

1. Create a folder (open git bash in that folder) type git init
2. Create any two random file in which contains code or any text in that repo.
3. Open git Bash again in the same folder type **git add .**
4. **git commit –m “Version 1”**
5. **git status** (this will tell you two committed files)
6. Do change in the txt file *any change*
7. **git status** (This will give you the result as two uncommitted files.)
8. **git add .**
9. **git commit –m “Version 2”**

To check whatever the changes done by which user use command **git log**

**Imp.** To change the user name for *a particular repo/folder* use the code in git bash in that folder

**git config user.name “\*y/n\*”**

**git config user.email “\*y/email\*”**

Some of the things you should know before having a git hub account.

1. In the new policy you can have three contributors in a private repo.
2. Once you create your repo in git hub add the **ssh** of repo to your Local PC.

Steps are:

Open git bash in that repo folder of your local PC type:

**git remote add origin \*y/ssh\***

*V/Imp. Note:*

*You cannot push your codes to your repo in git hub unless your PC has ssh key access to git hub. To generate the ssh key follow the steps:-*

Generating ssh key

1. To get the local PC (ssh key) open git bash in the repo folder (Check first email in the repo –

**git config user.email**) *copy the email*

1. **ssh-keygen –t rsa –b 4096 –c “\*y/email\*”**

or **ssh-keygen -t ed25519 -C "*\*y/email\**"**

note: This creates a new ssh key, using the provided email

**>***Generating public/private ed25519 key pair.*

1. press **Enter** this will save the key in the default location. c/user/you/.ssh/id\_ed25519.
2. Enter a strong password or press directly enter if you don’t want it to be secure, here password won’t be visible and have to be entered twice.

Adding the ssh key to ssh-agent

1. The ssh key should be running on background. And in git bash type **eval “$(ssh-agent -s)”**
2. **ssh-add~/ *.ssh/id\_ed25519*** file location ssh name

So the name of the ssh-key may vary according to what you give.

Copping the local ssh-key and adding to Git hub

1. **clip < ~/.ssh/id\_ed25519.pub**

*for legacy system* **clip < ~/.ssh/id\_rsa.pub**

1. Go to your git hub account then *profile>settings>SSH and GPG keys>New SSH Key*
2. *Paste the key* and flow the few steps.

*Now You can push the codes to your desired git hub repo in your account.*

To push the code use **git push –u origin *master*** note: ***after origin it’s the branch name we don’t need to write anything if there is only the main branch which will be always master.***

Making any change in git Hub repo

Changing anything in git hub repo won’t bring any change to your local PC unless you update yourself neither it will be shown in git log for that we have to do pull request. i.e. **git pull origin *master***  \**Now it will show the changes*\*

Note:- *Changing a file in github I meant it can be done by a collaborator but also by self just by going into the website and manually changing or editing it.*

More command in Git:

QNA. Suppose if by chance a file data gets changed then, it can be tracked in github, but how to do that in local Pc?

* **git diff \*file\_name.*extension*\*** note: git bash should be opened in that folder and it can track only till what it was last committed.
* **git diff** (If you write only this, it will still work if the change is done to only one file or consisting only one file)
* **git diff \*file\_name.*extension*\*** won’t show the difference for a staged file, to check staged file difference use the command **git diff --staged \*file\_name.*extension***\*

Q. How to get the previous version of the code in your local PC using git commands only? Ans. Suppose you have a file that edited and unstaged to get back the previous code that was last committed use the command **git checkout \*file\_name.*extension\****

Imp Note : *Similarly if the code was committed and then we need the previous version then first unstaged them by reset them and then checkout you get the previous version.* The codes are mentioned bellow:-

**git reset \*file\_name.*extension***\*

**git checkout \*file\_name.*extension***\*

Note: It cannot be undone once a file is committed but the above mentioned code can bring you the last committed file. Or even earlier version.

Types of staging basics in Git

Three types :-

1. git add –A stages all
2. git add . stages new and modified without deleted
3. git add –u stages modified and deleted, without new

*Modified files*

*New files*  ***add .*** ***add -u*** *Deleted files*

***add –A***

Q. How to copy from github into your local PC? Ans. Open github repo you want to get the code and copy the code ssh id of the repo then *open the folder* you want to paste it *start git bash* and then follow the command **git clone \*ssh\_id\_of-the-repo\***

Note*: This will create a repository inside that folder to avoid making extra folder i.e. you want to extract the code directly on that folder just add* ***.*** *at the end by giving space after the ssh id of the repo.*

Some short important topic

Ignoring files: Like consider in C programing we keep testing the files which creates lots of .exe clutter in our repo. to avoid committing every time this clutter and the programs that are not necessarily need to keep track off we can use git ignore. To start with in git bash type:

**touch .gitignore**

This will create a .gitignore file. Open in *any text editor* and type **\*.exe** to ignore *.exe* clutter files. For more info check github/gitignore.

Branching

This is used to make an experiment branch so that we don’t screw up the master *(main)* branch. We can create unlimited branches. Steps are:

1. First; check how many branch are there: **git branch**
2. Second; creating a new branch: **git branch \*y/branch-name\***

note: *In git bash when you check* **git branch** , *One with the* ***green in color*** *that branch is* *currently active*.

1. Third; Switching branch: **git checkout \*y/branch-name\***

Merge of Branching

Consider after experimenting in you new branch you find its ok, then you want to merge it into the main branch. Steps are:

1. 1st ; **git checkout master** *🡪 this will login into master branch*
2. 2nd ; git merge \*y/branch-name\*

Note: *When you checkout into master your new edits will be gone or not visible in that folder unless you merge that branch.*

Deleting a Branch

Q. How delete the branch? Ans. As we know, if we have to encounter some huge no. of codes and we want to delete the other branches then :-

**git branch –d \*y/branch-name\***

Note: *Above code is for deleting the branch from your local computer only not that are on github. One can do manually by going into their github account and delete the branch.*

To *delete the branch from github remotely* from *your local PC* is

**git push origin ­­--delete \*y/branch-name\***

Most Imp:

*In above code if a line is enclosed with double cote “” that means it’s necessary to use double cotes and any line inside the asterisk \* \* are the naming lines.*

*When you are pushing the code into Github make sure that which branch you are pushing i.e.* **git push origin \**y/branch-name*\*** *which it not necessary to be master, if you are pushing different branch write its name.*