Install GIT & make sure it is added into PATH.

Section 0 -Use GIT as local VCS. Steps to follow:

1. Create a directory ‘project\_dir’ & cd to ‘project\_dir’.

Mkdir project\_dir ; cd project\_dir

1. Initialize git version database. (git init)

git init

1. Create a new file index.html.

Touch index.html

1. Check the git status. You should find index.html as untracked file.

Git status

1. Stage the index.html file.

Git add index.html

1. Commit index.html

Git commit -m “some changes made”

1. Make few changes in index.html & create a new file info.txt file.

Touch info.txt

1. Check git status. You should find index.html & info.txt as untracked files.

Git status

1. Configure GIT to ignore all txt files.

Touch .gitignore

1. Again check the git status. You should find only index.html as untracked file.

Git status

1. State & commit index.html

git add index.html

git commit -m “changes made”

1. Log all your comments so far.

Git log

1. Make some changes in index.html.
2. Revert the change made in the previous step using git command.

git reset –soft

git checkout index.html

1. Again change index.html.
2. Stage index.html

git add index.html

1. Revert back the last stage.

git stash

git checkout index.html

1. Rename ‘add’ command to ‘my-add’.

git config –global alias.myadd add

1. Using my\_add command Stage index.html again & commit the changes.

git myadd index.html

git commit -m “changes were made”

1. Revert the last commit.

Get reset --soft

*GIT Branching*

Objective: Commit HTML, CSS & JavaScript assignments into GIT.

SECTION-1 (HTML assignments) - Steps to follow:

1. First take a backup of your assignments & projects. This is required because due to incorrect GIT operation you may lose your files.
2. Create an empty directory ‘Assignments’ & cd to ‘Assignments’.

Mkdir Assignments

Cd Assignments

1. Create a file README.txt inside ‘Assignments’ & write few lines about the contents of ‘Assignments’ folder.

Touch README.txt

1. Commit README.txt file.

git commit -m “created README.txt”

1. Now create a new branch ‘html-assignments’.
2. Switch to ‘html-assignments’ branch.

git checkout -b html-assignments

1. Copy all HTML assignments inside ‘Assignments’ folder.

cp a1.html Assignments cp a2.html Assignemnts

1. Commit HTML assignments into ‘html-assignments’ branch.

git commit -m “copy html files”

1. Make minor changes into few files belonging to ‘html-assignments’ branch.
2. Commit those changed files.

git commit -m “changes made in html-assignments branch”

1. Switch to master branch.

git checkout master

1. Make minor changes into README.txt file & commit those changes into master.
2. Again switch to ‘html-assignments’ branch.

git checkout html-assignments

1. Make minor changes into few files belonging to ‘html-assignments’ branch.
2. Commit those changes.

git commit -m “Again made changes in html-assignments”

1. Switch to master.

git checkout master

1. Merge ‘html-assignments’ branch into master. Confirm all html assignments are shown in master.

git merge html-assignments

1. Finally delete the ‘html-assignments’ branch.

git branch -d html-assignments

Deleted branch html-assignments (was b16ce5d)

SECTION-2 - (CSS assignments) Steps to follow:

1. Create a new branch ‘css-assignments’. Switch to ‘css-assignments’ branch.

git checkout -b css-assignments

1. Copy all CSS assignments inside ‘Assignments’ folder.

cp m1.css Assignments

cp m2.css Assignments

1. Commit CSS assignments into ‘css-assignments’ branch.

git commit -m ”first commit”

1. Make minor changes into README.txt file on line 1 belonging to ‘css-assignments’ branch.
2. Commit those changed files.

git commit -m “README.txt changed”

1. Switch to master branch.

git checkout master

1. Make minor changes into README.txt file on line 3 & commit those changes into master.

git commit -m “made changes”

1. Again switch to ‘css-assignments’ branch.

git checkout css-Assignments

1. Make minor changes into few files belonging to ‘css-assignments’ branch.
2. Commit those changes.

git commit -m “changes in m1.css file”

1. Switch to master.

git checkout master

1. Merge ‘css-assignments’ branch into master. Confirm all css assignments are shown in master.

git merge css-assignments

1. Finally delete the ‘css-assignments’ branch.

git branch -d css-assignments

deleted branch css-assignments (was b16ce5d)

SECTION-3 - (JavaScript assignments) Steps to follow:

1. Create a new branch ‘js-assignments’. Switch to ‘js-assignments’ branch.

git checkout -b css-assignments

1. Copy all JavaScript assignments inside ‘Assignments’ folder.

cp sample.js Assignments

1. Commit JavaScript assignments into ‘js-assignments’ branch.

git commit -m “js assignments”

1. Make minor changes into README.txt file on line 1 belonging to ‘js-assignments’ branch.
2. Commit those changed files.

git commit -m “README.txt assignments”

1. Switch to master branch.

git checkout master

1. Make minor changes into README.txt file on line 1 & commit those changes into master.
2. Again switch to ‘js-assignments’ branch.

git checkout js-assignments

1. Make minor changes into few files belonging to ‘js-assignments’ branch.
2. Commit those changes.
3. git commit -m “changes made in sample.js file assignments”
4. Switch to master.

git checkout master

1. Merge ‘js-assignments’ branch into master. Confirm all JavaScript assignments are shown in master.

git merge js-assignments

1. Finally delete the ‘js-assignments’ branch.

git branch -d js-assignments

*GIT Remoting*

Objective: Pushing source code into GITHUB & collaborate team members.

SECTION-3 (Pushing assignments to remote repository) - Steps to follow:

1. Create a github account if you do not have already.
2. Login on into github account.
3. Create new public repository ‘freshersbatch-oct16’.
4. Commit & push any sample file to this repository under ‘Assignments’ directory.

**Commands :**

git remote add origin <https://github.com/abhialagi77/freshers-28dec.git>

git add .

git commit -m “Add existing file”

git push origin master

SECTION-4 (Pushing source code to remote repository using Eclipse GIT plugin) - Steps to follow:

1. One developer from project team will create eclipse projects ‘SampleProj’ & add sample source code files. Then commit all files through eclipse GIT plugin.
2. Collaborate other team members with your github account so that they can also modify the committed files.
3. Other developers from same team will checkout all files from remote repository. This might get conflicts since certain files fail to merge. In such case, merge it manually.
4. Commit & push the ‘SampleProj’ project.