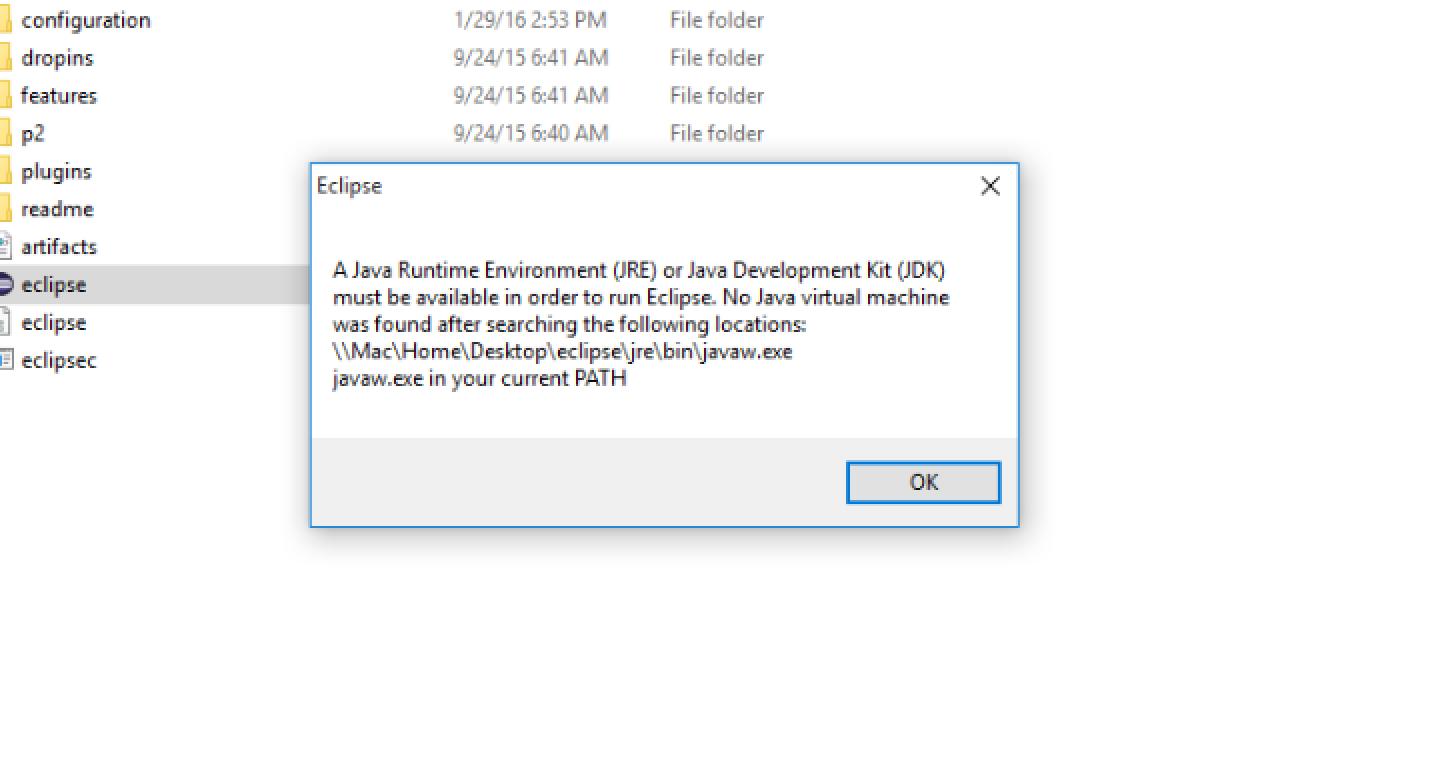
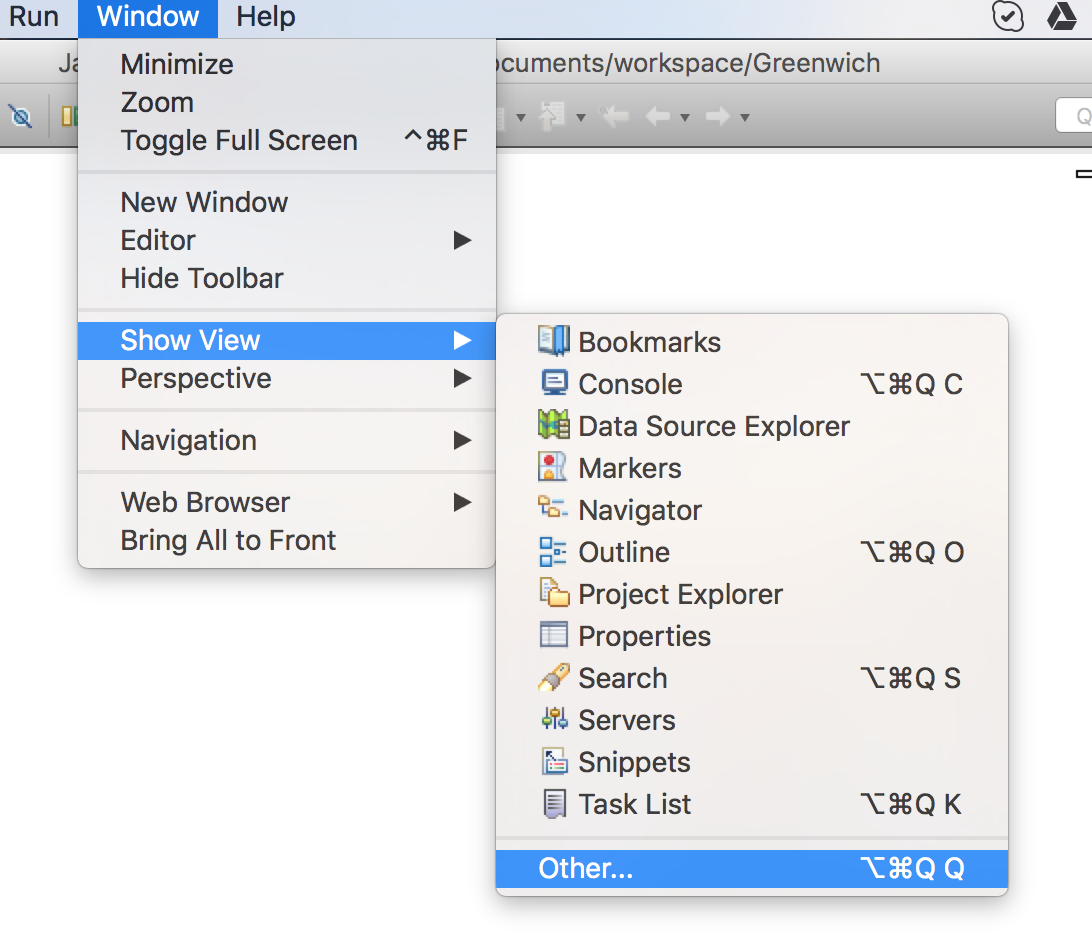
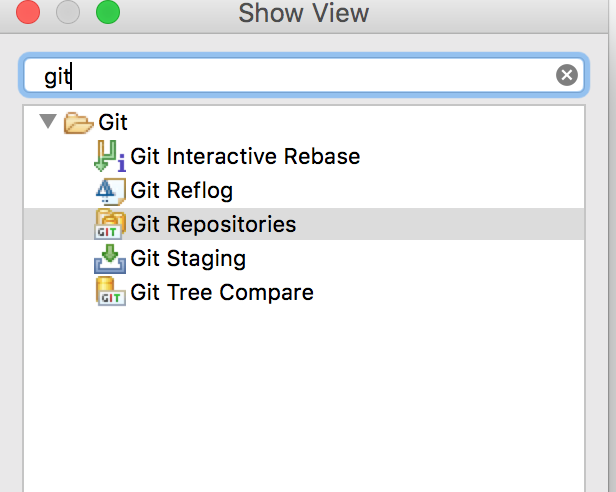
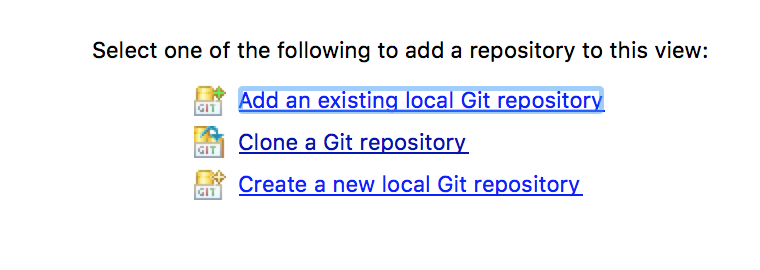
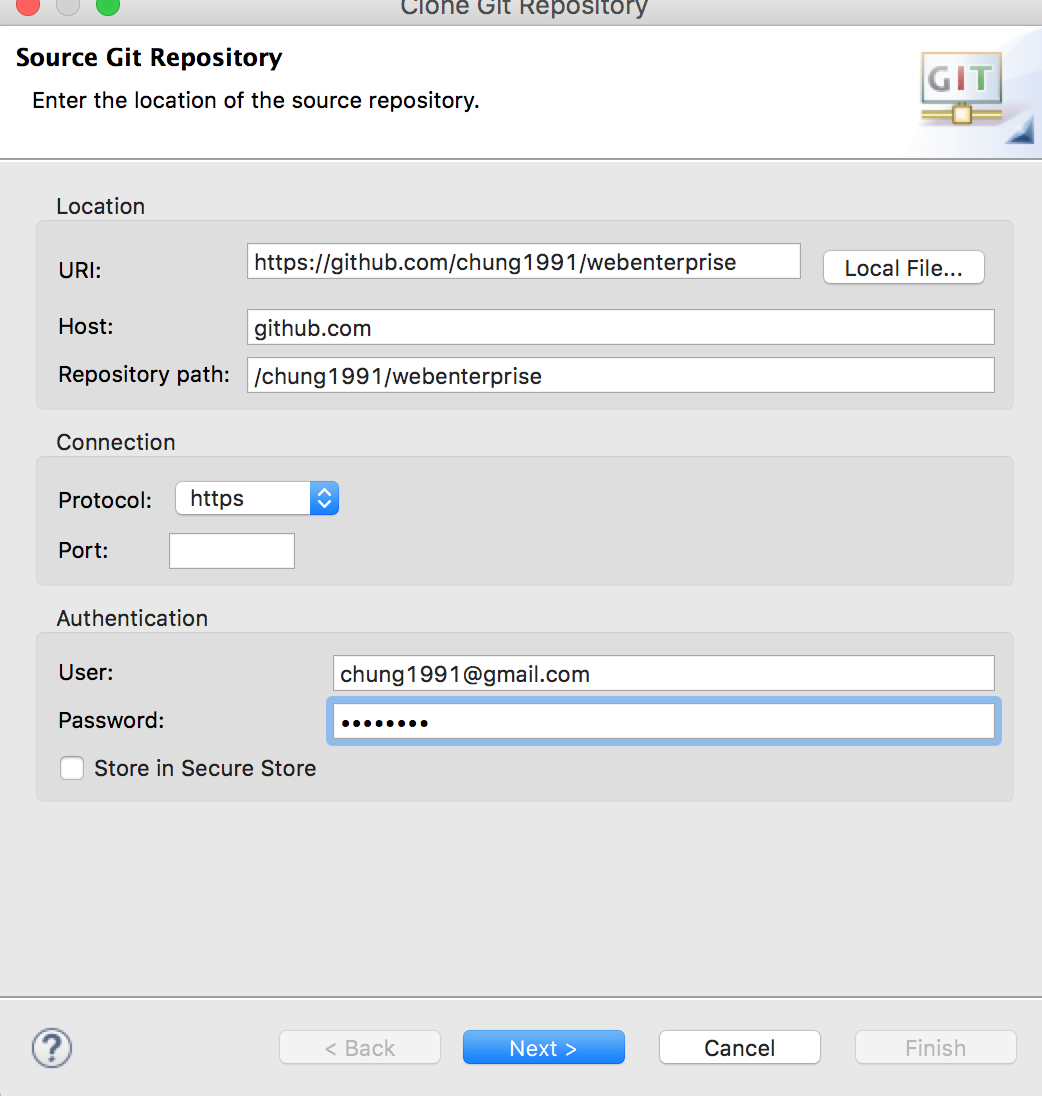
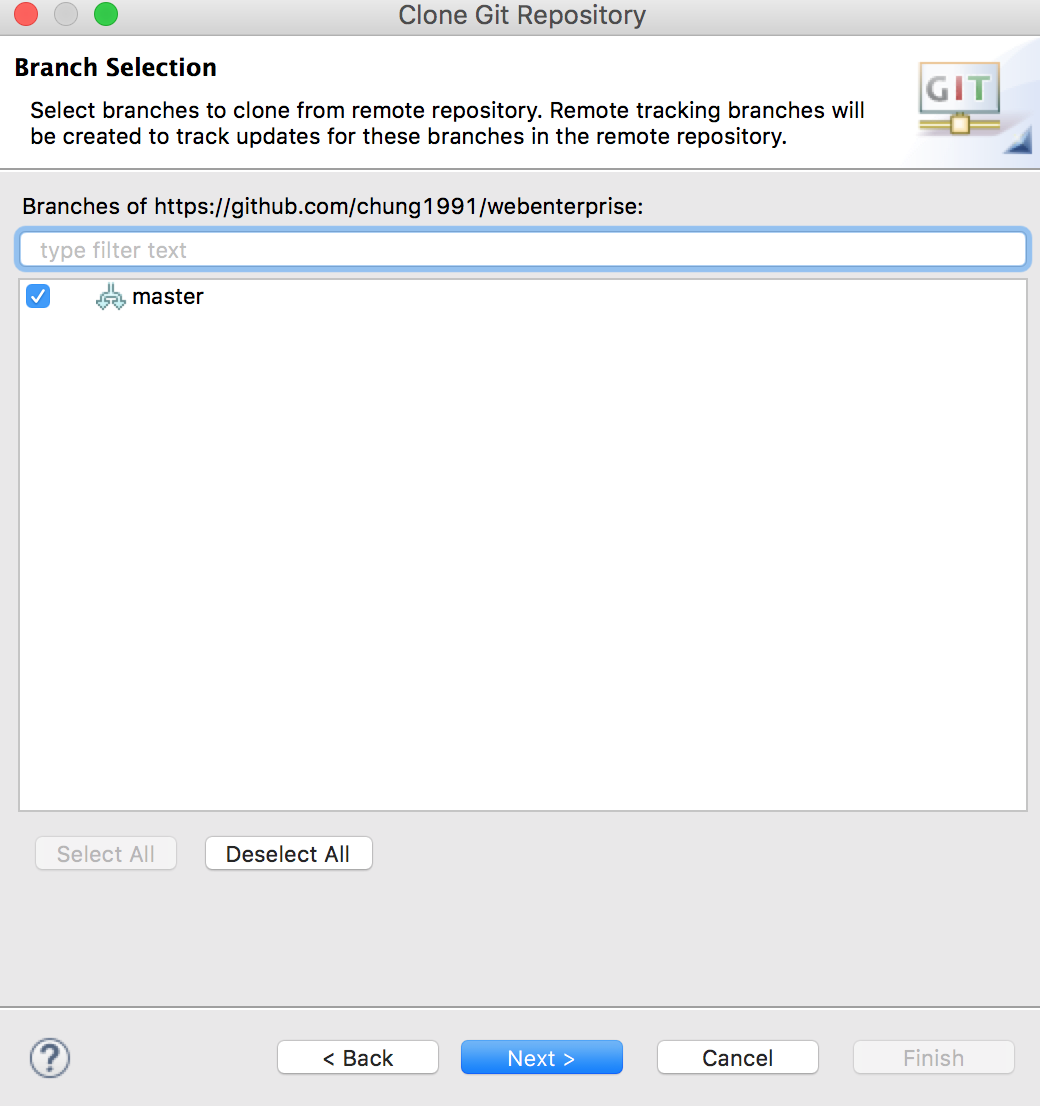
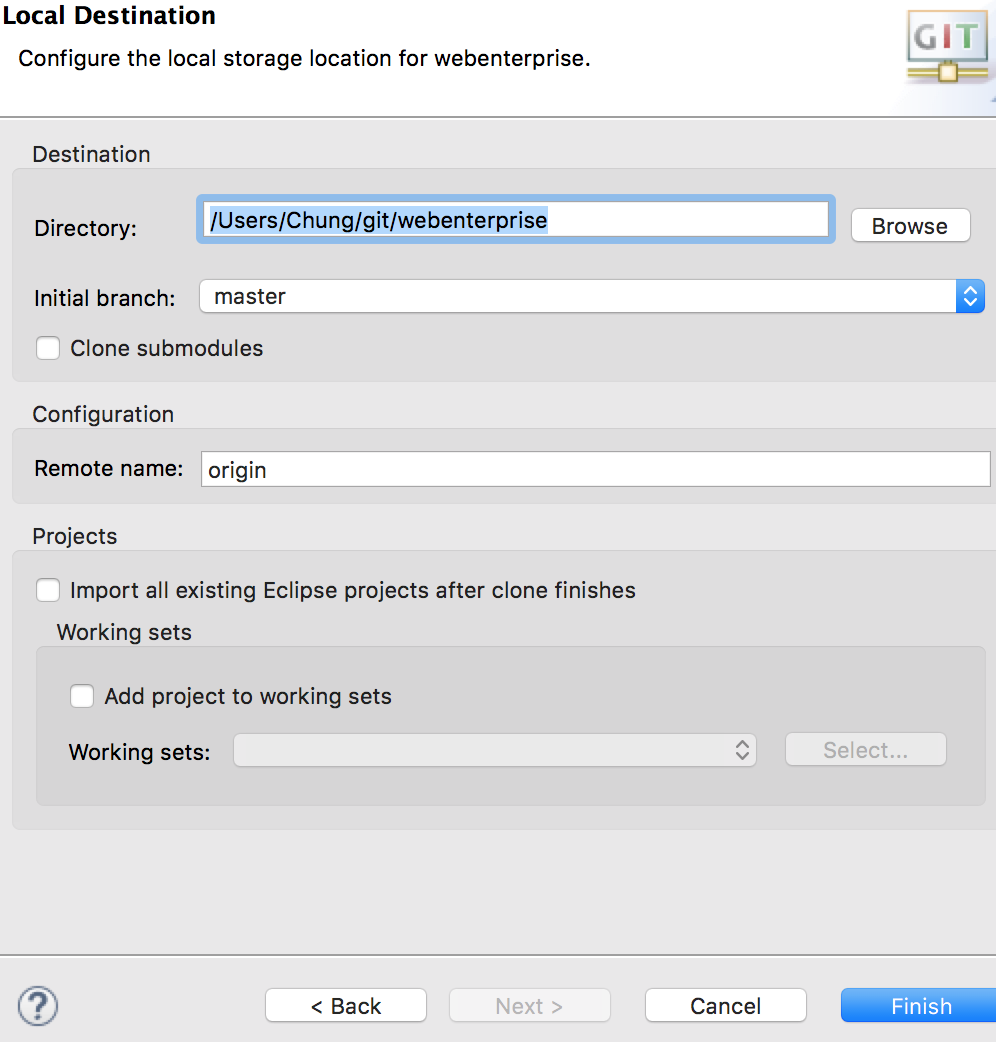
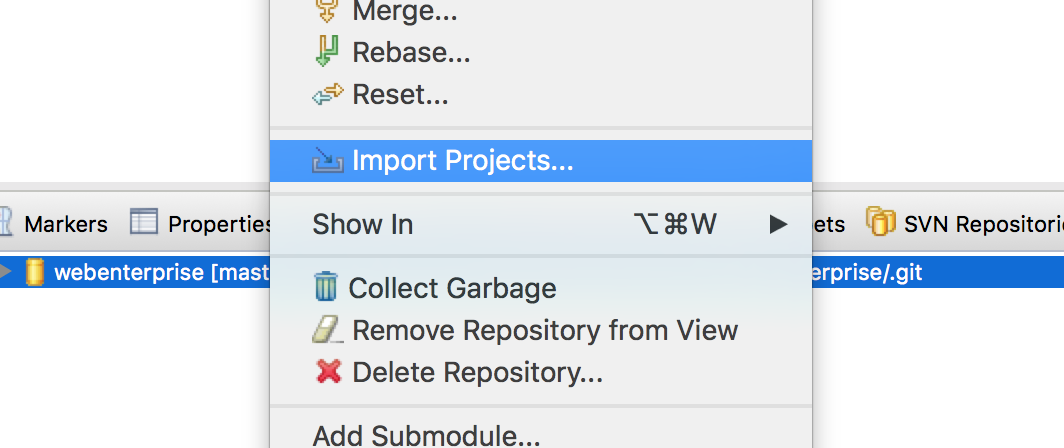
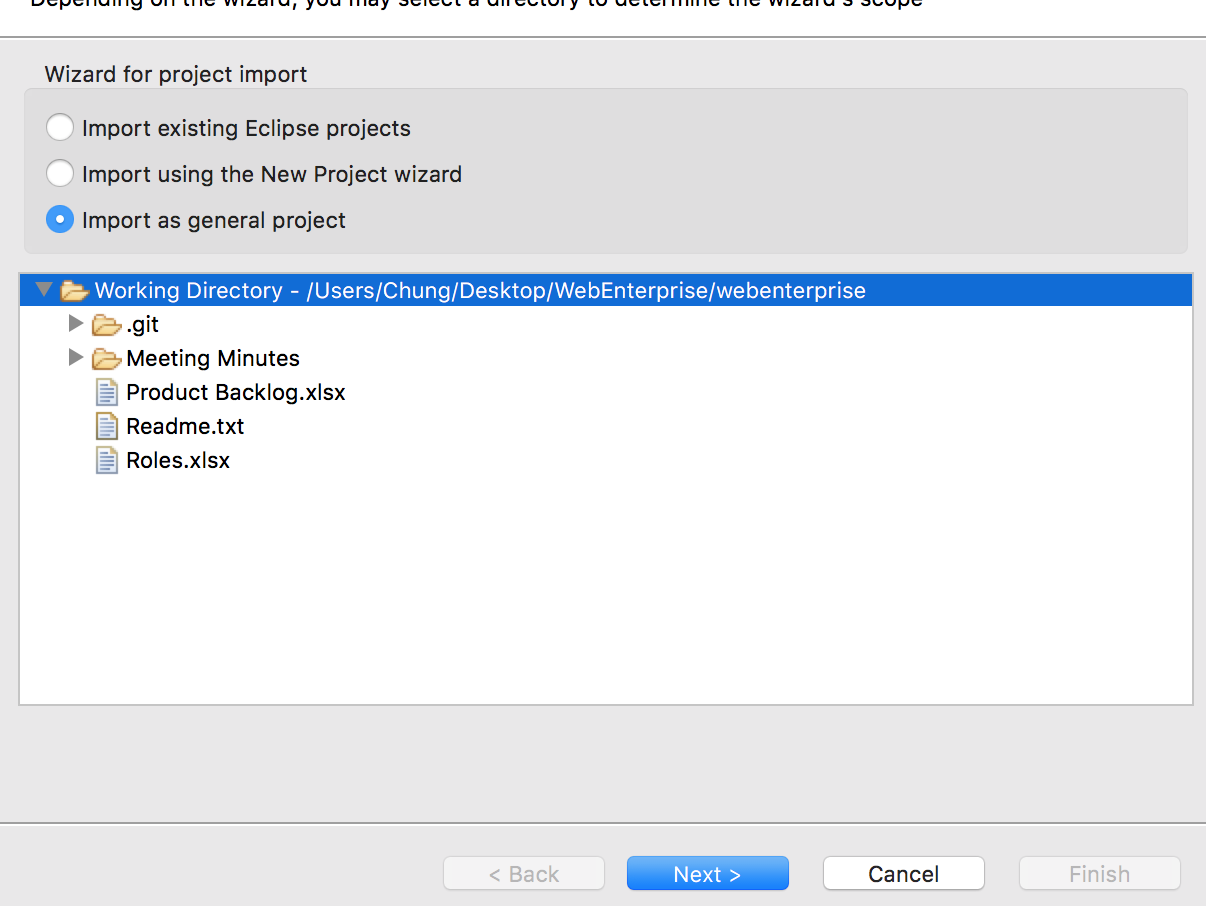
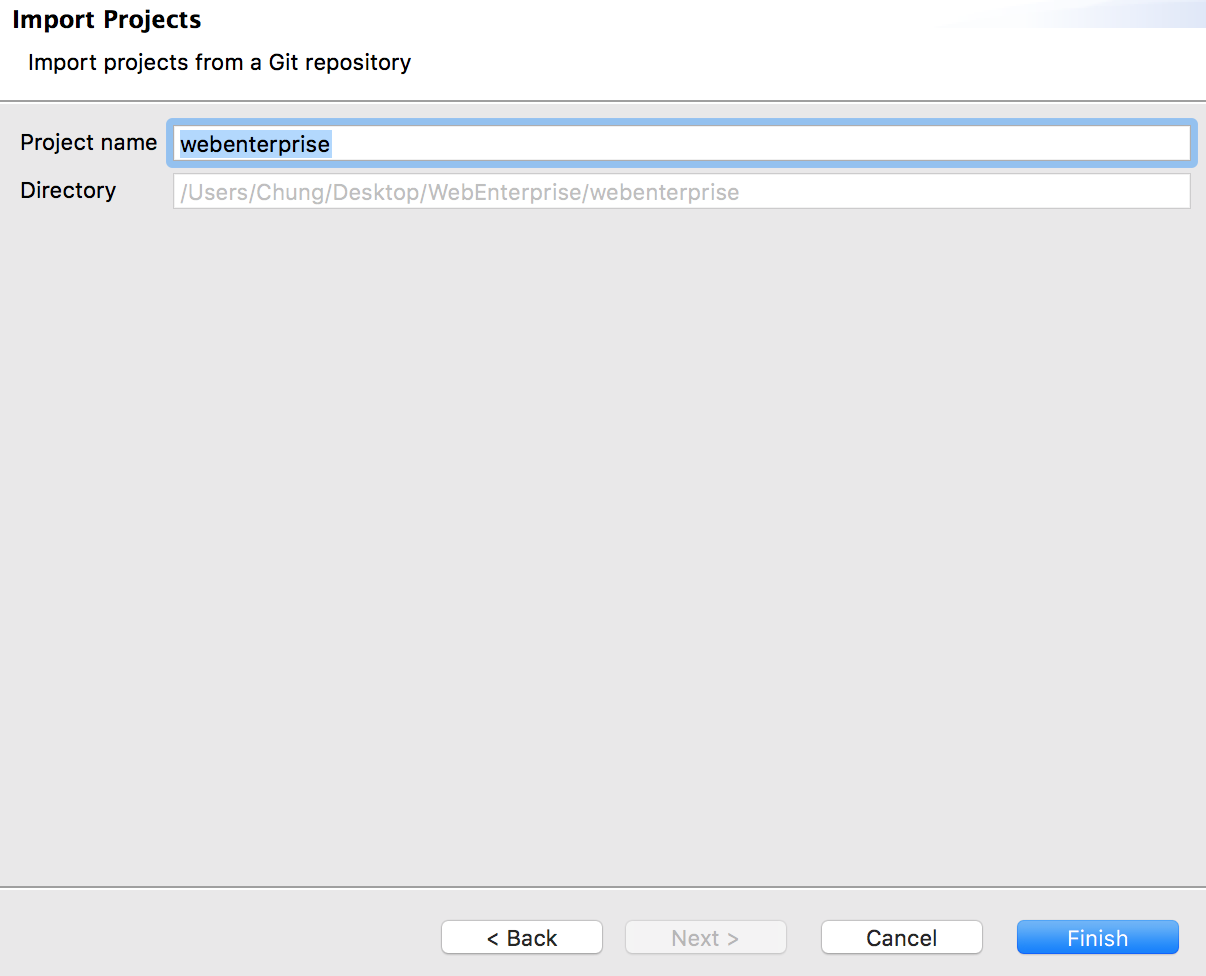
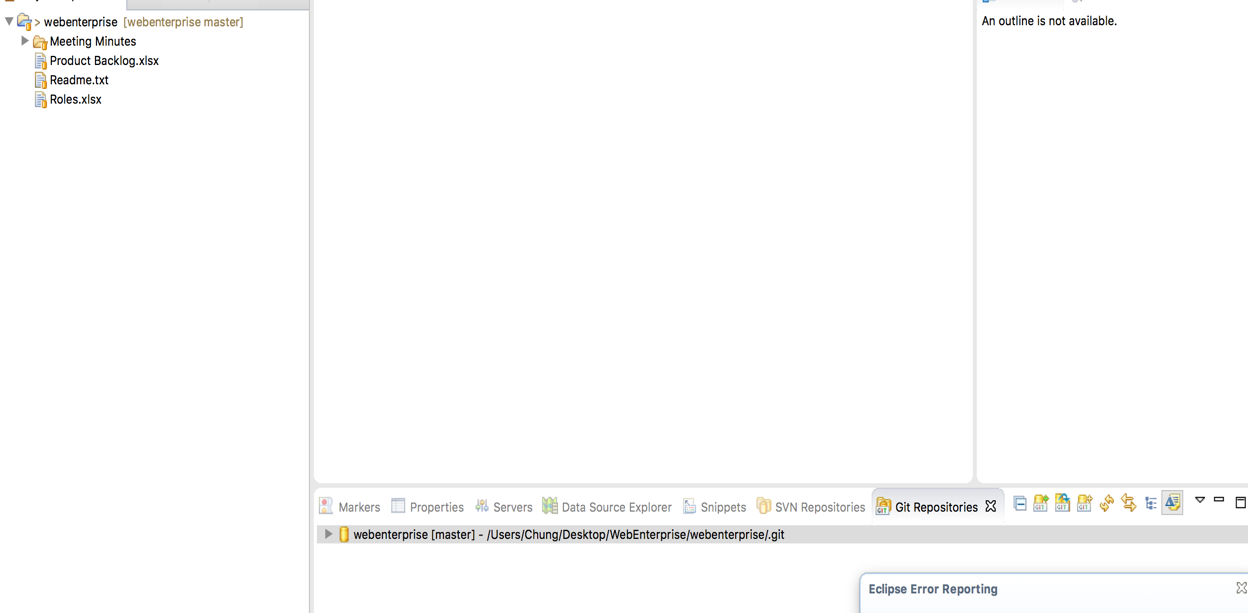
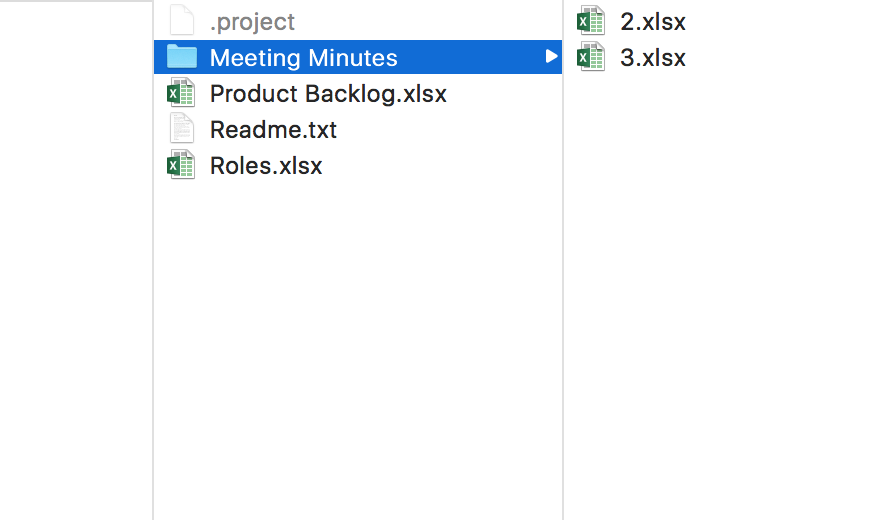
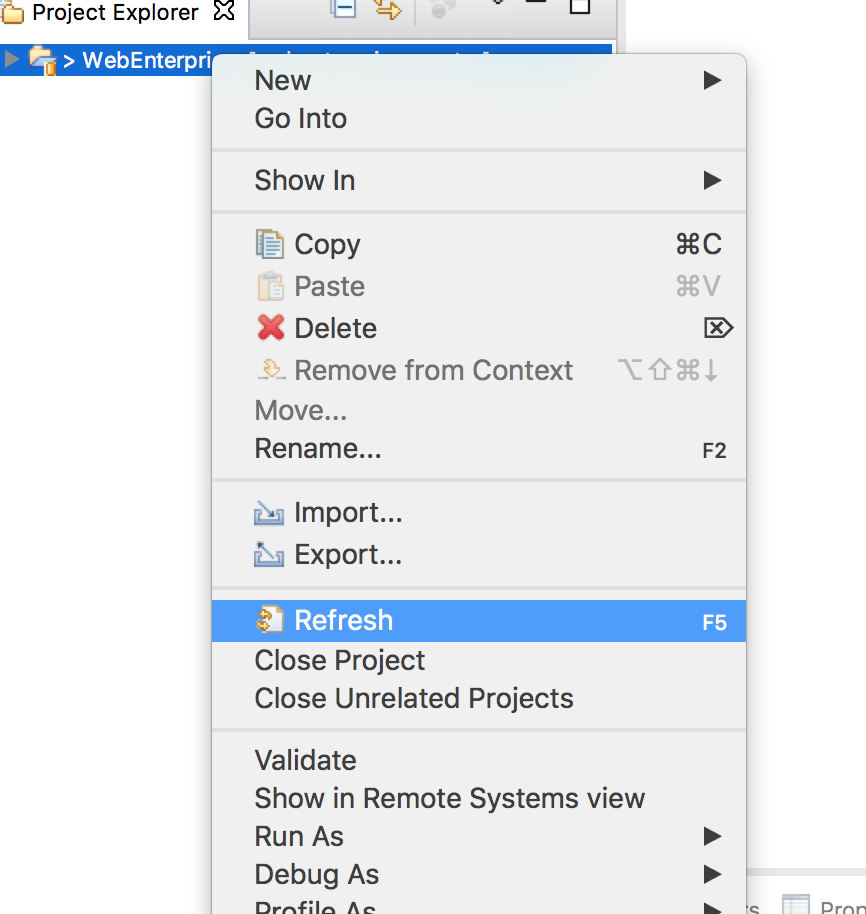
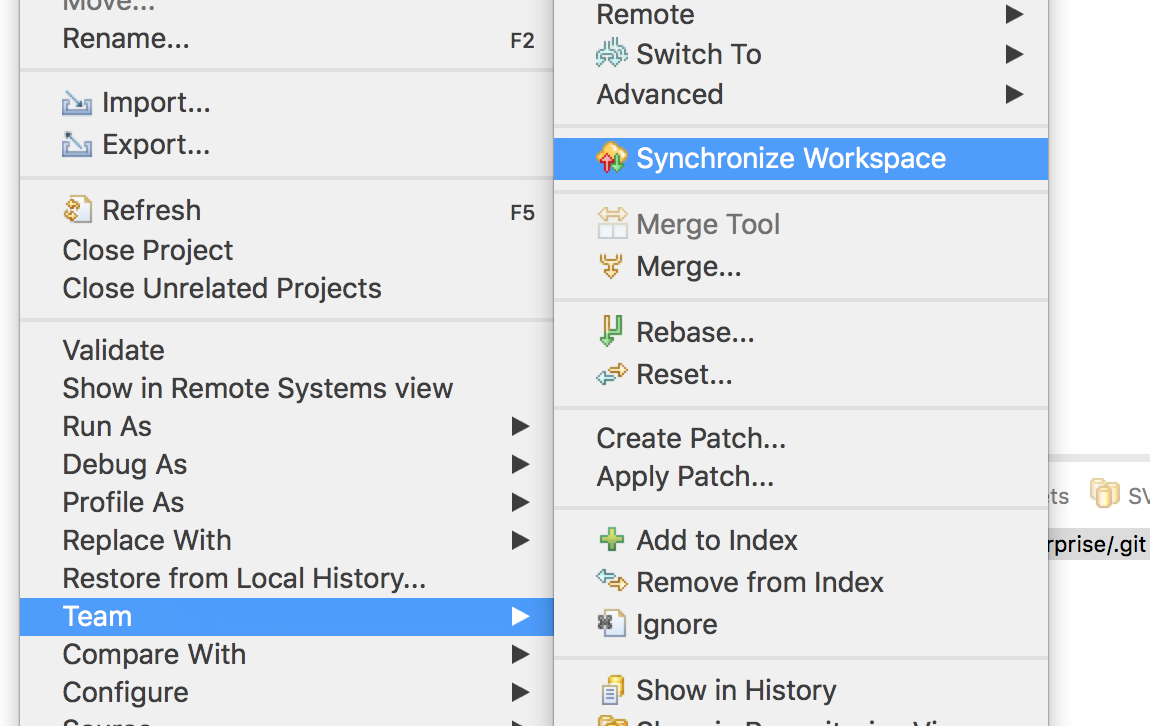
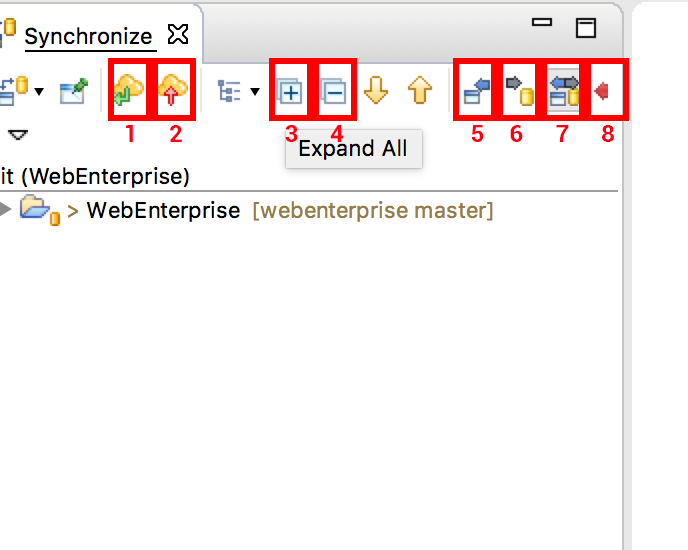
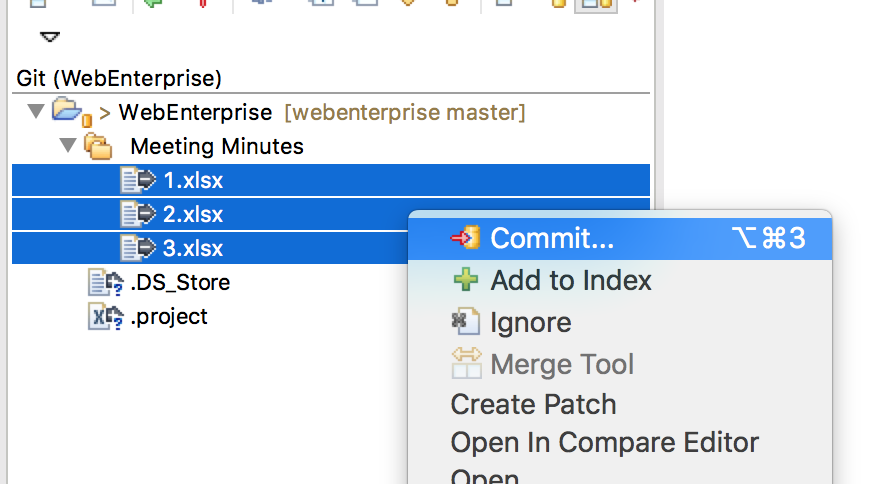
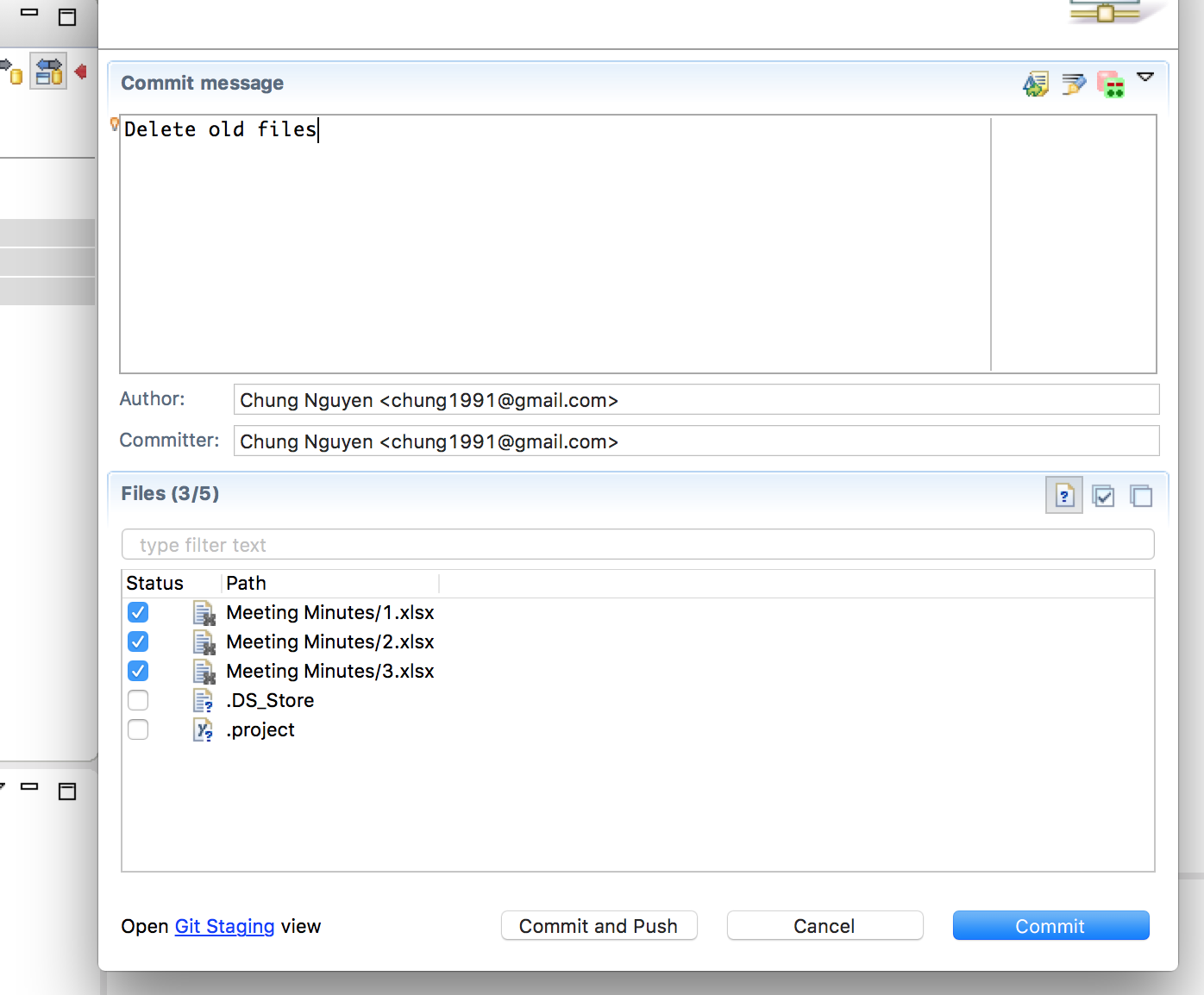
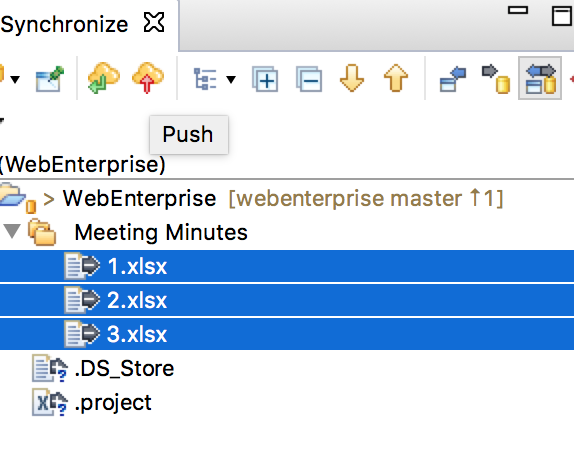
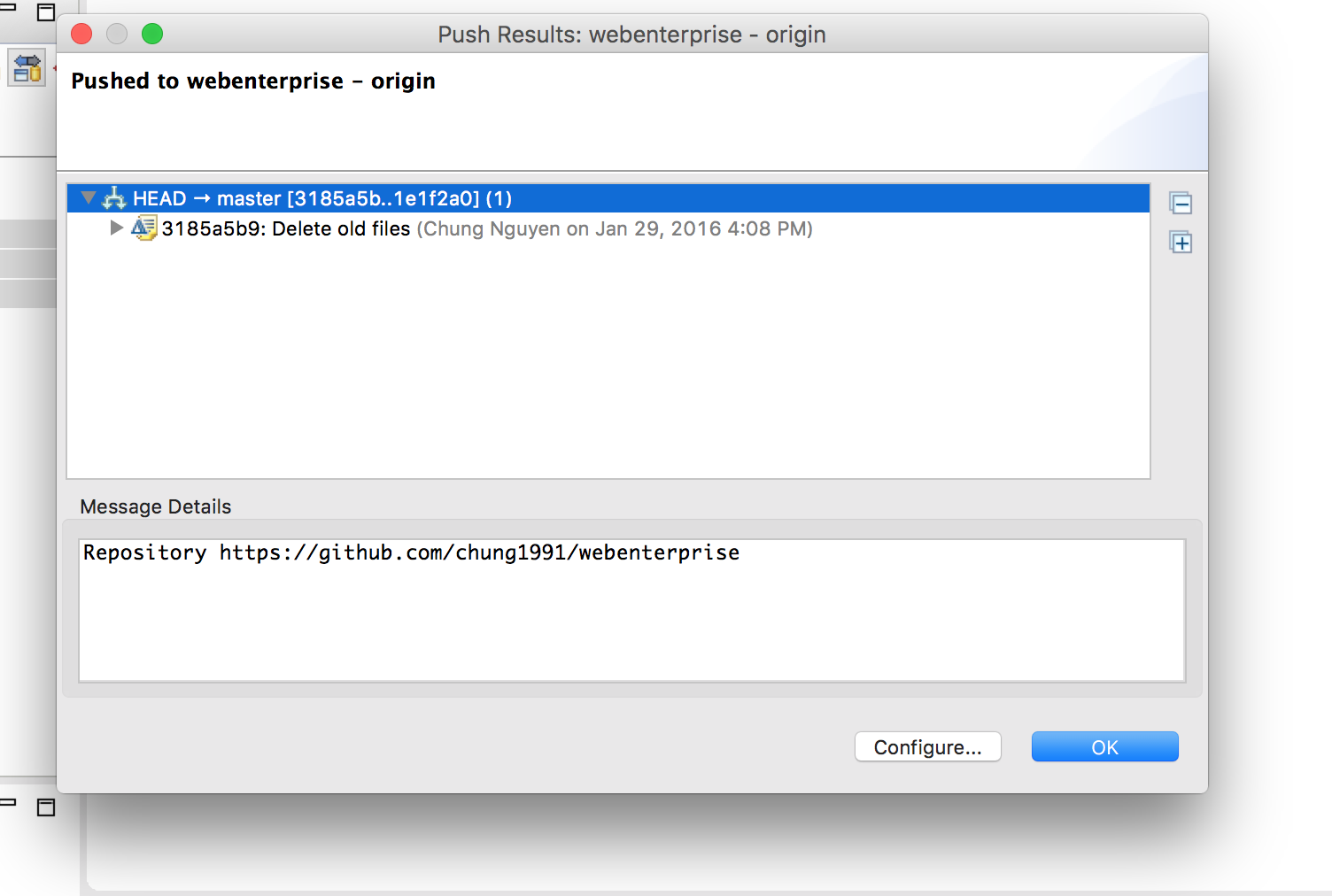
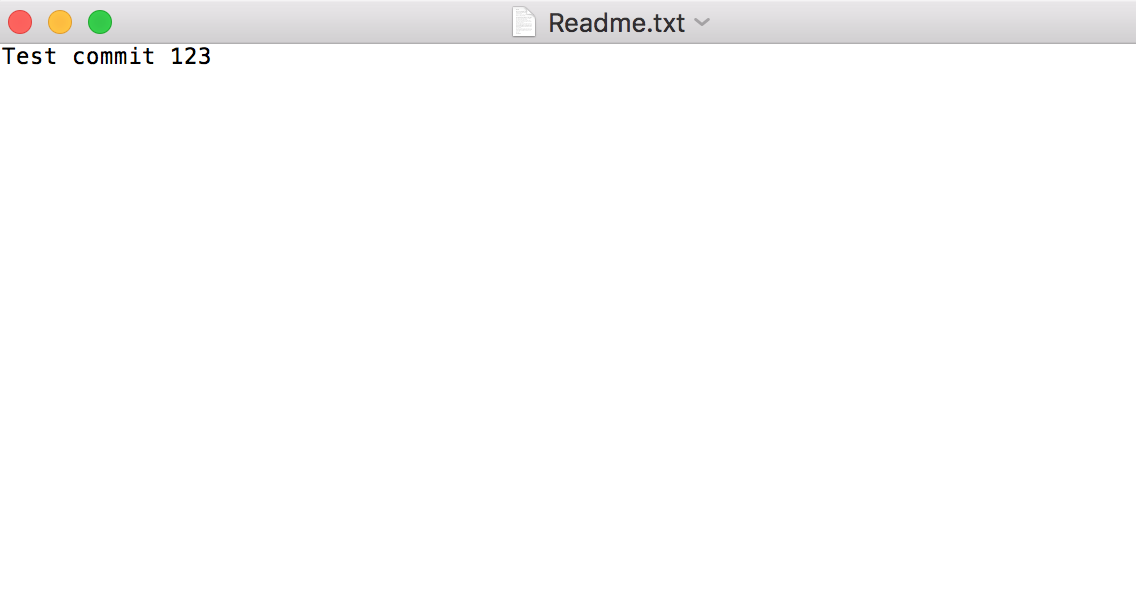
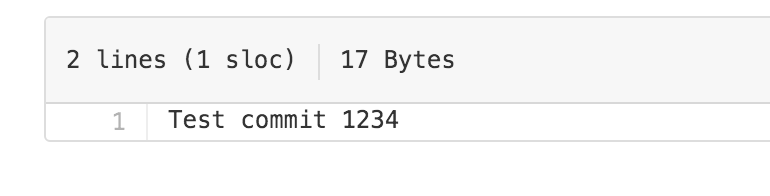
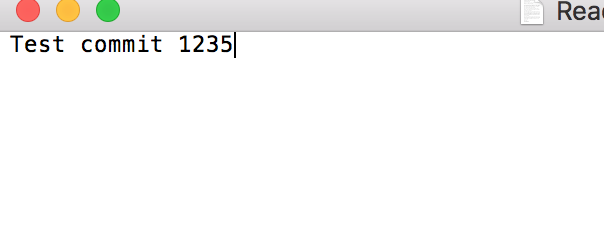
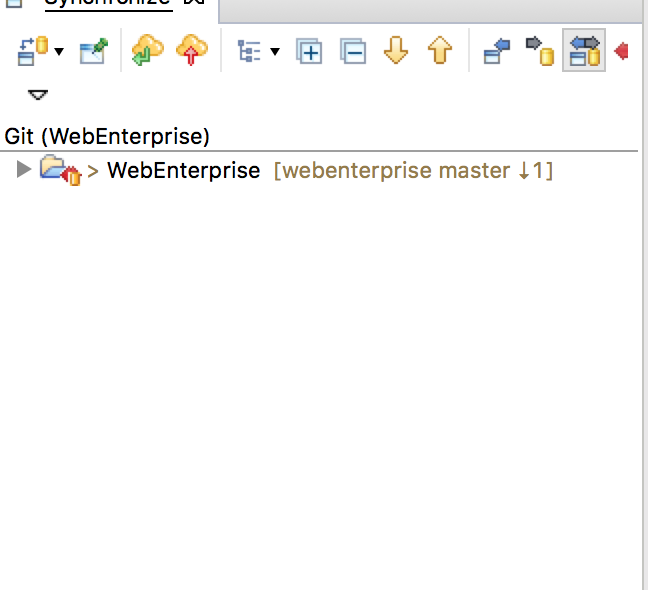
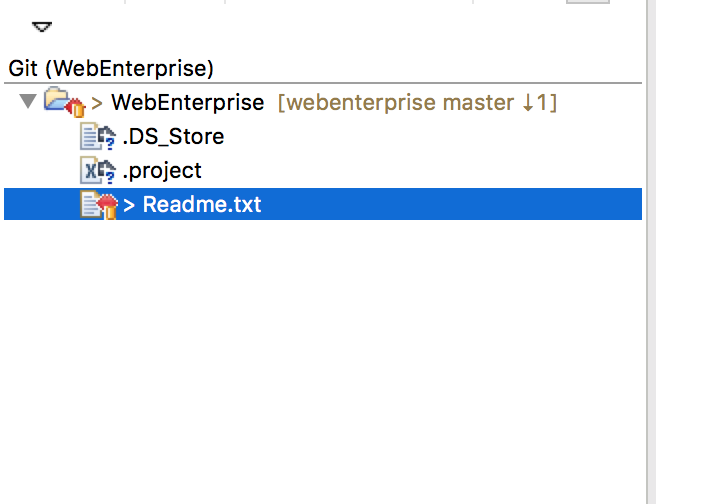
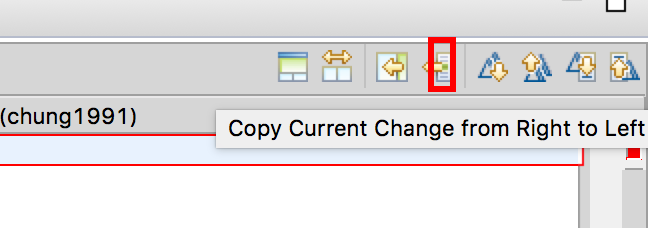
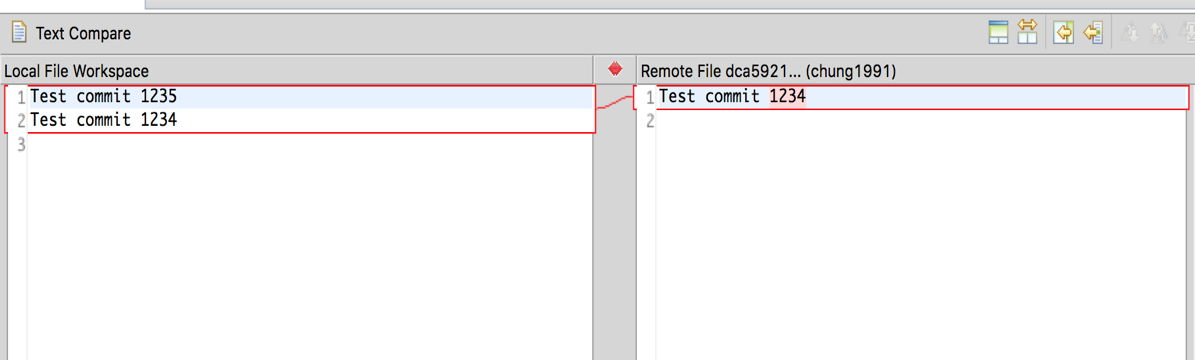
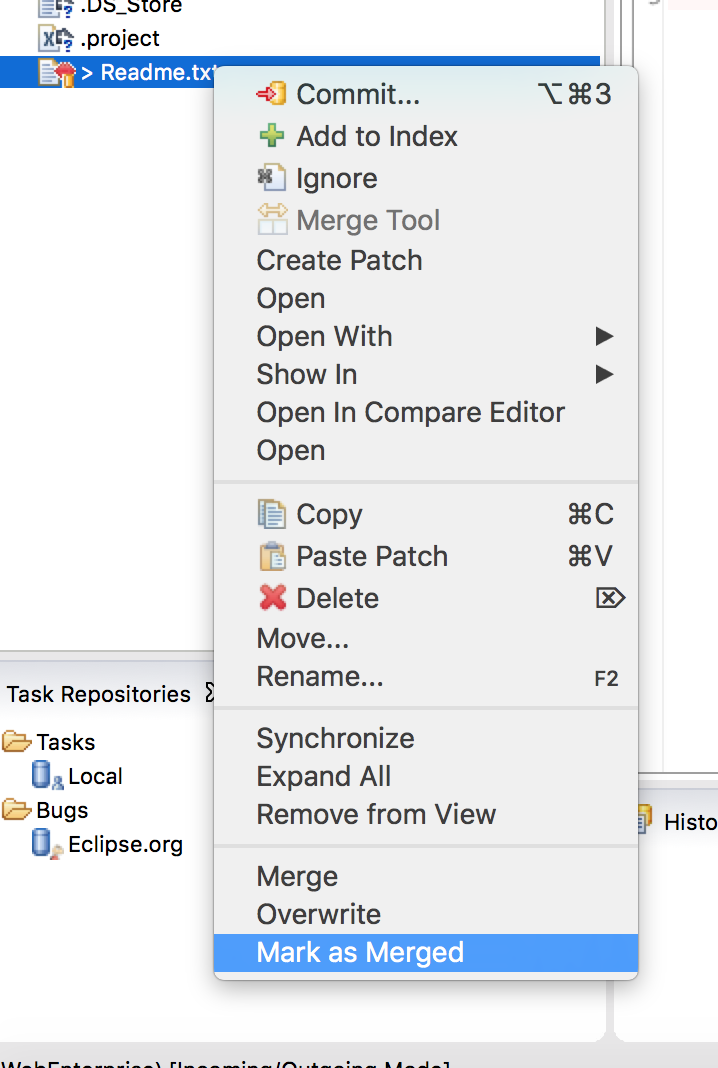
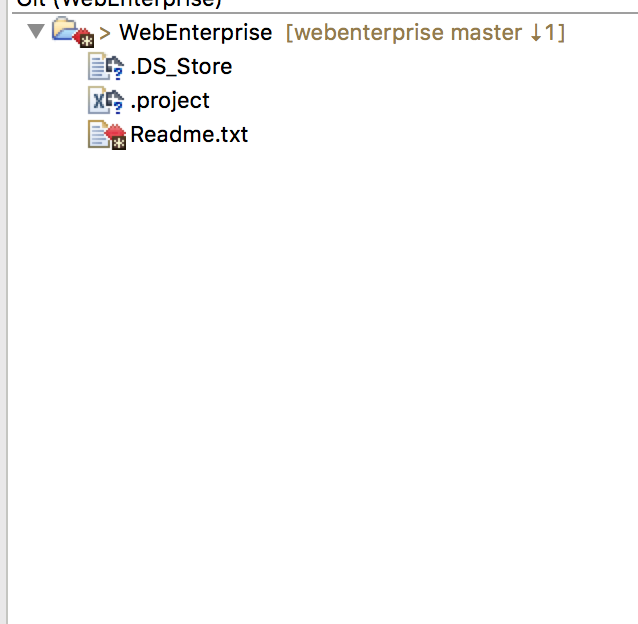
1. **DOWNLOAD ECLIPSE**
   1. <https://eclipse.org/downloads/>



* 1. Choose version which compatible with your OS
  2. When download finished, extract and click to “**eclipse.exe**”
  3. You must install the latest JDK if you don’t want to be prompted this dialog

1. **SETUP GIT REPOSITORY**
   1. Click on **Window** in menu bar then choose **/Show Views/Other**
   2. Typing “**git**” on filter textbox to find “**Git Repositories**”
   3. Now, look down to bottom window, you will see picture like below. Click on “**Clone a Git repository**”
   4. Paste your repository URL like below 
   5. Next
   6. You will see the popup has been filled a default local path. You also possibly change the path by click “**Browse**” button, after all, click on “**Finish**” button
   7. Now look at bottom window, right click on your repository, and import your project from path you saved project in previous step
   8. Tick **“Import as General project”,** then Next****
   9. Finish
   10. You have import your repository as a project successfully
2. **PERFORM GIT NORMAL CASE**
   1. Try to delete something, in below example, I want to delete “**Meeting Minutes**” folder
   2. Keep in mind that in anytime you don’t make changes files by Eclipse tool, you must refresh it before further continue working, like this situation, I must refresh project
   3. Right click again and choose “**Team/Synchronize Workspace**”  
        
      **Synchronize Action**: This action will hit and get all modified files from remote. This will provide comparing-hierarchy between old revision and new revision of each file (modified) to check which code is better to get. This action actually doesn’t merge automatically as well as cause conflict.   
        
      
   4. From here, we need to understand some signs:  
      1) Pull action: To get latest code  
      2) Push action: To send merging request  
      3) Expand all: Expand to see all files which have been modified  
      4) Collapse all: Collapse to hide all files which have been modified  
      5) See all changes from remote: To see all files which be deleted, modified, added from remote.   
      6) See all changes from local: To see all files which changes from local, they can be deleted, modified, added.  
      7) See both all changes from remote and local (and conflict): See both 5,6,7  
      8) See all conflict between remote and local: To see files from remote which conflict to files from local  
        
      For now click (3) plus sign to see all files from (7) filter.
   5. Select 3 files which marked with black right arrow (with minus inside), and right click and click “**Commit**”
   6. Enter description which describe what you make changes, do notice that all files, which be committed, must ticked/checked in “**Files**” window. Then hit “**Commit**” button. You completely can do 2 actions (commit & push) by click “**Commit and Push**” button
   7. If you just only commit files and you need an action more to made changes to remote. Now you click push button (which described above, it visualizes as yellow cloud with red up-arrow in center itself).   
        
      Notice: In this step you don’t need to select files was changed because this action will push all your current commits all to remote.
   8. Review then click “**OK**” button, so you are now familiar with commit and push something.
3. **PERFORM GIT CONFLICT CASE**
   1. Try another example to demonstrate situation which can raise confliction. In this situation. The local file “Readme.txt” looks like below
   2. But in the remote, this file was changed into
   3. There is no problem if I pull this file to local. My local file’s content will be changed into “**Test commit 1234**” without problem. But now I try to change it into “**Test commit 1235**” before I do pull action.
   4. Back to Eclipse, don’t forget to refresh project. Right click on project and “**Team/Synchronize Workspace**” and see what’s happen. I can see the conflict indicator near project. Now try to expand the project
   5. Select file was be marked in conflicted (but actually, local file still be OK, nothing was merge or causing real conflicted) and double click on it.
   6. You can see the Compare Splitting Window is shown. In the left splitting window, you can see the changes are made in local, next is in remote. So now you must determine which code you need to bring on the local.  
      1) If you determine “123**5**” is right (**5** represent as your code changes) you should do nothing.   
      2) If you determine “**4**” is right, you can click “Copy current change from right to left”  
        
      Now left splitting window will look like  
        
      Now you just delete the code “1235” from your left and retain the code “1234” (In case both “1234” and “1235” all right. You do not need to delete code “1235”)
   7. Now, your editing was done, it time to “Mark it merged”. This action will tell that “I merged code which has content “1234” from remote Readme.txt to my own local Readme.txt, in further synchronizing, don’t compare my own local Readme.txt to that revision anymore”. Right click on “Readme.txt” and click “Mark as merged”  
       
   8. You can see the indicator near Readme.txt had a little changes, tells you this file was marked as MERGED
   9. Finally, you just commit and push it to remote.