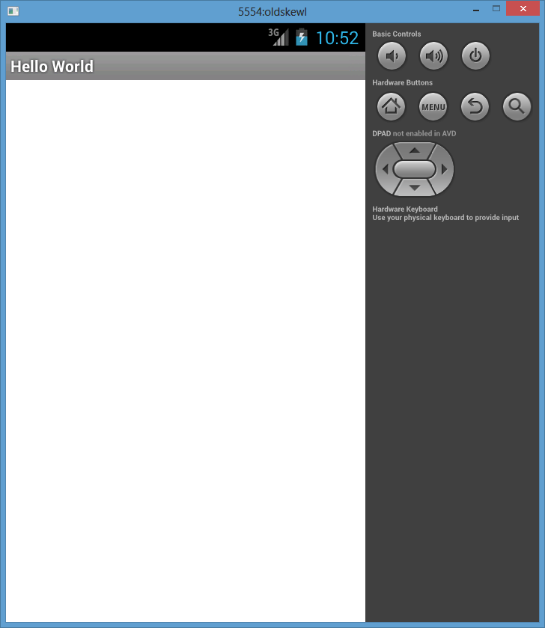
# Lab 1: Your First Android App

Welcome to the world of Android App development. Today we are going to jump right in and start building an Android app. First thing is first, **Open the Hello World Project** with Eclipse. Now let’s run our Android App in the Emulator. Right click the Hello World project folder and choose **Run As --> Android Application**. You will notice the Android Emulator starts up and Runs the Hello World App. Awesome!



Well, not so awesome yet. The app is very boring and doesn’t do anything yet. It’s just a blank screen with a title bar. Now let’s add some text to display to the user. Navigate to the **res/layout/activity\_main.xml** file. This file is known as a Layout File. Layout files define the visual structure for a User Interface. Using Android’s XML vocabulary, you can quickly design UI layouts with a series of nested elements. You can edit Layout Files in two ways, with the XML editor, or the Android Graphical UI Editor. Both display the same thing and you can toggle between the two by clicking on the “Graphical Layout” tab or the “activity\_main.xml” tab at the bottom. **Click on the “activity\_main.xml” tab** and edit your file. Insert a new TextView to display a message to the user.

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

xmlns:tools=*"http://schemas.android.com/tools"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

android:orientation=*"vertical"* >

<TextView

android:id=*"@+id/my\_text"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:textSize=*"20sp"*

android:text=*"Android is fun!"*/>

</LinearLayout>

Now run the app again by chooseing **Run As --> Android Application** or **Ctrl+F11** on the keyboard. Sweet, you make some text appear on the screen. Now let’s add a button to our App. Go back to the res/layout/activity\_main.xml file and add a Button element under the TextView.

<Button

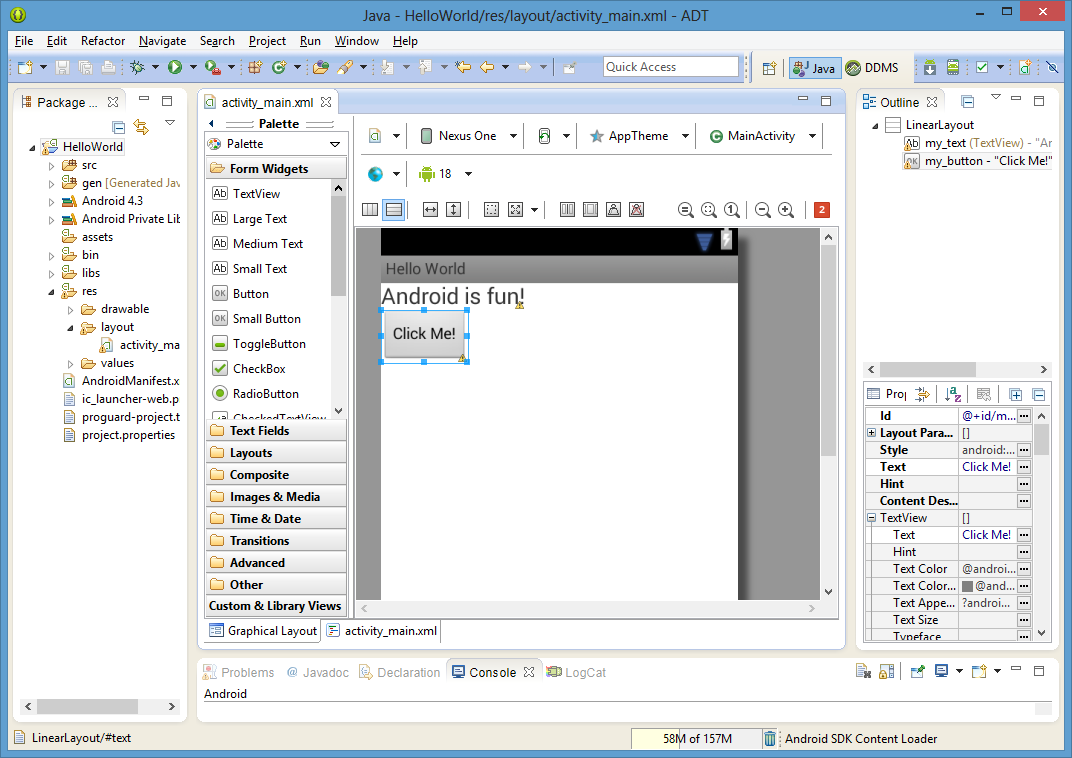
android:id=*"@+id/my\_button"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:text=*"Click Me!"* />

Run your app again. Checkout your new awesome button. You can get a preview of your layout before you run your app with the Android Graphical Layout tool. **Click on the Graphical Layout** tab at the bottom of the activity\_main.xml editor. The Graphical Layout editor is a really powerful tool to help you quickly build Android UI layouts.



You add new UI elements to your layout by dragging them from the Palette bar and positioning them right on your layout. Try out adding another TextView by **dragging TextView from the Palette** **onto your layout.** Pretty cool right? Now switch back to the XML editor by **clicking the activity\_main.xml tab.** Notice how a new XML TextView element has been added to the file. The graphical editor makes changes to the XML code in real time. As you move around elements and change attributes with the graphical editor, it’s actually changing your code too.

Alright that’s all well and good, but now let’s have some real fun. Let’s make the background of our TextView turn green when we click the button. To do so, we will need to write some Java code. **Open src/com.example.helloworld/MainActivity.java** and edit to look like this:

**package** com.example.helloworld;

**import** android.app.Activity;

**import** android.graphics.Color;

**import** android.os.Bundle;

**import** android.view.View;

**import** android.widget.Button;

**import** android.widget.TextView;

**public** **class** MainActivity **extends** Activity {

**private** Button mMyButton;

**private** TextView mMyText;

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_main*);

mMyText = (TextView)findViewById(R.id.*my\_text*);

mMyButton = (Button) findViewById(R.id.*my\_button*);

mMyButton.setOnClickListener(**new** View.OnClickListener(){

@Override

**public** **void** onClick(View v) {

mMyText.setBackgroundColor(Color.*GREEN*);

}

});

}

}

Run your app and try it out.