

Hibernate

Java Persistence API (JPA)

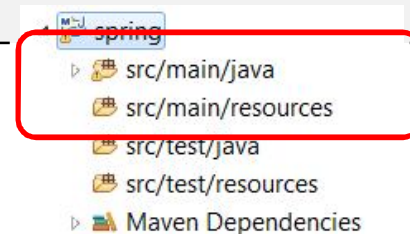
JPA introduce la idea de “**unidad de persistencia**”. Una unidad de persistencia provee una manera de especificar un conjunto de clases anotadas o clases que persistirán, junto con las propiedades del proveedor de JPA que se utilizará para esa unidad. La unidad de persistencia tiene un nombre y ese nombre es usado para crear en *run time* un **EntityManagerFactory**.

persistence.xml

```
<persistence xmlns="http://java.sun.com/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd" version="2.0">
<persistence-unit name="unlp">
    <provider>org.hibernate.ejb.HibernatePersistence</provider>
    <class>paquete.clase1</class>
    .
    .
    .
    <properties>
        <property name="hibernate.connection.driver_class" value="com.mysql.jdbc.Driver"/>
        <property name="hibernate.connection.password" value="admin"/>
        <property name="hibernate.connection.url" value="jdbc:mysql://localhost:3306/sbarra"/>
        <property name="hibernate.connection.username" value="root" />
        <property name="hibernate.dialect" value="org.hibernate.dialect.MySQL5InnoDBDialect />
        <property name="hibernate.hbm2ddl.auto" value="update/create"/>
        .
        .
        .
    </properties>
</persistence-unit>
</persistence>
```

← Recordar configurar el C3P0
para el pool de conexiones

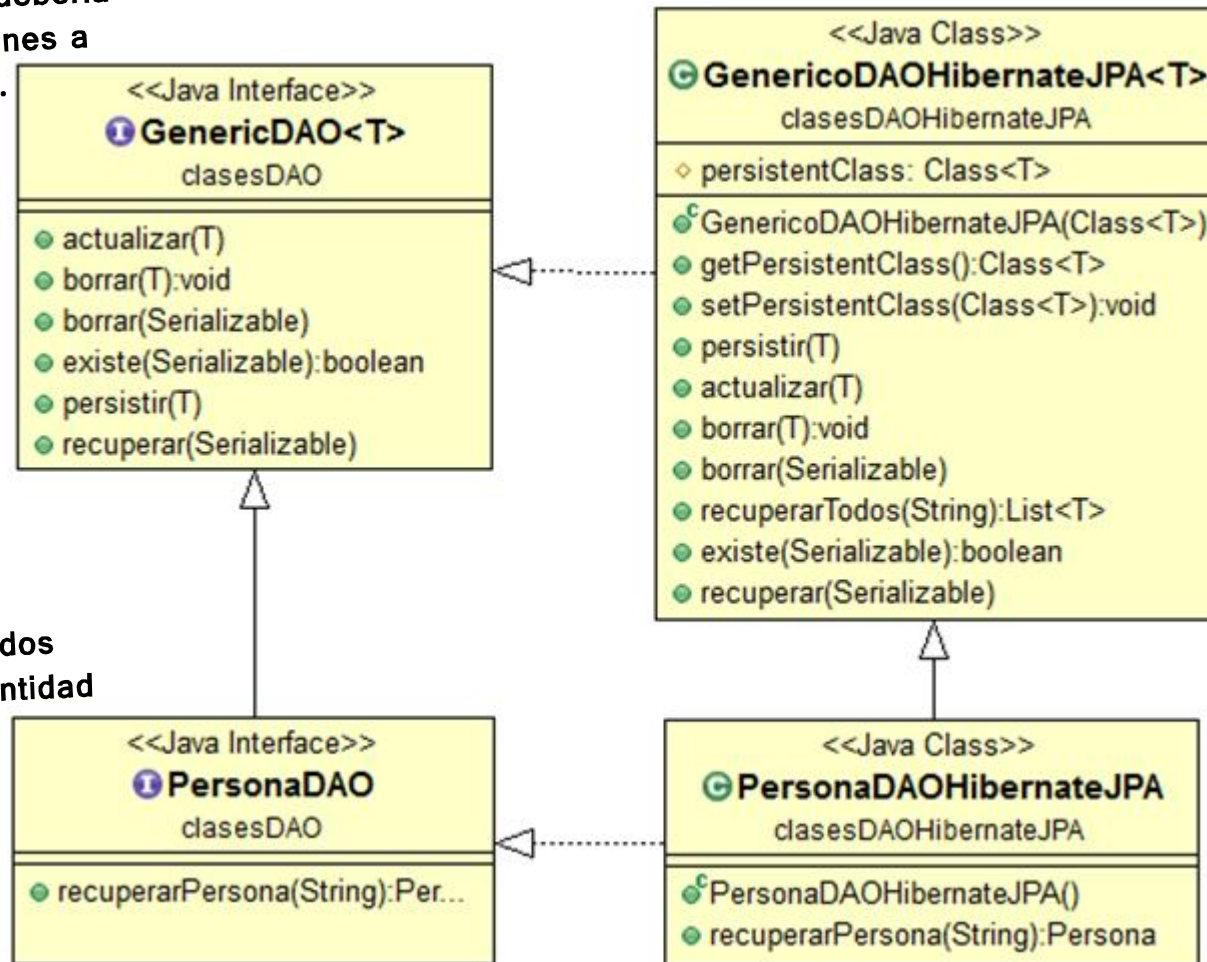
El archivo se ubica en alguna de las carpetas cuyo contenido es enviado a la carpeta **WEB-INF/classes/META-INF**



DAOs genéricos

La interface GenericDAO debería incluir los métodos comunes a todas las entidades.

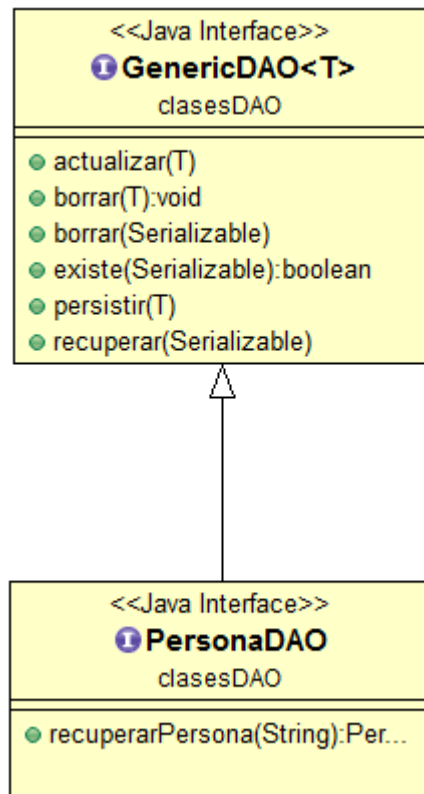
Se agregan métodos particulares de la entidad Persona.



Se implementan los métodos particulares de PersonaDAO

DAOs genéricos

Las interfaces



```
package clasesDAO;
import java.io.Serializable;
import java.util.List;

public interface GenericDAO<T> {
    public T actualizar(T entity);
    public void borrar(T entity);
    public T borrar(Serializable id);
    public boolean existe(Serializable id);
    public T persistir(T entity);
    public T recuperar(Serializable id);
}
```

```
package clasesDAO;
import modelos.Persona;

public interface PersonaDAO extends GenericDAO<Persona> {
    public Persona recuperarPersona(String usuario);
}
```

DAOs genéricos

Implementación del DAO genérico

```
package dao.impl;
//imports
public class GenericDAOHibernateJPA<T>
    implements GenericDAO<T> {
    protected Class<T> persistentClass;

    @Override
    public T persistir(T entity) {
        EntityManager em = EMF.getEMF().createEntityManager();
        EntityTransaction tx = null;
        try {
            tx = em.getTransaction();
            tx.begin();
            em.persist(entity);
            tx.commit();
        }
        catch (RuntimeException e) {
            if ( tx != null && tx.isActive() ) tx.rollback();
            throw e; // escribir en un log o mostrar un mensaje
        }
        finally {
            em.close();
        }
        return entity;
    }

    public T actualizar(T entity) {
        EntityManager em= EMF.getEMF().createEntityManager();
        EntityTransaction etx= em.getTransaction();
        etx.begin();
        T entity = em.merge(entity);
        etx.commit();
        em.close();
        return entity;
    }
    . . .
}
```

```
@Override
public void borrar(T entity) {
    EntityManager em = EMF.getEMF().createEntityManager();
    EntityTransaction tx = null;
    try {
        tx = em.getTransaction();
        tx.begin();
        em.remove(entity);
        tx.commit();
    }
    catch (RuntimeException e) {
        if ( tx != null && tx.isActive() ) tx.rollback();
        throw e; // escribir en un log o mostrar un mensaje
    }
    finally {
        em.close();
    }
}

public T borrar(Serializable id) {
    T entity =
    EMF.getEMF().createEntityManager().find(this.getPersistentClass(), id);
    if (entity != null) {
        this.borrar(entity);
    }
    return entity;
}

public List<T> recuperarTodos(String columnOrder) {
    Query consulta= EMF.getEMF().createEntityManager().createQuery("select
        e from " + getPersistentClass().getSimpleName()+" e order by
        e."+columnOrder);
    List<T> resultado = (List<T>)consulta.getResultList();
    return resultado;
}
. . .
}
```

DAOs genéricos

Clases y utilitarias y PersonaDAOHibernateJPA

```
package dao.impl;

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

public class EMF {
    private static final EntityManagerFactory em =
        Persistence.createEntityManagerFactory("mapeos");

    public static EntityManagerFactory getEMF() {
        return em;
    }
}
```

```
package dao.impl;

import dao.*;
import dao.impl.*;

public class DaoFactory {

    public static PersonaDAO getPersonaDAO() {

        return new PersonaDAOHibernateJPA();
    }
    . . . // Retornaría los diferentes DAOs
}
```

```
package dao.impl;

import javax.persistence.EntityTransaction;
import javax.persistence.Query;
import dao.PersonaDAO;
import entities.Persona;

public class PersonaDAOHibernateJPA extends
    GenericDAOHibernateJPA<Persona> implements PersonaDAO {

    public PersonaDAOHibernateJPA() {
        super(Persona.class);
    }

    /** esté método es a modo de ejemplo, se agregaría para cosas
     * particulares de la entidad Persona
     */
    @Override
    public Persona recuperarPersona(String nombre) {
        Query consulta = EMF.getEMF().createEntityManager().
            createQuery("select p from Persona p where p.nombre =?");
        consulta.setParameter(1, nombre);
        Persona resultado = (Persona)consulta.getSingleResult();
        return resultado;
    }
}
```