**Git**  is a [version control system](https://en.wikipedia.org/wiki/Version_control_system) (VCS) that is used for [software development](https://en.wikipedia.org/wiki/Software_development)

 Allows many [software developers](https://en.wikipedia.org/wiki/Software_developer) to work on a given project without requiring them to share a common network.

every Git [directory](https://en.wikipedia.org/wiki/Directory_(computing)) on every [computer](https://en.wikipedia.org/wiki/Node_(networking)) is a full-fledged [repository](https://en.wikipedia.org/wiki/Repository_(version_control)) with complete history and full version tracking abilities, independent of network access or a central server.

* How to download and install git.

Prog that Keep track of changes in file.

Ex – ok this is my essay and its done . file save as final.txt

Edit text above and again save as final1.txt

Git do a lot better and do not need to rename.

* Best prog to manage source code.
* We can use it with word doc , psd etc.
* Download dit git-scm.com/downloads

1. **Git - - version**
2. Set username and pass to know who committed   
   git config - - global user.name “Kaif Akhtar”  
   (global means this user)  
   git config - - global user.email [k@gmail.com](mailto:k@gmail.com)  
   git config user.email  
   git config - - list (returns all settings of user)
3. Clear
4. Git help
5. Git help commit (opens help page of commit command)

**Make a folder tuna at desktop**

* Pwd (returns where/path in comp git is looking at)
* Cd ~ (~ means home) means change directory to home
* Cd .. take u back one directory
* Ls returns all folder in current directory
* Cd Bucky move to bucky folder
* Cd desktop move to desktop
* Cd tuna move to tuna
* Pwd returns c/users/bucky/desktop/tuna

**how to turn tuna folder in git**

* Git init (git repo initialized/started) .. now tuna folder is git project..git is now going to track the changes.
* Ls –la (also shows hidden files in folder)
* Git add . (add all changes made in proj/directory)
* Commit is like taking a snapshot at that point
* Git commit –m “This is our first commit”

**How to view commit history**

* Git log (commit history)
* Git log - - author = “Kaif” show commits of Kaif
* Git log - - author = “wendy” no show as wendy does not exist
* Git status – tells u if there are any changes
* Untracked files msg, if we don’t make commit
* Adding and commiting files is different
* Git add second.txt (specific file added on staging area)
* Working copy > staging area > repository
* Staging area is just these are the files ready to commit
* Git status – now check the status of files in folder
* Git commit –m “adding the second file to repo” – push files from staging to repo
* Git log

**Edit files**

* Edit notepad file. Now the file in repo is diff from the file in here.
* Git status – file is modified now
* Git add first.txt – staged
* Git commit –m “modified file” – 1 file will be inserted in git and 1 deleted
* Git status – now file on comp is same as on repo

View changes u made

* Git status
* Git diff – shows diff bw file on comp and on repo
* Git add .
* Git comit –m “”
* Git diff – now no diff – if diff more than 2 files it will show all diff one by one

**Compare staging with repo**

* Edit file
* Git diff – shows diff
* Git add .
* Git diff – no diff , means it shows diff bw staging area and working file.
* Git diff - -staged – compares files in staging against repo

**How to delete file**

* Delete one file
* Git rm first.txt – removes the file from repo and working copy
* Git status – need to be committed still
* Git commit – m “”

**Move and Rename files**

* Rename one file
* Make file 2.txt and commit to repo , now rename 3.txt after that remove 2.txt using cmd rm 2.txt and add 3.txt to staging ,now commit the changes.
* Git status
* Git add home.txt
* Git rm first.txt
* Git status – now it will show file is renamed
* Git commit “here file renamed”
* Easy way to rename
* Git mv second.txt renamed.txt – moving is same as renaming in git
* git mv renamed.txt paddy/renamed1.txt – moved to another folder (paddy ) and also renamed file
* git commit

**Commit directly to repository**

* commit from working directory to repo instead of adding to staging area
* git commit –am “cleaned up html comments” - it will commit everything in ur working copy

**checkout command**

* when u messed the working local copy and want to get the files from repo.
* Modify one working copy file
* Git status (one modified file)
* Git checkout index.html (take something from repo and make working copy)
* Git status (working copy is same as repo)

**Unstage files**

* Bring back files from staging area.
* Modify file
* Git add . (files staged)
* Git status
* Git reset HEAD index.html (removes from staging area)
* Git status

**Getting old version of project from repo**

* Get old version from repo.
* First commit>second commit>third commit > (now you want to go back to second so u not need to del third you need to goto second point of time and commit again ) > second commit
* Git log
* Make some changes in file and do hard commit git commit - - am “made a icon for forum”
* Git status
* Git log (shows first commit and second commit)
* Mess up with file and commit again
* Git commit - - am “I am dumb”
* Git log
* Git checkout version - - index.html (goto version and get copy of index.html)
* Git status
* Git commit - - am “undoing kevin’s mistakes”

**GitHub**

* We know that git allows u to keep track of changes in proj.
* Well in github make public project for world to see .. it’s a website. Any 1 can access and suggest the changes. Any1 can download file and upload it to github. Every can collaborate for working on project.
* U need a github account
* Make new public repo.
* Give proj or repo name and give desp (if reqired).
* Initialize a repo with readme (uncheck it as it is not required) becoz we initialize from cmd using git init.
* Initialize a project in ur comp and commit to repo using git bash in local comp.
* Now we have to take this project and publish to github so that everyone can access it.

**Pushing to GitHub**

* Setup remote i.e connection bw local comp and public directory(Github)
* Git remote add nickname (.git url) - now conn is made from gitbash
* Git push –u nickname master – it will ask for username and pass.
* Uploading starts. Check github.com for files.

Cloning a repository

git clone https://github.com/*YOUR-USERNAME*/*YOUR-REPOSITORY*

-git cmd?

- access right to diff users.

- checkout process? How to pull frsh code from repo (first time)

These commands are very useful when interacting with [a remote repository](https://help.github.com/articles/about-remote-repositories). clone and fetchdownload remote code from a repository's remote URL to your local computer, merge is used to merge different people's work together with yours, and pull is a combination of fetch and merge.

git pull remotename branchname --allow-unrelated-histories