**GitHub Notes**

Repository

Branch

Pull

Push

Commit

Working Directory

Merge

When you start working with Git in a repository, it will clone the repository to your local machine.

3 states

1. Working directory

2. Staging Area

3. Repository

Author speaks of a 4th state viz. Remote (repository on Git sever)

Workflow

Working in the master branch initially

1. Create a file in the working directory. This file is untracked as far as Git is concerned

The file name for example is start.txt

2. Add to Git.

in Linux, the command is: get add start.txt

The check status git status

It is now in the Staging area and can be removed without any ramification.

Can build several changes and add to staging area

3. Commit

git commit -m "<comment why commit is taking palce>"

By this command, the start.txt file has been moved to the local repository

4. At this point the file is still on local repository

It needs to be 'pushed' to the github repository

When moving a file to Github server, it is called a push.

Command: git push origin master

origin is the name that github gave the master during the cloing process. Refers to the GitHub copy of your repository

master is the branch that is being pushed

After running this command, the file is pushed to the github server. This file shld be visible in the github website.

---

Creating a new Repository

1. Command for new repository

git init fresh-project

2. ls will show a directory called fresh-project

change to this directory and you can start adding code files here. This is yr workign directory master branch for fresh-project repository

3. This folder also contains the .git folder

The actual repsitory lives in this .git folder

4. git status command will show where you are

On branch master

master is the default branch

---------------

For an existing project, how to add git?

Go into the project directory and enter the command:

git init

This will create a new git repository using the directory name

However, existing files are not committed or pushed automatically

Since there cld be multiple files in the project and cannot be added one by one to the repository, there is a way to commit all files in one shot

git add .

Will move all files to staging

Then run the commit statement:

git commit -m "<comment>"

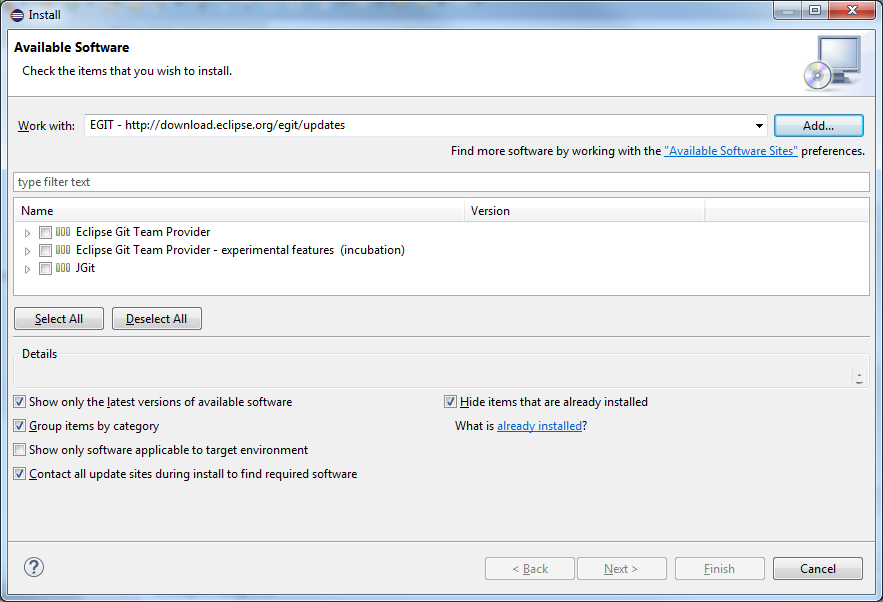
git status will show nothing to commit

--------------

**Working within Eclipse**

In Eclipse, Go to Help->Install new software for installing Git.

Install Window pops up



Click on Add

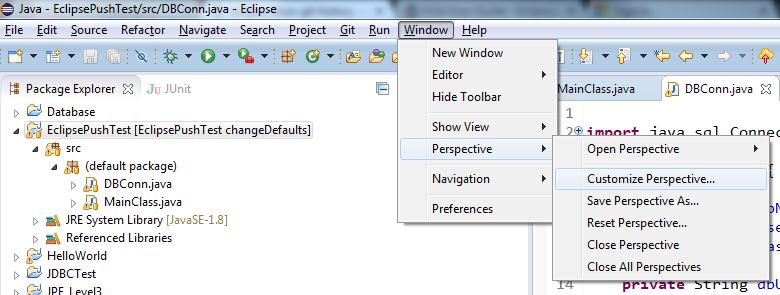
In the name, type EGIT and Location is: <http://download.eclipse.org/egit/updates>

Select all options Eclipse Git Team Provider, etc. and select Next and then Finish.

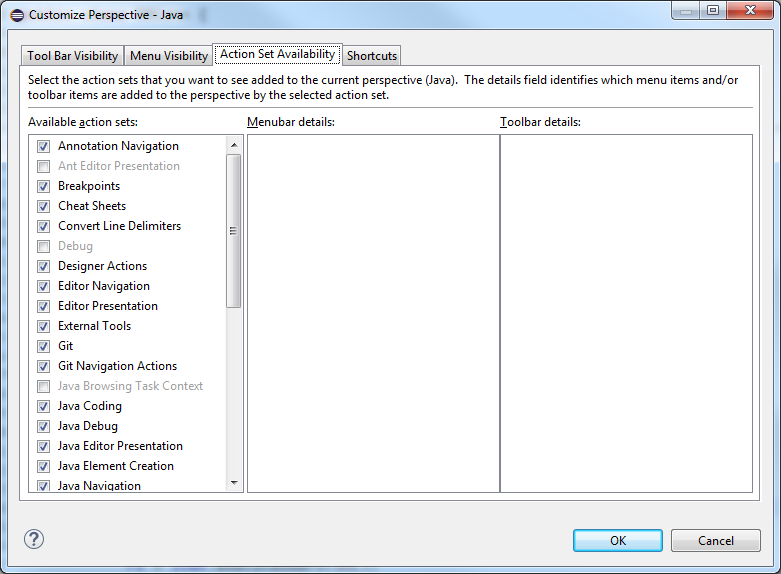
This will install EGit.

**Setup**

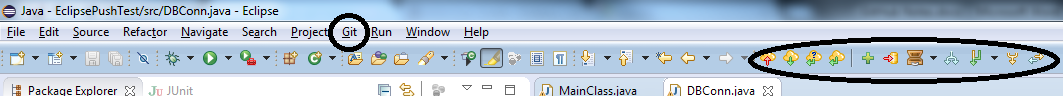
**Add Git menu to the menu bar and menu buttons.**  
1. Select Window->Perspective->Customize Perspective



2. Go to Action Set Availability and Select Git and Git Navigation Actions

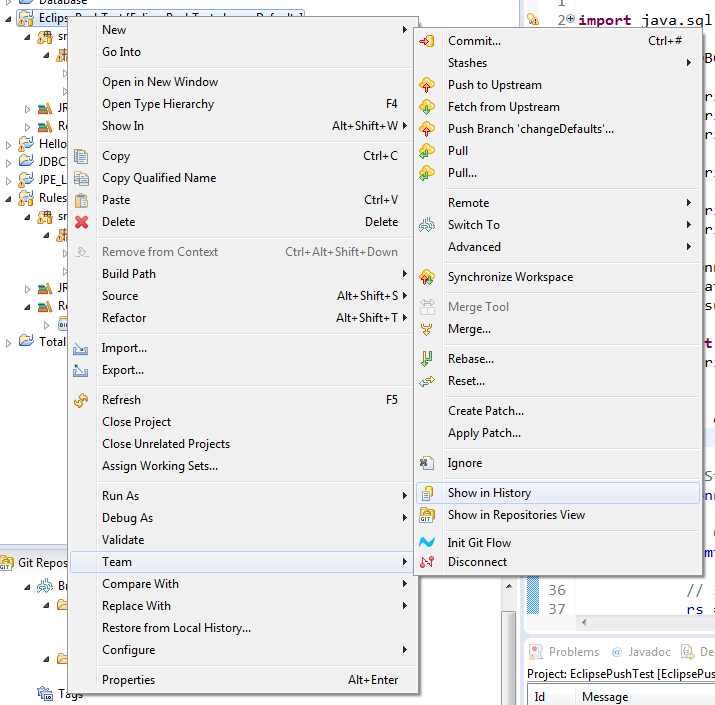


1. Then go to Tool Bar Visibility and Menu Visibility and select Git.
2. The Git buttons and Git should appear in the menu bar.



**Add History panel**

To view the history, right click on the project and choose Team->Show in History



The history panel will open up.

<https://www.youtube.com/watch?v=1XNVWpjPoio>

In this video, the author shows how to work with eclipse and git

1. Create a new Java project

2. On the project Properties, select Team->Share Project and select Git

3. On the Configure Git Repository window, select to create a Repository and a target location in the Git folder:

C:\Users\Kunal\git\RBE

This will map the folder in the workspace to the git directory

4. Open the Git Repositories view by going to Window->Show View->Other->Git Repositories

This shows the local repositories and the Working Dir in this

5. Working Dir contains a folder with the name of the project

6. There is a sub folder called .git which contains the repository

7. As per the workflow, the project files have to be moved to staging and then committed. This can be done in the Staging view.

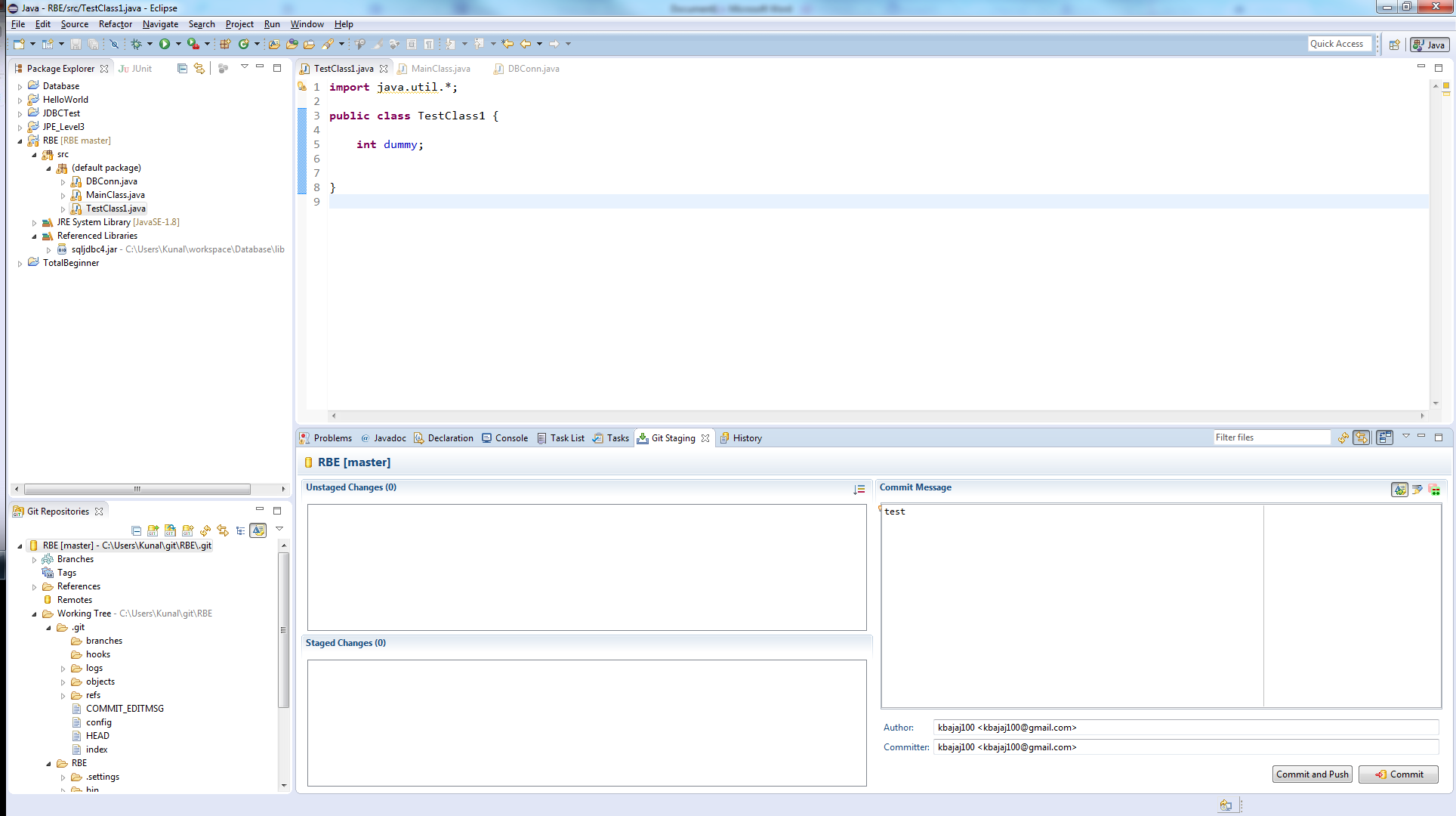
8. The History view chows the history of commits.

Next video in this series: https://www.youtube.com/watch?v=g5h1VLgC85o

1. Important to understand that each new commit creates a new version.

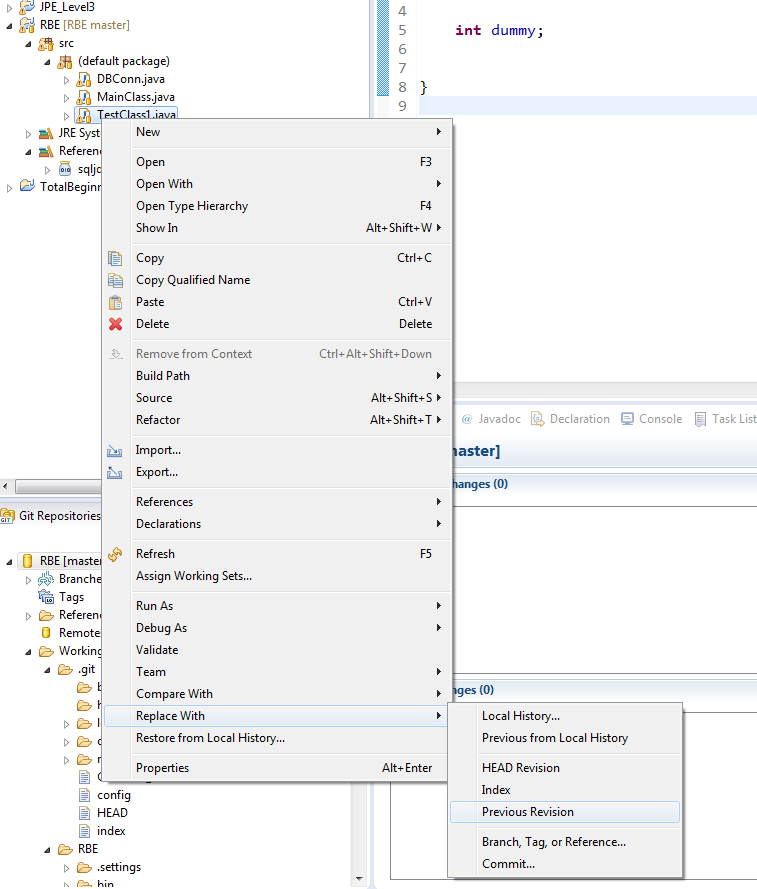
2. Supposing a commit is done and then another file is created which needs to be included in the previous commit (maybe this file was forgotten to be included).

3. This can be done by staging the additional file and then clicking on the Amend Commit button in the Staging tab.



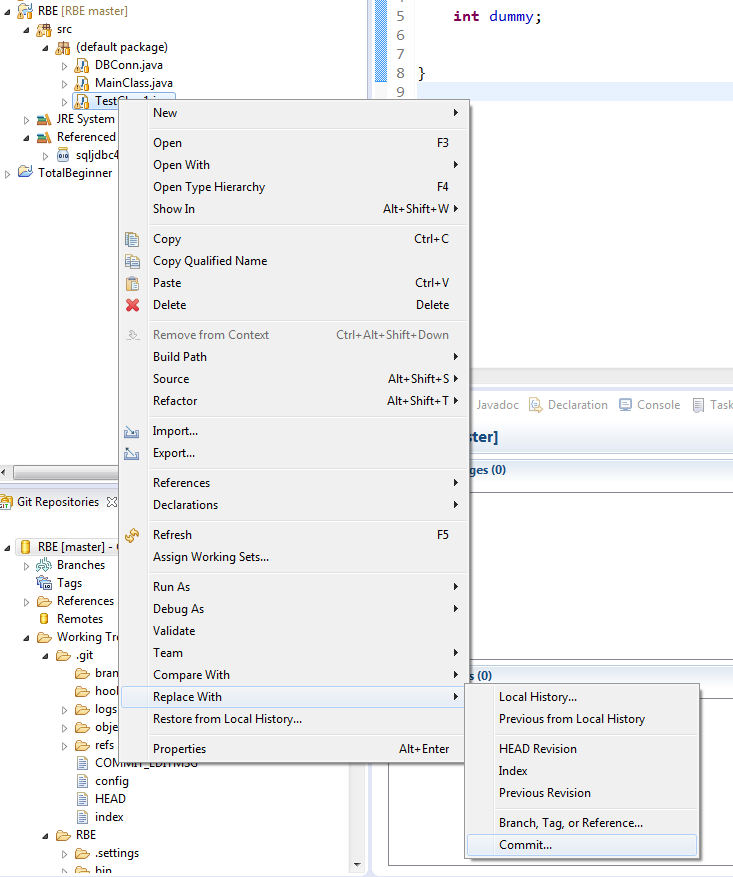
Pitfall of this in a team environment: Developers may have already started to work on the previously committed version. In this example, so far everything is based on a local repository.

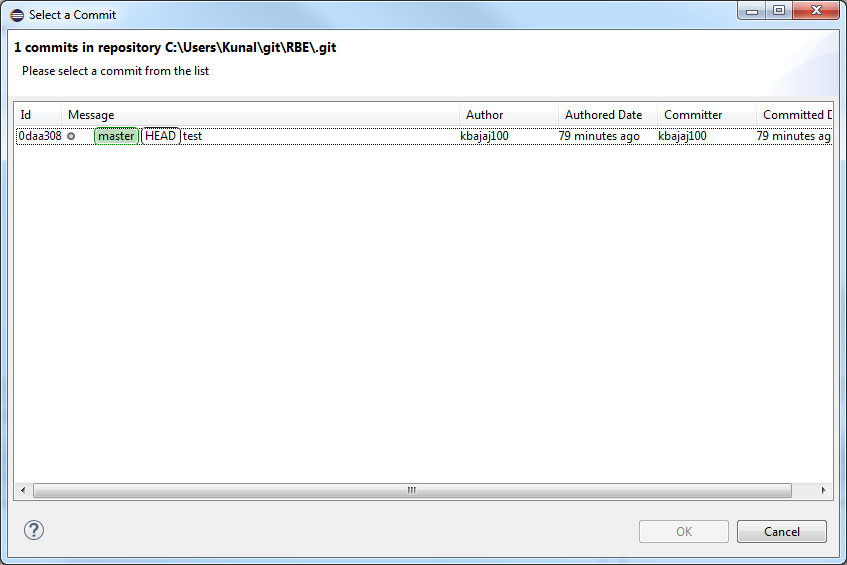
To revert to a previous version of a committed file, right click on the file and select Replace with->Previous Revision



To revert to a previously committed version from 2 or 3 commits ago, right click on the file and select Replace with->Commit

See diagram below.

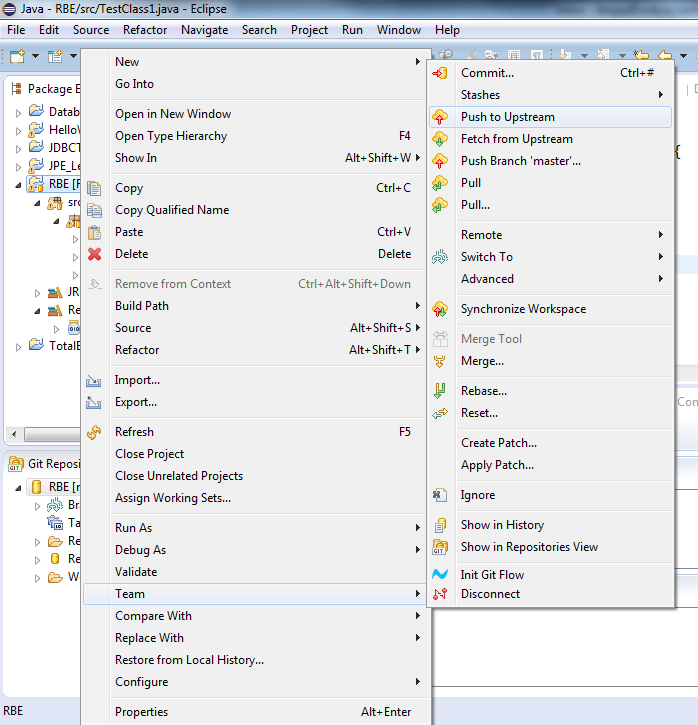


This will bring up the window listing all committed versions. You can choose the version you want.

<https://www.youtube.com/watch?v=KfeqnernMmE>

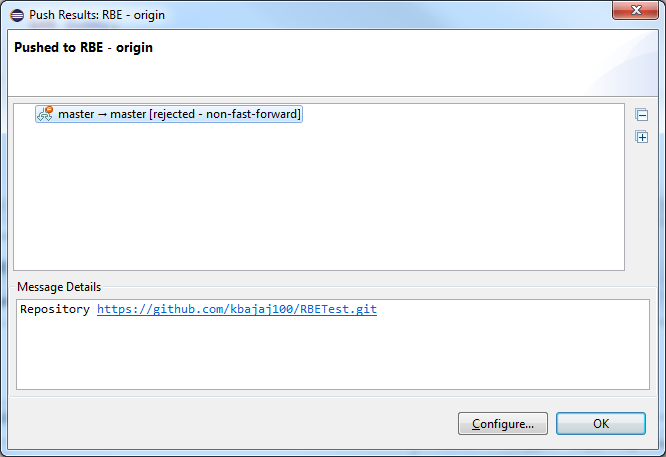
So far, the previous 2 videos showed working with Git in a local repository. Now, to link the local repository to an online repository, follow these steps.

1. On the project, right click and select Team->Push to Upstream



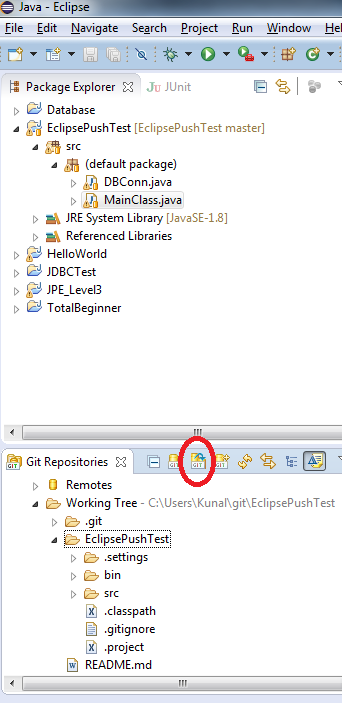
1. This will bring up the Login page. Enter the credentials.
2. If setting up for the first time, you will need to copy the HTTPS link form the github site and paste here. It will automatically populate the remaining settings.

The issue I have run into as part of this is that I keep getting the following error when trying to push.



The resolution for this is as follows.

1. Create the repository in the Github site
2. In eclipse, choose the Clone a Git repository and add to this view. See below:



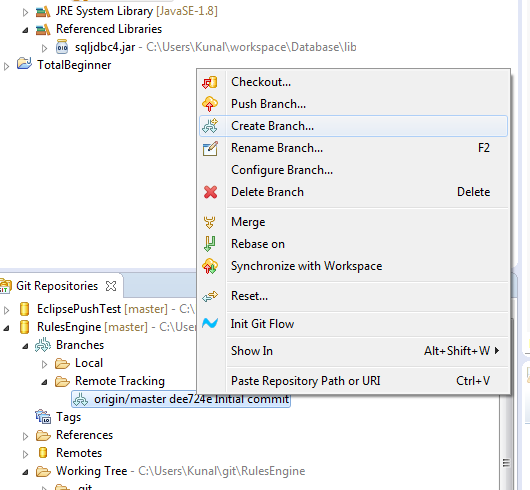
1. Edit the Readme file and try to commit and push. This was successful
2. Create a new Java project with the same name as the repository. This is important as the push seems to go to the wrong project or repository.
3. Then add the code files.
4. Then right click on the project and select Team->Share Project and select Git
5. It will ask for the URL. Paste the URL for the repository that was cloned.
6. Then try to stage the files and commit and push.
7. This was successful
8. Now, try just committing and then pushing individually.
9. So it seems you cannot push an individual file.
10. I was able to commit an individual file but then had to push the entire branch.
11. I was able to revert to a previously committed file and commit and push that as well. It seems that to push an individual file, you have to do that from the staging window.

<https://www.youtube.com/watch?v=KfeqnernMmE>

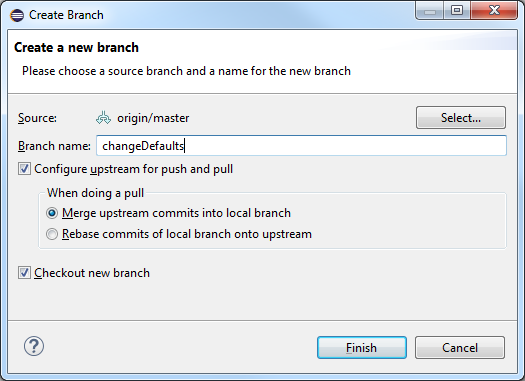
Using Git Server and Eclipse

How to create a Branch?

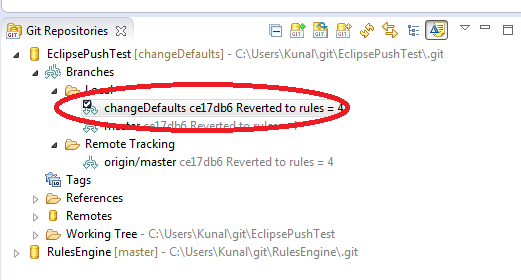
1. Right click on the Remote tracking and select create branch.
2. A branch can be created in the local repository or the remote repository. Better to create in the remote.



1. Enter a name for the branch and select Merge. This is important. The other option Rebase does not change the version numbers properly.



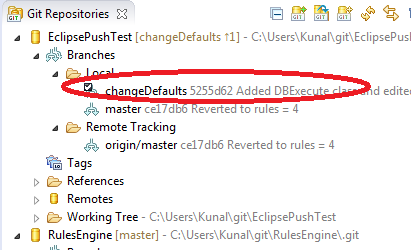
1. The changeDefaults branch shows up under Local with a check on it implying this branch is checked out by you.



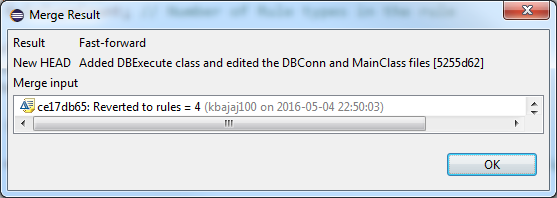
1. Now that the local branch is setup, you can make your changes.

In this case, I have added a new file and modified the MainClass file

1. By committing the changes, the changes get committed to the branch only. This can be verified as the branch ID changes from ce17db6 to 5255d62 while id of the master remains the same

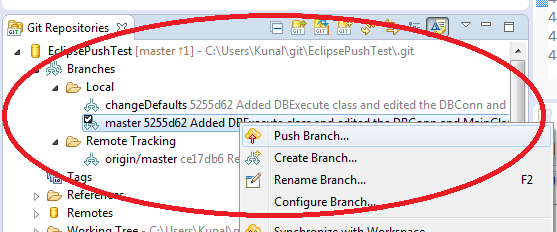


1. To merge the changes into the master (local) branch, here are the steps:
   1. Double click on master (local)
   2. Right click on the branch and select Merge
   3. You shld get a Result popup with a fast forward message

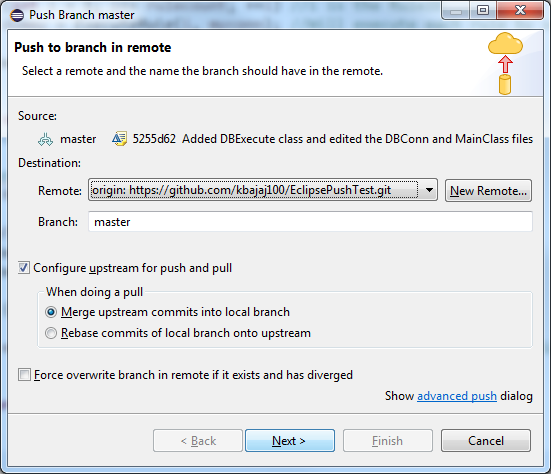


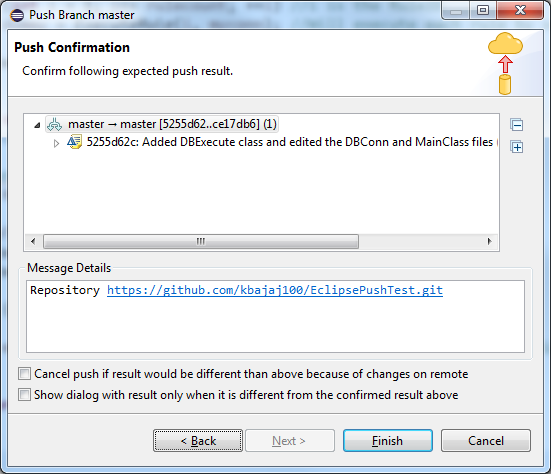
Fast forward implies that Git has to do no work and the merge can be done cleanly.

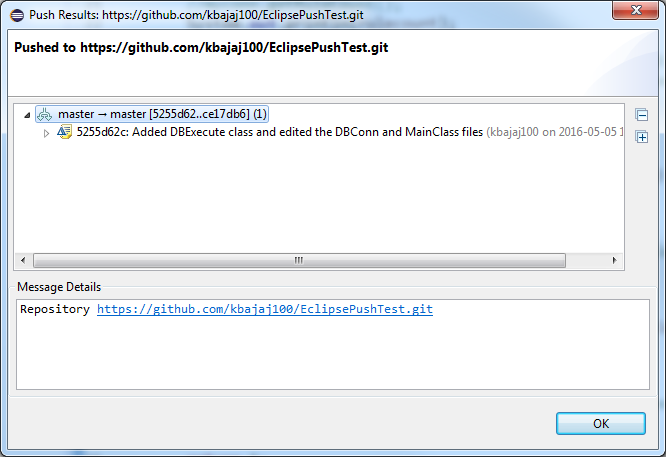
1. By doing this merge, the ID of the branch and master will be the same.
2. To push the changes to the master (remote), right click on the master branch and click on Push



1. Follow the popups and push.







The history should show all actions. The history is currently not doing this. This issue needs to be resolved.

**Conflict resolution**

**This is a major topic which is covered in the youtube link.**