Android User Interface (Basic Widgets)

Mobile Applications

ANDROID

Objectives

Basic Widgets:

- Label
- Button
- Image View and Image Button
- EditText
- CheckBox
- RadioButton

Unit measurement in Android

Using @ in XML Layouts

```
<?xml version="1.0" encoding="utf-8"?>
<Button
xmlns:android="http://schemas.android.com/apk/res/android"
   android:id="@+id/myButton"
   android:text=""
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
/>
```

Anything you *do want to use in your Java source needs an* **android:id="…"**

The convention is to use ***@+id/nnn*** as the id value***,*** where ***the nnn*** represents your locally-unique name for the widget ***(eg. @+id/myButton).***

Attaching Layouts to Java Code

Assume ***res/layout/main.xml*** has been created. This layout could be called by an application using the statement

setContentView(R.layout.*main);*

Individual widgets, such as *myButton* could be accessed by the application using the statement *findViewByID(...)* as in

Button btn= (Button) findViewById(R.id.*myButton);*

Where **R** is a class automatically generated to keep track of resources available to the application. In particular R.id... is the collection of widgets defined in the XML layout.

Attaching Layouts to Java Code(Cont.)

Attaching Listeners to the Widgets

The button of our example could now be used, for instance a listener for the click event could be written as:

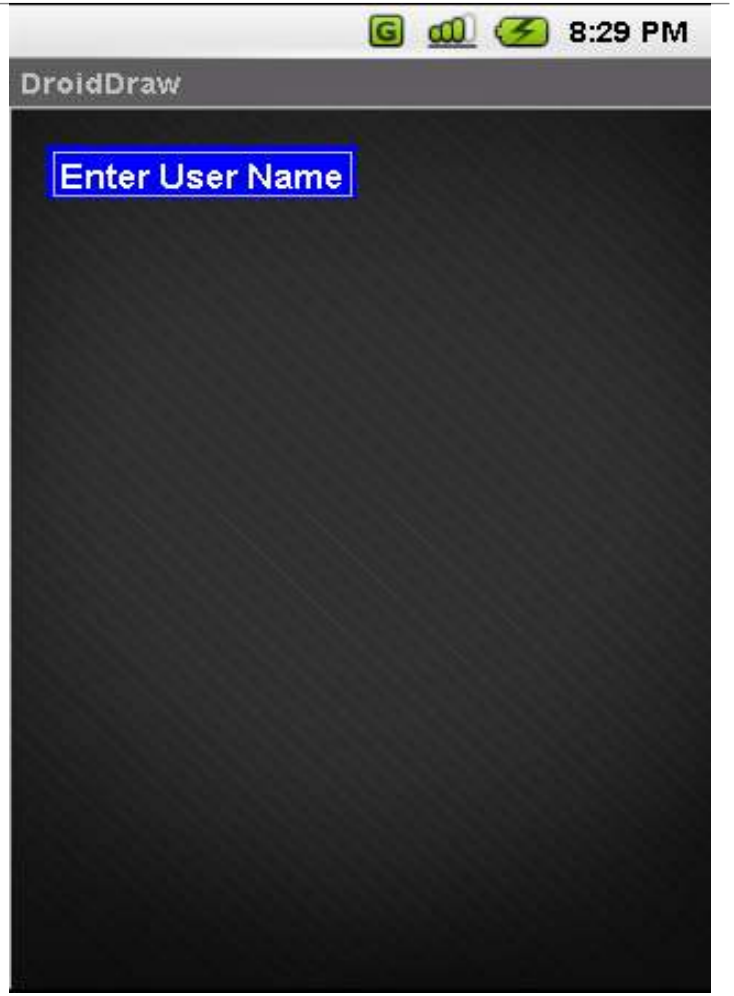
```
btn.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        updateTime();
    }
});

private void updateTime() {
    btn.setText(new Date().toString());
}
```

1-Basic Widgets: Labels

A label is called in android a **TextView.**

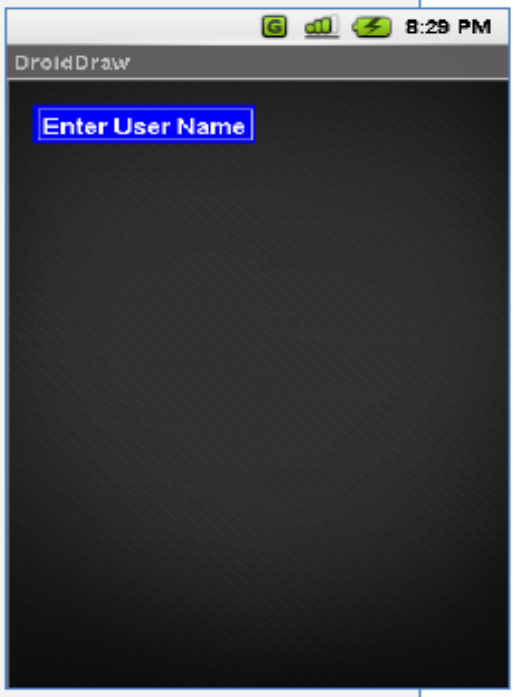
TextViews are typically used to display a caption.

TextViews are *not* editable, therefore they take no input.



2- Basic Widgets: Labels(Cont.)

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout
    android:id="@+id/absLayout"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    xmlns:android="http://schemas.android.com/apk/res/android"
>
<TextView
    android:id="@+id/myTextView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#ff0000ff"
    android:padding="3px"
    android:text="Enter User Name"
    android:textSize="16sp"
    android:textStyle="bold"
    android:gravity="center"
    android:layout_x="20px"
    android:layout_y="22px" >
</TextView>

</AbsoluteLayout>
```



2- Basic Widgets: Buttons

```
...
<Button
android:id="@+id/btnExitApp"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:padding="10px"
android:layout_marginLeft="5px"
android:text="Exit Application"
android:textSize="16sp"
android:textStyle="bold"
android:gravity="center"
android:layout_gravity="center_horizontal"
>
</Button>
```

3- Basic Widgets: Images

ImageView and **ImageButton** are two Android widgets that allow embedding of images in your applications.

Each widget takes an **android:src** or **android:background** attribute (in an XML layout) to specify what picture to use.

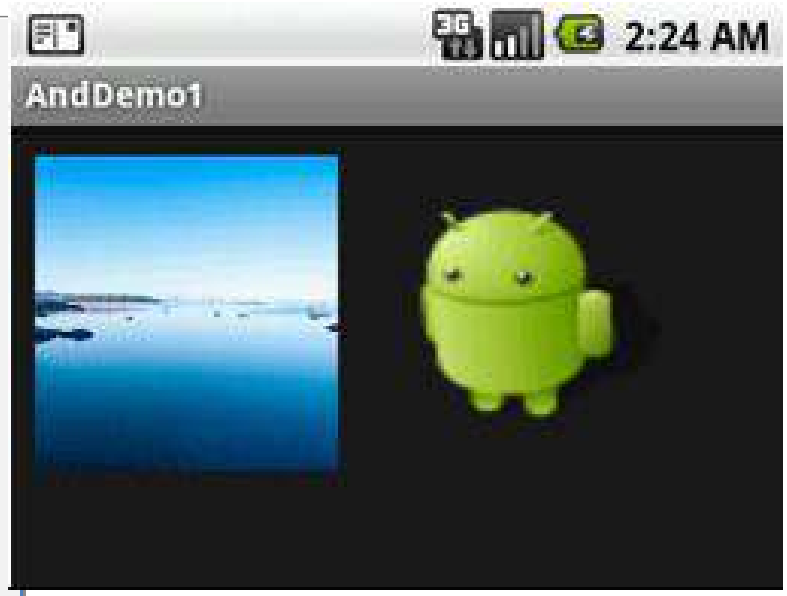
Pictures are usually reference a drawable resource.

You can also set the image content based on a URI from a content provider via setImageURI().

ImageButton, is a subclass of ImageView. It adds the standard Button behavior for responding to clickevents.

3- Basic Widgets: Images(Cont.)

```
. . .
<ImageButton
        android:id="@+id/myImageBtn1"
        android:background="@drawable/default_wallpaper"
        android:layout_width="125px"
        android:layout_height="131px"
>
</ImageButton>

<ImageView
        android:id="@+id/myImageView1"
        android:background="@drawable/ic_launcher_android"
        android:layout_width="108px"
        android:layout_height="90px"
>
</ImageView>
```

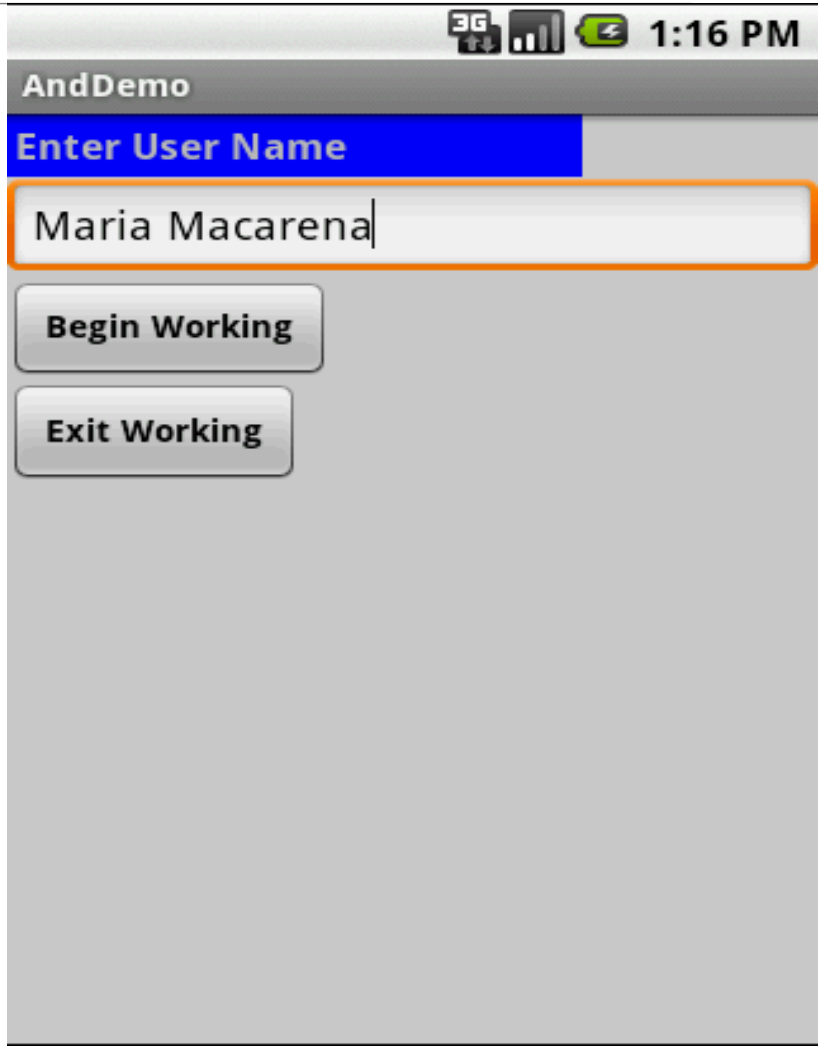


4- Basic Widgets: EditText

The **EditText**(or textBox) widget is an extension of TextView that allows updates.

The control configures itself to be *editable.*

Important Java methods are:
txtBox.setText("someValue") and
txtBox.getText().toString()



4- Basic Widgets: EditText(Cont.)

Example

```
...
<EditText
android:id="@+id/txtUserName"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:textSize="18sp"
android:autoText="true"
android:capitalize="words"
android:hint="First Last Name"
>
</EditText>
```

Basic Widgets: Example 1

Basic Widgets: Example 1(Cont.)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
android:id="@+id/linearLayout"
android:layout_width="fill parent"
android:layout_height="fill parent"
android:background="#ffcccccc"
android:orientation="vertical"
xmlns:android="http://schemas.android.com/apk/res/
android"
>

<TextView
android:id="@+id/labelUserName"
android:layout_width="227px"
android:layout_height="wrap content"
android:background="#ff0000ff"
android:padding="3px"
android:text="Enter User Name"
android:textSize="16sp"
android:textStyle="bold"
>
</TextView>
```

```
<EditText
android:id="@+id/txtUserName"
android:layout_width="fill parent"
android:layout_height="wrap_content"
android:textSize="18sp"
android:autoText="true"
android:capitalize="words"
android:hint="First Last Name"
>
</EditText>

<Button
android:id="@+id/btnBegin"
android:layout_width="wrap content"
android:layout_height="wrap content"
android:text=" Begin Working "
android:textSize="14px"
android:textStyle="bold"
>
</Button>

</LinearLayout>
```

Basic Widgets: Example 1

```java
package cis493.gui;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
//////////////////////////////////////////////////////////////////////////
// "LOGIN" - a gentle introduction to UI controls

public class AndDemo extends Activity {
    TextView labelUserName;
    EditText txtUserName;
    Button btnBegin;

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

        //binding the UI's controls defined in "main.xml" to Java code
        labelUserName = (TextView) findViewById(R.id.labelUserName);
        txtUserName = (EditText) findViewById(R.id.txtUserName);
        btnBegin = (Button) findViewById(R.id.btnBegin);
```

Basic Widgets: Example (Cont.)

```
        //LISTENER: wiring the button widget to events-&-code
        btnBegin.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
            String userName = txtUserName.getText().toString();
            if (userName.compareTo("Maria Macarena")==0){
                labelUserName.setText("OK, please wait...");
                Toast.makeText(getApplicationContext(),
                        "Bienvenido " + userName,
                        Toast.LENGTH_SHORT).show();
            }
            Toast.makeText(getApplicationContext(),
                    "Bienvenido " + userName,
                    Toast.LENGTH_SHORT).show();
        }

         });// onClick

    }//onCreate

}//class
```

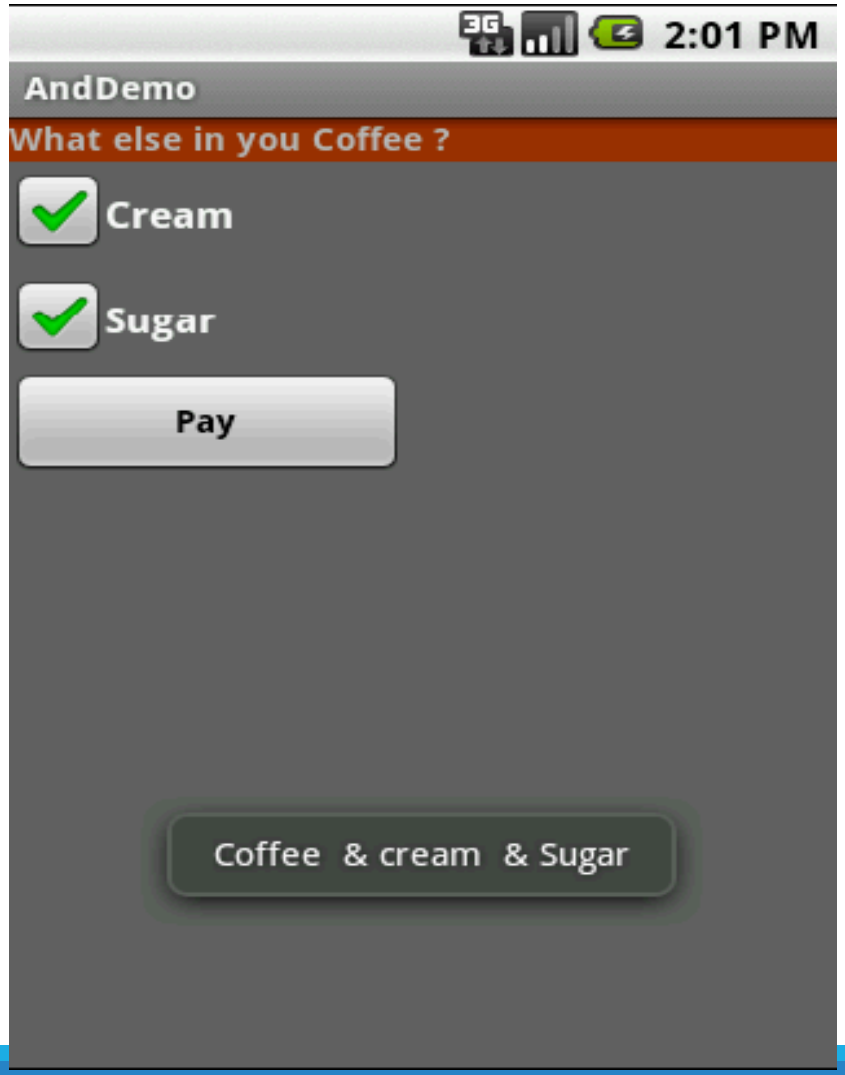
Basic Widgets: Example 1

```java
public class AndDemo extends Activity implements OnClickListener {
    Button btnBegin;
    Button btnExit;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        //binding the UI's controls defined in "main.xml" to Java code
        btnBegin = (Button) findViewById(R.id.btnBegin);
        btnExit = (Button) findViewById(R.id.btnExit);

        //LISTENER: wiring the button widget to events-&-code
        btnBegin.setOnClickListener(this);
        btnExit.setOnClickListener(this);
    }//onCreate

    @Override
    public void onClick(View v) {
        if (v.getId()==btnBegin.getId() ){
            Toast.makeText(getApplicationContext(), "1-Begin", 1).show();
        }
        if (v.getId()==btnExit.getId() ){
            Toast.makeText(getApplicationContext(), "2-Exit", 1).show();
        }
    }//onClick
}//class
```

5- Basic Widgets: CheckBox

A checkbox is a specific type of two-states button that can be either *checked* or *unchecked*.

An example usage of a checkbox inside your activity would be the following:



5- Basic Widgets: CheckBox (Cont.)

Layout: main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
android:id="@+id/linearLayout"
android:layout_width="fill parent"
android:layout_height="fill parent"
android:background="#ff666666"
android:orientation="vertical"
xmlns:android="http://schemas.android.com/apk/res/android"
>

<TextView
android:id="@+id/labelCoffee"
android:layout_width="fill parent"
android:layout_height="wrap content"
android:background="#ff993300"
android:text="What else in you Coffee ?"
android:textStyle="bold"
>
</TextView>
```

```
<CheckBox
android:id="@+id/chkCream"
android:layout_width="wrap content"
android:layout_height="wrap content"
android:text="Cream"
android:textStyle="bold"
>
</CheckBox>
<CheckBox
android:id="@+id/chkSugar"
android:layout_width="wrap content"
android:layout_height="wrap content"
android:text="Sugar"
android:textStyle="bold"
>
</CheckBox>
<Button
android:id="@+id/btnPay"
android:layout_width="153px"
android:layout_height="wrap content"
android:text="Pay"
android:textStyle="bold"
>
</Button>
</LinearLayout>
```

5- Basic Widgets: CheckBox (Cont.)

```
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.Toast;

public class AndDemo extends Activity {
    CheckBox chkCream;
    CheckBox chkSugar;
    Button btnPay;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        //binding XMl controls with Java code
        chkCream = (CheckBox)findViewById(R.id.chkCream);
        chkSugar = (CheckBox)findViewById(R.id.chkSugar);
        btnPay = (Button) findViewById(R.id.btnPay);
```

5- Basic Widgets: CheckBox (Cont.)

```
        //LISTENER: wiring button-events-&-code
        btnPay.setOnClickListener(new OnClickListener() {

                @Override
                public void onClick(View v) {
                        String msg = "Coffee ";
                        if (chkCream.isChecked()) {
                                msg += " & cream ";
                        }
                        if (chkSugar.isChecked(){
                                msg += " & Sugar";
                        }
                        Toast.makeText(getApplicationContext(),
                                        msg, Toast.LENGTH_SHORT).show();
                        //go now and compute cost...

                }//onClick


        });
    }//onCreate
}//class
```

6- Basic Widgets: RadioButtons

A radio button is a two-states button that can be either ***checked*** *or* ***unchecked.***

When the radio button is unchecked, the user can press or click it to check it.

Radio buttons are normally used together in a **RadioGroup.**

When several radio buttons live inside a radio group, checking one radio button ***unchecks*** all *the others.*

Similarly, you can call **isChecked()** on a RadioButtonto see if it is selected, **toggle()** to select it, and so on, like you can with a CheckBox.

6- Basic Widgets: RadioButtons(Cont.)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
android:id="@+id/myLinearLayout"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:orientation="vertical"
xmlns:android="http://schemas.android.com/apk/res/android"
>

<RadioGroup
android:id="@+id/radGroupCoffeeType"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:orientation="vertical"
>
<TextView
android:id="@+id/labelCoffeeType"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:background="#ff993300"
android:text="What type of coffee?"
android:textStyle="bold"
>
<RadioButton
android:id="@+id/radDecaf"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Decaf"
>
</RadioButton>
<RadioButton
android:id="@+id/radExpresso"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Expresso"
>
</RadioButton>
<RadioButton
android:id="@+id/radColombian"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Colombian"
>
</RadioButton>
</RadioGroup>
```

6- Basic Widgets: RadioButtons(Cont.)

```
package cis493.demoui;
// example using RadioButtons
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.Toast;

public class AndDemoUI extends Activity {
    CheckBox chkCream;
    CheckBox chkSugar;
    Button btnPay;
    RadioGroup radCoffeeType;
    RadioButton radDecaf;
    RadioButton radExpresso;
    RadioButton radColombian;
```

6- Basic Widgets: RadioButtons(Cont.)

```
@Override
   public void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.main);
       //binding XMl controls to Java code
       chkCream = (CheckBox)findViewById(R.id.chkCream);
       chkSugar = (CheckBox)findViewById(R.id.chkSugar);
       btnPay = (Button) findViewById(R.id.btnPay);

       radCoffeeType = (RadioGroup)findViewById(R.id.radGroupCoffeeType);
       radDecaf = (RadioButton)findViewById(R.id.radDecaf);
       radExpresso = (RadioButton)findViewById(R.id.radExpresso);
       radColombian = (RadioButton)findViewById(R.id.radColombian);
```

6- Basic Widgets: RadioButtons(Cont.)

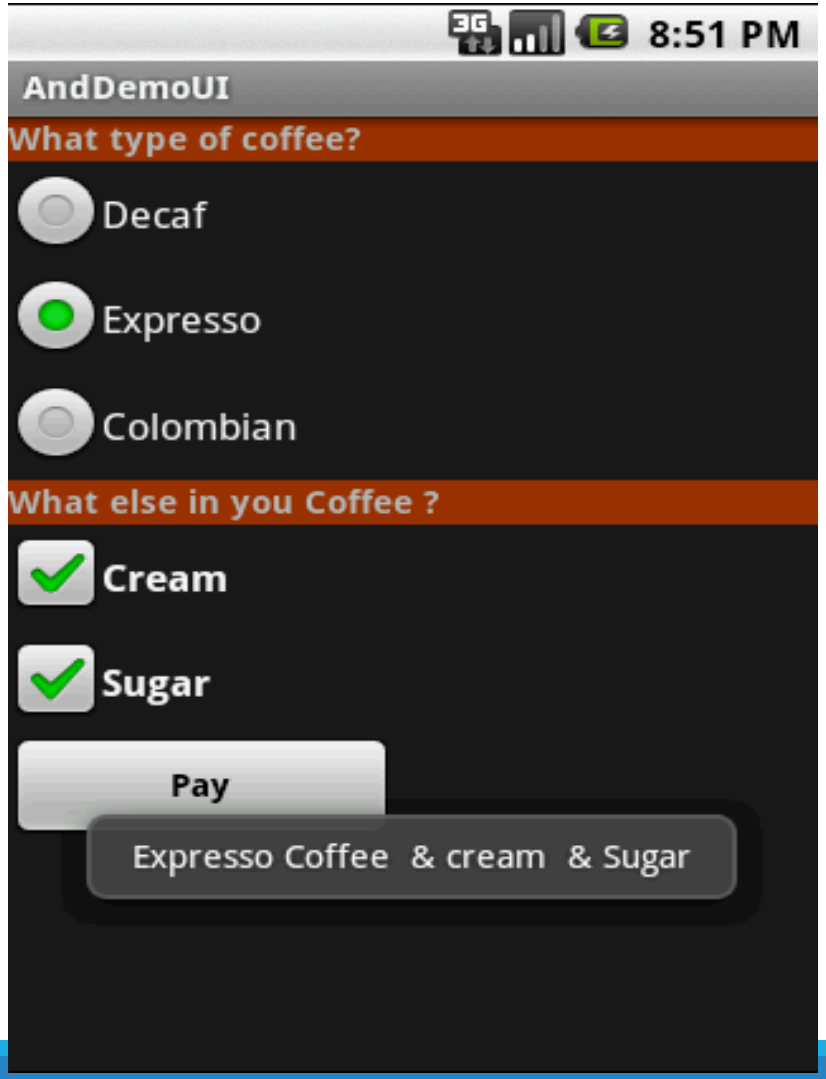
```
//LISTENER: wiring button-events-&-code
btnPay.setOnClickListener(new OnClickListener() {
  @Override
  public void onClick(View v) {
      String msg = "Coffee ";
      if (chkCream.isChecked())
        msg += " & cream ";

      if (chkSugar.isChecked())
        msg += " & Sugar";

      // get radio buttons ID number
      int radioId = radCoffeeType.getCheckedRadioButtonId();
      // compare selected's Id with individual RadioButtons ID
      if (radColombian.getId()== radioId)
          msg = "Colombian " + msg;
      // similarly you may use .isChecked() on each RadioButton
      if (radExpresso.isChecked())
          msg = "Expresso " + msg;

      Toast.makeText(getApplicationContext(), msg, Toast.LENGTH_SHORT).show();
      // go now and compute cost...
      }// onClick
      });
 }// onCreate
}// class
```

6- Basic Widgets: RadioButtons(Cont.)

Example

This UI uses *RadioButtons and CheckBoxes* to define choices

RadioGroup



Summary of choices

Unit measurement in Android

- px = Pixels – corresponds to actual pixels on the screen.
- pt = Points – 1/72 of an inch based on the physical size of the screen.
- dp

Density-independent Pixels – an abstract unit that is based on the physical density of the screen. These units are relative to a 160 dpi screen, so one dp is one pixel on a 160 dpi screen. The ratio of dp-to-pixel will change with the screen density, but not necessarily in direct proportion. Note: The compiler accepts both “dip” and “dp”, though “dp” is more consistent with “sp”.

- sp

Scale-independent Pixels – this is like the dp unit, but it is also scaled by the user’s font size preference. It is recommend you use this unit when specifying font sizes, so they will be adjusted for both the screen density and user’s preference.

For more Details

This is a power point “beginning” lecture on Layout and we can’t cover all the many and elaborate examples --- that is best served by websites and tutorials and books.

Resources

https://www.slideshare.net/