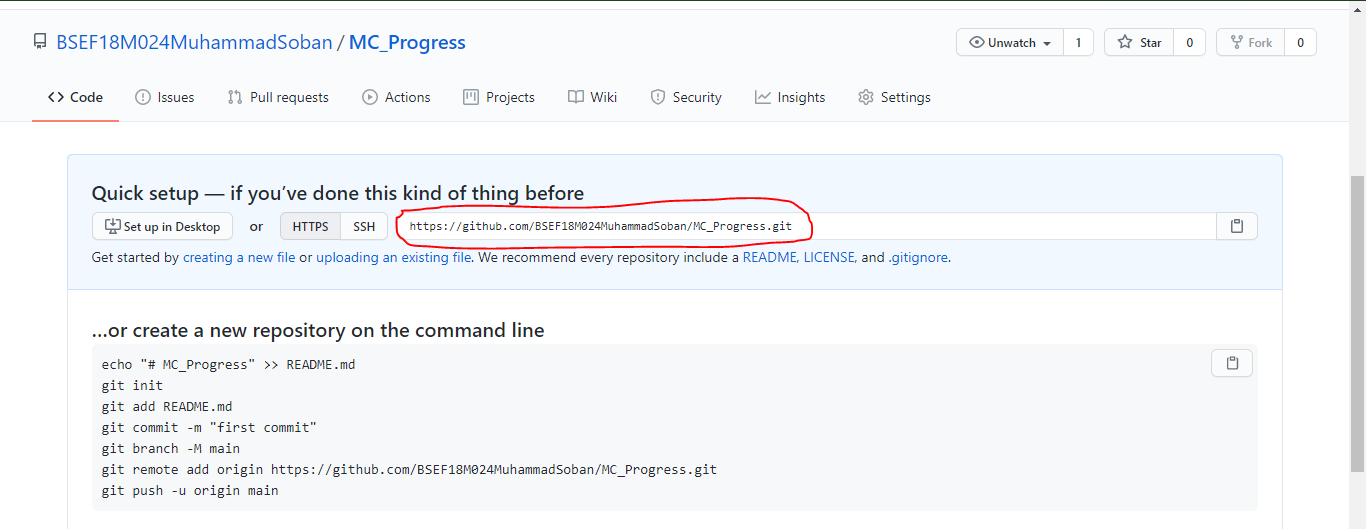
# Mobile Computing Progress

This Document include lecture by lecture progress of Mobile computing course.

## **Lecture 1**

In the first lecture we discussed about importance of “Niyat” and description of “Ahsaan” in the light of Quran and Hadith. We also got introduction about Version Control System and GitHub.

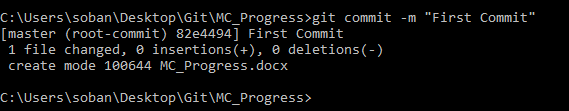
We also learned about how to create repository and get the link.

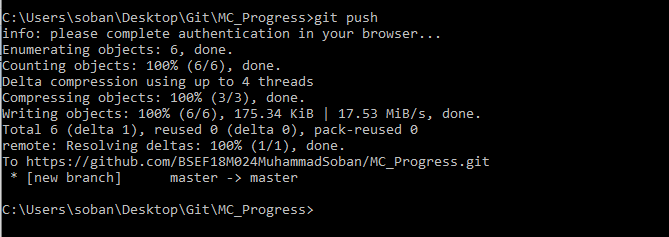
## **Lecture 2**

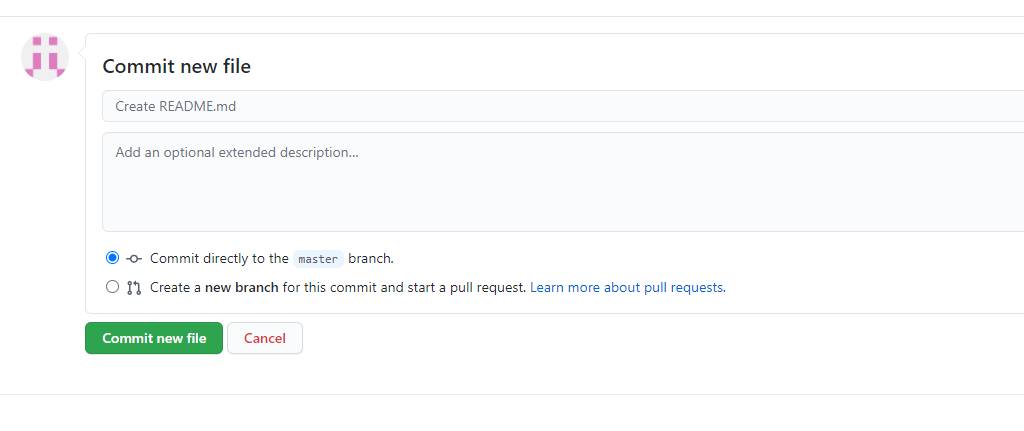
We learned some basic commands of Git like clone as shown below to get copy of master repository on your pc.

Then we add some file named MC\_Progress in local repository and below is the method to add this using git add command.

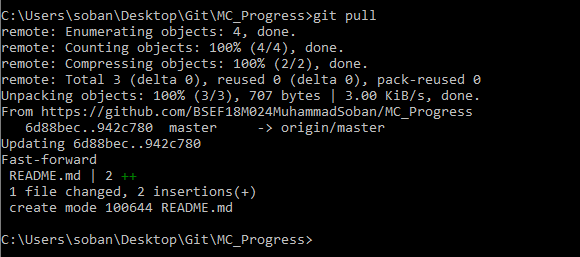
Now to commit (to make changes trackable) in local repository we use commit command as below

Now after making changes in local repository if we want to push changes in central repository, we use push command a below

and we also learned how to edit and commit files online



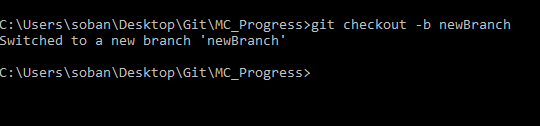
and how to pull in local repository using git pull command as shown below

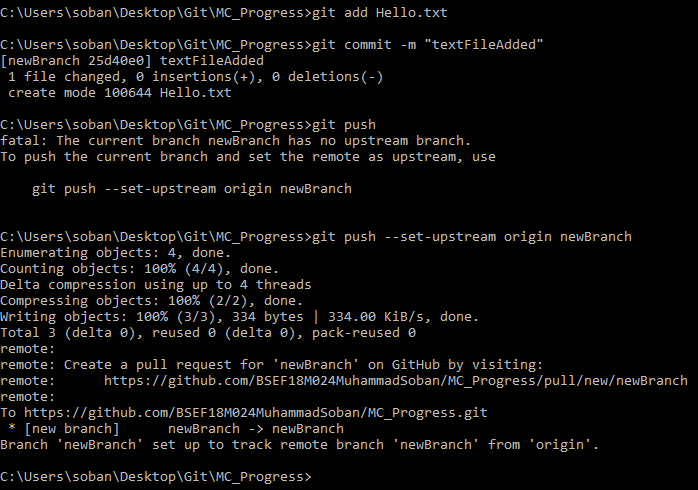


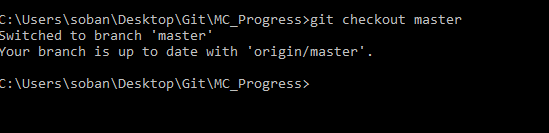
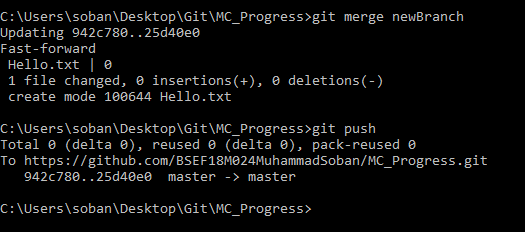
## **Lecture 3**

First of all, in video lecture sir recalled the previous learning regarding GitHub. Sir told us about merge conflict issue which occur if different people made different changes to same file. Now let’s play with git branches. To check available branches, we use command of git branch

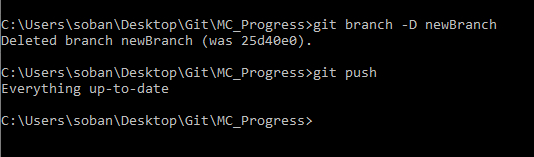
To create new branch, we use command as below

Now the controls shifted to newBranch where you can made changes and commit them as below

now to transfer control to master branch do as below

 Now we see ho to merge 2 branches

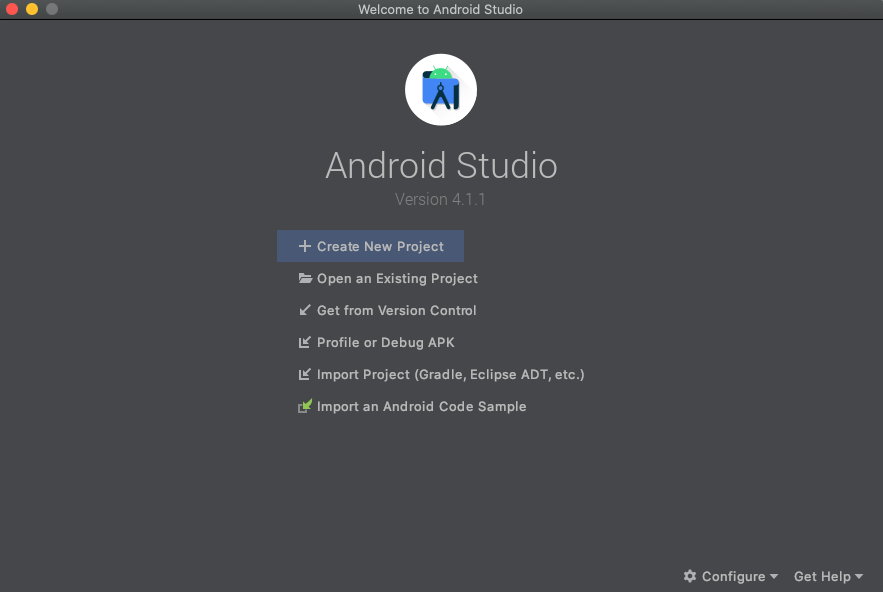
Now to delete newly created branch you can do follow

instead of directly pushing, after deleting branch you should run git add . command and then git commit and then git push to made change in online repository also.

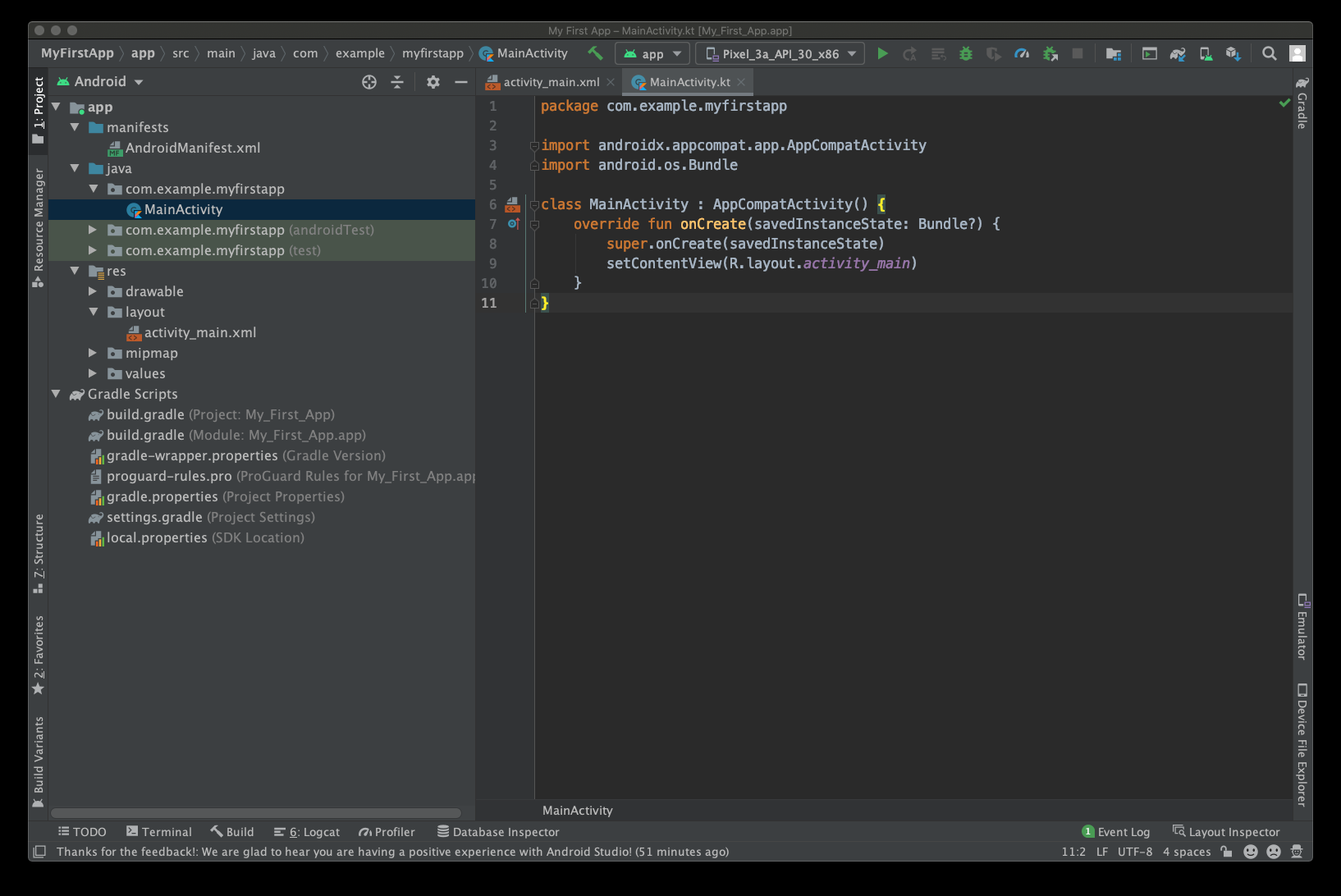
## **Lecture 4**

### Creating First Project in Android Studio

To create your new Android project, follow these steps:

1. Install the latest version of Android Studio
2. In the **Welcome to Android Studio** window, click **Create New Project**.
3. If you have a project already opened, select **File > New > New Project**.
4. In the **Select a Project Template** window, select **Empty Activity** and click **Next**.
5. In the **Configure your project** window, complete the following:
   * Enter "My First App" in the **Name** field.
   * Enter "com.example.myfirstapp" in the **Package name** field.
   * If you'd like to place the project in a different folder, change its **Save** location.
   * Select either **Java** or **Kotlin** from the **Language** drop-down menu.
   * Select the lowest version of Android you want your app to support in the **Minimum SDK** field.
   * If your app will require legacy library support, mark the **Use legacy android.support libraries** checkbox.
   * Leave the other options as they are.
6. Click **Finish**.

After some processing time, the Android Studio main window appears.



Now take a moment to review the most important files.

First, be sure the Project window is open (select View > Tool Windows > Project) and the Android view is selected from the drop-down list at the top of that window. You can then see the following files:

app > java > com.example.myfirstapp > MainActivity

This is the main activity. It's the entry point for your app. When you build and run your app, the system launches an instance of this Activity and loads its layout.

app > res > layout > activity\_main.xml

This XML file defines the layout for the activity's user interface (UI). It contains a TextView element with the text "Hello, World!"

app > manifests > AndroidManifest.xml

The manifest file describes the fundamental characteristics of the app and defines each of its components.

Gradle Scripts > build.gradle

There are two files with this name: one for the project, "Project: My\_First\_App," and one for the app module, "Module: My\_First\_App.app." Each module has its own build.gradle file, but this project currently has just one module. Use each module's build.gradle file to control how the Gradle plugin builds your app. For more information about this file, see Configure your build.

## **Lecture 5**

We Learned about Layouts

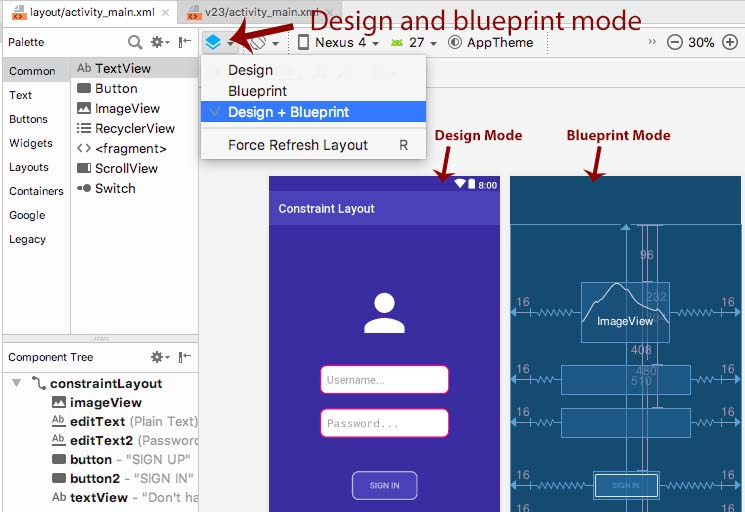
**Step 01:** Firstly, we can make a new empty project as taught in Lecture 4.

**Step 02:** To create a new layout file in your project:

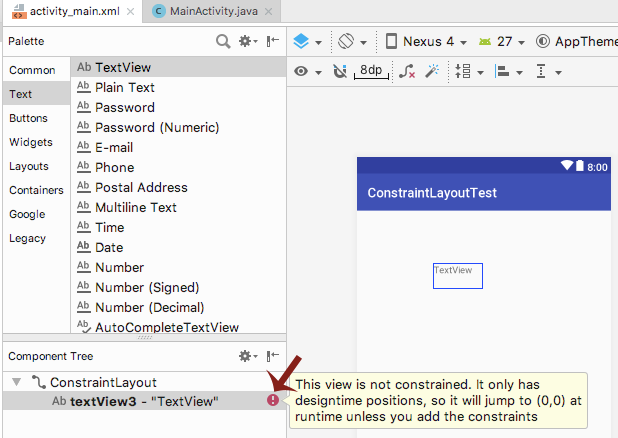
1. Right click on the layout folder.
2. Choose New.
3. Choose Layout Resource File.
4. Name your file.
5. Choose your Root Element.
6. Click ok.

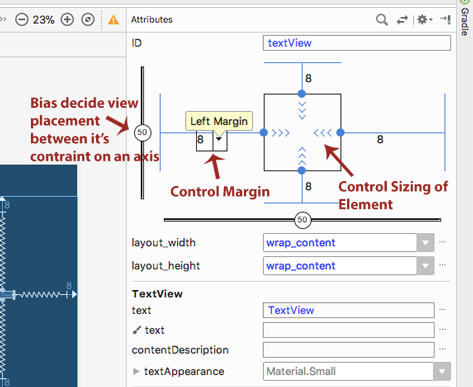
**Default Layout (Constraint)**

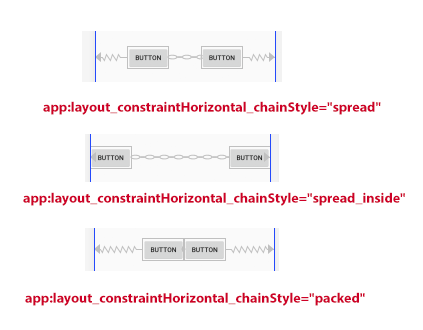
The default layout is constraint layout. Constraint Layout is a ViewGroup (i.e. a view that holds other views) which allows you to create large and complex layouts with a flat view hierarchy, and also allows you to position and size widgets in a very flexible way. It was created to help reduce the nesting of views and also improve the performance of layout files.



Here details and option given in this layout are provided in following Images





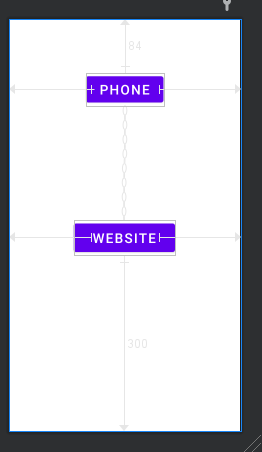


## **Lecture 6**

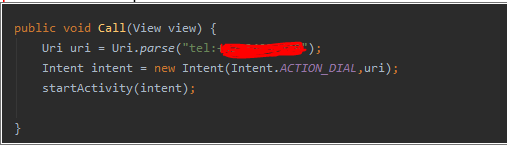
**Implicit Intents**

Implicit intents do not name a specific component, but instead declare a general action to perform, which allows a component from another app to handle it. For example, if you want to show the user a location on a map, you can use an implicit intent to request that another capable app show a specified location on a map.

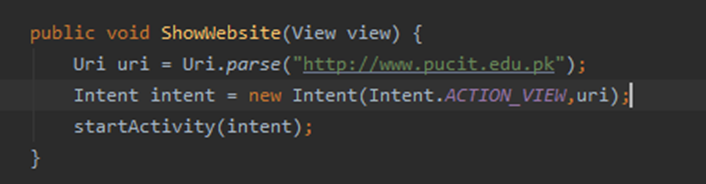
**Step 01:** On MainActivity we make two Button with the name of Phone and Website.



**Step 02:** On click of Phone Button we call the function of Call, in which phone dial bar is open.



**Step 03:** On click of Website Button we call the function of ShowWebsite, in which website is open.



## **Lecture 7**

**Explicit Intents**

Android Explicit intent specifies the component to be invoked from activity. In other words, we can call another activity in android by explicit intent. We can also pass the information from one activity to another using explicit intent.

**Step01:** Firstly, we create first activity like below

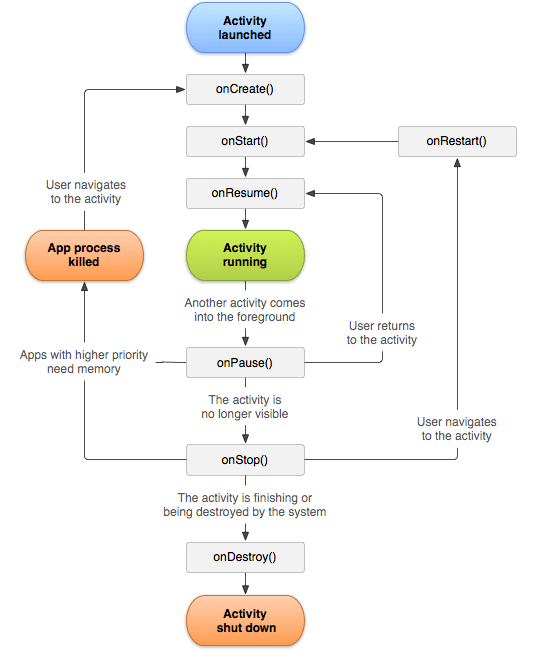
**Step02:** Now we create 2nd activity

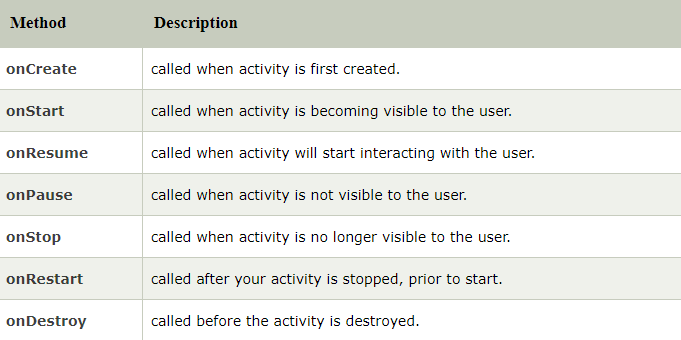
So we can call 2nd activity from 1st and 1st from 2nd.

## **Lecture 8**

**Activity** **Life** **Cycle**

Android Activity Lifecycle is controlled by 7 methods of android.app.Activity class. An activity is the single screen in android. It is like window or frame of Java.

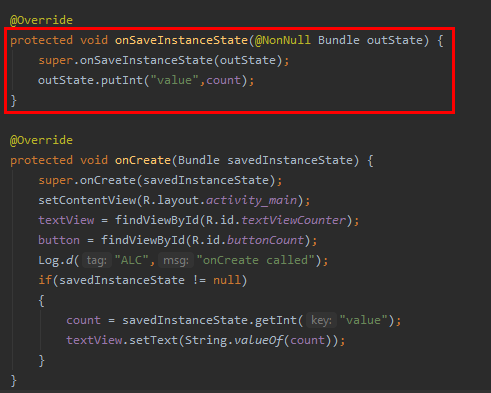


The Sequence is as follow.

**Step01:** we make increaseNumber in mainActivity.java function that increase the counter value after clicking the button

Now when we increment then after the screen rotated the counter reset to 0.

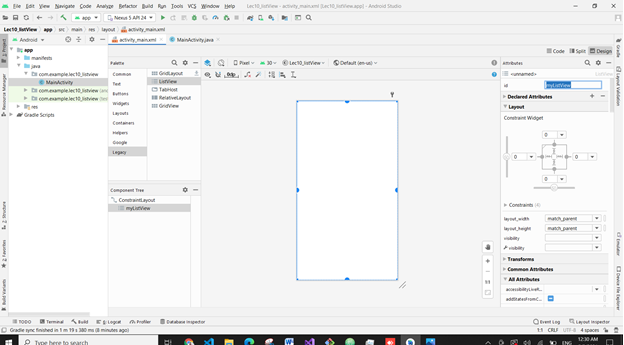
**Step02:** We can solve this problem as follow.



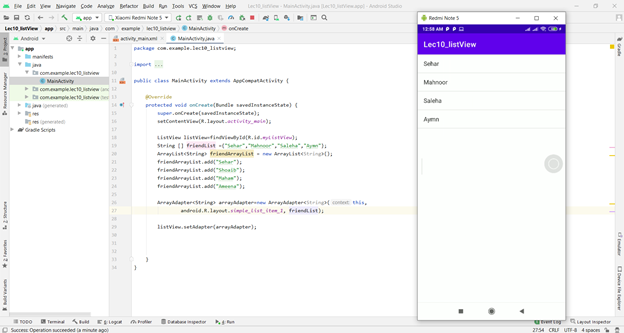
## **Lecture 9**

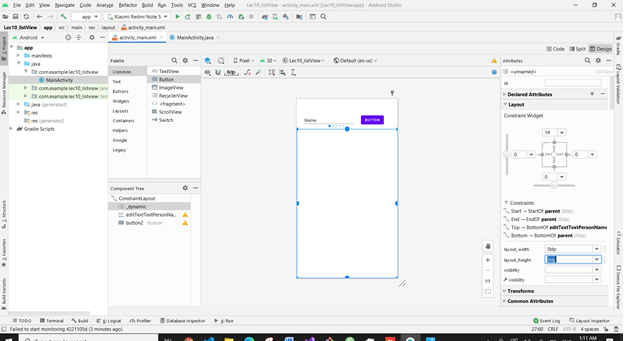
**List View:**

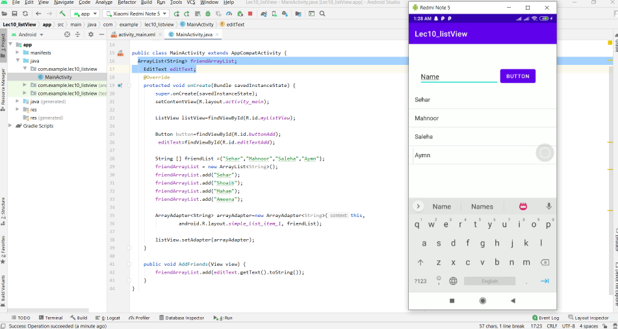
**Step01:** First we have drawn the list view as follow.



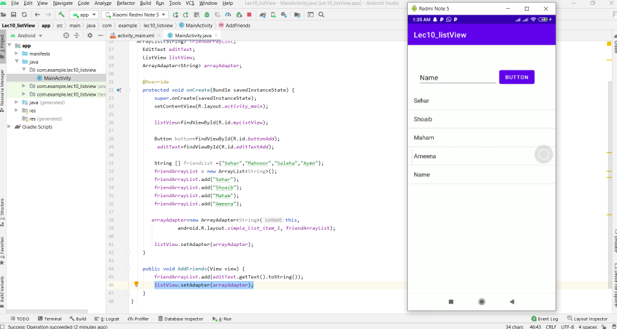
**Step02:** Then in main activity we write some code In the below pic we can also use friendArraylist



**Step03:** Then we have added a button and plain text and assign them ids above list view and make list view match constraint



**Step04**: Its not working because that adapter part should be in function So we copy it into function and make the instances of listview and adapter above override.

****

.

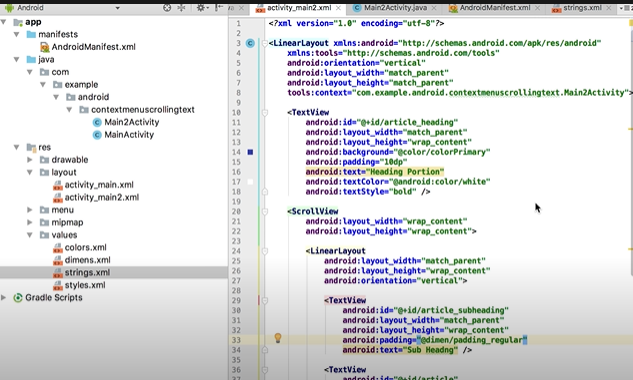
**Step05**: where you click on name it show us on logcat that you clicked this

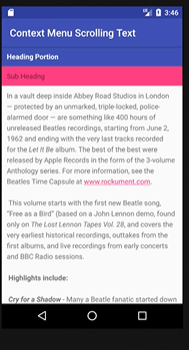


## **Lecture 10**

**Scroll View**

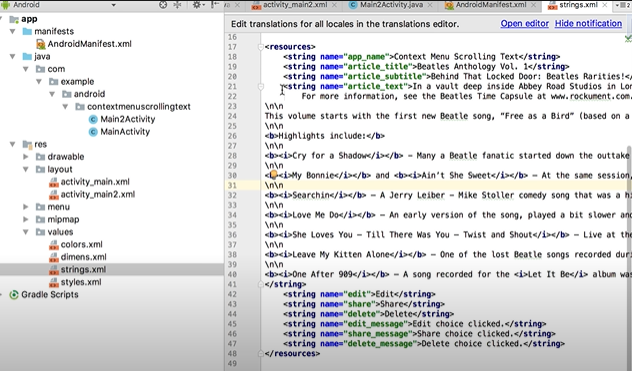
**Step01:** Create Xml as below

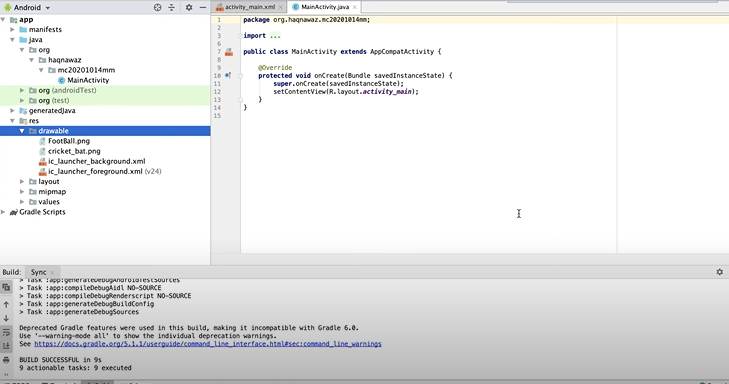




**Step02**: We added the article text in string.xml fil in values folder as follow

## **Lecture 11**

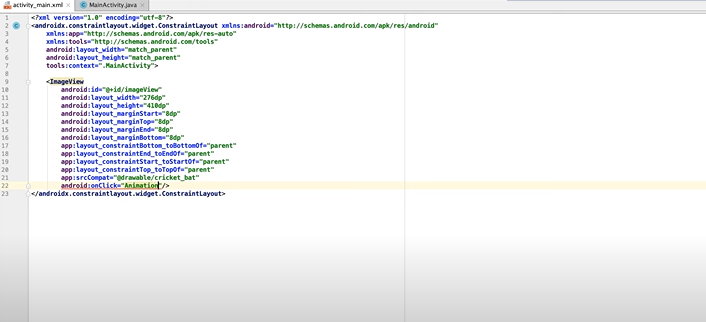
**Image Animation:**

**Step01:** First we add images in drawable folder as follow.

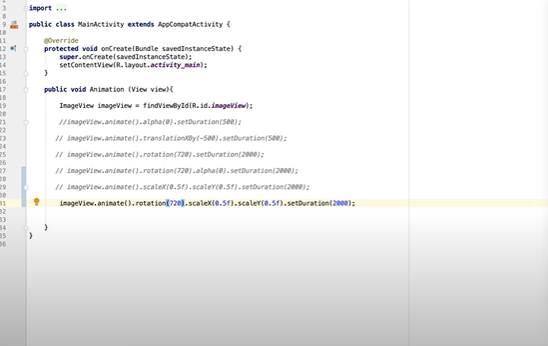
**Step02:** Add image view in constraint layout and set the constraints.

**Step03:** Define Button for animation.

**Step04:** Define Function for animation.



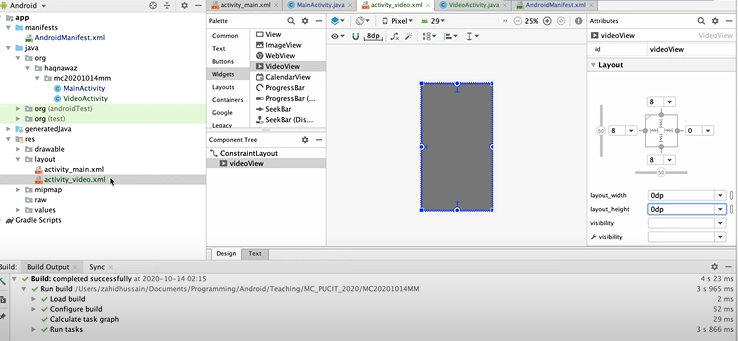
In this way we can apply many animations to image.



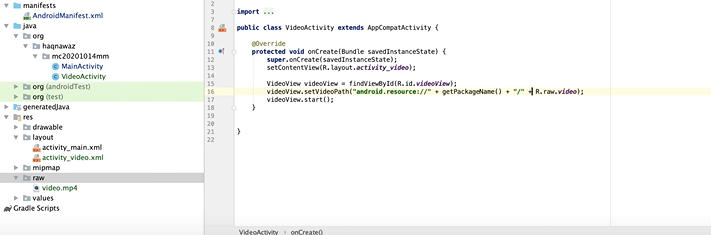
## **Lecture 12**

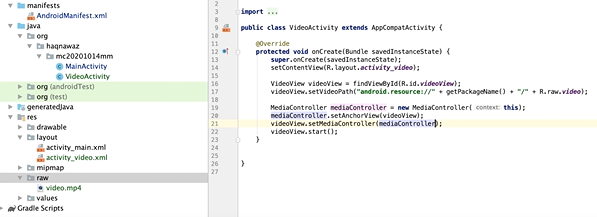
**Video Animation:**

**Step01:** In previous project add new activity and add video view in xml of that activity as follow.



**Step02:** Put video file in res/raw folder and design activity code in java class as follow



**Step03:** Add media controller to control the video i.e lay/pause etc. as follow

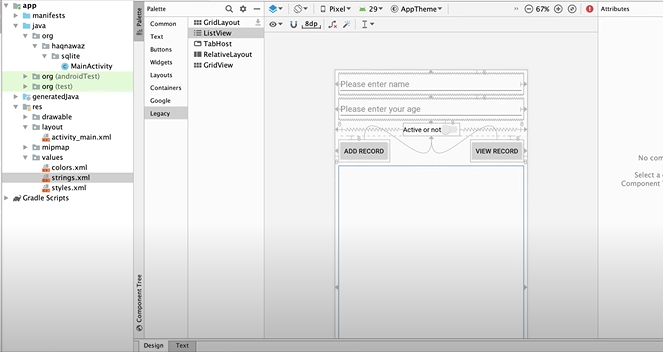
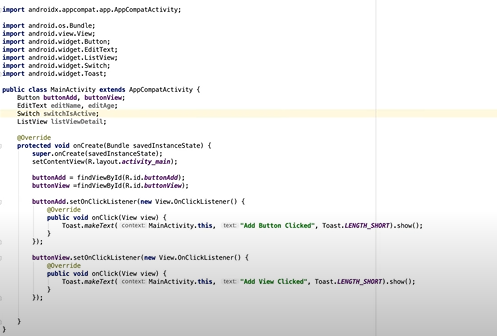
**Step04:** In similar way instead of using media controller, we use MediaPlayer class to play and control audio.

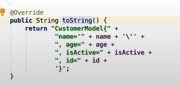
## **Lecture 13**

**Sqlite Database:**

**Step01**: First of all, we create the view as follow

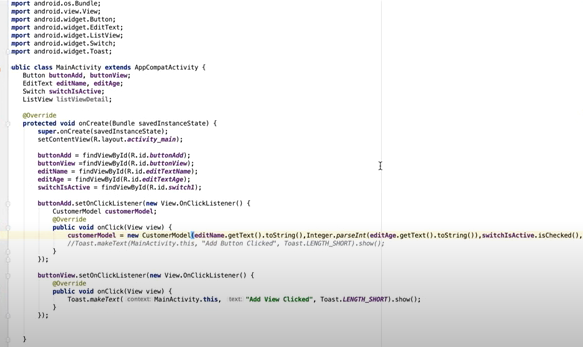
**Step02**: Then we add following code in java so that we can add/remove date=a in list view and toast for message.

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

**Step03**: Now we develop customer model class as follow and override the tostrind() method.

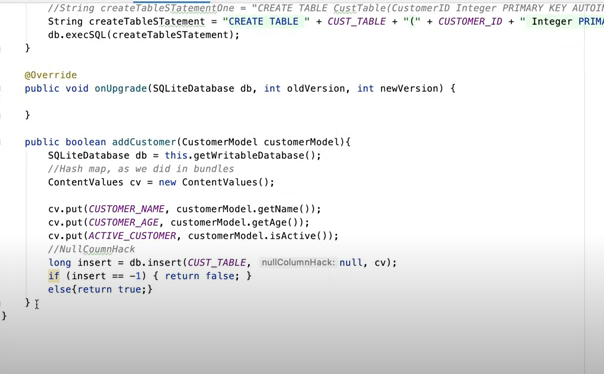


**Step04**: We also changed the activity java class as follow



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

**Step05**: Now we wrote DBHelper class as follow



## **Lecture 14**

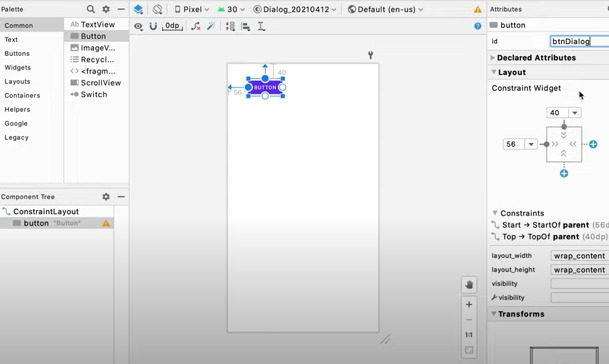
**Dialogs**

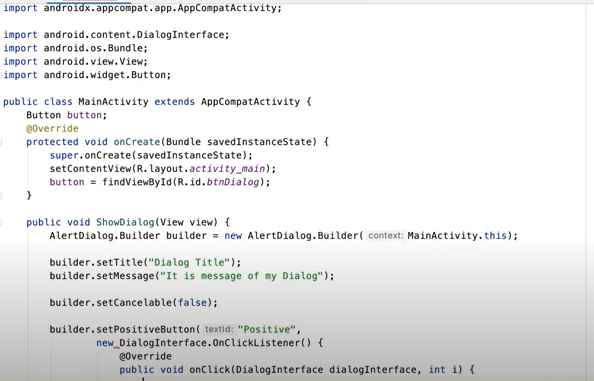
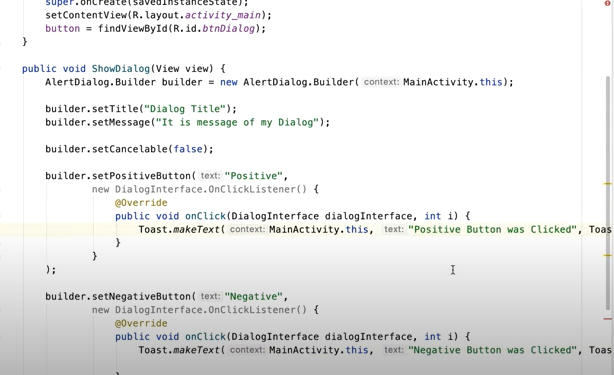
**Step01:** First of all, we design the xml view as follow.

**Step02:** Now we change design in code section

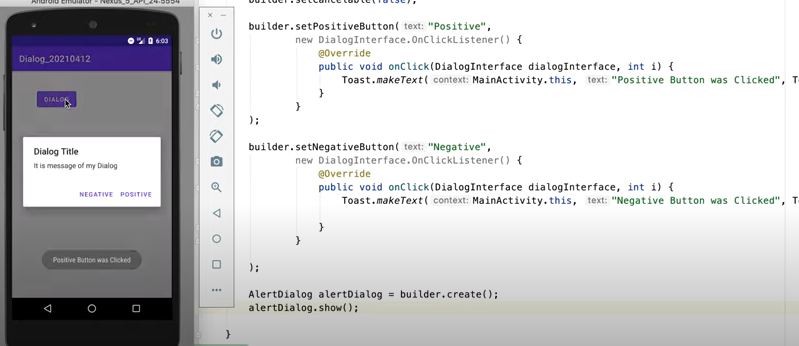


**Step03:** Now we change java code section in main activity

-

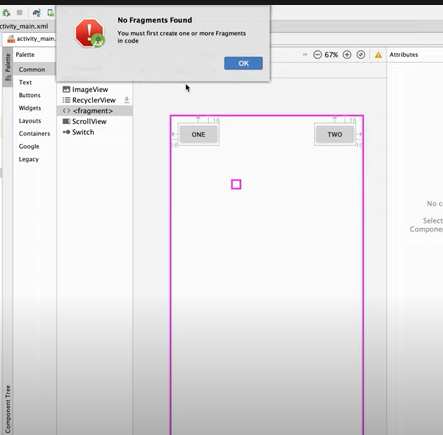
-

## **Lecture 15**

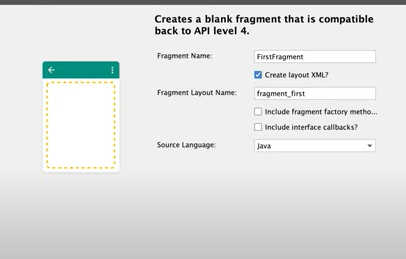
**Fragments:**

**Step01:** Create a new project with empty main activity.

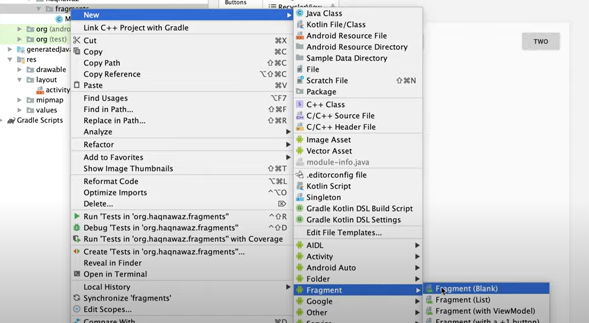
**Step02:** Add 2 buttons in xml view and try to add fragments as follow

-

**Step03:** To resolve the error create the fragments as follow



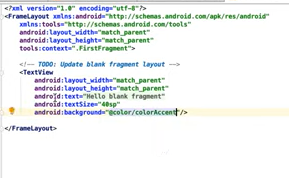
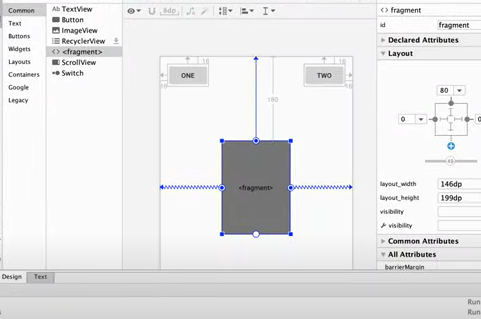
**Step04:** Manage the xmls of fragments as follow



**Step04:** Placement of fragments in view as follow

**Step05:**  Add code behind buttons in main activity java

-

-

## **Lecture 16**

**Notifications**

**Step01:** After creating the project, we add dependency in built.gradle file as follow

**Step02:** now add following code in main java file as follow.



-

**Step03:**  Add the channel code as follow

