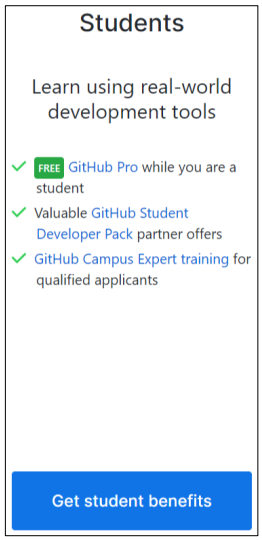
GitHub Setup Guide

Contents

Read up

1. Sign up for an account @ https://education.github.com/pack
   Do remember to use the RP student email account.



2. Some initial read up on GitHub to understand the various operations that can be done using it
   https://guides.github.com/activities/hello-world/
   http://rogerdudler.github.io/git-guide/
   http://www.vogella.com/tutorials/Git/article.html
   https://code.tutsplus.com/tutorials/working-with-git-in-android-studio--cms-30514

Git installation

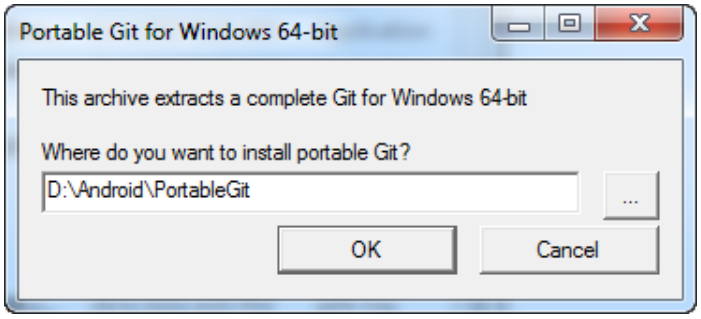
Step 0 - Installation file preparation

Copy the file PortableGit-2.26.0-64-bit.7z.exe from the class lecturer.

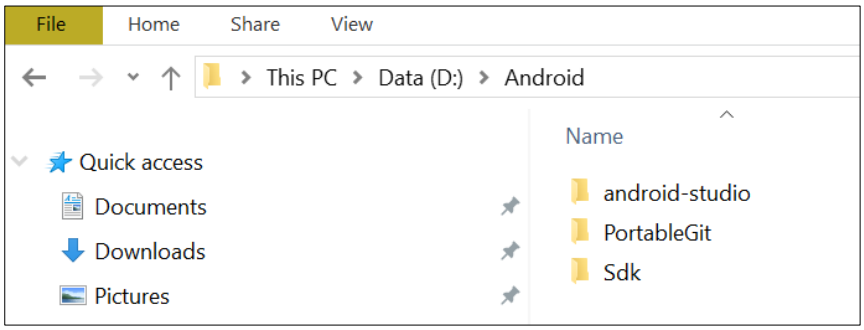
Alternatively, you can download it at https://github.com/git-for-windows/git/releases/download/v2.26.0.windows.1/PortableGit-2.26.0-64-bit.7z.exe

Step 1 - Installing Git

Run the installation file from Step 0

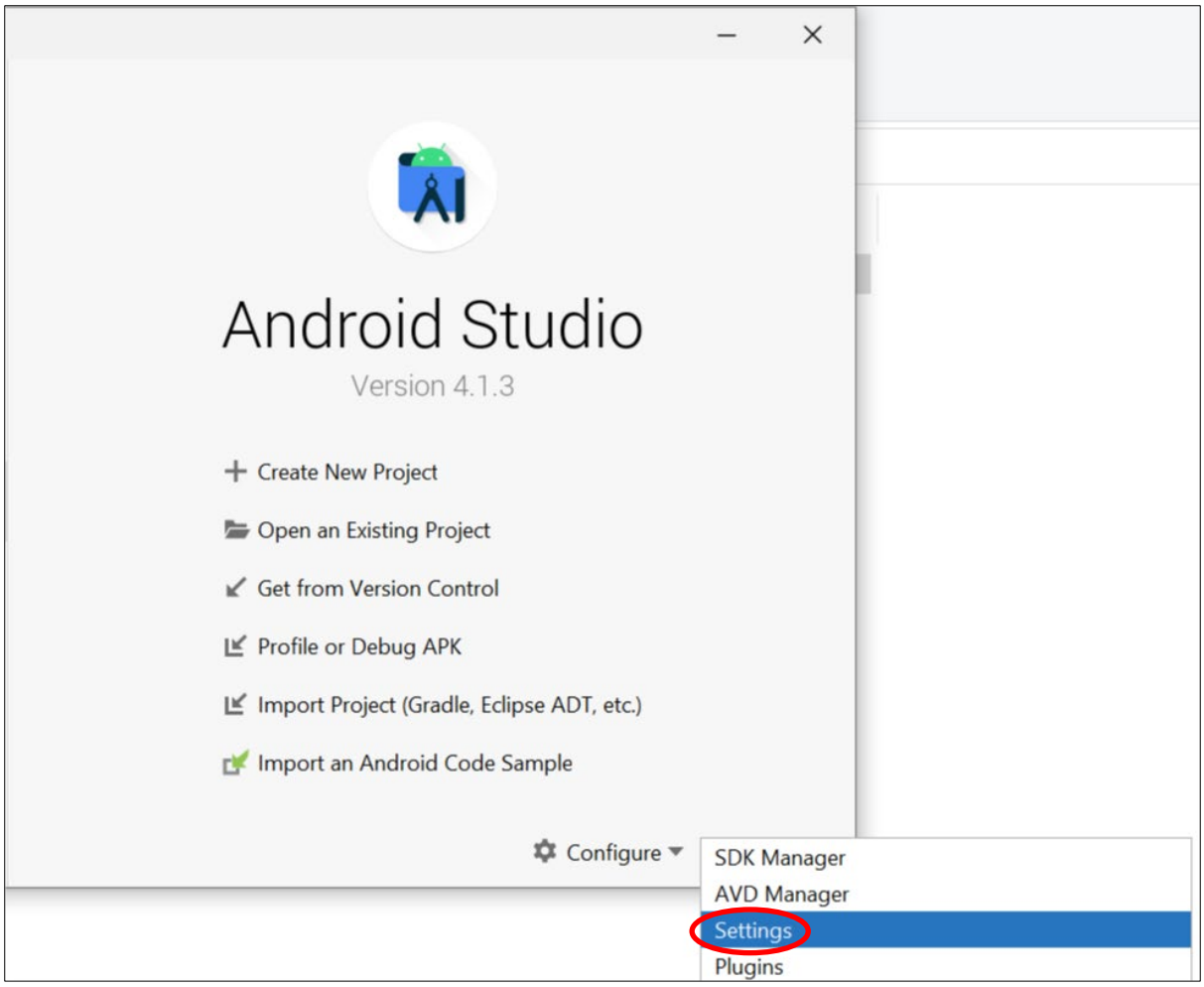


Install Git at the folder created for Android IDE installation. For example, D:\Android\PortableGit. Remember to create the "PortableGit" folder to install to.

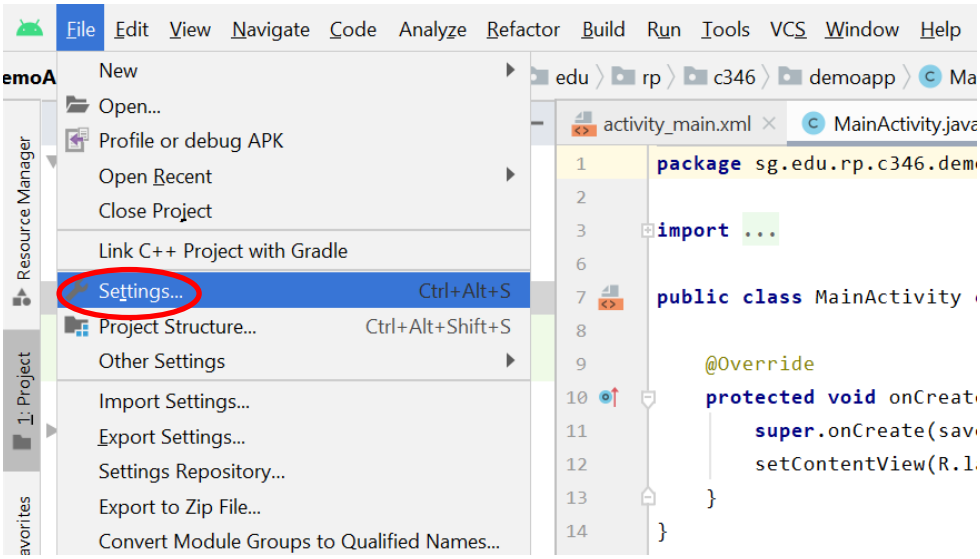


Step 2 - Setting up Git in Android Studio

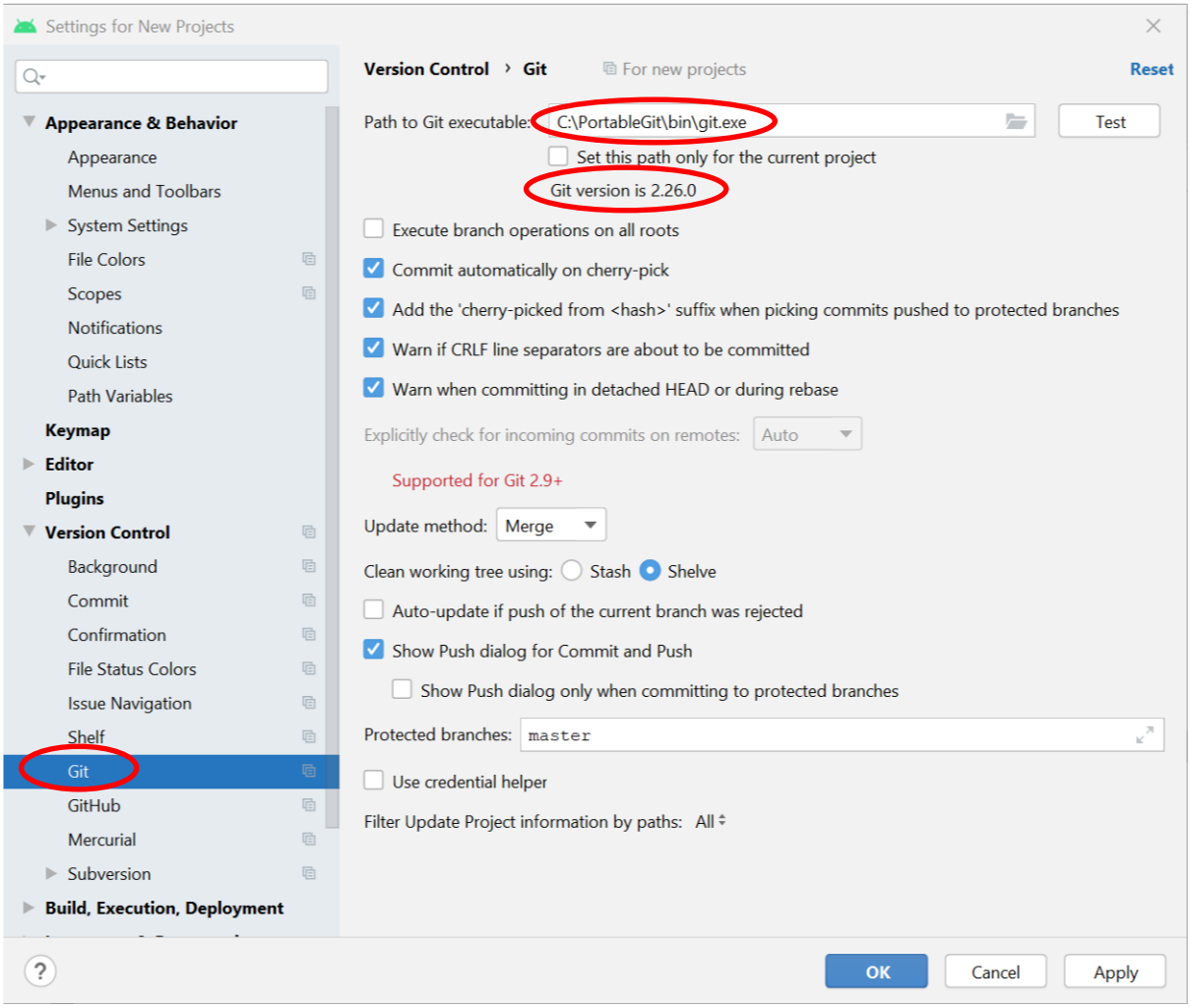
1. Go to the **Settings** page. It can be accessed from the main starting screen shown below



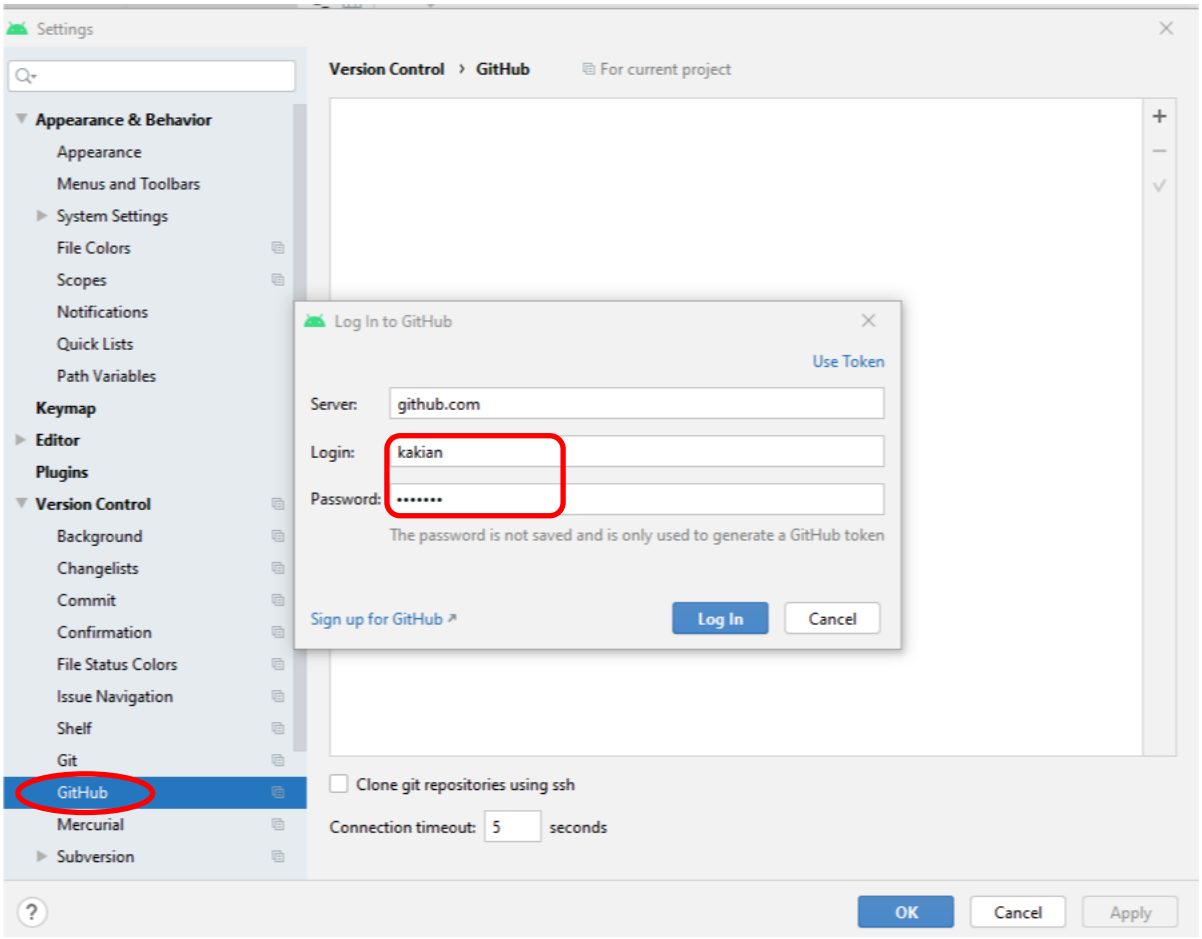
Alternatively, the **Settings** page can be accessed from here too!

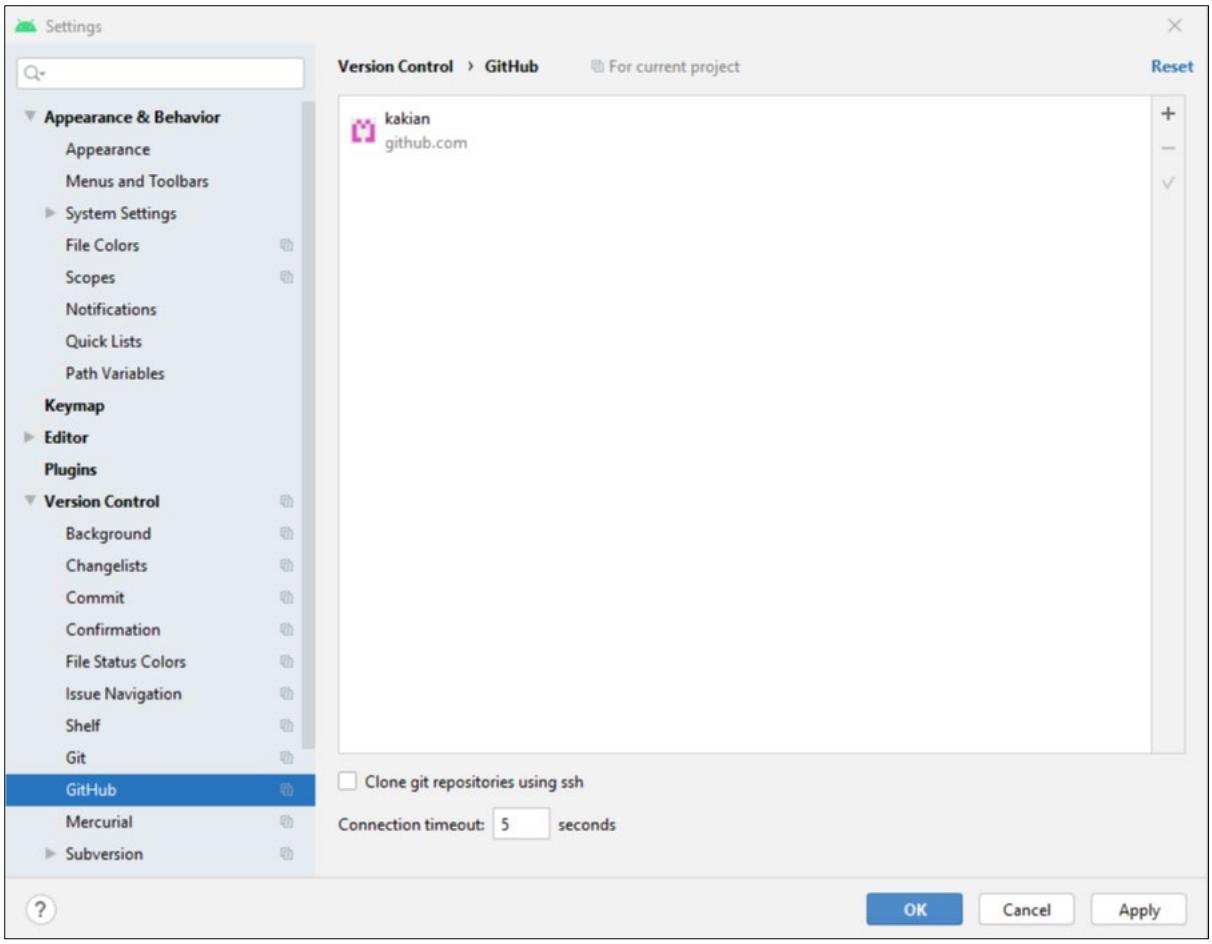


2. Enter the path to the **git.exe** from the Git installation at Step 1. Click the “Test” button and you should see “Git version is 2.26.0” displayed below.



3. Next, in the GitHub section, click the "Add account" link in the main panel to enter the Github account credentials.

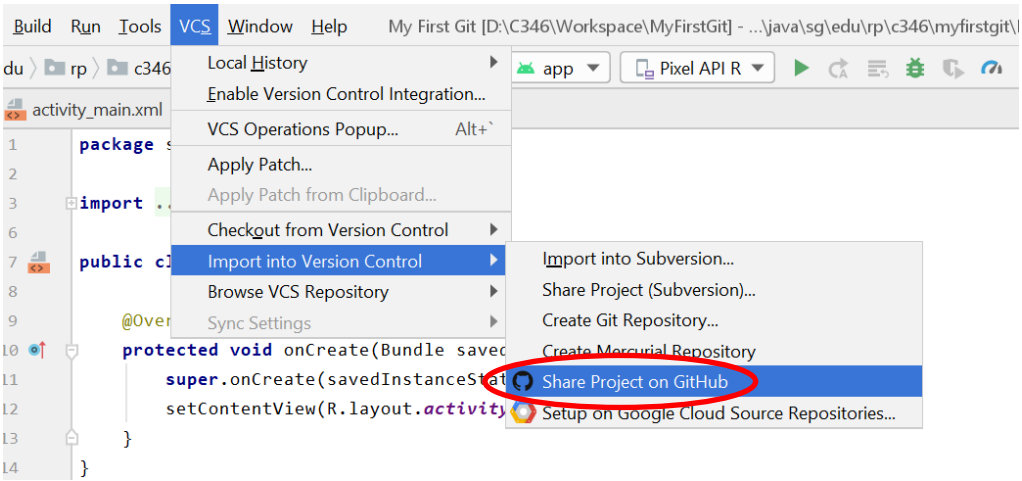

Settings
Version Control › GitHub
For current project
Appearance & Behavior
Appearance
Menus and Toolbars
System Settings
File Colors
Scopes
Notifications
Quick Lists
Path Variables
Keymap
Editor
Plugins
Version Control
Background
Changelists
Commit
Confirmation
File Status Colors
Issue Navigation
Shelf
Git
GitHub
Mercurial
Subversion
Log In to GitHub
Use Token
Server: github.com
Login: kakian
Password: •••••••
The password is not saved and is only used to generate a GitHub token
Sign up for GitHub ↗
Log In
Cancel
Clone git repositories using ssh
Connection timeout: 5 seconds
OK
Cancel
Apply



Importing into Github (initial commit)

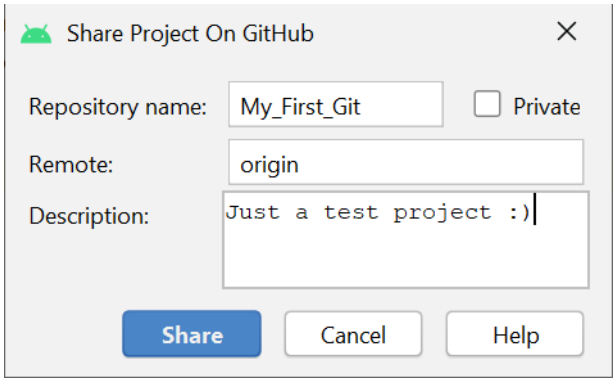
Step 1 - Create an Android project in Android Studio or choose any existing project to be imported

Step 2 - At the VCS menu, choose Import into Version Control -> Share Project on Github



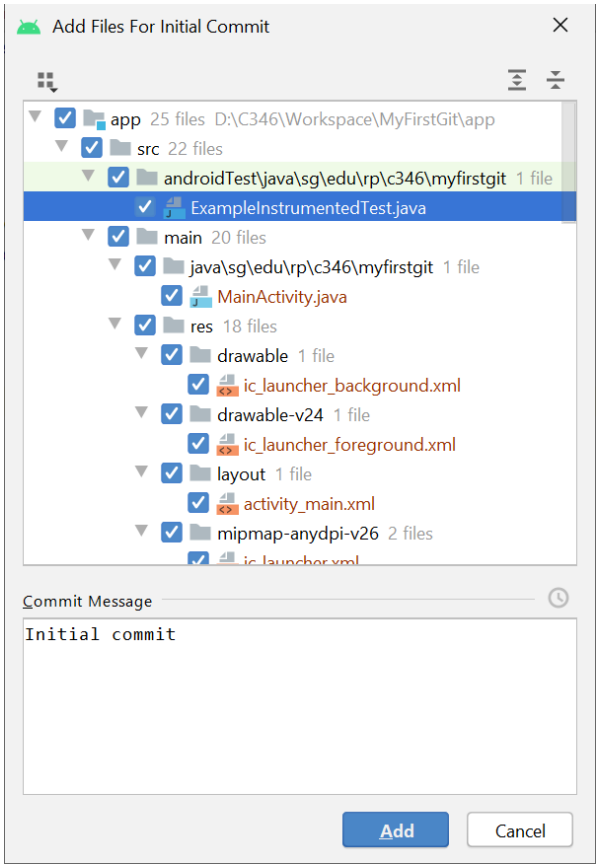
Step 3 - Project creation and upload @ Github

1. With the Github settings done prior to this, you would be seeing the below window



You can leave the remote name as the default value. As for the description, do input something useful and descriptive of your project. This could be your portfolio :)

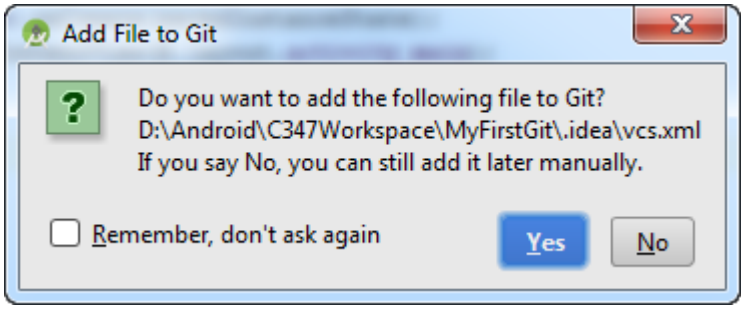
2. The following window will pop up, showing the files going to be committed to the repository



The commit message is supposed to capture the changes done for every commit. Do note that *commit* is not referring to "save" in your Android Studio. The *commit* is akin to the closing of sales of a day or whenever there's something major.

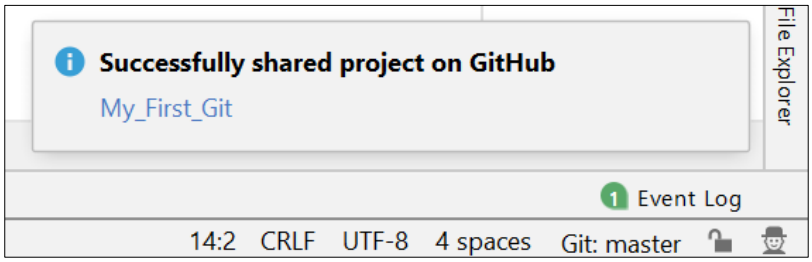
3. If you see the following pop up window, you can just click "no". Basically, we would want to commit only essential files, transient files are not required.

   Transient files here are referring to files that are generated by the IDE pertaining to user specific settings, definitely not critical to the project.

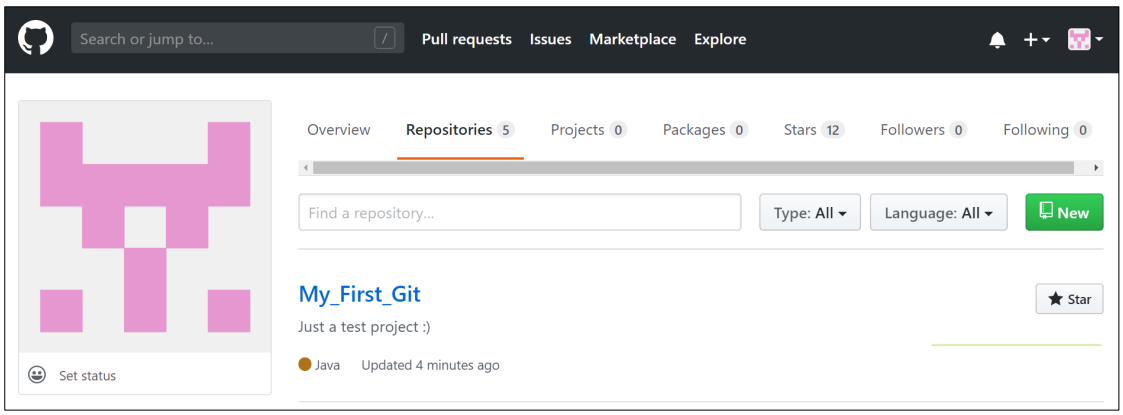


Note: To know which are not required to be committed into a repository, you can refer to this template https://github.com/github/gitignore/blob/master/Android.gitignore
You can also copy its content into .gitignore of your project to automate file exclusion

4. Upon completion, you'll see this pop up box on the lower right.



5. Go to http://github.com with your account logged on, you'll see the repository is created and the files are committed in it.



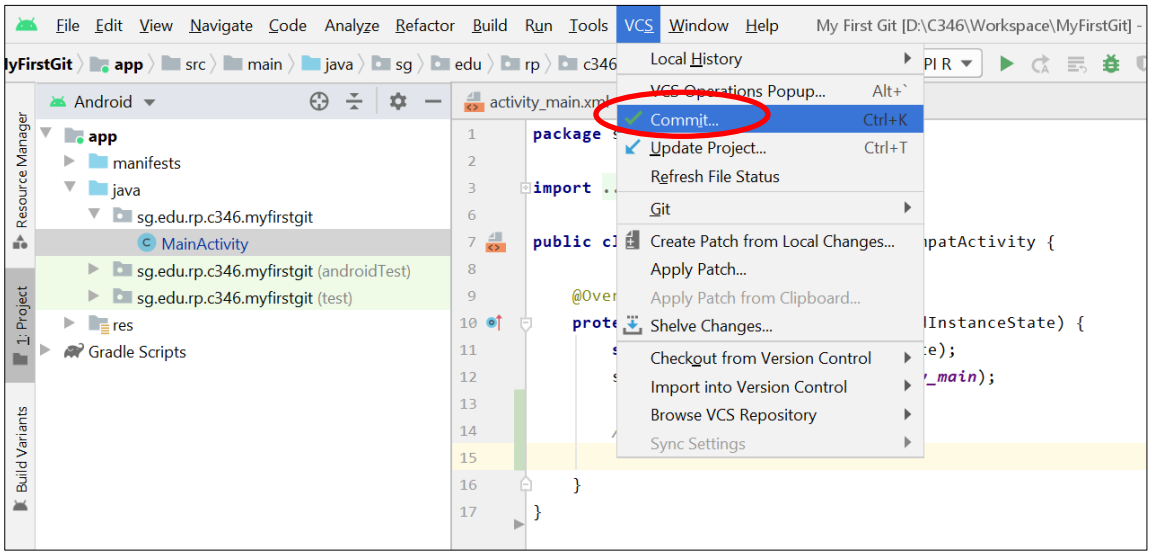
Commit changes to repository

1. Code changes

```
1  package com.myapplicationdev.android.myfirstgit;
2
3  import ...
5
6  public class MainActivity extends AppCompatActivity {
7
8      @Override
9      protected void onCreate(Bundle savedInstanceState) {
10         super.onCreate(savedInstanceState);
11         setContentView(R.layout.activity_main);
12
13         //This is a new line
14
15     }
16 }
```

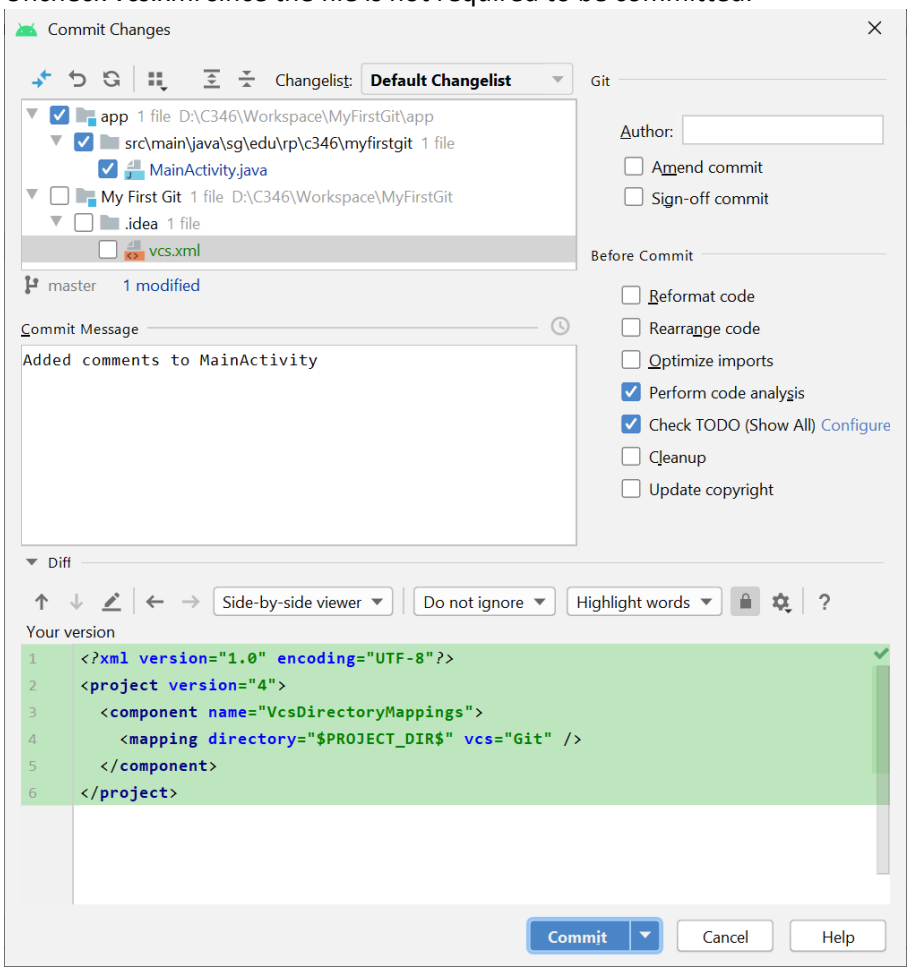
As you make changes to the project on the Android Studio, the IDE is keeping track of what you have written. The changes will be saved in your local storage as you continue on your development.

Only when you are ready to *commit*, you will need to push the changes into the repository.

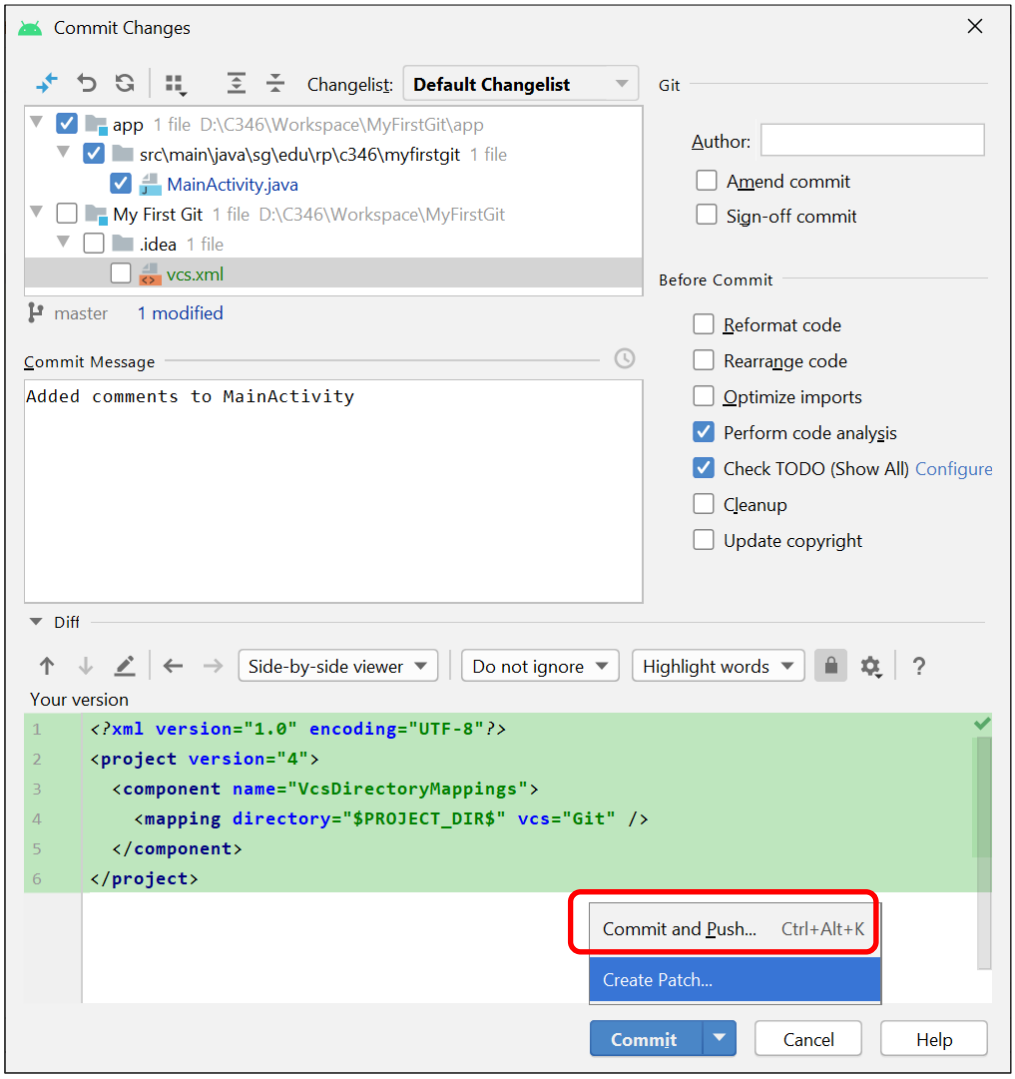


The following screen will appear. It's a necessary practice to summarize the code being committed. The idea is to capture the essence of what is being committed into the repository.

Uncheck vcs.xml since the file is not required to be committed.



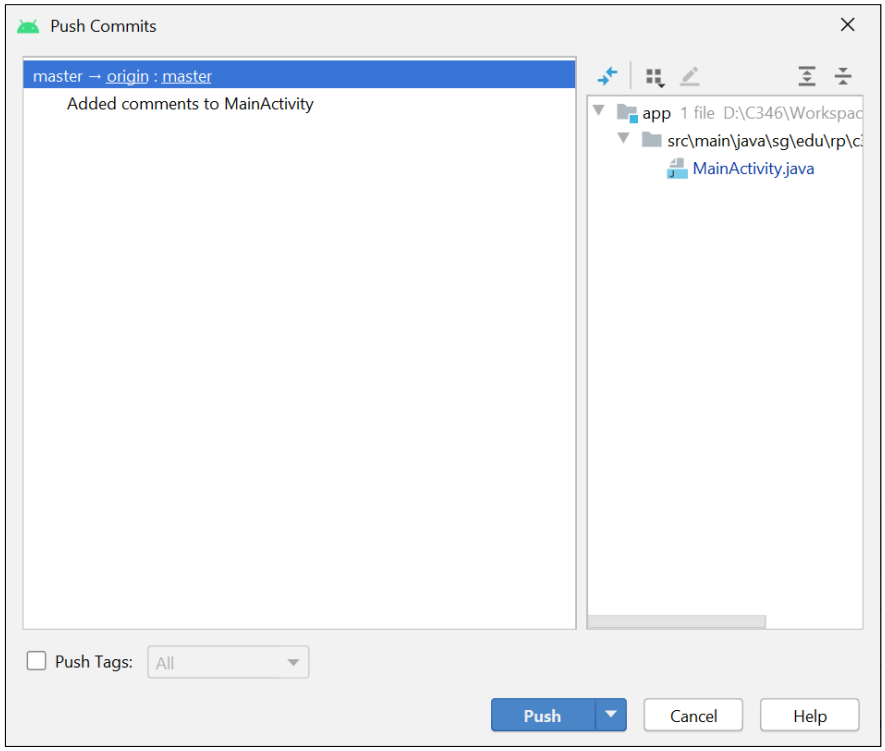
To effect the changes to the remote repository (in this case, Github itself), you need to push the changes over. Click **Commit** button and select **Commit and Push**.



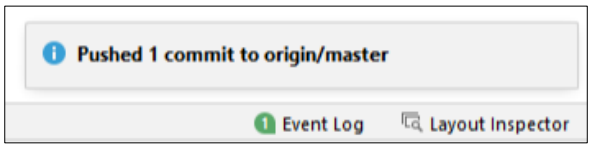
2. File adding/removal

The screen below will appear and do provide the summary for the commit as well.

Now, you'll need to push the changes over to Github. Click the **Push** button.

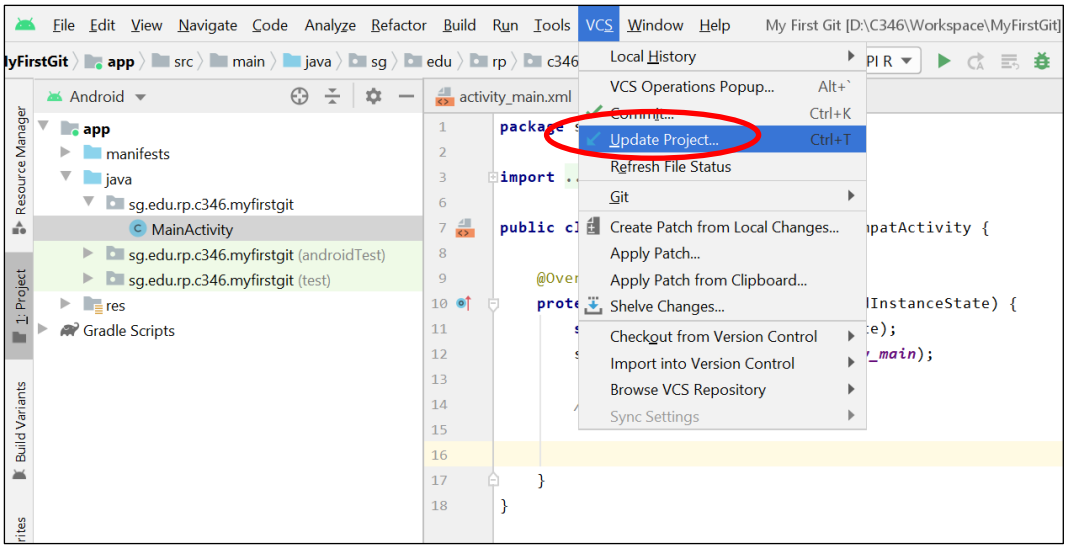


Now, your file has been uploaded successfully to your GitHub repository. Look for the event log pop-up at the bottom right corner of Android Studio.

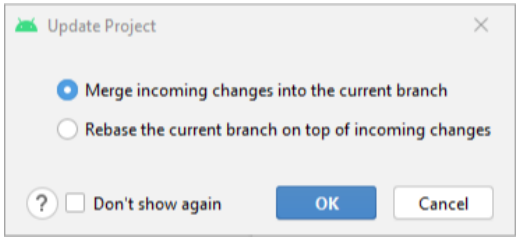


Update changes from the repository

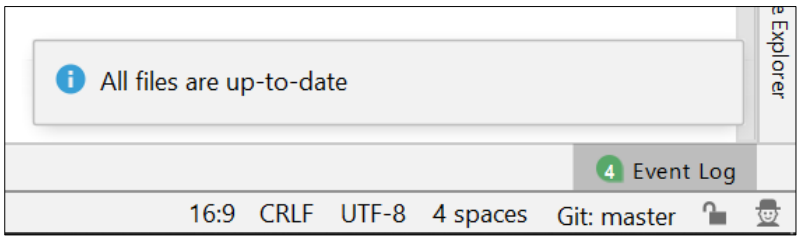
1. To update changes to the code from the repository, you can choose **Update Project** from the CVS menu



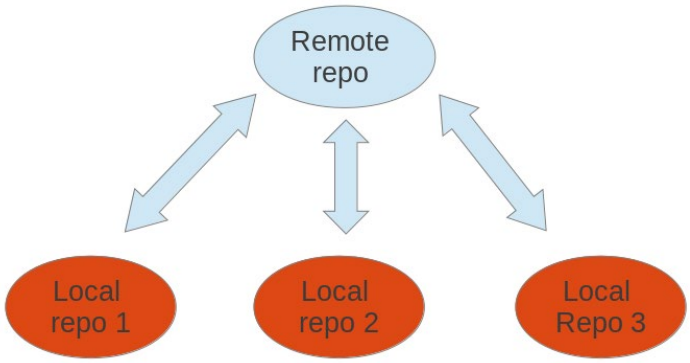
This is useful if you happen to use 2 different laptops to develop a project. Or the project is jointly developed by multiple authors.



Click **OK** to proceed. All the files in your Android Studio will be up-to-date.



2. Due to the projects can be located in a distributed manner, a Github project can be cloned to many different locations or by different collaborators. As the different local repo can be accessed and modified independent of one another some concurrency issue might be raised.



There's a possibility that there will be a conflict during update or commit. A conflict is due to changes made to the repository from the last time your copy of the project was updated.

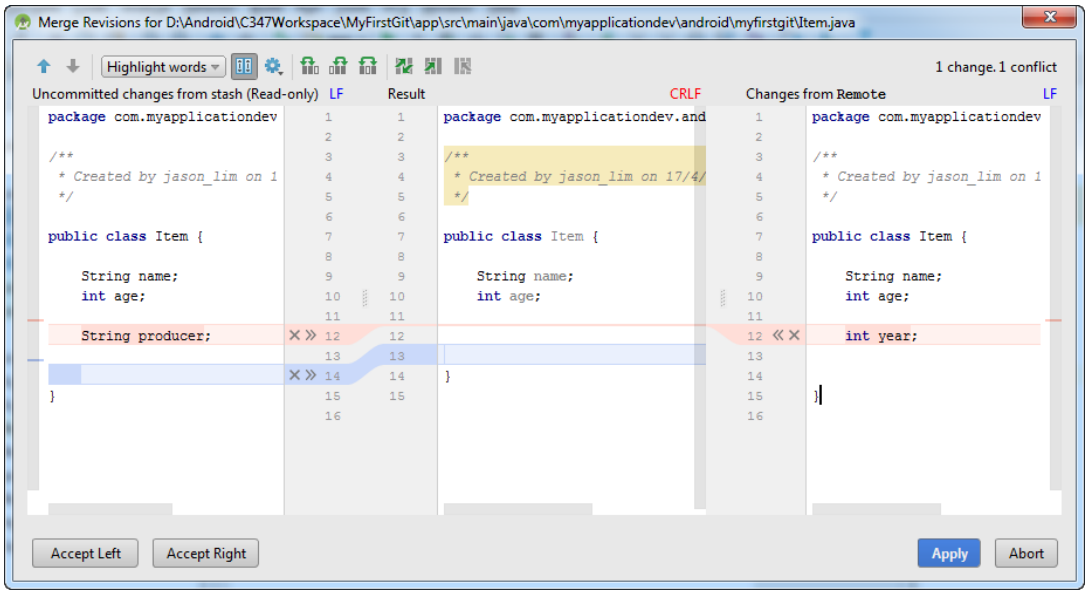
This could happen when there are more than 1 contributor/author for the project and they are making changes concurrently over the same period. Example, in a day, both authors tries to commit the same file.

When a conflict happens, it can be due to the following:

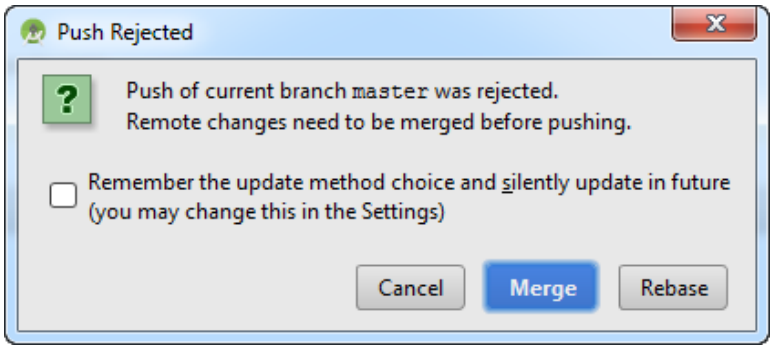
i) There are changes to the code in the repository since the last time you updated your local code
ii) Multiple collaborators doing some changes to the same file

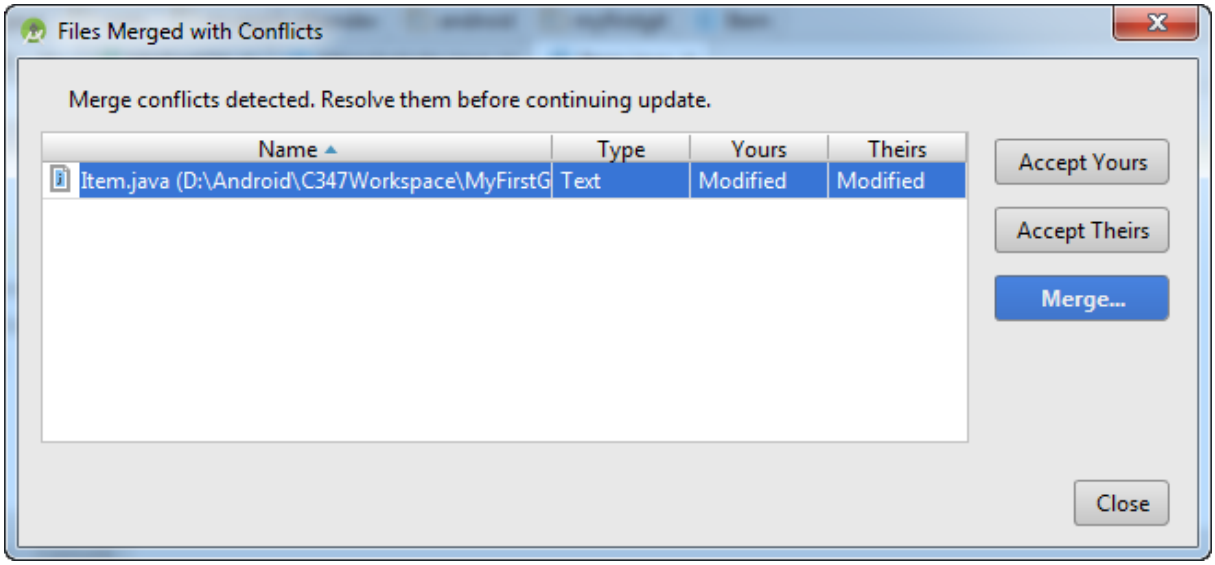
3. Resolving conflicts

You need to resolve the conflict. Typically, a window like this might show up. This is for you to decide which version of the code to keep.



There will be times, the Android Studio will just prompt for merge. That's to merge your changes to the changes another collaborator has done, before you commit your code and after you check out the project.

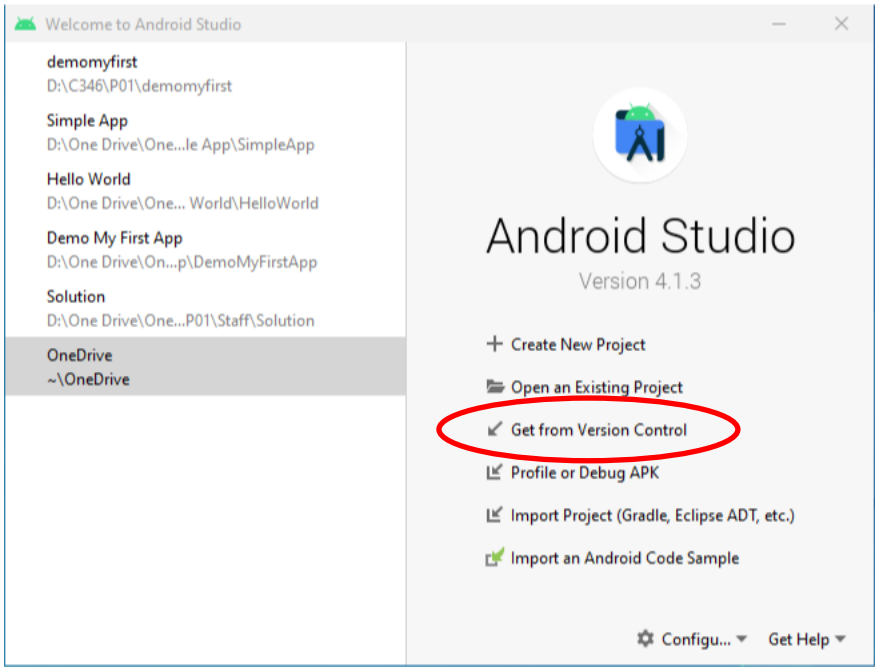




The idea is to look at the changes and to decide whether to keep your copy, other's copy or merge the code.

Import a project from Github

To check out a project from the Github, it's quite straight forward as the Android Studio has an option for it.



What you need is the link to the project on Github as well as where do you want to store the project.