LECTURE 16: UI PROGRAMMING PART 3

BY LINA HAMMAD & AHMAD BARGHASH

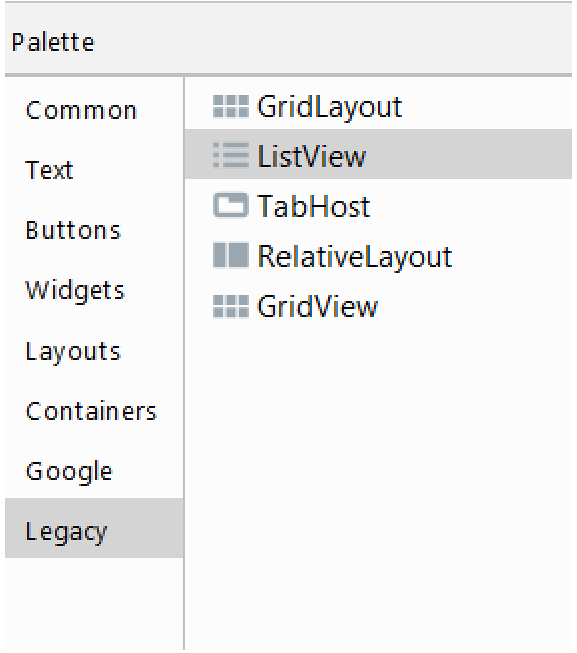
In this tutorial, we shall learn how to display elements of an array using Android ListView with the help of Kotlin Android Application. Then we will proceed further by adding ListView Item Click Listener so that a particular action would be taken when a click is made on an item in ListView

LEGACY – LISTVIEW

- Android **ListView** is a view which groups several items and display them in vertical scrollable list.

- key attributes:

android:clickable="***bool***"	set to false to disable the list
android:id="@+id/***theID***"	unique ID for use in Kotlin code
android:entries="@array/***array***"	set of options to appear in the list (must match an array in strings.xml)



LEGACY – LISTVIEW

- Xml code:

```
<ListView
        android:id="@+id/listView1"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>
```

- Kotlin code:

```
// use arrayadapter and define an array
val arrayAdapter: ArrayAdapter<*>
var cities = arrayOf("Melbourne", "Vienna", "Vancouver", "Toronto", "Calgary", "Adelaide",
        "Perth", "Auckland", "Helsinki", "Hamburg", "Munich", "New York", "Sydney",
        "Paris", "Cape Town", "Barcelona", "London", "Bangkok")

// access the listView from xml file
var myListView = findViewById<ListView>(R.id.listView1)
arrayAdapter = ArrayAdapter(this,
        android.R.layout.simple_list_item_1, cities)
myListView.adapter = arrayAdapter
```

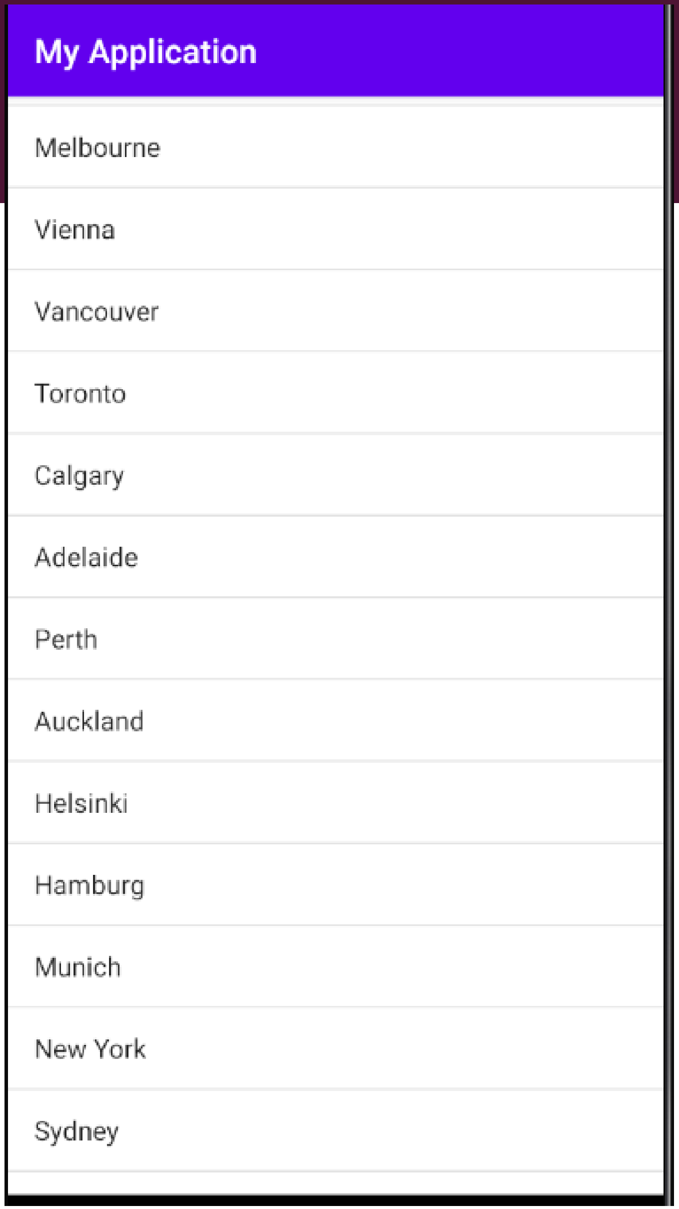
Remember to add the Koltin code inside the onCreate function.

LEGACY – LISTVIEW

- On click item, kotlin code:

```
myListView.onItemClickListener = object : OnItemClickListener {
    override fun onItemClick(parent: AdapterView<*>, view: View,
                             position: Int, id: Long) {

        // value of item that is clicked
        val itemValue = myListView.getItemAtPosition(position) as String

        // Toast the values
        Toast.makeText(applicationContext,
                "Position :$position\nItem Value : $itemValue", Toast.LENGTH_LONG).show()
    }
}
```

After run the program



My Application

Melbourne

Vienna

Vancouver

Toronto

Calgary

Adelaide

Perth

Auckland

Helsinki

Hamburg

Munich

New York

Sydney

LIST ADAPTERS

- **adapter**: Helps turn list data into list view items.
- common adapters: ArrayAdapter, CursorAdapter
- Syntax for creating an adapter:

```
ArrayAdapter name = ArrayAdapter(activity, layout, array)
```

  - the ***activity*** is usually this
  - the default ***layout*** for lists is android.R.layout.simple_list_item_1
  - get the ***array*** by reading your file or data source of choice (it can be an array like String[ ], or a list like ArrayList<String>)
- Once you have an adapter, you can attach it to your list by calling the setAdapter method of the ListView object in the Kotlin code.

LIST EVENTS

- List views respond to the following events:

1. **setOnItemClickListener**(AdapterView.OnItemClickListener)
   - Listener for when an item in the list has been clicked.
2. **setOnItemLongClickListener**(AdapterView.OnItemLongClickListener)
   - Listener for when an item in the list has been clicked and held.
3. **setOnItemSelectedListener**(AdapterView.OnItemSelectedListener)
   - Listener for when an item in the list has been selected.
4. Others:
   - onDrag, onFocusChanged, onHover, onKey, onScroll, onTouch, ...

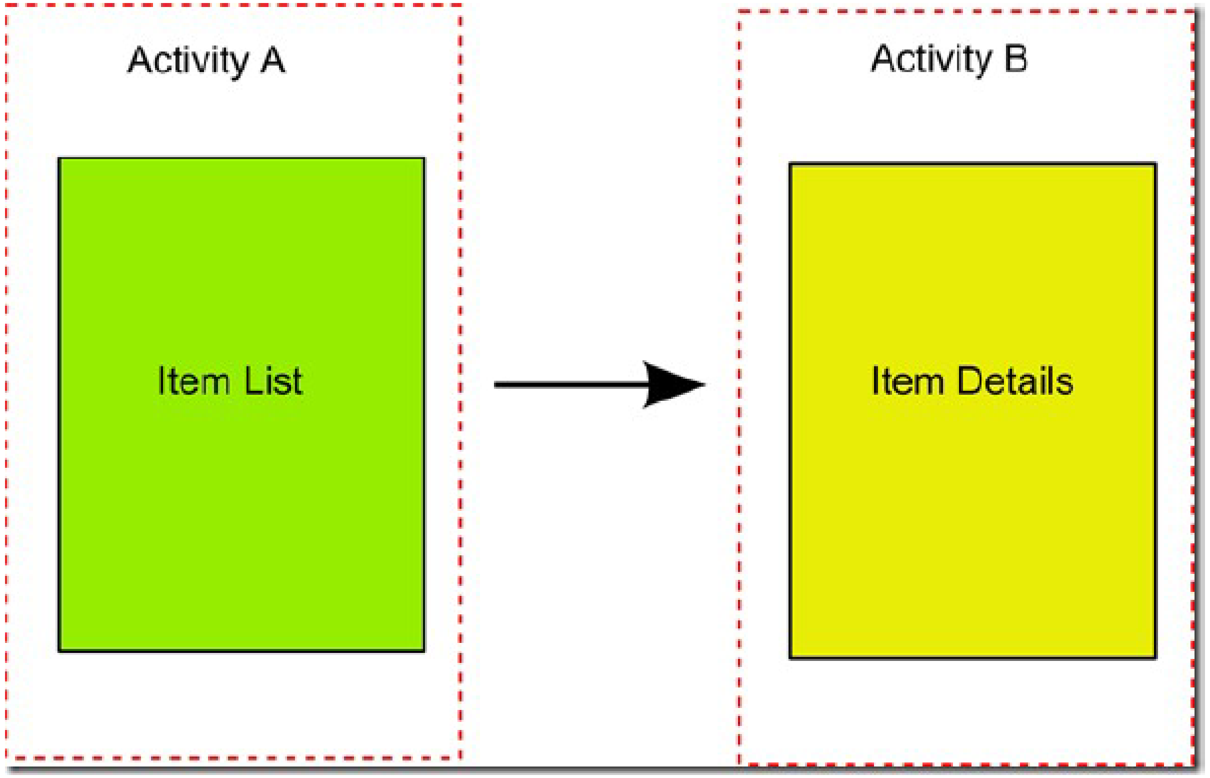
LECTURE 17: INTENTS

BY LINA HAMMAD & AHMAD BARGHASH

In this tutorial, you will learn Intents in Android. This tutorials describes the usage of intents to communicate between Android components.

MULTIPLE ACTIVITIES

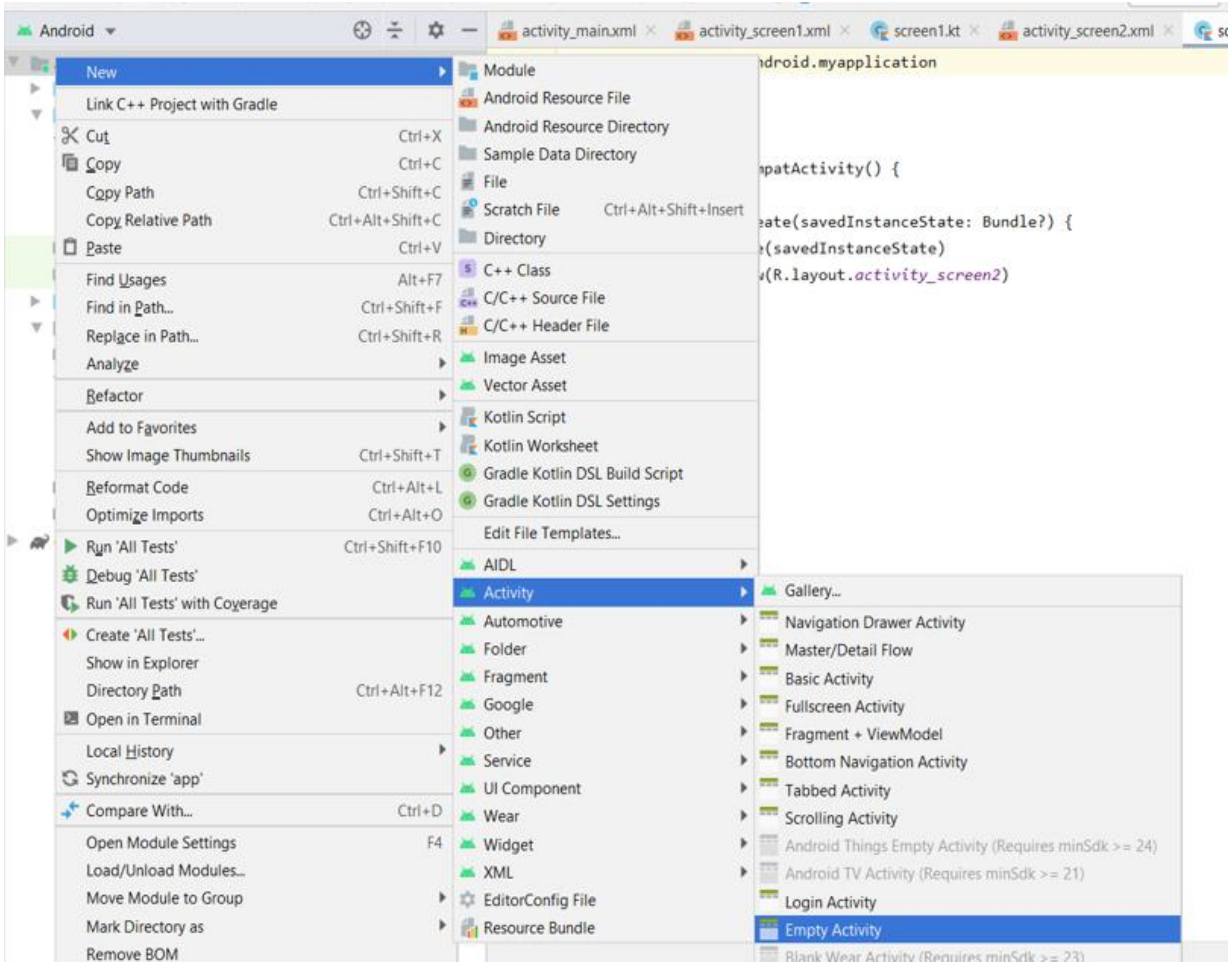
- Example: In an address book app, the main activity is a list of contacts, and clicking on a contact goes to another activity for viewing details.
- An activity A can launch another activity B in response to an event.
- The activity A can pass data to B.
- The second activity B can send data back to A when it is done



ADDING AN ACTIVITY

- In Android Studio, right click "**app**" at left: **New -> Activity.** This step will:

1. creates a new .XML file in res/layouts
2. creates a new .kt class in src/java
3. adds information to AndroidManifest.xml about the activity (without this information, the app will not allow the activity)



```
ndroid.myapplication

npatActivity() {

eate(savedInstanceState: Bundle?) {
e(savedInstanceState)
v(R.layout.activity_screen2)
```

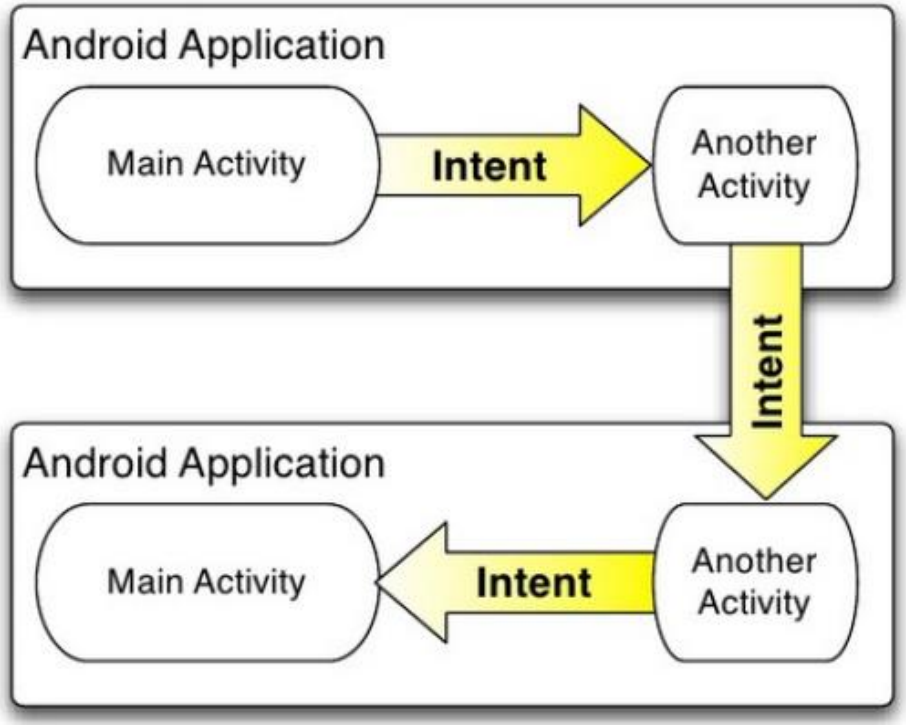
ACTIVITIES IN MANIFEST

- Every activity has an entry in project's **AndroidManifest.xml**, added automatically by Android Studio:

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".ThirdActivity" />
        <activity android:name=".SecondActivity" />
        <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>
```

INTENTS

- **Intent**: a bridge between activities; a way for one activity to invoke another.
- The activity can be in the same app or in a different app.
- Can store **extra data** to pass as "parameters" to that activity.
- Second activity can "**return**" information back to the caller if needed.



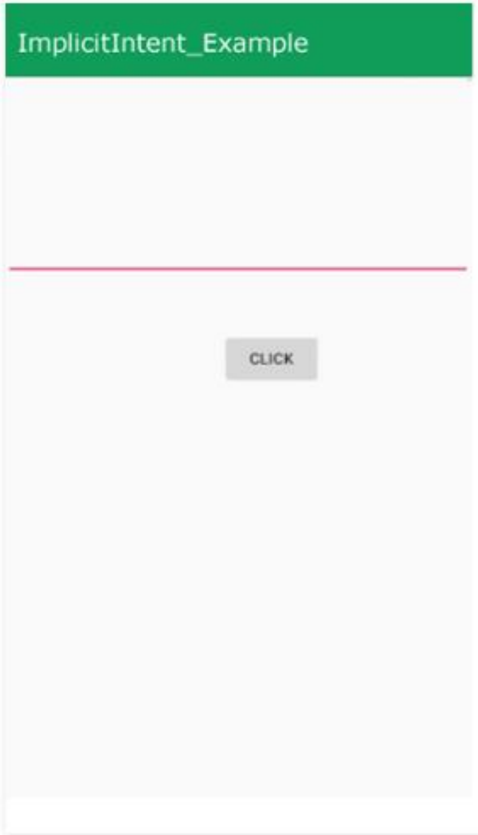
EXPLICIT VS IMPLICIT INTENTS

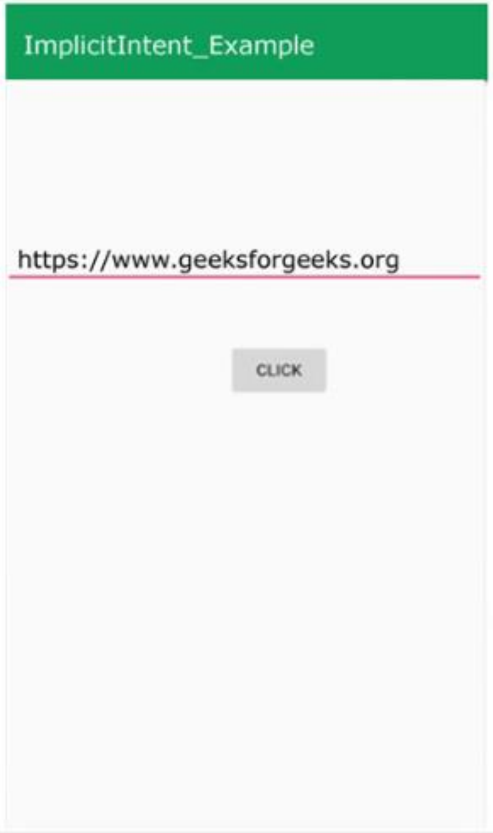
- Intents come in two varieties: Explicit and Implicit. We use explicit intents when we explicitly name the class of the target activity that will handle the intent. Implicit intents are used without a class name, where Android will help determine an appropriate Activity to handle the intent.

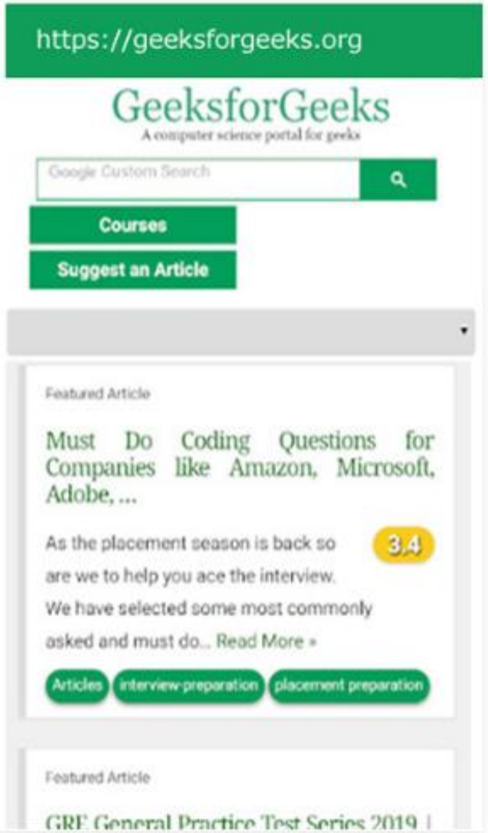
IMPLICIT INTENT

- **Implicit Intent:** Using implicit Intent, component can't be specifying. An action to be performed is declared by implicit intent. Then android operating system will filter out component which will response to the action.

- **Note**: In this example, no component is specified, instead an action is performed i.e. a webpage is going to be opened. As you type the name of your desired webpage and click on 'CLICK' button. Your webpage is opened.







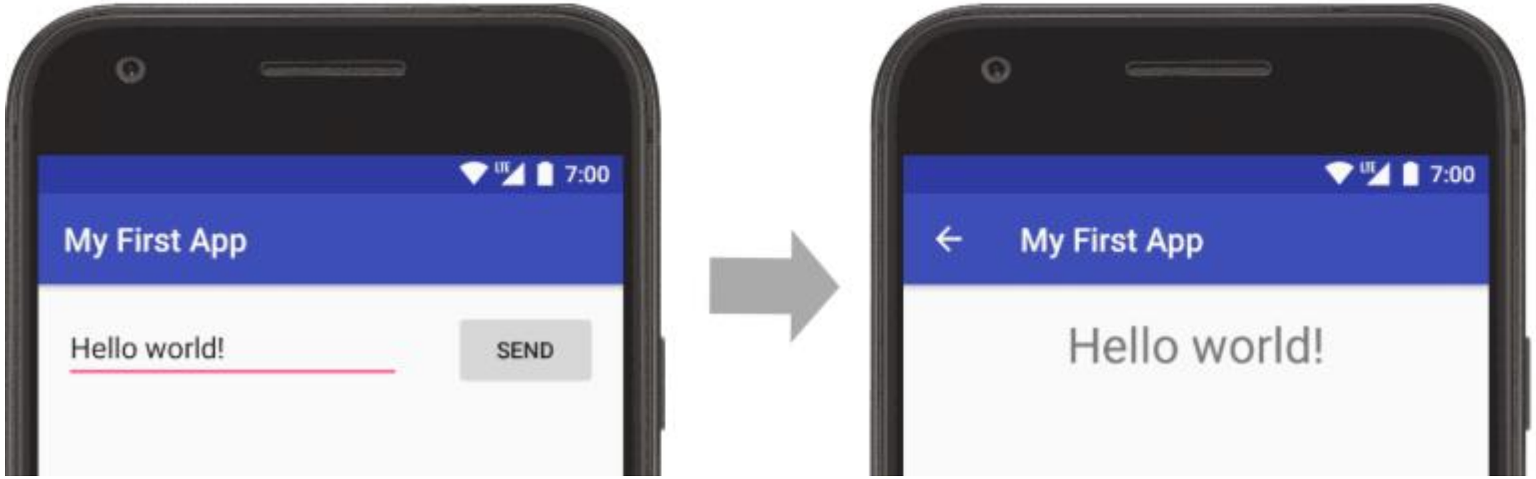
Must Do Coding Questions for Companies like Amazon, Microsoft, Adobe, ...

As the placement season is back so are we to help you ace the interview. We have selected some most commonly asked and must do... Read More »

GRE General Practice Test Series 2019 |

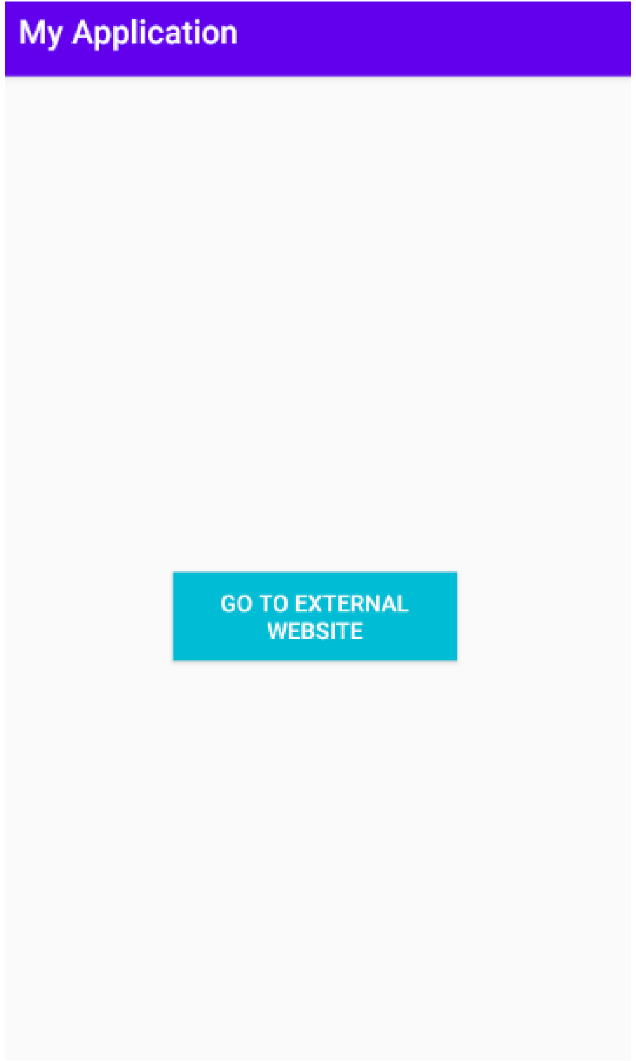
EXPLICIT INTENT

- **Explicit Intent:** Using explicit intent any other component can be specified. In other words, the targeted component is specified by explicit intent. So only the specified target component will be invoked.

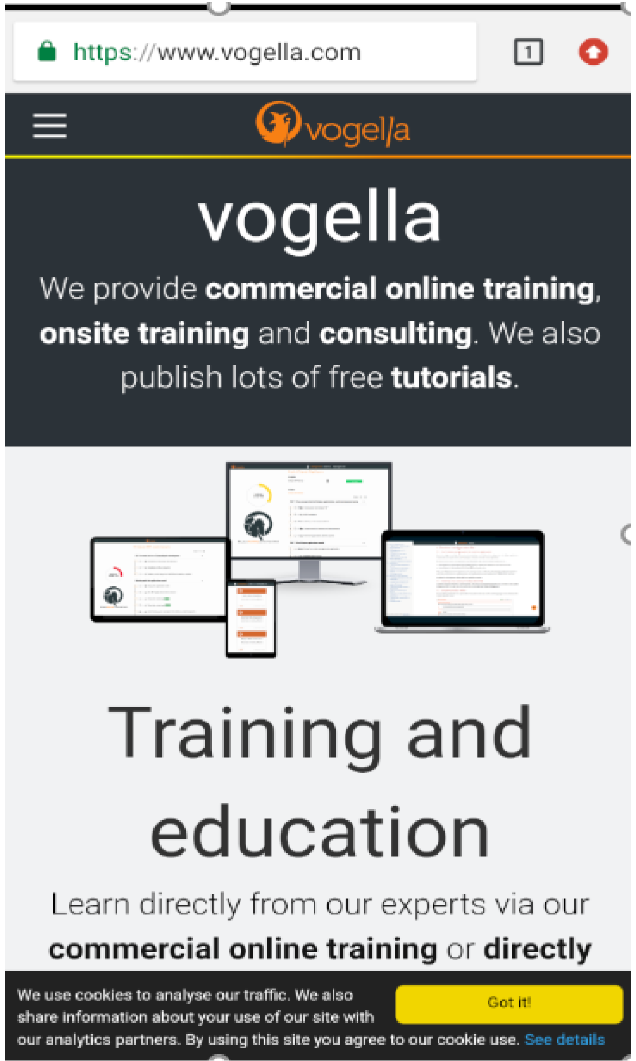


- In the above example, there are two activities (FirstActivity, SecondActivity). When you click on 'SEND' button in the first activity, then you move to second activity with text entered on the first activity and displayed on the second activity.

IMPLICIT INTENT EXAMPLE



The vogella website will open after click on the Go to external website button



vogella

We provide **commercial online training**, **onsite training** and **consulting**. We also publish lots of free **tutorials**.

Training and education

Learn directly from our experts via our **commercial online training** or **directly**

We use cookies to analyse our traffic. We also share information about your use of our site with our analytics partners. By using this site you agree to our cookie use. See details

IMPLICIT INTENT EXAMPLE

- Activity_main.xml code:

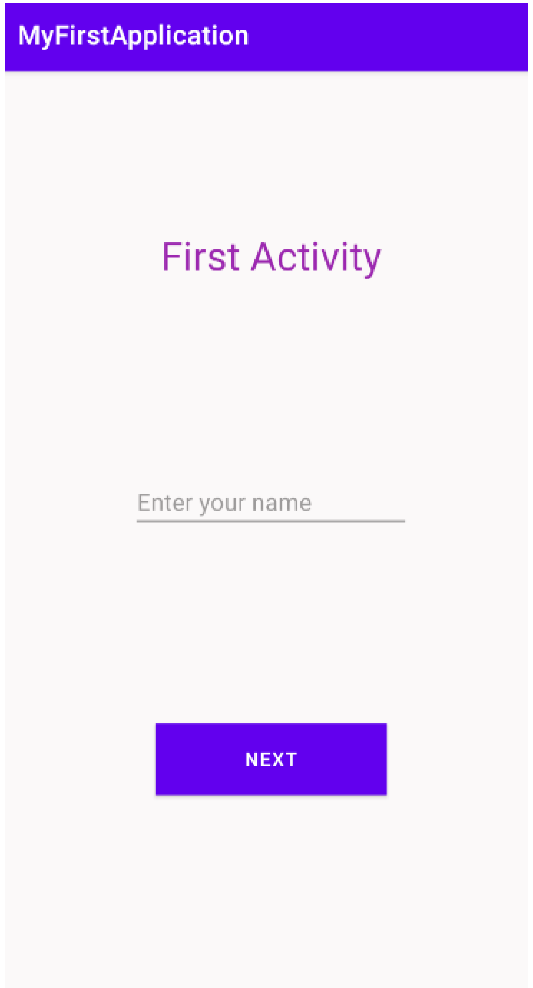
```
<Button
    android:id="@+id/btn1"
    android:layout_width="173dp"
    android:layout_height="54dp"
    android:background="#00BCD4"
    android:text="Go to external website"
    android:textColor="#FFFFFF"/>
```

- ActivityMain.kt code:

```
var btn = findViewById<Button>(R.id.btn1)
btn.setOnClickListener(){
    val i = Intent(Intent.ACTION_VIEW, Uri.parse("https://www.vogella.com/"))
    startActivity(i)
}
```

EXPLICIT INTENT EXAMPLE

- activity_main.xml code:



```
<TextView
    android:id="@+id/txtView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="First Activity"
    android:textColor="#9C27B0"
    android:textSize="30sp"/>

<EditText
    android:id="@+id/nameEditText"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:hint="Enter your name"
    android:inputType="textPersonName"
    app:layout_constraintTop_toBottomOf="@+id/txtView1"
/>

<Button
    android:id="@+id/nextBtn1"
    android:layout_width="173dp"
    android:layout_height="54dp"
    android:background="#00BCD4"
    android:text="Click"
    android:textColor="#FFFFFF"
    app:layout_constraintTop_toBottomOf="@+id/editText1"
/>
```

EXPLICIT INTENT EXAMPLE

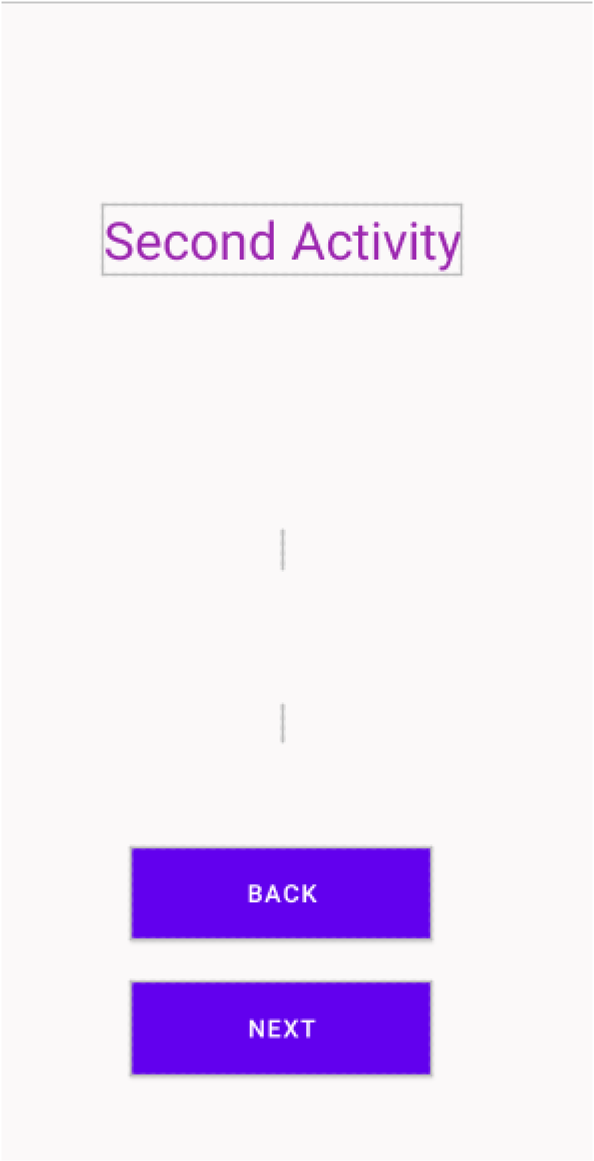
- MainActivity.kt code:

```
val courseName: String = "Mobile Application Development"

var nextBtn1 = findViewById<Button>(R.id.nextBtn1)
var editText1 = findViewById<EditText>(R.id.nameEditText)

nextBtn1.setOnClickListener() {
    intent = Intent(this, SecondActivity::class.java)
    intent.putExtra("course_name", courseName)
    intent.putExtra("your_name", editText1.text.toString())
    startActivity(intent)
}
```

EXPLICIT INTENT EXAMPLE

- activity_second.xml code:

```
<TextView
    android:id="@+id/txtView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Second Activity"
    android:textColor="#9C27B0"
    android:textSize="30sp"/>

<TextView
    android:id="@+id/nameTxtView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#8BC34A"
    android:textSize="16sp"/>

<TextView
    android:id="@+id/courseTxtView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#FF5722"
    android:textSize="16sp"/>

<Button
    android:id="@+id/backBtn1"
    android:layout_width="173dp"
    android:layout_height="54dp"
    android:background="#00BCD4"
    android:text="Back"
    android:textColor="#FFFFFF"/>


<Button
    android:id="@+id/nextBtn2"
    android:layout_width="173dp"
    android:layout_height="54dp"
    android:background="#00BCD4"
    android:text= "Next"
    android:textColor="#FFFFFF"/>
```



EXPLICIT INTENT EXAMPLE

- SecondActivity.kt code (in order to load the layout):

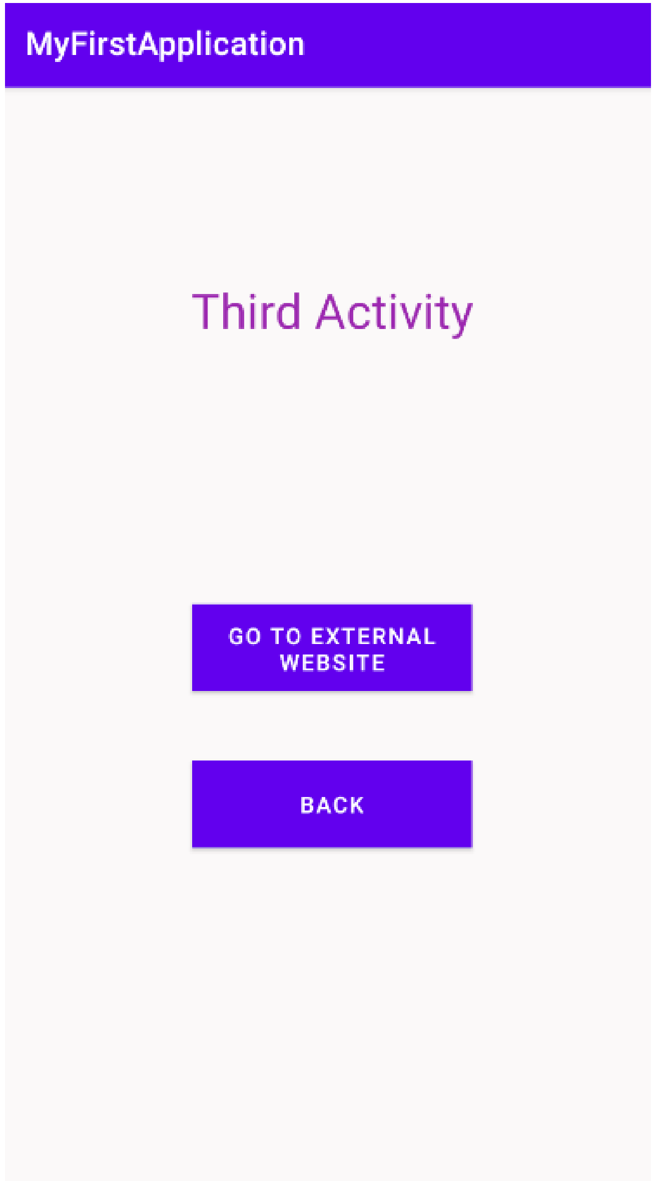
```
var intent = intent
val courseName =intent.getStringExtra("course_name")
val yourName =intent.getStringExtra("your_name")

var txtView2 = findViewById<TextView>(R.id.nameTxtView)
var txtView3 = findViewById<TextView>(R.id.courseTxtView)

txtView2.text = "Your name is: $yourName"
txtView3.text = "Course name is: $courseName"
```

- SecondActivity.kt:

```
//back button
var btn2 = findViewById<Button>(R.id.backBtn1)
btn2.setOnClickListener(){
    intent = Intent(this,MainActivity::class.java)
    startActivity(intent)
}

//next button
var btn3 = findViewById<Button>(R.id.nextBtn2)
btn3.setOnClickListener(){
    intent = Intent(this,ThirdActivity::class.java)
    startActivity(intent)
}
```

ACTIVITY_THIRD.XML CODE

```
<TextView
    android:id="@+id/txtView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp"
    android:text="Third Activity"
    android:textColor="#9C27B0"
    android:textSize="30sp"/>

<Button
    android:id="@+id/externalWebBtn"
    android:layout_width="173dp"
    android:layout_height="54dp"
    android:background="#00BCD4"
    android:text="Go to external website"
    android:textColor="#FFFFFF"/>

<Button
    android:id="@+id/backBtn2"
    android:layout_width="173dp"
    android:layout_height="54dp"
    android:background="#00BCD4"
    android:text="Back"
    android:textColor="#FFFFFF" />
```

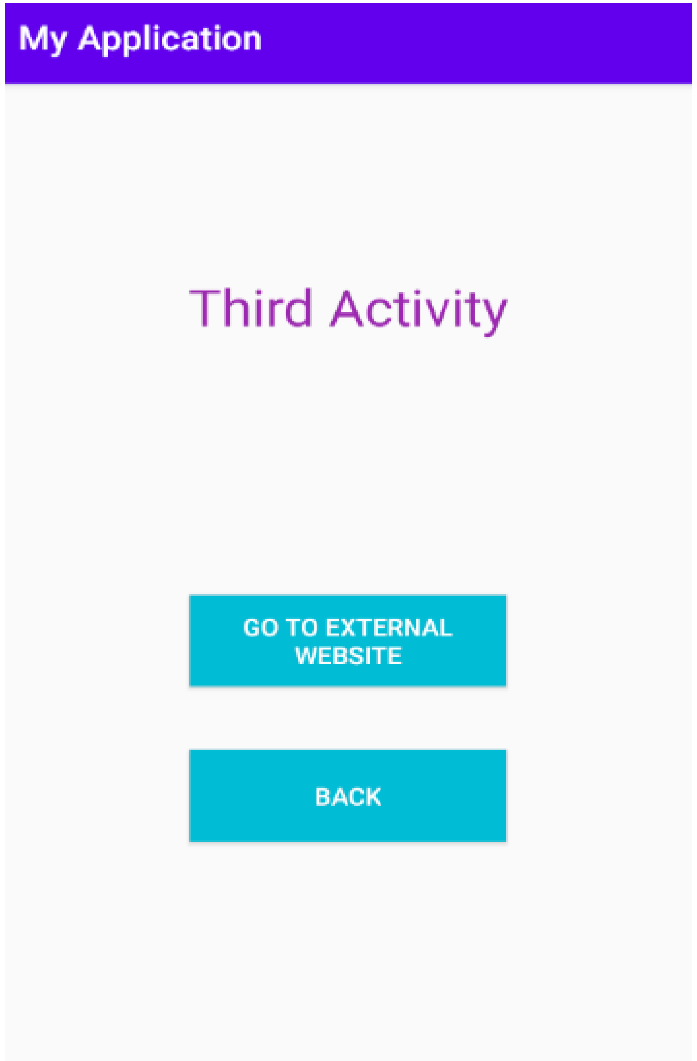


EXPLICIT INTENT EXAMPLE

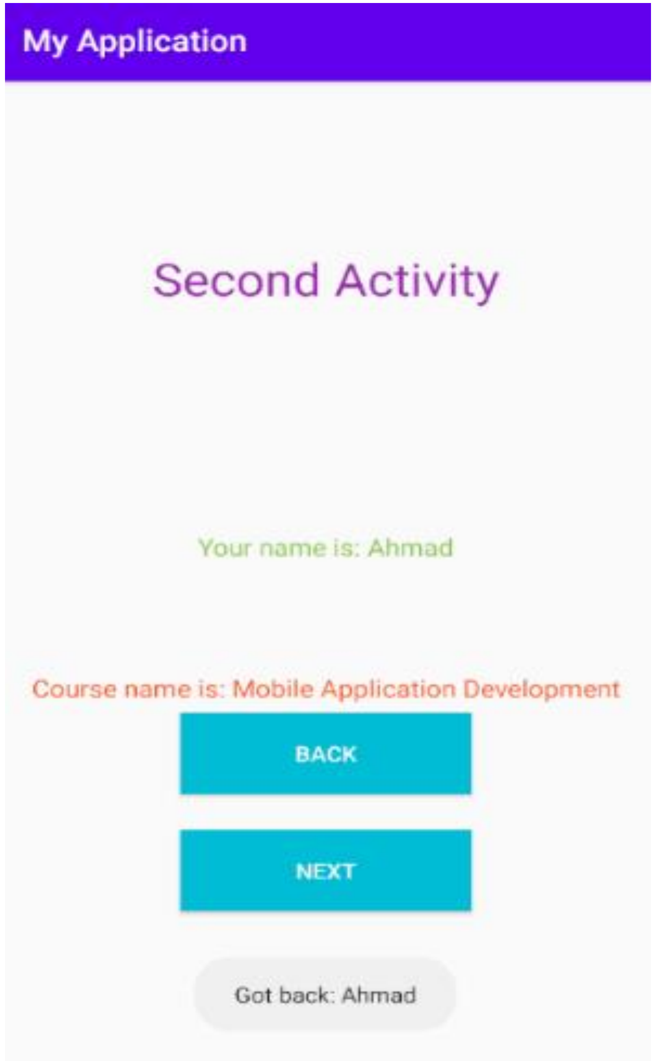
- ThirdActivity.kt code (in order to load the layout):

```kotlin
//next button
var btn4 = findViewById<Button>(R.id.externalWebBtn)
btn4.setOnClickListener() {
    val i = Intent(Intent.ACTION_VIEW, Uri.parse("https://www.vogella.com/"))
    startActivity(i)
}

//back button
var btn5 = findViewById<Button>(R.id.backBtn2)
btn5.setOnClickListener() {
    val intent = Intent(this, SecondActivity::class.java)
    startActivity(intent)
}
```

GRABBING THE RESULT



After click on the back button, you will come back to the second activity

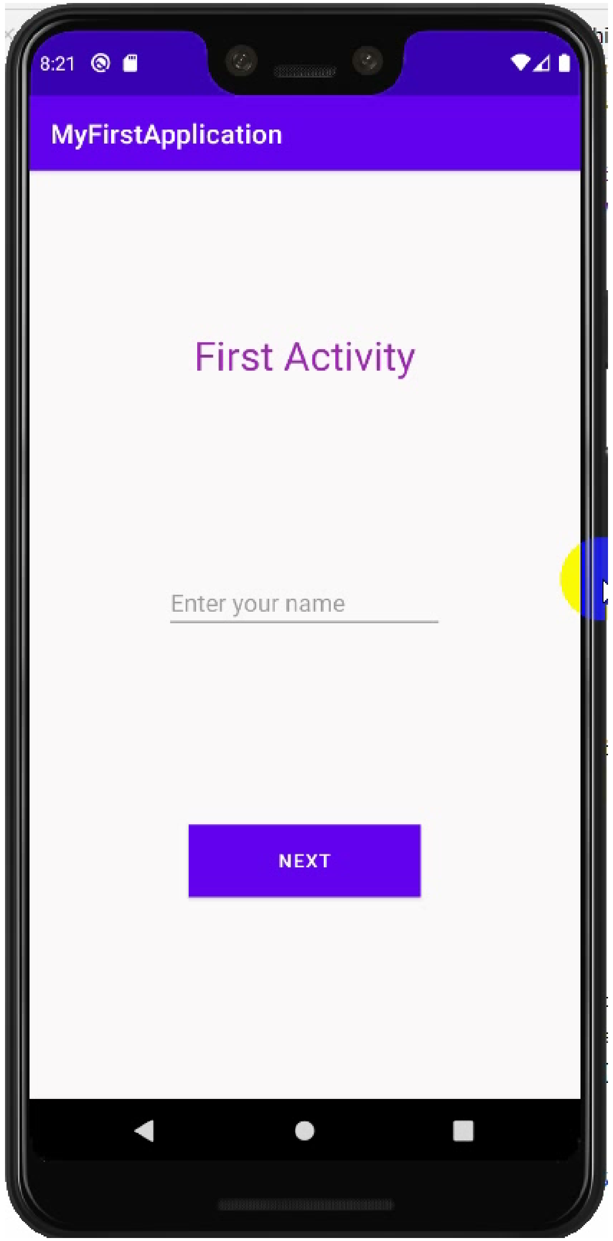


Your name is: Ahmad

Course name is: Mobile Application Development

Got back: Ahmad

RESULT OF THE EXPLICIT INTENT EXAMPLE


8:21
MyFirstApplication
First Activity
Enter your name
NEXT