**Section2:**

**Lec3: Install JDK on Windows**

Installed jdk on windows and set the environment variable

**Lec4: Install eclipse on windows**

Installed oxygen version of eclipse on windows

**Lec5: Setting Up Eclipse for git usage**

Go to eclipse->window->preferences-> search git

Go to configuration

Add key(user.name) and value(syedchay)

Add another key(user.email) and value([syedchay@gmail.com](mailto:syedchay@gmail.com))

To have git related options available in toolbar, we need to customize the windows perspective

Go to windows->perspective->customize perspective

See git option disabled. Then go to action set availability-> select git click on ok

Again go to windows->perspective->customize perspective

Now see git option enabled

Now go to windows->show view->other ->git->select git repositories and git staging

It will start showing in tool bars of eclipse in the console panel one

**Section3:**

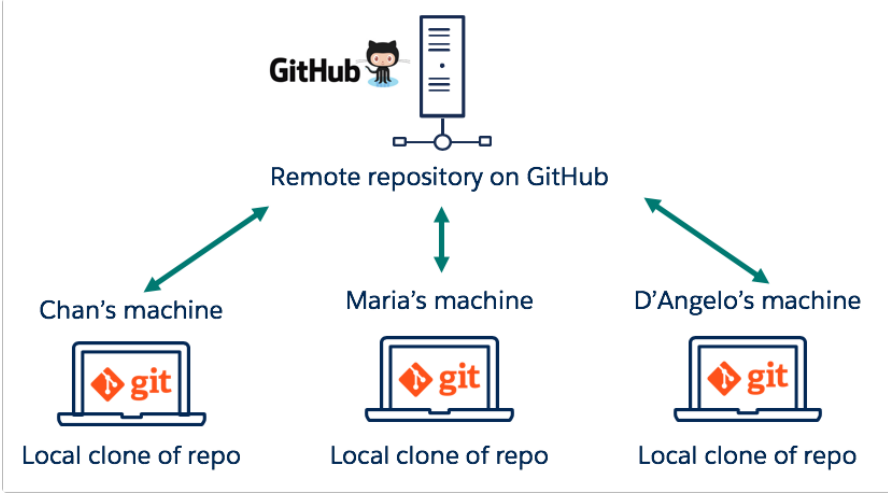
**Lec7: Git Repository- Local and Remote**

Git is a distributed version control system that can track code changes and helps in code collaboration

Git is a tool/software that is installed in a pc

GitHub is a code hosting platform for collaboration and version control

It’s a git based repository hosting platform. We can host our projects that uses Git on GitHub.



Git Repository-Local and Remote

Git local repository is the storage place where local changes are made.

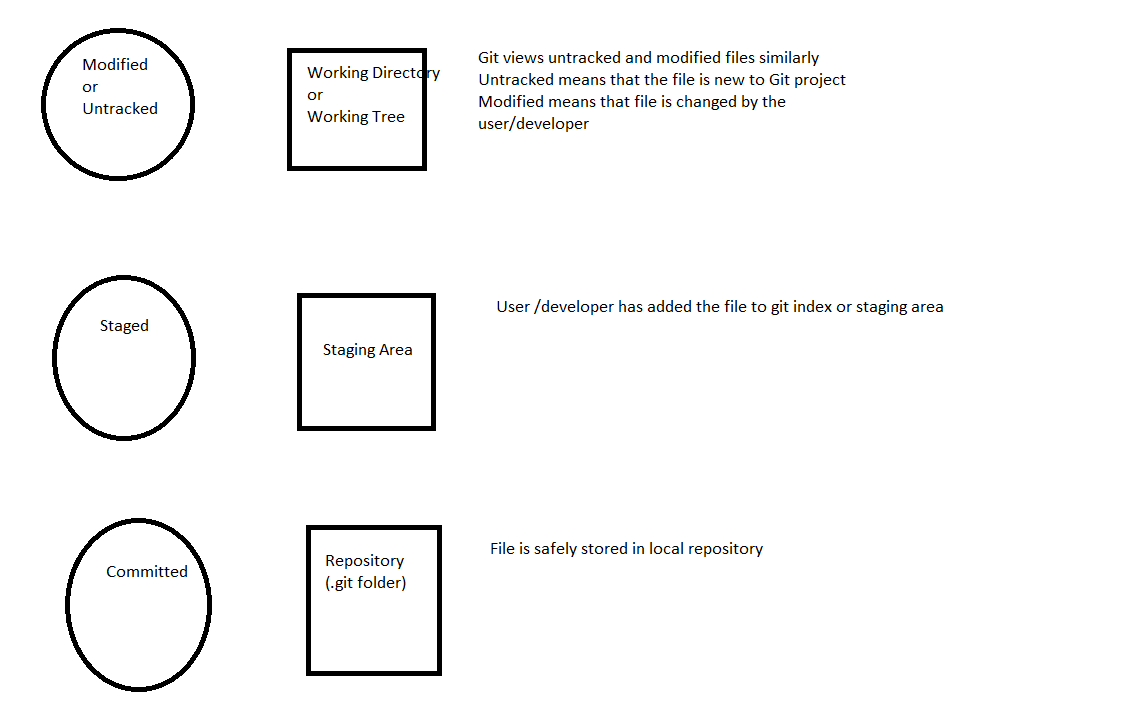
Typically this local repository resides inside our desktop/laptop

Remote Repository is common repository/storage that all team members use to exchange their changes

Remote Repository is generally a code hosting service like github or on an internal server

**Lec8: How Git Works – The Basic Flow and Concepts**

Three States of Git File(s) and three areas in Git



We are going to clone git project residing in git hub

Go to github-> click on code->copy url

Go to eclipse->window->show view->other->type git->select git repositories->click on open

In the tabs, select clone git repository ->do next next-> click on finish

Now inside git repositories tab, we see local repository at top and working tree at bottom

Imported working tree from git

Convert the project to maven project if pom file is present in the project imported

Modify one file among others

Now open git staging view by navigating to:

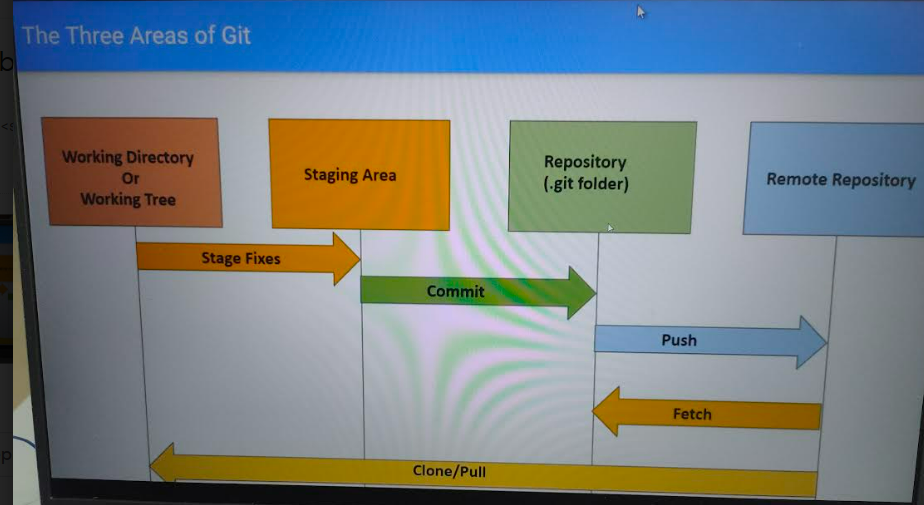
Windows->Show View->Other-> Git staging

The file modified will be showed to unstaged changes . We move it to staged changes

Git commit is a snapshot of your git repository at one point in time. It is a snapshot of the exact state of every tracked file in your staging directory.

Every time commit is added to a git repository, a hash string which identifies this commit is generated. This hash is computed with the SHA-1 algorithm and when expressed in hexadecimal notation, such hashes are 40 digit strings.

A commit contains metadata such as the author, the commit message, and the time the commit happened. For each commit, git maintains data for tracking what changed and some metadata such as author, committer, commit message, time of commit , while files were affected etc.



In Staging area tab of eclipse(where console is also in the tab) there is commit message, commit and push button, commit button

If I just click on commit button it will just stay in local repository and if I do commit and push it will go to remote repository.

When I click only commit then we will see in staging area tab there it will show one up arrow with 1 written beside that showing there is one un-pushed commit.

Now to push into remote repository , right click on project->team-> push to upstream->click ok

Now go to github-> your project-> open the java file where change was pushed

**Understanding Fetch**

We did some changes in one java file in remote repository in git hub itself.

Now go to eclipse-> right click on project-> team-> fetch from upstream. Click on ok

Open the changed file in eclipse, its not showing the changes

Pull= fetch+merge

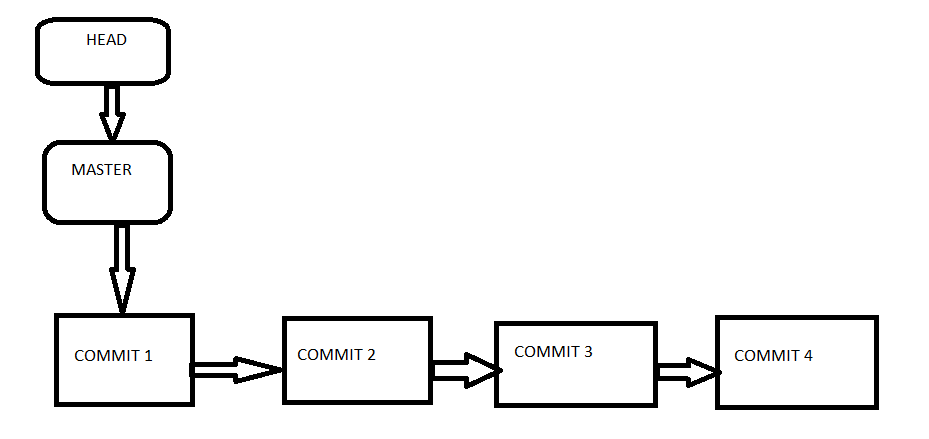
Go to git repositories tab->expand local repository at top->expand branches->expand local-> expand remote tracking-> right click on the remote tracking file which has changes -> select merge-> click ok

Now open the file in eclipse, we will see the changes

Git Head

Head is a pointer which is used by git to keep track of the tip of the current branch in repository

When new commits are made, the pointer moves to the last new commit.



Head has nothing to do with Staging index or working directory

It stores the status of head in .git\refs\heads directory

**Lec9: Configuring Git and Creating Local Repository for a project**

**Create local repository for project:**

Right Click on Project-> Teams-> Share Project->click on Use or create repository in parent folder of project->click on create repository->click on finish

Local repository of project is created (see the project explorer)



We can check if local repositories are created or not by:

go to windows->show view->other ->git->select git repositories and git staging

in git repositories u will see the repository created

Or to see in other view for the git repository created in explorer do:

windows->show view->other ->type navigator->select navigator-> select open

**Lec10: Creating First Commit to Local Repository with .gitignore file**

Go to navigator view on right hand side of project explorer

The files that we want to ignore

Right click on those files-> teams->ignore

Like .setting

To move into staging part,2 ways

1. Either go to staging view and move from unstaged to satged
2. Right click on file->Team->Add to index(this moves from unstaged to staged)

So .gitignore contains the files that we do not want to check in and staged has files that we want to push into remote repository

Give commit message like initial commit. Click on commit.

**Lec11: Adding several commits to local repository and browsing changes**

Changed logintest.java by adding comment1

We can commit by git staging or go to file that u want to commit->right click->teams->commit

We will see the staging view having the updated file-> add the commit comment and click on commit

Add commit 2 and then again commit the file

To see what commits we have done

windows->show view->other->type history->select history->click on open

**history view** is open in toolbar . we cant see history of changes.

To get history of changes-> select project->click on link with editor and selection (right handside of **history view** tab opened in toolbar)

U can select the file also and it will show if any commit is made

**Section 4:**

**Lec13: Push Local repository code to remote github repository**

Navigate to github->signin-> click on your repositories-> click on new ->repository name(sample-git-project)->click on create repository

Repository got created. Copy the path of repository

Go to eclipse-> Git repositories tab->Right click on **Remote->**Create Remote->Remote name(Origin)->Select configure push->Ok->click on change->uri(github) gets automatically populated->provide authentication(user and password)->finish->click on advanced->all branches spec->add all tags spec->finish->2 options (save and push:it will directly push to remote repository , save)->clicked on save

Now remotes have origin and 2 url’s(pulling and pushing). Since we have to push the code ->right click push url->push->give the password->ok->click no->click ok

Or if unable to push local repository to remote(follow below step)

Go to github->Settings->developer setting->personal access token->delete token->generate token->Note(test), expiration(90 days),scope(repo)->generate token

Give generated token as password. It will work

**Lecture14: Pushing changes to git hub**

Changed the logintest.java file and now doing commit to local repository

Righclick on logintest.java file->team->commit

Go to git staging tab view-> add comment-> click commit

Added main.java and again pushed it to local repository by Righclick on main.java file->team->add to index->go to git staging tab view-> add comment-> click commit

To push the changes to remote repository(2 ways)

1. Go to git repositories tab view->branches->local->right click on master->click on push branch->next->finish->close

**Lec15: Cloning from Github**

Go to github-> click on code->copy url

Go to eclipse->window->show view->other->type git->select git repositories->click on open

In the tabs, select clone git repository ->do next next-> click on finish

Or

Go to github-> click on code->copy url

Go to eclipse->file->import->type git->select projects from git->select clone uri->next->next->

next->next->import as general project->next->finish

right click on project->configure->convert to maven

Lec16: Creating, Editing, Renaming and Deleting Files on Github

**Create**

Go to git->click on add file->create new file->name of file(dummy.txt)->commit new file

**Edit**

Click on file->edit->make changes->commit changes

**Rename**

Click on file->edit-> change name of file->commit changes

**Delete**

Click on file->click on delete icon->commit changes

**Lec17: Reviewing your repository as of a particular commit**

Go to commit->go to particular commit id->click on that->review the commit done at that point in time

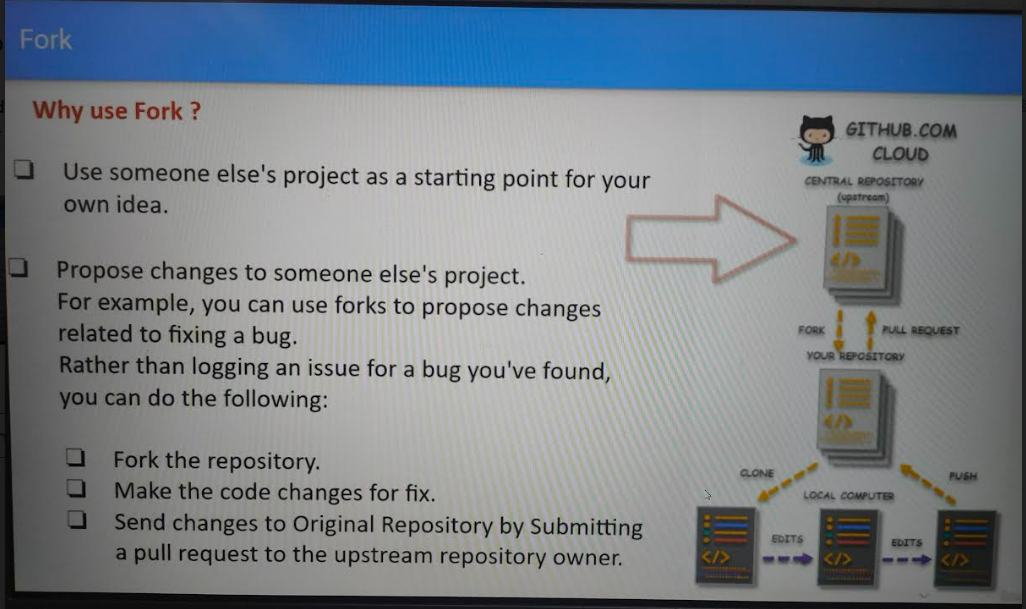
**Lec18: Using GitHub Fork and Pull Request**

**Fork or Forking**

GitHub repositories can be clone to a new Git repository hosted at Github. Github uses the term fork or forking for creating such clones

Forking a repository means creating a personal copy of someone else’s repository. Forks act as a sort of bridge between the original repository(also called upstream repository) and the forked personal copy.

This process does not have any effect on the original/upstream repository code



Forking process ex:

Go to a different github account and search for syedchay/sample-git-project->navigate to the project searched

Click on fork->create fork

Forked project gets created

Now clone this project to eclipse

Updated file logintest.java and added comment7

Right click on file->team->commit and push(change the author id to syedahsanraza07861)-> click on ok

Check the syedahsanraza07861 git repo and see if the changes are pushed .

Now we have to perform the pull request-> click on pull request->click on create new pull request

If there is any merge issue, it will show else it will tell able to merge and then -> click on create new pull request-> add any comment if u want to(I have added added comment 7 by syedahsanraza07861 -> click on create new pull request

After all the above steps completed we will see 1 pull request raised

Sign in to syedchay github from where the fork request was raised

When we will go to the rep , we will see 1 pull request shown-> click on 1 pull request->click on the pull request(added comment 7 shown as link)

It is showing the branch has no conflict with the base branch. Hence we can merge the pull request

Click on merge pull request->click on confirm merge

Now we want to delete the fork repository created in syedahsanraza07861

Login to syedahsanraza07861 github->go to the fork repository and go to settings-> under danger zone(click on delete repository)->click on I understand the consequences, delete this repository

**Lec19: Create, Delete and Restore branches at github**

Create a branch

Go to git repo->click on master->development(write down the branch that u want to create)->click on create

Delete branch:

Go to git repo->branches->click on delete icon of the branch that u want to delete

Restore branches

Go to git repo->branches->click on restore (restore will happen only when u have deleted and u remain on the same page and have not navigated to other pages)

**Lec20: How to add a collaborator to a git hub repository**

Go to git repo->settings->manage access->invite collaborator->write down the collaborator name->click on add

Collaborators have push access to the git repo

We can remove collaborator too

**Lec21: Create a pull request at github**

**This pull request is raised by syedahsanraza07861 saying syedchay as the reviewer**

Pull request is made from the branch where the changes are made to the branch where we want the changes to be merged

For ex: we want development(changes made) to be merged with master

So source :development and target: master

I have added comment 8 in logintest.java file directly in git hub repo and want this change to be merged with master.

2 ways of pull request:

1. When u navigate to git repo it will show development has new commits done a compare and pull request button is enabled. We did in this way

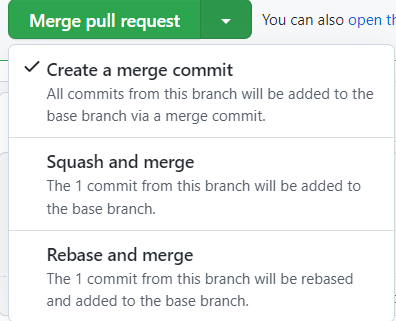
We will see base showing as master and development showing as compare . Add reviewer, assignees, comment->click on create pull request->no conflicts with merge->add comments

1. Go to branches and u will see pull request enabled for those branches

**Lec22: Merge the pull request at github**

Go to git repo->pull request->see the changes done->u can either comment(submit general feedback without explicit approval), approve, request changes(submit feedback that must be addressed before merging, here line by line review comments can also be given).

Since we have to approve the pull request there are 3 options available in this too:



We will choose the first option and click on merge pull request->click on confirm merge

**Lec23: Resolving merge conflicts on github**

Merge conflicts can be:

1). Suppose in 2 branches 2 files get changed at the same line with different code(competing line conflicts can be resolved in github directly) **This pull request is raised by syedahsanraza07861 saying syedchay as the reviewer. When syedchay logs in and goes to repo->pull requests->open pull request by clicking on its label-> scroll down and click on resolve conflicts->do the necessary changes in files and click on mark as resolved -> commit merge->now we will get this branch has no conflicts with base branch->click on merge pull request->click on confirm merge**

2) One branch a file gets deleted and other branch that file got updated. This conflict cant be resolved through githubui

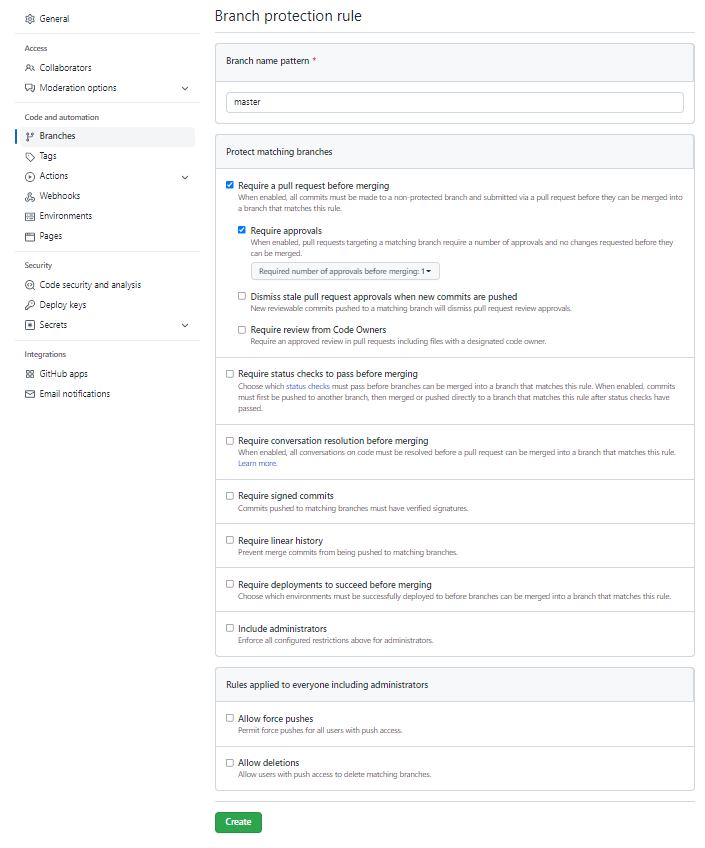
**Lec24:Protecting branches on github**

Protect branches against commits which are not properly reviewed

So we want master as a production branch and development as the branch which needs to have all code changes and until and unless pull request is not reviewed it should not be committed to master

Go to git repo->branches->settings->branches->add rule->branch name pattern(master)->check the first rule->click on create

Check various rules



Lec25: Compare branches at github

**Lec26: Fetch from Upstream Versus Pull**