GIT VERSION CONTROL TOOL

TYPES:

1. Version Control System (VCS)
2. Distrubuted or Decentralized Version Control System (DVCS)

Advantages of GIT:

1. Free and Open source
2. Fast and Small
3. Implicit Backup
4. Security
5. No need of powerful hardware
6. Easier Branching

DVCS TERMINOLOGIES:

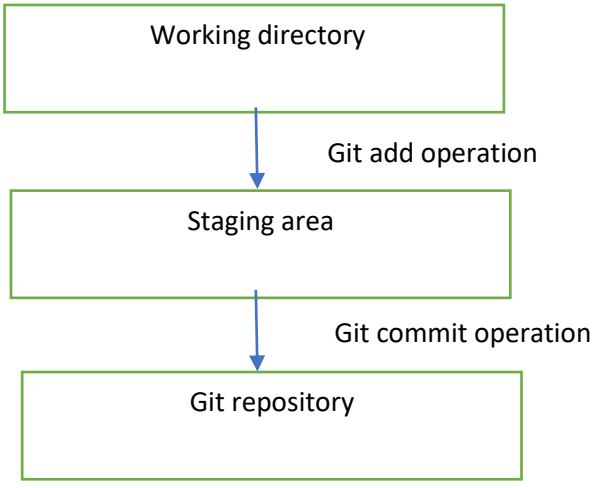
1. Load Repository
2. Working Directory and staging area or index

Basic work flow of git:

step 1: you modify a file from the working directory

step 2: you add this file to the staging area

step 3: you perform commit operation. this moves the files from staging area to local repository. after that perform push operation which stores the changes permanently to the git repository



GIT TERMINOLOGY:

Blobs, Trees, Commits, Tags, Clone, Pull, Push, Head, URL

Git download to windows:

URL: http://git-scm.com/download

AFTER INSTALLATION:

Go to windows and select ‘Git bash’

git --version [displays version of it]

Create a new folder and open it, Right click on folder and select Git Bash

git init [is for initialising the Git]

ls -la [list the files and must display .git/ (Base repository)]

SETTING GIT ACCOUNT:

git config - - global user.name “XXXXX”

git config - - global user,mail “ XXXX@gmail.com”

git config - - list

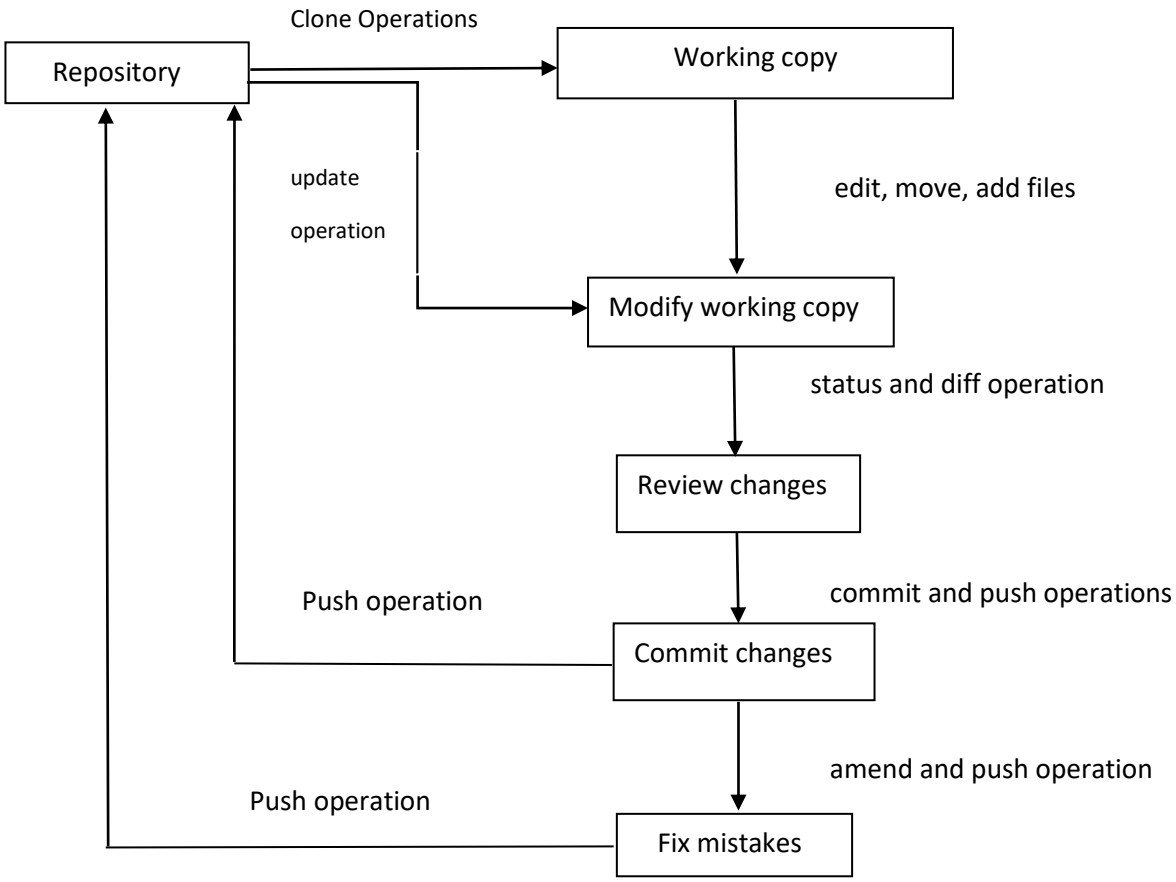
UNSETTING GIT ACCOUNT:

git config - - global - - unset user.name “XXXX”

git config - - global - - unset user.mail “XXXX@gmail.com”

git config - - list

GIT LIFE CYCLE:



CREATE A GITHUB ACCOUNT:

Go to https://github.com and signup

copy repository link in github

go to git folder and type

git clone repo URL

ls -l

cd my repo

GIT COMMENTS:

git status → displays the location

git log → displays the no. of commits

git show [commit ID] → list the commit

In select branch

vi filename and save it and

git add filename

git status

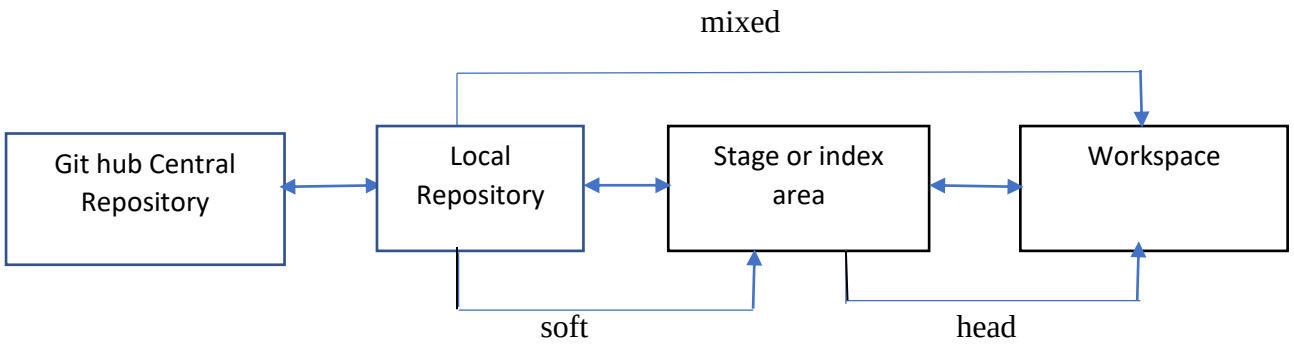
git commit -m "new commit" file.txt

git status

GIT WORKFLOW:

workspace→staging→local repository→add and commit

Local repository→workspace→mixed



local Repo → staging or index area →workspace → soft and head

ex: vi file2.txt [in master branch]

git status

git add file 2.txt

git add . [it moves all files to local to index]

git status

[if change the file in staging area]

git rest HEAD file name

ls

git status

git push

git pull [repo URL]

git push [repo URL]

git reset - - mixed commit ID

git reset [local repo→workspace]

git reset - - soft commit ID

git status

GIT BRANCHING STRATEGY:

git - - version

ex: 2.18.0

2 → major release

18→ minor release

0 → hot fix

git branch [displays branch list]

ex: *master, *current branch

master→hotfix→release branch→developer→ feature branch

CREATE A BRANCH:

git branch [branch name]

CHANGE BRANCH:

git checkout [branch name]

vi file.txt →enter content

git status

git add .

git status

git commit -m “reason/message”

CENTRAL REPOSITORY:

git push origin.branch name

MERGE FILES FROM MASTER TO NEW BRANCH

git status → ls

git checkout master

git merge new branch

git push origin – d new branch [deletes the branch]

git branch -d new branch [this is branch delete]

git checkout -b branch name

git branch

git branch -m old branch new branch

git branch

GIT STASH:

temporary memory is called stash

ls→git status→vi file9.txt

git add. → git status

git stash list

git stash save “file9.txt & file10.txt”

git stash list

git status

git stash show [stash ID]

git stash show-p [stash ID]

vi file11.txt

vi file12.txt

git stash save “file11.txt & file12.txt”

git add.

git status

git stash list

git stash show [status ID]

git stash show-p [stash ID]

git stash show-p [stash ID]

git stash apply [stash ID]

git status

git stash list

git stash drop [status ID]

git stash pop

git stash list

GIT TAGS:

Tag is a meaningful name or simply to identify the file

git commit -am “files”

git status

git log

git tag

git tag tagname (12.0.1)

git tag

Push the file from local repo to central repo with tags:

git push origin tag 1.0.0

git show tag 1.0.0

Delete tags:

git push origin -d tag 1.0.0

git tag

git tag -d tag 1.0.0