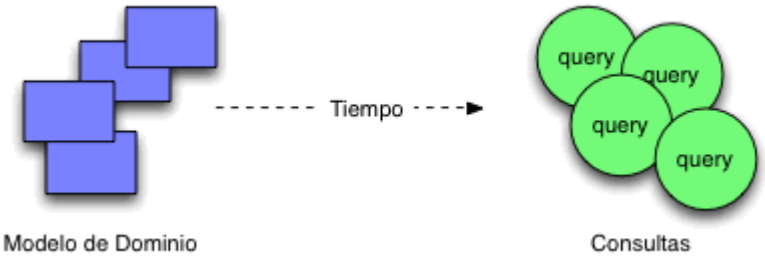
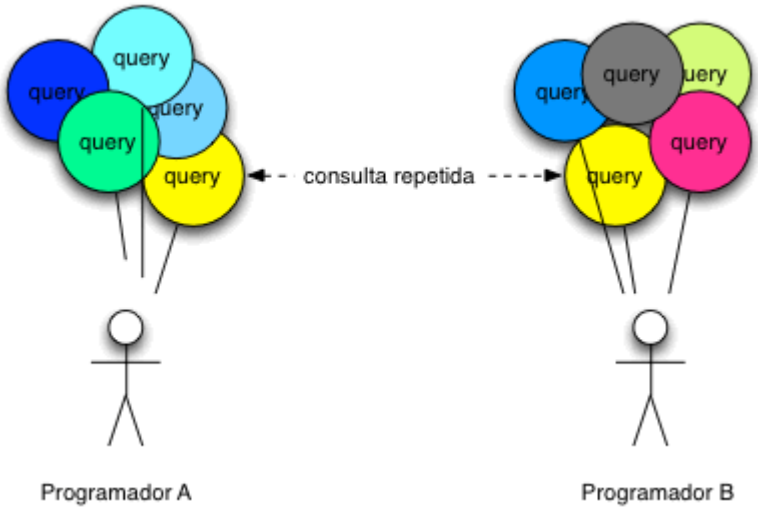
JPA es algo con lo que trabajamos de forma habitual. Una parte del trabajo que tenemos que realizar es construir el modelo de dominio que normalmente lleva una esfuerzo inicial importante . Sin embargo cuando estamos construyendo la aplicación el modelo de dominio suele estar ya asentado y el mayor esfuerzo de desarrollo pasa por construir las diferentes consultas a realizar contra el modelo.

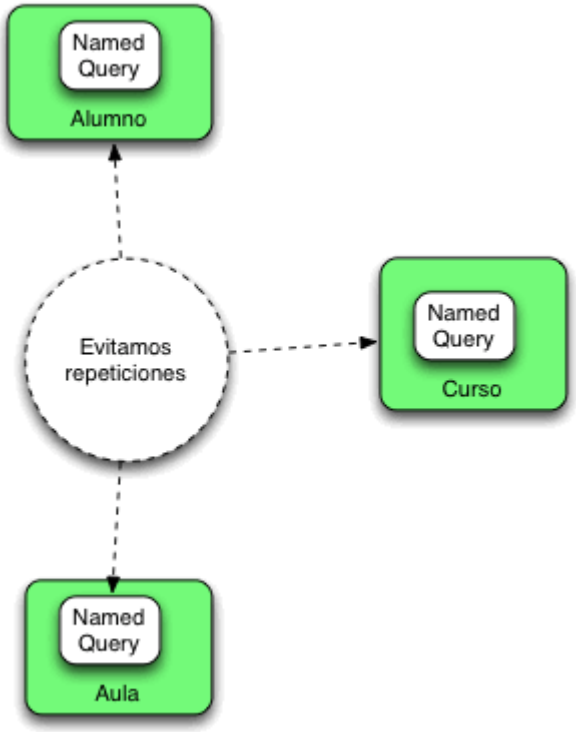


Consultas

Uno de los problemas que nos podemos encontrar cuando construimos las consultas es que varios desarrolladores van a trabajar en el mismo proyecto construyendo consultas. Con lo cual puede que acabemos con consultas repetidas en distintas partes del código .



Para evitar este problema podemos apoyarnos en JPA NamedQueries que nos permiten definir las consultas a nivel de clase de dominio evitando las repeticiones.



Vamos a verlo a través de un ejemplo en código:

```
package com.arquitecturajava;

import java.io.Serializable;

import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
@Entity
@NamedQueries({
@NamedQuery(name="AlumnosNombre", query="select a from Alumno a where
a.nombre=:nombre")
})
public class Alumno implements Serializable{

private static final long serialVersionUID = 1L;

@Id
private String dni;
private String nombre;
private String apellidos;
private int edad;

public Alumno() {
super();
}

public Alumno(String dni, String nombre, String apellidos, int edad) {
super();
this.dni = dni;
this.nombre = nombre;
```

```
this.apellidos = apellidos;
this.edad = edad;
}

public String getDni() {
return dni;
}
public void setDni(String dni) {
this.dni = dni;
}
public String getNombre() {
return nombre;
}
public void setNombre(String nombre) {
this.nombre = nombre;
}
public String getApellidos() {
return apellidos;
}
public void setApellidos(String apellidos) {
this.apellidos = apellidos;
}
public int getEdad() {
return edad;
}
public void setEdad(int edad) {
this.edad = edad;
}
}

&amp;amp;amp;nbsp;
```

En este caso hemos definido una clase Alumno y una NamedQuery denominada “AlumnosNombre” que nos busca a los alumnos por nombre. Una vez hecho esto podemos usar JPA y realizar la consulta en un programa sencilla.

```
package com.arquitecturajava.main;

import java.util.List;

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

import com.arquitecturajava.Alumno;
public class Principal003SeleccionAlumnosWhereNamed2 {

public static void main(String[] args) {

EntityManagerFactory emf =
Persistence.createEntityManagerFactory("UnidadCurso");
EntityManager em = emf.createEntityManager();
TypedQuery&amp;amp;lt;Alumno&amp;amp;gt; consultaAlumnos=
```

```
em.createNamedQuery("seleccionarAlumnosNombre", Alumno.class);
consultaAlumnos.setParameter("nombre", "miguel");
List&amp;amp;lt;Alumno&amp;amp;gt; lista=
consultaAlumnos.getResultList();

System.out.println("*************Alumnos*********");
for (Alumno a :lista) {

System.out.println(a.getNombre()+"," +a.getApellidos());

}

em.close();
System.out.println("termino");

}

}
```

En este caso hemos usado el EntityManager y su método createNamedQuery para referenciar a la consulta que hemos definido a nivel de la clase Alumno a través de las anotaciones.

```
Hibernate: select alumno0_.dni
*************Alumnos*********
angel,perez
angel,sanchez
termino
```

APUNTATE!!

Otros artículos relacionados:

Introducción a JPA

JPA OneToMany

Libro de JPA