**Android Programming Practical**

HOW TO CREATE A NEW PROJECT:

step 1: open android studio

step 2: select empty activity

step 3: enter app name and select java as language

step 4: go to res folder right click > new directory name it as **layout**

step 5: right click on LAYOUT folder -> new -> xml

step 6: name it ACTIVITY\_MAIN(front end file)

step 7: go to java + kotlin folder and select the first package and right click -> new java class file and name it Main\_Actvity(backend file)

**hello world practical**  
Also if you get hello world practical just follow this step:  
step 1: open android studio  
Step 2:new file

step 2: select empty view activity

step 3: enter app name and select java as language

And hello program will be ready no need to do anything else

**1. Implicit and explicit intent**

**Ans:**

### **XML Code (activity\_main.xml)**

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

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center"

android:orientation="vertical">

<Button

android:id="@+id/btnExplicit"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Explicit Intent"/>

<Button

android:id="@+id/btnImplicit"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Implicit Intent"/>

</LinearLayout>

### **Java Code (MainActivity.java)**

java

CopyEdit

package com.example.intents;

import android.content.Intent;

import android.net.Uri;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

Button btnExplicit = findViewById(R.id.btnExplicit);

Button btnImplicit = findViewById(R.id.btnImplicit);

btnExplicit.setOnClickListener(v -> {

Intent intent = new Intent(this, SecondActivity.class);

startActivity(intent);

});

btnImplicit.setOnClickListener(v -> {

Intent intent = new Intent(Intent.ACTION\_VIEW, Uri.parse("https://www.google.com"));

startActivity(intent);

});

}

}

### **Java Code (SecondActivity.java)**

java

CopyEdit

package com.example.intents;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

public class SecondActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_second);

}

}

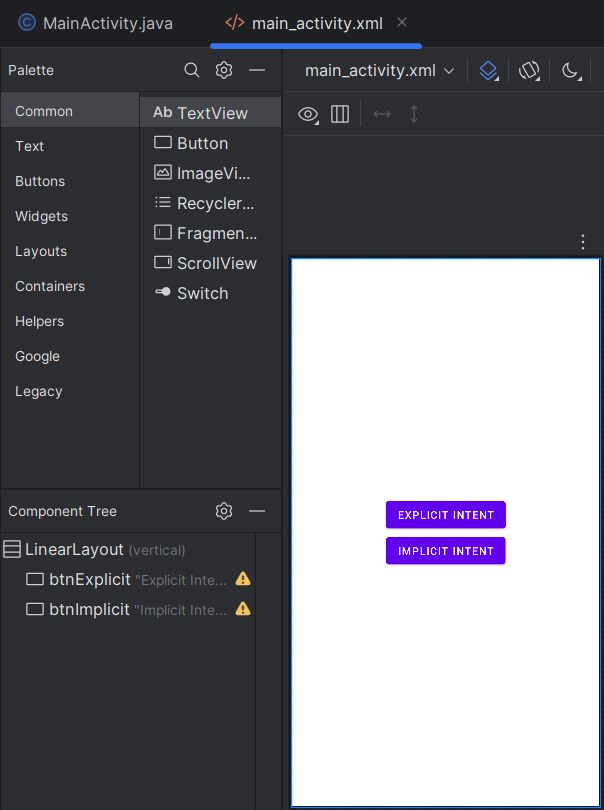
### **Steps to Implement**

1. Create a new **Empty Activity** project in Android Studio.
2. Replace **activity\_main.xml** and **MainActivity.java** with the above code.
3. Create **SecondActivity.java** and a corresponding **activity\_second.xml** file.
4. Run the app and test the buttons for **Explicit and Implicit intents**. 🚀

This code demonstrates **Explicit and Implicit Intents** in Android.

### **Functionality**

* **Explicit Intent:** Clicking the "Explicit Intent" button opens **SecondActivity**.
* **Implicit Intent:** Clicking the "Implicit Intent" button opens **Google** in a web browser.



**2. Hello world code**

**Ans:**  
**XML Code (activity\_main.xml)**

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

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center"

android:orientation="vertical">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Hello, World!"

android:textSize="24sp"/>

</LinearLayout>

### **Java Code (MainActivity.java)**

package com.example.helloworld;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

}

}

### **Steps to Run**

1. Open **Android Studio**, create a new **Empty Activity** project.
2. Replace **activity\_main.xml** and **MainActivity.java** with the above code.
3. Run the app – it will display **"Hello, World!"**

**3. Android activity life cycle**

**Ans:**

**Android Activity Lifecycle**

An Android activity goes through these lifecycle states:

1. **onCreate()** – Activity is created.
2. **onStart()** – Activity becomes visible.
3. **onResume()** – Activity is in the foreground.
4. **onPause()** – Activity is partially visible (e.g., another activity is opening).
5. **onStop()** – Activity is no longer visible.
6. **onDestroy()** – Activity is destroyed.
7. **onRestart()** – Called when restarting after stopping.

**XML Code (activity\_main.xml)**

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

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:gravity="center">

<TextView

android:id="@+id/textView"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Activity Lifecycle"

android:textSize="20sp"/>

</LinearLayout>

### **Java Code (MainActivity.java)**

package com.example.lifecycle;

import android.os.Bundle;

import android.util.Log;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

private static final String TAG = "Lifecycle";

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

Log.d(TAG, "onCreate");

}

@Override

protected void onStart() {

super.onStart();

Log.d(TAG, "onStart");

}

@Override

protected void onResume() {

super.onResume();

Log.d(TAG, "onResume");

}

@Override

protected void onPause() {

super.onPause();

Log.d(TAG, "onPause");

}

@Override

protected void onStop() {

super.onStop();

Log.d(TAG, "onStop");

}

@Override

protected void onDestroy() {

super.onDestroy();

Log.d(TAG, "onDestroy");

}

@Override

protected void onRestart() {

super.onRestart();

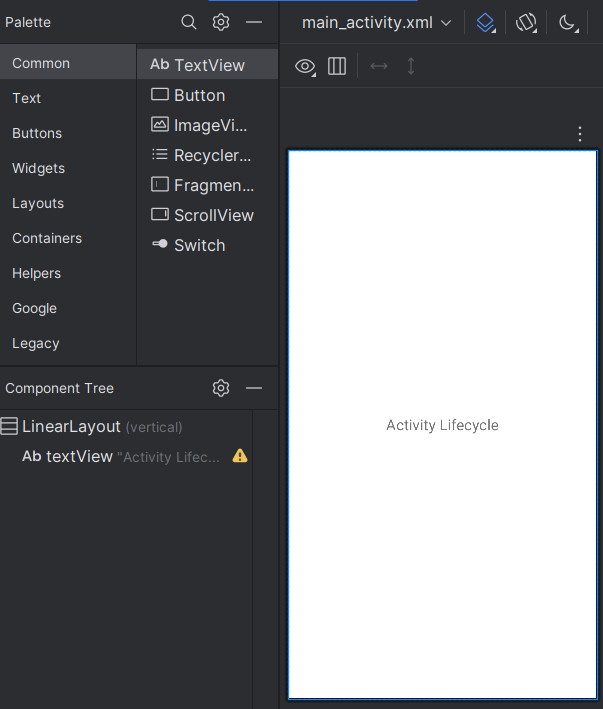
Log.d(TAG, "onRestart");

}

}

### **Steps to Implement**

1. Open **Android Studio**, create a new **Empty Activity** project.
2. Replace **activity\_main.xml** with the given XML code.
3. Replace **MainActivity.java** with the Java code.
4. Run the app and observe **Logcat** for lifecycle method calls.



**4. Registration form : checkbox, textview, button, radio button.**

**Ans:**

w

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Register"

android:textSize="20sp" />

<EditText

android:id="@+id/etName"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Enter Name"/>

<RadioGroup

android:id="@+id/radioGroup"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<RadioButton android:id="@+id/rbMale" android:text="Male"/>

<RadioButton android:id="@+id/rbFemale" android:text="Female"/>

</RadioGroup>

<CheckBox android:id="@+id/cbAgree" android:text="I agree to terms"/>

<Button

android:id="@+id/btnSubmit"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Submit"/>

</LinearLayout>

### **Java Code (MainActivity.java)**

package com.example.registration;

import android.os.Bundle;

import android.view.View;

import android.widget.\*;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

EditText etName = findViewById(R.id.etName);

RadioGroup radioGroup = findViewById(R.id.radioGroup);

CheckBox cbAgree = findViewById(R.id.cbAgree);

Button btnSubmit = findViewById(R.id.btnSubmit);

btnSubmit.setOnClickListener(v -> {

int selectedId = radioGroup.getCheckedRadioButtonId();

RadioButton rbSelected = findViewById(selectedId);

String name = etName.getText().toString();

boolean isChecked = cbAgree.isChecked();

if (isChecked && !name.isEmpty() && rbSelected != null) {

Toast.makeText(this, "Registered Successfully!", Toast.LENGTH\_SHORT).show();

} else {

Toast.makeText(this, "Fill all fields!", Toast.LENGTH\_SHORT).show();

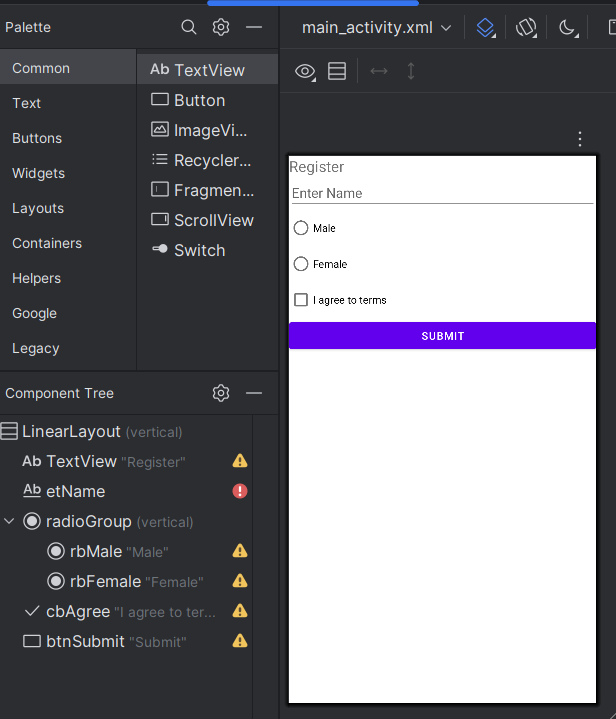
}

});

}

}

**Functionality**

* User enters a name, selects gender (RadioButton), agrees to terms (CheckBox), and clicks **Submit**.
* Displays **"Registered Successfully!"** if all fields are filled, else shows an error.  
    
  

**5. Menu dialog**

**Ans:**

### **XML Code (activity\_main.xml)**

xml

CopyEdit

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

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center"

android:orientation="vertical">

<Button

android:id="@+id/btnMenu"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Open Menu"/>

</LinearLayout>

### **Java Code (MainActivity.java)**

java

CopyEdit

package com.example.menudialog;

import android.app.AlertDialog;

import android.content.DialogInterface;

import android.os.Bundle;

import android.widget.Button;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

Button btnMenu = findViewById(R.id.btnMenu);

btnMenu.setOnClickListener(v -> {

String[] options = {"Option 1", "Option 2", "Option 3"};

new AlertDialog.Builder(this)

.setTitle("Select an Option")

.setItems(options, (dialog, which) ->

Toast.makeText(this, "You selected: " + options[which], Toast.LENGTH\_SHORT).show())

.setNegativeButton("Cancel", null)

.show();

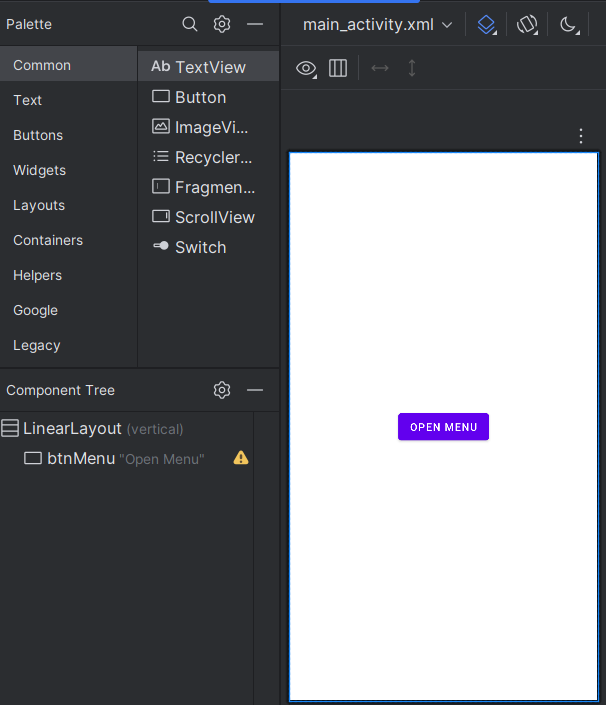
});

}

}

### **Functionality**

* Clicking **"Open Menu"** shows a **dialog menu** with three options.
* Selecting an option shows a **Toast message** with the selected choice.



**6.Radio button group**

**Ans:**

### **XML Code (activity\_main.xml)**

xml

CopyEdit

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

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:padding="16dp">

<RadioGroup android:id="@+id/radioGroup"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content">

<RadioButton android:id="@+id/rbOption1" android:text="Option 1"/>

<RadioButton android:id="@+id/rbOption2" android:text="Option 2"/>

</RadioGroup>

<Button android:id="@+id/btnSubmit" android:text="Submit"

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

</LinearLayout>

### **Java Code (MainActivity.java)**

java

CopyEdit

package com.example.radiogroup;

import android.os.Bundle;

import android.view.View;

import android.widget.\*;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

RadioGroup radioGroup = findViewById(R.id.radioGroup);

Button btnSubmit = findViewById(R.id.btnSubmit);

btnSubmit.setOnClickListener(v -> {

int selectedId = radioGroup.getCheckedRadioButtonId();

if (selectedId != -1) {

RadioButton selected = findViewById(selectedId);

Toast.makeText(this, "Selected: " + selected.getText(), Toast.LENGTH\_SHORT).show();

} else {

Toast.makeText(this, "Select an option", Toast.LENGTH\_SHORT).show();

}

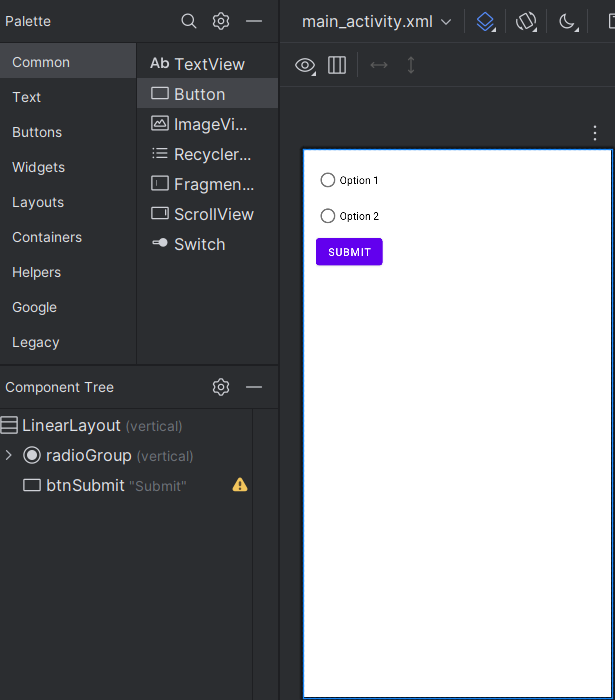
});

}

}

### **Functionality**

* User selects an option (RadioButton) and clicks **Submit**.
* Displays **selected option** in a **Toast message**

****

**7. Layouts: linear,table,grid**

**Ans:**

**Linear Layout (activity\_main.xml)**

xml

CopyEdit

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<TextView android:text="Linear Layout" android:textSize="18sp"/>

<Button android:text="Button 1"/>

<Button android:text="Button 2"/>

</LinearLayout>

### **Table Layout (activity\_main.xml)**

xml

CopyEdit

<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

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

<TableRow>

<TextView android:text="Row 1, Col 1"/>

<TextView android:text="Row 1, Col 2"/>

</TableRow>

<TableRow>

<TextView android:text="Row 2, Col 1"/>

<TextView android:text="Row 2, Col 2"/>

</TableRow>

</TableLayout>

**Grid Layout (activity\_grid.xml )**

xml

CopyEdit

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

<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:columnCount="2"

android:padding="16dp">

<TextView android:text="Item 1"/>

<TextView android:text="Item 2"/>

<Button android:text="Button 1"/>

<Button android:text="Button 2"/>

</GridLayout>

### **Functionality**

* **LinearLayout** → Arranges elements **vertically/horizontally**.
* **TableLayout** → Organizes elements in **rows & columns**.

**8.Notification code**

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

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:gravity="center">

<Button

android:id="@+id/btnShowNotification"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Show Notification" />

</LinearLayout>

import android.app.\*;

import android.content.Context;

import android.os.Build;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.app.NotificationCompat;

public class MainActivity extends AppCompatActivity {

private static final String CHANNEL\_ID = "SimpleChannel";

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

createNotificationChannel();

findViewById(R.id.btnShowNotification).setOnClickListener(v -> showNotification());

}

private void showNotification() {

Notification notification = new NotificationCompat.Builder(this, CHANNEL\_ID)

.setContentTitle("Hello!")

.setContentText("This is a simple notification.")

.setSmallIcon(R.drawable.ic\_launcher\_foreground)

.build();

NotificationManager manager = getSystemService(NotificationManager.class);

if (manager != null) {

manager.notify(1, notification);

}

}

private void createNotificationChannel() {

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {

NotificationChannel channel = new NotificationChannel(

CHANNEL\_ID, "Simple Notifications",

NotificationManager.IMPORTANCE\_DEFAULT);

NotificationManager manager = getSystemService(NotificationManager.class);

if (manager != null) {

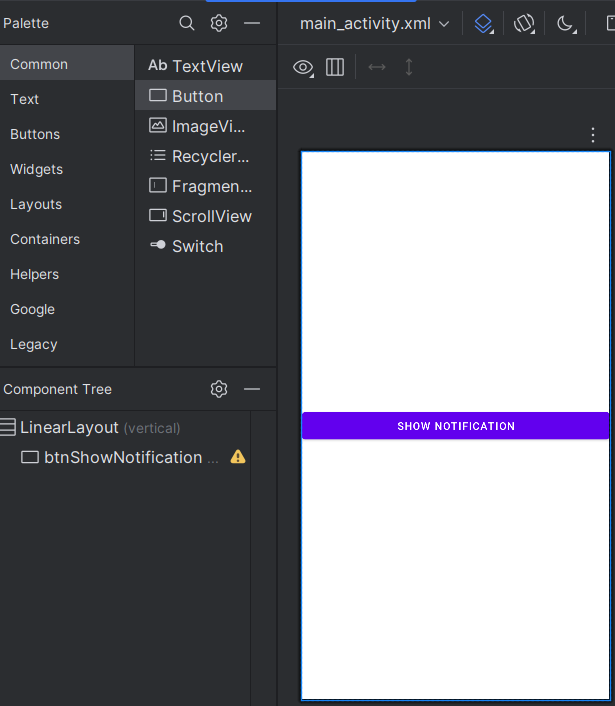
manager.createNotificationChannel(channel);

}

}

}

}



**9.Program to pass data from one activity to another activity using intent**

**Ans:**   
  
**Steps to Pass Data Between Activities using Intent**

1️⃣ **Create Two Activities:** MainActivity.java and SecondActivity.java  
 2️⃣ **Send Data from MainActivity using Intent** 3️⃣ **Receive Data in SecondActivity**

### **Java Code**

#### **MainActivity.java (Sender)**

java

CopyEdit

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

Button btnSend = findViewById(R.id.btnSend);

btnSend.setOnClickListener(v -> {

Intent intent = new Intent(this, SecondActivity.class);

intent.putExtra("message", "Hello, Second Activity!");

startActivity(intent);

});

}

}

#### **SecondActivity.java (Receiver)**

java

CopyEdit

import android.os.Bundle;

import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class SecondActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_second);

String message = getIntent().getStringExtra("message");

((TextView) findViewById(R.id.txtMessage)).setText(message);

}

}

### **XML Layouts**

#### **activity\_main.xml**

xml

CopyEdit

<Button

android:id="@+id/btnSend"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Send Data" />

#### **activity\_second.xml**

xml

CopyEdit

<TextView

android:id="@+id/txtMessage"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

**Viva**

1. **What is Android?** Android is an open-source operating system developed by Google for mobile devices like smartphones and tablets.
2. **Android Components:**
   * **Activity:** Represents a single screen in an Android app.
   * **Services:** Background processes running independently of UI.
   * **Content Providers:** Manage and share app data between applications.
   * **Broadcast Receiver:** Listens for system-wide broadcast messages like battery low or network change.
3. **Fragments** Fragments are reusable UI components within an activity that manage their own lifecycle.
4. **Stages in Activity Lifecycle**
   * **onCreate()** → Initialization
   * **onStart()** → Visible to the user
   * **onResume()** → Active & running
   * **onPause()** → Partially visible
   * **onStop()** → No longer visible
   * **onDestroy()** → Cleanup before removal
5. **Stages in Fragment Lifecycle**
   * **onAttach(), onCreate(), onCreateView()** → Initialization
   * **onStart(), onResume()** → Visible & active
   * **onPause(), onStop()** → Inactive
   * **onDestroyView(), onDestroy(), onDetach()** → Cleanup
6. **What is Layout?** A layout defines the structure and appearance of UI elements in an Android app.
7. **Different Types of Layouts**
   * **LinearLayout** (arranges elements in a row/column)
   * **RelativeLayout** (positions elements relative to others)
   * **ConstraintLayout** (flexible, responsive design)
   * **TableLayout** (tabular format)
   * **GridLayout** (grid-based structure)
8. **What is a Widget?** A widget is a UI element like Button, TextView, ImageView, or EditText used in Android applications.
9. **What are Notifications?** Notifications alert users about background events like messages, updates, or reminders.
10. **What are Intents?** Intents are used to start activities, services, or communicate between components (explicit or implicit).
11. **What is SQLite Database?** SQLite is a lightweight, local database for storing structured data in Android applications.
12. **Explain JSON** JSON (JavaScript Object Notation) is a lightweight data format used for data exchange between a server and a client.
13. **Programming Threats** Common threats include malware, phishing, SQL injection, and unauthorized access to sensitive data.
14. **How to Create and Run an Android Project?** Use Android Studio → Create a new project → Write code → Build & run on an emulator or real device.
15. **How to Deploy/Publish an Android App?** Generate a signed APK → Upload to Google Play Store with app details, screenshots, and pricing options.