



Accelerating Information Technology Innovation

<http://aiti.mit.edu>

Cali, Colombia
Summer 2012

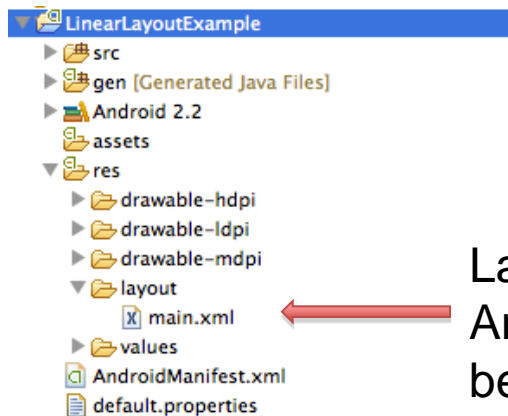
Lesson 03 – Android Layouts, Views,
and Menus

Agenda

- Layouts
- Views and Widgets
- Menus

Layouts

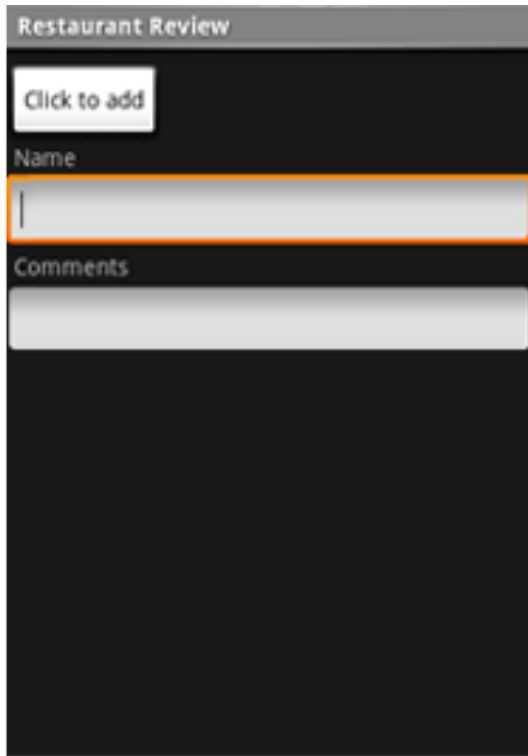
- Defined in two ways
 - XML layout files



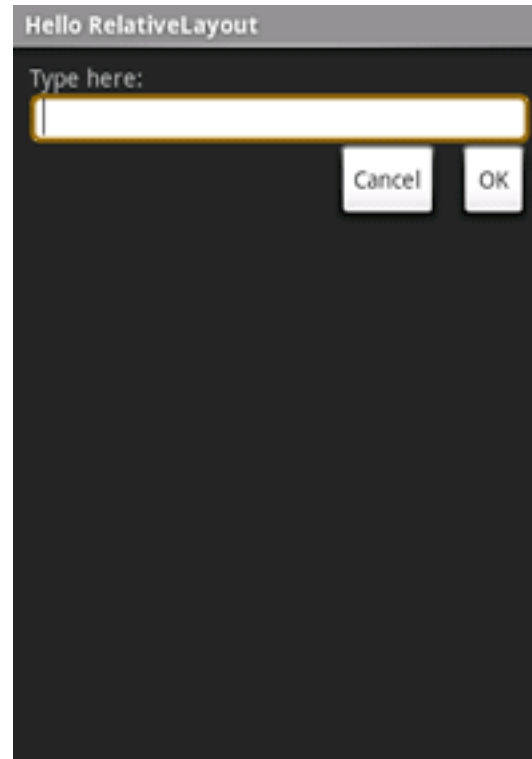
Layout file main.xml is auto-generated when an Android project is created in Eclipse. App layout can be defined in this file in XML.

- using code (e.g. in the onCreate() method)

Some Layouts



LinearLayout



RelativeLayout



TableLayout

LinearLayout

- Arrange components one after another, left-to-right, top-to-bottom:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a TextView" />
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a Button" />
</LinearLayout>
```

Hello, I am a TextView

Hello, I am a Button

RelativeLayout

- Position and align components relative to other components:


```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:background="@drawable/blue"
    android:padding="10px" >

    <TextView android:id="@+id/label"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Type here:" />

    <EditText android:id="@+id/entry"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:background="@android:drawable/editbox_background"
        android:layout_below="@id/label" />

</RelativeLayout>
```

Type here:

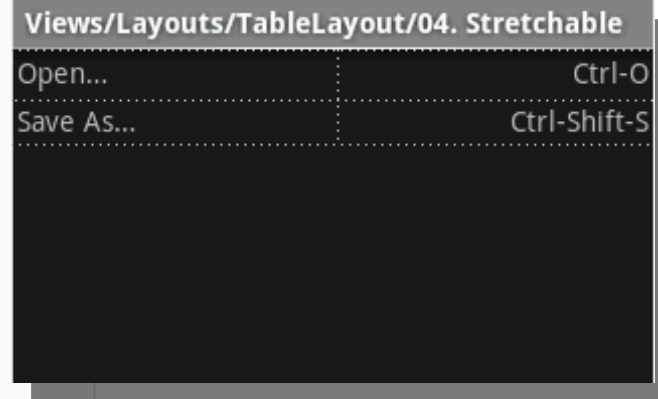
 `android:layout_below` is an attribute that can be used only with `RelativeLayout`. Other such attributes include `layout_alignParentRight`, and `layout_toLeftOf`.

TableLayout

- Position components in rows and columns:

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:stretchColumns="1">
    <TableRow>
        <TextView
            android:text="@string/table_layout_4_open"
            android:padding="3dip" />
        <TextView
            android:text="@string/table_layout_4_open_shortcut"
            android:gravity="right"
            android:padding="3dip" />
    </TableRow>

    <TableRow>
        <TextView
            android:text="@string/table_layout_4_save"
            android:padding="3dip" />
        <TextView
            android:text="@string/table_layout_4_save_shortcut"
            android:gravity="right"
            android:padding="3dip" />
    </TableRow>
</TableLayout>
```



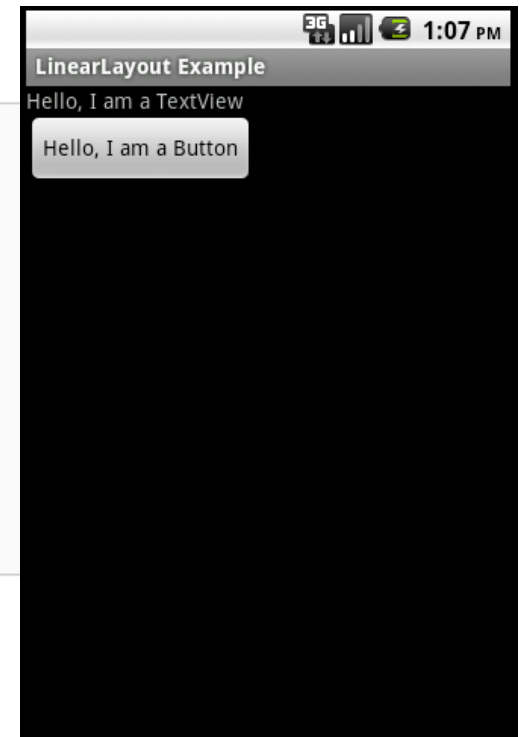
Views

- What they are: UI components
- Some common views and widgets:
 - Button
 - EditText (a text box)
 - TextView (a text label)
 - ListView
 - GridView
 - TabView
 - Spinner (a drop-down menu)
 - CheckBox
 - RadioButton
 - ToggleButton
 - RatingBar
 - MapView (for embedding Google Maps objects in applications)
 - WebView (for embedding web browsers in applications)

Adding Views to Layouts

- Example: adding a button and text label to a LinearLayout:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a TextView" />
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a Button" />
</LinearLayout>
```

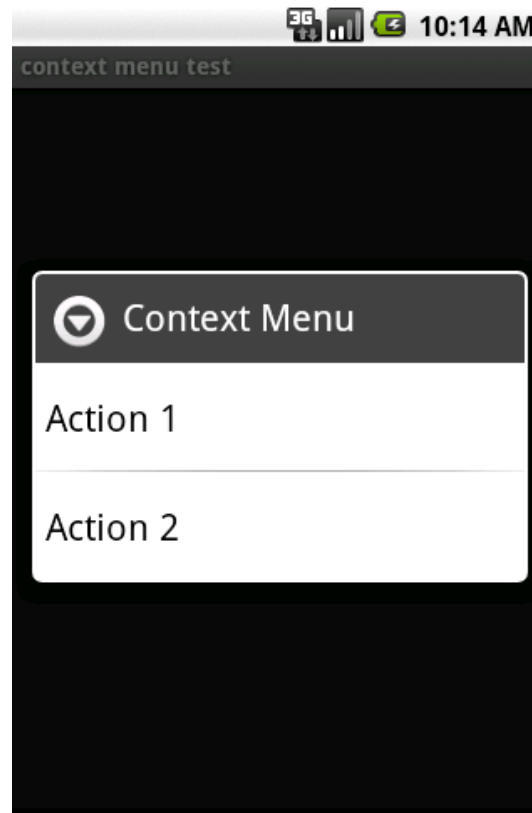


Menu

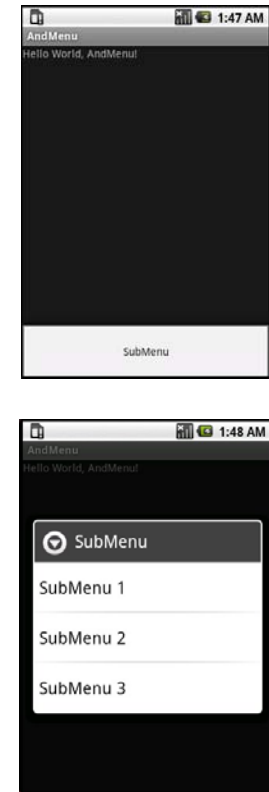
Options Menu



Context Menu



SubMenu



OptionsMenu Example

- Step 1: Implement `onCreateOptionsMenu()` method

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    boolean result = super.onCreateOptionsMenu(menu);

    menu.add(Menu.NONE, 0, 0, "Activity One");
    menu.add(Menu.NONE, 1, 1, "Activity Two");

    return result;
}
```

- Step 2: Implement `onOptionsItemSelected()` method

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    int itemIndex = item.getItemId();

    if (itemIndex == 0){
        //first menu button pressed. do something here
    }
    else if (itemIndex == 1){
        // second menu button pressed. do something here
    }

    return super.onOptionsItemSelected(item);
}
```

