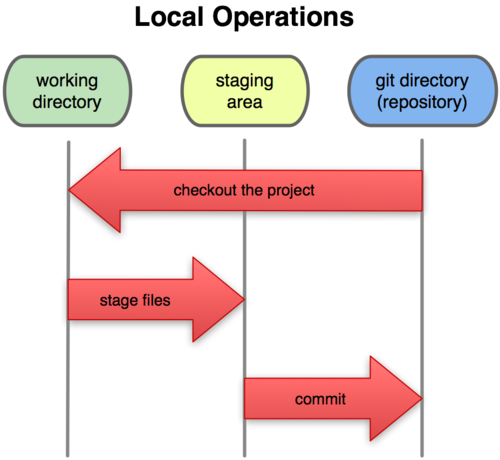
***Topics***

1. [***GIT Hub Introduction***](#_1._Git_Hub)
2. [***How to work with Git Hub***](#_2_How_to)
   1. [*Create an account on Git Hub*](#_2.1_Create_an)
   2. [*Installation of EGit on Eclipse*](#_2.2_Installation_of)
   3. [*Create a repository on Git Hub*](#_2.2_Create_a)
   4. [*Create Branches on Git Hub*](#_2.3_Create_Branches)
   5. *[Delete Branches on Git Hub](#_2.5_Delete_a)*
   6. [*Add Project on Git Repository*](#_2.4_Add_Project)
   7. [*Import Project from Git Repository*](#_2.5_Import_Project)
3. [***Operations On Git***](#_3.Operations_On_Git)
   1. [*Commit*](#_3.1_Commit)
   2. [*Adding files*](#_3.2_Adding_Files)
   3. [*Merge*](#_3.3_Merge)
   4. [*Resolving Conflicts*](#_3.4_Resolving_Conflicts)
   5. [*Fetch & Pull*](#_3.5_Fetch_&)
   6. [*Push*](#_3.6_Push)
   7. [*Synchronize*](#_3.7_Synchronize)
4. [***Review Document History***](#_4._Review_Document)

# *1.* *Git Hub Introduction*

EGit *is an Eclipse plug-in (software component) which allows you to use the distributed version control system* Git *directly within the Eclipse IDE.* EGit *is based on the* JGit *library.* JGit *is a library which implements the* Git *functionality in Java.*

*There are 3 main sections in EGit project .*



*1.* ***Git Directory*** *:* where Git stores the metadata and object database for your project. This is the most important part of Git, and it is what is copied when you clone a repository from another computer.

*2.* ***Staging area :*** is a simple file, generally contained in your Git directory, that stores information about what will go into your next commit. It’s sometimes referred to as the index, but it’s becoming standard to refer to it as the staging area.

*3.* ***Working directory :*** is a single checkout of one version of the project. These files are pulled out of the compressed database in the Git directory and placed on disk for you to use or modify.

# *2. How to work with Git Hub*

# *2.1 Create an account on Git Hub*

Signup on <https://github.com/> for creating your account on Git Hub.

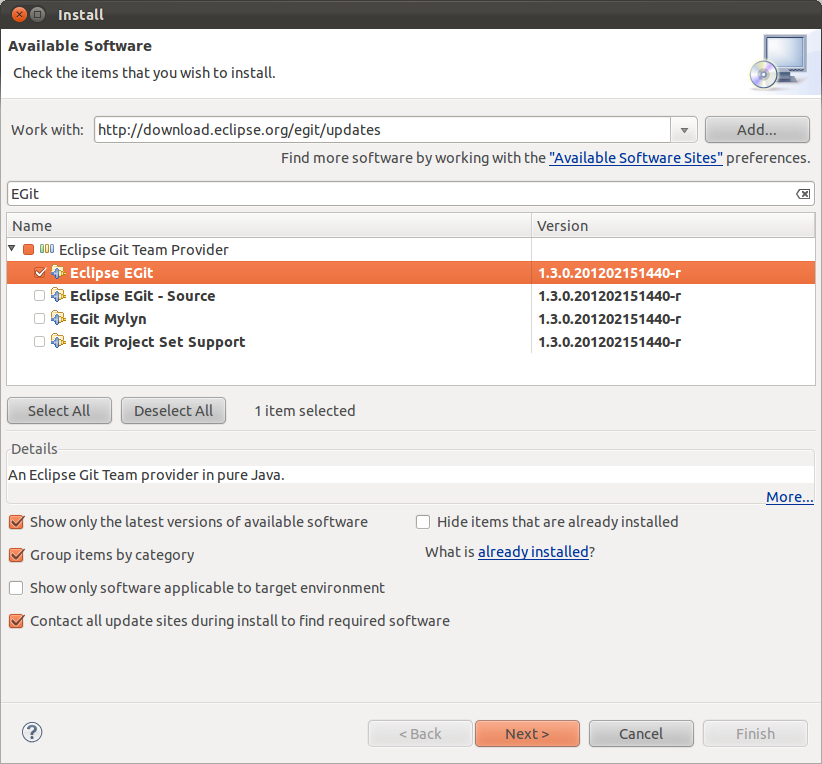
# *2.2 Installation of EGit on Eclipse*

Goto Help menu on Eclipse → ***Install new Software*** menu entry.

EGit can be installed from the following URL:

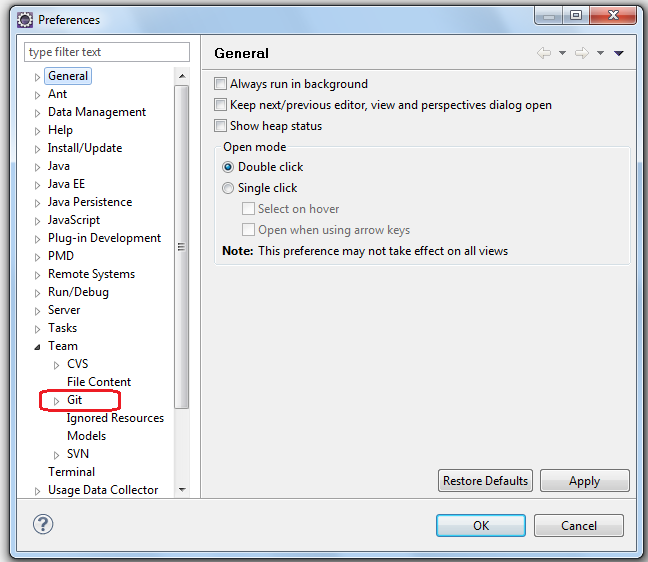
[http:*//download.eclipse.org/egit/updates*](http://download.eclipse.org/egit/updates)

The dialog to install EGit is depicted in the following screenshot.



To check whether EGit plugin is installed or not , go to ***window -> Preferences***

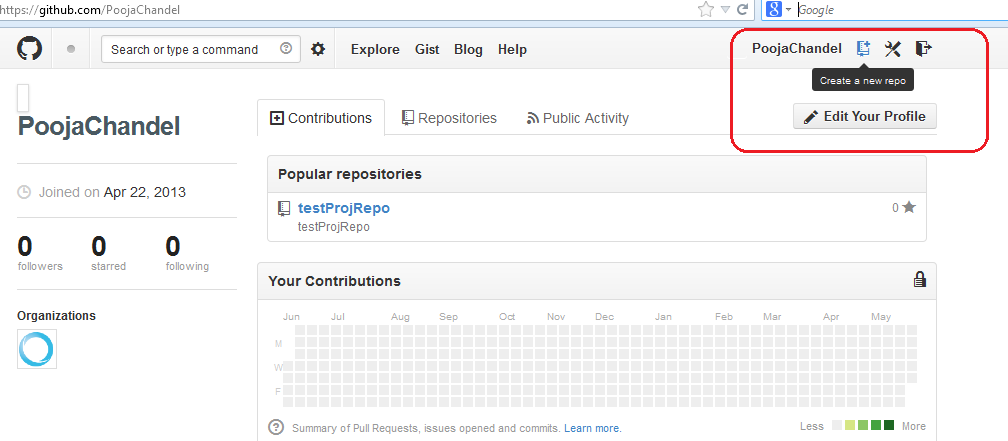
For ref. please see the below image :-



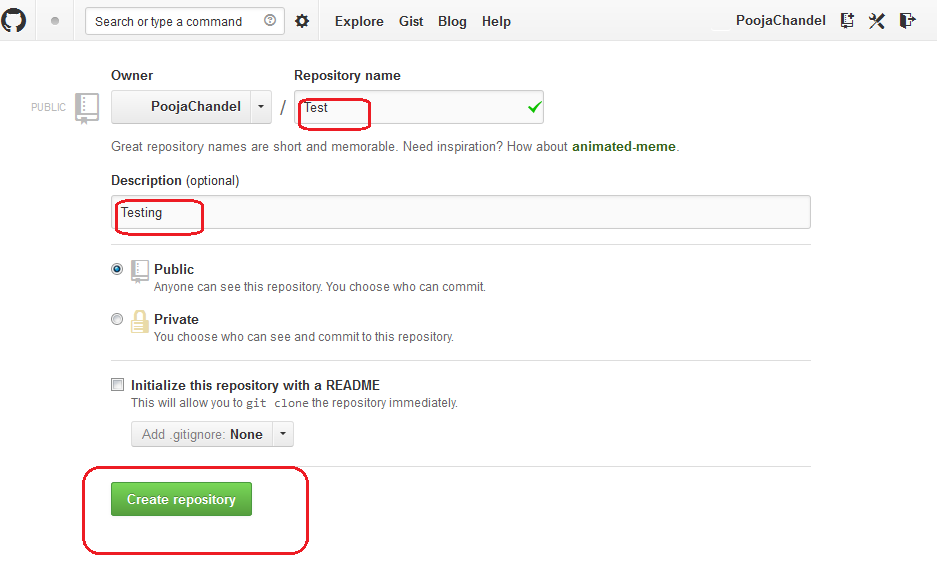
# *2.3 Create a Repository on Git Hub*

Login on <https://github.com> . Follow the steps as provided in below screen shots :

1. Click on create a new repo icon as shown below :

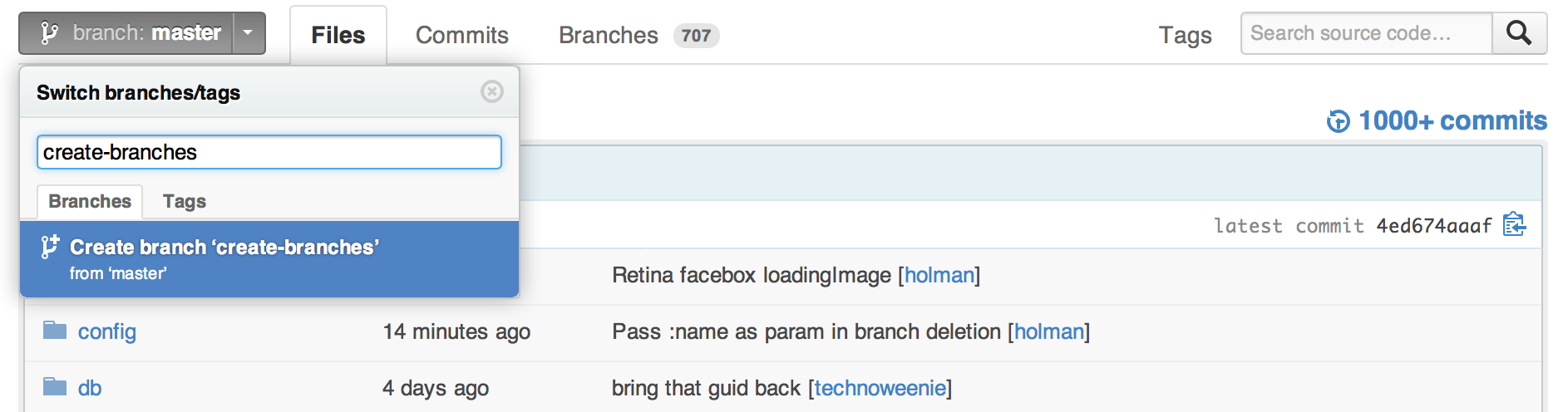


1. Fill the repository name and its description as shown below and click on create repository.



# *2.4 Create Branches on Git Hub*

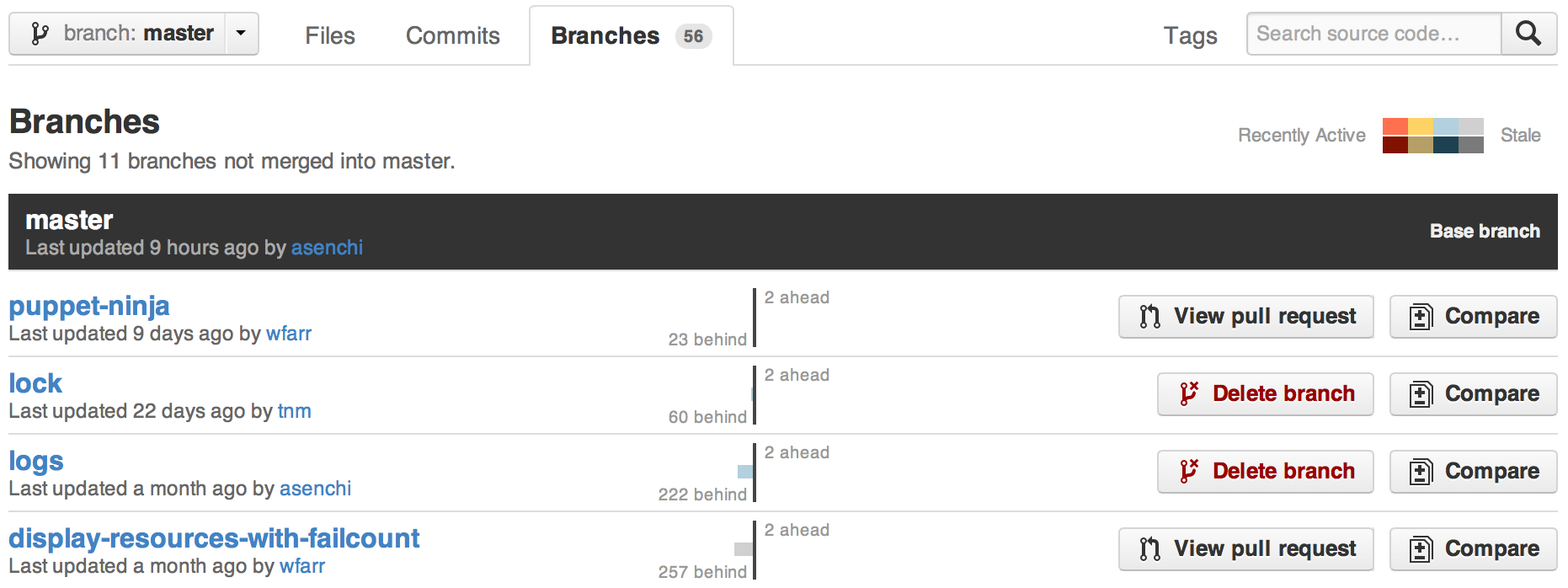
In your repository's branch selector, just start typing a new branch name. We'll give you the option to create a new branch:

[](https://f.cloud.github.com/assets/2723/45822/c26b6cd2-57e3-11e2-88e2-267ef1952025.png)

We'll branch off of your current context. For example, if you're on the bugfix branch, we'll create a new branch from bugfix instead of master

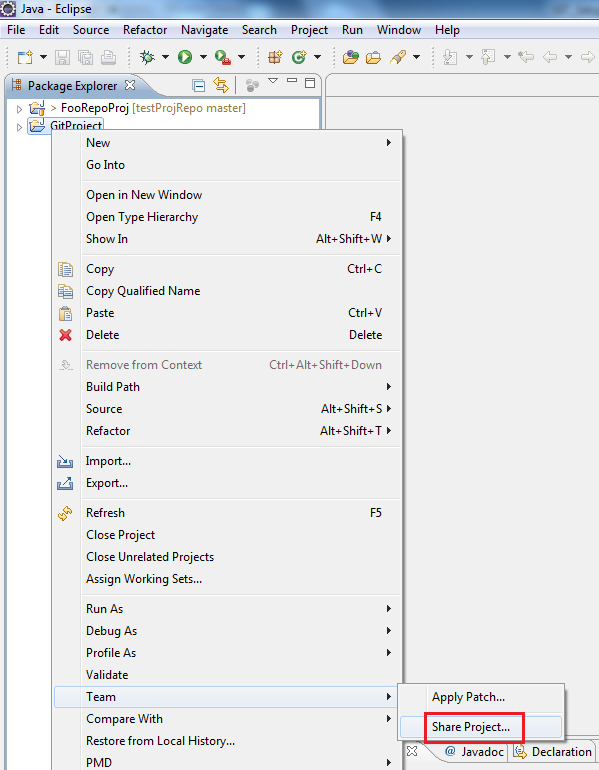
# *2.5 Delete a branch*

*You'll also see a delete button in your repository's Branches page:*

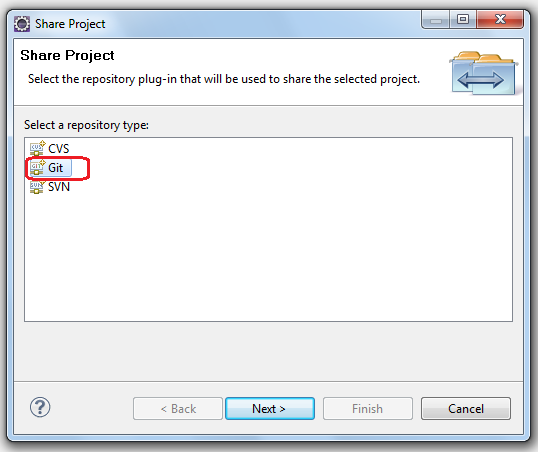
[](https://f.cloud.github.com/assets/2723/52171/a28b2ade-59ef-11e2-94c0-cbf11494a858.png)

# *2.6 Add Project on Git Repository*

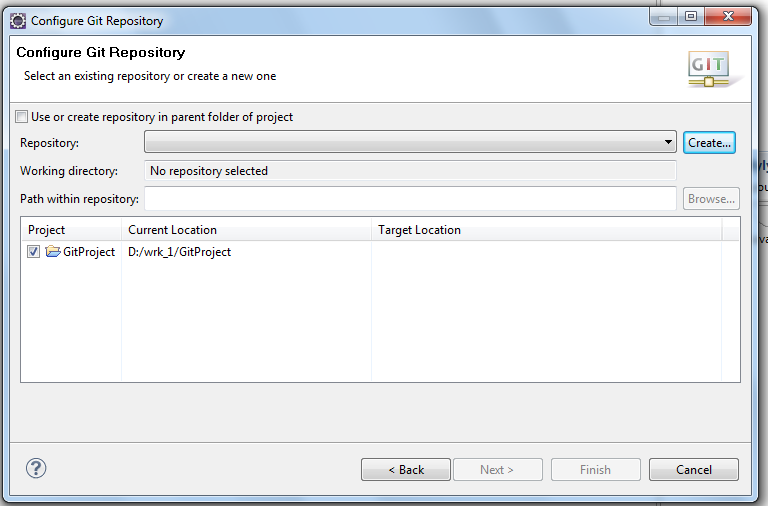
1. Create a project on Eclipse . Rightclick on project and navigate to ***Team -> Share Project*** (as shown below)



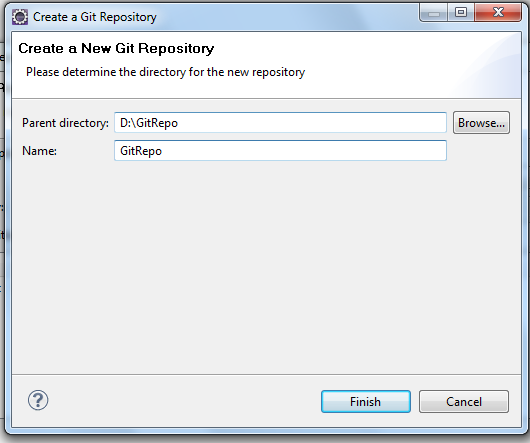
1. Select Git and click Next as shown below :-



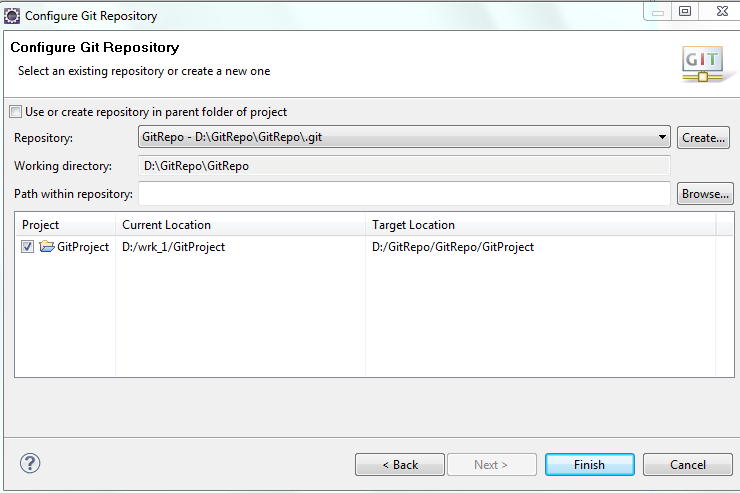
1. Create local repository by clicking create



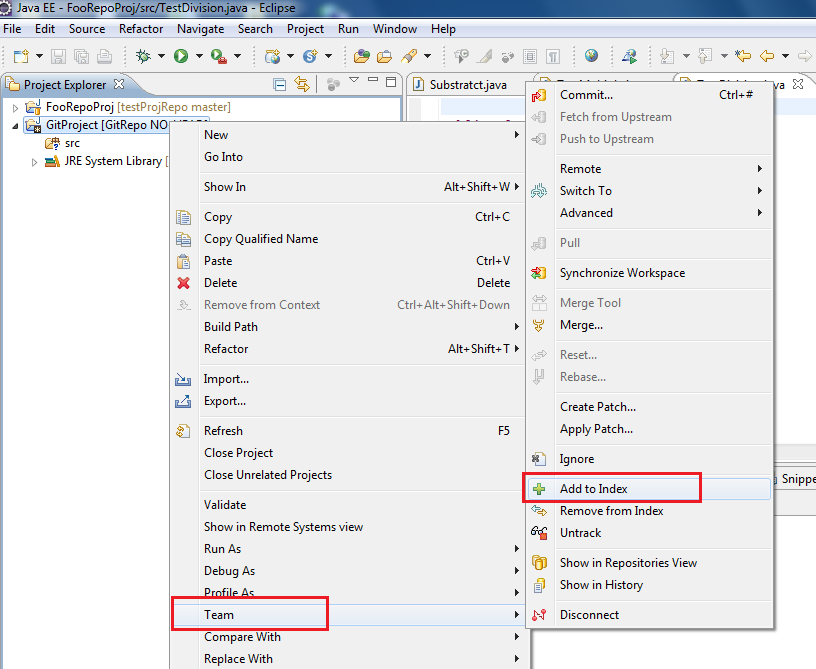
1. Create a folder for repository on local and pass the path as shown below. Also give the name of your repository of your choice and click Finish.



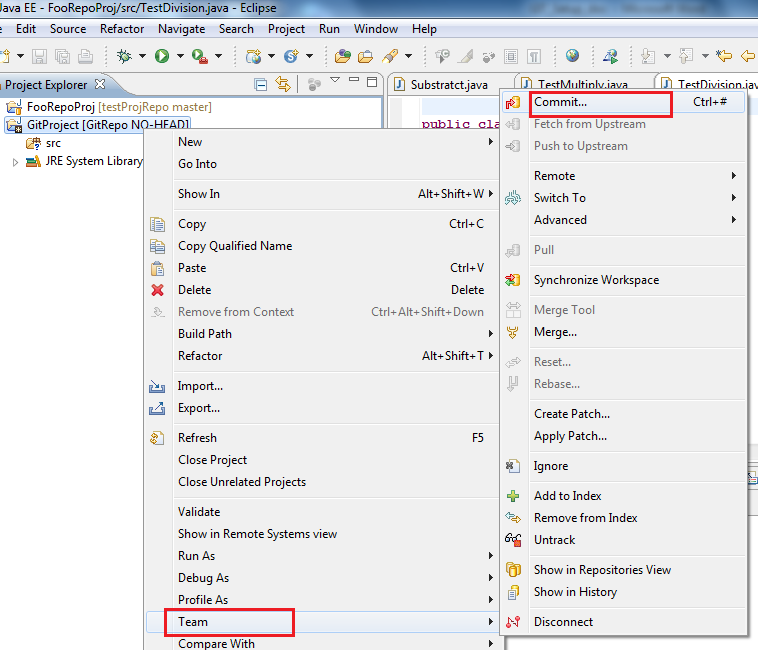
1. Click Finish.



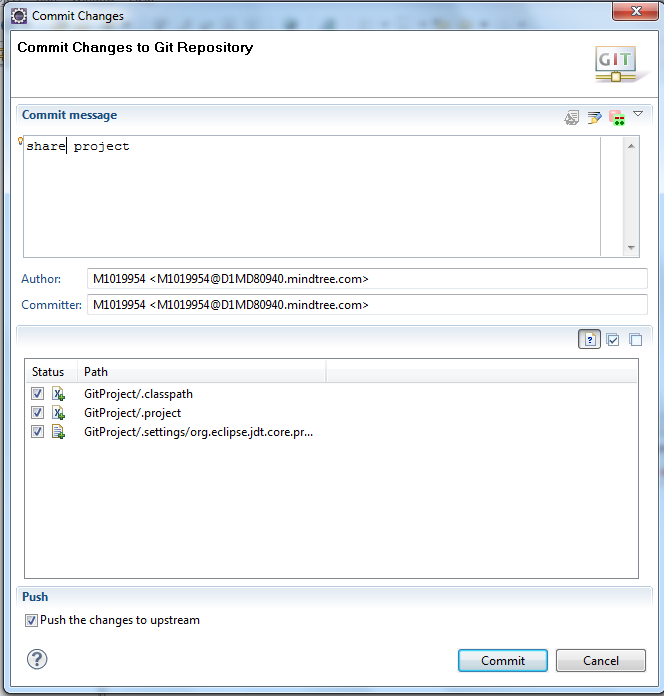
1. Now the project is present in your local repository to add project at remote repository , rightclick on project and navigate to ***Team-> Add to Index.***

******

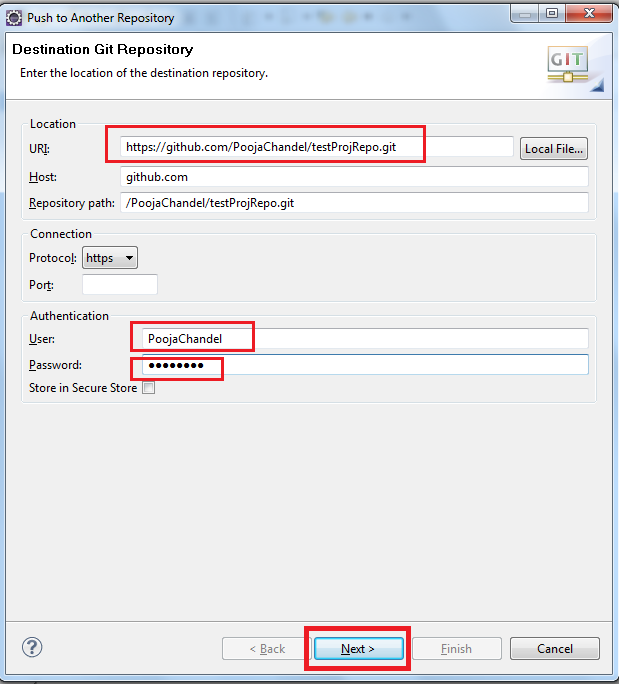
1. Now rightclick on project and navigate to ***Team -> Commit***.

******

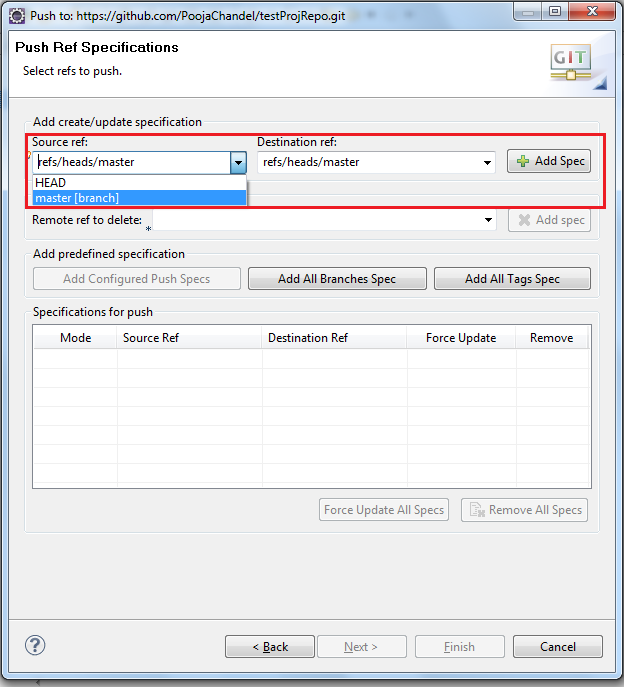
1. Add commit message and ensure that checkbox for push should be checked and click commit.

******

1. Enter the GIT repository uri path , Git Credentials as shown below and click next.

******

1. Choose branch from dropdown list and click on Add spec as shown below :



1. Check the Force update checkbox and click Finish (image 1.1)Confirmation window(1.2) will get displayed .

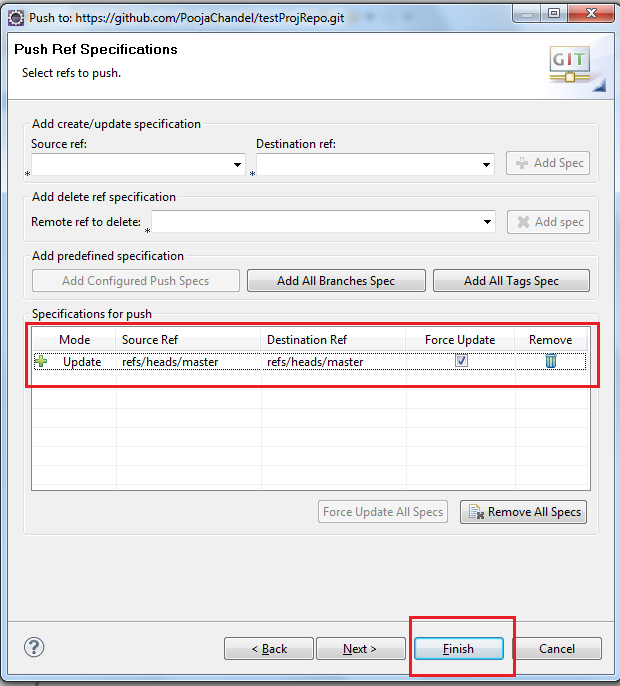


Image 1.1

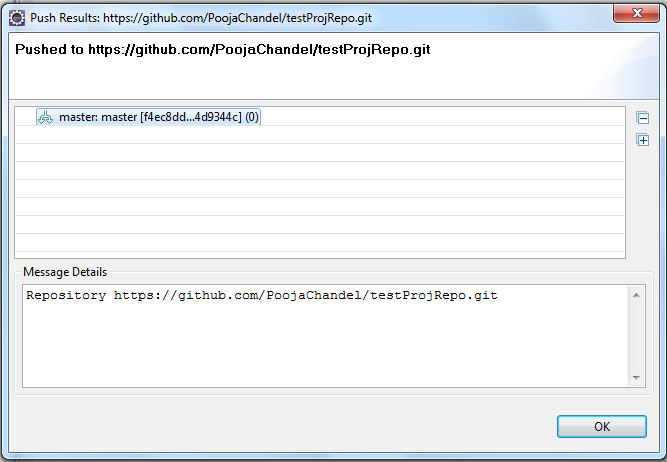
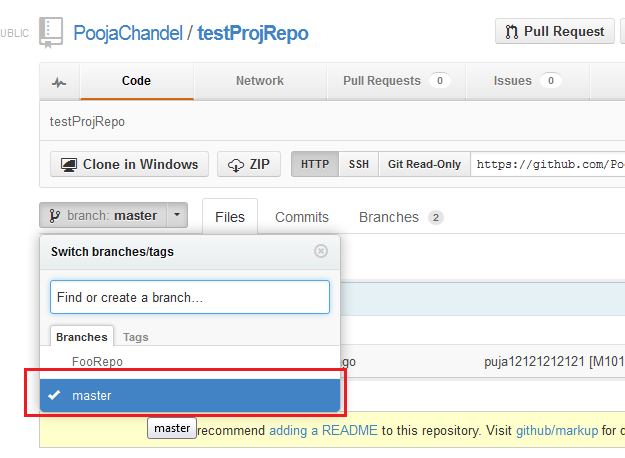


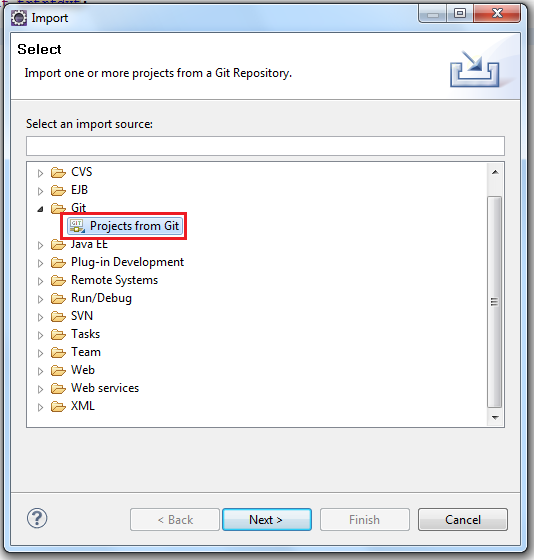
Image 1.2

1. Now check the project at Git Hub , go to repository and click on master branch in dropdown list. You can find your project added there.

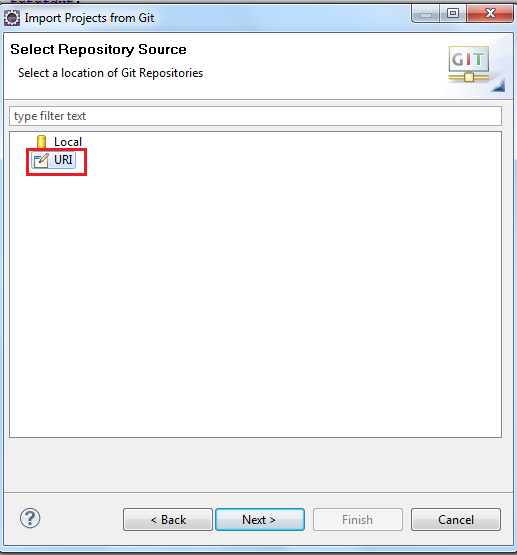


# *Import Project from Git Repository*

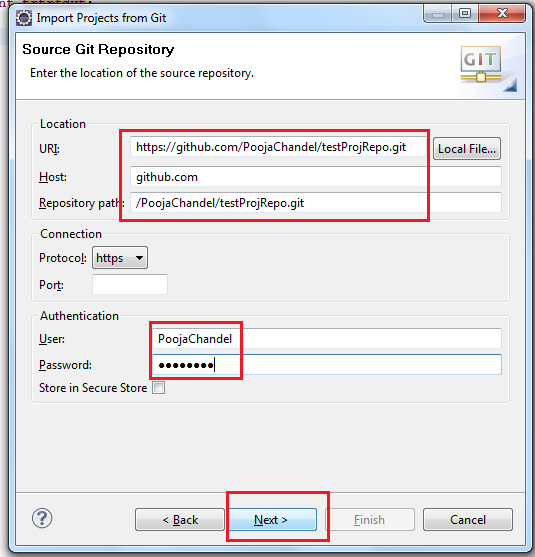
# For importing project from GIT repository right click on package explorer in Eclipse and click Import. Now Select GIT-> Projects from GIT and click next.



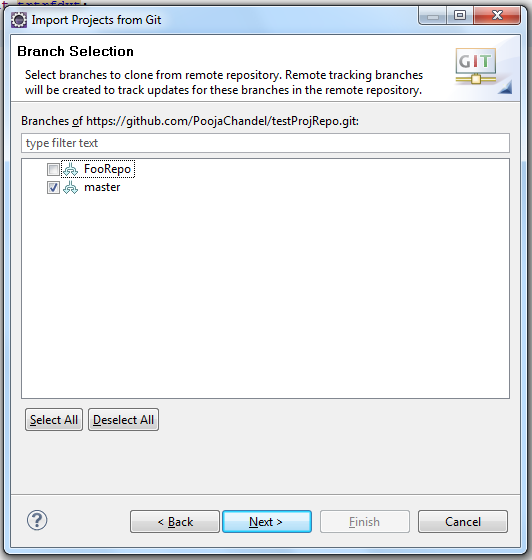
Now select URI and click next.



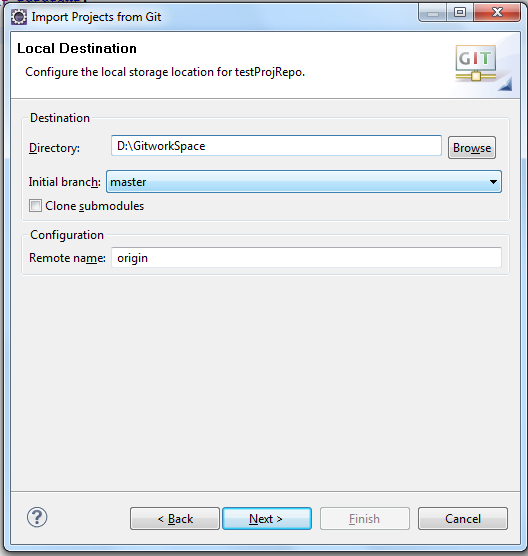
Now enter the uri and Git hub credentials and click next.



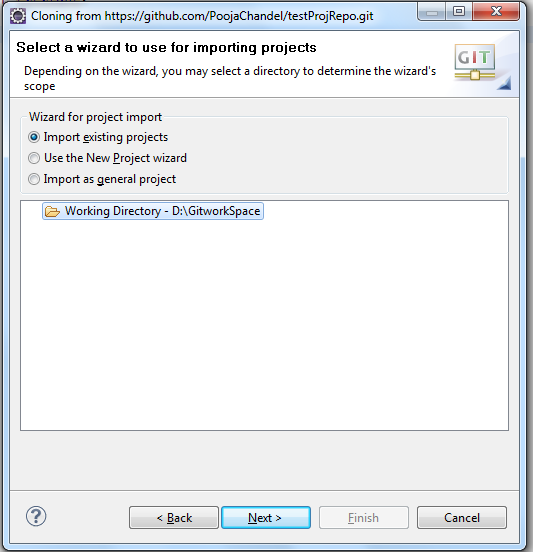
Now select the branch and click next. Suppose master is the project which is containing the project which you want to import and click next.



Now enter the path of workspace and click next.



Select options as shown below and click next.



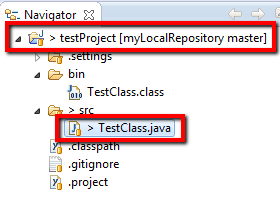
Now select the projects which you want to checkout and Click on Finish.

# 

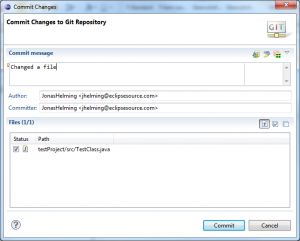
# *3. Operations On Git*

# *3.1 Commit*

Now you can start to modify files in your project. To save changes made in your workspace to your repository, you will have to commit them. After changing files in your project, a “>” sign will appear right after the icon, telling you the status of these files as dirty. Any parent folder of this file will be marked as dirty as well.

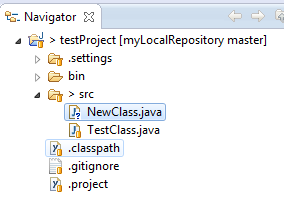


If you want to commit the changes to your repository, right click the project(or the files you want to commit) and select Team => Commit. This will open a new window, allowing you to select the files you want to commit. Before you can commit the files, you will have to enter a commit message in upper text box. After you are done, click commit to commit the selected files to your repository.

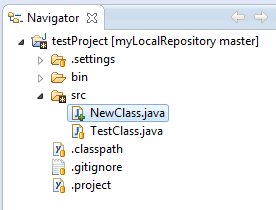


# *3.2 Adding Files*

To add a new file to the repository, you will have to create it in your shared project first. The new file will, again, appear, appear with a question mark.



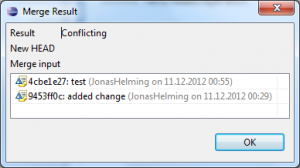
Right click it and navigate to Team=> Add. The question mark will turn into a plus symbol and the file will be tracked by git, but it is not yet commited. All of the files parent folders should now have a symbol that looks like an asterisk indicating that it is ‘staged’. In the next commit, the file will be added to the repository and the plus symbol will turn in to a repository icon. The repository icons of all the file’s parents (packages/project) will turn into staged icons.



# *3.3 Merge*

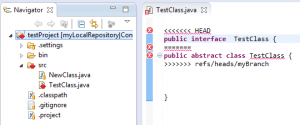
To merge one branch into another, you will have to checkout the branch you want to merge with. Right click the project node and navigate to Team=> Merge. Select any branch (other than checked out branch) and hit Merge.The merge will execute and a window will pop up with the results. The possible results are Already-up-to-date, Fast-forward, Merged, Conflicting result (see image bellow) will leave the merge process incomplete. You will have to resolve the conflicts(see next section).

A failed result may occur when there are already conflicting changes in the working directory.



# *3.4 Resolving Conflicts*

If your merge resulted in conflicts (note the red symbols on the file icons), you will have to resolve these manually. Open the conflicting files and scroll to the conflicting changes marked with “<<<<<<<”.



# *3.5 Fetch & Pull*

When cloning remote repositories, Git creates copies of the branches as local branches and as remote branches. A Fetch operation will update the remote branches only. To update your local branches as well, you will have to perform a Merge operation after fetching. The operation pull combines Fetch and Merge. To perform a Fetch, select Team=>Fetch from.. from the projects context menu. Enter the repository you want to fetch branches from.(If you cloned this repository, the remote branch will be selected by default)In the following window you will have to select what you want to fetch. As default, all branches are selected. The result of the fetch operation will be shown in final confirmation window. Follow the same steps to apply to Pull.

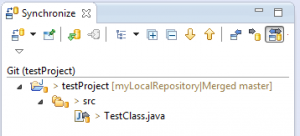
# *3.6 Push*

Local changes made to your local branches can be pushed to remote repositories causing a merge from your branches into the branches of remote repository(X pulls from Y is same as Y pushes to X) The Push wizard is pretty much the same astheFetch wizard. First, right click the project node and navigate to Team=> Push… . Enter the repository you want to push your branches to (the default for this will be the same as the Fetch default if you didn’t configure a Push default) and hit Next. Choose the branches you want to push or click Add all branches spec if you want to push all branches. You can also select branches you want to delete from the remote repository. If you are done hit Finish. A final window will show the results of the Push.

# *3.7 Synchronize*

Comparisons between your workspace and the local repository or between the current branch and others and are done via the Synchronize operation. If you right click Team => Sychronize Workspace, your local workspace will be compared with the current branch showing uncommitted changes. If you select Team => Advanced => Synchronize… . , you can select other branches to compare your current branch with. In this case you can also include local uncommitted changes.

To compare the branches you may want to switch to the Synchronizing perspective, where you can get a more detailed view of several changes.  Here is an example of a Synchronize operation in the Synchronizing perspective:



## *4. Review Document History*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.no. | Created by | Creation Date | Reviewed by | Review Comments |
| 1.  2. | Pooja Chandel  Jeevan Raghvendra Dhanak | 12 June 2013  12 June 2013 |  |  |