16

Android
External Resources

Victor Matos
Cleveland State University

Notes are based on:

The Busy Coder's Guide to Android Development
by Mark L. Murphy
Copyright © 2008-2009 CommonsWare, LLC.
ISBN: 978-0-9816780-0-9
&
Android Developers
http://developer.android.com/index.html

Android Resources

Resources and Internationalization

Resources are external files (that is, non-code files) that are used by your code and compiled into your application at build time.

Android supports a number of different kinds of resource files, including XML, PNG, and JPEG files.

Resources are externalized from source code, and XML files are compiled into a binary, fast loading format for efficiency reasons. Strings, likewise, are compressed into a more efficient storage form.

http://developer.android.com/guide/topics/resources/resources-i18n.html

Android Resources

Using Resources

The Android resource system keeps track of all *non-code assets* associated with an application.

You use the **Resources** class to access your application's resources; the Resources instance associated with your application can generally be found through **Context.getResources()**.

To use a resource, you must install it correctly in the **source tree** and build your application.

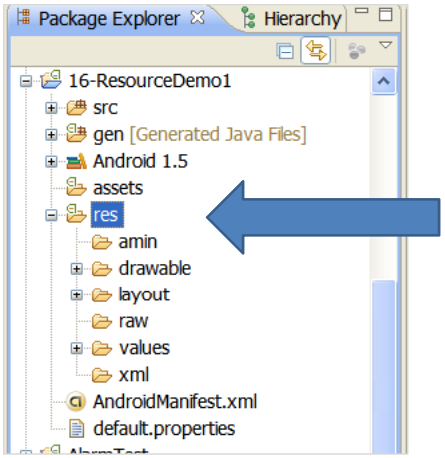
Android Resources

Copy/Paste Resources

You will create and store your resource files under the appropriate subdirectory under the **res/** directory in your project.

Resources are compiled into the final APK file.

Android creates a wrapper class, called **R**, that you can use to refer to these resources in your code. R contains subclasses named according to the path and file name of the source file

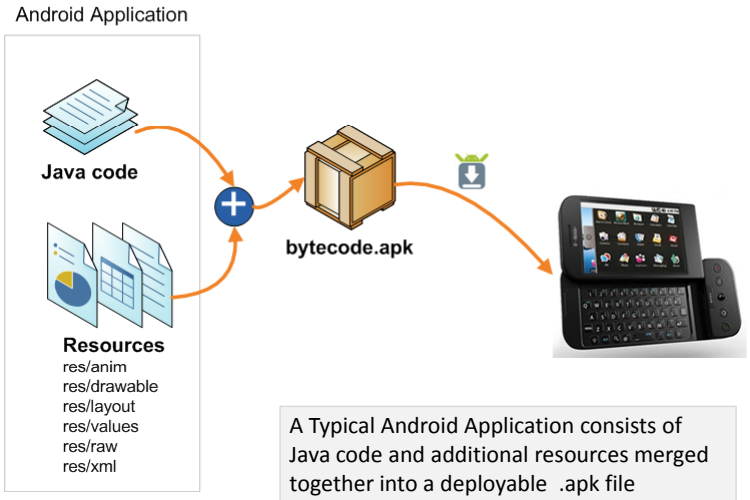


Android Resources

Directory	Resource Types
`res/anim/`	XML files that are compiled into frame by frame animation or tweened animation objects
`res/drawable/`	.png, .9.png, .jpg files. To get a resource of this type, use `mContext.getResources().getDrawable(R.drawable.imageId)`
`res/layout/`	XML files that are compiled into screen layouts (or part of a screen).
`res/values/`	XML files that can be compiled into many kinds of resource. **arrays.xml** to define arrays **colors.xml** to define color drawables and color string values. Use `Resources.getDrawable()` and `Resources.getColor(), respectively,` to get these resources. **dimens.xml** to define dimension value. Use `Resources.getDimension()` to get these resources. **strings.xml** to define string values (use either `Resources.getString` or preferably `Resources.getText()` to get these resources. `getText()` will retain any rich text styling which is usually desirable for UI strings. **styles.xml** to define style objects.
`res/xml/`	Arbitrary XML files that are compiled and can be read at run time by calling Resources.getXML().
`res/raw/`	Arbitrary files to copy directly to the device. They are added uncompiled to the compressed file that your application build produces. To use these resources in your application, call Resources.openRawResource() with the resource ID, which is R.raw.*somefilename*

5

Android Resources



A Typical Android Application consists of Java code and additional resources merged together into a deployable .apk file

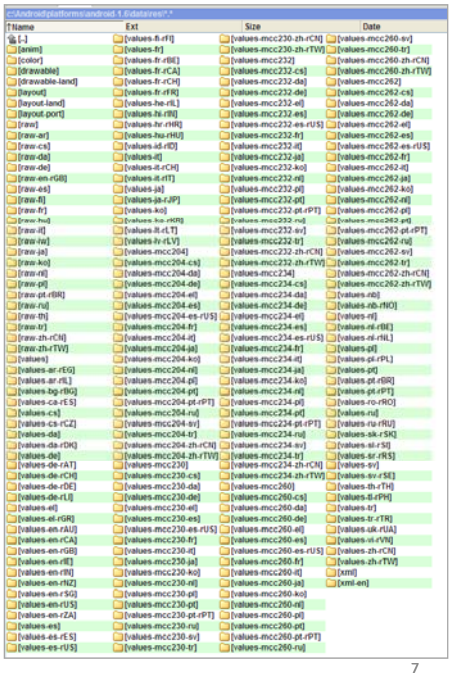
6

Resources

Examples.

To see a number of samples you should explore the folder:

c:\Android\platforms\android-1.5\data\res



Resources

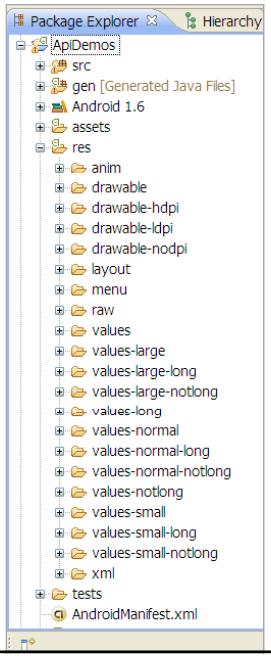
More Examples.

Try to install the **ApiDemos** application. Explore its resource folder. Find the source code in the folder:

c:\Android\platforms\android-1.6\samples

How to install the App:

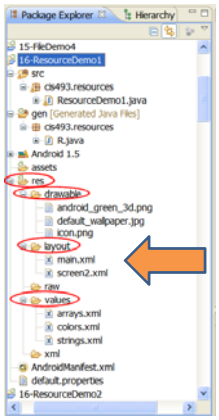
File -> New -> Project -> **Android** Project -> Next
Select "*Create project form existing source*"
Select the **ApiDemos** folder (all remaining fields will be self adjusted)



Android Resources

Java Statements for Using Resources

Displaying a screen layout:



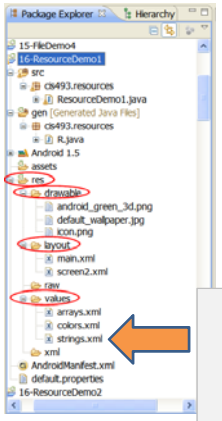

```
setContentView(R.layout.main);
```

```
setContentView(R.layout.screen2);
```

Android Resources

Java Statements for Using Resources

Retrieving **String** Resources from: res/values/…



/res/values/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
<string name="hello">Hola Mundo!, ResourceDemo1!</string>
<string name="app_name">ResourceDemo1</string>
<string name="good_bye">Hasta luego</string>
<string name="color_caption">Color:</string>
<string name="color_prompt">Seleccione un Color</string>
<string name="planet_caption">
      <b>Planeta </b>Planeta <i>Planeta </i><u>Planeta:
</u></string>
<string name="planet_prompt">Seleccione un Planeta</string>
</resources>
```

```
String msg =
   this.getString(R.string.color_prompt);
```

Android Resources

Java Statements for Using Resources

Enhancing externalized **String** resources from: res/values/…

/res/values/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
<string name="hello">Hola Mundo!, ResourceDemo1!</string>
<string name="app_name">ResourceDemo1</string>
<string name="good_bye">Hasta luego</string>
<string name="color_caption">Color:</string>
<string name="color_prompt">Seleccione un Color</string>
<string name="planet_caption">
       <b>Planeta </b>Planeta <i>Planeta </i><u>Planeta: </u></string>
<string name="planet_prompt">Seleccione un Planeta</string>
</resources>
```

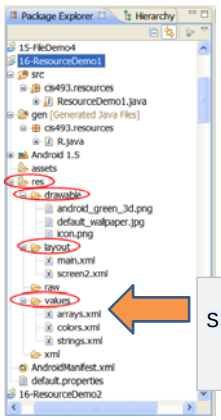
As in HTML a string using **<b>, <i>, <u>** modifiers will be rendered in: bold, italics, and, underlined modes. In our example:

Planeta *Planeta* Planeta Planeta

Android Resources

Java Statements for Using Resources

Retrieving **Array** Resources from: res/values/…



/res/values/arrays.xml

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

    <string-array name="colors">
        <item>red</item>
        <item>orange</item>
        <item>yellow</item>
        <item>green</item>
        <item>blue</item>
        <item>violet</item>
    </string-array>

    <string-array name="planets">
        <item>Mercury</item>
        <item>Venus</item>
        <item>Earth</item>
        <item>Mars</item>
        <item>Jupiter</item>
        <item>Saturn</item>
        <item>Uranus</item>
        <item>Neptune</item>
        <item>Pluto</item>
    </string-array>

</resources>
```

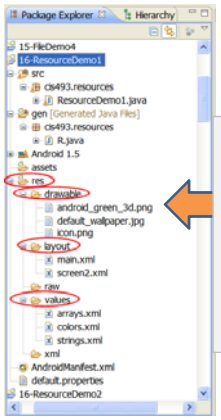
```
String myColors[] =
    this.getResources().getStringArray(R.array.colors);
```

Android Resources

Java Statements for Using Resources

Retrieving a **drawable image** from: res/drawable/…

/res/drawable/ 




```
//same as xml layout attribute
//android:src="@drawable/android_green_3d“

imageView1.setImageResource(
                R.drawable.android_green_3d);
```


Android Resources

Example1. Using Embedded Resources (drawable, string, array).

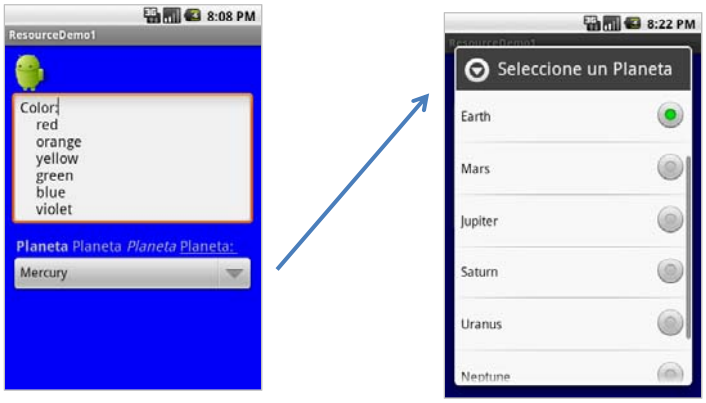
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:padding="10dip"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/solid_blue"
    >

    <ImageView
        android:id="@+id/ImageView01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        >
        </ImageView>

    <EditText
        android:id="@+id/txtColorBox"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
    />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dip"
        android:textSize="18px"
        android:text="@string/planet_caption"
    />

    <Spinner android:id="@+id/spinner2"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:drawSelectorOnTop="true"
        android:prompt="@string/planet_prompt"
    />

</LinearLayout>
```

/res/values/color.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <drawable name="red">#7f00</drawable>
    <drawable name="blue">#770000ff</drawable>
    <drawable name="green">#7700ff00</drawable>
    <color name="solid_red">#f00</color>
    <color name="solid_blue">#0000ff</color>
    <color name="solid_green">#f0f0</color>
    <color name="solid_yellow">#ffffff00</color>
</resources>
```

Android Resources

Example1. Using Embedded Resources (drawable, string, array).



Android Resources

Example1. Using Embedded Resources (drawable, string, array).

```
// using Resources (adapted from Android - ApiDemos)
package cis493.resources;

import android.app.Activity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.Spinner;

public class ResourceDemo1 extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        //TRY: setContentView(R.layout.screen2);

        ImageView imageView1 = (ImageView)findViewById(R.id.ImageView01);
        //similar to xml layout android:src="@drawable/android_green_3d"
        imageView1.setImageResource(R.drawable.android_green_3d);
```

Android Resources

Example1. Using Embedded Resources (drawable, string, array).

```
        EditText txtColorBox = (EditText)findViewById(R.id.txtColorBox);

        String msg = this.getString(R.string.color_caption);

       String myColors[] = this.getResources().getStringArray(R.array.colors);

       for ( int i=0; i<myColors.length; i++){
            msg +=  "\n\t" + myColors[i];
       }
       txtColorBox.setText(msg);


       Spinner s2 = (Spinner) findViewById(R.id.spinner2);
       ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(
           this,
           R.array.planets,
           android.R.layout.simple_spinner_item);

       adapter.setDropDownViewResource(
             android.R.layout.simple_spinner_dropdown_item);
             s2.setAdapter(adapter);
   }
}
```

Android Resources

mmmm

Android Resources

mmmm

Android Resources

mmmm

Resources

Questions ?