

UNIVERSIDAD DEL QUINDÍO
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

Información general	
ACTUALIZADO POR:	Einer Zapata
DURACIÓN ESTIMADA EN MINUTOS:	120
DOCENTE:	Christian Andrés Candela
GUÍA NO.	03
Nombre de la guía:	Proyecto Maven

Información de la Guía

Objetivo

Estudiar el uso de las entidades y su aplicación en el modelamiento de datos.

Conceptos Básicos

Manejo de Eclipse, Java, Bases de Datos, JDBC, XML, Glassfish.

Contextualización Teórica

Maven es un software cuya función principal es la gestión de dependencias de proyectos Java. Maven es uno de los proyectos de Apache Software Foundation que permite gestionar y crear proyectos Java. Para la configuración y gestión de los proyectos usa archivos de configuración POM (Project Object Model), en los cuales se especifican los diferentes elementos a usar en el proyecto, como lo pueden ser plugins y dependencias. Maven cuenta con un comando el cual puede ser usado a través de una consola, sin embargo, los principales IDEs de desarrollo ya poseen herramientas que permiten la fácil integración con Maven.

La estructura mínima de un archivo pom es:

```
<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/xsd/maven-
4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>co.edu.uniquindio.grid</groupId>
  <artifactId>proyectomaven</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>Proyecto Maven</name>
  <description>Proyecto Maven de Prueba</description>
</project>
```

Donde se tiene como principales elementos el groupId y el artifactId, el primero corresponde al identificador del grupo de artefactos (proyectos) al cual pertenece nuestro proyecto, el artifactId por su parte es el identificador único de nuestro proyecto dentro del grupo de artefactos. Ambos, groupId y artifactId, conforman un identificador único para nuestro proyecto.

En caso de que el proyecto así lo requiera se pueden adicionar dependencias hacia otros proyectos o librerías.

```
<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/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>co.edu.uniquindio.grid</groupId>
  <artifactId>proyectomaven</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>Proyecto Maven</name>
  <description>Proyecto Maven de Prueba</description>
  <dependencies>
    <dependency>
      <groupId>javax</groupId>
      <artifactId>javaee-api</artifactId>
      <version>8.0</version>
      <scope>provided</scope>
    </dependency>
  </dependencies>
</project>
```

Como puede verse en este caso se adiciono una dependencia hacia el proyecto javaee-api en su versión 8. El atributo scope indica en que momento de la vida de nuestro proyecto necesita la dependencia agregada, sus valores pueden ser: compile, provided, runtime, test y system.

- Compile: Es el valor por defecto, y la dependencia es usada a lo largo del ciclo de vida del proyecto.
- Provided: Similar a Compile, pero indica que se espera a que el JDK decida el momento en que se debe proveer la dependencia.
- Runtime: Indica que la dependencia no es requerida para compilar el proyecto, pero si para su ejecución
- Test: Indica que la dependencia no es requerida en la ejecución normal de la aplicación, pero si lo es para la ejecución de sus pruebas.
- System: Similar a provided, pero indica que el jar no se encuentra en ningún repositorio y el mismo deberá ser indicado de forma específica.

Para indicar o realizar tareas específicas en un proyecto se puede hacer uso de plugins, el más usado es el que permite indicar la versión de java con que se debe compilar el proyecto.

```
<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/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>co.edu.uniquindio.grid</groupId>
  <artifactId>proyectomaven</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>Proyecto Maven</name>
```

```
<description>Proyecto Maven de Prueba</description>
<dependencies>
  <dependency>
    <groupId>javax</groupId>
    <artifactId>javaee-api</artifactId>
    <version>8.0</version>
    <scope>provided</scope>
  </dependency>
</dependencies>
<build>
  <plugins>
    <!-- Establecer la version de java a ser usada por los proyectos -->
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.5</version>
      <configuration>
        <source>1.8</source>
        <target>1.8</target>
      </configuration>
    </plugin>
  </plugins>
</build>
</project>
```

Para más información consulte: <https://maven.apache.org/guides/>

Precauciones y Recomendaciones

Recuerde verificar que el servidor de aplicaciones soporte el motor de base de datos que usará, de igual forma debe verificar que eclipse este haciendo uso del JDK y no del JRE y recuerde adicionar al workspace el servidor de aplicaciones Glassfish antes de crear cualquier proyecto. También puede ser importante verificar que los puertos usados por Glassfish no estén ocupados (Para ello puede hacer uso del comando **netstat -npl** o **netstat -a**).

Artefactos

Se requiere tener instalado el JDK y el IDE para el desarrollo de aplicaciones Eclipse (JEE en su última versión), un servidor de aplicaciones que cumpla con las especificaciones de JEE, para esta práctica Glassfish y el motor de base de datos Mysql.

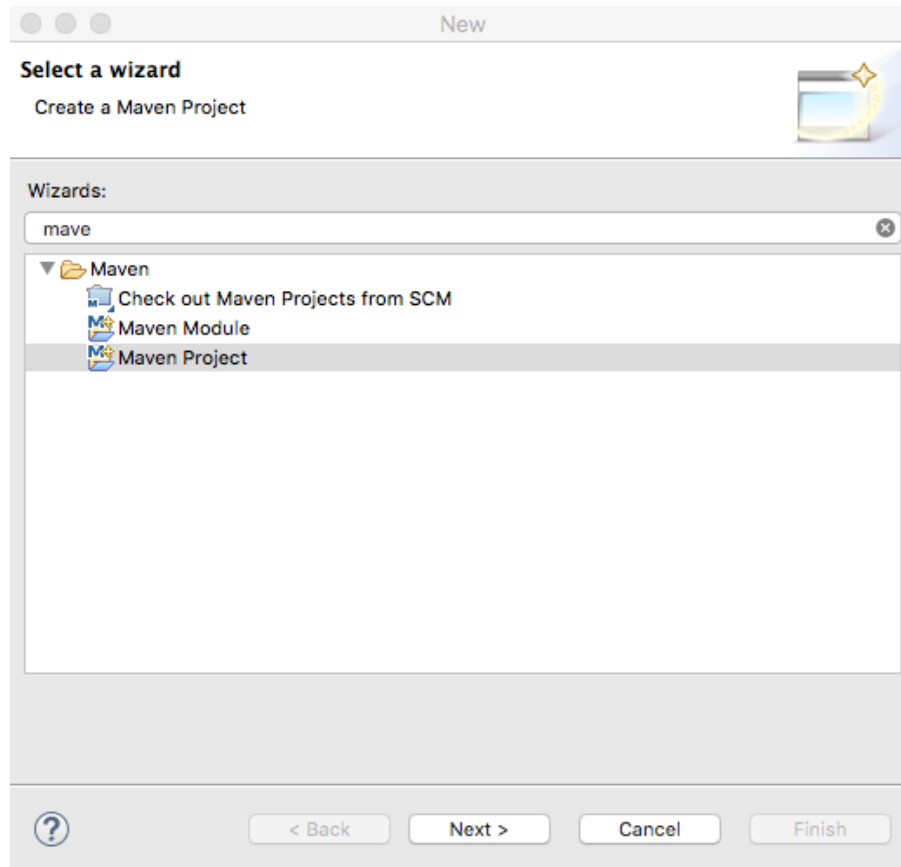
Evaluación o Resultado

Se espera que el alumno logre crear y realizar configuraciones básicas de proyectos usando maven.

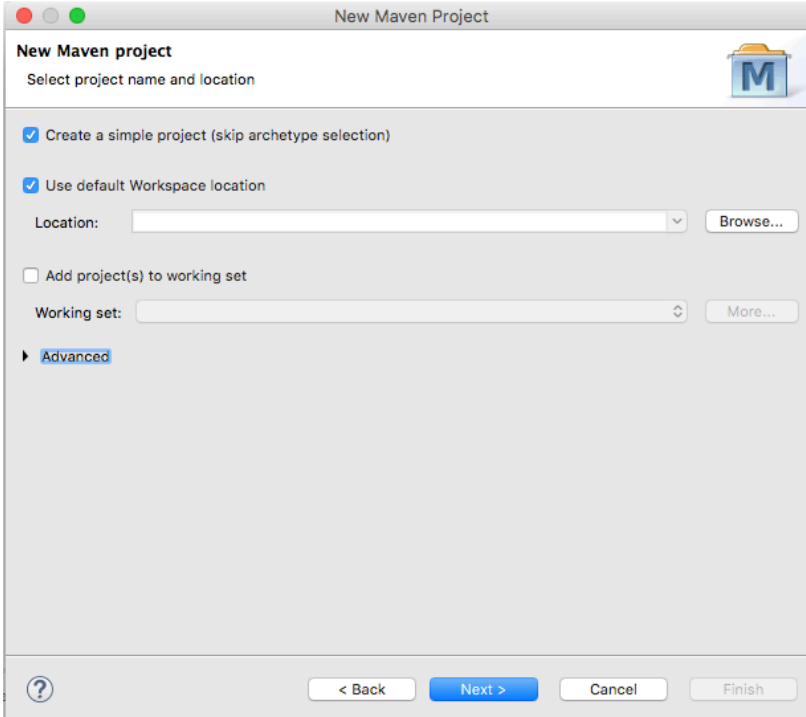
Procedimiento

1. Se iniciará creando un proyecto maven que permita contener otros proyectos, este tipo de proyecto es llamado proyecto padre. Para ello deberá:

En el menú file – new – Maven Project, Cree un Maven Project con configuración por defecto

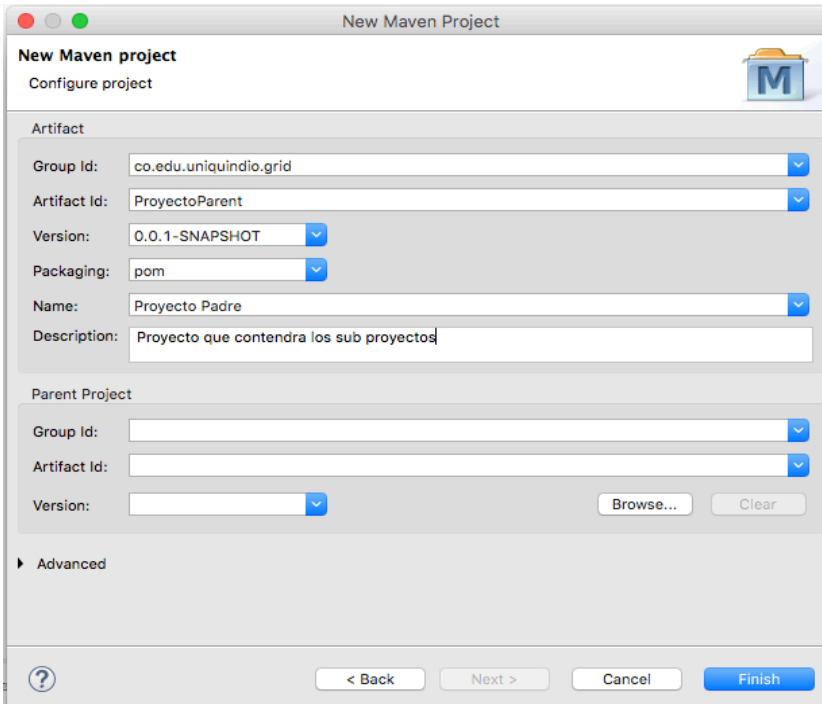


Next,



The screenshot shows the 'New Maven Project' dialog box with the title 'New Maven Project'. Below the title is the instruction 'Select project name and location'. There are two checked options: 'Create a simple project (skip archetype selection)' and 'Use default Workspace location'. A 'Location:' text box is followed by a dropdown arrow and a 'Browse...' button. Below this is an unchecked option 'Add project(s) to working set' with a 'Working set:' text box, a dropdown arrow, and a 'More...' button. A section titled 'Advanced' is collapsed. At the bottom are buttons for '< Back', 'Next >', 'Cancel', and 'Finish'.

Next,



The screenshot shows the 'New Maven Project' dialog box with the title 'New Maven Project'. Below the title is the instruction 'Configure project'. The 'Artifact' section contains: 'Group Id:' with the value 'co.edu.uniquindio.grid', 'Artifact Id:' with the value 'ProyectoParent', 'Version:' with the value '0.0.1-SNAPSHOT', 'Packaging:' with the value 'pom', 'Name:' with the value 'Proyecto Padre', and 'Description:' with the value 'Proyecto que contendrá los sub proyectos'. The 'Parent Project' section contains: 'Group Id:', 'Artifact Id:', and 'Version:' (all empty), with 'Browse...' and 'Clear' buttons. An 'Advanced' section is collapsed. At the bottom are buttons for '< Back', 'Next >', 'Cancel', and 'Finish'.

En este punto debe indicar al menos el groupid, el artefactid la versión y el tipo de empaquetado de su proyecto, en este caso el empaquetado es **POM**.

Finish.

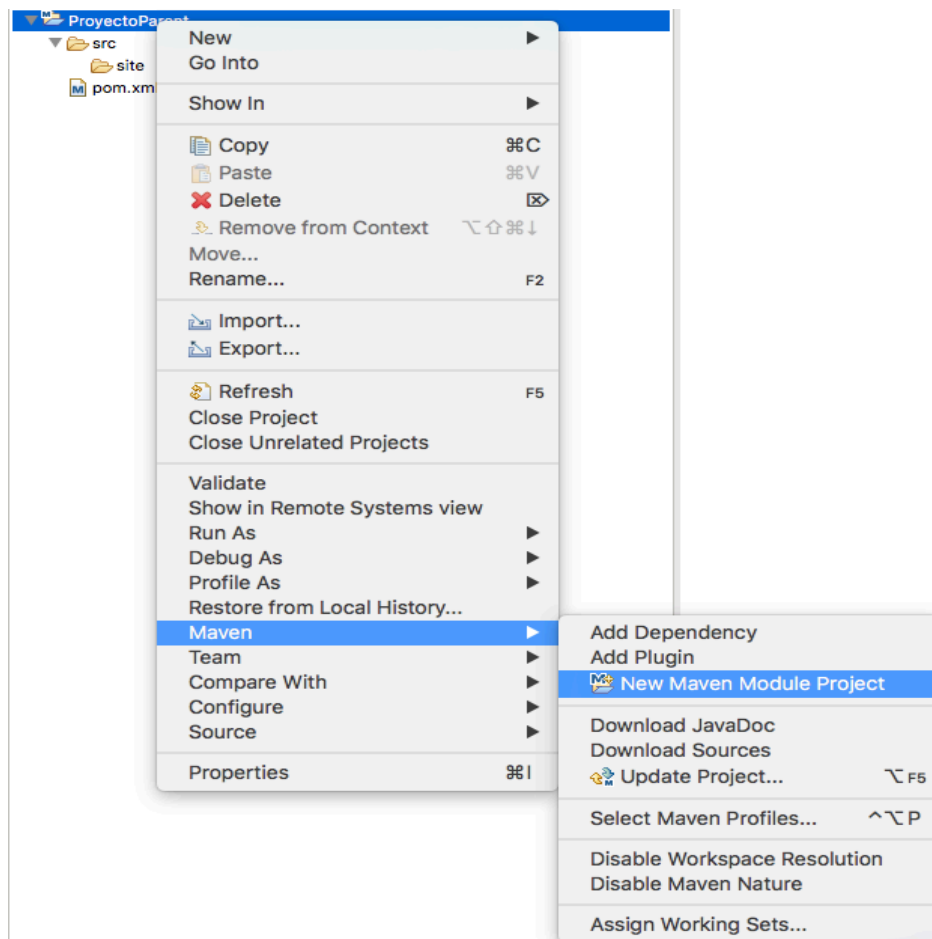
2. Modifique el archivo pom para establecer las configuraciones base de los plugins a ser usados en los proyectos hijos (agregue lo siguiente al final del pom).

```
<build>
  <pluginManagement>
    <plugins>
      <!-- Establecer la version de java a ser usada por los proyectos -->
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.7.0</version>
        <configuration>
          <source>1.8</source>
          <target>1.8</target>
        </configuration>
      </plugin>
      <!-- Establecer la version de EAR a ser usada -->
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-ear-plugin</artifactId>
        <version>2.10.1</version>
        <configuration>
          <version>7</version>
          <defaultLibBundleDir>lib</defaultLibBundleDir>
          <defaultJavaBundleDir>lib</defaultJavaBundleDir>
          <skinnyWars>true</skinnyWars>
          <generateModuleId>true</generateModuleId>
          <fileNameMapping>no-version</fileNameMapping>
          <archive>
            <manifest>
              <addClasspath>true</addClasspath>
            </manifest>
          </archive>
        </configuration>
      </plugin>
      <!-- Establecer la version de EJB -->
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-ejb-plugin</artifactId>
        <version>3.0.0</version>
        <configuration>
          <ejbVersion>3.2</ejbVersion>
        </configuration>
      </plugin>
      <!-- Establecer la version de WAR -->
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-war-plugin</artifactId>
        <version>3.2.0</version>
```

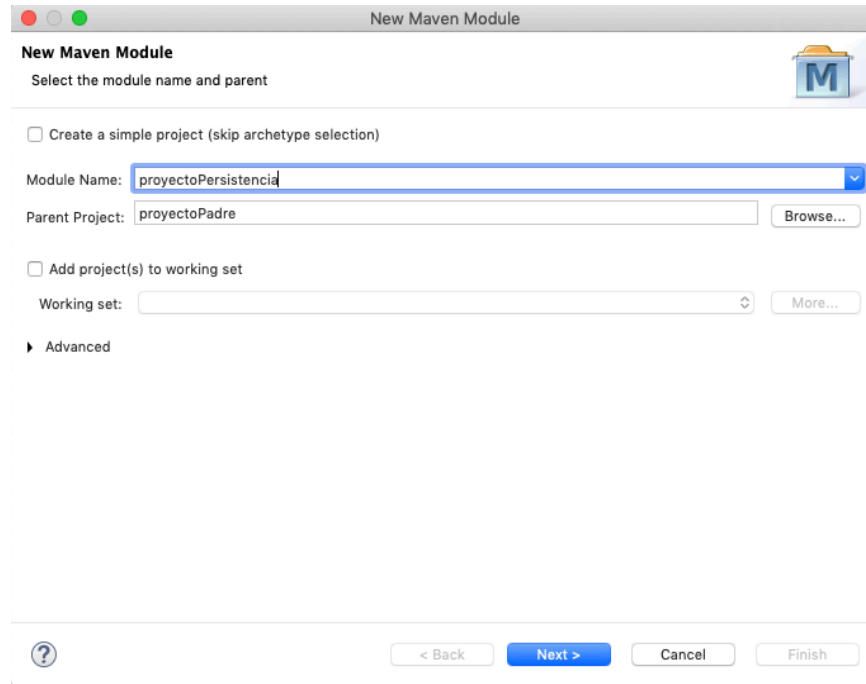
```
<configuration>
<failOnMissingWebXml>>false</failOnMissingWebXml>
</configuration>
</plugin>
</plugins>
</pluginManagement>
</build>
```

3. Crear proyecto para persistencia como hijo del proyecto padre.

Debe dar clic derecho sobre el proyecto padre y en la opción maven seleccionar new maven module Project, use la configuración por defecto.



De un nombre a su módulo de persistencia. Tal como se muestra en el siguiente ejemplo.



New Maven Module

Select the module name and parent

☐ Create a simple project (skip archetype selection)

Module Name:

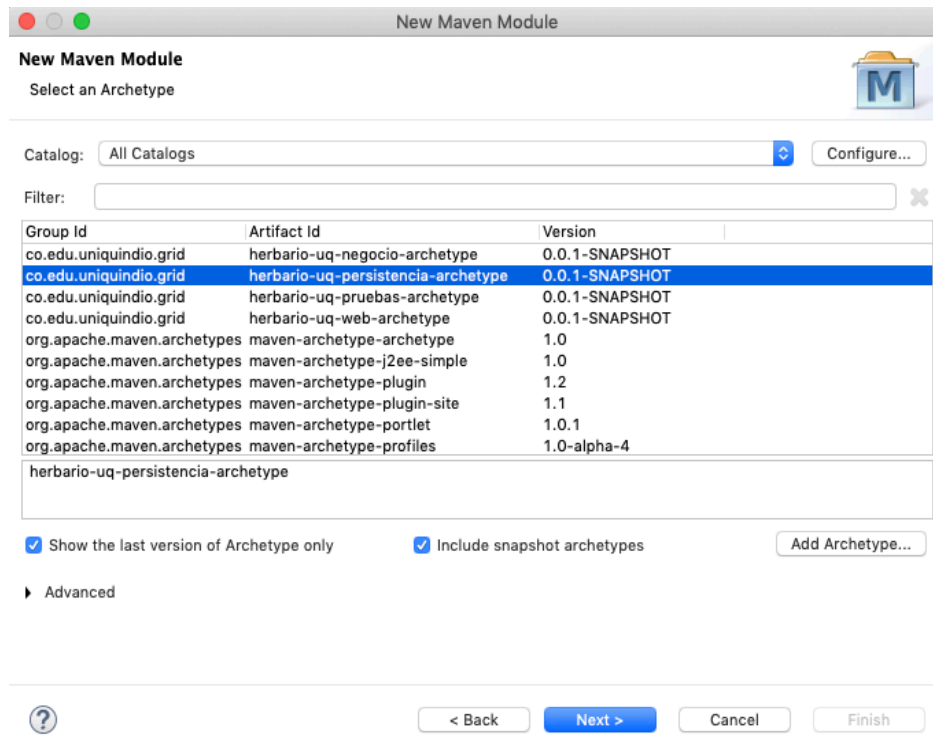
Parent Project:

☐ Add project(s) to working set

Working set:

► Advanced

Presione *Next*. Seleccione la opción *include snapshot archetypes* y seleccione el arquetipo de persistencia.



New Maven Module

Select an Archetype

Catalog:

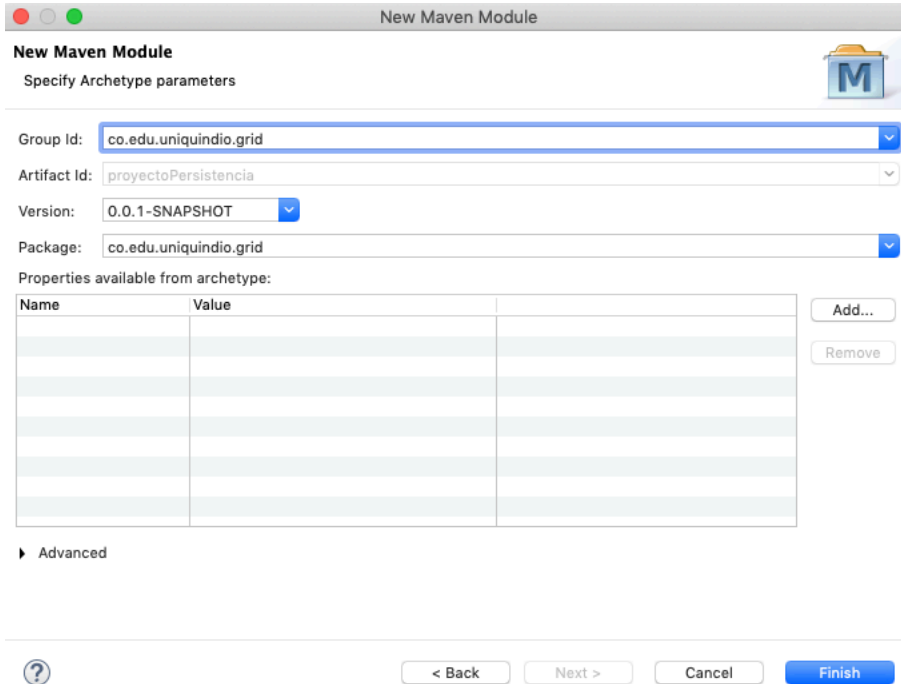
Filter:

Group Id	Artifact Id	Version
co.edu.uniquindio.grid	herbario-uw-negocio-archetype	0.0.1-SNAPSHOT
co.edu.uniquindio.grid	herbario-uw-persistencia-archetype	0.0.1-SNAPSHOT
co.edu.uniquindio.grid	herbario-uw-pruebas-archetype	0.0.1-SNAPSHOT
co.edu.uniquindio.grid	herbario-uw-web-archetype	0.0.1-SNAPSHOT
org.apache.maven.archetypes	maven-archetype-archetype	1.0
org.apache.maven.archetypes	maven-archetype-j2ee-simple	1.0
org.apache.maven.archetypes	maven-archetype-plugin	1.2
org.apache.maven.archetypes	maven-archetype-plugin-site	1.1
org.apache.maven.archetypes	maven-archetype-portlet	1.0.1
org.apache.maven.archetypes	maven-archetype-profiles	1.0-alpha-4
herbario-uw-persistencia-archetype		

☒ Show the last version of Archetype only ☒ Include snapshot archetypes

► Advanced

Presione *Next*, y agregue su *group id* y su paquete.



New Maven Module

Specify Archetype parameters

Group Id:

Artifact Id:

Version:

Package:

Properties available from archetype:

Name	Value

Advanced

< Back Next > Cancel Finish

Ahora presione *Finish*.

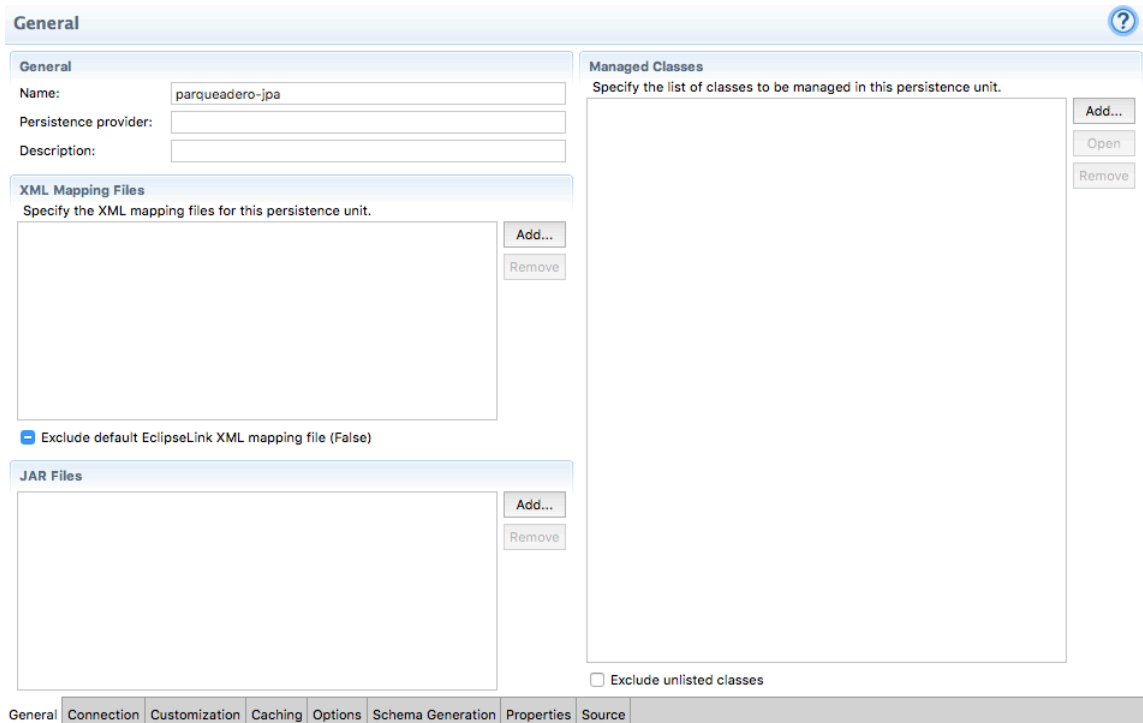
4. Ingrese al pom del proyecto de persistencia. Ahora se debería de ver similar a lo siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<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/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <parent>
    <groupId>co.edu.uniquindio.grid</groupId>
    <artifactId>proyectoPadre</artifactId>
    <version>0.0.1-SNAPSHOT</version>
  </parent>
  <artifactId>proyectoPersistencia</artifactId>
  <name>proyectoPersistencia</name>
  <dependencies>
    <dependency>
      <groupId>org.hibernate.javax.persistence</groupId>
      <artifactId>hibernate-jpa-2.1-api</artifactId>
      <version>1.0.0.Final</version>
      <scope>provided</scope>
    </dependency>
    <dependency>
      <groupId>javax</groupId>
      <artifactId>javaee-api</artifactId>
      <version>8.0</version>
```

```
<scope>provided</scope>
</dependency>
</dependencies>
</project>
```

Revise especialmente que los *group id* y los *artifactId* que sean correctos (los que usted ha elegido).

- Es necesario disponer de un Datasource en su servidor glassfish con acceso a una base de datos en mysql, si así lo desea puede hacer uso del creado en la Guia 02 o crear uno nuevo. Otra opción para aquellos que trabajan en los equipos de la Universidad es hacer uso del datasource jdbc/prueba.
- Configure el archivo persistence.xml para que haga uso de su datasource y descubra las entidades de forma automática. Para abrir el archivo, acceda a la sección Source Code de su proyecto, en ella al `src/java/resources` y al `Metainf`. De clic derecho sobre el archivo y seleccione abrir con Persistence Editor. En la pestaña General deselectione la opción *Exclude unlisted classes*. En la pestaña de Connection en la opción JTA Data Source ingrese el nombre de su data source. Luego ingrese a la pestaña *Schema Generation* y en la opción *Database* seleccione el valor *Create* y de igual forma en la opción *DDL Generation Type*.



The screenshot shows the 'General' tab of the Persistence Editor. It contains several sections: 'General' with fields for Name (parqueadero-jpa), Persistence provider, and Description; 'XML Mapping Files' with an 'Add...' button and a checkbox 'Exclude default EclipseLink XML mapping file (False)'; 'JAR Files' with an 'Add...' button; and 'Managed Classes' with a large text area for specifying classes to be managed, an 'Add...' button, and a checkbox 'Exclude unlisted classes'. At the bottom, there is a tabbed interface with 'General' selected, and other tabs like 'Connection', 'Customization', 'Caching', 'Options', 'Schema Generation', 'Properties', and 'Source'.

Connection

Persistence Unit Connection
Configure the data source or JDBC connection properties.

Transaction type:

Batch writing:

☒ Statement caching:

☒ Native SQL (False)

Database

JTA data source:

Non-JTA data source:

EclipseLink connection pool

[Populate from connection...](#)

Driver:

URL:

User:

Password:

☐ Bind parameters (True)

Read Connection

☐ Shared (False)

Minimum:

Maximum:

Write Connection

Minimum:

Maximum:

Exclusive connections

General **Connection** Customization Caching Options Schema Generation Properties Source

Schema Generation

Schema Generation

Schema generation

Database action:

Scripts generation:

Metadata and script creation:

Metadata and script dropping:

☒ Create database schemas (False)

Scripts create target:

Scripts drop target:

Database product name:

Database major version:

Database minor version:

Create script source:

Drop script source:

Connection:

Data loading

SQL load script source:

EclipseLink Schema Generation

DDL generation type:

Output mode:

DDL generation location:

Create DDL file name:

Drop DDL file name:

General **Connection** Customization Caching Options **Schema Generation** Properties Source

- De clic derecho en el proyecto padre y acceda al menú **maven – update project**

8. Adicione a su proyecto padre un nuevo maven module para pruebas usando el arquetipo de pruebas. Ajuste el pom de este nuevo proyecto agregando su *group id* y los *artifactId* que sean correctos.
9. Ahora ingrese al pom de pruebas, y elimine la siguiente dependencia.

```
<dependency>
  <groupId>co.edu.uniquindio.grid</groupId>
  <artifactId>herbario-uq-persistencia</artifactId>
  <version>0.0.1-SNAPSHOT</version>
</dependency>
```

10. Ahora ingrese ahora en su proyecto de pruebas a `src/test/resources/sun-resources.xml`. Este archivo será el encargado de crear de forma automática el pool de conexiones y el data source que se usaran en las pruebas.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE resources PUBLIC "-//Sun Microsystems, Inc.//DTD Application Server 9.0
Resource Definitions //EN" "http://www.sun.com/software/appserver/dtds/sun-
resources_1_3.dtd">
<resources>
  <jdbc-connection-pool
    name="mysql-proyecto-pool"
    datasource-classname="com.mysql.jdbc.jdbc2.optional.MysqlDataSource"
    res-type="javax.sql.DataSource">
    <property name="user" value="root"/>
    <property name="password" value="12345"/>
    <property name="url" value="jdbc:mysql://127.0.0.1:3306/test"/>
  </jdbc-connection-pool>
  <jdbc-resource
    enabled="true"
    jndi-name="jdbc/test"
    object-type="user"
    pool-name="mysql-proyecto-pool"/>
</resources>
```

Note que el atributo *user* debe corresponder el usuario empleado para conexión a su base de datos, así mismo el atributo *password* deberá contener la clave de acceso de dicho usuario. Por otra parte, el atributo *url* tiene la siguiente estructura:

jdbc:mysql://SERVIDOR_DONDE_ESTA_LA_BD:PUERTO_DEL_MOTOR_BD/NOMBRE_BD

en el caso del ejemplo se ha establecido 127.0.0.1 como la dirección del equipo donde está la base de datos, dicha dirección indica que la base de datos está en el equipo local. Para el puerto se ha dejado el puerto por defecto 3306, finalmente se está haciendo uso de la base de datos *test* de mysql.

11. En la misma carpeta del punto anterior puede encontrar un archivo con nombre **arquillian.xml**, este archivo contiene la configuración básica de arquillian.

```
<?xml version="1.0" encoding="UTF-8"?>
<arquillian xmlns="http://jboss.org/schema/arquillian"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```

        xsi:schemaLocation="http://jboss.org/schema/arquillian
        http://jboss.org/schema/arquillian/arquillian_1_0.xsd">
    <container qualifier="glassfish" default="true">
        <configuration>
            <property name="resourcesXml">src/test/resources/sun-resources.xml
            </property>
        </configuration>
    </container>
    <extension qualifier="persistence-script">
    <!-- Estos son para desactivar la verificación de llaves foraneas en mysql -->
    <property name="scriptsToExecuteBeforeTest">SET foreign_key_checks = 0;</property>
    <property name="scriptsToExecuteAfterTest">SET foreign_key_checks = 1;</property>
    </extension>
</arquillian>

```

12. En la misma carpeta de los puntos anteriores observe el archivo persistenceForTest.xml, este archivo contiene la configuración básica de la persistencia usada para las pruebas.

```

<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1"
    xmlns="http://xmlns.jcp.org/xml/ns/persistence"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
    http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
    <persistence-unit name="asesoriauq-persistencia">
        <jta-data-source>jdbc/test</jta-data-source>
        <exclude-unlisted-classes>>false</exclude-unlisted-classes>
        <properties>
            <property
                name="javax.persistence.schema-generation.database.action"
                value="create" />
            <property name="eclipselink.ddl-generation"
                value="create-tables" />
        </properties>
    </persistence-unit>
</persistence>

```

Para la próxima clase

Leer y entender el concepto de entidad en Java
 Leer y entender el concepto de base de datos relacional.
 Leer y entender el concepto de JavaBeans.
 Leer y entender el concepto de Enterprise JavaBeans.
 Leer y entender el concepto de Java Persistence Api.