Lab 1 - Introduction to Android Studio and Git Introduction

Due Sep 20 by 3:45pm **Points** 10 **Submitting** a file upload
Available Sep 13 at 12am - Sep 20 at 11:59pm 8 days

This assignment was locked Sep 20 at 11:59pm.

Lab 1: Introduction to Android Studio and Git

Introduction

This lab is to help you set up your Android development environment. We will be using Android Studio which is the official integrated development environment for Google's Android operating system. There are two main goals for this lab. The first one is to successfully install Android Studio on your personal computer and run a basic “Hello World” application along with setting up VCS(git) and pushing some changes to github. The second one is to make a clone of our class GitHub repository and make a few changes that will be described in more detail below. The successful completion of this lab will help you get more comfortable with Android Studio and version control with Git. This lab will help you complete more complex Android projects in the future.

Milestone 1.1 - Successfully compile and run “Hello World” app on an Android Virtual Device (AVD) in Android Studio.

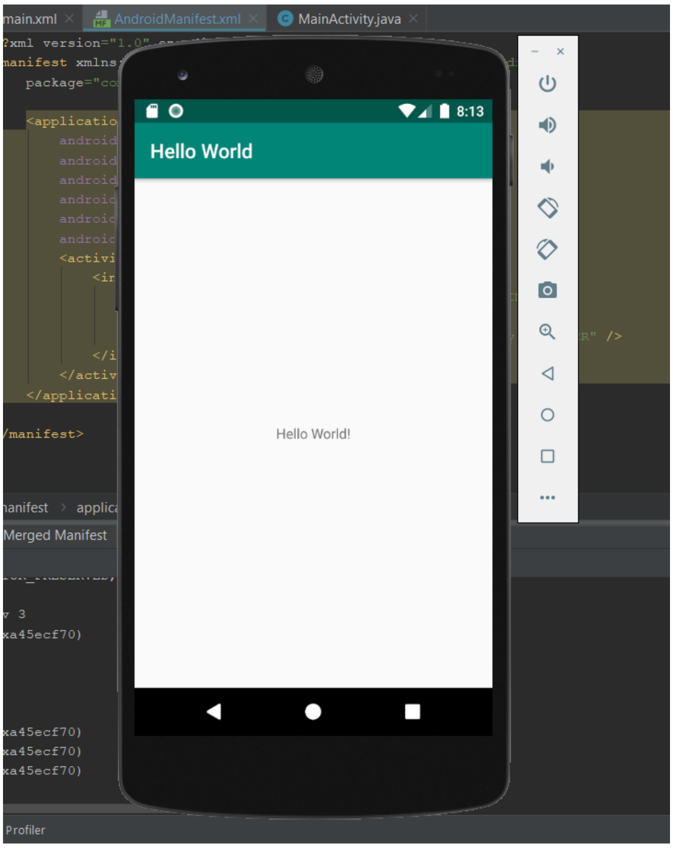
1. First step is to download Android Studio from the following link provided by Google: (https://developer.android.com/studio) https://developer.android.com/studio (https://developer.android.com/studio) . Then follow these steps:
   a. Navigate to the Android developers site and follow the instructions to download and install Android Studio.
   b. Accept the default configurations for all steps.
   c. Make sure that all components are selected for installation.
   d. After finishing the install, the Setup Wizard will download and install some additional components.
   e. When the download completes, Android Studio will start, and you are ready to create your “Hello World” project.
2. Create the “Hello World” app:
   a. Launch Android Studio if it is not already opened.
   b. In the main Welcome to Android Studio window, click "Start a new Android Studio project".
   c. Customize the Activity window under the Phone and Tablet section. Every app needs at least one activity. An activity represents a single screen with a user interface and Android Studio provides templates to help you get started. For the Hello World project, choose the **Empty Activity** template.

  d. In the New Project window, give your application an Application Name, such as "Hello World".
  e. Verify the Project location, or choose a different directory for storing your project.
  f. Make sure that you select the language as **Java**.
  g. Verify that the default Project location is where you want to store your Hello World app and other Android Studio projects, or **change it to your preferred directory**. Click Next.
  h. On the Target Android Devices screen, "Phone and Tablet" should be selected. Select **API 22: Android 5.1 Lollipop as the Minimum SDK**. Click Next.
  i. If your project requires additional components for your chosen target SDK, **Android Studio will install them automatically**. Click Next.
  j. Last step is to verify all the details that you have provided and click **Finish**.

3. Now that you have started an Android project, you will need a virtual device to run the app. To create an Android Virtual Device follow these steps:
   a. In Android Studio, select Tools > AVD Manager, or click the AVD Manager icon in the toolbar.
   b. Click the **+Create Virtual Device**…. (If you have created a virtual device before, the window shows all of your existing devices and the button is at the bottom.)
   c. The **Select Hardware** screen appears showing a list of preconfigured hardware devices. For each device, the table shows its diagonal display size (Size), screen resolution in pixels (Resolution), and pixel density (Density).
   d. Choose the Nexus 5 hardware device and click **Next**. If Nexus 5 device is not shown, select any other device.
   e. On the **System Image** screen, from the **Recommended** tab, choose which version of the Android system to run on the virtual device. You can select the latest system image.
   f. If a **Download** link is visible next to a system image version, it is not installed yet, and you need to download it. If necessary, click the link to start the download, and click **Finish** when it's done.
   g. On the **Verify Configuration** screen, you can select a name for your AVD (all the other settings are default) and click **Finish**.
4. Now that you have setup an Android Virtual Device and a “Hello World” project all that remaining to do is **run** this application on the AVD.
   a. In Android Studio, select **Run > Run app** or click the **Run icon** in the toolbar.

      (or)

   b. In Android Studio, select **Run > Select Device**. Under **Available Emulators**, select your **AVD name** (Nexus 5 API 23 by default) and click **OK**.
5. If you complete all these steps successfully you should see the emulator showing something like this:



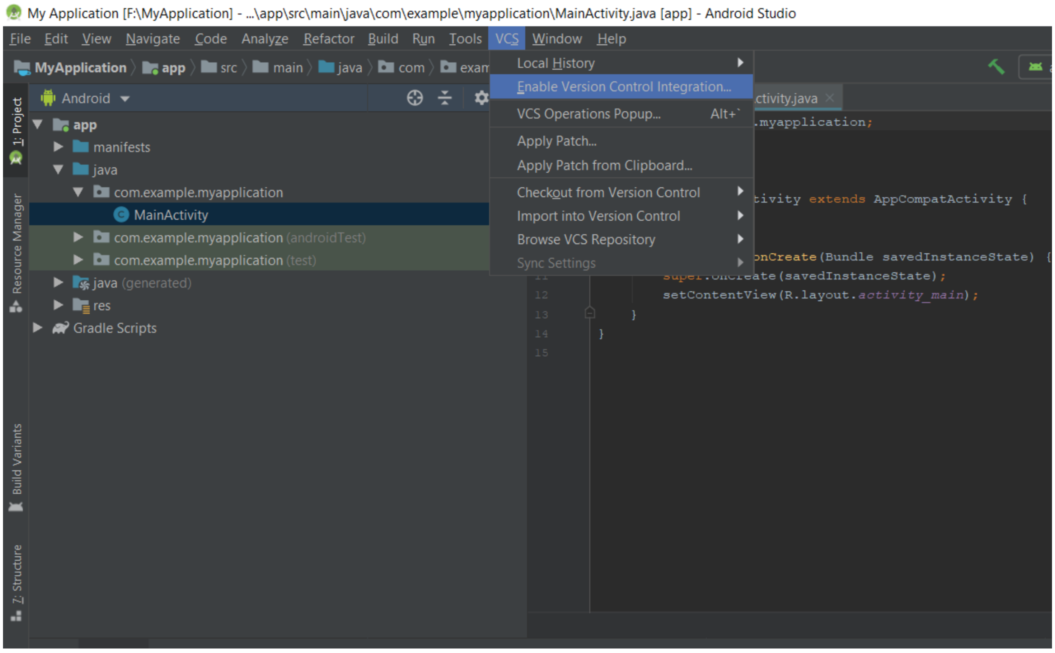
Milestone 1.2 - Successfully setup VCS (Git) and push changes to GitHub.

Version Control Setup - Now you will be learning how to use and integrate a Version Control System (VCS) with Android development. We will be using Git as our version control software to track your progress. In this milestone you will connect your GitHub account to your Android Studio project. If you do not already have a GitHub account then you can sign up at this link: **https://github.com/ (https://github.com/)**

We will be using GitHub classroom in this course to monitor your progress. Here is an easy tutorial to set up your first GitHub repository on Android Studio:

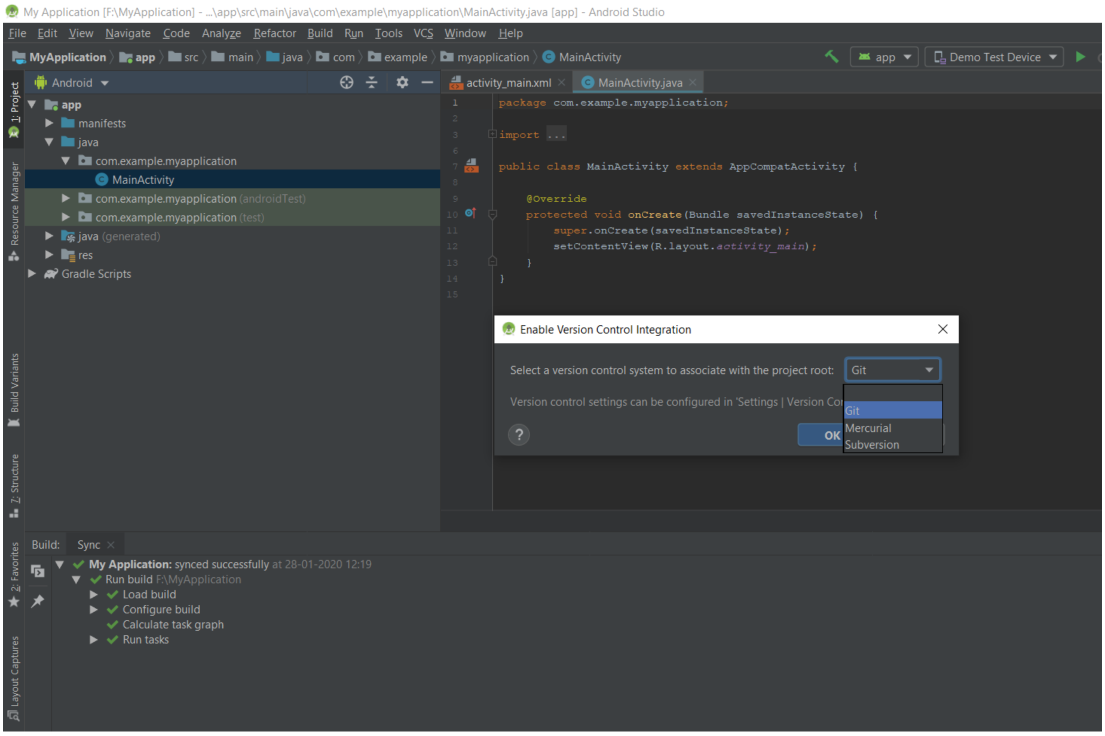
1. Install Git on your computer : Visit this **official site (https://git-scm.com/book/en/v2/Getting-Started-Installing-Git)** to download git on your computer. Once you do that, you can start using it with android studio.

2. Enable Version Control Integration on android studio : Go to Menu>VCS>Enable Version Control Integration


My Application [F:\MyApplication] - ...\app\src\main\java\com\example\myapplication\MainActivity.java [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication app src main java com exam
Android
app
manifests
java
com.example.myapplication
MainActivity
com.example.myapplication (androidTest)
com.example.myapplication (test)
java (generated)
res
Gradle Scripts
Local History
Enable Version Control Integration...
VCS Operations Popup... Alt+`
Apply Patch...
Apply Patch from Clipboard...
Checkout from Version Control
Import into Version Control
Browse VCS Repository
Sync Settings
ctivity.java
1: Project
Resource Manager
Build Variants
7: Structure

```
.myapplication;

tivity extends AppCompatActivity {

onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
  }
}
```

After that, select Git as your version control system


My Application [F:\MyApplication] - ...\app\src\main\java\com\example\myapplication\MainActivity.java [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication app src main java com example myapplication MainActivity
app
Demo Test Device
Android
activity_main.xml
MainActivity.java
app
manifests
java
com.example.myapplication
MainActivity
com.example.myapplication (androidTest)
com.example.myapplication (test)
java (generated)
res
Gradle Scripts
Enable Version Control Integration
Git
Mercurial
Subversion
OK
?
Build:
Sync
1: Project
Resource Manager
Build Variants
7: Structure
2: Favorites
Layout Captures

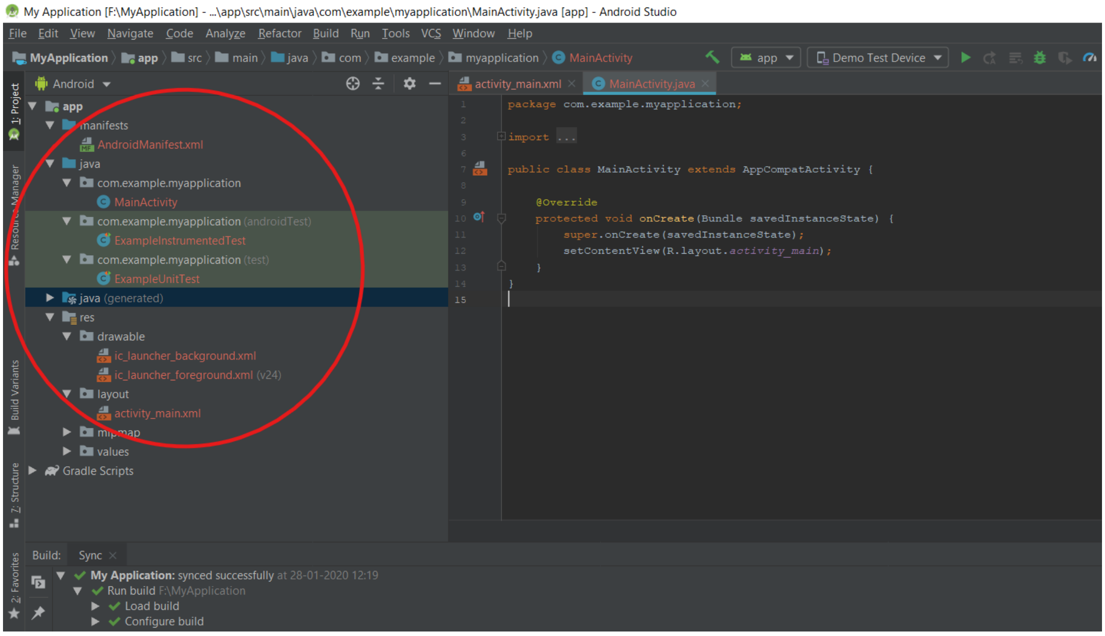
```
package com.example.myapplication;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Select a version control system to associate with the project root: Git

Version control settings can be configured in 'Settings | Version Co

```
My Application: synced successfully at 28-01-2020 12:19
  Run build F:\MyApplication
    Load build
    Configure build
    Calculate task graph
    Run tasks
```

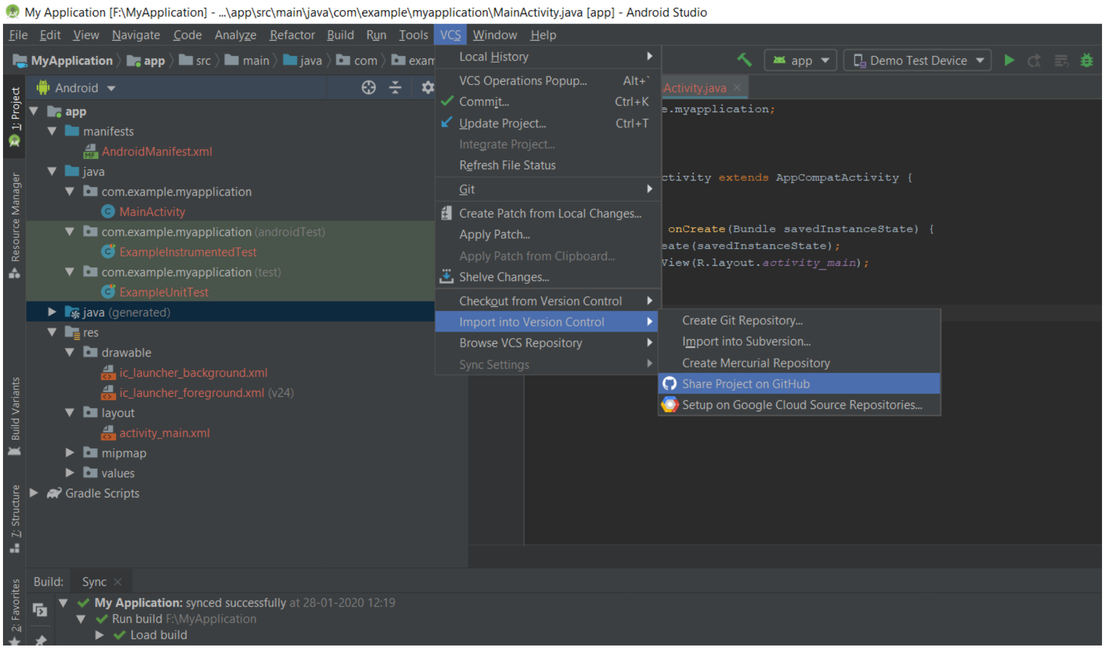
On doing this, you will notice that all your files will turn red.


My Application [F:\MyApplication] - ...\app\src\main\java\com\example\myapplication\MainActivity.java [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 myapplication 〉 MainActivity
app
Demo Test Device
Android
app
manifests
AndroidManifest.xml
java
com.example.myapplication
MainActivity
com.example.myapplication (androidTest)
ExampleInstrumentedTest
com.example.myapplication (test)
ExampleUnitTest
java (generated)
res
drawable
ic_launcher_background.xml
ic_launcher_foreground.xml (v24)
layout
activity_main.xml
mipmap
values
Gradle Scripts
activity_main.xml
MainActivity.java
1: Project
Resource Manager
Build Variants
Z: Structure
2: Favorites
Build:
Sync

```
package com.example.myapplication;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

```
My Application: synced successfully at 28-01-2020 12:19
  Run build F:\MyApplication
    Load build
    Configure build
```

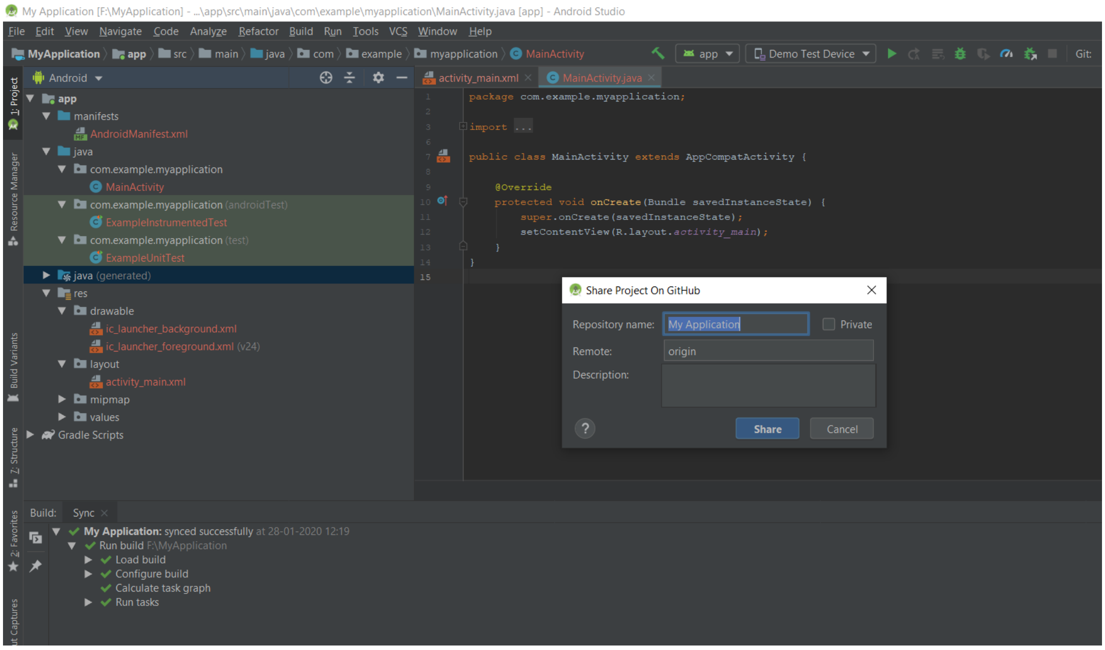
3. Share on Github : Now, go to VCS>Import into Version Control>Share project on Github **(or)** Git>GitHub>Share project on Github.


My Application [F:\MyApplication] - ...\app\src\main\java\com\example\myapplication\MainActivity.java [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication 〉 app 〉 src 〉 main 〉 java 〉 com 〉 exam
app
Demo Test Device
Android
app
manifests
AndroidManifest.xml
java
com.example.myapplication
MainActivity
com.example.myapplication (androidTest)
ExampleInstrumentedTest
com.example.myapplication (test)
ExampleUnitTest
java (generated)
res
drawable
ic_launcher_background.xml
ic_launcher_foreground.xml (v24)
layout
activity_main.xml
mipmap
values
Gradle Scripts
Local History
VCS Operations Popup... Alt+`
Commit... Ctrl+K
Update Project... Ctrl+T
Integrate Project...
Refresh File Status
Git
Create Patch from Local Changes...
Apply Patch...
Apply Patch from Clipboard...
Shelve Changes...
Checkout from Version Control
Import into Version Control
Browse VCS Repository
Sync Settings
Create Git Repository...
Import into Subversion...
Create Mercurial Repository
Share Project on GitHub
Setup on Google Cloud Source Repositories...
Activity.java
1: Project
Resource Manager
Build Variants
Z: Structure
2: Favorites
Build:
Sync

```
e.myapplication;

ctivity extends AppCompatActivity {

 onCreate(Bundle savedInstanceState) {
eate(savedInstanceState);
View(R.layout.activity_main);
```

```
My Application: synced successfully at 28-01-2020 12:19
  Run build F:\MyApplication
    Load build
```

Now give a name to your repository and write a description if you want**.** This step may require you to enter github credentials. In that case, enter login credentials. If you are facing any problems(Invalid Credentials/404 Data not Found) while using your credentials, try using “Access Token” feature present on the prompt. This access token has to be generated from the github website(after logging in) by going to Settings > Developer Settings > Personal Access Tokens > Generate new Token. Store this token on your system, as this will not be visible the next time you log in. You can always remove existing token and generate a new one.


My Application [F:\MyApplication] - ...\app\src\main\java\com\example\myapplication\MainActivity.java [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication app src main java com example myapplication MainActivity
app
Demo Test Device
Android
app
manifests
AndroidManifest.xml
java
com.example.myapplication
MainActivity
com.example.myapplication (androidTest)
ExampleInstrumentedTest
com.example.myapplication (test)
ExampleUnitTest
java (generated)
res
drawable
ic_launcher_background.xml
ic_launcher_foreground.xml (v24)
layout
activity_main.xml
mipmap
values
Gradle Scripts
activity_main.xml
MainActivity.java
Build: Sync

```
package com.example.myapplication;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Share Project On GitHub

Repository name: My Application ☐ Private

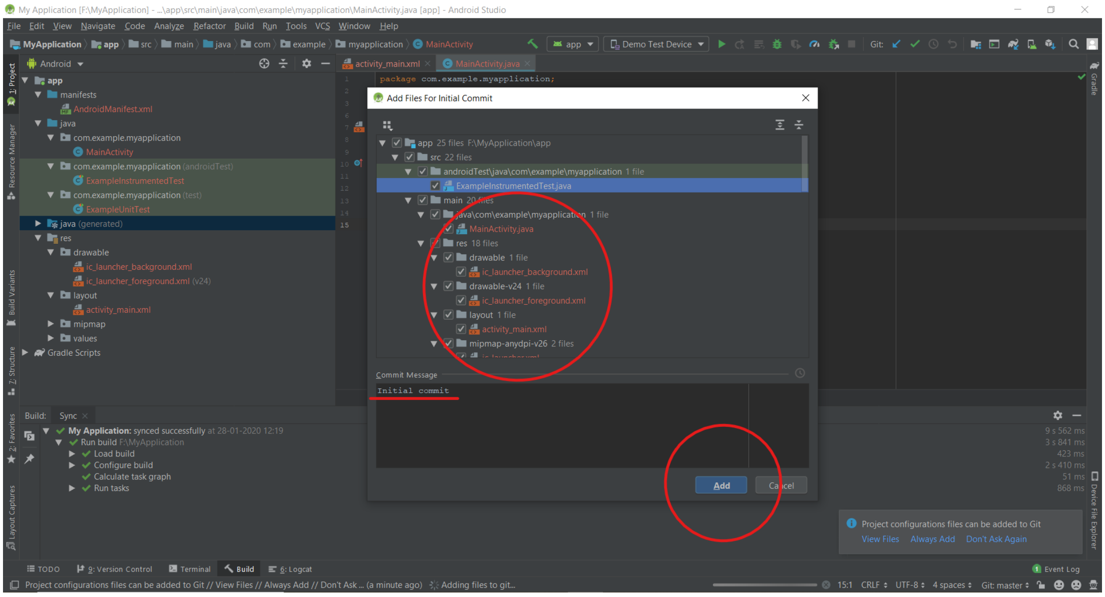
Remote: origin

Description:

Share Cancel

```
My Application: synced successfully at 28-01-2020 12:19
  Run build F:\MyApplication
    Load build
    Configure build
    Calculate task graph
    Run tasks
```

4. Push your project files on Github : Now select the files you want to share and press Ok. You can also write a specific commit message and after you do so, all your files will turn white.


My Application [F:\MyApplication] - ...\app\src\main\java\com\example\myapplication\MainActivity.java [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication app src main java com example myapplication MainActivity
app
Demo Test Device
Git:
Android
activity_main.xml
MainActivity.java
TODO
9: Version Control
Terminal
Build
6: Logcat
Event Log

Add Files For Initial Commit

```
☑ app 25 files F:\MyApplication\app
  ☑ src 22 files
    ☑ androidTest\java\com\example\myapplication 1 file
      ☑ ExampleInstrumentedTest.java
    ☑ main 20 files
      ☑ java\com\example\myapplication 1 file
        ☑ MainActivity.java
      ☑ res 18 files
        ☑ drawable 1 file
          ☑ ic_launcher_background.xml
        ☑ drawable-v24 1 file
          ☑ ic_launcher_foreground.xml
        ☑ layout 1 file
          ☑ activity_main.xml
        ☑ mipmap-anydpi-v26 2 files
```

Commit Message

```
Initial commit
```

Add Cancel

```
My Application: synced successfully at 28-01-2020 12:19
  Run build F:\MyApplication
    Load build
    Configure build
    Calculate task graph
    Run tasks
```

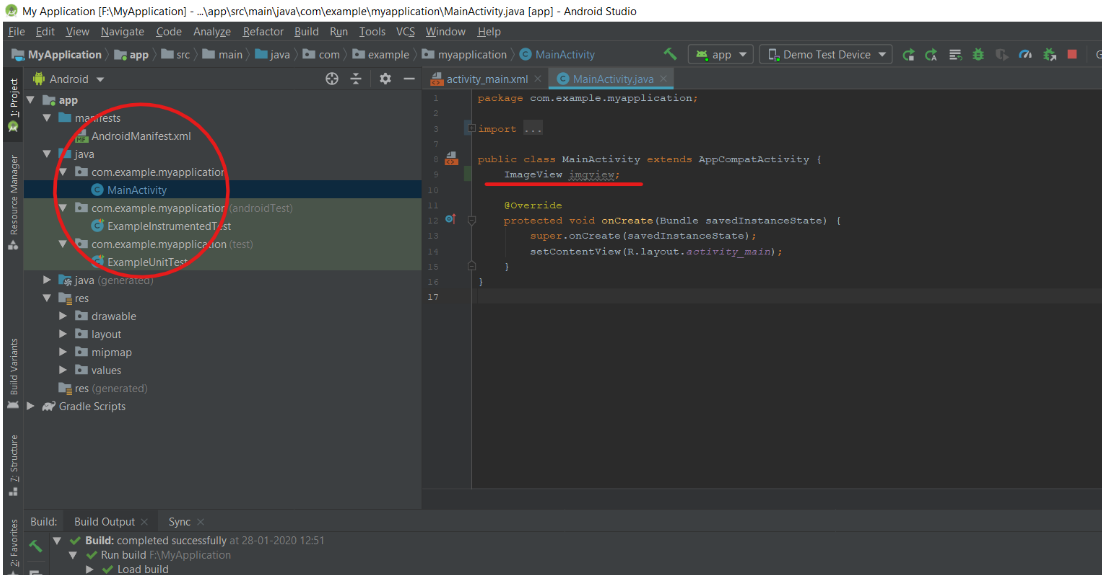
Project configurations files can be added to Git

View Files Always Add Don't Ask Again

Project configurations files can be added to Git // View Files // Always Add // Don't Ask ... (a minute ago) Adding files to git...

15:1 CRLF UTF-8 4 spaces Git: master

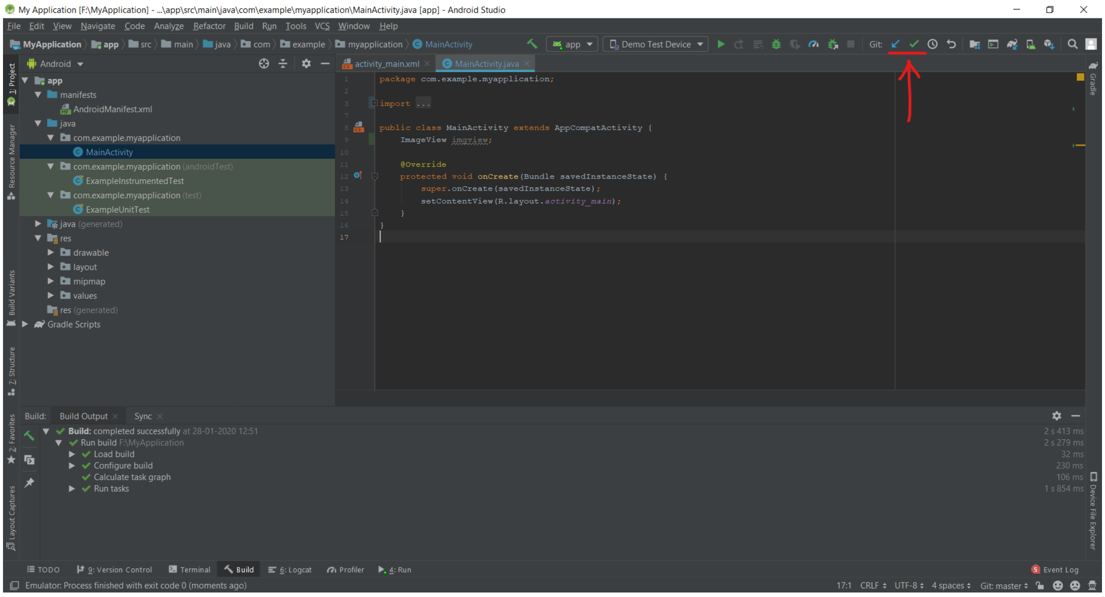
5. Make changes : Your project is now under version control and shared on Github, you can start making changes to commit and push. Add a new ImageView reference (code underlined in red) to the MainActivity.java. Notice that after the changes are made, the MainActivity.java file turned blue. That means the file now has uncommitted changes (and it doesn't match the file in the Github repository).


My Application [F:\MyApplication] - ...\app\src\main\java\com\example\myapplication\MainActivity.java [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication app src main java com example myapplication MainActivity
app
Demo Test Device
Android
activity_main.xml
MainActivity.java
app
manifests
AndroidManifest.xml
java
com.example.myapplication
MainActivity
com.example.myapplication (androidTest)
ExampleInstrumentedTest
com.example.myapplication (test)
ExampleUnitTest
java (generated)
res
drawable
layout
mipmap
values
res (generated)
Gradle Scripts
1: Project
Resource Manager
Build Variants
Z: Structure
2: Favorites
Build:
Build Output
Sync

```
package com.example.myapplication;

import ...

public class MainActivity extends AppCompatActivity {
    ImageView imgview;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

```
Build: completed successfully at 28-01-2020 12:51
  Run build F:\MyApplication
    Load build
```

6. Commit and Push : Its up to you now when you want to commit and push. You can do it after every new change or after a lot of changes. To commit and push, go to Menu>VCS (with green tick)

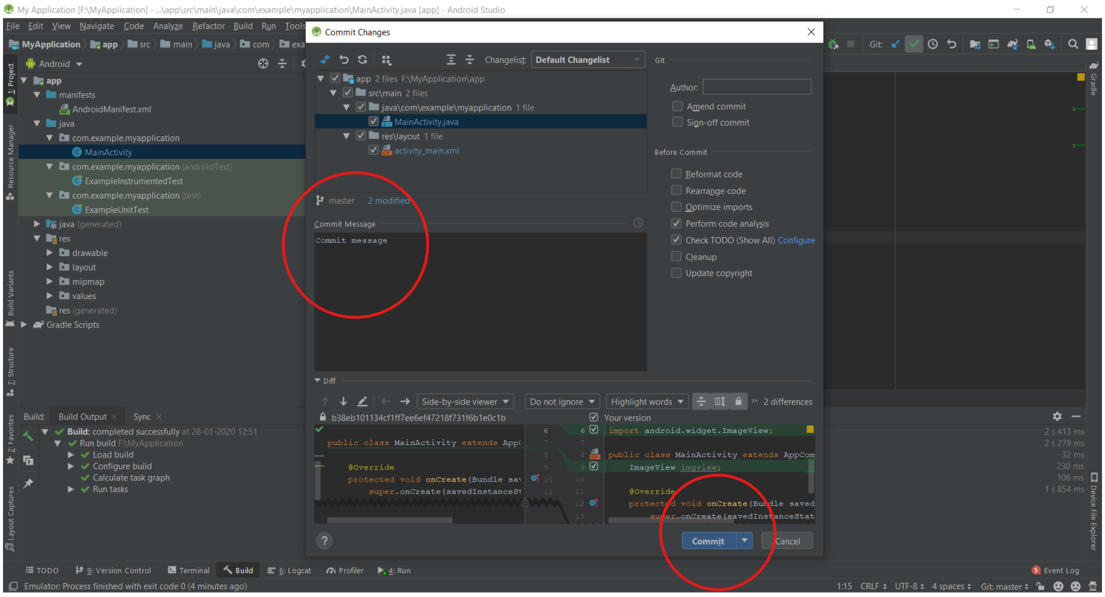

My Application [F:\MyApplication] - ...\app\src\main\java\com\example\myapplication\MainActivity.java [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication app src main java com example myapplication MainActivity
app
Demo Test Device
Git:
Android
activity_main.xml
MainActivity.java
app
manifests
AndroidManifest.xml
java
com.example.myapplication
MainActivity
com.example.myapplication (androidTest)
ExampleInstrumentedTest
com.example.myapplication (test)
ExampleUnitTest
java (generated)
res
drawable
layout
mipmap
values
res (generated)
Gradle Scripts
Build:
Build Output
Sync
TODO
9: Version Control
Terminal
Build
6: Logcat
Profiler
4: Run
Event Log
17:1 CRLF UTF-8 4 spaces Git: master
Gradle
Device File Explorer
Layout Captures

```
package com.example.myapplication;

import ...

public class MainActivity extends AppCompatActivity {
    ImageView imgview;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

```
Build: completed successfully at 28-01-2020 12:51        2 s 413 ms
  Run build F:\MyApplication                              2 s 279 ms
    Load build                                                 32 ms
    Configure build                                           230 ms
    Calculate task graph                                      106 ms
    Run tasks                                            1 s 854 ms
```

Emulator: Process finished with exit code 0 (moments ago)

After that, you will see a commit changes window and you have to write a commit message (mandatory) and then press on commit and push.



```
Commit message
```

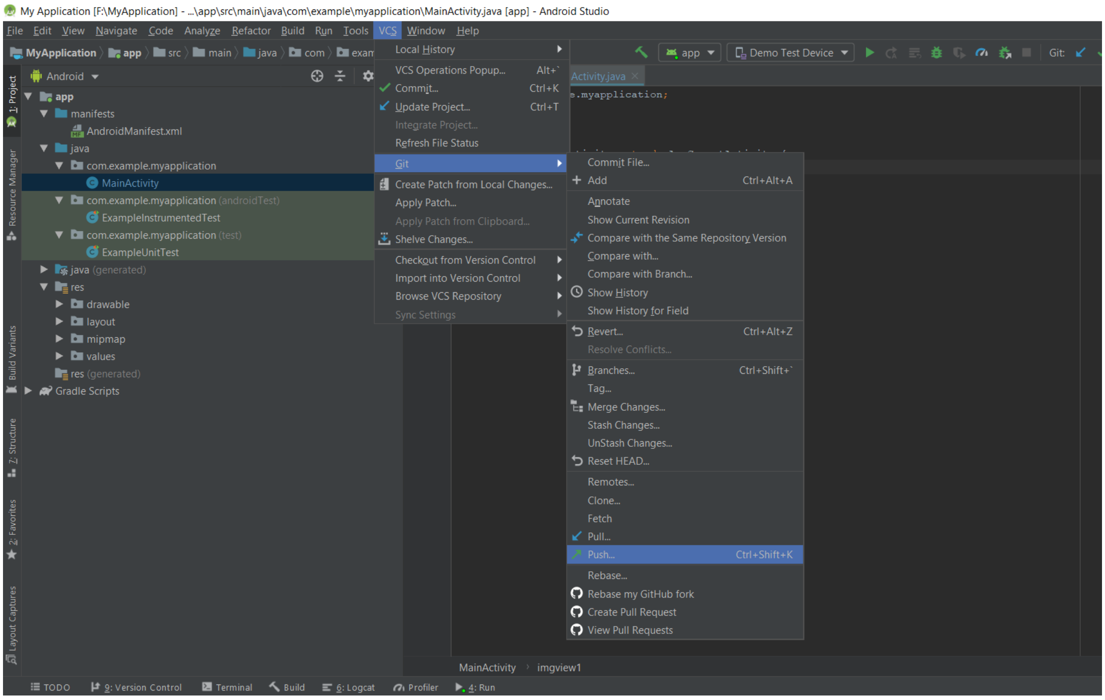
```
b38eb101134cf1ff7ee6ef47218f731f6b1e0c1b
public class MainActivity extends App(
    @Override
    protected void onCreate(Bundle sa
        super.onCreate(savedInstanceSt
```

```
import android.widget.ImageView;

public class MainActivity extends AppCom
    ImageView imgview;

    @Override
    protected void onCreate(Bundle saved
        super.onCreate(savedInstanceStat
```

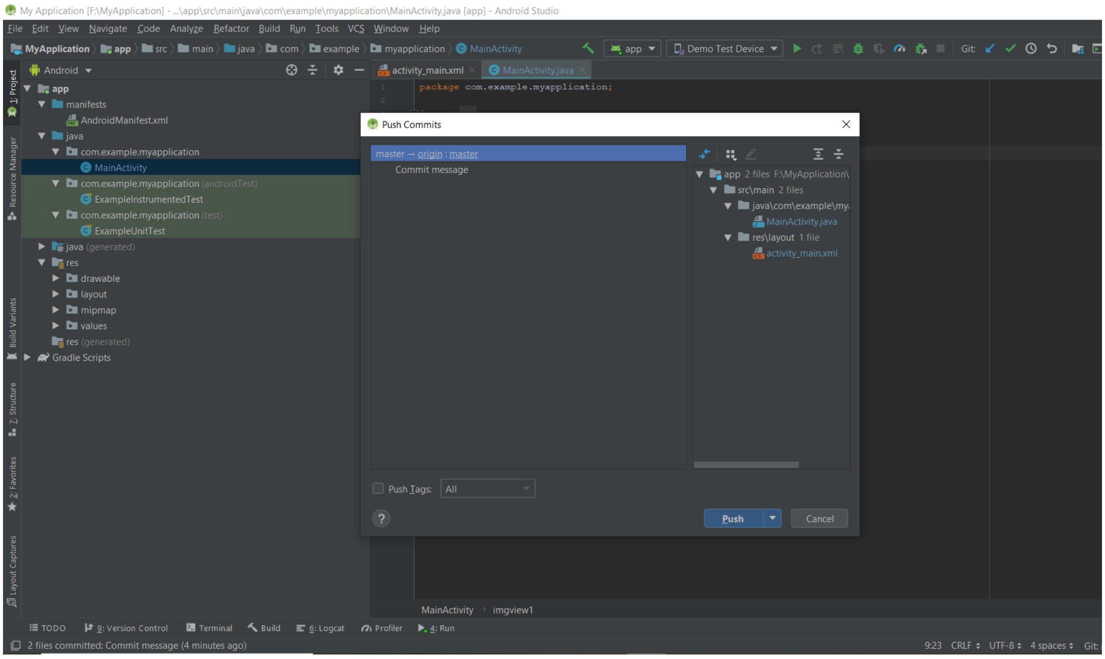
```
Build: completed successfully at 28-01-2020 12:51        2 s 413 ms
  Run build F:\MyApplication                              2 s 279 ms
    Load build                                                 32 ms
    Configure build                                           230 ms
    Calculate task graph                                      106 ms
    Run tasks                                            1 s 854 ms
```

Emulator: Process finished with exit code 0 (4 minutes ago)

After this step you will now push your changes to the GitHub repository. You will do this by navigating to VCS > Git > Push.



Finally, you will see another push window. Just press on push button and all your changes will be reflected in you repository.



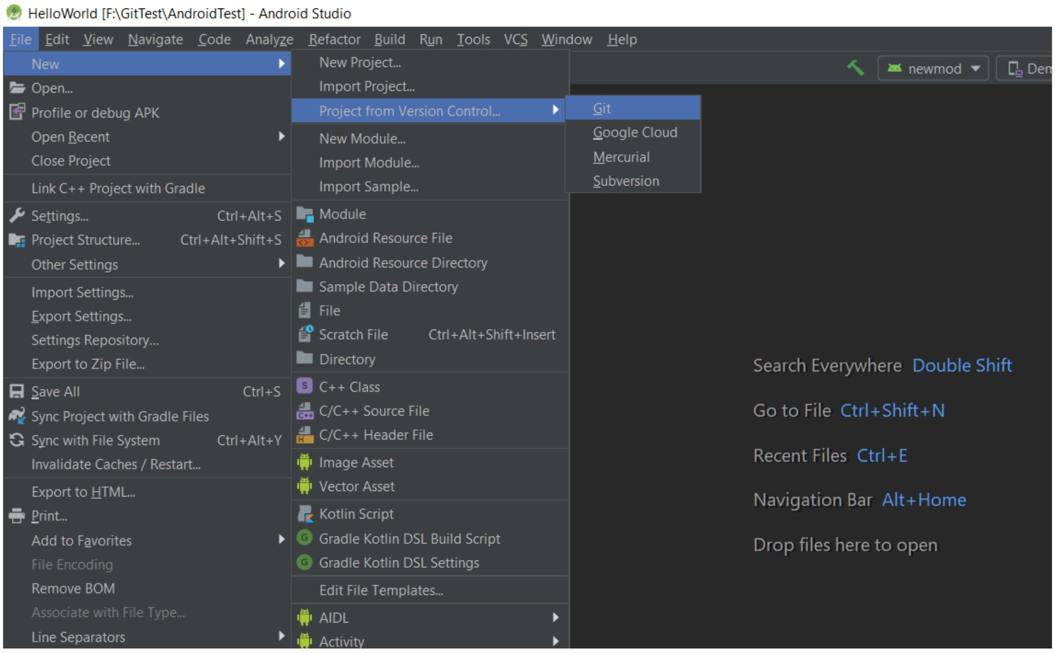
Congratulations on setting up your first project with version control.

Milestone 2

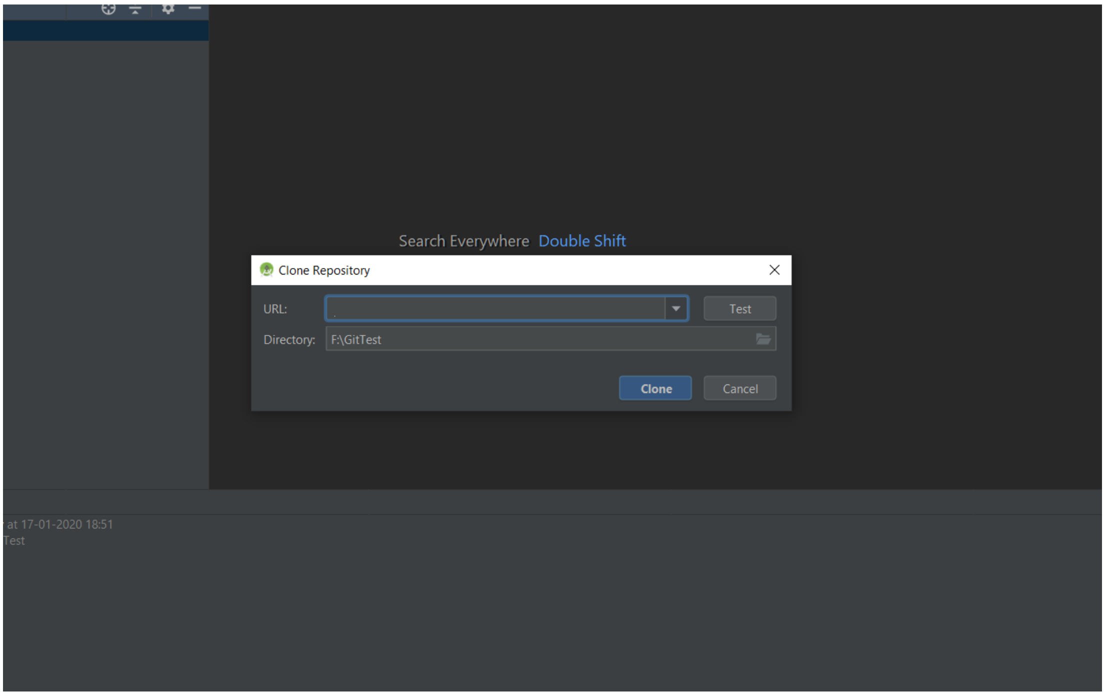
In this milestone you will have to successfully make a local clone of the repository that we have set up. Then you will have to make a git pull to get the most updated code. Once you have the most updated copy of the code you will have to make a few changes to the code which will be described below.

Now that you have Git setup on your computer and have connected your own private GitHub repositories to Android Studio, we can try using it to clone our classroom repository. To clone a repository follow these steps.

1. Clone a repository by selecting File > New > Project from Version Control > Git



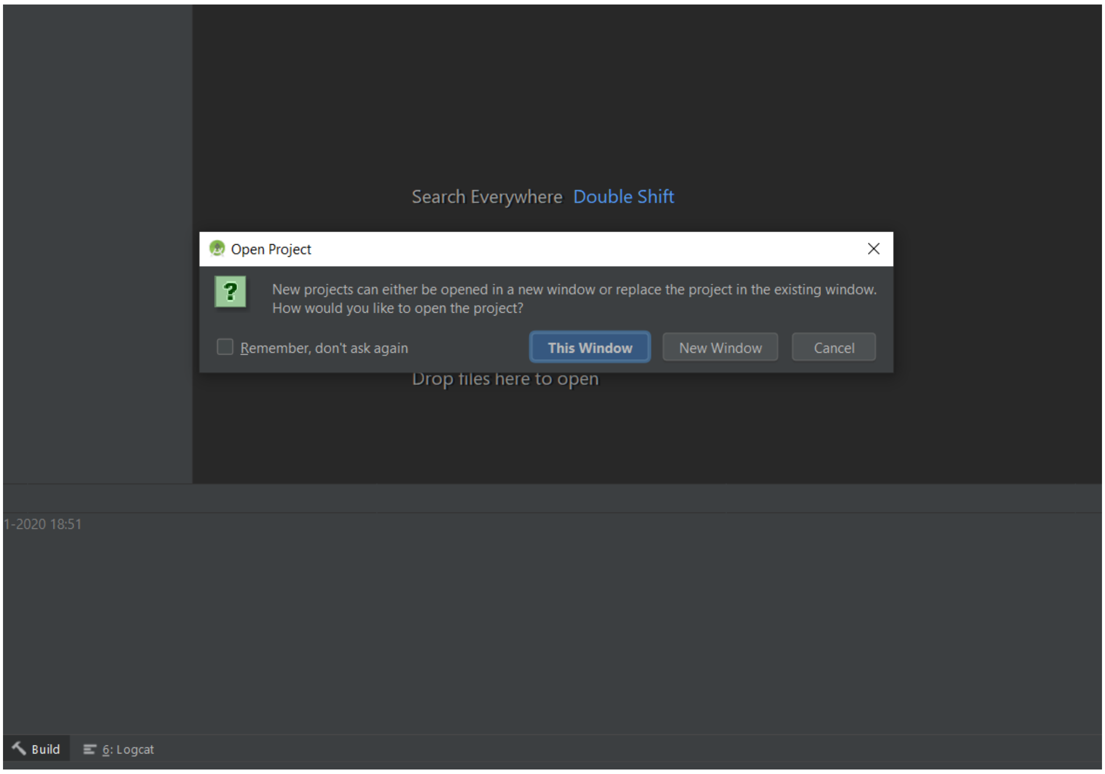
2. Then click the following link: [https://classroom.github.com/a/IVJ_ayLI (https://classroom.github.com/a/IVJ_ayLI)](https://classroom.github.com/a/IVJ_ayLI) to accept the invitation for this assignment. Once you click authorize you will receive a link (it will be something like https://github.com/CS407-Fall-2021/lab1- (https://github.com/CS-407-Spring2020/helloworld-kushalvpatel) yourgithubnamehere) which you can paste in the URL text field as shown in the picture below and hit **Clone.**



URL:

Directory: F:\GitTest

3. Then you will get this prompt which will ask you whether you want to open this project in the current window or in a new window. You can choose whichever option you are comfortable with.



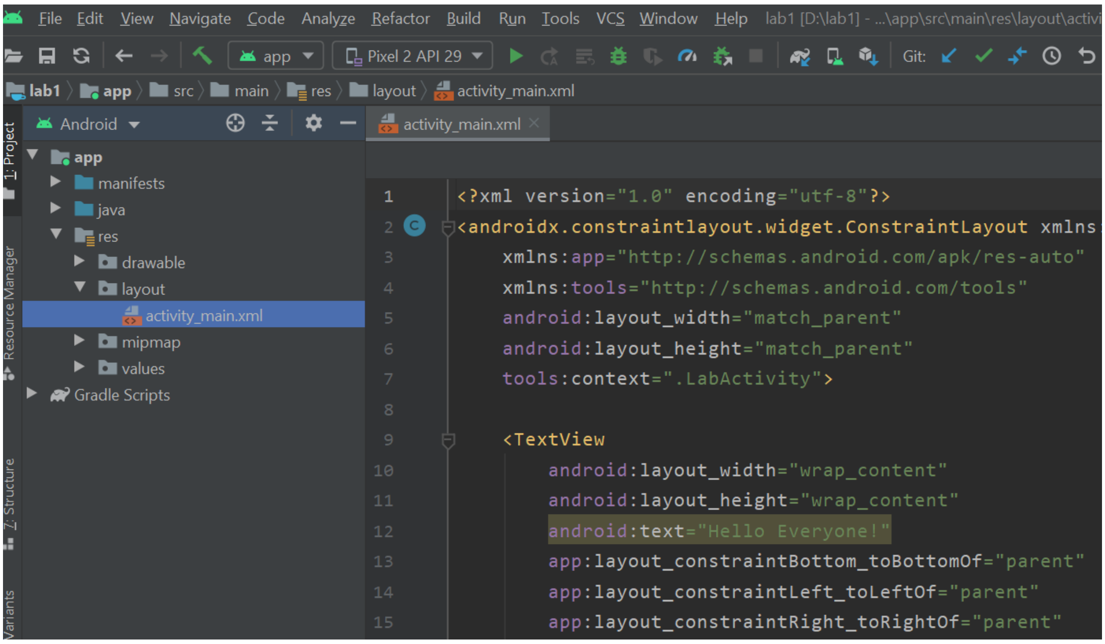
Open Project

New projects can either be opened in a new window or replace the project in the existing window. How would you like to open the project?

□ Remember, don't ask again

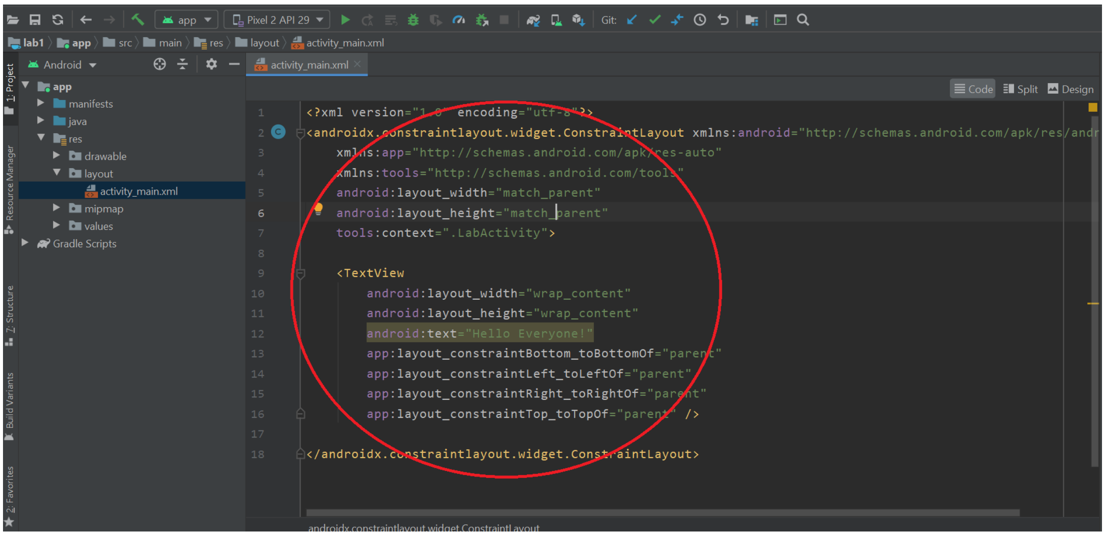
Once you have completed this step you will be left with a local clone of our GitHub classroom repository. Now you have the most updated copy of the code. You will have to make the following changes to the code given to you:

Open the activity_main.xml file (found in the files on the left hand side : app > res > layout):


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help lab1 [D:\lab1] - ...\app\src\main\res\layout\activi
app
Pixel 2 API 29
Git:
lab1 app src main res layout activity_main.xml
Android
activity_main.xml
app
manifests
java
res
drawable
layout
activity_main.xml
mipmap
values
Gradle Scripts
1: Project
Resource Manager
Z: Structure
Variants

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".LabActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello Everyone!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
```

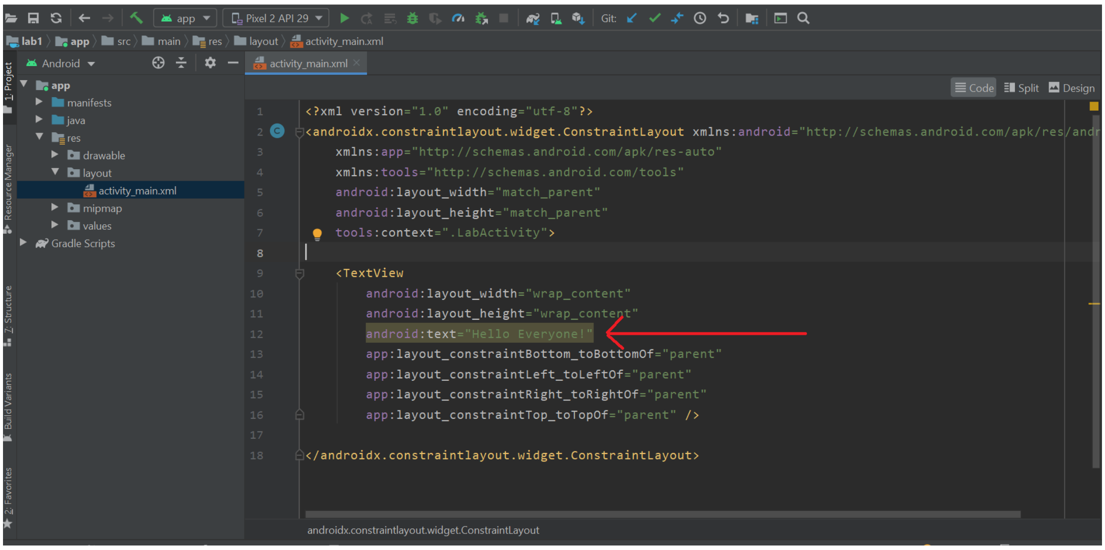
Once you open this file from the file explorer, you should be looking at some code like the one in the screenshot below.



```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/andr
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".LabActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello Everyone!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

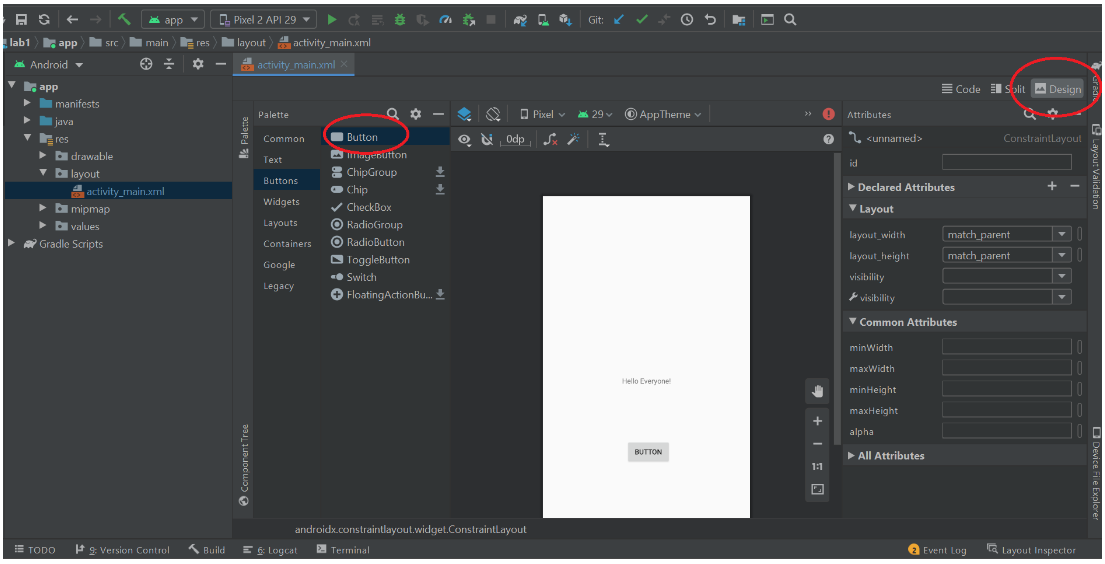
lab1 › app › src › main › res › layout › activity_main.xml
Android
app
manifests
java
res
drawable
layout
activity_main.xml
mipmap
values
Gradle Scripts
Code
Split
Design
androidx.constraintlayout.widget.ConstraintLayout

Your first task is to change message from “Hello Everyone!” to “Welcome to Android!”. You will need to make this change in the following location.



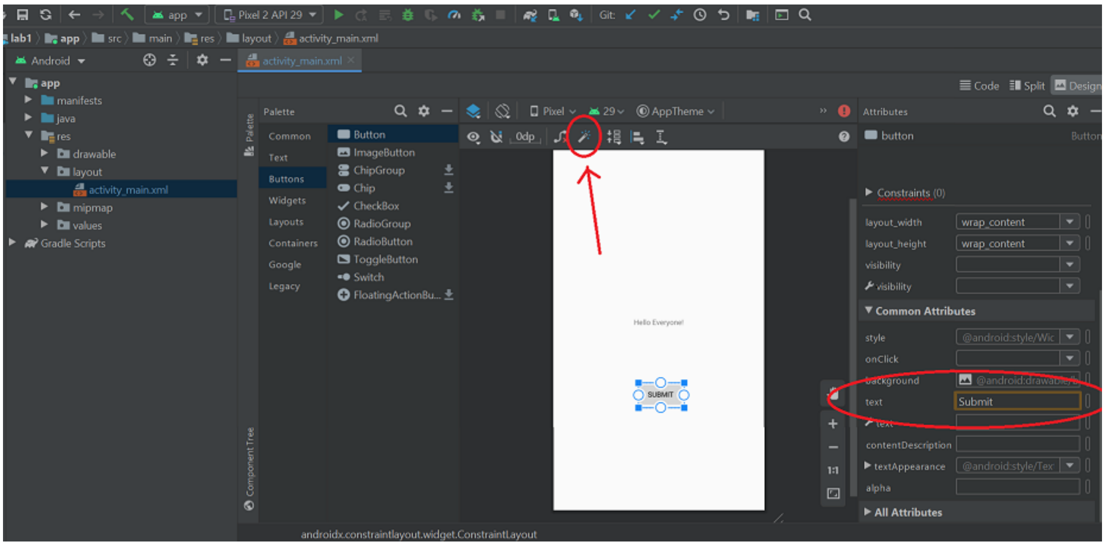
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/andr
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".LabActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello Everyone!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Your second task is to add a Button by using the **design** tab. You can use this design tab to drag and drop various components onto your app. For this task you should select Buttons > Button and drag and drop it onto the app. You will now be able to see it on the app preview. You can move it around and place it wherever you’d like as shown in the screenshot below.



Attributes
<unnamed> ConstraintLayout
id
Declared Attributes
Layout
layout_width match_parent
layout_height match_parent
visibility
visibility
Common Attributes
minWidth
maxWidth
minHeight
maxHeight
alpha
All Attributes

Click on “Infer Constraints Symbol” shown by arrow in the below screenshot to create automatic constraints. Change the text that the Button displays by going back to the **text/code** tab or you can change it in the design tab by first selecting the button and then editing the common attributes (common attributes > text) as shown below.



layout_width: wrap_content
layout_height: wrap_content
visibility:
visibility:
style: @android:style/Wic
onClick:
background: @android:drawable/
text: Submit
text:
contentDescription:
textAppearance: @android:style/Tex
alpha:

As the semester progresses, you will get more and more comfortable with the various resources that Android Studio provides. Now that you have made these changes, commit these changes to your local version control repo as you have learned from the git tutorial above.

Milestone 3

After you have made these changes to your android project, the last milestone is to Git push all these changes to your remote GitHub classroom repository. We will be checking the time and date of the last push to your individual repositories and grading you accordingly.

Conclusion

You have done a great job to get this far. Generally, the setup is the most boring part of any software development. Now that you have successfully completed this you are ready to take on more complex challenges that are waiting for you in the next many weeks of this semester.

References

- Google has great documentation for Android and this a good link to help get you started with this project. **https://google-developer-training.github.io/android-developer-fundamentals-course-practicals/en/Unit%201/11_p_hello_world.html** (https://google-developer-training.github.io/android-developer-fundamentals-course-practicals/en/Unit%201/11_p_hello_world.html)
- Github set up tutorial. **https://medium.com/code-yoga/how-to-link-android-studio-with-github-312037a13b99** (https://medium.com/code-yoga/how-to-link-android-studio-with-github-312037a13b99)

Lab 1 Rubric

Criteria	Ratings			Pts
Milestone 1 - Run Hello World app in Emulator	**2 pts** **Full Marks**	**0 pts** **No Marks**		2 pts
Milestone 1 - Integrate VCS to Android Studio and push Hello world code to GitHub	**2 pts** **Full Marks**	**0 pts** **No Marks**		2 pts
Milestone 2 - Pull source code from GitHub	**1 pts** **Full Marks**	**0 pts** **No Marks**		1 pts
Milestone 2 - Show demo after adding Button to the app	**3 pts** **Full Marks**	**1.5 pts** **Relevant code present, but demo is not working**	**0 pts** **No Marks**	3 pts
Milestone 3 - Git push all changes to remote GitHub classroom repository	**2 pts** **Full Marks**	**0 pts** **No Marks**		2 pts
				Total Points: 10