2015

Android Project Structure

V1.0

JUAN RONDON

Android Project Structure: The Team[1]

Every great team is composed of people who play different roles. Do you want to do the job right? You need the right team. Android Projects have a few key elements and each has a role to play:

1. Java: *The Professional*
2. Resources: *The Artist*
3. AndroidManifest.xml: *The Boss*
4. Intent: *The Job itself*

Java: The Professional

It's the job of your Java code to get things done. Your code is all going to be in the ***src/main/java*** directory under your main project folder.

Resources: *The Artist*

It's not enough to just get the job done. It needs to be done *in style*. Your App is never going to stand out unless it has great icons and images, well-designed layouts, engaging copy text, and maybe even some smooth animations.
Initially, the ***app/res*** (Resources) folder contains:

- ***Drawable*** folders that hold images.
- The ***layout*** folder with XML that represents the screen designs.
- The ***menu*** folder with XML of the items that will appear on the Action Bar.
- The ***values*** folder with XML containing dimensions, strings, and styles.

AndroidManifest.xml: *The Boss*

Someone's got to call the shots. That "someone" would be the Android Manifest. This XML file informs your system of the app's hardware and software requirements and contains your app's name, icon, and version.
The manifest also filters the **`Intents`** coming in. You need a job done by your app? Talk to the boss first. Now, more about the jobs themselves…

[1] http://www.raywenderlich.com/

Intent: The Job itself

Want to show the user a screen? Want to navigate to a website? Whatever the job is, in Android it is going to take the form of an **`Intent`**.

The Android system knows that you will potentially have a lot of Apps on your device, and wants to make it easy for them to talk to each other. So, it allows you to send and receive what are essentially requests for jobs to be done.

A job could get picked up by your App's own boss (the manifest) or another App.

When creating an **`Intent`**, it's up to you to either write it very generally to have the option of picking from several Apps to perform the job (*implicit*), or very specifically to follow a certain path (*explicit*). You'll see an example of each type of **`Intent`** later in this course.

For an immediate example, your App already has an **`Activity`** called **`MainActivity`**. Your manifest has it labeled with an intent filter that causes the **`MainActivity`** to launch when the user selects the App icon from their home screen. You could potentially move that filter to another **`Activity`** and then *that* activity would launch instead of **`MainActivity`**. Basically, the App does whatever the boss says.

If you don't fully grasp everything about **`Intents`** right away, don't worry. Just keep the concept in mind as you see **`Intents`** throughout the code, and eventually you will start to get an idea of their potential.