BHARATI VIDYAPEETH (DEEMED TO BE UNIVERSITY)

COLLEGE OF ENGINEERING, PUNE

Lab Manual

Mobile Computing

(Course 2014)

DEPARTMENT OF COMPUTER ENGINEERING

VISION OF THE INSTITUTE

"To be World Class Institute for Social Transformation through Dynamic Education"

MISSION OF THE INSTITUTE

- To provide quality technical education with advanced equipment, qualified faculty members, infrastructure to meet needs of profession and society.
- To provide an environment conductive to innovation, creativity, research and entrepreneurial leadership.
- To practice and promote professional ethics, transparency and accountability for social community, economic and environmental conditions.

VISION OF THE DEPARTMENT

"To pursue and excel in the endeavour for creating globally recognized computer engineers through quality education".

MISSION OF THE DEPARTMENT

1. To impart fundamental knowledge and strong skills conforming to a dynamic curriculum leading to developing professional competencies and continuing intellectual growth.

2. To develop professional, entrepreneurial and research competencies encompassing continuing intellectual growth.

3. To strive for producing qualified graduates possessing proficient abilities and ethical responsibilities adapting to the demand of working environment.

PROGRAM EDUCATIONAL OBJECTIVES

Graduates upon completion of B. Tech Computer Engineering Programme will able to:

1. Demonstrate technical and professional competencies by applying engineering fundamentals, computing principles and technologies.
2. Learn, Practice, and grow as skilled professionals/ entrepreneur/researchers adapting to the evolving computing landscape.
3. Demonstrate professional attitude, ethics, understanding of social context and interpersonal skills leading to a successful career.

PROGRAM OUTCOMES

1. To apply knowledge of computing and mathematics appropriate to the domain.
2. To logically define, analyse and solve real world problems.
3. To apply design principles in developing hardware/software systems of varying complexity that meet the specified needs.
4. To interpret and analyse data for providing solutions to complex engineering problems.
5. To use and practise engineering and IT tools for professional growth.
6. To understand and respond to legal and ethical issues involving the use of technology for societal benefits.
7. To develop societal relevant projects using available resources.
8. To exhibit professional and ethical responsibilities.
9. To work effectively as an individual and a team member within the professional environment.
10. To prepare and present technical documents using effective communication skills.
11. To demonstrate effective leadership skills throughout the project management life cycle.
12. To understand the significance of lifelong learning for professional development.

GENRAL INSTUCTIONS:

- Equipment in the lab is meant for the use of students. Students need to maintain a proper decorum in the computer lab. Students must use the equipment with care.
- Students are required to carry their reference materials, files and records with completed assignment while entering the lab.
- Students are supposed to occupy the systems allotted to them and are not supposed to talk or make noise in the lab.
- All the students should perform the given assignment individually.
- Lab can be used in free time/lunch hours by the students who need to use the systems should take prior permission from the lab in-charge.
- All the Students are instructed to carry their identity cards when entering the lab.
- Lab files need to be submitted on or before date of submission.
- Students are not supposed to use pen drives, compact drives or any other storage devices in the lab.
- For Laboratory related updates and assignments students should refer to the notice board in the Lab.

COURSE NAME: MOBILE COMPUTING

WEEKLY PLAN:

Week No.	Problem Definition	Practical/Assignment Name
1	To develop a Simple Android Application that uses GUI components, Font and Colors.	Developing a Simple Android Application that uses GUI components, Font and Colors.
2	To develop a Simple Android Application that uses Layout Managers and Event Listeners.	Developing a Simple Android Application that uses Layout Managers and Event Listeners.
3	To develop a Simple Android Application for Native Calculator.	Developing a Simple Android Application for Native Calculator.
4	To Develop android application for spinning bottle game.	Developing android application for spinning bottle game.
5	To develop Android Application for Rolling Dice Game	Developing Android Application for Rolling Dice Game.
6	To develop android application for Login Page.	Developing android application for Login Page.
7	To develop android application for web application.	Developing android application for web application.
8	To develop a Simple Android Application that draws basic Graphical Primitives on the screen.	Developing a Simple Android Application that draws basic Graphical Primitives on the screen.
9	To develop a Simple Android Application that makes use of Database.	Developing a Simple Android Application that makes use of Database.
10	To develop a Android Application that creates Alarm Clock.	Developing a Android Application that creates Alarm Clock.
11	To develop a Android Application that creates an alert upon receiving a message.	Developing a Android Application that creates an alert upon

EXAMINATION SCHEME

Practical Exam: 25 Marks
Term Work: 25 Marks
Total: 50 Marks
Minimum Marks required: 20 Marks

PROCEDURE OF EVALUATION

Each practical/assignment shall be assessed continuously on the scale of 25 marks. The distribution of marks as follows.

Sr. No	Evaluation Criteria	Marks for each Criteria	Rubrics
1	**Timely Submission**	**07**	➢ Punctuality reflects the work ethics. Students should reflect that work ethics by completing the lab assignments and reports in a timely manner without being reminded or warned.
2	**Presentation**	**06**	➢ Students are expected to write the technical document (lab report) in their own words. The presentation of the contents in the lab report should be complete, unambiguous, clear, and understandable. The report should document Approach/algorithm/design and code with proper explanation.
3	**Understanding**	**12**	➢ Correctness and Robustness of the code is expected. The Learners should have an in-depth knowledge of the practical assignment performed. The learner should be able to explain methodology used for designing and developing the program/solution. He/she should clearly understand the purpose of the assignment and its outcome.

LABORATORY USAGE

Students use Mobile Development Lab to perform Mobile Computing practical. This Lab is equipped with following hardware configuration: Intel Core i5 Processor CPU, Installed Memory: 4GB, HDD: 500GB and Software Configuration: OS-Windows 8.1, Tools: Android Studio, Turbo C,TASM, CISCO Packet Tracer, JDK, Rational Rose, Netbeans for executing the lab experiments, document the results and to prepare the technical documents for making the lab reports.

OBJECTIVE

The objective of this lab is to make students aware and practice implementation of various mobile applications and projects through Android Studio.

PRACTICAL PRE-REQUISITE

- Java Programming Language
- Visual Studio
- Network and Internet

SOFTWARE REQUIREMENTS

- Android Studio

COURSE OUTCOMES

1. Explain the basic of mobile telecommunication system
2. Choose the required functionality at each layer
3. Application identity solution for each functionality at each layer
4. Use similar tool and design Ad- hoc networks
5. Develop mobile application

HOW OUTCOMES ARE ASSESSED?

Outcome	Assignment Number	Level	Proficiency evaluated by
1. Explain the basic of mobile telecommunication system	**1,2,3,4,5,9,10,11**	**3,3,3,3,3,2,3,2**	Developed applications and executed programming codes
2. Choose the required functionality at each layer	**1,2,3,4,5,6,7**	**3,3,2,3,3,2,2**	Developed applications and executed programming codes
3. Application identity solution for each functionality at each layer	**3,4,5,6,7,8,9**	**3,3,3,2,3,2,3**	Developed applications and executed programming codes
4. Use similar tool and design Ad-hoc networks	**1,2,4,6,7**	**3,3,3,3,3**	Developed applications and executed programming codes
5. Develop mobile application	**1,2,3,4,5,6,7,8,9,10,11**	**3,3,3,3,3,3,3,3,3,3,3**	Developed applications and executed programming codes

DESIGN EXPERIENCE GAINED

Students will gain the knowledge about Mobile Application development. Before starting with Mobile Development part, student should follow the learned principles required to develop application and execute the code.

LEARNING EXPERIENCE GAINED

Students will learn to develop an android application. Application can be considered as Project also. So at the end of semester, students can develop their own project also based on Android operating system.

LIST OF PRACTICAL ASSIGNMENTS:

1. To develop a Simple Android Application that uses GUI components, Font and Colors.
2. To develop a Simple Android Application that uses Layout Managers and Event Listeners.
3. To develop a Simple Android Application for Native Calculator.
4. Develop android application for spinning bottle game.
5. Android Studio Code to Develop Rolling Dice Game.
6. Develop android application for Login Page.
7. Develop android application for web application.
8. To develop a Simple Android Application that draws basic Graphical Primitives on the screen.
9. To develop a Simple Android Application that makes use of Database.
10. To develop a Android Application that creates Alarm Clock.
11. To develop a Android Application that creates an alert upon receiving a message.

EXPERIMENT 01

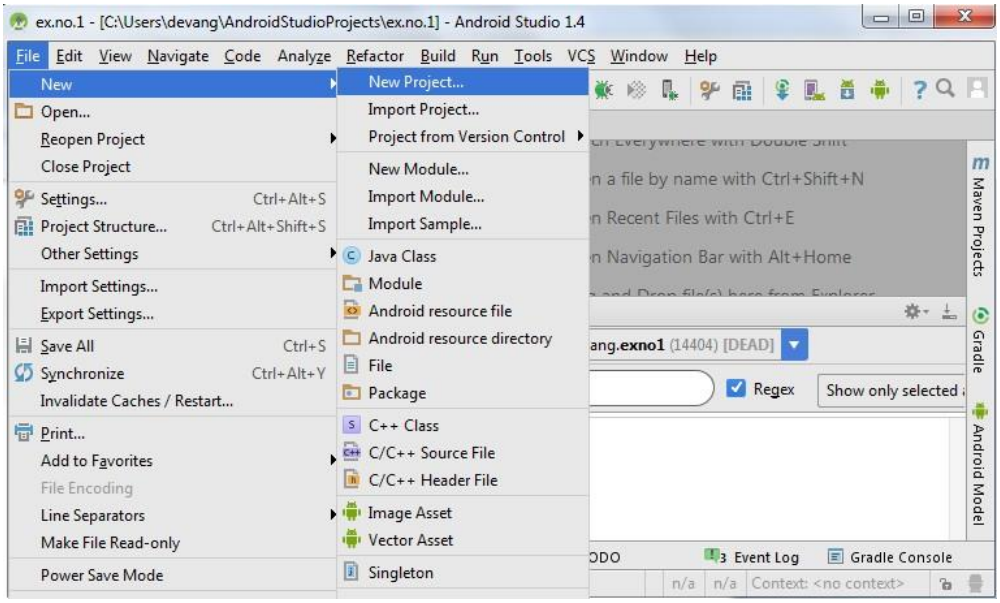
Aim:

To develop a Simple Android Application that uses GUI components, Font and Colors.

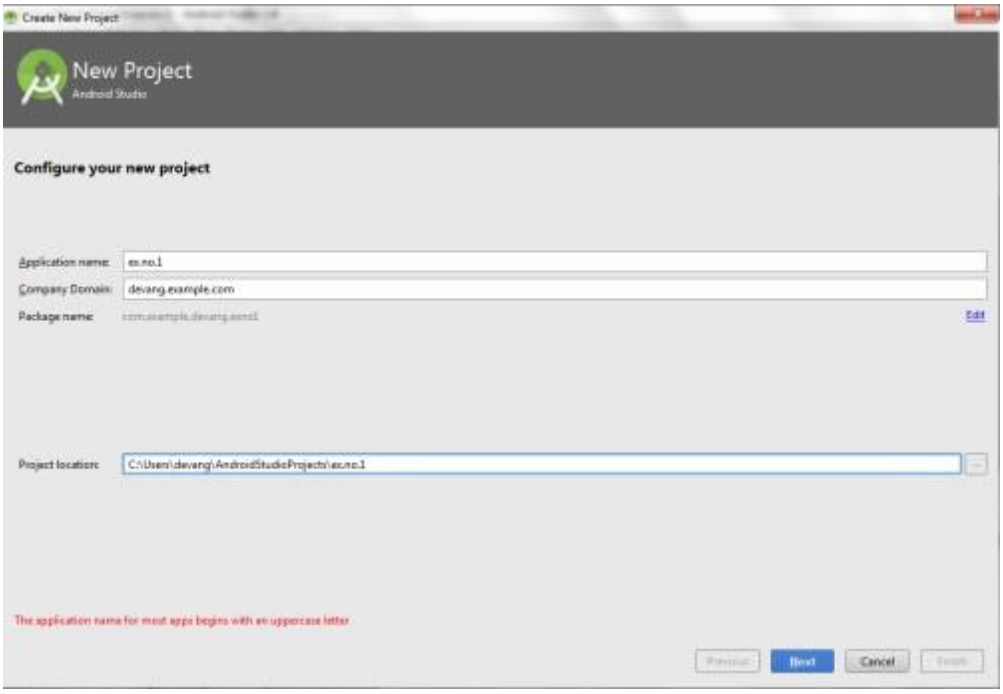
Procedure:

Creating a New project:

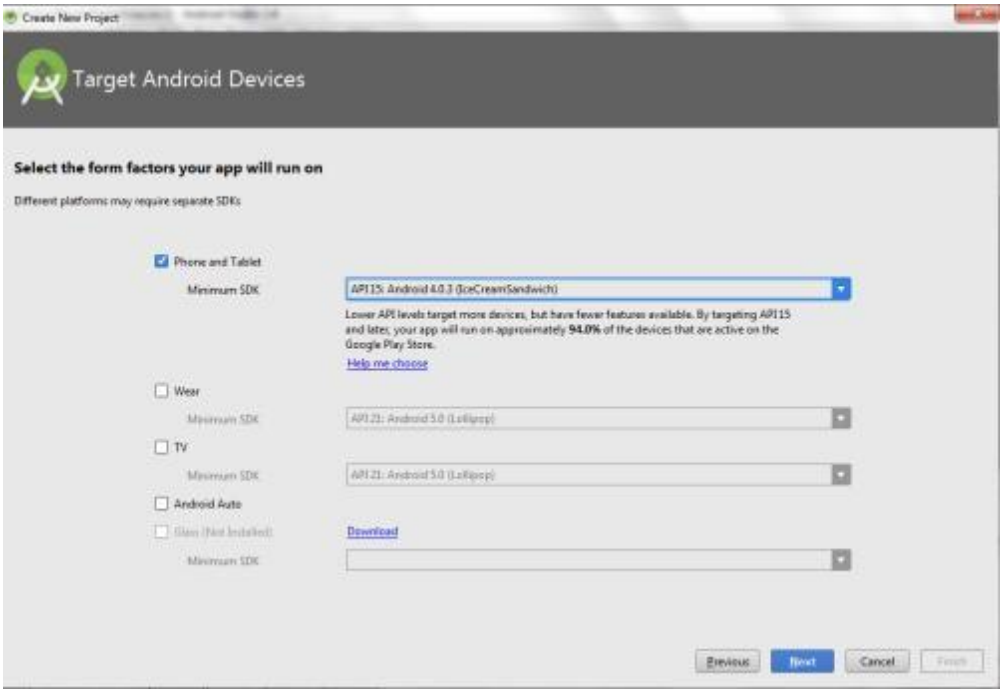
- Open Android Stdio and then click on **File -> New -> New project.**



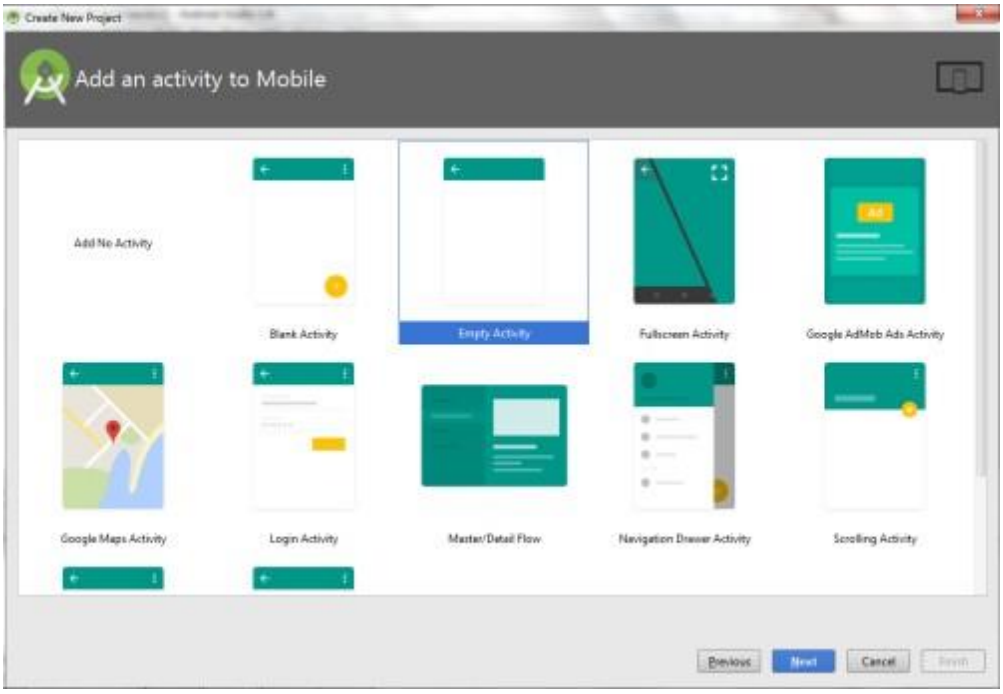
- Then type the Application name as "**ex.no.1"** and click **Next.**



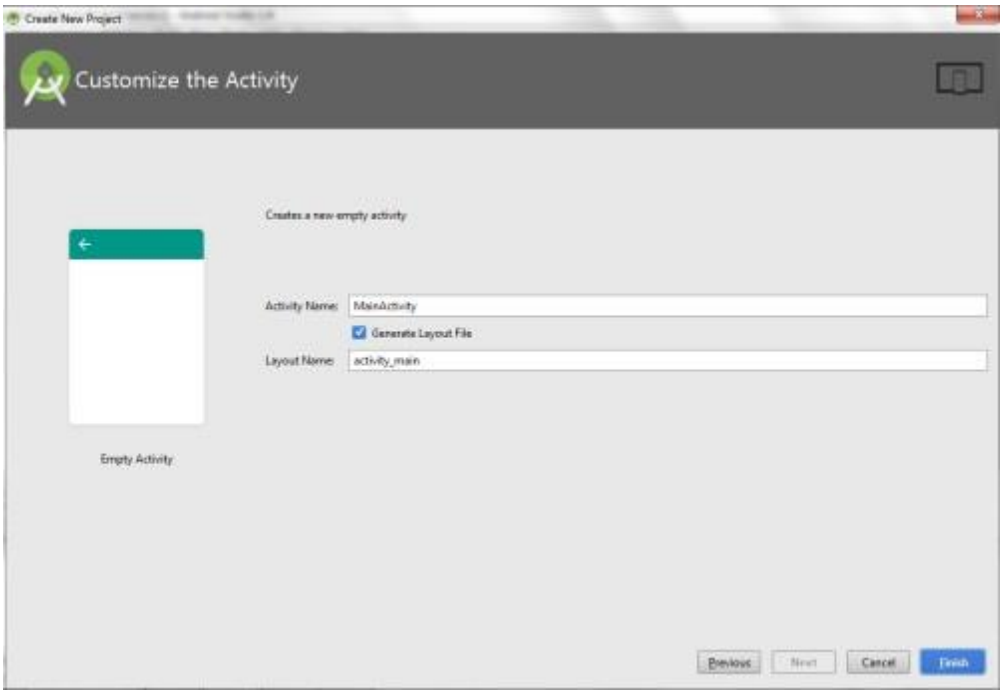
- Then select the **Minimum SDK** as shown below and click **Next**.



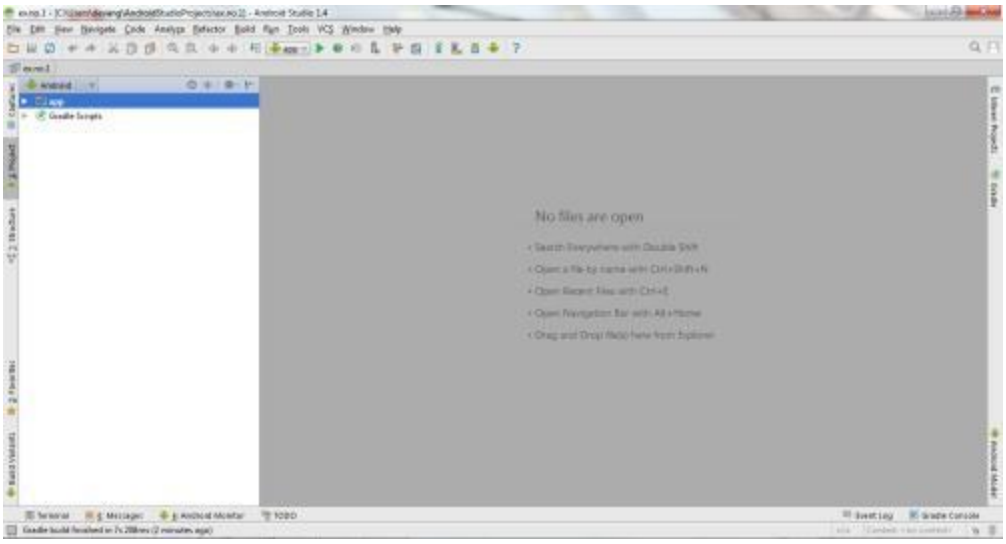
- Then select the **Empty Activity** and click **Next.**



- Finally click F**inish**.

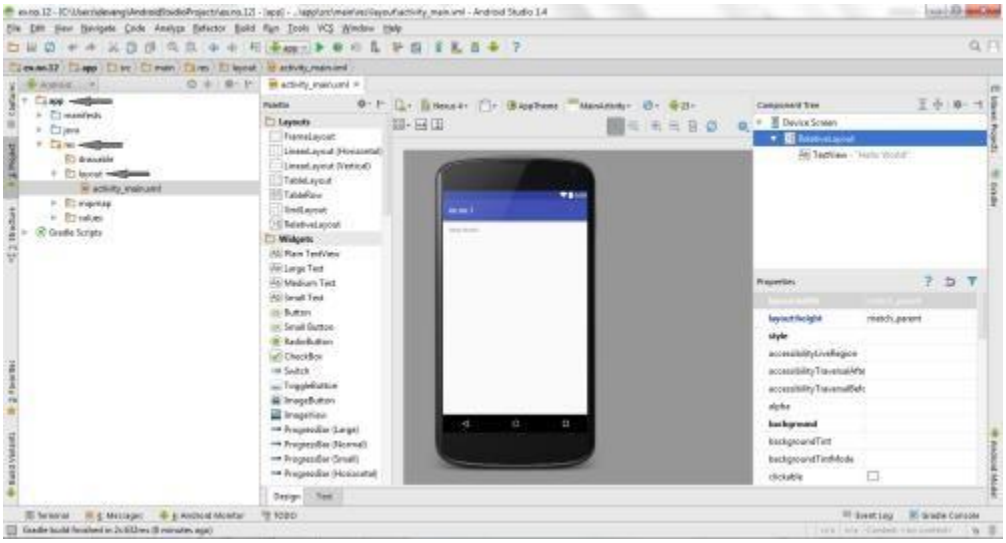


- It will take some time to build and load the project.
- After completion it will look as given below.

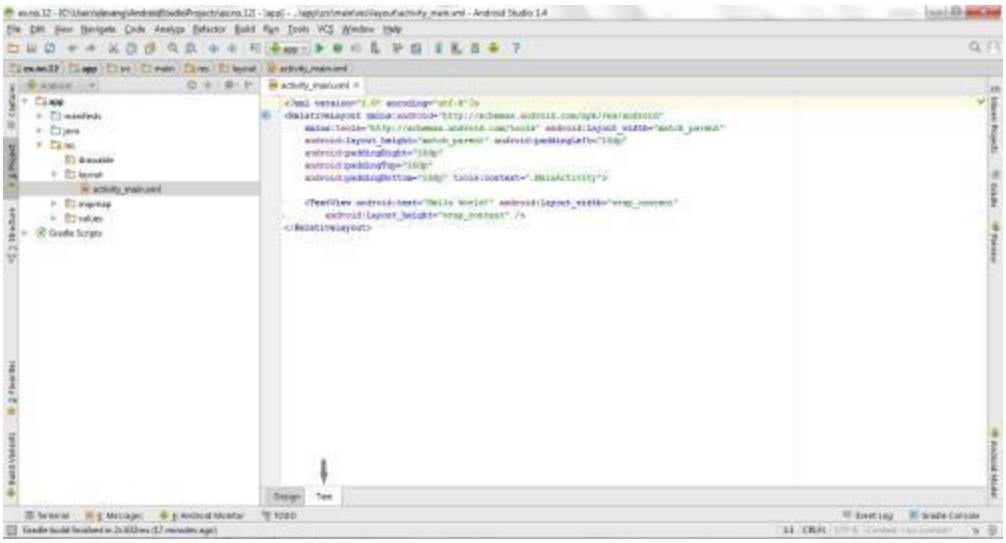


Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml.**



- Now click on **Text** as shown below.



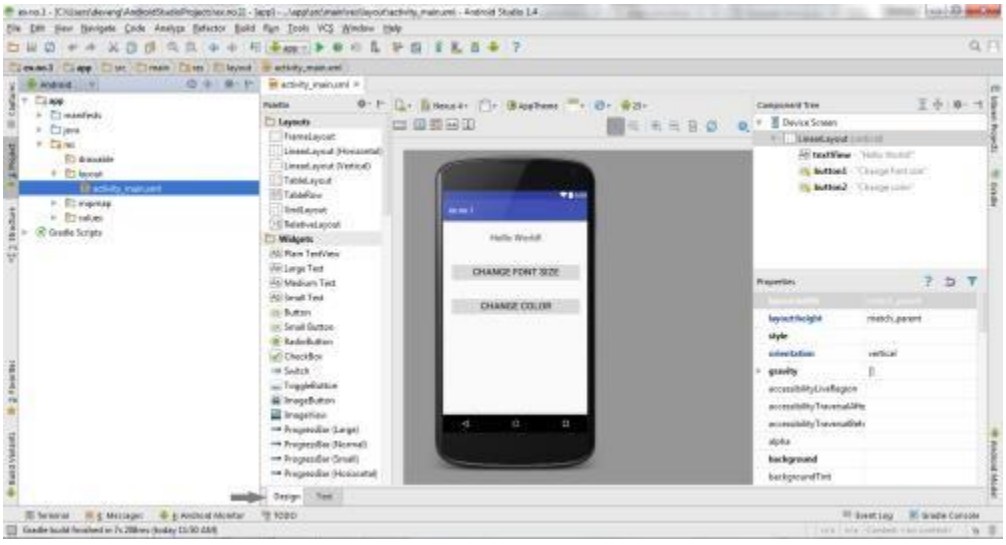
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

```
<?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">

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="30dp"
        android:gravity="center"
        android:text="Hello World!"
        android:textSize="25sp"
        android:textStyle="bold" />
```

```
<Button
    android:id="@+id/button1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="20dp"
    android:gravity="center"
    android:text="Change font size"
    android:textSize="25sp" />
<Button
    android:id="@+id/button2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="20dp"
    android:gravity="center"
    android:text="Change color"
    android:textSize="25sp" />
</LinearLayout>
```

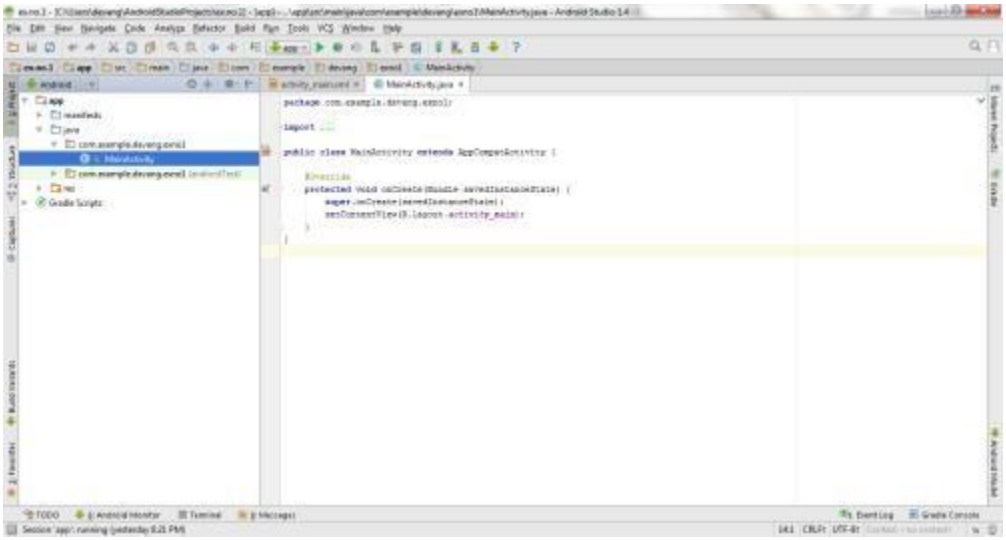
- Now click on Design and your application will look as given below.



- So now the designing part is completed.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno1 -> MainActivity.**



- Then delete the code which is there and type the code as given below.

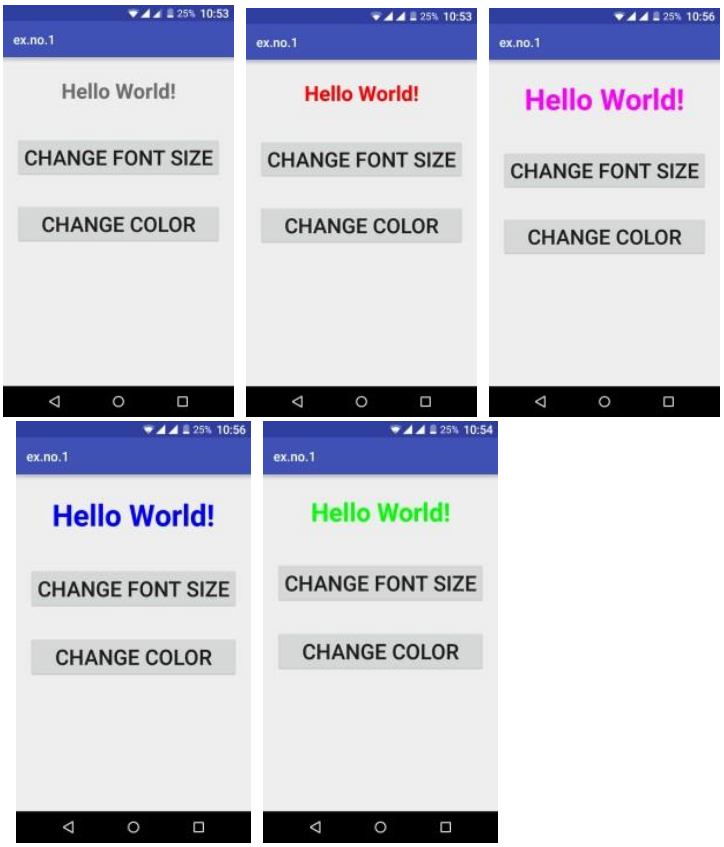
Code for MainActivity.java:

```
package com.example.exno1;

import android.graphics.Color;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity
{
   int ch=1;
   float font=30;
   @Override
   protected void onCreate(Bundle savedInstanceState)
   {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      final TextView t= (TextView) findViewById(R.id.textView);
      Button b1= (Button) findViewById(R.id.button1);
      b1.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            t.setTextSize(font);
```

```
                font = font + 5;
                if (font == 50)
                    font = 30;
            }
        });
        Button b2= (Button) findViewById(R.id.button2);
        b2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                switch (ch) {
                    case 1:
                        t.setTextColor(Color.RED);
                        break;
                    case 2:
                        t.setTextColor(Color.GREEN);
                        break;
                    case 3:
                        t.setTextColor(Color.BLUE);
                        break;
                    case 4:
                        t.setTextColor(Color.CYAN);
                        break;
                    case 5:
                        t.setTextColor(Color.YELLOW);
                        break;
                    case 6:
                        t.setTextColor(Color.MAGENTA);
                        break;
                }
                ch++;
                if (ch == 7)
                    ch = 1;
            }
        });
    }
}
```

s

- So now the Coding part is also completed.
- Now run the application to see the output.
- Output:

- 

- **Result:**
- Thus a Simple Android Application that uses GUI components, Font and Colors is developed and executed successfully.

EXPERIMENT NO 02

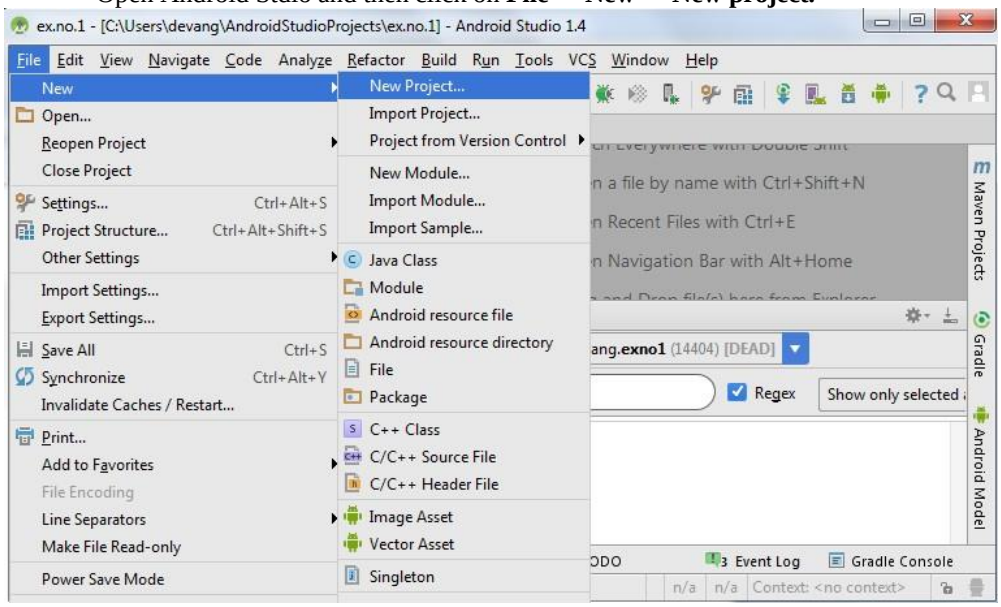
Aim:

To develop a Simple Android Application that uses Layout Managers and Event Listeners.

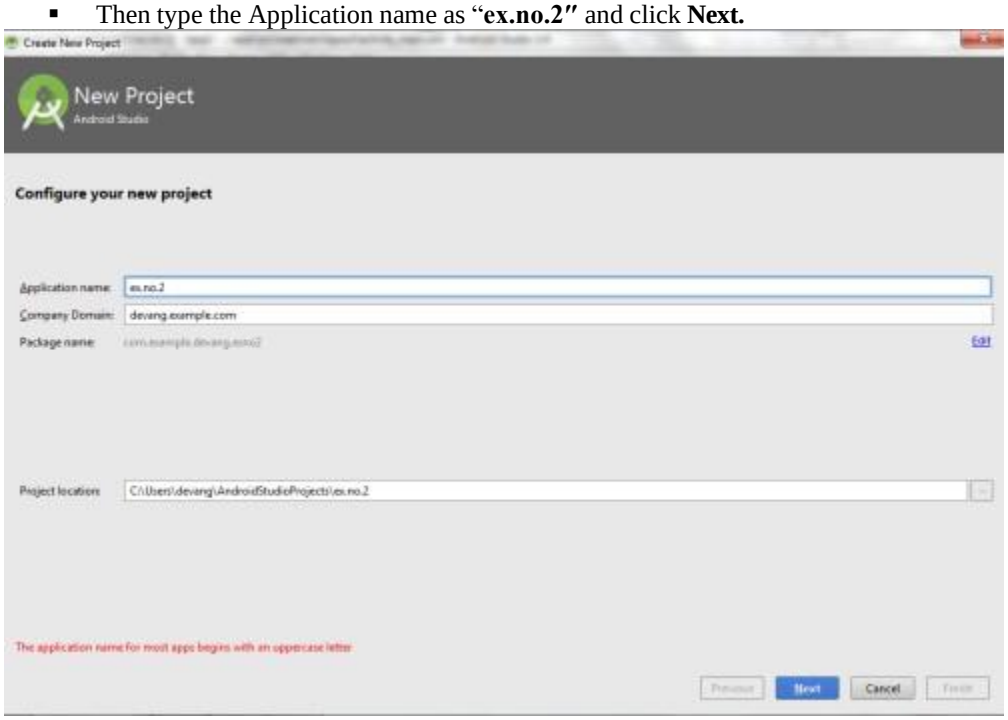
Procedure:

Creating a New project:

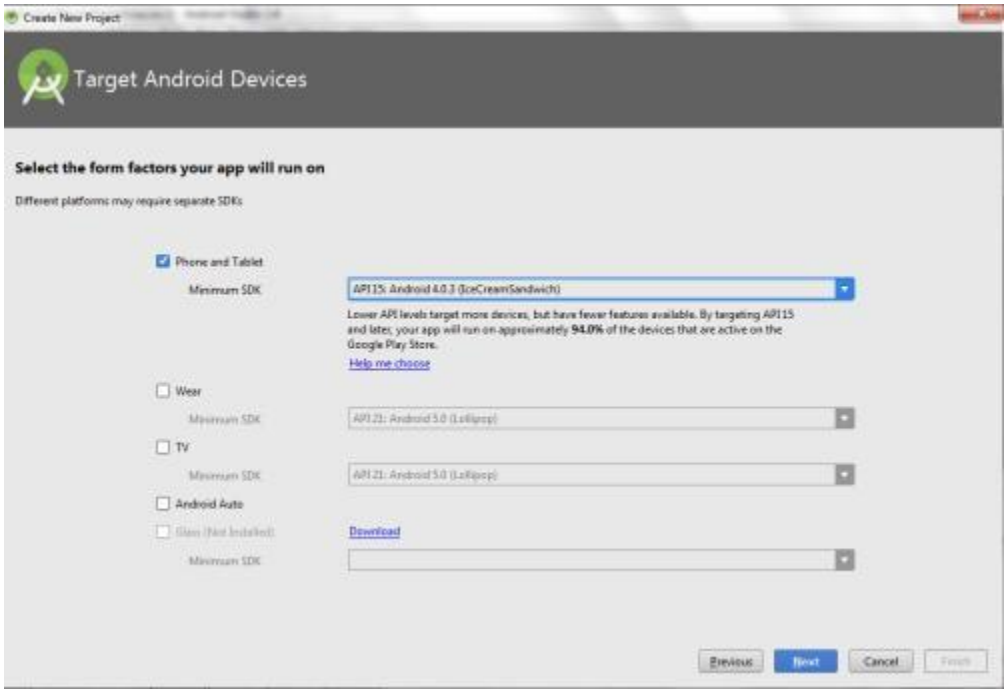
- Open Android Stdio and then click on **File -> New -> New project.**



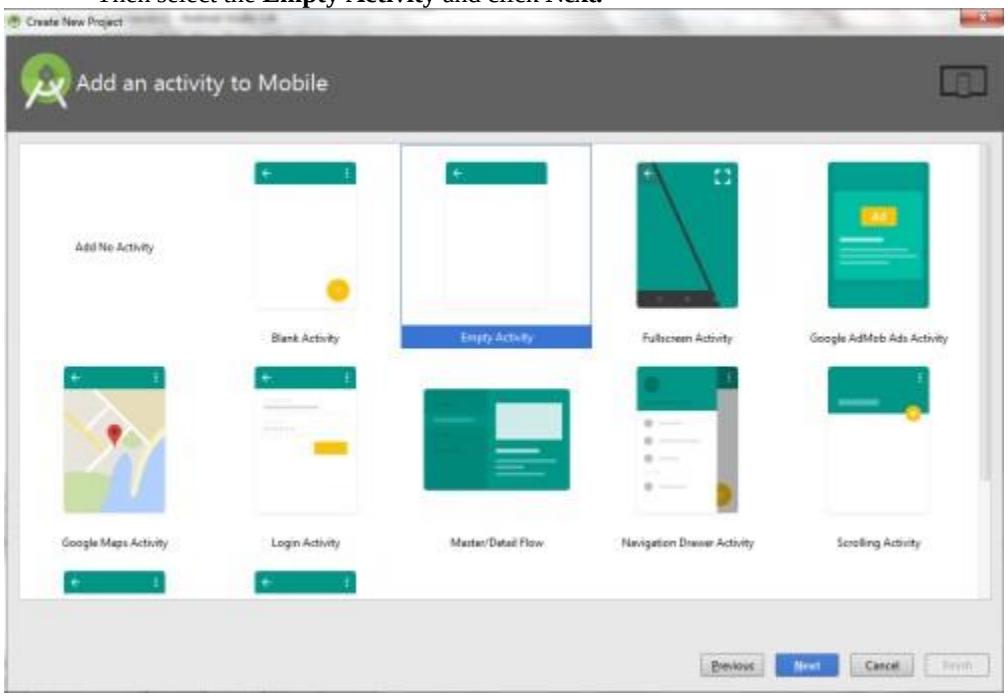
- Then type the Application name as "**ex.no.2"** and click **Next.**



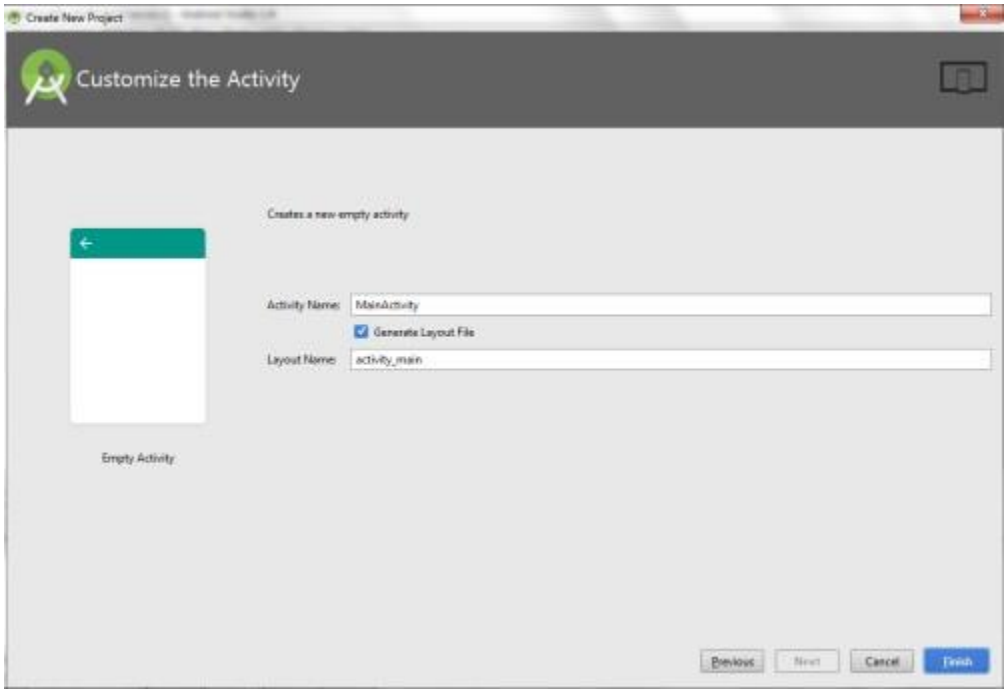
- Then select the **Minimum SDK** as shown below and click **Next**.



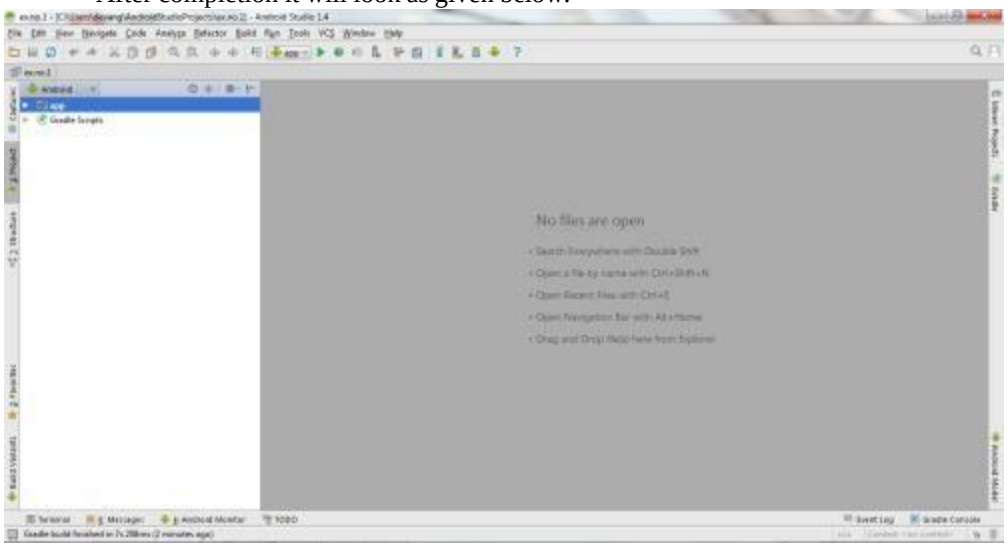
- Then select the **Empty Activity** and click **Next.**



- Finally click F**inish**.

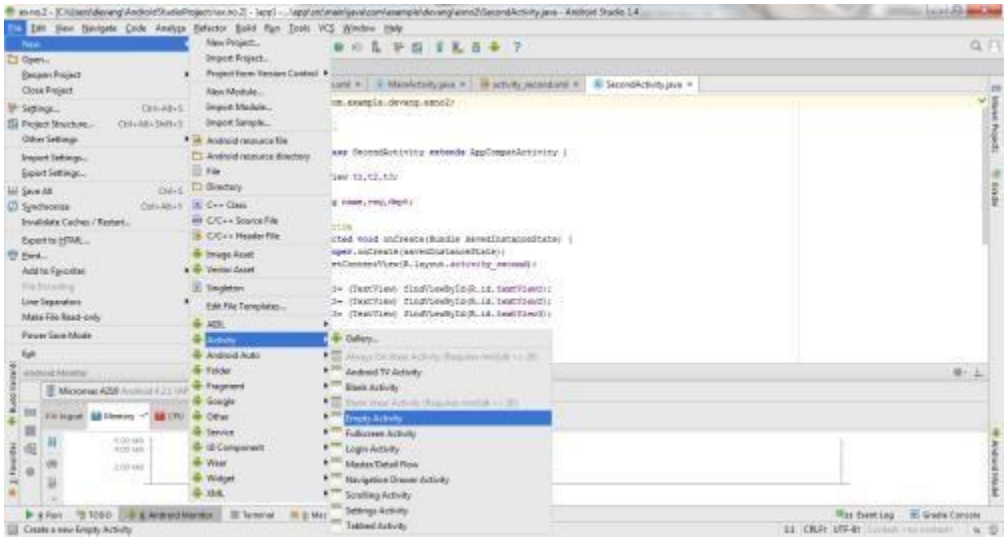


- It will take some time to build and load the project.
- After completion it will look as given below.

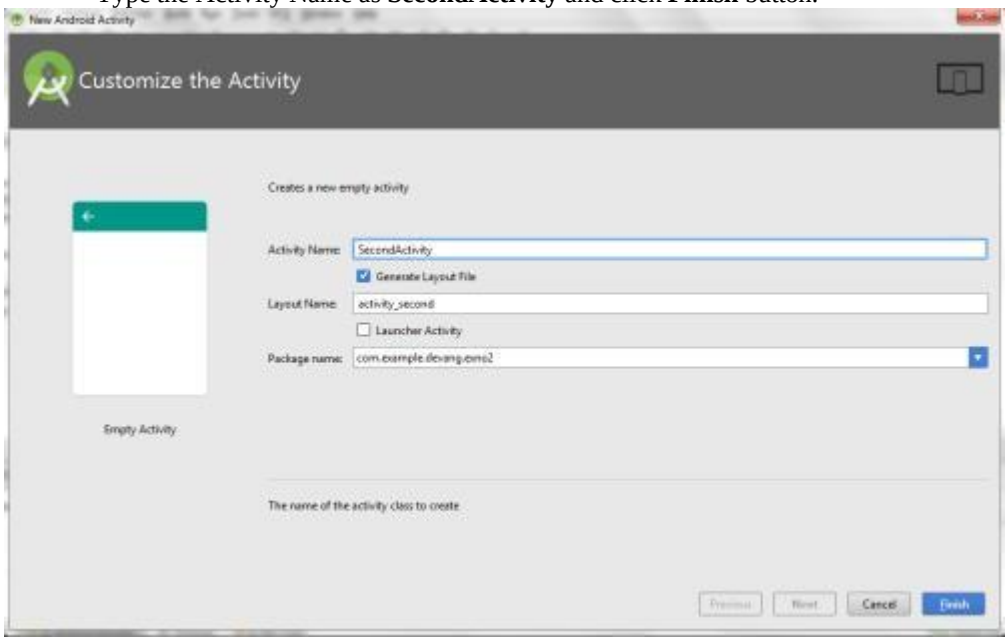


Creating Second Activity for the Android Application:

- Click on **File -> New -> Activity -> Empty Activity.**



- Type the Activity Name as **SecondActivity** and click **Finish** button.

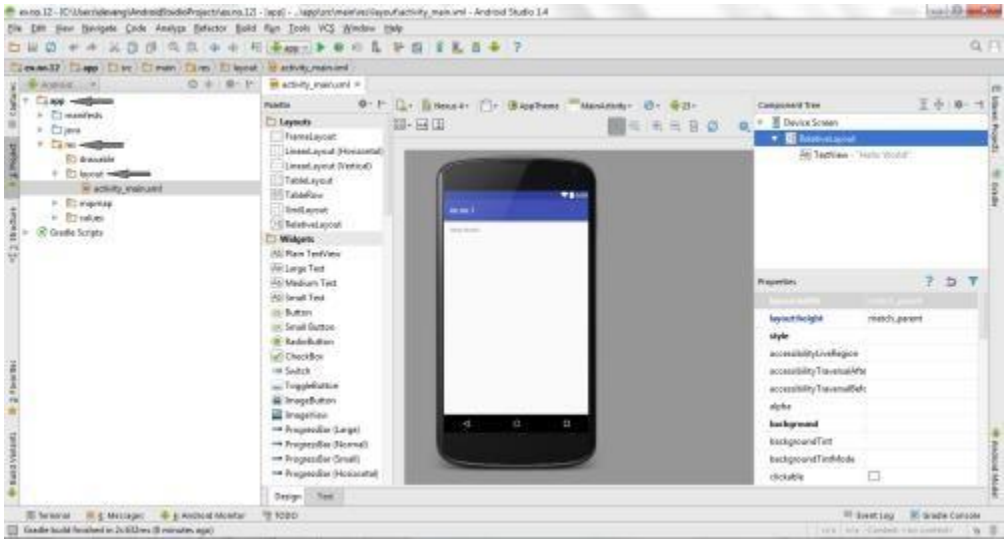


- Thus Second Activity For the application is created.

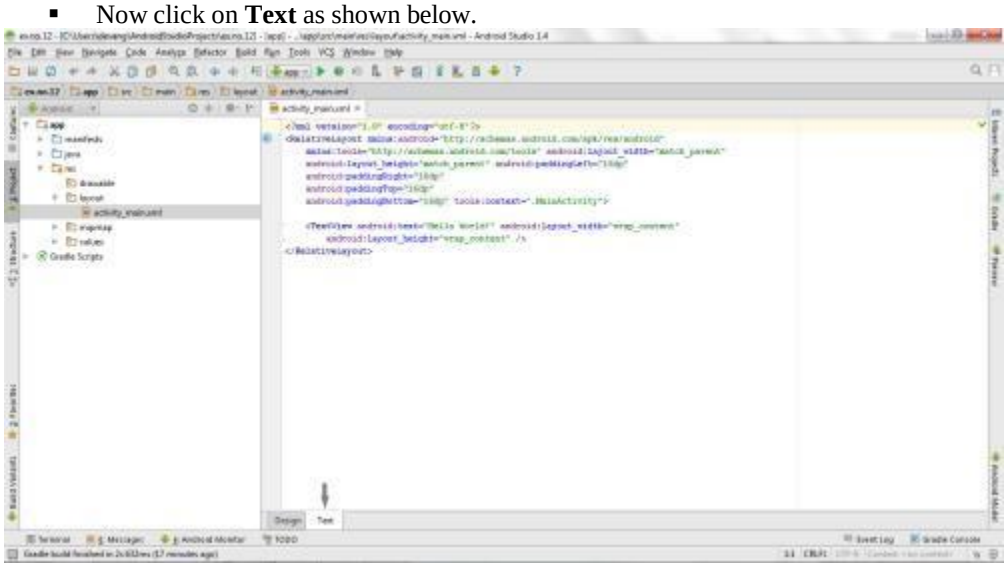
Designing layout for the Android Application:

Designing Layout for Main Activity:

- Click on **app -> res -> layout -> activity_main.xml.**



- Now click on **Text** as shown below.



- Then delete the code which is there and type the code as given below.

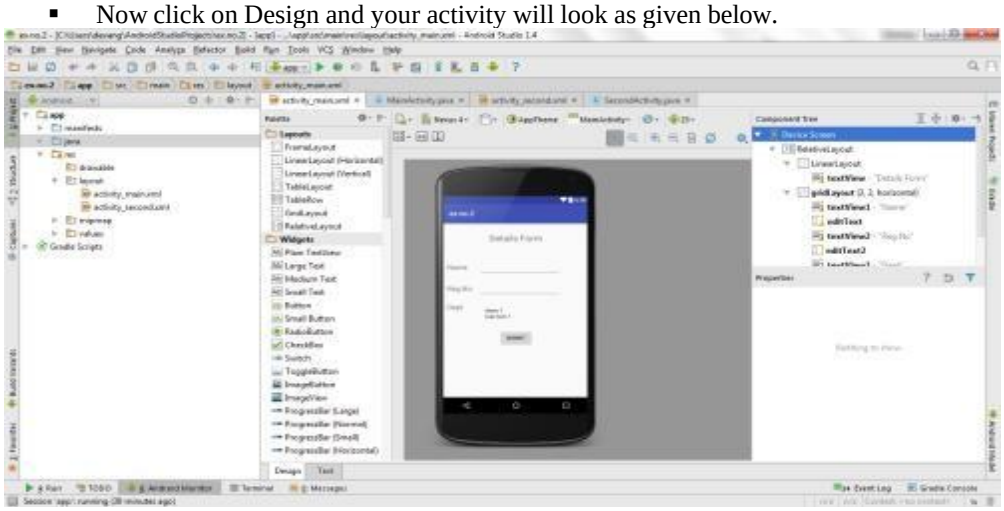
Code for Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="100dp">
        <TextView
            android:id="@+id/textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_margin="30dp"
            android:text="Details Form"
            android:textSize="25sp"
```

```
        android:gravity="center"/>
</LinearLayout>

<GridLayout
    android:id="@+id/gridLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="100dp"
    android:layout_marginBottom="200dp"
    android:columnCount="2"
    android:rowCount="3">
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:layout_row="0"
        android:layout_column="0"
        android:text="Name"
        android:textSize="20sp"
        android:gravity="center"/>

    <EditText
        android:id="@+id/editText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:layout_row="0"
        android:layout_column="1"
        android:ems="10"/>

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:layout_row="1"
        android:layout_column="0"
        android:text="Reg.No"
        android:textSize="20sp"
        android:gravity="center"/>

    <EditText
        android:id="@+id/editText2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:layout_row="1"
        android:layout_column="1"
        android:inputType="number"
        android:ems="10"/>

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
```

```
            android:layout_height="wrap_content"
            android:layout_margin="10dp"
            android:layout_row="2"
            android:layout_column="0"
            android:text="Dept"
            android:textSize="20sp"
            android:gravity="center"/>

        <Spinner
            android:id="@+id/spinner"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="10dp"
            android:layout_row="2"
            android:layout_column="1"
            android:spinnerMode="dropdown"/>

    </GridLayout>

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerInParent="true"
        android:layout_marginBottom="150dp"
        android:text="Submit"/>

</RelativeLayout>
```

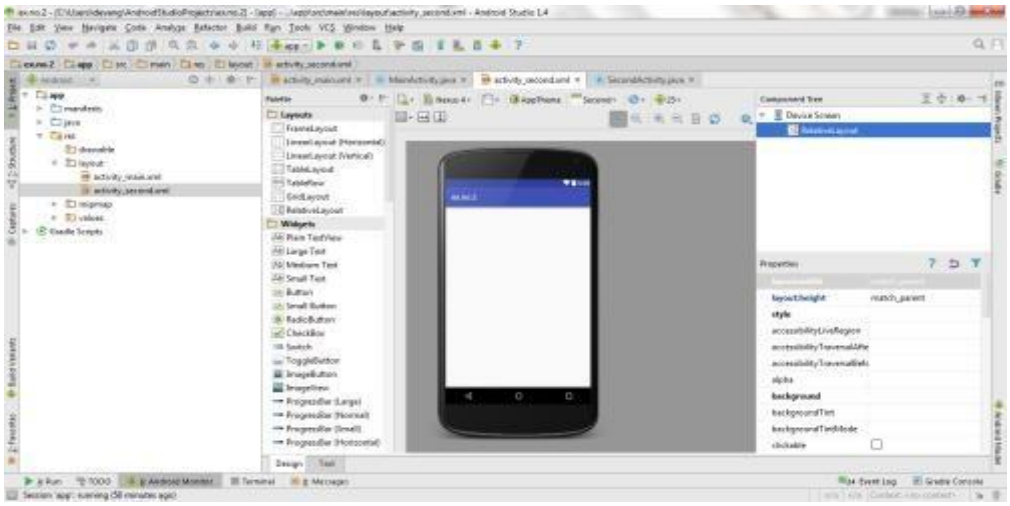
- Now click on Design and your activity will look as given below.



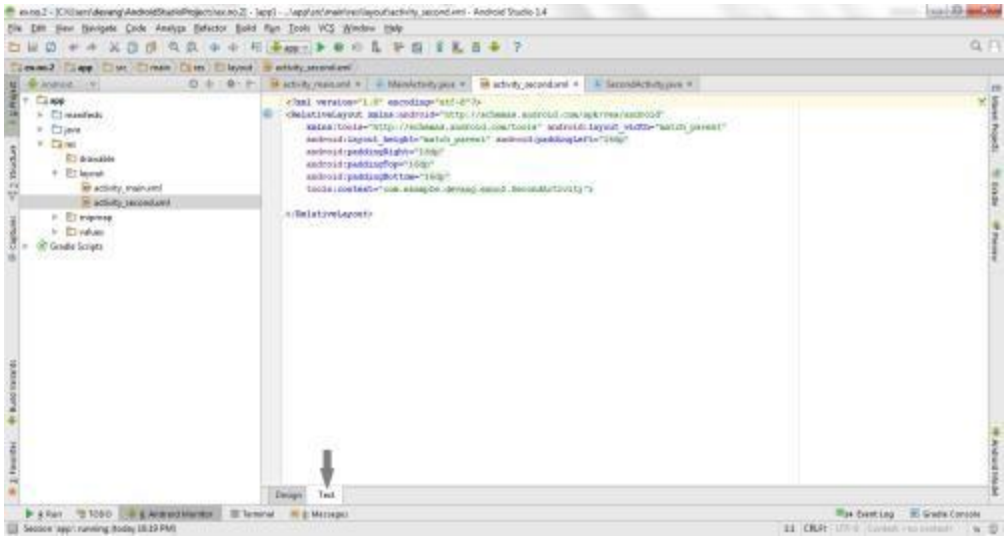
- So now the designing part of Main Activity is completed.

Designing Layout for Second Activity:

- Click on **app -> res -> layout -> activity_second.xml.**



- Now click on **Text** as shown below.



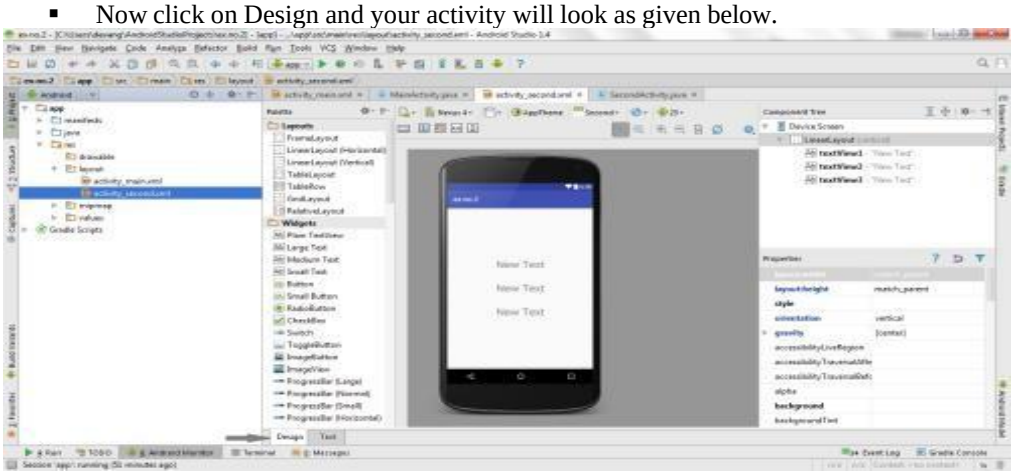
- Then delete the code which is there and type the code as given below.

Code for Activity_second.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.devang.exno2.SecondActivity"
    android:orientation="vertical"
    android:gravity="center">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="20dp"
        android:text="New Text"
        android:textSize="30sp"/>

    <TextView
        android:id="@+id/textView2"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="20dp"
        android:text="New Text"
        android:textSize="30sp"/>

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="20dp"
        android:text="New Text"
        android:textSize="30sp"/>

</LinearLayout>
```

- Now click on Design and your activity will look as given below.

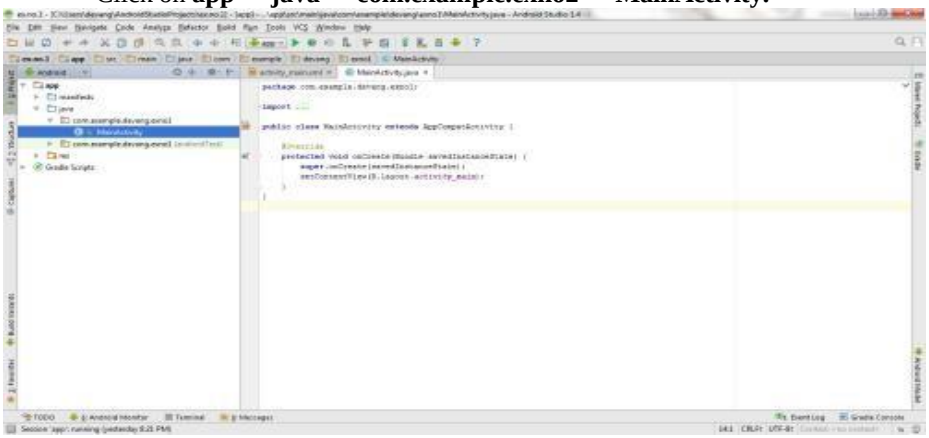


- So now the designing part of Second Activity is also completed.

Java Coding for the Android Application:

Java Coidng for Main Activity:

- Click on **app -> java -> com.example.exno2 -> MainActivity.**



- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exno2;
 import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
```

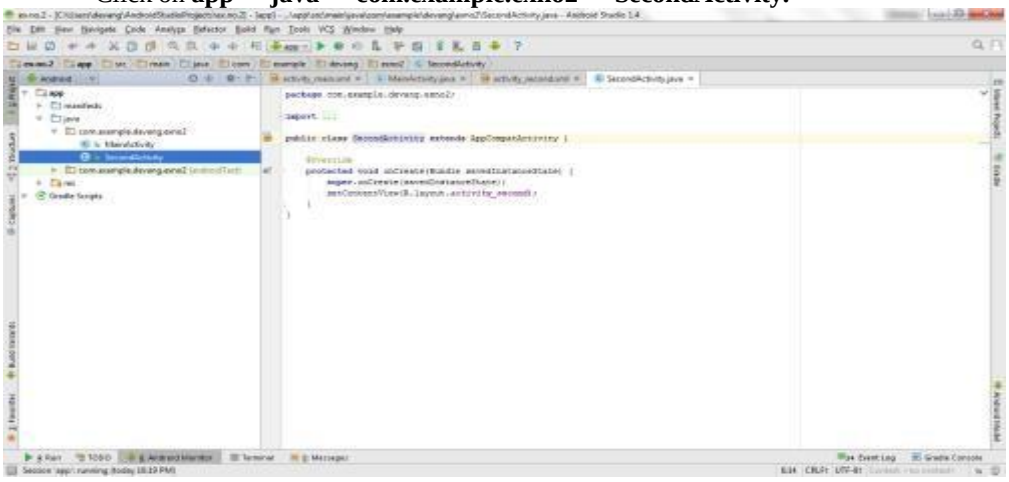
```
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;

public class MainActivity extends AppCompatActivity {

    //Defining the Views
    EditText e1,e2;
    Button bt;
    Spinner s;

    //Data for populating in Spinner
    String [] dept_array={"CSE","ECE","IT","Mech","Civil"};

    String name,reg,dept;

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

        //Referring the Views
        e1= (EditText) findViewById(R.id.editText);
        e2= (EditText) findViewById(R.id.editText2);

        bt= (Button) findViewById(R.id.button);

        s= (Spinner) findViewById(R.id.spinner);

        //Creating Adapter for Spinner for adapting the data from array to Spinner
        ArrayAdapter adapter= new ArrayAdapter(MainActivity.this,android.R.layout.simple_spinner_item,dept_array);
        s.setAdapter(adapter);

        //Creating Listener for Button
        bt.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                //Getting the Values from Views(Edittext & Spinner)
                name=e1.getText().toString();
                reg=e2.getText().toString();
                dept=s.getSelectedItem().toString();

                //Intent For Navigating to Second Activity
                Intent i = new Intent(MainActivity.this,SecondActivity.class);

                //For Passing the Values to Second Activity
                i.putExtra("name_key", name);
                i.putExtra("reg_key",reg);
                i.putExtra("dept_key", dept);

                startActivity(i);
```

```
        }
      });
    }
}
```

- So now the Coding part of Main Activity is completed.

Java Coding for Second Activity:

- Click on **app -> java -> com.example.exno2 -> SecondActivity.**



- Then delete the code which is there and type the code as given below.

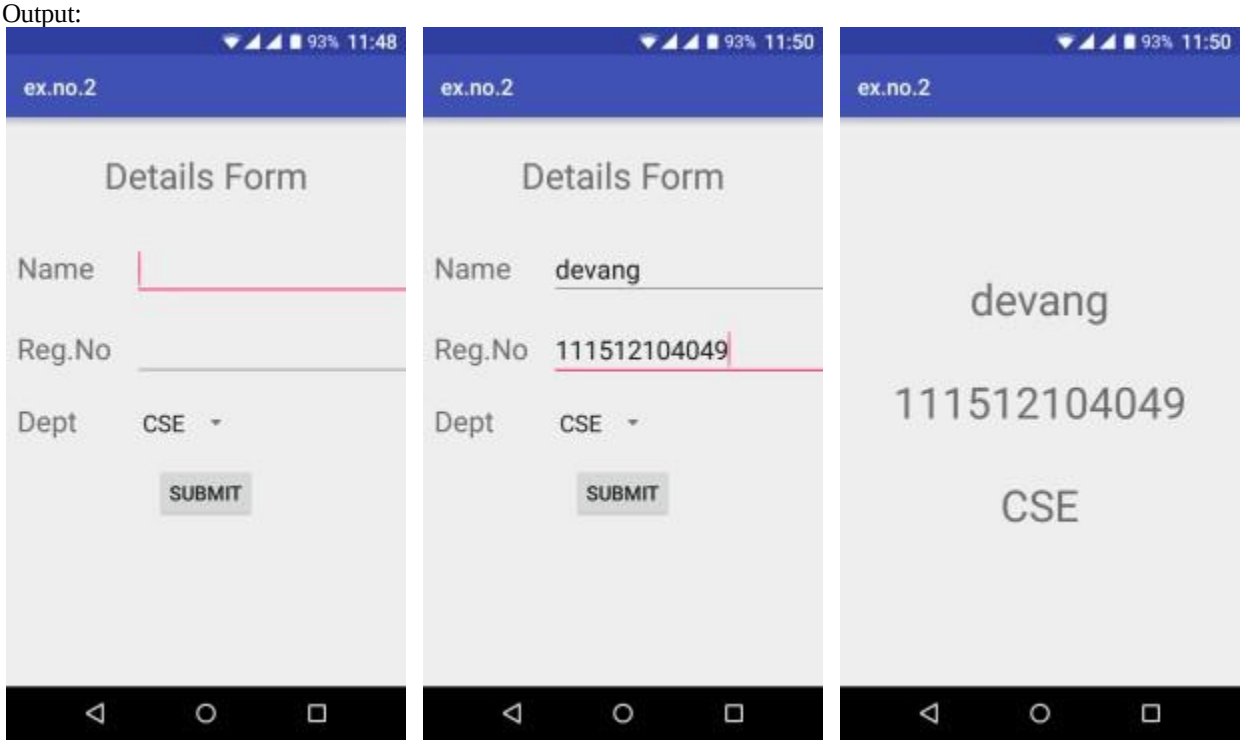
Code for SecondActivity.java:

```
package com.example.exno2;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;

public class SecondActivity extends AppCompatActivity {

    TextView t1,t2,t3;

    String name,reg,dept;

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

        t1= (TextView) findViewById(R.id.textView1);
        t2= (TextView) findViewById(R.id.textView2);
        t3= (TextView) findViewById(R.id.textView3);

        //Getting the Intent
        Intent i = getIntent();

        //Getting the Values from First Activity using the Intent received
        name=i.getStringExtra("name_key");
        reg=i.getStringExtra("reg_key");
        dept=i.getStringExtra("dept_key");
```

```
            //Setting the Values to Intent
            t1.setText(name);
            t2.setText(reg);
            t3.setText(dept);

        }
    }
```

- So now the Coding part of Second Activity is also completed.
- Now run the application to see the output.

Output:



Result:

Thus a Simple Android Application that uses Layout Managers and Event Listeners is developed and executed successfully.

EXPERIMENT 03

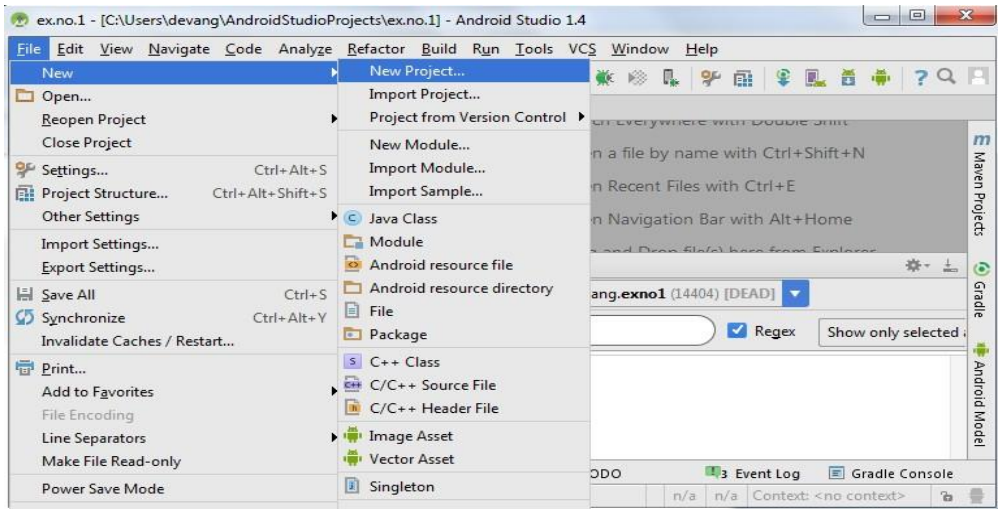
Aim:

To develop a Simple Android Application for Native Calculator.

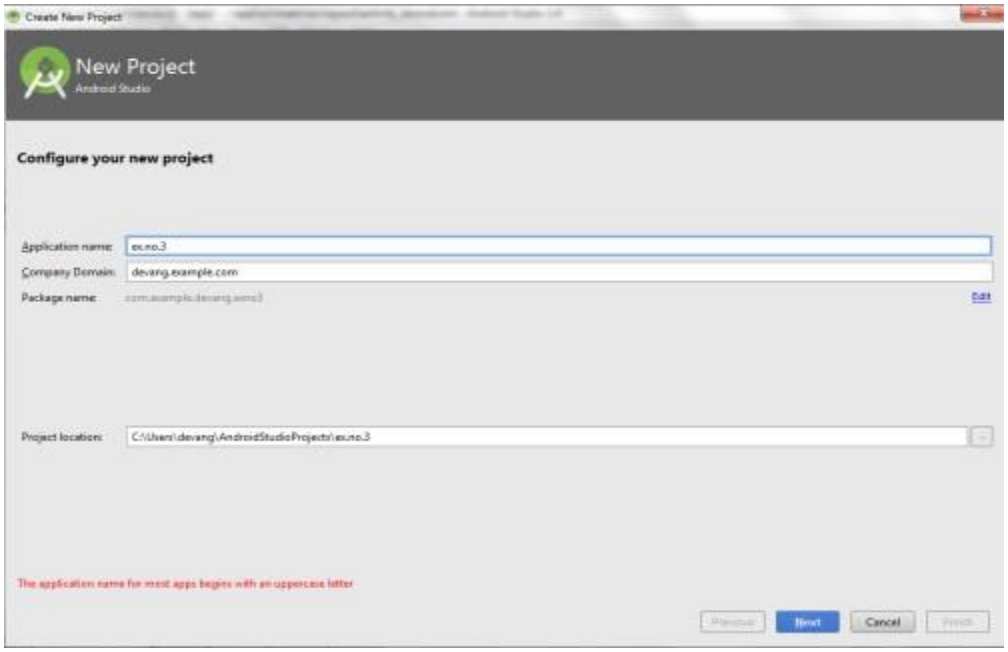
Procedure:

Creating a New project:

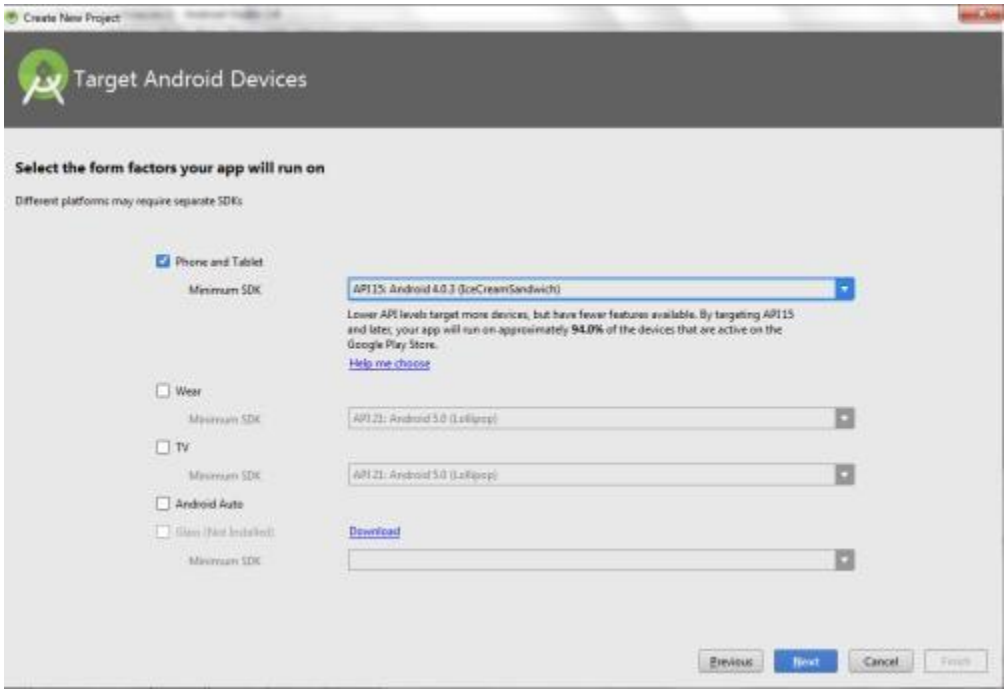
- Open Android Stdio and then click on **File -> New -> New project.**



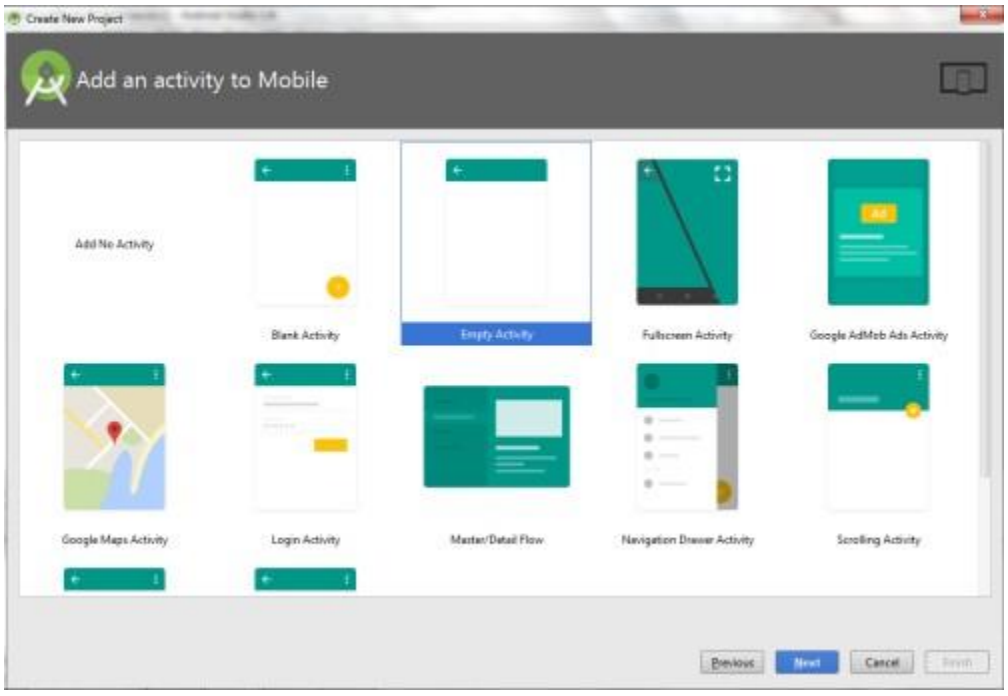
- Then type the Application name as "**ex.no.3"** and click **Next.**



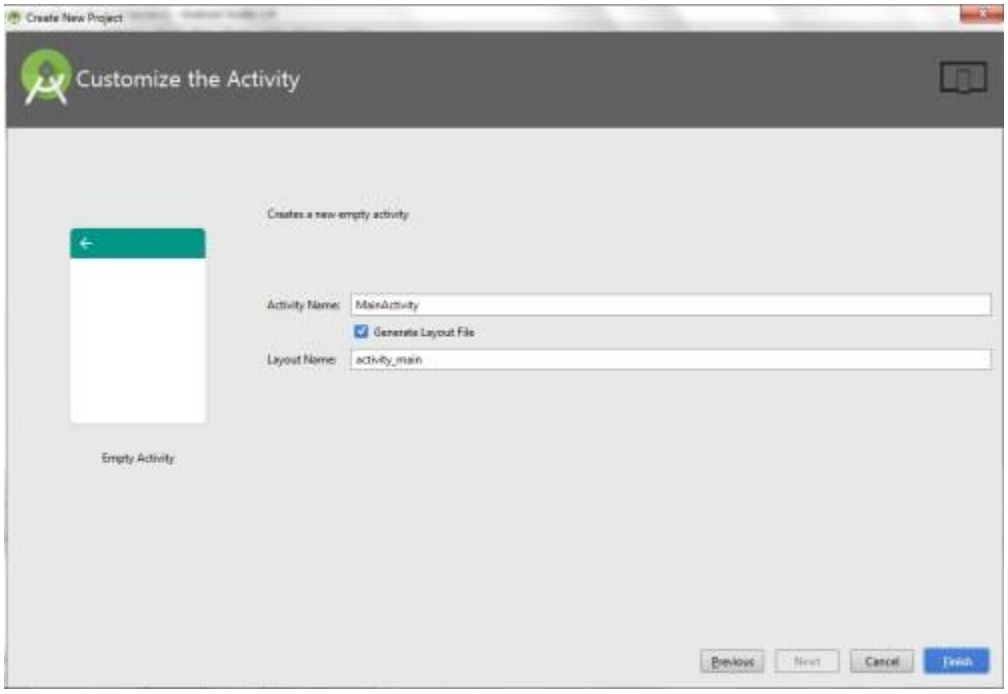
- Then select the **Minimum SDK** as shown below and click **Next**.



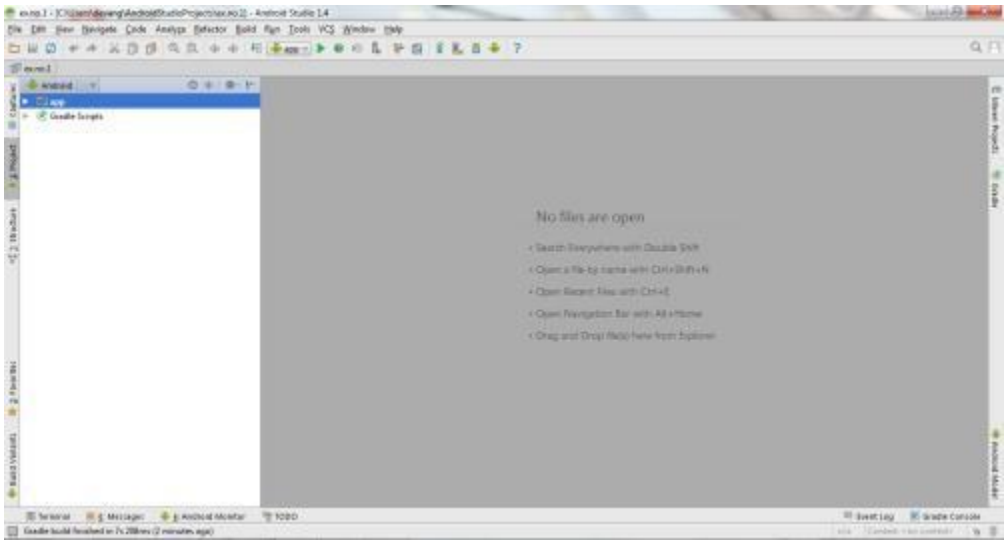
- Then select the **Empty Activity** and click **Next.**



- Finally click F**inish**.

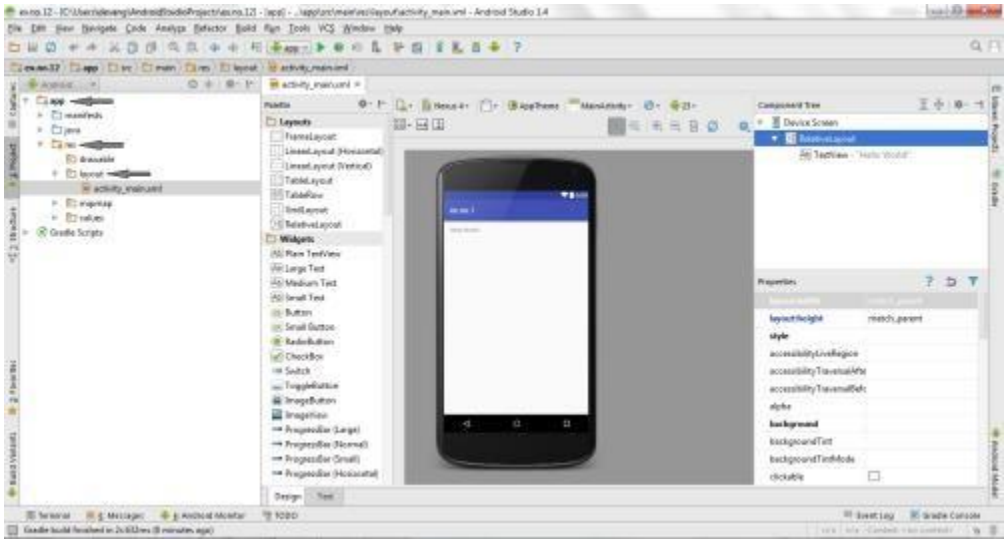


- It will take some time to build and load the project.
- After completion it will look as given below.

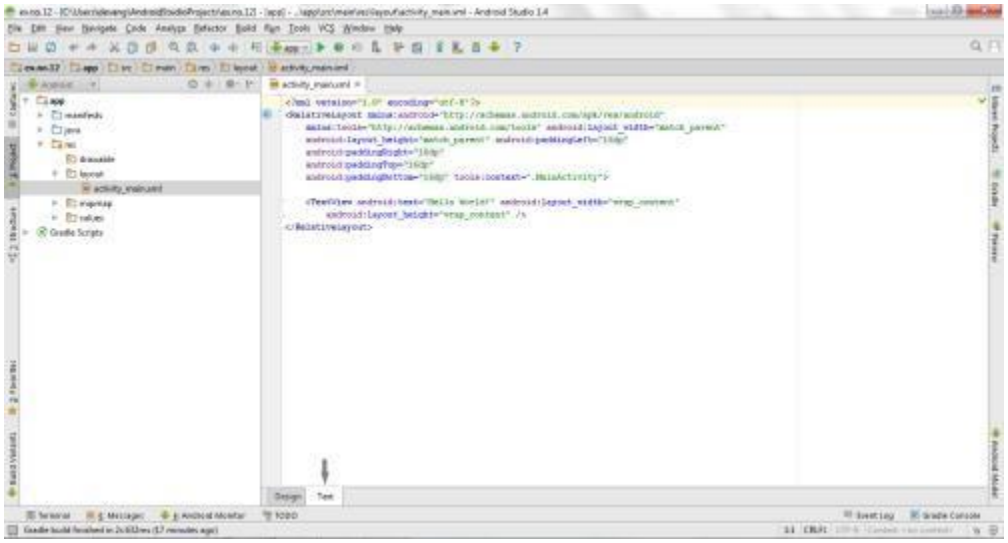


Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml.**



- Now click on **Text** as shown below.



- Then delete the code which is there and type the code as given below.

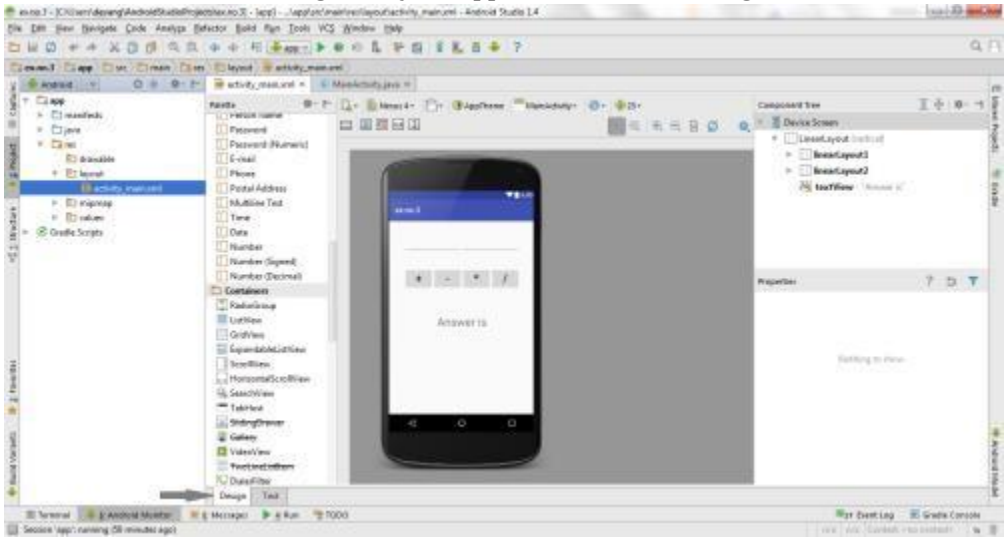
Code for Activity_main.xml:

```
<?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:layout_margin="20dp">

    <LinearLayout
```

```
    android:id="@+id/linearLayout1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="20dp">

    <EditText
        android:id="@+id/editText1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:inputType="numberDecimal"
        android:textSize="20sp" />

    <EditText
        android:id="@+id/editText2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:inputType="numberDecimal"
        android:textSize="20sp" />

</LinearLayout>

<LinearLayout
    android:id="@+id/linearLayout2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="20dp">

    <Button
        android:id="@+id/Add"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="+"
        android:textSize="30sp"/>

    <Button
        android:id="@+id/Sub"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="-"
        android:textSize="30sp"/>
```

```
    <Button
        android:id="@+id/Mul"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="*"
        android:textSize="30sp"/>

    <Button
        android:id="@+id/Div"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="/"
        android:textSize="30sp"/>

</LinearLayout>

<TextView
    android:id="@+id/textView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="50dp"
    android:text="Answer is"
    android:textSize="30sp"
    android:gravity="center"/>

</LinearLayout>
```

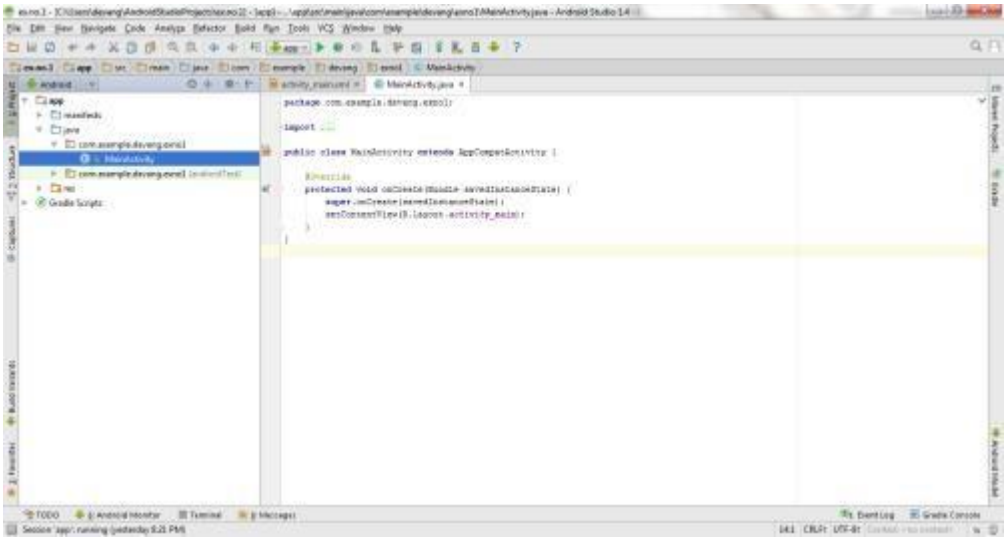
- Now click on Design and your application will look as given below.



- So now the designing part is completed.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno3 -> MainActivity.**



- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

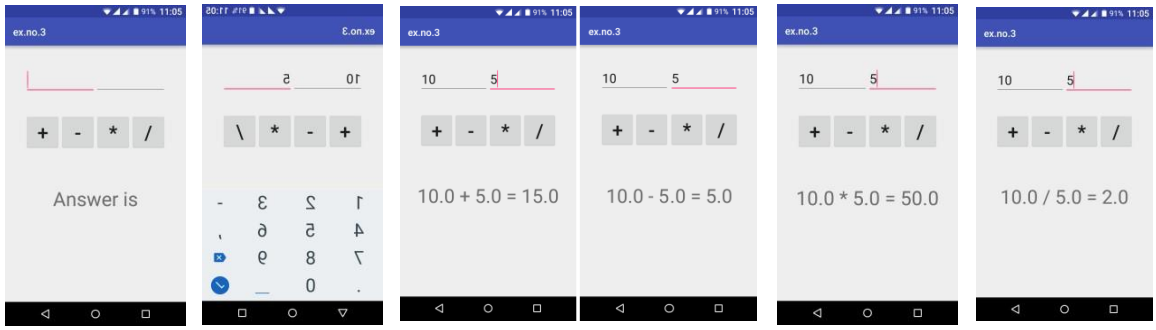
```
package com.example.devang.exno3;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.text.TextUtils;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity implements OnClickListener
{
   //Defining the Views
   EditText Num1;
   EditText Num2;
   Button Add;
   Button Sub;
   Button Mul;
   Button Div;
   TextView Result;
```

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

   //Referring the Views
   Num1 = (EditText) findViewById(R.id.editText1);
   Num2 = (EditText) findViewById(R.id.editText2);
   Add = (Button) findViewById(R.id.Add);
   Sub = (Button) findViewById(R.id.Sub);
   Mul = (Button) findViewById(R.id.Mul);
   Div = (Button) findViewById(R.id.Div);
   Result = (TextView) findViewById(R.id.textView);

   // set a listener
   Add.setOnClickListener(this);
   Sub.setOnClickListener(this);
   Mul.setOnClickListener(this);
   Div.setOnClickListener(this);
}

@Override
public void onClick (View v)
{

   float num1 = 0;
   float num2 = 0;
   float result = 0;
   String oper = "";

   // check if the fields are empty
   if (TextUtils.isEmpty(Num1.getText().toString()) || TextUtils.isEmpty(Num2.getText().toString()))
         return;

   // read EditText and fill variables with numbers num1
   = Float.parseFloat(Num1.getText().toString()); num2
   = Float.parseFloat(Num2.getText().toString());

   // defines the button that has been clicked and performs the corresponding operation
   // write operation into oper, we will use it later for output
   switch (v.getId())
   {
```

```
            case R.id.Add:
                oper = "+";
                result = num1 + num2;
                break;
            case R.id.Sub:
                oper = "-";
                result = num1 - num2;
                break;
            case R.id.Mul:
                oper = "*";
                result = num1 * num2;
                break;
            case R.id.Div:
                oper = "/";
                result = num1 / num2;
                break;
             default:
                break;
        }
        // form the output line
        Result.setText(num1 + " " + oper + " " + num2 + " = " + result);
    }
}
```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus a Simple Android Application for Native Calculator is developed and executed successfully.

EXPERIMENT 04

Aim: Develop android application for spinning bottle game
Procedure:

First step is to find two specific images: one for the floor which will be defined as background of the game and other for the bottle which will be rotated during the game.
We have chosen to use these both images:

Floor image for the background

Bottle image

Now, we need to define the layout of our game “Spin The Bottle”. We have a Relative Layout with the floor image as background and we add a bottle in the center of this layout :
Code:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:id="@+id/root"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="@drawable/background">

<ImageView
    android:id="@+id/bottle"
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:src="@drawable/beer_bottle"
    android:layout_centerInParent="true" />
```

</RelativeLayout>

The next step is to write the Java code of the Main Activity. First, we get references for the views. Then, we install a click listener on the main layout of the Main Activity. Thus, the user will just have to touch the screen to spin the bottle and to start the game.

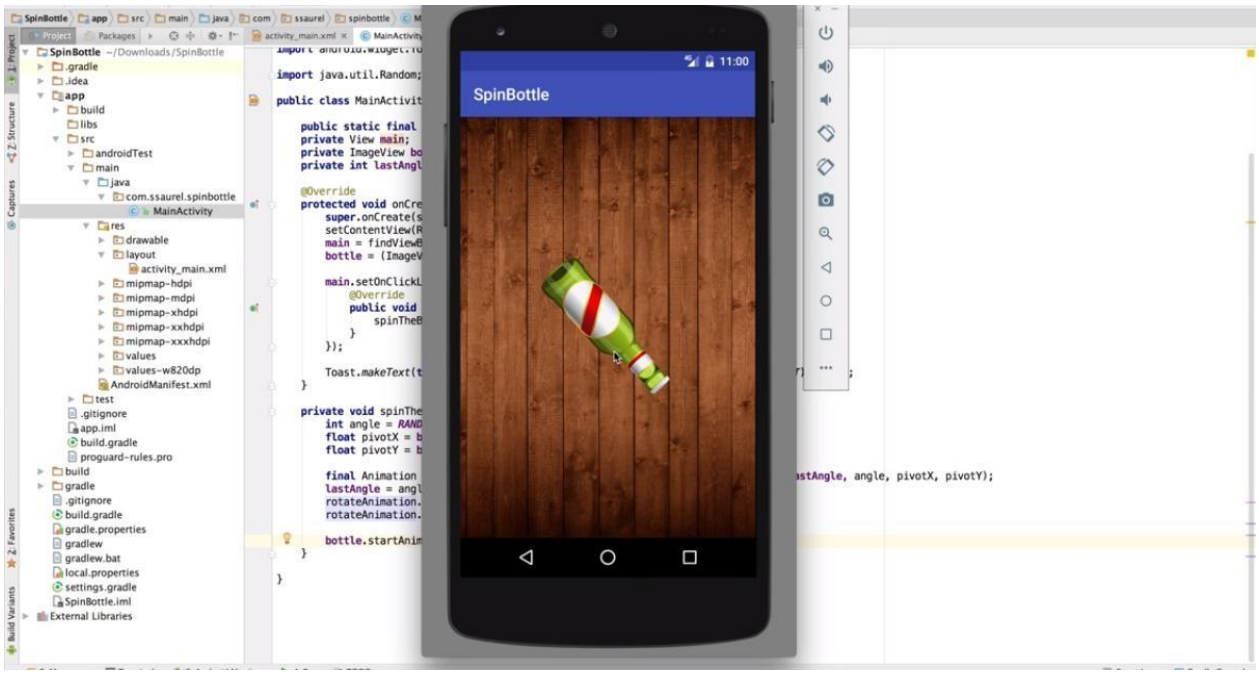
The core of the game is in the spinTheBottle() method. We are going to define a RotateAnimation from start angle to end angle centered on the center of the bottle. The end angle will be defined randomly to simulate really the game "Spin The Bottle". We should store the last end angle value to restart the bottle in the same position for the next turn of the game.

The Code of the Main Activity will have the following form :

```
package com.ssaurel.spinthebottle;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.RotateAnimation;
import android.widget.ImageView;
import android.widget.Toast;
import java.util.Random;

public class MainActivity extends AppCompatActivity {

  public static final Random RANDOM = new Random();
  private View main;
  private ImageView bottle;
  private int lastAngle = -1;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    main = findViewById(R.id.root);
    bottle = (ImageView) findViewById(R.id.bottle);
    main.setOnClickListener(new View.OnClickListener() {
    @Override
      public void onClick(View view) {
        spinTheBottle();
      }
    });
    Toast.makeText(this, R.string.touch_to_spin, Toast.LENGTH_SHORT).show();
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.main, menu);
    return true;
```

```
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem item) {
    switch(item.getItemId()) {
      case R.id.action_spin :
        spinTheBottle();
        break;
      case R.id.action_zero :
        resetTheBottle();
        break;
    }
    return super.onOptionsItemSelected(item);
  }

  private void spinTheBottle() {
    int angle = RANDOM.nextInt(3600 - 360) + 360;
    float pivotX = bottle.getWidth() / 2;
    float pivotY = bottle.getHeight() / 2;
    final Animation animRotate = new RotateAnimation(lastAngle == -1 ? 0 : lastAngle, angle, pivotX, pivotY);
    lastAngle = angle;
    animRotate.setDuration(2500);
    animRotate.setFillAfter(true);
    bottle.startAnimation(animRotate);
  }

  private void resetTheBottle() {
    float pivotX = bottle.getWidth() / 2;
    float pivotY = bottle.getHeight() / 2;
    final Animation animRotate = new RotateAnimation(lastAngle == -1 ? 0 : lastAngle, 0, pivotX, pivotY);
    lastAngle = -1;
    animRotate.setDuration(2000);
    animRotate.setFillAfter(true);
    bottle.startAnimation(animRotate);
  }
}
```

Output:



Result:

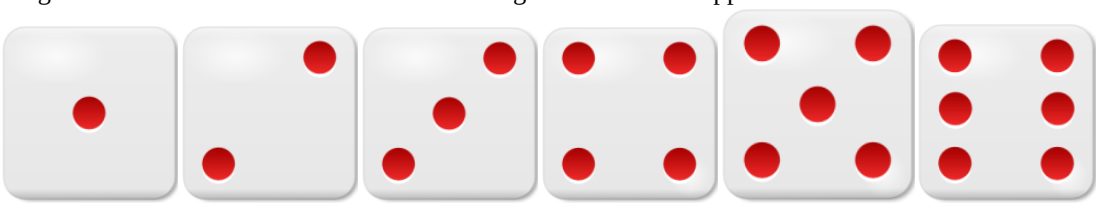
After implementing this, students will understand the use of multimedia files and random function in android studio.

EXPERIMENT 05

Aim: Android Studio Code to Develop Rolling Dice Game
The Dice Face Images and Sound

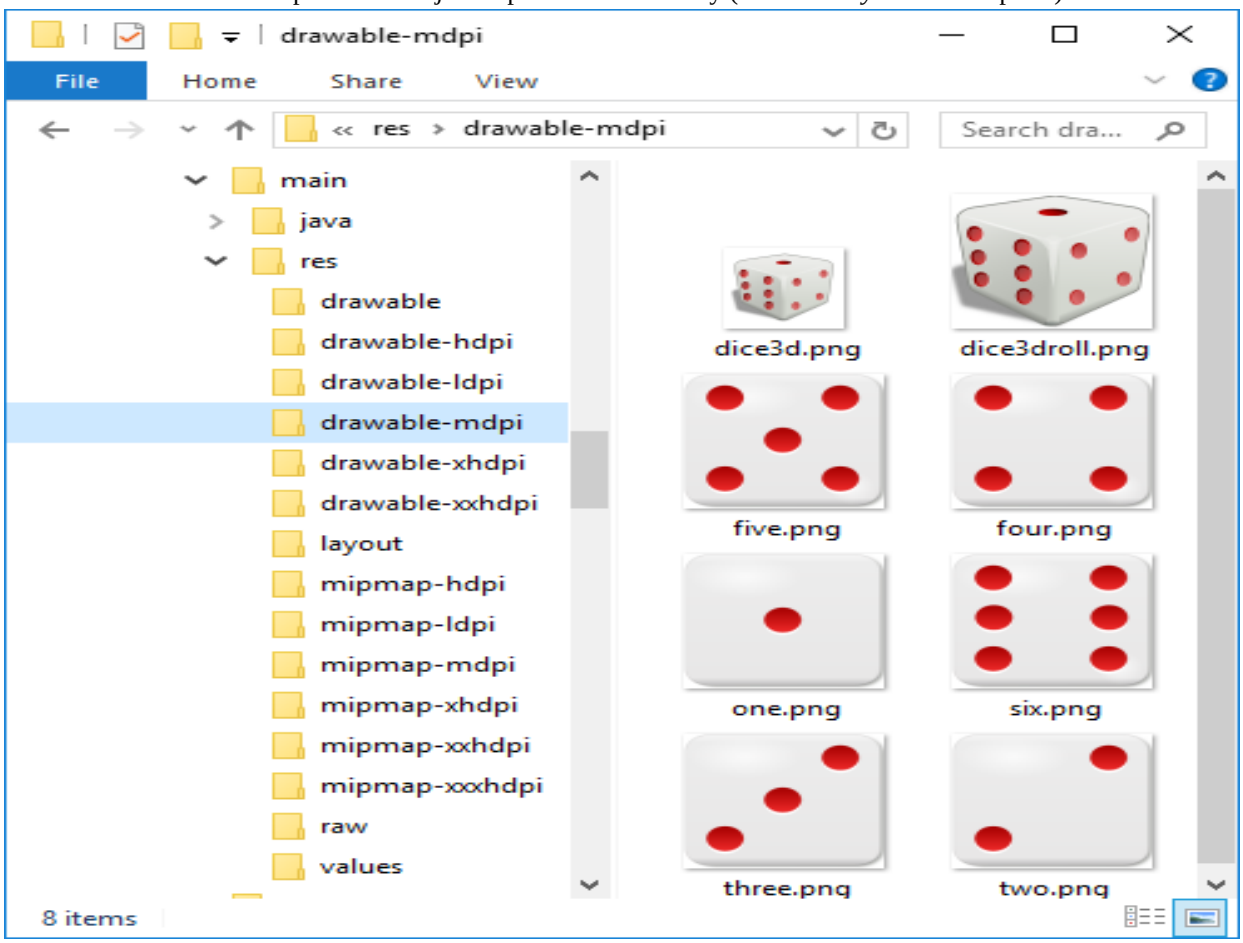
Procedure:

The code given in this article is for a common six sided dice. A dice is a cube, and each side of the cube (each face) has one of the numbers from one to six. Here six PNG images are used to show the dice roll result. Plus there is a 3D image shown for the dice roll. The same 3D image is used for the app icon.



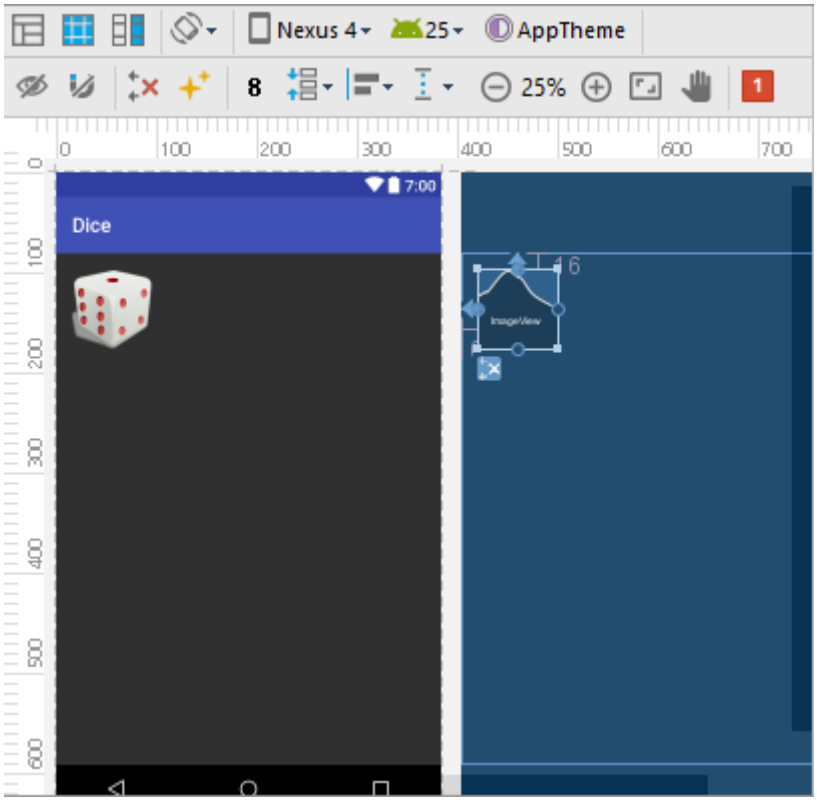
The sound of a dice roll is stored in shake_dice.mp3. It is by Mike Koenig and is from SoundBilble.com.
All the resources ready to add into a Android Studio project are available from this dice resources Zip file.
Android Studio Dice Roller Project

For this tutorial start a new Android Studio project. Here the *Application name* is *Dice* and an *Empty Activity* is used with other settings left at default values. Add the dice resources (from above) to the project by copying them to the *res* folder. Studio will update the *Project* explorer automatically (or use the *Synchronize* option).



Optionally change the app icon entries in the *AndroidManifest.xml* file to *dice3d* and *dice3d_rounded.png*, i.e. android:icon="@drawable/dice3d" and android:roundIcon="@mipmap/dice3d_rounded".
Android Dice Roller Project Layout

The screen for the dice roller is very simple. Just an ImageView which when pressed the roll is performed. If using the basic starting empty screen then open the *activity_main.xml* layout file. Delete the *Hello World!* default TextView (if present). From the widget *Pallete*, under *Images*, drag and drop an ImageView onto the layout and select *dice3droll* on the *Resources* chooser dialog. (Add the constraints if using a ConstraintLayout.)



The layout source should be similar to the following:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.dice.MainActivity">

<ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="16dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:contentDescription="Press for Dice Roll"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/dice3droll" />
</android.support.constraint.ConstraintLayout>
```

Android Dice Roller Source Code

A Java Random class (to generate the dice numbers) is declared along with an Android SoundPool (to play the dice roll sound). To load the right picture for each roll a *switch* statement is used. Some feedback is provided to the user so that they can see that a new roll has occurred. An identical number can be rolled in succession. To provide the feedback a 3D dice image will be displayed. However, because the roll happens so fast (unlike a real dice) the 3D image would not be seen. Therefore a Timer is used to provide a delay. This allows the UI to update. The Timer sends a Handler message signal to a Callback to perform the roll (the same method as described in the post Why Your TextView or Button Text Is Not Updating). Finally the roll value is used to update the UI.

Update: The creation of the SoundPool object has changed due to the deprecation of the constructor in Android API 21 and later. The code has been changed to allow for this, using a SoundPool.Builder. To support pre-Lollipop Android versions a new class is added to the project called *PreLollipopSoundPool*. The code for this new class follows this *MainActivity* class:

```
package com.example.dice;

import android.media.AudioAttributes;
import android.media.SoundPool;
import android.os.Build;
import android.os.Handler;
import android.os.Message;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;

import java.util.Random;
import java.util.Timer;
import java.util.TimerTask;

publicclassMainActivityextendsAppCompatActivity {
   ImageView dice_picture;     //reference to dice picture
   Random rng=new Random();    //generate random numbers
   SoundPool dice_sound;       //For dice sound playing
int sound_id;              //Used to control sound stream return by SoundPool
   Handler handler;           //Post message to start roll
   Timer timer=new Timer();    //Used to implement feedback to user
boolean rolling=false;      //Is die rolling?

   @Override
protectedvoidonCreate(Bundle  savedInstanceState)  {
super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      //Our function to initialise sound playing
      InitSound();
      //Get a reference to image widget
      dice_picture = (ImageView) findViewById(R.id.imageView);
      dice_picture.setOnClickListener(new HandleClick());
      //link handler to callback
      handler=new Handler(callback);
   }
```

```
    //User pressed dice, lets start
privateclassHandleClickimplementsView.OnClickListener {
publicvoidonClick(View arg0) {
if (!rolling) {
                rolling = true;
                //Show rolling image
                dice_picture.setImageResource(R.drawable.dice3droll);
                //Start rolling sound
                dice_sound.play(sound_id, 1.0f, 1.0f, 0, 0, 1.0f);
                //Pause to allow image to update
                timer.schedule(new Roll(), 400);
            }
        }
    }

    //New code to initialise sound playback
voidInitSound() {
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
            //Use the newer SoundPool.Builder
            //Set the audio attributes, SONIFICATION is for interaction events
            //uses builder pattern
            AudioAttributes aa = new AudioAttributes.Builder()
                    .setUsage(AudioAttributes.USAGE_ASSISTANCE_SONIFICATION)
                    .setContentType(AudioAttributes.CONTENT_TYPE_SONIFICATION)
                    .build();

            //default max streams is 1
            //also uses builder pattern
            dice_sound= new SoundPool.Builder().setAudioAttributes(aa).build();

        } else {
            //Running on device earlier than Lollipop
            //Use the older SoundPool constructor
            dice_sound=PreLollipopSoundPool.NewSoundPool();
        }
        //Load the dice sound
        sound_id=dice_sound.load(this,R.raw.shake_dice,1);
    }

    //When pause completed message sent to callback
classRollextendsTimerTask  {
publicvoidrun() {
            handler.sendEmptyMessage(0);
        }
    }

    //Receives message from timer to start dice roll
    Handler.Callback callback = new Handler.Callback() {
```

```
publicbooleanhandleMessage(Message msg) {
        //Get roll result
        //Remember nextInt returns 0 to 5 for argument of 6
        //hence + 1
Switch (rng.nextInt(6)+1) {
case 1:
            dice_picture.setImageResource(R.drawable.one);
break;
case 2:
            dice_picture.setImageResource(R.drawable.two);
break;
case 3:
            dice_picture.setImageResource(R.drawable.three);
break;
case 4:
            dice_picture.setImageResource(R.drawable.four);
break;
case 5:
            dice_picture.setImageResource(R.drawable.five);
break;
case 6:
            dice_picture.setImageResource(R.drawable.six);
break;
default:
        }
        rolling=false;  //user can press again
returntrue;
      }
   };

   //Clean up
protectedvoidonPause() {
super.onPause();
      dice_sound.pause(sound_id);
   }
protectedvoidonDestroy()  {
super.onDestroy();
      timer.cancel();
   }
}
```

To support APIs pre Lollipop add a new class called *PreLollipopSoundPool* in the same folder as the *MainActivity* class. Creating it as a separate class supports Java lazy loading, ensuring the app works when the deprecated method is removed from Android.

```
package com.example.dice;

import android.media.AudioManager;
import android.media.SoundPool;

/**
```

```
 * Created a pre Lollipop SoundPool
 */
finalclassPreLollipopSoundPool {
    @SuppressWarnings("deprecation")
publicstatic SoundPool NewSoundPool() {
returnnew SoundPool(1, AudioManager.STREAM_MUSIC,0);
    }
}
```

The Android dice roller source code is now ready to run, either in an AVD or an actual Android device.

Output:

Result: The Spinning Bottle game is developed through Android Studio and all required knowledge and skills have understood

EXPERIMENT 06

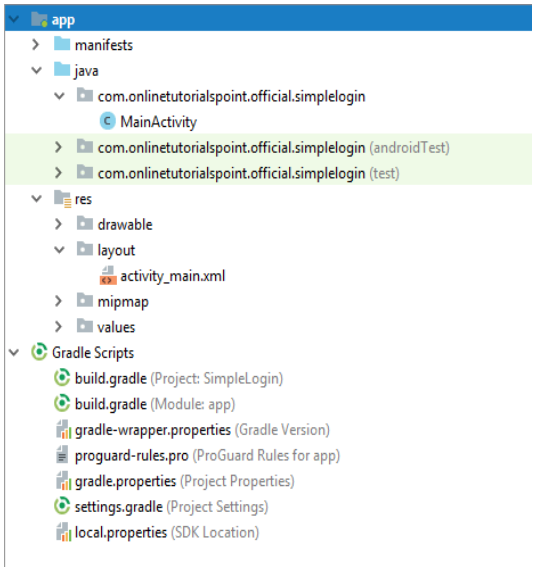
Aim: Develop android application for Login Page

Procedure:

Step1:

Creating a simple android login screen under Linear Layout.

Project Structure:



Gradle Dependencies:

```
apply plugin: 'com.android.application'

android {
compileSdkVersion 27
defaultConfig {
applicationId "com.onlinetutorialspoint.official.simplelogin"
minSdkVersion 23
targetSdkVersion 27
versionCode 1
versionName "1.0"
testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
}
buildTypes {
release {
minifyEnabled false
proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
}
}
}

dependencies {
implementation fileTree(dir: 'libs', include: ['*.jar'])
implementation 'com.android.support:appcompat-v7:27.0.1'
implementation 'com.android.support.constraint:constraint-layout:1.0.2'
testImplementation 'junit:junit:4.12'
```

```
    androidTestImplementation 'com.android.support.test:runner:1.0.1'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.1'
    }
```

activity_main.xml

Step2:

Creating a basic login form layout using LinearLayout having one text box, password and Login button.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context="com.onlinetutorialspoint.official.simplelogin.MainActivity">

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="vertical"
android:layout_marginLeft="16dp"
android:layout_marginRight="16dp"

android:layout_centerInParent="true">

<EditText
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="UserName"
android:id="@+id/username"/>
<EditText
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="password"
android:id="@+id/password"
android:inputType="textPassword"
/>

<Button
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="Login"
android:background="#3f76ff"
android:textColor="#fff"
android:id="@+id/login"/>

</LinearLayout>

</RelativeLayout>
```

MainActivity.java

Step3:

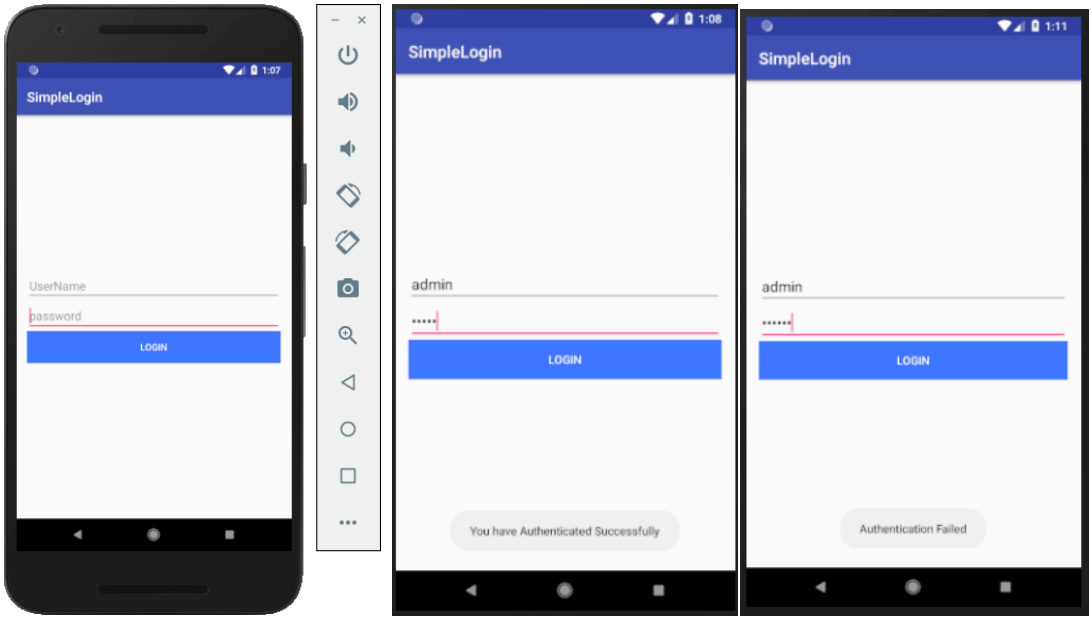
Creating MainActivity.java, responsible to read the input from the about layout and validate the user inputs.

```
package com.onlinetutorialspoint.official.simplelogin;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import java.util.Objects;

public class MainActivity extends AppCompatActivity {
EditText username,password;
Button login;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
username=findViewById(R.id.username);
password=findViewById(R.id.password);
login=findViewById(R.id.login);
login.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View view) {
if(Objects.equals(username.getText().toString(),
"admin")&&Objects.equals(password.getText().toString(),"admin"))
{
Toast.makeText(MainActivity.this,"You have Authenticated
Successfully",Toast.LENGTH_LONG).show();

}else
{
Toast.makeText(MainActivity.this,"Authentication  Failed",Toast.LENGTH_LONG).show();
}

}
});

}
}
```

Run Application:

Run Android Virtual Device (AVD):

Output:



Results:

After developing this application, students will be able to connect multiple classes in android studio

EXPERIMENT 07

Aim: Develop android application for web application

Procedure:

1. Introduction

An Android web app is an application that uses WebView to render part of the app's GUI. As name suggests, WebView is a View that displays Web pages. Let's say you are in Facebook and it shares some link. Instead of opening the link in browser, it uses the WebView component to open the link in the app itself. It is an extension of Android's View class that allows to display web pages as a part of Activity layout. WebView by default displays only webpage i.e. it does not include features of fully developed web browser like navigation and address bar. There are additional methods used to manually include all the features of a Web browser (managing cookies, history, navigating forward and backward, bookmarks and so on). We will discuss these methods in the next section.

WebView uses **Webkit** rendering engine to display web pages and includes all of the above methods. To know more about the Webkit check here.

2. Methods & Description

To have a more control over WebView, we can use the following methods defined in the WebView class.

- canGoBack() – This method specifies if the WebView has a back history item.
- canGoForward() – This method specifies if the WebView has a forward history item.
- clearHistory() – This method clears up the WebView forward and backward history.
- destroy() – This method destroys the internal state of WebView. It is required not to call any other method of WebView after calling destroy(), as this might can result in app crash.
- findAllAsync() – This method finds of instances of the string and highlight them. Works the same way as WebBrowser when we press **CTRL+F** to look for any String.
- getsProgress() – This method gets the progress of the current page. It returns the value between 1 and 100.
- getsTitle() – This method gets the title of the current page.
- getUrl() – This method gets the **URL** of the current page.

Signature of above methods

```
Webview webview = (Webview) findviewbyId(R.id.webview);

if(webView.canGoBack()){
 webView.goBack();

if(webView.canGoForward()){
 webView.goForward();

webView.clearHistory();

webView.destroy();

int findAllAsync(String find)

int progress = webview.getProgress()

String title = webview.getTitle()

String url = webview.getUrl()}
```

Apart from these methods, there are several other customization points where you can add your own behavior. These are:

- Creating and setting a WebChromeClient subclass. This class is called when something that might impact a browser UI happens, for instance, progress updates and JavaScript alerts are sent here.
- Creating and setting a WebViewClient subclass. It will be called when things happen that impact the rendering of the content(For example: errors or form submissions). You can also intercept URL loading here (via shouldOverrideUrlLoading()).
- Modifying the WebSettings, such as enabling JavaScript with setJavaScriptEnabled().
- Injecting Java objects into the WebView using the addJavascriptInterface(Object, String) method. This method allows you to inject Java objects into a page's JavaScript context, so that they can be accessed by JavaScript in the page.

Let's start creating a Android Web Application example that illustrates the use of WebView. IDE used – Android Studio 3.0 (You can use Eclipse as well, but Android studio works well with Gradle. Fore more info about Gradle read here – https://developer.android.com/studio/build/index.html
Download Android Studio 3.0: https://developer.android.com/studio/index.html

3. Simple Android WebView Example

Firstly, we will create a basic app which will show you how to add WebView excluding the additional methods which will be discussed in the latter section.

Layout File

To add a WebView to the application, simple include the element in activity layout.

activity_web.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical">

    <WebView
        android:id="@+id/webView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"  />
</LinearLayout>
```

Activity File

Now in the Activity use the loadUrl() to load a particular url.

WebActivity.java

```
public class WebActivity extends AppCompatActivity {
    WebView webView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_web);
        webView = (WebView) findViewById(R.id.webView);
        webView.loadUrl("www.google.com");
    }
}
```

Manifest File

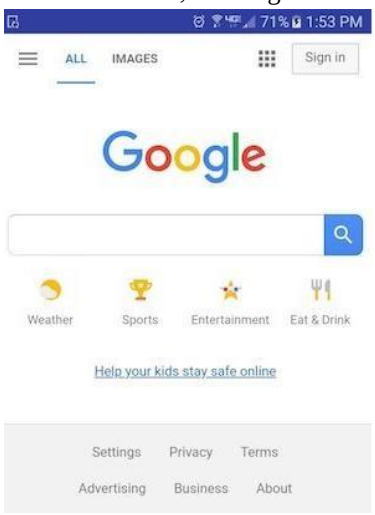
Since we we will be connecting to the web, add Internet permissions in the Manifest File.

AndroidManifest.xml

```
1    <uses-permission android:name="android.permission.INTERNET" />
```

Run the application

In the screenshot, the url gets loaded in the app browser.



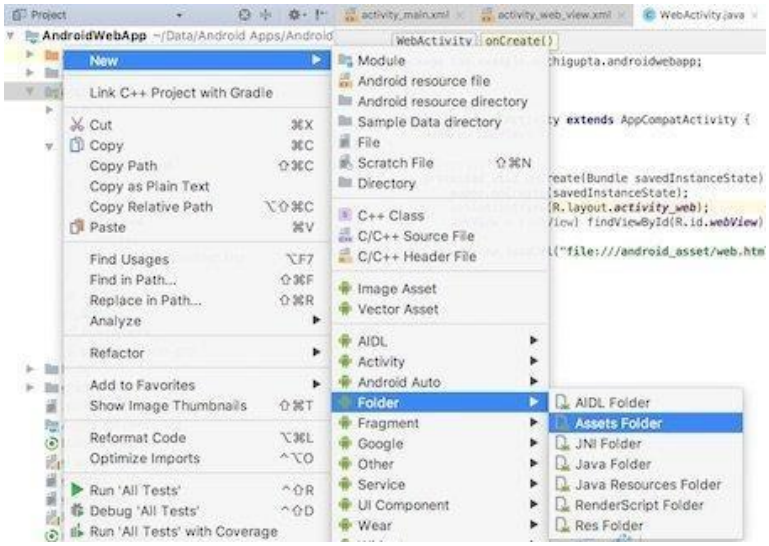
4. Display Custom HTML

Sometimes, there's need to open web page from the app's local storage instead from a url. In order to do so, keep all the HTML, CSS and Fonts in Assets folder and load them from there.

Create HTML file

Add the Assets folder in the application.

You can create the assets folder by: Right Click on app -> New -> Folder -> Assets Folder.



Create a HTML file in assets Folder

Now locate the assets folder here. src -> main -> assets. Create a new HTML file : Right Click on assets -> New -> File.

web.html

```
<!doctype html>
<html>
<head>
   <title>Android WebView Example</title>
</head>
```

```
<body>
  <h1 style='font-family:roboto;color:green;margin-left:30px;'>This is custom HTML loaded from app storage</h1>
  <p>Loading custom HTML is useful where you want to show static HTML, like privacy policy or user agreement in your app
</body>
</html>
```

Activity File

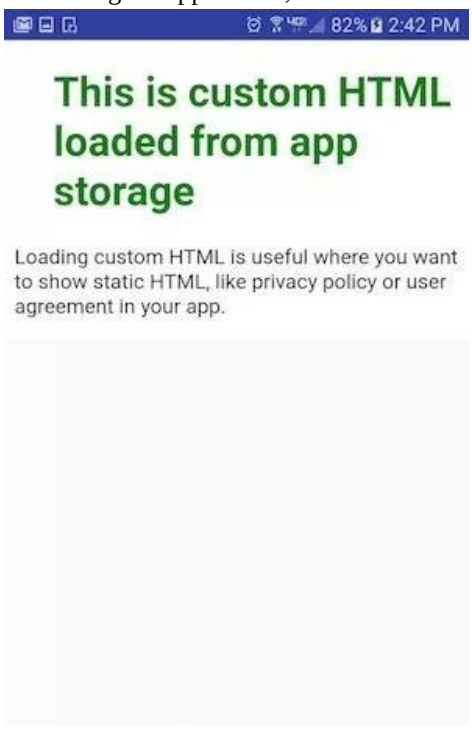
Now use the loadUrl() method to load a custom url.

WebActivity.java

```
1    webView.loadUrl("file:///android_asset/web.html");
```

Run the Application

On running the application, custom HTML will be showed up.



5. Using JavaScript in WebView

If the web page you plan to load in your WebView uses **JavaScript**, you must enable JavaScript for your WebView. Once JavaScript is enabled, you can also create interfaces between your application code and your JavaScript code. JavaScript is disabled in a WebView by default. It can be enabled through the WebSettings attached to the WebView. You can retrieve WebSettings with getSettings(), then enable JavaScript with setJavaScriptEnabled(). It's also possible to bind the JavaScript code in your web page to android WebView code like calling a method in your app to display alert dialog rather than using alert() in JavaScript.

Enabling JavaScript

```
WebView myWebView = (WebView) findViewById(R.id.webview);
WebSettings webSettings = myWebView.getSettings();
webSettings.setJavaScriptEnabled(true);
```

WebSettings provides access to a variety of other settings that you might find useful. For example, if you're developing a web application that's designed specifically for the WebView in your Android application, then you

can define a custom user agent string with setUserAgentString(), then query the custom user agent in your web page to verify that the client requesting your web page is actually your Android application.

6. Handling Page Navigation

When the user clicks a link from a web page in your WebView, the default behavior is for Android to launch an application in a web browser. However, you can override this behavior for your application and so, links will open within your WebView. You can then allow the user to navigate backward and forward through their web page history that's maintained by WebView.

To open the links clicked by user, simply provide a WebViewClient for your WebView, using setWebViewClient().

Signature of setWebViewClient()

```
myWebView.setWebViewClient(new WebViewClient());
```

If you want to add backward and forward arrows in the application just like Web application, it can be done manually by adding the icons in the ActionBar and call goBack() and goForward() respectively. By doing so, you would be able to handle the already browsed pages. We will use it in our code in next section.

7. Creating the Application with Page Navigation

In this example we will learn how to add back and forward arrows in Activity.

Layout File

This is the Main Activity. We are passing the url to the WebViewActivity.

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Button
        android:id="@+id/url"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="20dp"
        android:layout_marginStart="64dp"
        android:layout_marginTop="181dp"
        android:text="Go to https://www.w3schools.com/"
        android:textAllCaps="false"
        app:layout_constraintBottom_toTopOf="@+id/custom_html_button"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/custom_html_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="214dp"
        android:layout_marginStart="44dp"
        android:text="Load Custom HTML"
        android:textAllCaps="false"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="@+id/url"
        app:layout_constraintTop_toBottomOf="@+id/url" />
```

```
</android.support.constraint.ConstraintLayout>
```

Here we have another layout to display the web pages.

activity_web_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical">
    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme">
        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />
    </android.support.design.widget.AppBarLayout>

    <WebView
        android:id="@+id/webView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"  />
</LinearLayout>
```

Activity File

This the main activity. On clicking the Upper button, WebViewActivity will be opened. On clicking the Load Custom HTML button, WebActivity will be opened discussed in Section 4.

MainActivity.java

```
public class MainActivity extends Activity {

    private Button customButton;
    private Button urlButton;
    String url = "https://www.w3schools.com/";

    public void onCreate(Bundle savedInstanceState) {
        final Context context = this;

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        urlButton = (Button)findViewById(R.id.url);
        customButton = (Button) findViewById(R.id.custom_html_button);

        urlButton.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View arg0) {
                Intent intent = new Intent(MainActivity.this, WebViewActivity.class);
                intent.putExtra("url", url);
                startActivity(intent);
            }

        });
        customButton.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View arg0) {
```

```
                Intent intent = new Intent(context, WebActivity.class);
                startActivity(intent);
            }
        });
    }
}
```

Here is the WebViewActivity which will display the webpage and handling navigation.

WebViewActivity.java

```
public class WebViewActivity extends AppCompatActivity {

    private String url;
    private WebView webView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_web_view);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        getSupportActionBar().setTitle("");

        url = getIntent().getStringExtra("url");
        webView = (WebView) findViewById(R.id.webView);

        webView.setWebViewClient(new WebViewClient() {
            @Override
            public void onPageStarted(WebView view, String url, Bitmap favicon) {
                super.onPageStarted(view, url, favicon);
                invalidateOptionsMenu();
```

```
        }
        @Override
        public boolean shouldOverrideUrlLoading(WebView view, String url) {
            webView.loadUrl(url);
            return true;
        }
        @Override
        public void onPageFinished(WebView view, String url) {
            super.onPageFinished(view, url);
            invalidateOptionsMenu();
        }
        @Override
        public void onReceivedError(WebView view, WebResourceRequest request, WebResourceError error) {
            super.onReceivedError(view, request, error);
            invalidateOptionsMenu();
        }
    });
    webView.clearCache(true);
    webView.clearHistory();
    webView.loadUrl(url);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.brower, menu);
    return super.onCreateOptionsMenu(menu);
}

@Override
public boolean onPrepareOptionsMenu(Menu menu) {
```

```
        if (!webView.canGoBack()) {
            menu.getItem(1).setEnabled(false);
        } else {
            menu.getItem(1).setEnabled(true);
        }

        if (!webView.canGoForward()) {
            menu.getItem(2).setEnabled(false);
        } else {
            menu.getItem(2).setEnabled(true);
        }
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()){
            case R.id.action_home :
                webView.loadUrl("https://www.w3schools.com/");
                break;
            case R.id.action_back :
                if(webView.canGoBack()){ webView.goBack(); }
                break;
            case R.id.action_forward :
                if(webView.canGoForward()){ webView.goForward(); }
                break;
        }
        return super.onOptionsItemSelected(item);
    }
}
```

For your reference, to add the icons: Right click on Drawable Folder -> New -> Vector Asset. From here you can choose the icon to be added to the application.

Manifest File

This is the AndroidManifest File.

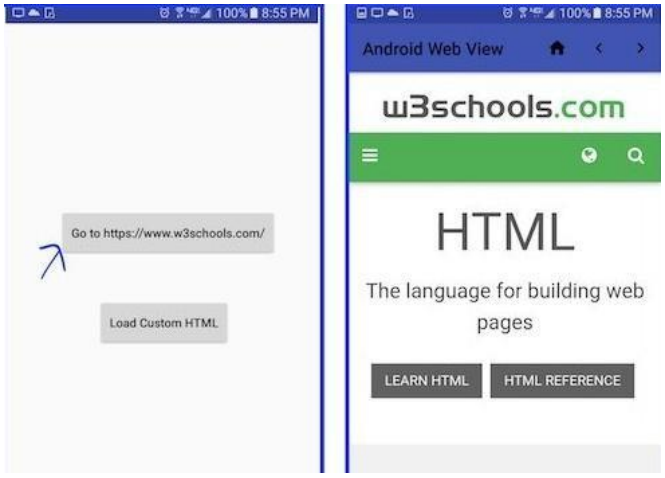
AndroidManifest.xml

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

    <activity
      android:name=".MainActivity"
      android:label="@string/title_activity_web_view">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>

    <activity
      android:name=".WebViewActivity"
      android:label="@string/title_activity_web_view">
    </activity>
```

```
        <activity
            android:name=".WebActivity"
            android:label="@string/title_activity_web_view">
        </activity>

    </application>
    </manifest>
```

Run the application See

the screenshots here.



On clicking the Load Custom HTML button, the Image in Section 4.3 will be opened up.

Result:

We have seen how to use WebView in the application. A big advantage of using a WebView is that you can store assets inside the app. This lets your app work offline and improves load times, since the WebView can retrieve assets directly from the local file system. Also, it provides interesting methods for Web pages to interact with the Android app using Javascript interface. Another scenario where it is helpful is when you want to provide information in your application that you might need to update, such as an end-user agreement or a user guide. Within your Android application, you can create an Activity that contains a WebView, then use that to display your document that's hosted online.

EXPERIMENT 08

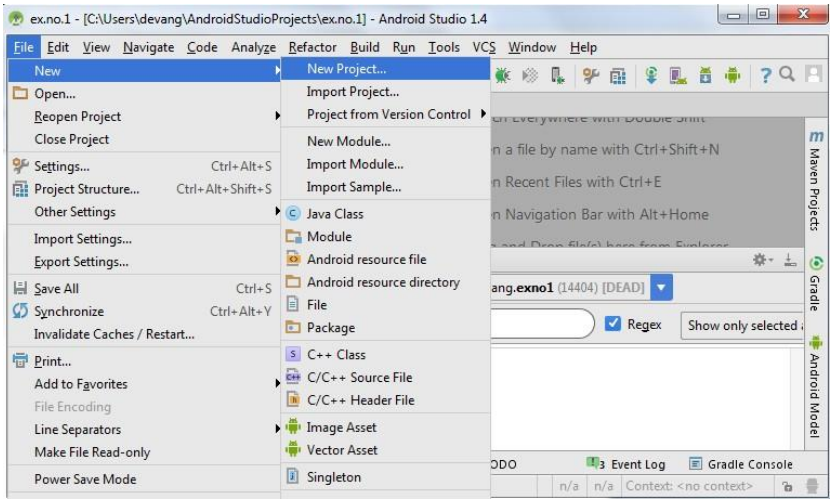
Aim:

To develop a Simple Android Application that draws basic Graphical Primitives on the screen.

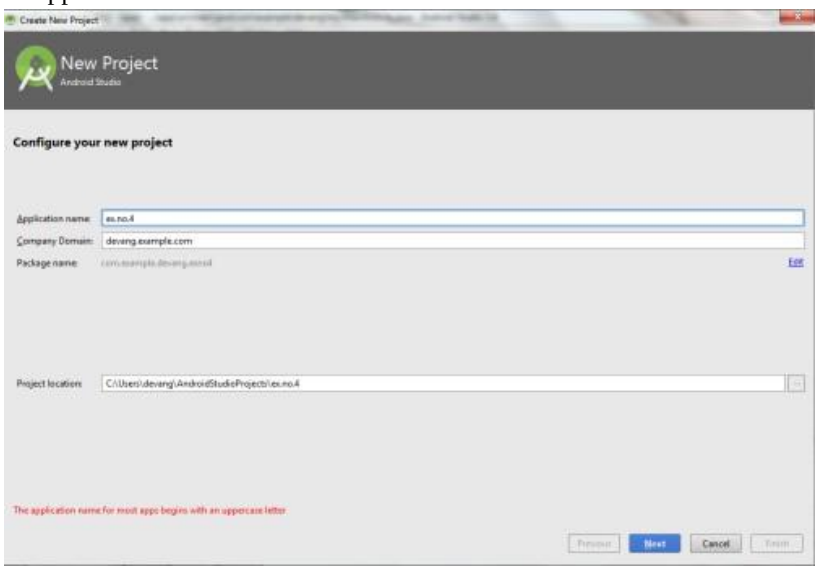
Procedure:

Creating a New project:

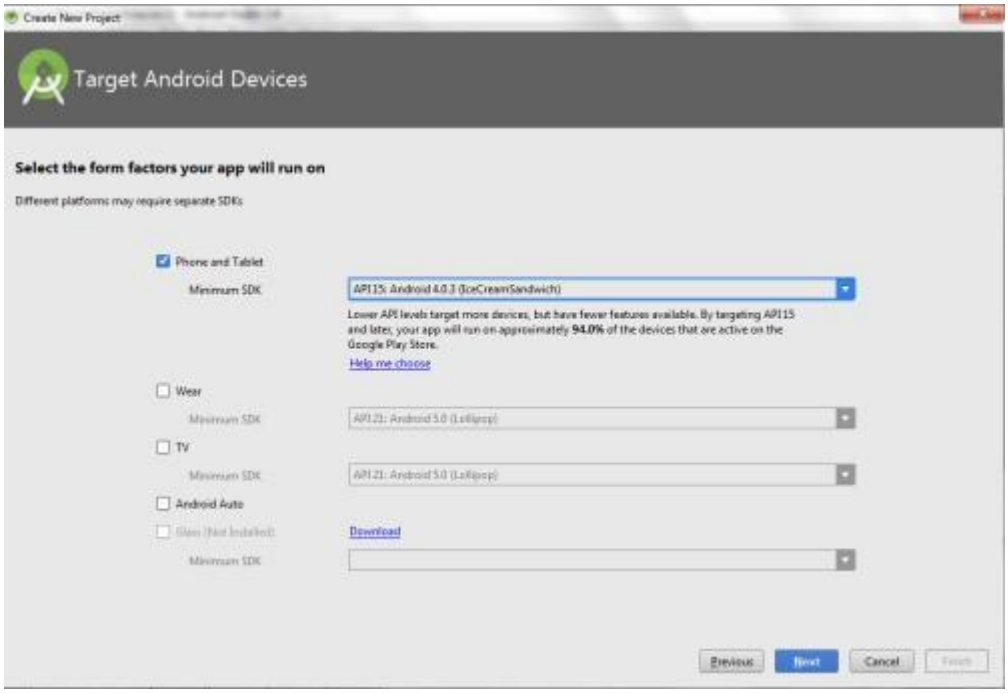
- Open Android Studio and then click on **File -> New -> New project.**



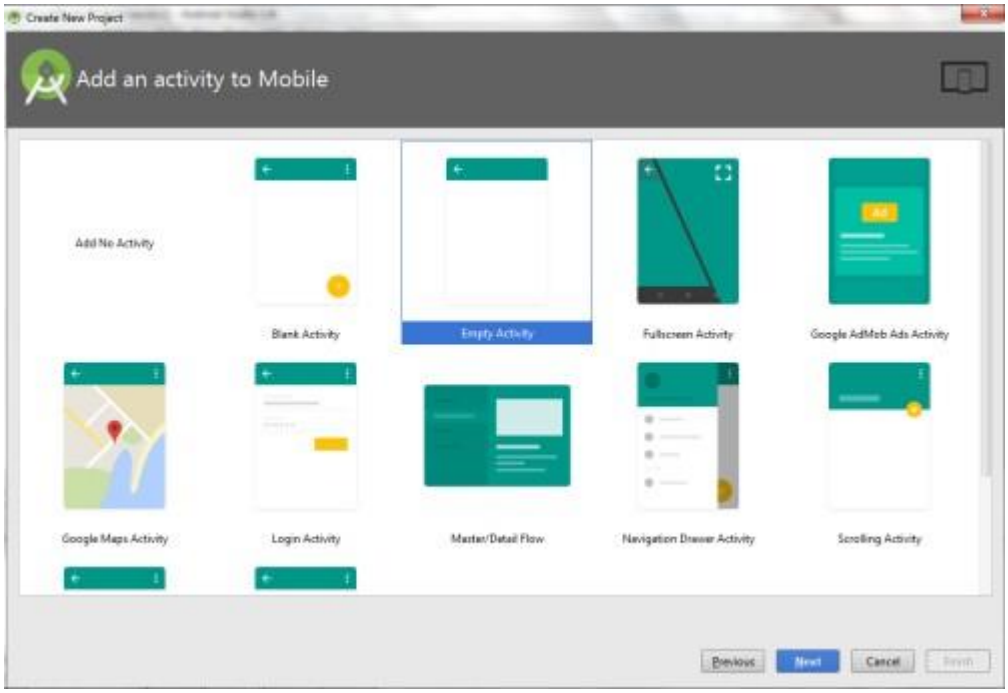
- Then type the Application name as "**ex.no.4"** and click **Next.**



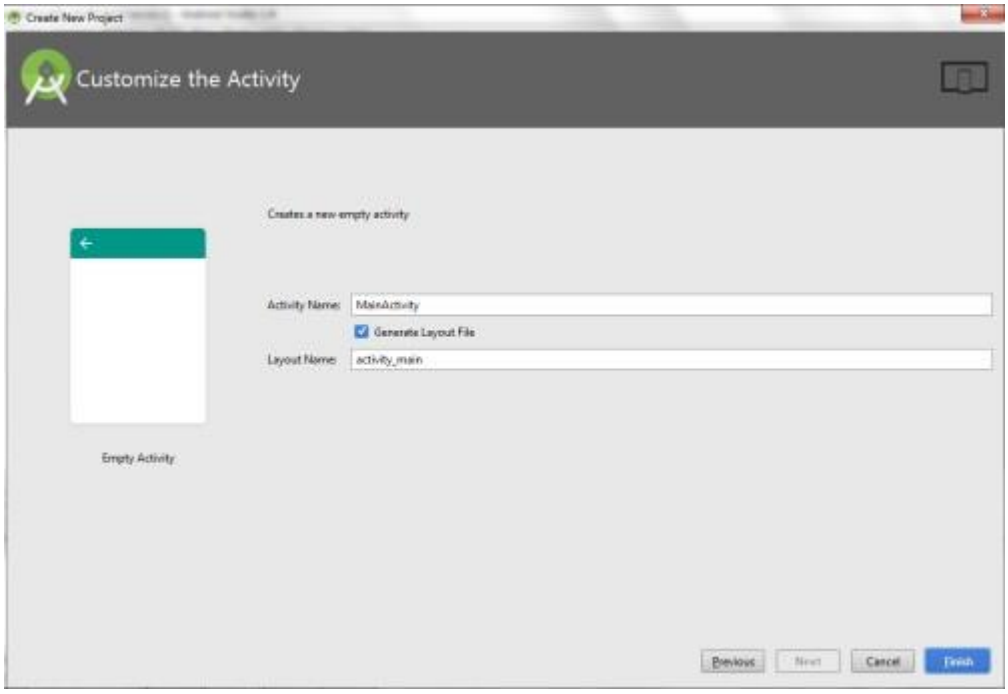
- Then select the **Minimum SDK** as shown below and click **Next**.



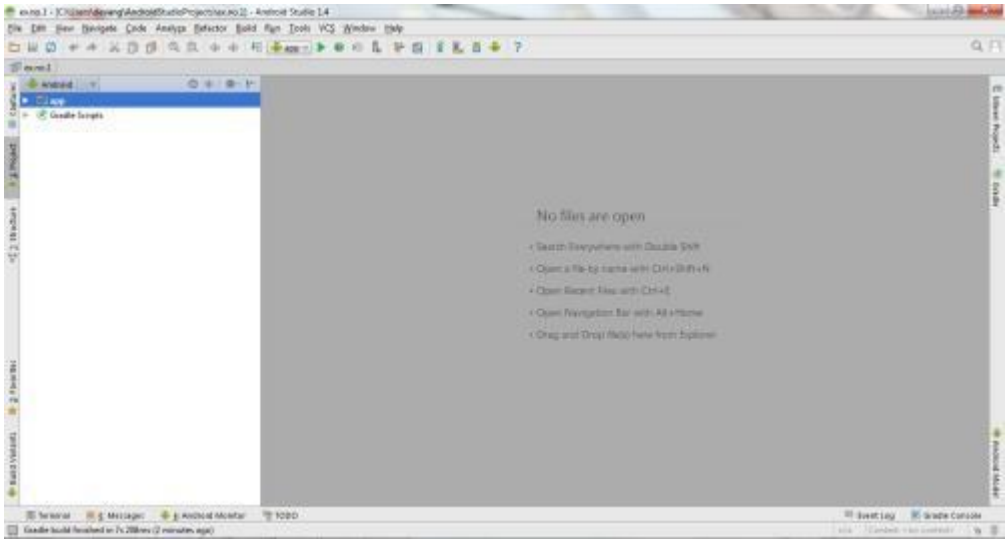
- Then select the **Empty Activity** and click **Next.**



- Finally click F**inish**.

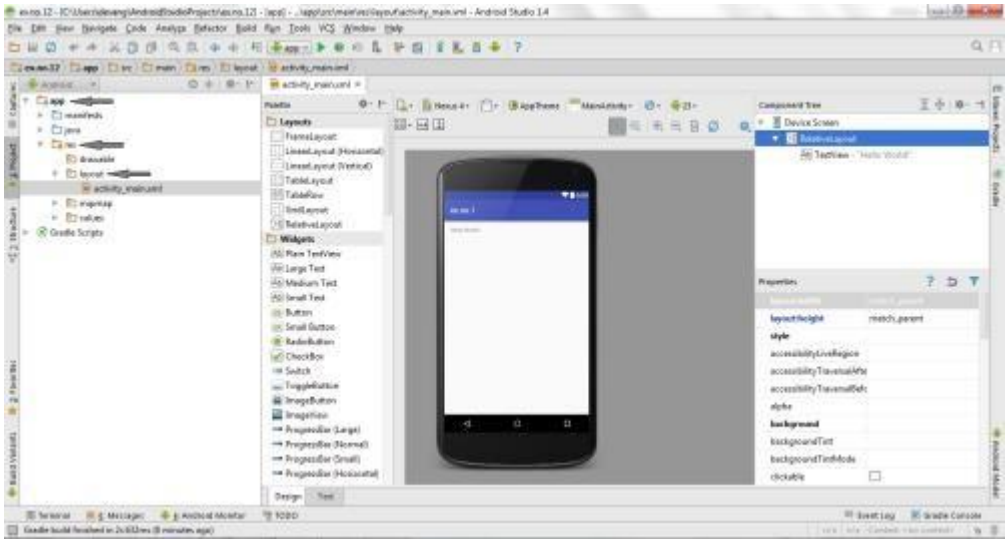


- It will take some time to build and load the project.
- After completion it will look as given below.

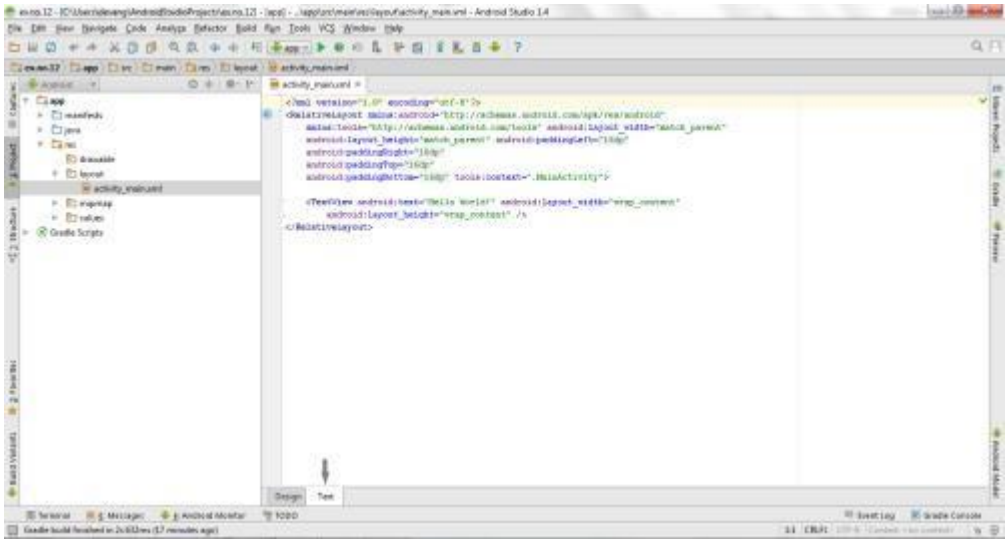


Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml.**



- Now click on **Text** as shown below.

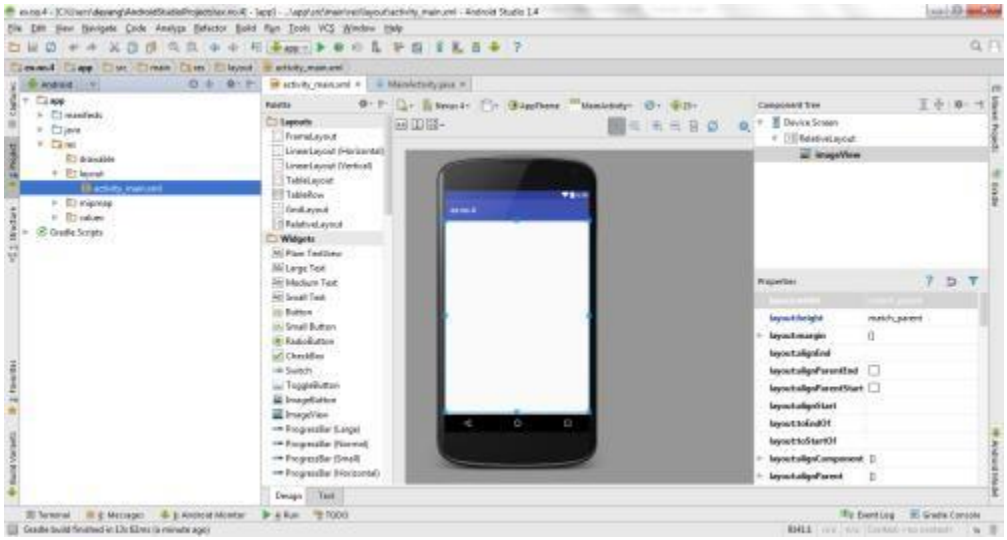


- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

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

   <ImageView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/imageView" />
</RelativeLayout>
```

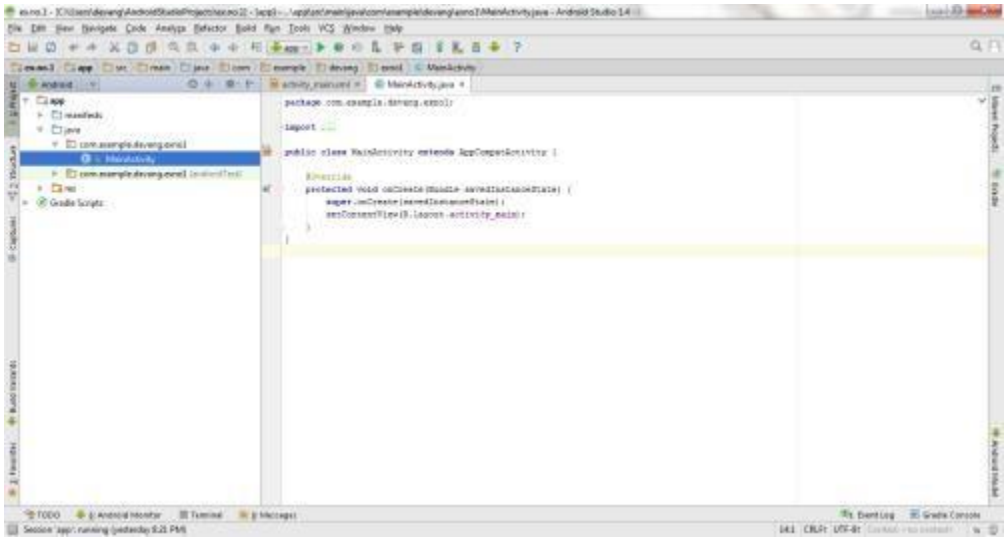
- Now click on **Design** and your application will look as given below.



- So now the designing part is completed.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno4 -> MainActivity.**



- Then delete the code which is there and type the code as given below.

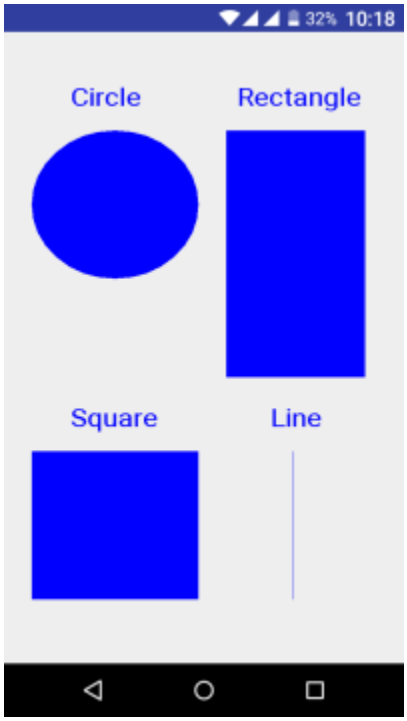
Code for MainActivity.java:

```
package com.example.exno4;

import android.app.Activity;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.drawable.BitmapDrawable;
import android.os.Bundle;
import android.widget.ImageView;

public class MainActivity extends Activity
```

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

        //Creating a Bitmap
        Bitmap bg = Bitmap.createBitmap(720, 1280, Bitmap.Config.ARGB_8888);

        //Setting the Bitmap as background for the ImageView
        ImageView i = (ImageView) findViewById(R.id.imageView);
        i.setBackgroundDrawable(new BitmapDrawable(bg));

        //Creating the Canvas Object
        Canvas canvas = new Canvas(bg);

        //Creating the Paint Object and set its color & TextSize
        Paint paint = new Paint();
        paint.setColor(Color.BLUE);
        paint.setTextSize(50);

        //To draw a Rectangle
        canvas.drawText("Rectangle", 420, 150, paint);
        canvas.drawRect(400, 200, 650, 700, paint);

        //To draw a Circle
        canvas.drawText("Circle", 120, 150, paint);
        canvas.drawCircle(200, 350, 150, paint);

        //To draw a Square
        canvas.drawText("Square", 120, 800, paint);
        canvas.drawRect(50, 850, 350, 1150, paint);

        //To draw a Line
        canvas.drawText("Line", 480, 800, paint);
        canvas.drawLine(520, 850, 520, 1150, paint);
    }
}
```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus a Simple Android Application that draws basic Graphical Primitives on the screen is developed and executed successfully.

EXPERIMENT 09

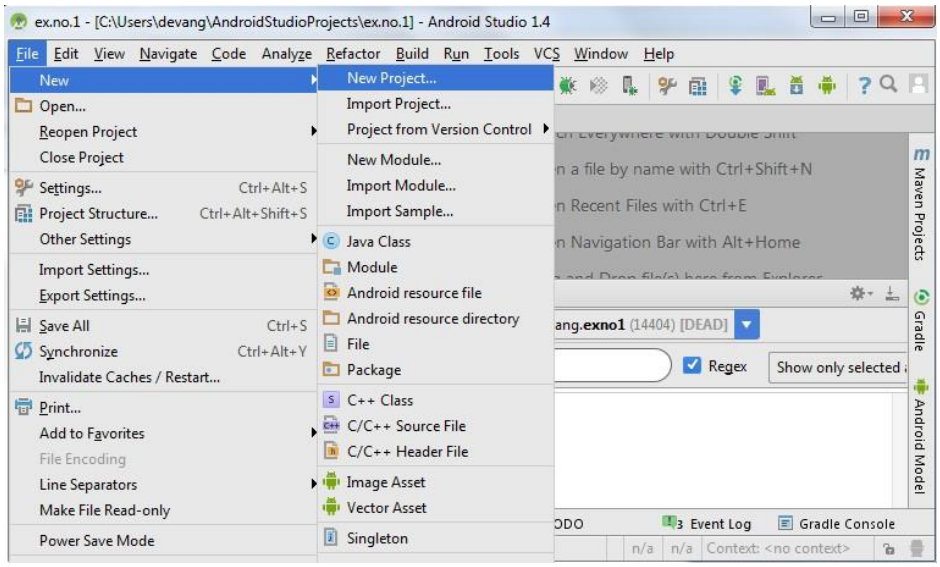
Aim:

To develop a Simple Android Application that makes use of Database.

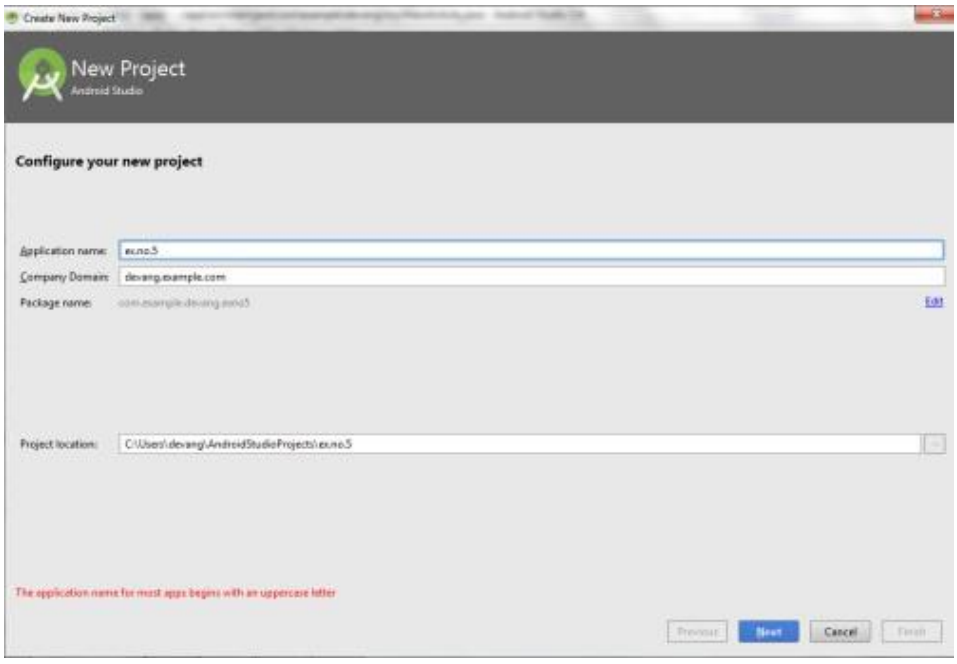
Procedure:

Creating a New project:

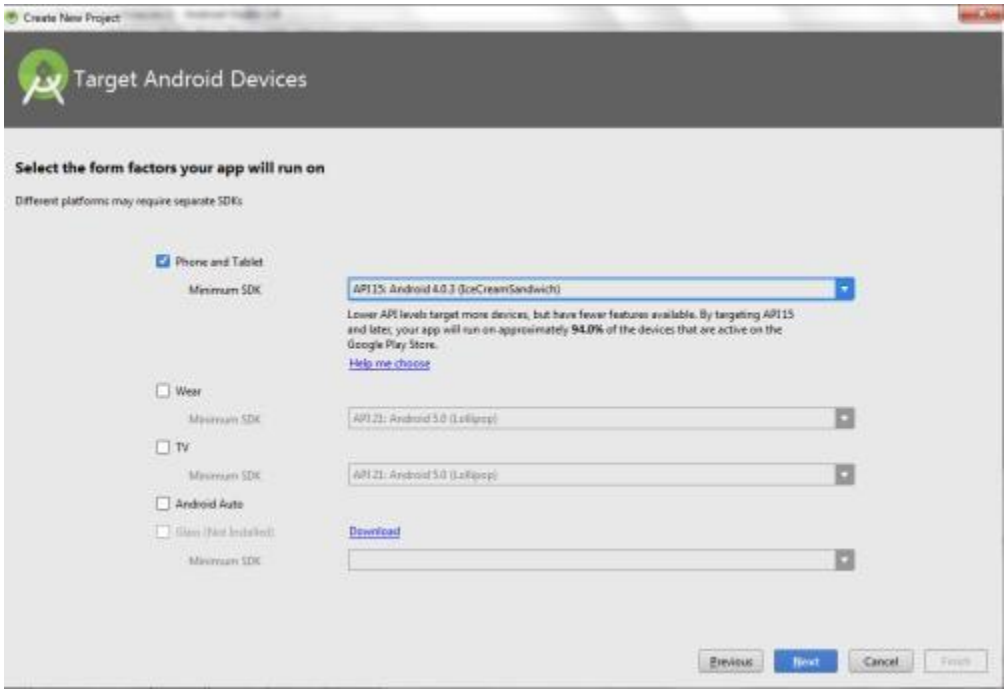
- Open Android Studio and then click on **File -> New -> New project.**



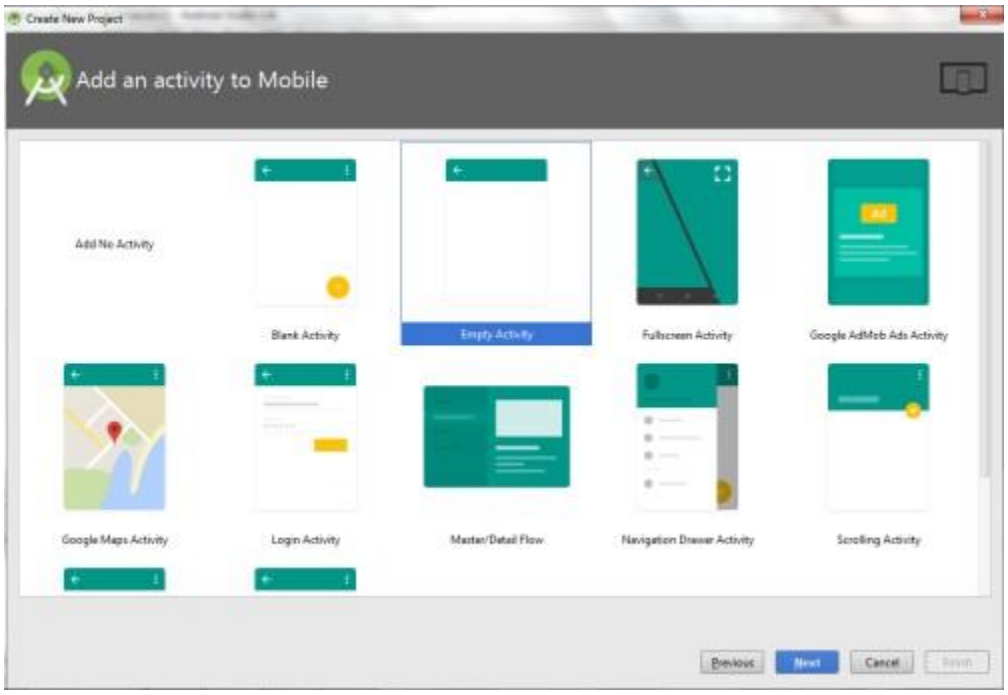
- Then type the Application name as "**ex.no.5"** and click **Next.**



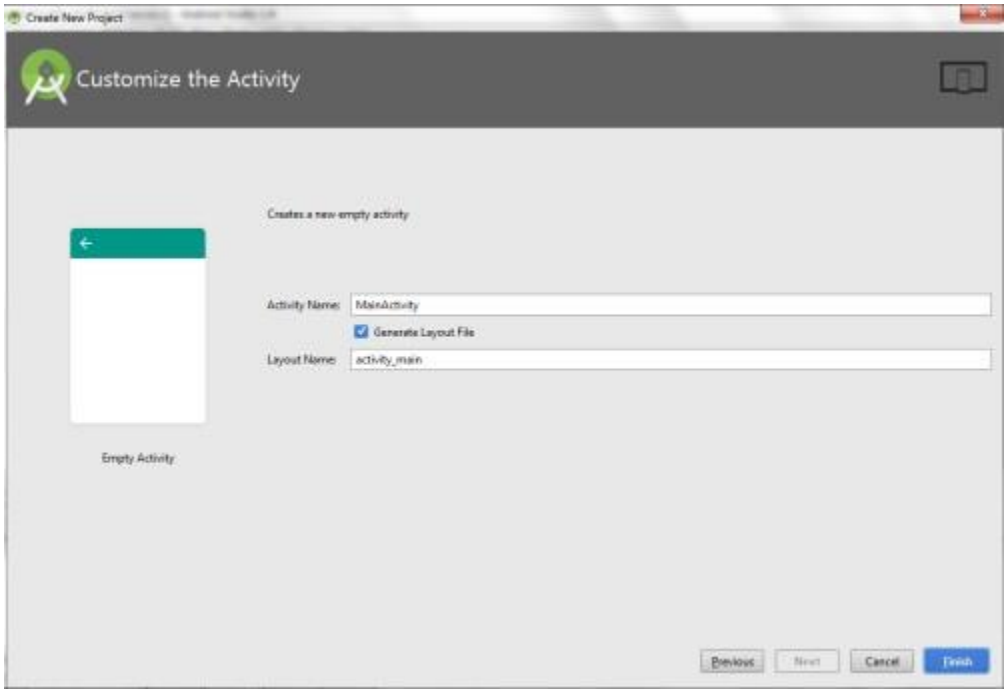
- Then select the **Minimum SDK** as shown below and click **Next**.



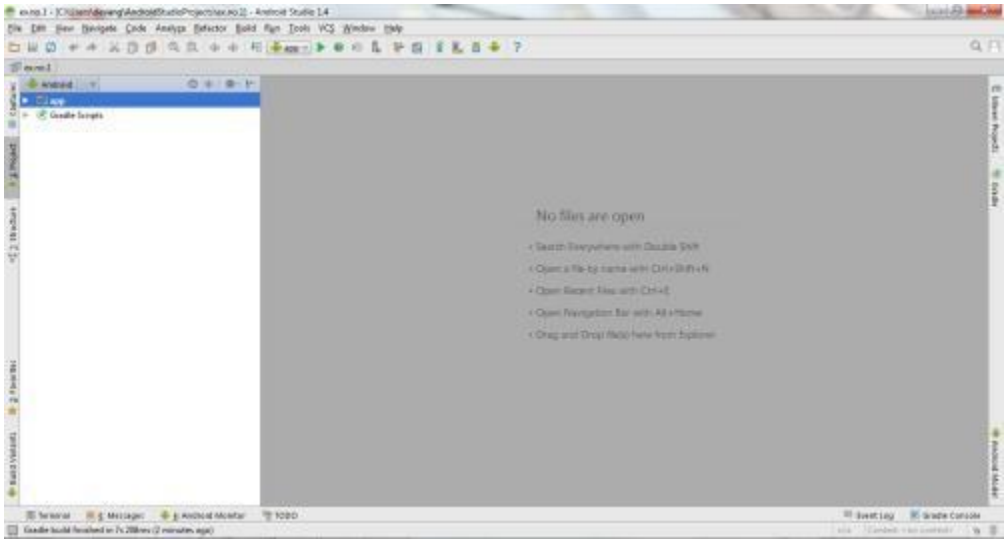
- Then select the **Empty Activity** and click **Next.**



- Finally click F**inish**.

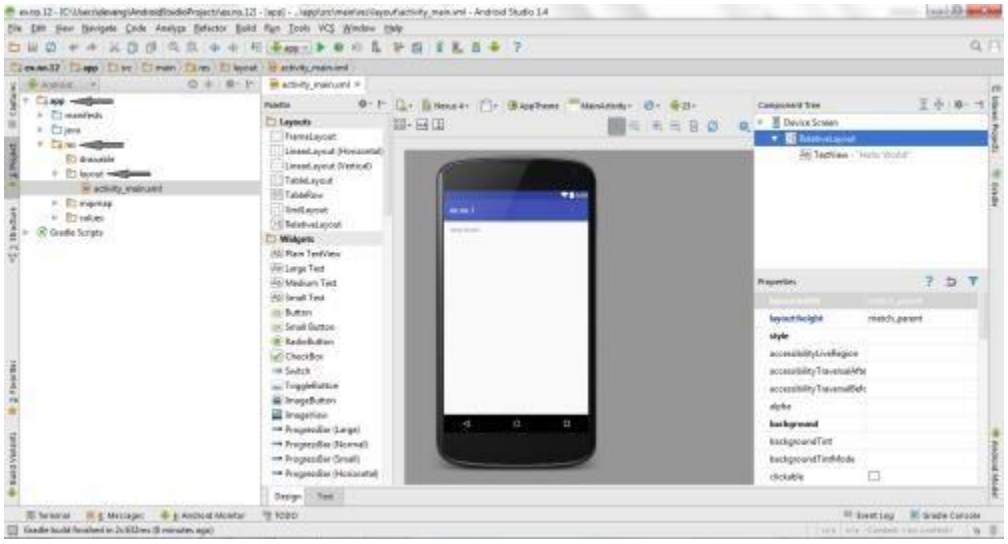


- It will take some time to build and load the project.
- After completion it will look as given below.

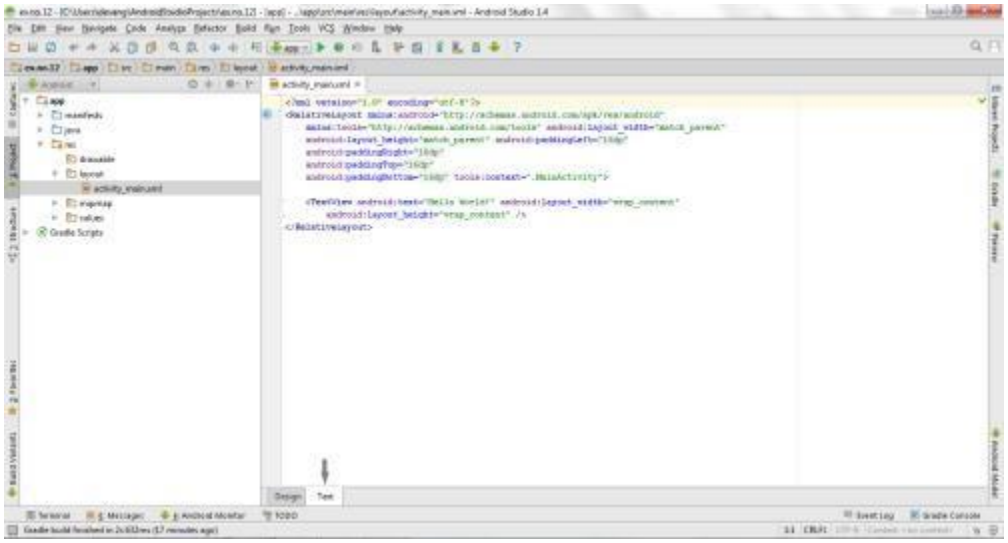


Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml.**



- Now click on **Text** as shown below.



- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

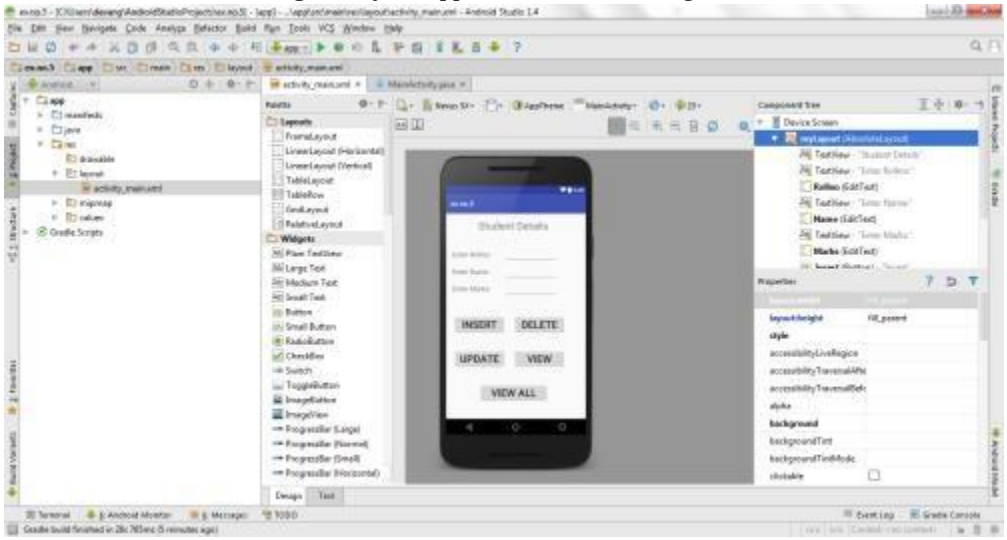
```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_x="50dp"
        android:layout_y="20dp"
        android:text="Student Details"
        android:textSize="30sp" />
```

```
<TextView
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_x="20dp"
   android:layout_y="110dp"
   android:text="Enter Rollno:"
   android:textSize="20sp" />

<EditText
   android:id="@+id/Rollno"
   android:layout_width="150dp"
   android:layout_height="wrap_content"
   android:layout_x="175dp"
   android:layout_y="100dp"
   android:inputType="number"
   android:textSize="20sp" />

<TextView
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_x="20dp"
   android:layout_y="160dp"
   android:text="Enter Name:"
   android:textSize="20sp" />

<EditText
   android:id="@+id/Name"
   android:layout_width="150dp"
   android:layout_height="wrap_content"
   android:layout_x="175dp"
   android:layout_y="150dp"
   android:inputType="text"
   android:textSize="20sp" />

<TextView
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_x="20dp"
   android:layout_y="210dp"
   android:text="Enter Marks:"
   android:textSize="20sp" />

<EditText
   android:id="@+id/Marks"
   android:layout_width="150dp"
   android:layout_height="wrap_content"
   android:layout_x="175dp"
```

```
    android:layout_y="200dp"
    android:inputType="number"
    android:textSize="20sp" />

<Button
    android:id="@+id/Insert"
    android:layout_width="150dp"
    android:layout_height="wrap_content"
    android:layout_x="25dp"
    android:layout_y="300dp"
    android:text="Insert"
    android:textSize="30dp" />

<Button
    android:id="@+id/Delete"
    android:layout_width="150dp"
    android:layout_height="wrap_content"
    android:layout_x="200dp"
    android:layout_y="300dp"
    android:text="Delete"
    android:textSize="30dp" />

<Button
    android:id="@+id/Update"
    android:layout_width="150dp"
    android:layout_height="wrap_content"
    android:layout_x="25dp"
    android:layout_y="400dp"
    android:text="Update"
    android:textSize="30dp" />

<Button
    android:id="@+id/View"
    android:layout_width="150dp"
    android:layout_height="wrap_content"
    android:layout_x="200dp"
    android:layout_y="400dp"
    android:text="View"
    android:textSize="30dp" />

<Button
    android:id="@+id/ViewAll"
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:layout_x="100dp"
    android:layout_y="500dp"
    android:text="View All"
    android:textSize="30dp" />
```

</AbsoluteLayout>

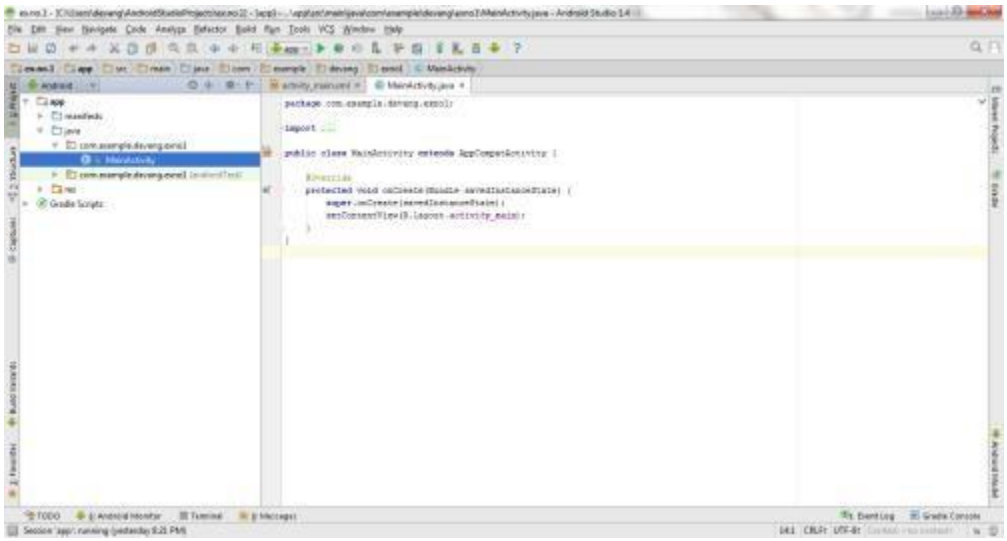
- Now click on **Design** and your application will look as given below.



- So now the designing part is completed.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno5 -> MainActivity.**



- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exno5;

import android.app.Activity;
import android.app.AlertDialog.Builder;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends Activity implements OnClickListener
{
   EditText Rollno,Name,Marks;
   Button Insert,Delete,Update,View,ViewAll;
   SQLiteDatabase db;
   /** Called when the activity is first created. */
   @Override
   public void onCreate(Bundle savedInstanceState)
   {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      Rollno=(EditText)findViewById(R.id.Rollno);
      Name=(EditText)findViewById(R.id.Name);
      Marks=(EditText)findViewById(R.id.Marks);
      Insert=(Button)findViewById(R.id.Insert);
      Delete=(Button)findViewById(R.id.Delete);
      Update=(Button)findViewById(R.id.Update);
      View=(Button)findViewById(R.id.View);
      ViewAll=(Button)findViewById(R.id.ViewAll);

      Insert.setOnClickListener(this);
      Delete.setOnClickListener(this);
      Update.setOnClickListener(this);
      View.setOnClickListener(this);
      ViewAll.setOnClickListener(this);

      // Creating database and table
      db=openOrCreateDatabase("StudentDB", Context.MODE_PRIVATE, null);
      db.execSQL("CREATE TABLE IF NOT EXISTS student(rollno VARCHAR,name VARCHAR,marks VARCHAR );");
   }
   public void onClick(View view)
   {
```

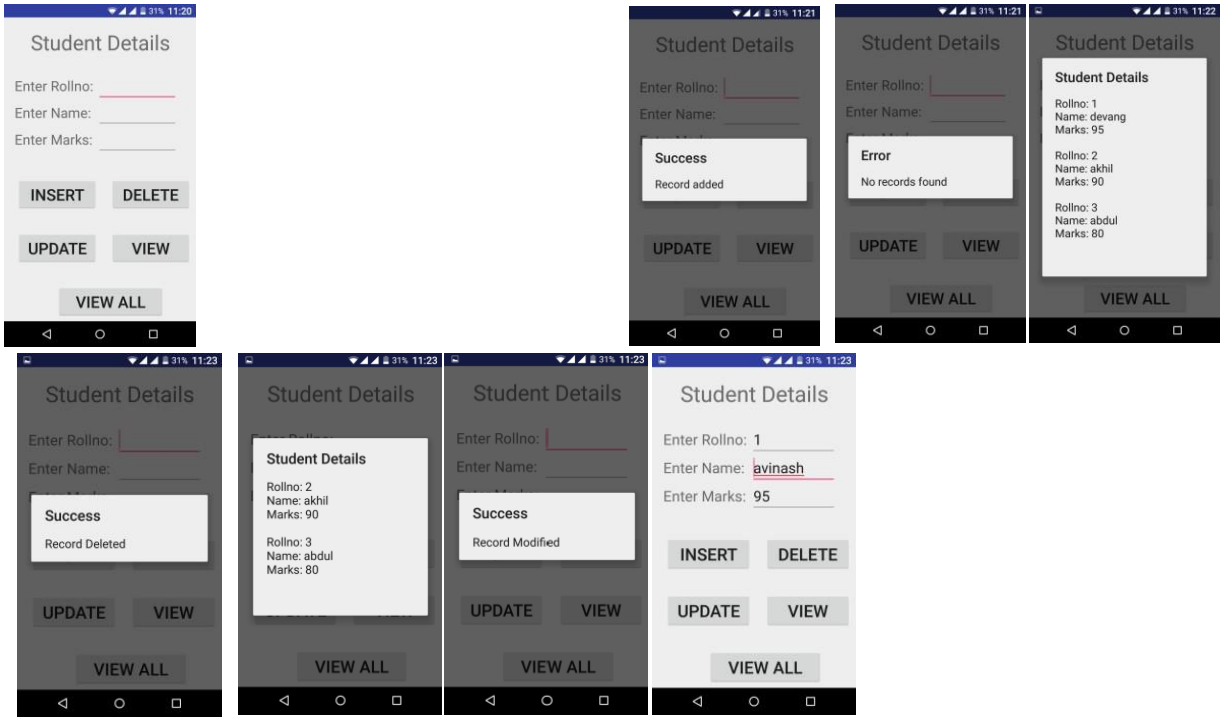
```
// Inserting a record to the Student table
if(view==Insert)
{
   // Checking for empty fields
   if(Rollno.getText().toString().trim().length()==0||
         Name.getText().toString().trim().length()==0||
         Marks.getText().toString().trim().length()==0)
   {
      showMessage("Error", "Please enter all values");
      return;
   }
   db.execSQL("INSERT INTO student VALUES('"+Rollno.getText()+"','"+Name.getText()+
         "','"+Marks.getText()+"');");
   showMessage("Success", "Record added");
   clearText();
}
// Deleting a record from the Student table
if(view==Delete)
{
   // Checking for empty roll number
   if(Rollno.getText().toString().trim().length()==0)
   {
      showMessage("Error", "Please enter Rollno");
      return;
   }
   Cursor c=db.rawQuery("SELECT * FROM student WHERE rollno='"+Rollno.getText()+"'", null);
   if(c.moveToFirst())
   {
      db.execSQL("DELETE FROM student WHERE rollno='"+Rollno.getText()+"'");
      showMessage("Success", "Record Deleted");
   }
   else
   {
      showMessage("Error", "Invalid Rollno");
   }
   clearText();
}
// Updating a record in the Student table
if(view==Update)
{
   // Checking for empty roll number
   if(Rollno.getText().toString().trim().length()==0)
   {
      showMessage("Error", "Please enter Rollno");
      return;
   }
   Cursor c=db.rawQuery("SELECT * FROM student WHERE rollno='"+Rollno.getText()+"'", null);
   if(c.moveToFirst()) {
```

```
            db.execSQL("UPDATE student SET name='" + Name.getText() + "',marks='" + Marks.getText() +
                  "' WHERE rollno='"+Rollno.getText()+"'");
            showMessage("Success", "Record Modified");
        }
        else {
            showMessage("Error", "Invalid Rollno");
        }
        clearText();
    }
    // Display a record from the Student table
    if(view==View)
    {
        // Checking for empty roll number
        if(Rollno.getText().toString().trim().length()==0)
        {
            showMessage("Error", "Please enter Rollno");
            return;
        }
        Cursor c=db.rawQuery("SELECT * FROM student WHERE rollno='"+Rollno.getText()+"'", null);
        if(c.moveToFirst())
        {
            Name.setText(c.getString(1));
            Marks.setText(c.getString(2));
        }
        else
        {
            showMessage("Error", "Invalid Rollno");
            clearText();
        }
    }
    // Displaying all the records
    if(view==ViewAll)
    {
        Cursor c=db.rawQuery("SELECT * FROM student", null);
        if(c.getCount()==0)
        {
            showMessage("Error", "No records found");
            return;
        }
        StringBuffer buffer=new StringBuffer();
        while(c.moveToNext())
        {
            buffer.append("Rollno: "+c.getString(0)+"\n");
            buffer.append("Name: "+c.getString(1)+"\n");
            buffer.append("Marks: "+c.getString(2)+"\n\n");
        }
        showMessage("Student Details", buffer.toString());
    }
```

```
    }
    public void showMessage(String title,String message)
    {
       Builder builder=new Builder(this);
       builder.setCancelable(true);
       builder.setTitle(title);
       builder.setMessage(message);
       builder.show();
    }
    public void clearText()
    {
       Rollno.setText("");
       Name.setText("");
       Marks.setText("");
       Rollno.requestFocus();
    }
}
```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus a Simple Android Application that makes use of Database is developed and executed successfully.

EXPERIMENT 10

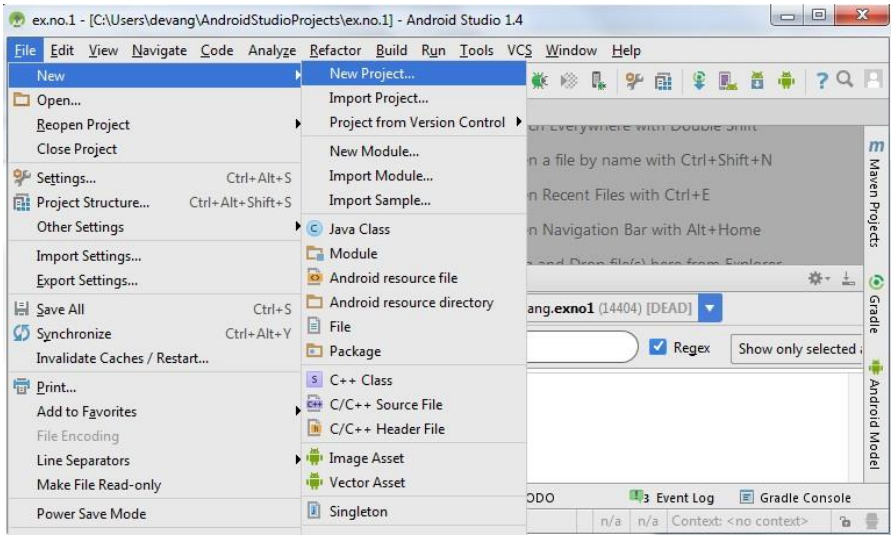
Aim:

To develop a Android Application that creates Alarm Clock.

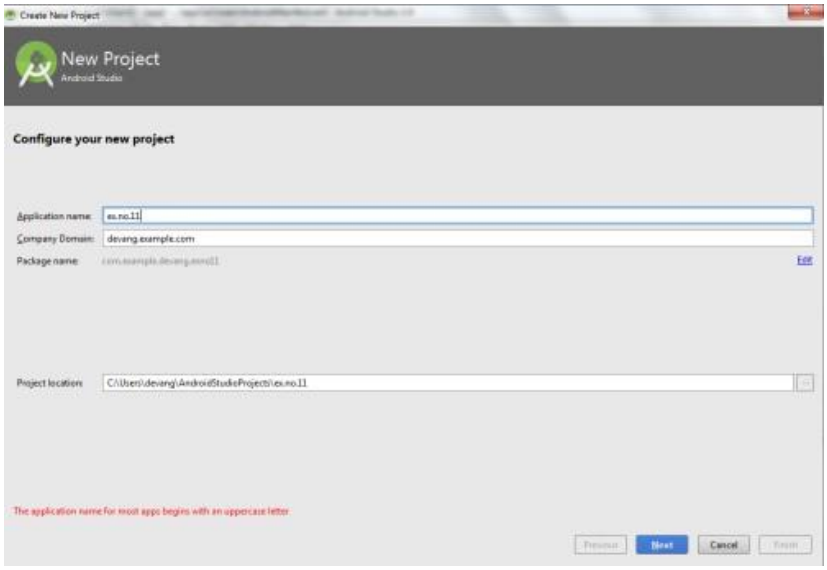
Procedure:

Creating a New project:

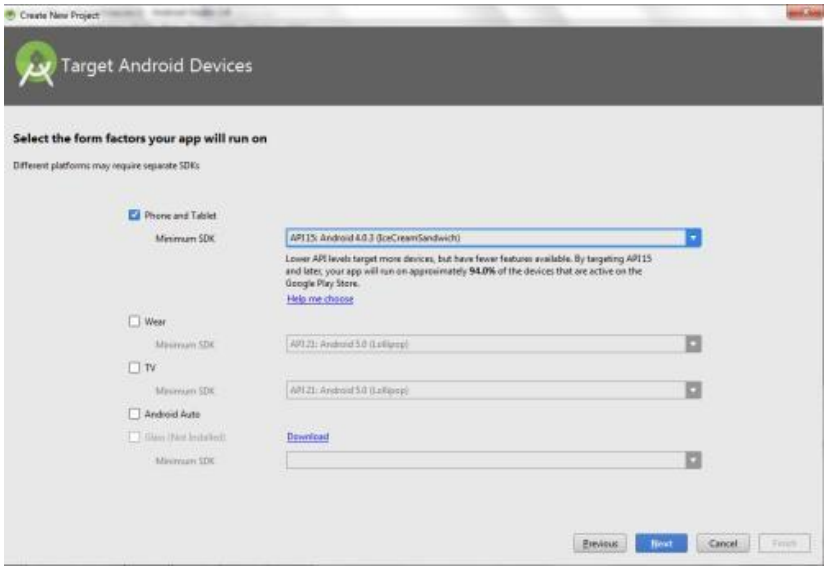
- Open Android Studio and then click on **File -> New -> New project.**



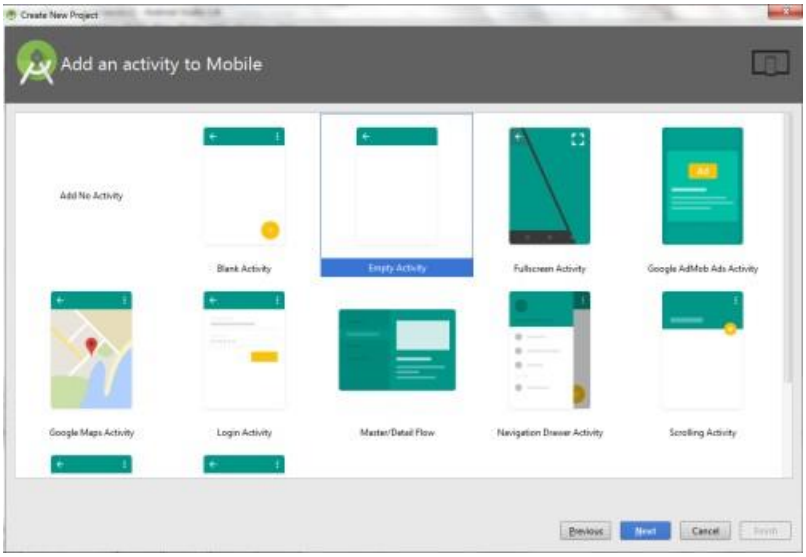
- Then type the Application name as "**ex.no.11"** and click **Next.**



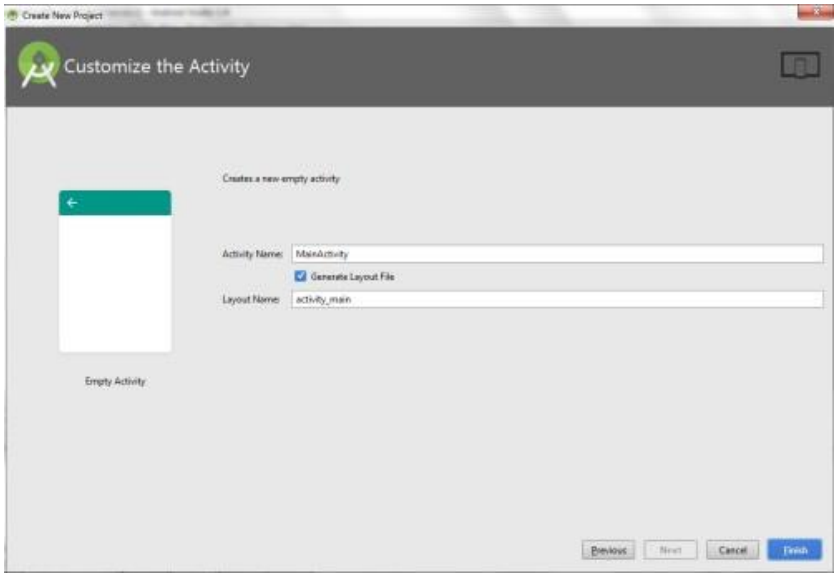
- Then select the **Minimum SDK** as shown below and click **Next**.



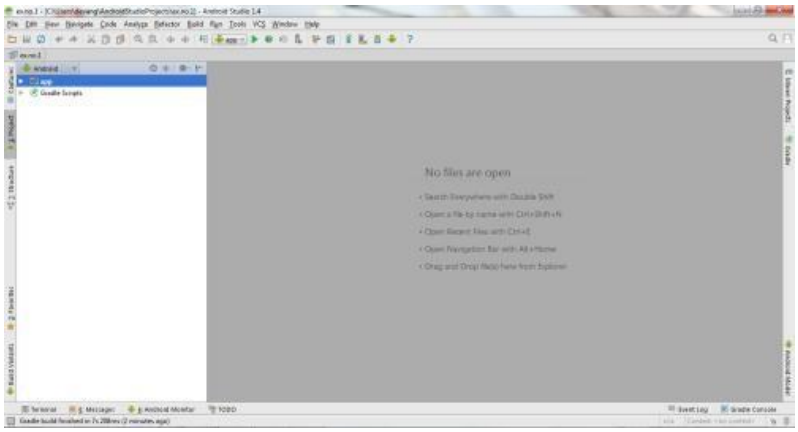
- Then select the **Empty Activity** and click **Next.**



- Finally click F**inish**.

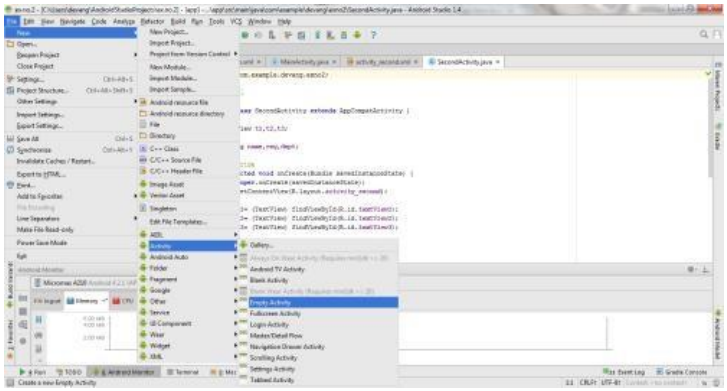


- It will take some time to build and load the project.
- After completion it will look as given below.

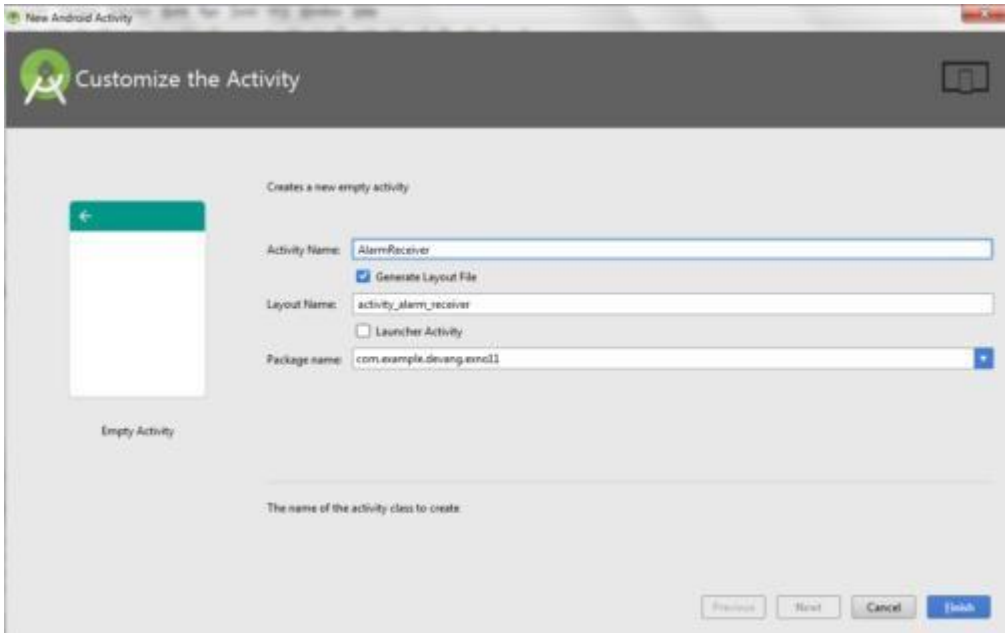


Creating Second Activity for the Android Application:

- Click on **File -> New -> Activity -> Empty Activity.**



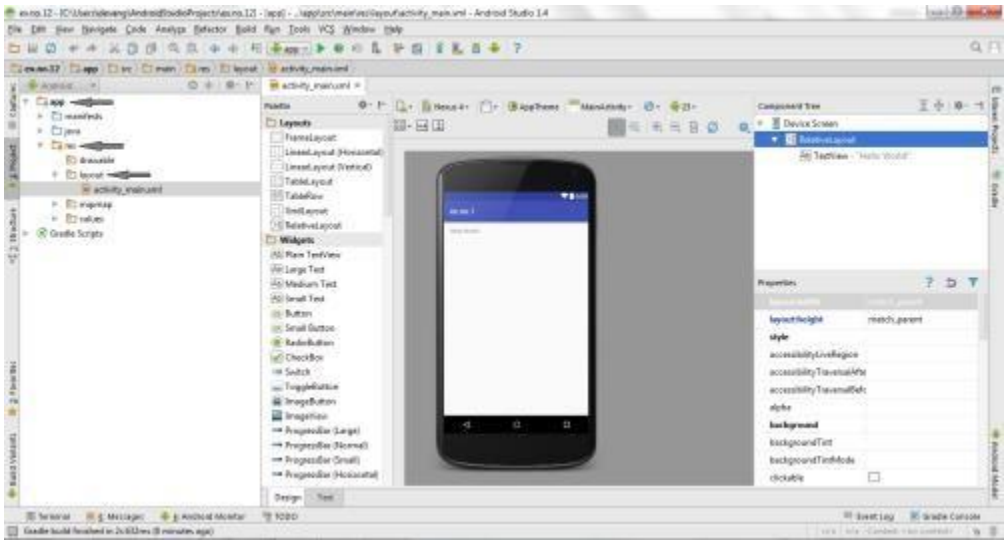
- Type the Activity Name as **AlarmReceiver** and click **Finish** button.



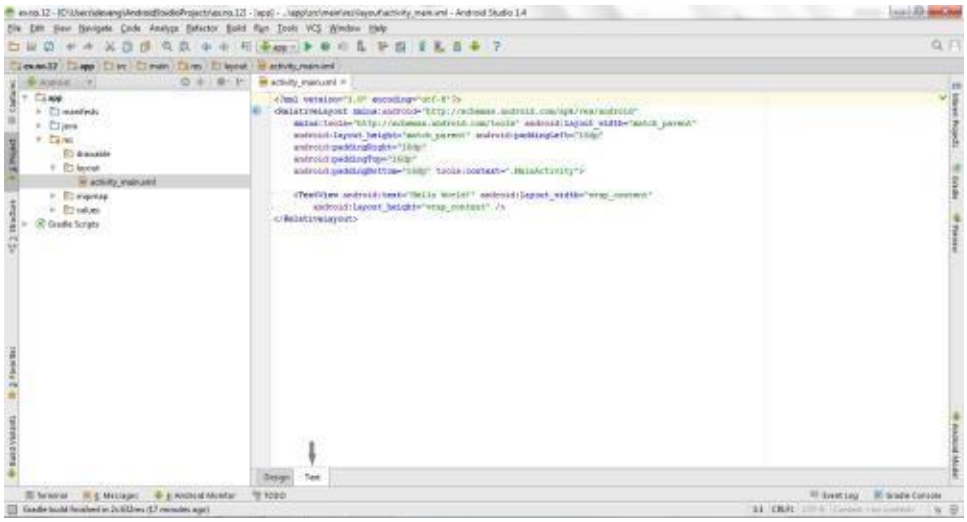
- Thus Second Activity For the application is created.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml.**



- Now click on **Text** as shown below.

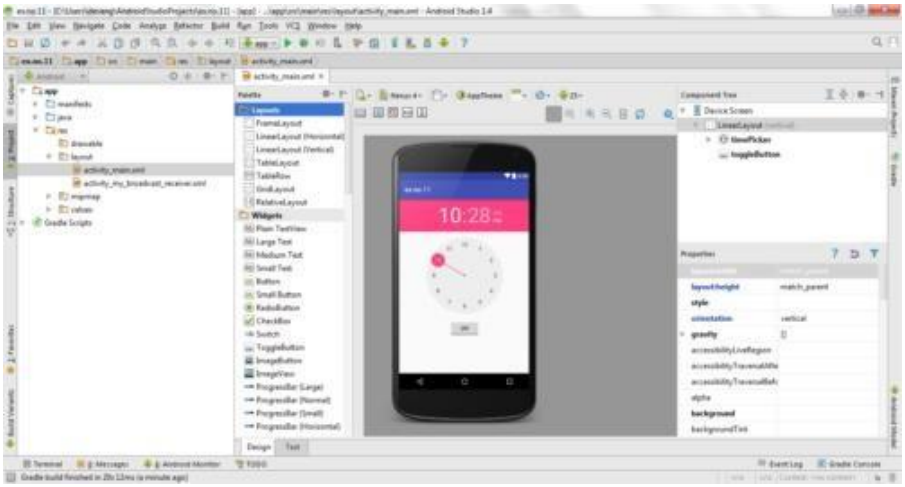


- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

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

   <TimePicker
      android:id="@+id/timePicker"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_gravity="center" />

   <ToggleButton
      android:id="@+id/toggleButton"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_gravity="center"
      android:layout_margin="20dp"
      android:checked="false"
      android:onClick="OnToggleClicked" />

</LinearLayout>
```

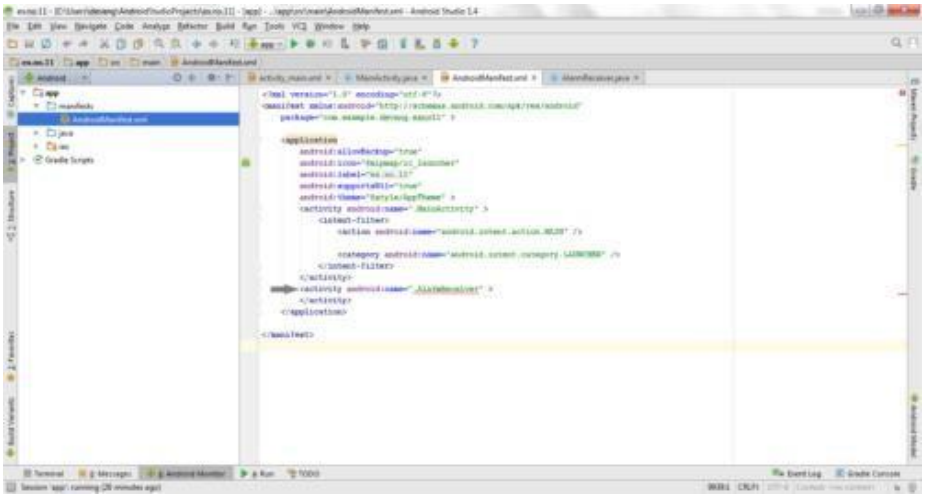
- Now click on **Design** and your application will look as given below.



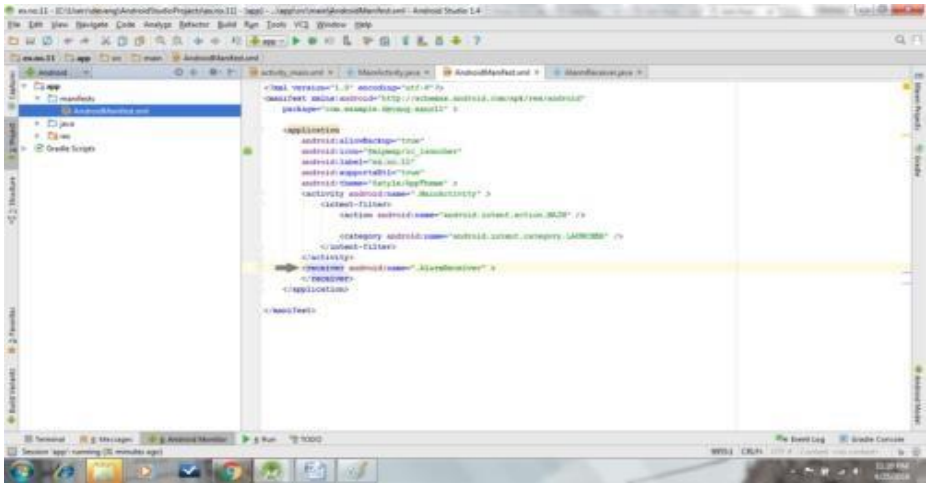
- So now the designing part is completed.

Changes in Manifest for the Android Application:

- Click on **app -> manifests -> AndroidManifest.xml**



- Now change the **activity tag** to **receiver tag** in the AndroidManifest.xml file as shown below



Code for AndroidManifest.xml:

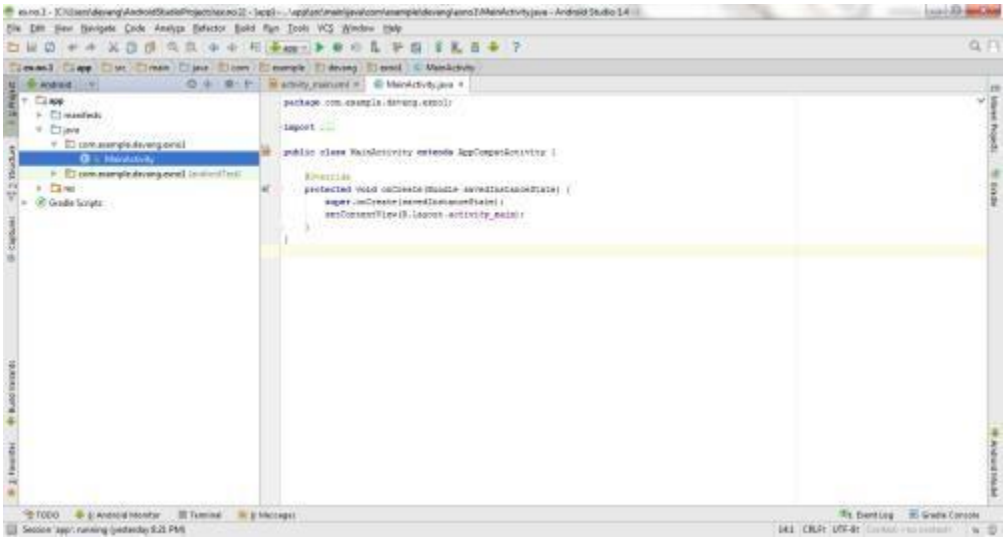
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.example.exno11" >
   <application
      android:allowBackup="true"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:supportsRtl="true"
      android:theme="@style/AppTheme" >
      <activity android:name=".MainActivity" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
      <receiver android:name=".AlarmReceiver" >
      </receiver>
   </application>
</manifest>
```

- So now the changes are done in the Manifest.

Java Coding for the Android Application:

Java Coding for Main Activity:

- Click on **app -> java -> com.example.exno11 -> MainActivity.**



- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

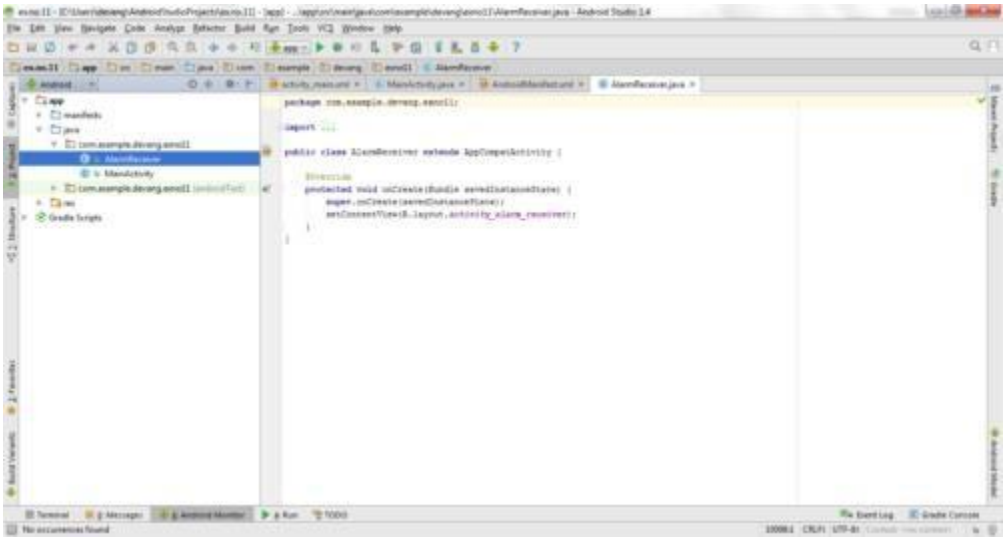
```
package com.example.exno11;

import android.app.AlarmManager;
import android.app.PendingIntent;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.TimePicker;
import android.widget.Toast;
import android.widget.ToggleButton;

import java.util.Calendar;

public class MainActivity extends AppCompatActivity
{
   TimePicker alarmTimePicker;
   PendingIntent pendingIntent;
   AlarmManager alarmManager;

   @Override
   protected void onCreate(Bundle savedInstanceState)
   {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      alarmTimePicker = (TimePicker) findViewById(R.id.timePicker);
```

```
        alarmManager = (AlarmManager) getSystemService(ALARM_SERVICE);
    }
    public void OnToggleClicked(View view)
    {
        long time;
        if (((ToggleButton) view).isChecked())
        {
            Toast.makeText(MainActivity.this, "ALARM ON", Toast.LENGTH_SHORT).show();
            Calendar calendar = Calendar.getInstance();
            calendar.set(Calendar.HOUR_OF_DAY,  alarmTimePicker.getCurrentHour());
            calendar.set(Calendar.MINUTE,  alarmTimePicker.getCurrentMinute());
            Intent intent = new Intent(this, AlarmReceiver.class);
            pendingIntent = PendingIntent.getBroadcast(this, 0, intent, 0);

            time=(calendar.getTimeInMillis()-(calendar.getTimeInMillis()%60000));
            if(System.currentTimeMillis()>time)
            {
                if (calendar.AM_PM == 0)
                    time = time + (1000*60*60*12);
                else
                    time = time + (1000*60*60*24);
            }
            alarmManager.setRepeating(AlarmManager.RTC_WAKEUP, time, 10000, pendingIntent);
        }
        else
        {
            alarmManager.cancel(pendingIntent);
            Toast.makeText(MainActivity.this, "ALARM OFF", Toast.LENGTH_SHORT).show();
        }
    }
}
```

- So now the Coding part of Main Activity is completed.

Java Coding for Alarm Receiver:

- Click on **app -> java -> com.example.exno11 -> AlarmReceiver.**



- Then delete the code which is there and type the code as given below.

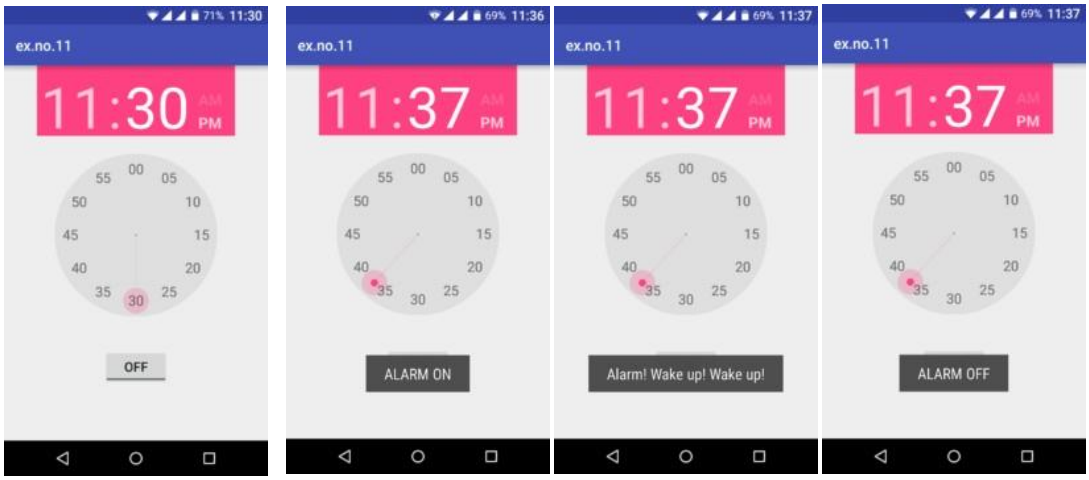
Code for AlarmReceiver.java:

```
package com.example.exno11;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.media.Ringtone;
import android.media.RingtoneManager;
import android.net.Uri;
import android.widget.Toast;

public class AlarmReceiver extends BroadcastReceiver
{
   @Override
   public void onReceive(Context context, Intent intent)
   {
      Toast.makeText(context, "Alarm! Wake up! Wake up!", Toast.LENGTH_LONG).show();
      Uri alarmUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_ALARM);
      if (alarmUri == null)
      {
          alarmUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
      }
      Ringtone ringtone = RingtoneManager.getRingtone(context, alarmUri);
      ringtone.play();
   }
}
```

- So now the Coding part of Alarm Receiver is also completed.

- Now run the application to see the output.

Output:



Result:

Thus Android Application that creates Alarm Clock is developed and executed successfully.

EXPERIMENT 11

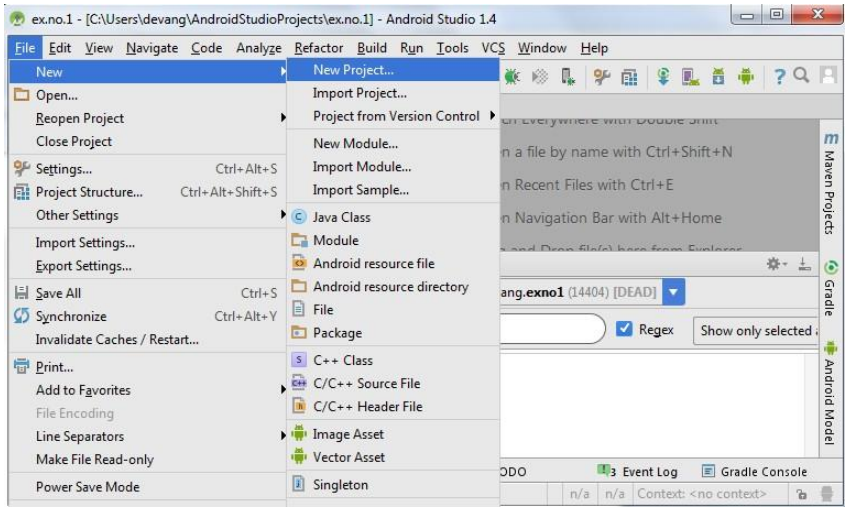
Aim:

To develop a Android Application that creates an alert upon receiving a message.

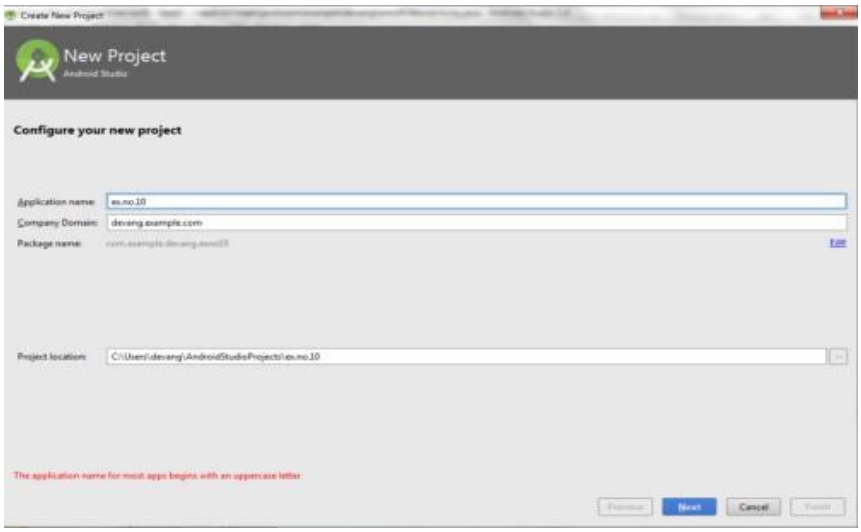
Procedure:

Creating a New project:

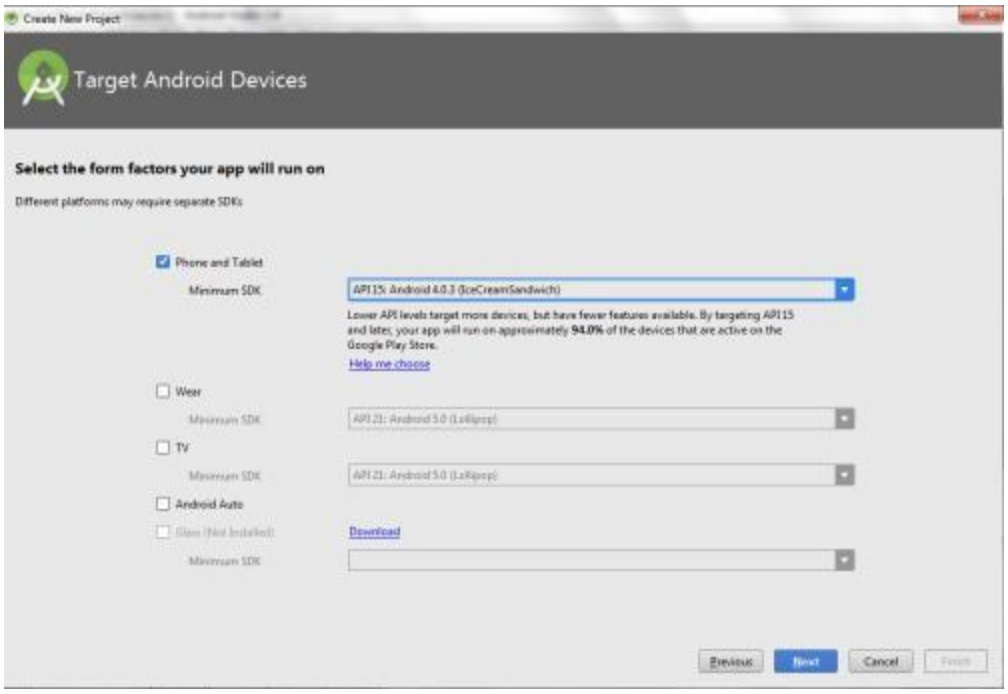
- Open Android Studio and then click on **File -> New -> New project.**



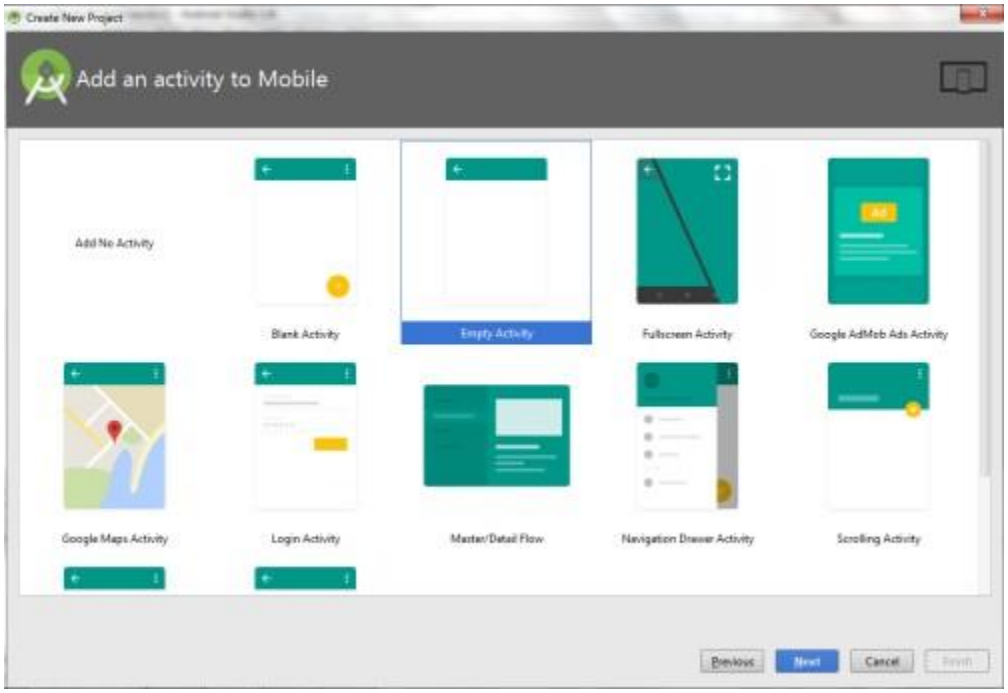
- Then type the Application name as "**ex.no.10"** and click **Next.**



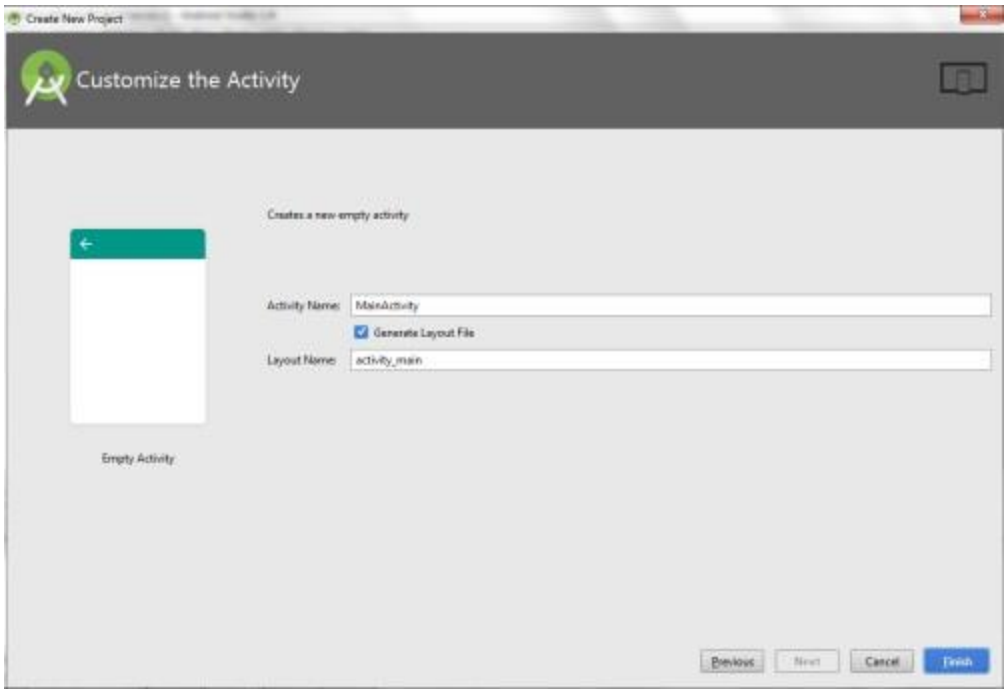
- Then select the **Minimum SDK** as shown below and click **Next**.



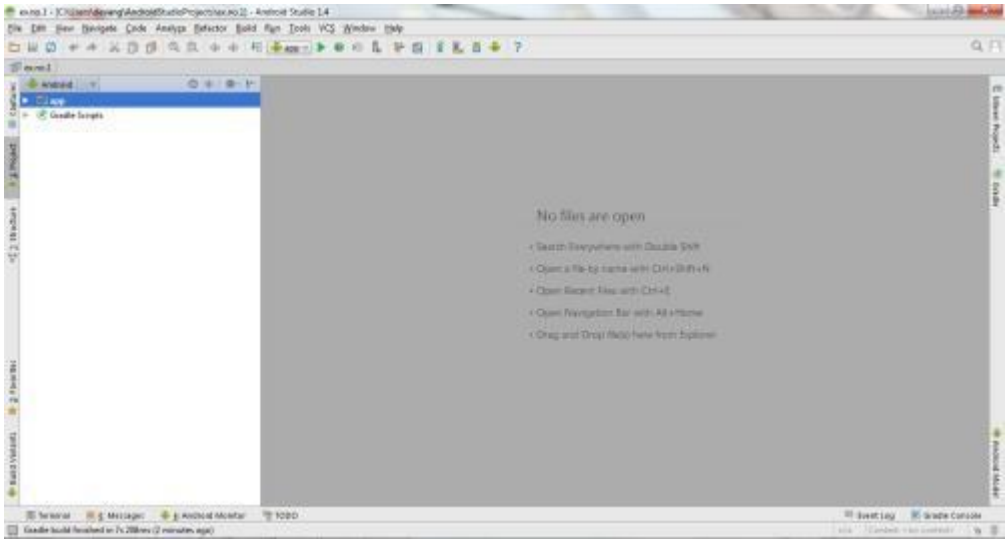
- Then select the **Empty Activity** and click **Next.**



- Finally click F**inish**.

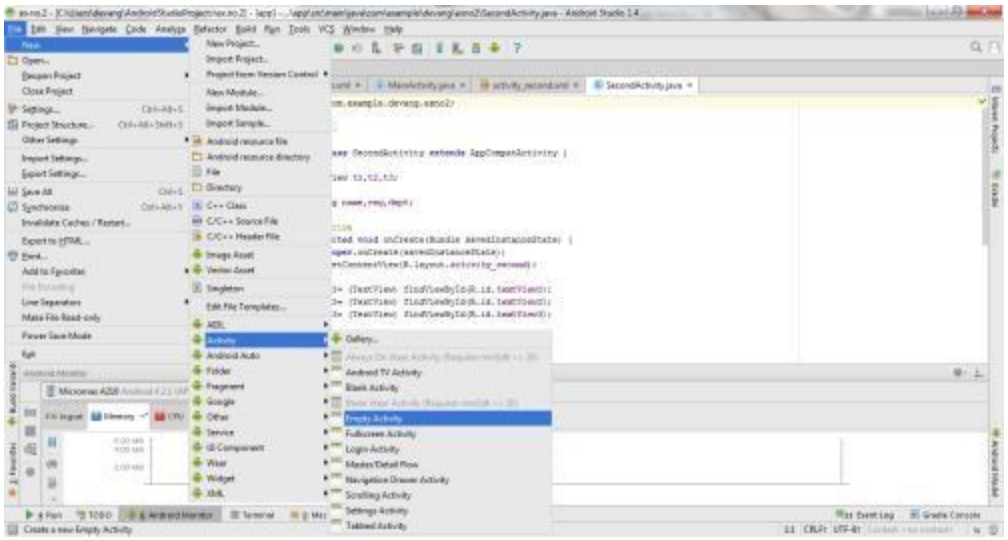


- It will take some time to build and load the project.
- After completion it will look as given below.

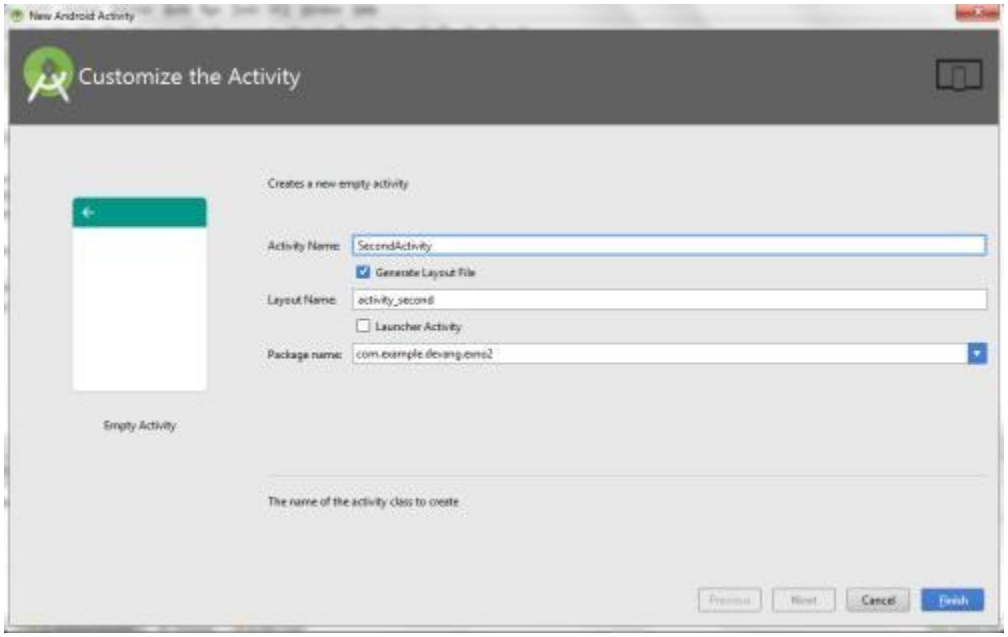


Creating Second Activity for the Android Application:

- Click on **File -> New -> Activity -> Empty Activity.**



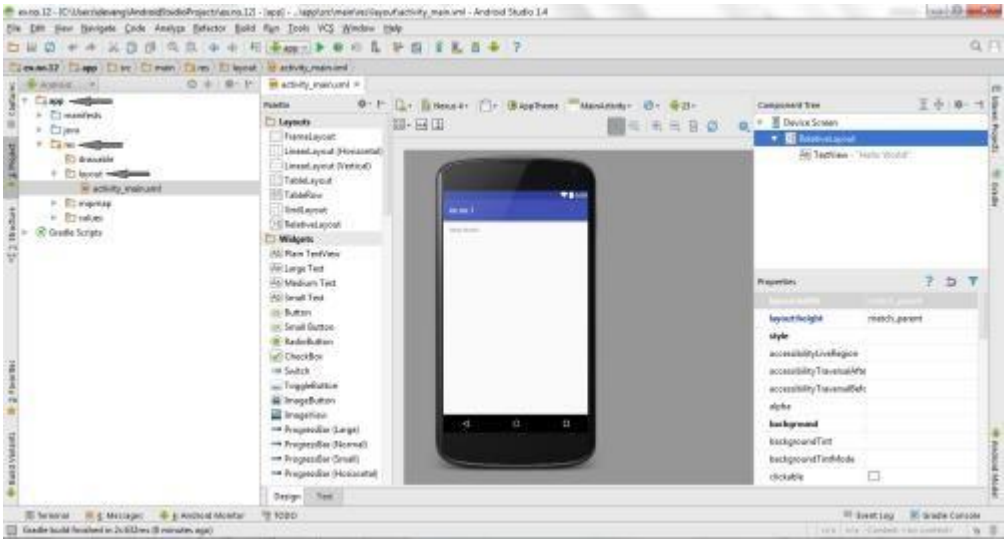
- Type the Activity Name as **SecondActivity** and click **Finish** button.



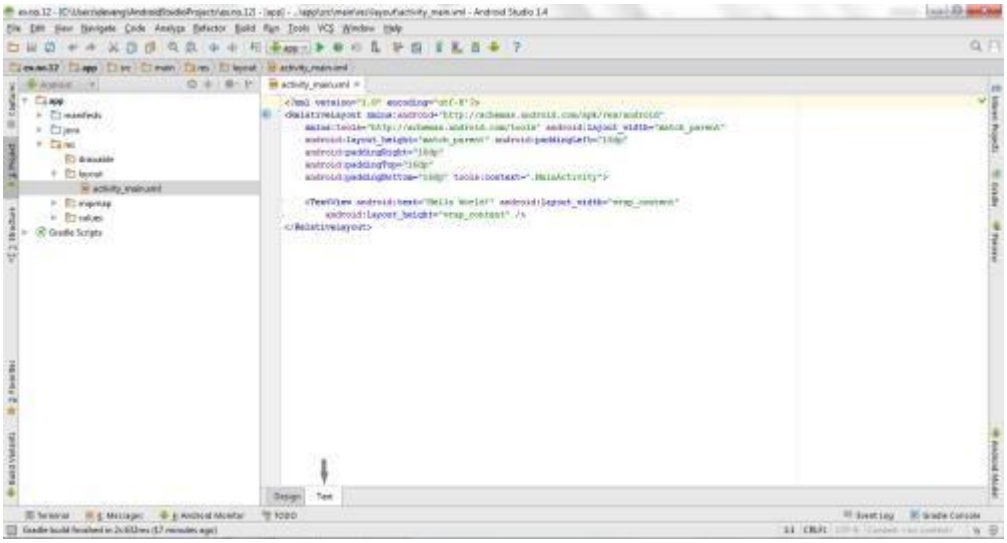
- Thus Second Activity For the application is created.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml.**



- Now click on **Text** as shown below.



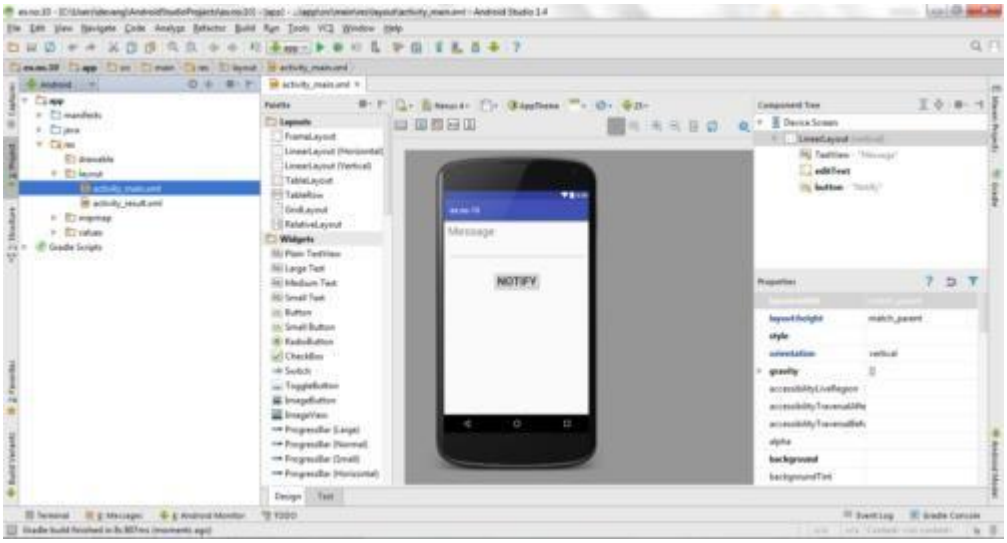
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="10dp"
    android:orientation="vertical">

    <TextView
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Message"
    android:textSize="30sp" />

<EditText
    android:id="@+id/editText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:singleLine="true"
    android:textSize="30sp" />

<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="30dp"
    android:layout_gravity="center"
    android:text="Notify"
    android:textSize="30sp"/>

</LinearLayout>
```

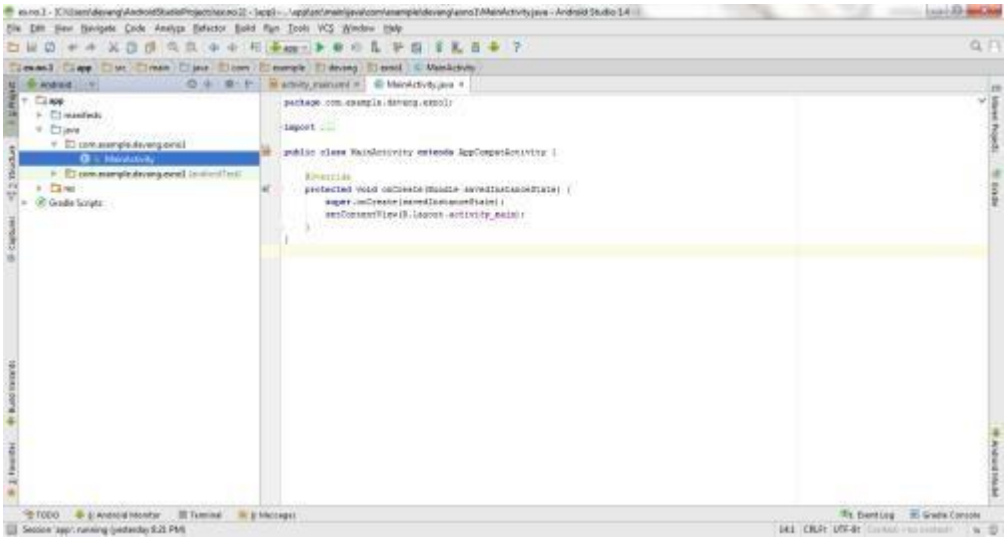
- Now click on **Design** and your application will look as given below.



- So now the designing part is completed.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno10 -> MainActivity.**



- Then delete the code which is there and type the code as given below.

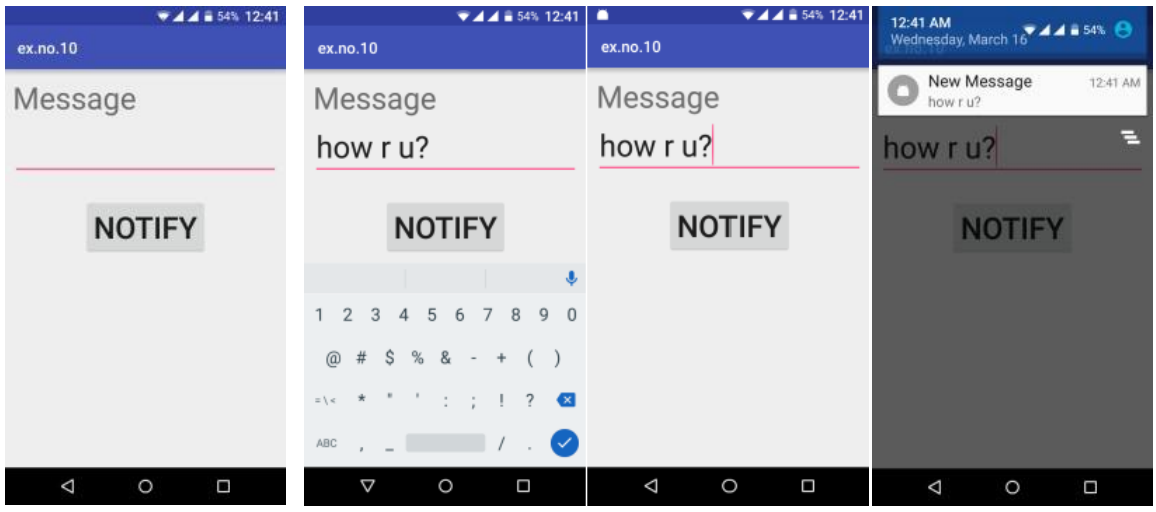
Code for MainActivity.java:

```
package com.example.exno10;

import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent; import
android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

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

      notify= (Button) findViewById(R.id.button);
      e= (EditText) findViewById(R.id.editText);
```

```
        notify.setOnClickListener(new  View.OnClickListener()
        {
           @Override
           public void onClick(View v)
           {
              Intent intent = new Intent(MainActivity.this, SecondActivity.class);
              PendingIntent pending = PendingIntent.getActivity(MainActivity.this, 0, intent, 0);
     Notification noti = new Notification.Builder(MainActivity.this).setContentTitle("New Message")
 .setContentText(e.getText().toString()).setSmallIcon(R.mipmap.ic_launcher).setContentIntent(pending).build();
        NotificationManager manager = (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
              noti.flags |= Notification.FLAG_AUTO_CANCEL;
              manager.notify(0, noti);
           }
        });
    }
}
```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus Android Application that creates an alert upon receiving a message is developed and executed successfully.