



Enter The Gradle

Ken Sipe - @kensipe

www.11.jdd.org.pl

About Speaker

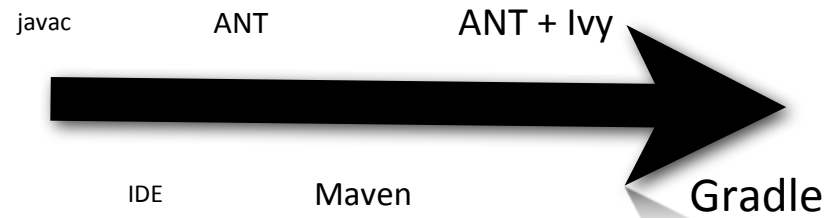


<http://kensipe.blogspot.com/>
<http://del.icio.us/kensipe>
twitter: @kensipe
ken.sipe@gmail.com

Developer: Embedded, C++, Java, Groovy, Grails, C#, Objective C
Speaker: **JavaOne 2009 Rock Star**, NFJS, JAX
Microsoft MCP
Sun Certified Java 2 Architect
Master of Scrums
Agile Coach
Instructor: VisiBroker CORBA
Rational Rose, OOAD



www.11.jdd.org.pl



What is Gradle?



- Build **DSL** written in Groovy
- Built on top of **Ant + Ivy**
- Apache 2 License
- Uses Groovy AntBuilder
 - ant.compile, ant.jar
- Plugins define common tasks to build different types of projects
 - java, groovy, war, ...

Gradle is Declarative



Specify **what...**

...not how

www.11.jdd.org.pl

Gradle Benefits



- Convention over Config
 - project layout is maven
- Flexible
 - Many artifacts per project
- Incremental Builds
- Rich Object Model
- Groovy

www.11.jdd.org.pl

Gradle Benefits



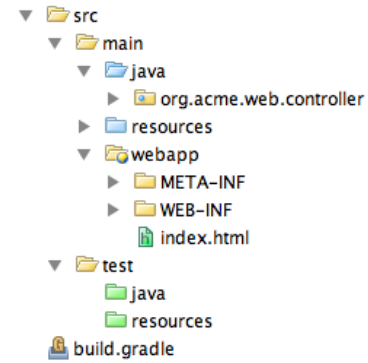
- Convention over Config
 - project layout is maven
- Flexible
 - Many artifacts per project
- Incremental Builds
- Rich Object Model
- Groovy (**NOT** XML)

www.11.jdd.org.pl

Getting Started



- Project Name
 - Project Directory
- Project Structure
 - Maven Style
- Build.gradle



User Tasks

Create a task



```
task hello
```

```
task hello << {  
    println 'Hello world!'  
}
```

```
task intro(dependsOn: hello) << {  
    println "I'm Gradle"  
}
```

```
project.tasks.add('someTask').doFirst {  
    // do something  
}
```

DSL Syntax And Tasks



```
task hello << { println 'Hello' }  
// direct API access is fine for single statements  
hello.dependsOn otherTask  
// for multiple access we prefer closure syntax  
hello {  
    onlyIf { day == 'monday' }  
    dependsOn otherTask  
}  
// combining Configuration and Actions  
task hello {  
    onlyIf {  
        day == 'monday'  
    }  
    doFirst {println 'Hello'}  
}
```

Task Types



```
task wrapper(type: Wrapper) {  
    gradleVersion = '1.0-milestone-3'  
}
```

Applying Plugins



- Any gradle script can be a plugin.
- Binary plugins must be in the build script classpath
 - can have id's (meta properties in the jar).

```
apply from: 'otherScript.gradle'  
apply from: 'http://mycomp.com/otherScript.gradle'
```

```
apply plugin: org.gradle.api.plugins.JavaPlugin  
apply plugin: 'java'
```

Standard Gradle Plugins



Plugin-Id	applies
base	
java-base	base
groovy-base	java-base
groovy	groovy-base
scala-base	java-base
scala	scala-base
war	java
osgi	
code-quality	
maven	
eclipse	

Sample Simple Build File



```
apply plugin:'war'  
version = 0.1  
  
repositories {  
    mavenCentral()  
}  
  
dependencies {  
    compile "commons-lang:commons-lang:2.4"  
}
```

Demo

simple java project, tasks, github

Beyond the Basics

Free yourself from the **XML**

Non Declarative



```
version = "1.0-${new Date().format('yyyyMMdd')}"
```

Extensible Object Model



```
tasks.withType(Jar).allObjects { jarTask ->
    jarTask.osgi = new DefaultOsgiManifest()
    jarTask.doFirst { task ->
        importOsgiManifestIntoManifest(task) }
}
```

Rich and Extensible API



```
tasks.withType(Jar).allObjects { jarTask ->
    jarTask.manifest.mainAttributes(Provider: "CodeMentor Inc.")
}
```

Rich and Extensible API



```
configure(nonWebProjects()) {  
    jar.manifest.attributes  
        Implementor: 'Code Mentor'  
}  
  
def nonWebProjects() {  
    subprojects.findAll {project ->  
        !project.name.startsWith('web')  
    }  
}
```

Jump Between Phases



```
task release(dependsOn: assemble) << {
    println 'We release now'
}

build.taskGraph.whenReady { taskGraph ->
    if (taskGraph.hasTask(':release')) {
        version = '1.0'
    } else {
        version = '1.0-SNAPSHOT'
    }
}
```

Gradle Dependencies

Runtime Dependencies



```
dependencies {  
  runtime group: 'org.springframework', name: 'spring-core', version: '2.5'  
  runtime 'org.springframework:spring-core:2.5', 'org.springframework:spring-aop:2.5'  
}
```

Separate Classpaths



```
dependencies {  
    compile 'org.springframework:spring-webmvc:3.0.0.RELEASE'  
  
    testCompile 'org.springframework:spring-test:3.0.0.RELEASE'  
    testCompile 'junit:junit:4.7'  
}
```

Transitive



Options 1: Everything

```
configurations.compile.transitive = true

dependencies {
    compile 'org.springframework:spring-webmvc:3.0.0.RC2'

    testCompile 'org.springframework:spring-test:3.0.0.RC2'
    testCompile 'junit:junit:4.7'
}
```

Options 2: Selective

```
runtime('org.hibernate:hibernate:3.0.5') {
    transitive = true
}
runtime group: 'org.hibernate', name: 'hibernate', version: '3.0.5', transitive: true
runtime(group: 'org.hibernate', name: 'hibernate', version: '3.0.5') {
    transitive = true
}
```

File Dependencies



```
dependencies {  
    runtime files('libs/a.jar', 'libs/b.jar') runtime  
    fileTree(dir: 'libs', includes: ['*.jar'])  
}
```

Naming Dependencies



```
List groovy = ["org.codehaus.groovy:groovy-all:1.5.4@jar",  
"commons-cli:commons-cli:1.0@jar",  
"org.apache.ant:ant:1.7.0@jar"]  
List hibernate = ['org.hibernate:hibernate:3.0.5@jar',  
'somegroup:someorg:1.0@jar']  
  
dependencies {  
    runtime groovy, hibernate  
}
```

Maven Repos



```
repositories {  
  mavenCentral()  
}
```

Or

```
repositories {  
  mavenCentral name: 'single-jar-repo',  
  urls: "http://repo.mycompany.com/jars"  
  mavenCentral name: 'multi-jar-repos',  
  urls: ["http://repo.mycompany.com/jars1", "http://repo.mycompany.com/jars1"]  
}
```

Or

```
repositories {  
  mavenRepo urls: "http://repo.mycompany.com/maven2"  
}
```

Flat File Repo



```
repositories {  
  flatDir name: 'localRepository',  
  dirs: 'lib' flatDir dirs: ['lib1', 'lib2']  
}
```

Demo

Jetty WAR Project, Wrapper

Common Interests

Bootstrap Gradle



- Jars can be added to the buildscript classpath
 - Custom build logic
 - Plugins
 - Helper classes (e.g. commons-math)

```
buildscript {  
    repositories { mavenCentral() }  
    dependencies {  
        classpath "commons-lang:commons-lang:3.1"  
        classpath files('lib/foo.jar')  
    }  
}
```

Implicit Depends



21 - 22 Listopada 2011, Kraków

```
build.gradle
1  apply plugin: 'java'
2
3  task zip(type: Zip) {
4
5      from jar.outputs.files
6
7      from('scripts/') {
8          fileType = 0755
9          include '**/runODP.sh'
10         include '**/runODP.bat'
11     }
12     from('lib/') {
13         include '**/*.jar'
14         into('lib')
15     }
16     from('.') {
17         include 'odp.config'
18     }
19 }
```

Line: 1 Column: 21 Groovy Tab Size: 4

Test Task Example



21 - 22 Listopada 2011, Kraków

Disables Auto Detection

```
test {
  jvmArgs: ["-Xmx512M"]
  include "**/tests/special/**/*Test.class"
  exclude "**/Old*Test.class"
  forkEvery = 30
  maxParallelForks = guessMaxForks()
}

def guessMaxForks() {
  int processors =
    Runtime.runtime.availableProcessors()
  return Math.max(2, (int) (processors / 2))
}
```

Ant Tasks



- Gradle provides an instance of the Groovy AntBuilder

```
ant.delete dir: 'someDir'
ant {
  ftp(server: "ftp.comp.org", userid: 'me', ...) {
    fileset(dir: "htdocs/manual") {
      include name: "**/*.html"
    }
    // high end
    myFileTree.addToAntBuilder(ant, 'fileset')
  }
  mkdir dir: 'someDir'
}
```

Importing Ant Builds



build.xml

```
<project>
  <target name="hello" depends="intro">
    <echo>Hello, from Ant</echo>
  </target>
</project>
```

build.gradle

```
ant.importBuild 'build.xml'
hello.doFirst { println 'Here comes Ant' }
task intro << { println 'Hello, from Gradle' }
```

```
>gradle hello
Hello, from Gradle
Here comes Ant
[ant:echo] Hello, from Ant
```

Integration with Maven



- Integration with Maven repositories
 - autogeneration of pom.xml
 - install to local Maven repo
 - deploy to any remote Repo
 - full maven metadata generation
- Integration of Maven builds in the future

Mult-Project

Configuration Injection



- **ultimateApp**

- api
- webservice
- shared

```
subprojects {
    apply plugin: 'java'
    dependencies {
        compile "commons-lang:commons-lang:3.1"
        testCompile "junit:junit:4.4"
    }
    test {
        jvmArgs: ['Xmx512M']
    }
}
```

Project Dependencies



- ultimateApp
 - api
 - webservice
 - shared

```
dependencies {  
  compile "commons-lang:commons-lang:3.1",  
  project(':shared')  
}
```

First Class Citizen

Settings File



- `settings.gradle`
 - location defines root
- root project is implicitly included

```
include 'project1', 'project2', 'project2:child1'  
  
// Everything is configurable  
rootProject.name = 'main'  
project(':project1').projectDir = '/myLocation'  
project(':project1').buildFileName =  
    'project1.gradle'
```



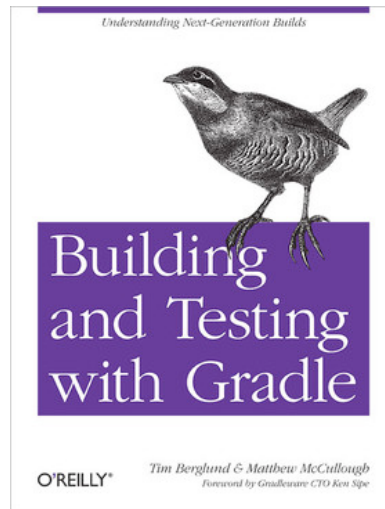
Demo

Multi-Project

www.11.jdd.org.pl



Gradle Book



www.11.jdd.org.pl



■ Resources

- <http://forums.gradle.org>
- <http://gradleware.com>

- Road Map
 - <http://www.gradle.org/roadmap>

www.11.jdd.org.pl

Q&A



- Gradle: The Groovy Way to Build
- Ken Sipe
 - ken.sipe@gmail.com
 - kensipe.blogspot.com
 - twitter: @kensipe