Taking advantage of Apache maven for your Android builds

Manfred Moser

simpligility technologies inc.
http://www.simpligility.com
@simpligility

simpligility technologies inc

Wanted:

Your Feedback!

simpligility technologies inc

Audience Background

Android development

Eclipse

Apache Ant

Apache Maven

simpligility technologies inc

About Manfred

- Android application developer and consultant
- Author Maven: The Complete Reference, The Hudson Book
- Core Committer on Android Maven Plugin
- Author of Maven Android SDK Deployer
- Vancouver Island Java User Group Founder/Leader

Agenda

Motivation

Brief Apache Maven introduction

Maven Android tool chain with examples

Lots of tips and tricks

Q & A

Building Android Apps Currently

Eclipse ADT

Or

Apache Ant based build

Whats wrong with that?

You don't like Eclipse or Ant or don't want to depend on it for build.

You need additional features in the build.

You want command line and continuous integration server usage.

You need to work with multiple dependencies...

You tried extending the Ant scripts...

Example Dependencies

Make “libs” folder and copy jars files into it.

Check them into version control.

Upgrade means replace jar file and transitive dependencies.

Example Dependencies

Known to be a hassle from years of Apache Ant usage on Java projects:

- Unknown dependencies
- Transitive dependencies
- No documentation
- No collision detection
- Large project checkout

Led to Apache Maven, Apache Ivy ...

Introduction to Apache Maven

"Software project management
and comprehension tool"

Builds your software and much more

De-facto standard for Java software builds

Convention over configuration (like RoR)

Declarative rather than procedural

Maven and Android

Android Maven Plugin – core tool

M2Eclipse, m2e-android – for Eclipse users

Maven Android SDK Deployer – for SDK libraries use

Android4Maven – all optional for setup

Maven Android Archetype – project blueprints

Maven RIndirect Plugin – for reuse/library setup

Maven Installation

Unzip, create M2_HOME, add to PATH

apt-get install maven2

port install maven3

and then just run

mvn -version

Needs to be 3.0.3+

Getting started

Convert your existing app – add pom.xml file

Create new project – use command line or IDE wizard to create normal Android project, then add pom.xml

Use Maven Android Archetype – on command line or in IDE wizard

Hello Flash Example – pom.xml

Packaging apk

Android dependency

Java source folder

Android Maven Plugin configuration

Doing a build

mvn clean install

Invokes the **clean** and the **install**

build life-cycle phase

Phases

- predefined order of tasks for build
- what happens is defined in pom (+ super-pom)
- additional plugin goals can added

pre-clean, clean, post-clean
validate, generate-sources, process-sources, generate-resources, process-resources, compile, process-classes, generate-test- sources, process-test-sources, generate-test-resources, process- test-resources, test-compile, test, prepare-package, package, pre-integration-test, integration-test, post-integration-test, verify, install, deploy

Deploy apk

mvn android:deploy

Invokes the
deploy **goal**
of the
android maven plugin

Goals

different per plugin
behaviour can be defined in pom
e.g. mvn install:install

but does not have to be

mvn archetype:create ... mvn install:file ....
pass parameters in with -Dparameter=value

Plugins

Maven itself does nearly nothing
– just a IOC container (using Guice)

Super POM defines standard configuration
which can do a LOT

Lots of plugins available at apache, codehaus and beyond

Maven Invocation

mvn [options] [<goal(s)>] [<phase(s)>]

- Options – get list with mvn -h

- Goals – with syntax Plugin:PluginGoal
- e.g. mvn android:deploy

- Phases – e.g. clean compile test install package
- e.g. mvn clean install

Downloading the internet? Not!

Core install of Apache Maven very small (3mb)

What is needed gets downloaded into repository..

Repository

Storage for plugins and artifacts

Local user repository ~/.m2/repository

Maven central

Other public repositories

Proxy repository server

Recap

pom.xml

Default build life-cycle with phases

Plugins with goals

Repositories

Now:

Use Cases
For
Android Development

Using external dependencies

Add dependency

Everything else happens “automagically”

Roboguice – Astroboy example

Other external dependencies

Google Guice IoC
KSOAP2-Android for SOAP parsing
Jackson JSON Parser
Robolectric
Robotium
GoogleAPI's
and so on and so on

How did it find android.jar?

Deployed to Maven Central repository

Built from AOSP with Android4Maven

Or

Deployed to repository with
Maven Android SDK Deployer

Google Maps jar

Can't be in Maven central

Use Maven Android SDK Deployer

Copies jars from SDK install to repository

Same for usb.jar, compatibility libraries...

Maps Dependency

```
<dependency>
  <groupId>com.google.android.maps</groupId>
  <artifactId>maps</artifactId>
  <version>7_r1</version>
  <scope>provided</scope>
</dependency>
```

Dependency Scopes

Default = compile

provided

test

Working with the Android emulator

mvn android:emulator-start

mvn android:emulator-stop

mvn android:emulator-stop-all

With the parameters
android.emulator.avd
android.emulator.wait
android.emulator.options

Configuring the Plugin

Using the plugin configuration in

- pom.xml
- settings.xml

```
<plugin>
    <groupId>com.jayway.maven.plugins.android.generation2</groupId>
    <artifactId>android-maven-plugin</artifactId>
    <configuration>
        <sdk>
            <platform>8</platform>
        </sdk>
        <emulator>
            <avd>22</avd>
        </emulator>
```

Configuring the Plugin

Using properties

- in Maven files

```
<properties>
    <android.emulator.avd>22</android.emulator.avd>
</properties>
```

- on command line

```
mvn android:emulator-start -Dandroid.emulator.avd=22
```

Device Interaction

android.device parameter

usb, emulator or serialnumber

Works against all adb attached devices!

- deploy, undeploy and redeploy
- run
- pull and push
- instrument

Demo

Install and Run Apk

Goals

- deploy
- undeploy
- redeploy
- run

undeployBeforeDeploy

remove old version of application before testing

Deploy any apk

- Deploy an application as part of e.g. preparing avd for ci build
  - Droidreader for PDF testing
  - OpenOffice document reader for odf
- Example droidreader apk install (android.file parameter)

Pull and Push

android:pull and android:push goals

Works against all attached devices

Upload database or image resources to device or emulator avd

Download performance .trace files

Testing

Unit & Integration Testing

Various tools Junit, TestNG, EasyMock, Robolectric, Robotium ...

All as part of build

Tests run by default

-DskipTests or -Dmaven.test.skip=true

Instrumentation Testing

SDK Instrumentation as well as Robotium

Automatically runs on all attached devices

Produces junit xml reports per device

AndroidManifest Changes

manifest-update goal

Support for

- manifest.versionName
- manifest.versionCode
- manifest.versionCodeAutoIncrement
- manifest.versionCodeUpdateFromVersion
- manifest.sharedUserId
- manifest.debuggable

Use in release profile or manually

See Morseflash example

Maven resource filtering

Define properties or use an existing one

Define filtered resources

Embed properties for replacement

Can also be used to exclude files (e.g. SVG source for images)

Development vs Production

Resource filtering with different values for properties

Depending on build time switch (e.g. automatic or provided at invocation)

Uses profiles

Morseflash example

Profiles

Separate build definition
e.g. properties
but also
plugin configuration/execution/...

Activation via switches, files, OS...

Releasing Application

Perform release with Maven Release Plugin

Will handle scm operations, version changes...

Take advantage of repo server for artifact storage

MorseFlash Example

Sign apk

Maven Jarsigner Plugin

keys, password ... settings.xml

Morseflash example

Zipalign

Android Maven Plugin

zipalign goal

Needs to be bound to lifecycle phase

Morseflash example

Proguard

Obfuscate, shrink and optimize

Use Proguard Maven Plugin

Only in release profile

Morseflash example

Proguard path configuration

rt.jar/classes.jar and jsse.jar

different path/filename for different OS

Morseflash example

Native Application Build

Fully supported with Android NDK install

Various examples as part of Android Maven Plugin Samples – native

Scala

special use of a library jar

enables Scala programming language

scala plugin for compilation

proguard for shrinking

Scala example

Screen size, platform level

emulator-start, emulator-stop

different profiles

different parameters for deploy

deploy to different emulators/devices

Produce project website

Maven site plugin with javadoc and so on

Sonar for historical trending and more

Static code analysis

As part of build

Findbugs
PMD
Checkstyle

Reuse projects

Plain java library project from server side application

Android projects – dependency type “apklib”

Android projects – Maven RIndirect Plugin

Android library projects – with r14 jar files

Android NDK .so files – as normal dependency

Efficient usage patterns

- IDE integration in Eclipse, IntelliJ and Netbeans
- Command line alias
  - alias mci='mvn clean install'
  - alias mct='mvn clean test'
  - alias mcis='mvn clean install -Dmaven.test.skip=true'
  - alias mad='mvn android:deploy'
  - alias mau='mvn android:undeploy'
  - alias mar='mvn android:run'
- Bash command line completion
- Run CI server like Hudson in background

Related Tools

- Maven Android SDK Deployer – maps jars, usb.jar, compatiblity jars... into repository or repository server
- Android4Maven – builds jars in Maven central

Your Android build needs?

Easy to script more features (Ant, Groovy,...)

All open source tools

We take pull requests!

Maven Resources

Apache Maven
http://maven.apache.org

Maven: The Complete Reference
and other free books
http://www.sonatype.com/Support/Books

M2 Eclipse
http://eclipse.org/m2e/

Maven Bash Completion
https://github.com/juven/maven-bash-completion

Android Maven Resources

Maven Android Developers Mailing List

http://groups.google.com/group/maven-android-developers

Android Maven Plugin

http://code.google.com/p/maven-android-plugin/

M2 Eclipse Android Integration

http://rgladwell.github.com/m2e-android/

Maven: The Complete Reference

Android Chapter

http://www.sonatype.com/index.php/Support/Books/Maven-The-Complete-Reference
http://www.sonatype.com/books/mvnref-book/reference/android-dev.html

Android Maven Resources

Android4Maven
http://sourceforge.net/projects/android4maven/

Android Maven Archetypes
https://github.com/akquinet/android-archetypes/wiki

Maven RIndirect Plugin
http://www.github.com/akquinet/android-rindirect

Examples

Android Maven Plugin Samples
http://code.google.com/p/maven-android-plugin/wiki/Samples

RoboGuice
http://roboguice.org

Robotium
http://robotium.org

Robolectric
http://robolectric.org

ActionBarSherlock
http://actionbarsherlock.com/

Some more examples at
http://code.google.com/p/maven-android-plugin/wiki/PeopleUsingThis

and many many more

Summary

Proven, ready alternative

More flexibility and power out of the box

Better for test driven development
and continuous integration

Better reuse of components
and therefore in team environment

You Want More...

Fireside chats
about tablet development and markets
Monday evening

Free signed copy
Maven: The Complete Reference
Tuesday, 3pm

The End

Thank you for your attention.

Contact me for consulting
and open source hacking

@simpligility

manfred@simpligility.com

http://simpligility.com