Repository Management and Sonatype Nexus

Contents

1 Objectives

- Understand the benefits of using a repository manager
- Know how to start using Nexus
- Learn about Component Lifecycle Management

2 Development Today

Uses components. Lots of them.

→ more than 80% of a common enterprise software

- Facilitate the power of open source
- Don't reinvent the wheel

3 But What Is A Component?

i. any artifact or library that your software needs in order to be built, released and to run

Contains code, class files, object files, binary resources like images, property files, xml files...

In jar, war, ear, swf, so, bin, apk, apklib, zip, tar.gz, rpm, deb files and so on

Examples

- Google Guice jar file needed during runtime
- JUnit jar file needed for unit test execution
- JDBC driver for your database needed at runtime

4 Manual Dependency Management

- Painful
- Unreliable
- Overloads SCM
- Hard to maintain and document

Note
Unbelievably lots of developers still do this today!

---

5 Declarative Dependency Managment

- Automatic
- Including transitive dependencies
- Declarative - so easy to read and understand
- Support from tools

6 "Problems" With Declarative Approach

- Common complaint "Maven is downloading the internet, again!"
- In fact everything is cached locally (~/.m2/repository)
- Components are used from local repo in **all** your projects built with Maven
- Other tools also need to download components, and all use
  - Central Repository
  - Maven repository format

Tip
This is where Sonatype Nexus can help!

---

7 Adoption Stages and Advantages

Proxy external repositories

- Starting with Central Repository
- Reduced downloads, faster builds, increased stability
- Adding more proxy repositories
  - only needs to be done on the server
  - developers get access to more components without any work

8 Adoption Stages and Advantages

Host external and internal artifacts

- Deploy once for everybody
- Share binary components like open source projects
- Improve cooperation between multiple, different teams (dev, qa, ops. . .)

9 Adoption Stages and Advantages

Lifecycle Integration

- Addition of CI server
- Controlling component usage - Procurement
- Improving release process - Staging
- Gaining license and security understanding of the components

10 Sonatype Nexus as Center Hub

images/nexus-tool-suite-integration.png

→ Nexus will be a key component of your enterprise development infrastructure

11 Installing Nexus

1. Install Java 7
2. Get the bundle with the embedded Jetty server from the download page
3. Extract archive, create symbolic link and run

```
sudo cp nexus-professional-x.y.z-bundle.tar.gz /usr/local
cd /usr/local
sudo tar xvzf nexus-professional-x.y.z-bundle.tar.gz
ln -s nexus-professional-x.y.z nexus
cd nexus
./bin/nexus console
```

4. Go to http://localhost:8081/nexus and log in with admin/admin123

Note
Nexus Professional has enterprise benefits, but open source edition is perfect for getting started.

---

12 Relationship Apache Maven and Nexus

Apache Maven introduced repository concept:

- storage for plugins
- and dependencies

All are retrieved from repositories on the internet, by the default the Central Repository

- Nexus runs Open Source Repository Hosting OSSRH as input for the Central Repository
- Nexus can run as proxy on site for you
- Best of breed Maven Repository Manager MRM (and beyond)

Tip
Read more about the scale needed to run OSSRH on the blog.

---

13 Connecting Maven to Nexus

Establish system/user wide setting for Maven to use Nexus:

- modify/create ~/.m2/settings.xml to point to Nexus (see labs/settings/)
- build a few Maven projects
- see how it starts proxying

```
<settings>
  <mirrors>
    <mirror>
      <id>nexus</id>
      <mirrorOf>*</mirrorOf>
      <url>http://localhost:8081/nexus/content/groups/public</url>
    </mirror>
  </mirrors>
  <profiles>
    <profile>
      <id>nexus</id>
      <repositories>
        <repository>
          <id>central</id>
          <url>http://central</url>
          <releases><enabled>true</enabled></releases>
          <snapshots><enabled>true</enabled></snapshots>
        </repository>
      </repositories>
     <pluginRepositories>
        <pluginRepository>
          <id>central</id>
          <url>http://central</url>
          <releases><enabled>true</enabled></releases>
          <snapshots><enabled>true</enabled></snapshots>
        </pluginRepository>
      </pluginRepositories>
    </profile>
  </profiles>
  <activeProfiles>
    <activeProfile>nexus</activeProfile>
  </activeProfiles>
</settings>
```

Tip
For other build tools this will be different.

14 Component Coordinates

Structure storage for components using unique "GAV" coordinates:

- **g** roupId, **a** rtifactId, **v** ersion - GAV
- optionally classifier and packaging

```
<dependency>
  <groupId>org.testng</groupId>
  <artifactId>testng</artifactId>
  <version>6.1.1</version>
</dependency>

<dependency>
  <groupId>com.google.inject</groupId>
  <artifactId>guice</artifactId>
  <version>3.0</version>
  <classifier>no_aop</classifier>
</dependency>

<dependency>
  <groupId>org.glassfish.admingui</groupId>
  <artifactId>war</artifactId>
  <version>10.0-b28</version>
  <type>war</type>
</dependency>
```

15 Maven Repository Format

Uses the GAV component coordinates. Coordinates map to specific locations in a Maven repository.

```
<dependency>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-core</artifactId>
  <version>3.4.1</version>
</dependency>
```

Maps to:

```
org/apache/camel/
                  camel-core/
                             3.4.1/
    camel-core-3.4.1.pom
    camel-core-3.4.1.jar
```

File names are created using

```
artifactId-version-classifier.packaging
```

Classifiers javadoc and sources are appended to file name:

```
    camel-core-3.4.1-javadoc.jar
    camel-core-3.4.1-sources.jar
```

Tip
Other repository formats use a different structure, but the Maven structure is understood and used by many tools.

---

16 What is a Repository?

- Organized storage and access container for artifacts
- Uses artifact coordinates for structure

→ A Repository Manager helps with administration and usage

17 Repository Manager Tasks

- Proxy and managing access to public repositories
- Storing components that are not in public repositories
- Managing releases and snapshots
- Controlling available and allowed dependencies
- Facilitate internal collaboration across components and teams

18 Repository Manager Advantages

- Increased speed
- Reduced bandwidth usage
- Predictability
- Ability to control and audit - all components under your control
- Improved management of 3rd party artifacts
- Internal collaboration enabled
- Distribution of components made possible

19 Nexus User Interface Tour

- Search for components, including advanced search
- View component details including security and license details
- Repositories
- Server administration
- Security

20 Proxying

Public Group is exposed to users →

- can be changed on server for all users
- takes security access rights into account

Examples:

- add an additional external proxy repository
- add an internal hosted repository
- manually deploy component into 3rd party hosted repository

Tip
Demo time!

---

21 Release vs Snapshot Repositories

Release Repositories

- Store "point-in-time" Releases
- Releases never change
- Publish a Release → Both the artifact and meta-data "live forever"

Snapshot Repositories

- Used for development-only
- Transient
- No promise SNAPSHOT artifacts will remain the same

Tip
Repositiory Groups merge them and expose the all under one URL.

---

22 Deploying Internal Components

is when the benefits step up to the next level:

- Sharing of binary components and not specification documents
- No more building each others components
- End of large multi-module builds
- Choice of build system

23 Deployments with Maven. . .

```
mvn clean deploy
```

- pom.xml → distributionManagement
  - snapshotRepository
  - releaseRepository
- settings.xml → server

24 pom.xml - distributionManagement

```
<distributionManagement>
  <repository>
    <id>nexus-releases</id>
    <url>http://localhost:8081/nexus/content/repositories/releases</url>
  </repository>
  <snapshotRepository>
    <id>nexus-snapshots</id>
    <url>http://localhost:8081/nexus/content/repositories/snapshots</url>
  </snapshotRepository>
</distributionManagement>
```

25 settings.xml - server

```
<servers>
  <server>
    <id>nexus</id>
    <username>admin</username>
    <password>admin123</password>
  </server>
</servers>
```

26 Maven Deploy Plugin

Use the example project in labs/maven-deploy-example

```
mvn clean deploy
mvn versions:set -DnewVersion=1.0.0
mvn clean deploy
```

- Snapshot versions can be deployed multiple times.
- Releases only once.

Now components are available for everybody via the public group.

Tip
Your continuous integration server could do the deployment.

---

27 Advanced Features

Procurement
Control availability of components

Staging
multi-step, controlled release process including reruns

Maven Settings Distribution
via Nexus Maven Plugin

Security
Enhanced LDAP, Atlassian Crowd

Other repository formats
NuGet, Site, P2, OBR, YUM

28 Distributed Deployments

Scale your organization, while maintaining performance for everybody!

images/nexus-smart-proxy.png

Various scenarios and setups are common, including:

- integration with component providers
- cooperation with external development teams
- component distribution to clients

29 Component Lifecycle Management

Component lifecycle management can be defined as the **practice of**

- **analyzing**,
- **controlling**, and
- **monitoring**

the components used in your software development lifecycle.

Sonatype CLM integration in

- Hudson/Jenkins
- Eclipse
- Nexus
- ...

30 Integrating with Nexus

- Lots of build tools can integrate with Maven repositories
  - Ant/Ivy, Gradle, SBT, Grails, ...
- All functionality is available in REST API
- Java Client for REST API available
- Plugin architecture with examples to create your own

31 Want to learn more?

- Nexus Opens Source OSS website
- Nexus Professional website
- Screen cast recordings
- Repository Management with Nexus - free book
- Nexus Professional Trial Bundle and Guide - some examples can be used with Nexus OSS as well
- Mailing lists
- Talk to the developers/support - HipChat
- Training classes