# * GitHub *

Page
Date

* What is version control ?

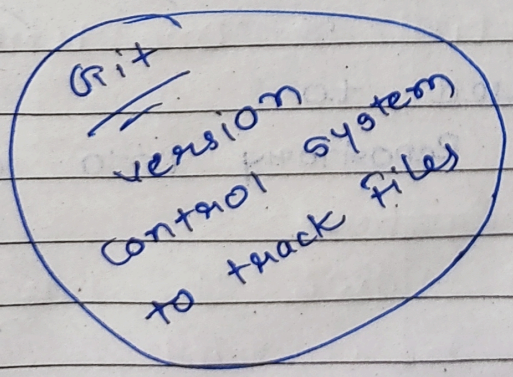
→ version control is system that helps manage changes to files and documents over time.

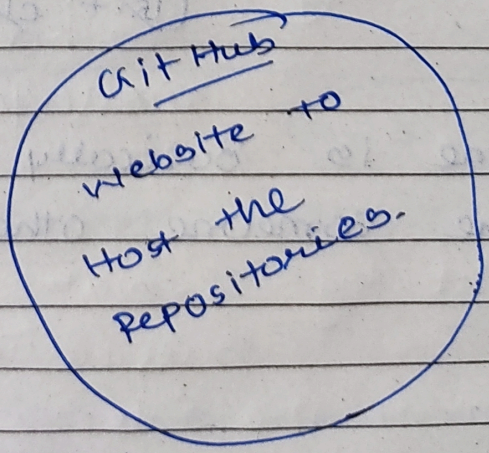
→ The primary purpose of version control is to track modification made to file or a set of files and enable collaboration between multiple users.

## Git vs GitHub.

- Git is version control tool (software) to track the file

→ GitHub is a platform to Host the Repository.



* Creating a Git Repo.

Git init :- To Initialize the Git Repo

→ Go to the directory of the file

→ git init

→ git status (to see the file.)

ls - list
cd = change directory

.git folder would be hidden by default to see that folder use command

⊞ + shift + .
Window Period
command

Window to see .git folder use dir/a

Git clone

→ Git clo

Git clone is Basically used to clone the someone other's Repository into our System.

Command.

git clone <url> Foldername

copy the url of the Repository which you want to clone.

Provided by you

* git diff

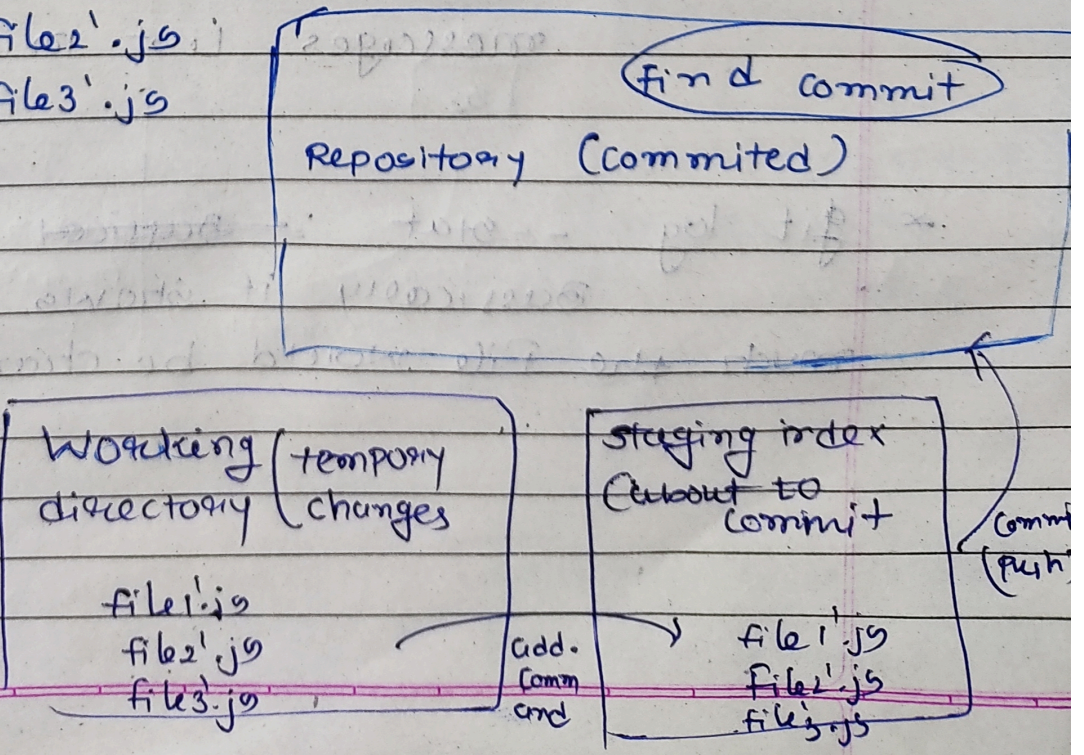
It show us to see the which things we have changed in Repository after cloning.

* Life cycle of change

→ suppose you are working on one Project dot_dummy_Repo

dot_dummy_Repo

| Original Fil | After changes |
|---|---|
| file1.js | file1'.js |
| file2.js | file2'.js |
| file3.js | file3'.js |



* Reviewing A Repo History

* git log :-
So Basically git log Provides a a commits by id. (by latest)

Press q to go out from commit list

* git log -3 → latest 3 } commits.
git log -2 → latest 2

* git log -P : Bassicaly it provides a actual changes in the file

* git log --Oneline :-
to see the id and commits messages list in one line.

* git log --stat :- ~~Bassical~~
Bassicaly it shows the how much the file would be changed.

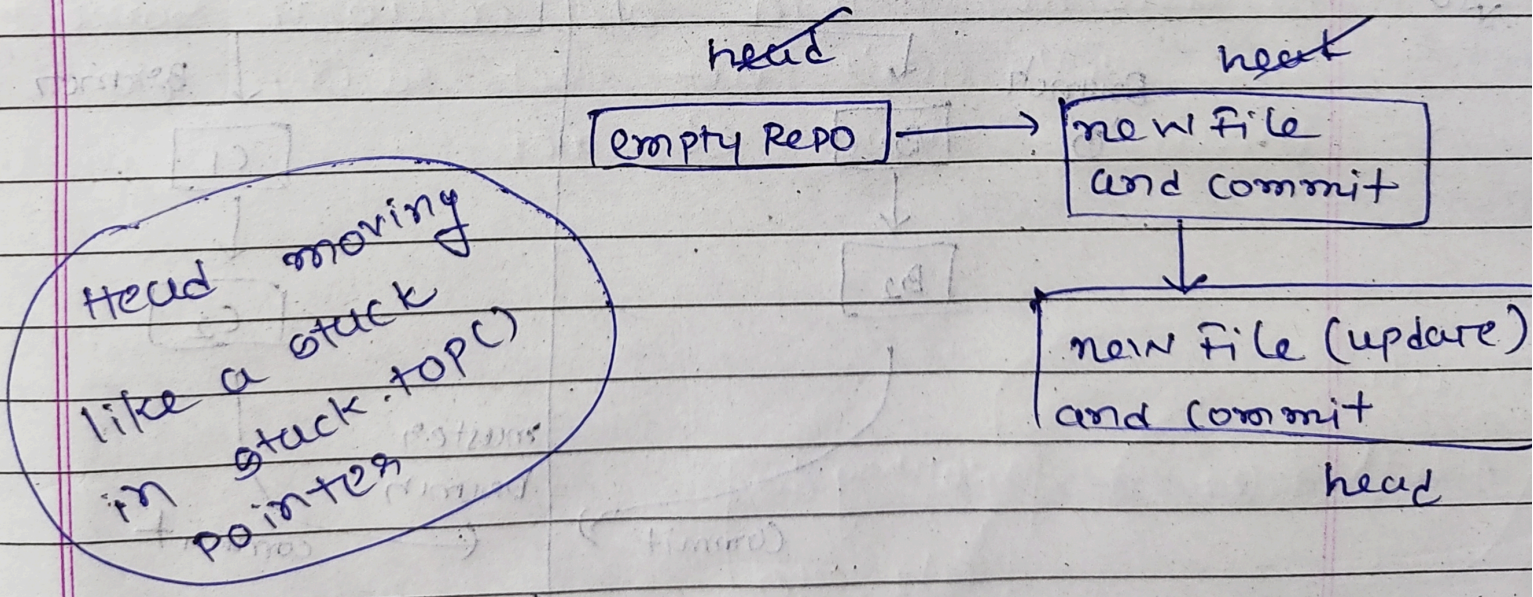
* git show <commit Id>
Bassically It show the what is the changes made in that particular commit.

* Let's make a commit.

* git add .
Bassically It adds files to track.

* git commit -m "~~init mai~~
New file

* ~~git~~ Head provides




* git restore filename
↑
bassically backtrack from that ~~file name~~ changes

* How to add a .gitignore file.

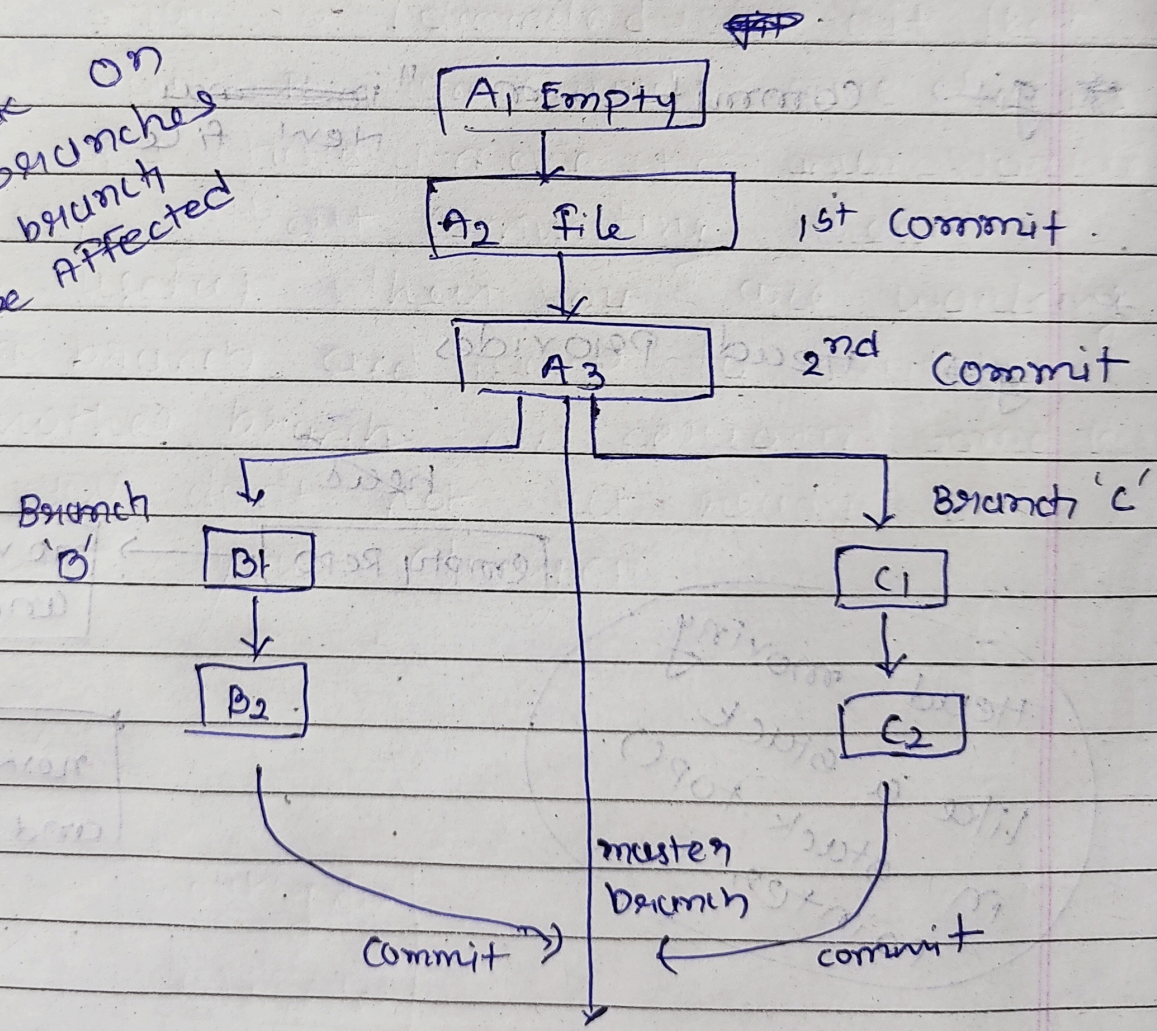
→ create file .gitignore

→ add the file type like (*.txt)

ignore txt fi

* Branching, Tagging & Merging.

→ People can work on the different branches but the main branch would not be affected



Steps:-

① → git branch (check the branches How many branches are there)

→ Create a new branch Run command

② git branch <branchname>

③ If you want to move to the new created branch Run Command.

③ git checkout <Created branchname>

↓

Now commit like we do in Normal commit because we have changed our working directory.

If you want to create new branch and move both Run comma

git checkout -b <branchname>

→ If you want merge this branch then Run command.

- move to master branch (git checkout master)
- git merge (branchname)

Branchname which you want to merge with master branch)

* After commiting want to delete branch 9

git branch -d branchname

* git tag :-

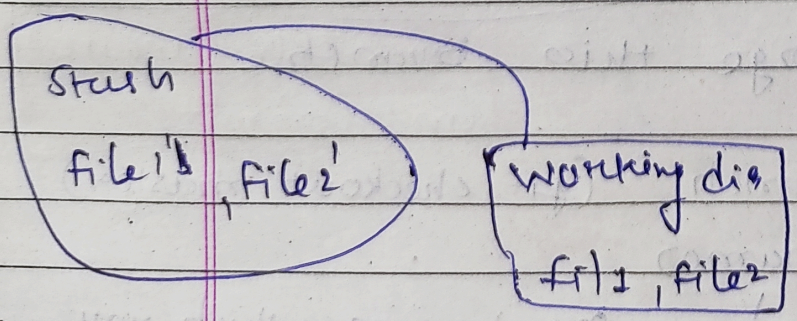
git tag basically we can give a tag to the specific commit

git tag -a <Commit Id> -m "message"

To delete tag :- git tag -d beta V1.0

* git stash

→ Git stash is command in Git that allow you to temporarily save changes that you have made to your working directory without commiting them. It is useful when you are working on a branch and want to switch to another branch or perform some tasks but you don't not yet ready to commit your changes.



* Undo commit.

* Git commit --amend → amend the most Recent Commit.

→ Git Revert → Revert given Commit

→ Git Reset → delete Commit (dangerous command)

Revert
Git <commit Id>

↓

Enter esc then : then wq

New file changes would be done

→ Git Reset meaning Commit would be delete and Head change.

git reset --soft <commit Id where you want to move>

# How to push a commit

Page
Date

- make git Repo on GitHub web page.
→ ~~local~~ go to your folder
→ follow ~~as~~ steps as github repo.
→ before push → set username

```
git config --global user.name "username"
git config --global user.email "useremail"
git push -u origin master
```