

# Global Startup

## Meet-up 4: Intro to Android

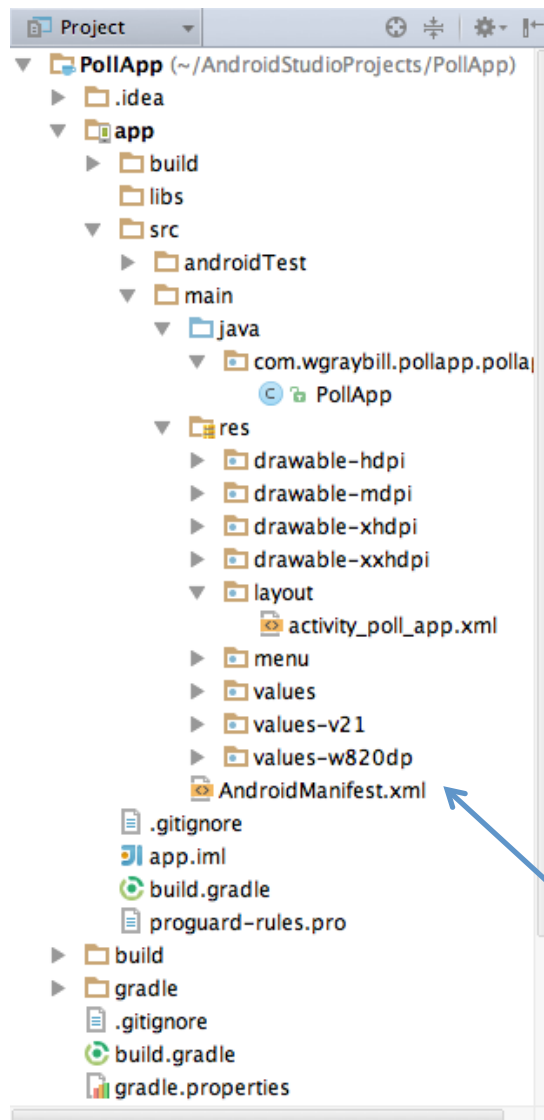


# Today's Meet-up

- Anatomy of an Android Project
- Android Layouts
- Practice with Layouts

# Android Project Structure

# Android Project Structure



Automatically generated files, act as “glue” between .java and .xml files

Your Java Code goes in here

Folders for graphic resources

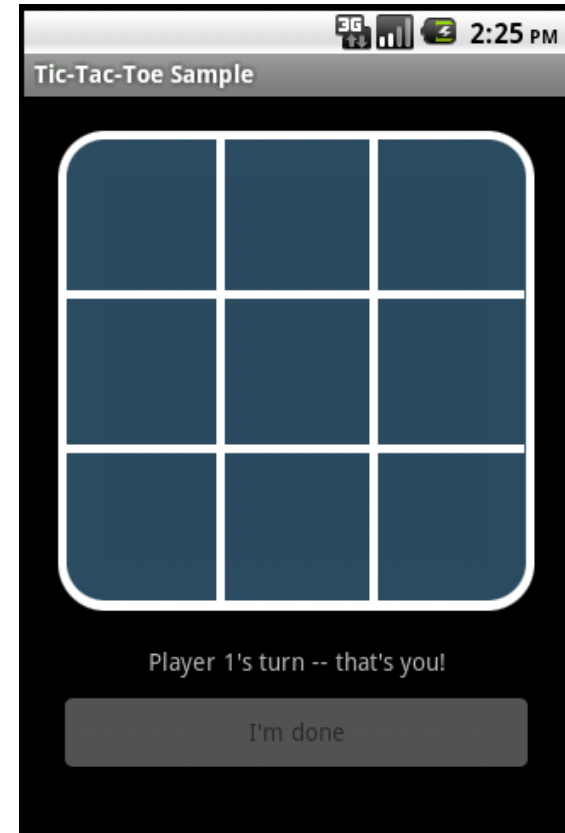
Defines UI layout

Values (strings, styles, dimensions) are defined here

**AndroidManifest.xml**


First file to be executed, points to Java code and contains system definitions/permission


# Example: Tic-Tac-Toe






Malay English Spanish Detect language ▾

 ↔ Malay Spanish ▾ Translate

|  
app screen  


Activity  


# Activity File .java

MainActivity.java

```
package com.example.exampleapp;
```

Package declaration

```
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
```

Import statements

```
public class MainActivity extends Activity {
```

Extend Activity Class

```
@Override
```

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}
```

Create UI (as defined in  
Layout .xml)

```
@Override
```

```
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.main, menu);
    return true;
}
```

Create Options Menu

```
}
```

# AndroidManifest.xml

```
ExampleApp Manifest ✕
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.exampleapp"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="17" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.exampleapp.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Compatibility Information (in build.gradle file in Android Studio)

Icon/ App name/  
Style defined

Points to main Activity file



# The Layout File .xml

Activities – one page of app, only one main activity can exist

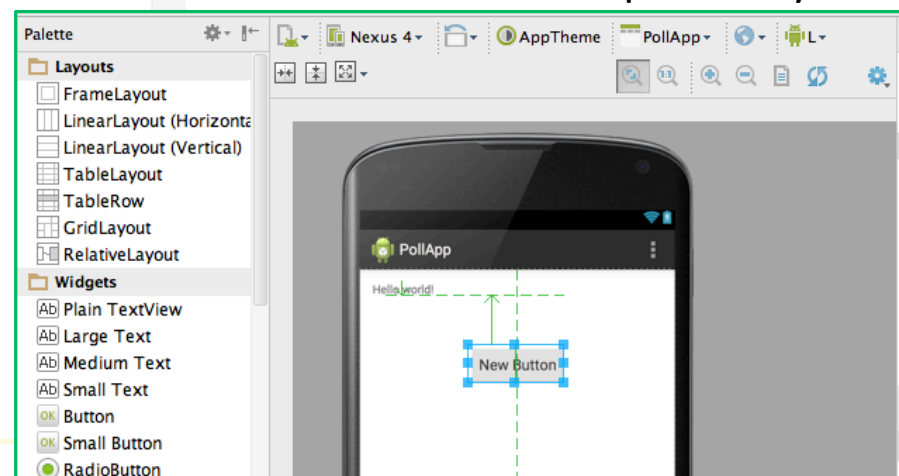
```
PollApp.java x activity_poll_app.xml x strings.xml x
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="64dp"
    android:paddingRight="64dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp"
    tools:context=".PollApp">

    <TextView
        android:text="@string/hello_world"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/textView" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Button"
        android:id="@+id/button"
        android:layout_marginTop="59dp"
        android:layout_below="@+id/textView"
        android:layout_centerHorizontal="true" />

</RelativeLayout>
```

Graphical Layout



# Value Files .xml

The screenshot displays the Android Studio interface for editing an activity layout. On the left, a 'Form Widgets' palette lists various UI components. The main workspace shows a graphical preview of an Android application with the title 'ExampleApp' and the text 'Hello world!' on a white background. Below the preview, the XML code for 'activity\_main.xml' is shown in the 'Graphical Layout' view. The code defines a `RelativeLayout` containing a `TextView` with the text resource `@string/hello_world`. A red circle highlights the `android:text="@string/hello_world"` attribute in the XML code. To the right, the 'strings.xml' file is open, showing the string resource definition for 'hello\_world' with the value 'Hello world!'. A green circle highlights the string value in the XML code, and a red arrow points from the red circle in the activity\_main.xml code to the green circle in the strings.xml code.

```
activity_main.xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hello_world" >

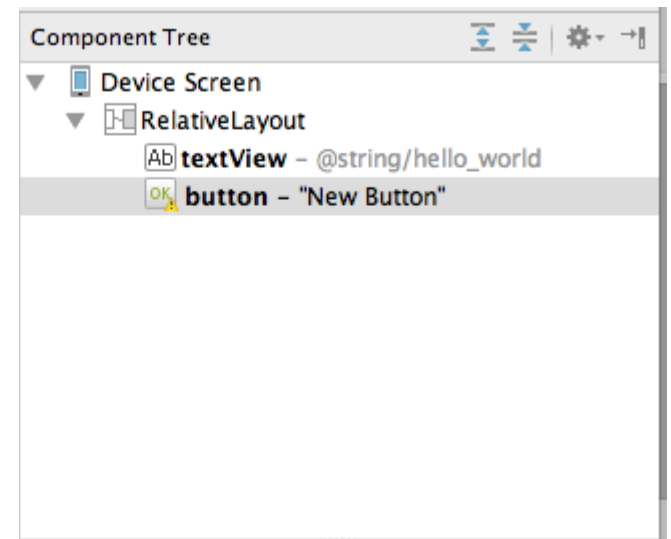
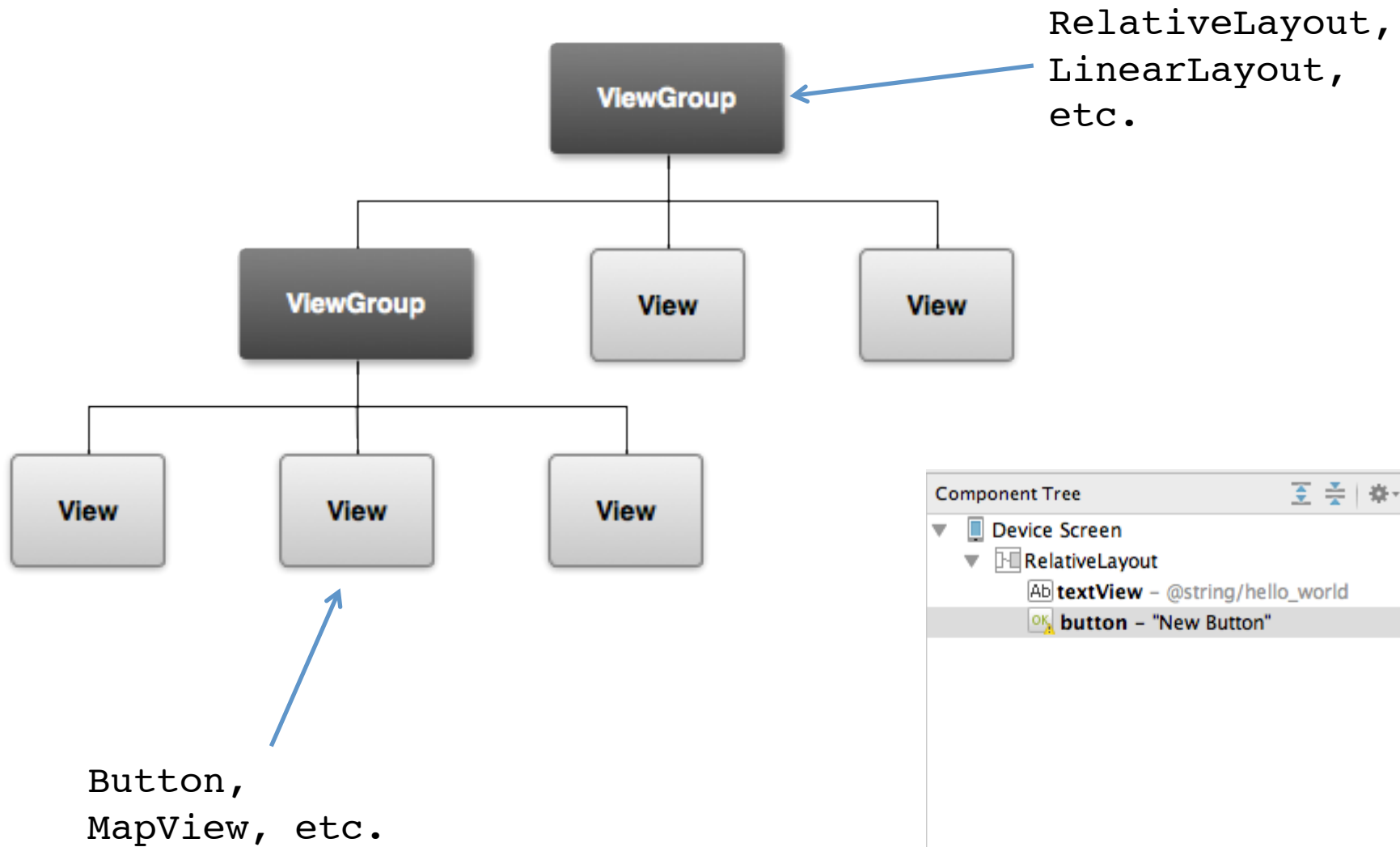
</RelativeLayout>
```

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

    <string name="app_name">ExampleApp</string>
    <string name="action_settings">Settings</string>
    <string name="hello_world">Hello world!</string>

</resources>
```

# Building a UI: Layouts



# 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

# Relative Layout

Position widgets relative to each other: good for more complicated UIs


```
<?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:" />
```

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>
```

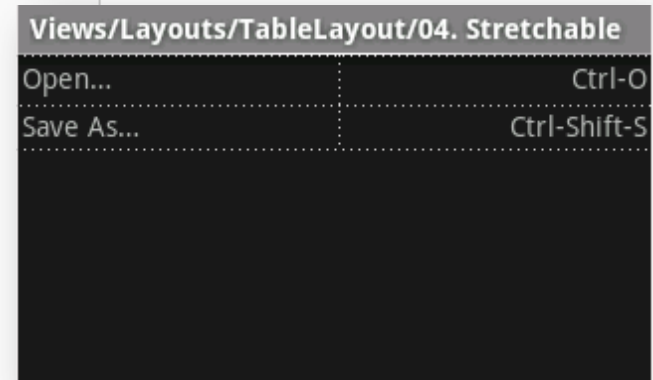
 `android:layout_below` is an attribute that can be used only with `RelativeLayout`. Other such attributes include `layout_alignParentRight`, and `layout_toLeftOf`.

# 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>
```



# Many more layouts and widgets...

- Linear Layout
- Relative Layout
- Grid Layout
- Frame Layout
- Table Layout
- Context Menu
- Option Menu
- Sub Menu
- Sliders
- Graphics
- Animations
- Videos
- Etc, etc, etc...
- 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)

...which are furthermore all customizable



# Today's Assignment

# Play with Layout design

- Work in pairs
- Use both xml and graphical interface
- Be creative! You don't need to match the example

