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

$ git status

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

index.html

nothing added to commit but untracked files present (use "git add" to track)

1. Stage the index.html file.

git add index.html

git status

1. Commit index.html

git commit -m "adding index.html file"

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

git add index.html

git commit -m “changes in index.html file”

git commit

touch info.txt

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

git status

On branch master

Untracked files:

(use "git add <file>..." to include in what will be committed)

info.txt

nothing added to commit but untracked files present (use "git add" to track)

1. Configure GIT to ignore all txt files.

touch .gitignore

in .gitignore add .\*txt

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

git status

1. State & commit index.html

git add .

git commit -m “changes in index.html”

git commit

1. Log all your comments so far.

git log

1. Make some changes in index.html.

Vi index.html

git add .

git commit -m “changes in index.html”

git commit

1. Revert the change made in the previous step using git command.

git diff head

1. Again change index.html.

vi index.html

1. Stage index.html

Git add .

1. Revert back the last stage.

Git log –one line

Git log f23c28abddc08ce2eece406e065a344241afb607

1. Rename ‘add’ command to ‘my-add

git mv add my-add

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

git add .

git commit -m “changes by adding”

git commit

1. Revert the last commit.

git revert f23c28abddc08ce2eece406e065a344241afb607

*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’.

cd ..

mkdir Assignments

cd Assignments

git init

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 add README.txt

git commit -m "adding readme file"

git commit

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

git branch html-assignments

1. Switch to ‘html-assignments’ branch.

git checkout html-assignments

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

Create html files

cp \*.html Assignments

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

git add .

git commit -m “changes in html files”

git commit

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

git commit -m "i changes"

git commit

1. Switch to master branch.

git checkout master

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

git add .

git commit -m “changes in readme files”

git commit

1. 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 "i changes"

git commit

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

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

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

git branch css-assignments

1. Switch to ‘css-assignments’ branch.

git checkout css-assignments

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

Create css files

cp \*.css Assignments

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

git add .

git commit -m “changes in css files”

git commit

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

git add .

git commit -m “changes in readme file”

git commit

1. Switch to master branch.

git checkout master

1. Make minor changes into README.txt file on line 3 & commit those changes into master.
2. Again switch to ‘css-assignments’ branch.

Git checkout css-assignment

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

Git add ,

git commit -m “changes in readme”

git commit

1. Switch to master.

git checkout master

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

git merger css-assignments

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

git branch -d css-assignments

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

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

git branch js-assignments

1. Switch to ‘js-assignments’ branch.

git checkout js-assignments

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

Create js files

cp \*.js Assignments

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

git add .

git commit -m “changes in js files”

git commit

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

Git add ,

git commit -m “changes in js files”

git commit

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.

Git add .

git commit -m “changes in js files”

git commit

1. Switch to master.

git checkout master

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

git merger 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.

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.

Downloaded the eclipse and created a project Sampleproj and added source code files named as p1.html,p2.js and committed the code and pushed to git hub

1. Collaborate other team members with your github account so that they can also modify the committed files.

I able to make some changes like adding, delating, etc

1. 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.
2. Commit & push the ‘SampleProj’ project.

I changes the commit other people can see the changes which I give permission in teams.