web@hosain:~$ About

Getting Started With Android Development Using Groovy 2.4 and Android Studio

Feb 7, 2015

Groovy 2.4 was released last month with native support for Android development.

In this post I'll explain how to set up a new Android Studio project with Groovy support as I couldn't find a decent tutorial to explain it for newcomers (like myself).

Why Groovy?

If you're new to Groovy, you may ask why would I want to do this?

In short: less code = (hopefully) less buggy and more readable code. Here are some of the benefits of using Groovy over Java:

- Syntax: with features like closures and dynamic typing, Groovy allows you to write much more readable code in fewer lines of code. Some examples:

```
ListView myListView = (ListView) findViewById(android.R.id.list)
/* Implicit typing - and no semicolons! */
def myListView = findViewById(android.R.id.list) as ListView


Bundle extras = getIntent().getExtras();
/* 'get' is optional in groovy for getSomething() methods */
def extras = intent.extras


['a', 'b', 'c'].each {
    println it
}
```

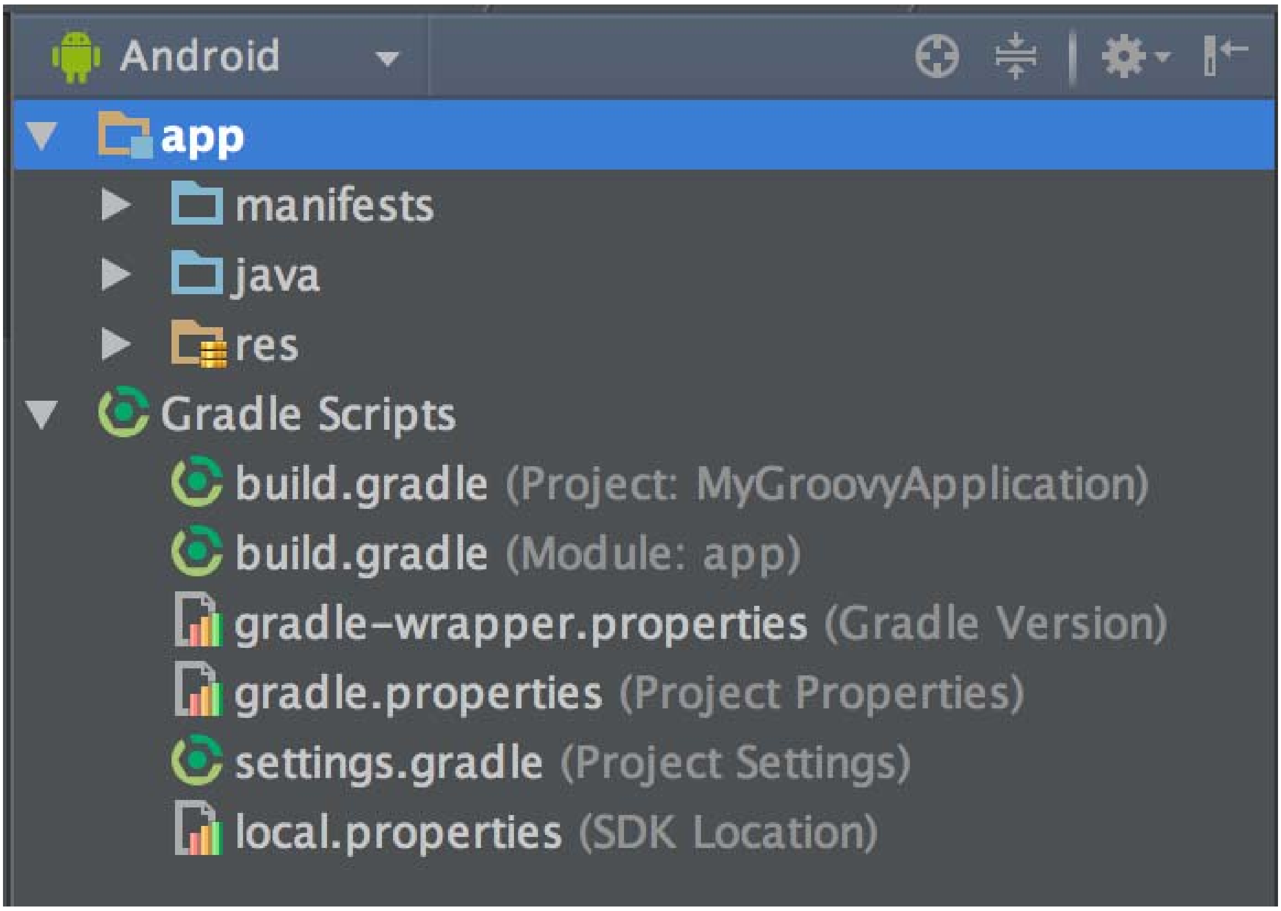
- Powerful collections: creating and manipulating lists / maps is very easy and powerful out of the box (each, findAll, collect, etc), which is a good way to shorten your code.
- Dynamic: Groovy's metaclassing capabilities allow you to extend or replace an

existing class behaviour if you like.

Setup

So let's get started by creating a new project using Android Studio. For this example I'm using the Basic Activity template with the default settings.

You should see the following file structure created:



Android

app
manifests
java
res
Gradle Scripts
build.gradle (Project: MyGroovyApplication)
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Version)
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)

The key files for us here are the two build.gradle files.

The first one is project-wide build.gradle which applies to all project modules. The second build.gradle for the app's main module.

Add Groovy dependencies

In the app module build.gradle, we need to make two changes:

- Apply the groovy-android plugin below the line "apply plugin: 'com.android.application'" - which adds support for the Groovy language in the Android Gradle toolchain.

```
apply plugin: 'groovyx.grooid.groovy-android'
```

- Add the grooid depedency - a special version of the Groovy compiler for Android:

```
 dependencies {
     //..
     compile 'org.codehaus.groovy:groovy:2.4.0:grooid'
 }
```

The final file should look similar to this:

```
apply plugin: 'com.android.application'
apply plugin: 'groovyx.grooid.groovy-android'

android {
    compileSdkVersion 21
    buildToolsVersion "21.1.2"

    defaultConfig {
        applicationId "net.hosain.mygroovyapplication"
        minSdkVersion 15
        targetSdkVersion 21
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:21.0.3'
    compile 'org.codehaus.groovy:groovy:2.4.0:grooid'
}
```

In the project build.gradle, add the plugin dependency:

```
classpath 'org.codehaus.groovy:gradle-groovy-android-plugin:0.3.5'
```

The final file should look like:

```
// Top-level build file where you can add configuration options common

buildscript {
    repositories {
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:1.0.0'
        classpath 'org.codehaus.groovy:gradle-groovy-android-plugin:0
        // NOTE: Do not place your application dependencies here; they
        // in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        jcenter()
    }
}
```

You should see a gradle sync message in Android Studio, click “Sync Now” and it should complete successfully.

Groovify it!

- The groovy plugin looks for files in app/src/main/groovy, so rename main/java to main/groovy (The folder name on the left hand side will remain ‘java’ due to the lack of full Groovy support in Android Studio).

At this point, your app should run fine! This is because you can mix and match groovy/java code.

- Rename all files in src/groovy/**/*.java to end with ‘.groovy’

Now you should be able to make use of Groovy across your application.

Let’s take MainActivity.groovy as an example: try removing semicolons at the end of each line and rerun the app!

Issues and tips

- As I mentioned before, Android Studio doesn’t fully support Groovy yet. For example you can’t right click and create a Groovy class. You would have to create an empty file ending with ‘.groovy’, and the annoying left hand side menu won’t update to source folder name from ‘java’.

- Annotate your classes with @CompileStatic - this is especially good if you’re just starting out or don’t need to use much of the dynamic stuff as it forces strict typing at compile time. It also helps greatly in performance.

```
@CompileStatic
class MainActivity extends ActionBarActivity {

}
```

- ProGuard rules: Add the following to your app/proguard-rules.pro - this helps in keeping your app APK size to the minimum and ensures some Groovy classes don’t get removed by ProGuard.

```
-dontobfuscate
-keep class org.codehaus.groovy.vmplugin.**
-keep class org.codehaus.groovy.runtime.dgm*
-keepclassmembers class org.codehaus.groovy.runtime.dgm* {
    *;
}

-keepclassmembers class ** implements org.codehaus.groovy.runtim
    *;
}

-dontwarn org.codehaus.groovy.**
-dontwarn groovy**
```

- Gradle build task fails due to lint errors:

  If you run ‘./gradlew clean build’ in your project’s root directory, it’ll fail with an InvalidPackage lint error - full error message.

  It seems like some of the base Groovy classes make calls to unsupported java code which Android doesn’t like.

  For now, you can suppress these errors by adding the following to the app module build.gradle inside the android block:

```
android {
    ...
    lintOptions {
        disable 'InvalidPackage'
    }
}
```

1 Comment Hosain's blog ● 1 Login ▾

Recommend Share Sort by Best ▾

Join the discussion…

erichelgeson • 4 months ago

Great article! You can integrate into InteliJ with the instructions here: http://www.mscharhag.com/2015/...

Keep posting about your journey with groovy and android, thanks!

^ | ˅ • Reply • Share ›