Apache Maven: Best Practices

Brett Porter - brett@apache.org

http://www.devzuz.org/blogs/bporter

Leading the Wave
of Open Source

Maven without the PAIN

- Sometimes unpleasant
- It’s for your own good!
- Can avoid or alleviate the problems

Who am I?

Apache Maven developer since 2003

Started the Archiva project

Co-author of Better Builds with Maven

Co-founder of DevZuz

Why are you using Maven?

- Consider this from the beginning
- Right tool for the job
- Ensure you reap the benefits

Best Practices

- Simplicity
- Planning ahead
- Portability
- Reproducibility

ApacheCon

Simplicity

Getting Started

- Write the build like you write code
- Utilise conventions
- Use multiple modules

Inheritance

- Multi-module inheritance
- Organisational POM hierarchy

Build Pipeline

- Depends on your team
- Use profiles for controlling complexity
- Applies to testing as well
- Key is to keep the build fast

Scripting

- Maven is declarative by design
- Integrate scripting if necessary
- Consider writing plugins

Planning Ahead

Development Environment

- Maven is best used in the bigger picture
- Plan for the infrastructure you will use

Automated Build Servers

- Maven is intended to highly automated
- More than continuous integration

Repository Management

- Centralise storage of artifacts
- Store the artifacts you build
- Store third-party artifacts you consume
- Servers can help manage the artifacts

Using Repositories

```
<repository>
  <id>apache-snapshots</id>
  <name>Apache Snapshots Repository</name>
  <url>http://people.apache.org/repo/m2-snapshot-repository</url>
  <releases>
    <enabled>false</enabled>
  </releases>
</repository>

<repository>
  <id>java-net-m1</id>
  <name>Java.net Repository</name>
  <url>http://download.java.net/maven/1/</url>
  <layout>legacy</layout>
  <snapshots>
    <enabled>false</enabled>
  </snapshots>
</repository>
```

Using Repositories

```
<repository>
  <id>apache-releases</id>
  <name>Apache Releases Repository</name>
  <url>http://people.apache.org/repo/m2-ibiblio-rsync-repository</url>
  <snapshots>
    <enabled>false</enabled>
  </snapshots>
</repository>
```

```
<repository>
  <id>apache-snapshots</id>
  <name>Apache Snapshots Repository</name>
  <url>http://people.apache.org/repo/m2-snapshot-repository</url>
  <releases>
    <enabled>false</enabled>
  </releases>
</repository>

<repository>
  <id>java-net-m1</id>
  <name>Java.net Repository</name>
  <url>http://download.java.net/maven/1/</url>
  <layout>legacy</layout>
  <snapshots>
    <enabled>false</enabled>
  </snapshots>
</repository>
```

```
<repository>
  <id>codehaus-snapshots</id>
  <name>Codehaus Snapshots Repository</name>
  <url>http://snapshots.repository.codehaus.org/</url>
  <releases>
    <enabled>false</enabled>
  </releases>
</repository>
```

```
<repository>
  <id>codehaus-releases</id>
  <name>Codehaus Releases Repository</name>
  <url>http://repository.codehaus.org/</url>
  <snapshots>
    <enabled>false</enabled>
  </snapshots>
</repository>
```

```
<repository>
  <id>java-net-m2</id>
  <name>Java.net Repository</name>
  <url>http://download.java.net/maven/2/</url>
  <snapshots>
    <enabled>false</enabled>
  </snapshots>
</repository>
```

Using Repositories

- Minimise the number of repositories
- Only in the POM if you redistribute
- Use repository manager to centralise

Settings and Installation

- Three levels of configuration
  - project (pom.xml)
  - user (~/.m2/settings.xml)
  - installation (<maven>/conf/settings.xml)

Portability

The Goal

- When a new developer builds the project
  - it works first go
  - ... and it keeps working

Hard Coding

- Don’t hard code paths
- Don’t hard code databases
- Don’t hard code properties
- Don’t do it in the tests either

Profiles

- Very useful - but don’t abuse them
- Document them all
- Avoid depending on the environment

Portable Artifacts

X

```
<bean
 id="dataSource"
 class="org.springframewor[illegible]dbc[illegible]tasource.DriverManagerDataSource">
  <property name="driverClas[illegible]" value="org.hsqldb.jdbcDriver" />
  <property name="url" value[illegible]c:hsqldb:database" />
  <property name="username"[illegible]alu[illegible]"sa" />
  <property name="passwor[illegible]value[illegible] />
</bean>
```

Portable Artifacts

- Artifacts in repository must be unique
- Either by classifier, or being portable
- Recommend externalising configuration
  - database
  - target environment
  - properties

Resource Filtering

- Use with great care!
- Centralisation, not substitution

  ✓ `google.analytics.code=UA-1234567-1`

  X `database.username=admin`
- Useful for once-off alterations
- Consider externalising configuration

Reproducibility

Reproducibility

- Important for releases
- More important for source releases
- Depends on portability too
- Build must be isolated from change

The Enforcer

```
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-enforcer-plugin</artifactId>
  <executions>
    <execution>
      <goals>
        <goal>enforce</goal>
      </goals>
      <configuration>
        <rules>
          <requirePluginVersions>
            <banLatest>true</banLatest>
            <banRelease>true</banRelease>
          </requirePluginVersions>
        </rules>
      </configuration>
      ...
```

The Enforcer

- Ensure build will be reproducible
- Based on rules
  - force specific plugin versions
  - ban snapshots
  - force Maven/Java/OS version
  - can write your own

Dependencies

- Specify only what you need
- Specify scope
- Use `dependencyManagement` to:
  - coerce Maven to use a particular version
  - enforce consistency within a project

Releases

- Set the project version to a `-SNAPSHOT`
- Make them early and often
- Automate as much as possible
  - Use Maven tools to assist
- Apply strict criteria, but no more than CI

Archetypes

- Standard project layouts
- Include organisational POM
- Facilitates consistency

Sites and Reports

- A whole other topic of best practices!
- But apply the same principles
  - set up what you'll actually use
  - set up enforcement checks, not just reports

Questions?

Brett Porter - brett@apache.org

Better Builds with Maven, blog at

http://www.devzuz.org/

Leading the Wave
of Open Source