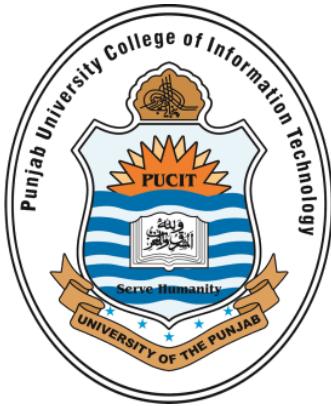


MOBILE COMPUTING-PROGRESS FILE



Submitted to:

Sir Haq Nawaz

Submitted by:

Amber Shafique (Bsef18a019)

LECTURE#1

❖ VCS (Version Control System):

A **version control system** is a kind of software that helps the developer team to efficiently communicate and manage track of all the changes that have been made to the source code along with the information like who made and what change has been made.

- Version control software keeps track of every modification to the code in a special kind of database.
- Using a VCS also generally means that if you screw things up or lose files, you can easily recover

Benefits:-

1. **Code Synchronizing:** Same code available to all members on different repositories connected to a central repository. All the changes in the files are tracked under the central repository. The central repository includes all the information of versioned files, and list of users that check out files from that central place using **VCS**.
2. **Code Testing :**VCS helps to test changes without losing the original version of the application.
3. **Revert:** VCS allows us to revert back to previous versions of the file because it separately maintains each version of file.

❖ Git :

Git is software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development.

Step-0

1.1 GitHub Account:

Sign up to github.com and create a new account.

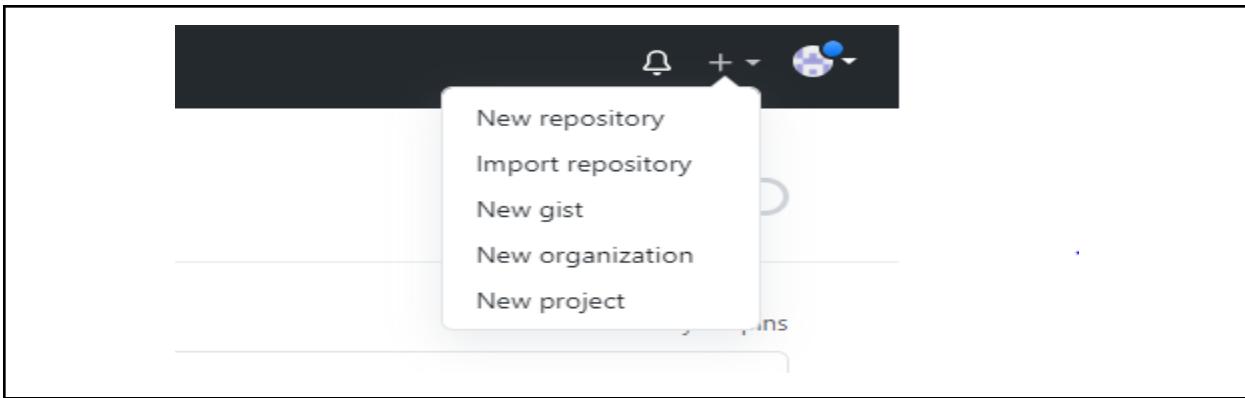
1.2 Git CMD:

Install git cmd and configure it on your PC.

Step-1

1.1 Create a new Repository:

1.1.1 Click on add new repository button on your GitHub account.



1.1.2 Then provide details of your repository => Name, Description, Type.

Public Repository: It is visible to every user on your GitHub.

Private Repository: It is only available to the repository owner.

1.1.3 Click on the create repository button.

The screenshot shows the "Create repository" form on GitHub. The "Owner" field is set to "amber-shafique". The "Repository name" field contains "VCS". The "Description (optional)" field contains "Mobile Computing Lecture#1". The "Public" radio button is selected, with the note: "Anyone on the internet can see this repository. You choose who can commit." The "Private" radio button is also present with its note. Under "Initialize this repository with:", there are three checkboxes: "Add a README file", "Add .gitignore", and "Choose a license". Each checkbox has a descriptive subtitle below it. At the bottom is a large green "Create repository" button.

Owner * Repository name *

amber-shafique / VCS ✓

Great repository names are short and memorable. Need inspiration? How about [didactic-octo-adventure](#)?

Description (optional)

Mobile Computing Lecture#1

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

Add a README file
This is where you can write a long description for your project. [Learn more](#).

Add .gitignore
Choose which files not to track from a list of templates. [Learn more](#).

Choose a license
A license tells others what they can and can't do with your code. [Learn more](#).

Create repository

1.2 Repository URL : On creating a new repository GitHub will provide its url that is used to uniquely identify our repository on GitHub.

Quick setup — if you've done this kind of thing before

 Set up in Desktop

or

HTTPS

SSH

<https://github.com/amber-shafique/VCS.git>

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a

LECTURE#2

❖ Git Clone:

- It is primarily used to point to an existing repository and make a clone or copy of that repository in a new directory, at another location.
- Makes connection between **local repository**(on PC) and **central repository** (on GitHub).

Step-1

Open **git CMD** and go to that folder path (using **cd**) where you want to clone the repository.

Step-2

Run the following command and give the **git url** of the repository you want to clone locally.

git clone URL

```
C:\Users\HP\Desktop>cd Git
C:\Users\HP\Desktop\Git>git clone https://github.com/amber-shafique/VCS.git
Cloning into 'VCS'...
warning: You appear to have cloned an empty repository.

C:\Users\HP\Desktop\Git>
```

Repository cloned Locally (see using **dir**)

```
C:\Users\HP\Desktop>dir
Volume in drive C has no label.
Volume Serial Number is F8F2-04AE

Directory of C:\Users\HP\Desktop\Git

04/14/2021  12:25 AM    <DIR>      .
04/14/2021  12:25 AM    <DIR>      ..
04/14/2021  12:21 AM    <DIR>      vcs
              0 File(s)   0 bytes
              3 Dir(s)  40,968,900,608 bytes free
```

❖ Add File:

Add locally created file to the central(online) repository.

Step-1

Creating file in local repository.



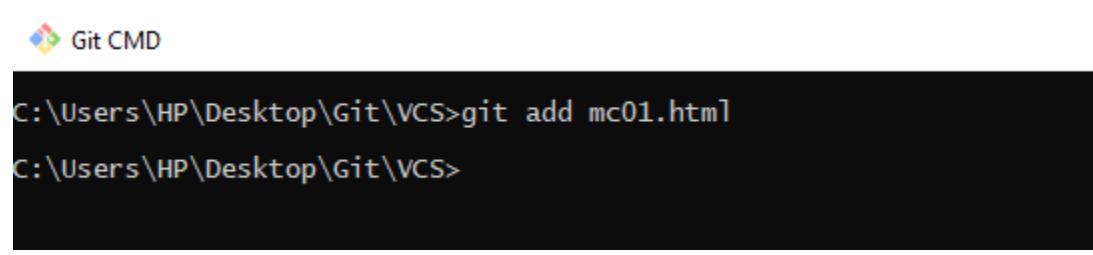
```
C: > Users > HP > Desktop > mc01.html > html > body
1   <!DOCTYPE html>
2   <html>
3   <head>
4       <title>Mobile Computing</title>
5   </head>
6   <body>
7
8       <h1>Lecture-2</h1>
9       <p>Testing for GitHub</p>
10      </body>
```

Step-2

Run the following command and give the **File Name** you want to add on git repository.

```
git add FileName
```

No error, so file has been added successfully.



```
Git CMD
C:\Users\HP\Desktop\Git\VCS>git add mc01.html
C:\Users\HP\Desktop\Git\VCS>
```

❖ Add all files :

- [git add --all](#)
- [git add .](#)

❖ Add multiple files at same time:

```
git add FileName FileName FileName ...
```

❖ Git Commit

The "commit" command is used to save your changes and track messages.

Step-3

Run the following command and give the **Message** you want to get displayed with added file.

```
git commit -m "message"
```

Message=> to inform the users about the change.



Git CMD

```
C:\Users\HP\Desktop\Git\VCS>git add mc01.html
C:\Users\HP\Desktop\Git\VCS>git commit -m "First commit done successfully"
[master (root-commit) 6f5a409] First commit done successfully
 1 file changed, 11 insertions(+)
 create mode 100644 mc01.html
```

❖ Add and commit simultaneously

```
git commit -am "message"
```

❖ Git Push:

To push the changes made in file locally to server (online repository).

Step-4

Run the following command to complete the synchronization.

```
git push
```

Now we are synchronized with data available on local disk and online repository.



```
Git CMD  
C:\Users\HP\Desktop\Git\VCS>git push  
info: please complete authentication in your browser...  
Enumerating objects: 3, done.  
Counting objects: 100% (3/3), done.  
Delta compression using up to 8 threads  
Compressing objects: 100% (2/2), done.  
Writing objects: 100% (3/3), 345 bytes | 69.00 KiB/s, done.  
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0  
To https://github.com/amber-shafique/VCS.git  
 * [new branch]      master -> master
```

Check the status of the online repository by refreshing the GitHub Repository URL.



amber-shafique First commit done successfully

6f5a409 17 minutes ago 1 commit



mc01.html First commit done successfully

17 minutes ago

Help people interested in this repository understand your project by adding a README.

Add a README

Content in online and local repository is synchronized.

```
11 lines (10 sloc) | 146 Bytes  
  
1 <!DOCTYPE html>  
2 <html>  
3 <head>  
4     <title>Mobile Computing</title>  
5 </head>  
6 <body>  
7  
8     <h1>Lecture-2</h1>  
9     <p>Testing for GitHub</p>  
10 </body>  
11 </html>
```

❖ Git Status:

Gives the status of repository.

```
git status
```

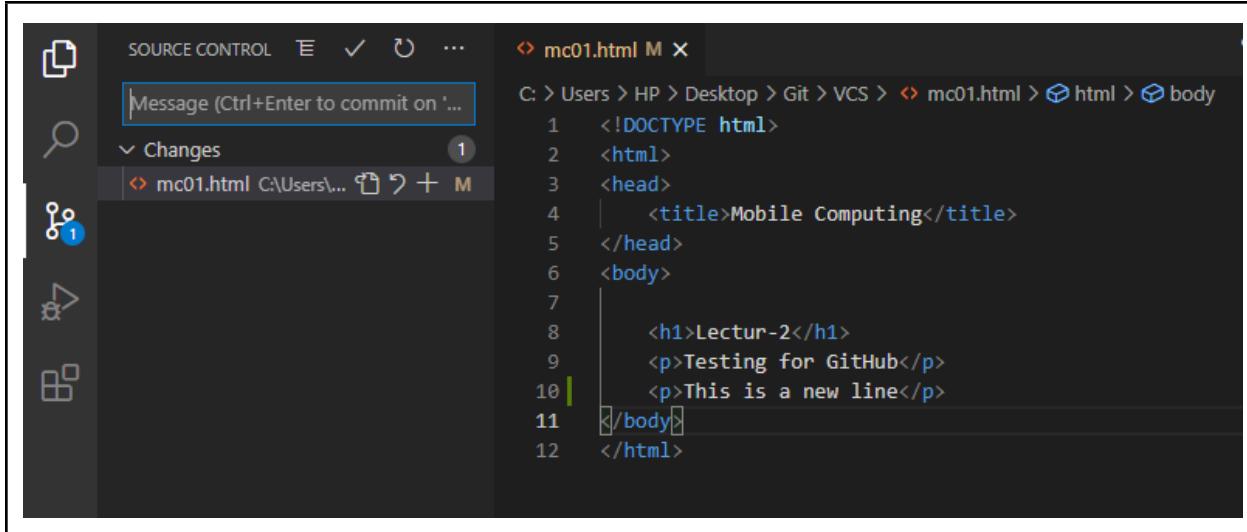
```
C:\Users\HP\Desktop\Git\VCS>git status
On branch master
Your branch is based on 'origin/master', but the upstream is gone.
  (use "git branch --unset-upstream" to fixup)

nothing to commit, working tree clean

C:\Users\HP\Desktop\Git\VCS>
```

Making change in file locally using IDE:

Whenever we make any change in file locally editor will show some change message



❖ Adding New Commit (To save locally made changes online) :

Run these previously discussed commands on git CMD in sequence.

Step-1

```
git add FileName
```

Step-2

```
git commit -m "message"
```

Step-3

```
git push
```



Git CMD

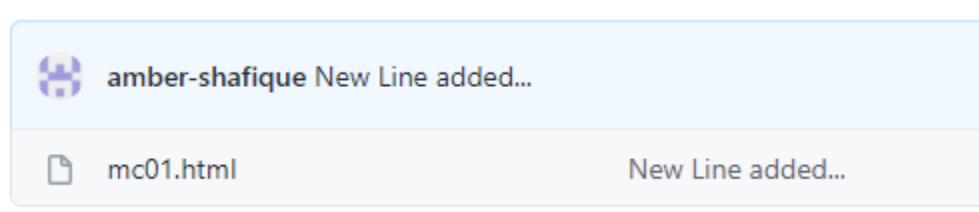
```
C:\Users\HP\Desktop\Git\VCS>git add mc01.html
C:\Users\HP\Desktop\Git\VCS>git commit -m "New Line added..."
[master 0381540] New Line added...
 1 file changed, 1 insertion(+)

C:\Users\HP\Desktop\Git\VCS>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 319 bytes | 106.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/amber-shafique/VCS.git
 6f5a409..0381540 master -> master

C:\Users\HP\Desktop\Git\VCS>
```

Track of Changes:

To preview change online refresh the repository url and see the changes have been made.

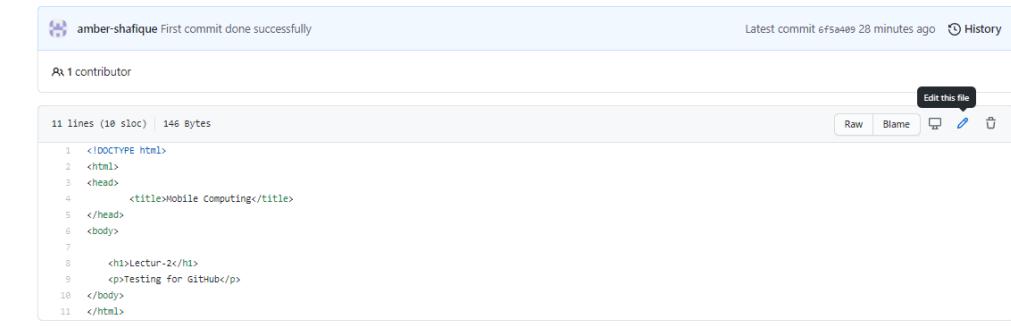


The screenshot shows a GitHub repository page. At the top, there's a commit message from 'amber-shafique' with the title 'New Line added...'. Below it, a file named 'mc01.html' is listed with the same commit message 'New Line added...'.

❖ Editing a file online in github repository:

Step-1

Edit the file online using the edit button on file.



The screenshot shows the GitHub file editor for 'mc01.html'. The file content is as follows:

```
11 lines (10 sloc) | 146 Bytes
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Mobile Computing</title>
5 </head>
6 <body>
7
8   <h1>Lecture-2</h1>
9   <p>Testing for GitHub</p>
10 </body>
11 </html>
```

At the bottom right of the code editor, there is a button labeled 'Edit this file'.

Step-2

Save commit changes.

The screenshot shows a 'Commit changes' dialog box. At the top, there's a text input field with placeholder text 'Changes made online...'. Below it is another text input field with placeholder text 'Add an optional extended description...'. Underneath these fields are two radio button options: one selected (blue outline) labeled 'Commit directly to the master branch.' and another unselected (grey outline) labeled 'Create a new branch for this commit and start a pull request. L'. At the bottom left is a green 'Commit changes' button, and at the bottom right is a red 'Cancel' button. Below the dialog box, a preview area shows a user profile icon, the name 'amber-shafique', and the text 'Changes made online...'. It also indicates '1 contributor'.

❖ Git Pull:

To get Changes from GitHub to local repository.

Step-1

After making changes online, run the following command on git CMD.

```
git push
```

Changes have been made to local repository.

Git CMD

```
C:\Users\HP\Desktop\Git\VCS>git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 789 bytes | 39.00 KiB/s, done.
From https://github.com/amber-shafique/VCS
  0381540..80df6ba master      -> origin/master
Updating 0381540..80df6ba
Fast-forward
  mc01.html | 5 +---+
  1 file changed, 4 insertions(+), 1 deletion(-)
```

```
C:\Users\HP\Desktop\Git\VCS>
```

❖ Version Hierarchy:

File showing with latest commit.

- **Red Highlighted lines:** The lines that have been deleted.
- **Green Highlighted lines:** The lines that have been changed.

```
5      5      </head>
6      6      <body>
7      7
8      -      <h1>Lectur-2</h1>
8      +      <h1>Lectur-2 changed by git</h1>
9      9      <p>Testing for GitHub</p>
10     10
11     11      <p>This is a new line</p>
```

Parents=> previous versions of file.

1 parent 80df6ba commit

LECTURE#3

❖ Merge Conflicts:

A merge conflict is an event that takes place when Git is unable to automatically resolve differences in code between two commits. Git can merge the changes automatically only if the commits are on different lines or branches.

Step-1

Changes in lin-8 online.

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>Mobile Computing</title>
5  </head>
6  <body>
7
8      <h1>Lectur-2 changed by git</h1>
^          +-----+-----+
```

Step-2

Changes in same line-8 locally.

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4  |  <title>Mobile Computing</title>
5  </head>
6  <body>
7
8 |  <h1>Lectur-2 chaged locally</h1>
9  <p>Testing for GitHub</p>
```

Step-3

Run commands step by step to save file changes in git CMD.

Merge Conflict has occurred because we are trying to change the same line locally and in online repository.

```
Git CMD

C:\Users\HP\Desktop\Git\VCS>git add mc01.html
C:\Users\HP\Desktop\Git\VCS>git commit -m "Git Merge Conflict"
[master 2b363d5] Git Merge Conflict
 1 file changed, 1 insertion(+), 1 deletion(-)

C:\Users\HP\Desktop\Git\VCS>git push
To https://github.com/amber-shafique/VCS.git
 ! [rejected]          master -> master (fetch first)
error: failed to push some refs to 'https://github.com/amber-shafique/VCS.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.

C:\Users\HP\Desktop\Git\VCS>git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 684 bytes | 25.00 KiB/s, done.
From https://github.com/amber-shafique/VCS
 80df6ba..b2332d9  master      -> origin/master
Auto-merging mc01.html
CONFLICT (content): Merge conflict in mc01.html
Automatic merge failed; fix conflicts and then commit the result.
```

Step-4

Open the file in IDE to see the conflict and given options.

```
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
8 <<<<< HEAD (Current Change)
9 <h1>Lectur-2 chaged locally</h1>
10 =====
11 <h1>Lectur-2 changed by git</h1>
12 >>>>> b2332d9db23e90b369df00877e17db07d3dd6303 (Incoming Change)
13 <p>Testing for GitHub</p>
```

Step-5

Accept Change and resolve the merge conflict and **again** Run commands step by step to save file changes in git CMD.

Now changes have been saved successfully as conflict is resolved.

```
Git CMD
C:\Users\HP\Desktop\Git\VCS>git add mc01.html
C:\Users\HP\Desktop\Git\VCS>git commit -m "Git Merge Conflict Resolved"
[master 7d7438d] Git Merge Conflict Resolved
C:\Users\HP\Desktop\Git\VCS>git push
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 678 bytes | 339.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/amber-shafique/VCS.git
  b2332d9..7d7438d master -> master
C:\Users\HP\Desktop\Git\VCS>
```

❖ Git Logs:

To see the details of all commits.

git log

```
Git CMD - "C:\Program Files\Git\cmd\git.exe" log
C:\Users\HP\Desktop\Git\VCS>git log
commit 7d7438d3afbd57dcf9fa087cd89dd524c93e276a (HEAD -> master, origin/master)
Merge: 2b363d5 b2332d9
Author: amber-shafique <bsef18a019@pucit.edu.pk>
Date:   Wed Apr 14 02:25:18 2021 +0500

    Git Merge Conflict Resolved

commit 2b363d5a3fa0c2651c944f5acf3b105deab9d3c2
Author: amber-shafique <bsef18a019@pucit.edu.pk>
Date:   Wed Apr 14 02:15:59 2021 +0500

    Git Merge Conflict

commit b2332d9db23e90b369df00877e17db07d3dd6303
Author: amber-shafique <81466246+amber-shafique@users.noreply.github.com>
Date:   Wed Apr 14 02:08:41 2021 +0500

    Merge Conflict by online changes...

commit 80df6ba5cf23b9e65b5e7cfb318838d2e56a367c
Author: amber-shafique <81466246+amber-shafique@users.noreply.github.com>
```

❖ Remove a File:

Files already present

```
C:\Users\HP\Desktop\Git\MC-1>dir
Volume in drive C has no label.
Volume Serial Number is F8F2-04AE

Directory of C:\Users\HP\Desktop\Git\MC-1

04/14/2021  03:44 PM    <DIR>        .
04/14/2021  03:44 PM    <DIR>        ..
04/14/2021  02:42 AM            6 file1.txt
04/14/2021  03:44 PM            6 file2.txt
04/14/2021  03:06 AM            7 file3.txt
               3 File(s)       19 bytes
               2 Dir(s)   41,993,297,920 bytes free
```

Step-1

To remove file type the following command on git CMD, and give the name of file you want to delete.

```
git rm FileName
```

File deleted locally

```
Git CMD

C:\Users\HP\Desktop\Git\MC-1>git rm file1.txt
rm 'file1.txt'

C:\Users\HP\Desktop\Git\MC-1>
```

Now to update delete changes to git:

Step-2

```
git add .
```

Step-3

```
git commit -m "message"
```

Step-4

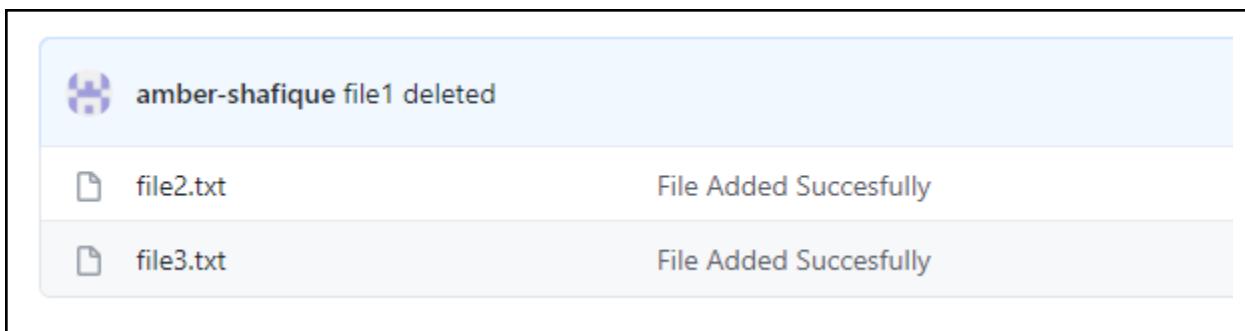
```
git push
```

```
C:\Users\HP\Desktop\Git\MC-1>git add .
C:\Users\HP\Desktop\Git\MC-1>git commit -m "file1 deleted"
[master 37a39ee] file1 deleted
 1 file changed, 1 deletion(-)
 delete mode 100644 file1.txt

C:\Users\HP\Desktop\Git\MC-1>git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 264 bytes | 88.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/amber-shafique/MC-1.git
  f837866..37a39ee master -> master

C:\Users\HP\Desktop\Git\MC-1>
```

Preview delete changes on GitHub



The screenshot shows a GitHub commit history for a repository named 'MC-1'. The commit message is 'file1 deleted'. Below the commit, there are two entries: 'file2.txt' and 'file3.txt', both listed as 'File Added Successfully'. This indicates that while 'file1.txt' was deleted, two new files were added.

❖ Branching:

Branching is the practice of creating copies of programs or objects in development to work in parallel versions, retaining the original and working on the branch or making different changes to each.

Step-1

Git Branch:

Run the following command on cmd to check available branches.

```
git branch
```

Step-2

Git Checkout:

To create a new branch, type the following command on git CMD and give the name of branch.

```
git checkout -b newbranch
```

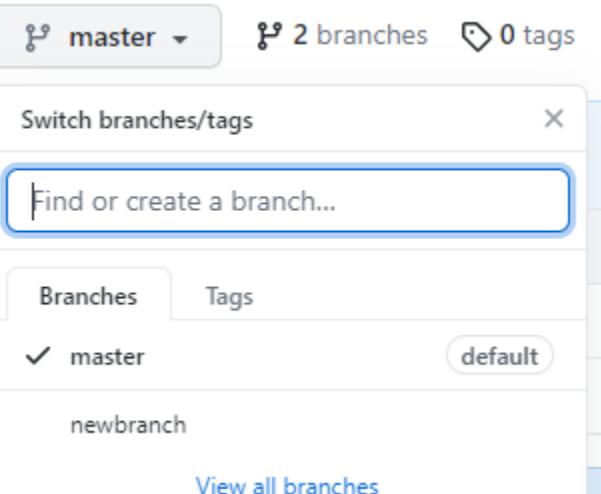
```
Git CMD
C:\Users\HP\Desktop\Git\MC-1>git branch
* master

C:\Users\HP\Desktop\Git\MC-1>git checkout -b newbranch
Switched to a new branch 'newbranch'

C:\Users\HP\Desktop\Git\MC-1>git branch
* master
* newbranch

C:\Users\HP\Desktop\Git\MC-1>
```

Branch Added on GitHub



Step-3

Git Checkout Master:

To navigate from one branch to another. (master=>branch name)

[git checkout master](#)

```
Git CMD
C:\Users\HP\Desktop\Git\MC-1>git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

C:\Users\HP\Desktop\Git\MC-1>
```

To Check Branching:

- Step-1** Add changes to a file on master branch.
- Step-2** Move to new branch and add file on the new branch.
- Step-3** Checkout file on both branches and see the changes accordingly on IDE.

New branch showing changes...

The screenshot shows two terminal windows. The left window is titled 'file2.txt' and displays a file with three lines of code. The right window is titled 'Git CMD' and shows the command 'git checkout master' being run, followed by a message indicating the switch to the 'master' branch and its status relative to 'origin/master'. Then, the command 'git checkout newbranch' is run, switching to the 'newbranch' branch and updating it from 'origin/newbranch'. The command prompt 'C:\Users\HP\Desktop\Git\MC-1>' is visible at the bottom of the CMD window.

Master branch showing previous version without changes...

The screenshot shows a terminal window with a file named 'file2.txt' containing three lines of code. The right side of the screen shows the command 'git checkout master' being run, which switches the branch to 'master' and updates it from 'origin/master'. The command prompt 'C:\Users\HP\Desktop\Git\MC-1>' is visible at the bottom.

❖ Merging:

Merging is Git's way of putting a forked history back together again. The git merge command lets you take the independent lines of development created by git branch and integrate them into a single branch.

Merge Branch:

To merge the changes in branches.

[Git merge branchname](#)

```
C:\Users\HP\Desktop\Git\MC-1>git merge newbranch
Removing file3.txt
Auto-merging file2.txt
CONFLICT (content): Merge conflict in file2.txt
CONFLICT (add/add): Merge conflict in BranchingPractice.html
Auto-merging BranchingPractice.html
Automatic merge failed; fix conflicts and then commit the result.

C:\Users\HP\Desktop\Git\MC-1>
```

❖ Delete Branch:

To delete a branch. NewBranch=>you want to delete.

Git branch -D NewBranch

Checking out branches before and after deletion...

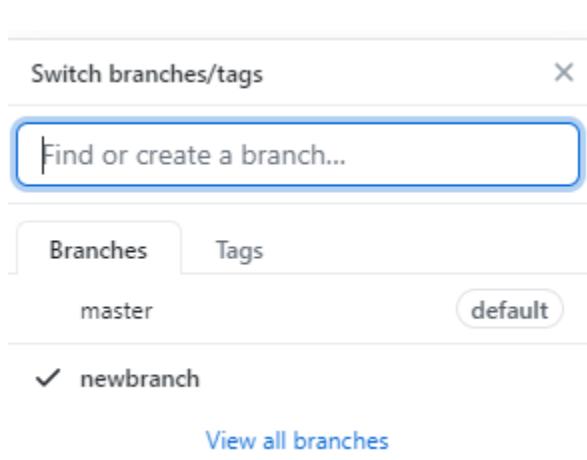
```
Git CMD
C:\Users\HP\Desktop\Git\MC-1>git branch
* master
  newbranch

C:\Users\HP\Desktop\Git\MC-1>git branch -D newbranch
Deleted branch newbranch (was ba2149a).

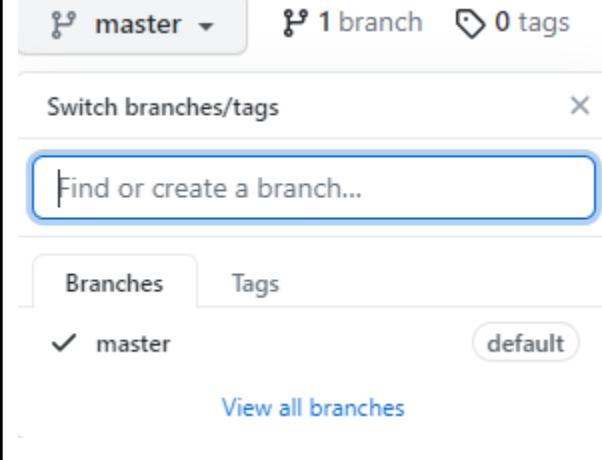
C:\Users\HP\Desktop\Git\MC-1>git branch
* master

C:\Users\HP\Desktop\Git\MC-1>
```

Before Deletion



After Deletion



LECTURE#4

❖ Android Studio:

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA. A unified environment where you can develop for all Android devices. Apply Changes to push code and resource changes to your running app without restarting your app.

Step-1

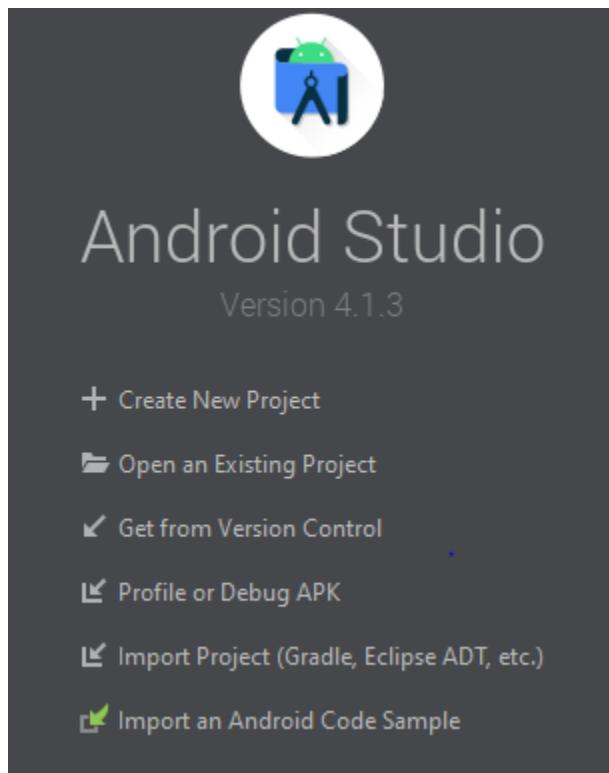
Installation:

Download any latest version of **Android Studio** and configure it on your PC.
[\(https://developer.android.com/studio\)](https://developer.android.com/studio)

Step-2

Create a New Project:

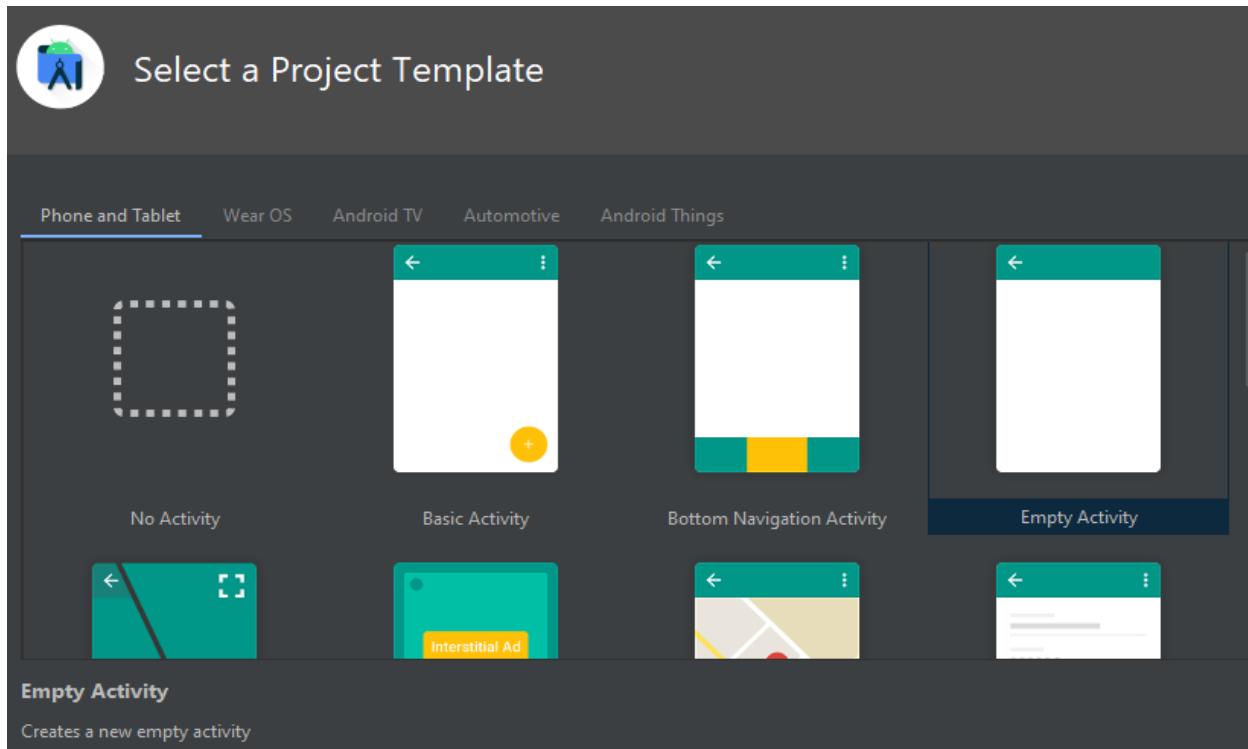
Open Android Studio and select the option to create a new project.



Step-3

Template Selection:

From templates choose Empty Activity.



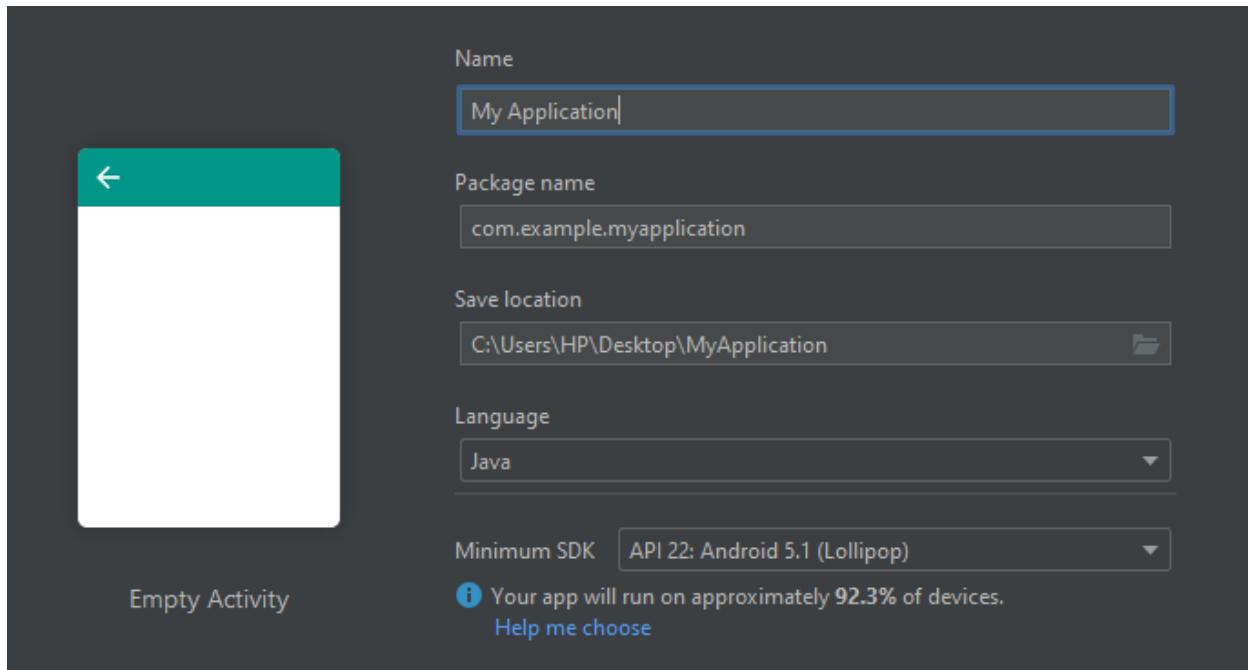
Step-4

Project Name:

Select a name and location for project for project.

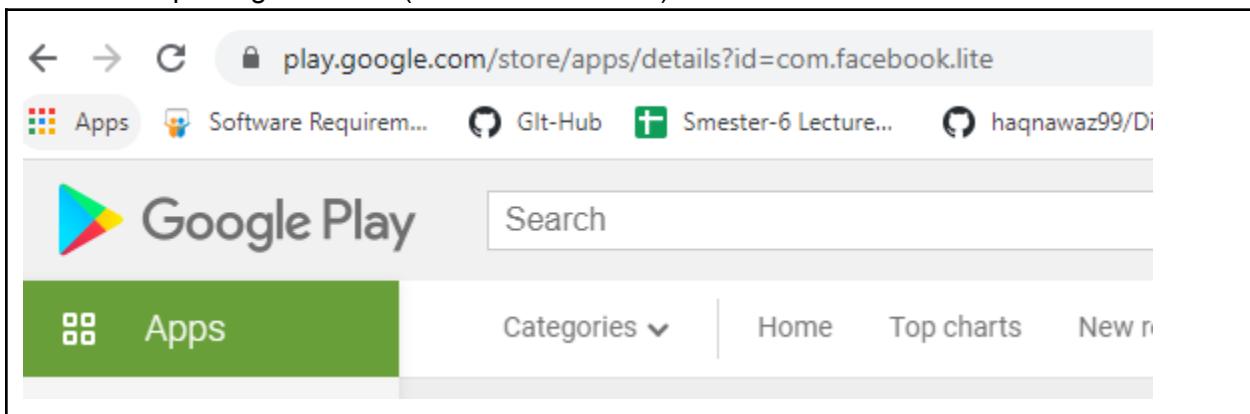
Package Name:

It is used to uniquely identify the APK file of our application and it should must be unique to publish the App on play store.

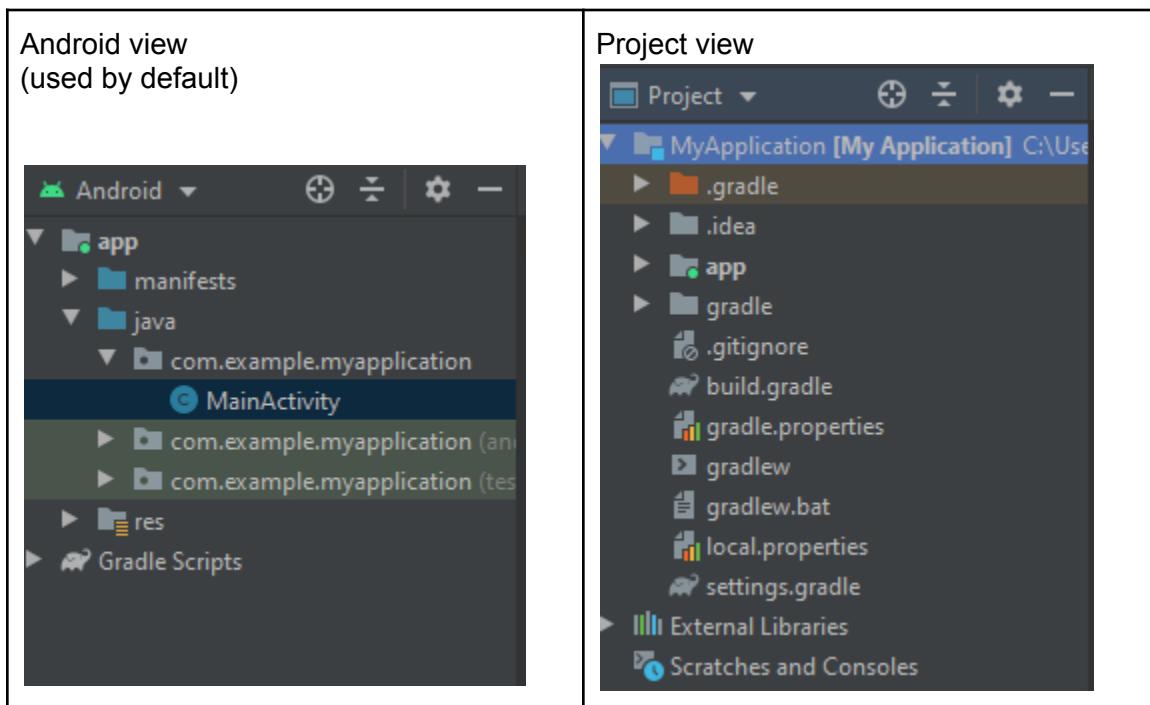


Package Name Example:

Here package name is (**com.facebook.lite**)

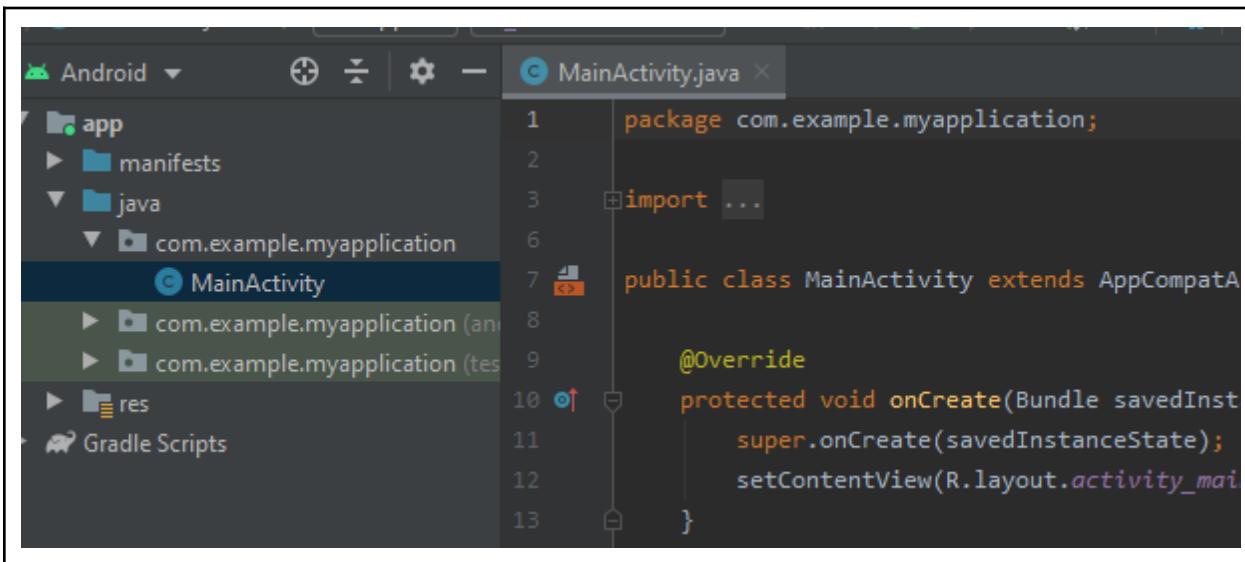


❖ Android Studio File Structure:



Files for coding:

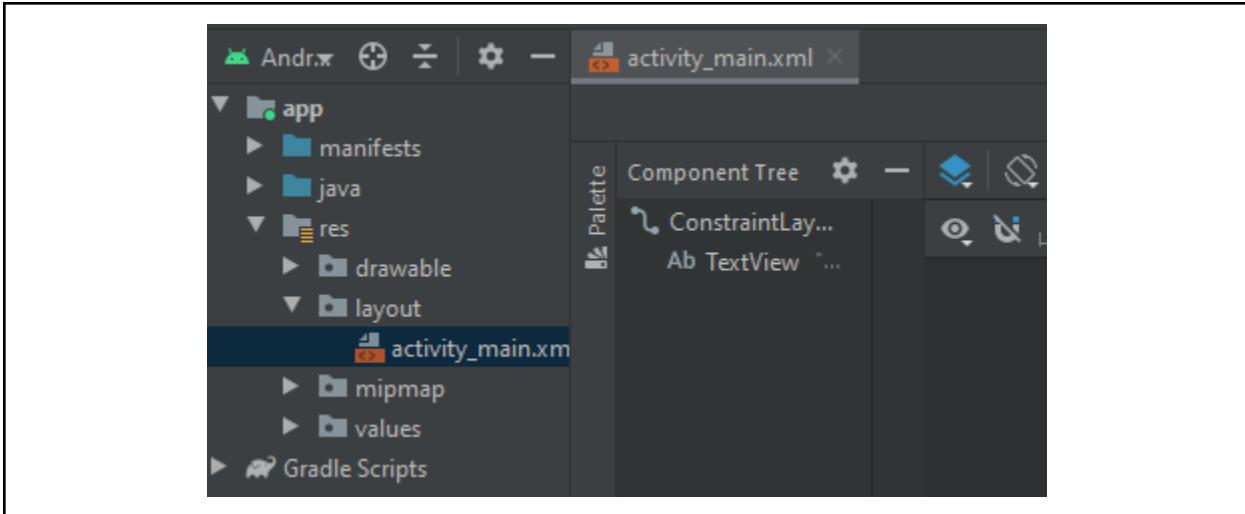
- **MainActivity.java** file for coding.



The screenshot shows the Android Studio interface with the Java editor open. The file path in the left sidebar is `app/java/com.example.myapplication/MainActivity.java`. The code in the editor is:

```
1 package com.example.myapplication;
2
3 import ...
4
5 public class MainActivity extends AppCompatActivity {
6
7     @Override
8     protected void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10        setContentView(R.layout.activity_main);
11    }
12 }
```

Activity_main.xml file for Graphical user interface(GUI).



Step-4

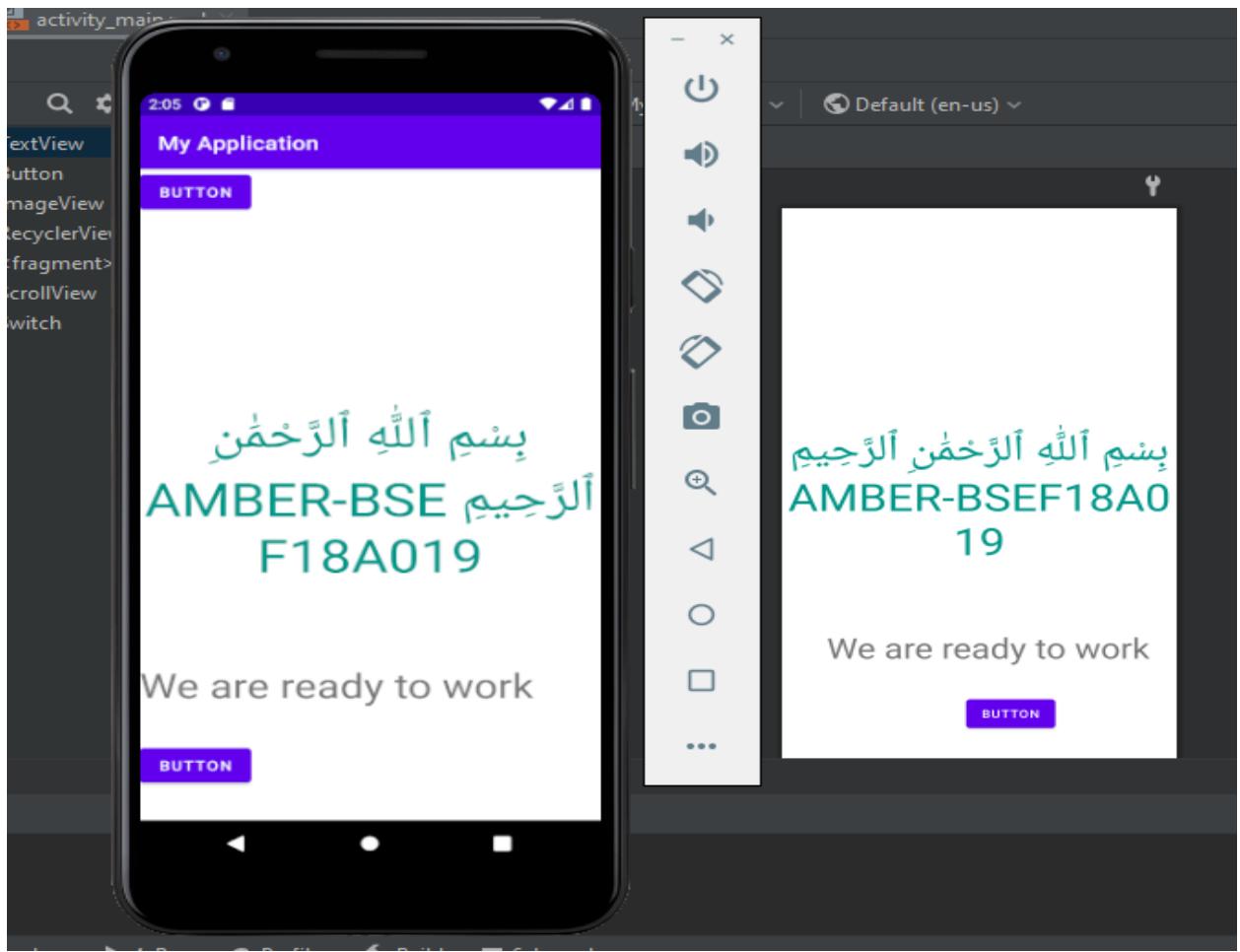
Select Virtual Device:

Select **Virtual Devices** from AVD(Android Virtual Device) Manager to view output.

Step-4

To view output:

Run the **Emulator** choosing any virtual device and see the output.



To Do list:

```
6
7  public class MainActivity extends AppCompatActivity {
8
9      @Override
10     protected void onCreate(Bundle savedInstanceState) {
11         super.onCreate(savedInstanceState);
12         //TODO: it is testing one
13         setContentView(R.layout.activity_main);
14
15         String name="Amber";
16         //It will be a comment
17         //TODO: it is testing two
18     }
19 }
```

TODO: Project Current File Scope Based Default Changelist

Found 2 TODO items in 1 file

com.example.myapplication 2 items

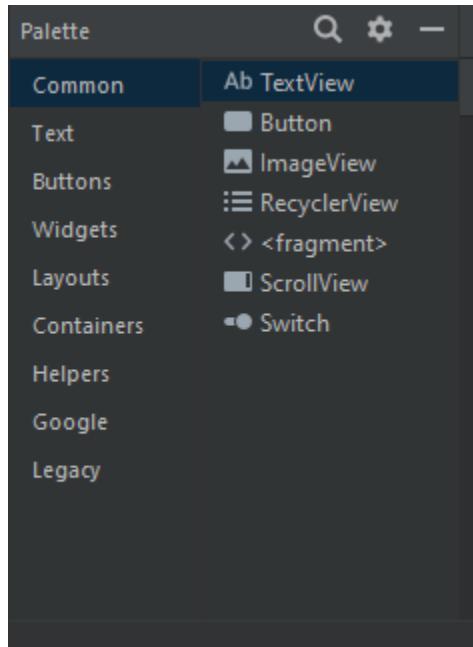
MainActivity.java 2 items

(12, 11) //TODO: it is testing one

(17, 11) //TODO: it is testing two

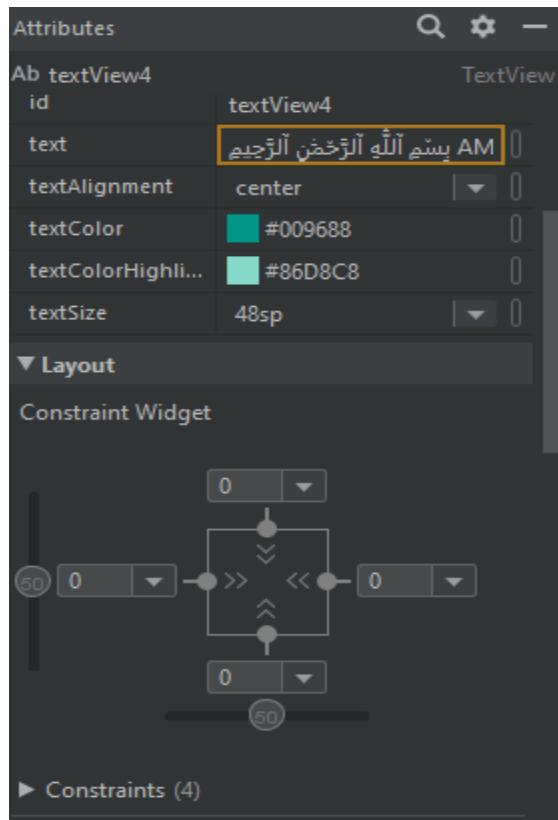
Design Palette:

We can add text,Button,image and other controls by using drag and drop from this palette.



Attributes:

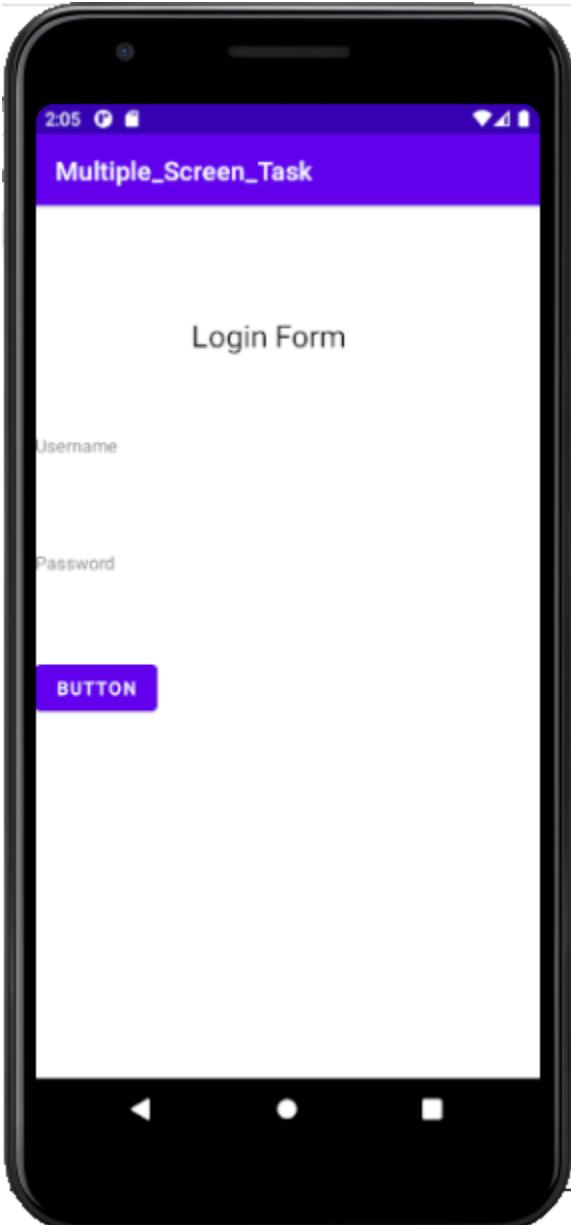
To set the values of elements added in xml.



❖ Multiple Screen Task:

Screen-1

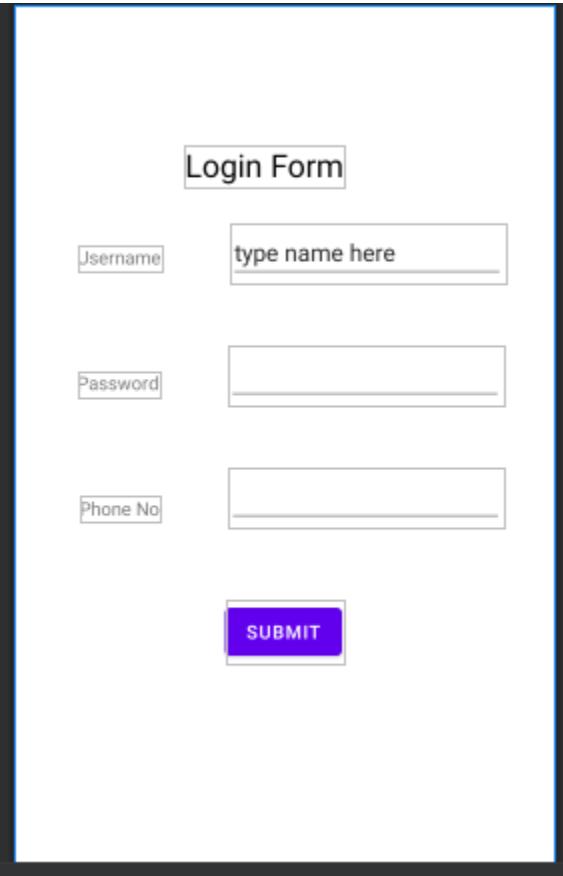
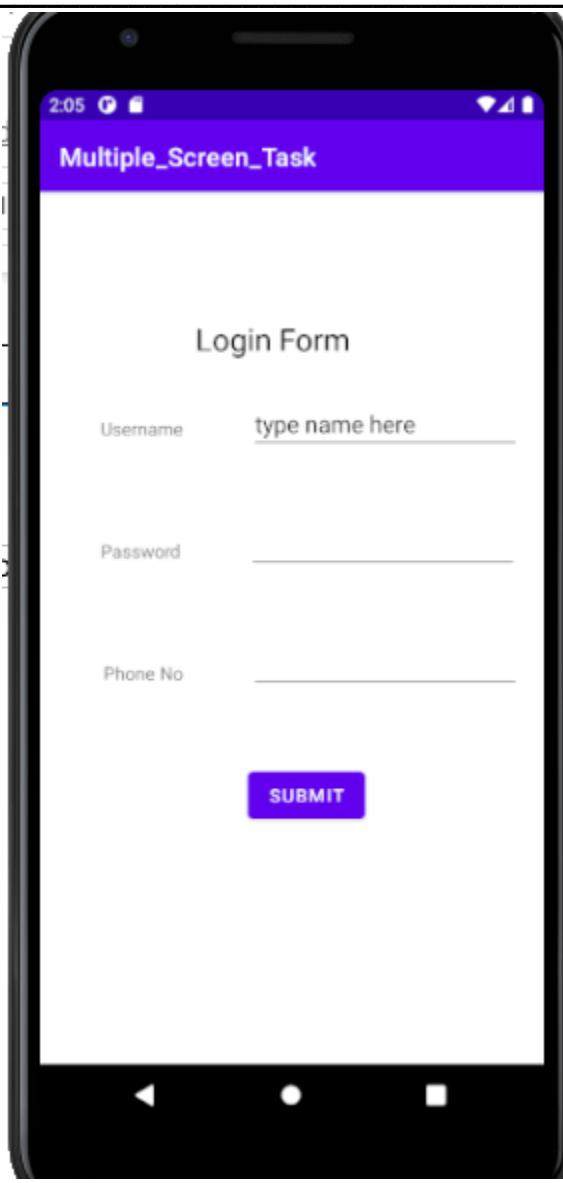
Step-1 Add elements of text box and buttons on the screen from palette by drag and drop.

XML View	Emulator View
<p>XML View content:</p> <p>Username</p> <p>Password</p> <p>Login Form</p> <p>BUTTON</p>	<p>Emulator View content:</p> <p>2:05</p> <p>Multiple_Screen_Task</p> <p>Login Form</p> <p>Username</p> <p>Password</p> <p>BUTTON</p> 

Screen-2

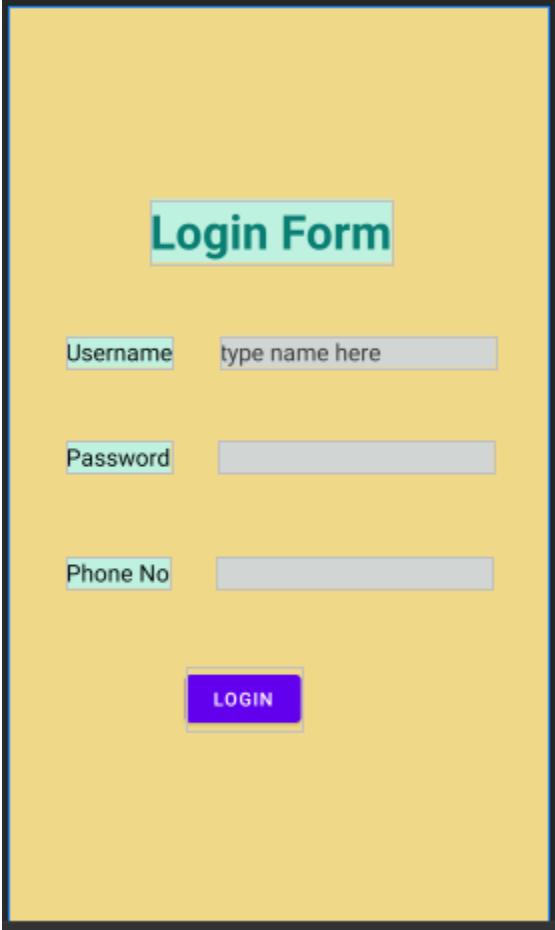
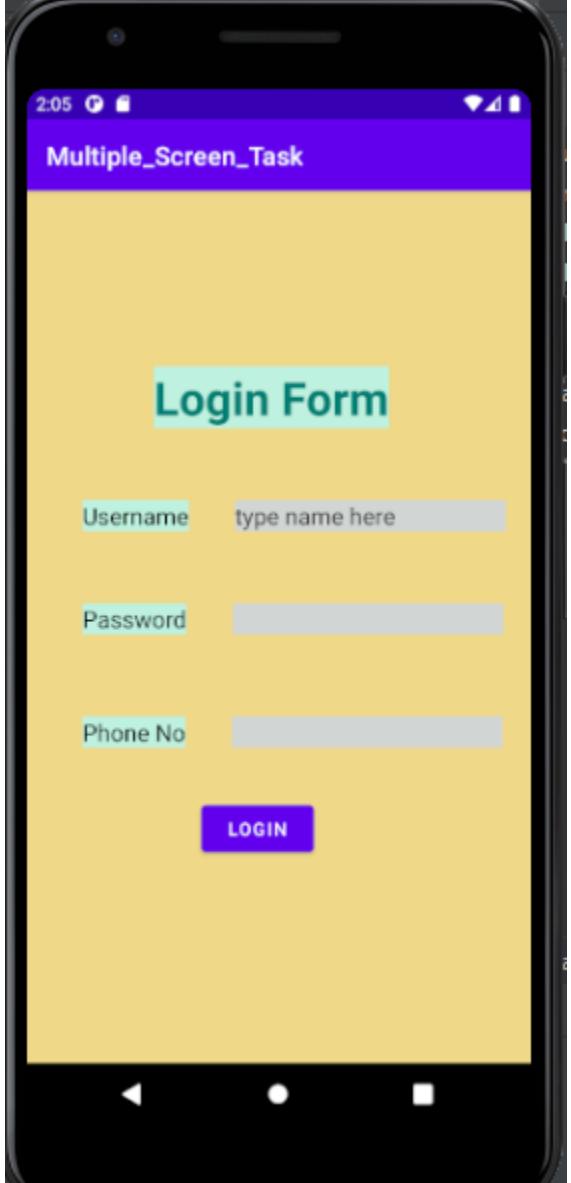
Step-2

Adding more elements of plain text and password to take input.

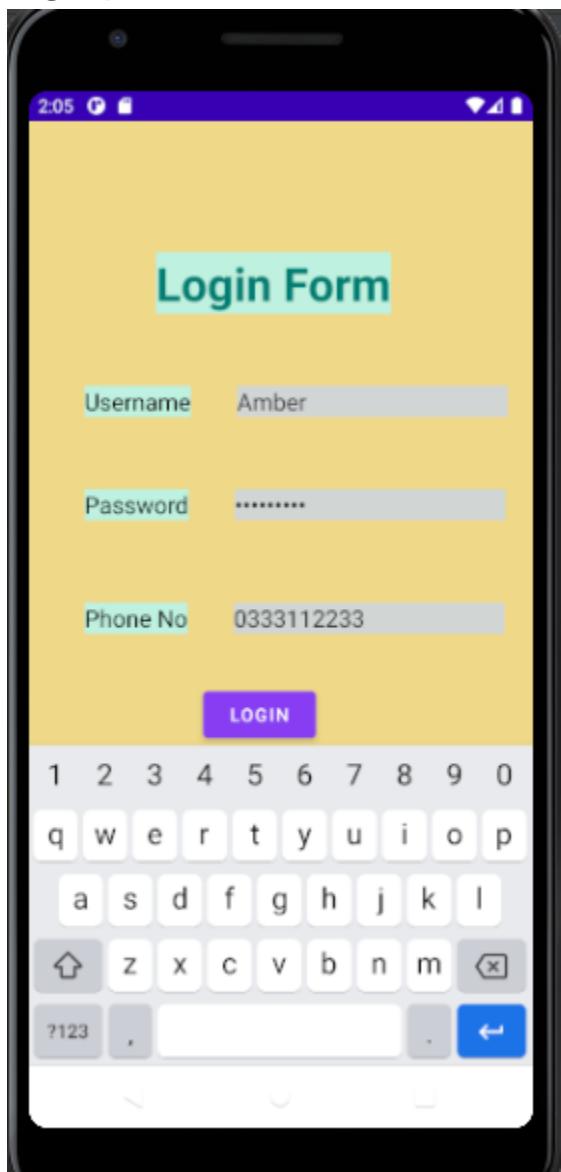
XML View	Emulator View
	

Screen-3

Step-3 Setting constraints and others attributes to each element to set the screen presentation.

XML View	Emulator View
 The XML View shows a yellow-themed login interface. It features a main title "Login Form" in a teal box at the top. Below it are three sets of labels and input fields: "Username" with a placeholder "type name here", "Password" with a placeholder "type name here", and "Phone No" with a placeholder "type name here". At the bottom is a purple "LOGIN" button.	 The Emulator View shows the same login interface as the XML View, but it is displayed on a virtual smartphone screen. The title "Login Form" is in a teal box. The input fields have light gray backgrounds. The "LOGIN" button is purple. The entire screen has a yellow background. The emulator status bar at the top shows the time as 2:05 and various connectivity icons.

Final Screen view taking input:



LECTURE#5

❖ View Group:

A ViewGroup is an **invisible container**. It is a special view that can contain other views (called children.)

- **Layouts:**

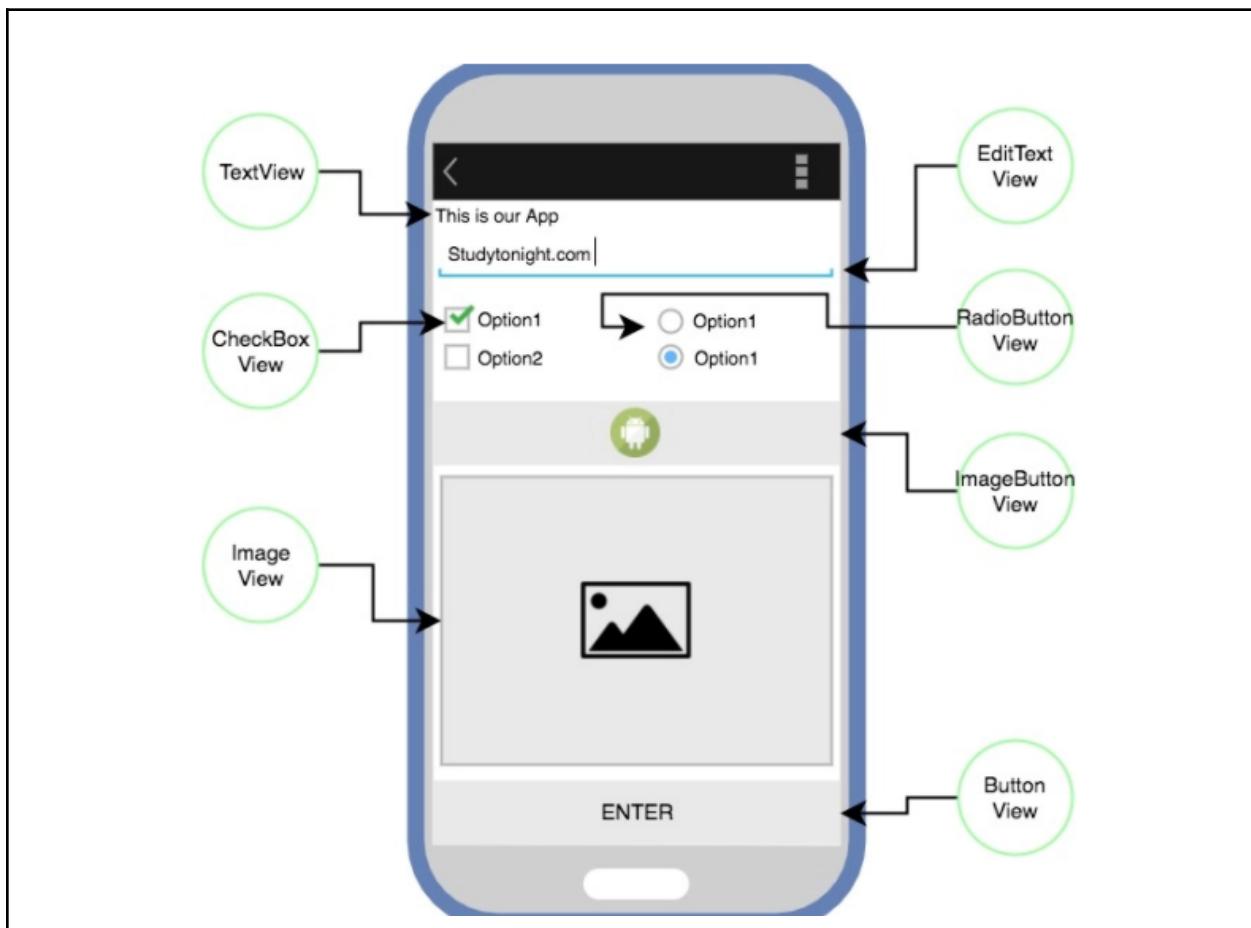
View group objects are called "**layouts**" can be one of many types that provide a different layout structure, such as LinearLayout or ConstraintLayout .

❖ View:

View is the basic building block of **UI**(User Interface) in android. View refers to the android. view class, which is the super class for all the GUI components like TextView, ImageView, Button etc. It draws something that user can see and interact with.

- **Widgets:**

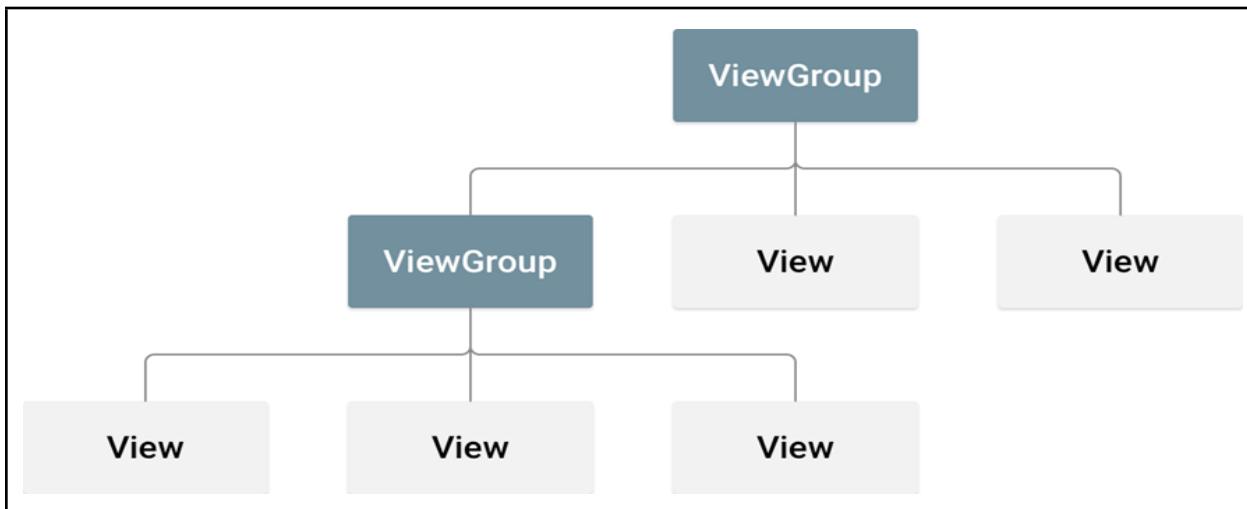
The View objects are usually called "**widgets**" and can be one of many subclasses, such as Button or TextView.



Root Element:

Each layout file must contain exactly one **root element**, which must be a **View or ViewGroup** object. Once you've defined the root element, you can add additional layout objects or widgets as child elements to gradually build a View hierarchy that defines your layout. For example, here's an XML layout that uses a vertical LinearLayout to hold a TextView and a Button.

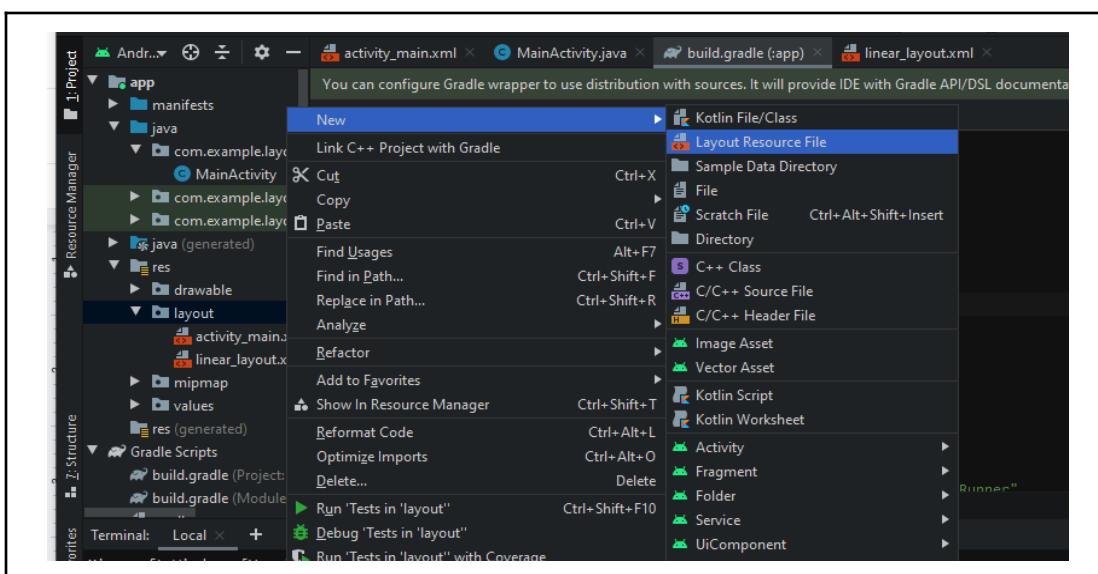
Constraint Layout is the root element by default.



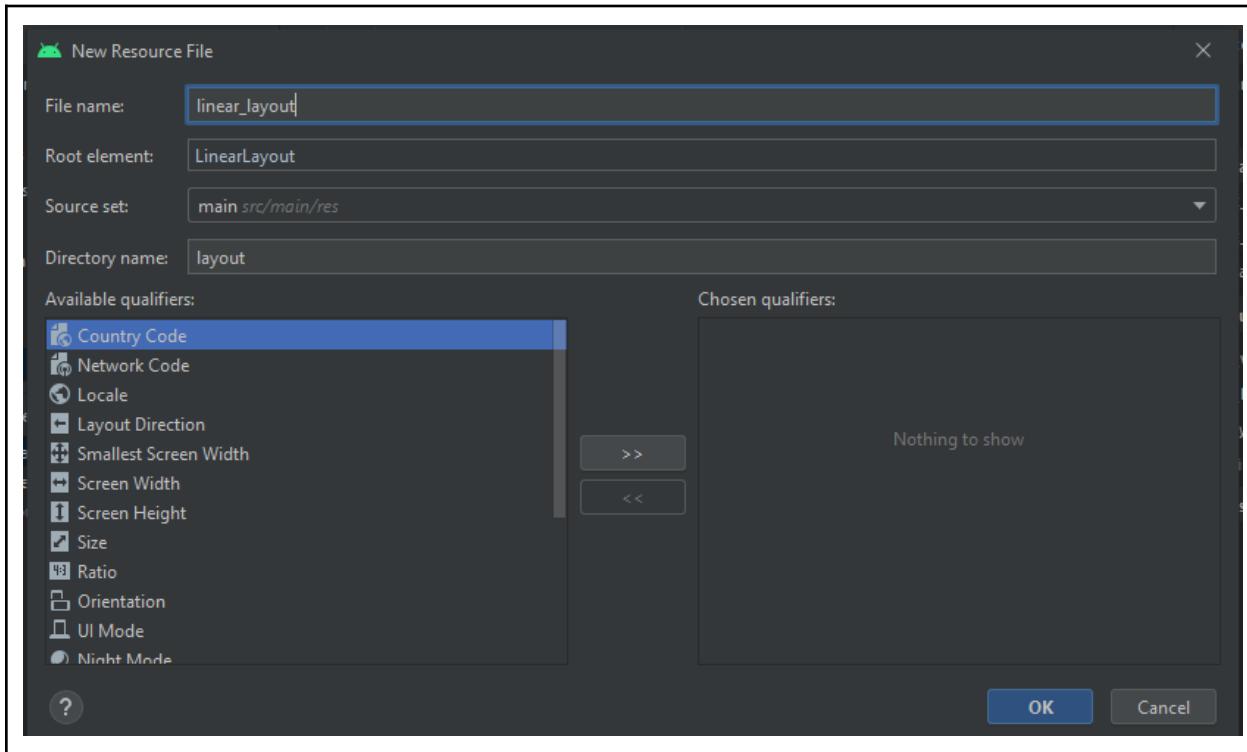
❖ Layouts Practice:

Step-1 Create a new project with empty template.

Step-2 Add a new resource file.



Step-3 Select the root element of **LinearLayout**.

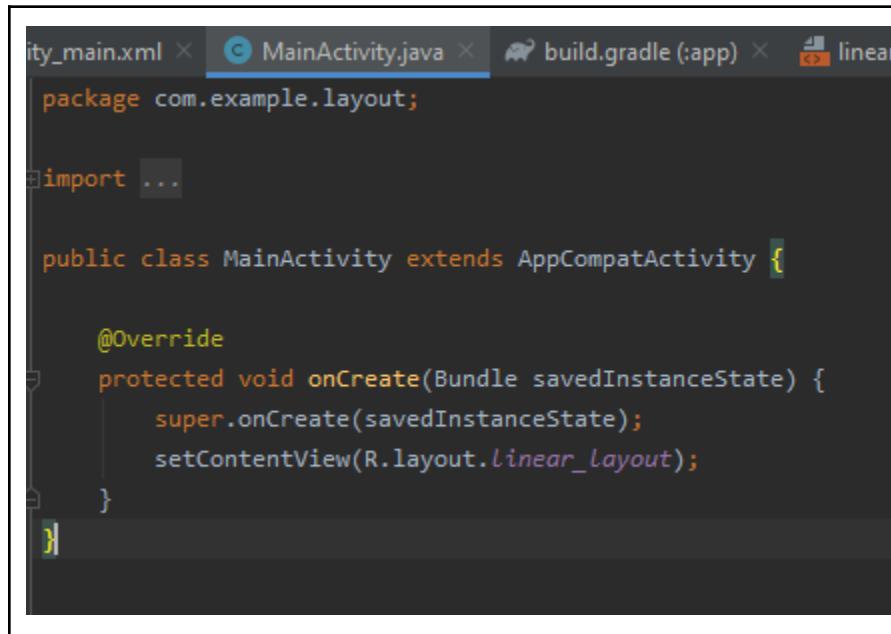


Step-4 Add a text view and button, and you will see that these elements are displayed in a linear layout on xml design view.



❖ Load XML Resources:

Step-5 In MainActivity.java file change the **setContentView(R.layout .__)**, to the layout file you want to run on emulator.



A screenshot of the Android Studio code editor. The current file is MainActivity.java, which contains the following code:

```
activity_main.xml × MainActivity.java × build.gradle (:app) × linear_layout.xml ×
package com.example.layout;

import ...

public class MainActivity extends AppCompatActivity {

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

Layout Parametres:

They are used by views to tell their parents how they want to be laid out.

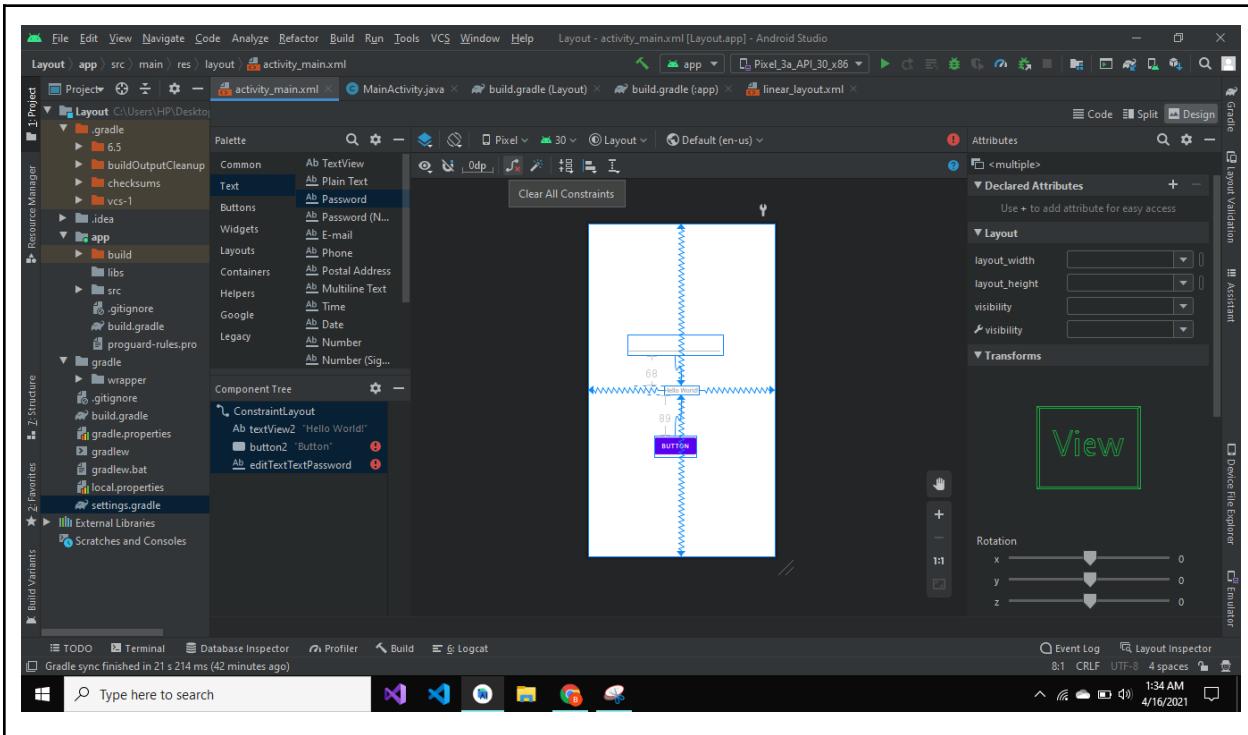
❖ Constraint Layout:

In Constraint Layout, you can connect all the view components to each other and place them on screen.

- To define a **view's position** in ConstraintLayout , you must add at least one horizontal and one vertical constraint for the view. Each constraint represents a connection or alignment to another view, the parent layout, or an invisible guideline.

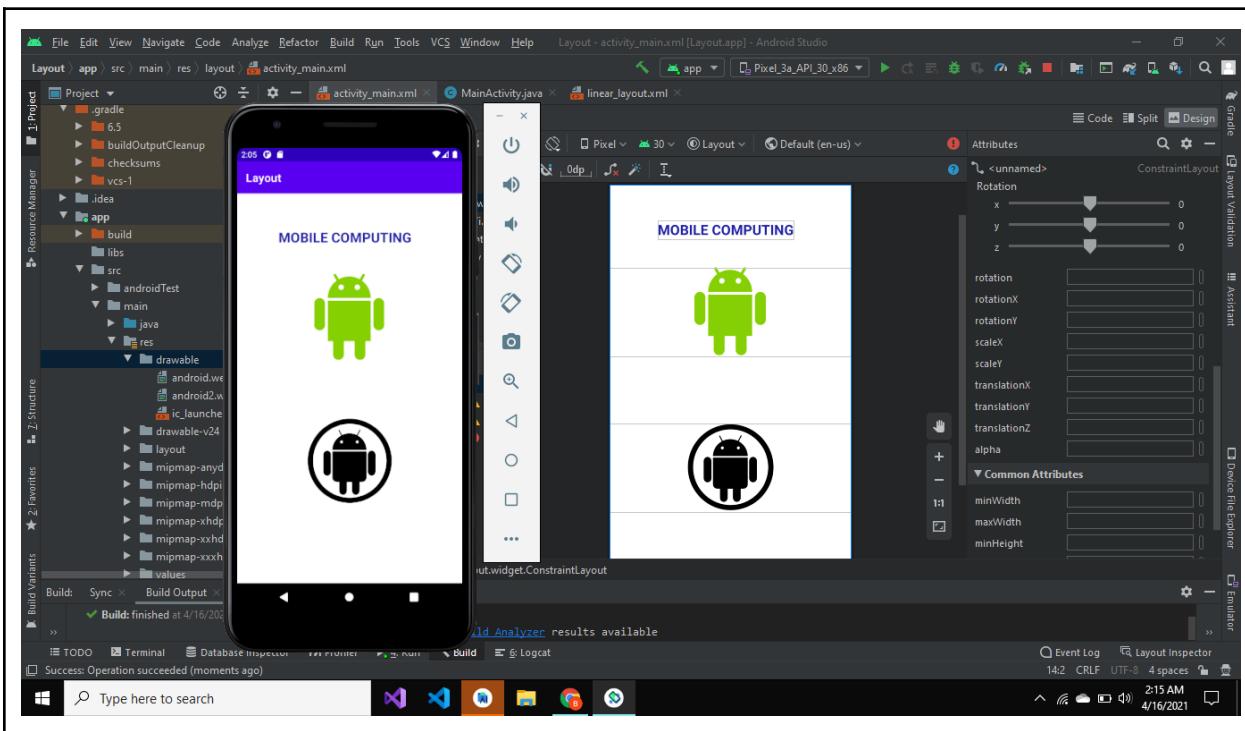
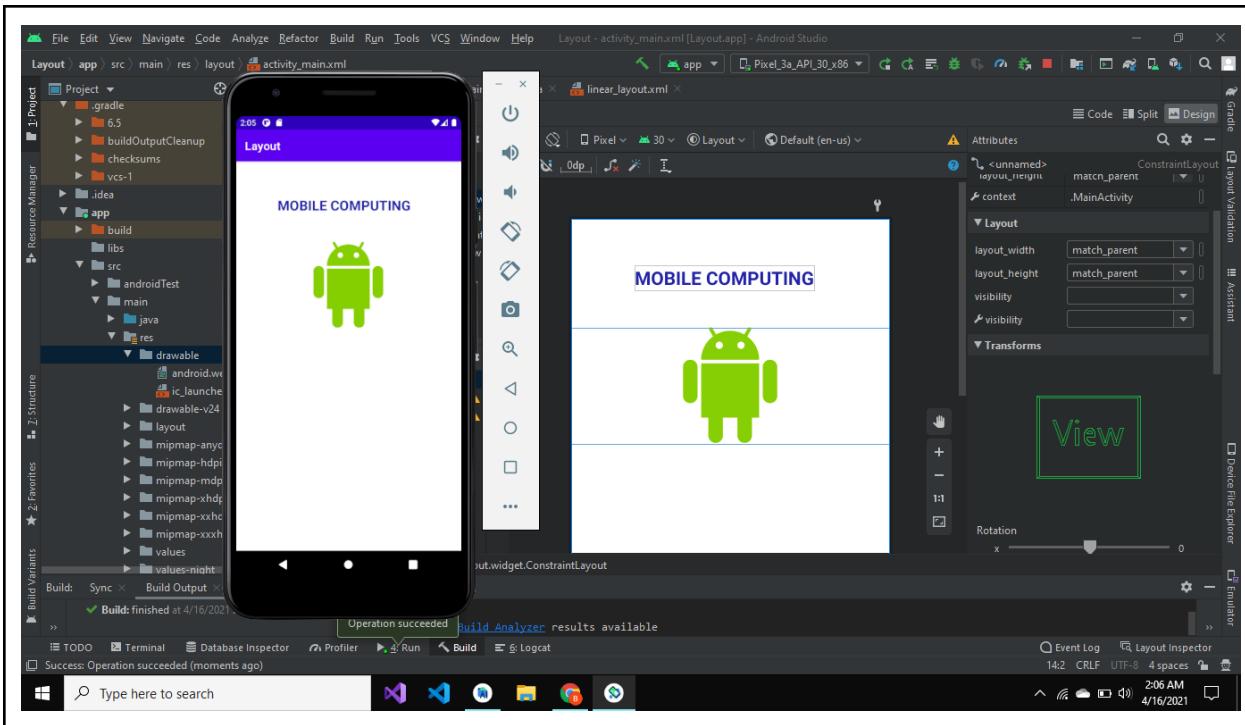
Clear All constraints:

- Step-1** Select all on xml design layout by **ctrl+a**
Step-2 Select the option Clear All constraints.

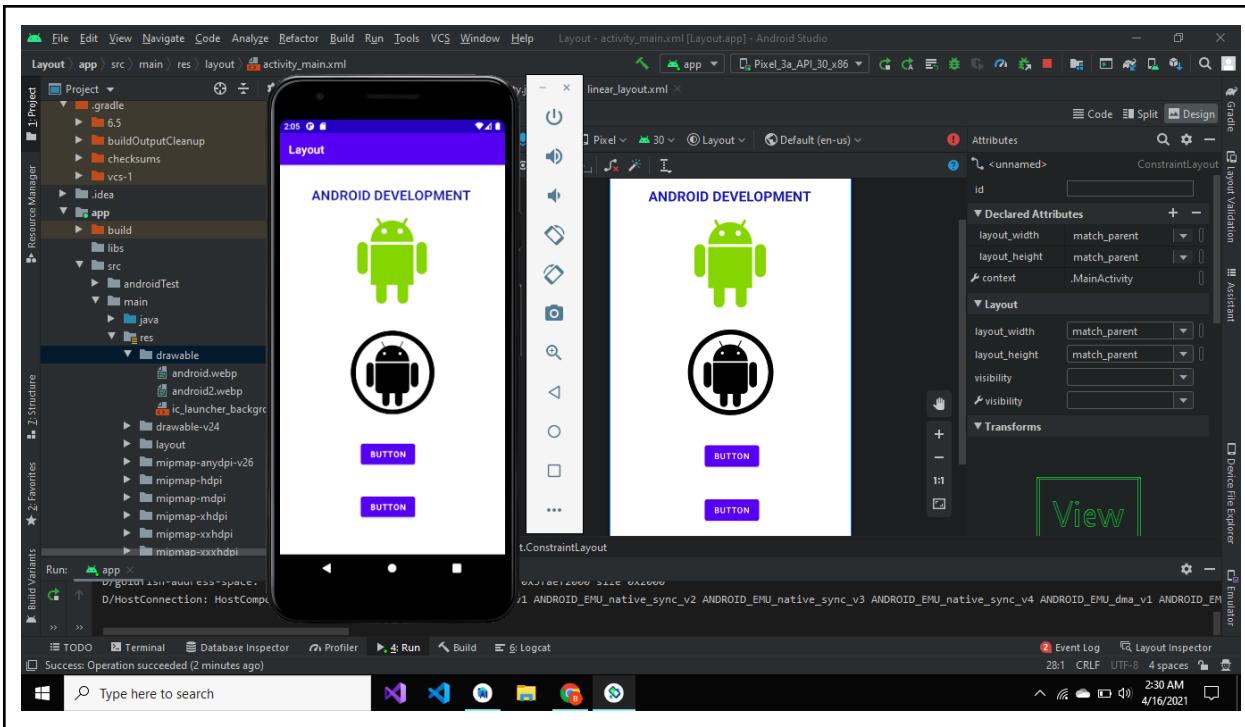


❖ Adding Image in Android Layout:

- Step-1** In drawable paste any image from your system
Step-2 Add an image view from palette and select the image you want to add.

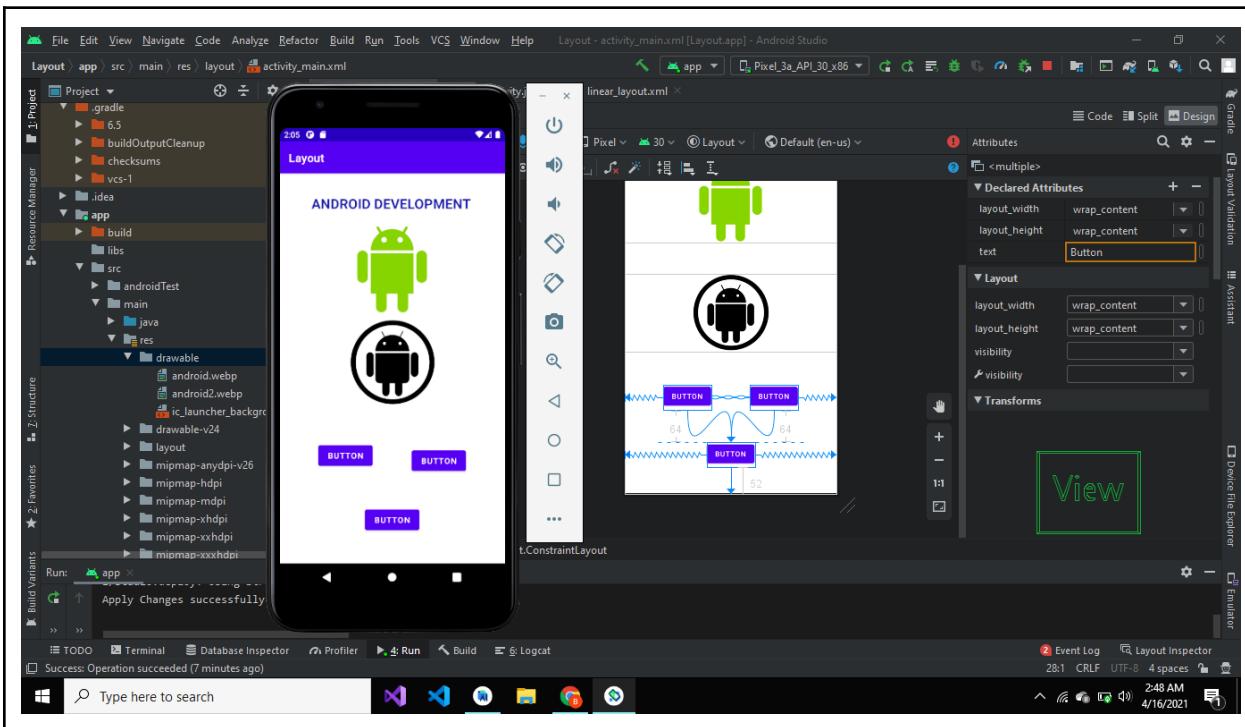


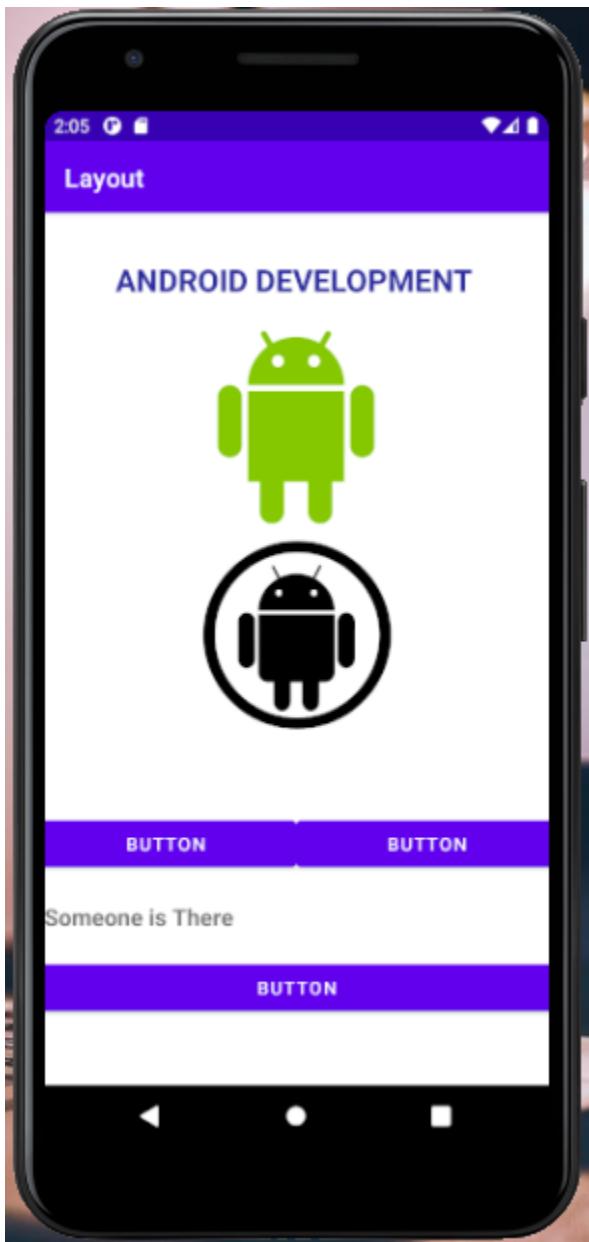
❖ Adding buttons with alignment:



❖ Buttons using Layout Constraint Center-Horizontally:

Reminder place details





❖ **Making Instances for buttons:**

Reminder place details

LECTURE#6

❖ Activity:

An Activity is an application component that represents one window, one hierarchy of views.

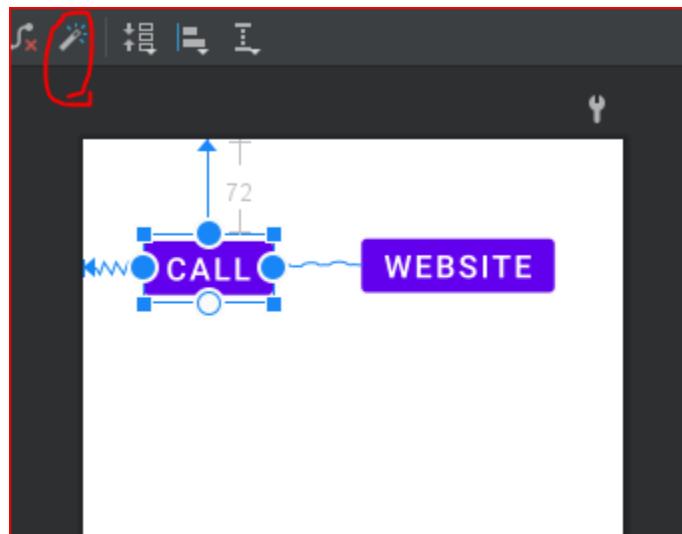
- Typically fills the screen, but can be embedded in other Activity or appear as floating window.
- Java class, typically one Activity in one file.

❖ Intent:

An intent is to perform an action on the screen. It is mostly used to start activity, send broadcast receiver, start services and send message between two activities.

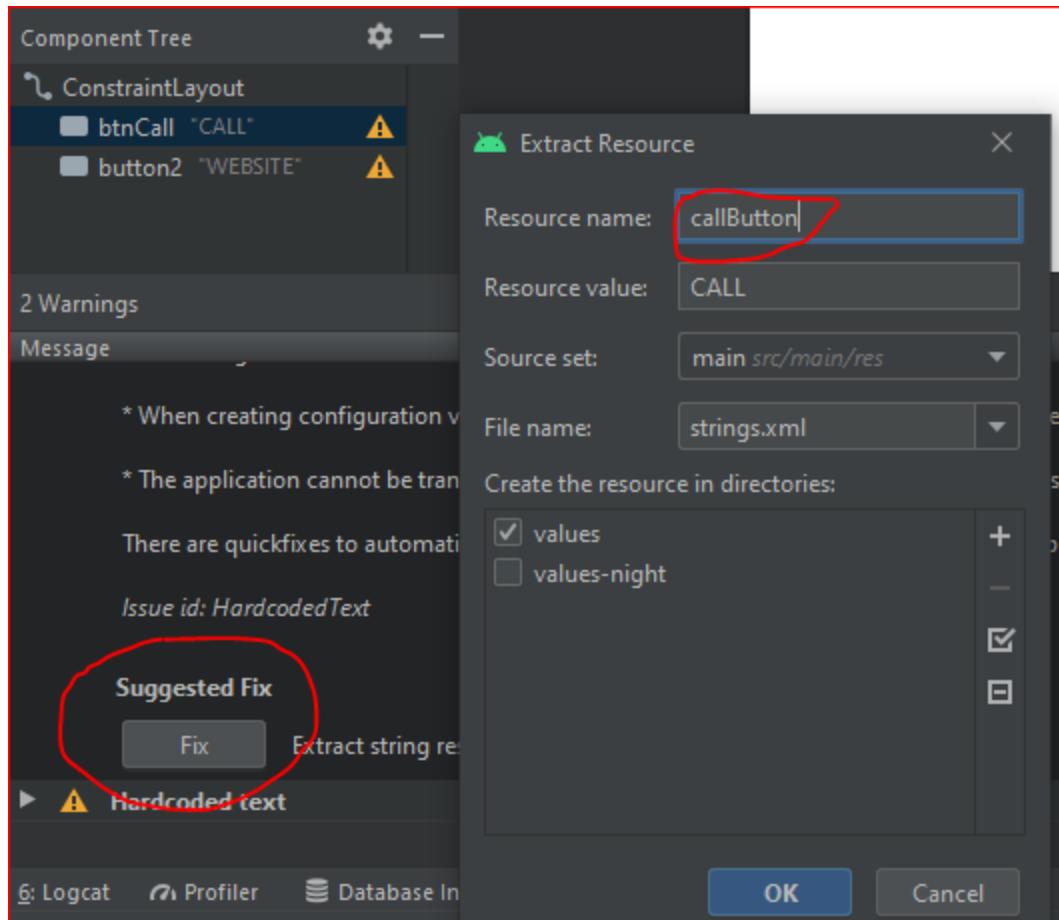
To Set Constraints Automatically:

Step-1 Use the highlighted tool to set constraints automatically.

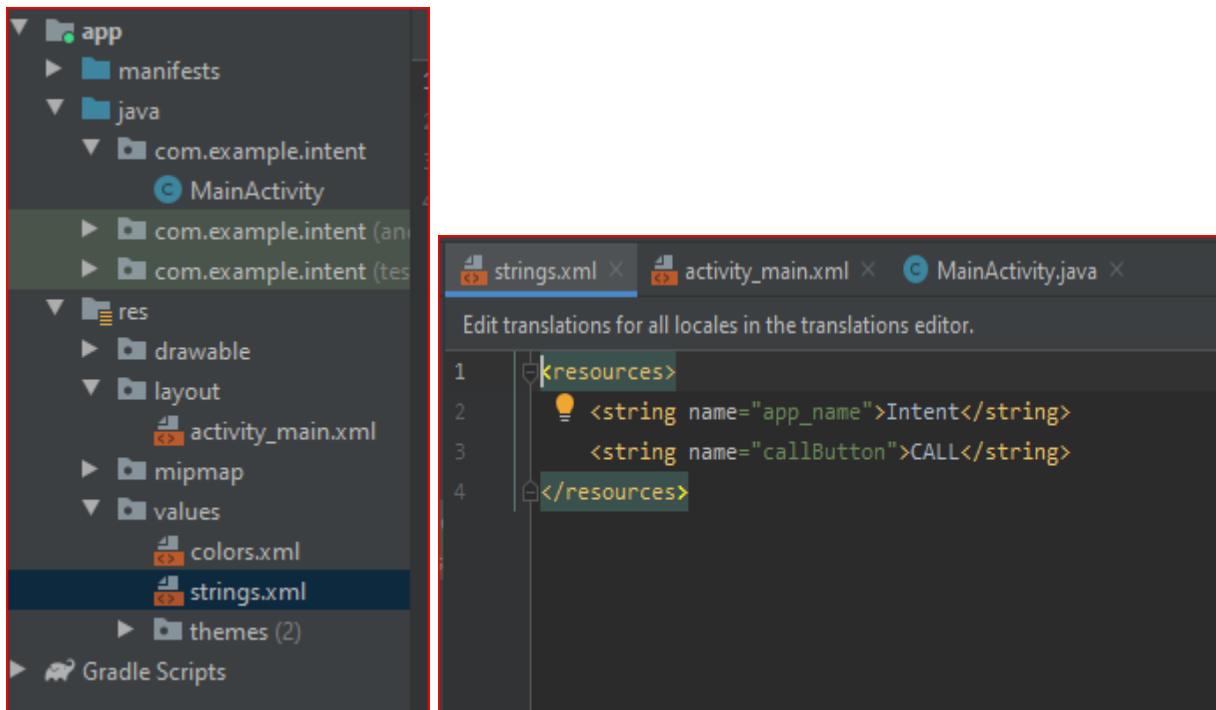


To define String Resources:

Step-2 Click on the fixes button and set a string name for the element.



- The string name will be defined in string.xml file that is



❖ Intent Uses



1. To open Website using Intent:

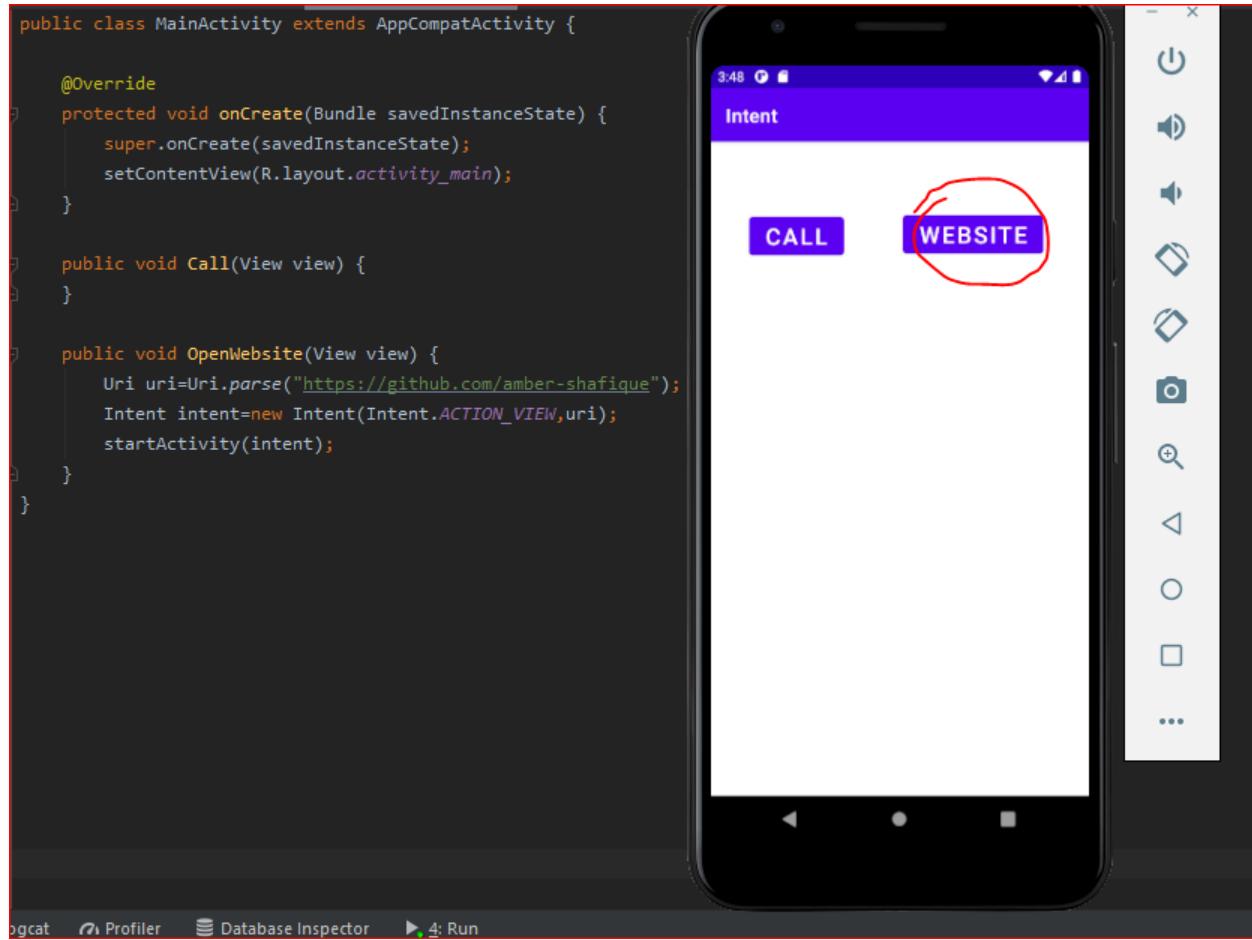
```
Uri uri=Uri.parse("https://github.com/amber-shafique");
Intent intent=new Intent(Intent.ACTION_VIEW,uri);
startActivity(intent);
```

Step-1

Add the above code onClick instance of button.

Step-2

Click on the website button on click of which you have added intent to open website.



- Just after Clicking the **website** button , the url will open

The screenshot shows the Android Studio interface with the following details:

- MainActivity.java:** The code is displayed in the main editor. It contains an `onCreate` method and a `OpenWebsite` method. The `OpenWebsite` method uses `Uri.parse` to parse a URL and creates an `Intent` with `ACTION_VIEW` to start the activity.
- Emulator:** A virtual device is running, displaying a GitHub profile page for "amber-shafique". The page shows basic information like "Joined 22 days ago" and "Follow". Below this, there's a section for "Popular repositories" listing "MC-1" and "20210225_A019".
- Toolbar:** The bottom toolbar includes icons for Logcat, Profiler, Database Inspector, and Run.

2. To Dial a Call using Intent:

```
Uri uri=Uri.parse("tel: +923485677744");
Intent intent=new Intent(Intent.ACTION_DIAL,uri);
startActivity(intent);
```

Step-1

Add the above code onClick instance of button.

Step-2

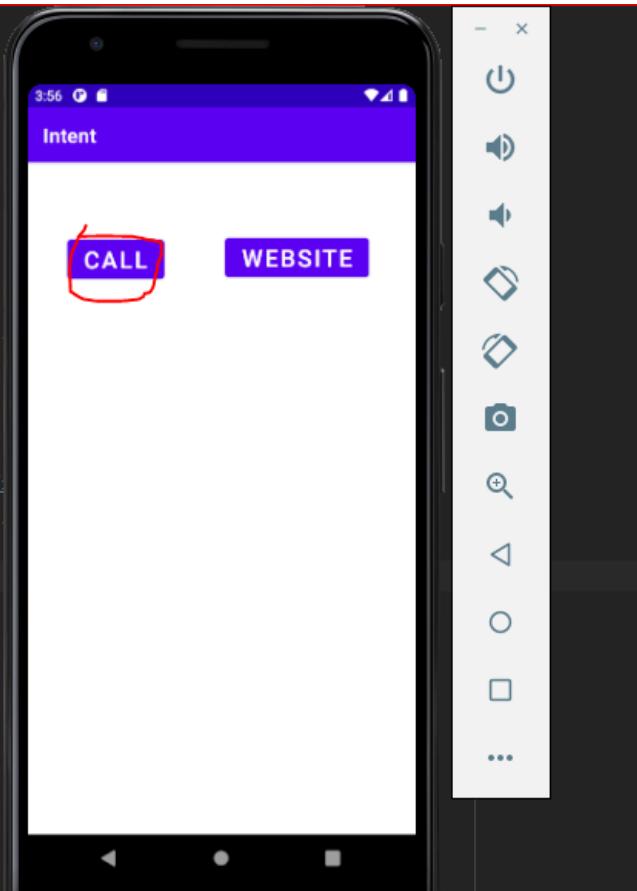
Click on the website button on click of which you have added intent to open call dialer.

```
public class MainActivity extends AppCompatActivity {

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

    public void Call(View view) {
        Uri uri=Uri.parse("tel: +923485677744");
        Intent intent=new Intent(Intent.ACTION_DIAL,uri)
        startActivity(intent);
    }

    public void OpenWebsite(View view) {
        Uri uri=Uri.parse("https://github.com/amber-shaf");
        Intent intent=new Intent(Intent.ACTION_VIEW,uri)
        startActivity(intent);
    }
}
```



- Just after Clicking the **call** button , the call dialer will open

The image shows a developer's environment with two main components: an IDE window on the left and an emulator window on the right.

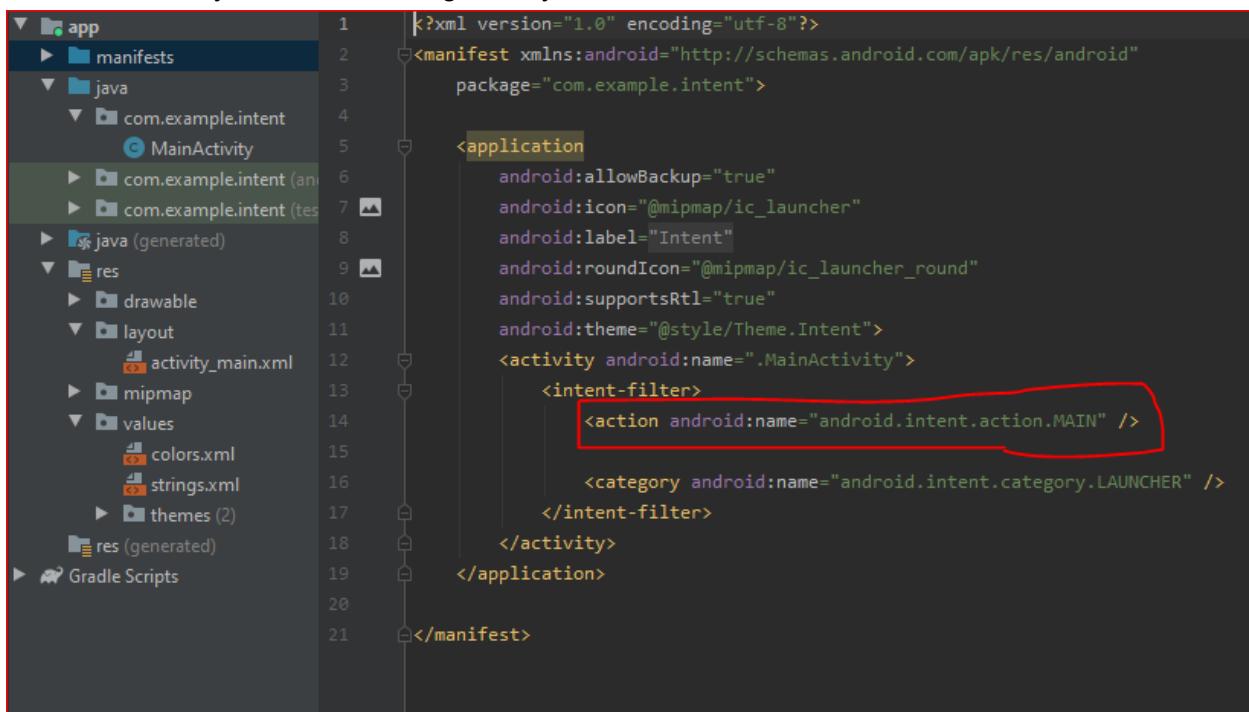
IDE Window (Left): Displays the Java code for `MainActivity`. The code includes an `onCreate` method and two click listeners: `Call` and `OpenWebsite`.

```
10 public class MainActivity extends AppCompatActivity {  
11  
12     @Override  
13     protected void onCreate(Bundle savedInstanceState) {  
14         super.onCreate(savedInstanceState);  
15         setContentView(R.layout.activity_main);  
16     }  
17  
18     public void Call(View view) {  
19         Uri uri=Uri.parse("tel: +923485677744");  
20         Intent intent=new Intent(Intent.ACTION_DIAL,uri)  
21         startActivity(intent);  
22     }  
23  
24     public void OpenWebsite(View view) {  
25         Uri uri=Uri.parse("https://github.com/amber-shaf");  
26         Intent intent=new Intent(Intent.ACTION_VIEW,uri)  
27         startActivity(intent);  
28     }  
29 }  
30  
31  
32  
33  
34  
35  
36  
37  
38
```

A green "Success" toast message is visible at the bottom of the IDE window.

Emulator Window (Right): Shows a virtual smartphone displaying a dialer interface. The screen shows a phone number (+92 348 5677744) and a numeric keypad. At the top, there are options: "Create new contact", "Add to a contact", and "Send SMS". On the right side of the emulator, there is a vertical toolbar with various icons for power, volume, camera, search, and more.

- **Setting another Activity as Launcher Activity:**
By default launching activity in **AndroidManifest.xml** file.

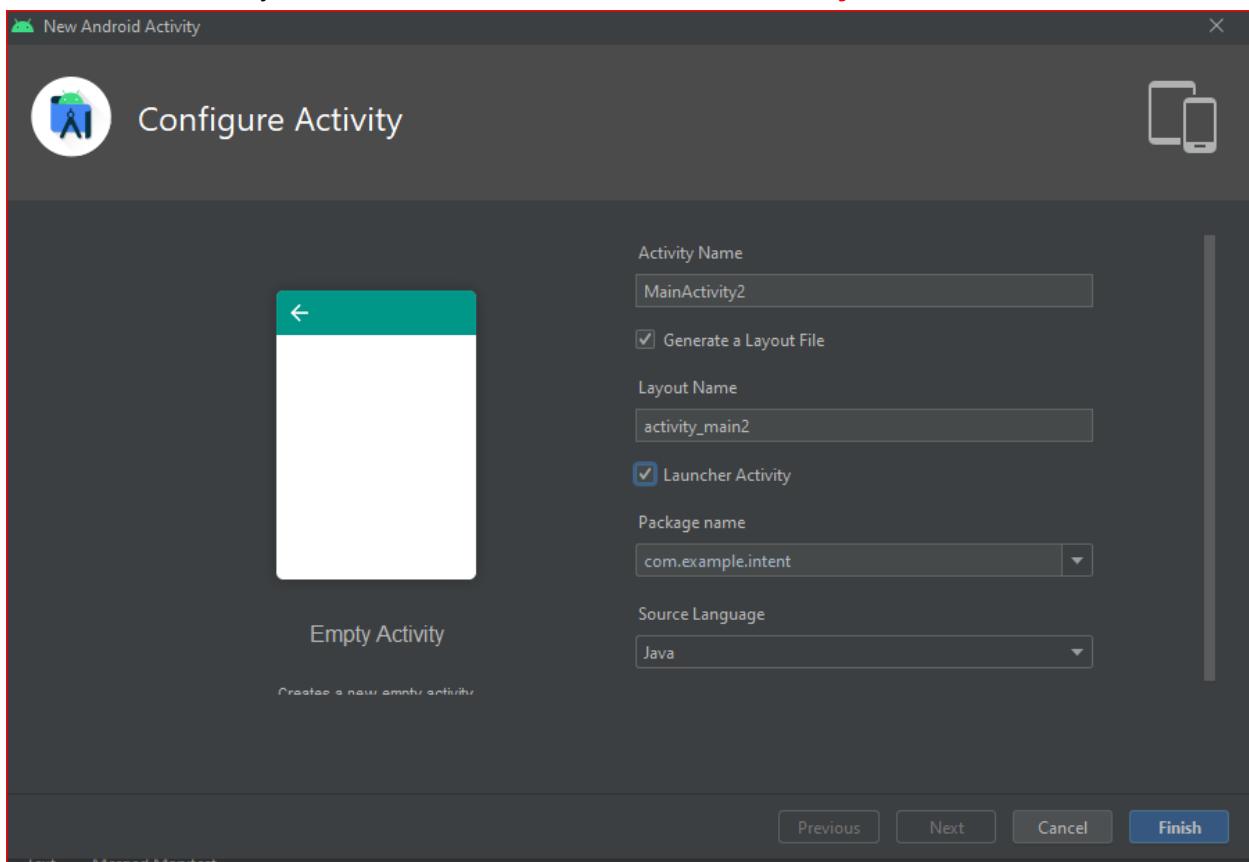


```

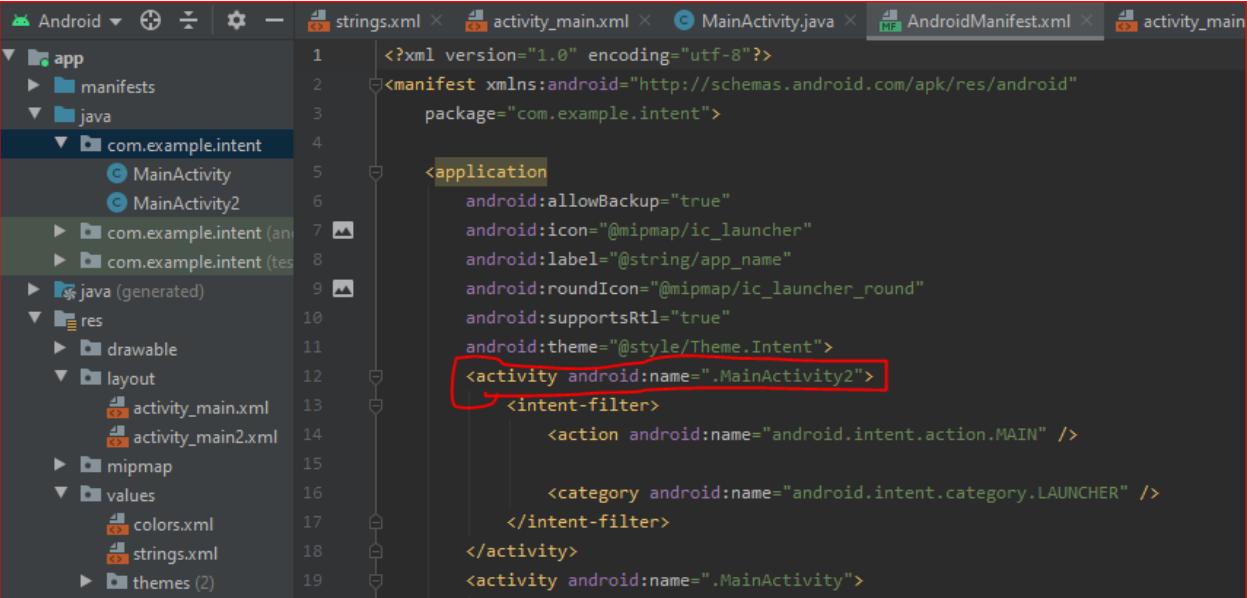
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="com.example.intent">
4
5     <application
6         android:allowBackup="true"
7         android:icon="@mipmap/ic_launcher"
8         android:label="Intent"
9         android:roundIcon="@mipmap/ic_launcher_round"
10        android:supportsRtl="true"
11        android:theme="@style/Theme.Intent">
12         <activity android:name=".MainActivity">
13             <intent-filter>
14                 <action android:name="android.intent.action.MAIN" />
15
16                 <category android:name="android.intent.category.LAUNCHER" />
17             </intent-filter>
18         </activity>
19     </application>
20
21 </manifest>

```

Create a new activity and check its attribute of **Launcher Activity**.



Launcher Activity changes in **AndroidManifest.xml** file



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.intent">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Intent">
        <activity android:name=".MainActivity2">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".MainActivity">
```

3. To move from one activity to another:

this=> present activity

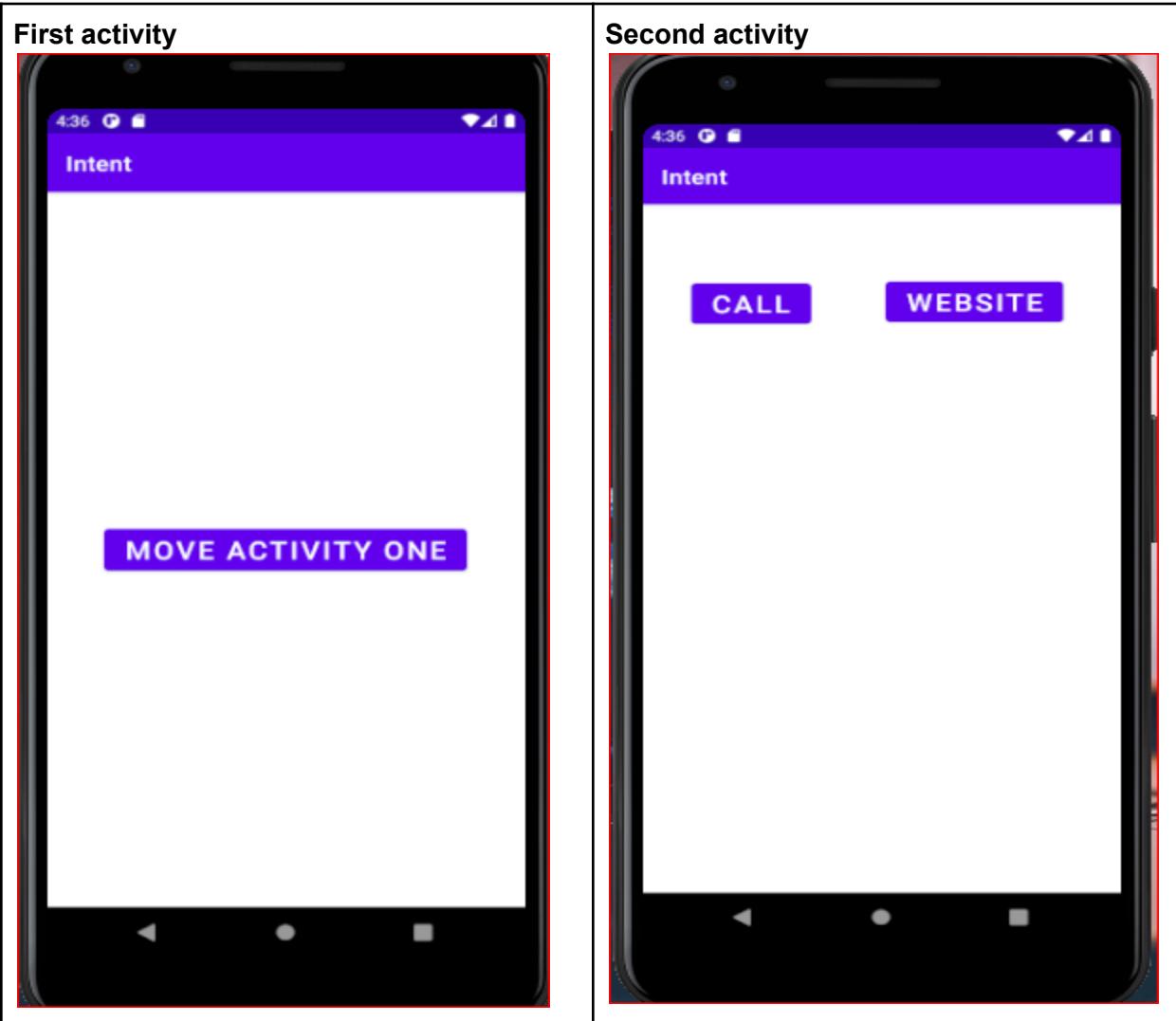
ActivitName.class=> activity where to move

```
Intent intent=new Intent(this, MainActivity.class);
startActivity(intent);
```

Step-1 Add the above code onClick instance of button.

Step-2 Click on the website button on click of which you have added intent to open another activity..

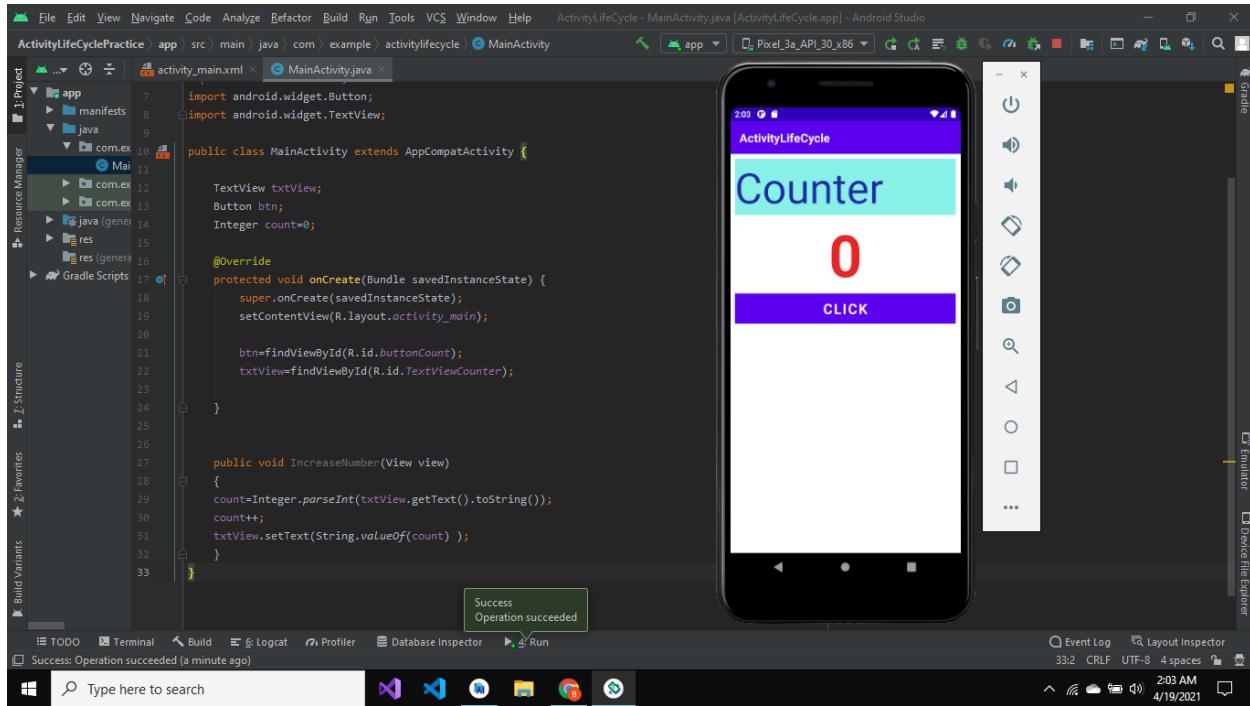
- Just after Clicking the **Move Activity One** button , the second activity will open.



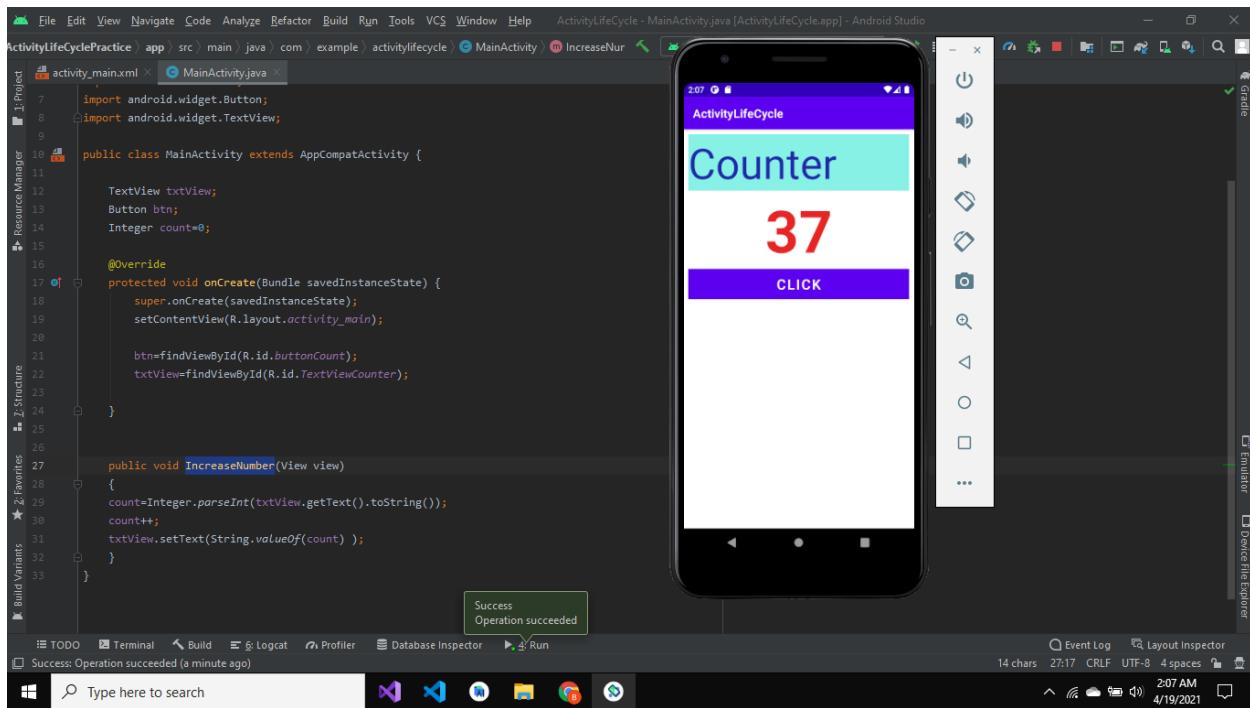
LECTURE#7

❖ Counter:

Adding code to display a counter on screen.



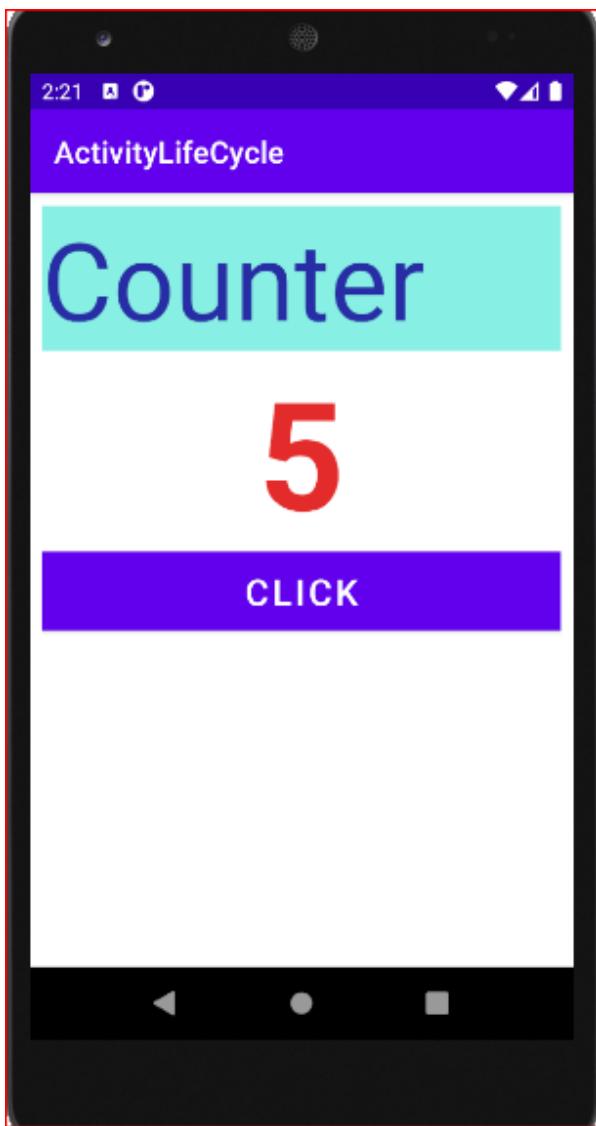
Click on the button to increase value of counter and see the result.



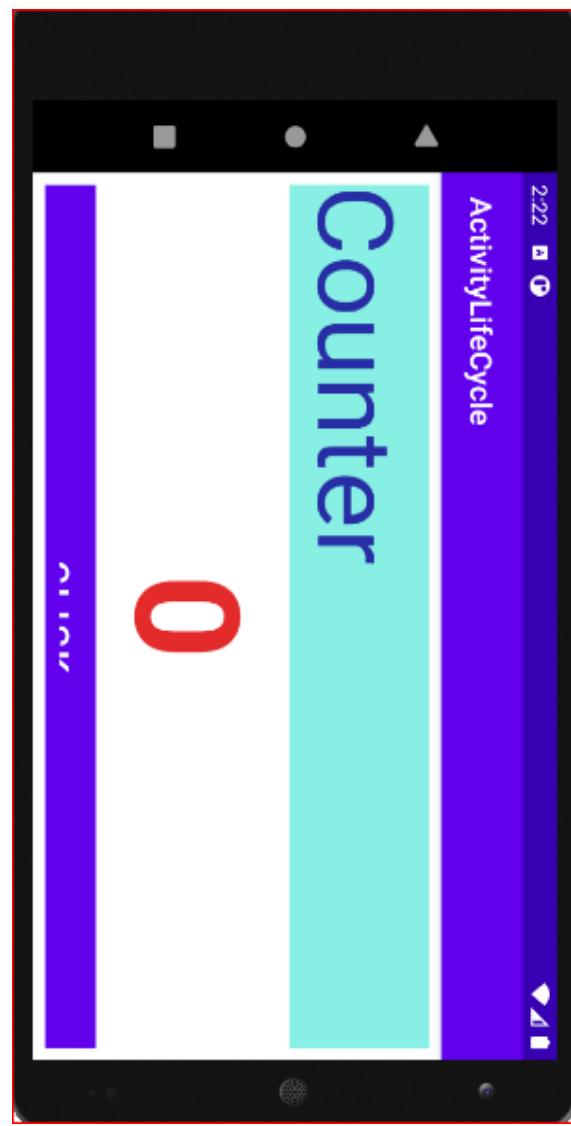
Rotation-Action Problem:

When we rotate the screen counter values goes back to zero

Before Rotation:



After Rotation:

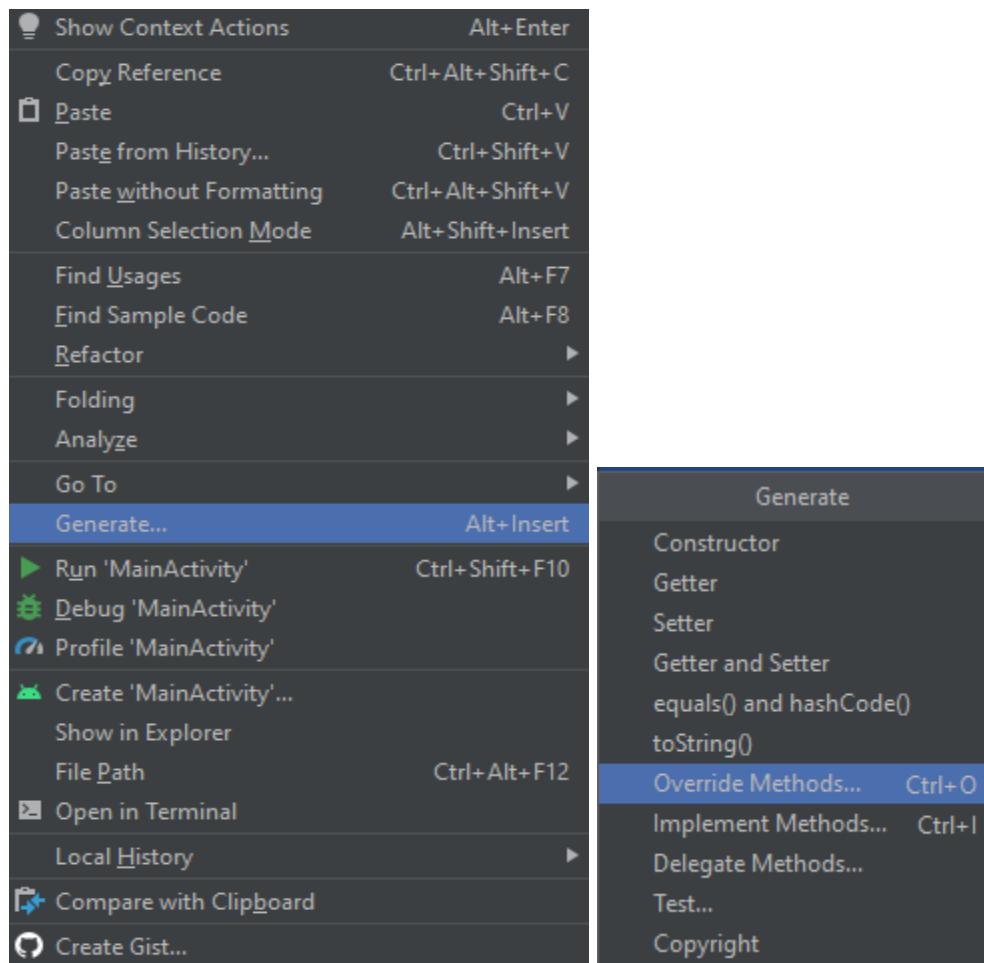


❖ Activity LifeCycle

- Create a LOG

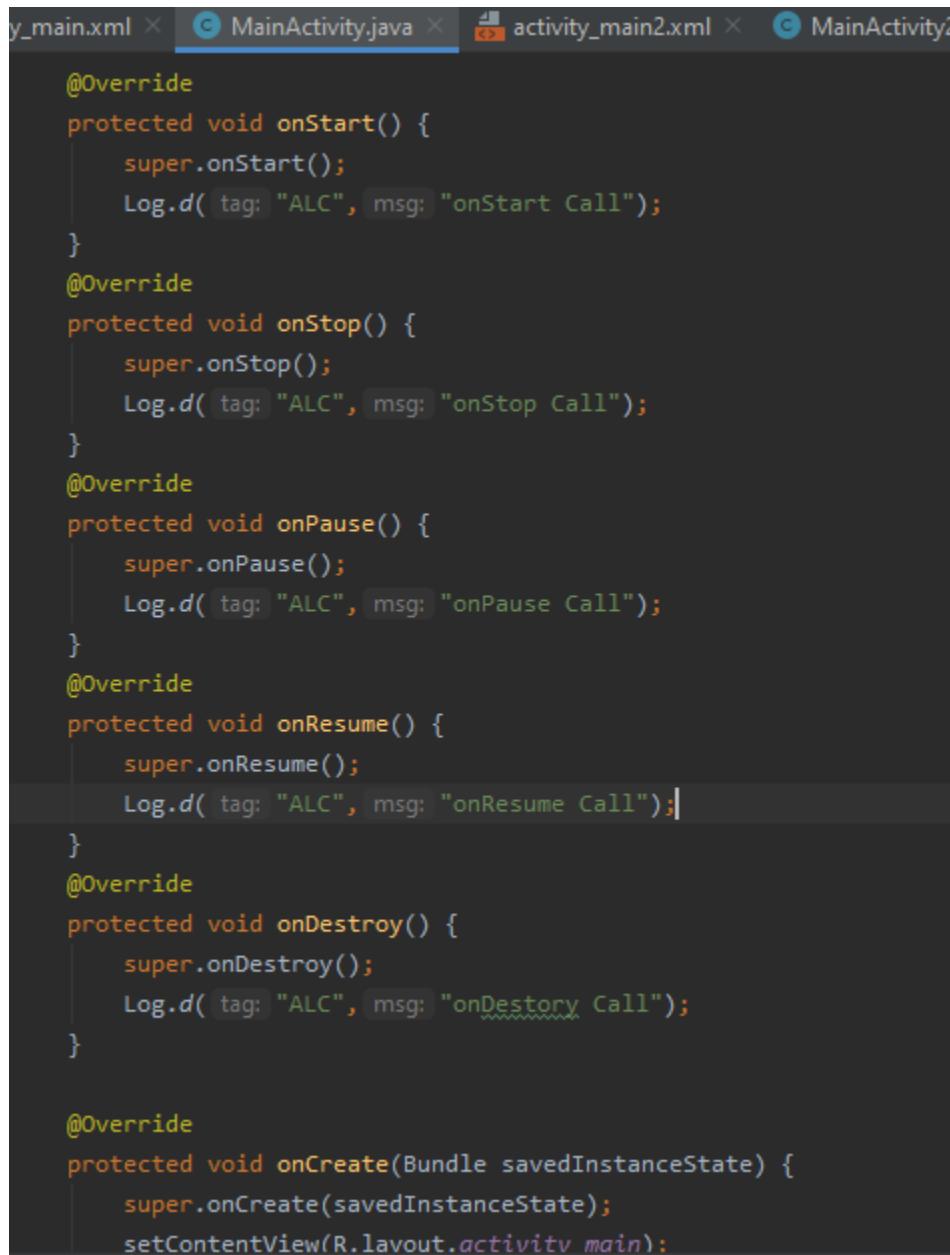
```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
  
    btn=findViewById(R.id.buttonCount);  
    txtView=findViewById(R.id.TextViewCounter);  
  
    Log.d( tag: "ALC", msg: "onCreate Call");
```

- To Override methods:



Override the 6 methods in main activity to check their logs in logcat.

- **onCreate()**
- **onStart()**
- **onResume()**
- **on pause()**
- **on Stop()**
- **onDestroy()**

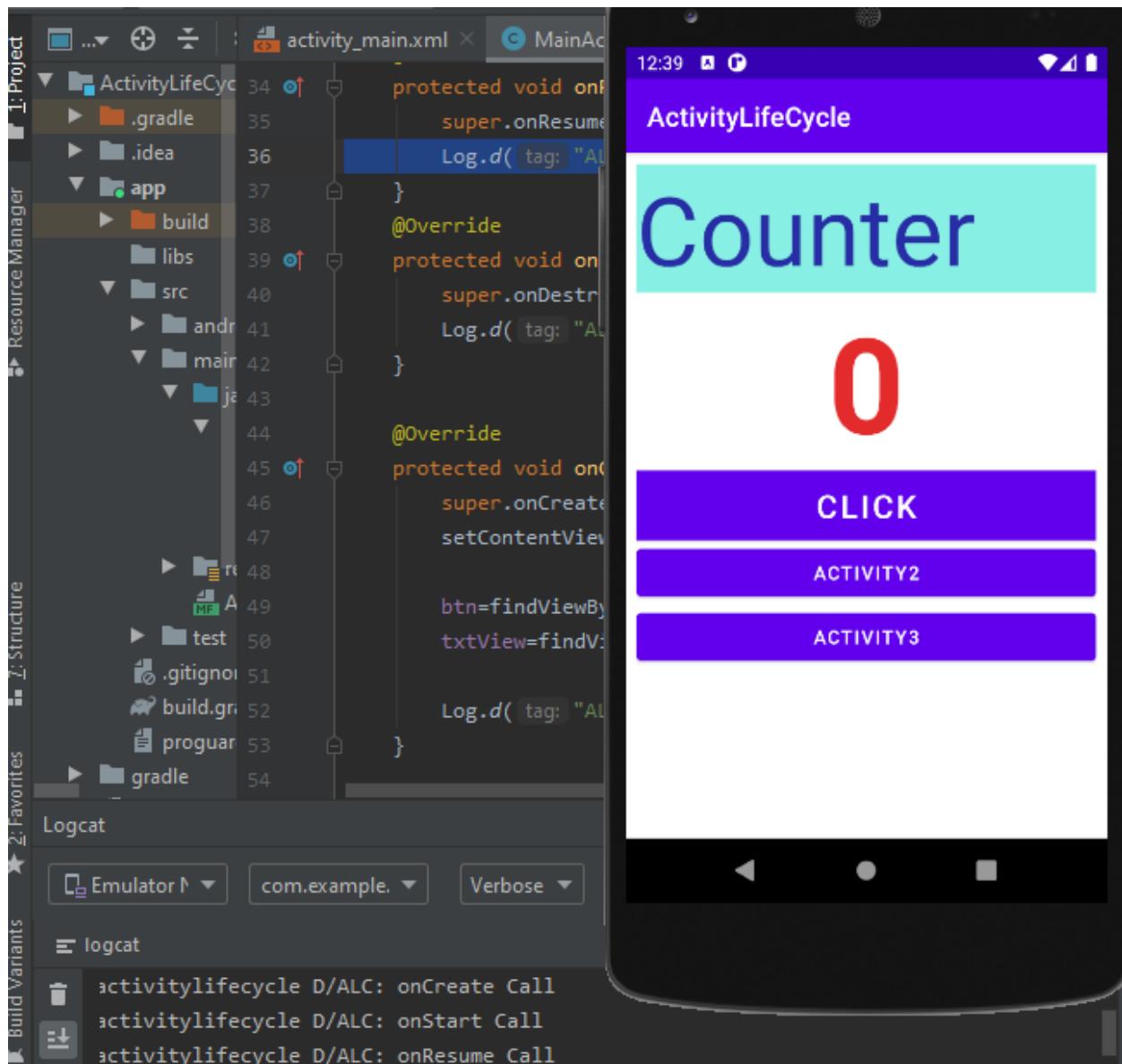


The screenshot shows the Android Studio interface with the tab bar at the top. The active tab is "MainActivity.java". Below it, other tabs include "activity_main.xml", "activity_main2.xml", and "MainActivity2". The code in the editor is as follows:

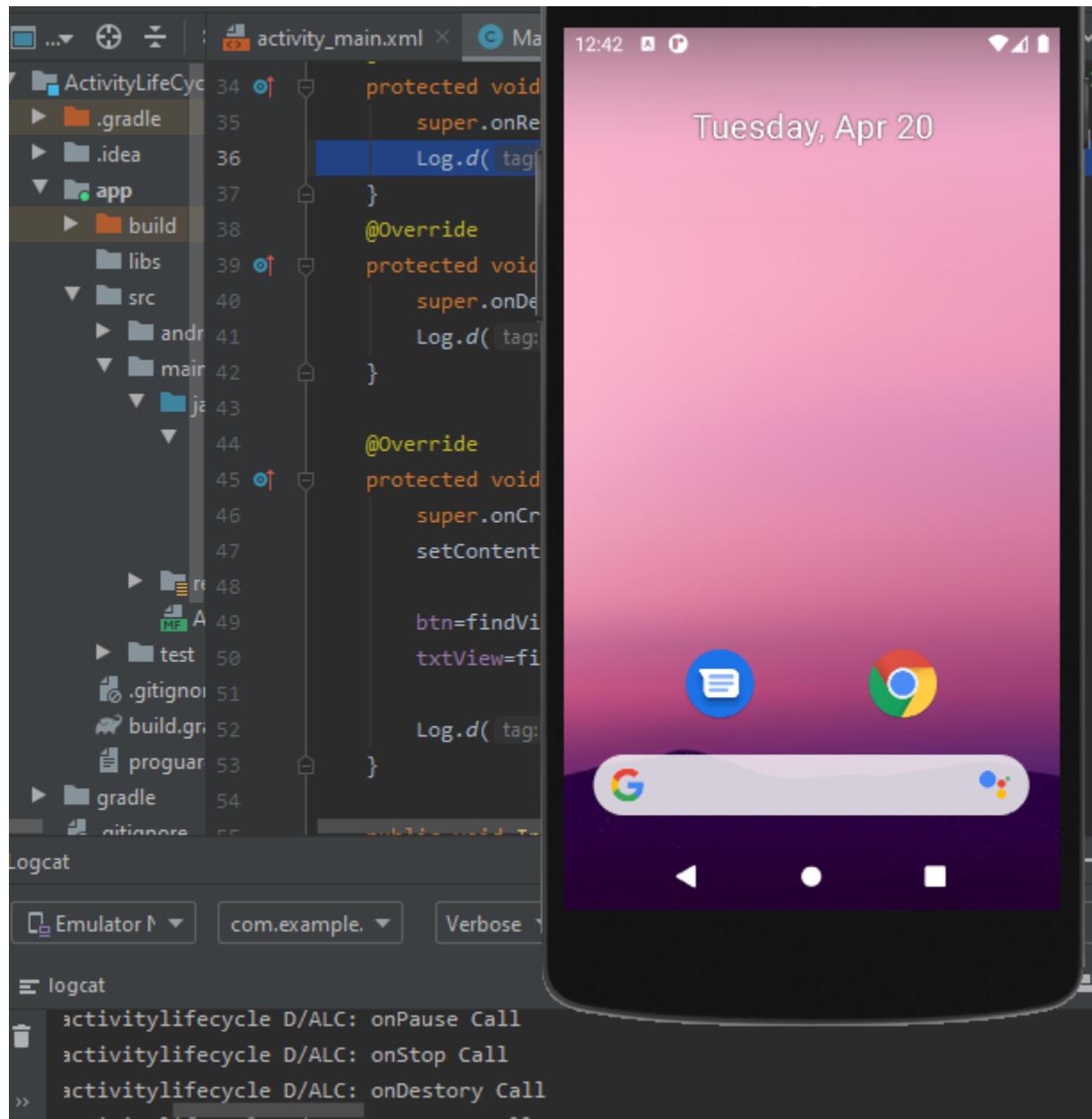
```
@Override  
protected void onStart() {  
    super.onStart();  
    Log.d( tag: "ALC", msg: "onStart Call");  
}  
  
@Override  
protected void onStop() {  
    super.onStop();  
    Log.d( tag: "ALC", msg: "onStop Call");  
}  
  
@Override  
protected void onPause() {  
    super.onPause();  
    Log.d( tag: "ALC", msg: "onPause Call");  
}  
  
@Override  
protected void onResume() {  
    super.onResume();  
    Log.d( tag: "ALC", msg: "onResume Call");  
}  
  
@Override  
protected void onDestroy() {  
    super.onDestroy();  
    Log.d( tag: "ALC", msg: "onDestory Call");  
}  
  
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);
```

❖ ALC of One Activity:

- Run the emulator and check the logcat.
- On running the application following three function will be called as shown in logcat
 1. onCreate()
 2. onStart()
 3. onResume()



- On just clicking back button on emulator screen following three functions will be called.
 1. on pause()
 2. on Stop()
 3. onDestroy()



❖ ALC of Main activity to any other activity:

- Override the same 6 functions in activity2.
- Rebuild the application and this time after opening the main activity navigate to second activity and see the logcat.
- Logcat is displaying functions calls by tag names as:
 ALC => for Main Activity
 ALC2 => for second activity when clicked

The screenshot shows an Android development environment with the following components:

- MainActivity.java:** Contains code for overriding onStart(), onStop(), and onPause() methods. It includes a tag message for each call.
- MainActivity2.java:** Contains code for overriding onStart(), onStop(), and onPause() methods. It includes a tag message for each call.
- activity_main2.xml:** XML layout file for Activity-2.
- activity_main3.xml:** XML layout file for the Main Activity.
- Logcat:** Shows the following log entries:

```
35.757 9724-9724/com.example.activitylifecycle D/ALC: onCreate Call
35.767 9724-9724/com.example.activitylifecycle D/ALC: onStart Call
35.772 9724-9724/com.example.activitylifecycle D/ALC: onResume Call
04.218 9724-9724/com.example.activitylifecycle D/ALC: onPause Call
04.549 9724-9724/com.example.activitylifecycle D/ALC2: onCreate Called
04.555 9724-9724/com.example.activitylifecycle D/ALC2: onStart Called
04.556 9724-9724/com.example.activitylifecycle D/ALC2: onResume Called
05.194 9724-9724/com.example.activitylifecycle D/ALC: onStop Call
```

The screenshot also shows a preview of the app running on a virtual device, titled "Activity-2". The title bar of the device screen displays "ActivityLifecycle".

- Now from second activity click back to move back to main activity and see the logcat.

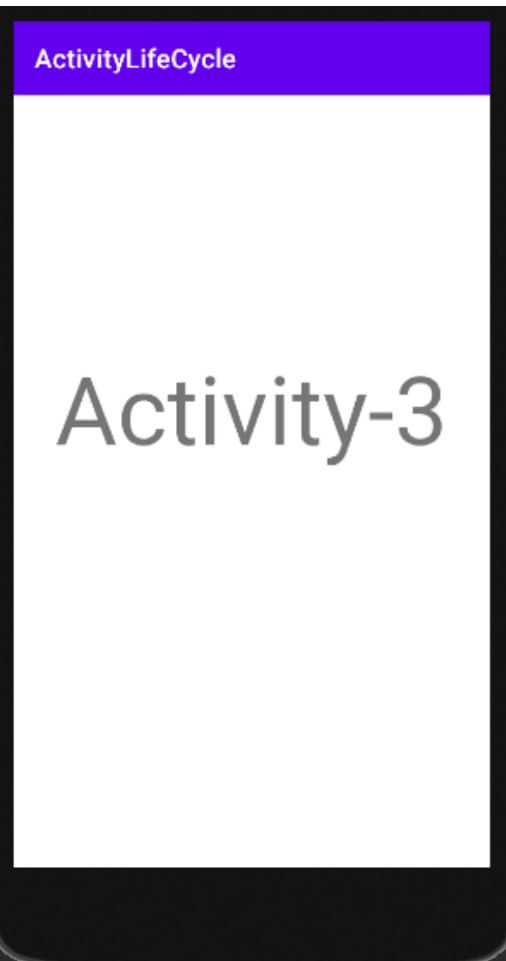
MainActivity.java

```
>MainActivity2 extends AppCompatActivity {
    void onStart() {
        onStart();
        ( tag: "ALC2", msg: "onStart Called");
    }
    void onStop() {
        onStop();
    }
}
```

Logcat:

```
35.757 9724-9724/com.example.activitylifecycle D/ALC: onCreate Call
35.767 9724-9724/com.example.activitylifecycle D/ALC: onStart Call
35.772 9724-9724/com.example.activitylifecycle D/ALC: onResume Call
35.772 9724-9724/com.example.activitylifecycle D/ALC: onPause Call
35.772 9724-9724/com.example.activitylifecycle D/ALC2: onCreate Called
35.772 9724-9724/com.example.activitylifecycle D/ALC2: onStart Called
35.772 9724-9724/com.example.activitylifecycle D/ALC2: onResume Called
35.772 9724-9724/com.example.activitylifecycle D/ALC2: onStop Call
35.772 9724-9724/com.example.activitylifecycle D/ALC2: onPause Called
35.772 9724-9724/com.example.activitylifecycle D/ALC2: onStart Call
35.772 9724-9724/com.example.activitylifecycle D/ALC2: onResume Call
35.772 9724-9724/com.example.activitylifecycle D/ALC2: onStop Called
35.772 9724-9724/com.example.activitylifecycle D/ALC2: onDestroy Called
```

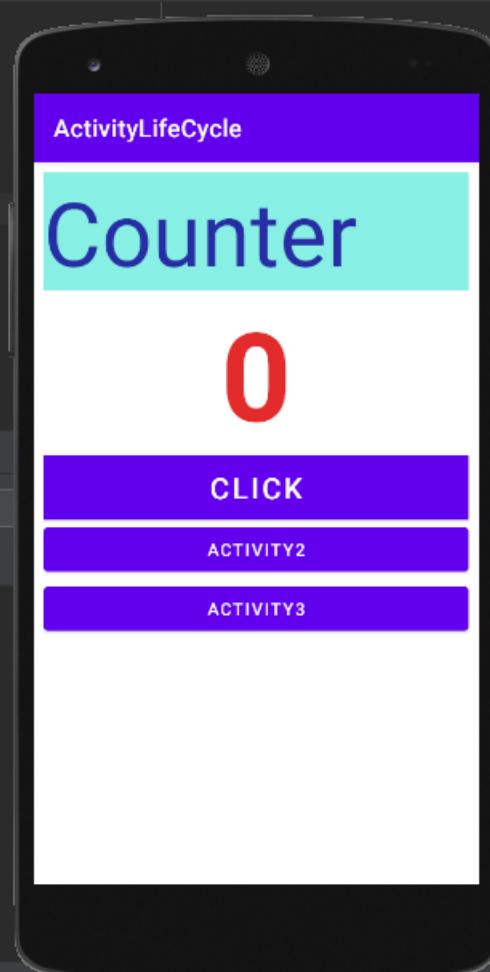
Repeating same steps for activity 3,navigate from main to 2nd activity then back to main and move to Activity-3. Then see the logcat.



The screenshot shows an Android application interface. At the top, there is a purple header bar with the text "ActivityLifecycle". Below it is a white content area containing the large text "Activity-3". On the left side of the screen, there is a dark-themed code editor or log viewer window. The title bar of this window shows three tabs: "activity_main2.xml", "MainActivity2.java", and "activity_main3.xml". The main content area of the window displays a series of log messages in white text on a black background, all starting with the prefix "/com.example.activitylifecycle D/ALC:". The messages are as follows:

```
/com.example.activitylifecycle D/ALC: onCreate Call  
/com.example.activitylifecycle D/ALC: onStart Call  
/com.example.activitylifecycle D/ALC: onResume Call  
/com.example.activitylifecycle D/ALC: onPause Call  
/com.example.activitylifecycle D/ALC2: onCreate Called  
/com.example.activitylifecycle D/ALC2: onStart Called  
/com.example.activitylifecycle D/ALC2: onResume Called  
/com.example.activitylifecycle D/ALC: onStop Call  
/com.example.activitylifecycle D/ALC2: onPause Called  
/com.example.activitylifecycle D/ALC: onStart Call  
/com.example.activitylifecycle D/ALC: onResume Call  
/com.example.activitylifecycle D/ALC2: onStop Called  
/com.example.activitylifecycle D/ALC2: onDestory Called  
/com.example.activitylifecycle D/ALC: onPause Call  
/com.example.activitylifecycle D/ALC3: onCreate Called  
/com.example.activitylifecycle D/ALC3: onStart Called  
/com.example.activitylifecycle D/ALC3: onResume Called  
/com.example.activitylifecycle D/ALC: onStop Call
```

❖ ALC of Main activity to other activity from other to another activity:



The screenshot shows an Android application interface and its corresponding Java code. The application title is "ActivityLifeCycle" and the main screen displays the word "Counter" in large blue text, followed by a large red number "0". Below the number are three purple buttons labeled "CLICK", "ACTIVITY2", and "ACTIVITY3".

The Java code in the MainActivity.java file contains the following logic for handling the activity life cycle:

```
onResume() {
    ...
    C2", msg: "onResume Called");
}

onDestroy() {
    ...
    C2", msg: "onDestroy Called");
}
```

The logcat output shows the following messages:

```
2045-12045/com.example.activitylifecycle D/ALC: onCreate Call
2045-12045/com.example.activitylifecycle D/ALC: onStart Call
2045-12045/com.example.activitylifecycle D/ALC: onResume Call
```

The application is running on a virtual device with a black background. The bottom of the screen shows the Android navigation bar.

```
activity_main2.xml MainActivity2.java activity_main3.xml MainActivity3.java
```

```
{  
    : "onResume Called");  
  
    {  
        : "onDestroy Called");  
  
    }  
}
```

example.activitylifecycle (120k) Verbose alc

```
45/com.example.activitylifecycle D/ALC: onCreate Call  
45/com.example.activitylifecycle D/ALC: onStart Call  
45/com.example.activitylifecycle D/ALC: onResume Call  
45/com.example.activitylifecycle D/ALC: onPause Call  
45/com.example.activitylifecycle D/ALC2: onCreate Called  
45/com.example.activitylifecycle D/ALC2: onStart Called  
45/com.example.activitylifecycle D/ALC2: onResume Called  
45/com.example.activitylifecycle D/ALC: onStop Call
```

The screenshot shows an Android development environment with the following components:

- Code Editor:** Displays `MainActivity2.java` with code related to activity lifecycle handling.
- Logcat:** Shows log messages from the application, including:

```
.example.activitylifecycle D/ALC: onCreate Call  
.example.activitylifecycle D/ALC: onStart Call  
.example.activitylifecycle D/ALC: onResume Call  
.example.activitylifecycle D/ALC: onPause Call  
.example.activitylifecycle D/ALC2: onCreate Called  
.example.activitylifecycle D/ALC2: onStart Called  
.example.activitylifecycle D/ALC2: onResume Called  
.example.activitylifecycle D/ALC2: onStop Called  
.example.activitylifecycle D/ALC3: onCreate Called  
.example.activitylifecycle D/ALC3: onStart Called  
.example.activitylifecycle D/ALC3: onResume Called  
.example.activitylifecycle D/ALC3: onStop Called
```
- Emulator:** A virtual device labeled "Nexus 5 (Edited) API 30" is running, displaying the text "Activity-3" on its screen.

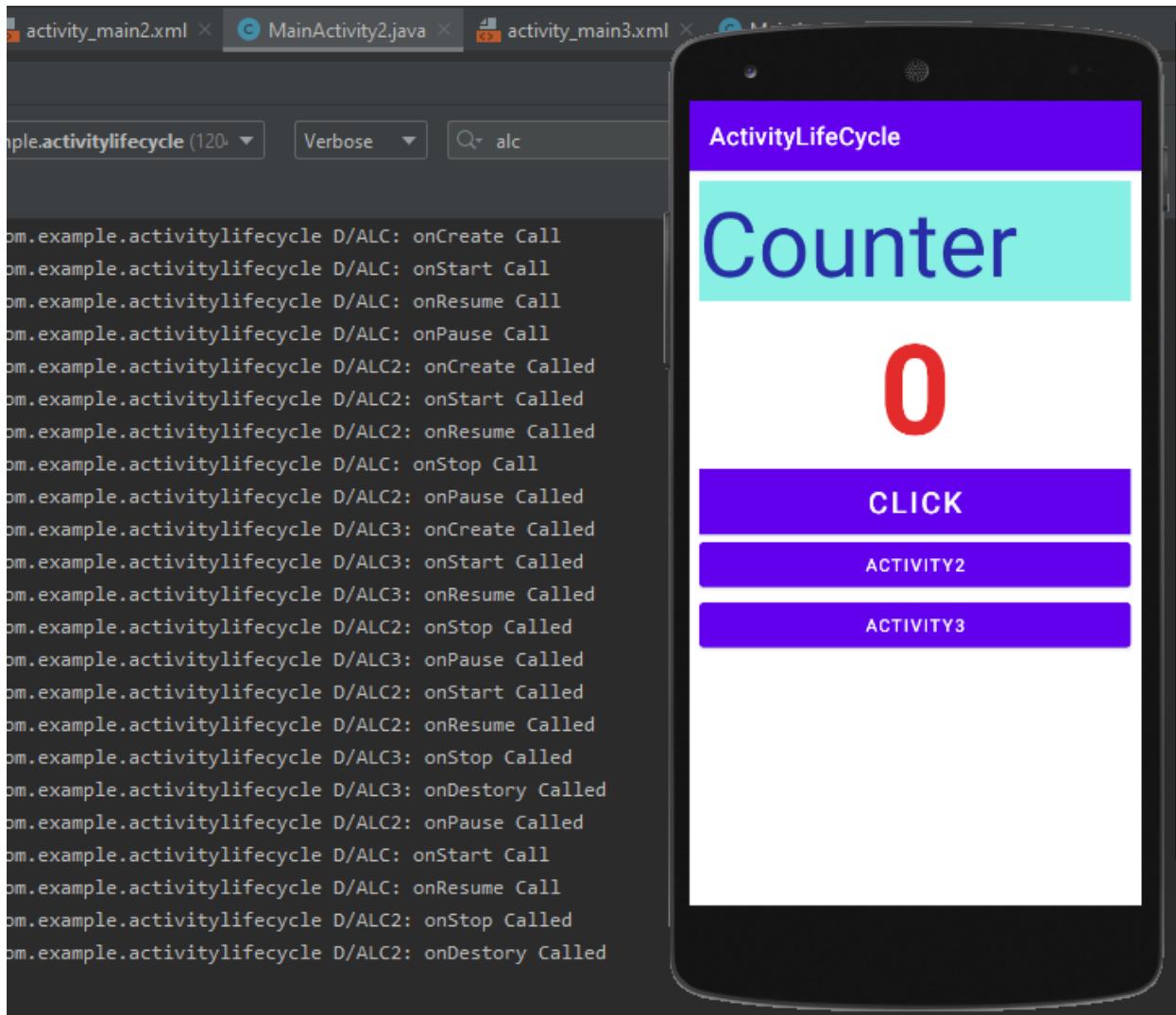
The screenshot shows an Android application interface. At the top, there is a purple header bar with the text "ActivityLifecycle". Below the header, the main content area displays the text "Activity-2" in a large, light gray font. In the bottom right corner of the content area, there is a purple rectangular button with the white text "MOVE ACTIVITY 3".

On the left side of the screen, there is a vertical stack of three tabs or cards, each with a purple header. The top card is labeled "activity_main2.xml", the middle one "MainActivity2.java", and the bottom one "activity_main3.xml".

Below these tabs, the Java code for "MainActivity2.java" is visible. It contains several log statements (Log.d) that output the names of various lifecycle methods being called. These logs are repeated multiple times, indicating a loop or a test scenario.

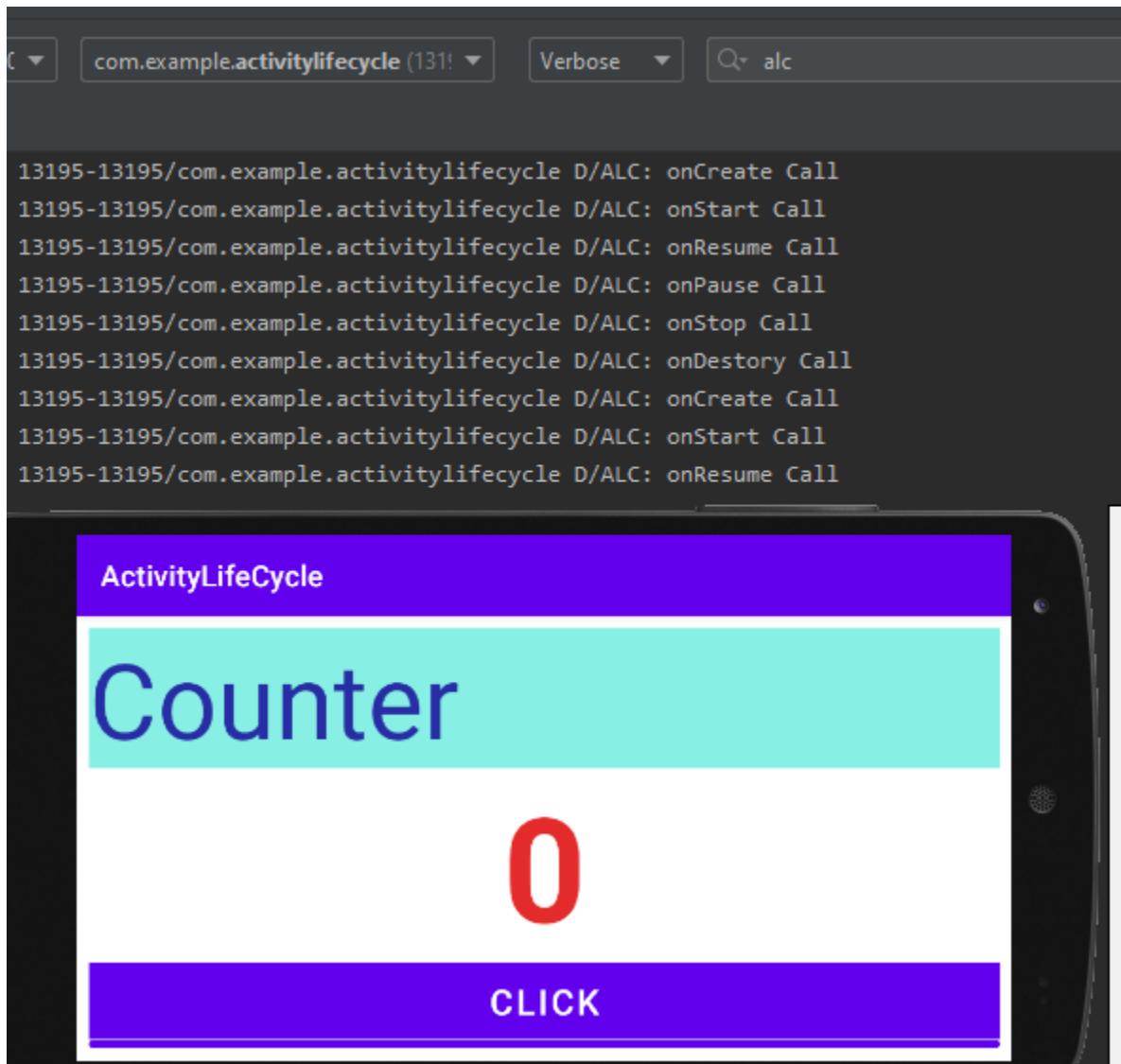
```
onDestroy Called");
}

com.example.activitylifecycle D/ALC: onCreate Call
com.example.activitylifecycle D/ALC: onStart Call
com.example.activitylifecycle D/ALC: onResume Call
com.example.activitylifecycle D/ALC: onPause Call
com.example.activitylifecycle D/ALC2: onCreate Called
com.example.activitylifecycle D/ALC2: onStart Called
com.example.activitylifecycle D/ALC2: onResume Called
com.example.activitylifecycle D/ALC2: onStop Call
com.example.activitylifecycle D/ALC2: onPause Called
com.example.activitylifecycle D/ALC3: onCreate Called
com.example.activitylifecycle D/ALC3: onStart Called
com.example.activitylifecycle D/ALC3: onResume Called
com.example.activitylifecycle D/ALC2: onStop Called
com.example.activitylifecycle D/ALC3: onPause Called
com.example.activitylifecycle D/ALC2: onStart Called
com.example.activitylifecycle D/ALC2: onResume Called
com.example.activitylifecycle D/ALC3: onStop Called
com.example.activitylifecycle D/ALC3: onDestroy Called
```



Reason =>Rotation-Action Problem:

- On screen rotation `onDestroy()` is called so value of count doesn't get saved and after rotation activity starts again .



Solution =>Rotation-Action Problem:

- Save values before destroy

Add the following method in the activity and save value of count in the bundle. So that value of count can be stored before call of `onDestory()` and reused later on.

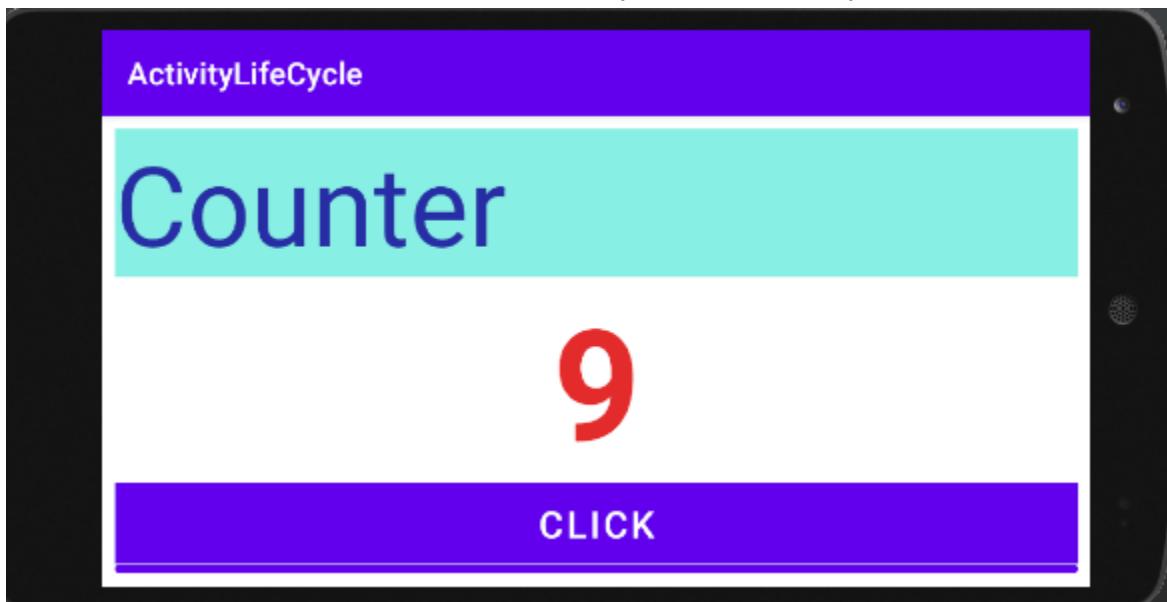
```
@Override  
protected void onSaveInstanceState(@NonNull Bundle outState) {  
    super.onSaveInstanceState(outState);  
    outState.putInt("value", count);
```

```
}
```

Put an if condition to check instance state and display value stored by bundle.

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
  
    if (savedInstanceState!=null)  
    {  
        count=savedInstanceState.getInt("value");  
        txtView.setText(String.valueOf(count));  
    }  
}
```

Now check the screen after rotation it will display the count exactly that was before rotation.



Checking Counter on Activity Navigation:

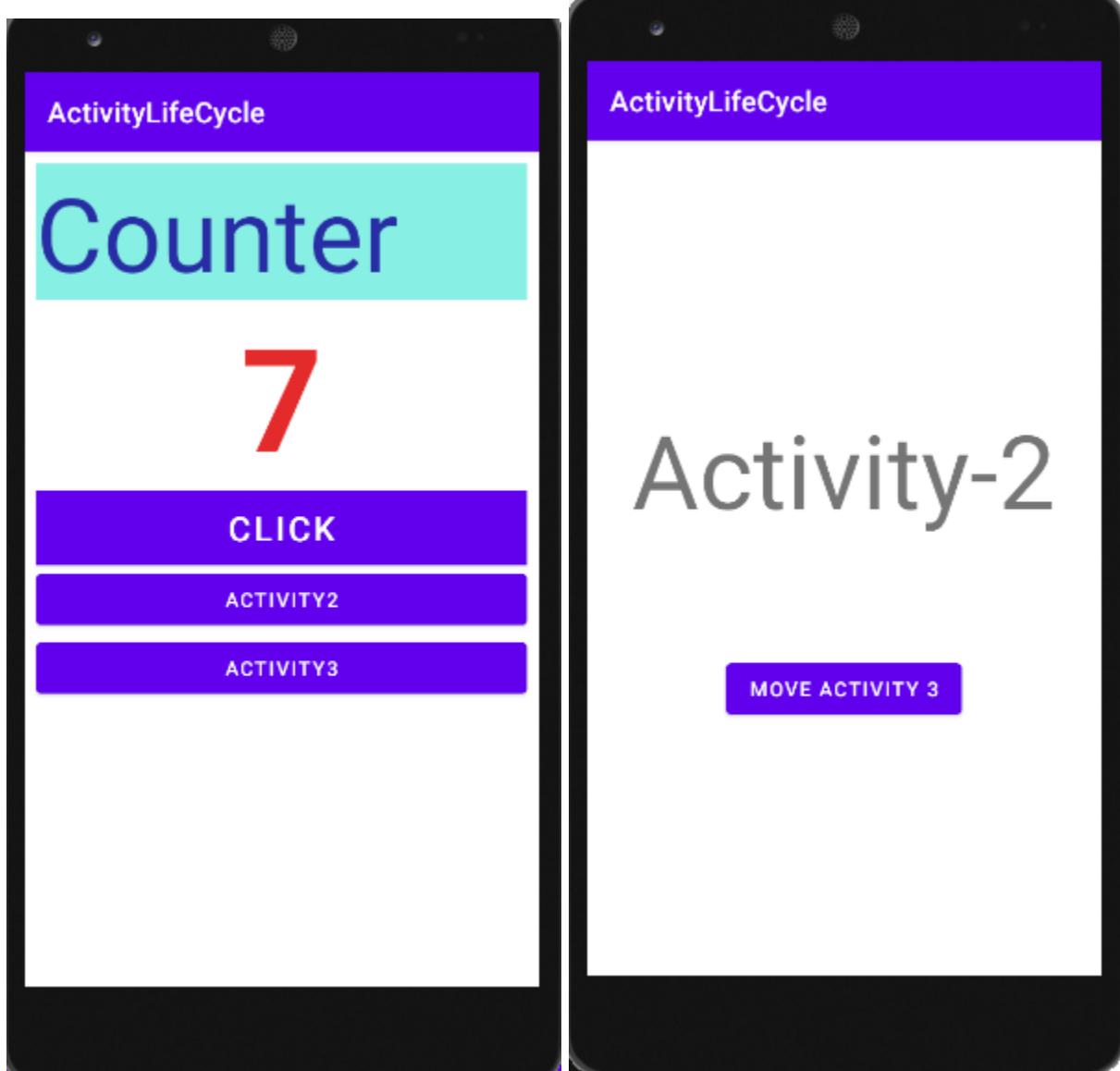
Now comment the display code in if condition.

```
if (savedInstanceState!=null)  
{  
    count=savedInstanceState.getInt( key: "value");  
    //txtView.setText(String.valueOf(count));  
}
```

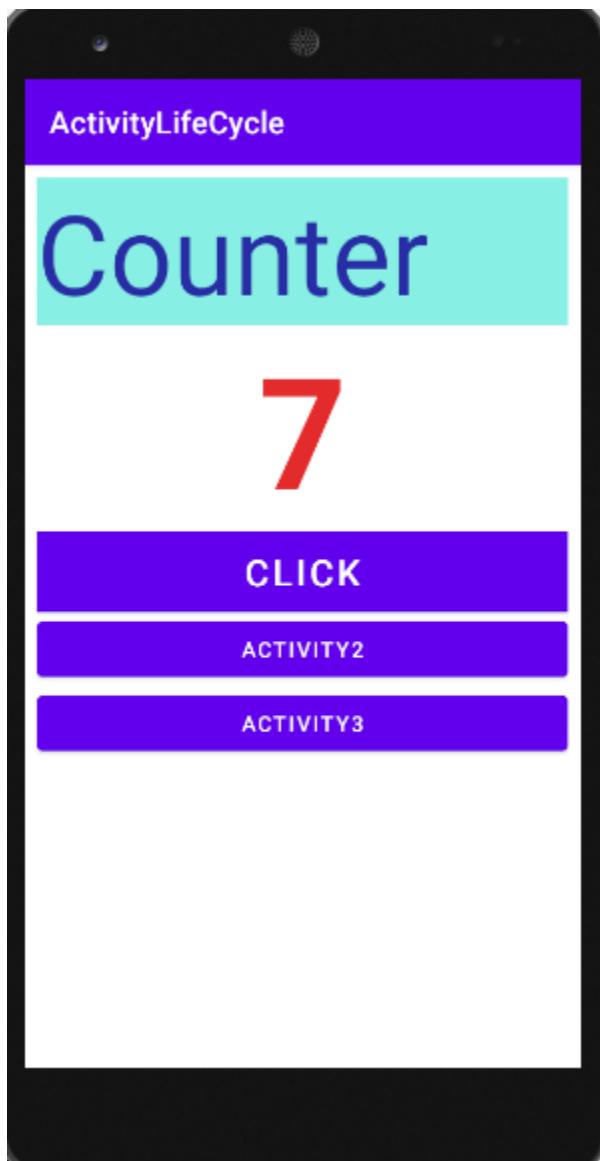
Run the emulator and see add come counts,then rotate and see behavoir.



Now get back to original rotation format and then move to other activity.



Move back from second activity and see the behaviour. The count value will be the same as you saved.



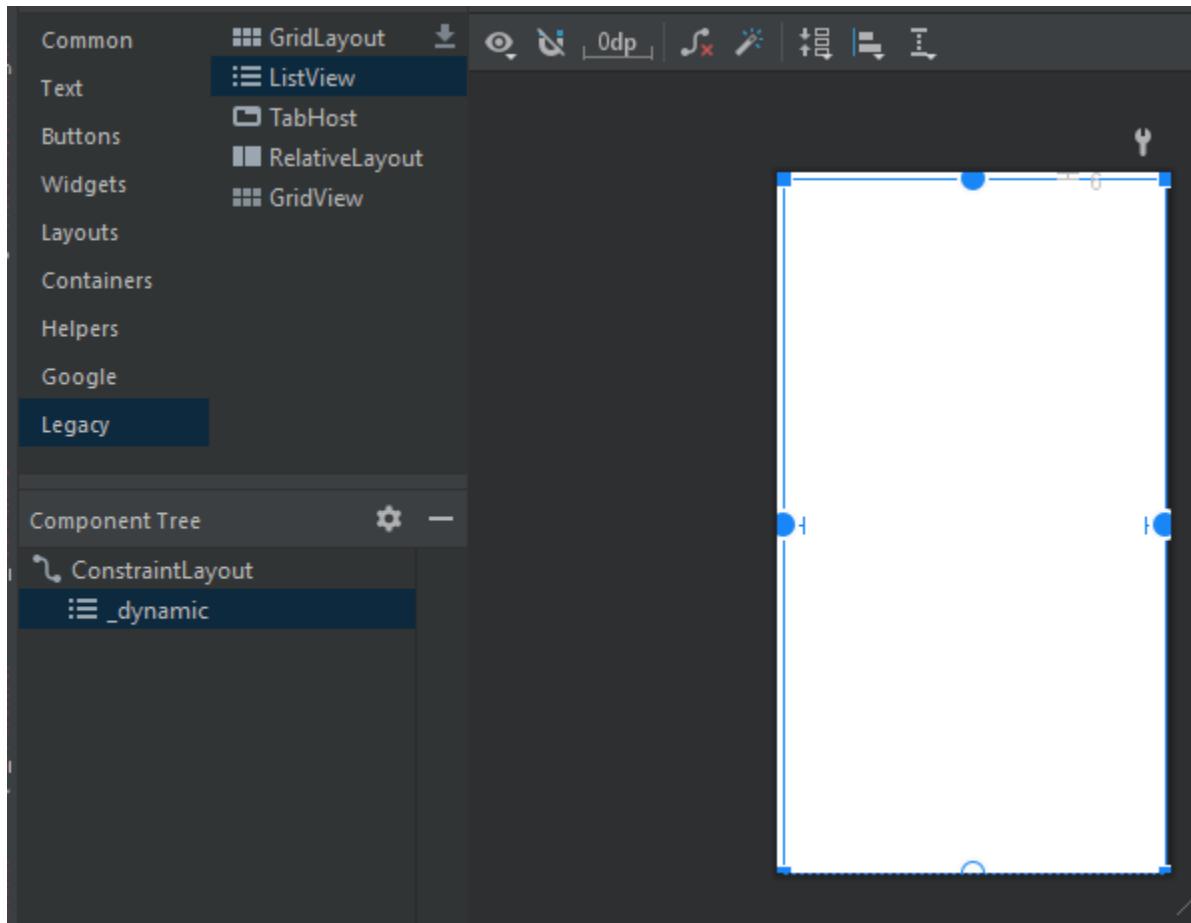
LECTURE#8

❖ Android Adapter:

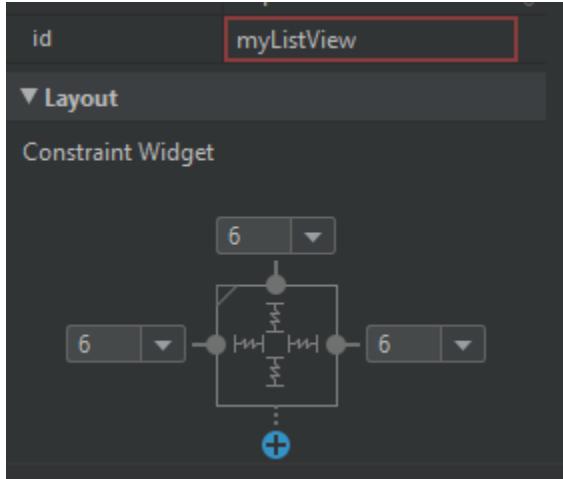
In Android, Adapter is a bridge between UI component and data source that helps us to fill data in UI component. It holds the data and send the data to an Adapter view then view can takes the data from the adapter view and shows the data on different views like as ListView, GridView, Spinner etc.

❖ List View:

- Add a list view from pallet on xml page.



- Set Constraints.



❖ Array :

- Make an array of strings, and display it using **array adapter**.

The screenshot shows the Android Studio IDE. On the left, the code editor displays `MainActivity.java` with the following content:

```

activity_main.xml x MainActivity.java x
import android.widget.ListView;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        ListView listView=findViewById(R.id.myList);

        String [] friendList={"Amber","Eman","Hiba","Rubab",
        ArrayList<String> friendArrayList=new ArrayList<String>;
        friendArrayList.add("ABC");
        friendArrayList.add("XYZ");
        friendArrayList.add("MNO");
        friendArrayList.add("PQR");
        friendArrayList.add("DEF");

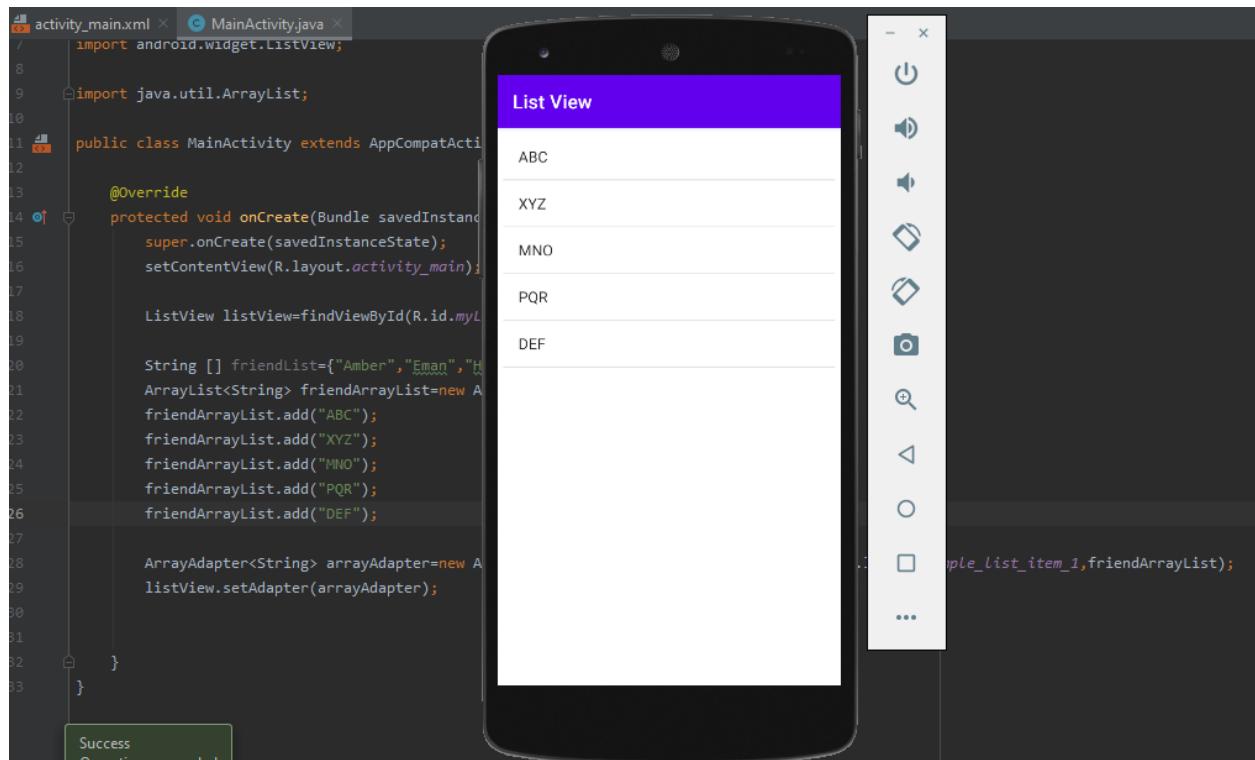
        ArrayAdapter<String> arrayAdapter=new ArrayAdapter<String>(this,R.layout.list_item,friendList);
        listView.setAdapter(arrayAdapter);
    }
}

```

A green box highlights the line `Success Operation succeeded`. On the right, a preview window shows a smartphone displaying a list titled "List View" with items: Amber, Eman, Hiba, Rubab, Amna.

❖ Array List:

Make an ArrayList of strings, and display it using **array adapter**.



The screenshot shows the Android Studio interface with two tabs open: `activity_main.xml` and `MainActivity.java`. The `MainActivity.java` tab contains the following Java code:

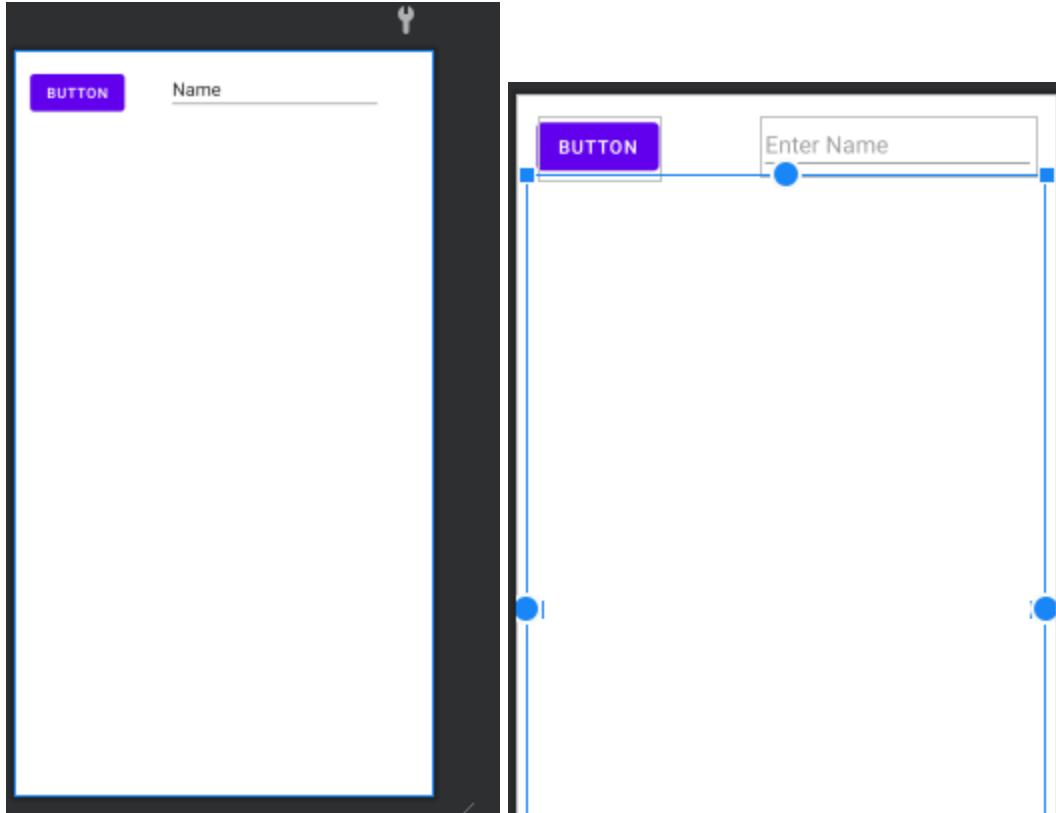
```
1 import android.widget.ListView;
2
3 import java.util.ArrayList;
4
5 public class MainActivity extends AppCompatActivity {
6
7     @Override
8     protected void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10        setContentView(R.layout.activity_main);
11
12        ListView listView=findViewById(R.id.listView);
13
14        String [] friendList={"Amber","Eman","Huda"};
15        ArrayList<String> friendArrayList=new ArrayList();
16        friendArrayList.add("ABC");
17        friendArrayList.add("XYZ");
18        friendArrayList.add("MNO");
19        friendArrayList.add("PQR");
20        friendArrayList.add("DEF");
21
22        ArrayAdapter<String> arrayAdapter=new ArrayAdapter(this,R.layout.simple_list_item_1,friendArrayList);
23        listView.setAdapter(arrayAdapter);
24    }
25}
```

The `activity_main.xml` tab shows a basic layout with a list view.

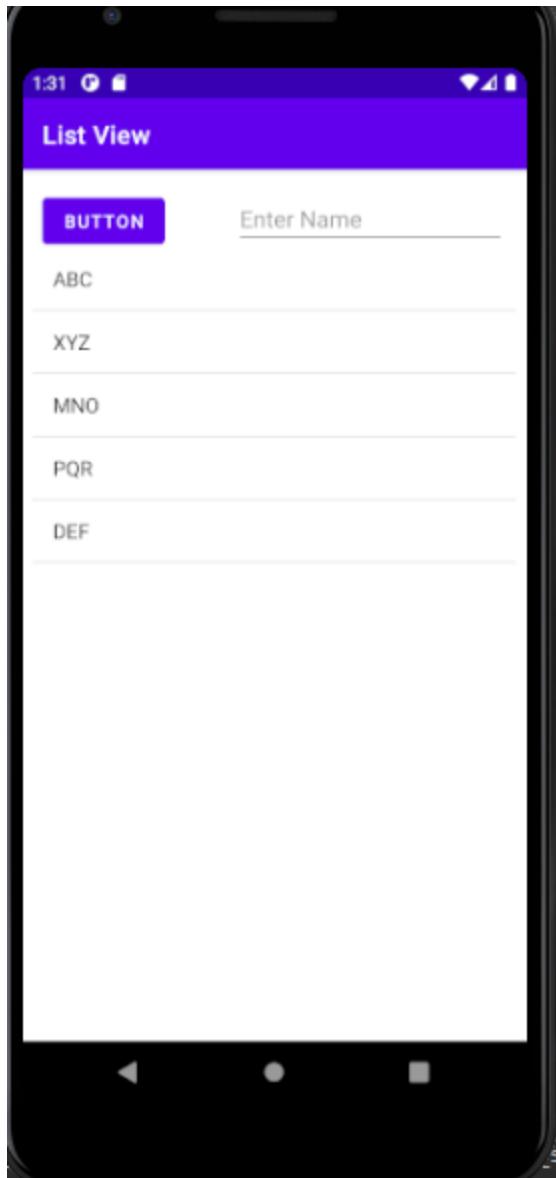
The right side of the screen shows a virtual device running the application. The title bar says "List View". The list view displays five items: ABC, XYZ, MNO, PQR, and DEF.

❖ Add new Record

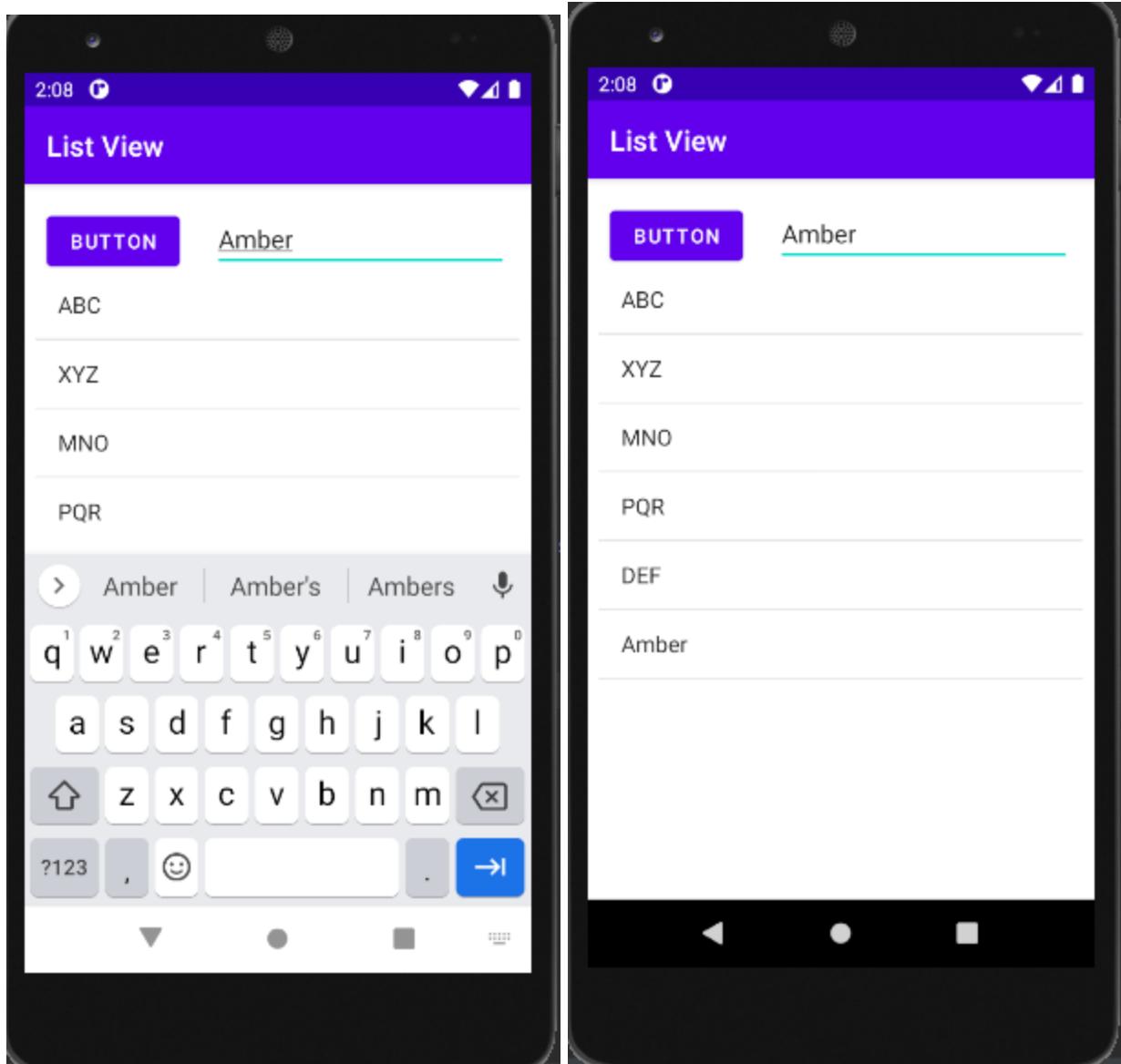
- In the list view, Add a button and plain text.
- Then set their constraints in such a way that list is displayed under them.



- The layout will be like the image below.

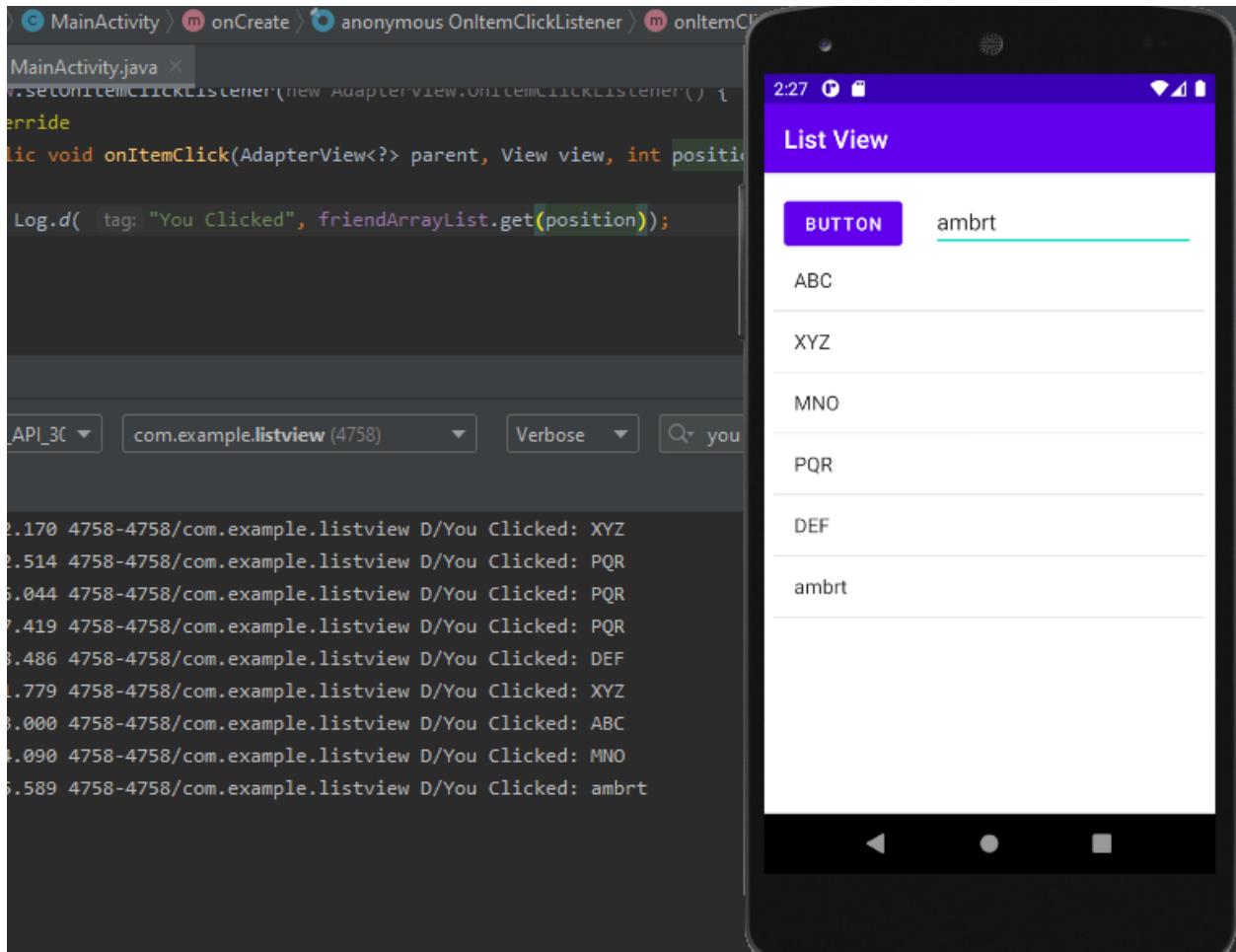


- On click event of listener add text to list using function.



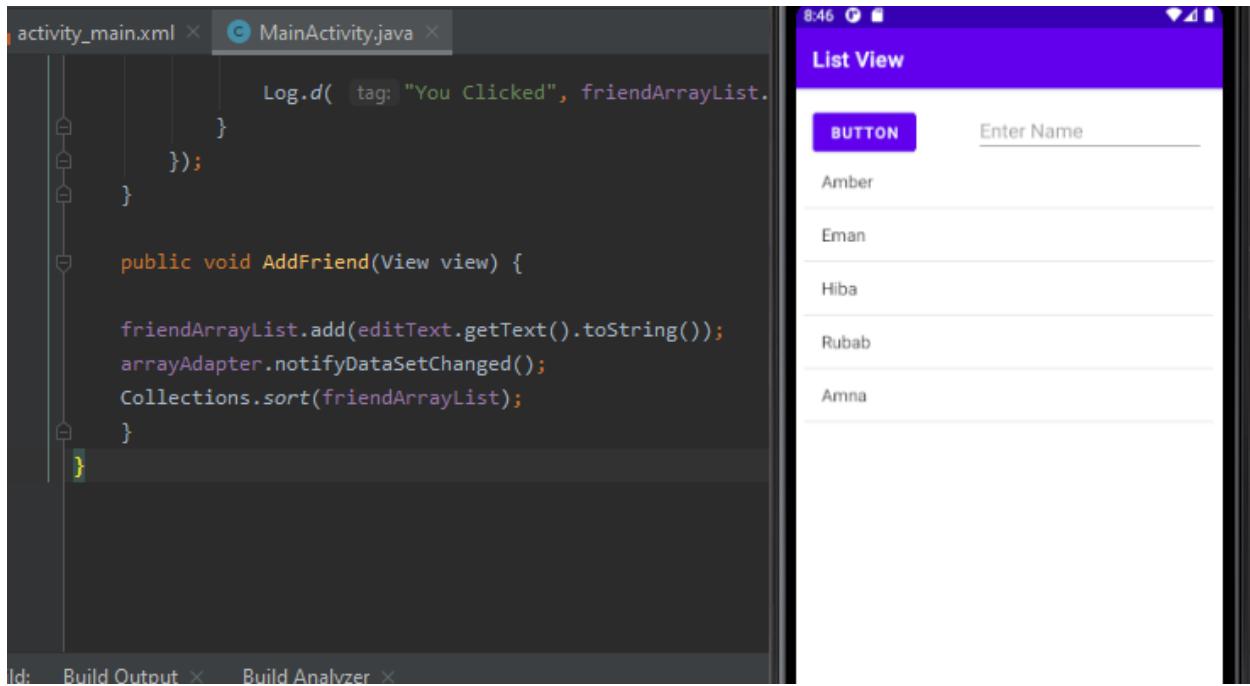
❖ Log Checking List Items :

- Check the clicked item in log using the log function in activity.java file.

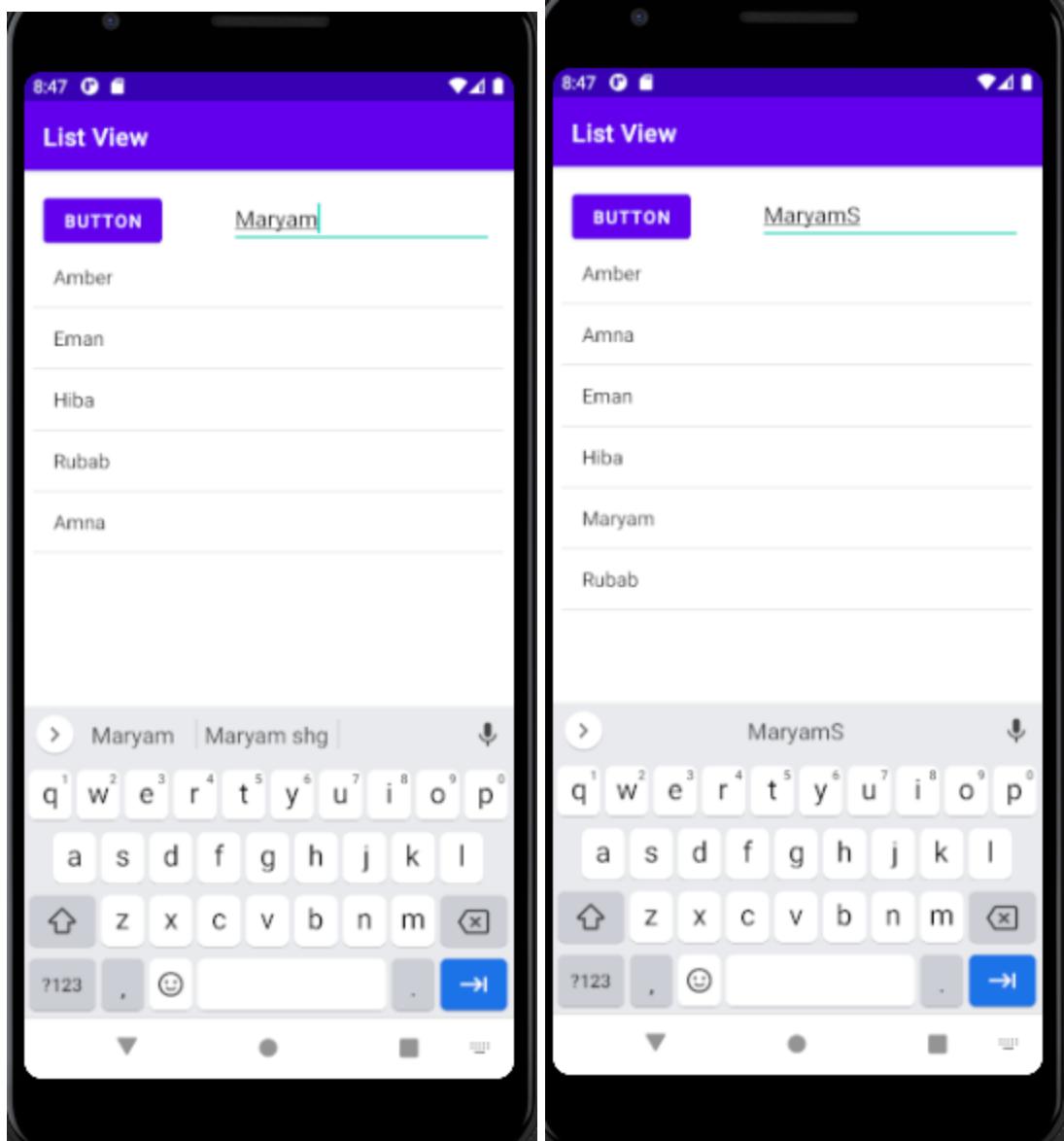


❖ Sorting:

- Array list items can be sorted using the functions as shown below.

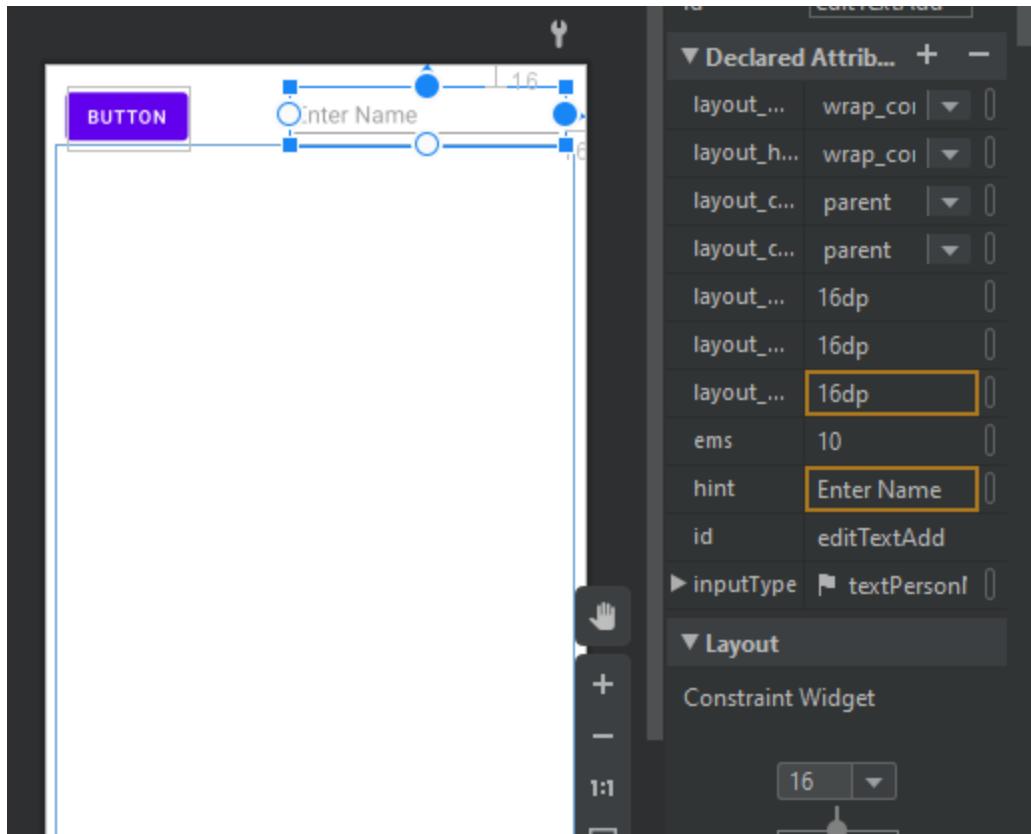


- After using the sort function add new items to list and check the their sorting accordingly.



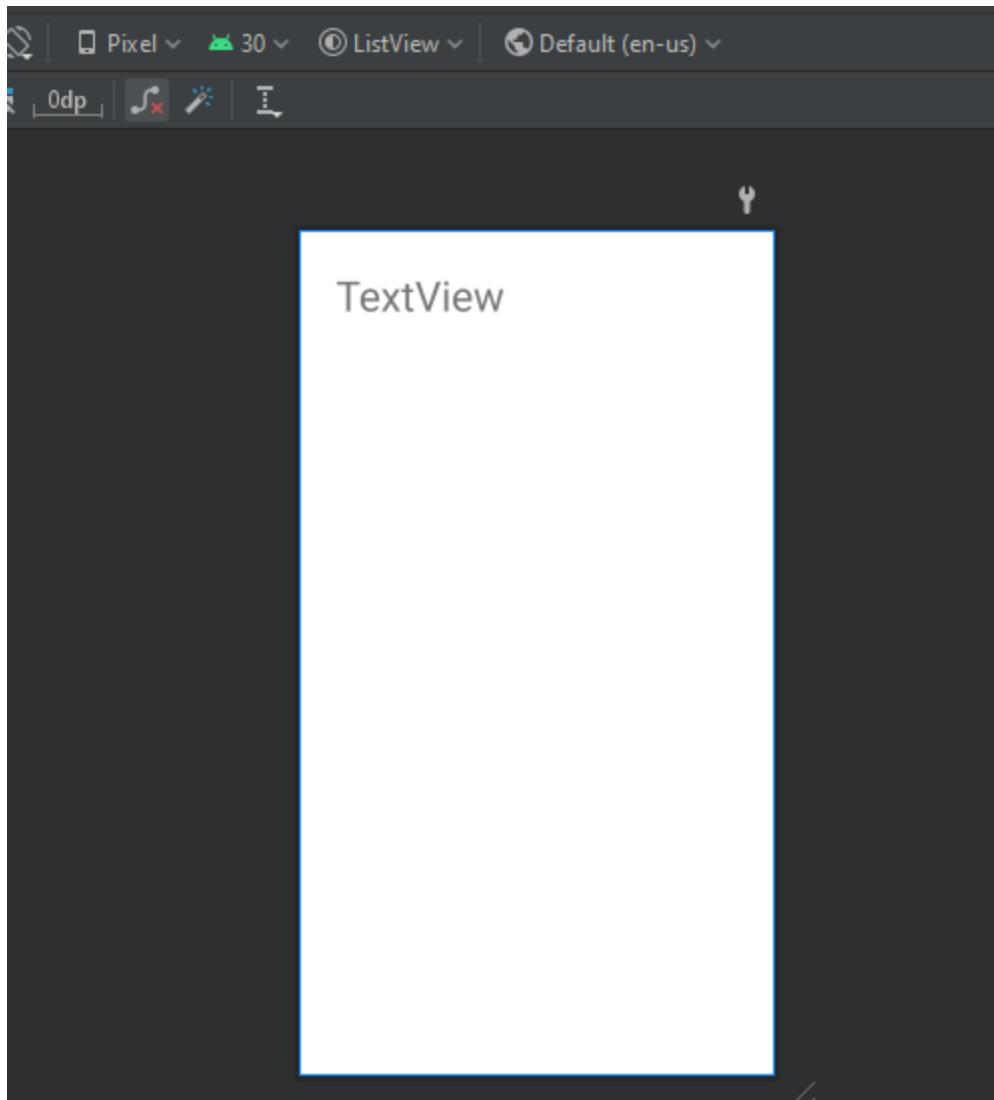
- **Add Hint in Text View:**

- Instead of using some text add hint to text view to facilitate user to add their own text while clicking on the view.



❖ Pass Data from One Activity to Another:

- To pass the data from one activity to another, add new activity and then add a text view in it.



- Then make an intent as below to move from 1st activity to second.

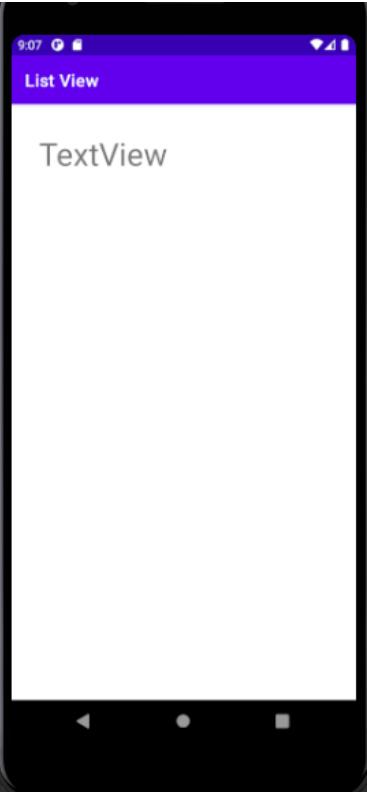
The image shows the Android Studio environment. On the left, the code editor displays Java code for a ListView adapter:

```
friendArrayList=new ArrayList<String>(); //Array List
friendArrayList.add("Amber");
friendArrayList.add("Eman");
friendArrayList.add("Hiba");
friendArrayList.add("Rubab");
friendArrayList.add("Amna");

arrayAdapter=new ArrayAdapter<String>( context: this, android.R.layout.simple_list_item_1,friendArrayList);
listView.setAdapter(arrayAdapter);
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        Log.d( tag: "You Clicked", friendArrayList.get(position));
        Intent intent=new Intent( packageContext: MainActivity.this,DetailActivity.class);
        startActivity(intent);
    }
})
```

On the right, an emulator window titled "List View" shows a list of names: Amber, Amna, Eman, Hiba, and Rubab. Each name is preceded by a small purple button icon. There is also a "BUTTON" button at the top left and an "Enter Name" input field at the top right.

- Check the activity navigation on emulator.



The screenshot shows a split-screen view. On the left is the Android Studio code editor with the file `MainActivity.java` open. It contains Java code for setting up a list view with friends' names. On the right is an Android emulator window titled "List View" showing a single item labeled "TextView".

```

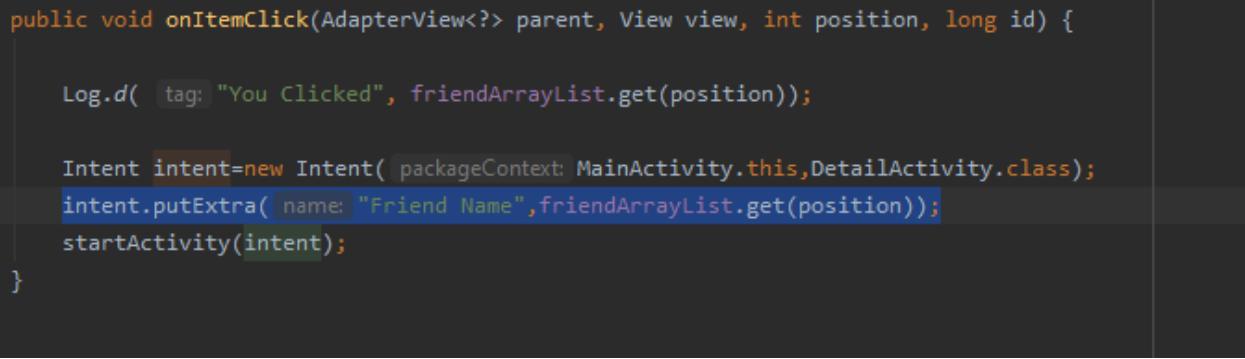
friendArrayList=new ArrayList<String>();//Array List
friendArrayList.add("Amber");
friendArrayList.add("Eman");
friendArrayList.add("Hiba");
friendArrayList.add("Rubab");
friendArrayList.add("Anna");

arrayAdapter=new ArrayAdapter<String>( context: this, android.R.layout.simple_list_item_1,friendArrayList);
listView.setAdapter(arrayAdapter);
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        Log.d( tag: "You Clicked", friendArrayList.get(position));
        Intent intent=new Intent( packageContext: MainActivity.this,DetailActivity.class);
        startActivity(intent);
    }
});

void AddFriend(View view) {
    friendArrayList.add(editText.getText().toString());
    adapter.notifyDataSetChanged();
    friendArrayList.sort(friendArrayList);
}

```

- Now set the clicked item of list of first activity as text of second activity.

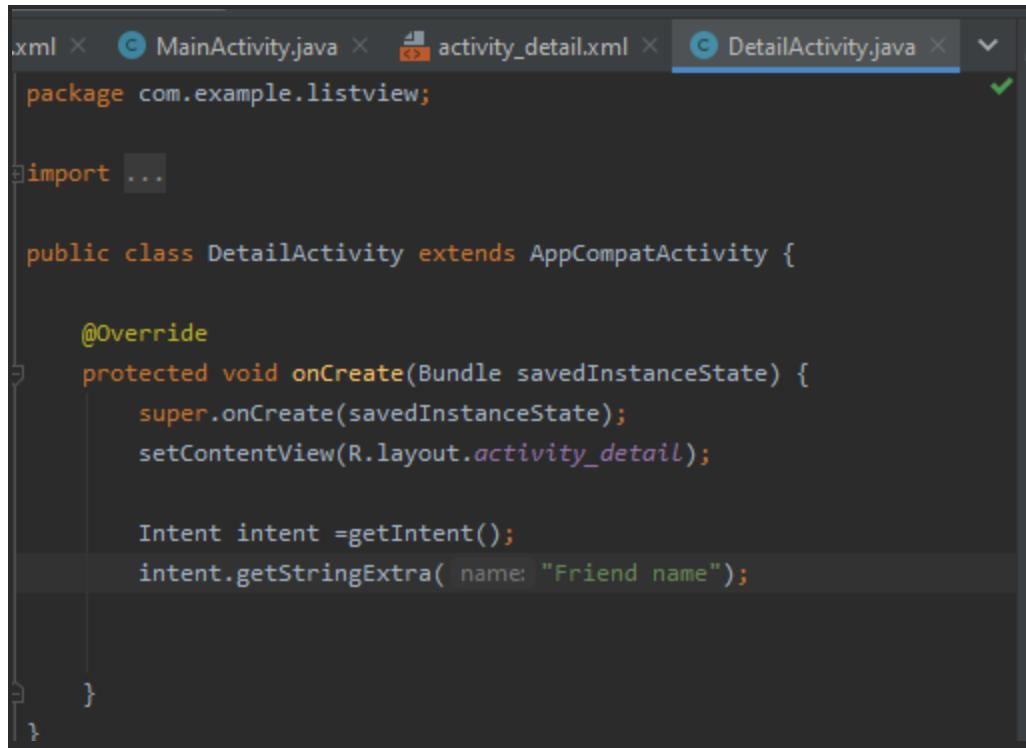


The screenshot shows the `MainActivity.java` code editor with the `onItemClick` method modified to include an intent extra for the friend's name.

```

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
    Log.d( tag: "You Clicked", friendArrayList.get(position));
    Intent intent=new Intent( packageContext: MainActivity.this,DetailActivity.class);
    intent.putExtra( name: "Friend Name",friendArrayList.get(position));
    startActivity(intent);
}

```



```
xml × C MainActivity.java × activity_detail.xml × C DetailActivity.java × ✓
package com.example.listview;

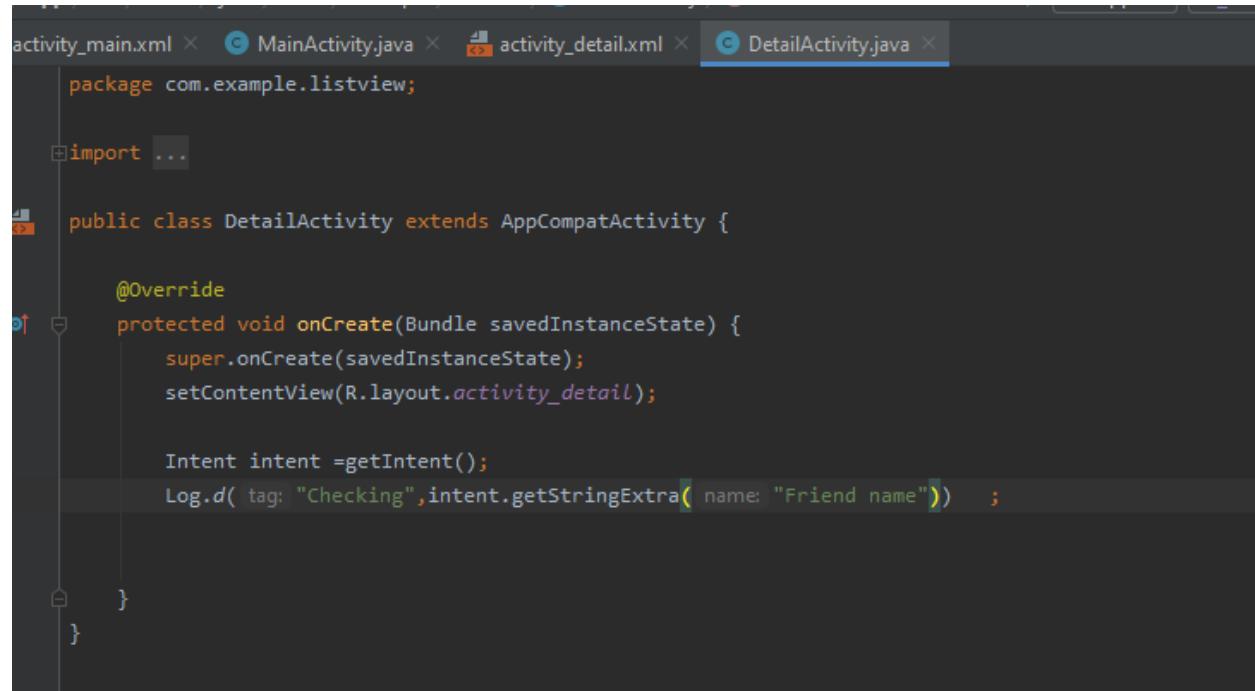
import ...

public class DetailActivity extends AppCompatActivity {

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

        Intent intent = getIntent();
        intent.getStringExtra("Friend name");
    }
}
```

- Place a log function to see the clicked items in one activity and checked in second activity.



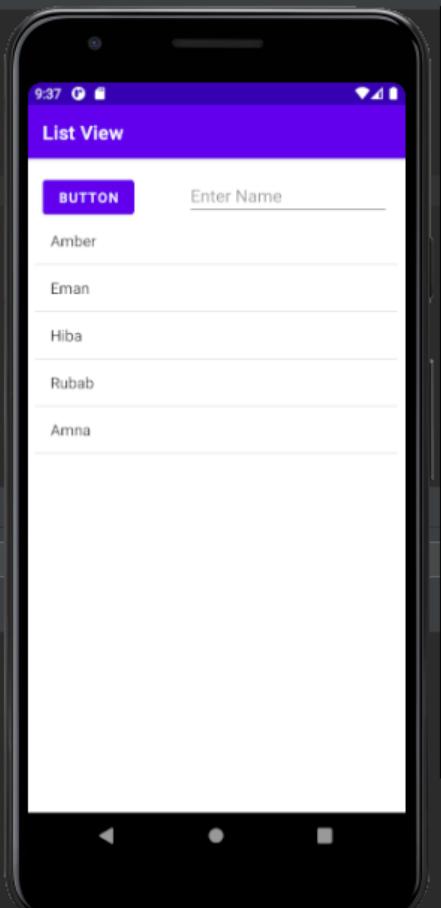
```
activity_main.xml × C MainActivity.java × activity_detail.xml × C DetailActivity.java ×
package com.example.listview;

import ...

public class DetailActivity extends AppCompatActivity {

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

        Intent intent = getIntent();
        Log.d("Checking", intent.getStringExtra("Friend name"));
    }
}
```



The screenshot shows an Android development environment with the following details:

- Code Editor:** The main.java file contains Java code for the DetailActivity. The relevant part of the code is:

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_detail);

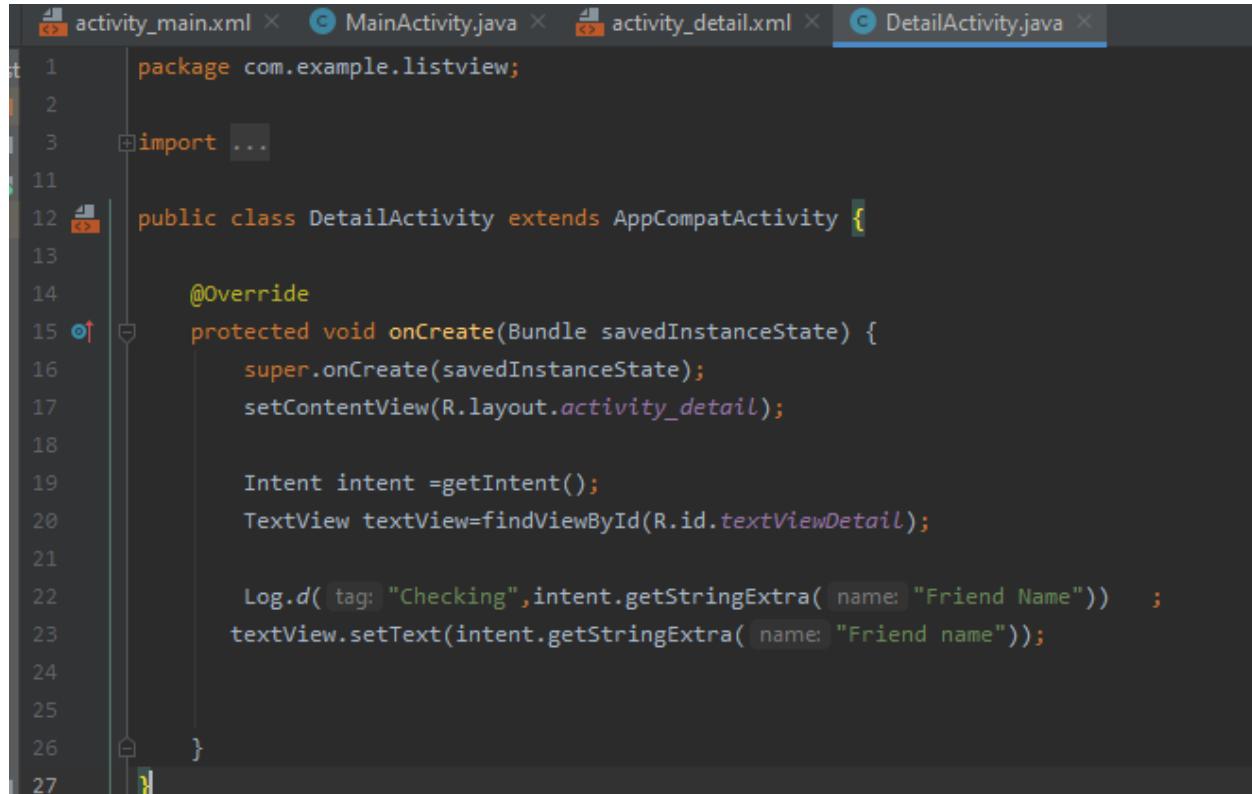
    Intent intent = getIntent();
    Log.d("Checking", intent.getStringExtra("Friend Name"));
    // intent.getStringExtra("Friend name");
}
```

- Logcat:** The log output shows several entries indicating name selection:

```
1:36:04.382 9039-9039/com.example.listview D/You Clicked: Eman
1:36:04.460 9039-9039/com.example.listview D/Checking: Eman
1:36:11.580 9039-9039/com.example.listview D/You Clicked: Amber
1:36:11.687 9039-9039/com.example.listview D/Checking: Amber
1:36:14.321 9039-9039/com.example.listview D/You Clicked: Rubab
1:36:14.505 9039-9039/com.example.listview D/Checking: Rubab
1:36:22.667 9039-9039/com.example.listview D/You Clicked: Hiba
1:36:22.754 9039-9039/com.example.listview D/Checking: Hiba
```

- Emulator:** The emulator displays a screen titled "List View" with a purple header. It features a "BUTTON" button and an "Enter Name" input field. Below these are five list items: "Amber", "Eman", "Hiba", "Rubab", and "Amna".

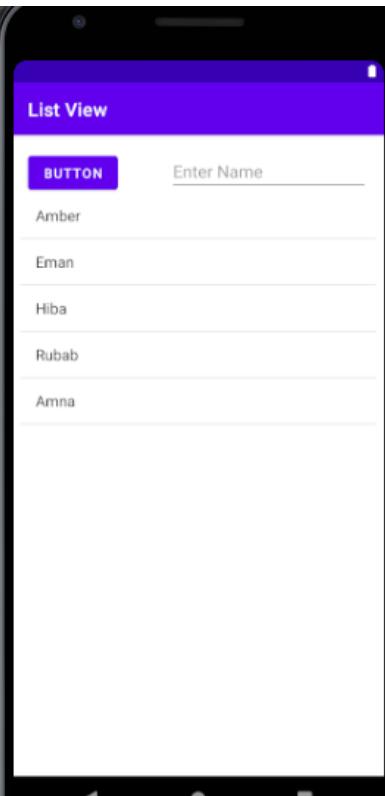
- Get the clicked item in second activity to display when clicked.



The screenshot shows the Android Studio interface with four tabs at the top: activity_main.xml, MainActivity.java, activity_detail.xml, and DetailActivity.java. The DetailActivity.java tab is active, displaying the following Java code:

```
1 package com.example.listview;
2
3 import ...
4
5 public class DetailActivity extends AppCompatActivity {
6
7     @Override
8     protected void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10        setContentView(R.layout.activity_detail);
11
12        Intent intent = getIntent();
13        TextView textView=findViewById(R.id.textViewDetail);
14
15        Log.d("tag: " + "Checking", intent.getStringExtra("name: " + "Friend Name"));
16        textView.setText(intent.getStringExtra("name: " + "Friend name"));
17    }
18}
```

- Now see the output on emulator, as we click any item in activity one it will be displayed in second activity.



```
package com.example.listview;

import ...

public class DetailActivity extends AppCompatActivity {

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

        TextView txtView=findViewById(R.id.textViewDetail);
        Intent intent =getIntent();

        Log.d( tag: "Checking",intent.getStringExtra( name: "Friend Name") );
        txtView.setText(intent.getStringExtra( name: "Friend Name"));

    }
}
```



```
activity_main.xml × activity_detail.xml × DetailActivity.java ×

package com.example.listview;

import ...

public class DetailActivity extends AppCompatActivity {

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

        TextView txtView=findViewById(R.id.textViewDetail);
        Intent intent =getIntent();

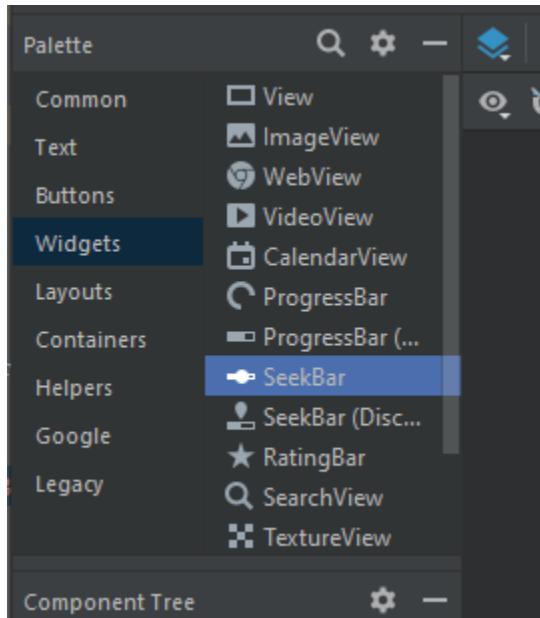
        Log.d( tag: "Checking",intent.getStringExtra( name: "Friend Name") );
        txtView.setText(intent.getStringExtra( name: "Friend Name"));

    }
}
```

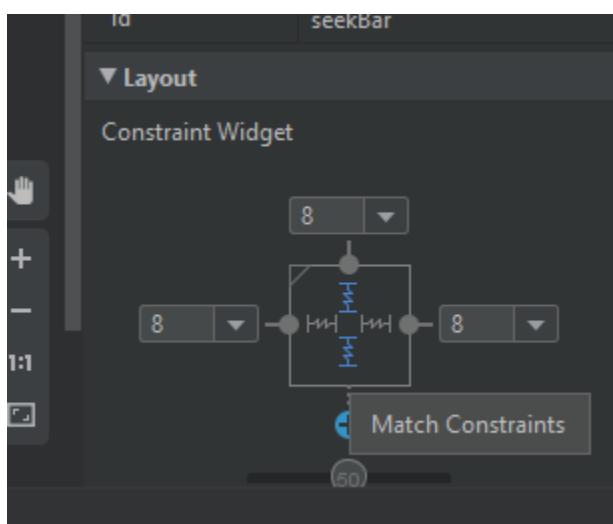
LECTURE#9

❖ Seek bar:

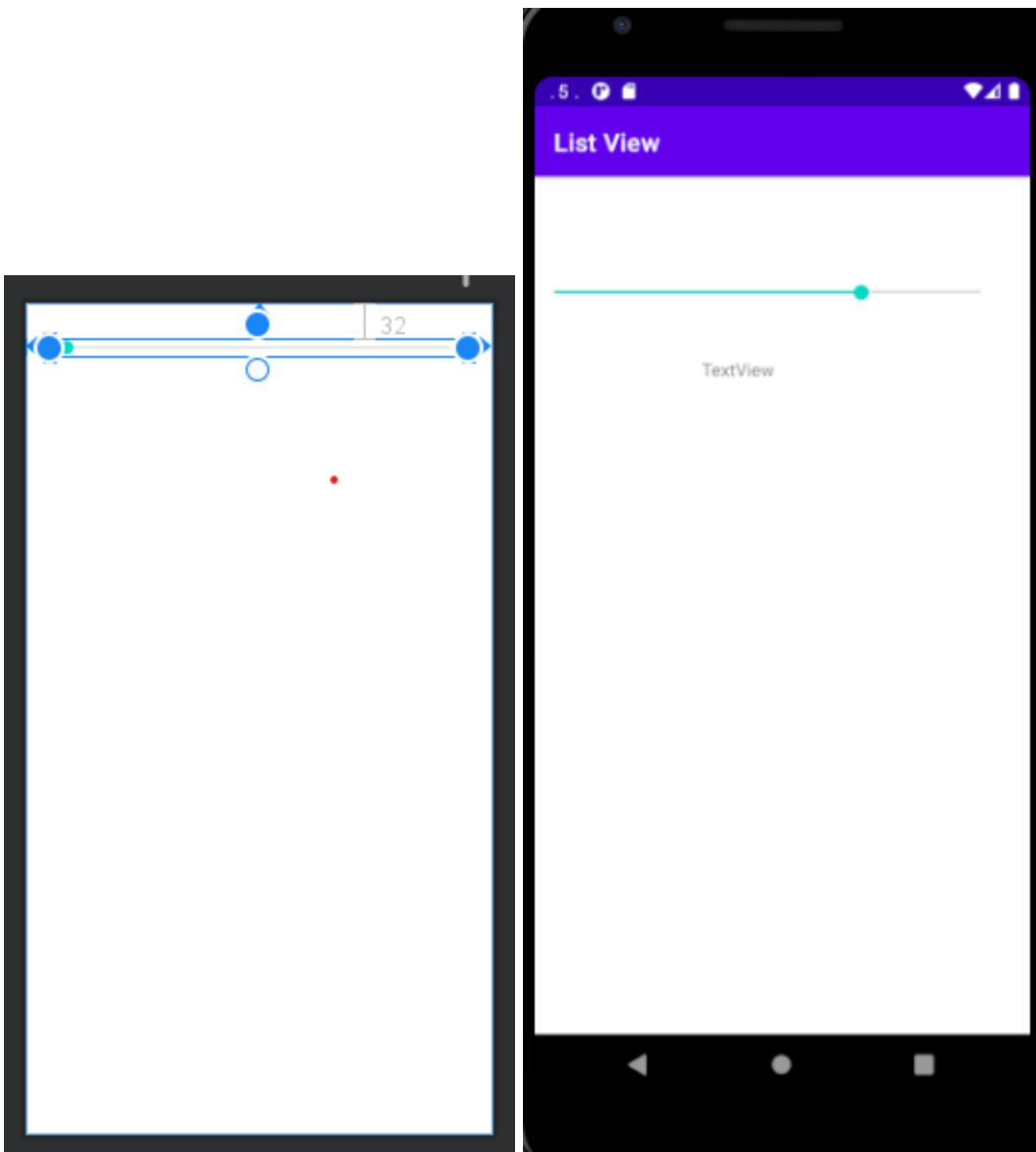
- Add seek bar from widgets on xml file.



- Set constraints of seek bar.

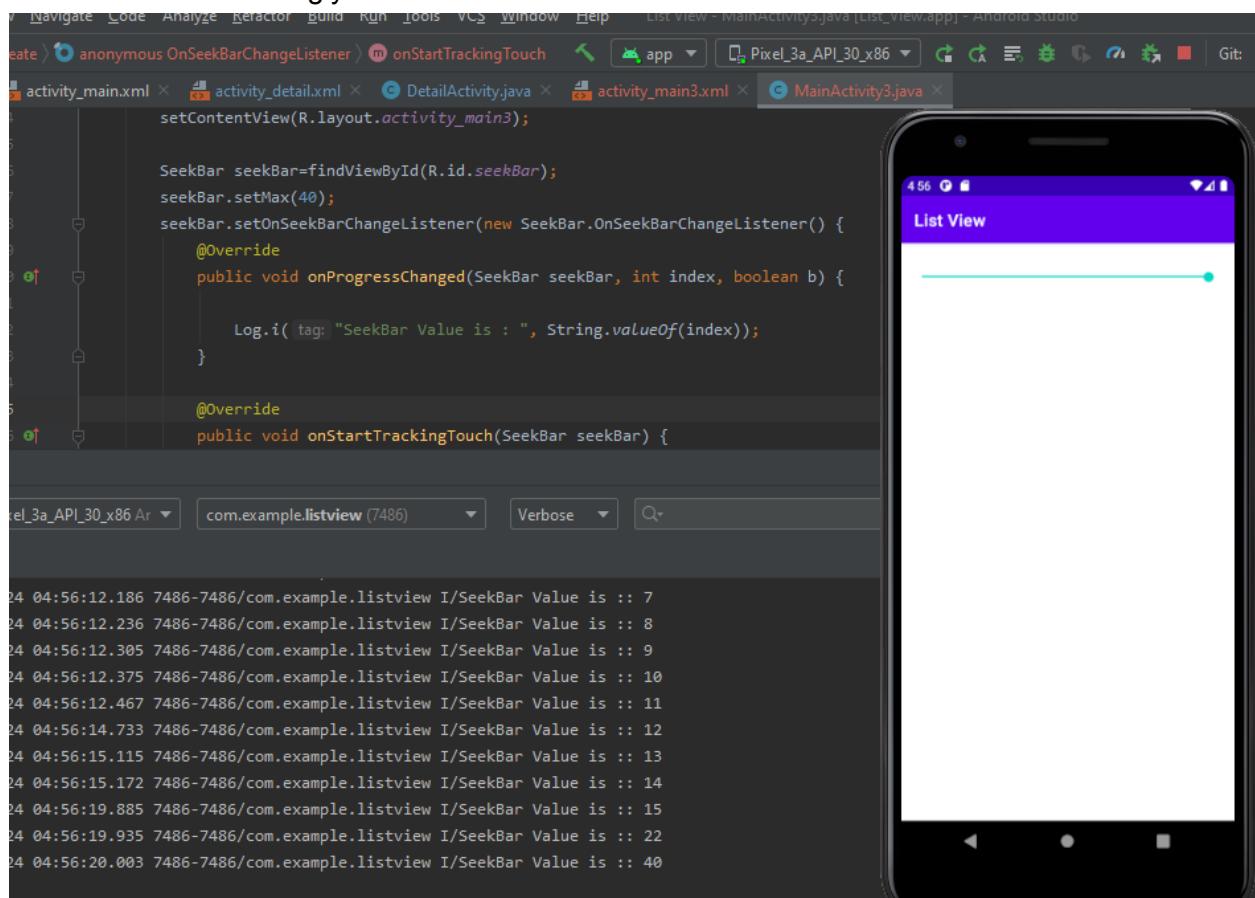


- After setting the position of seek bar , run it ti see its view on emulator.



❖ Setting Max value and read it in log:

- Set maximum value of seek bar using function as below and check its output values by using log.
- As we run the emulator and move the seek bar the values will be displayed in log accordingly.



The screenshot shows the Android Studio interface with the code editor open to `MainActivity3.java`. The code defines a `SeekBar` and sets its maximum value to 40. It also implements the `OnSeekBarChangeListener` interface to log the seek bar's progress whenever it changes.

```
    setContentView(R.layout.activity_main3);

    SeekBar seekBar=findViewById(R.id.seekBar);
    seekBar.setMax(40);
    seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
        @Override
        public void onProgressChanged(SeekBar seekBar, int index, boolean b) {
            Log.i("tag", "SeekBar Value is :: " + String.valueOf(index));
        }

        @Override
        public void onStartTrackingTouch(SeekBar seekBar) {
    
```

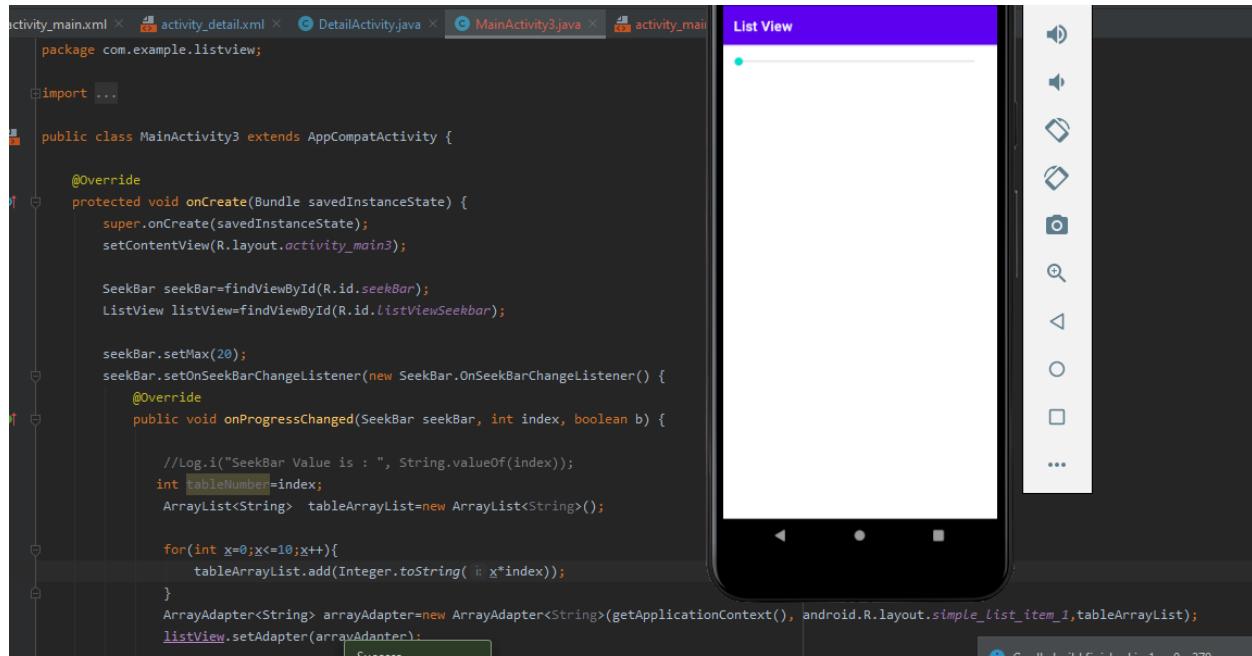
The logcat window shows the following output, indicating the seek bar's value is being printed every time it is moved:

```
24 04:56:12.186 7486-7486/com.example.listview I/SeekBar Value is :: 7
24 04:56:12.236 7486-7486/com.example.listview I/SeekBar Value is :: 8
24 04:56:12.305 7486-7486/com.example.listview I/SeekBar Value is :: 9
24 04:56:12.375 7486-7486/com.example.listview I/SeekBar Value is :: 10
24 04:56:12.467 7486-7486/com.example.listview I/SeekBar Value is :: 11
24 04:56:14.733 7486-7486/com.example.listview I/SeekBar Value is :: 12
24 04:56:15.115 7486-7486/com.example.listview I/SeekBar Value is :: 13
24 04:56:15.172 7486-7486/com.example.listview I/SeekBar Value is :: 14
24 04:56:19.885 7486-7486/com.example.listview I/SeekBar Value is :: 15
24 04:56:19.935 7486-7486/com.example.listview I/SeekBar Value is :: 22
24 04:56:20.003 7486-7486/com.example.listview I/SeekBar Value is :: 40
```

The right side of the interface shows an emulator running an application titled "List View". A horizontal seek bar is visible on the screen, which corresponds to the one defined in the Java code.

❖ To print table of number on seekbar:

- Use seek bar function to get its value and store it in table.



The screenshot shows the Android Studio interface with several tabs open: activity_main.xml, activity_detail.xml, DetailActivity.java, MainActivity3.java (which is the active tab), and activity_main. The code in MainActivity3.java is as follows:

```
activity_main.xml x activity_detail.xml x DetailActivity.java x MainActivity3.java x activity_main
package com.example.listview;

import ...

public class MainActivity3 extends AppCompatActivity {

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

        SeekBar seekBar=findViewById(R.id.seekBar);
        ListView listView=findViewById(R.id.listViewSeekBar);

        seekBar.setMax(20);
        seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
            @Override
            public void onProgressChanged(SeekBar seekBar, int index, boolean b) {

                //Log.i("SeekBar Value is : ", String.valueOf(index));
                int tableNumber=index;
                ArrayList<String> tableArrayList=new ArrayList<String>();

                for(int x=0;x<=10;x++){
                    tableArrayList.add(Integer.toString(x*index));
                }
                ArrayAdapter<String> arrayAdapter=new ArrayAdapter<String>(getApplicationContext(), android.R.layout.simple_list_item_1,tableArrayList);
                listView.setAdapter(arrayAdapter);
            }
        });
    }
}
```

The emulator window shows a seekbar at the top with a value of 1. Below it is a list view. The right side of the screen has a vertical toolbar with various icons.

- Add a list view under seek bar in xml file .
- Write a function to display the table of value selected on seek bar.
- Run the emulator to see the output on screen.

The screenshot shows an Android application running on an emulator. The application has a purple header bar with the title "List View". Below the header is a SeekBar with its thumb positioned at the center. Underneath the SeekBar is a ListView displaying a list of numbers from 6 to 60, each in its own row. At the bottom of the screen, there is a navigation bar with three icons. The status bar at the top shows the time as 5:47 and various signal strength indicators.

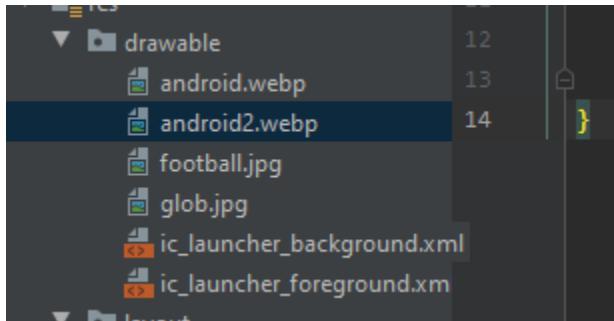
```
onProgressChanged(SeekBar seekBar, int index, boolean fromUser) {
    tag: "SeekBar Value is : ", String.valueOf(index))
    leNumber=index;
    list<String> tableArrayList=new ArrayList<String>();
    for(x=1;x<=10;x++){
        tableArrayList.add(Integer.toString( i: x*index));
    }
    arrayAdapter=new ArrayAdapter<String>(newView.getContext(), R.layout.list_item, R.id.textView1, tableArrayList);
    newView.setAdapter(arrayAdapter);
}

 onStartTrackingTouch(SeekBar seekBar) {
    Log.d("com.example.listview", "I/SeekBar Value is :: 1");
    Log.d("com.example.listview", "I/SeekBar Value is :: 2");
    Log.d("com.example.listview", "I/SeekBar Value is :: 6");
}

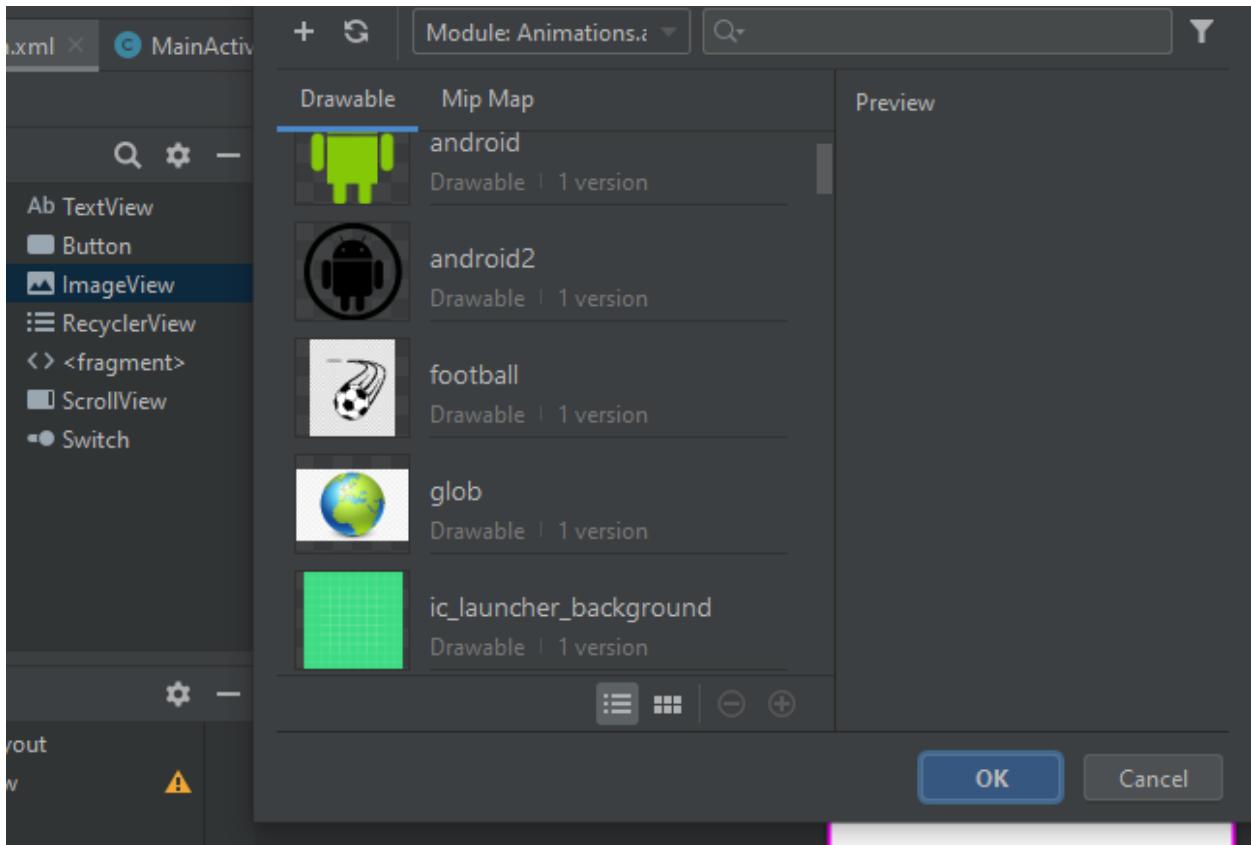
Logcat output:
com.example.listview W/xample.listvie: Checksum mismatch
com.example.listview I/SeekBar Value is :: 1
com.example.listview I/SeekBar Value is :: 2
com.example.listview I/SeekBar Value is :: 6
```

❖ Widgets and Animations:

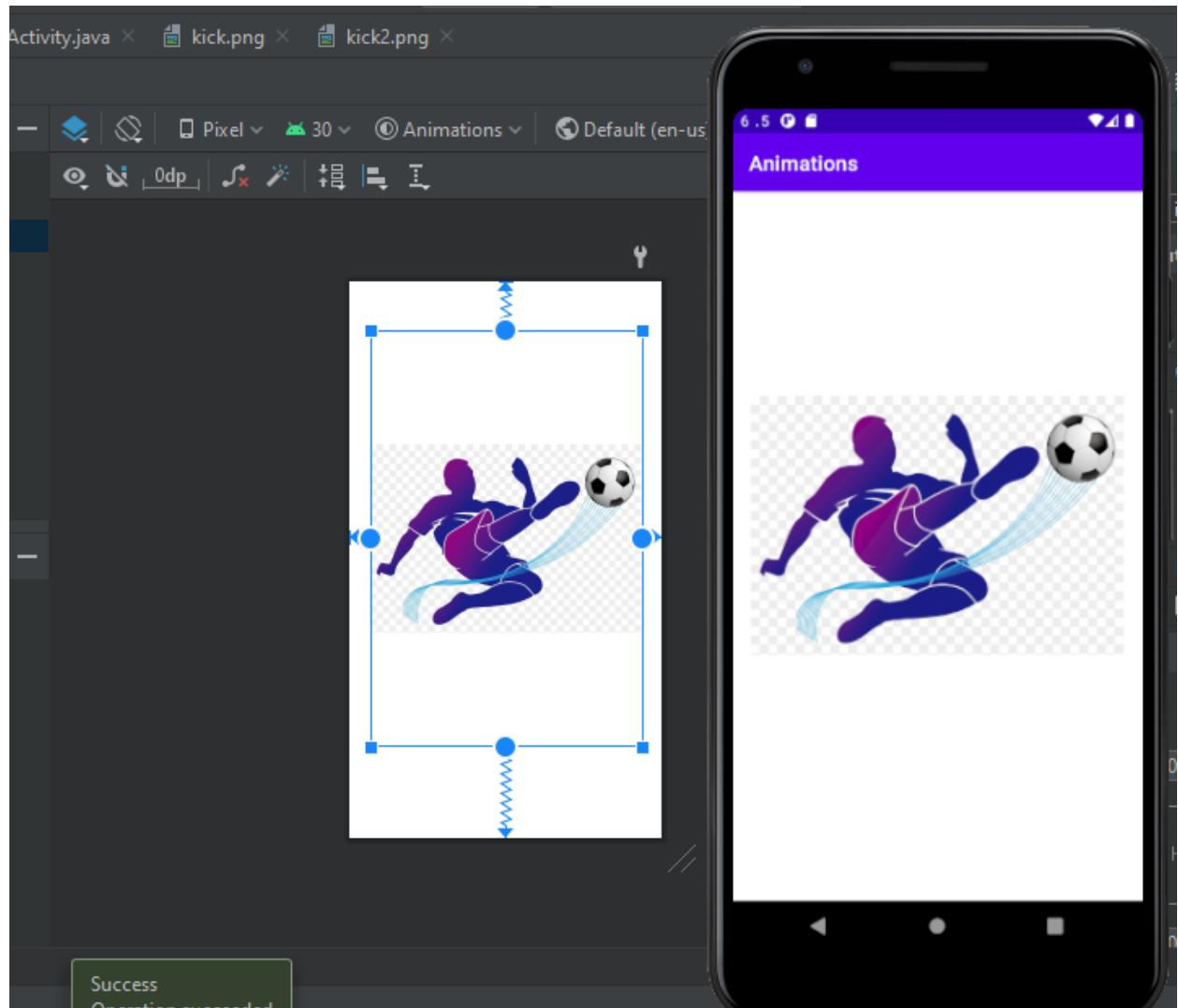
- Copy come images from your PC and paste then in **drawable** folder in your project.



- Add an image view on xml file and select any image you want to add.



- Set image constraints and position.
- Run the emulator to see the output of image.



❖ On click for image:

- Write an onclick function for image to perform actions on image click.

```
<ImageView  
    android:id="@+id/imageView"  
    android:layout_width="357dp"  
    android:layout_height="546dp"  
    android:onClick="Animation"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.351"
```

Basic functions of Animation:

❖ Image Disappear Animation :

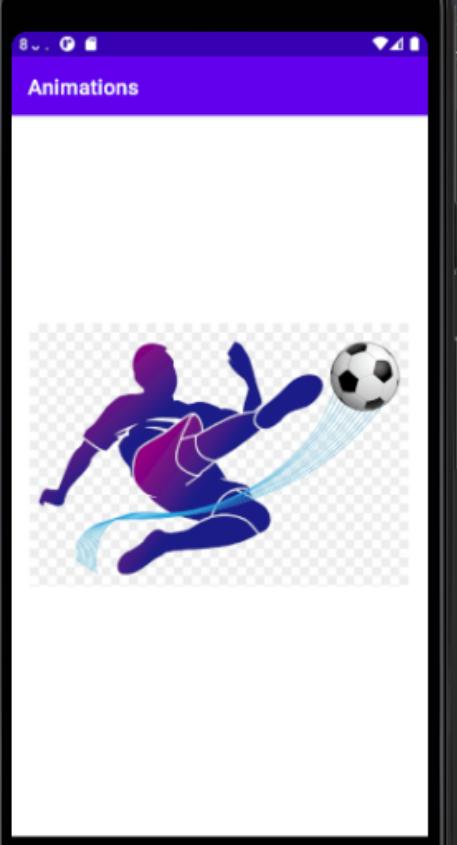
- Write the function as below to get the disappearing effect on image click.

```
activity_main.xml × MainActivity.java × kick.png × kick2.png ×
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

}

public void Animation(View view) {
    ImageView imageView=findViewById(R.id.imageView);
    imageView.animate().alpha(0).setDuration(500);
```

- Image view before click.

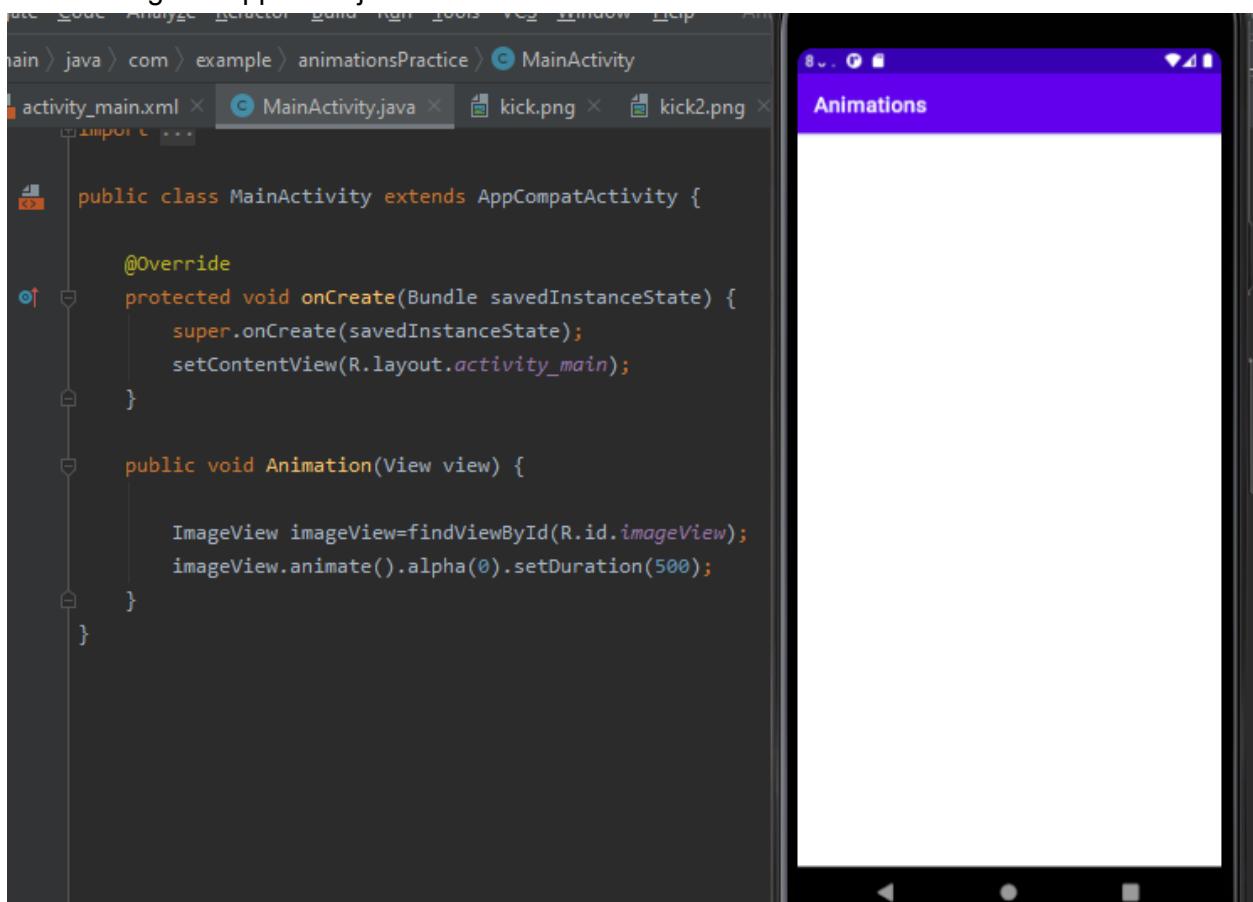


The screenshot shows the Android Studio interface with the code editor and a preview window. The code editor displays the MainActivity.java file with the following code:

```
main > java > com > example > animationsPractice > MainActivity
activity_main.xml × MainActivity.java × kick.png × kick2.png ×
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void Animation(View view) {
        ImageView imageView=findViewById(R.id.imageView);
        imageView.animate().alpha(0).setDuration(500);
    }
}
```

The preview window shows a soccer player in mid-air, performing a bicycle kick on a soccer ball. The background is white with a checkered pattern at the bottom.

- Image disappeared just after click.



❖ Image TranslationY Animation:

- Write the function as below to get Translation along Y effect on image click.

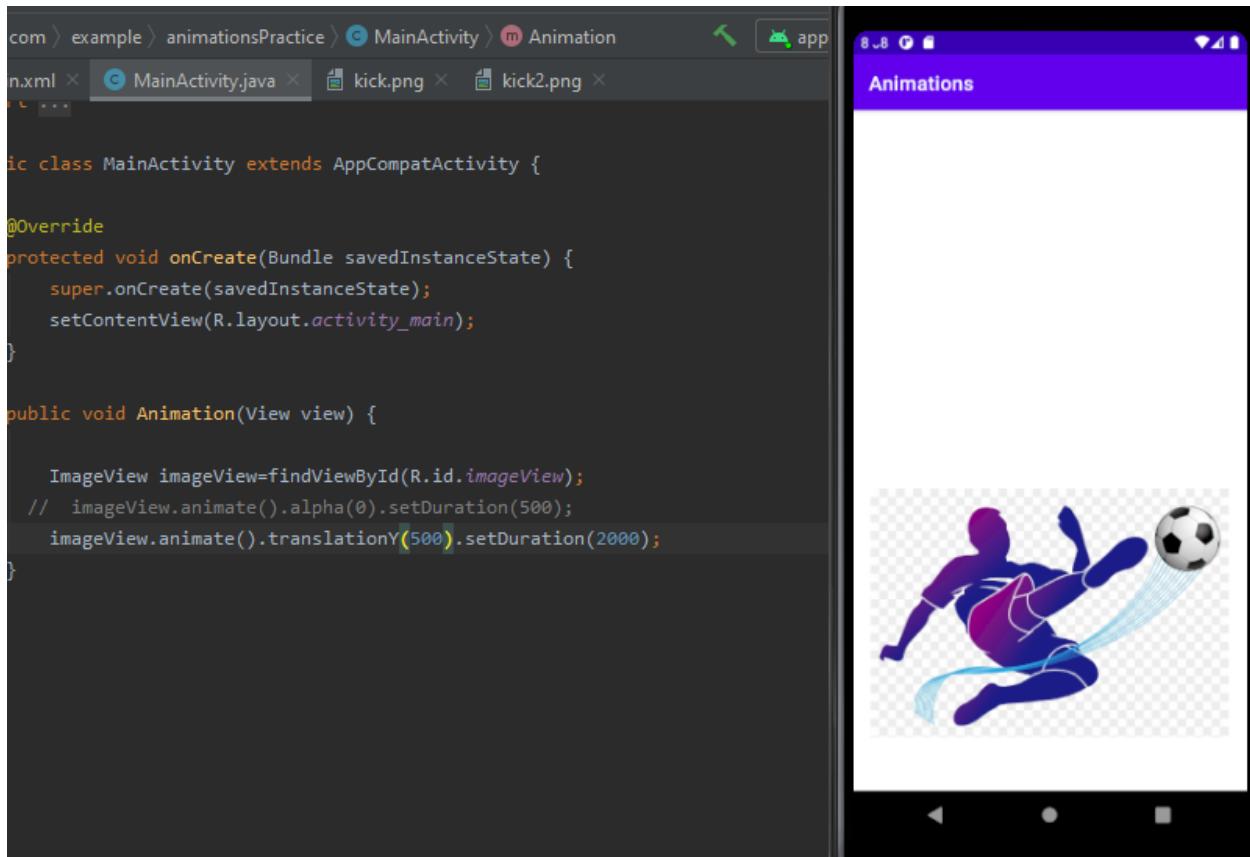
```

public void Animation(View view) {

    ImageView imageView=findViewById(R.id.imageView);
    // imageView.animate().alpha(0).setDuration(500);
    imageView.animate().translationY(500).setDuration(2000);
}

```

- Image moved along Y axis just after click.



❖ Image TranslationX Animation:

- Write the function as below to get Translation along X effect on image click.

```

public void Animation(View view) {

    ImageView imageView=findViewById(R.id.imageView);
    // imageView.animate().alpha(0).setDuration(500);
    // imageView.animate().translationY(500).setDuration(2000);
    imageView.animate().translationX(500).setDuration(2000);
}
}

```

- Image moved along X axis just after click.

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** Shows files like activity_main.xml, MainActivity.java, kick.png, and kick2.png.
- MainActivity.java Code:**

```

public class MainActivity extends AppCompatActivity {

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

    public void Animation(View view) {
        ImageView imageView=findViewById(R.id.imageView);
        // imageView.animate().alpha(0).setDuration(500);
        // imageView.animate().translationY(500).setDuration(2000);
        imageView.animate().translationX(500).setDuration(2000);
    }
}

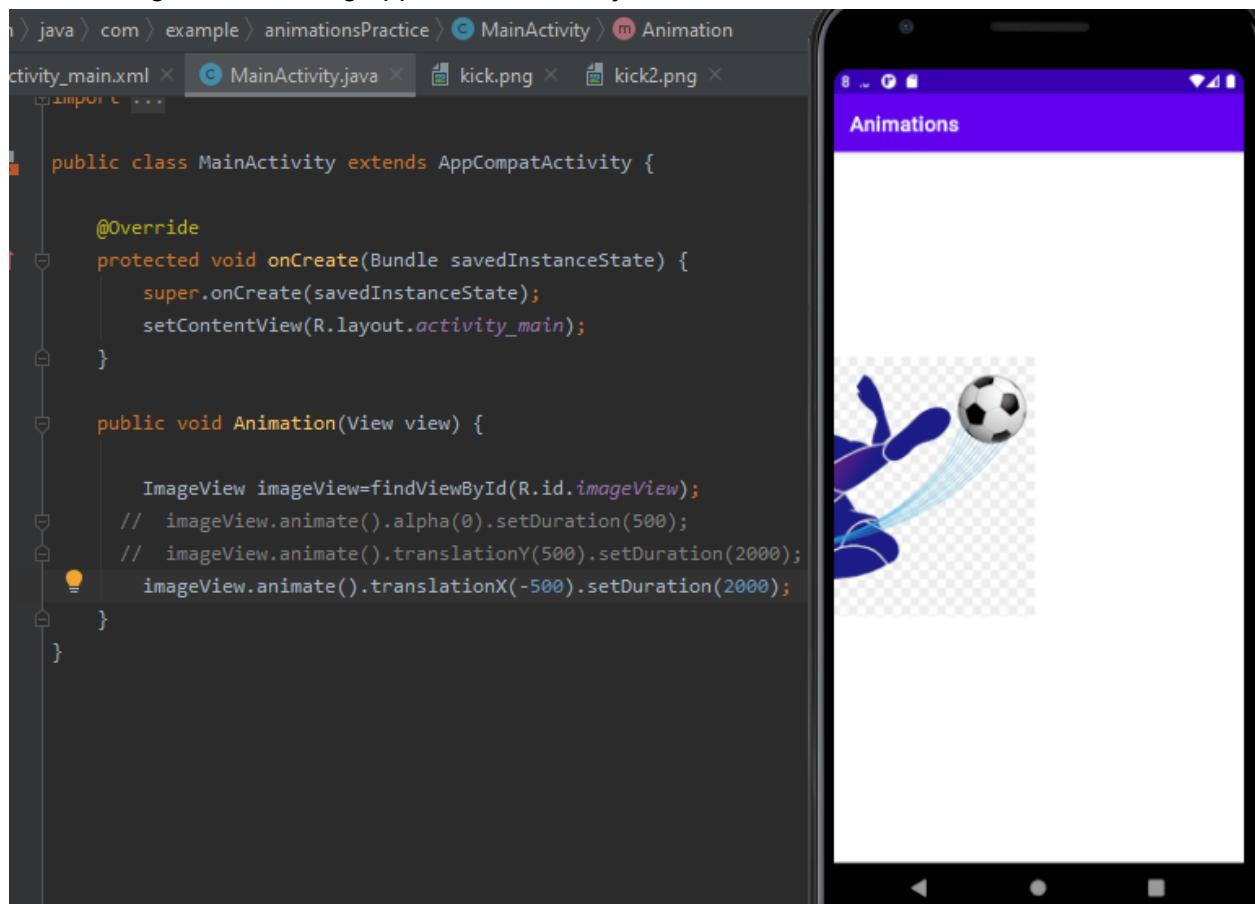
```
- Running Device Preview:** Displays a soccer player image with a purple-to-blue gradient overlay, showing the effect of the X-axis translation animation.

❖ Image -Ve Translation Animation:

Negative value => move to opposite side

- Write the function as below to get-Ve Translation Animation effect on image click.

- Image moved along opposite to X axis just after click.



The screenshot shows the Android Studio interface. On the left, the code editor displays `MainActivity.java` with the following code:

```
public class MainActivity extends AppCompatActivity {

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

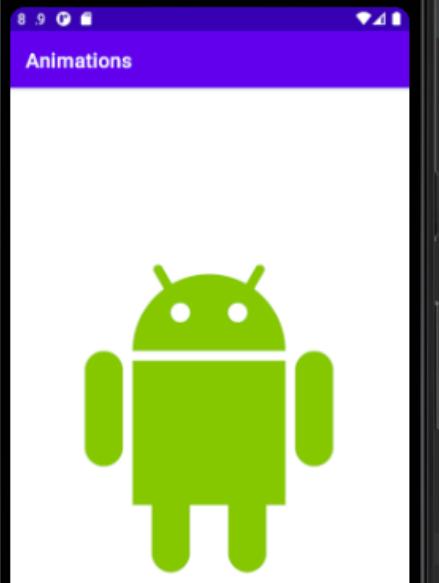
    public void Animation(View view) {

        ImageView imageView=findViewById(R.id.imageView);
        // imageView.animate().alpha(0).setDuration(500);
        // imageView.animate().translationY(500).setDuration(2000);
        imageView.animate().translationX(-500).setDuration(2000);
    }
}
```

On the right, the Android emulator window shows a purple title bar with the text "Animations". Below it is a white screen with a soccer ball and a blue kicking foot. A small checkered area indicates where the image was cut out from its original source.

❖ Image Rotation Animation:

- Write the function as below to get rotation effect on image click.



```
activity_main.xml x MainActivity.java x
```

```
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
  
    public void Animation(View view) {  
  
        ImageView imageView=findViewById(R.id.imageView);  
        // imageView.animate().alpha(0).setDuration(500);  
        //imageView.animate().translationY(500).setDuration(2000);  
        // imageView.animate().translationX(-500).setDuration(2000);  
        imageView.animate().rotation(180);  
    }  
}
```

- Image rotated to 180 degree just after click.



```
app > src > main > java > com > example > animationsPractice > MainActivity > Animations
```

```
activity_main.xml x MainActivity.java x
```

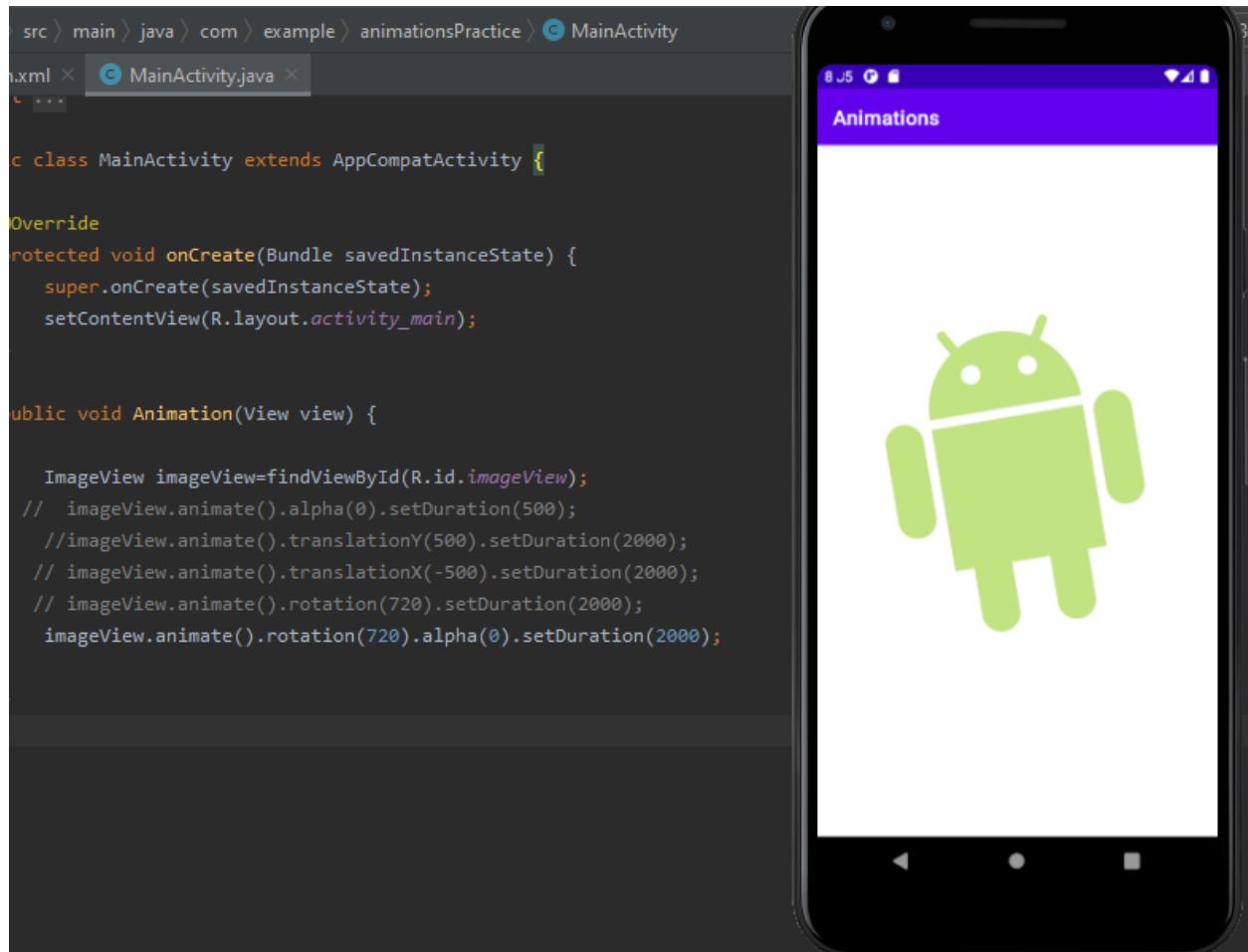
```
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
  
    public void Animation(View view) {  
  
        ImageView imageView=findViewById(R.id.imageView);  
        // imageView.animate().alpha(0).setDuration(500);  
        //imageView.animate().translationY(500).setDuration(2000);  
        // imageView.animate().translationX(-500).setDuration(2000);  
        imageView.animate().rotation(180);  
    }  
}
```

For Double Rotations:

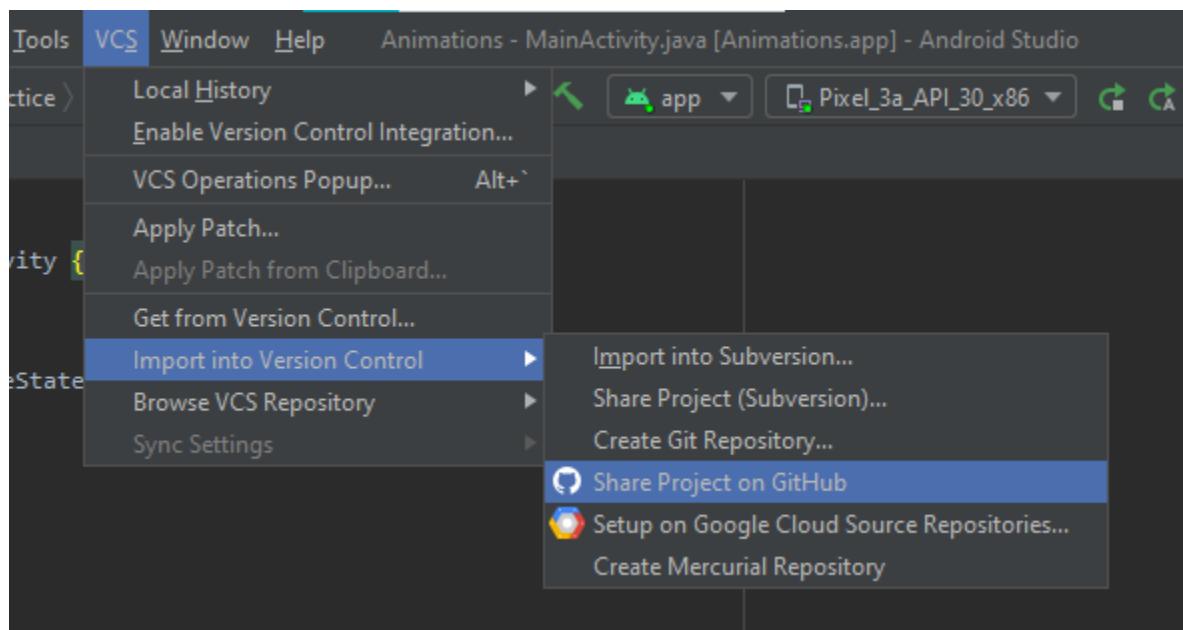
```
public void Animation(View view) {  
  
    ImageView imageView=findViewById(R.id.imageView);  
    // imageView.animate().alpha(0).setDuration(500);  
    //imageView.animate().translationY(500).setDuration(2000);  
    // imageView.animate().translationX(-500).setDuration(2000);  
  
    imageView.animate().rotation(720).setDuration(2000);
```

❖ Image Rotation,Disappear and Duration at same time:

- Write the function as below to get rotation, disappear effect with some duration on image click.
- Image rotated and disappeared in given duration.



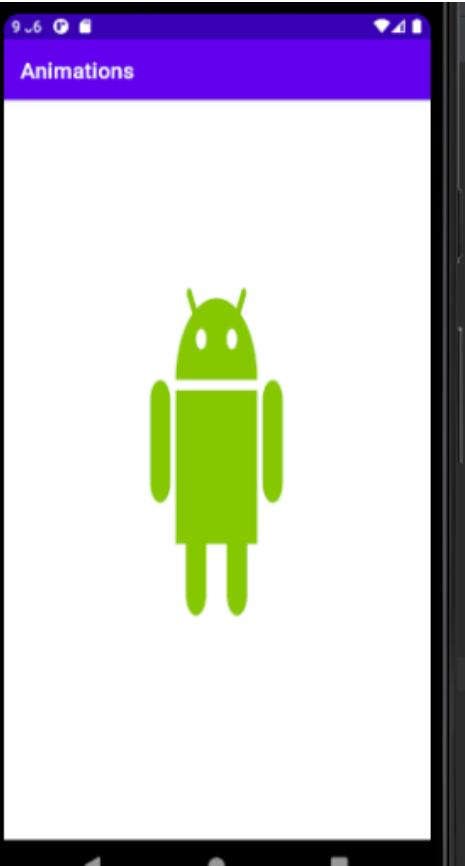
❖ Share to GitHub Using Android IDE:



❖ Image Scaling Animation:

- Write the function as below to get scaling effect on image click.

- Image scaled to x-axis just after click.



```
main.xml x MainActivity.java x
public class MainActivity extends AppCompatActivity {

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

    public void Animation(View view) {
        ImageView imageView=findViewById(R.id.imageView);
        // imageView.animate().alpha(0).setDuration(500);
        // imageView.animate().translationY(500).setDuration(2000);
        // imageView.animate().translationX(-500).setDuration(2000);
        // imageView.animate().rotation(720).setDuration(2000);
        // imageView.animate().rotation(720).alpha(0).setDuration(2000);
        imageView.animate().scaleX(0.5f);
    }
}
```

- Image scaled to y-axis just after click.

The screenshot shows the Android Studio interface. On the left, the code editor displays `MainActivity.java` with the following code:

```
activity_main.xml × MainActivity.java ×
```

```
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
  
    public void Animation(View view) {  
  
        ImageView imageView=findViewById(R.id.imageView);  
        // imageView.animate().alpha(0).setDuration(500);  
        // imageView.animate().translationY(500).setDuration(2000);  
        // imageView.animate().translationX(-500).setDuration(2000);  
        // imageView.animate().rotation(720).setDuration(2000);  
        // imageView.animate().rotation(720).alpha(0).setDuration(2000);  
        // imageView.animate().scaleX(0.5f);  
        imageView.animate().scaleY(0.5f);  
    }  
}
```

On the right, the emulator window shows a green Android icon on a purple background with the title "Animations".

❖ Image Scaling ALong Both XY axis:

- Write the function as below to get scaling effect on image click.



```
y_main.xml x MainActivity.java x
+-- MainActi... +-- MainActivity.java +-- R.java
+-- anim... +-- activity_main.xml +-- anim... +-- R.java

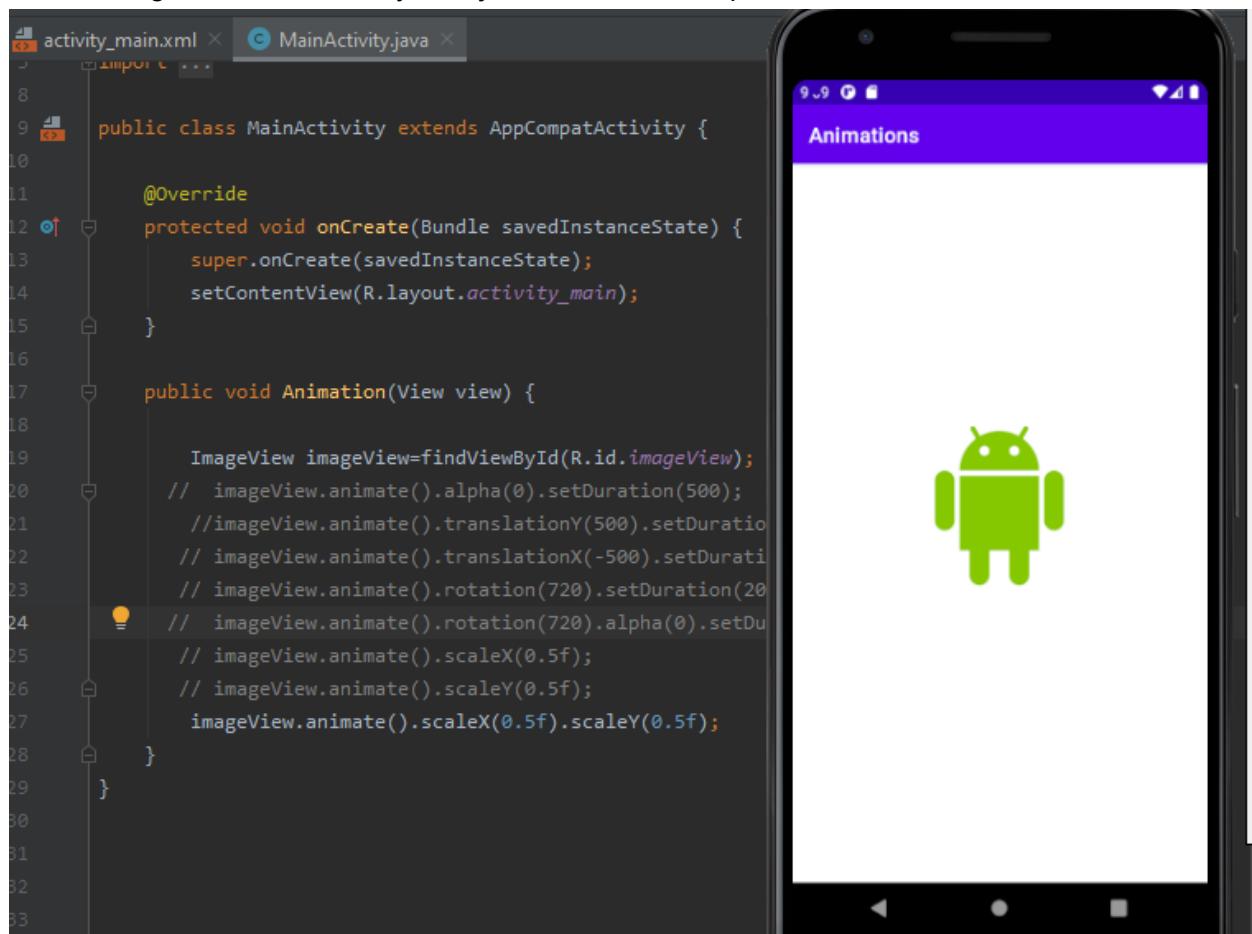
public class MainActivity extends AppCompatActivity {

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

    public void Animation(View view) {

        ImageView imageView=findViewById(R.id.imageView);
        // imageView.animate().alpha(0).setDuration(500);
        // imageView.animate().translationY(500).setDuratio
        // imageView.animate().translationX(-500).setDurati
        // imageView.animate().rotation(720).setDuration(20
        // imageView.animate().rotation(720).alpha(0).setDu
        // imageView.animate().scaleX(0.5f);
        // imageView.animate().scaleY(0.5f);
        imageView.animate().scaleX(0.5f).scaleY(0.5f);
    }
}
```

- Image scaled to both xy-axis just after click so squeezed.



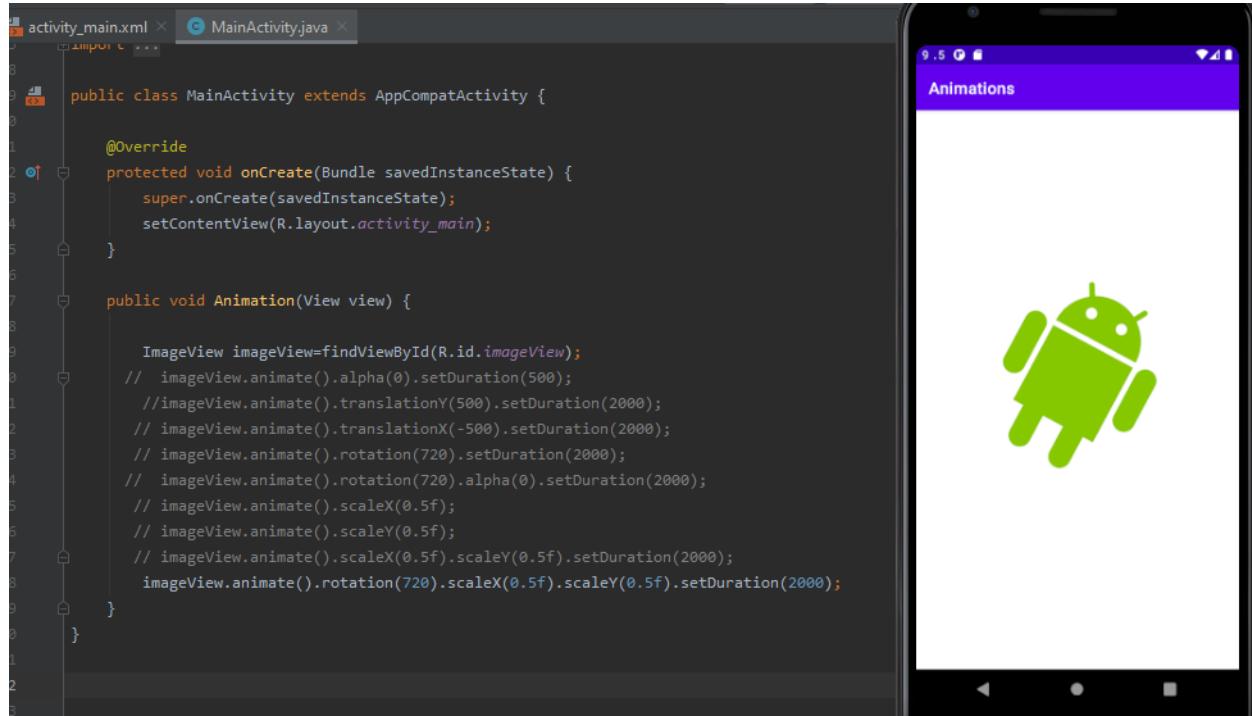
The image shows a screenshot of the Android Studio IDE. On the left, the code editor displays `MainActivity.java` with the following code:

```
activity_main.xml × MainActivity.java ×
8
9  public class MainActivity extends AppCompatActivity {
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_main);
15     }
16
17     public void Animation(View view) {
18
19         ImageView imageView=findViewById(R.id.imageView);
20         // imageView.animate().alpha(0).setDuration(500);
21         //imageView.animate().translationY(500).setDuratio
22         // imageView.animate().translationX(-500).setDurati
23         // imageView.animate().rotation(720).setDuration(20
24         //   // imageView.animate().rotation(720).alpha(0).setDu
25         //   // imageView.animate().scaleX(0.5f);
26         //   // imageView.animate().scaleY(0.5f);
27         imageView.animate().scaleX(0.5f).scaleY(0.5f);
28     }
29 }
30
31
32
33
```

The right side of the image shows a screenshot of an Android emulator. The title bar says "Animations". Inside the screen, there is a large green Android robot icon. The screen is mostly white, indicating that the animation has just started or is not yet fully applied.

❖ Image Scaling + Rotation at same time:

- Write the function as below to get scaling and rotation effect on image click.
- Image rotated and scaled to both xy-axis just after click.



The image shows the Android Studio interface. On the left, the code editor displays `MainActivity.java` with the following code:

```
public class MainActivity extends AppCompatActivity {

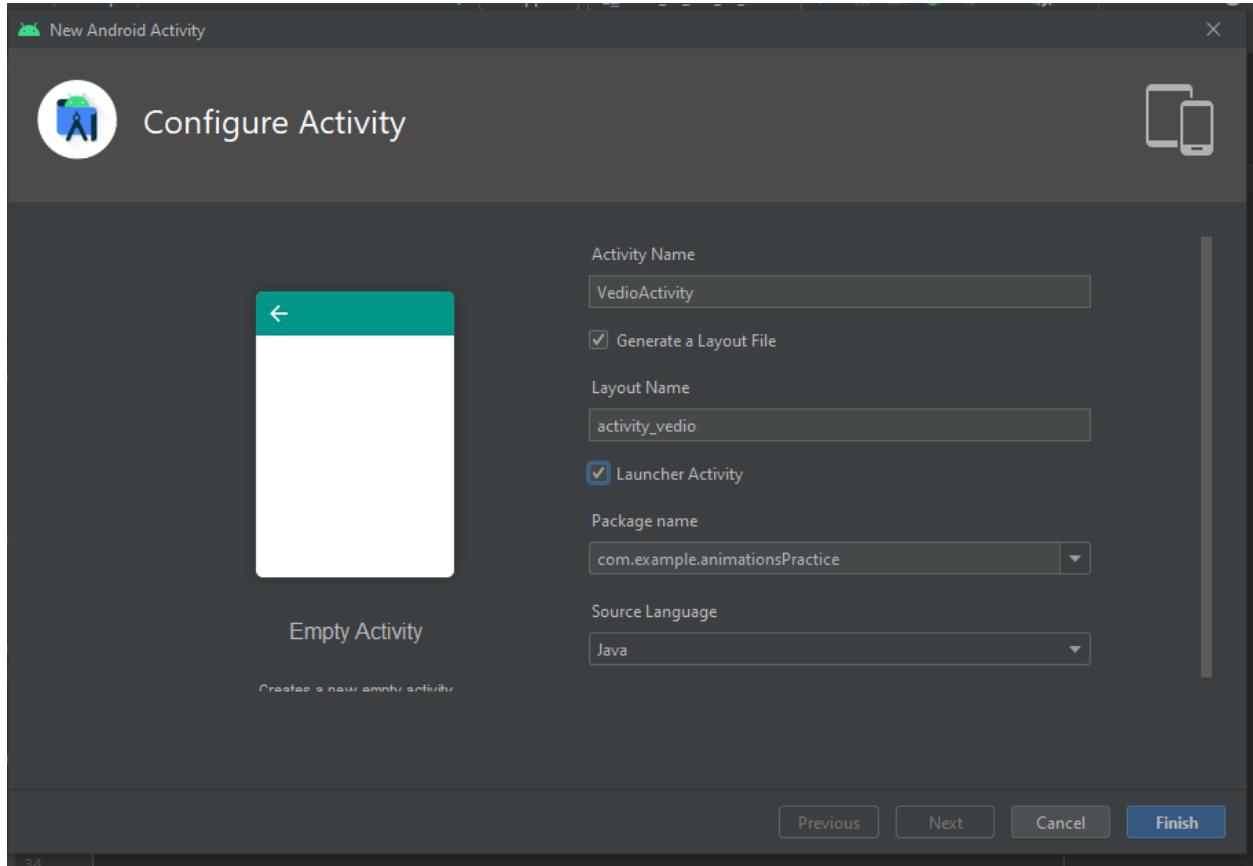
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void Animation(View view) {
        ImageView imageView=findViewById(R.id.imageView);
        // imageView.animate().alpha(0).setDuration(500);
        // imageView.animate().translationY(500).setDuration(2000);
        // imageView.animate().translationX(-500).setDuration(2000);
        // imageView.animate().rotation(720).setDuration(2000);
        // imageView.animate().rotation(720).alpha(0).setDuration(2000);
        // imageView.animate().scaleX(0.5f);
        // imageView.animate().scaleY(0.5f);
        // imageView.animate().scaleX(0.5f).scaleY(0.5f).setDuration(2000);
        imageView.animate().rotation(720).scaleX(0.5f).scaleY(0.5f).setDuration(2000);
    }
}
```

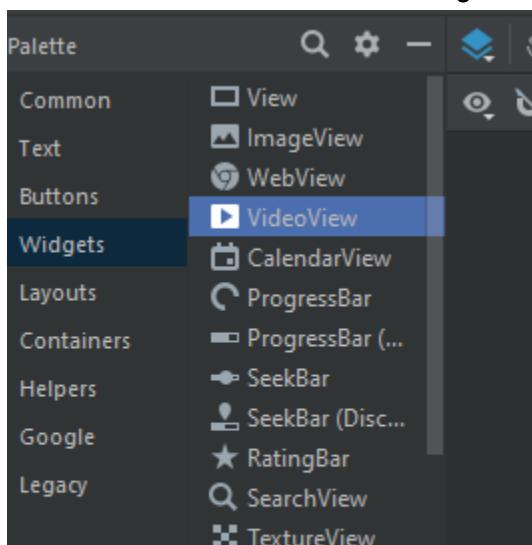
On the right, the preview window shows a green Android robot icon on a white background, with the title bar reading "Animations".

❖ Videos:

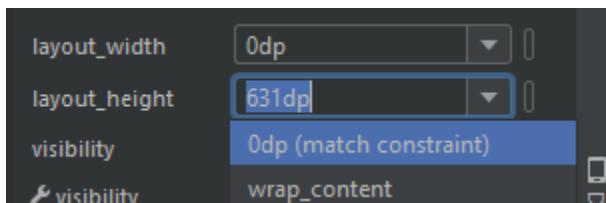
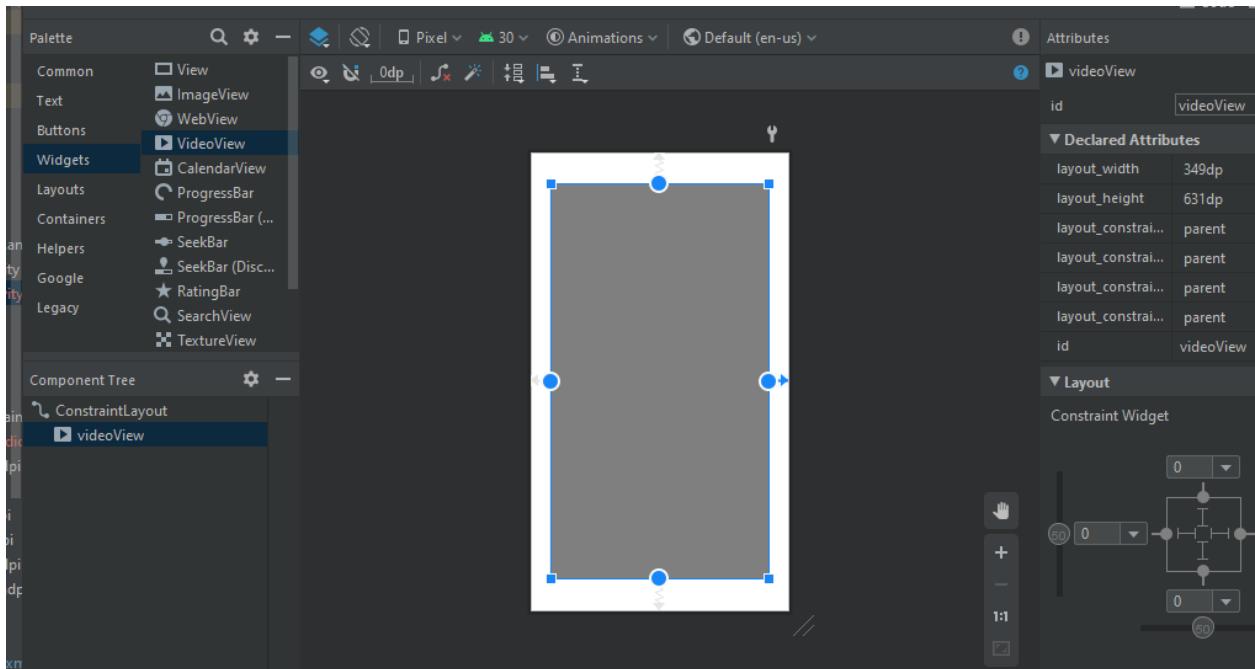
- Create a new activity and set it as launching activity.



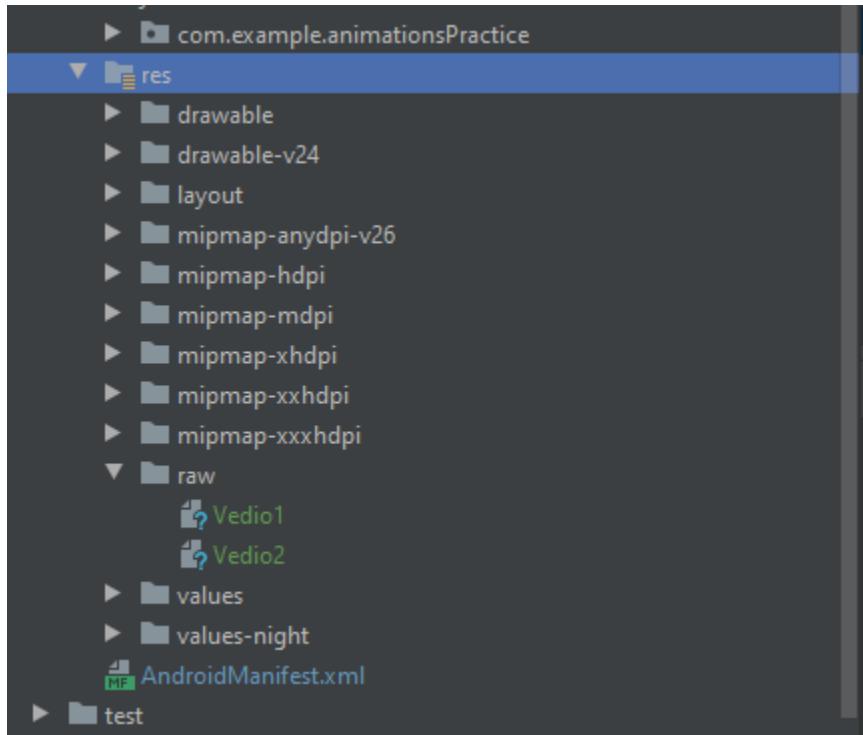
- Add an video view from widgets.



- Set the constraints of view and fix its position.



- In **res** folder make a folder named **raw**. Then copy videos you want to add from PC and paste in this folder.



- Add the code below to make a video view on screen. Run it and see the video output on emulator.

The screenshot shows the Android Studio interface with the following details:

- MainActivity.java:**

```

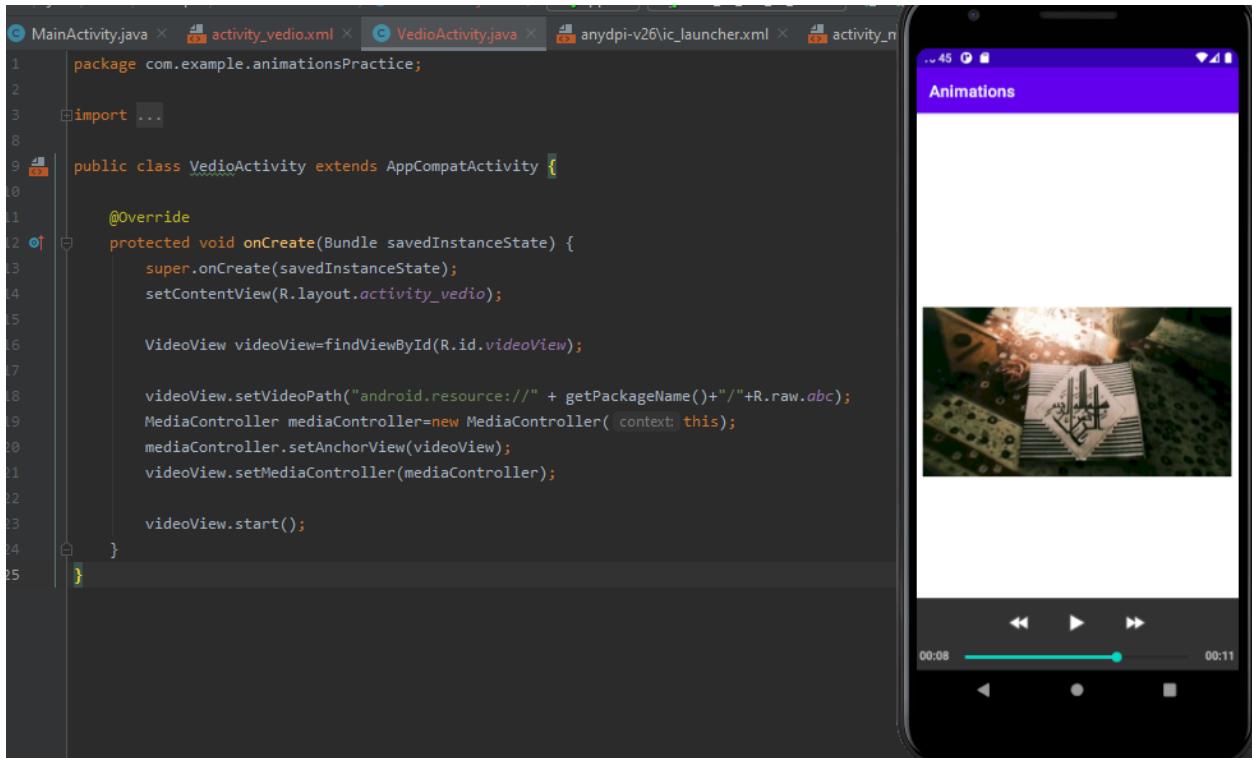
1 package com.example.animationsPractice;
2
3 import ...
4
5 public class VedioActivity extends AppCompatActivity {
6
7     @Override
8     protected void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10        setContentView(R.layout.activity_vedio);
11
12        VideoView videoView=findViewById(R.id.videoView);
13        videoView.setVideoPath("android.resource://" + getPackageName() + "/"+R.raw.abc);
14        videoView.start();
15    }
16}
```
- Emulator Preview:** A smartphone emulator displays the application's user interface. The title bar says "Animations". The main screen shows a video player with a video frame containing Arabic calligraphy.

❖ Set Video Controls:

- To set video controls add the code below.

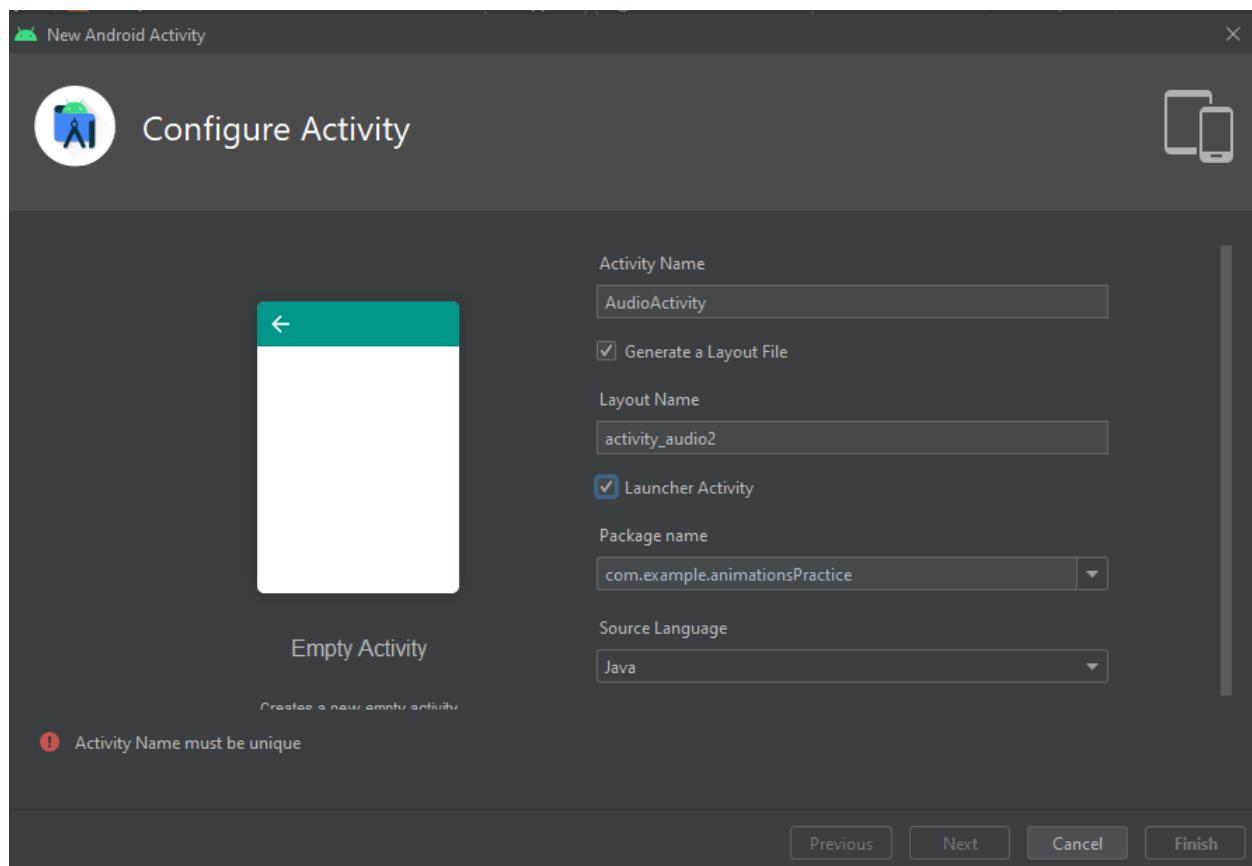
```
9  public class VedioActivity extends AppCompatActivity {
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_vedio);
15
16         VideoView videoView=findViewById(R.id.videoView);
17
18         videoView.setVideoPath("android.resource://" + getPackageName() + "/" + R.raw.abc);
19         MediaController mediaController=new MediaController(this);
20         mediaController.setAnchorView(videoView);
21         videoView.setMediaController(mediaController);
22
23         videoView.start();
24     }
}
```

- Run it and see the output on emulator, the video with controls.

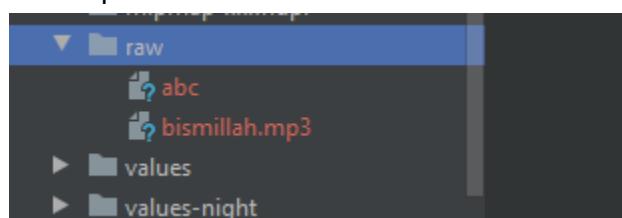


❖ Audio Files:

- Create a new activity and set it as launching activity.



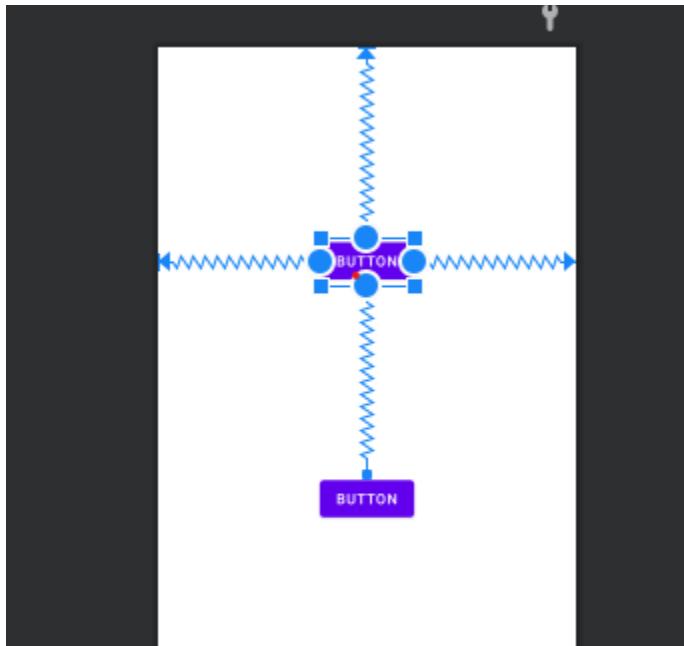
- In **res** folder make a folder named **raw**. Then copy audios you want to add from PC and paste in this folder.



❖ Set Audio Controls:

Android don't have built in functionality so we need to add buttons to set audio controls.

- Add 2 Buttons on xml file and set their constraints with position.



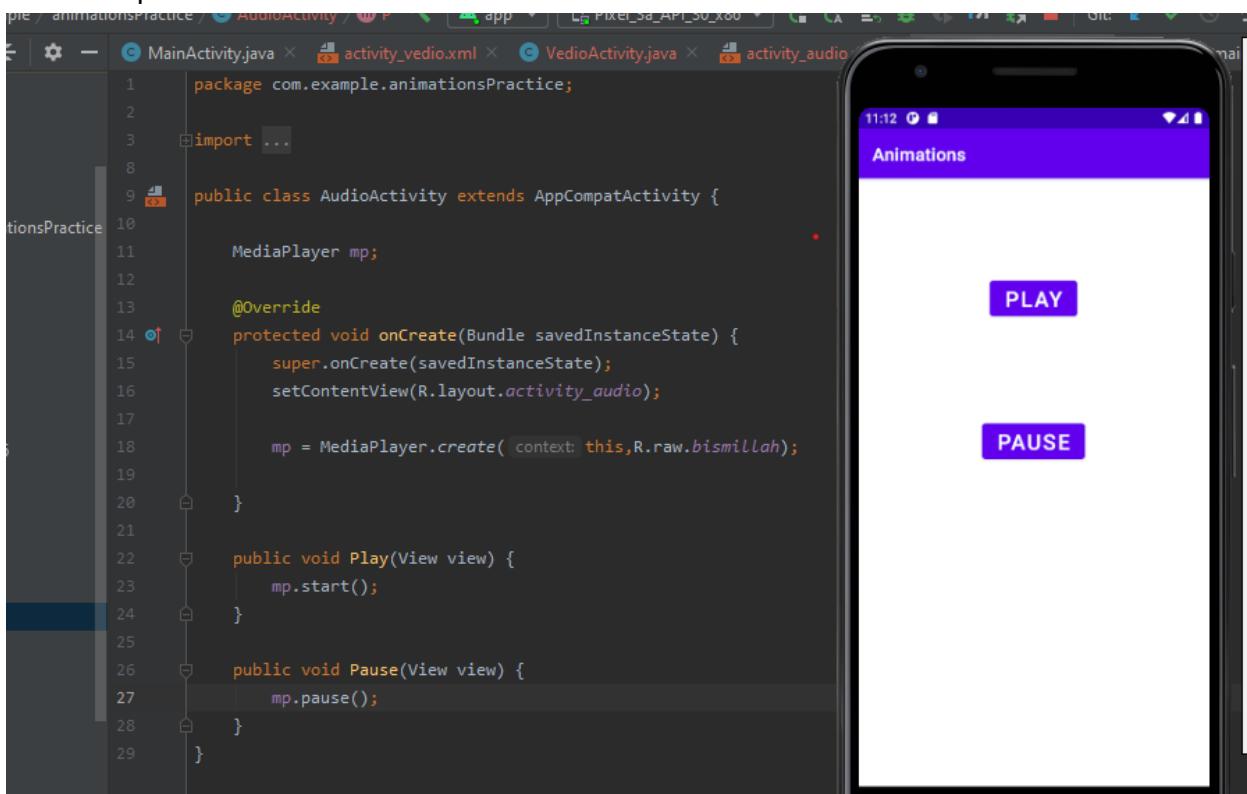
- Now write click event for one button as Play and other button as Pause.

```
<Button  
    android:id="@+id	btnPlay"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Button"  
    android:onClick="Play"  
    app:layout_constraintBottom_toTopOf="@+id/BtnPause"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent" />  
  
<Button  
    android:id="@+id/BtnPause"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginBottom="360dp"  
    android:text="Button"  
    android:onClick="Pause"
```

```
        public void Play(View view) {
            mp.start();
        }

        public void Pause(View view) {
            mp.pause();|}
        }
    }
```

- Write the code to add audio on screen and set controls in functions on clicked by buttons.
- Run the file and see output on emulator. Play button will play the audio and pause will stop the audio.



❖ Uploading the Project on GitHub Repository:

```
Git CMD
C:\Users\HP>cd C:\Users\HP\Desktop\Git\20210326_A019
C:\Users\HP\Desktop\Git\20210326_A019>git add .

C:\Users\HP\Desktop\Git\20210326_A019>git commit -m "Audio-Vedio Practice Added"
[master a557bda] Audio-Vedio Practice Added
 8 files changed, 127 insertions(+)
 create mode 100644 Animations/.idea/vcs.xml
 create mode 100644 Animations/app/src/main/java/com/example/animationsPractice/MainActivity.java
 create mode 100644 Animations/app/src/main/java/com/example/animationsPractice/VedioActivity.java
 create mode 100644 Animations/app/src/main/res/layout/activity_audio.xml
 create mode 100644 Animations/app/src/main/res/layout/activity_vedio.xml
 create mode 100644 Animations/app/src/main/res/raw/abc
 create mode 100644 Animations/app/src/main/res/raw/bismillah.mp3

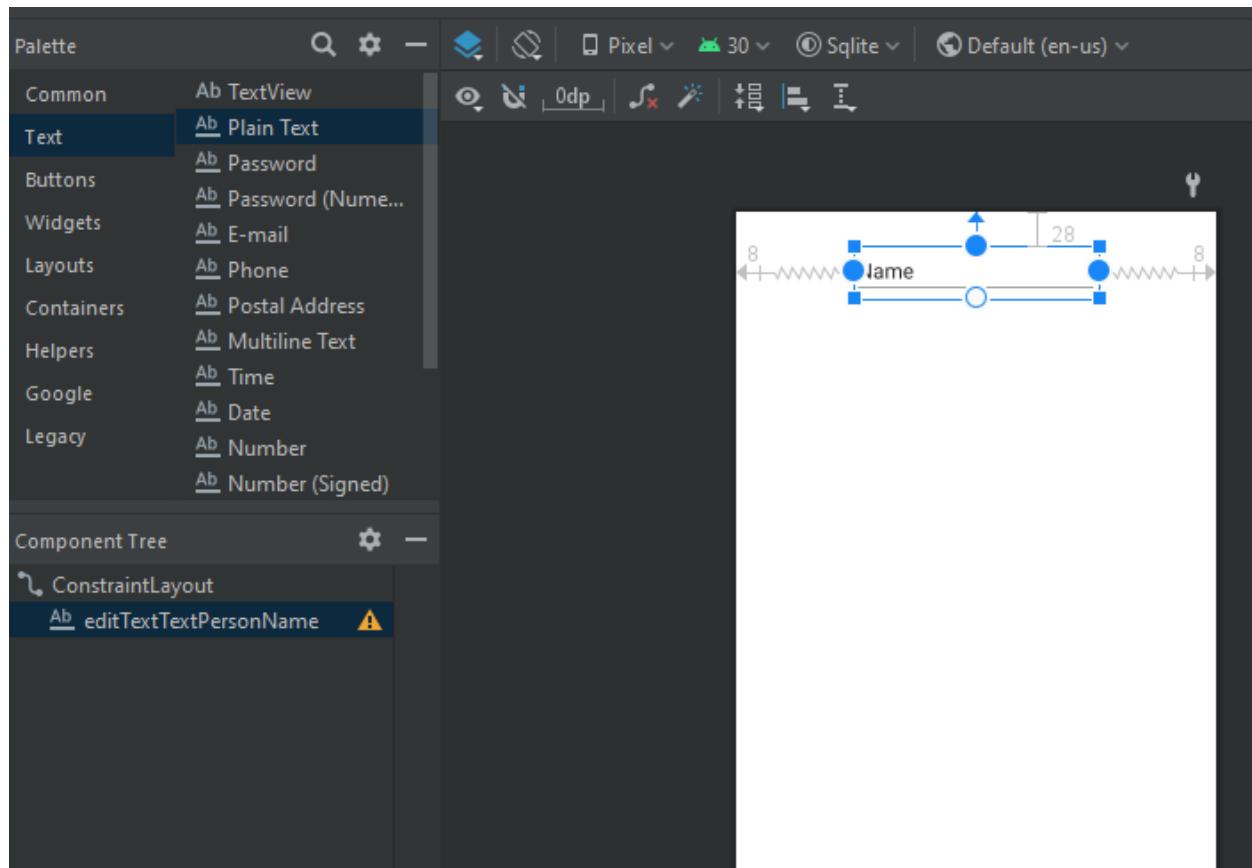
C:\Users\HP\Desktop\Git\20210326_A019>git push
Enumerating objects: 35, done.
Counting objects: 100% (35/35), done.
Delta compression using up to 8 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (22/22), 2.55 MiB | 93.00 KiB/s, done.
Total 22 (delta 7), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (7/7), completed with 5 local objects.
To https://github.com/amber-shafique/20210326_A019.git
 ea/f199..a557bda master -> master
```

LECTURE#10

❖ SQLite-Lec1:

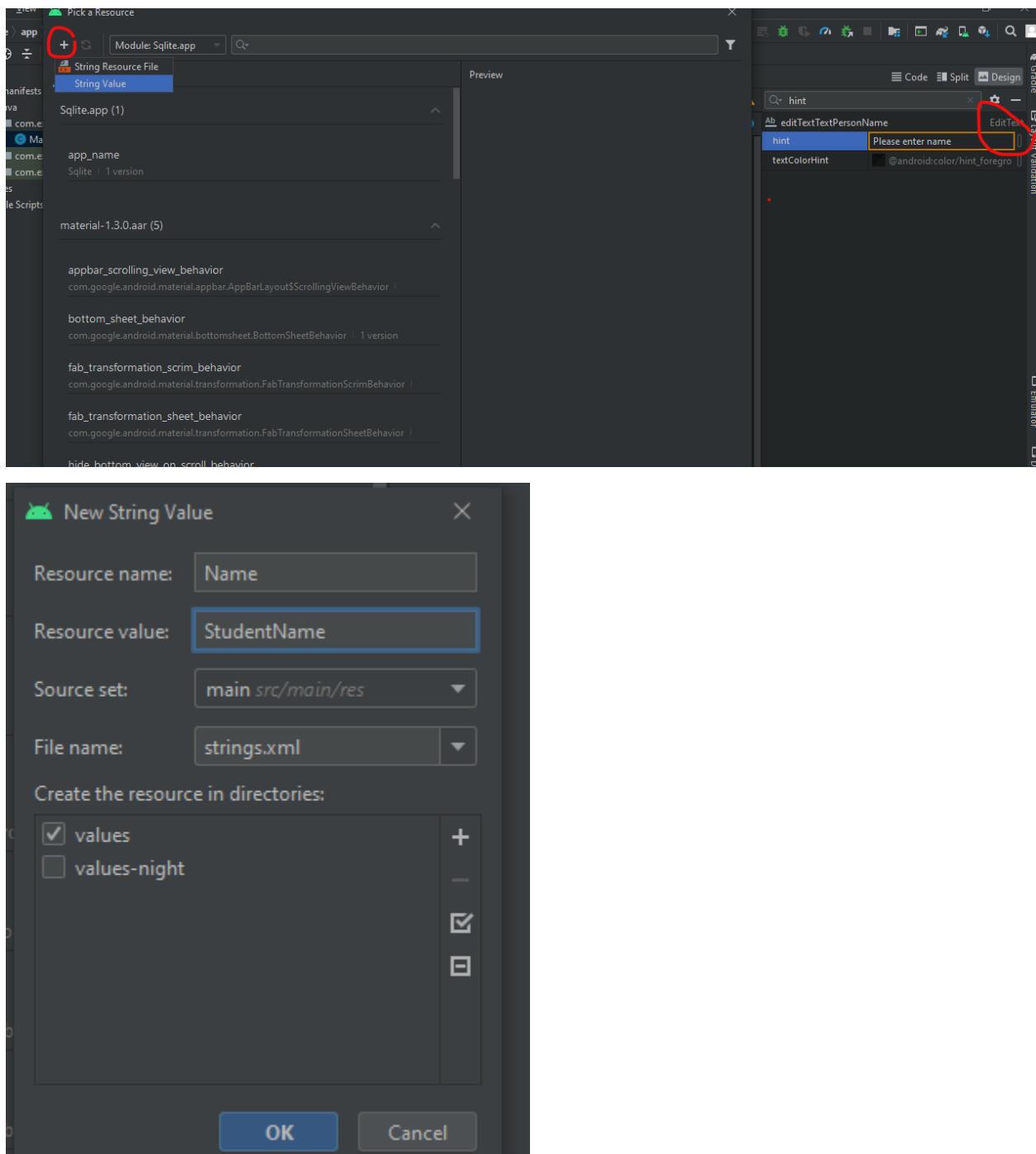
To make a view for entering and retrieving data from database:

- Add a plain text field and set its constraints.

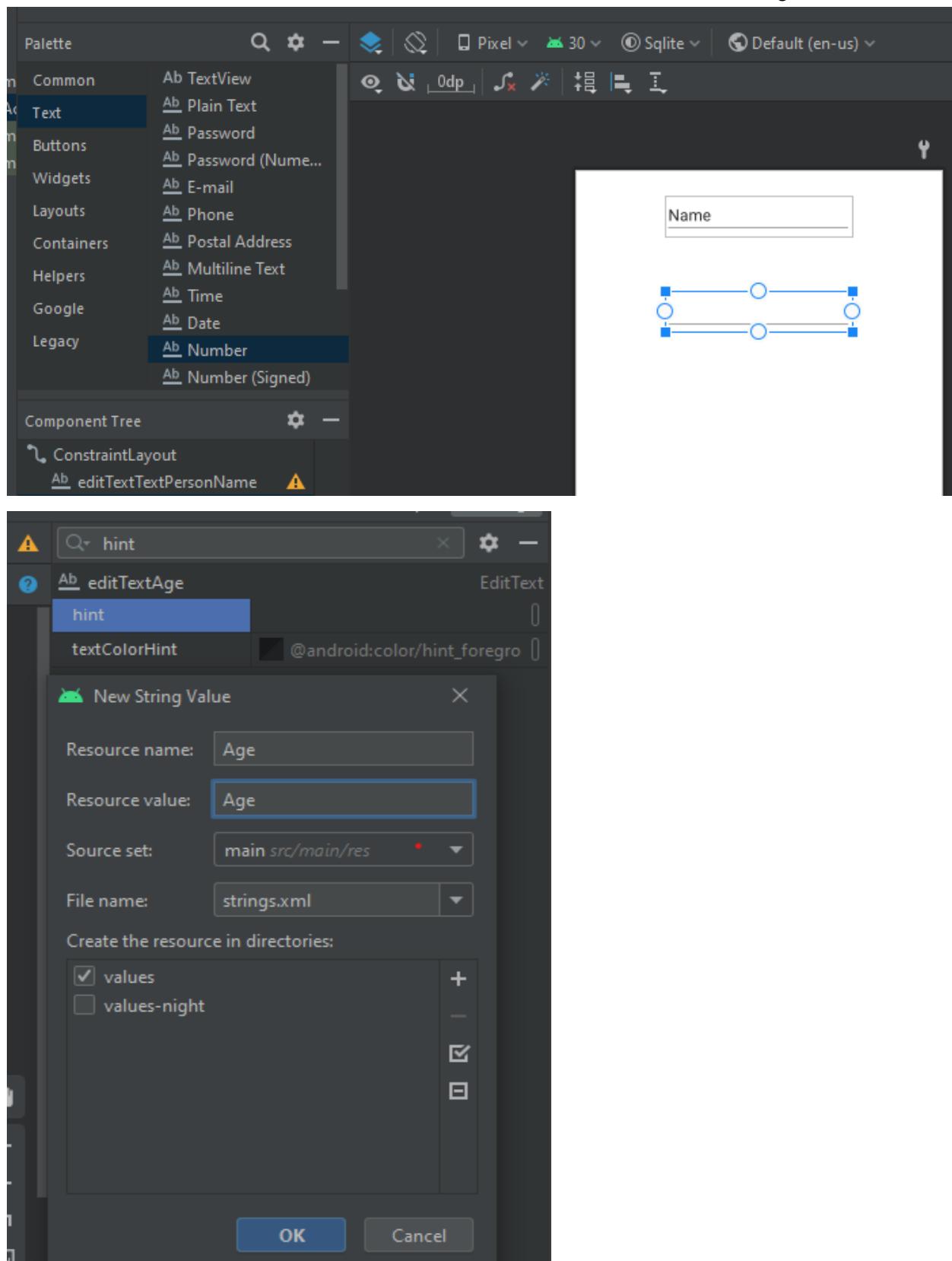


- **Add new string resource:**

To save the id of the text field create a new string resource.

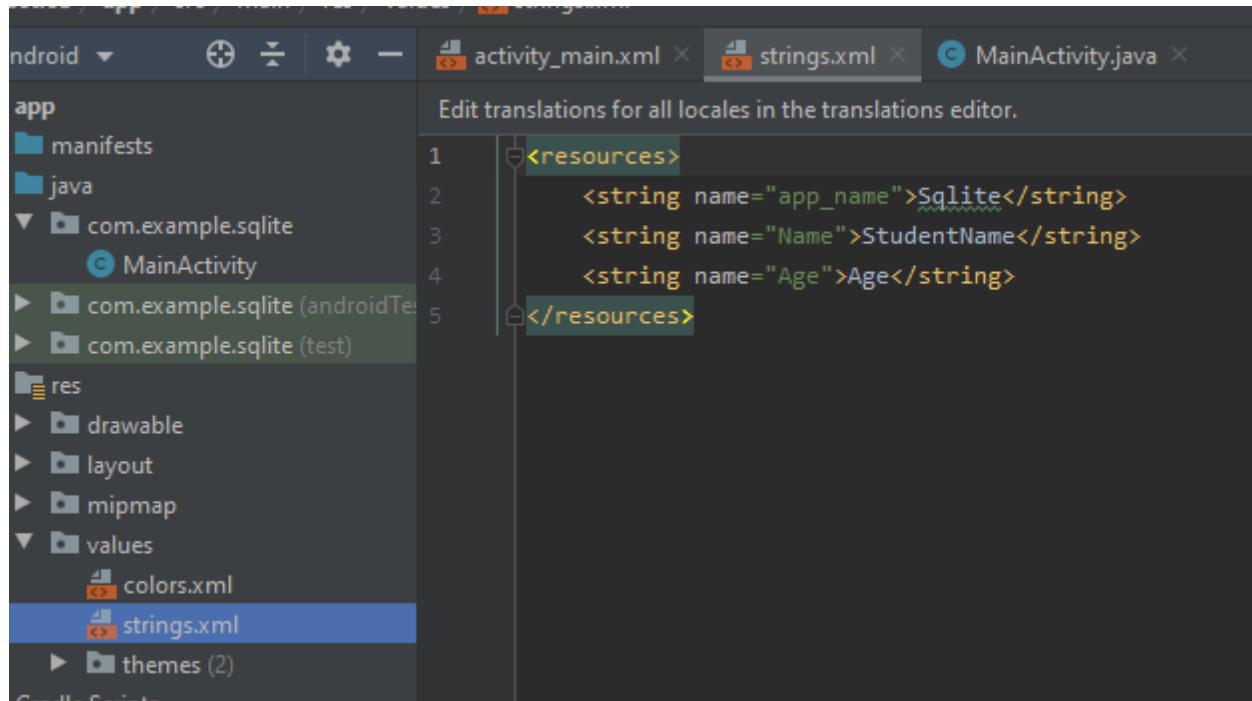


- Add another field of number & set its constraints too. Also add new string resource for it.



● Viewing String Resources:

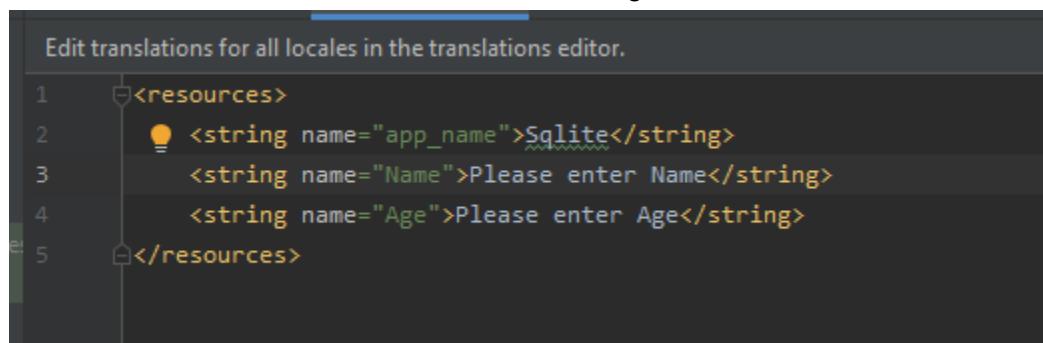
In resources=> values, string.xml file will be saving string resources.



The screenshot shows the Android Studio interface with the project navigation bar at the top. Below it is a tree view of the project structure under the 'app' folder. The 'values' folder is expanded, and the 'strings.xml' file is selected and highlighted with a blue background. The main editor area displays the XML code for the strings resource. The code includes a root <resources> tag, three <string> tags with names 'app_name', 'Name', and 'Age', and a closing </resources> tag.

```
<resources>
    <string name="app_name">Sqlite</string>
    <string name="Name">StudentName</string>
    <string name="Age">Age</string>
</resources>
```

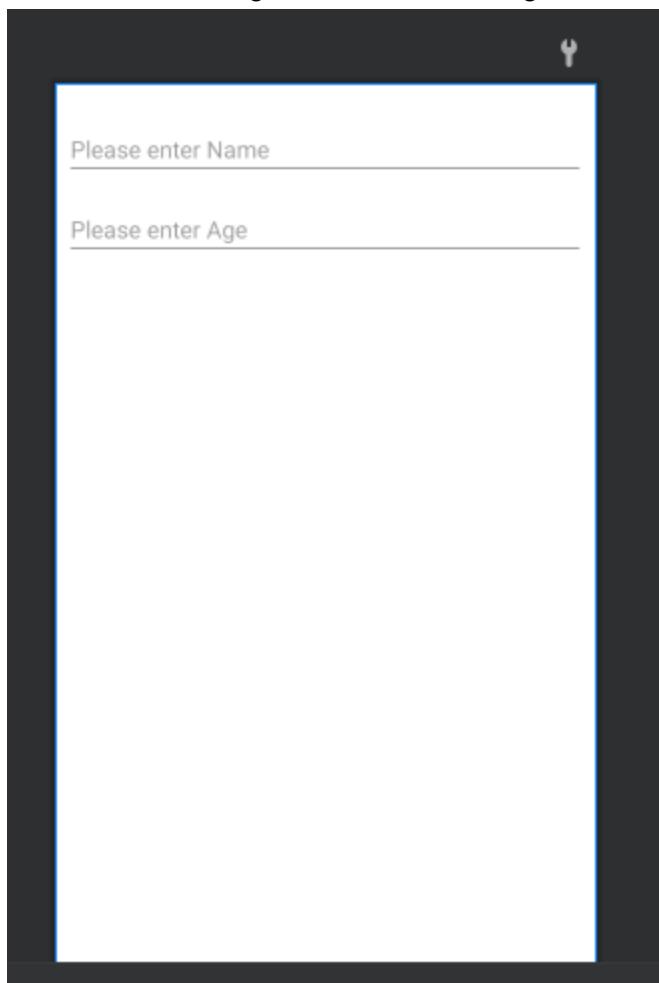
Set the default text for these strings and save.



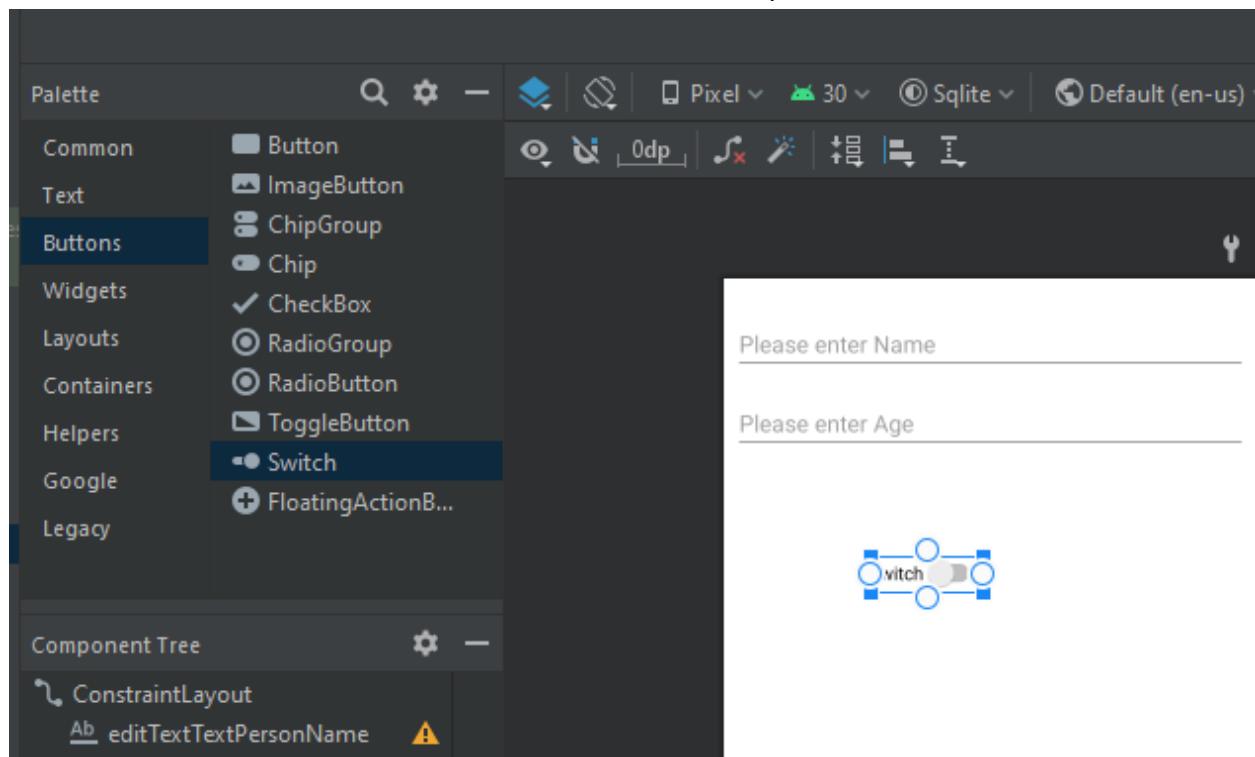
The screenshot shows the same Android Studio interface, but the 'strings.xml' file has been modified. The first string entry, 'app_name', now contains the placeholder text 'Sqlite'. The other two entries, 'Name' and 'Age', have been replaced by placeholder text 'Please enter Name' and 'Please enter Age' respectively. The rest of the XML structure remains the same.

```
<resources>
    <string name="app_name">Sqlite</string>
    <string name="Name">Please enter Name</string>
    <string name="Age">Please enter Age</string>
</resources>
```

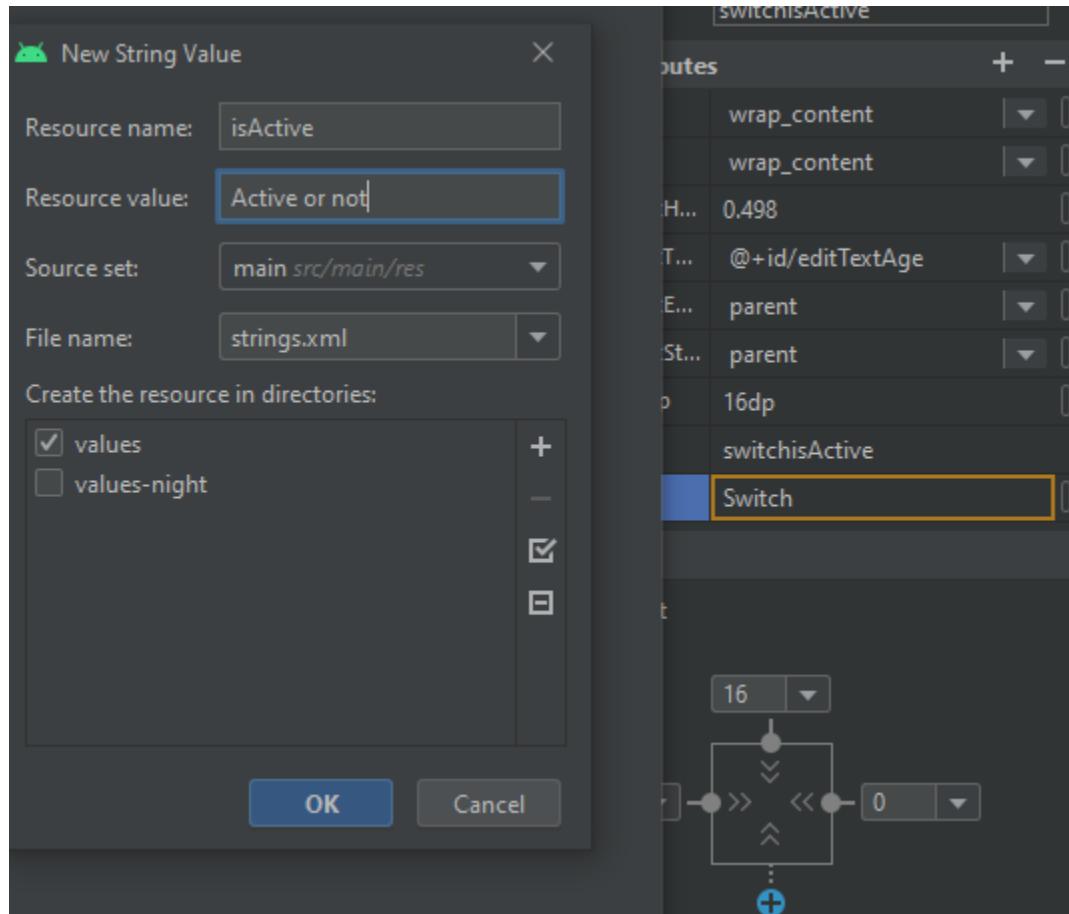
- After adding default text for strings run the emulator and see the output on screen.



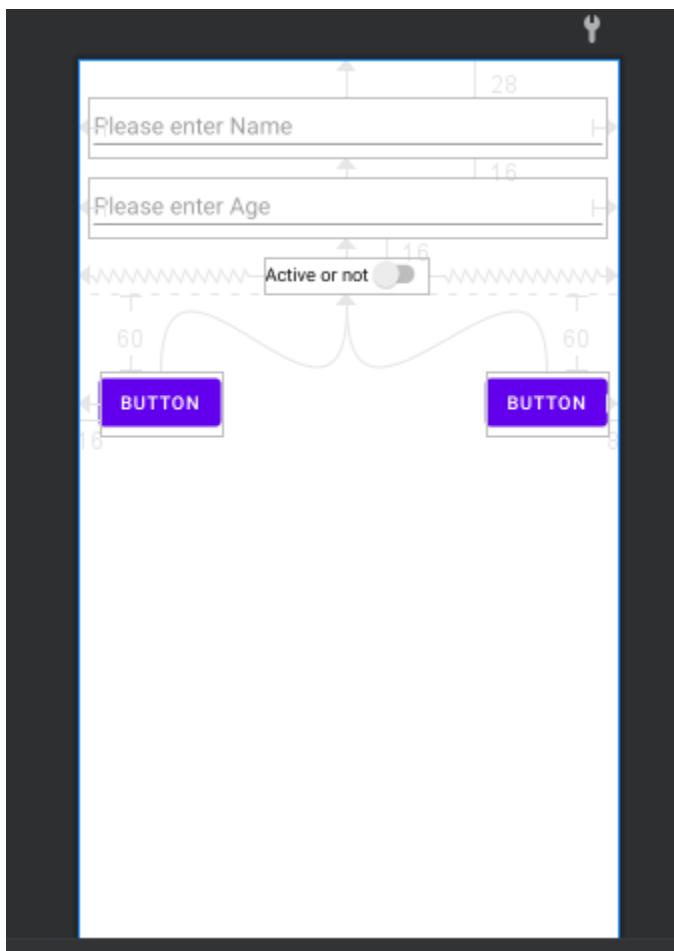
- Now add another field of switch on screen from the pallet.



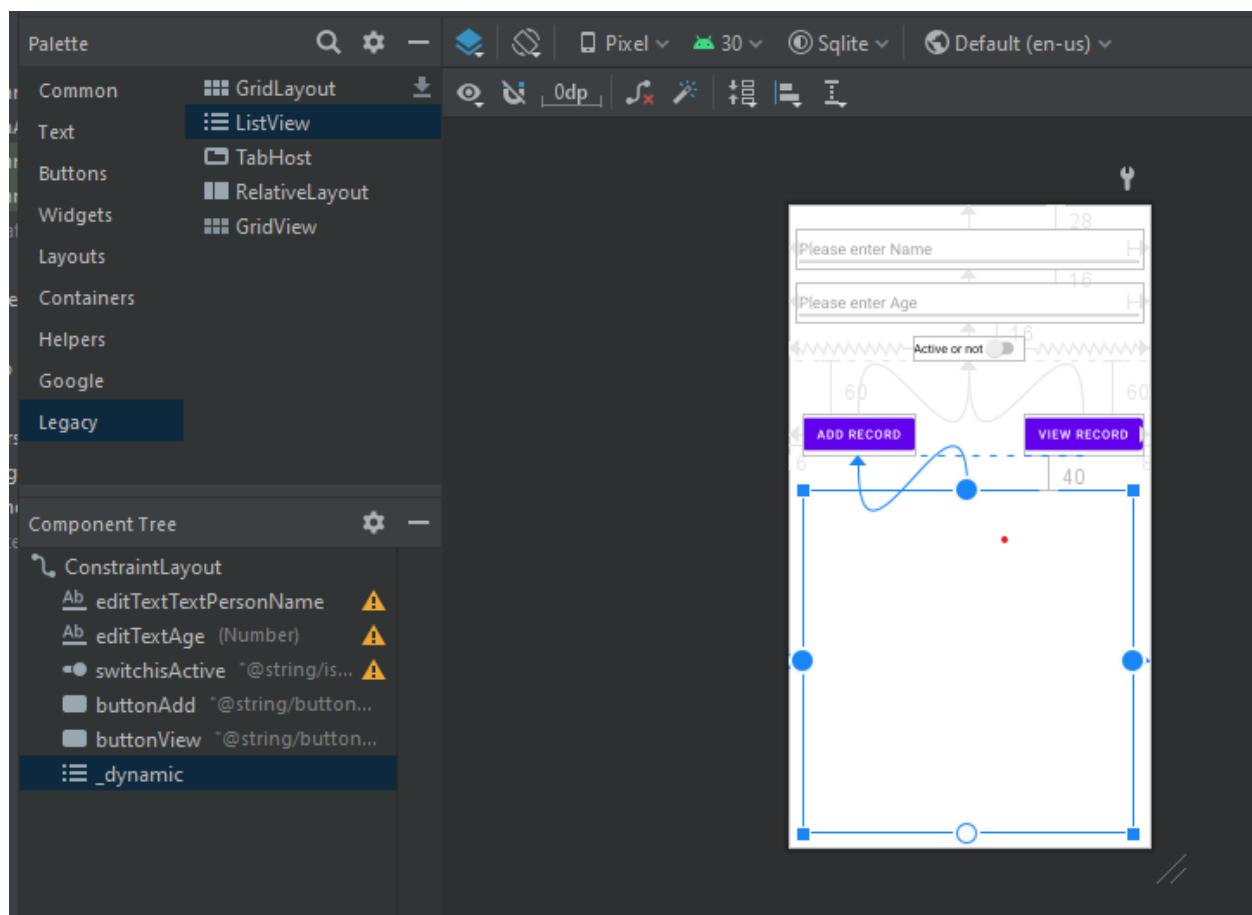
- Add a string resource for switch also.



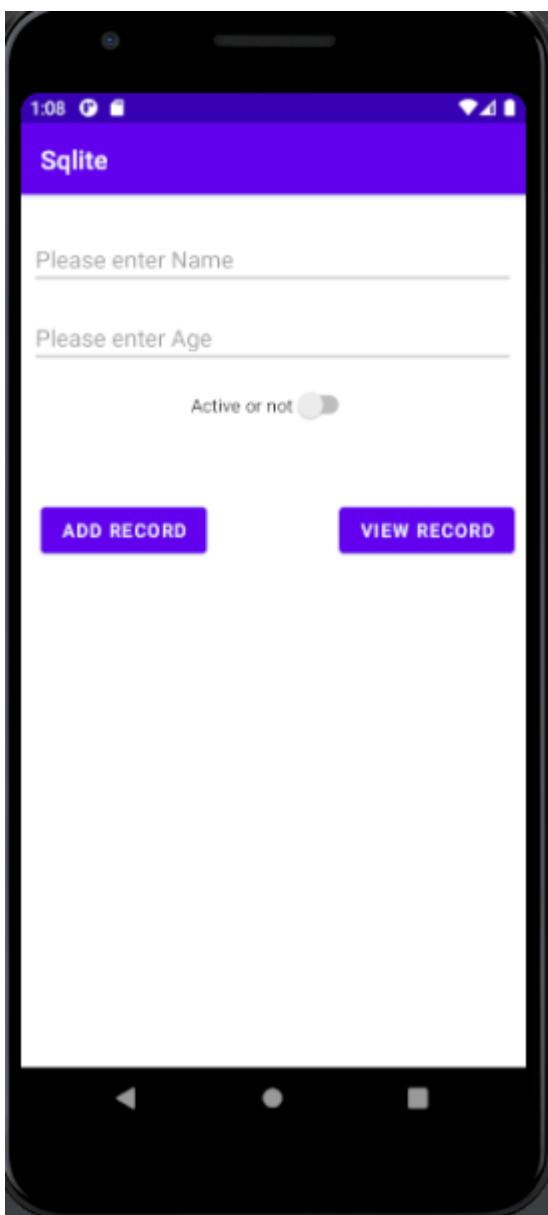
- Now add two buttons on the screen and set their constraints.



- Add a list view and set its constraints with buttons.



- Now run the emulator to see the whole screen view.



- **Using toast instead of log:**

To see the click events happening on screen add toast function instead of log.

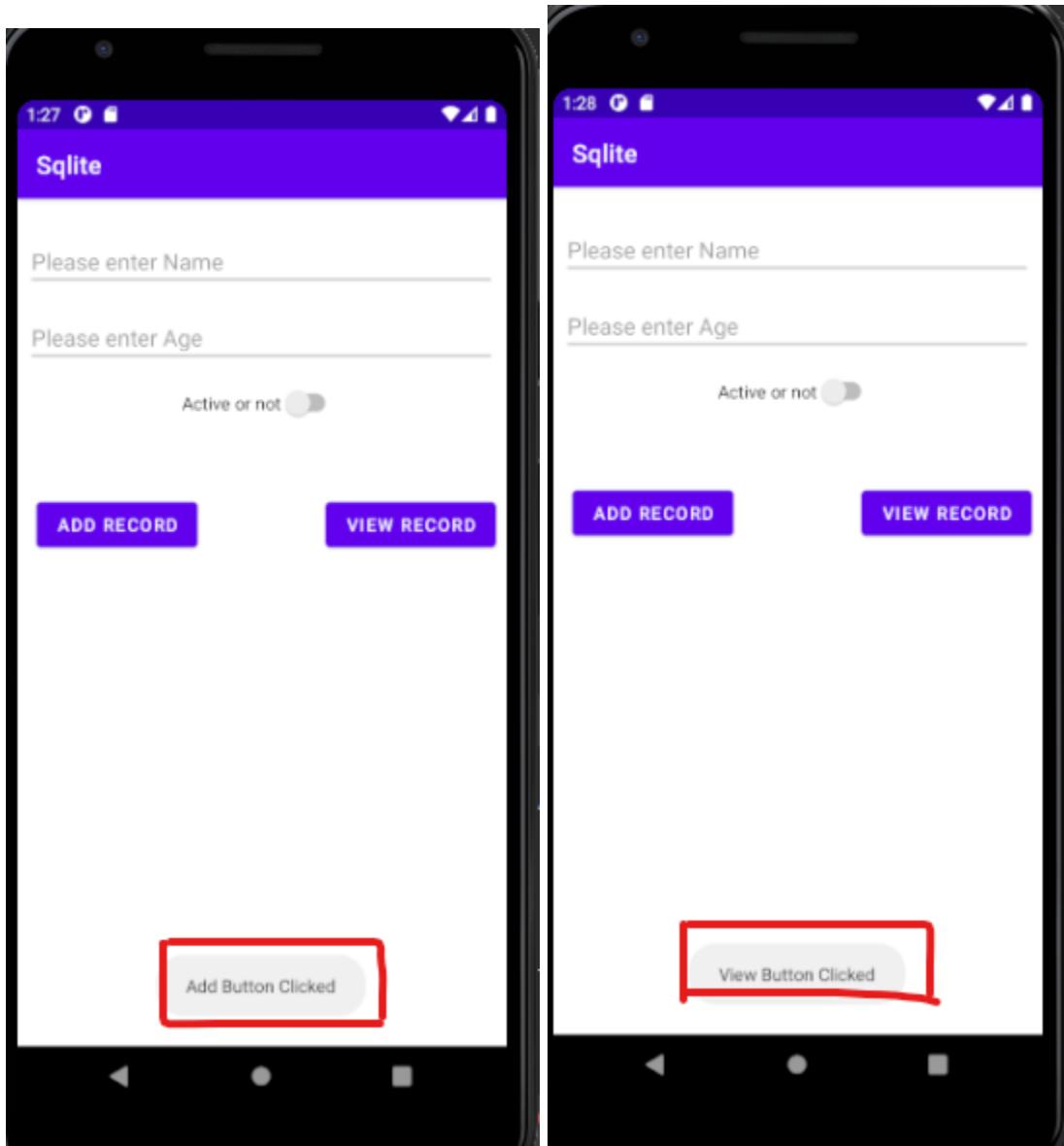
```
btnAdd= findViewById(R.id.buttonAdd);
btnView=findViewById(R.id.buttonView);

        .
        .
        .

        btnAdd.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View view) {
        Toast.makeText( context: MainActivity.this, text: "Add Button Clicked",Toast.LENGTH_SHORT).show();
    }
});
btnView.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View view) {

        Toast.makeText( context: MainActivity.this, text: "View Button Clicked",Toast.LENGTH_SHORT).show();
    }
});
```

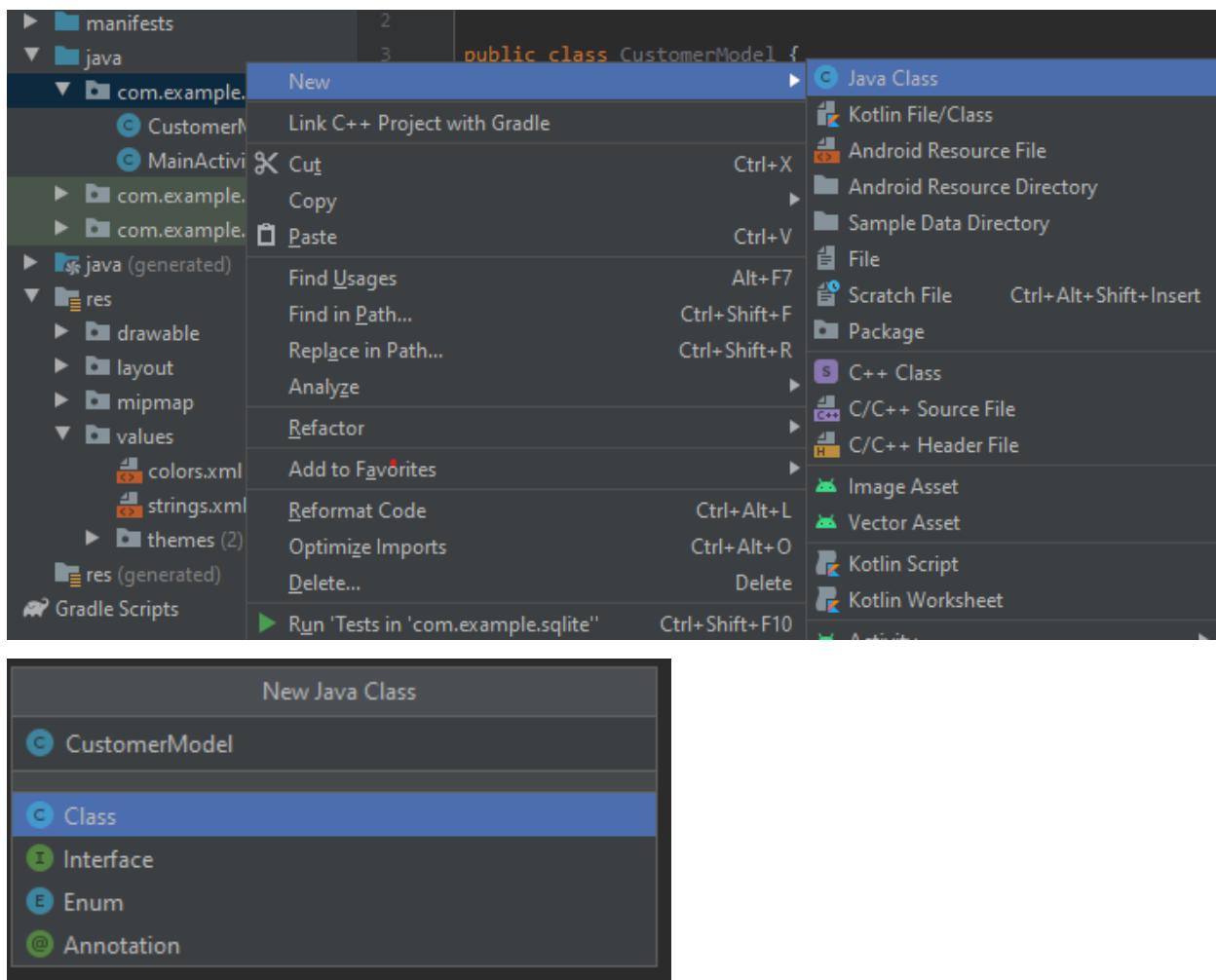
- After adding toast on click events of button run the emulator to see output.
- Just when you click the button the toast function will display a text notification in the bottom to let us know that which button is being clicked.



❖ SQLite-Lec2:

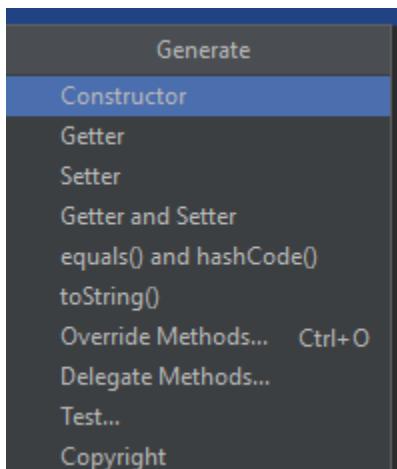
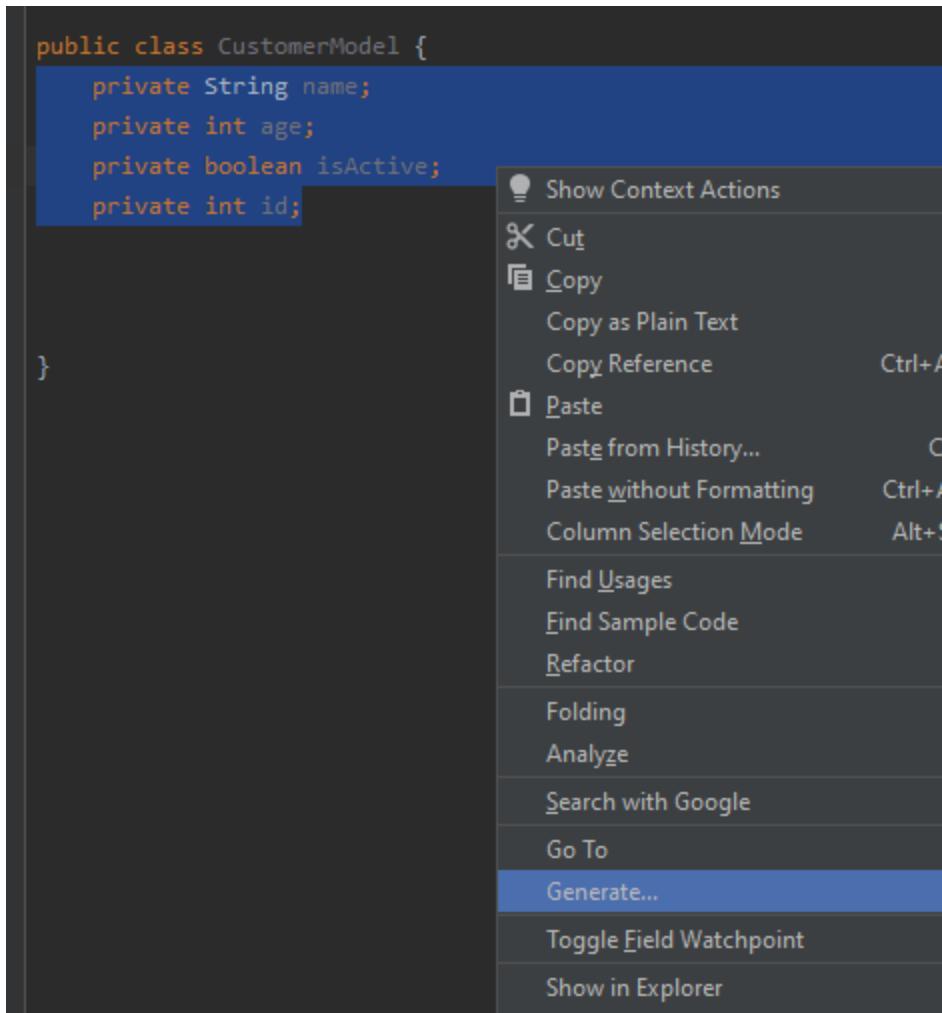
● Retrieving Values from the model:

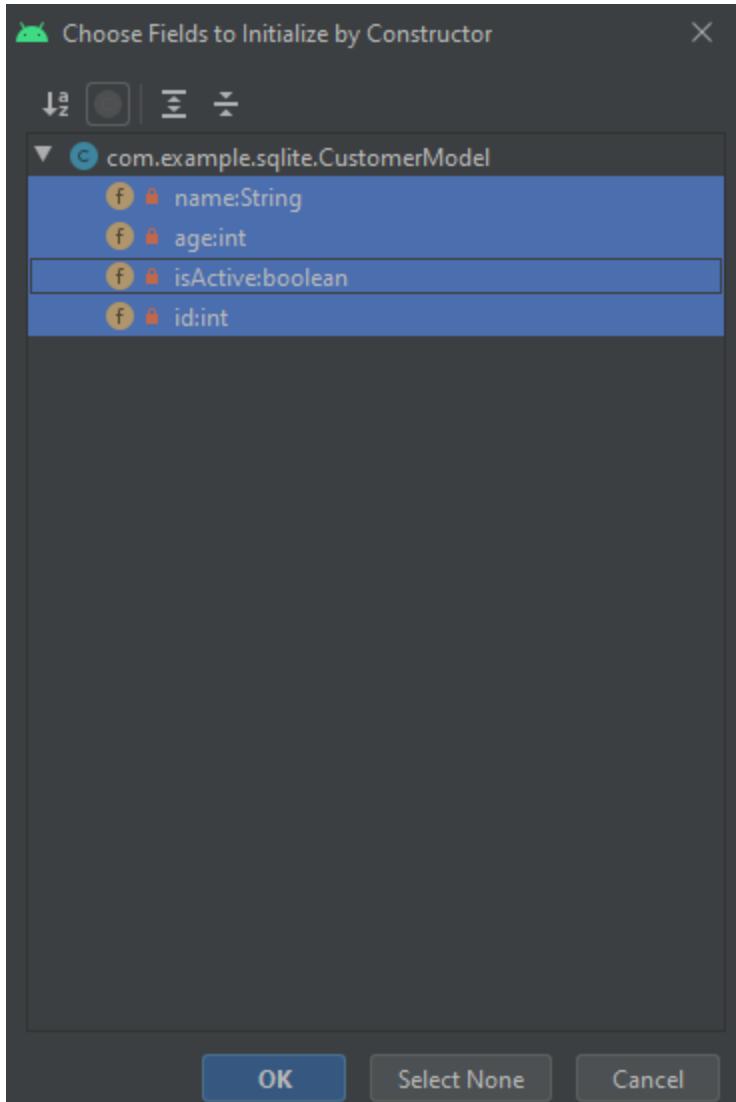
- Add a new Java class named as customerModel.



❖ Adding Constructor:

- After making a new java class add constructor using following steps.



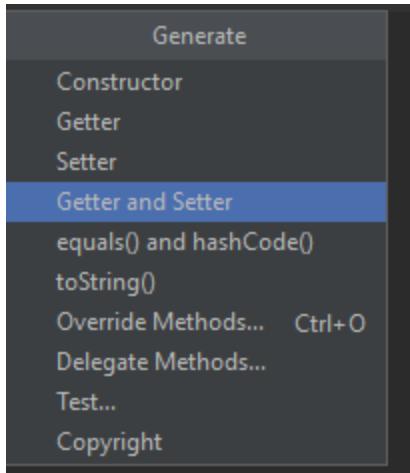


- After following above steps, following is the constructor that has been added to the class.

```
public CustomerModel(String name, int age, boolean isActive, int id) {  
    this.name = name;  
    this.age = age;  
    this.isActive = isActive;  
    this.id = id;  
}
```

❖ Adding Getter Setter:

- To add getter setter in the java class following are the steps to follow.

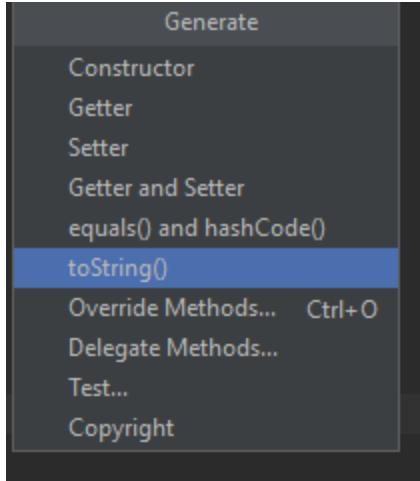


- After following the above steps, following is the view of getter setters that have been added to java class.

```
public String getName() {  
    return name;  
}  
  
public void setName(String name) {  
    this.name = name;  
}  
  
public int getAge() {  
    return age;  
}  
  
public void setAge(int age) {  
    this.age = age;  
}
```

Override `toString` method:

Now override the `toString()` method in class using following steps.



- After following the above steps, following is the view of `toString()` method that have been added to java class.

```
💡 @Override
public String toString() {
    return "CustomerModel{" +
        "name='" + name + '\'' +
        ", age=" + age +
        ", isActive=" + isActive +
        ", id=" + id +
        '}';
}
```

❖ Adding class object in main activity:

- Add instance of the above class in main activity so we can get its values.

```
setContentView(R.layout.activity_main);

btnAdd= findViewById(R.id.buttonAdd);
btnView=findViewById(R.id.buttonView);
editName=findViewById(R.id.editTextName);
editAge=findViewById(R.id.editTextAge);
switchIsActive=findViewById(R.id.switchisActive);

btnAdd.setOnClickListener(new View.OnClickListener()
{
    CustomerModel customerModel;

    @Override
    public void onClick(View view) {

        CustomerModel customerModel=new CustomerModel(editName.getText().toString(), Integer.parseInt(editAge.getText().toString()), switchIsActive.isChecked(), id:());

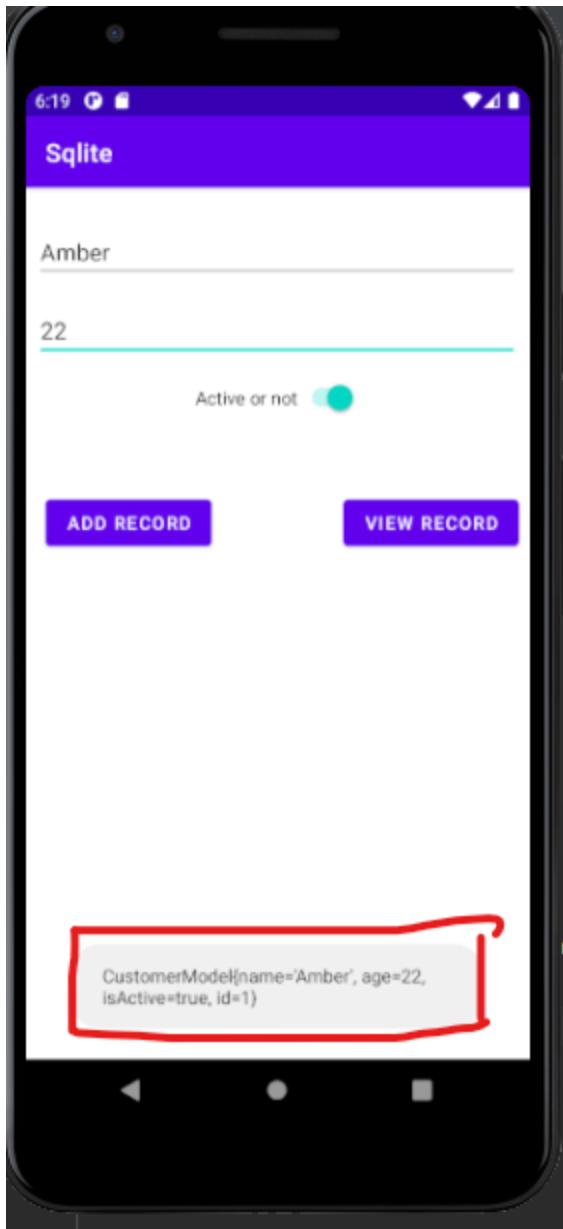
        // Toast.makeText(MainActivity.this,"Add Button Clicked",Toast.LENGTH_SHORT).show();
    }
});
```

❖ Add new toast to see data added:

- Now to see the data added on click of button we add a new toast to get string values that have been passed to customer's object.

```
Toast.makeText( context: MainActivity.this, customerModel.toString(), Toast.LENGTH_LONG).show();  
// Toast.makeText(MainActivity.this, "Add Button Clicked", Toast.LENGTH_SHORT).show();
```

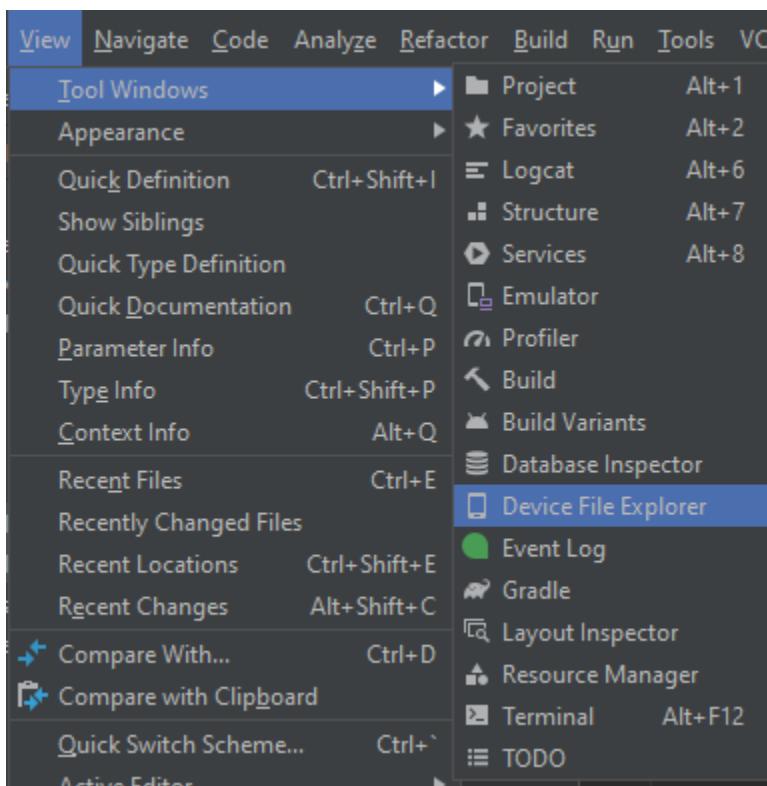
- Run the emulator to see the data added on click of add record button.
- As we enter the data in text fields and click the add button, we will get its values represented in bottom of screen that are displayed by toast.



To Open Created Database:

❖ To view files of the device:

- Run the emulator first, and then follow the steps bellow.

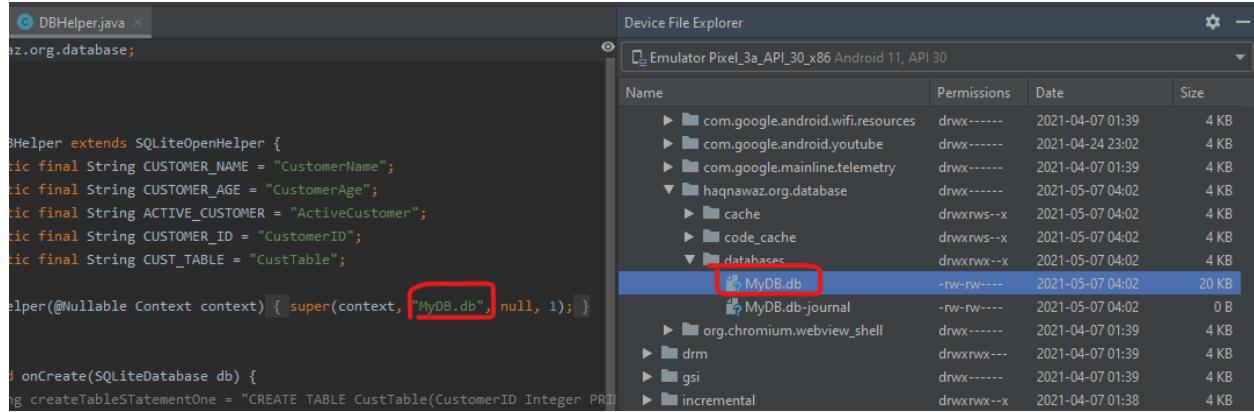


- Here in **data folder** are all applications that are available that we run on the emulator.

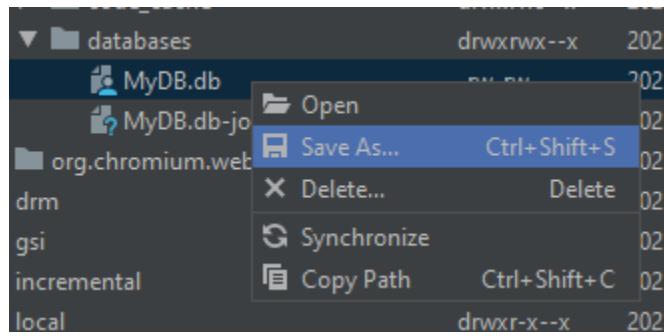
Name	Permissi...	Date	Size
► bin	lrw-r--r--	2020-10-14 03:35	11 B
► cache	lrw-r--r--	2020-10-14 03:35	11 B
► config	drwxr-xr-x	2021-04-24 23:01	0 B
▼ data	drwxrwx-->	2021-04-24 23:02	4 KB
► adb	drwx-----	2021-04-07 01:39	4 KB
► anr	drwxrwxr-x	2021-04-15 05:48	4 KB
► apex	drwxr-xr-x	2021-04-24 23:01	4 KB
► app	drwxrwx-->	2021-05-06 18:15	4 KB
► app-asec	drwx-----	2021-04-07 01:38	4 KB
► app-ephemeral	drwxrwx-->	2021-04-07 01:38	4 KB
► app-lib	drwxrwx-->	2021-04-07 01:38	4 KB
► app-private	drwxrwx-->	2021-04-07 01:38	4 KB
► app-staging	drwxr-x---	2021-04-07 01:38	4 KB
► backup	drwx-----	2021-05-06 18:15	4 KB
► bootchart	drwxr-xr-x	2021-04-07 01:38	4 KB
► cache	drwxrwx-->	2021-04-07 01:38	4 KB
► dalvik-cache	drwxrwx-->	2021-04-07 01:39	4 KB
▼ data	drwxrwx-->	2021-04-26 01:08	12 KB
► android	drwx-----	2021-04-24 23:02	4 KB
► android.auto_generated_i	drwx-----	2021-04-07 01:39	4 KB
► com.android.backupconf	drwx-----	2021-04-07 01:39	4 KB
► com.android.bips	drwx-----	2021-04-24 23:02	4 KB
► com.android.bips.auto_g	drwx-----	2021-04-07 01:39	4 KB
► com.android.bluetooth	drwx-----	2021-04-24 23:02	4 KB
► com.android.bluetoothm	drwx-----	2021-04-07 01:39	4 KB
► com.android.bookmarkpi	drwx-----	2021-04-07 01:39	4 KB

❖ To View Database file:

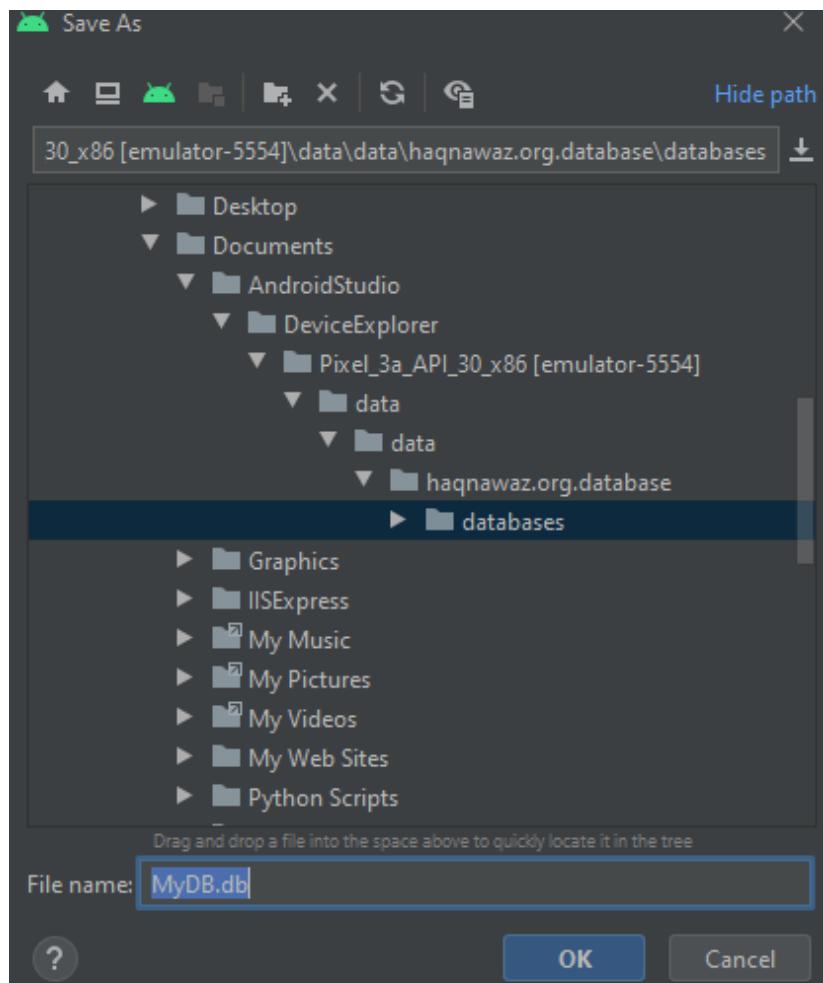
- To see the database file open the project's package and click on its database as named in the DB helper class.



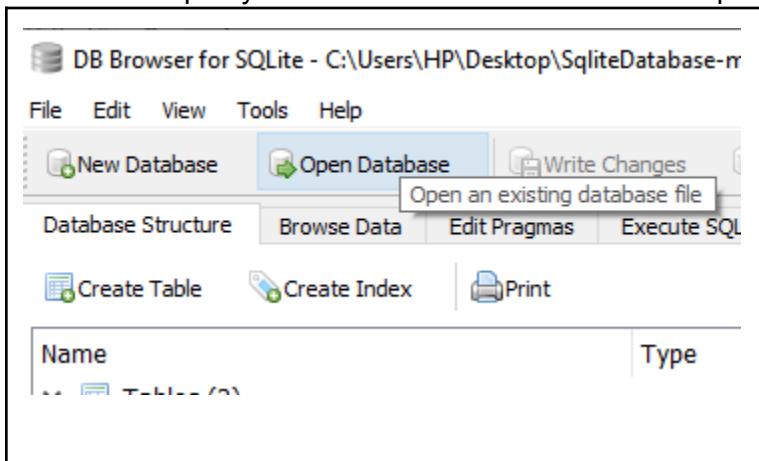
- Export the database on your system using save as.



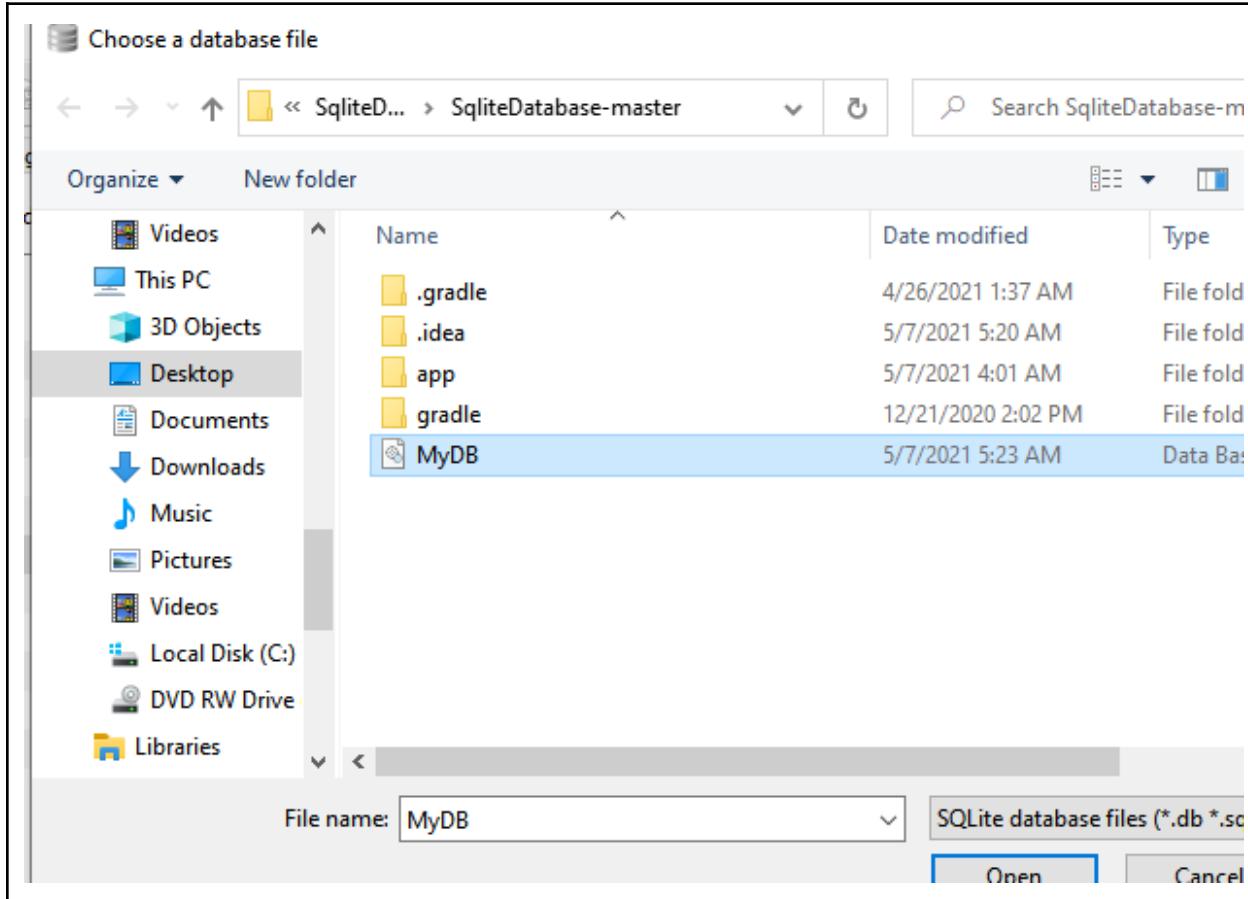
- Select a location to save your database, so you can view it later on in **SQLite Browser**.



- Now open your SQLite Browser and select the option to open database.



- Choose the database file you want to open.



- Here is the database you have opened in your SQLite Browser.

The screenshot shows the DB Browser for SQLite interface. The title bar reads "DB Browser for SQLite - C:\Users\HP\Desktop\SqliteDatabase-master\SqliteDatabase-master\MyDB.db". The menu bar includes File, Edit, View, Tools, and Help. The toolbar has buttons for New Database, Open Database, Write Changes, Revert Changes, Open Project, Save Project, Database Structure (selected), Browse Data, Edit Pragmas, Execute SQL, Create Table, Create Index, Modify Table, Delete Table, and Print. The main window displays a table with three columns: Name, Type, and Schema. The "Name" column lists Tables (3), Indices (0), Views (0), and Triggers (0). The "Type" column shows the table types. The "Schema" column shows the CREATE TABLE statements for each table. The "CustTable" row is currently selected, with its schema visible in the Schema column.

Name	Type	Schema
Tables (3)		
CustTable	CREATE TABLE CustTable(CustomerID Integer PRIMARY KEY, CustomerName Text, CustomerAddress Text, CustomerCity Text, CustomerState Text, CustomerZip Text, CustomerPhone Text, CustomerEmail Text, CustomerFax Text, CustomerNotes Text)	
android_metadata	CREATE TABLE android_metadata (locale TEXT)	
sqlite_sequence	CREATE TABLE sqlite_sequence(name,seq)	
Indices (0)		
Views (0)		
Triggers (0)		

LECTURE#11

SQLite Lec#3

❖ Retrieving Data from Database:

- Make a function `getAllRecords()` will return a list of type customer model.

```
}
```

```
public List<CustomerModel>getAllRecords(){
```

```
    List<CustomerModel> myList =new ArrayList<>();
```

```
    //SQL Query
```

```
    String query="SELECT *FROM "+ CUST_TABLE;
```

```
    Cursor cursor=db.rawQuery(query,null);
```

```
    if(cursor.moveToFirst())
```

```
        do{
```

```
            CustomerModel cm=new CustomerModel();
```

```
            cm.setCustId(cursor.getInt(cursor.getColumnIndex("CUST_ID")));
```

```
            cm.setName(cursor.getString(cursor.getColumnIndex("NAME")));
```

```
            cm.setAddress(cursor.getString(cursor.getColumnIndex("ADDRESS")));
```

```
            cm.setCity(cursor.getString(cursor.getColumnIndex("CITY")));
```

```
            cm.setState(cursor.getString(cursor.getColumnIndex("STATE")));
```

```
            cm.setZip(cursor.getString(cursor.getColumnIndex("ZIP")));
```

```
            myList.add(cm);
```

```
        }while(cursor.moveToNext());
```

```
    cursor.close();
```

```
    return myList;
```

```
}
```

Define Query:

- Define the following query to retrieve data from database.

```
}
```

```
public List<CustomerModel>getAllRecords(){
```

```
    List<CustomerModel> myList =new ArrayList<>();
```

```
    String query="SELECT *FROM "+ CUST_TABLE;
```

```
    Cursor cursor=db.rawQuery(query,null);
```

```
    if(cursor.moveToFirst())
```

```
        do{
```

```
            CustomerModel cm=new CustomerModel();
```

```
            cm.setCustId(cursor.getInt(cursor.getColumnIndex("CUST_ID")));
```

```
            cm.setName(cursor.getString(cursor.getColumnIndex("NAME")));
```

```
            cm.setAddress(cursor.getString(cursor.getColumnIndex("ADDRESS")));
```

```
            cm.setCity(cursor.getString(cursor.getColumnIndex("CITY")));
```

```
            cm.setState(cursor.getString(cursor.getColumnIndex("STATE")));
```

```
            cm.setZip(cursor.getString(cursor.getColumnIndex("ZIP")));
```

```
            myList.add(cm);
```

```
        }while(cursor.moveToNext());
```

```
    cursor.close();
```

```
    return myList;
```

```
}
```

❖ Make an object of SQLite Database:

- To get record make an object of database.

```
}
```

```
public List<CustomerModel>getAllRecords(){
```

```
    List<CustomerModel> myList =new ArrayList<>();
```

```
    String query="SELECT *FROM "+ CUST_TABLE;
```

```
    SQLiteDatabase DB=this.getReadableDatabase();
```

```
    Cursor cursor=DB.rawQuery(query,null);
```

```
    if(cursor.moveToFirst())
```

```
        do{
```

```
            CustomerModel cm=new CustomerModel();
```

```
            cm.setCustId(cursor.getInt(cursor.getColumnIndex("CUST_ID")));
```

```
            cm.setName(cursor.getString(cursor.getColumnIndex("NAME")));
```

```
            cm.setAddress(cursor.getString(cursor.getColumnIndex("ADDRESS")));
```

```
            cm.setCity(cursor.getString(cursor.getColumnIndex("CITY")));
```

```
            cm.setState(cursor.getString(cursor.getColumnIndex("STATE")));
```

```
            cm.setZip(cursor.getString(cursor.getColumnIndex("ZIP")));
```

```
            myList.add(cm);
```

```
        }while(cursor.moveToNext());
```

```
    cursor.close();
```

```
    return myList;
```

```
}
```

❖ **Curser:** used to return dataSet.(kind of a table)

- Define **rawQuery()** that will return us a cursor as dataset.

```
public List<CustomerModel> getAllRecords(){

    List<CustomerModel> myList = new ArrayList<>();
    String query="SELECT *FROM "+ CUST_TABLE;
    SQLiteDatabase DB=this.getReadableDatabase();
    Cursor cursor=DB.rawQuery(query, selectionArgs: null);

    return myList;
}
```

❖ **Checks and place records:**

- Place a check in if condition to check if records are available, then get them.

```
if(cursor.moveToFirst()){
    do{
        String custName=cursor.getString( columnIndex: 1);
        int custAge=cursor.getInt( columnIndex: 2);
        Boolean isActive=cursor.getInt( columnIndex: 3)==1?true:false;

    }while(cursor.moveToNext());
}
```

❖ **Using Model:**

- Now define a customer class model to get its values to database.

```
if(cursor.moveToFirst()){
    do{
        String custName=cursor.getString( columnIndex: 1);
        int custAge=cursor.getInt( columnIndex: 2);
        Boolean isActive=cursor.getInt( columnIndex: 3)==1?true:false;

        CustomerModel newCustomerModel=new CustomerModel(custName,custAge,isActive, id: 1);
    }while(cursor.moveToNext());
}
```

❖ Adding to list:

- Add values of model to list in order to display them.

```
if(cursor.moveToFirst()){
    do{
        String custName=cursor.getString( columnIndex: 1);
        int custAge=cursor.getInt( columnIndex: 2);
        Boolean isActive=cursor.getInt( columnIndex: 3)==1?true:false;

        CustomerModel newCustomerModel=new CustomerModel(custName,cu
        myList.add(newCustomerModel);
    }while(cursor.moveToNext());
}
```

❖ Closing Resources after use:

- Close the used resources after getting values from them.

```
if(cursor.moveToFirst()){
    do{
        String custName=cursor.getString( columnIndex: 1);
        int custAge=cursor.getInt( columnIndex: 2);
        Boolean isActive=cursor.getInt( columnIndex: 3)==1?true:false;

        CustomerModel newCustomerModel=new CustomerModel(custName,cu
        myList.add(newCustomerModel);
    }while(cursor.moveToNext());
}
cursor.close();
DB.close();
return myList;
}
```

❖ Add DB Helper & List:

- OnClick() Add DBHelper and list so we can get values from database on click.

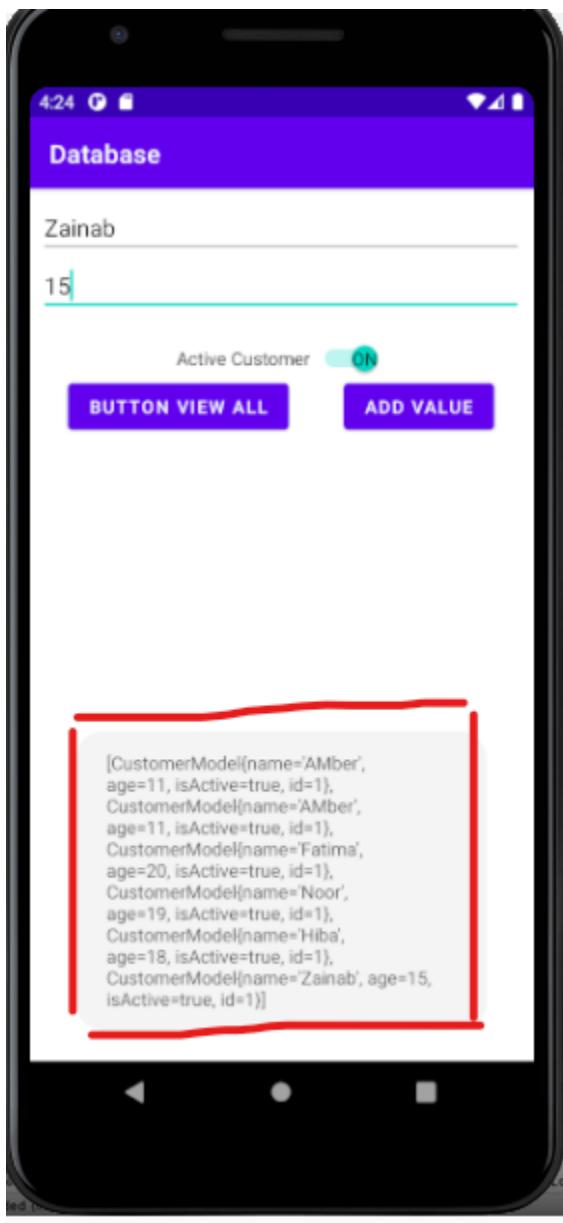
```
buttonViewAll.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        DBHelper dbHelper=new DBHelper( context: MainActivity.this);
        List<CustomerModel> allCustomers=dbHelper.getAllRecords();
```

❖ Displaying in toast:

- Use a toast to check if values are being added on click of add button.

```
public void onClick(View v) {  
    DBHelper dbHelper=new DBHelper( context: MainActivity.this)  
    List<CustomerModel> allCustomers=dbHelper.getAllRecords();  
  
    Toast.makeText( context: MainActivity.this, allCustomers.toString(), Toast.LENGTH_SHORT).show();  
}  
};
```

- Click on add value then view record



❖ To Display in ListView:

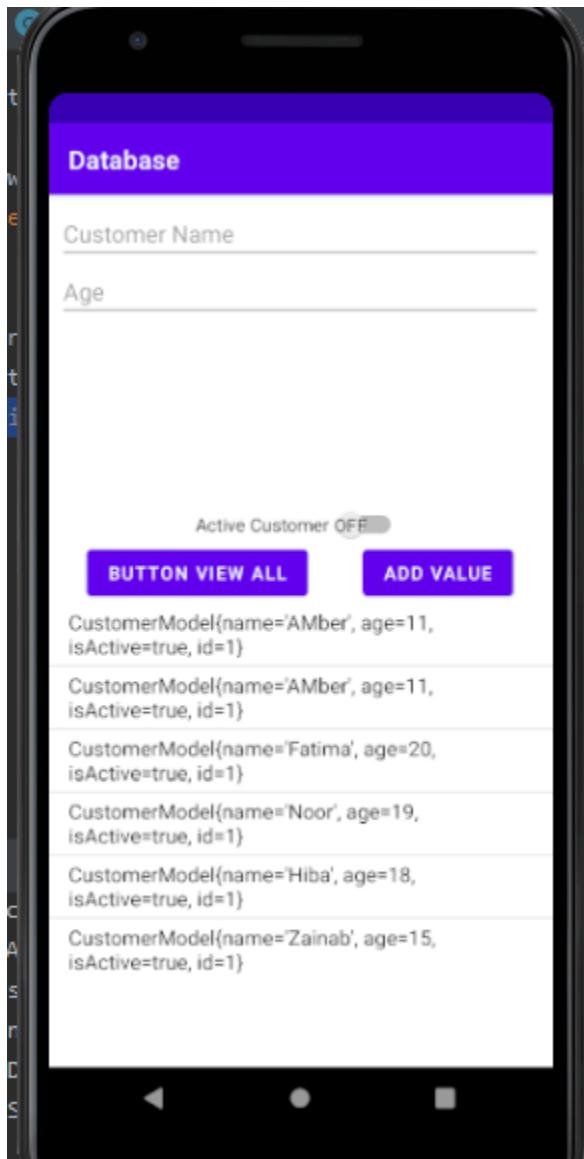
- List is made so make a new array adapter to show it.

```
ListView listViewCustomer;
ArrayAdapter<CustomerModel> arrayAdapter;
@Override
```

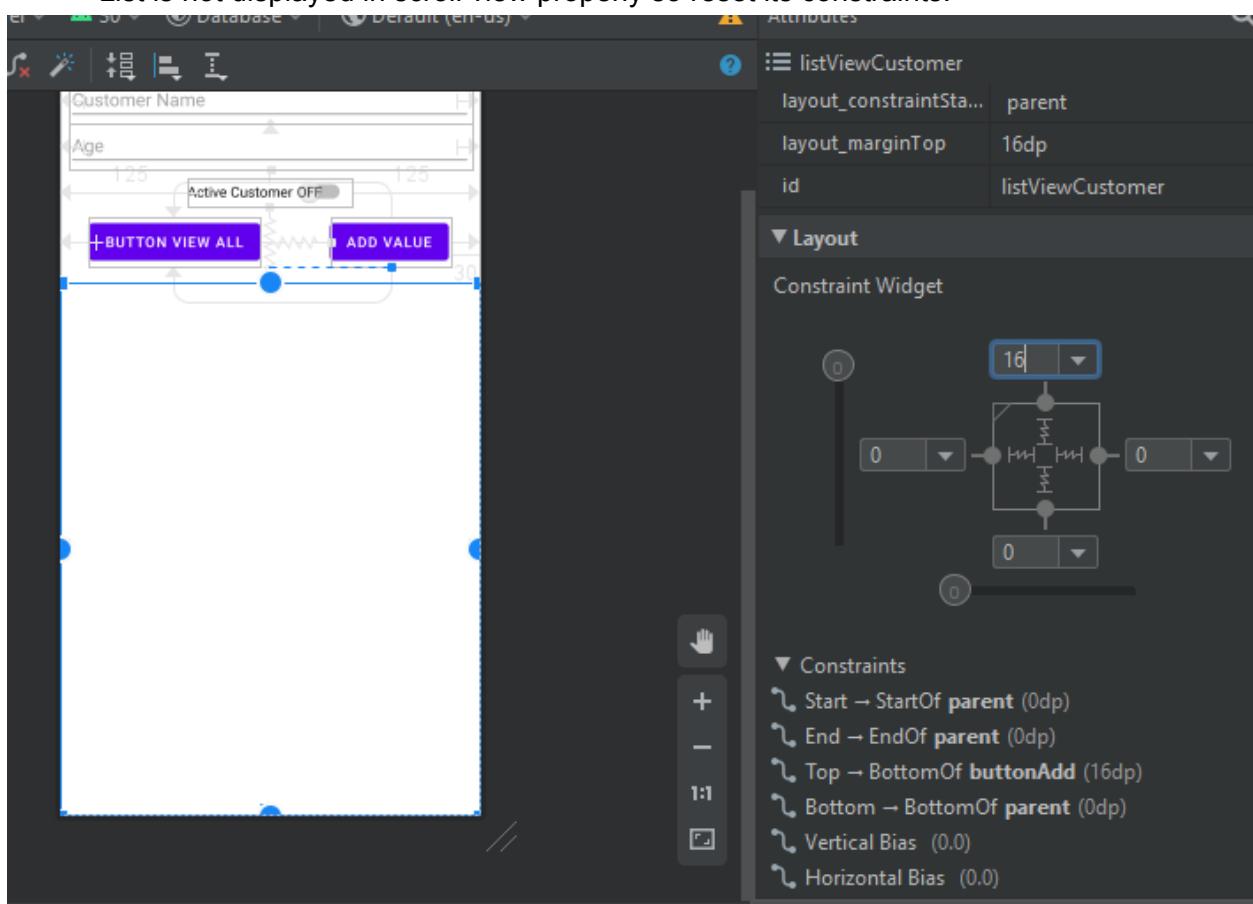
- Pass customer class values to array adapter then pass adapter to list to get displayed.

```
arrayAdapter=new ArrayAdapter<CustomerModel>( context: MainActivity.this, android.R.layout.simple_list_item
listViewCustomer.setAdapter(arrayAdapter);
```

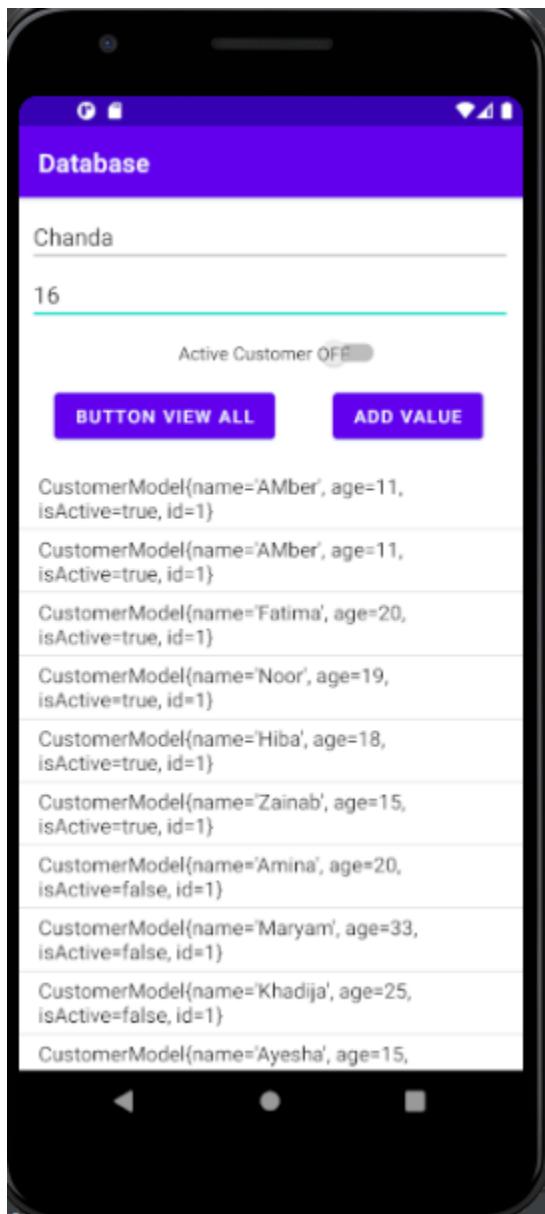
- Now as we run the emulator we can see values being displayed in list on clicking add value.



- List is not displayed in scroll view properly so reset its constraints.

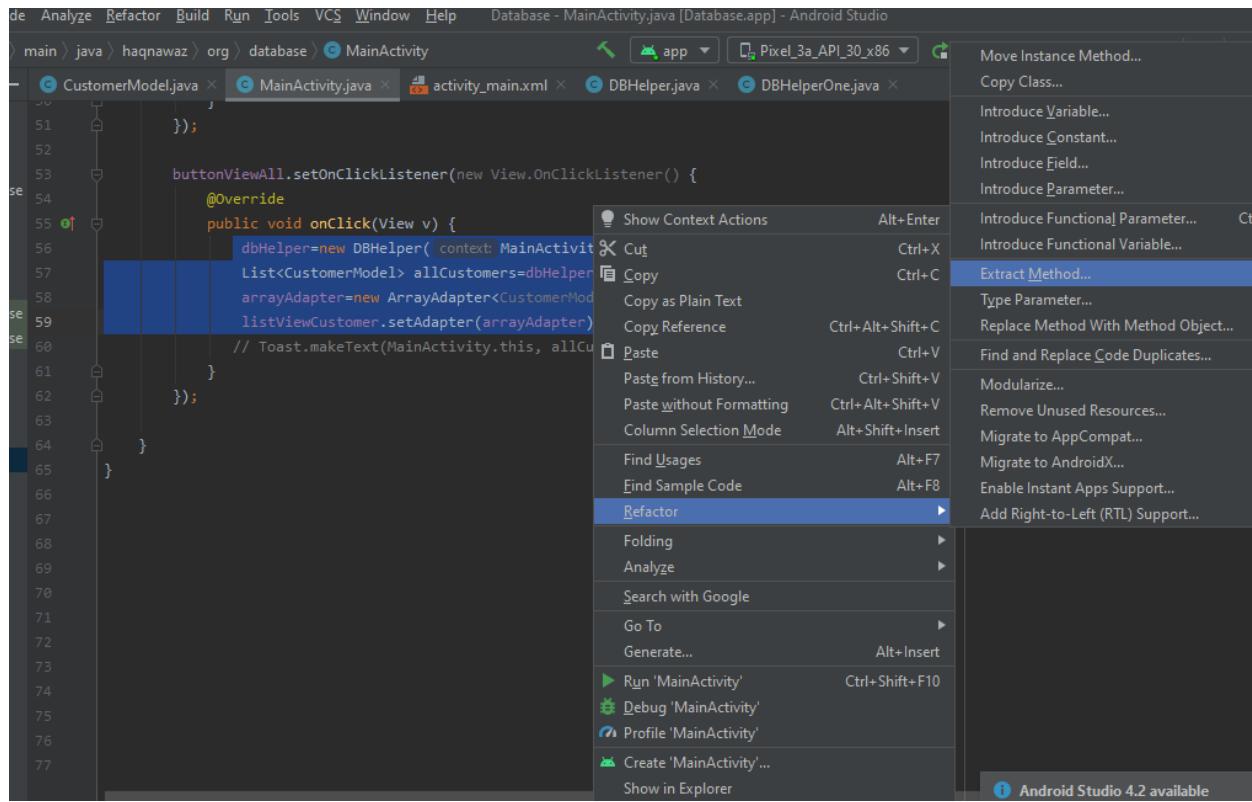


- After setting constraints rerun the emulator and see a scrolling view of the records being displayed in list.

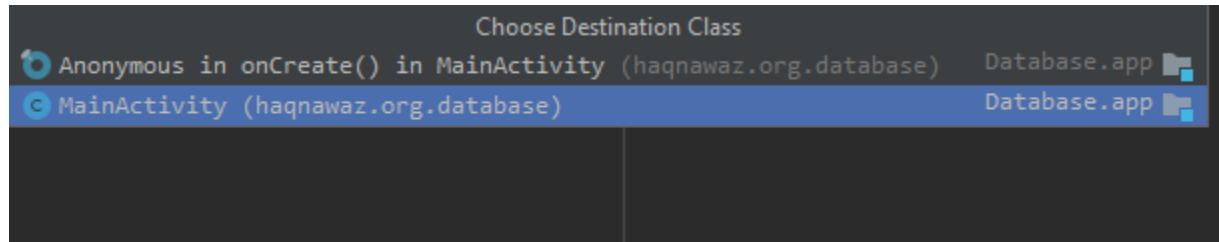


❖ Refactor Method:

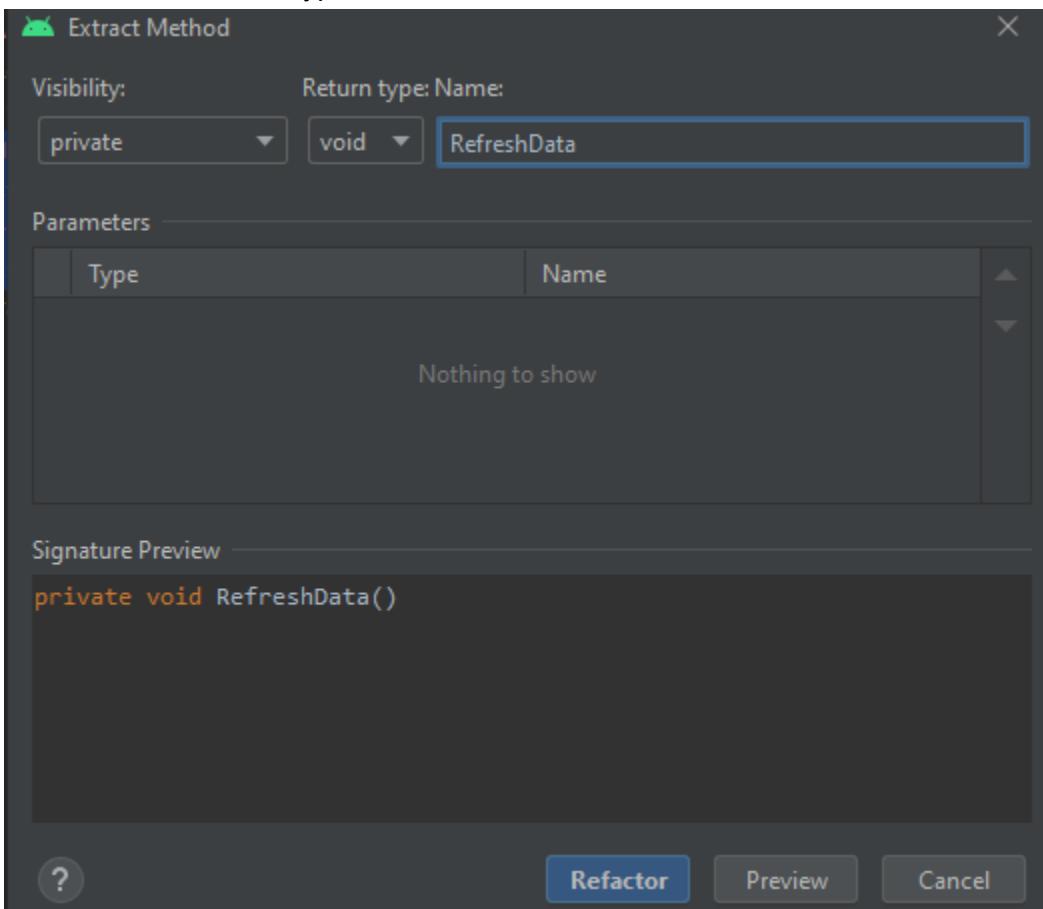
- To refactor method follow the steps bellow.



- Choose the activity where you want to refactor.



- Select a return type name for the method to be refactored.



- Now instead of using all the statements just call the refactored method in click function.

```
buttonViewAll.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        RefreshData();
        // Toast.makeText(MainActivity.this, allCustomers.toString(), Toast.LENGTH_LONG)
    }
});
```



```
private void RefreshData() {
    dbHelper=new DBHelper( context: MainActivity.this);
    List<CustomerModel> allCustomers=dbHelper.getAllRecords();
    arrayAdapter=new ArrayAdapter<CustomerModel>( context: MainActivity.this, android.R.layout.
    listViewCustomer.setAdapter(arrayAdapter);
}
```

❖ Refresh on Add new record:

- Also add refactor method to on click try so record being added to list when clicked.

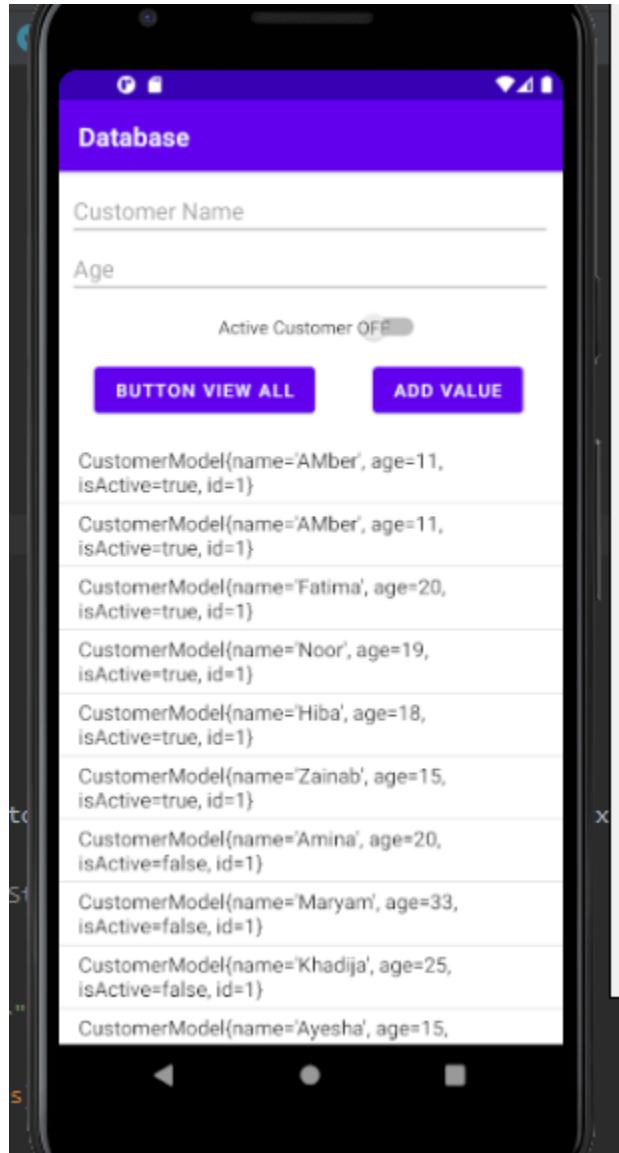
```
public void onClick(View v) {  
    try {  
        customerModel = new CustomerMo  
        RefreshData();  
        // Toast.makeText(MainActivity.this, "Customer  
    }  
}
```

❖ Refresh on OnCreate:

- Also add refactor method to on create so record being displayed not just on adding but also on first view of screen too.

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
  
    buttonAdd = findViewById(R.id.buttonAdd);  
    buttonViewAll = findViewById(R.id.buttonViewAll);  
    editTextName = findViewById(R.id.editTextName);  
    editTextAge = findViewById(R.id.editTextAge);  
    switchIsActive = findViewById(R.id.switchCustomer);  
    listViewCustomer = findViewById(R.id.listViewCustomer);  
    RefreshData();  
    buttonAdd.setOnClickListener(new View.OnClickListener() {  
        @Override  
        public void onClick(View v) {  
            String name = editTextName.getText().toString();  
            String age = editTextAge.getText().toString();  
            boolean isActive = switchIsActive.isChecked();  
            if (!name.isEmpty() & !age.isEmpty()) {  
                customerModel.addCustomer(name, age, isActive);  
                RefreshData();  
            } else {  
                Toast.makeText(MainActivity.this, "Please enter all  
            }  
        }  
    });  
}
```

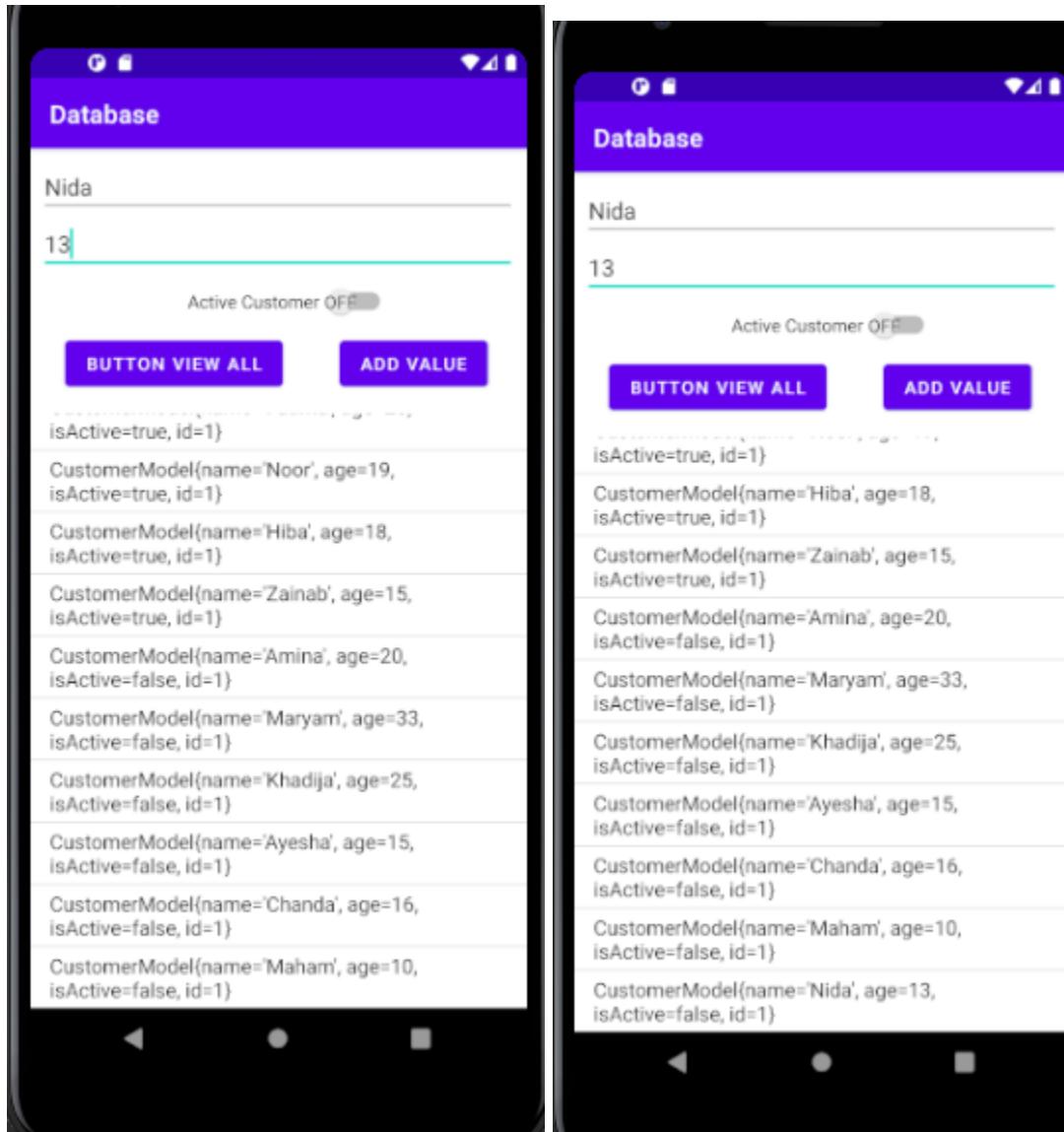
- Run the emulator and see the previous records on screen without even adding a new one.



- **Bug:**

- New Record added is displayed in list not on clicking add value but after clicking view all records.

Add value=> no record show **View all =>** record added show

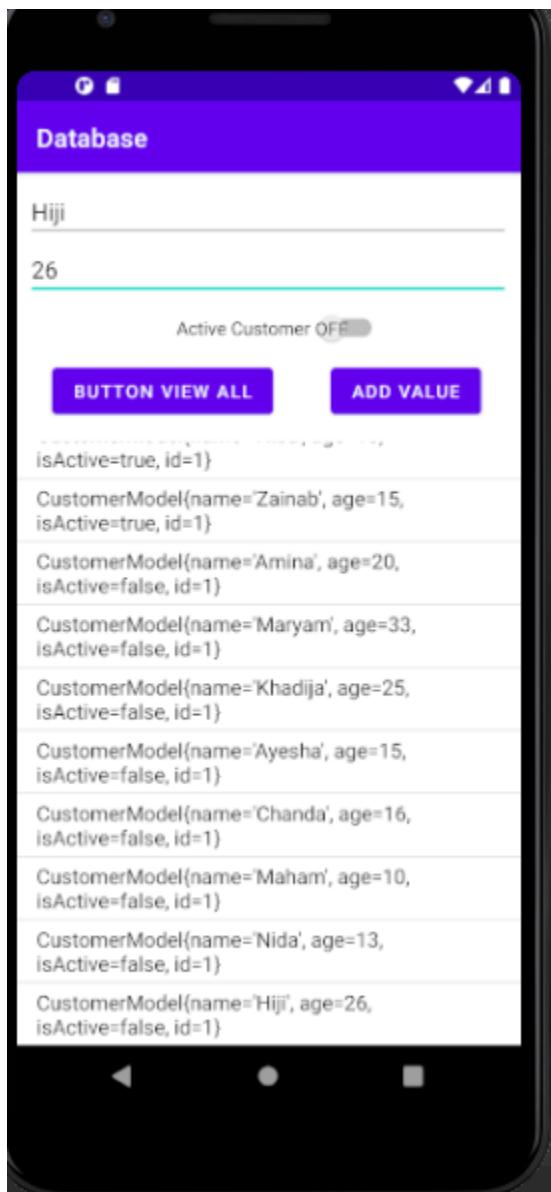


- Remove Bug:

- Add refresh method just after the values are being added to list.

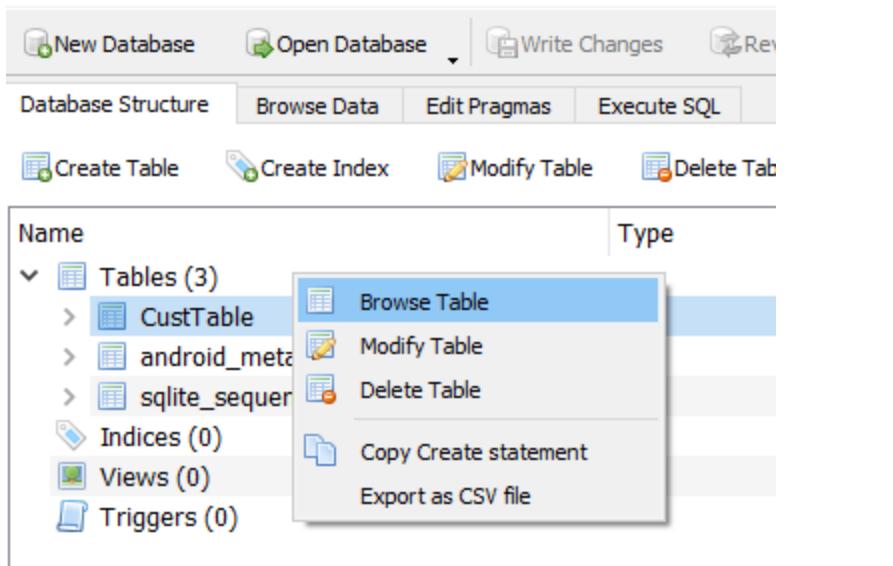
```
@Override
public void onClick(View v) {
    try {
        customerModel = new CustomerModel(editName.getText().to
            // Toast.makeText(MainActivity.this, customerModel.toSt
    }
    catch (Exception e){
        Toast.makeText( context: MainActivity.this, text: "Error"
    }
    DBHelper dbHelper = new DBHelper( context: MainActivity.this)
    boolean b = dbHelper.addCustomer(customerModel);
    RefreshData();
}
```

- Run the emulator and see ,Record displayed on clicking add value.



❖ To make changes in database:

- Open SQLite Browser, open your database and do the step below.



The screenshot shows the SQLite Browser interface with the 'Browse Data' tab selected. The 'Table' dropdown is set to 'CustTable'. The data grid displays 14 rows of customer information:

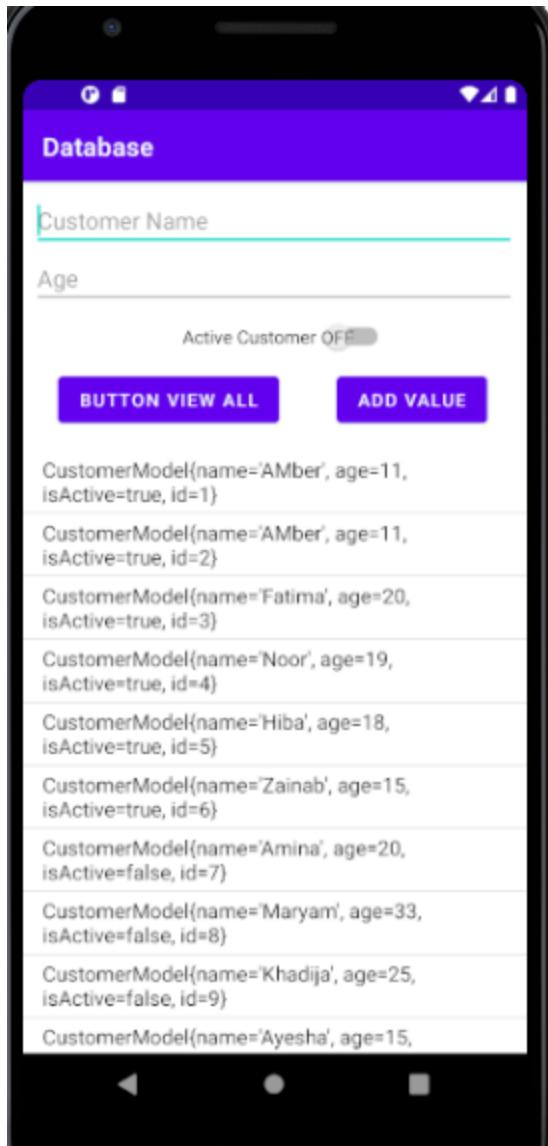
	CustomerID	CustomerName	CustomerAge	ActiveCustomer
	Filter	Filter	Filter	Filter
1	1	AMber	11	1
2	2	AMber	11	1
3	3	Fatima	20	1
4	4	Noor	19	1
5	5	Hiba	18	1
6	6	Zainab	15	1
7	7	Amina	20	0
8	8	Maryam	33	0
9	9	Khadija	25	0
10	10	Ayesha	15	0
11	11	Chanda	16	0
12	12	Maham	10	0
13	13	Nida	13	0
14	14	Hiji	26	0

Homework TASKS:

1. BUG=>Update id at data retrieval
2. On click delete record

```
Cursor cursor=db.rawQuery(query, selectionArgs, null);
if(cursor.moveToFirst()){
    do{
        int id= cursor.getInt( columnIndex: 0);
        String custName=cursor.getString( columnIndex: 1);
        int custAge=cursor.getInt( columnIndex: 2);
        Boolean isActive=cursor.getInt( columnIndex: 3)==1?true:false;

        CustomerModel newCustomerModel=new CustomerModel(custName,custAge,isActive,id);
        myList.add(newCustomerModel);
    }while(cursor.moveToNext());
}
```

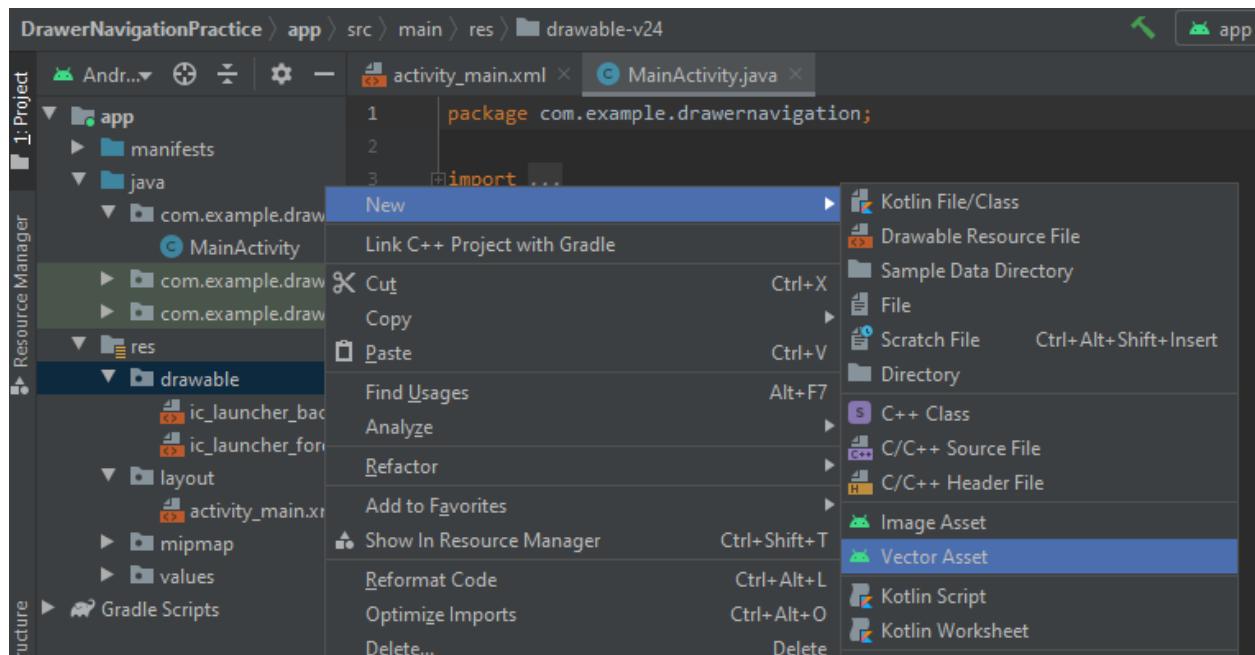


On click delete record

LECTURE#12

❖ Drawer Navigation:

To add an icon=>Vector Asset:



Asset Studio

Configure Vector Asset

Asset Type: Clip Art Local file (SVG, PSD)

Name: ic_android_black_24dp

Clip Art: 

Size: 24 dp X 24 dp

Color: #000000

Opacity: 100 %

Enable auto mirroring for RTL layout



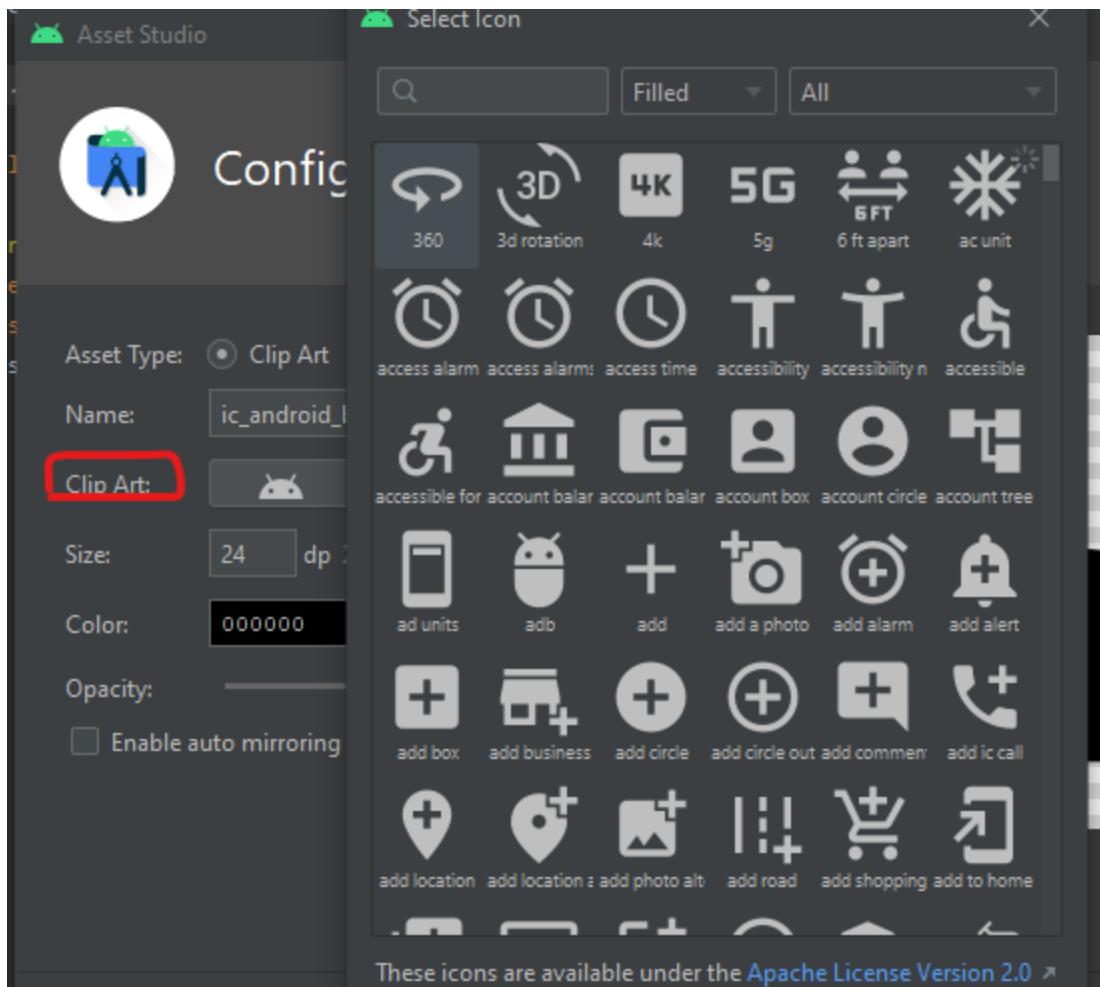
?

Previous

Next

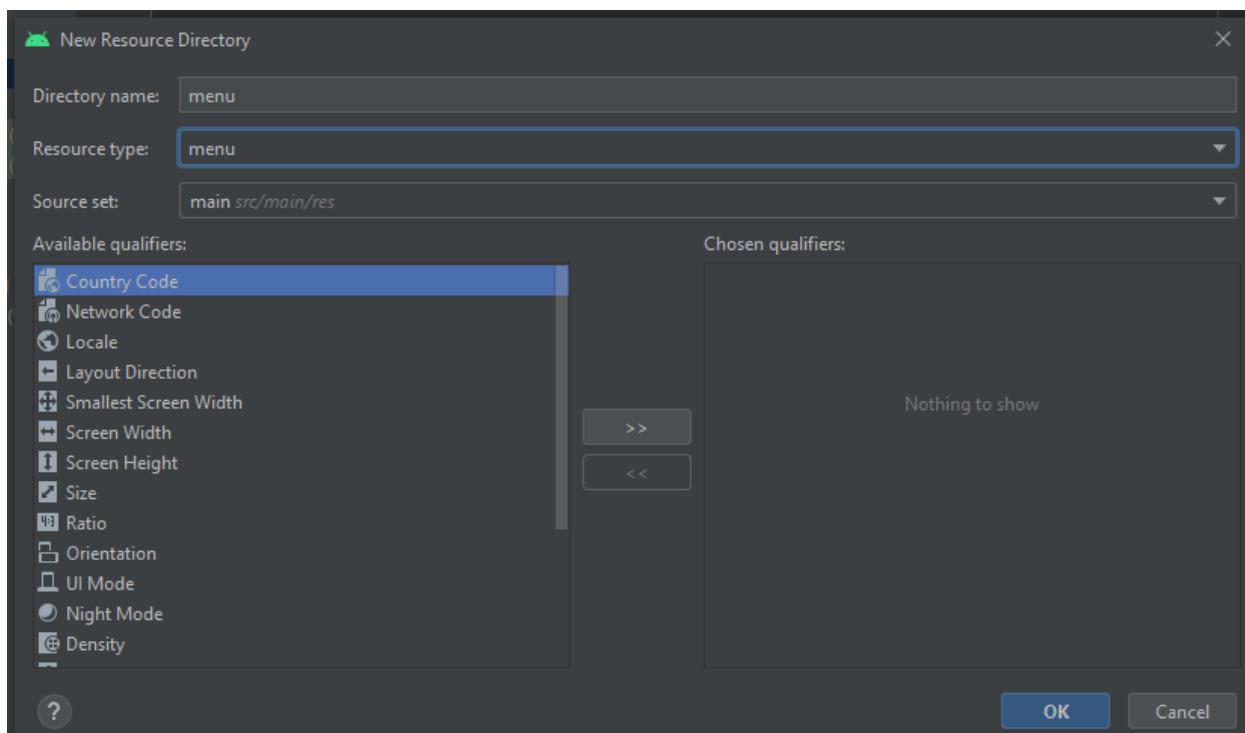
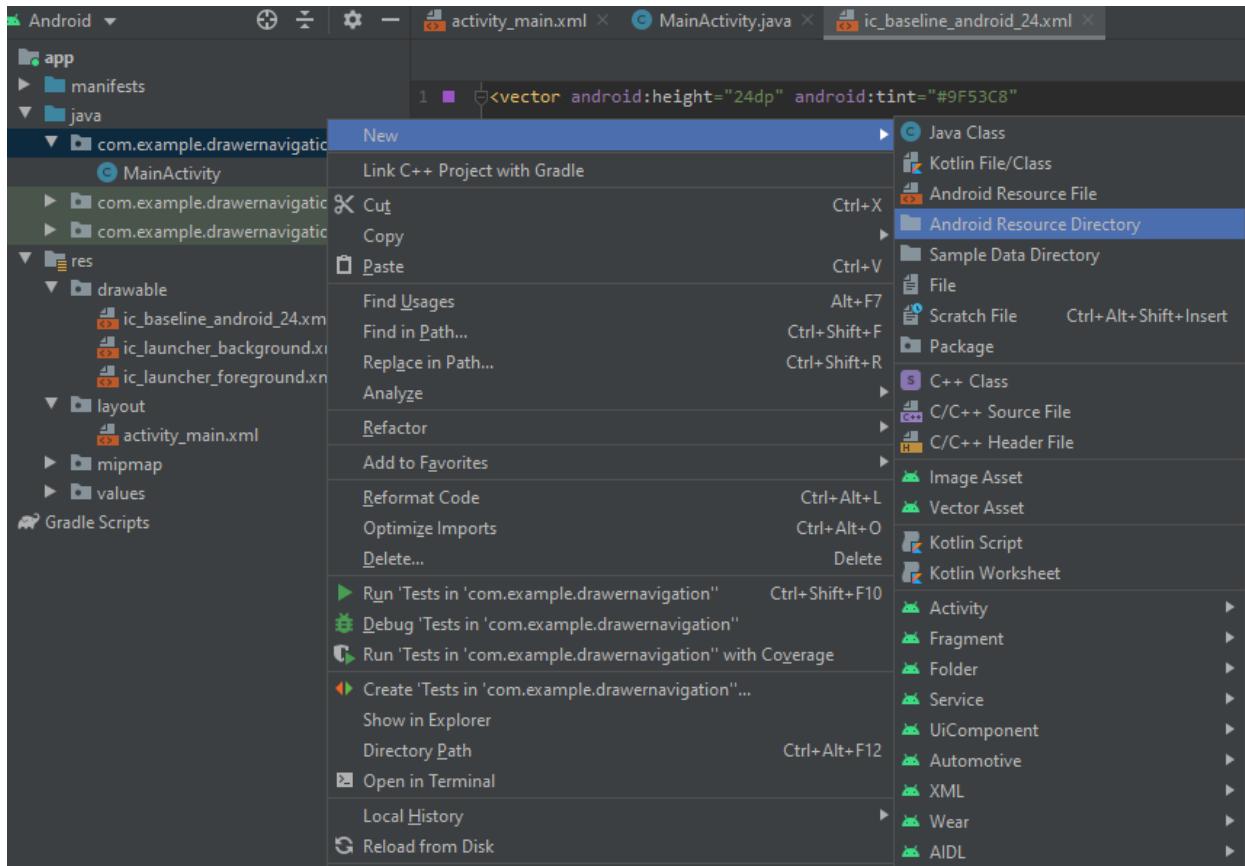
Cancel

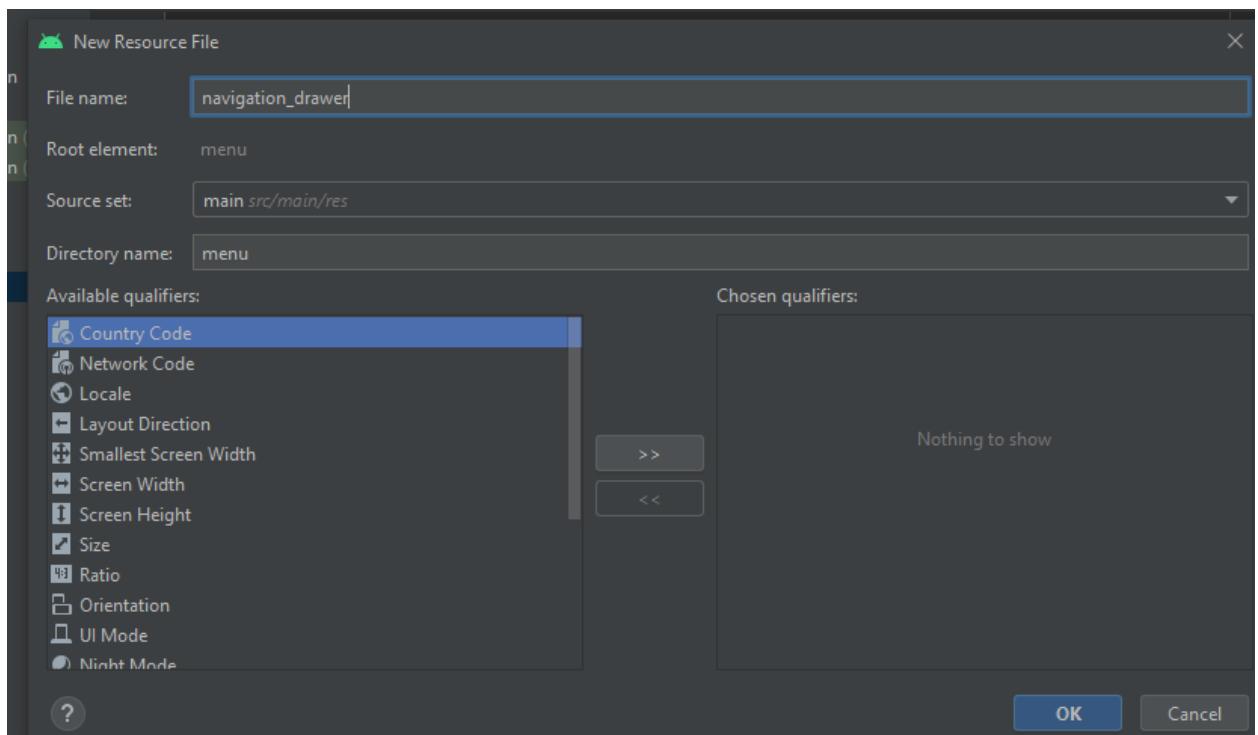
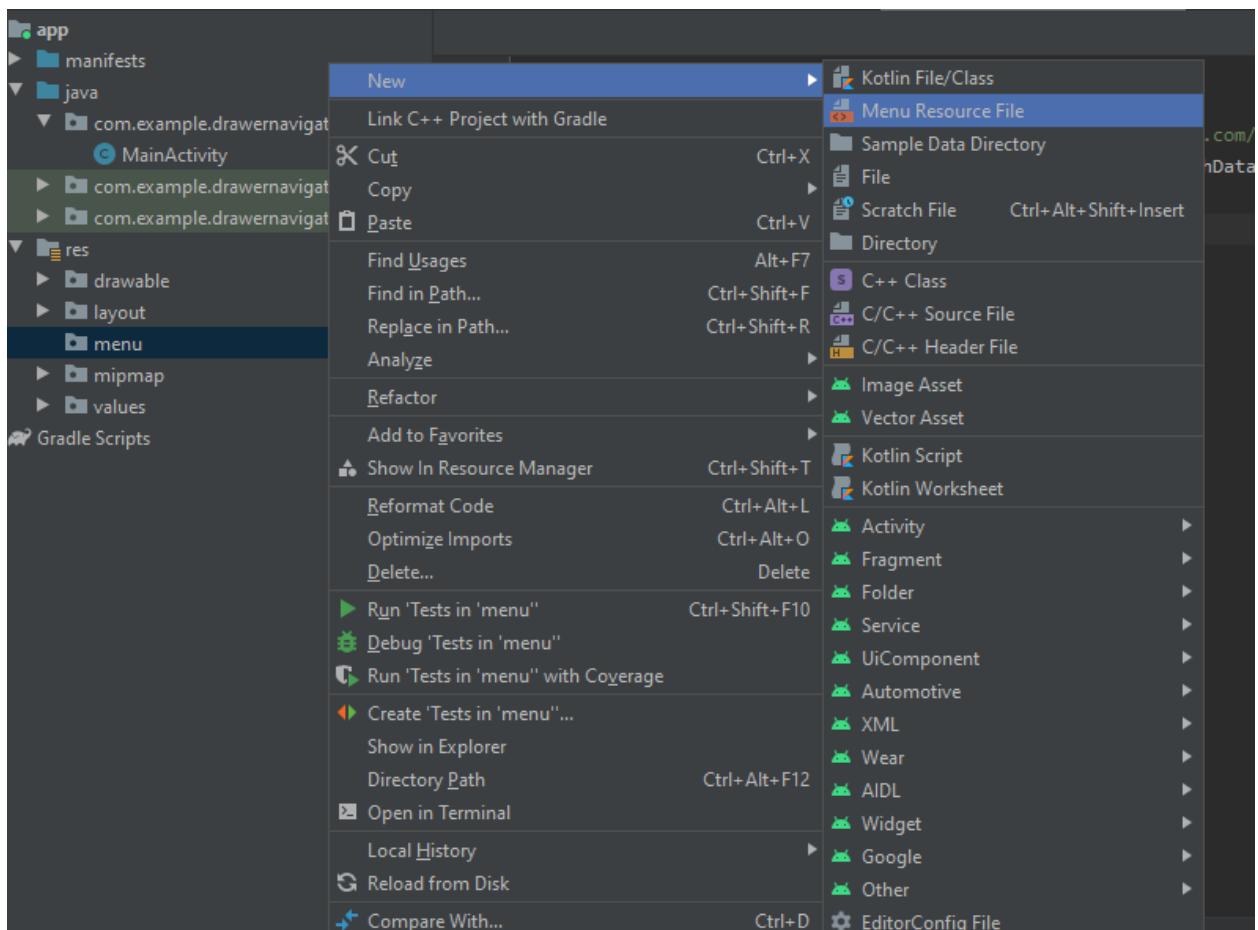
Finish

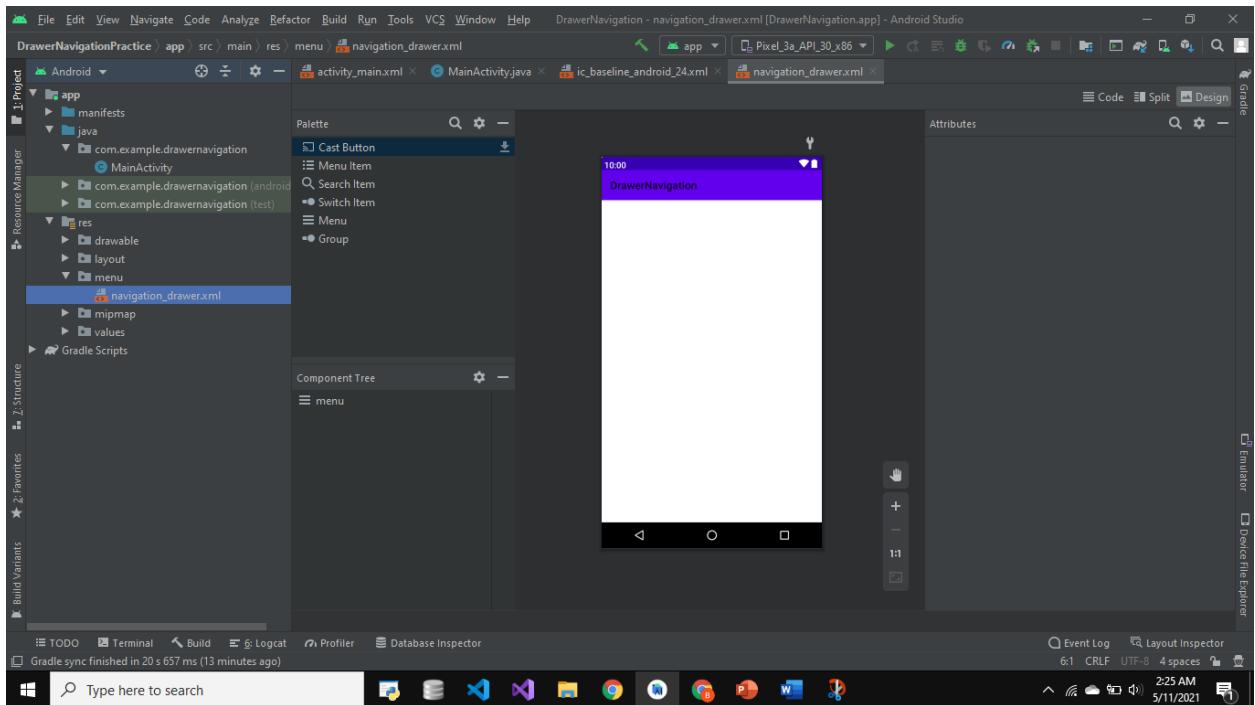


The screenshot shows the Android Studio code editor with the file 'ic_baseline_android_24.xml' selected. The code is as follows:

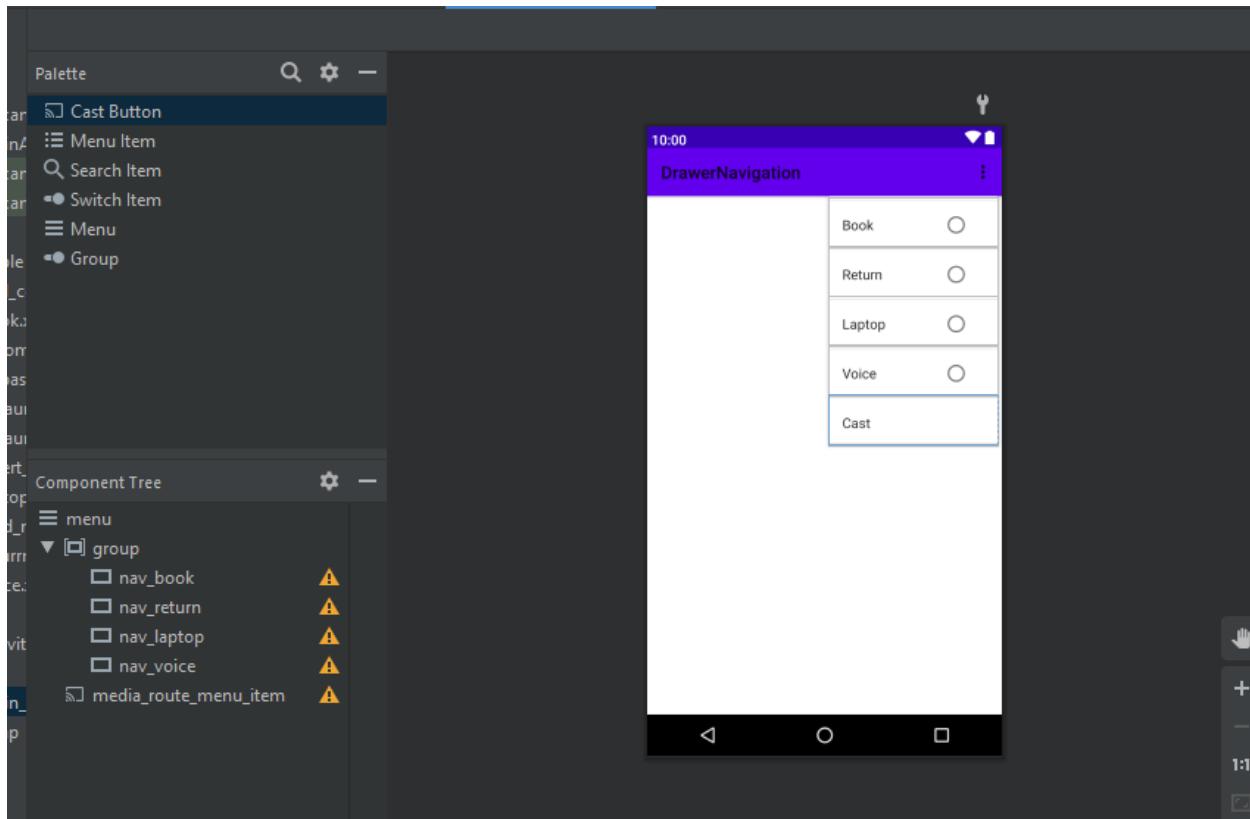
```
1 <vector android:height="24dp" android:tint="#9F53C8"
2     android:viewportHeight="24" android:viewportWidth="24"
3     android:width="24dp" xmlns:android="http://schemas.android.com/apk/res/android">
4     <path android:fillColor="@android:color/white" android:pathData="M17.6,9.48l1.84,-3.18c0.16,-
5         0.16,0.41,-0.41,0.41,-0.41"/>
6 </vector>
```

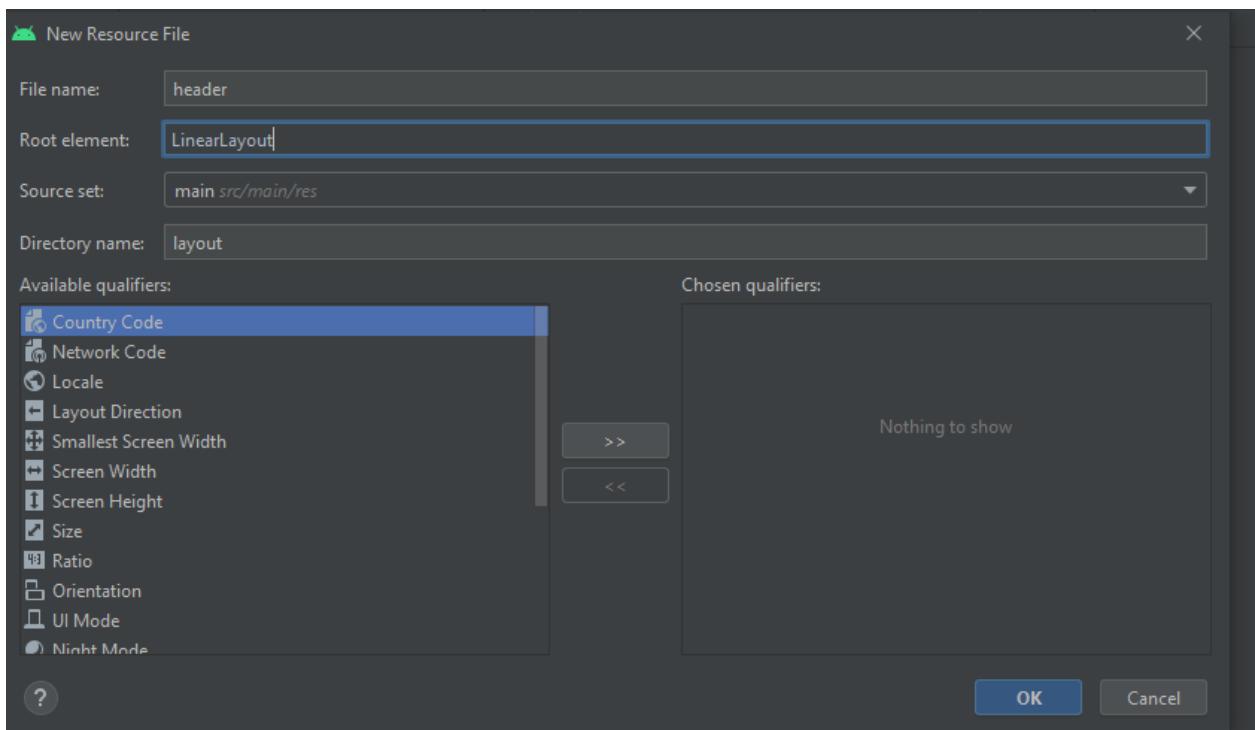
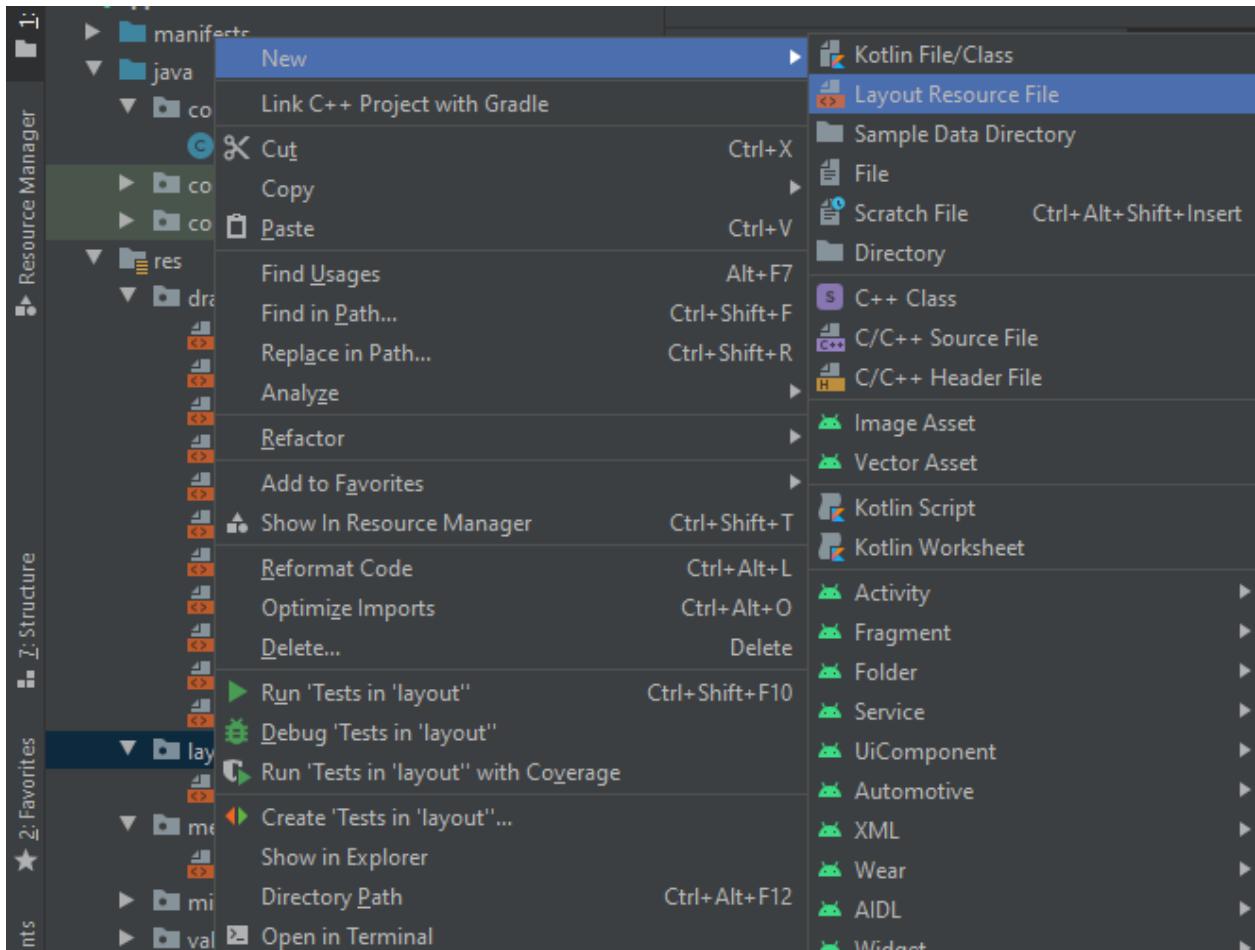






```
activity_main.xml × MainActivity.java × main_menue.xml ×
1   <?xml version="1.0" encoding="utf-8"?>
2   <menu xmlns:android="http://schemas.android.com/apk/res/android">
3
4       <group android:checkableBehavior="single">
5           <item
6               android:id="@+id/nav_book"
7               android:icon="@drawable/book"
8               android:title="Book" />
9
10          <item
11              android:id="@+id/nav_return"
12              android:icon="@drawable/returnn"
13              android:title="Return" />
14
15          <item
16              android:id="@+id/nav_laptop"
17              android:icon="@drawable/laptop"
18              android:title="Laptop" />
19
20          <item
21              android:id="@+id/nav_voice"
22              android:icon="@drawable/voice"
23              android:title="Voice" />
24
25      </group>
26      <item android:title="Reading">
27          <menu>
```





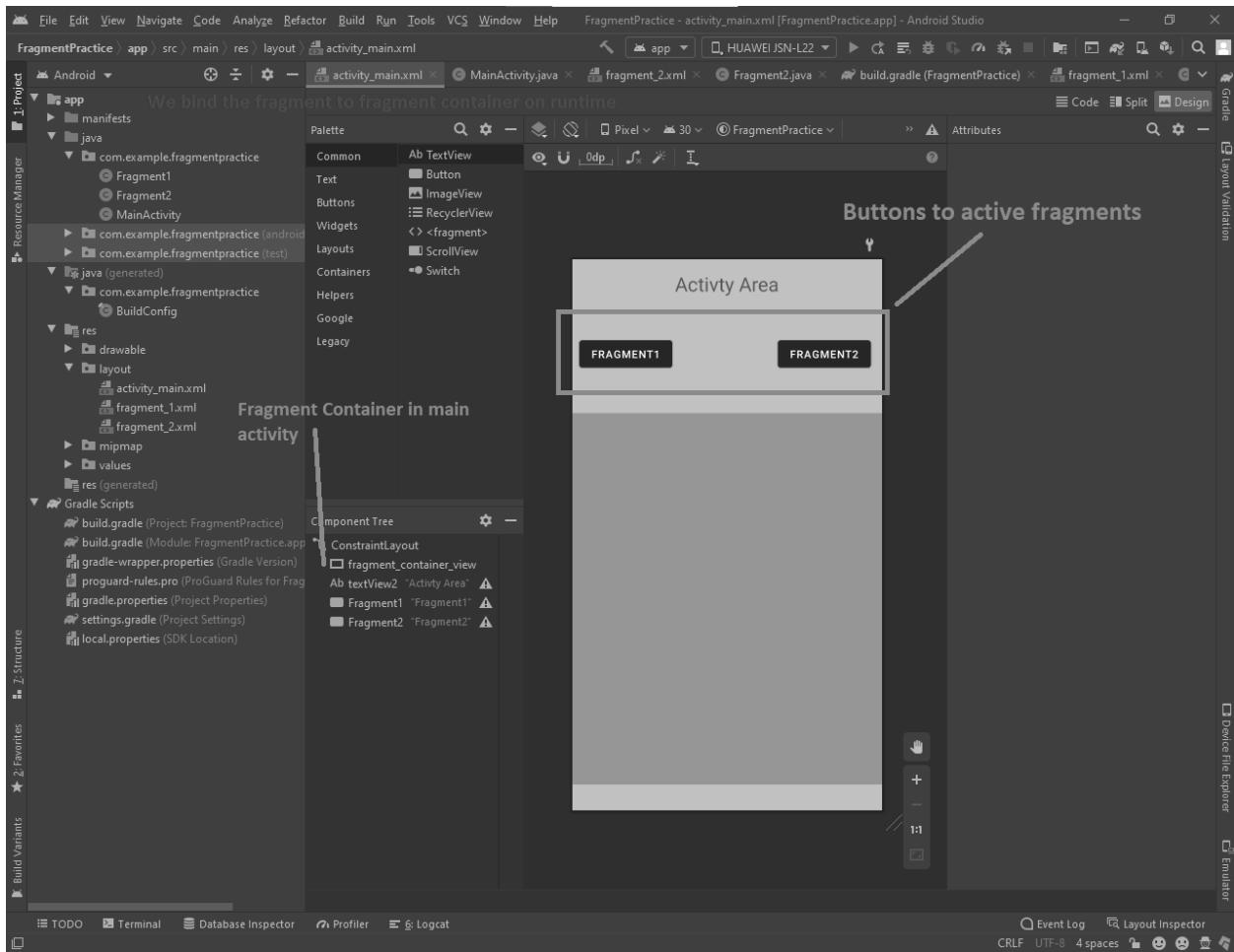
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"

    android:background="@color/purple_200">
<ImageView
    android:layout_width="125dp"
    android:layout_height="125dp"
    android:background="@color/teal_200"
    android:src="@drawable/local_library" />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Library"
    android:textSize="30sp" />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Read to explore more!"
    android:textColor="@color/white"
    android:background="@color/black"
    android:textSize="20sp" />
</LinearLayout>
```

Fragments

A Fragment represents a reusable portion of your app's UI. A fragment defines and manages its own layout, has its own lifecycle, and can handle its own input events. Fragments cannot live on their own--they must be hosted by an activity or another fragment.

Application



MainActivity with a fragment container (replace on button click)

The screenshot shows the Android Studio interface with the code editor open to `MainActivity.java`. The code implements a fragment container pattern where fragments are replaced based on button clicks.

```
1 package com.example.fragmentpractice;
2
3 import ...
4
5 public class MainActivity extends AppCompatActivity {
6
7     Button Fragment1Button, Fragment2Button;
8
9     @Override
10    protected void onCreate(Bundle savedInstanceState) {
11        super.onCreate(savedInstanceState);
12        setContentView(R.layout.activity_main);
13
14        Fragment1Button = findViewById(R.id.Fragment1);
15        Fragment2Button = findViewById(R.id.Fragment2);
16
17
18        Fragment1Button.setOnClickListener(new View.OnClickListener(){
19            @Override
20            public void onClick(View v) {
21                getSupportFragmentManager().beginTransaction()
22                    .setReorderingAllowed(true)
23                    .replace(R.id.fragment_container_view, Fragment1.class, null)
24                    .commit();
25            }
26        });
27
28        Fragment2Button.setOnClickListener(new View.OnClickListener(){
29            @Override
30            public void onClick(View v) {
31                getSupportFragmentManager().beginTransaction()
32                    .setReorderingAllowed(true)
33                    .replace(R.id.fragment_container_view, Fragment2.class, null)
34                    .commit();
35            }
36        });
37    }
38
39
40}
```

The code defines two buttons, `Fragment1Button` and `Fragment2Button`, which are assigned to their respective views in the XML layout. In the `onCreate` method, the `setOnClickListener` methods are set up for each button. When a button is clicked, a transaction is started using `getSupportFragmentManager()`. The `replace` method is used to replace the current fragment with either `Fragment1` or `Fragment2`, depending on which button was pressed. The `setReorderingAllowed(true)` call ensures that the fragments can be reordered.

Fragment – 1



Fragment - 2



