**MIND STORM SOFTWARE PVT LTD**

**Hands-On Exercise – ex02 - Multiple Activities.doc**

**Objective**Multiple Activities in an Android Application and how to pass data across activities.

**Assumptions**

* Development Environment for Android (Java SDK, Eclipse, Android SDK) has been setup successfully.
* You are familiar with using Eclipse.
* Android SDK 4.x is available and Android Virtual Devices are already created.
* Start the Android Virtual Device to save time.
  + Click on **Window 🡪 Android SDK and AVD Manager**.
  + Select an **Android 4.x** compatible AVD and click on **Start**
  + Select **Scale display to real size** and provide a **Screen Size (in)** as **5 inches** or any other appropriate size for your development machine.

**Step by Step Instructions**

**Step 1 – Create the Android Project**

1. Create a new project. Click on **File🡪New 🡪 Android Application Project**
2. Enter **Project Name or Application Name** as **MultipleActivitiesExercise.** Click on **Next.**
3. Enter **Package Name** as **com.mindstorm.multipleactivities** . Click on **Next**.
4. Deselect (Uncheck) the **Create custom launcher icon.** Click on **Next.**
5. In **Create Activity**, go with the default options i.e. go with Create Activity and Blank Activity as selected. Click on **Next.**
6. On the **New Blank Activity,** leave as default and Click on **Finish**
7. (Optional): Verify that the Project runs in your Emulator by **Right-click** the **Project** and **Run As 🡪 Android Application**

**Step 2 – Create a Second Activity**

1. Go to **Package Explorer** in Eclipse for the **MultipleActivitiesExercise** project.
2. Select the package **com.mindstorm.multipleactivities** from within the **src** directory.
3. **Right Click** and select **New 🡪 Class**
4. Enter the name as **ViewSubjectsActivity**
5. Enter the **Superclass** as **android.app.Activity**
6. Click on **Finish.**

**Step 3 – Create a Third Activity**

1. Go to **Package Explorer** in Eclipse for the **MultipleActivitiesExercise** project.
2. Select the package **com.mindstorm.multipleactivities** from within the **src** directory.
3. **Right Click** and select **New 🡪 Class**
4. Enter the name as **AboutAppActivity**
5. Enter the **Superclass** as **android.app.Activity**
6. Click on **Finish.**

**Step 4 – Code the Main Activity**

1. First we will add some strings to the resources which will be references in the activities. Go to **res/values** folder and modify **strings.xml** to have the following **additional** string elements.

<string name="viewsubjects\_label">Subjects</string>

<string name="aboutapp\_label">About App</string>

1. Define the Layout for **MainActivity**. This will consist of a two buttons which will launch the second activity and third activity i.e. **ViewSubjectsActivity and AboutAppActivity**

Go to **res/layout** and open the **activity\_main.xml**. Enter the following content in the **activity\_main.xml** file (You can simply copy this):

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

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

android:orientation=*"vertical"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

>

<ImageView

android:id=*"@+id/imageView1"*

android:layout\_width=*"wrap\_content"*

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

android:src=*"@drawable/ic\_launcher"* android:layout\_gravity=*"center\_horizontal"* />

<Button android:layout\_height=*"wrap\_content"* android:id=*"@+id/btnViewSubjects"* android:text=*"@string/viewsubjects\_label"* android:layout\_width=*"match\_parent"*></Button>

<Button android:layout\_height=*"wrap\_content"* android:id=*"@+id/btnAboutApp"* android:text=*"@string/aboutapp\_label"* android:layout\_width=*"match\_parent"*></Button>

</LinearLayout>

1. Go to **src** 🡪 **com.mindstorm.multipleactivities** and open **MainActivity.java** file
2. Modify the **onCreate()** method as show below:

**public void onCreate(Bundle savedInstanceState) {**

**super.onCreate(savedInstanceState);**

**setContentView(R.layout.activity\_main);**

**Button btnViewSubjects = (Button) findViewById(R.id.btnViewSubjects);**

**btnViewSubjects.setOnClickListener(new OnClickListener() {**

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

**Intent i = new Intent(getBaseContext(),ViewSubjectsActivity.class);**

**i.putExtra("data1", "Android JellyBean");**

**i.putExtra("data2", 10);**

**startActivity(i);**

**}**

**});**

**Button btnAboutApp = (Button) findViewById(R.id.btnAboutApp);**

**btnAboutApp.setOnClickListener(new OnClickListener() {**

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

**Intent i = new Intent(getBaseContext(),AboutAppActivity.class);**

**//Data to pass to the activity**

**i.putExtra("data1", "Android IceCream Sandwich");**

**i.putExtra("data2", 20);**

**startActivity(i);**

**}**

**});**

**}**

**Step 4 – Code the ViewSubjectsActivity**

1. Define the Layout for the **ViewSubjectsActivity**. Go to Right-click on **res/layout** , select **New 🡪 File** and provide the name as **viewsubjects.xml**. Enter the following content in the **viewsubjects.xml** file:

<?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" >

<TextView

android:id="@+id/textView1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="This is View Subjects Activity"

android:textAppearance="?android:attr/textAppearanceLarge" />

<TextView

android:id="@+id/txtDataPassed"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Medium Text"

android:textAppearance="?android:attr/textAppearanceMedium" />

</LinearLayout>

1. Go to **src** 🡪 **com.mindstorm.multipleactivities** and open **Activity2.java** file
2. Modify the source code as shown below for **Activity2.java**:

**package com.mindstorm.multipleactivities;**

**import android.app.Activity;**

**import android.os.Bundle;**

**import android.widget.TextView;**

**public class ViewSubjectsActivity extends Activity {**

**/\*\* Called when the activity is first created. \*/**

**@Override**

**public void onCreate(Bundle savedInstanceState) {**

**super.onCreate(savedInstanceState);**

**setContentView(R.layout.viewsubjects);**

**//Read the data passed**

**String data1 = getIntent().getExtras().getString("data1");**

**int data2 = getIntent().getExtras().getInt("data2");**

**//Set the text**

**TextView txtDataPassed = (TextView)findViewById(R.id.txtDataPassed);**

**txtDataPassed.setText(data1 + " and " + data2);**

**}**

**}**

**Step 5 – Code the AboutAppActivity**

1. Define the Layout for the **AboutAppActivity**. Go to Right-click on **res/layout** , select **New 🡪 File** and provide the name as **aboutapp.xml**. Enter the following content in the **aboutapp.xml** file:

<?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" >

<TextView

android:id="@+id/textView1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="This is About App Activity"

android:textAppearance="?android:attr/textAppearanceLarge" />

<TextView

android:id="@+id/txtDataPassed"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Medium Text"

android:textAppearance="?android:attr/textAppearanceMedium" />

</LinearLayout>

1. Go to **src** 🡪 **com.mindstorm.multipleactivities** and open **AboutAppActivity.java** file
2. Modify the source code as shown below for **AboutAppActivity.java**:

**package com.mindstorm.multipleactivities;**

**import android.app.Activity;**

**import android.os.Bundle;**

**import android.widget.TextView;**

**public class AboutAppActivity extends Activity {**

**/\*\* Called when the activity is first created. \*/**

**@Override**

**public void onCreate(Bundle savedInstanceState) {**

**super.onCreate(savedInstanceState);**

**setContentView(R.layout.aboutapp);**

**//Read the data passed**

**String data1 = getIntent().getExtras().getString("data1");**

**int data2 = getIntent().getExtras().getInt("data2");**

**//Set the text**

**TextView txtDataPassed = (TextView)findViewById(R.id.txtDataPassed);**

**txtDataPassed.setText(data1 + " and " + data2);**

**}**

**}**

1. **Step 6 – Update the Manifest**

We need to ensure that both Activity1 and Activity2 are defined in the AndroidManifest.xml file. Go to AndroidManifest.xml and add the following the following <activity> elements inside of the <application> tag as shown below.

<activity

android:name=*"com.mindstorm.multipleactivities.MainActivity"*

android:label=*"@string/app\_name"* >

<intent-filter>

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

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

</intent-filter>

</activity>

<activity

android:name=*"com.mindstorm.multipleactivities.ViewSubjectsActivity"*

android:label=*"@string/app\_name"* >

</activity>

<activity

android:name=*"com.mindstorm.multipleactivities.AboutAppActivity"*

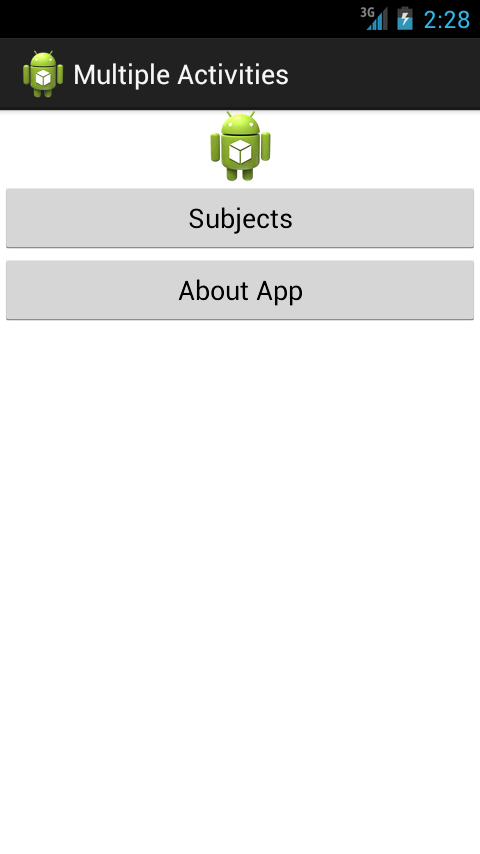
android:label=*"@string/app\_name"* >

</activity>

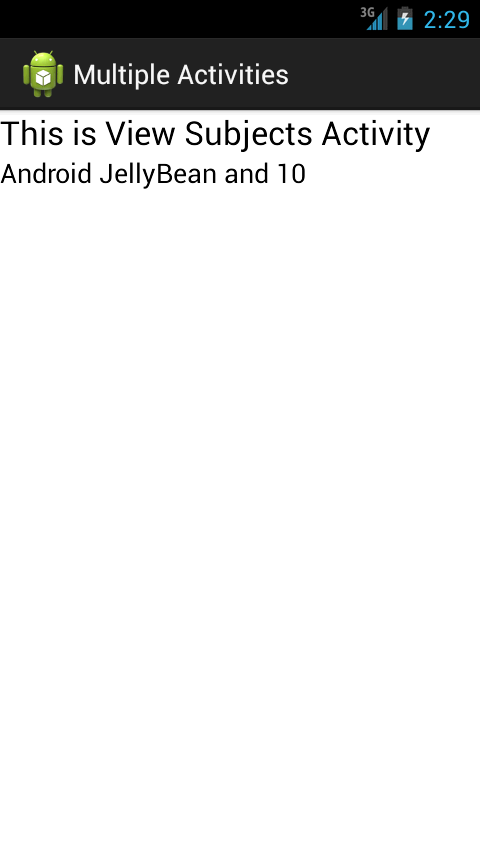
**Step 6 – Run the Example**

1. **Right Click** the **Project** in Eclipse.
2. Select **Run As 🡪 Android** Application
3. (Optional): If you have multiple compatible AVDs running, select the correct AVD. In our case it is the 4.x AVD.

You should see the **MainActivity** screen come up as shown below:



On clicking the Subjects button, you should see the second activity screen come up as shown below:



Similarly, when you click on the About App button, the About Activity screen will come up.

**Summary**

This hands-on exercise demonstrated how you can have multiple activities (screens) in your Android application. The pattern is simple. Define the Activity class and its layout. Ensure that the Activities are defined in the **AndroidManifest.xml** file. And launch the activity by explicitly providing the Activity class in the Intent and starting it with **startActivity** method.