DEPARTMENT OF INFORMATION TECHNOLOGY

LABORATORY MANUAL

IT3681-MOBILE APPLICATIONS DEVELOPMENT LABORATORY

Regulation 2021

Year / Semester: III / VI

2023 - 2024 (EVEN SEMESTER)

PREPARED BY

Mrs. C. Aparna

Assistant Professor

IT3681 MOBILE APPLICATIONS DEVELOPMENT LABORATORY

COURSE OBJECTIVES:

The objective of this course is to enable the students to

- Use Flutter/Kotlin multi-platform environment for building cross platform mobile applications.
- Demonstrate the knowledge of different programming techniques and patterns for mobile application development.
- Identify the components and structure of mobile application development frameworks.
- Understand the capabilities and limitations of different platforms.
- Design and develop real-time mobile applications.

LIST OF EXPERIMENTS:

- Study and installation of Flutter/Kotlin multi-platform environment
- Develop an application that uses Widgets, GUI components, Fonts, and Colors.
- Develop a native calculator application.
- Develop a gaming application that uses 2-D animations and gestures.
- Develop a movie rating application (similar to IMDB)
- Develop an application to connect to a web service and to retrieve data with HTTP.
- Develop a simple shopping application.
- Design a web server supporting push notifications.
- Develop an application by integrating Google maps
- Mini Projects involving Flutter/Kotlin multi-platform

TEXTBOOKS:

1. Simone Alessandria, Flutter Projects: A practical project-based guide to building real-world cross-platform mobile applications and games, Packt publishing.
2. Carmine Zaccagnino, Programming Flutter: Native, Cross-Platform Apps the Easy Way (The Pragmatic Programmers), Packt publishing.

REFERENCES

1. Gergely Orosz, Building Mobile Applications at Scale:39 Engineering Challenges
2. Souvik Biswas & Codemagic, Flutter Libraries we love
3. ED Freitas, Daniel Jebaraj, Flutter Succinctly
4. Antonio Leiva, Kotlin for Android Developers Learn Kotlin the easy way while developing an Android Applications

COURSE OUTCOMES:

On successful completion of this course, the student should be able to

CO1: Design and build simple mobile applications supporting multiple platforms.

CO2: Apply various programming techniques and patterns to build mobile applications.

CO3: Build real-time mobile applications for society/environment

CO4: Build gaming and multimedia based mobile applications

CO5: Build AI based mobile applications for society/environment following ethical practices

TABLE OF CONTENTS

S.NO.	DATE	EXPERIMENT TITLE	MARKS (50)	SIGN.
1.		Study and installation of Flutter/Kotlin multi-platform environment		
2.		Develop an application that uses Widgets, GUI components, Fonts, and Colors.		
3.		Develop a native calculator application.		
4.		Develop a gaming application that uses 2-D animations and gestures.		
5.		Develop a movie rating application (similar to IMDB)		
6.		Develop an application to connect to a web service and to retrieve data with HTTP.		
7.		Develop a simple shopping application.		
8.		Design a web server supporting push notifications.		
9.		Develop an application by integrating Google maps		
10.		Mini Projects involving Flutter/Kotlin multi-platform		
		Average		

Ex.No:1 Study and installation of Flutter/Kotlin multi-platform environment

Introduction

In general, developing a mobile application is a complex and challenging task. There are many frameworks available to develop a mobile application. Android provides a native framework based on Java language and iOS provides a native framework based on Objective-C / Swift language.

However, to develop an application supporting both the OSs, we need to code in two different languages using two different frameworks. To help overcome this complexity, there exists mobile frameworks supporting both OS. These frameworks range from simple HTML based hybrid mobile application framework (which uses HTML for User Interface and JavaScript for application logic) to complex language specific framework (which do the heavy lifting of converting code to native code). Irrespective of their simplicity or complexity, these frameworks always have many disadvantages, one of the main drawback being their slow performance.

In this scenario, Flutter – a simple and high performance framework based on Dart language, provides high performance by rendering the UI directly in the operating system's canvas rather than through native framework.

Flutter also offers many ready to use widgets (UI) to create a modern application. These widgets are optimized for mobile environment and designing the application using widgets is as simple as designing HTML.

To be specific, Flutter application is itself a widget. Flutter widgets also supports animations and gestures. The application logic is based on reactive programming. Widget may optionally have a state. By changing the state of the widget, Flutter will automatically (reactive programming) compare the widget's state (old and new) and render the widget with only the necessary changes instead of re-rendering the whole widget.

We shall discuss the complete architecture in the coming chapters.

Features of Flutter

Flutter framework offers the following features to developers −

- Modern and reactive framework.
- Uses Dart programming language and it is very easy to learn.
- Fast development.
- Beautiful and fluid user interfaces.
- Huge widget catalog.
- Runs same UI for multiple platforms.
- High performance application.

Installation in Windows

In this section, let us see how to install *Flutter SDK* and its requirement in a windows system.

Step 1 − Go to URL, https://flutter.dev/docs/get-started/install/windows and download the latest Flutter SDK. As of April 2019, the version is 1.2.1 and the file is flutter_windows_v1.2.1-stable.zip.

Step 2 − Unzip the zip archive in a folder, say C:\flutter\

Step 3 − Update the system path to include flutter bin directory.

Step 4 − Flutter provides a tool, flutter doctor to check that all the requirement of flutter development is met.

flutter doctor

Step 5 − Running the above command will analyze the system and show its report as shown below −

Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, v1.2.1, on Microsoft Windows [Version
10.0.17134.706], locale en-US)
[√] Android toolchain - develop for Android devices (Android SDK version
28.0.3)
[√] Android Studio (version 3.2)
[√] VS Code, 64-bit edition (version 1.29.1)
[!] Connected device
! No devices available
! Doctor found issues in 1 category.

The report says that all development tools are available but the device is not connected. We can fix this by connecting an android device through USB or starting an android emulator.

Step 6 − Install the latest Android SDK, if reported by flutter doctor

Step 7 − Install the latest Android Studio, if reported by flutter doctor

Step 8 − Start an android emulator or connect a real android device to the system.

Step 9 − Install Flutter and Dart plugin for Android Studio. It provides startup template to create new Flutter application, an option to run and debug Flutter application in the Android studio itself, etc.,

- Open Android Studio.
- Click File → Settings → Plugins.
- Select the Flutter plugin and click Install.
- Click Yes when prompted to install the Dart plugin.
- Restart Android studio.

Installation in MacOS

To install Flutter on MacOS, you will have to follow the following steps −

Step 1 − Go to URL, https://flutter.dev/docs/get-started/install/macos and download latest Flutter SDK. As of April 2019, the version is 1.2.1 and the file is flutter_macos_v1.2.1- stable.zip.

Step 2 − Unzip the zip archive in a folder, say /path/to/flutter

Step 3 − Update the system path to include flutter bin directory (in ~/.bashrc file).

> export PATH = "$PATH:/path/to/flutter/bin"

Step 4 − Enable the updated path in the current session using below command and then verify it as well.

source ~/.bashrc

source $HOME/.bash_profile

echo $PATH

Flutter provides a tool, flutter doctor to check that all the requirement of flutter development is met. It is similar to the Windows counterpart.

Step 5 − Install latest XCode, if reported by flutter doctor

Step 6 − Install latest Android SDK, if reported by flutter doctor

Step 7 − Install latest Android Studio, if reported by flutter doctor

Step 8 − Start an android emulator or connect a real android device to the system to develop android application.

Step 9 − Open iOS simulator or connect a real iPhone device to the system to develop iOS application.

Step 10 − Install Flutter and Dart plugin for Android Studio. It provides the startup template to create a new Flutter application, option to run and debug Flutter application in the Android studio itself, etc.,

- Open Android Studio
- Click **Preferences → Plugins**
- Select the Flutter plugin and click Install
- Click Yes when prompted to install the Dart plugin.
- Restart Android studio.

Result:

Thus, installation of Flutter/Kotlin multi-platform environment successfully completed.

Ex.No:2 Develop an application that uses Widgets, GUI components, Fonts, and Colors.

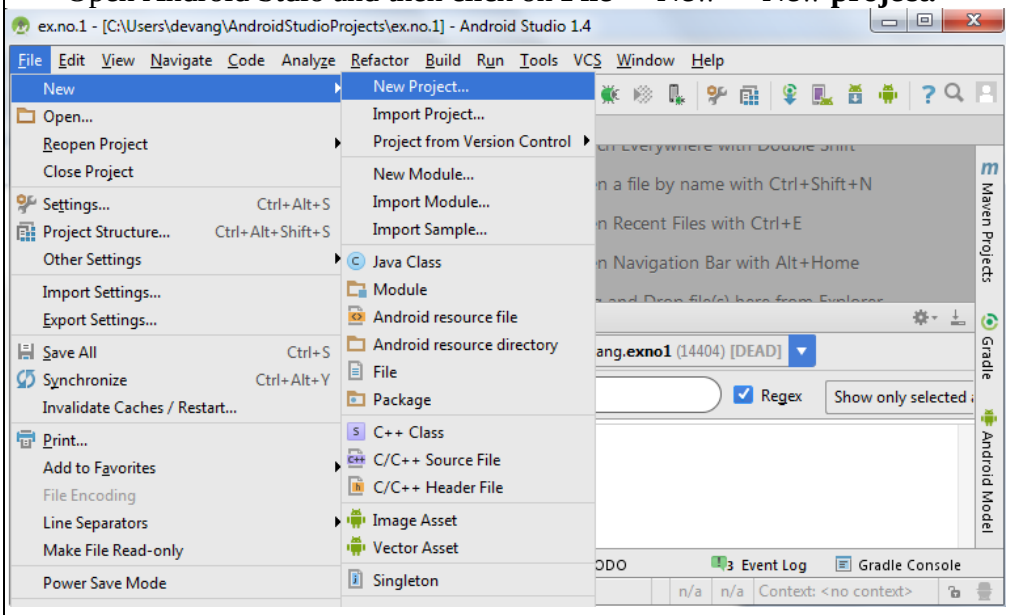
Aim:

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

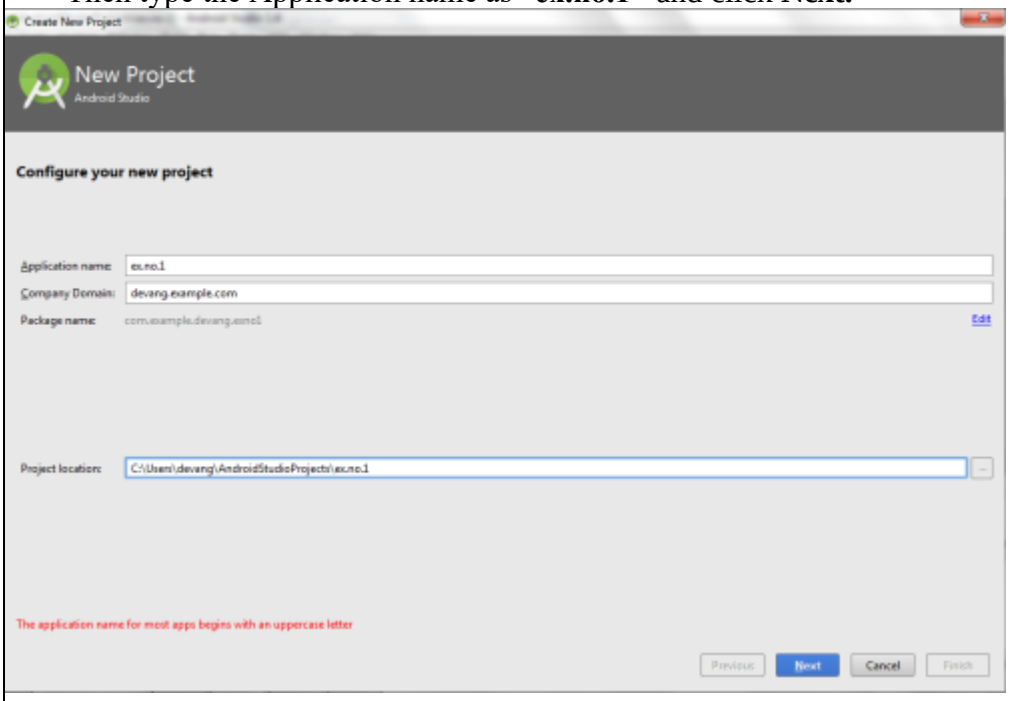
Procedure:

Creating a New project:

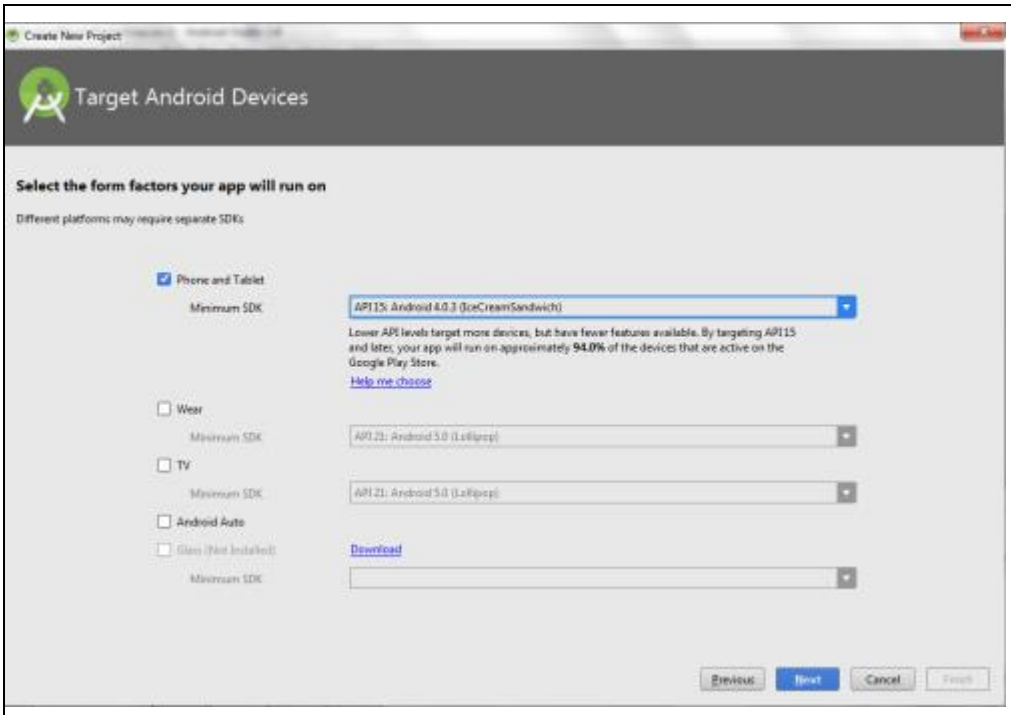
- Open Android Stdio and then click on **File -> New -> New project.**



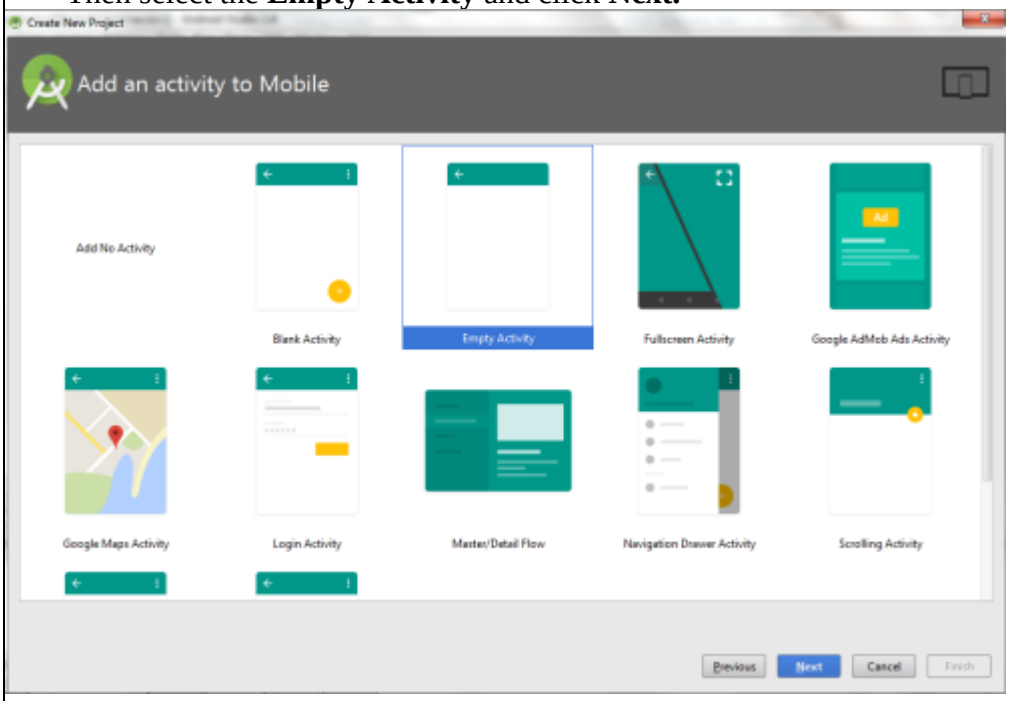
- Then type the Application name as "**ex.no.1"** and click **Next.**



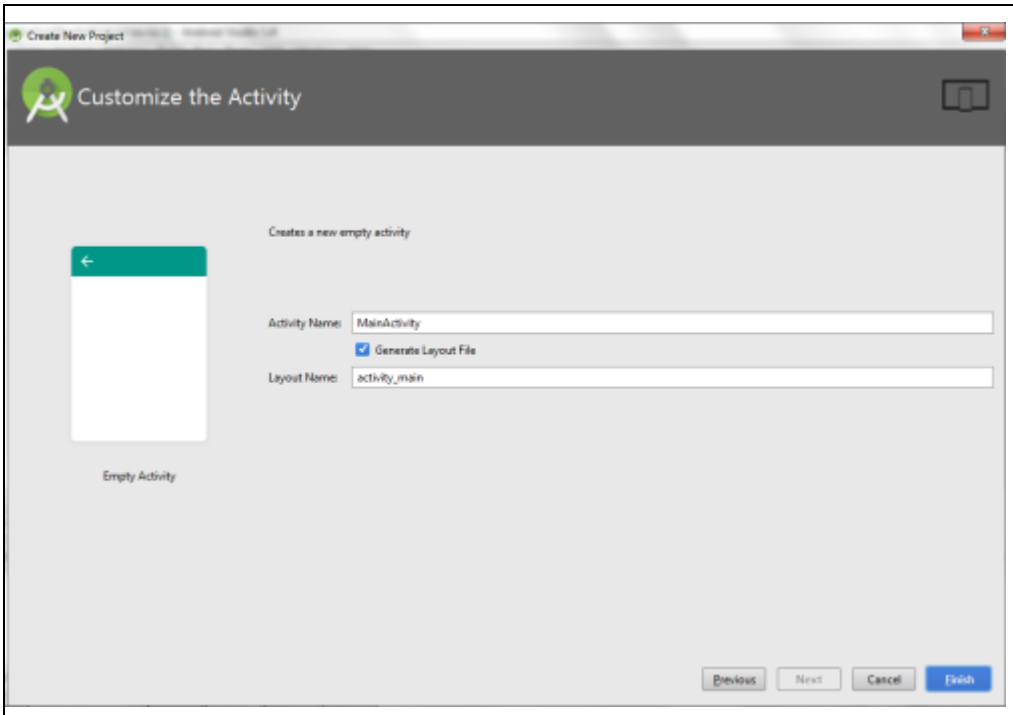
- Then select the **Minimum SDK** as shown below and click **Next**.



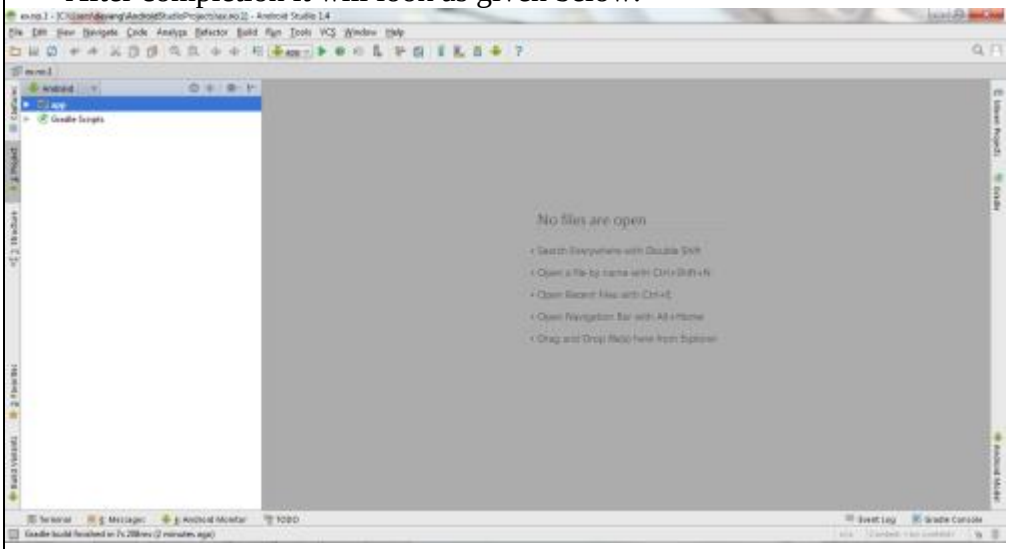
- Then select the **Empty Activity** and click **Next.**



- Finally click F**inish**.

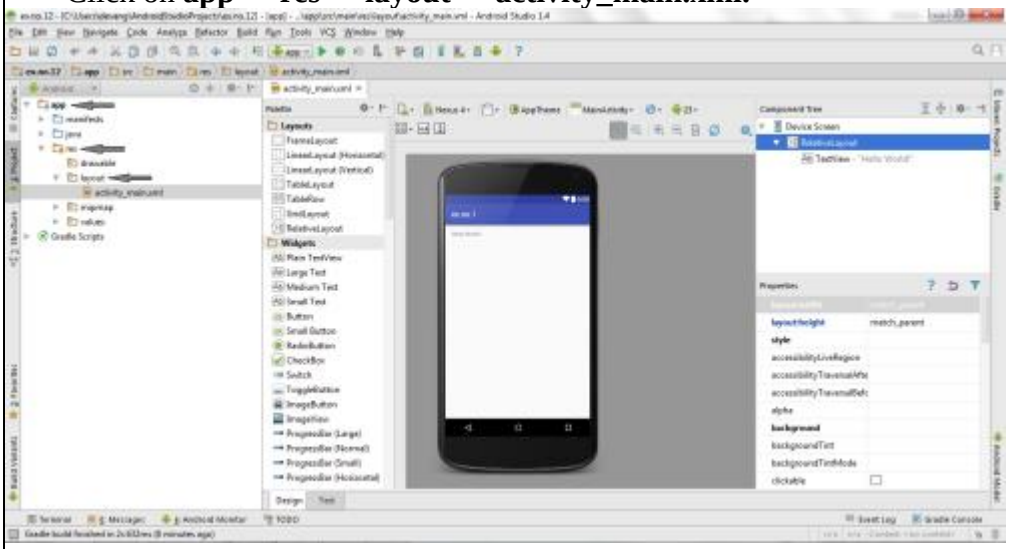


- It will take some time to build and load the project.
- After completion it will look as given below.

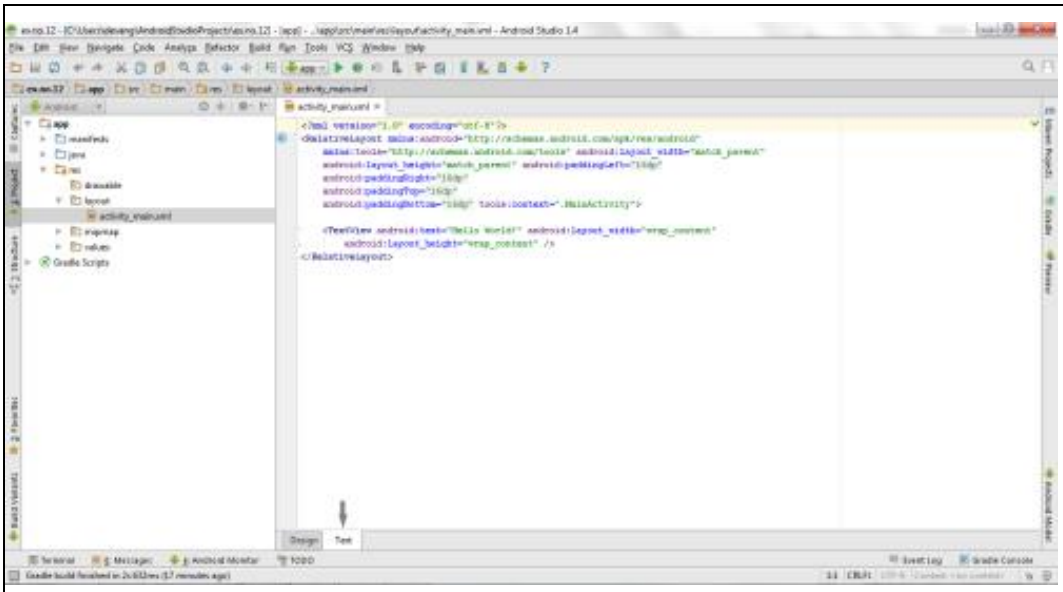


Designing layout for the Android Application:

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



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



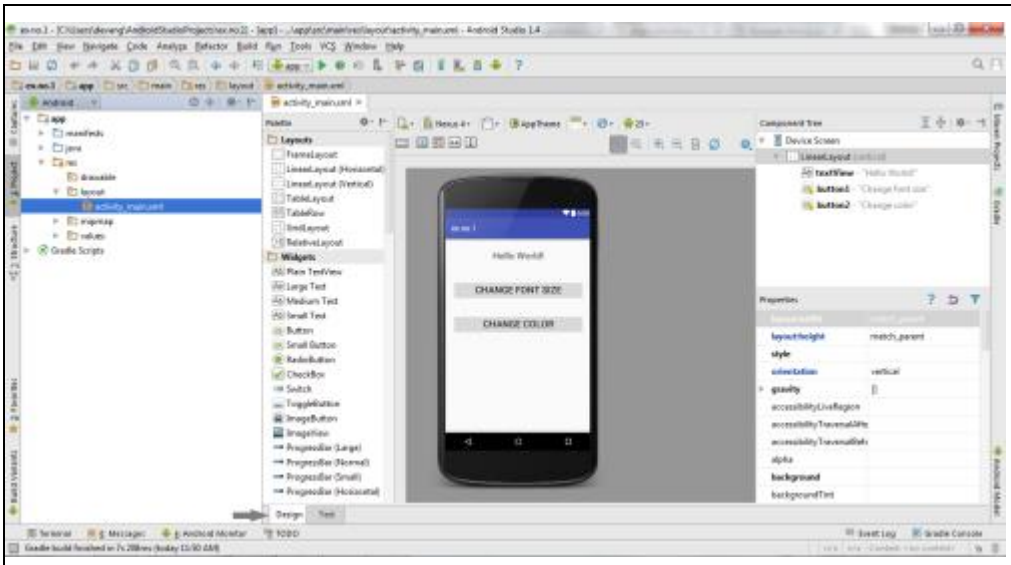
- Then delete the code which is there and type the code as given below.

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>
```

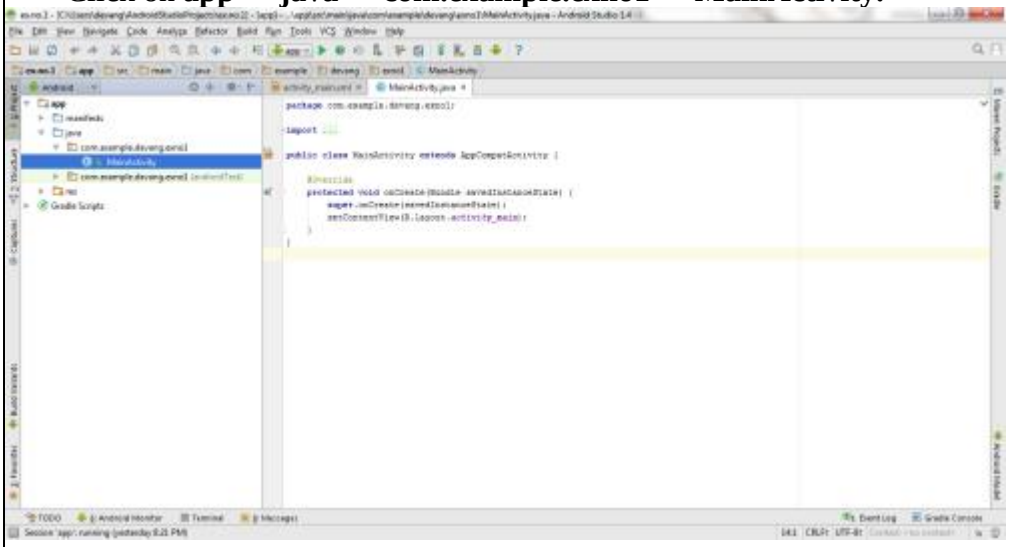
- Now click on Design and your application will look as given below.



- 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:

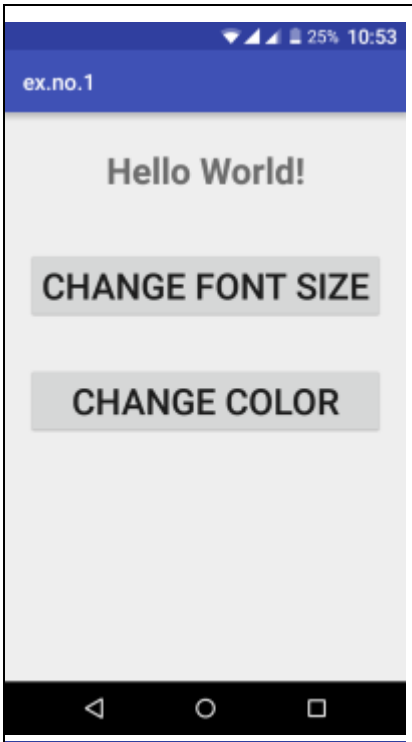
?

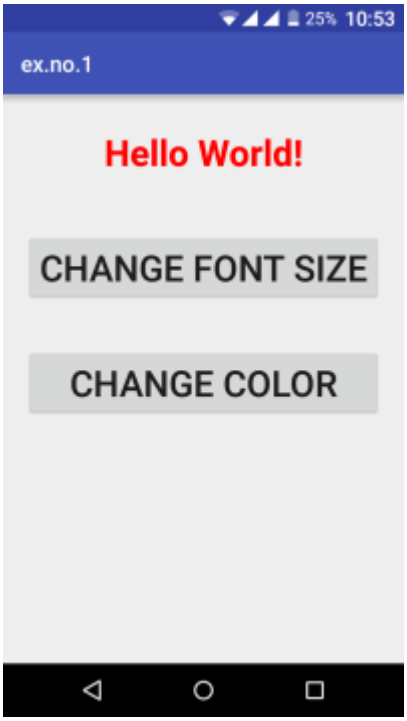
```
package com.example.exno1;

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);
```

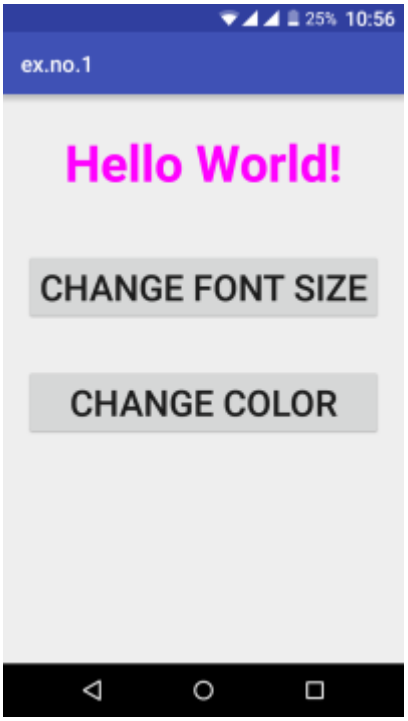
```
        final TextView t= (TextView) findViewById(R.id.textView);
        Button b1= (Button) findViewById(R.id.button1);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                t.setTextSize(font);
                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;
            }
        });
    }
}
```

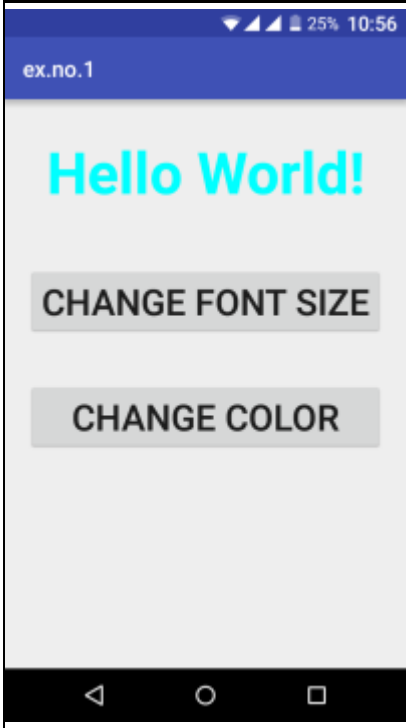
- So now the Coding part is also completed.
- Now run the application to see the output.

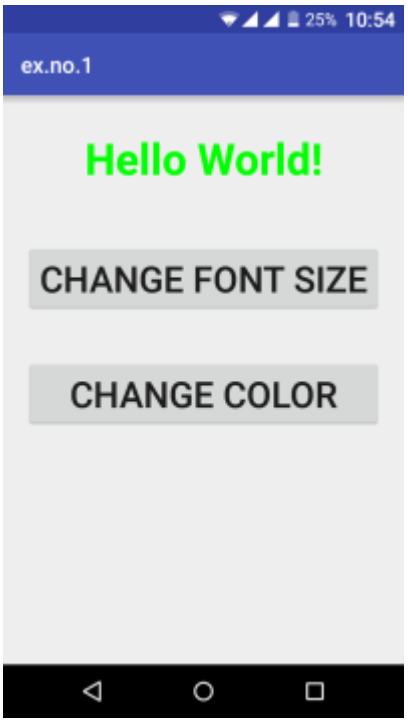
Output:











Result:

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

Ex.No:3 **Develop a native calculator application.**

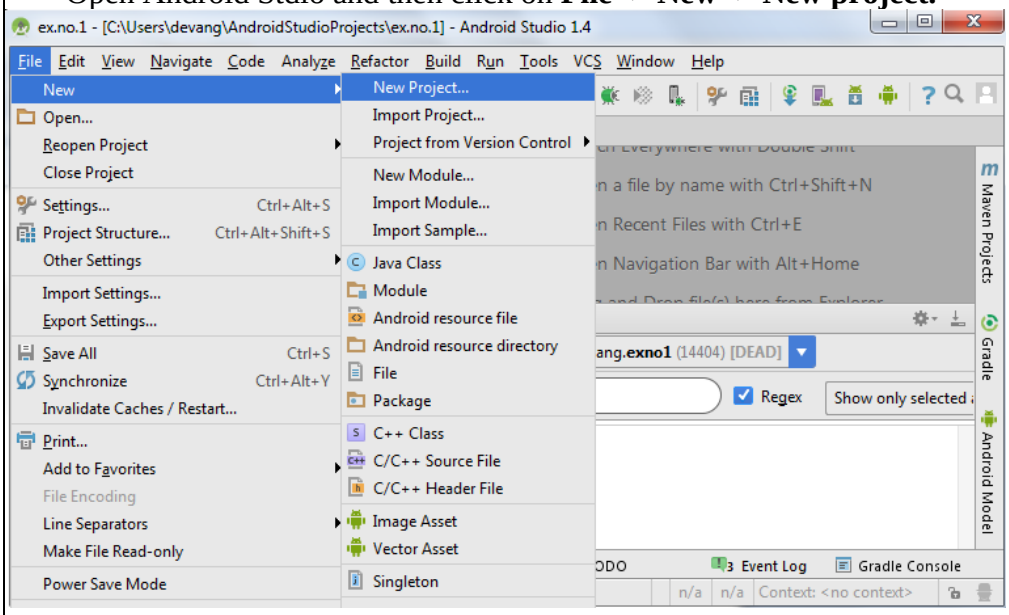
Aim:

To develop a Simple Android Application for Native Calculator.

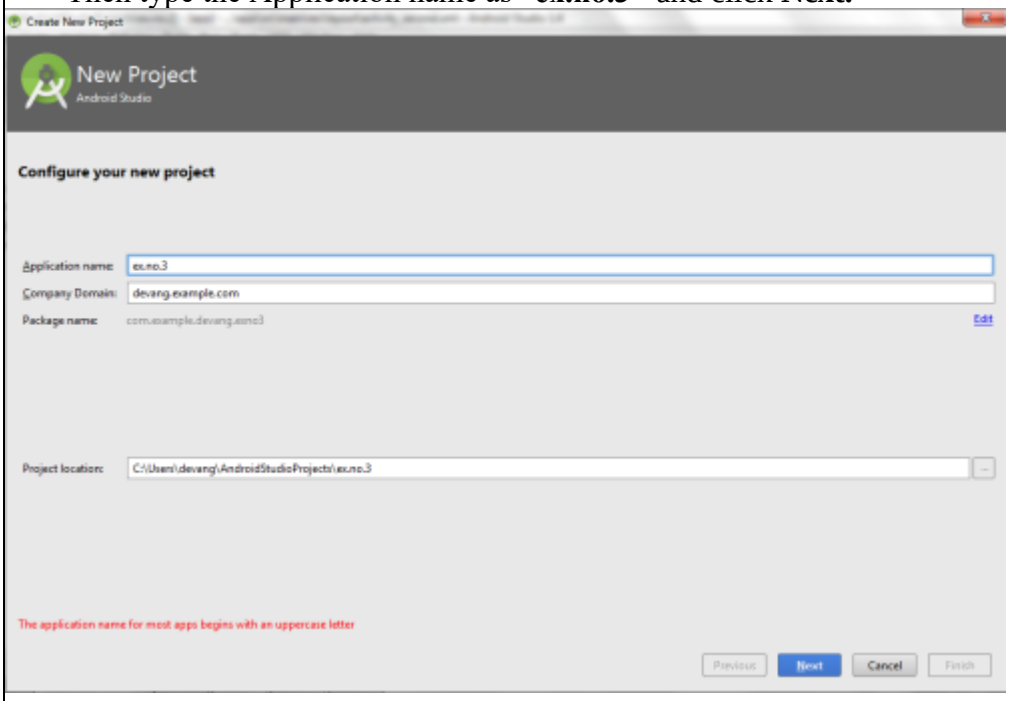
Procedure:

Creating a New project:

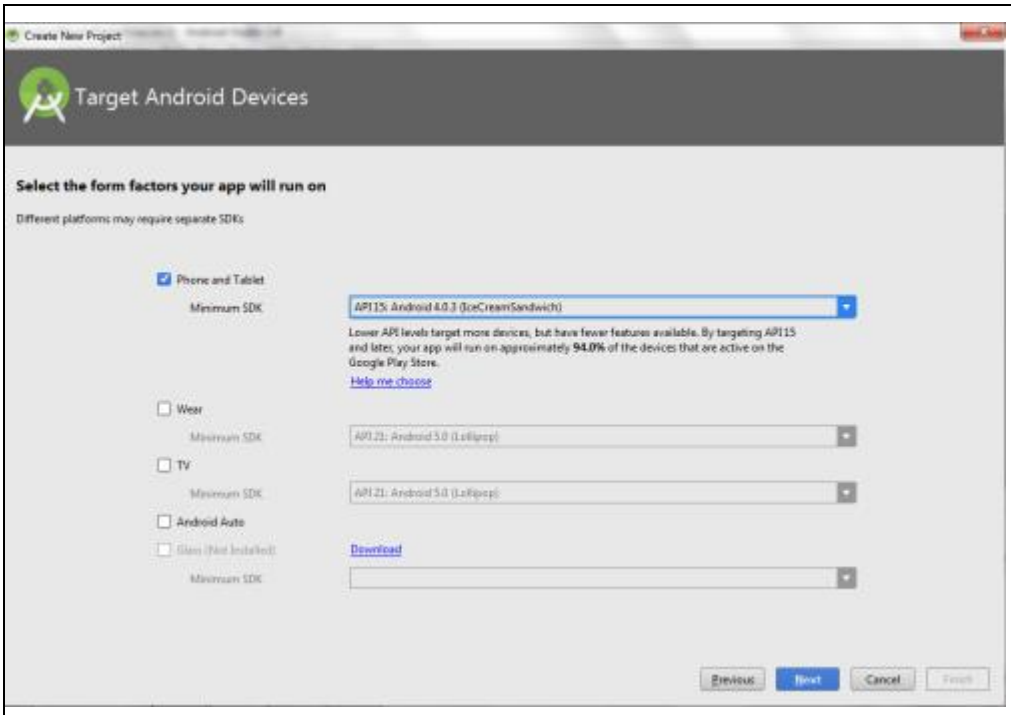
- Open Android Stdio and then click on **File -> New -> New project.**



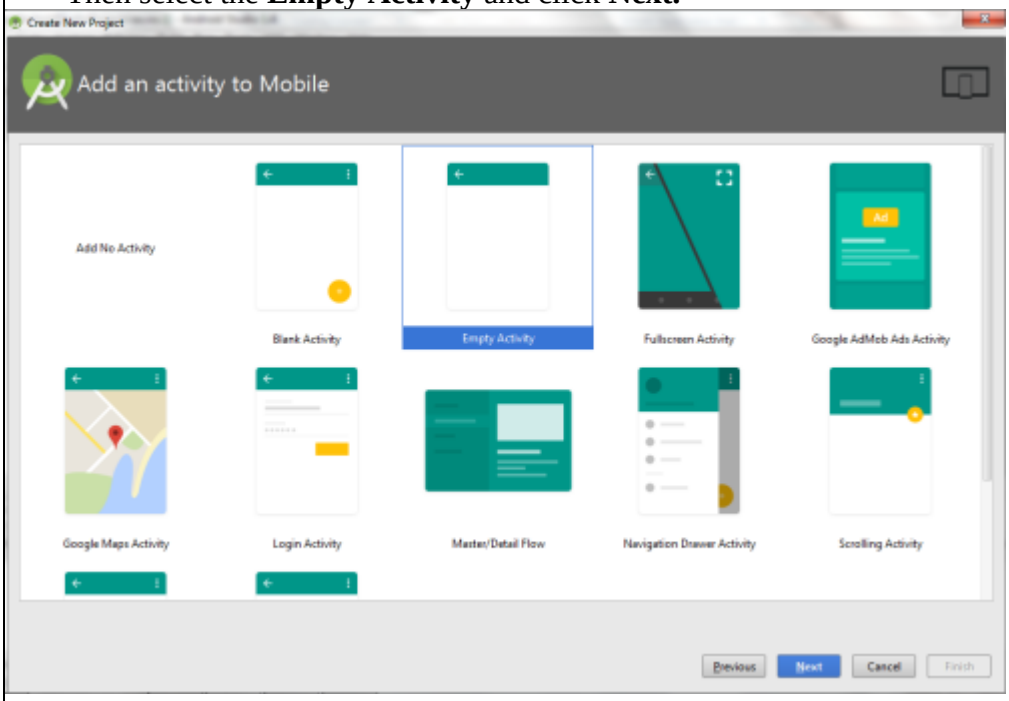
- Then type the Application name as "**ex.no.3**" and click **Next.**



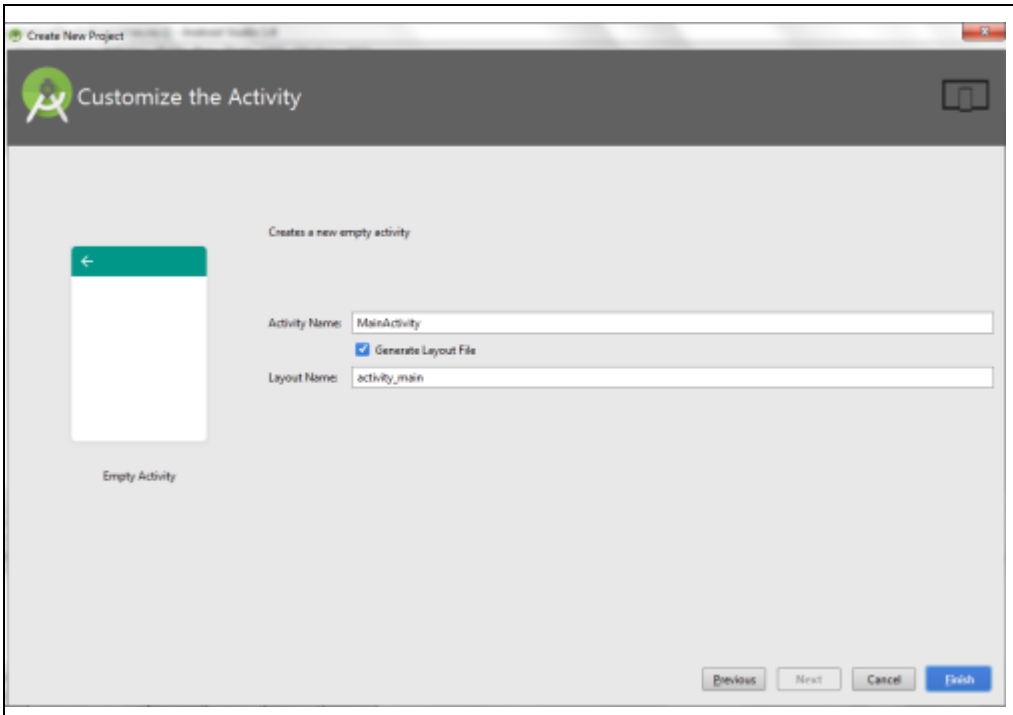
- Then select the **Minimum SDK** as shown below and click **Next**.



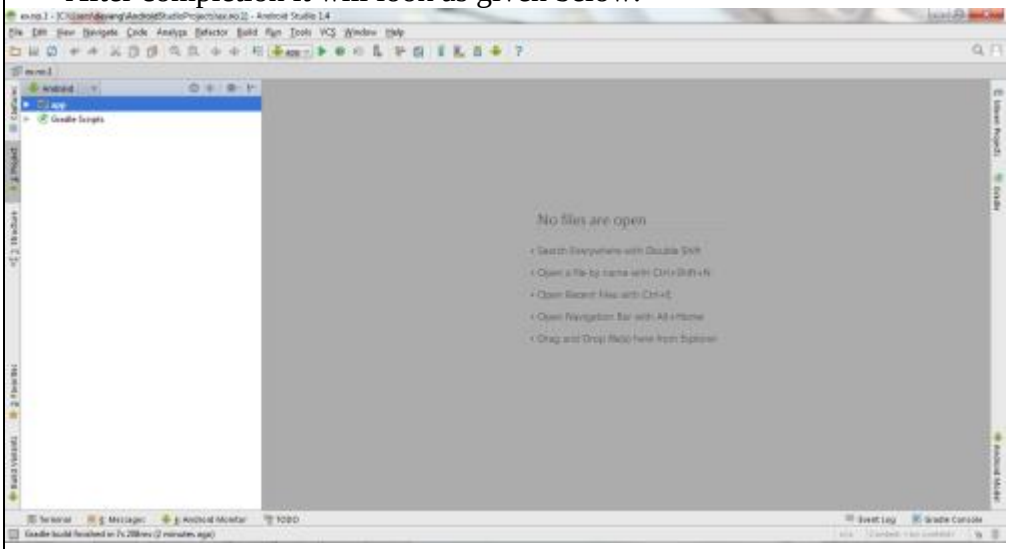
- Then select the **Empty Activity** and click **Next.**



- Finally click F**inish**.

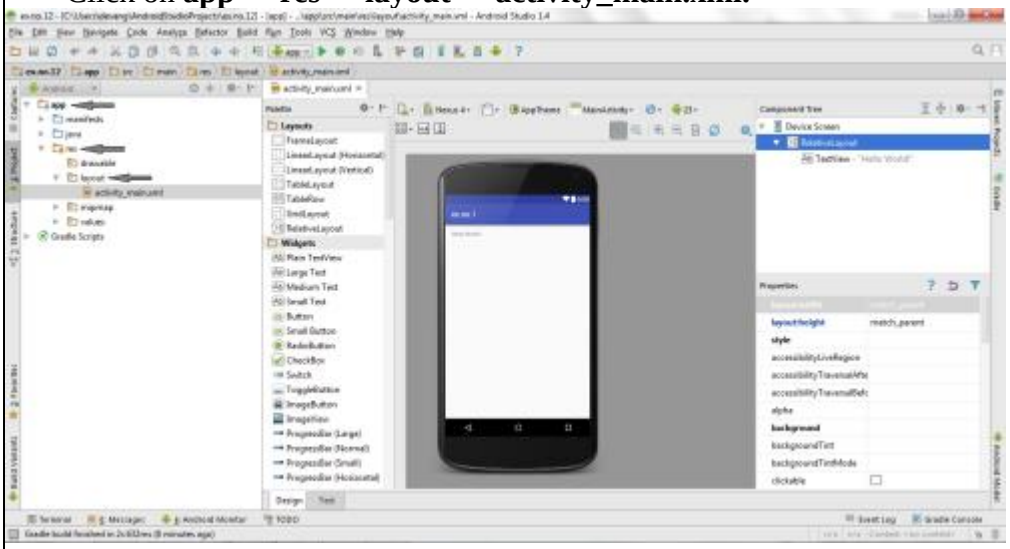


- It will take some time to build and load the project.
- After completion it will look as given below.

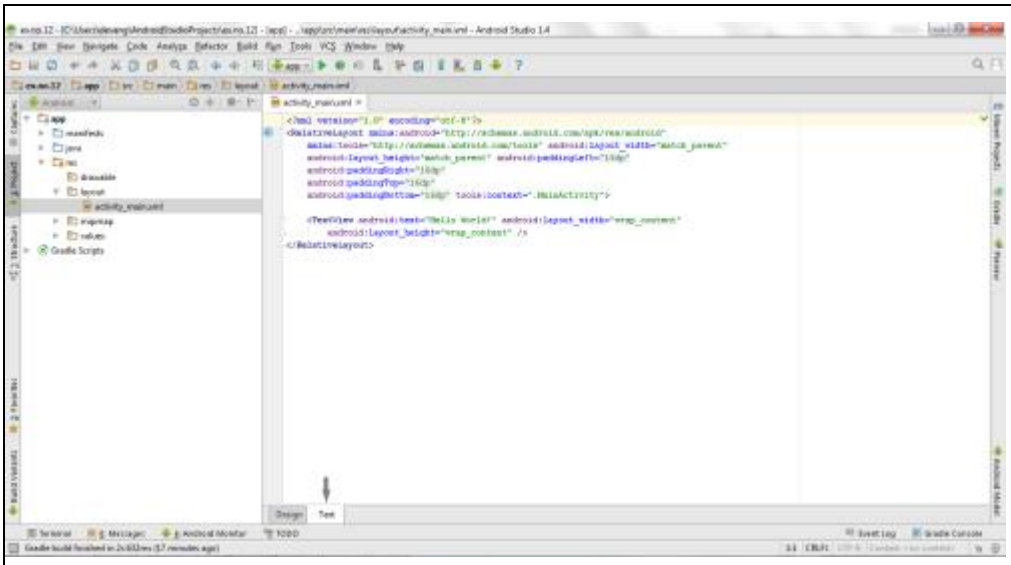


Designing layout for the Android Application:

- 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.

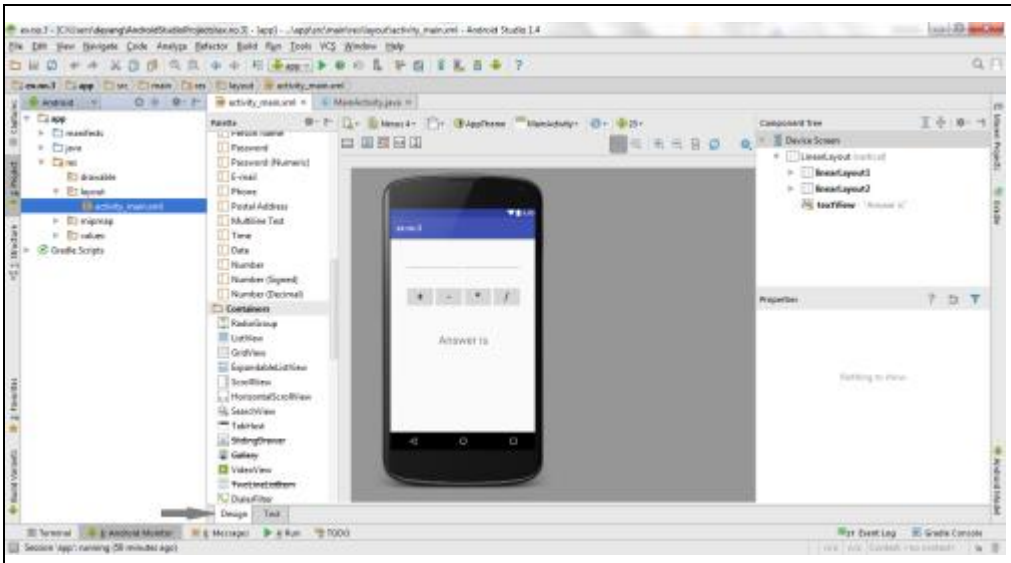
Code for Activity_main.xml:

?

```
<?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>
```

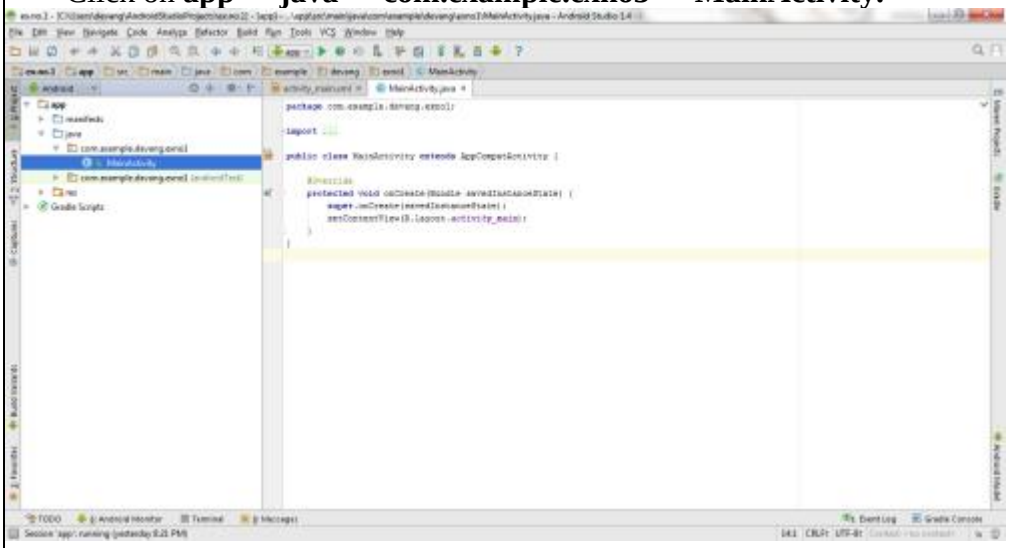
- Now click on Design and your application will look as given below.



- 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:

?

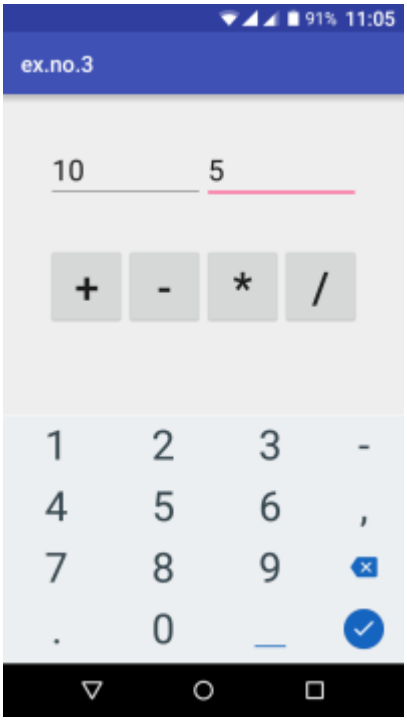
```
package com.example.devang.exno3;

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;
```

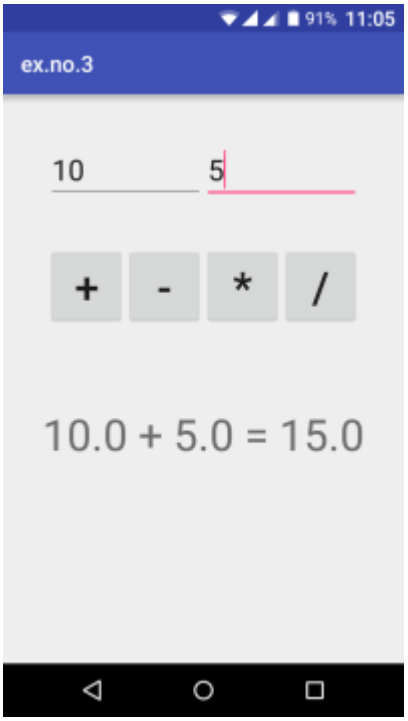
```
  Button Mul;
  Button Div;
  TextView Result;

  @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 = "-";
```

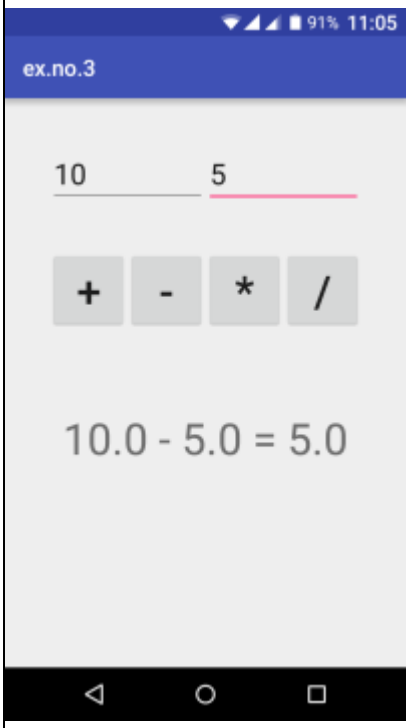
```
            result = num1 - num2;
            break;
        case R.id.Mul:
            oper = "*";
            result = num1 * num2;
            break;
        case R.id.Div:
            oper = "/";
            result = num1 / num2;
            break;
        default:
            break;
      }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

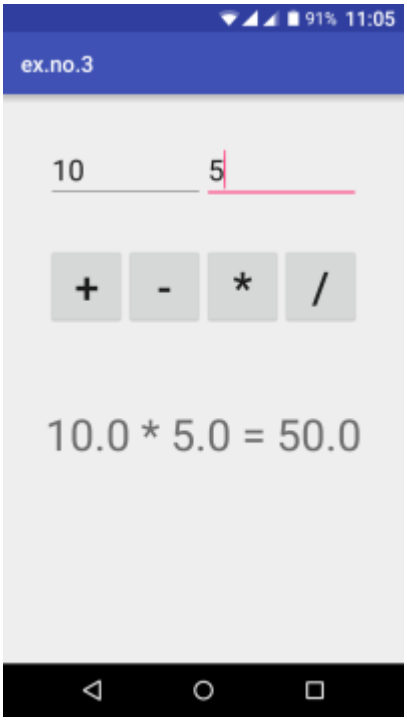
- So now the Coding part is also completed.
- Now run the application to see the output.

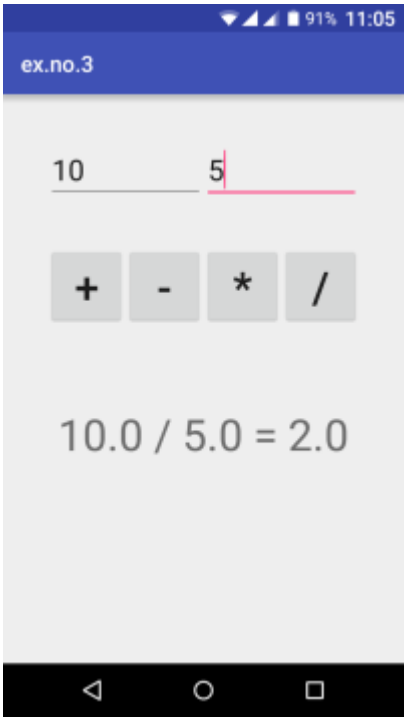
Output:











Result:

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

Ex.No:4 Develop a gaming application that uses 2-D animations and gestures

Aim:

To develop a gaming application that uses 2-D animations and gestures

Procedure:

Styles.xml

```
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
        <item name="windowNoTitle">true</item>
        <item name="windowActionBar">false</item>
        <item name="android:windowFullscreen">true</item>
        <item name="android:windowContentOverlay">@null</item>
    </style>

</resources>
```

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"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:background="@drawable/splash"
    tools:context="net.simplifiedcoding.simplegame.MainActivity">

    <ImageButton
        android:id="@+id/buttonPlay"
        android:background="@drawable/playnow"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/buttonScore"
        android:layout_centerHorizontal="true" />

    <ImageButton
        android:id="@+id/buttonScore"
        android:background="@drawable/highscore"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true" />

</RelativeLayout>
```

- When we tap the Play Now button our Game Activity will start.
- Now come inside MainActivity.java and write the following code.

MainActivity.java

```
package net.simplifiedcoding.simplegame;

import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.media.Image;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageButton;

public class MainActivity extends AppCompatActivity implements View.OnClickListener{

    //image button
    private ImageButton buttonPlay;

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

        //setting the orientation to landscape
        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE);

        //getting the button
        buttonPlay = (ImageButton) findViewById(R.id.buttonPlay);

        //adding a click listener
        buttonPlay.setOnClickListener(this);
    }

    @Override
    public void onClick(View v) {

        //starting game activity
```

```
        startActivity(new Intent(this, GameActivity.class));
    }
}
```

GameView.java

```
public class GameView extends SurfaceView implements Runnable {

    //boolean variable to track if the game is playing or not
    volatile boolean playing;

    //the game thread
    private Thread gameThread = null;


    //Class constructor
    public GameView(Context context) {
        super(context);

    }

    @Override
    public void run() {
        while (playing) {
      //to update the frame
            update();

      //to draw the frame
            draw();

      //to control
            control();
        }
    }



    private void update() {

    }

    private void draw() {

    }

    private void control() {
```

```
            try {
                gameThread.sleep(17);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }

        public void pause() {
        //when the game is paused
        //setting the variable to false
            playing = false;
            try {
            //stopping the thread
                gameThread.join();
            } catch (InterruptedException e) {
            }
        }

        public void resume() {
        //when the game is resumed
        //starting the thread again
            playing = true;
            gameThread = new Thread(this);
            gameThread.start();
        }
    }
```

- The above class is our GameView class. It is the actual game panel where we will play the game. The class is implementing Runnable interface. We have a volatile boolean type variable running that will track whether the game is running or not. After that we have our gameThread, it is the main game loop. Then we have the constructor to the class. We are not doing anything inside the constructor right now. Then we have the overriden method run(), here we are running a loop until the playing variable running is true. Inside the loop we are calling the following methods.
- update() -> Here we will update the coordinate of our characters.
- draw() -> Here we will draw the characters to the canvas.
- control() -> This method will control the frames per seconds drawn. Here we are calling the delay method of Thread. And this is actually making our frame rate to aroud 60fps.
- After these we have two more methods.
- pause() -> To pause the game, we are stopping the gameThread here.
- resume() -> To resume the game, here we are starting the gameThread.

GameActivity.java

```
package net.simplifiedcoding.spacefighter;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class GameActivity extends AppCompatActivity {

    //declaring gameview
    private GameView gameView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        //Initializing game view object
        gameView = new GameView(this);


        //adding it to contentview
        setContentView(gameView);
    }

    //pausing the game when activity is paused
    @Override
    protected void onPause() {
        super.onPause();
        gameView.pause();
    }

    //running the game when activity is resumed
    @Override
    protected void onResume() {
        super.onResume();
        gameView.resume();
    }
}
```

Player.java

```
package net.simplifiedcoding.spacefighter;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
```

```
public class Player {
    //Bitmap to get character from image
    private Bitmap bitmap;

    //coordinates
    private int x;
    private int y;

    //motion speed of the character
    private int speed = 0;

    //constructor
    public Player(Context context) {
        x = 75;
        y = 50;
        speed = 1;

        //Getting bitmap from drawable resource
        bitmap = BitmapFactory.decodeResource(context.getResources(), R.drawable.player);
    }

    //Method to update coordinate of character
    public void update(){
        //updating x coordinate
        x++;
    }

    /*
    * These are getters you can generate it autmaticallyl
    * right click on editor -> generate -> getters
    * */
    public Bitmap getBitmap() {
        return bitmap;
    }

    public int getX() {
        return x;
    }

    public int getY() {
        return y;
    }

    public int getSpeed() {
        return speed;
```

```
    }
}
```

Drawing Player to GameView: To draw the player to our GameView you need to come back to the GameView.java class and modify it as below.

GameView.java

```
public class GameView extends SurfaceView implements Runnable {

    volatile boolean playing;
    private Thread gameThread = null;

    //adding the player to this class
    private Player player;

    //These objects will be used for drawing
    private Paint paint;
    private Canvas canvas;
    private SurfaceHolder surfaceHolder;

    public GameView(Context context) {
        super(context);

        //initializing player object
        player = new Player(context);


        //initializing drawing objects
        surfaceHolder = getHolder();
        paint = new Paint();
    }

    @Override
    public void run() {
        while (playing) {
            update();
            draw();
            control();
        }
    }

    private void update() {
        //updating player position
        player.update();
    }
```

```
    private void draw() {
        //checking if surface is valid
        if (surfaceHolder.getSurface().isValid()) {
            //locking the canvas
            canvas = surfaceHolder.lockCanvas();
            //drawing a background color for canvas
            canvas.drawColor(Color.BLACK);
            //Drawing the player
            canvas.drawBitmap(
                    player.getBitmap(),
                    player.getX(),
                    player.getY(),
                    paint);
            //Unlocking the canvas
            surfaceHolder.unlockCanvasAndPost(canvas);
        }
    }

    private void control() {
        try {
            gameThread.sleep(17);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    public void pause() {
        playing = false;
        try {
            gameThread.join();
        } catch (InterruptedException e) {
        }
    }

    public void resume() {
        playing = true;
        gameThread = new Thread(this);
        gameThread.start();
    }
}
```

Output without Control:

Adding Controls:

```
@Override
public boolean onTouchEvent(MotionEvent motionEvent) {
    switch (motionEvent.getAction() & MotionEvent.ACTION_MASK) {
        case MotionEvent.ACTION_UP:
            //When the user presses on the screen
```

```
            break;
        case MotionEvent.ACTION_DOWN:
            //When the user releases the screen
            //do something here
            break;
    }
    return true;
}
```

Player.java

```
public class Player {
    private Bitmap bitmap;
    private int x;
    private int y;
    private int speed = 0;

    //boolean variable to track the ship is boosting or not
    private boolean boosting;

    //Gravity Value to add gravity effect on the ship
    private final int GRAVITY = -10;

    //Controlling Y coordinate so that ship won't go outside the screen
    private int maxY;
    private int minY;

    //Limit the bounds of the ship's speed
    private final int MIN_SPEED = 1;
    private final int MAX_SPEED = 20;

    public Player(Context context) {
        x = 75;
        y = 50;
        speed = 1;
        bitmap = BitmapFactory.decodeResource(context.getResources(), R.drawable.player);

        //setting the boosting value to false initially
        boosting = false;
    }



    //setting boosting true
    public void setBoosting() {
```

```
    boosting = true;
}

//setting boosting false
public void stopBoosting() {
    boosting = false;
}

public void update() {
    //if the ship is boosting
    if (boosting) {
        //speeding up the ship
        speed += 2;
    } else {
        //slowing down if not boosting
        speed -= 5;
    }
    //controlling the top speed
    if (speed > MAX_SPEED) {
        speed = MAX_SPEED;
    }
    //if the speed is less than min speed
    //controlling it so that it won't stop completely
    if (speed < MIN_SPEED) {
        speed = MIN_SPEED;
    }

    //moving the ship down
    y -= speed + GRAVITY;

    //but controlling it also so that it won't go off the screen
    if (y < minY) {
        y = minY;
    }
    if (y > maxY) {
        y = maxY;
    }
}

public Bitmap getBitmap() {
    return bitmap;
}

public int getX() {
    return x;
```

```
    }

    public int getY() {
        return y;
    }

    public int getSpeed() {
        return speed;
    }
}
```

GameActivity.java

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        //Getting display object
        Display display = getWindowManager().getDefaultDisplay();

        //Getting the screen resolution into point object
        Point size = new Point();
        display.getSize(size);

        //Initializing game view object
        //this time we are also passing the screen size to the GameView constructor
        gameView = new GameView(this, size.x, size.y);


        //adding it to contentview
        setContentView(gameView);
    }
```

Now to complete adding the boosters come inside GameView.java file and modify the onTouchEvent() as follows.

```
    @Override
    public boolean onTouchEvent(MotionEvent motionEvent) {
        switch (motionEvent.getAction() & MotionEvent.ACTION_MASK) {
            case MotionEvent.ACTION_UP:
                //stopping the boosting when screen is released
                player.stopBoosting();
                break;
            case MotionEvent.ACTION_DOWN:
                //boosting the space jet when screen is pressed
                player.setBoosting();
```

```
            break;
        }
        return true;
    }
```

Now we will add background stars to make the background looks animating.

```
package net.simplifiedcoding.spacefighter;

import java.util.Random;

public class Star {
    private int x;
    private int y;
    private int speed;

    private int maxX;
    private int maxY;
    private int minX;
    private int minY;



    public Star(int screenX, int screenY) {
        maxX = screenX;
        maxY = screenY;
        minX = 0;
        minY = 0;
        Random generator = new Random();
        speed = generator.nextInt(10);

        //generating a random coordinate
        //but keeping the coordinate inside the screen size
        x = generator.nextInt(maxX);
        y = generator.nextInt(maxY);
    }

    public void update(int playerSpeed) {
        //animating the star horizontally left side
        //by decreasing x coordinate with player speed
        x -= playerSpeed;
        x -= speed;
        //if the star reached the left edge of the screen
        if (x < 0) {
            //again starting the star from right edge
```

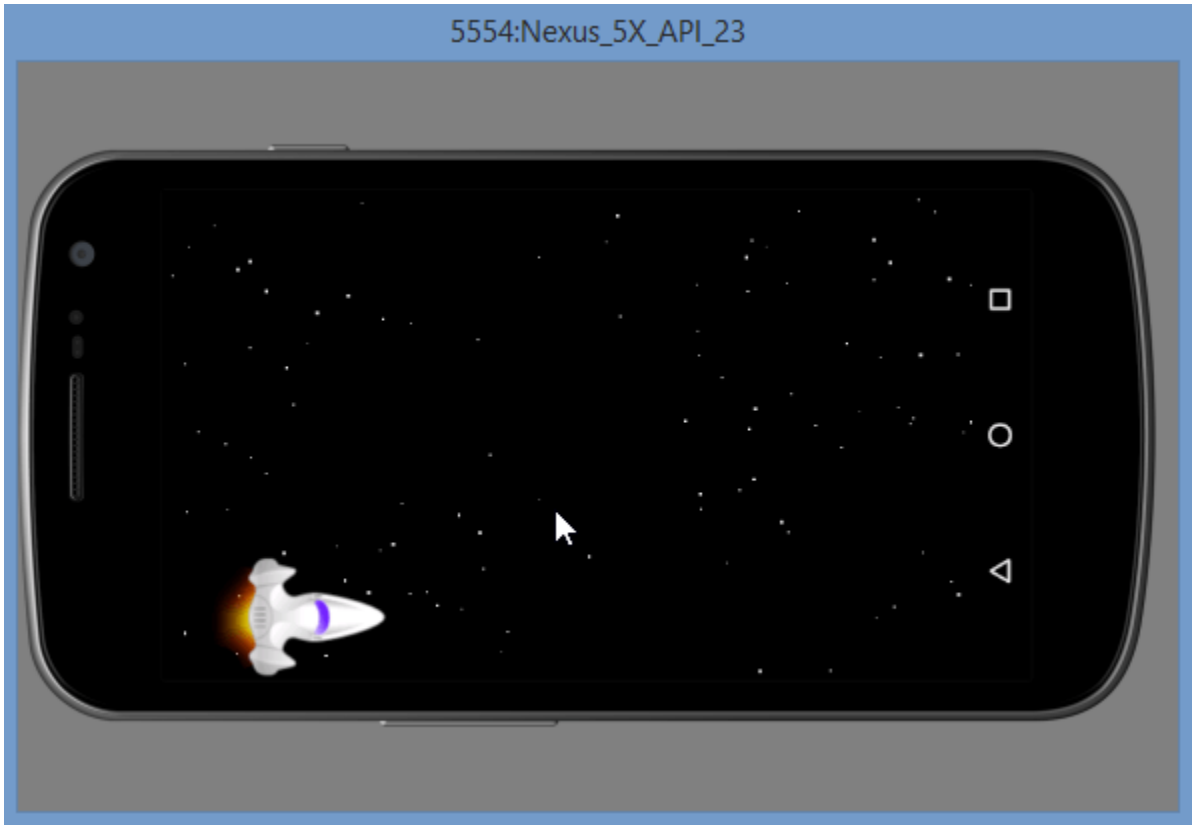
```
            //this will give a infinite scrolling background effect
            x = maxX;
            Random generator = new Random();
            y = generator.nextInt(maxY);
            speed = generator.nextInt(15);
        }
    }

    public float getStarWidth() {
        //Making the star width random so that
        //it will give a real look
        float minX = 1.0f;
        float maxX = 4.0f;
        Random rand = new Random();
        float finalX = rand.nextFloat() * (maxX - minX) + minX;
        return finalX;
    }

    public int getX() {
        return x;
    }

    public int getY() {
        return y;
    }
}
```

GameView.java

```
public class GameView extends SurfaceView implements Runnable {

    volatile boolean playing;
    private Thread gameThread = null;
    private Player player;

    private Paint paint;
    private Canvas canvas;
    private SurfaceHolder surfaceHolder;

    //Adding an stars list
    private ArrayList<Star> stars = new
            ArrayList<Star>();

    public GameView(Context context, int screenX, int screenY) {
        super(context);
```

```
        player = new Player(context, screenX, screenY);

        surfaceHolder = getHolder();
        paint = new Paint();

        //adding 100 stars you may increase the number
        int starNums = 100;
        for (int i = 0; i < starNums; i++) {
            Star s = new Star(screenX, screenY);
            stars.add(s);
        }
    }

    @Override
    public void run() {
        while (playing) {
            update();
            draw();
            control();
        }
    }

    private void update() {
        player.update();

        //Updating the stars with player speed
        for (Star s : stars) {
            s.update(player.getSpeed());
        }
    }

    private void draw() {
        if (surfaceHolder.getSurface().isValid()) {
            canvas = surfaceHolder.lockCanvas();
            canvas.drawColor(Color.BLACK);

            //setting the paint color to white to draw the stars
            paint.setColor(Color.WHITE);

            //drawing all stars
            for (Star s : stars) {
                paint.setStrokeWidth(s.getStarWidth());
                canvas.drawPoint(s.getX(), s.getY(), paint);
            }
```

```
            canvas.drawBitmap(
                    player.getBitmap(),
                    player.getX(),
                    player.getY(),
                    paint);
            surfaceHolder.unlockCanvasAndPost(canvas);
        }
    }

    private void control() {
        try {
            gameThread.sleep(17);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    public void pause() {
        playing = false;
        try {
            gameThread.join();
        } catch (InterruptedException e) {
        }
    }

    public void resume() {
        playing = true;
        gameThread = new Thread(this);
        gameThread.start();
    }


    @Override
    public boolean onTouchEvent(MotionEvent motionEvent) {
        switch (motionEvent.getAction() & MotionEvent.ACTION_MASK) {
            case MotionEvent.ACTION_UP:
                player.stopBoosting();
                break;
            case MotionEvent.ACTION_DOWN:
                player.setBoosting();
                break;
        }
        return true;
    }
}
```



Create a new java class named Enemy and write the following code.

```
package net.simplifiedcoding.spacefighter;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Rect;

import java.util.Random;

public class Enemy {

    //bitmap for the enemy
    //we have already pasted the bitmap in the drawable folder
    private Bitmap bitmap;

    //x and y coordinates
    private int x;
    private int y;
```

```
//enemy speed
private int speed = 1;

//min and max coordinates to keep the enemy inside the screen
private int maxX;
private int minX;

private int maxY;
private int minY;


public Enemy(Context context, int screenX, int screenY) {
   //getting bitmap from drawable resource
   bitmap = BitmapFactory.decodeResource(context.getResources(), R.drawable.enemy);

   //initializing min and max coordinates
   maxX = screenX;
   maxY = screenY;
   minX = 0;
   minY = 0;

   //generating a random coordinate to add enemy
   Random generator = new Random();
   speed = generator.nextInt(6) + 10;
   x = screenX;
   y = generator.nextInt(maxY) - bitmap.getHeight();

}

public void update(int playerSpeed) {
   //decreasing x coordinate so that enemy will move right to left
   x -= playerSpeed;
   x -= speed;
   //if the enemy reaches the left edge
   if (x < minX - bitmap.getWidth()) {
      //adding the enemy again to the right edge
      Random generator = new Random();
      speed = generator.nextInt(10) + 10;
      x = maxX;
      y = generator.nextInt(maxY) - bitmap.getHeight();
   }
}
```

```
    //getters
    public Bitmap getBitmap() {
        return bitmap;
    }

    public int getX() {
        return x;
    }

    public int getY() {
        return y;
    }

    public int getSpeed() {
        return speed;
    }

}
```

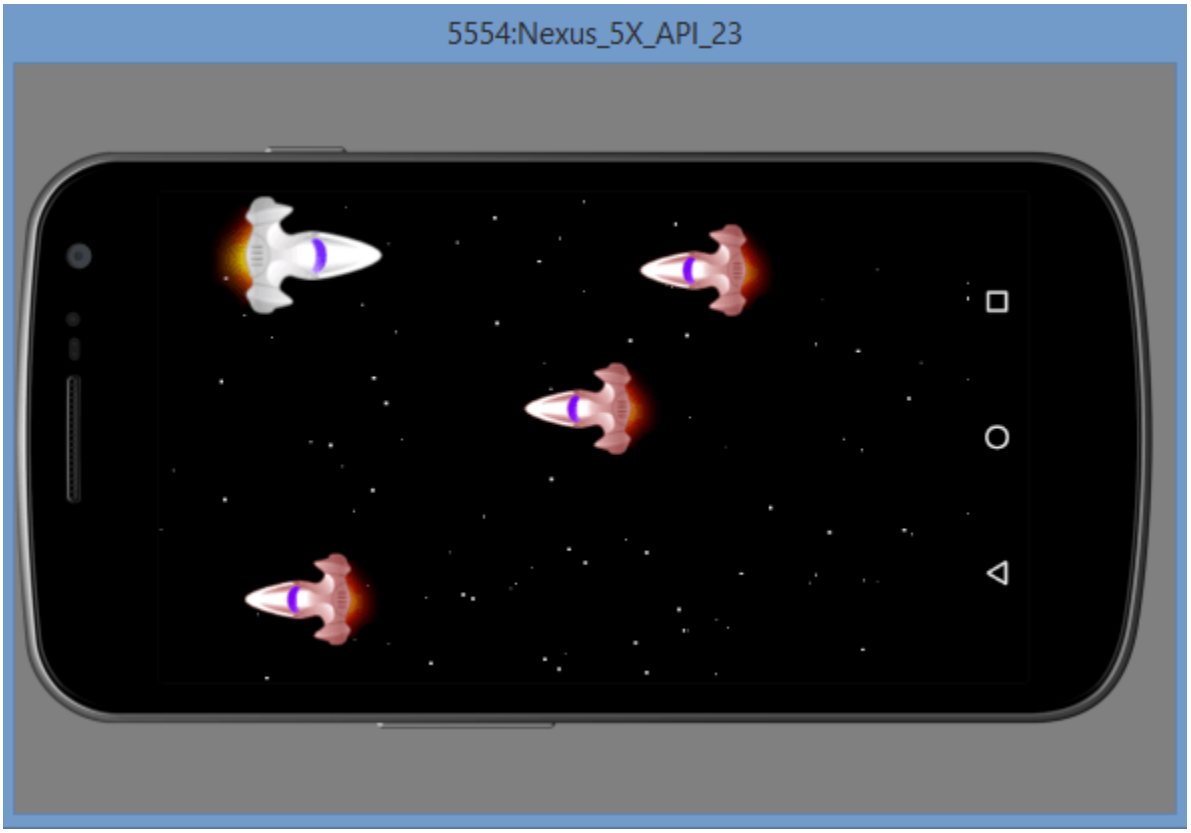
We need to add the enemies in the GameView now. So come inside GameView.java and modify the code as follows.

```
public class GameView extends SurfaceView implements Runnable {

    volatile boolean playing;
    private Thread gameThread = null;
    private Player player;

    private Paint paint;
    private Canvas canvas;
    private SurfaceHolder surfaceHolder;

    //Adding enemies object array
    private Enemy[] enemies;

    //Adding 3 enemies you may increase the size
    private int enemyCount = 3;

    private ArrayList<Star> stars = new
            ArrayList<Star>();

    public GameView(Context context, int screenX, int screenY) {
        super(context);
        player = new Player(context, screenX, screenY);
```

```
        surfaceHolder = getHolder();
        paint = new Paint();

        int starNums = 100;
        for (int i = 0; i < starNums; i++) {
            Star s = new Star(screenX, screenY);
            stars.add(s);
        }

        //initializing enemy object array
        enemies = new Enemy[enemyCount];
        for(int i=0; i<enemyCount; i++){
            enemies[i] = new Enemy(context, screenX, screenY);
        }
    }

    @Override
    public void run() {
        while (playing) {
            update();
            draw();
            control();
        }
    }

    private void update() {
        player.update();
        for (Star s : stars) {
            s.update(player.getSpeed());
        }

        //updating the enemy coordinate with respect to player speed
        for(int i=0; i<enemyCount; i++){
            enemies[i].update(player.getSpeed());
        }
    }

    private void draw() {
        if (surfaceHolder.getSurface().isValid()) {
            canvas = surfaceHolder.lockCanvas();
            canvas.drawColor(Color.BLACK);

            paint.setColor(Color.WHITE);

            for (Star s : stars) {
```

```
            paint.setStrokeWidth(s.getStarWidth());
            canvas.drawPoint(s.getX(), s.getY(), paint);
        }

        canvas.drawBitmap(
                player.getBitmap(),
                player.getX(),
                player.getY(),
                paint);

        //drawing the enemies
        for (int i = 0; i < enemyCount; i++) {
            canvas.drawBitmap(
                    enemies[i].getBitmap(),
                    enemies[i].getX(),
                    enemies[i].getY(),
                    paint
            );
        }

        surfaceHolder.unlockCanvasAndPost(canvas);

    }
}

private void control() {
    try {
        gameThread.sleep(17);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}

public void pause() {
    playing = false;
    try {
        gameThread.join();
    } catch (InterruptedException e) {
    }
}

public void resume() {
    playing = true;
    gameThread = new Thread(this);
    gameThread.start();
```

```
    }


    @Override
    public boolean onTouchEvent(MotionEvent motionEvent) {
        switch (motionEvent.getAction() & MotionEvent.ACTION_MASK) {
            case MotionEvent.ACTION_UP:
                player.stopBoosting();
                break;
            case MotionEvent.ACTION_DOWN:
                player.setBoosting();
                break;
        }
        return true;
    }
}
```



Detecting Collision

```
public class Enemy {
    private Bitmap bitmap;
    private int x;
    private int y;
    private int speed = 1;
```

```
private int maxX;
private int minX;

private int maxY;
private int minY;

//creating a rect object
private Rect detectCollision;

public Enemy(Context context, int screenX, int screenY) {
   bitmap = BitmapFactory.decodeResource(context.getResources(), R.drawable.enemy);
   maxX = screenX;
   maxY = screenY;
   minX = 0;
   minY = 0;
   Random generator = new Random();
   speed = generator.nextInt(6) + 10;
   x = screenX;
   y = generator.nextInt(maxY) - bitmap.getHeight();

   //initializing rect object
   detectCollision = new Rect(x, y, bitmap.getWidth(), bitmap.getHeight());
}

public void update(int playerSpeed) {
   x -= playerSpeed;
   x -= speed;
   if (x < minX - bitmap.getWidth()) {
      Random generator = new Random();
      speed = generator.nextInt(10) + 10;
      x = maxX;
      y = generator.nextInt(maxY) - bitmap.getHeight();
   }

   //Adding the top, left, bottom and right to the rect object
   detectCollision.left = x;
   detectCollision.top = y;
   detectCollision.right = x + bitmap.getWidth();
   detectCollision.bottom = y + bitmap.getHeight();
}

//adding a setter to x coordinate so that we can change it after collision
public void setX(int x){
   this.x = x;
```

```
    }

    //one more getter for getting the rect object
    public Rect getDetectCollision() {
        return detectCollision;
    }

    //getters
    public Bitmap getBitmap() {
        return bitmap;
    }

    public int getX() {
        return x;
    }

    public int getY() {
        return y;
    }

    public int getSpeed() {
        return speed;
    }

}
```

Player.java

```
public class Player {
    private Bitmap bitmap;
    private int x;
    private int y;
    private int speed = 0;
    private boolean boosting;
    private final int GRAVITY = -10;
    private int maxY;
    private int minY;

    private final int MIN_SPEED = 1;
    private final int MAX_SPEED = 20;

    private Rect detectCollision;

    public Player(Context context, int screenX, int screenY) {
        x = 75;
```

```
        y = 50;
        speed = 1;
        bitmap = BitmapFactory.decodeResource(context.getResources(), R.drawable.player);
        maxY = screenY - bitmap.getHeight();
        minY = 0;
        boosting = false;

        //initializing rect object
        detectCollision = new Rect(x, y, bitmap.getWidth(), bitmap.getHeight());
    }

    public void setBoosting() {
        boosting = true;
    }

    public void stopBoosting() {
        boosting = false;
    }

    public void update() {
        if (boosting) {
            speed += 2;
        } else {
            speed -= 5;
        }

        if (speed > MAX_SPEED) {
            speed = MAX_SPEED;
        }

        if (speed < MIN_SPEED) {
            speed = MIN_SPEED;
        }

        y -= speed + GRAVITY;

        if (y < minY) {
            y = minY;
        }
        if (y > maxY) {
            y = maxY;
        }

        //adding top, left, bottom and right to the rect object
        detectCollision.left = x;
```

```
        detectCollision.top = y;
        detectCollision.right = x + bitmap.getWidth();
        detectCollision.bottom = y + bitmap.getHeight();


    }

    //one more getter for getting the rect object
    public Rect getDetectCollision() {
        return detectCollision;
    }

    public Bitmap getBitmap() {
        return bitmap;
    }

    public int getX() {
        return x;
    }

    public int getY() {
        return y;
    }

    public int getSpeed() {
        return speed;
    }
}
```

Now to complete the collision detection, again to inside GameView.java file and modify the update() method as follows.

```
    private void update() {
        player.update();
        for (Star s : stars) {
            s.update(player.getSpeed());
        }

        for(int i=0; i<enemyCount; i++){
            enemies[i].update(player.getSpeed());

            //if collision occurrs with player
            if (Rect.intersects(player.getDetectCollision(), enemies[i].getDetectCollision())) {
                //moving enemy outside the left edge
                enemies[i].setX(-200);
            }
        }
```

```
    }
```

Adding Blast Effect

Boom.java

```
package net.simplifiedcoding.spacefighter;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;

public class Boom {

    //bitmap object
    private Bitmap bitmap;

    //coordinate variables
    private int x;
    private int y;

    //constructor
    public Boom(Context context) {
        //getting boom image from drawable resource
        bitmap = BitmapFactory.decodeResource
                (context.getResources(), R.drawable.boom);

        //setting the coordinate outside the screen
        //so that it won't shown up in the screen
        //it will be only visible for a fraction of second
        //after collission
        x = -250;
        y = -250;
    }

    //setters for x and y to make it visible at the place of collision
    public void setX(int x) {
        this.x = x;
    }

    public void setY(int y) {
        this.y = y;
    }

    //getters
```

```
    public Bitmap getBitmap() {
        return bitmap;
    }

    public void setBitmap(Bitmap bitmap) {
        this.bitmap = bitmap;
    }

    public int getX() {
        return x;
    }

    public int getY() {
        return y;
    }

}
```

Now again come inside GameView.java file and modify the code as follow.

```
public class GameView extends SurfaceView implements Runnable {

    volatile boolean playing;
    private Thread gameThread = null;
    private Player player;

    private Paint paint;
    private Canvas canvas;
    private SurfaceHolder surfaceHolder;

    private Enemy[] enemies;

    private int enemyCount = 3;

    private ArrayList<Star> stars = new
            ArrayList<Star>();

    //defining a boom object to display blast
    private Boom boom;

    public GameView(Context context, int screenX, int screenY) {
        super(context);
        player = new Player(context, screenX, screenY);

        surfaceHolder = getHolder();
```

```
        paint = new Paint();

        int starNums = 100;
        for (int i = 0; i < starNums; i++) {
            Star s = new Star(screenX, screenY);
            stars.add(s);
        }

        enemies = new Enemy[enemyCount];
        for (int i = 0; i < enemyCount; i++) {
            enemies[i] = new Enemy(context, screenX, screenY);
        }

        //initializing boom object
        boom = new Boom(context);
    }

    @Override
    public void run() {
        while (playing) {
            update();
            draw();
            control();
        }
    }

    private void update() {
        player.update();

        //setting boom outside the screen
        boom.setX(-250);
        boom.setY(-250);

        for (Star s : stars) {
            s.update(player.getSpeed());
        }

        for (int i = 0; i < enemyCount; i++) {
            enemies[i].update(player.getSpeed());

            //if collision occurrs with player
            if (Rect.intersects(player.getDetectCollision(), enemies[i].getDetectCollision())) {

                //displaying boom at that location
                boom.setX(enemies[i].getX());
```

```
            boom.setY(enemies[i].getY());

            enemies[i].setX(-200);

        }
    }
}

private void draw() {
    if (surfaceHolder.getSurface().isValid()) {
        canvas = surfaceHolder.lockCanvas();
        canvas.drawColor(Color.BLACK);

        paint.setColor(Color.WHITE);

        for (Star s : stars) {
            paint.setStrokeWidth(s.getStarWidth());
            canvas.drawPoint(s.getX(), s.getY(), paint);
        }

        canvas.drawBitmap(
                player.getBitmap(),
                player.getX(),
                player.getY(),
                paint);

        for (int i = 0; i < enemyCount; i++) {
            canvas.drawBitmap(
                    enemies[i].getBitmap(),
                    enemies[i].getX(),
                    enemies[i].getY(),
                    paint
            );
        }

        //drawing boom image
        canvas.drawBitmap(
                boom.getBitmap(),
                boom.getX(),
                boom.getY(),
                paint
        );

        surfaceHolder.unlockCanvasAndPost(canvas);
```

```
        }
    }

    private void control() {
        try {
            gameThread.sleep(17);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    public void pause() {
        playing = false; try
        {
            gameThread.join();
        } catch (InterruptedException e) {
        }
    }

    public void resume() {
        playing = true;
        gameThread = new Thread(this);
        gameThread.start();
    }


    @Override
    public boolean onTouchEvent(MotionEvent motionEvent) {
        switch (motionEvent.getAction() & MotionEvent.ACTION_MASK) {
            case MotionEvent.ACTION_UP:
                player.stopBoosting(); break;
            case MotionEvent.ACTION_DOWN:
                player.setBoosting();
                break;
        }
        return true;
    }
}
```

Now again execute the application and you will see a blast effect on collision.

Result:

Thus, the program was executed successfully.

Ex.No: 5 **Develop a movie rating application (similar to IMDB)**

Aim:

To develop a movie rating application.

Procedure:

MainActivity.java

```
package com.example.radiobutton;
import android.os.Bundle;
import android.app.Activity; import
android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import  android.widget.RadioGroup.OnCheckedChangeListener; import
android.widget.TextView;
import android.widget.Toast;
public class MainActivity extends Activity
{
private RadioGroup radioGroup;
private RadioButton sound, vibration, silent;
private Button button;
private TextView textView;
@Override
protected void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
radioGroup = (RadioGroup) findViewById(R.id.myRadioGroup);
radioGroup.setOnCheckedChangeListener(new OnCheckedChangeListener()
 {
@Override
public void onCheckedChanged(RadioGroup group, int checkedId)
 {
// find which radio button is selected
```

```
if(checkedId == R.id.silent)
    {
    Toast.makeText(getApplicationContext(), "choice: Silent", Toast.LENGTH_SHORT).show();

    }
     else if(checkedId ==R.id.sound)
    {
    Toast.makeText(getApplicationContext(), "choice: Sound", Toast.LENGTH_SHORT).show();

    }
    else
     {
    Toast.makeText(getApplicationContext(), "choice: Vibration",
    Toast.LENGTH_SHORT).show();

    }
    }
    });
    sound = (RadioButton) findViewById(R.id.sound);
    vibration=(RadioButton)findViewById(R.id.vibrate);
    silent = (RadioButton) findViewById(R.id.silent);
    textView = (TextView) findViewById(R.id.textView1);
    button = (Button)findViewById(R.id.button1);
    button.setOnClickListener(new OnClickListener()
    {
    @Override
    public void onClick(View v) {
    int selectedId = radioGroup.getCheckedRadioButtonId();

    //find which radioButton is checked by id
    if(selectedId ==sound.getId())
    {
    textView.setText("You chose 'Sound' option");
    }
    else if(selectedId == vibration.getId())
```

```
{
textView.setText("You chose 'Vibration' option");
}
else
{
textView.setText("Youchose 'Silent' option");
}
}
});
}
}
```

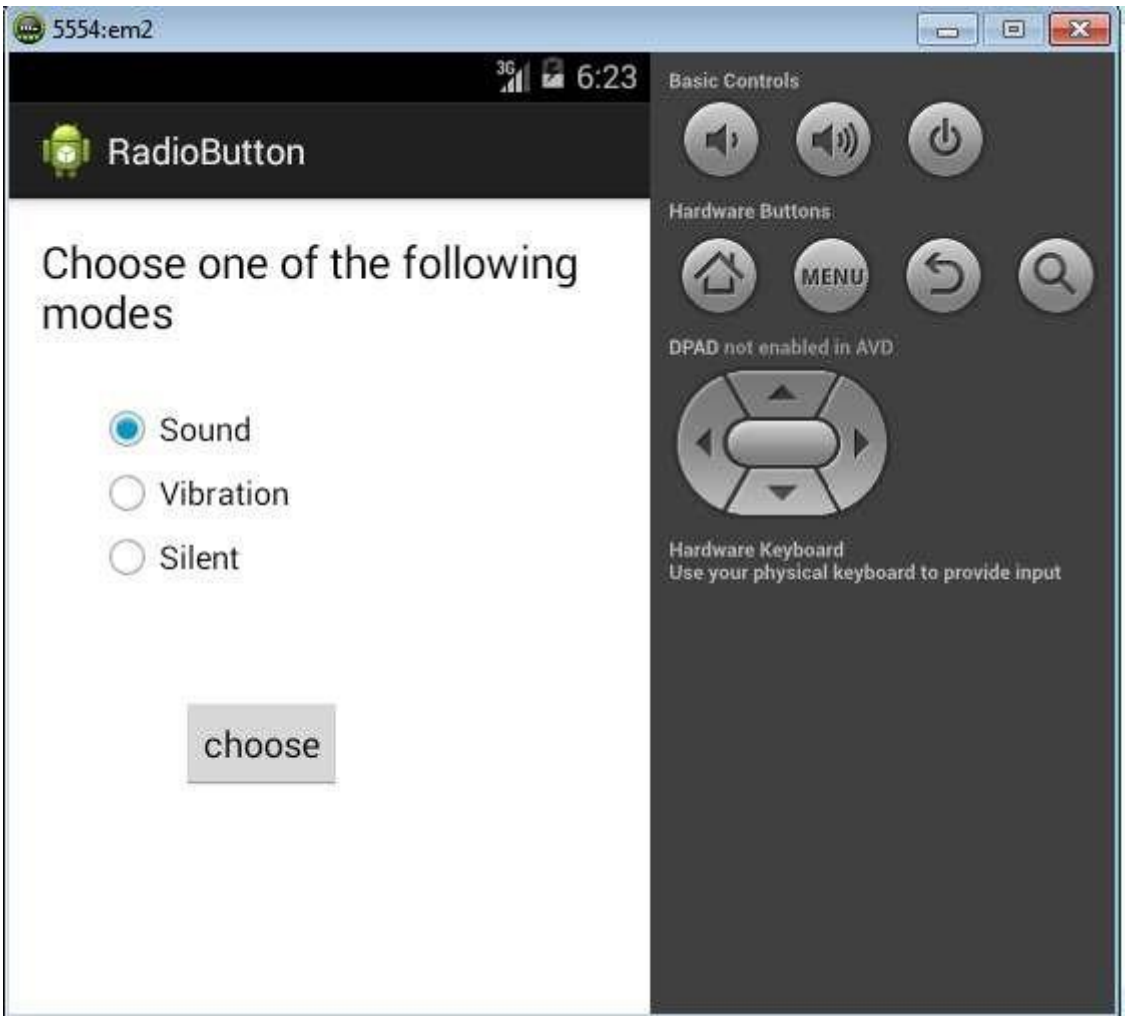
MainActivity.xml

```
<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"
android:paddingBottom="@dimen/activity_vertical
_margin"
android:paddingLeft="@dimen/activity_horizontal_
margin"
android:paddingRight="@dimen/activity_horizontal
_margin"
android:paddingTop="@dimen/activity_vertical_ma
rgin" tools:context=".MainActivity" >
<RadioGroup
android:id="@+id/myRadio
Group"
android:layout_width="wrap
_content"
android:layout_height="wrap_
content"
android:layout_alignParentLef
t="true"
```

```
android:layout_below="@+id/
textView1"
android:layout_marginLeft="
27dp"
android:layout_marginTop="
28dp" >

<RadioButton
android:id="@+id/sound"
android:layout_width="wrap_
content"

---
android:layout_height="wrap_
content"
android:checked="true"
android:text="Sound" />

<RadioButton
android:id="@+id/vibrate"
android:layout_width="wra
p_content"
android:layout_height="wra
p_content"
android:text="Vibration"
/>

<RadioButton
android:id="@+id/silent"
android:layout_width="wra
p_content"

android:layout_height="wra
p_content"
android:text="Silent" />

</RadioGroup>
<TextView
android:id="@+id/textVie
w1"
android:layout_width="wra
```

```
p_content"
android:layout_height="wra
p_content"
android:layout_alignParentL
eft="true"
android:layout_alignParentT
op="true"
android:text="Choose one of the following modes"
android:textAppearance="?android:attr/textAppearanceLarge" />
<Button
android:id="@+id/but
ton1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/myRadioGr
oup"
android:layout_below="@+id/myRadioGro
up" android:layout_marginLeft="42dp"
android:layout_marginTop="53dp"
android:text="choose" />
</RelativeLayout>
```

OUTPUT:



Result:

Thus, the program was executed and implemented successfully.

Ex.No: 6 Develop an application to connect to a web service and to retrieve data with HTTP

Aim:

To develop an application to connect to a web service and to retrieve data with HTTP

Algorithm:

1. Create a New Android Project:
   - Click New in the toolbar.
     - In the window that appears, open the Android folder, select Android Application Project, and click next.
     - Provide the application name and the project name and then finally give the desired package name.
     - Choose a launcher icon for your application and then select Blank Activity and then click Next
     - Provide the desired Activity name for your project and then click Finish.
2. Create a New AVD (Android Virtual Device):
   - click Android Virtual Device Manager from the toolbar.
   - In the Android Virtual Device Manager panel, click New.
   - Fill in the details for the AVD. Give it a name, a platform target, an SD card size, and a skin (HVGA is default).
   - Click Create AVD and Select the new AVD from the Android Virtual Device Manager and click Start.
3. Design the graphical layout.
4. Run the application.
5. When the application starts alarm sound will be invoked.
6. Stop alarm button is clicked to stop the alarm.
7. Close the Android project.

Program Code:
Mainactivity.Java

```
package com.example.admin.myapplication;

import
android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import
android.view.View;
import
android.widget.Button;
import
android.widget.Toast;

public class MainActivityextends

   AppCompatActivity {

@Override
```

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

        Button startBtn = (Button) findViewById(R.id.sendbttn);
startBtn.setOnClickListener(new View.OnClickListener() {
public void onClick(View view) {
sendEmail();
            }
        });
    }

protected void sendEmail() {
Log.i("Send email", "");

        String[] TO = {

"muthuramalingam566@gmail.com"
};
        String[] CC = {

"ramdurai25@gmail.com"
};
        Intent emailIntent = new Intent(Intent.ACTION_SEND);
emailIntent.setData(Uri.parse("mailto:"));
emailIntent.setType("text/plain");
emailIntent.putExtra(Intent.EXTRA_EMAIL, TO);
emailIntent.putExtra(Intent.EXTRA_CC, CC);
emailIntent.putExtra(Intent.EXTRA_SUBJECT, "Your subject");
emailIntent.putExtra(Intent.EXTRA_TEXT, "Email message goes here");
try {

startActivity(Intent.createChooser(emailIntent, "Send mail..."));
            finish();

Log.i("Finished sending email...", "");

        } catch (android.content.ActivityNotFoundException ex) {
Toast.makeText(MainActivity.this, "There is no email client installed.",
Toast.LENGTH_SHORT).show();

        }
    }
}
```

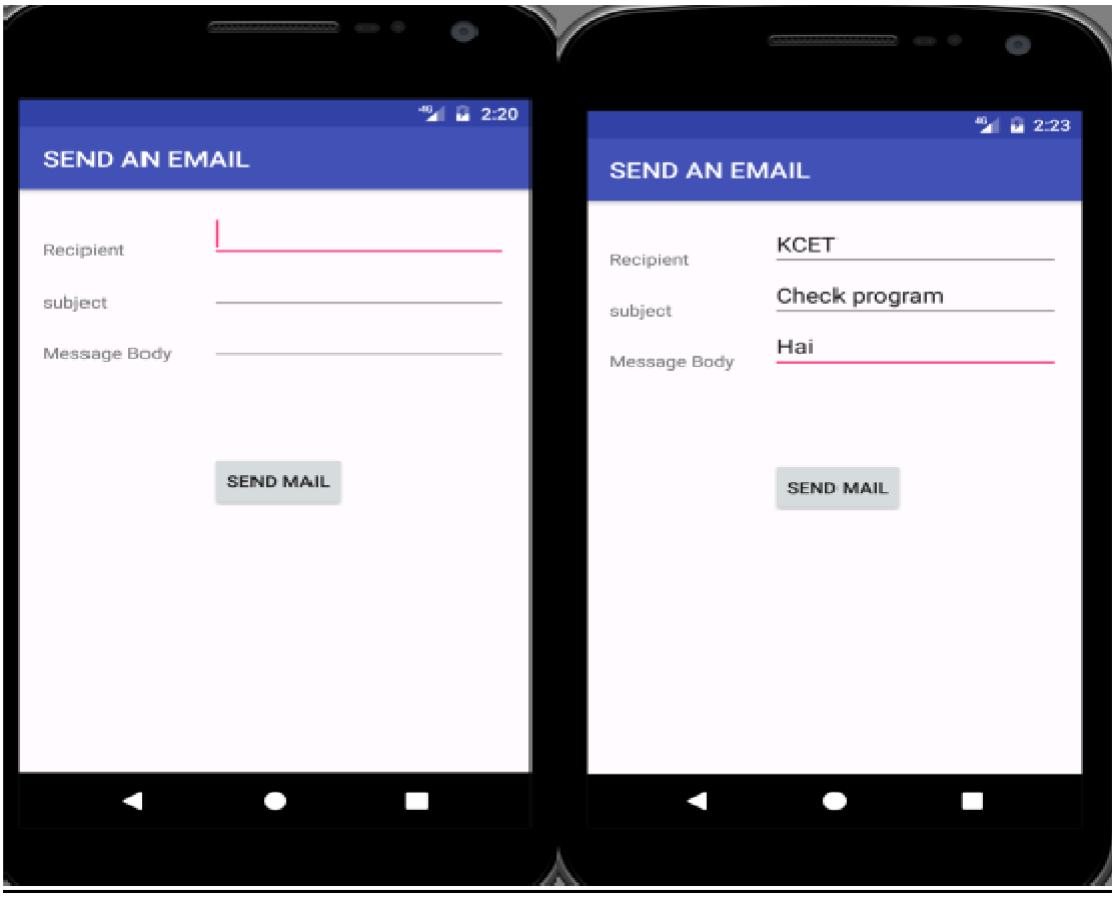
activity_main.xml

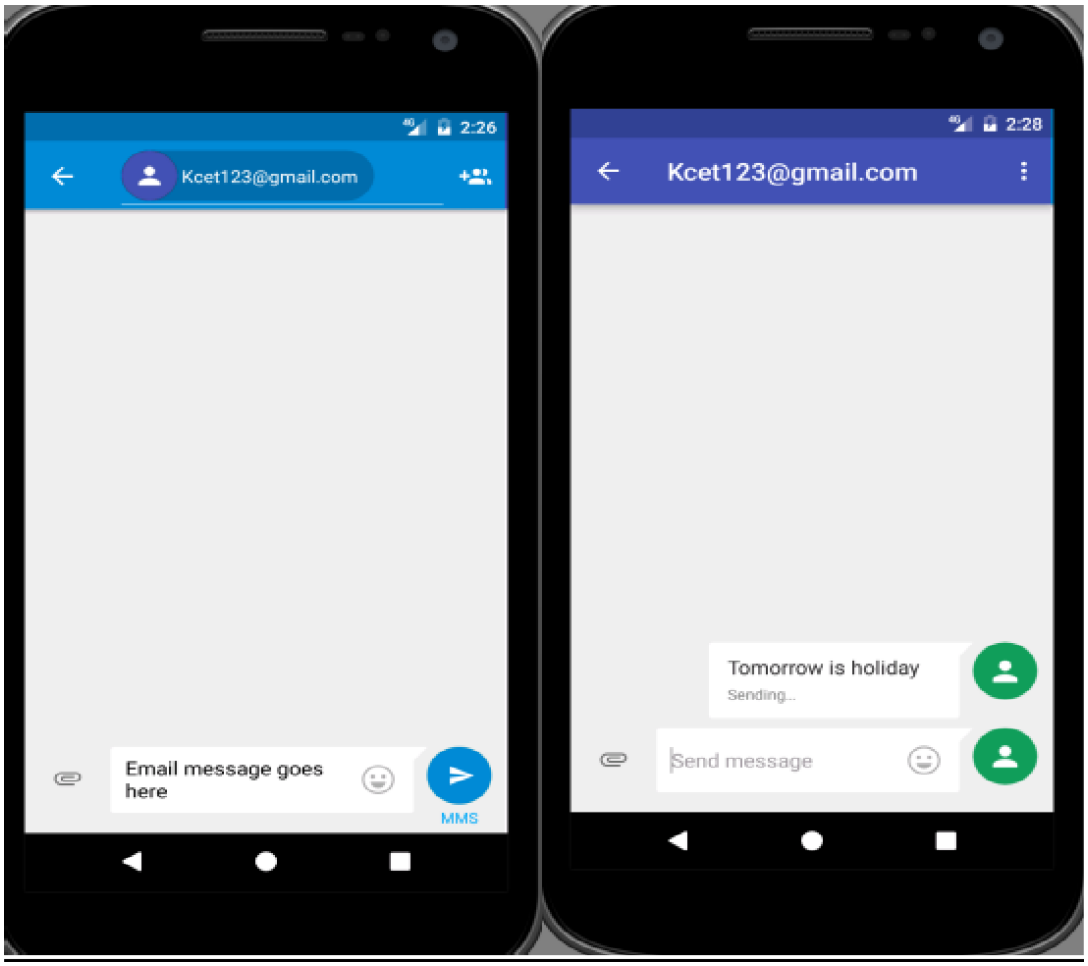
```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayoutxmlns: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"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
```

```
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.admin.myapplication.MainActivity">

<EditTextandroid:layout_width="wrap_content"
android:layout_height="wrap_content"
android:inputType="textEmailAddress"
android:ems="10"
```

```
android:id="@+id/editText"
android:layout_alignParentTop="true"
android:layout_alignParentRight="true"
android:layout_alignParentEnd="true" />

<EditText android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:inputType="textEmailAddress"
android:ems="10" android:id="@+id/editText2"
android:layout_below="@+id/editText"
android:layout_alignRight="@+id/editText"
android:layout_alignEnd="@+id/editText" />

<EditText android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:inputType="textEmailAddress"
android:ems="10" android:id="@+id/editText3"
android:layout_below="@+id/editText2"
android:layout_alignRight="@+id/editText2"
android:layout_alignEnd="@+id/editText2" />

<Button android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="SEND MAIL"
android:id="@+id/sendbttn"
android:layout_centerVertical="true"
android:layout_alignLeft="@+id/editText3"
android:layout_alignStart="@+id/editText3" />

<TextView android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Recipient"
android:id="@+id/textView"
android:layout_alignBottom="@+id/editText"
android:layout_alignParentLeft="true"
android:layout_alignParentStart="true" />

<TextView android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="subject"
android:id="@+id/textView2"
android:layout_alignBottom="@+id/editText2"
android:layout_alignParentLeft="true"
android:layout_alignParentStart="true" />

<TextView android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Message Body"
android:id="@+id/textView3"
android:layout_alignBottom="@+id/editText3"
android:layout_alignParentLeft="true"
android:layout_alignParentStart="true" />

</RelativeLayout>
```

OUTPUT:


2:20
SEND AN EMAIL
Recipient
subject
Message Body
SEND MAIL
2:23
SEND AN EMAIL
Recipient
KCET
subject
Check program
Message Body
Hai
SEND MAIL



Result:

Thus, the program was implemented and executed successfully.

Ex.No: 7 **Develop a simple shopping application**

Aim:

To develop a simple shopping application

Procedure:

MainActivity.java

```
package com.javatpoint.optionmenu;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;
public class MainActivity extends Activity
{
@Override
protected void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
}
@Override
public booleanonCreateOptionsMenu(Menu menu)
{
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.main, menu);//Menu Resource, Menu
return true;
}
@Override
public boolean onOptionsItemSelected(MenuItem item)
{
switch (item.getItemId())
{
case R.id.item1:
Toast.makeText(getApplicationContext(),"Item 1 Selected",Toast.LENGTH_LONG).show();
```

```
return true;
case R.id.item2:
Toast.makeText(getApplicationContext(),"Item 2 Selected",Toast.LENGTH_LONG).show();
return true;
case R.id.item3:
Toast.makeText(getApplicationContext(),"Item 3 Selected",Toast.LENGTH_LONG).show();
return true;
default:
return super.onOptionsItemSelected(item);
}
}
}
```

MainActivity.xml

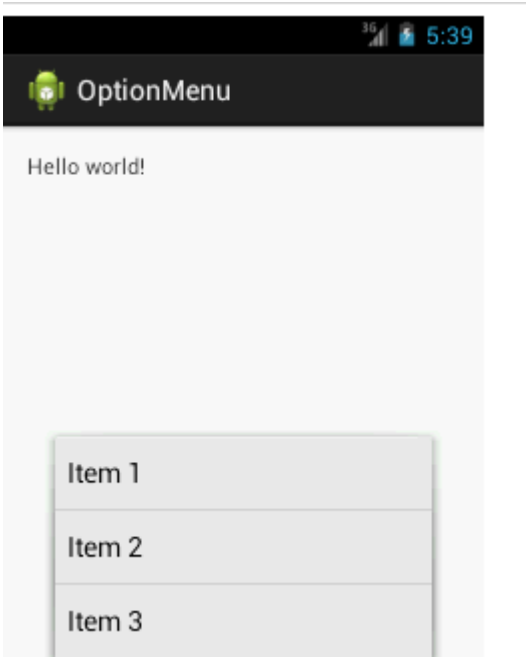
```
<RelativeLayout xmlns:androclass="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context=".MainActivity" >
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="@string/hello_world" />
</RelativeLayout>
```

SecondActivity.xml

```
<menu xmlns:androclass="http://schemas.android.com/apk/res/android" >
<item android:id="@+id/item1"
android:title="Item 1"/>
<item android:id="@+id/item2"
android:title="Item 2"/>
```

```
<item android:id="@+id/item3"
android:title="Item 3"/>
</menu>
```

OUTPUT:



Result:

Thus, the program was implemented and executed successfully

Ex.No: 8 **Design a web server supporting push notifications.**

Aim:

To develop a web server supporting push notifications.

Algorithm:

1. Create a New Android Project:
   - Click New in the toolbar.
   - In the window that appears, open the Android folder, select Android Application Project, and click next.
   - Provide the application name and the project name and then finally give the desired package name.
   - Choose a launcher icon for your application and then select Blank Activity and then click Next
   - Provide the desired Activity name for your project and then click Finish.
2. Create a New AVD (Android Virtual Device):
   - click Android Virtual Device Manager from the toolbar.
   - In the Android Virtual Device Manager panel, click New.
   - Fill in the details for the AVD. Give it a name, a platform target, an SD card size, and a skin (HVGA is default).
   - Click Create AVD and Select the new AVD from the Android Virtual Device Manager and click Start.
3. Design the layout by adding a text box and a command button.
4. Run the application.
5. If the entered E-mail doesn't match the given E-mail id, then an alert will be displayed.
6. If the entered E-mail id matches with the provided mail-id then login is successful.
7. Close the Android project.

PROGRAM CODE:

MainActivity.java

```
package com.pa.Alert;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
```

```
public class MainActivity extends Activity {
	private Button BTN;
	private EditText email;

	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_main);
		BTN = (Button) findViewById(R.id.btn);
		email = (EditText) findViewById(R.id.emailInput);
		BTN.setOnClickListener(new OnClickListener() {
			public void onClick(View v) {
				String val = email.getText().toString();
				if (val == null || val.length() <= 0) {
					Toast.makeText(getApplicationContext(),
							"Please Enter the email", Toast.LENGTH_LONG).show();
				} else if (val.equals("enpboss@gmail.com")) {
							Intent intent = new Intent(getApplicationContext(),
									SecondActivity.class);
							startActivity(intent);
							Toast.makeText(getApplicationContext(),
									"Login Success", Toast.LENGTH_LONG).show();
				 } else {
					Toast.makeText(getApplicationContext(),
					"Please Enter valid email", Toast.LENGTH_LONG)
											.show();
					}
			}
		});
	}
}
```

SecondActivity.java

```
package com.pa.Alert;

import android.app.Activity;
import android.os.Bundle;

public class SecondActivity extends Activity {
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		// TODO Auto-generated method stub
		super.onCreate(savedInstanceState);
		setContentView(R.layout.second_activity);
	}
}
```

Main_activity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayoutxmlns: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"
android:orientation="vertical" >

<EditText
android:id="@+id/emailInput"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:ems="10" />

<Button
android:id="@+id/btn"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="20sp"
android:gravity="center"
android:text="Login" />

</LinearLayout>
```

AndroidMainfest.Xml

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

<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">

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

<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
<activity android:name=".SecondActivity">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```

OUTPUT:

Result:

Thus, the program was implemented and executed successfully.

Ex.No: 9 **Develop an application by integrating Google maps**

AIM:

To develop an android application that uses Google Map location information.

ALGORITHM:

1. Create a New Android Project:
   - Click New in the toolbar.
   - In the window that appears, open the Android folder, select Android Application Project, and click next.
   - Provide the application name and the project name and then finally give the desired package name.
   - Choose a launcher icon for your application and then select Blank Activity and then click Next
   - Provide the desired Activity name for your project and then click Finish.
2. Create a New AVD (Android Virtual Device):
   - click Android Virtual Device Manager from the toolbar.
   - In the Android Virtual Device Manager panel, click New.
   - Fill in the details for the AVD. Give it a name, a platform target, an SD card size, and a skin (HVGA is default).
   - Click Create AVD and Select the new AVD from the Android Virtual Device Manager and click Start.
3. Design the graphical layout.
4. Run the application.
5. The requested data is retrieved from the database named myFriendsDb.
6. Close the Android project.

PROGRAM CODE

UseGps.java

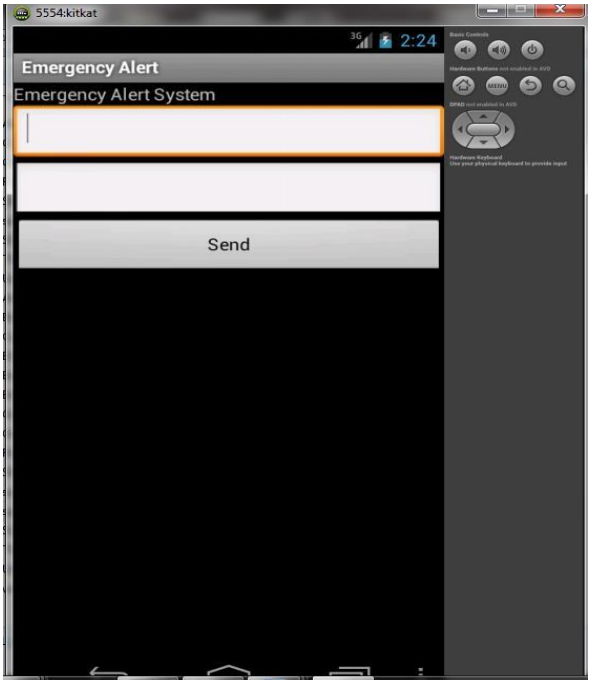
```
package com.emergency;
import android.app.Activity;
import android.content.Context;
import android.location.Location;
import  android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
```

```
public class UseGps extends Activity
{
        Button buttonSend;
        EditTexttextSMS;
        EditTexttextlon;
public void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
buttonSend = (Button) findViewById(R.id.buttonSend);
textSMS = (EditText) findViewById(R.id.editTextSMS);
textlon = (EditText) findViewById(R.id.textlon);
LocationManagermlocManager =
(LocationManager)getSystemService(Context.LOCATION_SERVICE);
LocationListenermlocListener = new MyLocationListener();
mlocManager.requestLocationUpdates( LocationManager.GPS_PROVIDER, 0, 0, mlocListener);
}
public class MyLocationListener implements LocationListener
{
public void onLocationChanged(Location loc)
{
loc.getLatitude();
loc.getLongitude();
Double lat=loc.getLatitude();
Double lon=loc.getLongitude();
textSMS.setText(lat.toString());
textlon.setText(lon.toString());
}
public void onProviderDisabled(String provider)
{
Toast.makeText( getApplicationContext(),"Gps Disabled",Toast.LENGTH_SHORT ).show();
}
public void onProviderEnabled(String provider)
{
Toast.makeText( getApplicationContext(), "Gps Enabled", Toast.LENGTH_SHORT).show();
}
}
}
```

main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
>
<TextView
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Emergency Alert System"
   />

<EditText
android:id="@+id/editTextSMS"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:gravity="top" />

<EditText
android:id="@+id/textlon"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:gravity="top" />

<Button
android:id="@+id/buttonSend"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Send" />

</LinearLayout>
```

OUTPUT:



Result:

Thus, the program for android application that makes use of Google Map was executed successfully.

Ex.No: 10 **Mini Projects involving Flutter/Kotlin multi-platform**

AIM:

To Write a mobile application that creates alarm clock.

PROCEDURE:

Create a new Android Application

1. In Eclipse go to **File->New->Project**
2. Select an **Android Project** from the Android Folder and press **Next.**
3. Fill in the details of your Android application.
    a. **Project Name**: The project name and folder that Eclipse will store the project files
    b. **Build Target:** The version of the Android SDK that will be used when you build your program. Select a platform that is equal to or lower than the target chosen for the AVD.
    c. **Application Name:** This is the name of the application.
    d. **Package Name**: The namespace that all of the source code will reside under.
    e. **Create Activity:** The name for that class stub that is generated by the plugin.
4. The values that are used in this example are:
    a. **Project Name**: Alarm
    b. **Build Target:** 2.3.3
    c. **Application Name:** Alarm
    d. **Package Name**: com. Alarm.example
    e. **Create Activity:** Alarm
5. Click on

Finish. Coding the Application

1. Open **AndroidMainfest.xml** which is located in **res->values->** **AndroidMainfest.xml.** This file will hold all of the text that our layout will use.
2. Click on the **AndroidMainfest**.xml at the bottom to bring up the raw xml file.

Editing the the java code

1. Open SampleApp.java from the left hand side.
2. Save the files.

Running the Application

1. Click on the green circle with the white arrow.
2. Choose the AVD that we created in a previous step.
3. The android AVD will load and the program will run.

PROGRAMS

FileName :MainActivity.java

```
Package
com.lab.alarmclock;
import java.util.Calendar;
import
android.app.Activity;
import
android.app.AlarmManager;
import
android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import
android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TimePicker;

public class AlarmActivity extends Activity
                        {

        private TimePicker
        timepicker; private Context
        context; private Button
        btnSetAlarm;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
                // TODO Auto-generated method stub
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);

               context = this;

                timepicker = (TimePicker) findViewById(R.id.timepicker);

                btnSetAlarm = (Button) findViewById(R.id.btnSetAlarm);
                btnSetAlarm.setOnClickListener(new OnClickListener() {

                        @Override
                        public void onClick(View v) {
                                // TODO Auto-generated method stub
```

```
                    Calendar calendar = Calendar.getInstance();
                    calendar.set(Calendar.HOUR_OF_DAY,
                    timepicker.getCurrentHour()); calendar.set(Calendar.MINUTE,
                    timepicker.getCurrentMinute());

                    Intent myIntent = new Intent(context, AlarmReceiver.class);
                    PendingIntent pendingIntent = PendingIntent.getBroadcast(
                                context, 0, myIntent, 0);

                    AlarmManager alarmManager =
(AlarmManager) getSystemService(ALARM_SERVICE);
                    alarmManager.set(AlarmManager.RTC, calendar.getTimeInMillis(),
                                pendingIntent);

                }
            });

        };}
```

```
}
```

File Name: Alaram Reciever.java

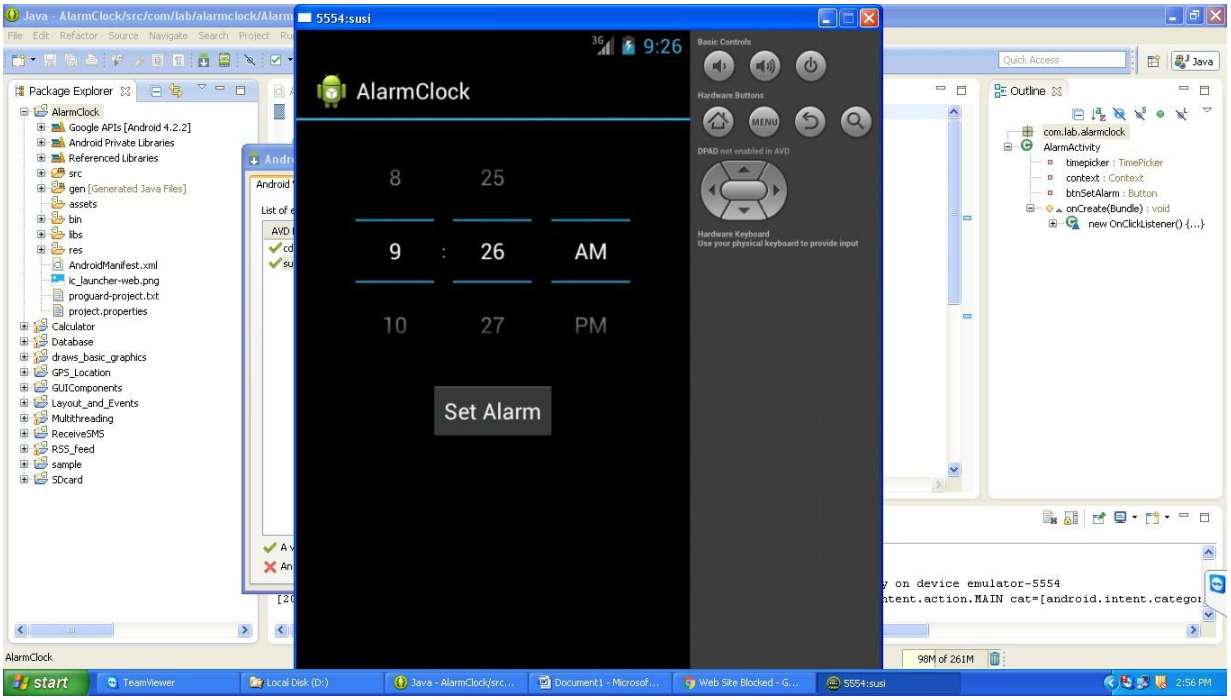
```
package com.lab.alarmclock;

import
android.content.Context;
import android.content.Intent;
import
android.media.Ringtone;
import
android.media.RingtoneManager;
import android.net.Uri;
import
android.support.v4.content.WakefulBroadcastReceiver;
import android.util.Log;
import android.widget.Toast;

public class AlarmReceiver extends

        WakefulBroadcastReceiver { @Override
        public void onReceive(Context context, Intent intent) {
                // TODO Auto-generated method stub

                Log.e("alarmreceiver", "alarmreceiver");
                Toast.makeText(context, "alarmreceiver", Toast.LENGTH_LONG).show();

                Uri alarmUri = RingtoneManager
                                .getDefaultUri(RingtoneManager.TYPE_ALARM);

                Ringtone ringtone = RingtoneManager.getRingtone(context, alarmUri);

                ringtone.play();

        }
}
```

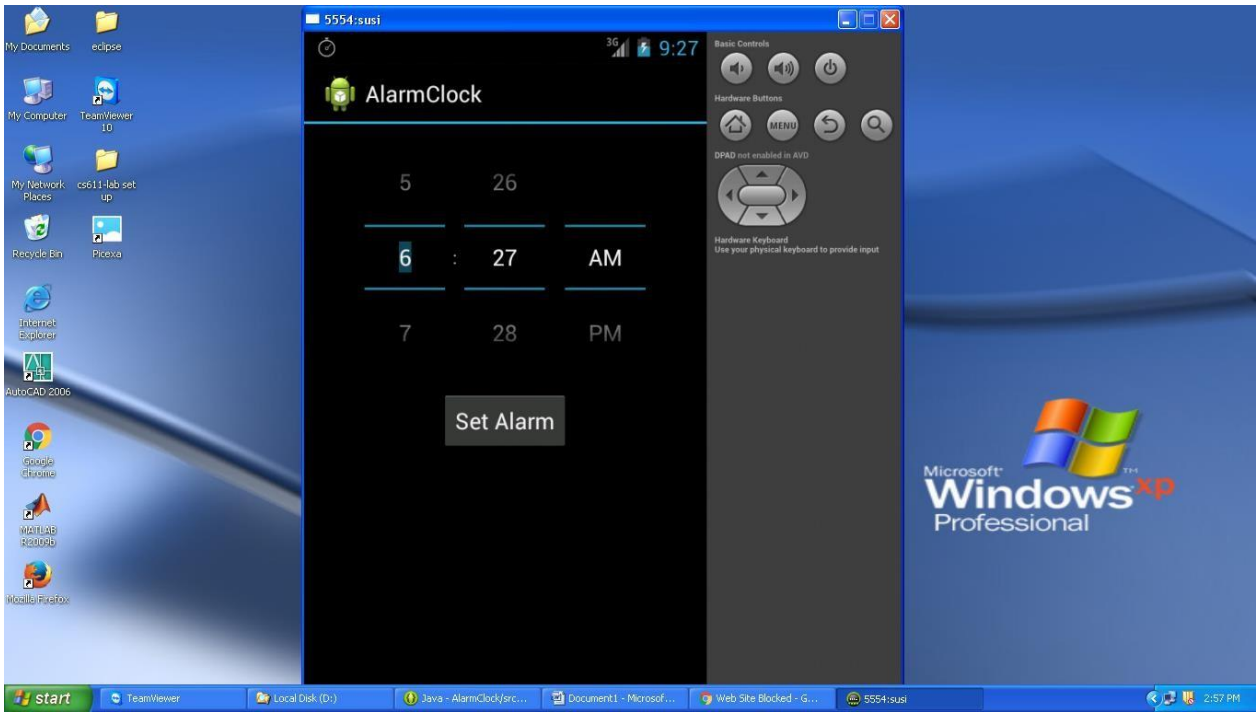
File Name: Androidmainfest.xml

```
<?xml version="1.0" encoding="utf-8"?>

<manifest
   xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.lab.alarmclock"

   android:versionCode="1"
   android:versionName="1.0"
   >

   <uses-permission android:name="android.permission.WAKE_LOCK" />

   <uses-sdk
      android:minSdkVersion="8"
      android:targetSdkVersion="21"
      />
```

```
    <application
        android:allowBackup="true
        "
        android:icon="@drawable/ic_launche
        r" android:label="@string/app_name"
        android:theme="@style/AppTheme" >

        <activity
            android:name=".AlarmActivity"
            android:label="@string/app_name"
            >

            <intent-filter>
                <action android:name="android.intent.action.MAIN" />


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

            </intent-filter>
        </activity>
        <receiver android:name=".AlarmReceiver" />

    </application>

</manifest>
```

OUTPUT:





RESULT:

Thus the mobile application that creates alarm clock