#### TOLANI COLLEGE OF COMMERCE

***(Affiliated to University of Mumbai)***

#### MUMBAI-MAHARASHTRA-400093 DEPARTMENT OF INFORMATION TECHNOLOGY

**CERTIFICATE**

This is to certify that, Ms. Mohammed Iqram Iqbal Patel Seat No. 35, studying in **Bachelor of Science Information Technology** has satisfactorily completed the Practical in the subject of **Advanced Mobile Programming** as prescribed by University of Mumbai, during the academic year 2019 - 2020.

Internal Examiner Co-ordinator

External Examiner

Date: College Seal

**INDEX**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr. No** | **Practical** | **Title** | **Page No** | **Date** | **Sign** |
| 1 | **1** | Introduction to Android, Introduction to Android Studio IDE, Application Fundamentals | **3** |  |  |
| 2 | **2** | Programming Resources | **8** |  |  |
| 3 | **3** | Programming Activities and fragments | **11** |  |  |
| 4 | **4** | Programs related to different Layouts | **15** |  |  |
| 5 | **5** | Programming UI elements | **25** |  |  |
| 6 | **6** | Programming menus, dialog, dialog fragments | **28** |  |  |
| 7 | **7** | Programs on Intents, Events, Listeners and Adapters | **36** |  |  |
| 8 | **8** | Programs on Services, notification and broadcast receivers | **39** |  |  |
| 9 | **9** | Database Programming with SQLite | **46** |  |  |
| 10 | **10** | Programming Security and permissions | **58** |  |  |
| 11 | **11A** | Programming Media API | **66** |  |  |
| 12 | **11B** | Programming Telephone API | **69** |  |  |

**Aim:-**

## PRACTICAL 1

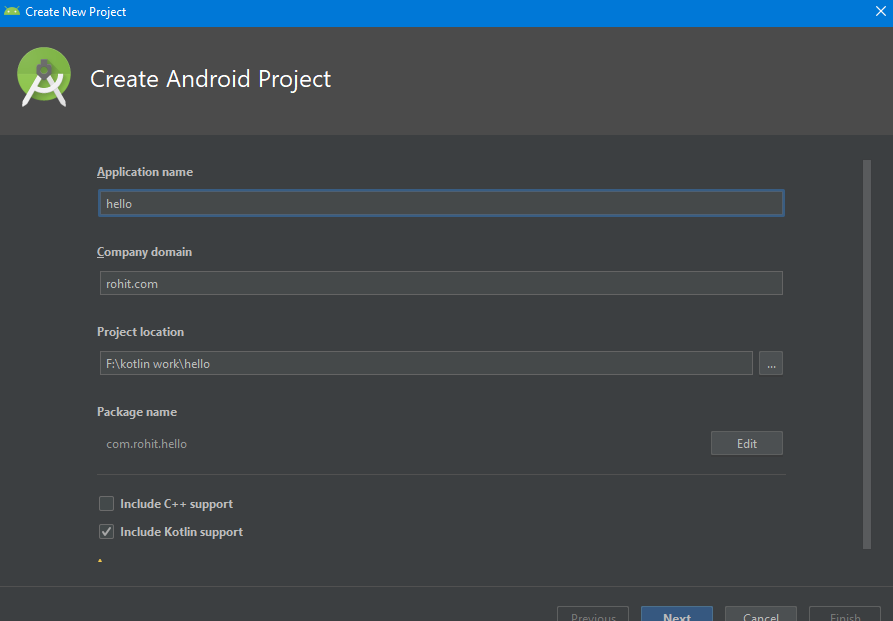
Introduction to Android, Introduction to Android Studio IDE, Application Fundamentals: Creating a Project, Android Components, Activities, Services, Content Providers, Broadcast Receivers, Interface overview, Creating Android Virtual device, USB debugging mode, Android Application Overview. Simple “Hello World” program.

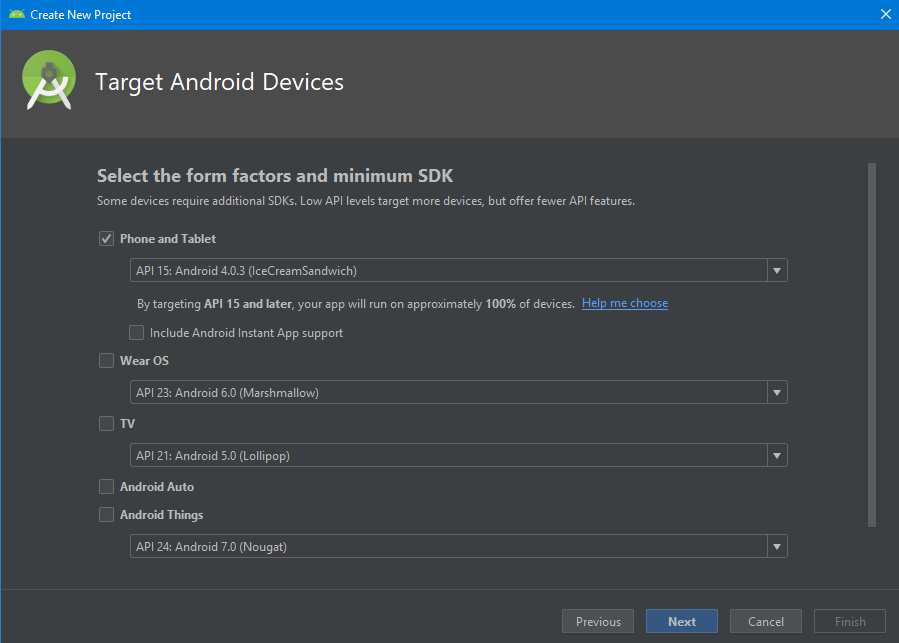
**Description:-**

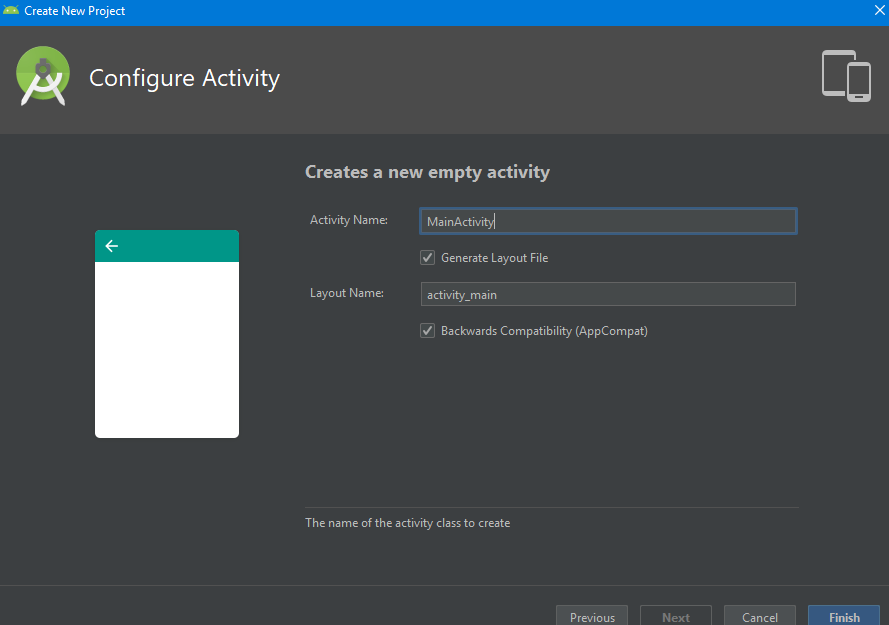
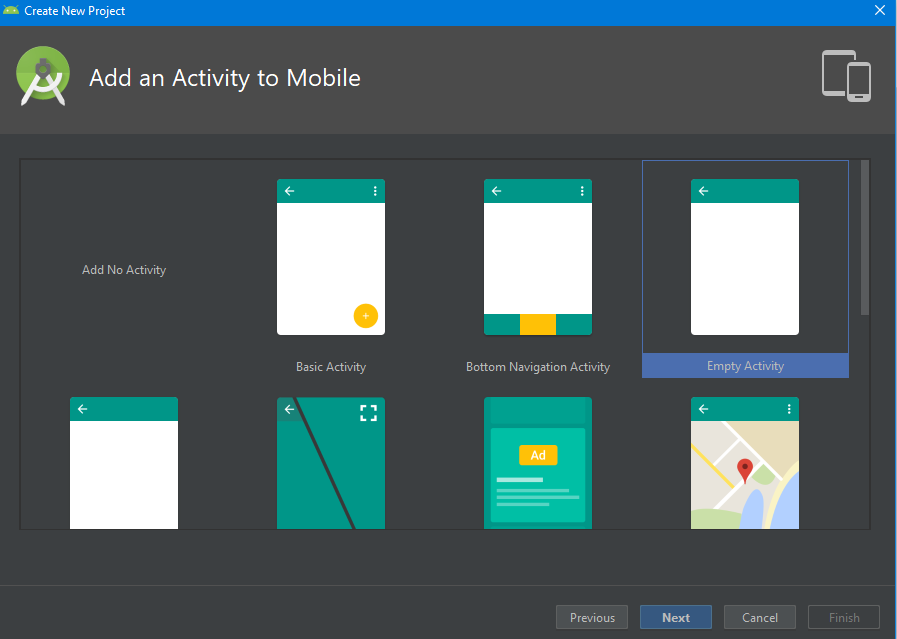
Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems

### Solution:-

Creating a project:



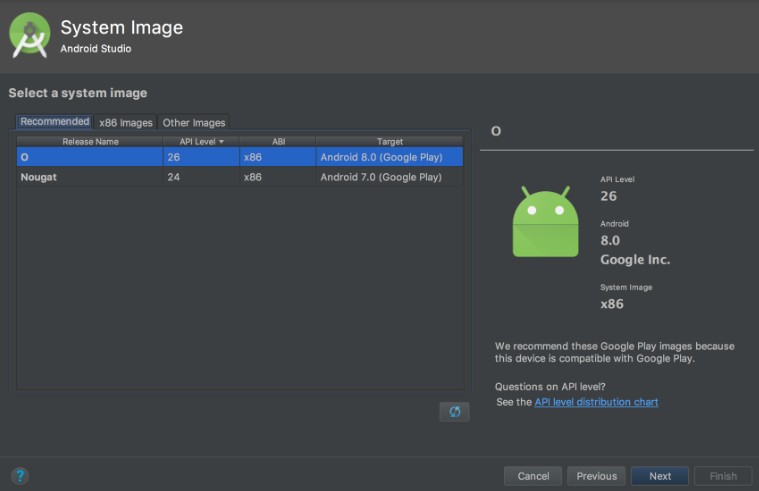
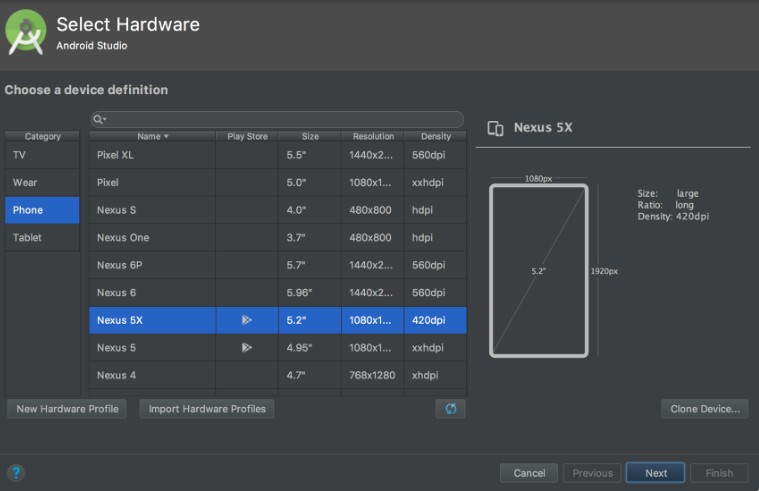
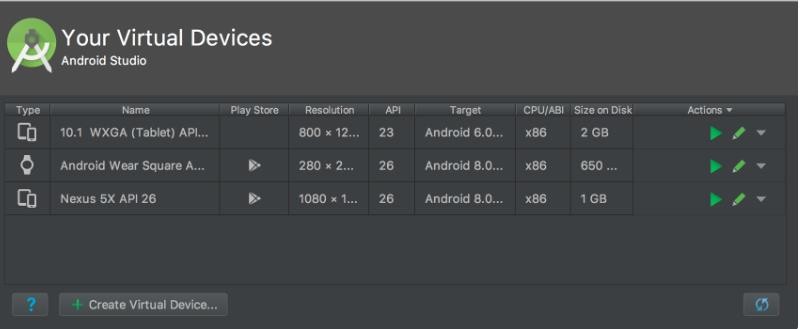




**Create and manage virtual devices:**

To open the AVD Manager, do one of the following:

* Select Tools > AVD Manager.
* Click AVD Manager AVD Manager icon in the toolbar.





#### Activity\_Main.Kt

**package** com.bscit.hello

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

**class** MainActivity : AppCompatActivity() {

**override fun** onCreate(savedInstanceState: Bundle?) { **super**.onCreate(savedInstanceState) setContentView(R.layout.*activity\_main*)

}

}

Activity\_Main.xml

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

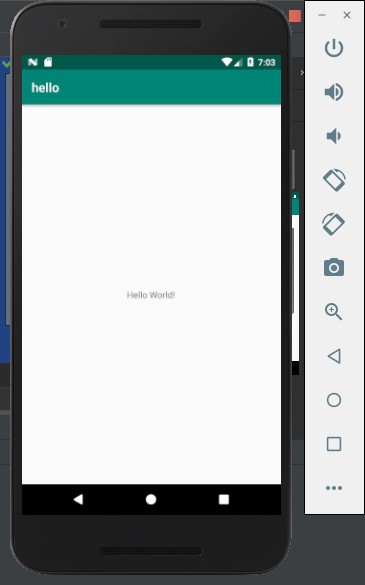
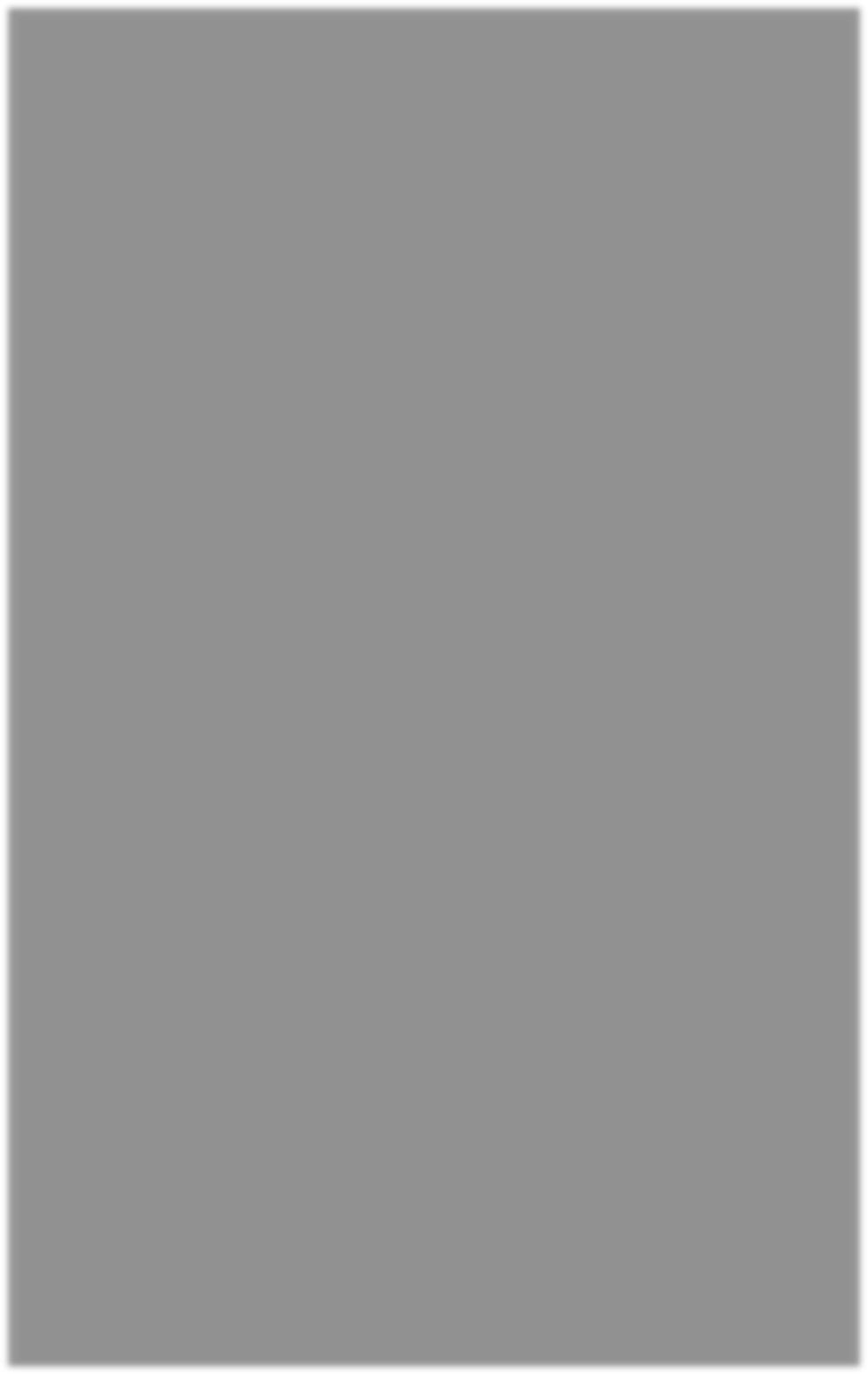
<android.support.constraint.ConstraintLayout xmlns:android="<http://schemas.android.com/apk/res/android>" xmlns:tools="<http://schemas.android.com/tools>" xmlns:app="[http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) android:layout\_width="match\_parent" android:layout\_height="match\_parent" tools:context=".MainActivity">

<TextView android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="Hello World!"

app:layout\_constraintBottom\_toBottomOf="parent" app:layout\_constraintLeft\_toLeftOf="parent" app:layout\_constraintRight\_toRightOf="parent" app:layout\_constraintTop\_toTopOf="parent"/>

</android.support.constraint.ConstraintLayout>

**Output:-**



## Practical 2

**Aim:**Programming Android Resources: Colour, Theme, String, Drawable, Dimension, Image

**Description:**Android Studio helps you add new resources and alternative resources in several ways, depending on the type of resource you want to add.

**Solution:**

#### activity\_main.xml

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

<LinearLayout

xmlns:android="<http://schemas.android.com/apk/res/android>" xmlns:app="[http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) xmlns:tools="<http://schemas.android.com/tools>" android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:orientation="vertical" android:background="@drawable/abc" tools:context=".MainActivity">

<TextView android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"

android:layout\_marginLeft="120dp" android:hint="@string/mystring"

android:textSize="30dp" android:textColorHint="@color/tx" app:layout\_constraintBottom\_toBottomOf="parent" app:layout\_constraintLeft\_toLeftOf="parent" app:layout\_constraintRight\_toRightOf="parent" app:layout\_constraintTop\_toTopOf="parent"/>

<ImageView android:layout\_width="match\_parent" android:layout\_height="wrap\_content" app:srcCompat="@mipmap/ic\_launcher"

tools:layout\_editor\_absoluteY="0dp" tools:layout\_editor\_absoluteX="3dp" android:id="@+id/imageView"

tools:ignore="MissingConstraints" android:layout\_alignParentStart="true" android:layout\_alignParentTop="true" android:layout\_marginTop="143dp" android:layout\_alignParentLeft="true"/>

</LinearLayout>

#### strings.xml

<resources>

<**string name="app\_name"**>My Application</**string**>

<**string name="mystring"**>"WELCOME"</**string**>

</**resources**>

#### color.xml

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

<resources>

<**color name="colorPrimary"**>#008577</**color**>

<**color name="colorPrimaryDark"**>#00574B</**color**>

<**color name="colorAccent"**>#D81B60</**color**>

<**color name="tx"**>#FFFFFF</**color**>

</**resources**>

#### MainActivity.kt

**package** com.example.gkaud.myapplication

**import** android.support.v7.app.AppCompatActivity

**import** android.os.Bundle

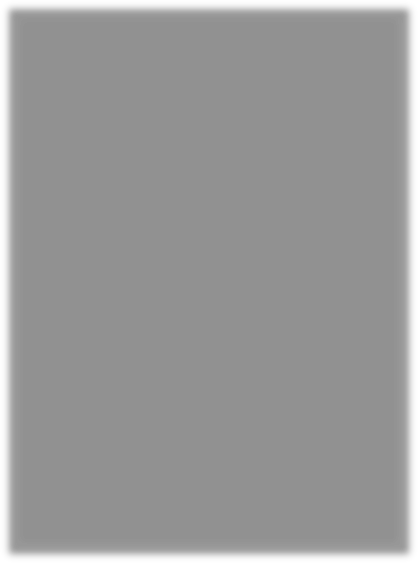
**class** MainActivity : AppCompatActivity() {

**override fun** onCreate(savedInstanceState: Bundle?) { **super**.onCreate(savedInstanceState) setContentView(R.layout.*activity\_main*)

}

}

**Output:-**



## Practical 3

**Aim:** Programming Activities and fragments: Activity Life Cycle, Activity methods, Multiple Activities, Life Cycle of fragments and multiple fragments.

**Description:**

A fragment is an independent Android component which can be used by an activity. A fragment encapsulates functionality so that it is easier to reuse within activities and layouts.

### Solution:

#### activity\_main.xml

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

<android.support.constraint.ConstraintLayoutxmlns:android="[http://schemas.android.](http://schemas.android/) com/apk/res/android"

xmlns:app="[http://schemas.android.com/apk/res-auto"](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=".MainActivity">

<TextView android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:textSize="90dp" android:text="Welcome" android:textColor="@color/colorPrimary"

app:layout\_constraintBottom\_toBottomOf="parent" app:layout\_constraintLeft\_toLeftOf="parent" app:layout\_constraintRight\_toRightOf="parent" app:layout\_constraintTop\_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>

#### MainActivity.java

**package** com.example.myapplication;

**import** android.os.Bundle; **import** android.app.Activity; **import** android.util.Log;

**public class** MainActivity**extends** Activity { String **msg**= **"Android : "**;

@Override

**public void** onCreate(Bundle savedInstanceState) { **super**.onCreate(savedInstanceState); setContentView(R.layout.***activity\_main***); Log.*d*(**msg**, **"The onCreate() event"**);

}

@Override

**protected void** onStart() {

**super**.onStart();

Log.*d*(**msg**, **"The onStart() event"**);

}

@Override

**protected void** onResume() {

**super**.onResume();

Log.*d*(**msg**, **"The onResume() event"**);

}

@Override

**protected void** onPause() {

**super**.onPause();

Log.*d*(**msg**, **"The onPause() event"**);

}

@Override

**protected void** onStop() {

**super**.onStop();

Log.*d*(**msg**, **"The onStop() event"**);

}

@Override

**public void** onDestroy() {

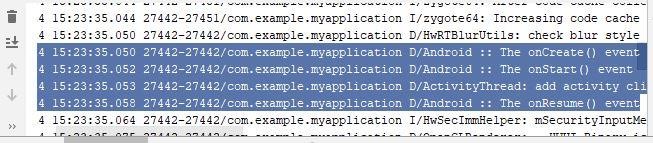
**super**.onDestroy();

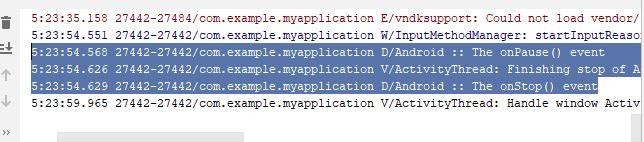
Log.*d*(**msg**, **"The onDestroy() event"**);

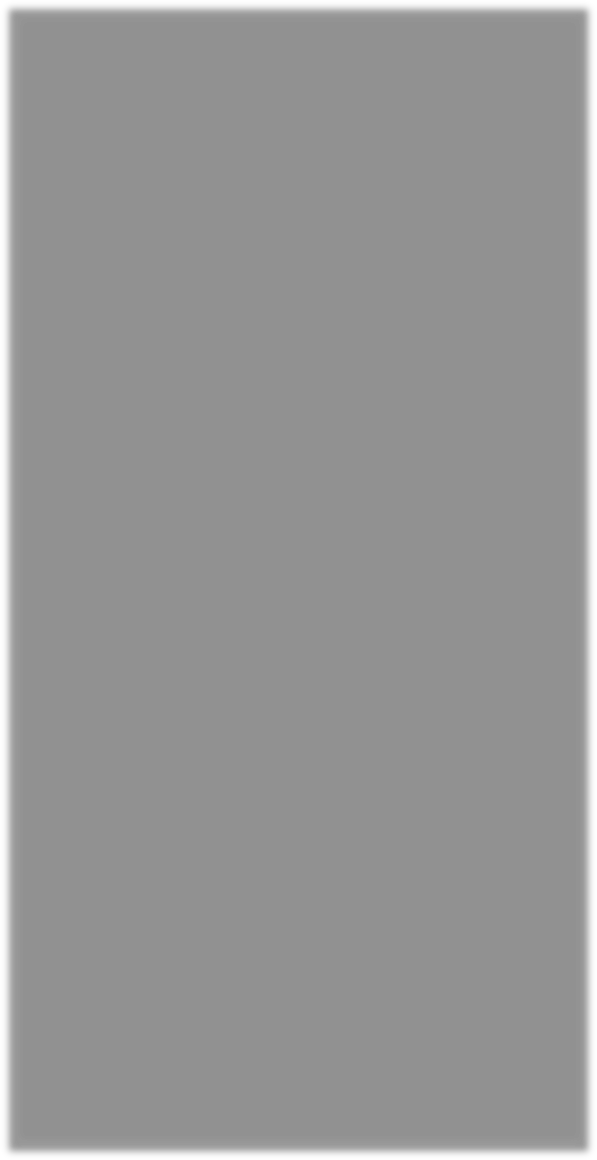
}

}

**Output: -**







## Practical 4

**Aim:**Programs related to different Layouts: Coordinate, Linear, Relative, Table, Absolute, Frame, List View, Grid View.

**Description:**

A layout defines the structure for a user interface in your app, such as in an activity. All elements in the layout are built using a hierarchy of View and ViewGroup objects. A View usually draws something the user can see and interact with. Whereas a ViewGroup is an invisible container that defines the layout structure for View and other ViewGroup objects.

### Solution:

#### Relative Layout and Linear Layout: - activity\_main.xml

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

<LinearLayoutxmlns:android="<http://schemas.android.com/apk/res/android>" xmlns:app="[http://schemas.android.com/apk/res-auto"](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=".MainActivity">

<RelativeLayout android:layout\_width="match\_parent" android:layout\_height="match\_parent" tools:layout\_editor\_absoluteX="16dp" tools:layout\_editor\_absoluteY="19dp">

<TextView android:id="@+id/Submit"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="TextView" />

<EditText android:id="@+id/Username" android:layout\_width="wrap\_content" android:layout\_height="48dp" android:layout\_marginLeft="50dp" android:layout\_marginTop="100dp" android:ems="10"

android:text="Name"

/>

<EditText android:id="@+id/Password" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_marginTop="20dp" android:layout\_below="@id/Username" android:ems="10" android:inputType="textPassword" android:text="Password" android:layout\_marginLeft="50dp"

/>

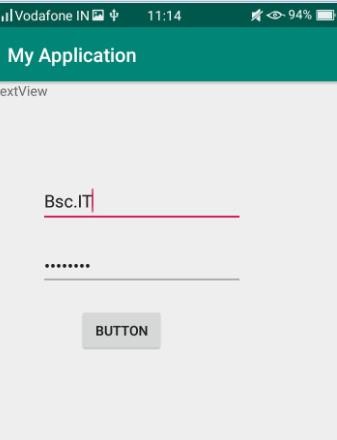
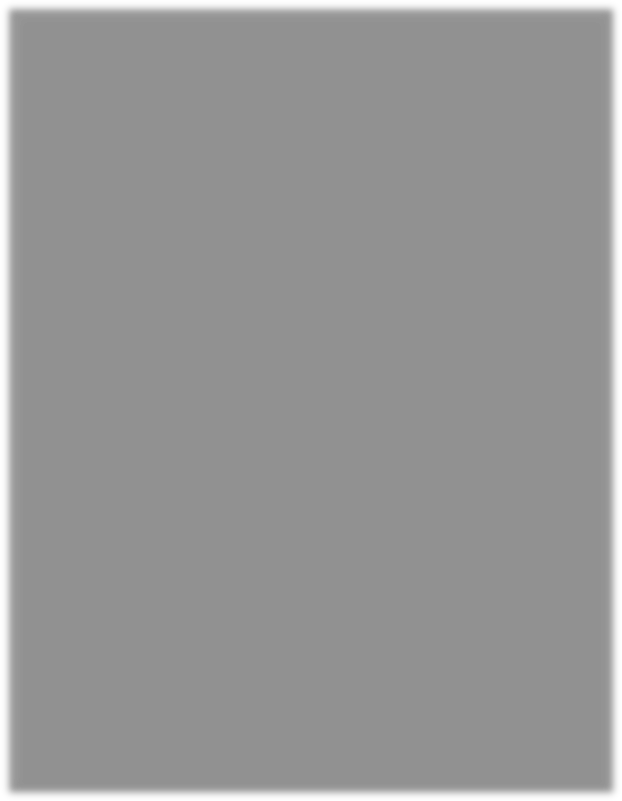
<Button android:id="@+id/bt1"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="Button" android:layout\_marginLeft="90dp" android:layout\_below="@id/Password" android:layout\_marginTop="20dp" tools:layout\_editor\_absoluteX="137dp" tools:layout\_editor\_absoluteY="363dp" />

</RelativeLayout>

</LinearLayout>

#### Output: -



**Table Layout activity\_main.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=".MainActivity">

<TableLayoutandroid:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_marginLeft="50dp" android:layout\_marginTop="150dp">

<TableRow>

<Button android:id="@+id/btn1" android:text="1" android:layout\_gravity="center"

/>

<Button android:id="@+id/btn2" android:text="2" android:layout\_gravity="center"

/>

<Button android:id="@+id/btn3" android:text="3" android:layout\_gravity="center"

/>

</**TableRow**>

<TableRow>

<Button android:id="@+id/btn4" android:text="4" android:layout\_gravity="center"

/>

<Button android:id="@+id/btn5" android:text="5" android:layout\_gravity="center"

/><Button android:id="@+id/btn6" android:text="6" android:layout\_gravity="center"

/>

</**TableRow**>

<TableRow>

<Button android:id="@+id/btn7" android:text="7" android:layout\_gravity="center"

/>

<Button android:id="@+id/btn8" android:text="8" android:layout\_gravity="center"

/><Button android:id="@+id/btn9" android:text="9" android:layout\_gravity="center"

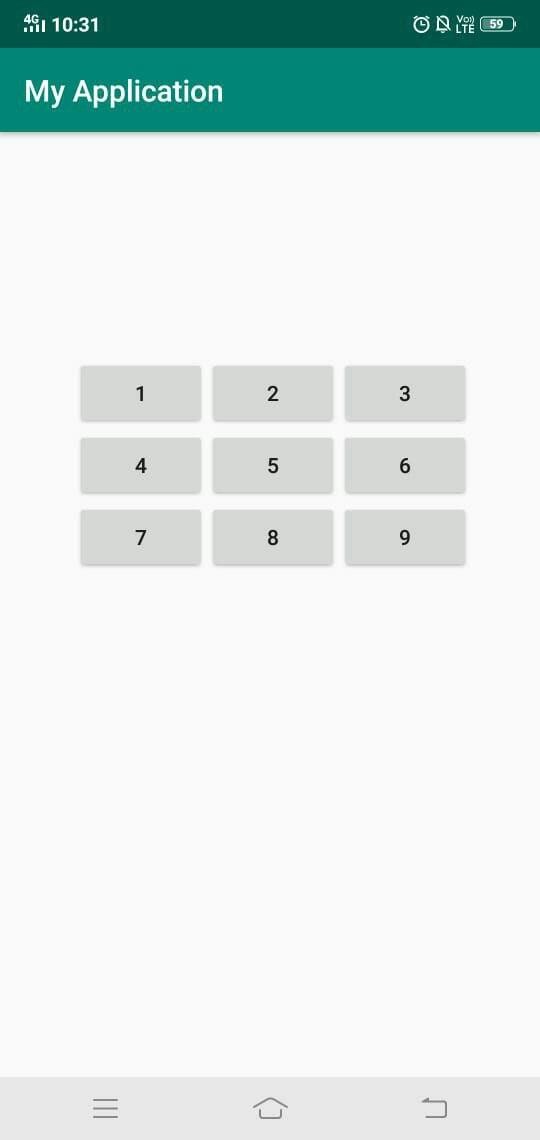
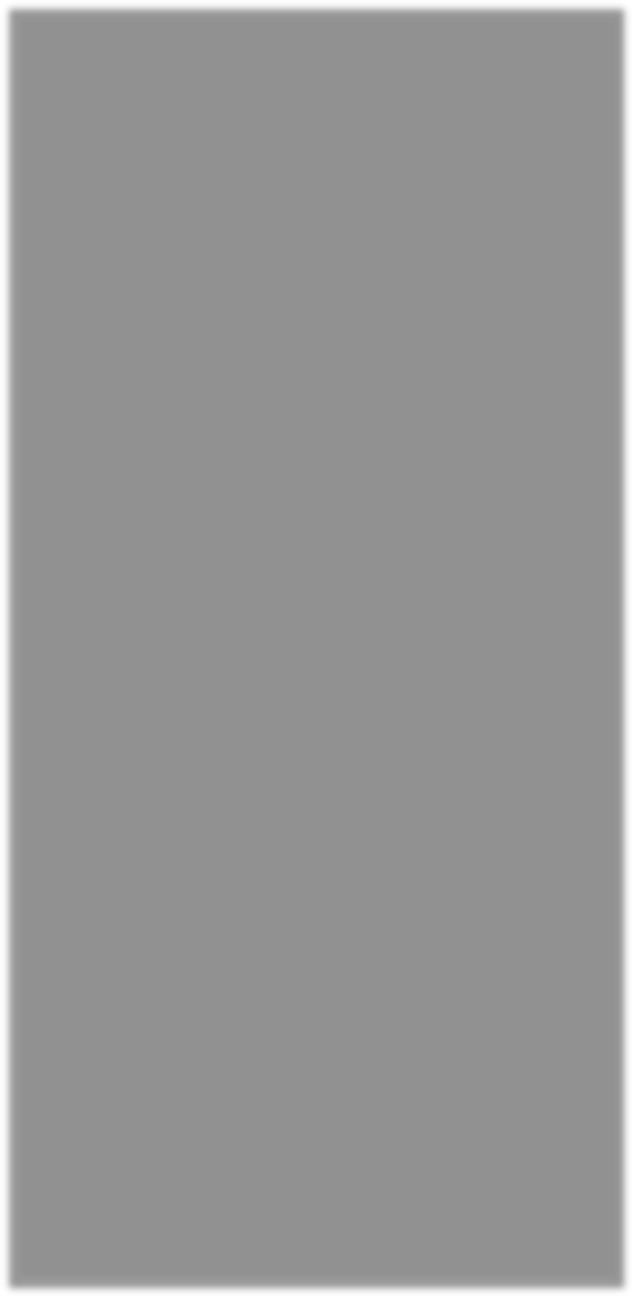
/>

</**TableRow**>

</**TableLayout**>

</LinearLayout>

### Output:



#### Frame:

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

<FrameLayout 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">

<ImageViewandroid:layout\_width="match\_parent" android:layout\_height="match\_parent" android:src="@drawable/red" android:scaleType="centerCrop"/>

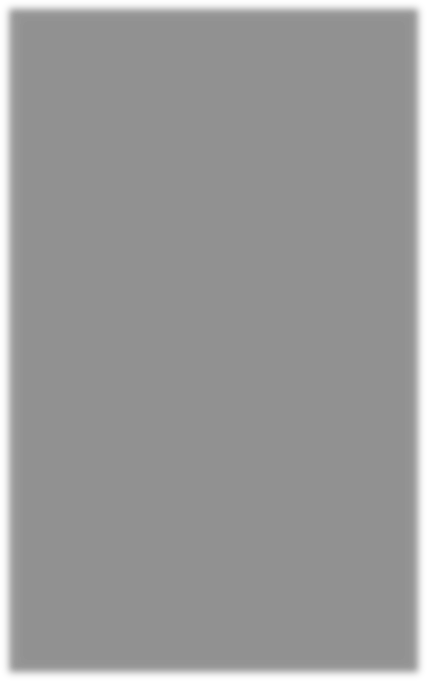
<TextView android:textSize="100dp"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="Hello World!" android:gravity="center" android:textColor="@color/rohit" android:layout\_marginTop="220dp"

/>

</**FrameLayout**>

### Output:



#### List Layout: - Activity\_main.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>" xmlns:app="[http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) android:layout\_width="match\_parent" android:layout\_height="match\_parent" tools:context=".MainActivity">

<Button android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:id="@+id/btn"

android:text="Click me to view list" android:layout\_marginTop="200dp" android:layout\_marginLeft="90dp"/>

</LinearLayout>

#### String.xml

<resources>

<**string name="app\_name"**>list</**string**>

<array name="insert\_list">

<**item**>one</**item**>

<**item**>two</**item**>

<**item**>three</**item**>

<**item**>four</**item**>

<**item**>five</**item**>

<**item**>six</**item**>

<**item**>seven</**item**>

<**item**>eight</**item**>

<**item**>nine</**item**>

<**item**>ten</**item**>

</**array**>

</**resources**>

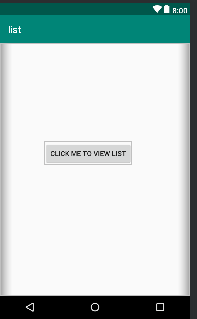
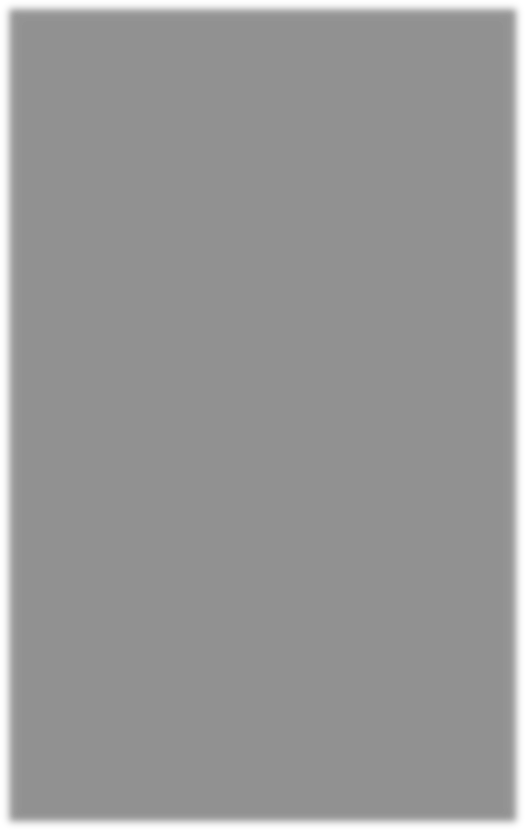
#### Activity\_list\_view.xml:

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

<ListView xmlns:android="<http://schemas.android.com/apk/res/android>" xmlns:tools="<http://schemas.android.com/tools>" xmlns:app="[http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) android:layout\_width="match\_parent" android:layout\_height="match\_parent" tools:context=".list\_view" android:entries="@array/insert\_list">

</**ListView**>

### Output: -



#### Grid layout: -

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

<GridLayout xmlns:android="<http://schemas.android.com/apk/res/android>" xmlns:tools="<http://schemas.android.com/tools>" xmlns:app="[http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) android:layout\_width="match\_parent" android:layout\_height="match\_parent" tools:context=".MainActivity"

android:rowCount="3" android:columnCount="3" android:padding="20dp">

<Button android:layout\_width="110dp" android:layout\_height="100dp" android:text="1"/>

<Button android:layout\_width="110dp" android:layout\_height="100dp" android:text="2"/>

<Button android:layout\_width="110dp"

android:layout\_height="100dp" android:text="3"/>

<Button android:layout\_width="110dp" android:layout\_height="100dp" android:text="4"/>

<Button android:layout\_width="110dp" android:layout\_height="100dp" android:text="5"/>

<Button android:layout\_width="110dp" android:layout\_height="100dp" android:text="6"/>

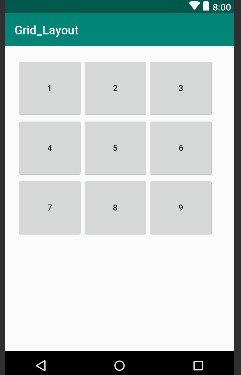
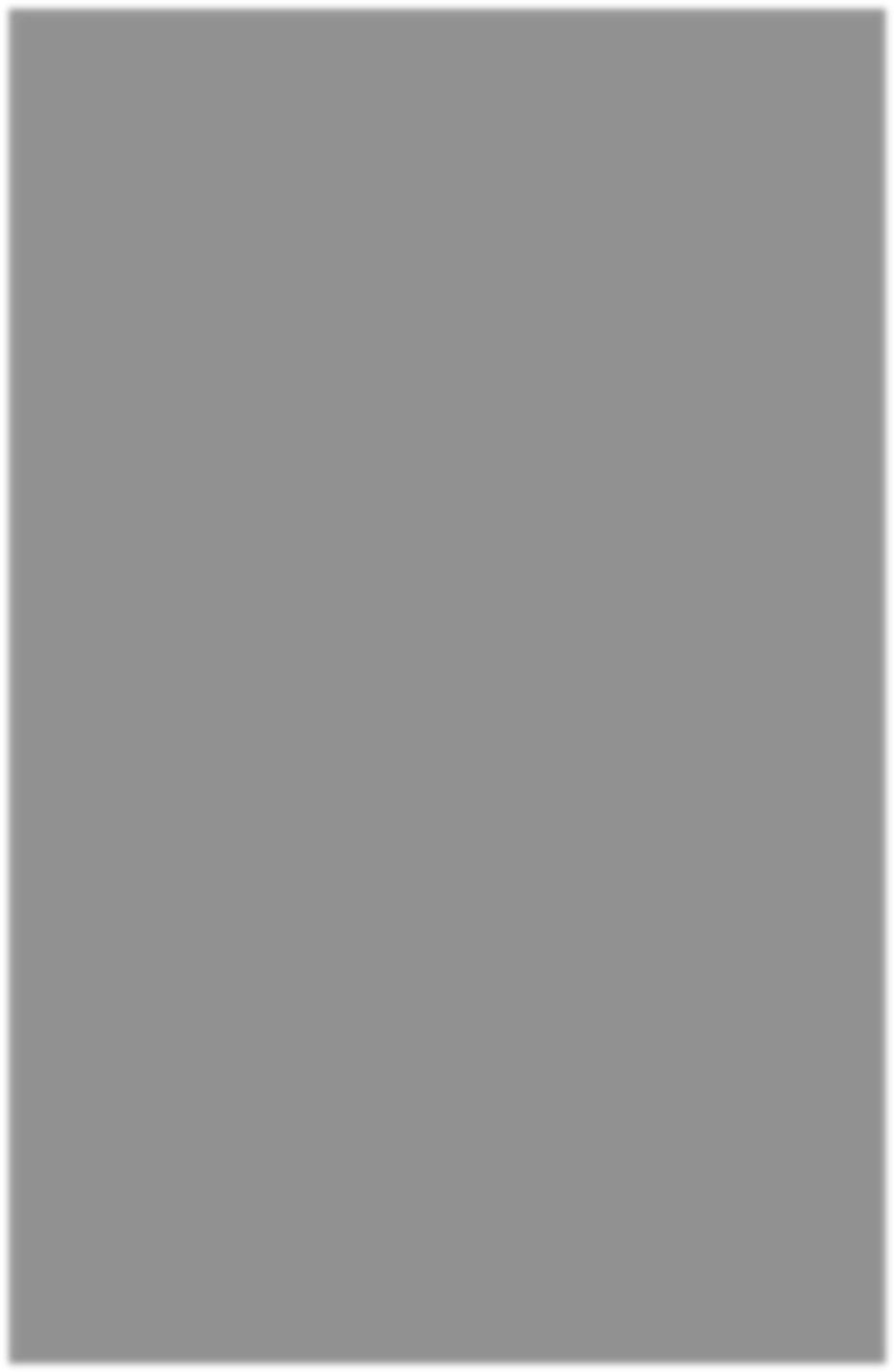
<Button android:layout\_width="110dp" android:layout\_height="100dp" android:text="7"/>

<Button android:layout\_width="110dp" android:layout\_height="100dp" android:text="8"/>

<Button android:layout\_width="110dp" android:layout\_height="100dp" android:text="9"/>

</**GridLayout**>

**Output: -**



# PRACTICAL 5

**Aim:** Programming UI elements: AppBar, Fragments, UI Components

**Description:**

Your app's user interface is everything that the user can see and interact with. Android provides a variety of pre-built UI components such as structured layout objects and UI controls that allow you to build the graphical user interface for your app. Android also provides other UI modules for special interfaces such as dialogs, notifications, and menus.

### Solution:

#### MainActivity.kt:

package bscit.technobeat

import android.content.Intent

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

import kotlinx.android.synthetic.main.activity\_login.\* import kotlinx.android.synthetic.main.activity\_main.\* import kotlinx.android.synthetic.main.activity\_register.\* import rohit.technobeat.R.id.login

import rohit.technobeat.R.id.newaccount class MainActivity : AppCompatActivity() {

override fun onCreate(savedInstanceState: Bundle?) { super.onCreate(savedInstanceState) setContentView(R.layout.*activity\_main*) login.setOnClickListener{

valintent = Intent(this, LoginActivity::class.*java*)

*// start your next activity*

startActivity(intent)

}

newaccount.setOnClickListener{

valintent = Intent(this, RegisterActivity::class.java)

*// start your next activity*

startActivity(intent)

}

}

}

#### activity\_main.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:gravity="center\_horizontal"

android:orientation="vertical" 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/home" tools:context=".MainActivity">

<ScrollView android:id="@+id/login\_form" android:layout\_width="match\_parent" android:layout\_height="match\_parent">

<LinearLayout android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:orientation="vertical" android:gravity="center">

<android.support.v7.widget.AppCompatTextView android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_marginTop="210dp" android:alpha="0.7" android:text="TECHNOBEAT" android:textColor="#000000" android:textSize="33dp" android:textStyle="bold" tools:layout\_marginLeft="85dp" />

<Button android:id="@+id/login"

style="?android:textAppearanceSmall" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:layout\_marginTop="16dp" android:text="Login" android:background="@drawable/round\_button" android:alpha="0.8"

android:textStyle="bold" />

<Button

android:id="@+id/newaccount" style="?android:textAppearanceSmall"

android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:layout\_marginTop="16dp" android:text="REGISTER" android:background="@drawable/round\_button" android:alpha="0.8"

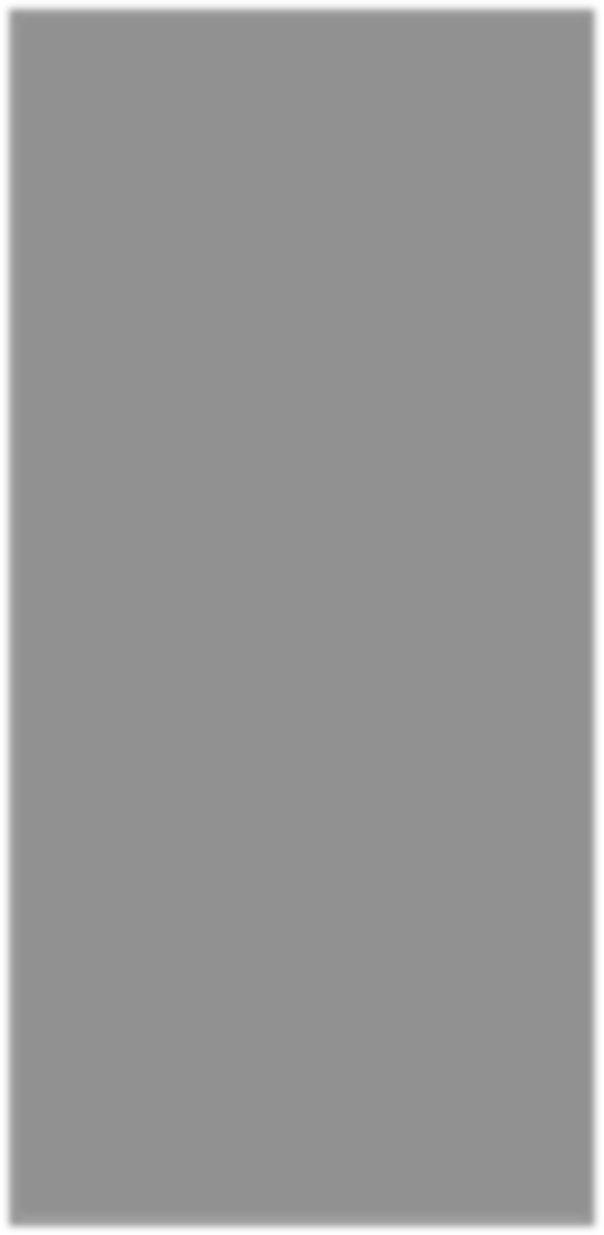
android:textStyle="bold" />

</LinearLayout>

</**ScrollView**>

</LinearLayout>

Output:-



**PRACTICAL 6**

**Aim**:Programming menus, dialog, dialog fragments

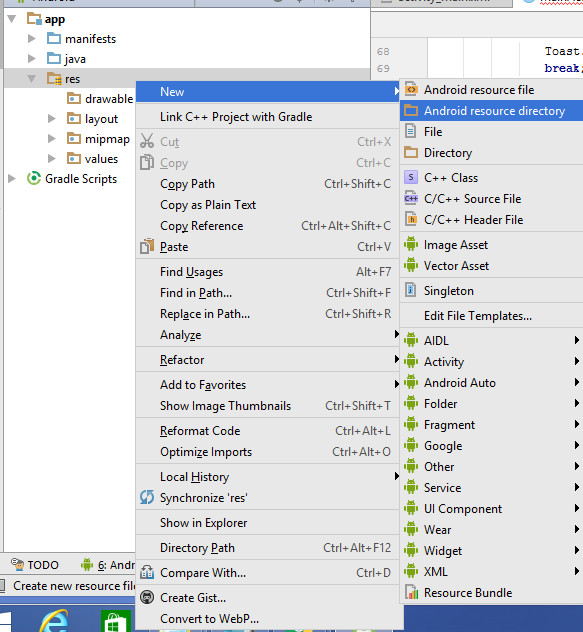
**Description:**

A dialog is a small window that prompts the user to make a decision or enter additional information. A dialog does not fill the screen and is normally used for modal events that require users to take an action before they can proceed.

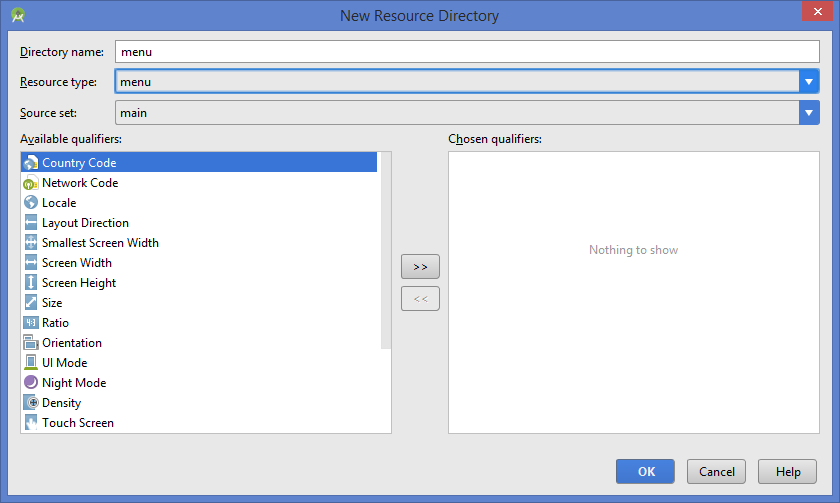
### Solution:

#### Menu.xml

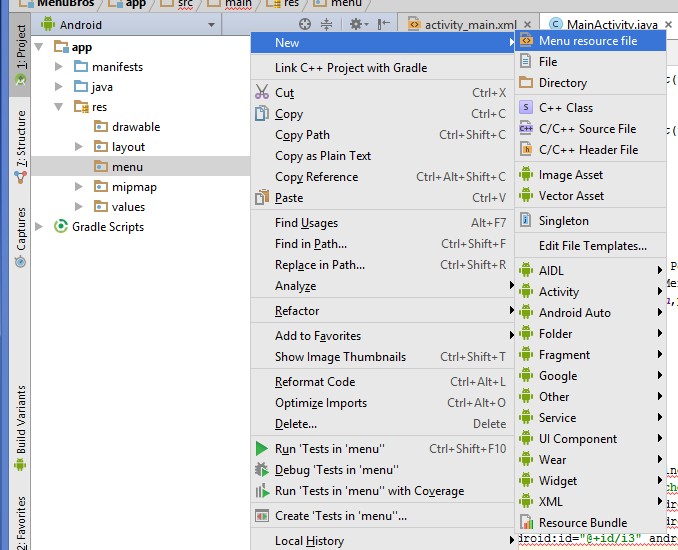
To define the menu\_file.xml file, first create a menu directory under res folder. This is done by right clicking on **res --> new --> Android resource directory**.



Then a new window will appear. Type menu in the directory name and choose menu in the Resource type. Then, click on OK.



A new menu directory would be made under res directory. Add menu\_file.xml file in menu directory by right clicking on **menu --> New --> Menu resource file**.



Give the name as **menu\_file.xml** and click on Ok. The **menu\_file.xml** file contains the following tags:

#### <menu>

It defines a Menu, which is a container for menu items. A <menu> element must be the root node for the file and can hold one or more <item> and

<group> elements.

#### <item>

It creates a MenuItem, which represents a single item in a menu. This element may contain a nested <menu> element in order to create a submenu.

#### <group>

It is an optional, invisible container for <item> elements. It allows you to categorize menu items so they share properties such as active state and visibility.

#### MainActivity.java

public class MainActivity extends AppCompatActivity { @Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState); setContentView(R.layout.*activity\_main*);

}

@Override

public booleanonCreateOptionsMenu(Menu menu) { MenuInflaterinflater = getMenuInflater(); inflater.inflate(R.menu.*menu\_file*, menu);

return true;

}

@Override

public booleanonOptionsItemSelected(MenuItem item) {

//Handle item selection switch (item.getItemId()) {

case R.id.*i1*:

//perform any action;

Toast.makeText(**this**, “Menu item **is** selected”, Toast.LENGTH\_SHORT).show()

return true; case R.id.*a*:

//perform any action;

Toast.makeText(**this**, “Menu submenu a **is** selected”, Toast.LENGTH\_SHORT).show() return true;

case R.id.*b*:

//perform any action;

Toast.makeText(**this**, “Menu sub menu b **is** selected”, Toast.LENGTH\_SHORT).show() return true;

default:

return super.onOptionsItemSelected(item);

}

}

}

#### menu\_file.xml

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

<menu xmlns:android="[http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android)>

<item android:id="@+id/i1" android:title="item" android:icon="@drawable/flower1" >

<!-- "item" submenu -->

<menu>

<item android:id="@+id/a" android:title="subitem a" android:icon="@drawable/flower1"/>

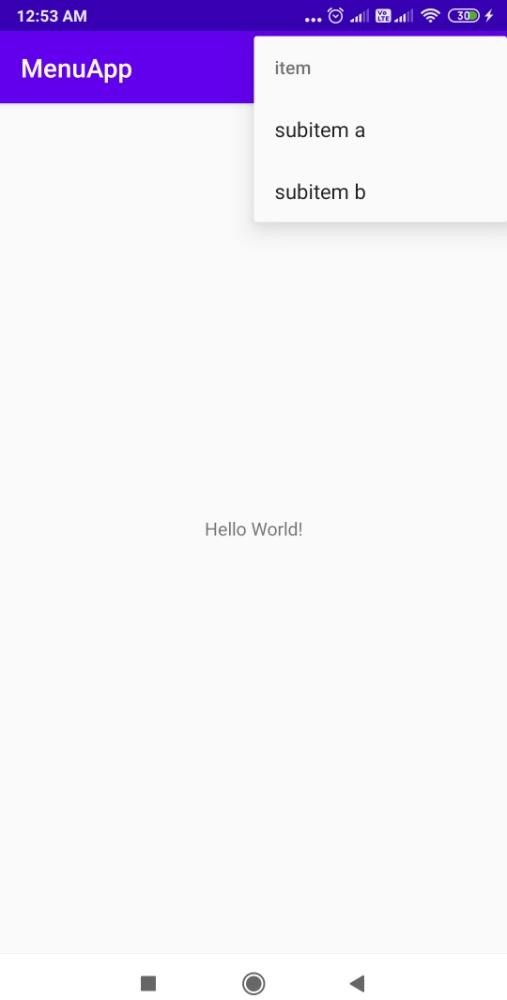
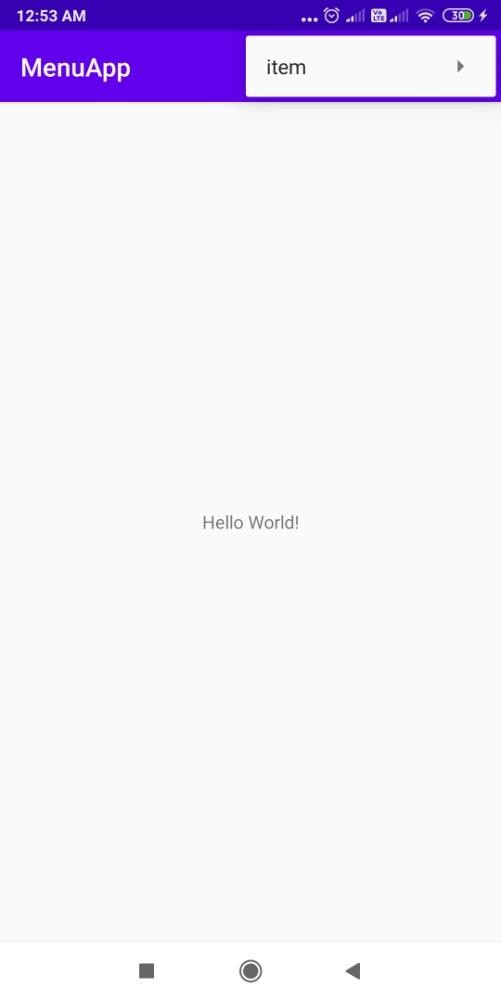
<item android:id="@+id/b" android:title="subitem b" android:icon="@drawable/flower2" />

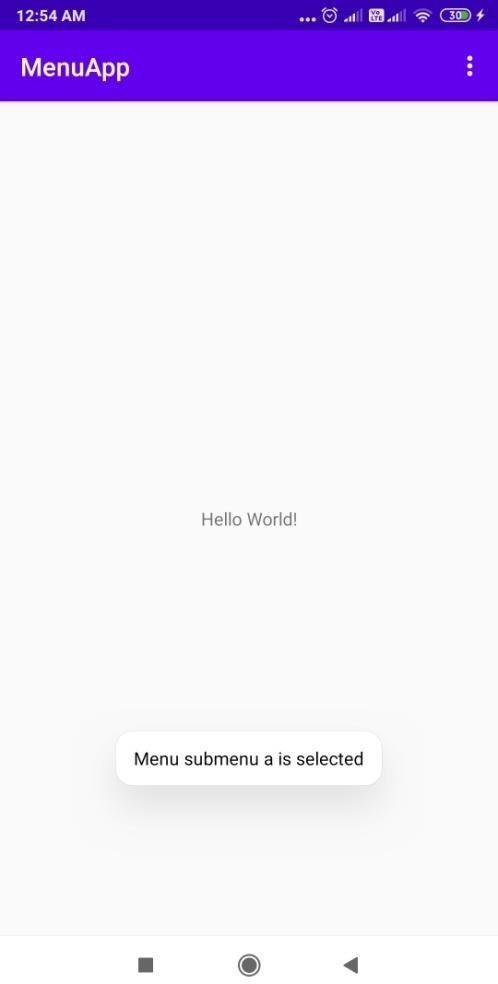
</menu>

</item>

</menu>

### Output:





#### Alert Dialog

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

<LinearLayout xmlns:android="<http://schemas.android.com/apk/res/android>" xmlns:tools="<http://schemas.android.com/tools>" xmlns:app="[http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:id="@+id/mylayout"

tools:context=".MainActivity">

<Button android:layout\_marginTop="100dp"

android:text="Button" android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content" android:id="@+id/button" android:layout\_weight="1"/>

</LinearLayout>

MainActivity.kt

**package** com.example.alertapp

**import** android.graphics.Color

**import** android.support.v7.app.AppCompatActivity

**import** android.os.Bundle

**import** android.support.v7.app.AlertDialog

**import** android.widget.\*

**import** kotlinx.android.synthetic.main.activity\_main.\*

**class** MainActivity : AppCompatActivity() {

**override fun** onCreate(savedInstanceState: Bundle?) { **super**.onCreate(savedInstanceState) setContentView(R.layout.activity\_main)

// Set a click listener for button widget button.setOnClickListener**{**

// Initialize a new instance of

**val**builder = AlertDialog.Builder(**this**@MainActivity)

// Set the alert dialog title builder.setTitle(**"App background color"**)

// Display a message on alert dialog

builder.setMessage(**"Are you want to set the app background color to RED?"**)

// Set a positive button and its click listener on alert dialog builder.setPositiveButton(**"YES"**)**{**dialog, which **->**

// Do something when user press the positive button Toast.makeText(applicationContext,**"Ok, we change the app background."**,Toast.LENGTH\_SHORT).show()

// Change the app background color mylayout.setBackgroundColor(Color.RED)

}

// Display a negative button on alert dialog builder.setNegativeButton(**"No"**)**{**dialog,which**->**

Toast.makeText(applicationContext,**"You are not agree."**,Toast.LENGTH\_SHORT).show() mylayout.setBackgroundColor(Color.WHITE)

}

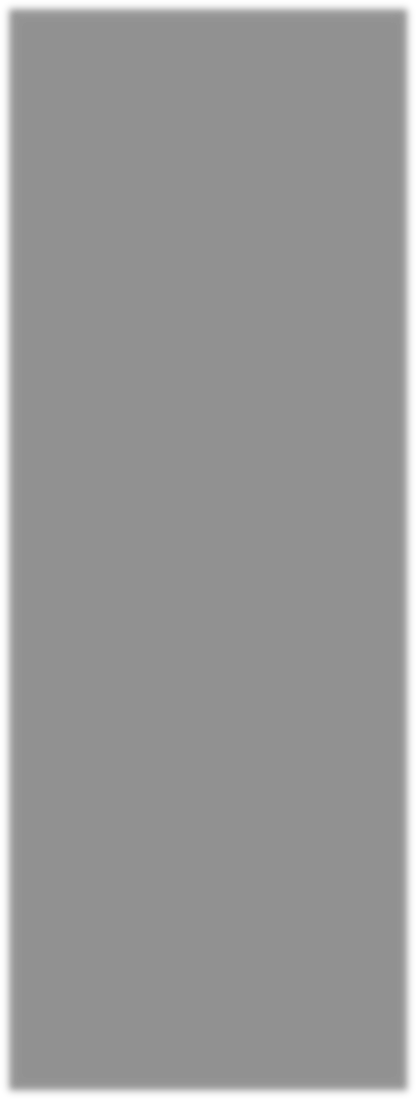
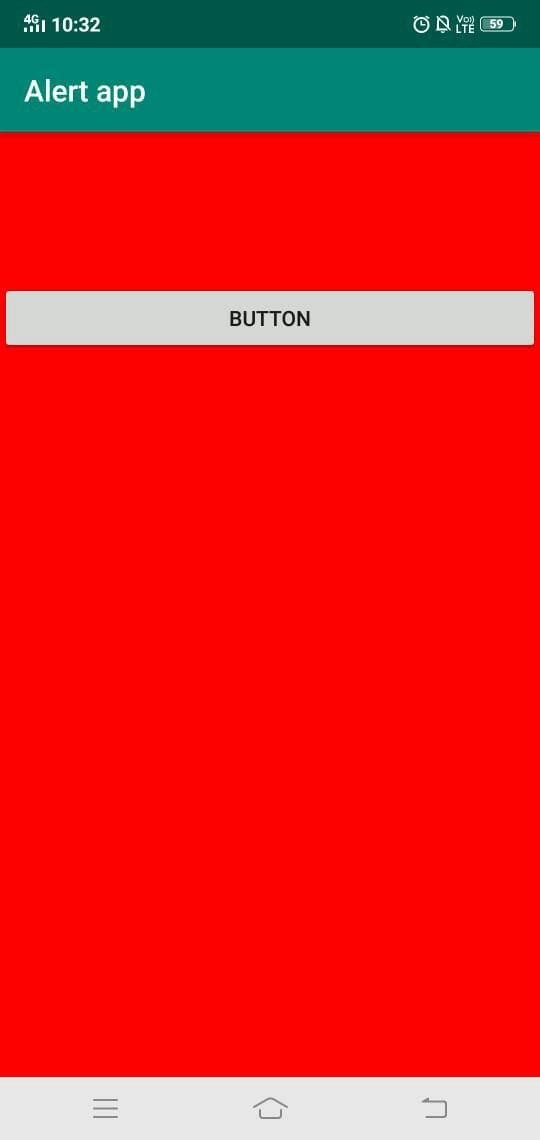
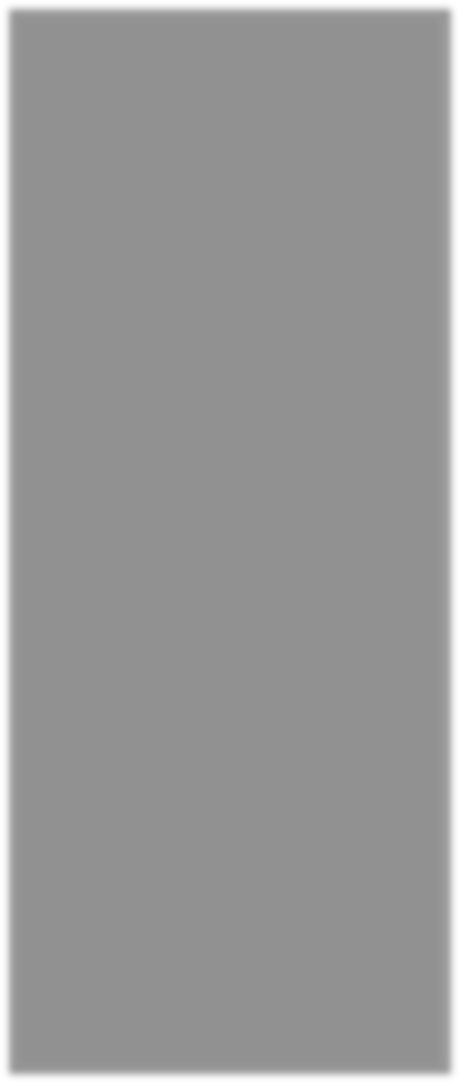
// Display a neutral button on alert dialog builder.setNeutralButton(**"Cancel"**)**{**\_,\_ **->** Toast.makeText(applicationContext,**"You cancelled the dialog."**,Toast.LENGTH\_SHORT).show()

}

// Finally, make the alert dialog using builder

**val**dialog: AlertDialog = builder.create()

// Display the alert dialog on app interface dialog.show()

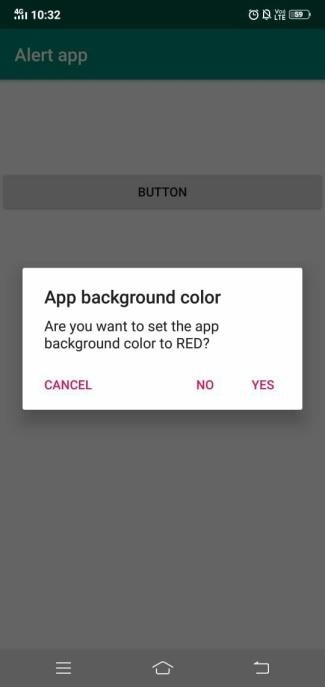
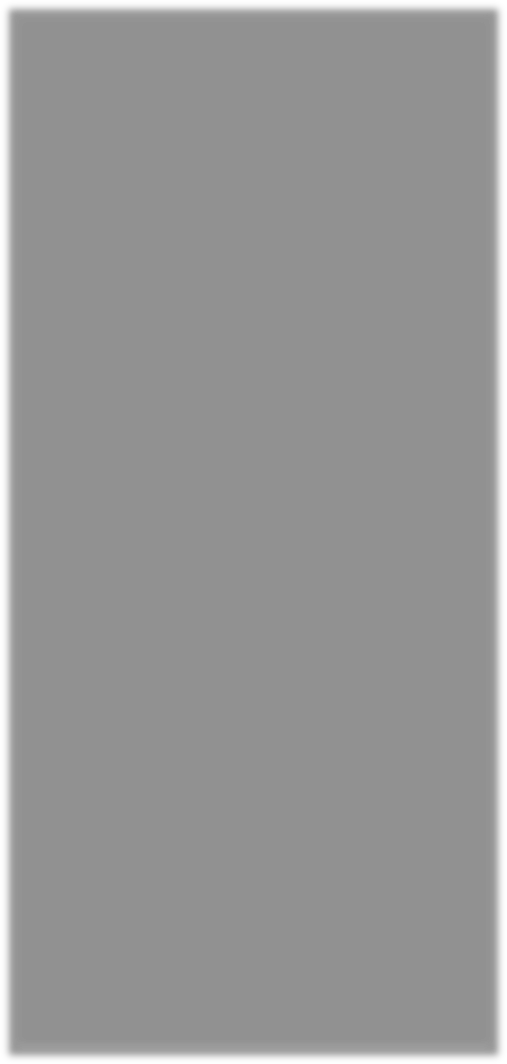


}

}

}

**Output:**



# Practical 7

**Aim:**Programs on Intents, Events, Listeners and Adapters: The Android Intent Class, Using Events and Event Listeners

**Description:**

Events are a useful way to collect data about a user's interaction with interactive components of Applications. Like button presses or screen touch etc. The Android framework maintains an event queue as first-in, first-out (FIFO) basis. You can capture these events in your program and take appropriate action as per requirements.

### Solution:

#### INTENTS

**MainActivity.java**

**package** com.example.intentapp;

**import** androidx.appcompat.app.AppCompatActivity;

**import** android.content.Intent; **import** android.os.Bundle; **import** android.view.View; **import** android.widget.EditText;

**public class** MainActivity**extends** AppCompatActivity { @Override

**protected void** onCreate(Bundle savedInstanceState) { **super**.onCreate(savedInstanceState); setContentView(R.layout.***activity\_main***);

}

**public void** showntext(View view)

{

EditText ed=(EditText)findViewById(R.id.***text1***); String msg=ed.getText().toString();

Intent in=**new** Intent(**this**,newpage.**class**); in.putExtra(**"my key"**, msg);

startActivity(in);

}

}

#### activity\_main.xml

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

<LinearLayoutxmlns:app="[http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) xmlns:tools="<http://schemas.android.com/tools>" xmlns:android="<http://schemas.android.com/apk/res/android>" android:layout\_width="match\_parent" android:layout\_height="match\_parent" tools:context=".MainActivity">

<EditText android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:hint="Enter value" android:id="@+id/text1"/>

<Button android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:id="@+id/btn" android:text="click" android:onClick="showntext"/>

</LinearLayout>

#### Add a new layout resource file newpage\_layout.xml

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

<androidx.constraintlayout.widget.ConstraintLayout 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">

<TextView android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="New Text" android:id="@+id/text2" android:textSize="20dp" tools:ignore="MissingConstraints" />

</androidx.constraintlayout.widget.ConstraintLayout>

#### Add a new empty activity newpage.java

**package** com.example.intentapp;

**import** androidx.appcompat.app.AppCompatActivity;

**import** android.os.Bundle;

**import** android.widget.TextView;

**public class** newpage**extends** AppCompatActivity { @Override

**protected void** onCreate(Bundle savedInstanceState) { **super**.onCreate(savedInstanceState); setContentView(R.layout.***activity\_newpage***); TextView tv1=(TextView)findViewById(R.id.***text2***);

String myval=getIntent().getExtras().getString(**"my key"**); tv1.setText(**"Value="** +myval);

}

}

**Output:**



**Practical 8**

**Aim:**Programs on Services, notification and broadcast receivers

**Description:**

#### Services

A **service** is a component that runs in the background to perform long-running operations without needing to interact with the user and it works even if application is destroyed. A service can essentially take two states -

#### Started

A service is **started** when an application component, such as an activity, starts it by calling *startService()*. Once started, a service can run in the background indefinitely, even if the component that started it is destroyed

#### Bound

A service is **bound** when an application component binds to it by calling *bindService()*. A bound service offers a client-server interface that allows components to interact with the service, send requests, get results, and even do so across processes with interprocess communication (IPC)

### Solution:

#### Activity\_main.xml

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

<LinearLayoutxmlns:app="[http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) xmlns:tools="<http://schemas.android.com/tools>" xmlns:android="<http://schemas.android.com/apk/res/android>" android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:orientation="vertical"

tools:context=".MainActivity">

<Button android:id="@+id/btnStart"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:onClick="startService" android:text="Start Service"

/>

<Button

android:id="@+id/btnStop" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:onClick="stopService" android:text="Stop Service"

/>

#### Create new class in java MyServices.java

**package** com.example.servicesapp;

**import** android.app.Service; **import** android.content.Intent; **import** android.os.IBinder; **import** android.widget.Toast;

**public class** MyServices**extends** Service

{

@Override

**public int** onStartCommand(Intent intent, **int** flags, **int** startId) { Toast.*makeText*(**this**, **"Services Started..."**, Toast.***LENGTH\_LONG***).show(); **return *START\_STICKY***;

}

@Override

**public void** onDestroy() {

Toast.*makeText*(**this**, **"Services Destroyed..."**, Toast.***LENGTH\_LONG***).show();

}

@Override

**public void** onCreate()

{

**super**.onCreate();

}

**public** IBinderonBind(Intent intent)

{

return null;

}

}

#### MainActivity.java

**package** com.example.servicesapp;

**import** androidx.appcompat.app.AppCompatActivity;

**import** android.content.Intent; **import** android.os.Bundle; **import** android.view.View;

**public class** MainActivity**extends** AppCompatActivity { @Override

**protected void** onCreate(Bundle savedInstanceState) { **super**.onCreate(savedInstanceState); setContentView(R.layout.***activity\_main***);

}

**public void** startService(View view)

{

Intent intent= **new** Intent(**this**, MyServices.**class**); startService(intent);

}

**public void** stopService()

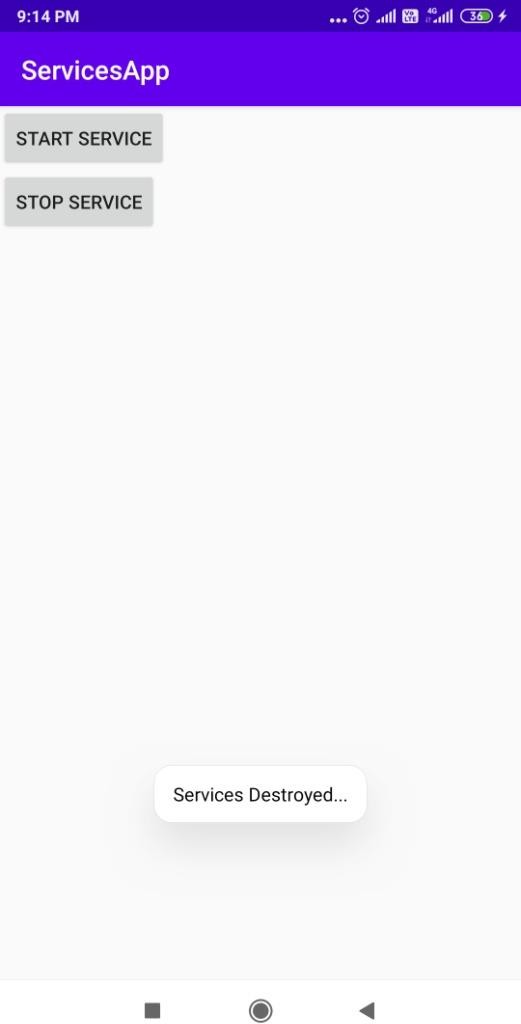
{

Intent intent= **new** Intent(**this**,MyServices.**class**); stopService(intent);

}

}

### Output:

#### Broadcast Receivers activity\_main.xml

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

<android.support.constraint.ConstraintLayoutxmlns:android="[http://schemas.android.](http://schemas.android/) com/apk/res/android"

xmlns:app="[http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) xmlns:tools="<http://schemas.android.com/tools>" android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:background="@color/colorPrimary" tools:context="MainActivity">

<TextView android:id="@+id/textView" android:layout\_width="300dp" android:layout\_height="36dp" android:layout\_marginEnd="8dp" android:layout\_marginStart="8dp" android:gravity="center\_vertical"

android:hint="Flight Mode"

android:textSize="24dp" app:layout\_constraintEnd\_toEndOf="parent"

/>

</android.support.constraint.ConstraintLayout>

#### MyReceiver.kt

**package** com.example.p1

**import** android.content.BroadcastReceiver

**import** android.content.Context **import** android.content.Intent **import** android.widget.Toast

**class** MyReceiver : BroadcastReceiver() {

**override fun** onReceive(context: Context, intent: Intent) {

*//* ***TODO: This method is called when the BroadcastReceiver is receiving***

*// an Intent broadcast.*

Toast.makeText(context, **"Broadcast : Flight mode changed."**, Toast.*LENGTH\_LONG*).show()

}

}

#### AndroidManifest.xml

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

<manifest xmlns:android="<http://schemas.android.com/apk/res/android>" package="com.example.p1">

<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**>

<receiver

android:name=".MyReceiver" android:enabled="true" android:exported="true">

<intent-filter>

<action android:name="android.intent.action.AIRPLANE\_MODE"/>

</intent-filter>

</**receiver**>

</**application**>

</**manifest**>

#### MainActivity.kt

**package** com.example.p1

**import** android.support.v7.app.AppCompatActivity

**import** android.os.Bundle

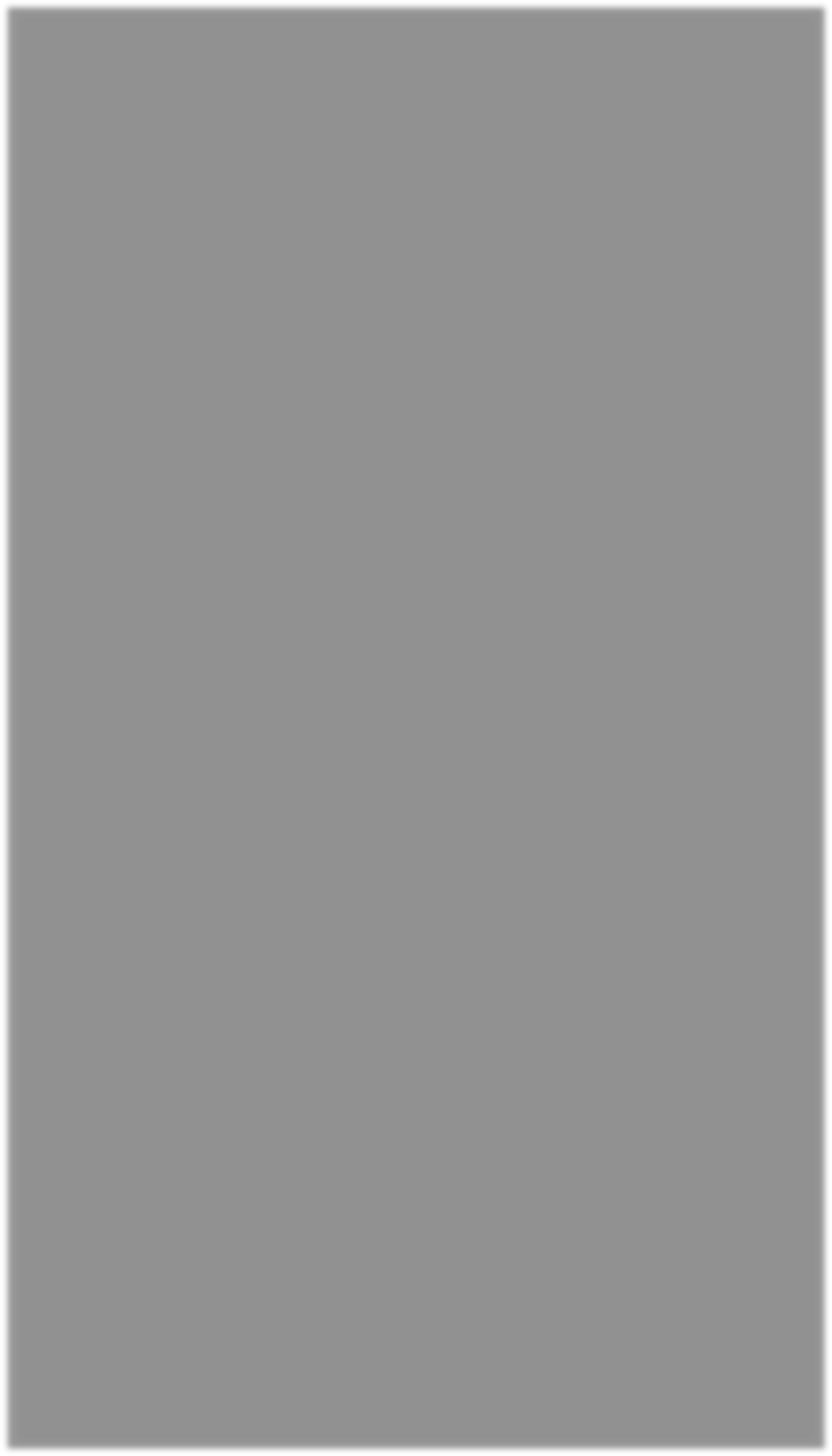
**class** MainActivity : AppCompatActivity() {

**override fun** onCreate(savedInstanceState: Bundle?) { **super**.onCreate(savedInstanceState) setContentView(R.layout.*activity\_main*)

}

}

### Output:



**Practical 9 Aim:**Database Programming with SQLite **Description:**

SQLite is an open-source relational database i.e. used to perform database operations on android devices such as storing, manipulating or retrieving persistent data from the database.

It is embedded in android bydefault. So, there is no need to perform any database setup or administration task.

Here, we are going to see the example of sqlite to store and fetch the data. Data is displayed in the logcat. For displaying data on the spinner or listview, move to the next page.

### Solution:

#### activity\_main.xml

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

<LinearLayoutxmlns:android="<http://schemas.android.com/apk/res/android>" xmlns:app="[http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) xmlns:tools="<http://schemas.android.com/tools>" android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:layout\_marginBottom="8dp"

android:layout\_marginEnd="8dp" android:layout\_marginStart="8dp" android:layout\_marginTop="8dp" android:orientation="vertical" tools:context="example.javatpoint.com.DB.MainActivity">

<TableLayout android:layout\_width="match\_parent" android:layout\_height="wrap\_content">

<TableRow>

<TextView android:text="User Id"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_column="1" />

<EditText

android:id="@+id/u\_id" android:layout\_width="200dp" android:layout\_height="wrap\_content" android:layout\_marginLeft="20sp" android:layout\_marginStart="20sp" android:width="150px" />

</**TableRow**>

<TableRow>

<TextView android:text="User Name"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_column="1" />

<EditText android:id="@+id/u\_name" android:width="200dp"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_column="2" android:layout\_marginStart="20sp" android:layout\_marginLeft="20sp"/>

</**TableRow**>

<TableRow>

<TextView android:text="User Email"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_column="1" />

<EditText android:id="@+id/u\_email" android:width="200dp"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_column="2" android:layout\_marginStart="20sp" android:layout\_marginLeft="20sp" />

</**TableRow**>

</**TableLayout**>

<LinearLayout android:layout\_width="wrap\_content" android:layout\_height="350sp" android:layout\_marginTop="20sp">

<ListView android:id="@+id/listView" android:layout\_width="wrap\_content"

android:layout\_height="350sp"/>

</LinearLayout>

<LinearLayout android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_marginTop="40sp" android:orientation="horizontal" android:layout\_gravity="center">

<Button android:text="Save"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:onClick="saveRecord"/>

<Button android:text="View"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:onClick="viewRecord"/>

<Button android:text="Update"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:onClick="updateRecord"/>

<Button android:text="Delete"

android:layout\_width="wrap\_content" android:layout\_height="match\_parent"

android:onClick="deleteRecord"/>

</LinearLayout>

</LinearLayout>

#### custom\_list.xml

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

<android.support.constraint.ConstraintLayoutxmlns:android="[http://schemas.android.](http://schemas.android/) com/apk/res/android"

android:layout\_width="match\_parent" android:layout\_height="match\_parent">

<LinearLayoutxmlns:android="<http://schemas.android.com/apk/res/android>" android:orientation="vertical" android:layout\_width="match\_parent" android:layout\_height="match\_parent"

android:id="@+id/linearLayout">

<TextView android:id="@+id/textViewId" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:text="Id"

android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium"/>

<TextView android:id="@+id/textViewName" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:text="Name"

android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium"/>

<TextView android:id="@+id/textViewEmail" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:text="Email"

android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium"/>

</LinearLayout>

</android.support.constraint.ConstraintLayout>

delete\_dialog.xml

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

<android.support.constraint.ConstraintLayoutxmlns:android="[http://schemas.android.](http://schemas.android/) com/apk/res/android"

android:layout\_width="match\_parent" android:layout\_height="match\_parent">

<LinearLayoutxmlns:android="<http://schemas.android.com/apk/res/android>" android:orientation="vertical"

android:padding="10dp" android:layout\_width="match\_parent" android:layout\_height="match\_parent">

<EditText android:id="@+id/deleteId"

android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:ems="10"

android:hint="enter id" />

</LinearLayout>

</android.support.constraint.ConstraintLayout>

#### update\_dialog.xml

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

<android.support.constraint.ConstraintLayoutxmlns:android="[http://schemas.android.](http://schemas.android/) com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<LinearLayoutxmlns:android="<http://schemas.android.com/apk/res/android>" android:orientation="vertical"

android:padding="10dp" android:layout\_width="match\_parent" android:layout\_height="match\_parent">

<EditText android:id="@+id/updateId" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:ems="10"

android:hint="enter id" />

<EditText android:id="@+id/updateName" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:ems="10"

android:hint="enter name"/>

<EditText android:id="@+id/updateEmail" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:ems="10"

android:hint="enter email"/>

</LinearLayout>

</android.support.constraint.ConstraintLayout>

#### MainActivity.kt

**package** com.example.db

**import** android.support.v7.app.AppCompatActivity

**import** android.os.Bundle **import** android.view.View **import** android.widget.EditText **import** android.widget.Toast

**import** kotlinx.android.synthetic.main.activity\_main.\*

**import** android.content.DialogInterface

**import** android.support.v7.app.AlertDialog

**class** MainActivity : AppCompatActivity() {

**override fun** onCreate(savedInstanceState: Bundle?) { **super**.onCreate(savedInstanceState) setContentView(R.layout.*activity\_main*)

}

*//method for saving records in database*

**fun** saveRecord(view: View){ **val**id = u\_id.*text*.toString() **val**name = u\_name.*text*.toString() **val**email = u\_email.*text*.toString()

**val**databaseHandler: DatabaseHandler= DatabaseHandler(**this**)

**if**(id.*trim*()!=**""** &&name.*trim*()!=**""** &&email.*trim*()!=**""**){

**val**status = databaseHandler.addEmployee(EmpModelClass(Integer.parseInt(id),name, email))

**if**(status > -1){

Toast.makeText(*applicationContext*,**"record save"**,Toast.*LENGTH\_LONG*).show() u\_id.*text*.clear()

u\_name.*text*.clear() u\_email.*text*.clear()

}

}**else**{

Toast.makeText(*applicationContext*,**"id or name or email cannot be blank"**,Toast.*LENGTH\_LONG*).show()

}

}

*//method for read records from database in ListView*

**fun** viewRecord(view: View){

*//creating the instance of DatabaseHandler class*

**val**databaseHandler: DatabaseHandler= DatabaseHandler(**this**)

*//calling the viewEmployee method of DatabaseHandler class to read the records* **val**emp: List<EmpModelClass> = databaseHandler.viewEmployee() **val**empArrayId = Array<String>(emp.**size**)**{"0"}**

**val**empArrayName = Array<String>(emp.**size**)**{"null"} val**empArrayEmail = Array<String>(emp.**size**)**{"null"}**

**var** index = 0

**for**(e **in** emp){

empArrayId[index] = e.**userId**.toString() empArrayName[index] = e.**userName** empArrayEmail[index] = e.**userEmail** index++

}

*//creating custom ArrayAdapter*

**val**myListAdapter = MyListAdapter(**this**,empArrayId,empArrayName,empArrayEmail) listView.*adapter*= myListAdapter

}

*//method for updating records based on user id* **fun** updateRecord(view: View){ **val**dialogBuilder = AlertDialog.Builder(**this**) **val**inflater = **this**.*layoutInflater*

**val**dialogView = inflater.inflate(R.layout.*update\_dialog*, **null**) dialogBuilder.setView(dialogView)

**val**edtId = dialogView.findViewById(R.id.*updateId*) **as** EditText

**val**edtName = dialogView.findViewById(R.id.*updateName*) **as** EditText

**val**edtEmail = dialogView.findViewById(R.id.*updateEmail*) **as** EditText

dialogBuilder.setTitle(**"Update Record"**) dialogBuilder.setMessage(**"Enter data below"**)

dialogBuilder.setPositiveButton(**"Update"**, DialogInterface.OnClickListener**{** \_, \_ **->**

**val**updateId = edtId.*text*.toString() **val**updateName = edtName.*text*.toString() **val**updateEmail = edtEmail.*text*.toString()

*//creating the instance of DatabaseHandler class*

**val**databaseHandler: DatabaseHandler= DatabaseHandler(**this**)

**if**(updateId.*trim*()!=**""** &&updateName.*trim*()!=**""** &&updateEmail.*trim*()!=**""**){

*//calling the updateEmployee method of DatabaseHandler class to update record*

**val**status = databaseHandler.updateEmployee(EmpModelClass(Integer.parseInt(updateId),updateName, updateEmail))

**if**(status > -1){

Toast.makeText(*applicationContext*,**"record update"**,Toast.*LENGTH\_LONG*).show()

}

}**else**{

Toast.makeText(*applicationContext*,**"id or name or email cannot be blank"**,Toast.*LENGTH\_LONG*).show()

}

**}**)

dialogBuilder.setNegativeButton(**"Cancel"**, DialogInterface.OnClickListener**{** dialog, which

->

*//pass*

**}**)

**val**b = dialogBuilder.create() b.show()

}

*//method for deleting records based on id*

**fun** deleteRecord(view: View){

*//creating AlertDialog for taking user id* **val**dialogBuilder = AlertDialog.Builder(**this**) **val**inflater = **this**.*layoutInflater*

**val**dialogView = inflater.inflate(R.layout.*delete\_dialog*, **null**) dialogBuilder.setView(dialogView)

**val**dltId = dialogView.findViewById(R.id.*deleteId*) **as** EditText dialogBuilder.setTitle(**"Delete Record"**) dialogBuilder.setMessage(**"Enter id below"**)

dialogBuilder.setPositiveButton(**"Delete"**, DialogInterface.OnClickListener**{** \_, \_ **->**

**val**deleteId = dltId.*text*.toString()

*//creating the instance of DatabaseHandler class* **val**databaseHandler: DatabaseHandler= DatabaseHandler(**this**) **if**(deleteId.*trim*()!=**""**){

*//calling the deleteEmployee method of DatabaseHandler class to delete record* **val**status = databaseHandler.deleteEmployee(EmpModelClass(Integer.parseInt(deleteId),**""**,**""**)) **if**(status > -1){

Toast.makeText(*applicationContext*,**"record deleted"**,Toast.*LENGTH\_LONG*).show()

}

}**else**{

Toast.makeText(*applicationContext*,**"id or name or email cannot be blank"**,Toast.*LENGTH\_LONG*).show()

}

**}**)

dialogBuilder.setNegativeButton(**"Cancel"**, DialogInterface.OnClickListener**{** \_, \_ **->**

*//pass*

**}**)

**val**b = dialogBuilder.create() b.show()

}

}

#### EmpClassModel.kt

**package** com.example.db

**class** EmpModelClass (**var userId**: Int, **valuserName**:String , **valuserEmail**: String)

#### MyListAdapter.kt

**package** com.example.db

**import** android.app.Activity **import** android.view.View **import** android.view.ViewGroup

**import** android.widget.ArrayAdapter

**import** android.widget.TextView

**class** MyListAdapter(**private val context**: Activity, **private val id**: Array<String>, **private val name**: Array<String>, **private val email**: Array<String>)

: ArrayAdapter<String>(context, R.layout.*custom\_list*, name) {

**override fun** getView(position: Int, view: View?, parent: ViewGroup): View {

**val**inflater = **context**.*layoutInflater*

**val**rowView = inflater.inflate(R.layout.*custom\_list*, **null**, **true**)

**val**idText = rowView.findViewById(R.id.*textViewId*) **as** TextView **val**nameText = rowView.findViewById(R.id.*textViewName*) **as** TextView **val**emailText = rowView.findViewById(R.id.*textViewEmail*) **as** TextView

idText.*text*= **"Id: ${id**[position]**}"**

nameText.*text*= **"Name: ${name**[position]**}"**

emailText.*text*= **"Email: ${email**[position]**}" return** rowView

}

}

#### DatabaseHandler.kt

**package** com.example.db

**import** android.content.Context

**import** android.database.sqlite.SQLiteDatabase **import** android.database.sqlite.SQLiteOpenHelper **import** android.content.ContentValues

**import** android.database.Cursor

**import** android.database.sqlite.SQLiteException

*//creating the database logic, extending the SQLiteOpenHelper base class* **class**DatabaseHandler(context: Context): SQLiteOpenHelper(context,**DATABASE\_NAME**,**null**,**DATABASE\_VERSION**) { **companion object** {

private val DATABASE\_VERSION = 1

private val DATABASE\_NAME = "EmployeeDatabase" private val TABLE\_CONTACTS = "EmployeeTable" private val KEY\_ID = "id"

private val KEY\_NAME = "name" private val KEY\_EMAIL = "email"

}

**override fun** onCreate(db: SQLiteDatabase?) {

*//* ***TODO("not implemented") //To change body of created functions use File | Settings | File Templates.***

*//creating table with fields*

**val**CREATE\_CONTACTS\_TABLE = (**"CREATE TABLE "** + **TABLE\_CONTACTS** +

"("

+ KEY\_ID + " INTEGER PRIMARY KEY," + KEY\_NAME + " TEXT,"

+ **KEY\_EMAIL** + **" TEXT"** + **")"**) db?.execSQL(CREATE\_CONTACTS\_TABLE)

}

**override fun** onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {

*//* ***TODO("not implemented") //To change body of created functions use File | Settings | File Templates.***

db!!.execSQL(**"DROP TABLE IF EXISTS "** + **TABLE\_CONTACTS**)

onCreate(db)

}

*//method to insert data*

**fun** addEmployee(emp: EmpModelClass):Long{

**val**db = **this**.*writableDatabase*

**val**contentValues = ContentValues() contentValues.put(**KEY\_ID**, emp.**userId**)

contentValues.put(**KEY\_NAME**, emp.**userName**) *// EmpModelClass Name*

contentValues.put(**KEY\_EMAIL**,emp.**userEmail**) *// EmpModelClass Phone*

*// Inserting Row*

**val**success = db.insert(**TABLE\_CONTACTS**, **null**, contentValues)

*//2nd argument is String containing nullColumnHack*

db.close() *// Closing database connection*

**return** success

}

*//method to read data*

**fun** viewEmployee():List<EmpModelClass>{ **val**empList:ArrayList<EmpModelClass> = ArrayList<EmpModelClass>() **val**selectQuery = **"SELECT \* FROM $TABLE\_CONTACTS"**

**val**db = **this**.*readableDatabase* **var** cursor: Cursor? = **null**

**try**{

cursor = db.rawQuery(selectQuery, **null**)

}**catch** (e: SQLiteException) { db.execSQL(selectQuery)

**return** ArrayList()

}

**var** userId: Int

**var** userName: String

**var** userEmail: String

**if** (cursor.moveToFirst()) {

**do** {

userId = cursor.getInt(cursor.getColumnIndex(**"id"**))

userName = cursor.getString(cursor.getColumnIndex(**"name"**)) userEmail = cursor.getString(cursor.getColumnIndex(**"email"**))

**val**emp= EmpModelClass(userId = userId, userName = userName, userEmail = userEmail) empList.add(emp)

} **while** (cursor.moveToNext())

}

**return** empList

}

*//method to update data*

**fun** updateEmployee(emp: EmpModelClass):Int{

**val**db = **this**.*writableDatabase* **val**contentValues = ContentValues() contentValues.put(**KEY\_ID**, emp.**userId**)

contentValues.put(**KEY\_NAME**, emp.**userName**) *// EmpModelClass Name*

contentValues.put(**KEY\_EMAIL**,emp.**userEmail**) *// EmpModelClass Email*

*// Updating Row*

**val**success = db.update(**TABLE\_CONTACTS**, contentValues,**"id="**+emp.**userId**,**null**)

*//2nd argument is String containing nullColumnHack*

db.close() *// Closing database connection*

**return** success

}

*//method to delete data*

**fun** deleteEmployee(emp: EmpModelClass):Int{

**val**db = **this**.*writableDatabase* **val**contentValues = ContentValues()

contentValues.put(**KEY\_ID**, emp.**userId**) *// EmpModelClassUserId*

*// Deleting Row*

**val**success = db.delete(**TABLE\_CONTACTS**,**"id="**+emp.**userId**,**null**)

*//2nd argument is String containing nullColumnHack*

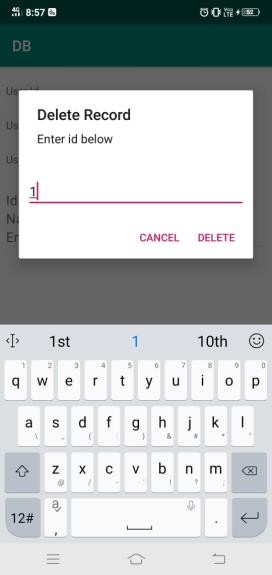
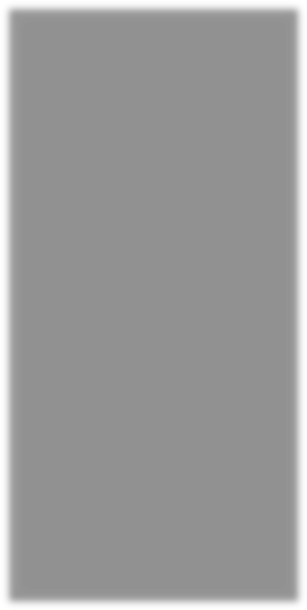
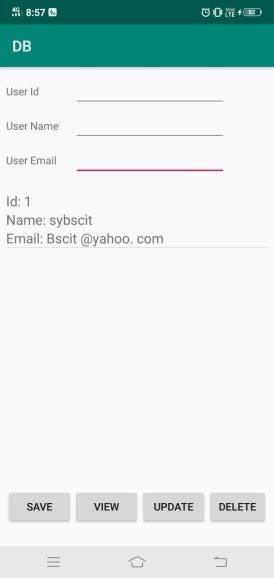
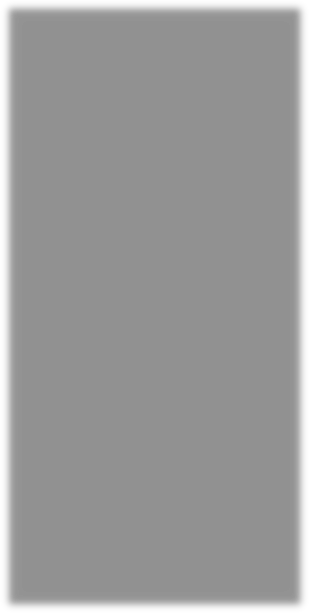
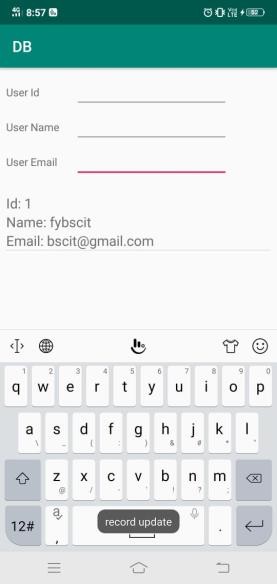
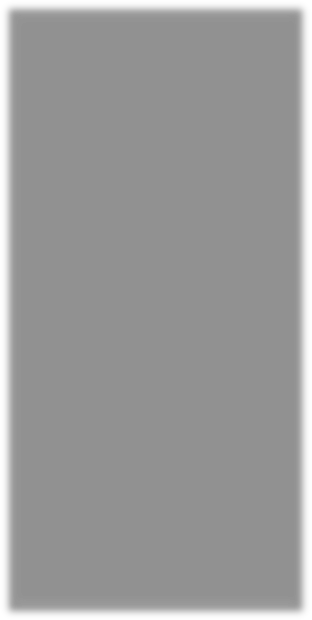
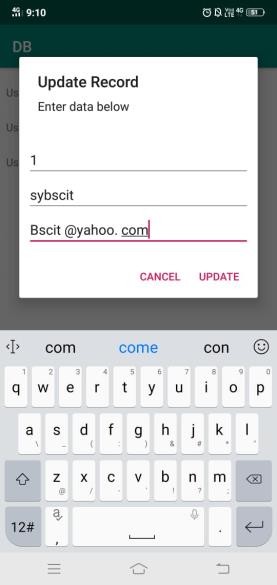
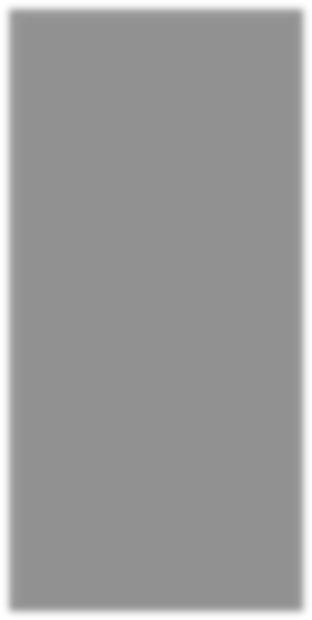
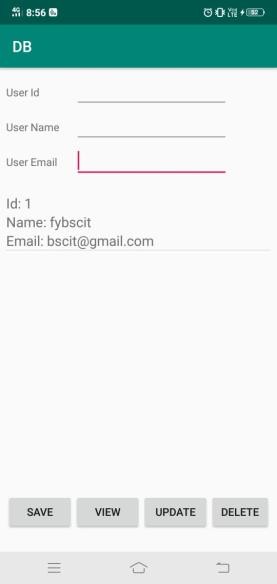
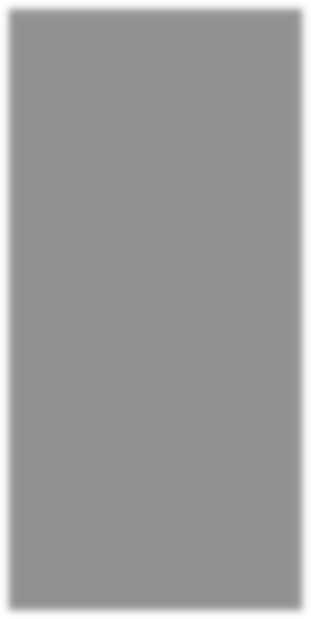
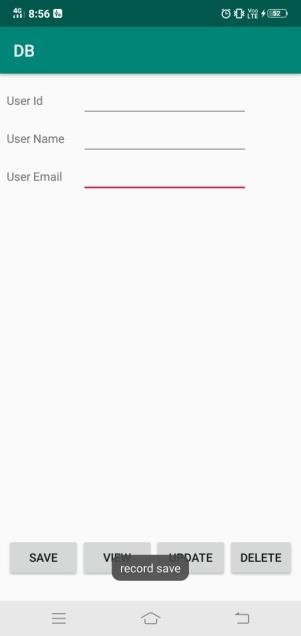
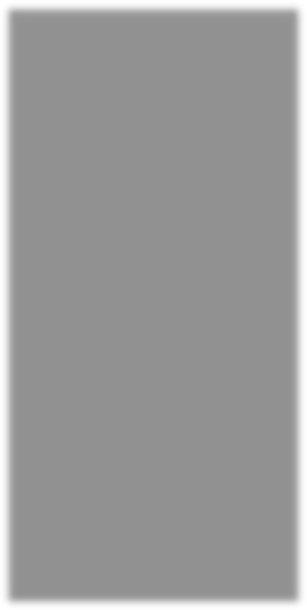
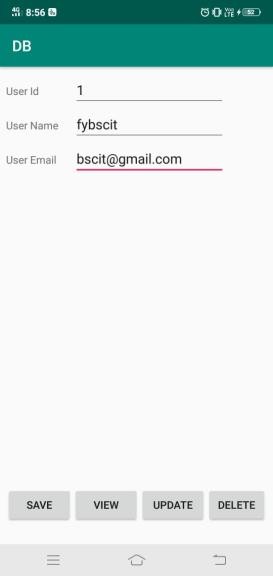
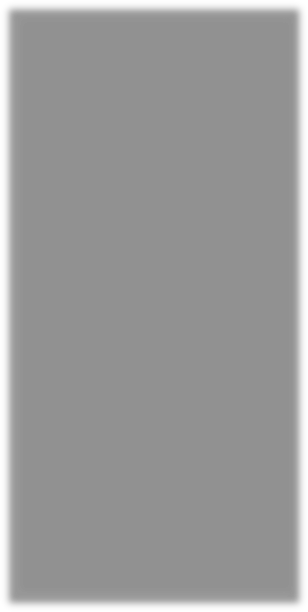
db.close() *// Closing database connection*

**return** success

}

}

### Output:



## PRACTICAL 10

**Aim**: Programming Security and permissions.

**Description:**

The purpose of a permission is to protect the privacy of an Android user. Android apps must request permission to access sensitive user data (such as contacts and SMS), as well as certain system features (such as camera and internet). Depending on the feature, the system might grant the permission automatically or might prompt the user to approve the request.

### Solution:

Extra Packagesrequied in ManagePermission.kt (Class File)

**import** android.app.Activity

**import** android.content.pm.PackageManager **import** android.support.v4.app.ActivityCompat **import** android.support.v4.content.ContextCompat **import** android.support.v7.app.AlertDialog

Extra Packagesrequied in MainActivity.kt

**import** android.Manifest **import** android.content.Context **import** android.os.Build **import** android.widget.Toast

**import** kotlinx.android.synthetic.main.activity\_main.\*

For Multple Permission Access,need to add following line in class

**MainActivity.kt**

private valPermissionsRequestCode= 123

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

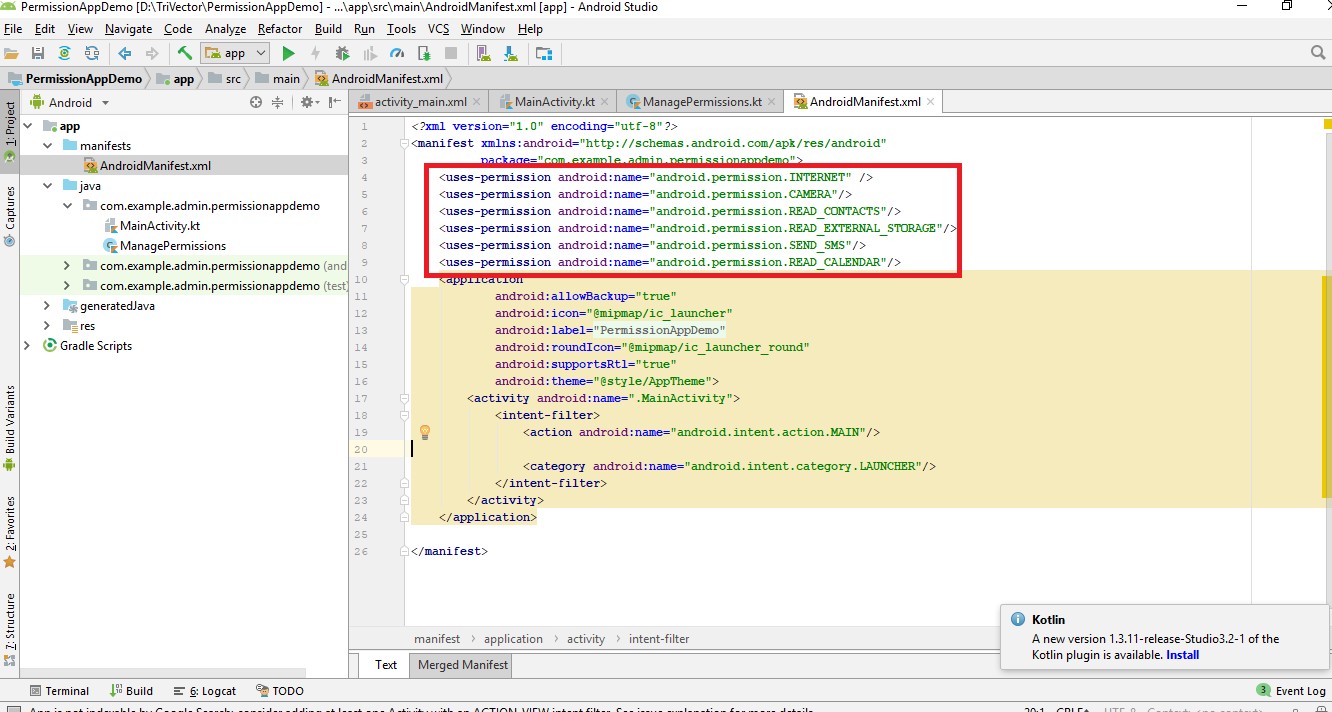
<uses-permission android:name="android.permission.CAMERA"/>

<uses-permission android:name="android.permission.READ\_CONTACTS"/>

<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE"/>

<uses-permission android:name="android.permission.SEND\_SMS"/>

<uses-permission android:name="android.permission.READ\_CALENDAR"/>



#### Activity\_main.xml

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

<androidx.constraintlayout.widget.ConstraintLayoutxmlns:android="[http://schemas.a](http://schemas.a/) ndroid.com/apk/res/android"

xmlns:app="[http://schemas.android.com/apk/res-auto"](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=".MainActivity">

<Button android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:id="@+id/button" android:text="Request Permissions" tools:ignore="MissingConstraints" />

</androidx.constraintlayout.widget.ConstraintLayout>

#### MainActivity.kt

**package** com.example.admin.permissionappdemo

**import** android.Manifest **import** android.content.Context **import** android.os.Build

**import** android.support.v7.app.AppCompatActivity

**import** android.os.Bundle

**import** android.widget.Toast

**import** kotlinx.android.synthetic.main.activity\_main.\*

**class** MainActivity : AppCompatActivity() {

private valPermissionsRequestCode= 123

**private lateinit var managePermissions**: ManagePermissions

**override fun** onCreate(savedInstanceState: Bundle?) { **super**.onCreate(savedInstanceState) setContentView(R.layout.*activity\_main*)

*// Initialize a list of required permissions to request runtime* **val**list = *listOf*<String>( android.Manifest.permission.*CAMERA*, android.Manifest.permission.*READ\_CONTACTS*, android.Manifest.permission.*READ\_EXTERNAL\_STORAGE*, android.Manifest.permission.*SEND\_SMS*, android.Manifest.permission.*READ\_CALENDAR*

)

*// Initialize a new instance of ManagePermissions class*

**managePermissions**= ManagePermissions(**this**,list,**PermissionsRequestCode**)

*// Button to check permissions states*

button.setOnClickListener**{**

**if** (Build.VERSION.*SDK\_INT*>= Build.VERSION\_CODES.*M*)

**managePermissions**.checkPermissions()

}

}

*// Receive the permissions request result*

**override fun** onRequestPermissionsResult(requestCode: Int, permissions: Array<String>, grantResults: IntArray) {

**when** (requestCode) { **PermissionsRequestCode**->{ **val**isPermissionsGranted = **managePermissions**

.processPermissionsResult(requestCode,permissions,grantResults)

**if**(isPermissionsGranted){

*// Do the task now*

*toast*(**"Permissions granted."**)

}**else**{

*toast*(**"Permissions denied."**)

}

return

}

}

}

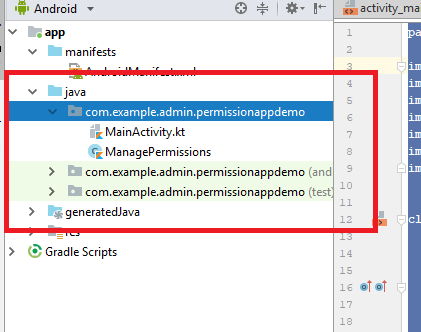
}

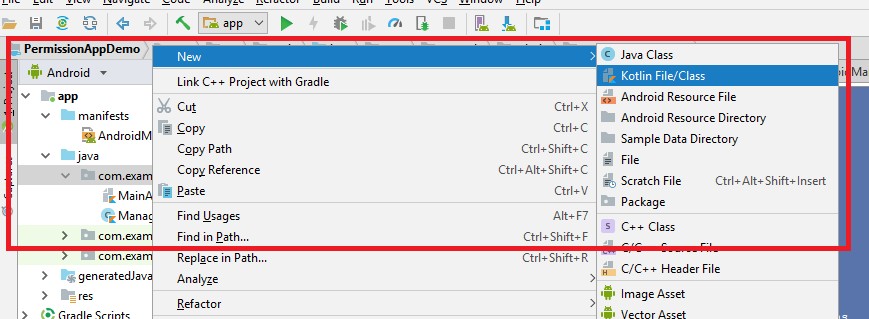
*// Extension function to show toast message*

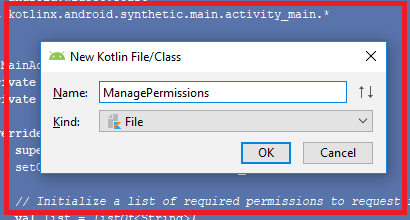
**fun** Context.toast(message: String) {

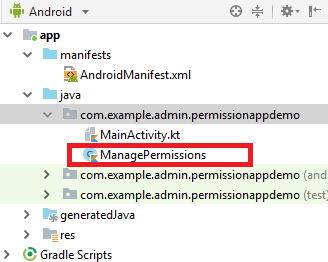
Toast.makeText(**this**, message, Toast.*LENGTH\_SHORT*).show()

}









#### ManagePermissions.kt

**import** android.app.Activity

**import** android.content.pm.PackageManager **import** android.support.v4.app.ActivityCompat **import** android.support.v4.content.ContextCompat **import** android.support.v7.app.AlertDialog

**class** ManagePermissions(**val activity**: Activity,**val list**: List<String>,**valcode**:Int) {

*// Check permissions at runtime*

**fun** checkPermissions() {

**if** (isPermissionsGranted() != PackageManager.*PERMISSION\_GRANTED*) { showAlert()

} **else** {

activity.*toast*("Permissions already granted.")

}

}

*// Check permissions status*

**private fun** isPermissionsGranted(): Int {

*// PERMISSION\_GRANTED : Constant Value: 0*

*// PERMISSION\_DENIED : Constant Value: -1*

**var** counter = 0;

**for** (permission **in list**) {

counter += ContextCompat.checkSelfPermission(**activity**, permission)

}

**return** counter

}

*// Find the first denied permission* **private fun** deniedPermission(): String { **for** (permission **in list**) {

**if** (ContextCompat.checkSelfPermission(**activity**, permission)

== PackageManager.*PERMISSION\_DENIED*) **return** permission

}

return ""

}

*// Show alert dialog to request permissions*

**private fun** showAlert() {

**val**builder = AlertDialog.Builder(**activity**) builder.setTitle(**"Need permission(s)"**)

builder.setMessage(**"Some permissions are required to do the task."**) builder.setPositiveButton(**"OK"**, **{** dialog, which **->**requestPermissions() **}**) builder.setNeutralButton(**"Cancel"**, **null**)

**val**dialog = builder.create() dialog.show()

}

*// Request the permissions at run time* **private fun** requestPermissions() { **val**permission = deniedPermission()

**if** (ActivityCompat.shouldShowRequestPermissionRationale(**activity**, permission)) {

*// Show an explanation asynchronously*

activity.*toast*("Should show an explanation.")

} **else** {

ActivityCompat.requestPermissions(**activity**, **list**.*toTypedArray*(), **code**)

}

}

*// Process permissions result*

**fun** processPermissionsResult(requestCode: Int, permissions: Array<String>, grantResults: IntArray): Boolean {

**var** result = 0

**if** (grantResults.*isNotEmpty*()) {

**for** (item **in** grantResults) {

result += item

}

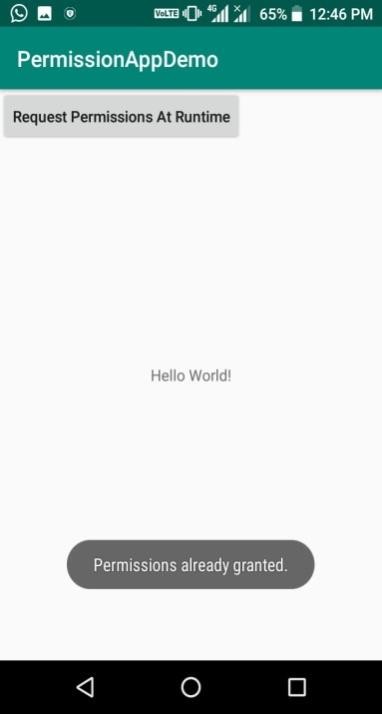
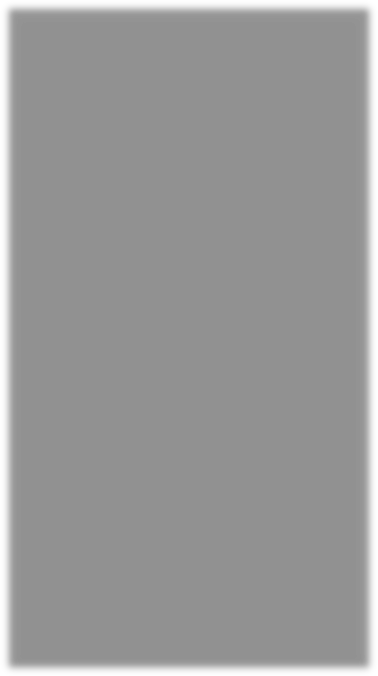
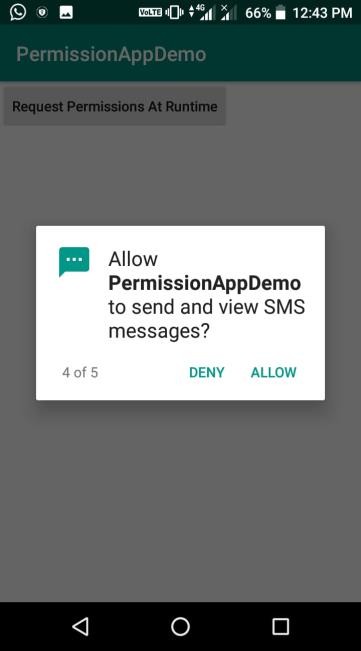
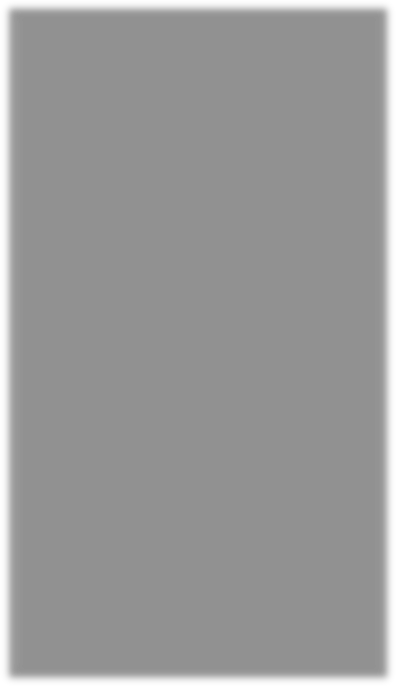
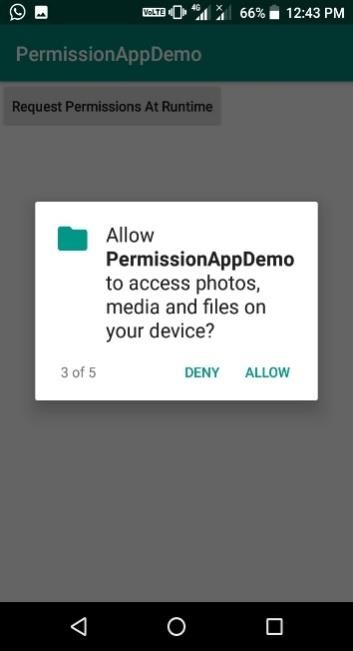
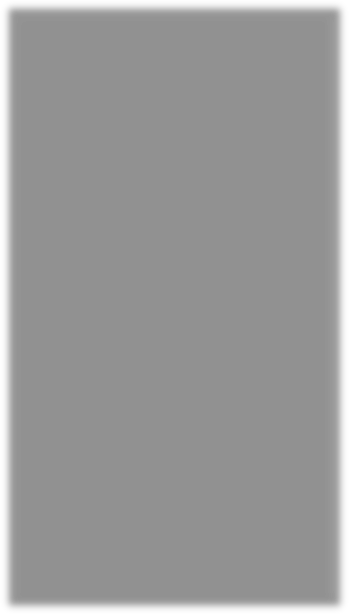
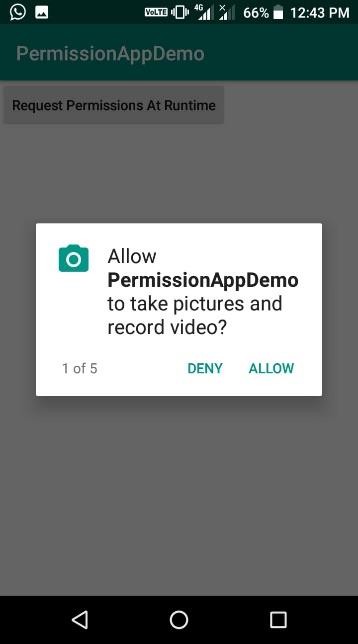
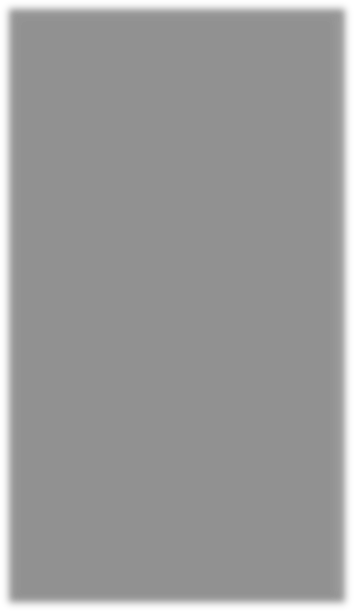
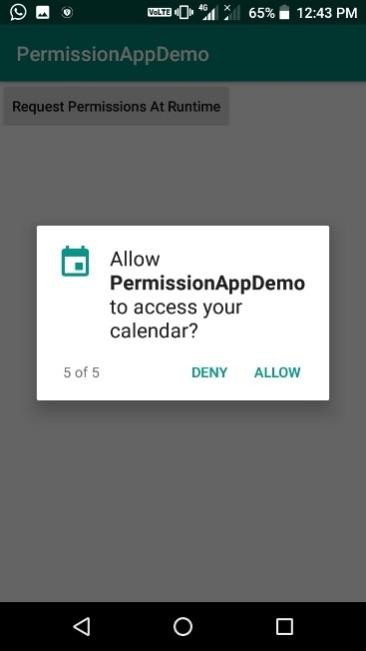
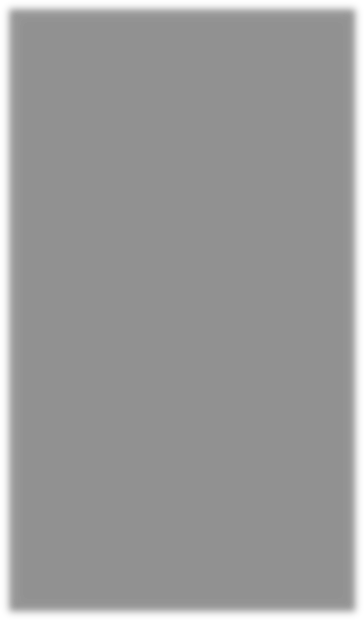
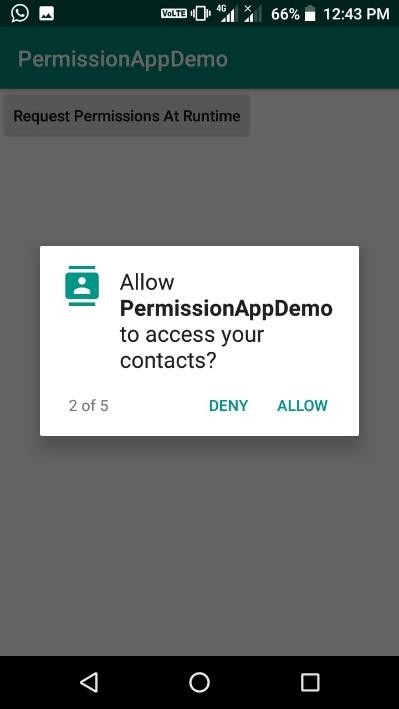
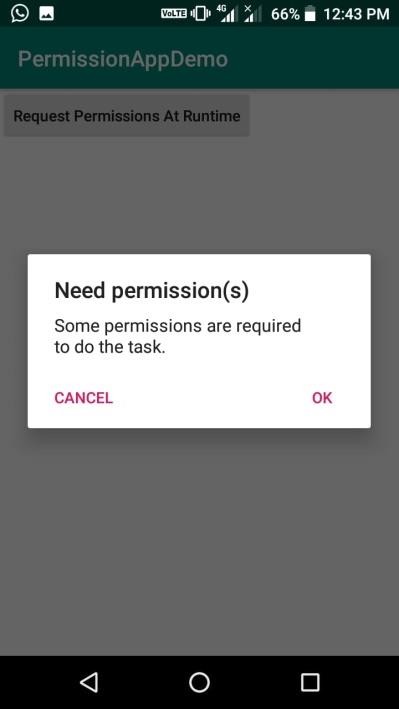
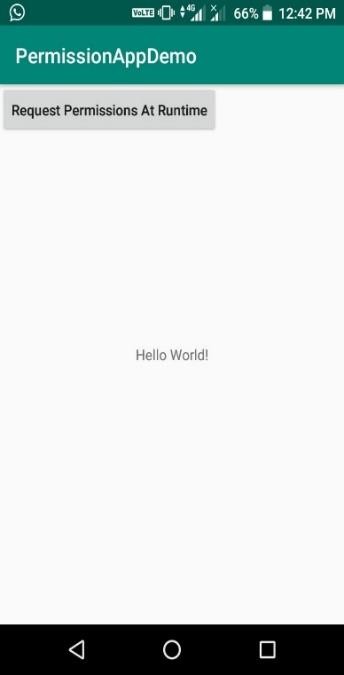
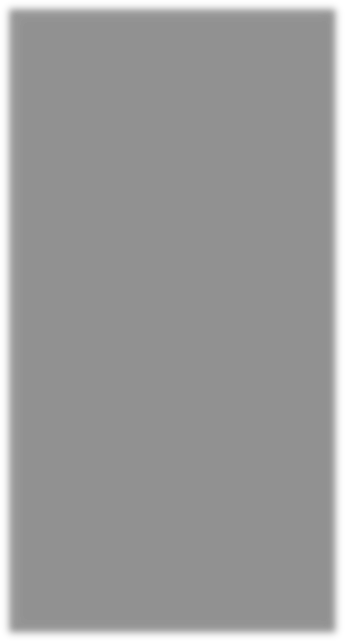
}

**if** (result == PackageManager.*PERMISSION\_GRANTED*) **return true return false**

}

}

### Output:-



## PRACTICAL 11A

**Aim**: Programming Media API

**Description:**

Provides classes that manage various media interfaces in audio and video.

The Media APIs are used to play and, in some cases, record media files. This includes audio (e.g., play MP3s or other music files, ringtones, game sound effects, or DTMF tones) and video (e.g., play a video streamed over the web or from local storage).

Other special classes in the package offer the ability to detect the faces of people in Bitmaps (FaceDetector), control audio routing (to the device or a headset) and control alerts such as ringtones and phone vibrations (AudioManager).

**Solution:**

#### Activity\_main.xml

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

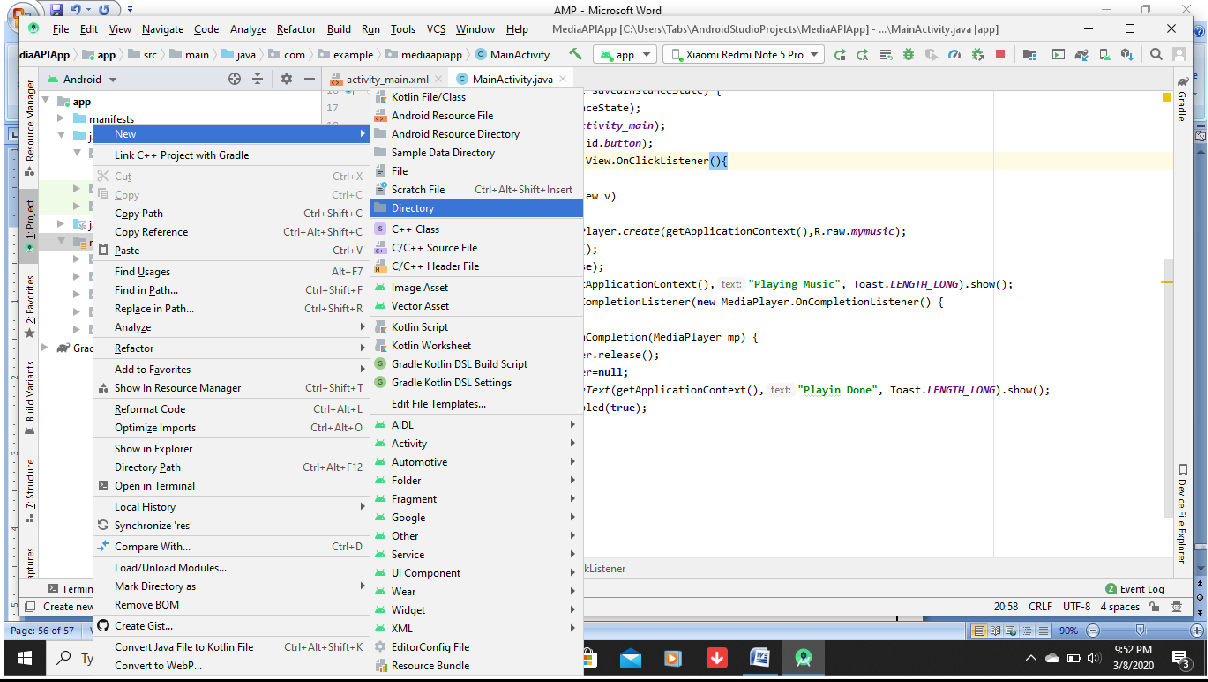
<LinearLayoutxmlns:app="[http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) xmlns:tools="<http://schemas.android.com/tools>" xmlns:android="<http://schemas.android.com/apk/res/android>" android:layout\_width="match\_parent" android:layout\_height="match\_parent" tools:context=".MainActivity">

<Button android:id="@+id/button"

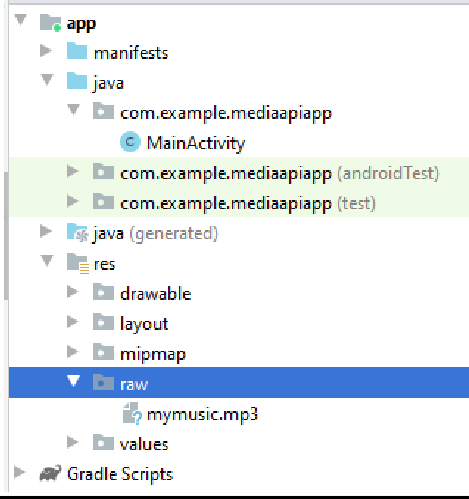
android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_gravity="center" android:text="START" />

</LinearLayout>

#### Create new directory under 'res' folder



**Copy mymusic.mp3 file and paste it in the raw folder created**



#### MainActivity.java

**package** com.example.mediaapiapp;

**import** androidx.appcompat.app.AppCompatActivity;

**import** android.media.MediaPlayer;

**import** android.os.Bundle; **import** android.view.View; **import** android.widget.Button; **import** android.widget.Toast;

**public class** MainActivity**extends** AppCompatActivity { MediaPlayer**mediaPlayer**;

Button **bn**;

@Override

**protected void** onCreate(Bundle savedInstanceState) { **super**.onCreate(savedInstanceState); setContentView(R.layout.***activity\_main***); **bn**=(Button)findViewById(R.id.***button***); **bn**.setOnClickListener(**new** View.OnClickListener(){

@Override

**public void** onClick(View v)

{

**mediaPlayer**=MediaPlayer.*create*(getApplicationContext(),R.raw.***mymusic***); **mediaPlayer**.start();

**bn**.setEnabled(**false**); Toast.*makeText*(getApplicationContext(),**"Playing Music"**, Toast.***LENGTH\_LONG***).show();

**mediaPlayer**.setOnCompletionListener(**new** MediaPlayer.OnCompletionListener() { @Override

**public void** onCompletion(MediaPlayermp) {

**mediaPlayer**.release(); **mediaPlayer**=**null**;

Toast.*makeText*(getApplicationContext(),**"Playin Done"**, Toast.***LENGTH\_LONG***).show();

**bn**.setEnabled(**true**);

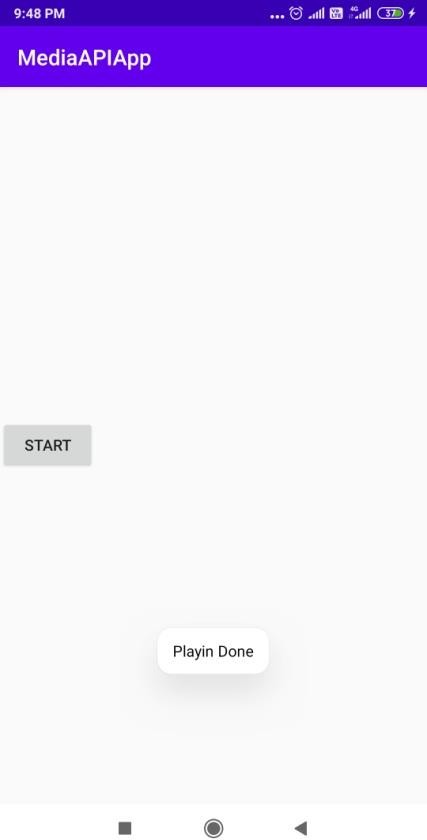
}

});

}

});

}

**Output:**

**PRACTICAL 11B**

**Aim:**Programming Telephone API

**Description:**

#### Android.telephony

Provides APIs for monitoring the basic phone information, such as the network type and connection state, plus utilities for manipulating phone number strings.

### Solution:

#### AndroidManifest.xml

*Add user-permission in this file.*

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

<manifest xmlns:android="<http://schemas.android.com/apk/res/android>" package="com.example.telephoneapiapp">

<uses-permission android:name="android.permission.CALL\_PHONE"/>

<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**>

</**application**>

</**manifest**>

#### Activity\_main.xml

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

<androidx.constraintlayout.widget.ConstraintLayoutxmlns:android="[http://schemas.a](http://schemas.a/) ndroid.com/apk/res/android"

xmlns:app="[http://schemas.android.com/apk/res-auto"](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=".MainActivity">

<Button android:id="@+id/btnCall"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="call 5556" tools:ignore="MissingConstraints" />

</androidx.constraintlayout.widget.ConstraintLayout>

#### MainActivity.java

**package** com.example.telephoneapiapp;

**import** androidx.appcompat.app.AppCompatActivity;

**import** androidx.core.app.ActivityCompat;

**import** android.Manifest;

**import** android.content.Intent;

**import** android.content.pm.PackageManager;

**import** android.net.Uri; **import** android.os.Bundle; **import** android.view.View; **import** android.widget.Button;

**public class** MainActivity**extends** AppCompatActivity {

**private** Button **button**; @Override

**protected void** onCreate(Bundle savedInstanceState) { **super**.onCreate(savedInstanceState); setContentView(R.layout.***activity\_main***); **button**=(Button)findViewById(R.id.***btnCall***); **button**.setOnClickListener(**new** View.OnClickListener() {

@Override

**public void** onClick(View arg0) {

Intent callIntent= **new** Intent(Intent.***ACTION\_CALL***); callIntent.setData(Uri.*parse*(**"tel:5556"**)); **if**(ActivityCompat.*checkSelfPermission*(MainActivity.**this**, Manifest.permission.***CALL\_PHONE***)!= PackageManager.***PERMISSION\_GRANTED***){ **return**;

}

startActivity(callIntent);

}

});

}

}

### Output:

