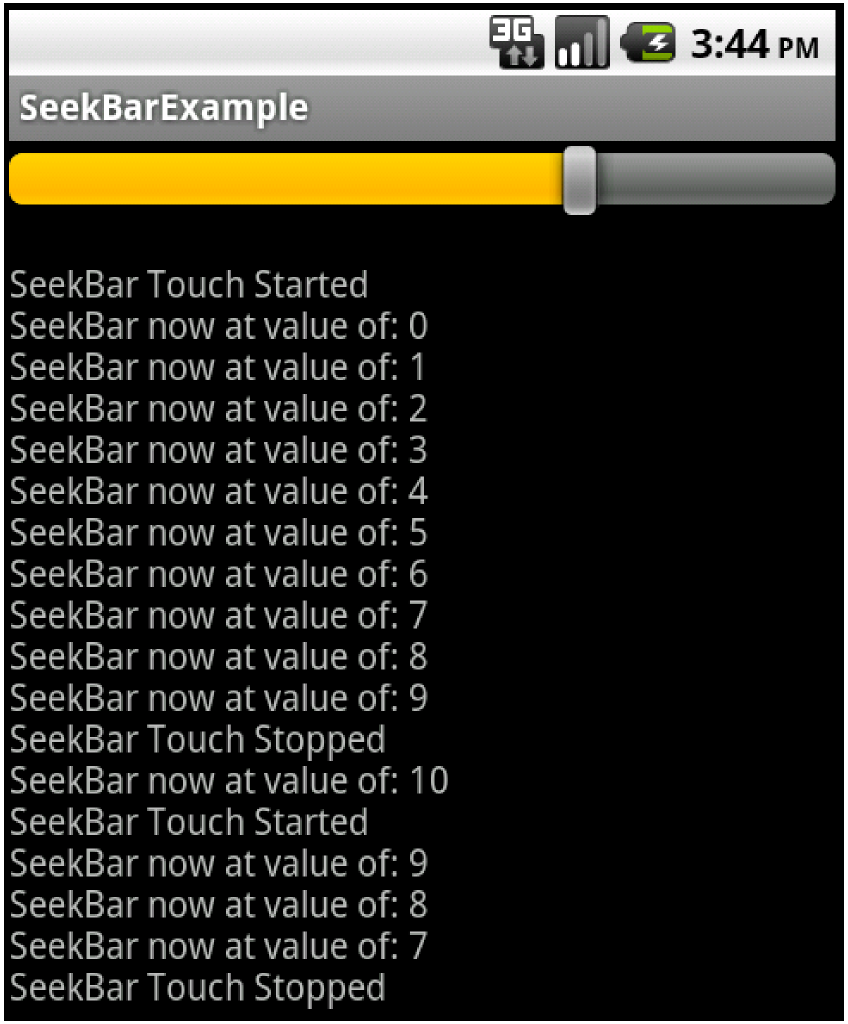
Android Programming
Lecture 6

9/21/2011

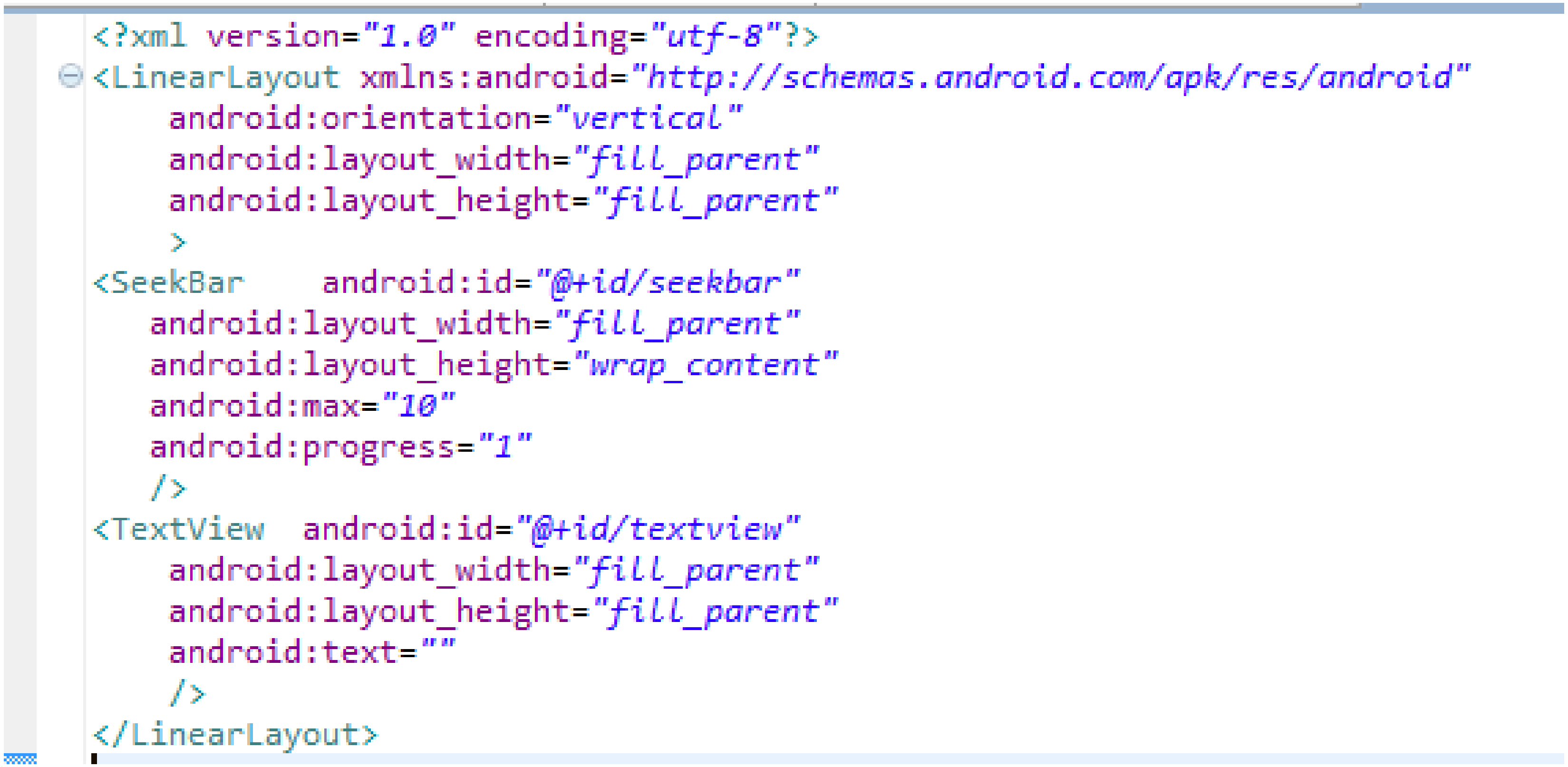
SeekBar

- SeekBar (slider) allow selection of *integer values* using a natural interface
- Constraints:
  - Min: 0
  - Max: settable
  - Changes by: 1
  - Starting point for knob can be set
- Senses initiation of touch ,ending of touch, and movement



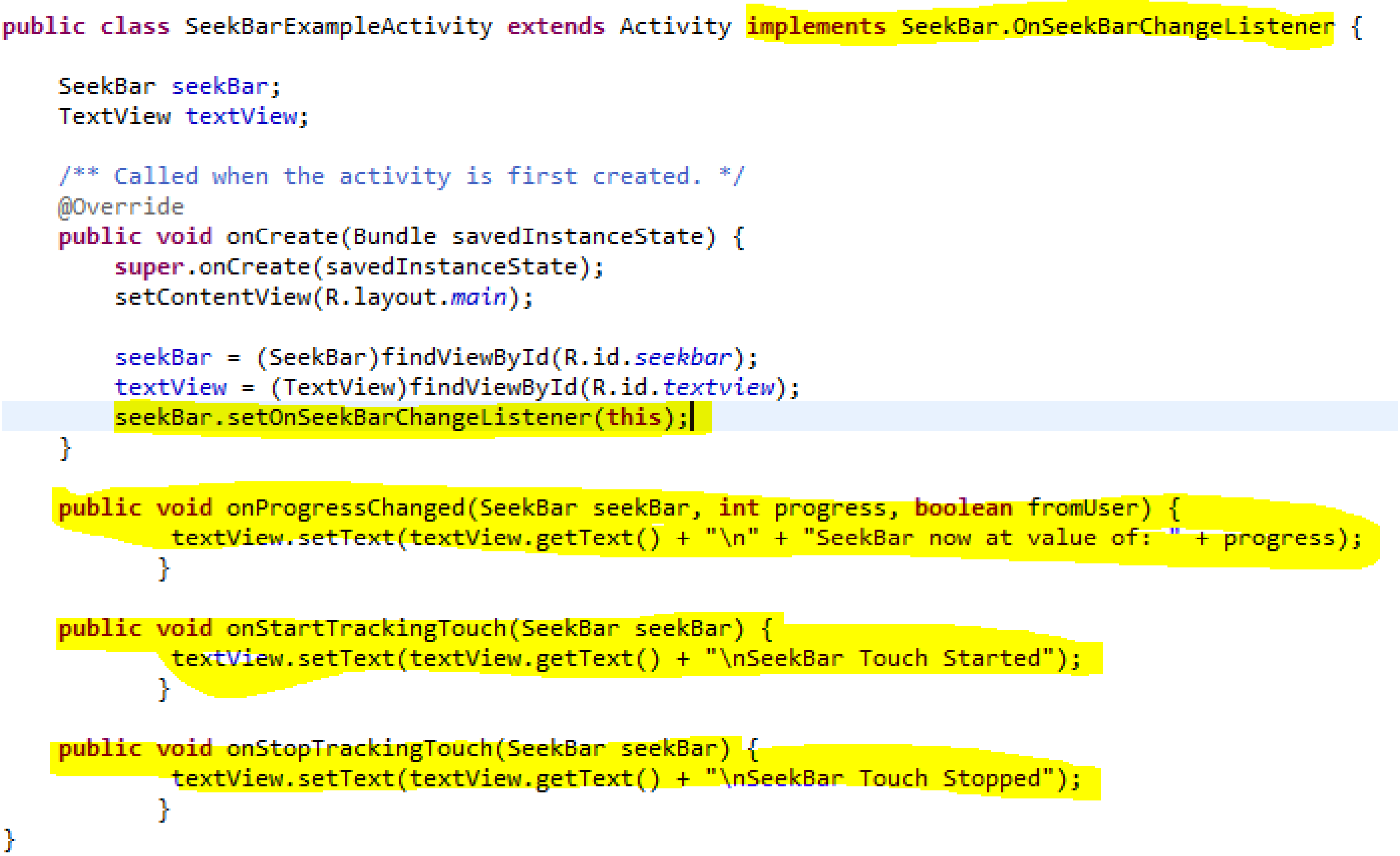
```
SeekBar Touch Started
SeekBar now at value of: 0
SeekBar now at value of: 1
SeekBar now at value of: 2
SeekBar now at value of: 3
SeekBar now at value of: 4
SeekBar now at value of: 5
SeekBar now at value of: 6
SeekBar now at value of: 7
SeekBar now at value of: 8
SeekBar now at value of: 9
SeekBar Touch Stopped
SeekBar now at value of: 10
SeekBar Touch Started
SeekBar now at value of: 9
SeekBar now at value of: 8
SeekBar now at value of: 7
SeekBar Touch Stopped
```

SeekBar



```
<?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"
    >
<SeekBar    android:id="@+id/seekbar"
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:max="10"
   android:progress="1"
   />
<TextView  android:id="@+id/textview"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:text=""
    />
</LinearLayout>
```

SeekBar



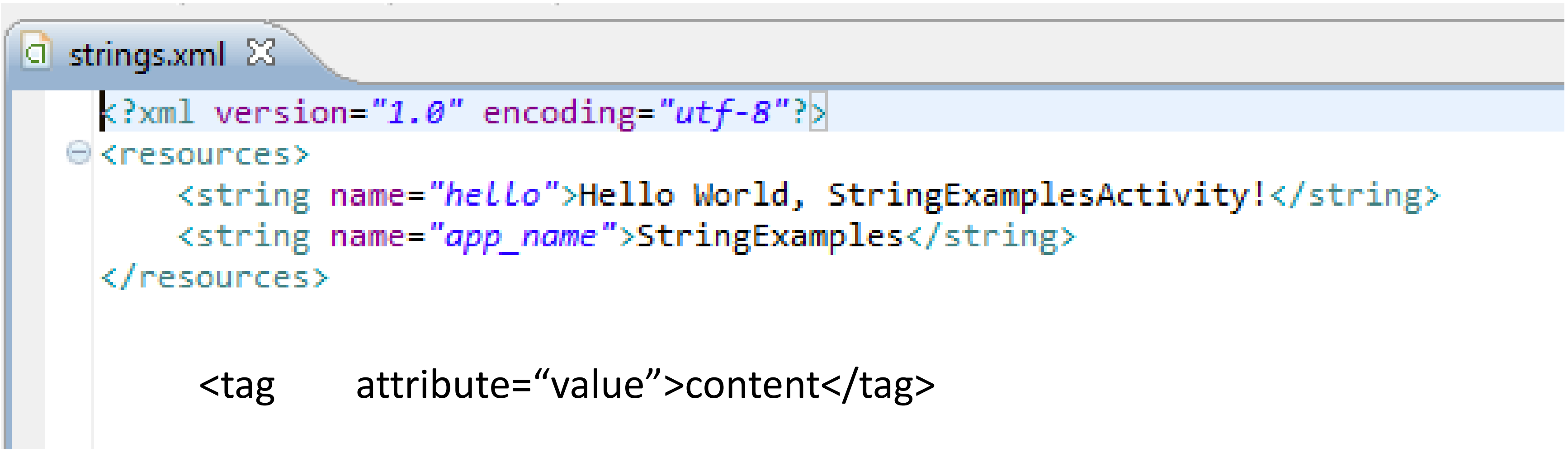
```
public class SeekBarExampleActivity extends Activity implements SeekBar.OnSeekBarChangeListener {

    SeekBar seekBar;
    TextView textView;

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        seekBar = (SeekBar)findViewById(R.id.seekbar);
        textView = (TextView)findViewById(R.id.textview);
        seekBar.setOnSeekBarChangeListener(this);
    }

    public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
            textView.setText(textView.getText() + "\n" + "SeekBar now at value of: " + progress);
           }

    public void onStartTrackingTouch(SeekBar seekBar) {
            textView.setText(textView.getText() + "\nSeekBar Touch Started");
           }

    public void onStopTrackingTouch(SeekBar seekBar) {
            textView.setText(textView.getText() + "\nSeekBar Touch Stopped");
           }
}
```

Strings in XML

- It is possible to provide mappings between names and strings for your app in XML
  - Referencing name will replace with actual string
  - Allows for:
    - Single find-and-replace of Strings
    - Simple customization
    - Internationalization
- Loaded in file: *res/values/strings.xml*

Strings in XML

- By default, Eclipse will generate a few string resource entries for you



```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, StringExamplesActivity!</string>
    <string name="app_name">StringExamples</string>
</resources>
```

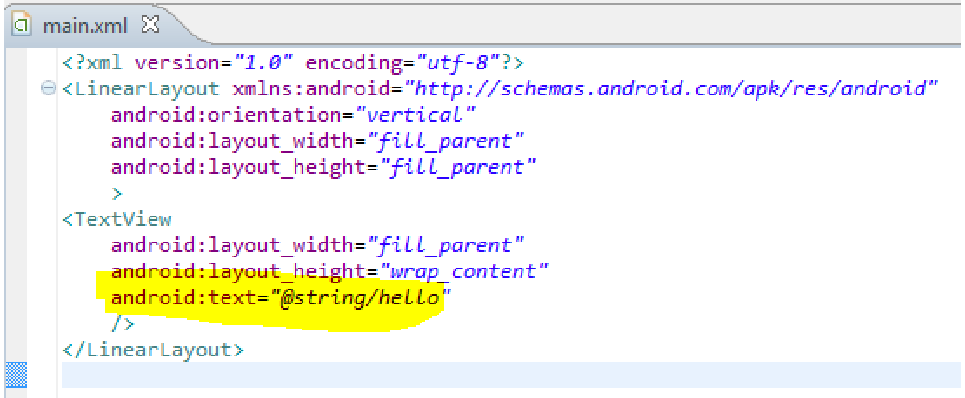
<tag   attribute=“value”>content</tag>

- Essentially equivalent to

```
String app_name =  “StringExamples”;
```

Strings in XML

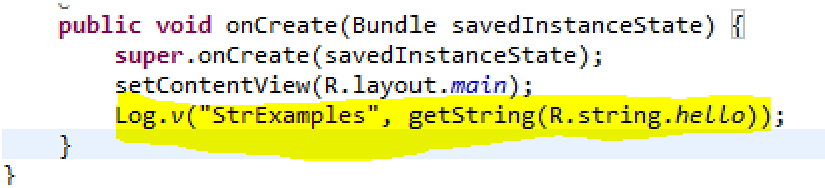
- To use in other XML files:



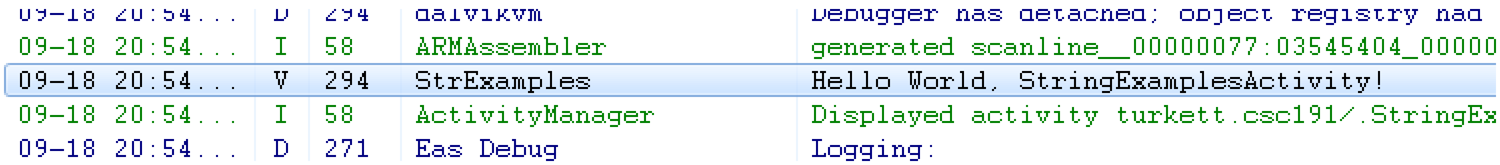
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
</LinearLayout>
```

@ refers
to a resource definition
(already saw with ids)

- To use in code: getString(R.string.*name*)



```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    Log.v("StrExamples", getString(R.string.hello));
  }
}
```



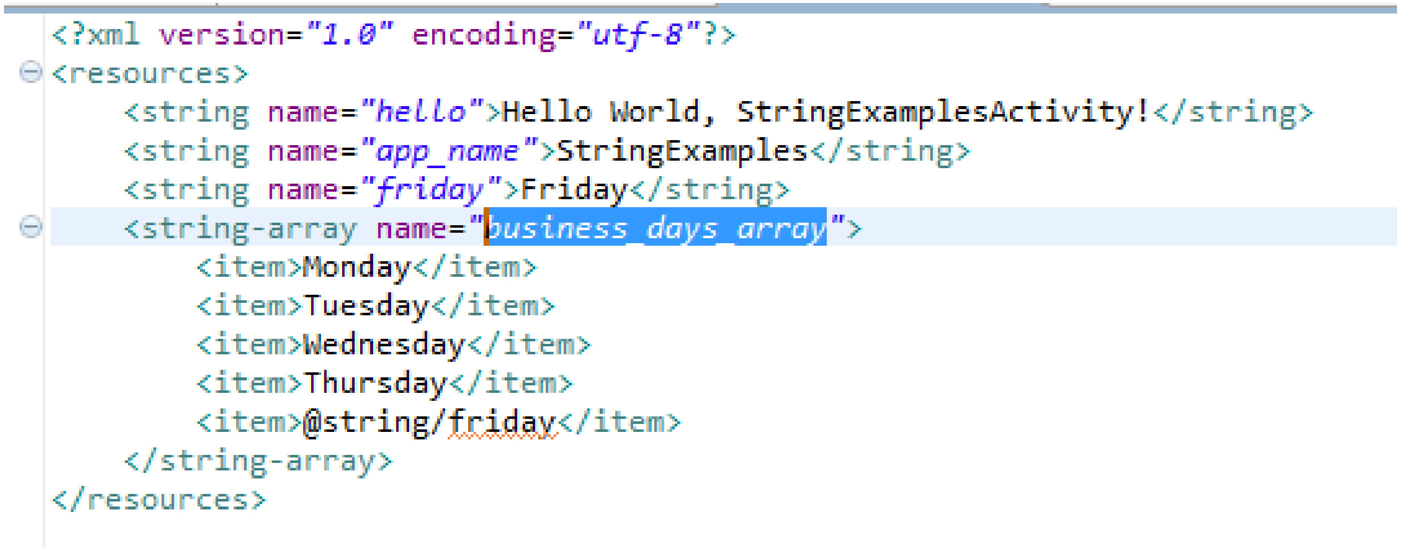
```
09-18 20:54...  D  294  dalvikvm         Debugger has detached; object registry had
09-18 20:54...  I  58   ARMAssembler     generated scanline__00000077:03545404_00000
09-18 20:54...  V  294  StrExamples      Hello World, StringExamplesActivity!
09-18 20:54...  I  58   ActivityManager  Displayed activity turkett.csc191/.StringEx
09-18 20:54...  D  271  Eas Debug        Logging:
```

Arrays in XML

- Two main types of arrays can be specified in XML:
  - String arrays
  - Integer arrays

String Arrays in XML

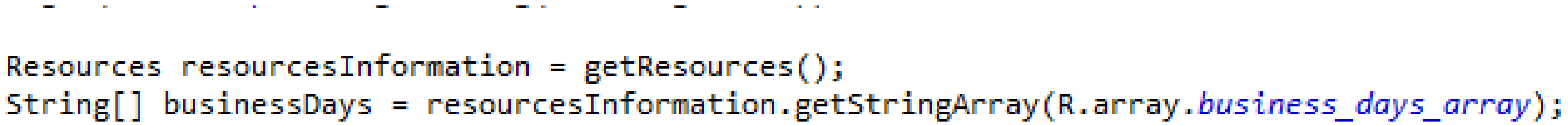
- Arrays of Strings can be loaded in the same strings XML file
- An array entry can be a reference to another pre-defined String (see Friday example)



```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, StringExamplesActivity!</string>
    <string name="app_name">StringExamples</string>
    <string name="friday">Friday</string>
    <string-array name="business_days_array">
        <item>Monday</item>
        <item>Tuesday</item>
        <item>Wednesday</item>
        <item>Thursday</item>
        <item>@string/friday</item>
    </string-array>
</resources>
```

String Arrays in XML

- To use in code: getStringArray(R.array.*name*)
  - Request reference to app resources via getResources()
  - Request the appropriately named array resource

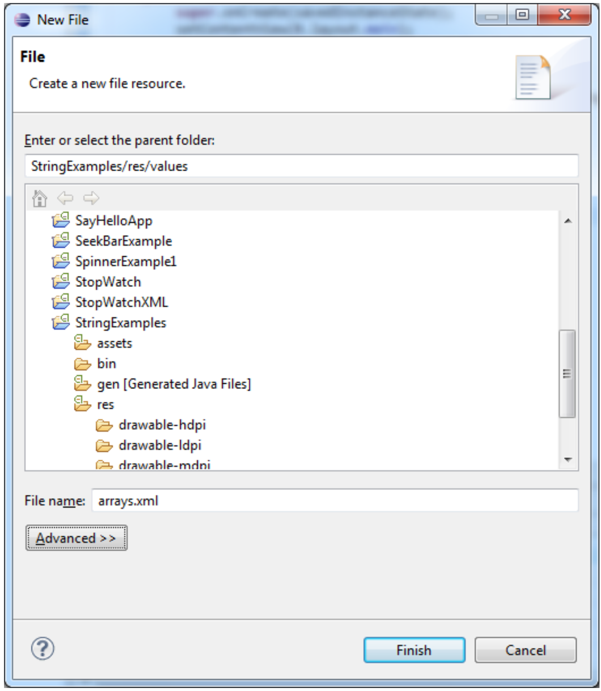


```
Resources resourcesInformation = getResources();
String[] businessDays = resourcesInformation.getStringArray(R.array.business_days_array);
```

Integer Arrays in XML

- Create a new resource values file with a name of your choosing
  - arrays.xml is reasonable (name doesn’t matter)
- Loaded in file: *res/values/chosennamehere.xml*
- To create new file
  - Right-click on values folder in project and choose “New File”

  - Provide filename in dialog box that pops up



File

Create a new file resource.

Enter or select the parent folder:

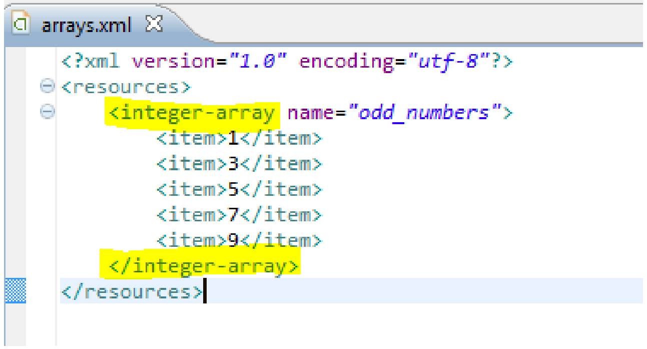
StringExamples/res/values

SayHelloApp
SeekBarExample
SpinnerExample1
StopWatch
StopWatchXML
StringExamples
assets
bin
gen [Generated Java Files]
res
drawable-hdpi
drawable-ldpi
drawable-mdpi

File name: arrays.xml

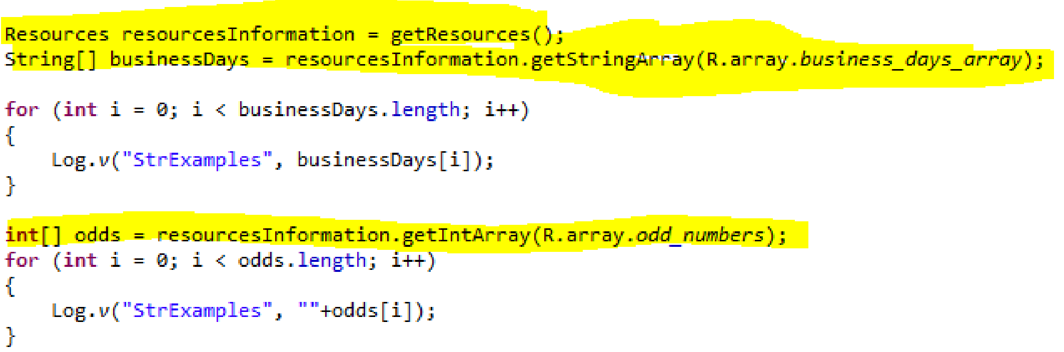
Integer Arrays in XML

- An integer array looks a lot like a string array, except small XML differences:

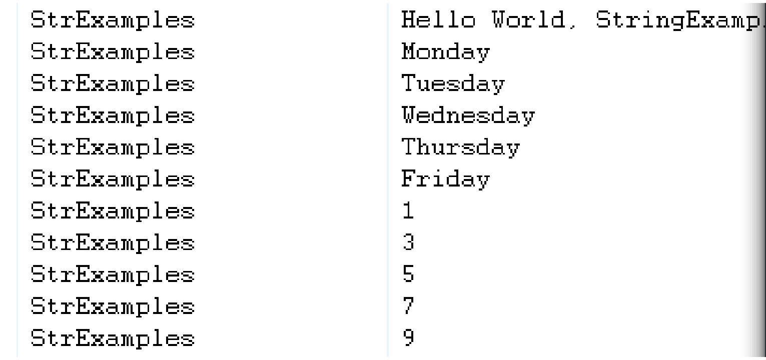


```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <integer-array name="odd_numbers">
        <item>1</item>
        <item>3</item>
        <item>5</item>
        <item>7</item>
        <item>9</item>
    </integer-array>
</resources>
```

- Accessed in code similarly:



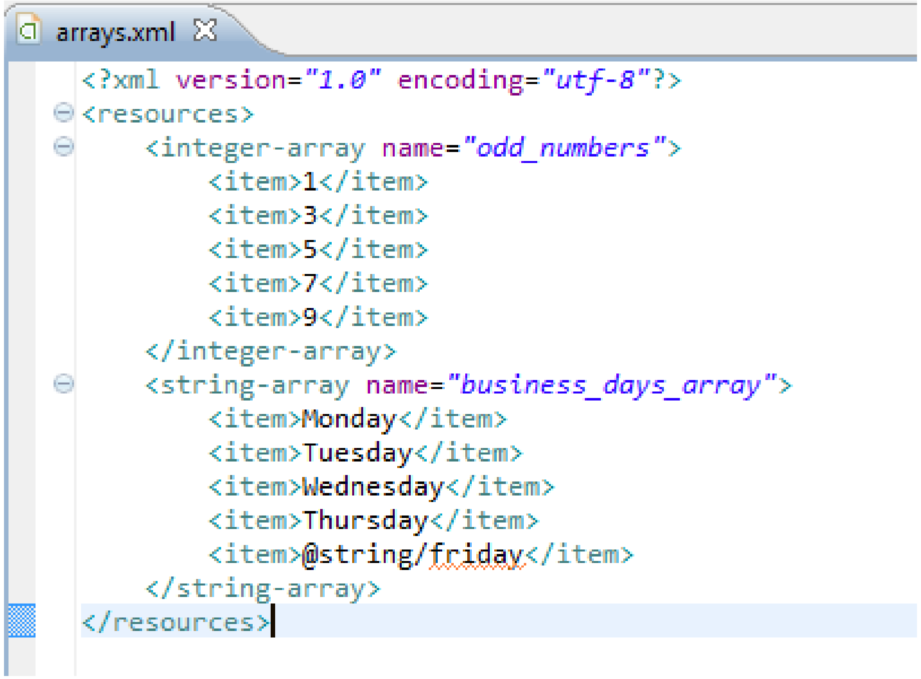
```
Resources resourcesInformation = getResources();
String[] businessDays = resourcesInformation.getStringArray(R.array.business_days_array);

for (int i = 0; i < businessDays.length; i++)
{
    Log.v("StrExamples", businessDays[i]);
}

int[] odds = resourcesInformation.getIntArray(R.array.odd_numbers);
for (int i = 0; i < odds.length; i++)
{
    Log.v("StrExamples", ""+odds[i]);
}
```



```
                            StrExamples      Hello World, StringExamp
                            StrExamples      Monday
                            StrExamples      Tuesday
                            StrExamples      Wednesday
                            StrExamples      Thursday
                            StrExamples      Friday
                            StrExamples      1
                            StrExamples      3
                            StrExamples      5
09-18 21:39...  V  353     StrExamples      7
09-18 21:39...  V  353     StrExamples      9
```

Integer Arrays in XML

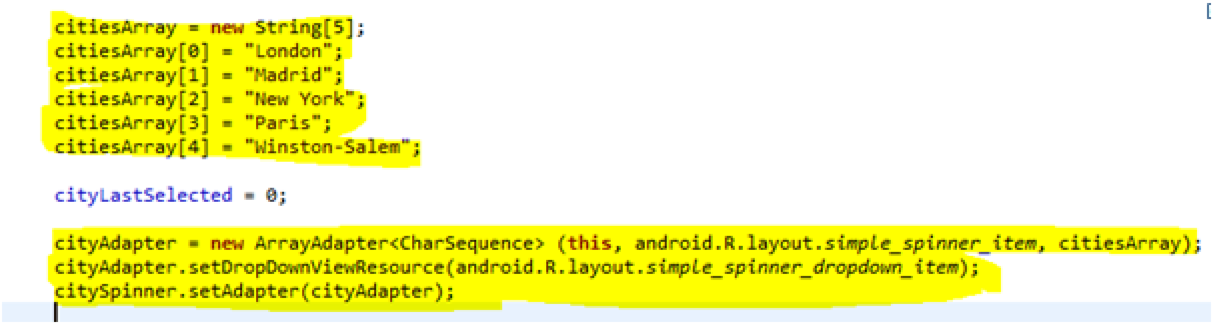
- Technically, string-array could have been defined in this arrays.xml file
  - It just has to be defined in a file in the *res/values* folder – filenames for array resources are irrelevant



```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <integer-array name="odd_numbers">
        <item>1</item>
        <item>3</item>
        <item>5</item>
        <item>7</item>
        <item>9</item>
    </integer-array>
    <string-array name="business_days_array">
        <item>Monday</item>
        <item>Tuesday</item>
        <item>Wednesday</item>
        <item>Thursday</item>
        <item>@string/friday</item>
    </string-array>
</resources>
```

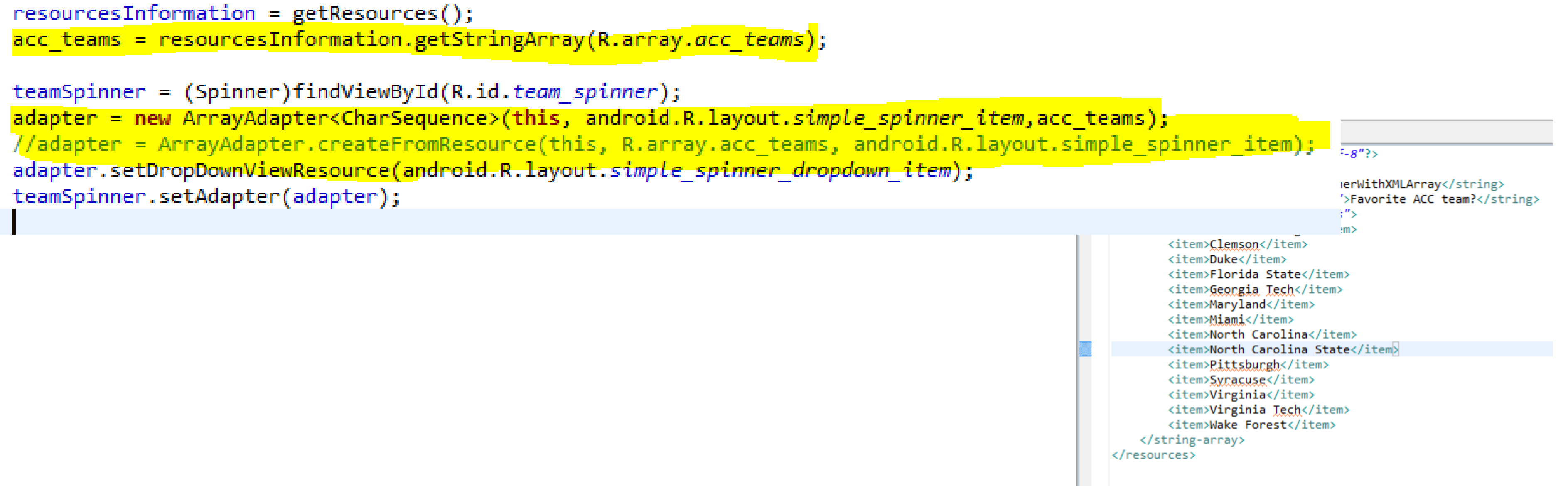
Using XML Arrays with Array Adapter

Previously:



```
citiesArray = new String[5];
citiesArray[0] = "London";
citiesArray[1] = "Madrid";
citiesArray[2] = "New York";
citiesArray[3] = "Paris";
citiesArray[4] = "Winston-Salem";

cityLastSelected = 0;

cityAdapter = new ArrayAdapter<CharSequence> (this, android.R.layout.simple_spinner_item, citiesArray);
cityAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
citySpinner.setAdapter(cityAdapter);
```

Now:



```
resourcesInformation = getResources();
acc_teams = resourcesInformation.getStringArray(R.array.acc_teams);

teamSpinner = (Spinner)findViewById(R.id.team_spinner);
adapter = new ArrayAdapter<CharSequence>(this, android.R.layout.simple_spinner_item,acc_teams);
//adapter = ArrayAdapter.createFromResource(this, R.array.acc_teams, android.R.layout.simple_spinner_item);
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
teamSpinner.setAdapter(adapter);
```

```
-8"?>

erWithXMLArray</string>
'>Favorite ACC team?</string>
;">
m>
        <item>Clemson</item>
        <item>Duke</item>
        <item>Florida State</item>
        <item>Georgia Tech</item>
        <item>Maryland</item>
        <item>Miami</item>
        <item>North Carolina</item>
        <item>North Carolina State</item>
        <item>Pittsburgh</item>
        <item>Syracuse</item>
        <item>Virginia</item>
        <item>Virginia Tech</item>
        <item>Wake Forest</item>
    </string-array>
</resources>
```

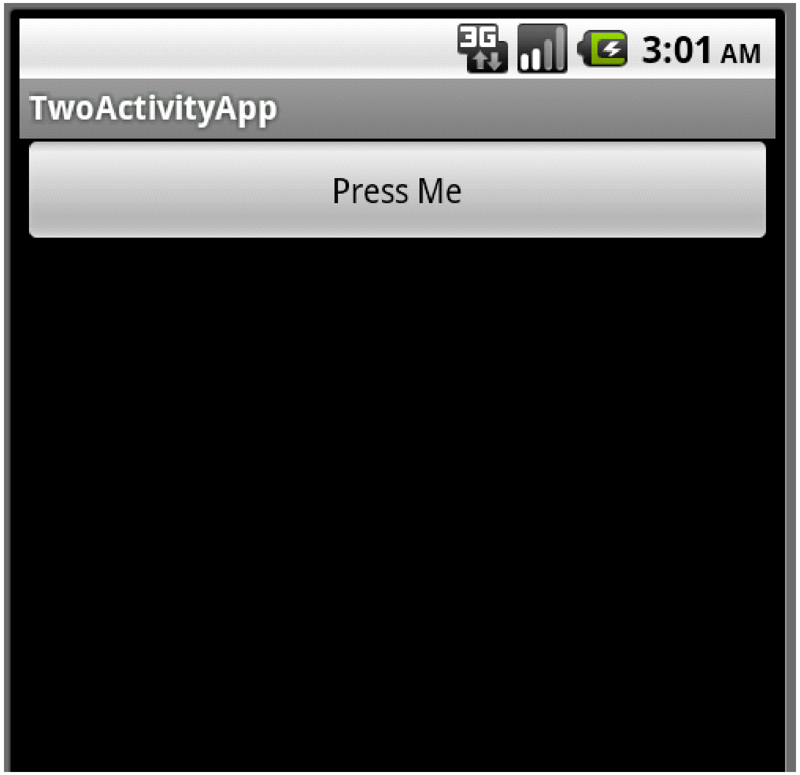
Multiple Activities

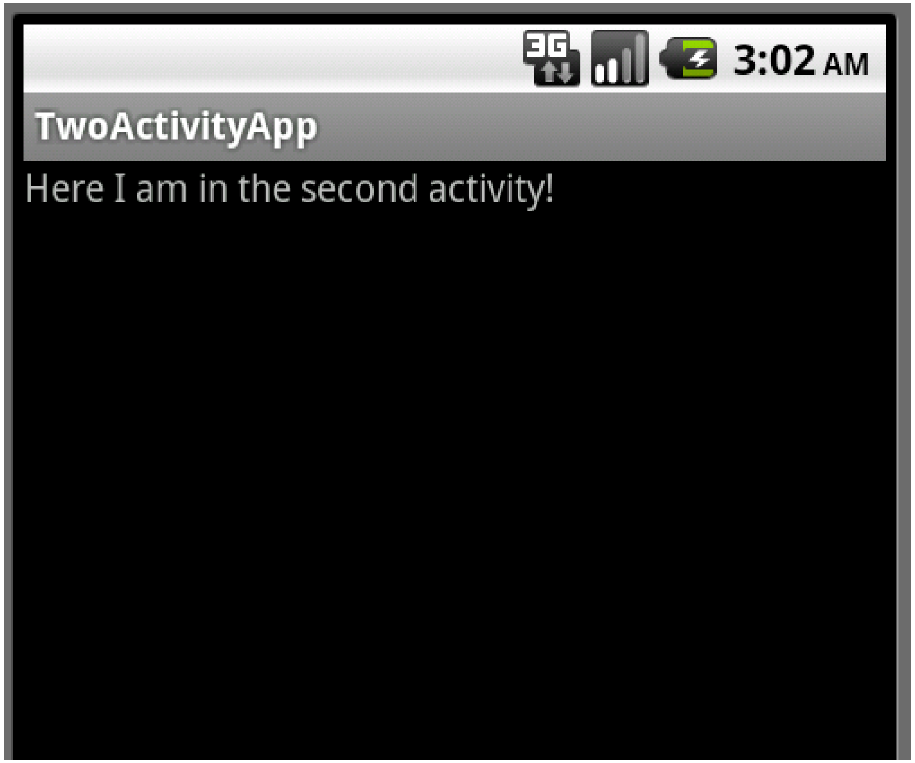
- So far, projects limited to one Activity

- Next step:
  - Intra-application communication
    - Having multiple activities within own application
  - Inter-application communication
    - Exploiting capabilities of other interactions

Multiple Activities: Code By Example

- First goal:
  - Press button in one Activity
  - Leads to opening of a second Activity





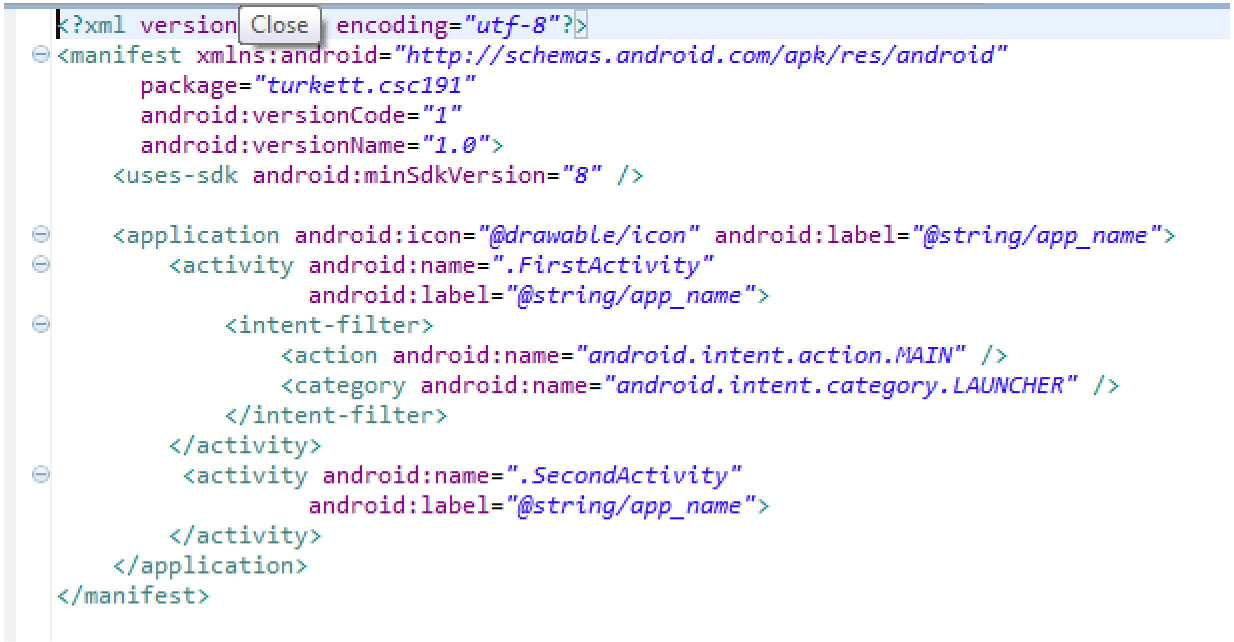
Here I am in the second activity!

Multiple Activities

- All Activities within an application need to be specified in application AndroidManifest.xml
  - – Only one is listed as the main launch Activity

First Activity
(launched)

Second Activity



```
<?xml version        encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="turkett.csc191"
      android:versionCode="1"
      android:versionName="1.0">
    <uses-sdk android:minSdkVersion="8" />

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".FirstActivity"
                  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=".SecondActivity"
                  android:label="@string/app_name">
        </activity>
    </application>
</manifest>
```

Intents

- Fundamental Android intra- and inter-application communication
- An abstract description of an activity to be performed
- Can be:
  - Directed to a specific component to be handled/performed
  - Broadcast on the device, triggering a response and handling from an appropriate component

Intents

Simplest intents are used to just trigger a specific known other Activity:

Have no specific description except target class

```
Intent intent = new Intent(MyActivity.this,
   MyOtherActivity.class);
startActivity(intent);
```

Intent parameters:

- current context (current Activity)
- target Activity

Intents

```
Intent intent = new Intent(MyActivity.this,
   MyOtherActivity.class);
startActivity(intent);
```

startActivity is a method of the Activity class, with one intent parameter

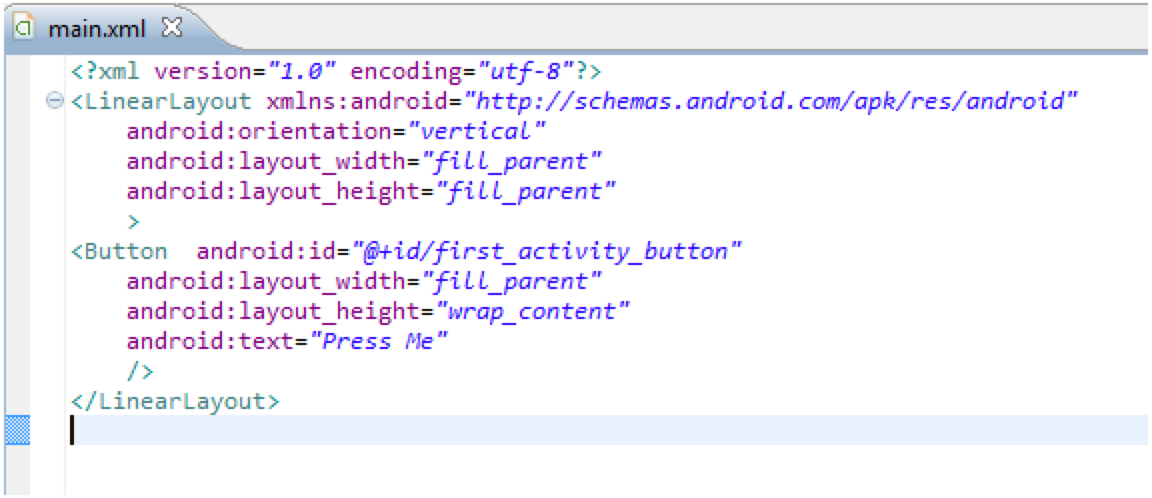
Forces creation of the new Activity

Passes intent to the new Activity

Second activity is an independent piece

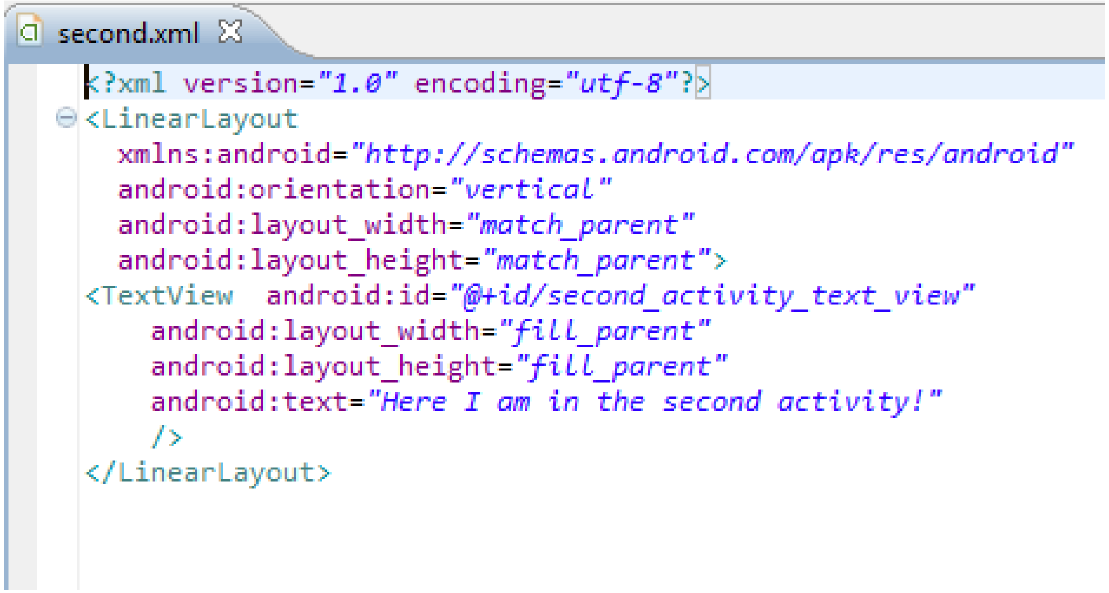
(no return of information to first)

Two Activity Example: Two Layouts



```
<?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"
    >
<Button  android:id="@+id/first_activity_button"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Press Me"
    />
</LinearLayout>
```

First Activity

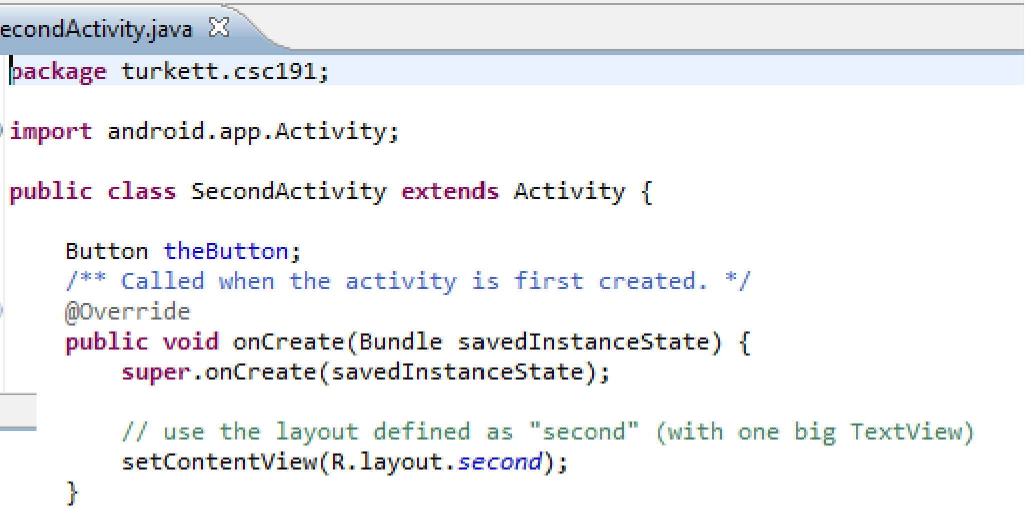


```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:layout_width="match_parent"
  android:layout_height="match_parent">
<TextView  android:id="@+id/second_activity_text_view"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:text="Here I am in the second activity!"
    />
</LinearLayout>
```

Second Activity

Two Activity Example: Two Activities

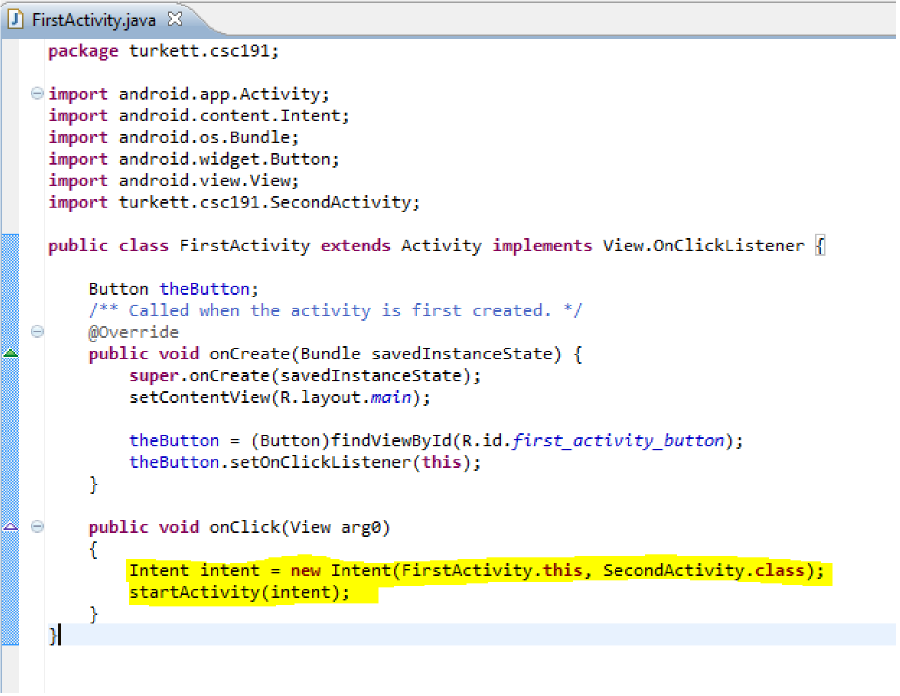
Second Activity



SecondActivity.java

```java
package turkett.csc191;

import android.app.Activity;

public class SecondActivity extends Activity {

    Button theButton;
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // use the layout defined as "second" (with one big TextView)
        setContentView(R.layout.second);
    }
```

First Activity



FirstActivity.java

```java
package turkett.csc191;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import android.view.View;
import turkett.csc191.SecondActivity;

public class FirstActivity extends Activity implements View.OnClickListener {

    Button theButton;
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        theButton = (Button)findViewById(R.id.first_activity_button);
        theButton.setOnClickListener(this);
    }

    public void onClick(View arg0)
    {
        Intent intent = new Intent(FirstActivity.this, SecondActivity.class);
        startActivity(intent);
    }
}
```