El concepto de PersistenceUnit es uno de los conceptos claves de JPA. Este elemento es el encargado de configurar el acceso a la base de datos de tal forma que nuestras entidades puedan ser persistidas o seleccionadas de forma transparente .Vamos a construir un ejemplo que lo use y ver sus opciones más importantes. Para ello el primer paso es instalar las dependencias que JPA necesita.

```
<dependencies>

		<!--
https://mvnrepository.com/artifact/org.hibernate.javax.persistence/hib
ernate-jpa-2.1-api -->
		<dependency>
<groupId>org.hibernate.javax.persistence</groupId>
			<artifactId>hibernate-jpa-2.1-api</artifactId>
			<version>1.0.2.Final</version>
		</dependency>
		<!--
https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->
		<dependency>
			<groupId>org.hibernate</groupId>
			<artifactId>hibernate-core</artifactId>
			<version>5.4.11.Final</version>
		</dependency>
		<!--
https://mvnrepository.com/artifact/mysql/mysql-connector-java -->
		<dependency>
			<groupId>mysql</groupId>
			<artifactId>mysql-connector-java</artifactId>
			<version>6.0.6</version>
		</dependency>
```

```
        </dependencies>
```

PersistenceUnit y Persistence.xml

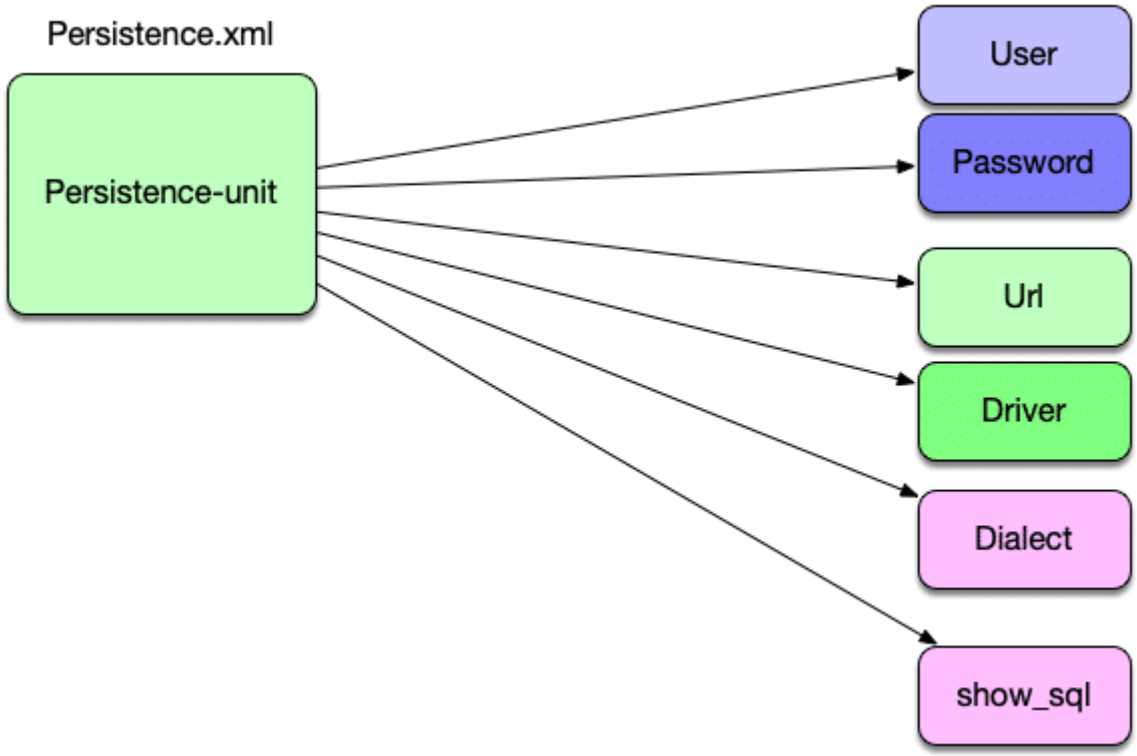
Es momento de ver el contenido del fichero de persistencia y hablar de sus opciones fundamentales.

```
<?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="unidadLibros">
 <class>com.arquitecturajava.jpabasico.dominio.Libro</class>
 <properties>
 <property name= "hibernate.show_sql" value="true" />
 <property name="hibernate.dialect"
value="org.hibernate.dialect.MySQLDialect" />
 <property name="javax.persistence.jdbc.driver"
value="com.mysql.jdbc.Driver" />
 <property name="javax.persistence.jdbc.user" value="root" />
 <property name="javax.persistence.jdbc.password" value="root" />
 <property name="javax.persistence.jdbc.url"
value="jdbc:mysql://localhost/jpa1" />
</properties>

 </persistence-unit>
</persistence>
```

El fichero dispone de lo que se denomina un PersistenceUnit o unidad de persistencia que

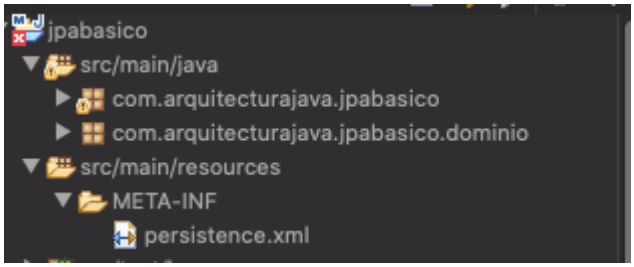
es la encargada de aglutinar todos los datos que nuestra aplicación Java necesitará para conectarse a la base de datos.



En este caso tenemos seis propiedades y vamos a hablar un poco de ellas

- User: Define el usuario de conexión a la base de datos
- Password: Define la contraseña
- URL: Define la url a través de la cual accedemos a la base de datos
- Driver: El tipo de Driver que queremos usar . En este caso nos conectamos a una base de datos mySQL.
- Dialect: El dialecto , esta ya es una propiedad muy ligada a hibernate ya que define el dialecto con el cual JPA QL transforma su query a una query SQL de la base de datos.
- show_sql: Uno de los parámetros más típicos ya que nos permite ver que consultas se ejecutan

Una vez configurado el fichero de persistencia . Vamos a utilizarlo en un proyecto muy sencillo de eclipse que seleccione libros de una base de datos.



Vamos a ver el código del programa principal

```
package com.arquitecturajava.jpabasico;

import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

import com.arquitecturajava.jpabasico.dominio.Libro;

public class Principal002Buscar {

        public static void main(String[] args) {

                Libro miLibro = new Libro("1A");

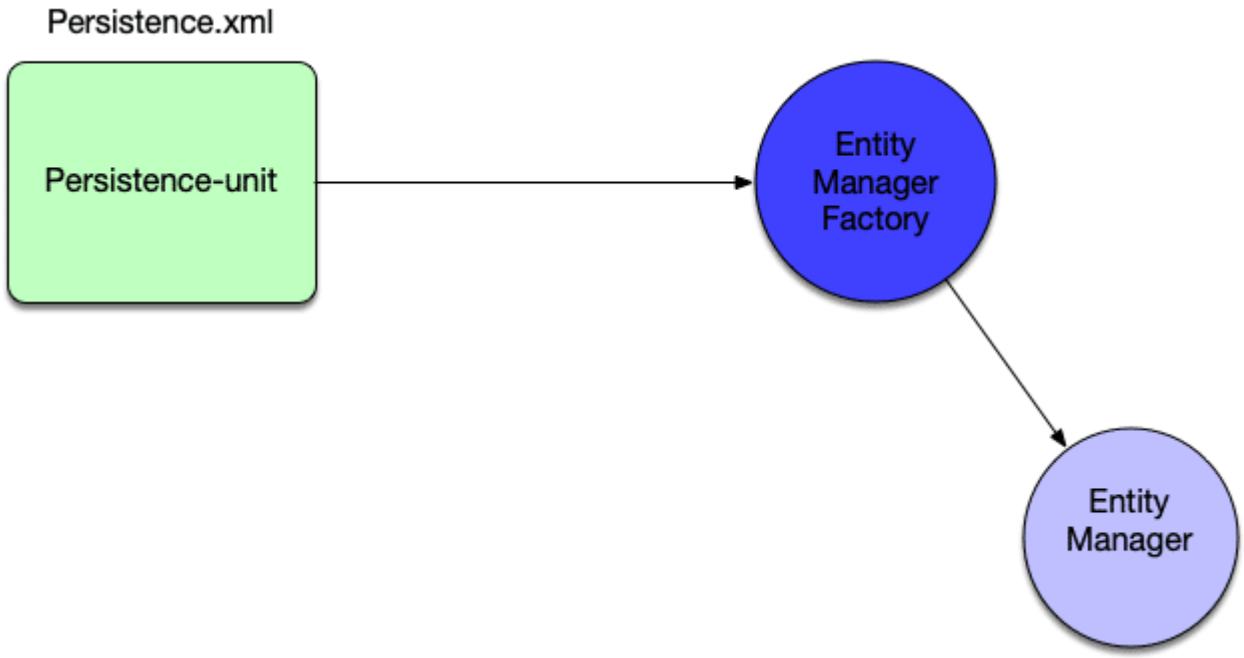
                EntityManagerFactory emf =
Persistence.createEntityManagerFactory("unidadLibros");
                EntityManager em = emf.createEntityManager();
                try {
                Libro libro= em.find(Libro.class,"1A");
                System.out.println(libro.getIsbn());
                System.out.println(libro.getAutor());
                System.out.println(libro.getTitulo());
                } catch (Exception e) {
                        e.printStackTrace();
```

```
                } finally {
                        em.close();

                }

        }
}
```

En este caso utilizamos el objeto Persistence para obtener un entityManagerFactory que apunte a un PersistenceUnit este nos genera un EntityManager que es el objeto que necesitamos.



Este es un ejemplo sencillo de como usar el Persistence.xml y conocer sus propiedades elementales.

Otros artículos relacionados

- JPA one to one
- JPA Merge

- JPA Criteria
- JPA