**1. Install Git :**

Windows : *git bash : https://gitforwindows.org/*

Ubuntu : *sudo apt-get install git-core*

RedHat : *sudo yum install git-core*

**2. Config Set :**

Set a Git username:

*git config --global user.name "Ethans Institute"*

*git config --global user.email "deepak.kumar@ethans.com"*

*git config --list`*

**3. Need HELP?**

Open the help page in browser or in console

*git help <verb>*

*git <verb> --help*

**4. Initialize a repository from existing code.**

*mkdir LearnGit*

*cd LearnGit*

*git init* (will create .git directory which makes the LearnGit directory a tracked Repo)

*touch .project*

*touch test.pyc*

*touch calc.py*

*git status* (will show the untracked file before git commit)

*touch .gitignore* (to block some files in repo to not get commited to remote repo)

*vi .gitignore* (and add the files to be ignored)

.project

\*.pyc

*git status* (will not show the files included in .gitignore)

**5. Basic WorkFlow**

Working Dir Staging Area .git Dir(Repo)

| | |

|<---------------------Checkout the Project-----------------------|

| | |

|---------------Stage Fixes---------->| |

| | |

| |------------Commit------->|

Add File to staging area

*git add .gitignore*

*git add calc.py*

Remove File to staging Area

*git reset calc.py* (bring back calc.py to untracked stage)

*git reset* (will bring all staged file to untracked)

**6. Our First Commit**

*git add -A* (add All file to staging area)

*git commit -m "Initial Commit"*

*git status*

*git log*

**7. Cloning a Remote Repo**

*mkdir RemoteRepo*

*cd RemoteRepo*

*git clone git@github.com:deepak2717/MyScripts.git .*

**8. Viewing Information about remote repository**

*git remote -v*

---------Note-----------

git pull does a git fetch followed by a git merge

You can do a git fetch at any time to update your remote-tracking branches under refs/remotes/<remote>/.

This operation never changes any of your own local branches under refs/heads, and is safe to do without changing your working copy

A git pull is what you would do to bring a local branch up-to-date with its remote version, while also updating your other remote-tracking branches

-------------------------

*git branch -a*

**9. create a .gitignore and calc.py in the cloned repo.**

*touch .gitignore*

*touch calc.py*

write a code in calc.py

def add(x,y):

pass

def subtract(x,y):

pass

def multiply(x,y):

pass

def divide(x,y):

pass

def square(x,y):

pass

*git add -A*

make change in the calc.py

def multiply(x,y):

return x\*y

*git diff* (show the changes made in the files)

*git status* (show the modified state)

git commit -m "Added calc.py and worked on Mulitiply Function"

git status

git pull origin master

*git add -A* (all the chnages in file)

**10. Pushing the changes**

we should always pull first then push when there is parallel development is going on

*git pull origin master*

From github.com:deepak2717/MyScripts

\* branch master -> FETCH\_HEAD

Already up to date.

*git remote add origin git@github.com:deepak2717/MyScripts.git* (its get added when clone is done, if remote is not added we can use this command)

*git push -u origin master*

Counting objects: 4, done.

Delta compression using up to 4 threads.

Compressing objects: 100% (3/3), done.

Writing objects: 100% (4/4), 407 bytes | 203.00 KiB/s, done.

Total 4 (delta 1), reused 0 (delta 0)

remote: Resolving deltas: 100% (1/1), completed with 1 local object.

To github.com:deepak2717/MyScripts.git

7257f74..6c795bb master -> master

Branch 'master' set up to track remote branch 'master' from 'origin'.

**11. Common Work Flow :**

a) Create a branch from desired feature

*git branch calc-divide* (creates the branch from master)

*git branch* (check the branch)

*git checkout calc-divide* ( move from master to the created branch)

b) Make changes to the calc.py in divide function

def divide(x,y):

return x/y

*git status* ( show the modified state in branch)

*git add -A*

*git commit -m “Divide Function”* ( only changed in the local branch, it has no effect in the local master and remote repository)

c) After commit push branch to remote

*git push -u origin calc-divide*

*git branch -a*

d) Merge a Branch

*git checkout master* ( switch to master as branch need to be merged with master)

*git pull origin master* (a good practice, to pull any new changes from remote to local, in our case we had not made any changes)

*git branch –merged* ( It shows with what all branched you have merged till now)

*git merge calc-divide*

*git push origin master*

**12. Deleting a Branch**

*git branch –merged* (to check the branch is merged with master, so we can delete the branch now)

*git branch -d calc-divide* (delete the branch calc-divide locally, not from the remote)

*git branch -a* (we can see the calc-divide branch is still in remote)

*git push origin –delete calc-divide* (delete the branch from remote)