Tarea No. 13. Archivo persistence.xml

Barrera Pérez Carlos Tonatihu
Profesor: José Asunción Enríquez Zárate
Web Application Development
Grupo: 3CM9

8 de abril de 2019

1. Introducción

El archivo persistence.xml es la parte fundamental en la configuración de JPA. Este archivo define una o mas unidades de persistencia en las cuales puedes configurar parámetros como:

- El nombre de la unidad de persistencia.
- Cuales seran las clases de persistencia que son parte de la unidad de persistencia.
- El como las clases son mapeadas en la base de datos.
- El proveedor de persistencia que sera utilizado.
- La fuente de datos que se utilizara para conectarse a la base de datos.
- Como crear y validar el esquema de la base de datos.
- El modo del cache de segundo nivel.
- Muchas configuraciones especificas de proveedor.

Es por esto que en esta tarea se exploraran los principales parámetros de configuración que se tienen en este archivo.

2. Desarrollo

El siguiente es un ejemplo de un archivo persistence.xml con la respectiva descripción de cada atributo y etiqueta.

```
1 <persistence xmlns="http://java.sun.com/xml/ns/persistence"
2     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
4 http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd"
5     version="2.0">
6   <persistence-unit name="manager1" transaction-type="JTA">
7     <provider>org.hibernate.ejb.HibernatePersistence</provider
    >
8     <jta-data-source>java:/DefaultDS</jta-data-source>
9     <mapping-file>ormap.xml</mapping-file>
10    <jar-file>MyApp.jar</jar-file>
11    <class>org.acme.Employee</class>
12    <class>org.acme.Person</class>
13    <class>org.acme.Address</class>
```

```
14        <shared-cache-mode>ENABLE_SELECTOVE</shared-cache-mode>
15        <validation-mode>CALLBACK</validation-mode>
16        <properties>
17            <property name="hibernate.dialect"
18 value="org.hibernate.dialect.HSQLDialect" />
19            <property name="hibernate.hbm2ddl.auto" value="create-
      drop" />
20        </properties>
21    </persistence-unit>
22 </persistence>
```

- **name**. Este atributo defiene el nombre del entity manager.
- **transaction-type**. Este atributo define el tipo de transacción a utilizar. Ya sea JTA o RESOURCE_LOCAL, JTA es la que se utiliza por defecto en un ambiente JavaEE y RESOURCE_LOCAL en JavaSE.
- **provider**. Es el nombre completo del proveedor de persistencia EJB.
- **jta-data-source, non-jta-data-source**. Este es el nombre del JDNI donde se ubica el DataSource.
- **mapping-file**. El elemento de clase que especifica un archivo de asignación XML compatible con EJB3 que se asignara.
- **jar-file**. Especifica un jar a analizar. Todos los elementos de este jar son agregados a la configuración de la unidad de persistencia. Este elemento es principalmente utilizado in JavaEE.
- **exclude-unlisted-classes**. No revisa el archivo jar principal para las clases anotadas. Solo las clases explicitas serán parte de la unidad de persistencia.
- **class**. Especifica el nombre completo de la clase que se va a mapear.
- **shared-cache-mode**. Por defecto, las entidades son elegidas para el segundo nivel de caché si son anotadas con @Cacheable. Sin embargo, se puede utilizar lo siguiente:
  - ALL. Forzar el caché para todas las entidades.
  - NONE. Deshabilitado para todas las entidades.
  - ENABLE_SELECTIVE (default). Habilitar caché explícitamente marcado.

    - DISABLE_SELECTIVE. Habilitar caché a menos que se encuentre explícitamente marcado con @Cacheable(false)

- **validation-mode**. Por defecto la validación Bean está activada. Cuando una entidad es creada, actualizada o borrada, es validada antes de mandarla a la base de datos. El esquema de la base de datos que es generado por Hibernate también refleja estas restricciones declaradas en la entidad. Se pueden hacer pequeños cambios si son necesarios, tales como:

    - AUTO. Si la validación bean está presente en el classpath, CALLBACK y DDL están activados.
    - CALLBACK. Las entidades son validadas en su creación, actualización y borrado. Si no hay proveedor de validación, se lanzara una excepción a la hora de la inicialización.
    - DDL. (No es estándar). El esquema de la base de datos son entidades que se validan en la creación, actualización y borrado. Si no hay proveedor de validación, se genera una excepción en el momento de la inicialización.
    - NONE. No se utiliza validación.

3. Conclusiones

Es evidente la gran cantidad de parámetros que se pueden configurar en el archivo persistene.xml, sin embargo, al conocernos evita que uno se pueda sentir abrumado por todas las posibles configuraciones que se pueden lograr con tan solo modificar este archivo. Y al igual que con el archivo web.xml, el conocer el como funciona cada parámetro nos puede ahorrar tiempo y facilitar el proceso de desarrollo.