Create and Modify Repos

Friday, June 1, 2018 5:01 PM

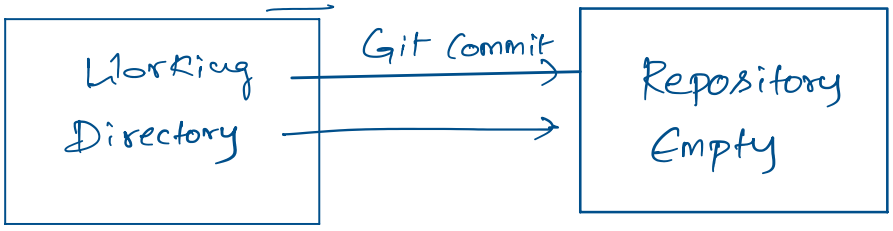
git init [to initialize git in a specific directory (folder to a git repository)]

Directory & Subdirectories inside → Repository

Diffirent repositories → Different directories

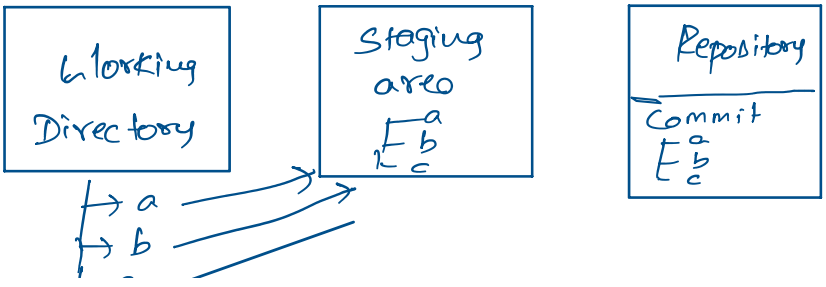
Some times the git only track some of the files in that directory rather than all of them.

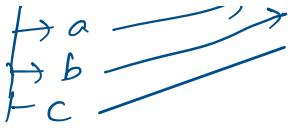
git Status – Shows which files are changed from the last commit



Can't RUN git commit to create a snapshot of the Current state of the files

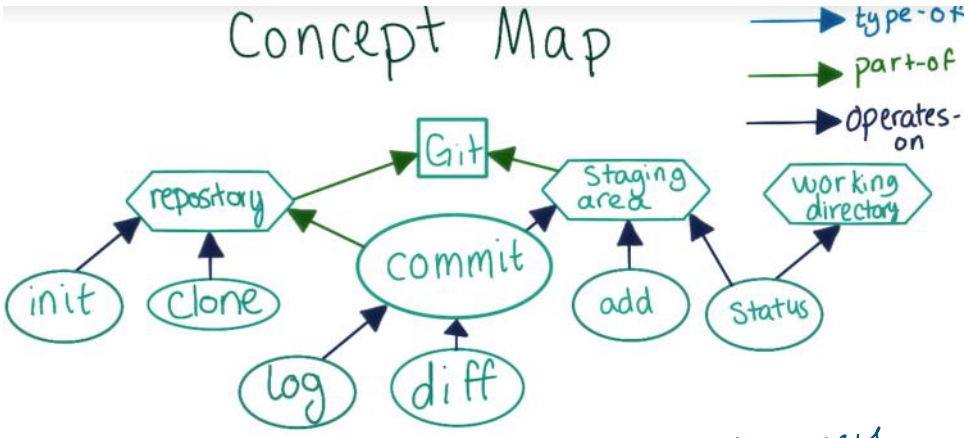
Keep Commit Small – one Commit per logical change





git add – The files will be in staging area

git reset :- Removes the file from the staging area.
Still be in the working directory



git Commit – to commit changes

```
git commit -m "commit message"
```

↓ commit full Command

Commit msg Structure

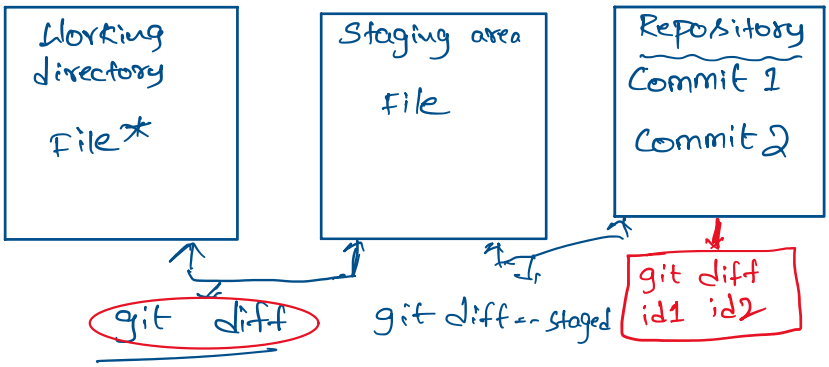
Type: Subject
body
Footer

Subject < 50 characters
What a commit does.
Body: What & why
each line length < 72
Footer: optional reference issue tracker IDs

feat: a new feature
fix: a bug fix
docs: changes to documentation
style: formatting, missing semi colons, etc; no code change
refactor: refactoring production code
test: adding tests, refactoring test; no production code change
chore: updating build tasks, package manager configs, etc; no production code change

git add → git status → git commit

one file at a time.



git diff → To track the changes that haven't been added to the staging area yet.

git diff --staged - To see the changes between the staging area & most recent commit

git reset --hard - Discards any changes in the working directory (or) the Staging area

once this done there is no going back.

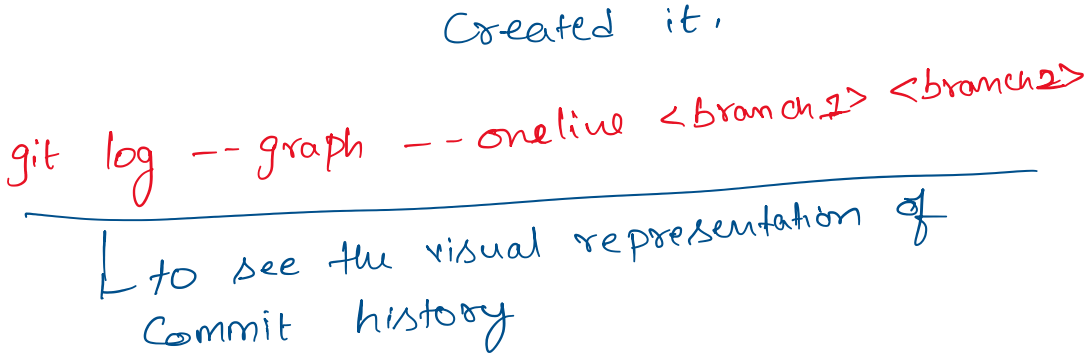
git checkout master - when the head is in detached state

git branch <branch name> - To create a new Branch

git branch → Shows the current branch.

Remote branch → Not created by you rather someone else created it.

<branch2>



Git config --global user.name "Your name"
Git config --global user.email "your email address"
Git reflog - to see all the recent checkouts

Each commit in GIT knows about its parent.
Stores a reference to the commit that was checked out when it was made.
It just stores the commit ID of what was then the tip of the branch.
Only the branches store anything about the position of the branch.

If a branch is deleted and leaves some commits unreachable from existing branches, those commits will continue to be accessible by commit id, until Git's garbage collection runs.

This will happen automatically from time to time, unless you actively turn it off.
You can also run this process manually with git gc.