CIS 470
Mobile App Development

Lecture 3

Wenbing Zhao
Department of Electrical Engineering and Computer Science
Cleveland State University
wenbing@ieee.org

Activities, Fragments, and Intents

- The life cycles of an activity
- Using fragments to customize your UI
- Understanding the concept of intents
- Displaying alerts to the user using notifications

Activities

- Typically, applications have one or more activities
- The main purpose of an activity is to interact with the user
- An activity's life cycle: from the moment an activity appears on the screen to the moment it is hidden, it goes through a number of stages

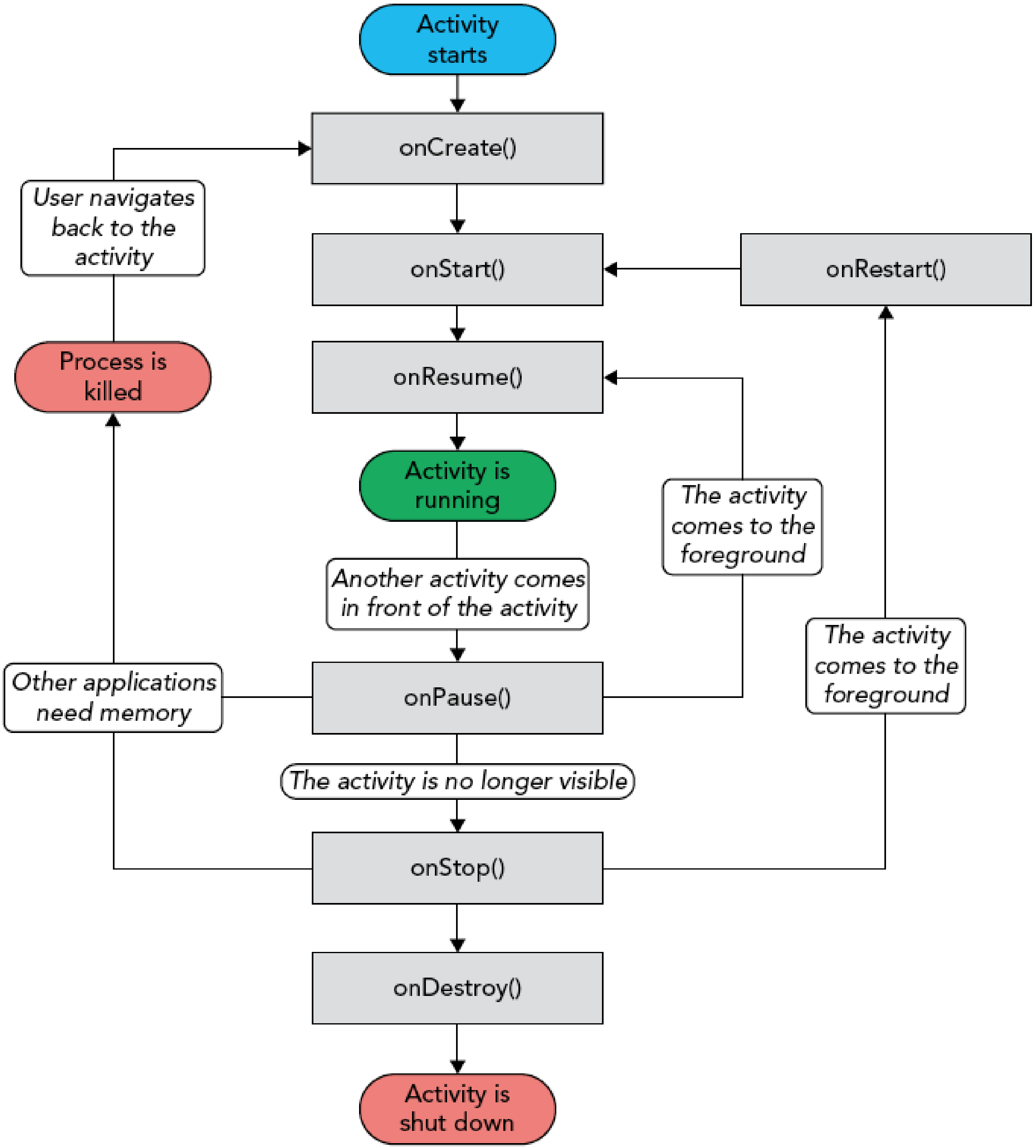
Fragments

- Fragment is a feature that was introduced for tablets in
- Android 3.0 and for phones in Android 4.0
- Think of fragments as “miniature” activities that can be grouped to form an activity

Intent

- An intent is the “glue” that enables activities from different applications to work together seamlessly, ensuring that tasks can be performed as though they all belong to one single application

Activity Life Cycle


Activity starts
onCreate()
User navigates back to the activity
onStart()
onRestart()
Process is killed
onResume()
Activity is running
The activity comes to the foreground
Another activity comes in front of the activity
The activity comes to the foreground
Other applications need memory
onPause()
The activity is no longer visible
onStop()
onDestroy()
Activity is shut down

Activity Life Cycle

- onCreate()—Called when the activity is first created
- onStart()—Called when the activity becomes visible to the user
- onResume()—Called when the activity starts interacting with the user
- onPause()—Called when the current activity is being paused and the previous activity is being resumed
- onStop()—Called when the activity is no longer visible to the user
- onDestroy()—Called before the activity is destroyed by the system (either manually or by the system to conserve memory)
- onRestart()—Called when the activity has been stopped and is restarting again

Observe Activity Life Cycle

- Using Android Studio, create a new Android project and name it Activity101
- In the Activity101Activity.java file, add the following highlighted statements
  - Throughout this example, be sure to change all references to "com.jfdimarzio" to whatever package name your project is using

```
package com.jfdimarzio.activity101;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
public class MainActivity extends AppCompatActivity
{
    String tag = "Lifecycle Step";
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Log.d(tag, "In the onCreate() event");
    }
   public void onStart()
   {
        super.onStart();
        Log.d(tag, "In the onStart() event");
    }
   public void onRestart()
   {
        super.onRestart();
        Log.d(tag, "In the onRestart() event");
    }
```

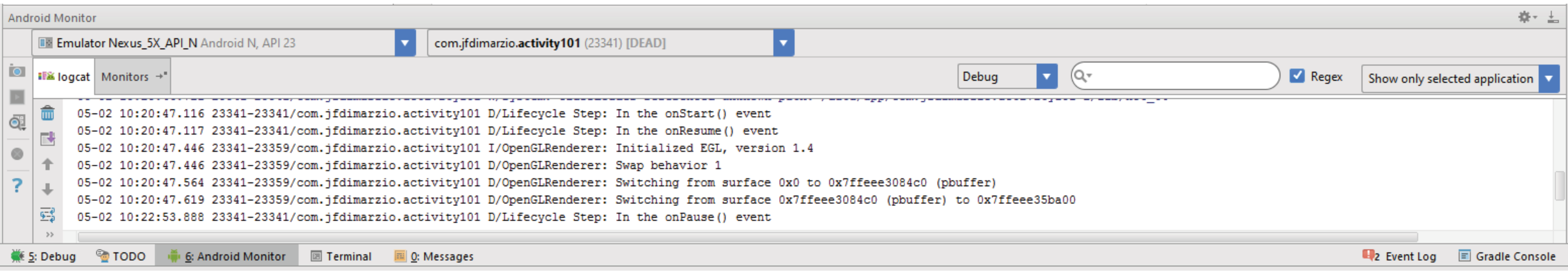
Observe Activity Life Cycle

- Run => Debug, or Shift + F9 to debug the app

```
    public void onResume()
    {
        super.onResume();
        Log.d(tag, "In the onResume() event");
    }
    public void onPause()
    {
        super.onPause();
        Log.d(tag, "In the onPause() event");
     }
    public void onStop()
    {
        super.onStop();
        Log.d(tag, "In the onStop() event");
     }
    public void onDestroy()
    {
         super.onDestroy();
         Log.d(tag, "In the onDestroy() event");
    }
}
```

Observe Activity Life Cycle

- When the activity is first loaded, you should see something very similar to the following in the logcat console



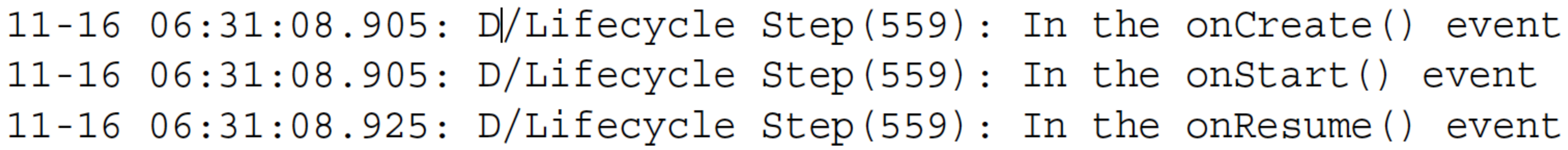
```
05-02 10:20:47.116 23341-23341/com.jfdimarzio.activity101 D/Lifecycle Step: In the onStart() event
05-02 10:20:47.117 23341-23341/com.jfdimarzio.activity101 D/Lifecycle Step: In the onResume() event
05-02 10:20:47.446 23341-23359/com.jfdimarzio.activity101 I/OpenGLRenderer: Initialized EGL, version 1.4
05-02 10:20:47.446 23341-23359/com.jfdimarzio.activity101 D/OpenGLRenderer: Swap behavior 1
05-02 10:20:47.564 23341-23359/com.jfdimarzio.activity101 D/OpenGLRenderer: Switching from surface 0x0 to 0x7ffeee3084c0 (pbuffer)
05-02 10:20:47.619 23341-23359/com.jfdimarzio.activity101 D/OpenGLRenderer: Switching from surface 0x7ffeee3084c0 (pbuffer) to 0x7ffeee35ba00
05-02 10:22:53.888 23341-23341/com.jfdimarzio.activity101 D/Lifecycle Step: In the onPause() event
```

- If you click the Back button on the Android emulator, you will see:

```
11-16 06:29:26.665: D/Lifecycle Step(559): In the onPause() event
11-16 06:29:28.465: D/Lifecycle Step(559): In the onStop() event
11-16 06:29:28.465: D/Lifecycle Step(559): In the onDestroy() event
```

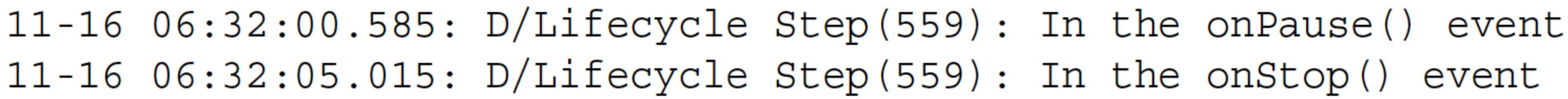
Observe Activity Life Cycle

- Click the Home button, click the Overview icon, select the Activity101 app, you will see:



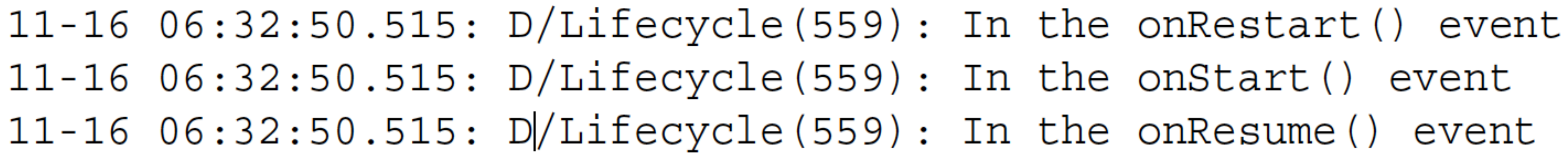
```
11-16 06:31:08.905: D/Lifecycle Step(559): In the onCreate() event
11-16 06:31:08.905: D/Lifecycle Step(559): In the onStart() event
11-16 06:31:08.925: D/Lifecycle Step(559): In the onResume() event
```

- Click the Home button and then click the Phone button on the Android emulator so that the activity is pushed to the background



```
11-16 06:32:00.585: D/Lifecycle Step(559): In the onPause() event
11-16 06:32:05.015: D/Lifecycle Step(559): In the onStop() event
```

- Exit the phone dialer by clicking the Back button, the activity is now visible again:



```
11-16 06:32:50.515: D/Lifecycle(559): In the onRestart() event
11-16 06:32:50.515: D/Lifecycle(559): In the onStart() event
11-16 06:32:50.515: D/Lifecycle(559): In the onResume() event
```

Observe Activity Life Cycle: Summary

- Use the onCreate() method to create and instantiate the objects that you will be using in your application
- Use the onResume() method to start any services or code that needs to run while your activity is in the foreground
- Use the onPause() method to stop any services or code that does not need to run when your activity is not in the foreground
- Use the onDestroy() method to free up resources before your activity is destroyed

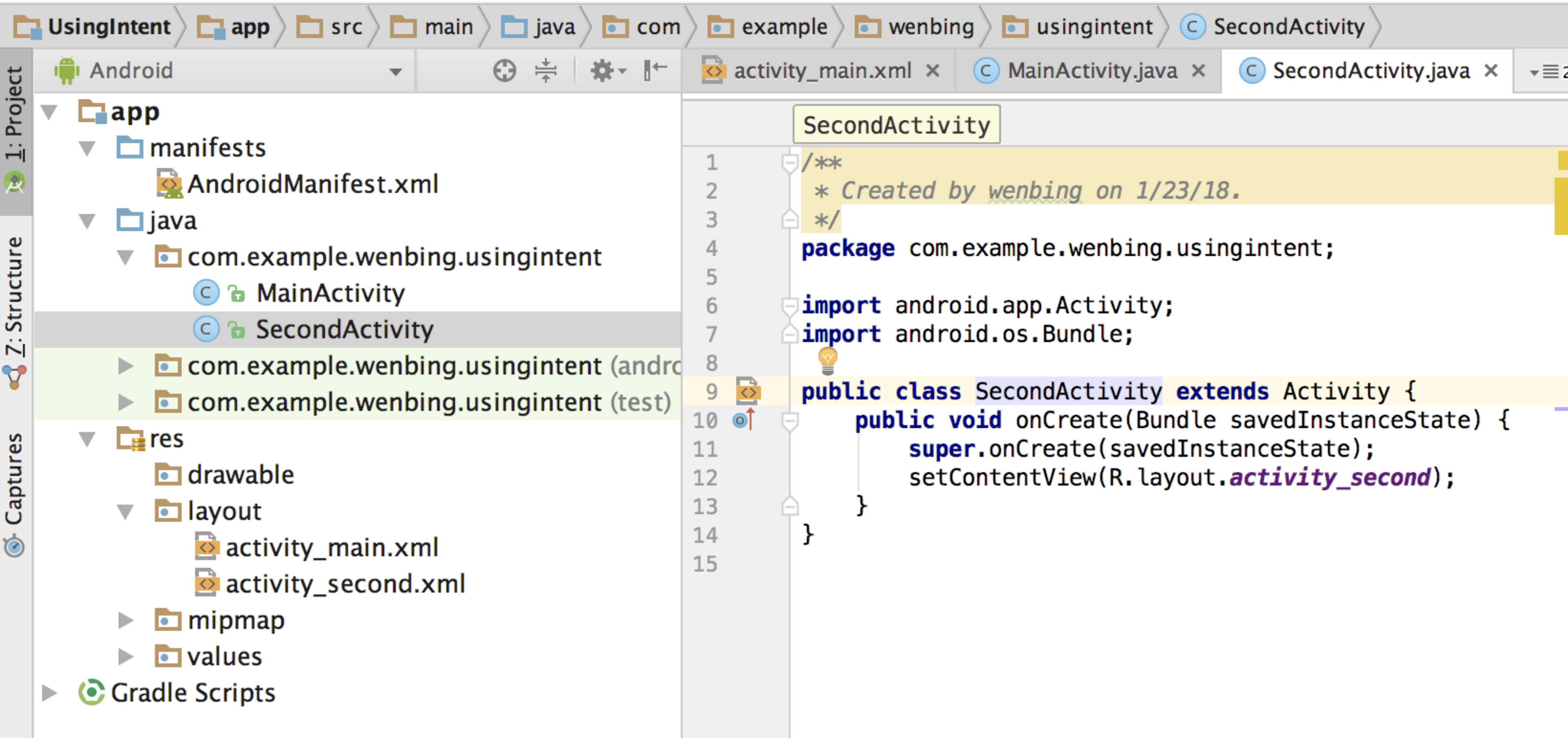
Intents

- When your application has more than one activity, you often need to navigate from one to another. In Android, you navigate between activities through what is known as an intent

Linking Activities with Intents: Example

- Using Android Studio, create a new Android project with an empty Activity named MainActivity; name the project **UsingIntent**
- Right-click your package name under the java folder in the Project Files windows and select New ➪ Java Class
- Name the new class SecondActivity and click OK
- Add the bolded statements from the following code to the AndroidManifest.xml file

Linking Activities with Intents: Example



```java
/**
 * Created by wenbing on 1/23/18.
 */
package com.example.wenbing.usingintent;

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

public class SecondActivity extends Activity {
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
    }
}
```

Linking Activities with Intents: Example

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.jfdimarzio.usingintent">
    <application
        android:allowBackup="true”
        ...
        <activity android:name=".MainActivity">
            .....
        </activity>
        <activity android:name=".SecondActivity" >
            <intent-filter >
                <action android:name="com.username.usingintent.SecondActivity" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
         </activity>
      </application>
</manifest>
```

Linking Activities with Intents: Example

- Make a copy of the activity_main.xml file (in the res/layout folder) by right-clicking it and selecting Copy. Then right-click the res/layout folder and select Paste. Name the file **activity_second.xml**
- Modify the activity_second.xml file as follows:

```
<?xml version="1.0" encoding="utf-8"?>
.....
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.jfdimarzio.usingintent.SecondActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="This is the Second Activity!" />
</RelativeLayout>
```

Linking Activities with Intents: Example

- In the SecondActivity.java file, add the bolded statements from the following code:

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

public class SecondActivity extends Activity {
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
    }
}
```

Linking Activities with Intents: Example

- Add the bolded lines in the following code to the activity_main.xml file:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android=http://schemas.android.com/apk/res/android
    .....
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Main Activity!"
        android:id="@+id/textView" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Display second activity"
        android:onClick="onClick"
        android:id="@+id/button"
        android:layout_below="@+id/textView"
        android:layout_alignParentStart="true"
        android:layout_marginTop="56dp" />
</RelativeLayout>
```

Linking Activities with Intents: Example

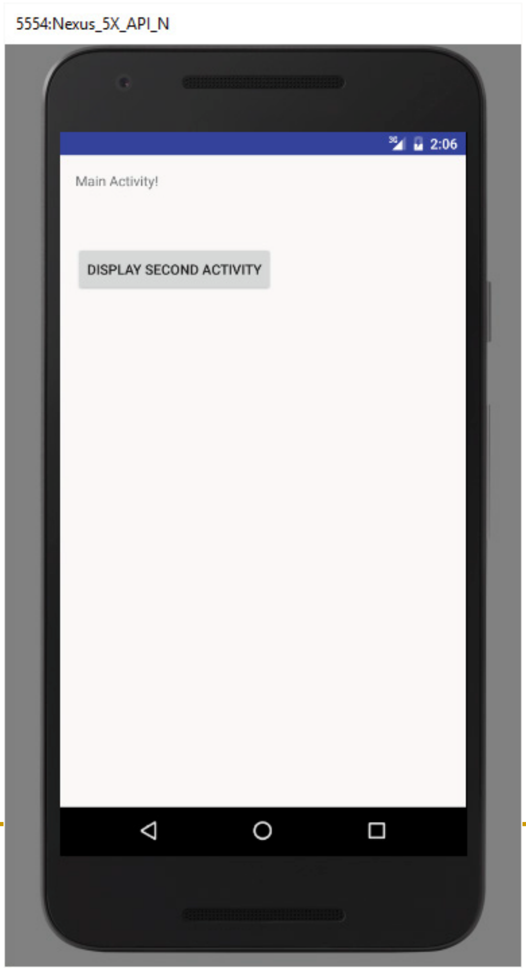
- Modify the MainActivity.java file as shown in the bolded lines in the following code:

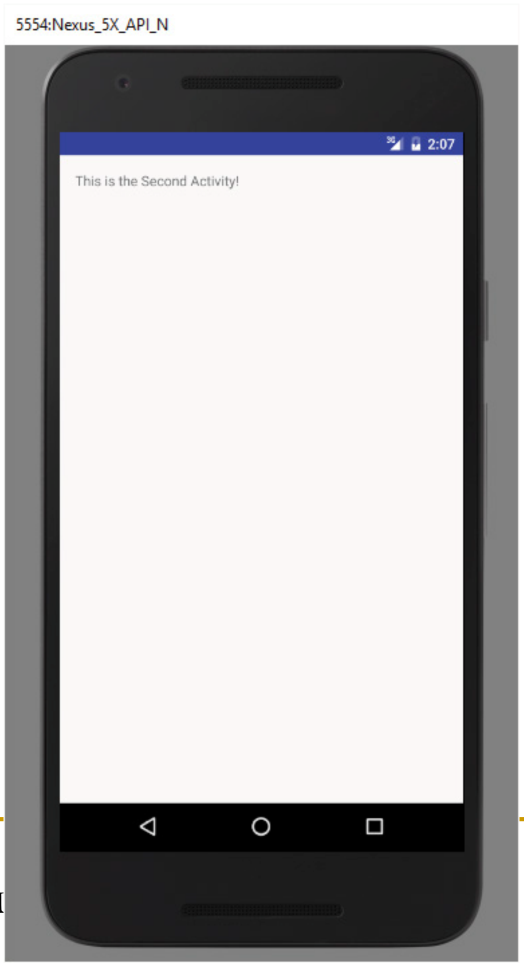
```
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.View;

public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
     }
     public void onClick(View view) {
          startActivity(new Intent("com.username.usingintent.SecondActivity"));
     }
}
```

Linking Activities with Intents: Example

- Press Shift+F9 to debug the application on the Android emulator
- When the first activity is loaded, click the button and the second activity also loads


5554:Nexus_5X_API_N


5554:Nexus_5X_API_N

Challenge Task#1

- Add a UI control on the screen of the second activity so that you can go back to the first activity (i.e., the main activity). In addition, on the main activity, display an iteration count on the number of times the main activity is displayed.