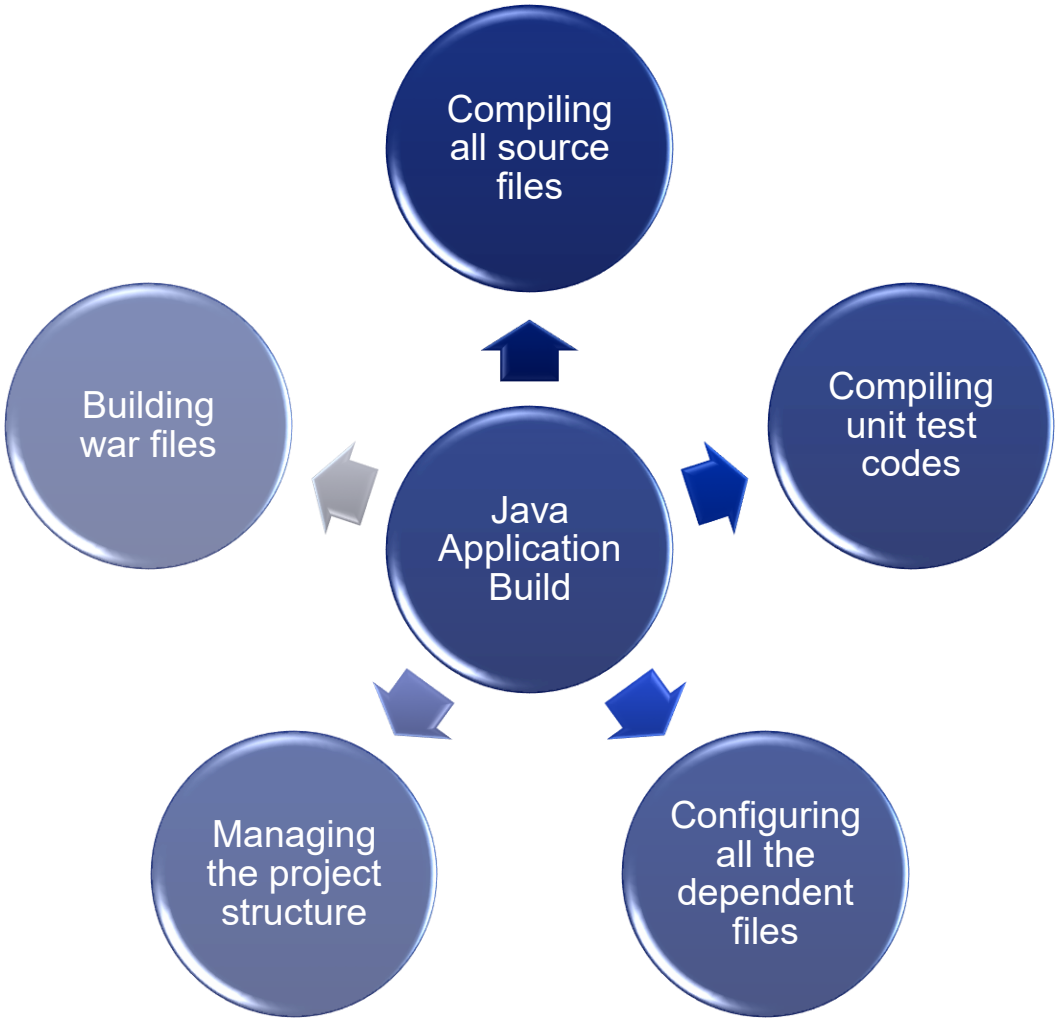
Introduction to Maven

Presented By

Sagar

Java Consultant

Java Application Build Steps



Common Issues with normal projects

What is Maven ?

- Build tool
- A software project management and comprehension tool.
- It can manage the build of project, documentation and reporting from a central piece of information
- Maven is hosted by Apache Software Foundation

What is Maven ?

- Maven uses POM (Project Object Model) in order to describe the software project which is being built, it's dependencies on other external modules and build order.

What is Maven ?

- Maven comes with predefined targets for performing certain tasks like compilation of code and it's packaging.
- Maven is network-ready. The core engine can dynamically download plug-ins from a repository.

Setting the environment for Maven

- Download Maven from http://maven.apache.org/download.html
- Unzip the installation archive to the desired folder

Setting the environment for Maven

- If installed correctly, you should be able to test it by opening a command prompt and typing :

```
C:\>mvn -version
```

- The output should be like this:

```
C:\>mvn -version
Apache Maven 2.2.1 (r801777; 2009-08-07 00:46:01+0530)
Java version: 1.6.0_16
Java home: C:\Program Files\Java\jdk1.6.0_16\jre
Default locale: en_US, platform encoding: Cp1252
OS name: "windows xp" version: "5.1" arch: "x86" Family: "windows"
```

Maven Key terms

- Archetype - is a template of a project which is combined with some user input to produce a working Maven project that has been tailored to the user's requirements
- POM - The pom.xml file is the core of a project's configuration in Maven. It is a single configuration file that contains the majority of information required to build a project.
- Dependencies(required .jar files) - Maven allows projects to declare what dependencies they have, and will automatically materialize those dependencies

Maven Repositories

- Maven repositories store a set of artifacts which are used by Maven during dependency resolution for a project.
- All repositories are downloaded from mvnrepository.com
- Local repositories can be accessed on the local hard disk.
- Remote repositories can be accessed through the network.
- An artifact is bundled as a JAR file which contains the binary library or executable.
- An artifact can also be a war or an ear.

Maven Archetypes

- An archetype is a complete project template
- Using an archetype, a project template can be created with a simple command
- Following is a list of archetypes and their purposes.

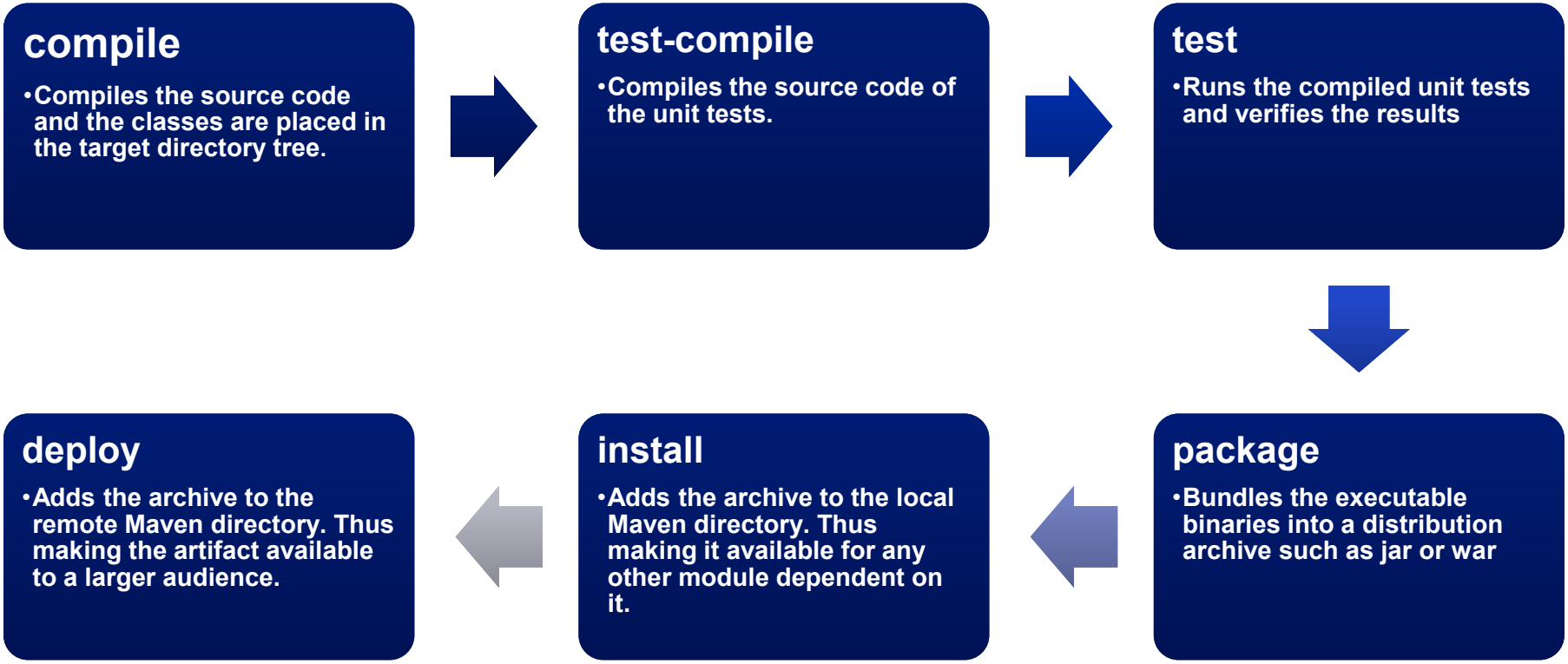
Archetype	Purpose
maven-archetype-archetype	To create our own project template (archetype)
Maven-archetype-j2ee-simple	To create a J2EE project (EAR).
Maven-archetype-webapp	To create a web application (WAR). This contains a HelloWorld JSP
Maven-archetype-quickstart	To create a simple Java project. Can be used to generate JAR. Default of Maven 2.

Maven Life cycle phases

- Plug-ins are software modules which are written to fit into the plug-in framework of Maven.
- Each task within a plugin is called a *mojo*.
- A mojo is executed when the Maven engine executes the corresponding phase on the build life cycle.
- The association between the phase of a life cycle and a mojo is called as a *binding*.

Maven Life cycle phases (Continued…)

The phases of the build life cycle are described below:



First Maven project

First Maven project

- In order to create everything which is needed for a simple Java project that can be built using Maven, you can use **Archetype** plug-in.
- Archetype is a standard plug-in which comes with Maven.
- The Archetype plug-in runs outside of a Maven project build life cycle and is used for creating Maven projects.

*** Show Archetype while creating maven app*

Maven Compiler Plugin

- The Maven Compiler Plugin helps you to compile Java source code.
- This plugin compiles your project's sources.
- Ex :

```
<plugins>
  <plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-compiler-plugin</artifactId>
    <version>3.13.0</version>
    <configuration>
      <source>1.8</source>
      <target>1.8</target>
    </configuration>
  </plugin>
</plugins>
```

Maven profile

- Profiles can be automatically triggered based on the detected state(various versions of JDK etc) of the build environment.
- These triggers are specified via an <activation> section in the profile itself.

- Ex: for JDK 1.4

```
<profiles>
  <profile>
    <activation>
      <jdk>1.4</jdk>
    </activation>
    ...
  </profile>
</profiles>
```

If range of versions are to be configured

```
<profiles>
  <profile>
    <activation>
      <jdk>[1.3,1.6)</jdk>
    </activation>
    ...
  </profile>
</profiles>
```

Creating Maven Project Using Command Line

- Navigate to your workspace
- Type the below command in command line from your workspace

```
D:\MphasisWorkspace>mvn archetype:generate -DgroupId=com.mycom -DartifactId=maven-hell
o-app -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

Import the workspace into eclipse as follows :

- File->Import->Existing Maven Projects
- Navigate to the project folder
- Ensure that pom.xml is cheked under Projects:
- Click on Finish

Adding Test-scoped Dependencies

```
<dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>3.8.1</version>
        <scope>test</scope>
</dependency>
```

Generating Test Reports

- Add the following reporting plugin in pom.xml

```
<reporting>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-surefire-report-plugin</artifactId>
      <version>3.3.1</version>
    </plugin>
  </plugins>
</reporting>
```

- Run the following command in command prompt:
- D:\YourWorkspace`\MyFirstMavenApp>mvn surefire-report:report`
- A test report will be created.
- Type the following command to see the report on web page
- D:\YourWorkspace`\MyFirstMavenApp\target\site>surefire-report.html`

Creating .jar file

Add the following in pom.xml

```
  <build>
    <plugins>
        <plugin>
           <groupId>com.mycom</groupId>
             <artifactId>exec-maven-plugin</artifactId>
             <version>4.0.0</version>
             <configuration>
                <mainClass>com.mycom.App</mainClass>
             </configuration>
        </plugin>
    </plugins>
  </build>
```

Type **mvn package**

Navigate to \tartet folder

Type **java -cp maven-hello-app-1.0-SNAPSHOT.jar com.mycom.App**

Site life cycle

After running mvn surefire-report:report, \MyFirstMavenApp\target\site> will be generated.

The plugins for site are :

- pre-site
- site
- post-site
- site-deploy

Site configuration

```
<plugins>
        <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-antrun-plugin</artifactId>
                <version>1.1</version>
                <executions>
                        <execution>
                                <id>id.pre-site</id>
                                <phase>pre-site</phase>
                                <goals>
                                        <goal>run</goal>
                                </goals>
                                <configuration>
                                        <tasks>
                                                <echo>in pre-site phase</echo>
                                        </tasks>
                                </configuration>
                        </execution>
                        <execution>
                                <id>id.site</id>
                                <phase>site</phase>
                                <goals>
                                        <goal>run</goal>
                                </goals>
                                <configuration>
                                        <tasks>
                                                <echo>in site phase</echo>
                                        </tasks>
                                </configuration>
                        </execution>
                        <execution>
                                <id>id.post-site</id>
                                <phase>post-site</phase>
                                <goals>
                                        <goal>run</goal>
                                </goals>
                                <configuration>
                                        <tasks>
                                                <echo>in post-site phase</echo>
                                        </tasks>
                                </configuration>
                        </execution>
                        <execution>
                                <id>id.site-deploy</id>
                                <phase>site-deploy</phase>
                                <goals>
                                        <goal>run</goal>
                                </goals>
                                <configuration>
                                        <tasks>
                                                <echo>in site-deploy phase</echo>
                                        </tasks>
                                </configuration>
                        </execution>
                </executions>
        </plugin>
</plugins>
```

Thank You