TESTING GRADLE PLUGINS

ANDREW REITZ

WHO AM I

- Android Developer @TheNerdey
- Maintainer of Android Groovy Gradle Plugin
- Twitter: @andrewreitz_
- GitHub: pieces029

WHY TEST?

So this doesn't happen to you.

WHY TEST (THE GROOVY ANDROID STORY)

Groovy Android Plugin has many dependencies

Java Version	Gradle Version	Android Build Tools Version
1.6	2.10	1.1.0
1.7	2.11	1.3.0
	2.12	1.5.0
	2.13	2.0.0
	2.14	2.1.2

TESTING YOUR PLUGIN WITH UNIT TESTS

STEP ONE WRITE UNIT TESTS

TESTING YOUR PLUGIN WITH UNIT TESTS

For classes that depend on Gradle you want to use ProjectBuilder.

```
@Rule TemporaryFolder dir

Project project

def setup() {
  project = ProjectBuilder.builder().withProjectDir(dir.root).build()
  project.pluginManager.apply(AppPlugin) // com.android.build.gradle.AppPlugin
  project.pluginManager.apply(GroovyAndroidPlugin) // my plugin
}
```

TESTING YOUR PLUGIN WITH UNIT TESTS

Testing small pieces of classes.

```
def "should set options on groovy compile"() {
  given:
  def groovyTask = project.tasks.create('Test Groovy Compile', GroovyCompile)

  project.androidGroovy {
    options { // must be explicit here as spock does not resolve like gradle
      project.configure(groovyTask.groovyOptions) {
        encoding = 'UTF-8'
        forkOptions.jvmArgs = ['-noverify']
      }
      sourceCompatibility = JavaVersion.VERSION_1_7
      targetCompatibility = '1.7'
    }
  }

  when:
  project.extensions.getByType(GroovyAndroidExtension).configure(groovyTask)
```

HOW TO FUNCTIONAL TEST

HOW TO FUNCTIONAL TEST

Add the dependencies

```
dependencies {
    testCompile gradleTestKit()
    // add junit, testNG, etc.
}
```

HOW TO FUNCTIONAL TEST

A simple project

```
@Rule final TemporaryFolder testProjectDir = new TemporaryFolder()
File buildFile

def setup() {
    buildFile = testProjectDir.newFile('build.gradle')
}

def "hello world task prints hello world"() {
    given:
    buildFile << """
        task helloWorld {
            doLast {
                println 'Hello world!'
            }
        }
    """

    when:
```

HOW TO FUNCTIONAL TEST

Let's get creative.

`projectLocalRepo.gradle`

```
plugins.withType(MavenPlugin) {
  ext.localRepoUrl = new File(rootProject.buildDir, 'localrepo').toURI()

  install {
    repositories {
      mavenDeployer {
        repository(url: localRepoUrl)
      }
    }
  }
}
```

HOW TO FUNCTIONAL TEST

build.gradle

```
apply plugin: 'maven'
apply from: "$rootDir/gradle/projectLocalRepo.gradle"

...

tasks.withType(Test) {
  dependsOn install
}
```

HOW TO FUNCTIONAL TEST

```
abstract class FunctionalSpec extends Specification {
  File getLocalRepo() {
    return new File('build/localrepo')
  }
}
```

HOW TO FUNCTIONAL TEST

```
buildFile << """
  buildscript {
    repositories {
      maven { url "${localRepo.toURI()}" }
      jcenter()
    }
    dependencies {
      classpath 'com.your.coolplugin:plugin:+'
    }
    ...
  }

  apply plugin: 'com.your.coolplugin'
  ...
"""
```

HOW TO FUNCTIONAL TEST

FunctionalSpec.groovy

```groovy
@Rule TemporaryFolder dir

File getBuildFile() {
  return makeFile('build.gradle')
}

File makeFile(String path) {
  def file = file(path)
  if (!file.exists()) {
    def parts = path.split('/')
    if (parts.size() > 1) {
      dir.newFolder(*parts[0..-2])
    }
    dir.newFile(path)
  }
  return file
}
```

TIPS AND TRICKS

Have a "Base" Functional Specification.

```
GradleRunner runner(String gradleVersion, String... args) {
    return GradleRunner.create()
        .withProjectDir(dir.root)
        .forwardOutput() // forward ouput to System.out
        .withArguments(args.toList())
        .withGradleVersion(gradleVersion?:GradleVersion.current().version)
  }
```

TIPS AND TRICKS

"Base" Functional Specification Cont.

```
BuildResult runWithVersion(String gradleVersion, String... args) {
  runner(gradleVersion, args).build()
}

BuildResult run(String... args) {
  runner(null, args).build()
}
```

TIPS AND TRICKS

Pass common easy to forget values to your tests.

```
tasks.withType(Test) {
  systemProperty 'androidPluginVersion', androidPluginVersion
  systemProperty 'buildToolsVersion', buildToolsVersion
  systemProperty 'compileSdkVersion', compileSdkVersion
}
```

TIPS AND TRICKS

Use your plugin in your project!

```
sourceSets {
  main {
    groovy.srcDirs = ['src/main/groovy', '../groovy-android-gradle-plugin/src/main/groovy']
    resources.srcDirs = ['src/main/resources', '../groovy-android-gradle-plugin/src/main/resources']
  }
}
```

REAL WORLD EXAMPLE

https://github.com/groovy/groovy-android-gradle-plugin

THANKS FOR COMING!

DO YOU HAVE ANY QUESTIONS?