

# **Introducción a Apache Maven**

Taller de Tecnologías de Producción de  
Software - Opción Java

Lic. Vanessa Aybar Rosales

# Agenda

## ❏ Conceptos básicos

- ❏ Gestión de aplicaciones Java SE/EE
- ❏ ¿Qué es Apache Maven?
- ❏ Project Object Model
- ❏ Archetypes
- ❏ Creando un proyecto con Maven

## ❏ Funcionamiento

- ❏ Ciclo de Vida, Fases, Plugins y Goals
- ❏ Dependencias, repositorios

## ❏ Instalación

# **Introducción a Apache Maven**

CONCEPTOS BÁSICOS

# Gestión de aplicaciones Java SE/EE

- ❑ Tareas de **compilación, empaquetado y despliegue**
- ❑ Inclusión de **librerías** propias y de terceros
- ❑ Gestión de **recursos estáticos** (HTML, JSP, Javascript, CSS, archivos de conf., etc.)
- ❑ Realización de **pruebas unitarias** (JUnit).
- ❑ Realización de **pruebas de integración** en sistemas de control de versiones (CVS, SVN, Git)

# Problemas en la gestión de aplicaciones Java SE/EE

- ❑ Heterogeneidad entre distintos sistemas
- ❑ Tiempo de adaptación del desarrollador al proyecto
- ❑ Administración de versiones del proyecto
- ❑ Administración de versiones de librerías
- ❑ Administración de perfiles: desarrollo, pre-producción, producción

# ¿Qué es Apache Maven?

- ❑ Herramienta para la administración del desarrollo de un proyecto de software
  - ❑ define el **ciclo de vida** del proyecto
  - ❑ proporciona un **modelo de gestión y descripción** del proyecto
- ❑ Convention over configuration
  - ❑ código fuente: `${basedir}/src/main/java`
  - ❑ recursos: `${basedir}/src/main/resources`
- ❑ Define una interfaz común

**mvn** clean | **mvn** package | **mvn** install

# ¿Qué es Apache Maven?

- ❑ Escribimos una descripción del proyecto conocido como el “Project Object Model” (**pom.xml**)
- ❑ Administración de **dependencias** e información de dependencias transitivas
- ❑ Permite incorporar servicios vía “**plugins**”

# Project Object Model (POM)

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId> edu.info.ttps </groupId>
  <artifactId> ttps </artifactId>
  <packaging> war </packaging>
  <version> 1.0-SNAPSHOT </version>
  <name>TTPS Maven Webapp</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
  <build>
    <finalName>ttps</finalName>
  </build>
</project>
```

COORDENADA:  
IDENTIFICA DE  
MANERA UNÍVOCA  
UN PROYECTO

NOMBRE DE  
NUESTRO  
PRODUCTO  
EMPAQUETADO

MAVEN NO TRABAJA SOBRE  
NUESTRO POM, SINO SOBRE  
EL "EFFECTIVE POM"

`mvn help:effective-pom`



# Archetype

- ❑ Es un template o modelo a partir del cual creamos un proyecto (ejb, war, jar, etc)

```
mvn archetype:generate
```

|                            |  |
|----------------------------|--|
| maven-archetype-webapp     | arquetipo de un proyecto web               |
| maven-archetype-quickstart | arquetipo de una app. java estándar (jar)  |
| maven-archetype-plugin     | arquetipo para desarrollar plugin de maven |

- ❑ Un desarrollador puede crear sus propios arquetipos

<http://maven.apache.org/guides/mini/guide-creating-archetypes.html>

# Creando un proyecto web con Maven

```
mvn archetype:generate
```

```
-DarchetypeArtifactId=maven-archetype-webapp
```

PROTOTIPO DE MI PROYECTO

```
-DgroupId=edu.info.ttps
```

GRUPO, ORGANIZACIÓN.  
POR CONVENCION ES  
INVERSA DEL DOMINIO

```
-Dpackaging=war
```

SALIDA DE MI PROYECTO

```
-DarchetypeId=ttps
```

IDENTIFICADOR DEL  
PROYECTO EN MI GRUPO

# Creando un proyecto web con Maven

```
Administrator: C:\Windows\system32\cmd.exe

C:\monTTPS>mvn archetype:generate -DgroupId=edu.info.ttps -Dpackage=edu.info.ttps -DartifactId=ttps -DarchetypeArtifactId=maven-archetype-webapp -Dversion=1.0-SNAPSHOT
[INFO] Scanning for projects...
[INFO]
[INFO] -----
[INFO] Building Maven Stub Project (No POM) 1
[INFO] -----
[INFO]
[INFO] >>> maven-archetype-plugin:2.0-alpha-4:generate (default-cli) @ standalone-pom >>>
[INFO]
[INFO] <<< maven-archetype-plugin:2.0-alpha-4:generate (default-cli) @ standalone-pom <<<
[INFO]
[INFO] --- maven-archetype-plugin:2.0-alpha-4:generate (default-cli) @ standalone-pom ---
[INFO] Setting property: classpath.resource.loader.class => 'org.codehaus.plexus.velocity.ContextClassLoaderResourceLoader'.
[INFO] Setting property: velocimacro.messages.on => 'false'.
[INFO] Setting property: resource.loader => 'classpath'.
[INFO] Setting property: resource.manager.logwhenfound => 'false'.
[INFO] Generating project in Interactive mode
Downloading: http://repo.maven.apache.org/maven2/org/apache/maven/archetypes/maven-archetype-webapp/1.0/maven-archetype-webapp-1.0.jar
Downloaded: http://repo.maven.apache.org/maven2/org/apache/maven/archetypes/maven-archetype-webapp/1.0/maven-archetype-webapp-1.0.jar (4 KB at 0.4 KB/sec)
Confirm properties configuration:
groupId: edu.info.ttps
artifactId: ttps
version: 1.0-SNAPSHOT
package: edu.info.ttps
Y: :
[INFO] -----
```

# Creando un proyecto web con Maven

```
---
[INFO] Using following parameters for creating OldArchetype: maven-archetype-web
app:1.0
[INFO] -----
---
[INFO] Parameter: groupId, Value: edu.info.ttps
[INFO] Parameter: packageName, Value: edu.info.ttps
[INFO] Parameter: package, Value: edu.info.ttps
[INFO] Parameter: artifactId, Value: ttps
[INFO] Parameter: basedir, Value: C:\monTTPS
[INFO] Parameter: version, Value: 1.0-SNAPSHOT
[INFO] ***** End of debug info from resources from generated POM
*****
[INFO] OldArchetype created in dir: C:\monTTPS\ttps
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 2:03.510s
[INFO] Finished at: Thu Oct 10 23:30:14 ART 2013
[INFO] Final Memory: 9M/22M
[INFO] -----
C:\monTTPS>
```

# Creando un proyecto web con Maven

Estructura resultante:

```
ttps/  
├── pom.xml  
└── src  
    ├── main  
    │   ├── resources  
    │   └── webapp  
    │       ├── index.jsp  
    │       └── WEB-INF  
    │           └── web.xml
```

No hay carpeta java ni test, hay que crearlas

# Creando un proyecto web con Maven

- ❑ **src/main/java:** código fuente de la aplicación que estamos desarrollando.
- ❑ **src/main/resources:** recursos de la aplicación que estamos desarrollando (imágenes, audio, documentos...).
- ❑ **src/main/config:** archivos de configuración utilizados en la aplicación (hibernate.cfg.xml, persistence.xml, etc).
- ❑ **src/main/webapp:** en caso de una aplicación web, aquí van los recursos tipo HTML, CSS, JS, etc.

-----

- ❑ **src/test/java:** código fuente de las clases de test de la aplicación.
- ❑ **src/test/resources:** recursos a utilizar por los tests de la aplicación (ej: archivo que debe ser leído por una clase de test)

# Creando un proyecto web con Maven

```
c:\...\ttps\mvn package
```



```
ttps/
```

```
|— pom.xml
```

```
|— src
```

```
|— target
```

```
    |— classes
```

```
    |— maven-archiver
```

```
    |— ttps
```

```
    |— ttps.war
```

# **Introducción a Apache Maven**

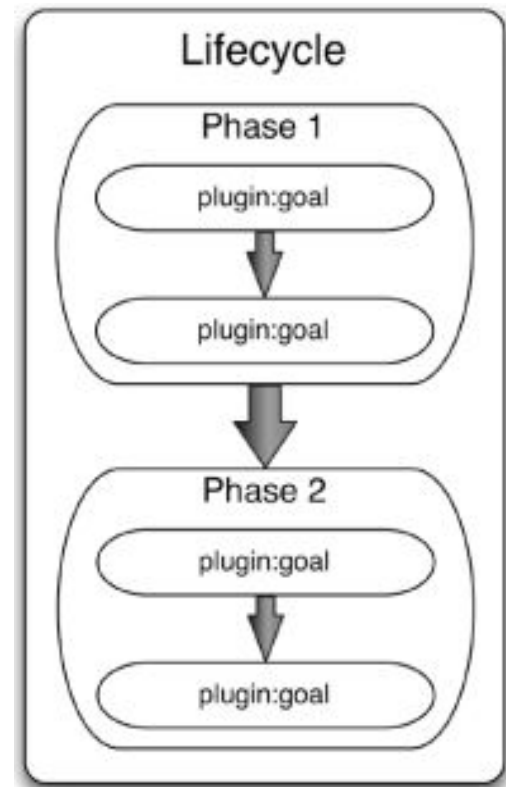
FUNCIONAMIENTO



# Ciclo de Vida, Fases, Plugins y Goals

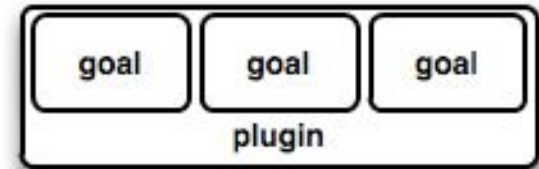
En maven se realizan una serie de pasos desde la compilación hasta el deploy (independiente del lenguaje)

- ❑ Un **Ciclo de vida** es un conjunto de fases ( cjto. de pasos) bien conocidos
- ❑ Cada **Fase** comprende plugins



# Ciclo de Vida, Fases, Plugins y Goals

- ❑ Los **Plugins** abarcan un conjunto de goals



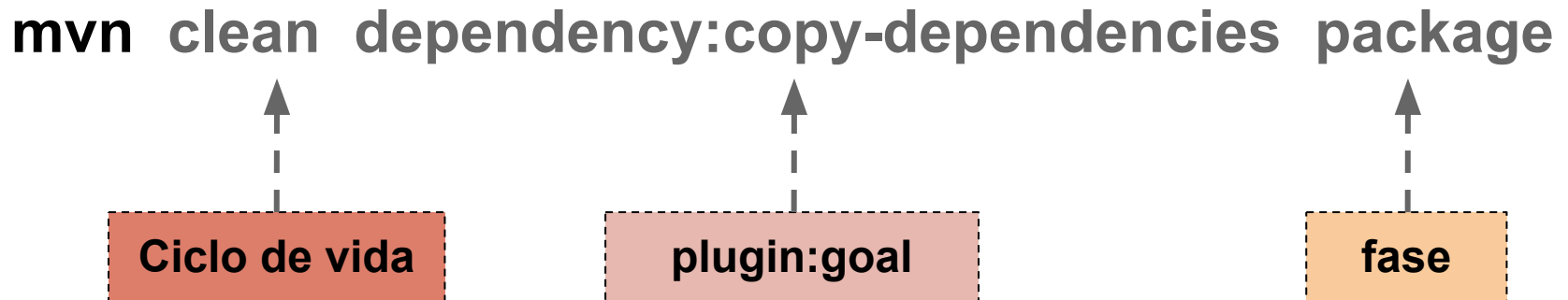
- ❑ Un **Goal** es una tarea específica

```
mvn jar:jar  
mvn jar:sign  
mvn jar:sign-verify
```

- ❑ Los plugins son parte de la descripción del proyecto y para cada goal se indica a qué fase está asociado

# Ciclo de Vida, Fases, Plugins y Goals

- ❏ mvn ejecuta los “pasos” que le indiquemos uno a continuación de otro

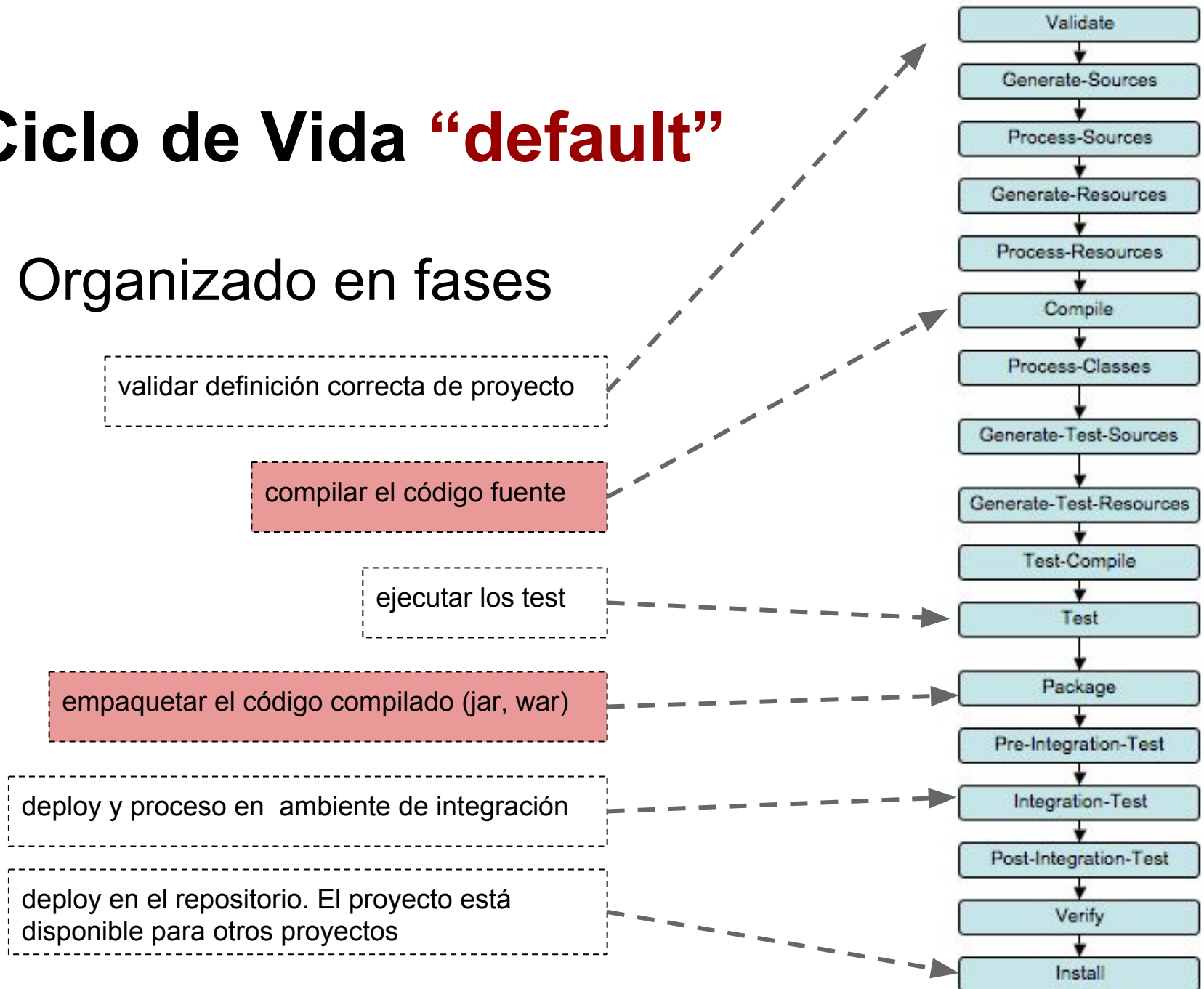


# Ciclos de vida en Maven

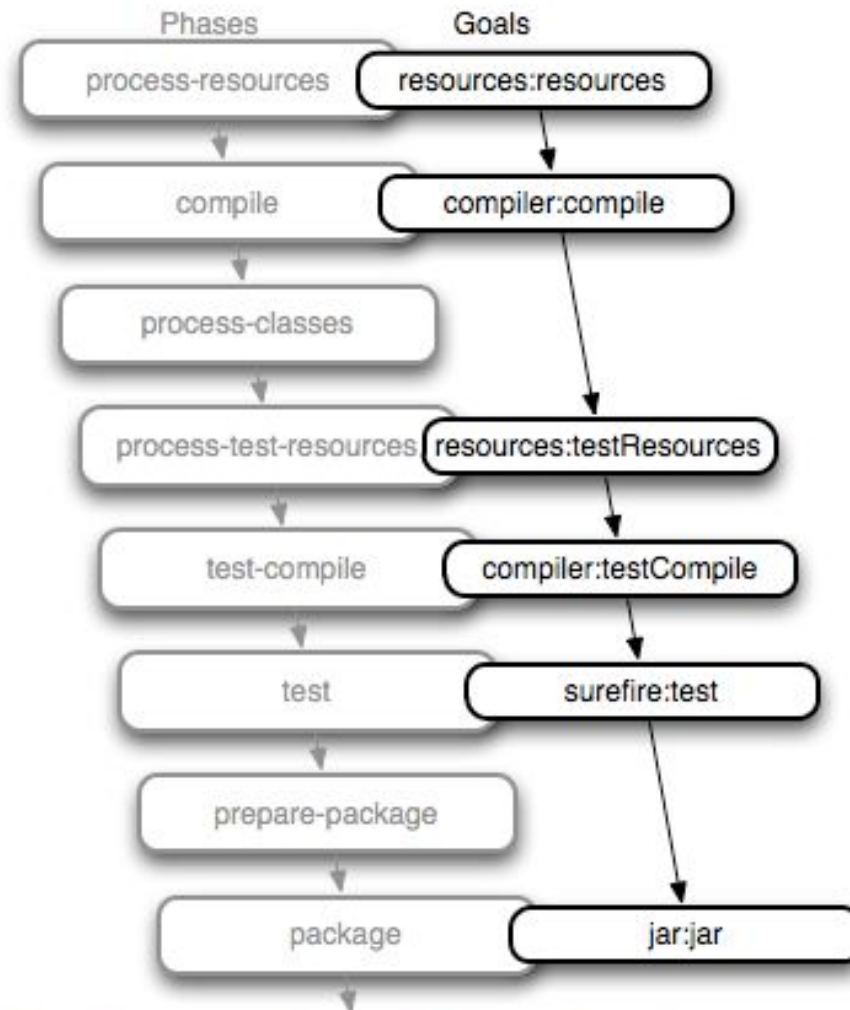
- ❑ **clean**: se eliminan los archivos compilados del directorio de salida
- ❑ **default**: abarca las fases de construcción de un programa hasta la instalación de nuestro artifact en un repositorio
- ❑ **site**: genera información html de nuestro proyecto y hace deploy de nuestro artifact en un web server

# Ciclo de Vida “default”

## Organizado en fases



# Ciclo de Vida “default”, Fases, Plugins y Goals



Note: There are more phases than shown above, this is a partial list

# Administración de dependencias

- ❑ En Maven una dependencia **NO** es un jar, es una referencia a otro artifact que se identifica unívocamente a través de su **coordenada**
- ❑ Las dependencias se descargan localmente en la carpeta “**repository**” desde repositorios remotos:

```
http://search.maven.org (repositorio central)
http://mvnrepository.com
http://repository.jboss.org/nexus/
http://repository.sonatype.org
```

- ❑ El proyecto tiene asociado un árbol (o grafo) de dependencias

```
mvn dependencies:tree
```

# Alcance de las dependencias

```
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>3.8.1</version>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>axis-wsdl4j</groupId>
  <artifactId>axis-wsdl4j</artifactId>
  <version>1.5.1</version>
  <scope>system</scope>
  <systemPath>${project.basedir}/src/main/webapp/WEB-INF/lib/axis-wsdl4j-1.5.1.jar</systemPath>
</dependency>
```

## Opciones

|                 |  |
|-----------------|--|
| <b>compile</b>  | (default) usada en todas las fases                   |
| <b>provided</b> | por el jre o el contenedor web                       |
| <b>runtime</b>  | requerida para ejecución, no para compilación        |
| <b>test</b>     | requerida para compilación y ejecución de test       |
| <b>system</b>   | similar a provided pero no es tomada del repositorio |



# Exclusión de dependencias

- ❑ Puede ocurrir que 2 dependencias referencien transitivamente a la misma dependencia (conflicto)

```
<dependency>
  <groupId>group-a</groupId>
  <artifactId>artifact-a</artifactId>
  <version>1.0</version>
  <exclusions>
    <exclusion>
      <groupId>group-c</groupId>
      <artifactId>excluded-artifact</artifactId>
    </exclusion>
  </exclusions>
</dependency>
```

# Configuración de repositorio en el POM

```
<project>
...
<repositories>
  <repository>
    <id>Sun Maven Repository</id>
    <url>http://download.java.net/maven/2/</url>
  </repository>
</repositories>
...
</project>
```

# Comandos útiles

```
mvn dependencies:tree
```

```
mvn clean
```

```
mvn compiler:compile
```

```
mvn package
```

```
mvn install
```

```
mvn help:describe -Dplugin=<pluginname>
```

```
mvn help:effective-pom
```

# Instalación

- ❑ Descargar del sitio

<http://maven.apache.org/download.html> (v 3.0+)

- ❑ Desempaquetar en un directorio

- ❑ Setear variables de entorno

```
C:\Users\vanessa> set JAVA_HOME=....
```

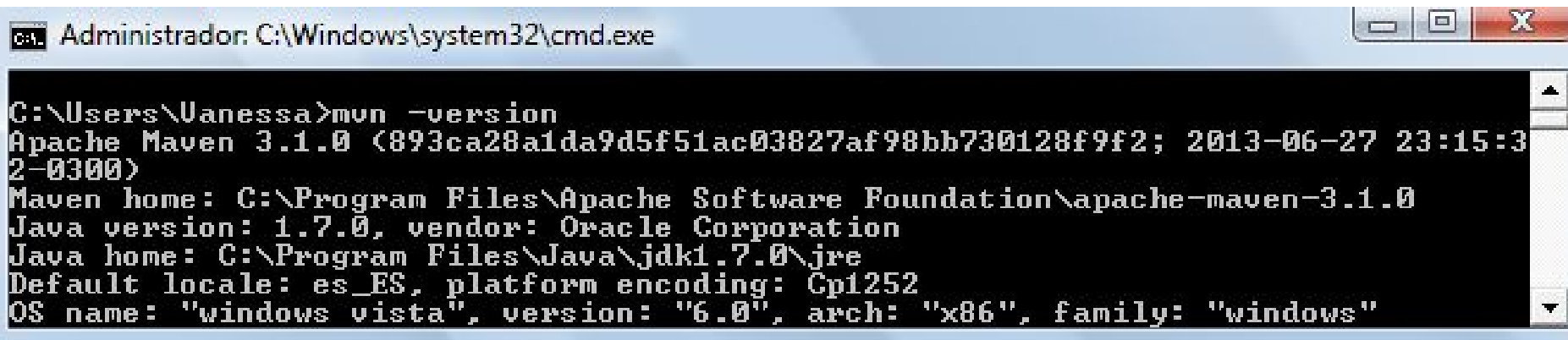
```
C:\Users\vanessa> set M2_HOME=c:\Program Files\apache-maven-3.0.3
```

```
C:\Users\vanessa> set M2_REPO=c:\...\m2\repository
```

```
C:\Users\vanessa> set PATH=%PATH%;%M2_HOME%\bin
```

# Instalación

❏ Probar la instalación: `mvn -version`



```
Administrador: C:\Windows\system32\cmd.exe

C:\Users\Vanessa>mvn -version
Apache Maven 3.1.0 (893ca28a1da9d5f51ac03827af98bb730128f9f2; 2013-06-27 23:15:32-0300)
Maven home: C:\Program Files\Apache Software Foundation\apache-maven-3.1.0
Java version: 1.7.0, vendor: Oracle Corporation
Java home: C:\Program Files\Java\jdk1.7.0\jre
Default locale: es_ES, platform encoding: Cp1252
OS name: "windows vista", version: "6.0", arch: "x86", family: "windows"
```

# Integración con eclipse

Usando **Maven Eclipse Plugin** podemos configurar nuestro proyecto para que eclipse lo reconozca como válido

```
mvn eclipse:configure-workspace -Declipse.workspace=<pathWS>
```

```
//agrega la variable M2_REPO al workspace
```

```
mvn eclipse:eclipse
```

```
//genera los archivos de configuración de eclipse:  
.classpath, .project, de modo que luego se puede importar  
como proyecto existente
```

```
mvn eclipse:clean
```

```
//elimina la configuración en el proyecto para eclipse
```

# Referencias

- ❑ Apache Maven Project (Documentation oficial)  
<http://maven.apache.org/guides>
- ❑ An Introduction to Maven 2  
<http://www.javaworld.com/javaworld/jw-12-2005/jw-1205-maven.html?page=1>
- ❑ Apache simplifica el proceso de construcción  
[http://www.programacion.com/java/tutorial/jap\\_maven/](http://www.programacion.com/java/tutorial/jap_maven/)