Git Course

\* Clone Existing Git Repository into local machine

1) create a local directory.

2)cmd -> **git clone git repo\_url** Clones a repository into a newly created directory, creates remote-tracking branches for each branch in the cloned repository (visible using git branch --remotes), and creates and checks out an initial branch that is forked from the cloned repository’s currently active branch.

3) After the clone, a plain git fetch without arguments will update all the remote-tracking branches, and a git pull without arguments will in addition merge the remote master branch into the current master branch

cmd- **git fetch & git pull**

<https://www.git-tower.com/learn/git/faq/difference-between-git-fetch-git-pull>

\*Create new git hub Repository

go to git hub and create a new repository there you can get the remote URL

1) create a local directory.

2) **git init** ->This command creates an empty Git repository - basically a .git directory with subdirectories for objects , refs/heads ,refs/tags, and template files. An initial HEAD file that references the HEAD of the master branch is also created.

3**) git add .** -> after copying all files into new local directory or you can add empty git repository into your existing folder using **git init**  Then execute **git add .** it will add all files

If you want to add a particular file then use **git add file\_name**

4) **git commit –m “log message”** -> creates a new commit containing the current contents and given log message describes the changes.

5) **git remote add origin remote\_url ->** by adding remote you can connect your local repository to github remote repository

6)**git push origin master** ->it will push all local files to git hub repository

origin-> your remote & master is your current working branch.

\*Create Git Branch

1) **git branch branch\_name** ->it will create a new local branch

2) **git checkout branch\_name** ->HEAD moves to given branch name

(HEAD is nothing but a pointer which pointes to git branches, HEAD pointing branch is our current working branch)

**or**

1)**git checkout –b branch\_name** ->it will create a new local branch and HEAD to the newly created branch.

(when we push the branch our local branch git will automatically create our local branch to remote branch)

\*Show list of branch and remote

1) **git branch –a** -> shows all branches

2) **git branch –r** ->shows all remote branches

3) **git branch - - merged** ->shows all merged branches

4) **git remote** ->shows all remote

\*Branch merging

1)first checkout to master branch

2)then use **git merge master branch\_name** -> it will merge the branch\_name branch into the master branch

\*Switch between branches

1) **git checkout branch\_name** ->switch to branch branch\_name.

\***Resolve CONFLICT**

git shows conflict when more than one branch is trying to merge the same change on master branch. When conflict occurs git also shows the conflict location like file name and the conflicted area is marked with starting with

<<<<<<<<<HEAD and ends with >>>>>>>>your branch\_name.

Below HEAD the original file content the ‘=======’ marks below that is the changes we are committing, to remove conflict we have to remove all HEAD and equal marks and keep the code which we want to commit.

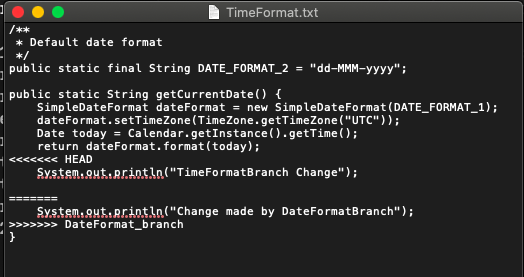


image: Before resolving commit

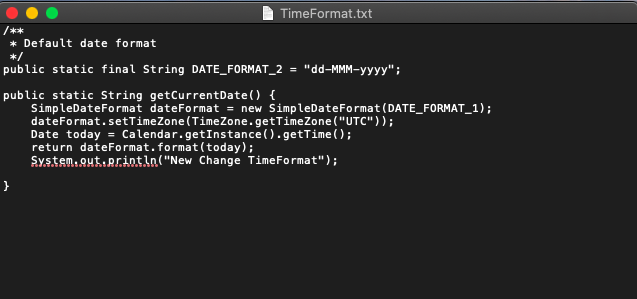


image: After conflict resolve

then 1)git add

2)git commit

3)git push

\***Discard the file changes**

1) git checkout -- or git checkout path/file\_name

\***Git Stash**

The command saves your local modifications away and reverts the working directory to match the HEAD commit