Embracing Gradle

Practical build automation for the Android world

Avram Lyon
Scopely

AnDevCon Boston, 2014

March-July 2012

One title, one codebase

Debug / Release

Google / Amazon

Free / Paid

=

6 Builds

the road to gradle is paved with home-brew and make-do

this is Scopely's road there

`rake` and `ant`

- `goog-free-prod.yaml`:

```
app_name: Dice with Buddies Free
facebook_app_id: 12345678912
```

- `config/AndroidManifest.xml.erb`:

```
<application app:label=<%=app_name%> ...>
```

- `config/strings.xml.erb`:

```
<string name="facebook_app_id"><%=facebook_app_id%></string>
```

```
$ rake configure env=goog-free-prod
    + config/goog-free-prod.yaml
      config/amzn-paid-prod.yaml
 config/AndroidManifest.xml.erb  => AndroidManifest.xml
 config/strings.xml.erb          => res/values/strings.xml
$ ant clean release
```

why use one build tool when you can use two?

`ant-raker`

1. Worked
2. `rake` and `ant` experts certainly would have done better
3. i18n… Not really.
4. Dependency management? What's that?
5. All builds had assets and jars for all versions
6. IDE treated as standard Ant-style project
    - But it wouldn't build until you `raked`

More builds

January 2013

- More codebases
  1. Android-Core (90%+ of code)
  2. Dice-Android (assets, code tweaks)
  3. MiniGolf-Android (+ more)
- Unity components!
- More developers!

Complication multiplies!

```
# generate templated files (Manifest, resources, etc.)
rake configure env=goog-free-prod
# Build Unity *.so's and assets, copy them into the Android project
rake unity unzipUnity
# Build the game APK
ant release
```

- Versioning specified as argument to `rake configure`
- `build.xml` fairly standard, with addition of simple package renaming task

The Good

- Encapsulated magic in `rake configure` and `rake unity`
- Arbitrary Ruby code in templates

The Bad

- Encapsulated magic in `rake configure` and `rake unity`
- Arbitrary Ruby code in templates
- Ruby? How did that happen?
- See previous slides.

May 2013

Three titles

Debug / Release

Google / Amazon

Free / Paid

=

18 Builds

Google I/O 2013
Gradle + Android Studio

Decision to Move

1. Flavors
2. Flavors
3. Testing for library projects
4. Flavors
5. Extensibility
6. Dependency management
7. Embrace the future
8. Flavors

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="<%=@gamepackage%>"
    android:versionCode="<%=@versionCode%>"
    android:installLocation="auto"
    android:versionName="<%=@version%>" >

    <uses-sdk android:minSdkVersion="8" android:targetSdkVersion="17" />
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.READ_CONTACTS" />
    <uses-permission android:name="android.permission.SEND_SMS" />
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <!-- Start game-specific permissions -->
<% ERB.new(open(@merge_permissions).read(), nil, nil, "@mergeperm").result(binding)  %>
<%=@mergeperm%>
    <!-- End game-specific permissions -->

    <% if @store == "GoogleCheckout" %>
      <uses-permission android:name="com.android.vending.BILLING" />
      <uses-permission android:name="com.android.vending.CHECK_LICENSE" />

      <!-- Used for GCM, a Google-only service -->
      <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
      <uses-permission android:name="android.permission.GET_ACCOUNTS" />
      <uses-permission android:name="android.permission.WAKE_LOCK" />

      <permission android:name="<%=@package%>.permission.C2D_MESSAGE" android:protectionLev
    <uses-permission android:name="<%=@package%>.permission.C2D_MESSAGE" />
    <% end %>
```

Typical values

```yaml
# Game-defining settings
version: 2.8.0
game: Dice
gamepackage: com.withbuddies.dice
gameurlscheme: dwb

# Build settings
merge_activities: config/AndroidManifest.merge_activities.erb
merge_permissions: config/AndroidManifest.merge_permissions.erb
intent_filter_overrides: []
```

`amzn-free-test.yaml`

```yaml
package: com.withbuddies.dice.free
store: AmazonMarketplace
```

(it keeps on going)

Converting to Gradle equivalents

We didn't actually convert everything immediately.

Gradle starts with sane, opionionated configuration by convention

```
/src/main/java
/src/main/res
/src/main/assets
/src/main/AndroidManifest.xml

/src/androidTest/java
/src/androidTest/res
...
```

Ant convention

```
/project/src
/project/res
/project/assets
/project/AndroidManifest.xml

/testProject/src
/testProject/res
/testProject/AndroidManifest.xml
```

Make Gradle bend to Ant's will

```
android {
  sourceSets {
        main {
            manifest {
                srcFile 'project/AndroidManifest.xml'
            }
            java {
                srcDirs = ['library/src']
            }
            res {
                srcDirs = ['library/res']
            }
            assets {
                srcDirs = ['library/assets']
            }
            // --snip--
        }

        androidTest {
          // no manifest
          java {
            srcDirs = ['test/src']
          }
          res {
            srcDirs = ['test/res']
          }
        }
    }
```

Custom package name

```
android {
    flavorGroups "store", "bundle", "config"

    productFlavors {
        free {
            flavorGroup "bundle"
            packageName 'com.withbuddies.dice.free'
        }
        paid {
            flavorGroup "bundle"
            packageName 'com.withbuddies.dice'
        }
        ...
    }
}
```

Build with: `./gradlew installFreeRelease`

But... GCM is broken!

Manifest fragments

`src/free/AndroidManifest.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android">
    <application>
        <receiver android:name="com.withbuddies.core.push.GCMReceiver" android:permission="
            <intent-filter>
                <action android:name="com.google.android.c2dm.intent.RECEIVE" />
                <action android:name="com.google.android.c2dm.intent.REGISTRATION" />
                <category android:name="com.withbuddies.dice.free" />
            </intent-filter>
        </receiver>
    </application>

    <permission android:name="com.withbuddies.dice.free.permission.C2D_MESSAGE" android:pro
    <uses-permission android:name="com.withbuddies.dice.free.permission.C2D_MESSAGE" />
</manifest>
```

`src/paid/AndroidManifest.xml` (trimmed out some detail):

```
    <category android:name="com.withbuddies.dice" />
    <permission android:name="com.withbuddies.dice.permission.C2D_MESSAGE" android:protecti
    <uses-permission android:name="com.withbuddies.dice.permission.C2D_MESSAGE" />
```

Intelligent "new" manifest merging

Optional as of Android plugin 0.10

```
android {
  useOldManifestMerger false
}
```

`src/main/AndroidManifest.xml`:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android">
    <application>
        <receiver android:name="com.withbuddies.core.push.GCMReceiver" android:permission="
            <intent-filter>
                <action android:name="com.google.android.c2dm.intent.RECEIVE" />
                <action android:name="com.google.android.c2dm.intent.REGISTRATION" />
                <category android:name="${packageName}" />
            </intent-filter>
        </receiver>
    </application>

    <permission android:name="${packageName}.permission.C2D_MESSAGE" android:protectionLeve
    <uses-permission android:name="${packageName}.permission.C2D_MESSAGE" />
</manifest>
```

Information: http://tools.android.com/tech-docs/new-build-system/user-guide/manifest-merger

Amazon vs. Google

```
android {
    flavorGroups "store", "bundle", "config"

    productFlavors {
        goog {
            flavorGroup "store"
        }
        amzn {
            flavorGroup "store"
        }
        ...
    }
}
```

Amazon vs. Google

`src/goog/AndroidManifest.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android">
      <uses-permission android:name="com.android.vending.BILLING" />
      <uses-permission android:name="com.android.vending.CHECK_LICENSE" />

      <!-- Used for GCM, a Google-only service -->
      <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
      <uses-permission android:name="android.permission.GET_ACCOUNTS" />
      <uses-permission android:name="android.permission.WAKE_LOCK" />
</manifest>
```

Because Amazon will reject you if you have the GCM or billing permissions in your manifest.

How do I know what build I am?

Old school: Set value in generated properties file or resource file

```
DEBUG=<%= @mode == "test" %>
BUNDLE=<%=@bundle%>   # dicefree or dicepaid
MODE=<%=@mode%>
TARGET=<%=@target%>
```

Likely use case is controlling ads or premium features.

Slightly newer school: Override a resource

- `src/main/res/values/configuration.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="configuration_bundle">dicefree</string>
</resources>
```

- `src/paid/res/values/paid_configuration.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="configuration_bundle">dicepaid</string>
</resources>
```

- `src/free/res/values/free_configuration.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="configuration_bundle">dicefree</string>
</resources>
```

Newest school: Key off `BuildConfig`

Remember old `BuildConfig.DEBUG`? It has family.

```
public static final String BUILD_TYPE = "debug";
public static final String FLAVOR = "googFree";
public static final String FLAVOR_store = "goog";
public static final String FLAVOR_bundle = "free";
```

This also means no overhead retrieving from properties files or resources.

Other uses

```
def buildTime = new Date().format("yyyy-MM-dd'T'HH:mm'Z'", TimeZone.getTimeZone("UTC"))

android {
    defaultConfig {
          buildConfigField "String", "BUILD_TIME", "\"$buildTime\""
    }
}
```

Versioning

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="<%=@gamepackage%>"
    android:versionCode="<%=@versionCode%>"
    android:installLocation="auto"
    android:versionName="<%=@version%>" >
```

Version Code and Version

```
android {
  versionCode getVersionCode()
  versionName getVersionName()
}
```

Getting groovy with versions

```
def getVersionName = { ->
    return project.hasProperty('versionName') ? versionName : null
}

 /*
  * Make from version name and build number
  */
def versionCodeFromVersionName = { ->
    if (project.hasProperty('versionName')) {
        String[] parts = versionName.tokenize(".");
        int[] numbers = new int[parts.length + 1];
        for (int i = 0; i < parts.length; i++)
            numbers[i] = Integer.parseInt(parts[i].trim());
        if (project.hasProperty('buildNumber')) {
            numbers[parts.length] = Integer.parseInt(buildNumber) % 10000;
        }
        int number = 0;
        number += numbers[0] * 1000000
        number += numbers[1] * 100000
        number += numbers[2] * 10000
        number += numbers[3]
        return number
    } else {
        return -1
    }
}
```

Getting groovy with versions

Properties can be set in a properties file:

`gradle.properties`:

```
versionName=3.1.4
```

Or on the command line:

```
./gradlew installGoogFreeRelease -PbuildNumber=35
```

Multiproject builds

- `./build.gradle` – Frequently empty, unless specifying defaults.
- `./settings.gradle` – Enumeration of all projects that will be used with `compile project(":Core-Project")`

"Project" dependencies

```
include 'ads-sdk'
include 'Core-Project'
include 'Game1'
include 'anotherDirectory/anotherProject'
```

Argument to `include` must be the path to the dependency's `build.gradle`

Typical defaults for parent project

```
subprojects {
    buildscript {
        repositories {
            mavenCentral()
        }
        dependencies {
            classpath 'com.android.tools.build:gradle:0.10.2'
        }
    }

    apply plugin: 'android'

    compileSdkVersion 19

    buildToolsVersion "19.1.0"

    // Configures the dexer. Our projects frequently exhaust the default heap allocated to
    dexOptions {
        incremental true
        javaMaxHeapSize "4g"
        preDexLibraries = "true".equals(System.getProperty("pre-dex", "true"))
    }

    defaultConfig {
        minSdkVersion 14
        targetSdkVersion 19
    }
}
```

Pulling code into a "plugin"

```
apply from: '../client-build-scripts/Android/Universal/base_build.gradle'
```

`base_build.gradle`:

```
task assembleFiles(type: Copy, dependsOn: )
    from(jar.outputs.files) {
        into 'lib'
    }

    from(configurations.runtime) {
        into 'lib'
    }

    from('build/scripts') {
        into 'bin'
    }

    from("$rootProject.projectDir/templates") {
        into 'scripts'
        include 'upstart-template.conf.template'
        rename { file -> "${project.name}.conf" }
        expand(serviceName: project.name)

    }
}
```

More on Groovy templating (optional)

(Non-Android example)

```
# Upstart service configuration
# Generated from /templates/upstart-template.conf.template

description "titan-${serviceName}"

start on filesystem and started networking
stop on shutdown

script
    export HOME="/opt/scopely"

    echo \$\$ > /var/run/${serviceName}.pid
    exec sudo -u ubuntu \$HOME/${serviceName}/bin/${serviceName} >> /var/log/${serviceName}
end script

pre-start script
    echo "[`date -u +%Y-%m-%dT%T.%3NZ`] (sys) Starting" >> /var/log/${serviceName}.sys.log
end script

pre-stop script
    rm /var/run/titan-${serviceName}.pid
    echo "[`date -u +%Y-%m-%dT%T.%3NZ`] (sys) Stopping" >> /var/log/${serviceName}.sys.log
end script
```

A real plugin

See https://github.com/ajlyon/unity-gradle-plugin

Plugin docs:

See http://www.gradle.org/docs/current/userguide/custom_plugins.html

Plugins you don't need to write

- `android-sdk-manager` @ `'com.jakewharton.sdkmanager:gradle-plugin:0.10.+'`

Automatically downloads the specified versions of Android and build tools. Perfect for CI.

- `android-test` @ `'org.robolectric.gradle:gradle-android-test-plugin:0.10.0'`

Runs Robolectric tests on the JVM. Run by Pivotal, which took over from Jake Wharton.

Gradle Support

IDEs

1. IntelliJ / Android Studio
2. Eclipse. Not worth trying at present; can grok the `java` plugin but not `android`

Build Servers

First class support:

1. TeamCity
2. Jenkins (+ Gradle plugin, of course)

Things move fast -- wait a bit before bumping versions of IDE or Gradle.

Future directions for Gradle

1. NDK development
2. Gradle and LibGDX: http://www.badlogicgames.com/wordpress/?p=3336
3. Gradle for C#
    - Defunct project by Unity3d: https://github.com/Unity-Technologies/kaizen
4. Gradle for Obj-C

Further reading:

- The manual, but it occasionally is a bit behind: http://tools.android.com/tech-docs/new-build-system/user-guide
- Every new version. Read the release notes carefully.
- Xavier Ducrohet on Google+: https://plus.google.com/+XavierDucrohet/posts
- Google Developer Tools on Google+: https://plus.google.com/communities/114791428968349268860
- adt-dev community on Google Groups: https://groups.google.com/forum/#!forum/adt-dev

Questions?

Feedback

eventmobi.com/andevconboston

Me

@ajlyon
ajlyon@gmail.com
Scopely is hiring!