Git / GitHub

configuring username

→ $ git config --global user.name "Adarsh Singh"

→ $ git config --global user.email "2001ada...@gmail.com"

$git --version ⟶ gives version.

$git config --list ⟶ gives the info about current user on git.
(space)
↘ if you want to see 1,
$ git config user.email ↵

clear ⟶ to clear screen.

$git help ⟶ menu.
↳ learning more about topics in help.
$ git help commit ⟶ opens things in browser.

---

creating Repository

$ pwd ⟶ where git is currently now. [I mean which folder]
/c/Users/Adarsh

- first navigate to the folder required

$ cd ~
I ↳ home.
change directory

- initialising the directory to be git project.
$ git init

* Trick →
$ ls -la
- not only the folders but shows also the hidden folders.

navigating in given directory.

⇒ backward. ; /c/users/adarsh
$ cd ..
it will give → /c/Users.

⇒ forward
$ ls → will show what all folders you can hop up to.
$ cd Adarsh
it will give → /c/Users/Adarsh

) Adding files and Commit log =>

adding files.

$ git add . → add all the changes that we made to our project. (.) → means folder or directory. → adding all.

but inorder to keep track of the changes, we have to commit them. 2-Step Process → first add & then commit.

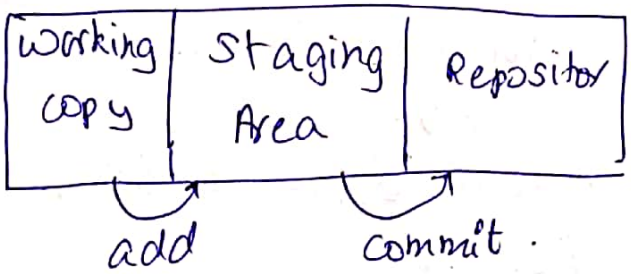
→ save this point in time (snapshot)

$ git commit -m "This is our first commit"

- commit: so anytime you want to comeback to.
- -m: space
- "This is our first commit": message.

$ git add second.txt

↳ adding single file in staging area.



$ git log

↳ shows all commits made.

$ git log --author="Adarsh"

↳ shows all commits made by this author.

$ git status.

↳ current status. Tells what all files need commits & what are in staging area.

) Editing files

$ git add filename.txt

$ git commit -m "modified filename.txt"

in this case $git status at halfway through tells you the files needing modification or modified.

- Viewing the changes.

  After changing any file and before adding them to repository.

  → $ git diff

  ↳ in red ⇒ + in Green shows the difference the thing in repo and on local computer.

  → $ git diff --staged

  ↳ It shows difference between staging area files and repository.

- Deleting a file.

  $ git rm thisd.txt

  ↳ removes.

  doing a $ git status will show that a file has been deleted.

  so now, you have to commit.

  $ git commit -m "We deleted thisd file".

- Moving and Renaming files ⇒

  if you rename a file, $ git status will tell that there is a file to be added [new name file] and there is a file which is deleted [old name file].

  so, $ git add newnamefile.txt

  $ git rm oldnamefile.txt

  $ git commit -m "Renamed file from oldnamefile to newname file".

(or)

$ git mv oldnamefile.txt newfilename.txt

↳ move, in git moving a file is somewhat similar to renaming it.

but for moving a file into a folder

$ git mv file.txt FileName/file.txt

It moves the file.txt to FileName. you can also change the name while moving by changing file.txt under filename and

$ git commit -m "moved to other files".

Trick

If you are editing or modifying things you can go from working directory to the repository without going to staging area by,

$ git commit -am "Cleaned up"

but be careful while using it. Only to be used for simple editing. And not for moving or deleting.

- replacing working files from the files in repository

$ git checkout -- filename.txt

- Unstaging files.

$ git reset HEAD filename.txt.

it will unstage the files, meaning remove the file from staging area and bring back to working copy. or stage.

- Getting Older versions from Repository.

$ git checkout 01e7dba -- filename.txt

first few bits of commit number

$ git commit -am "Called a commit file"

GIT-HUB

→ initialise and other stuffs on git bash.

→ setting up nickname,

$ git remote add githubRepo https:....

any nickname for use

so when you check,

$ git ~~repo~~ remote

githubRepo

then adding files to github.

→ $ git push -u githubRepo master

Nickname

then it will ask for username & password.

o) Want to commit or push again to github.

→ pull it first,

git pull 'remote_name' 'branch_name'

eg. git pull githubRepo master.

pull = fetch + merge

can just do

$ git pull

if you have pushed just before

Creating a Branch,

- $ git checkout -b 'branch_name'

  - -b : flag for creating a branch.
  - 'branch_name' : give branch a specific name.
  - switches to a new branch named 'branch_name'.

now changes can be seen as

$ git status (in /Testing Repository (Branch_name))

- modified : README.md

then, git add and commit.

now, push it to github using its branch name,

$ git push origin 'branch_name'

origin : Nickname