

CURSO HIBERNATE Y JPA

EJERCICIO

API DE CRITERIA CON HIBERNATE/JPA



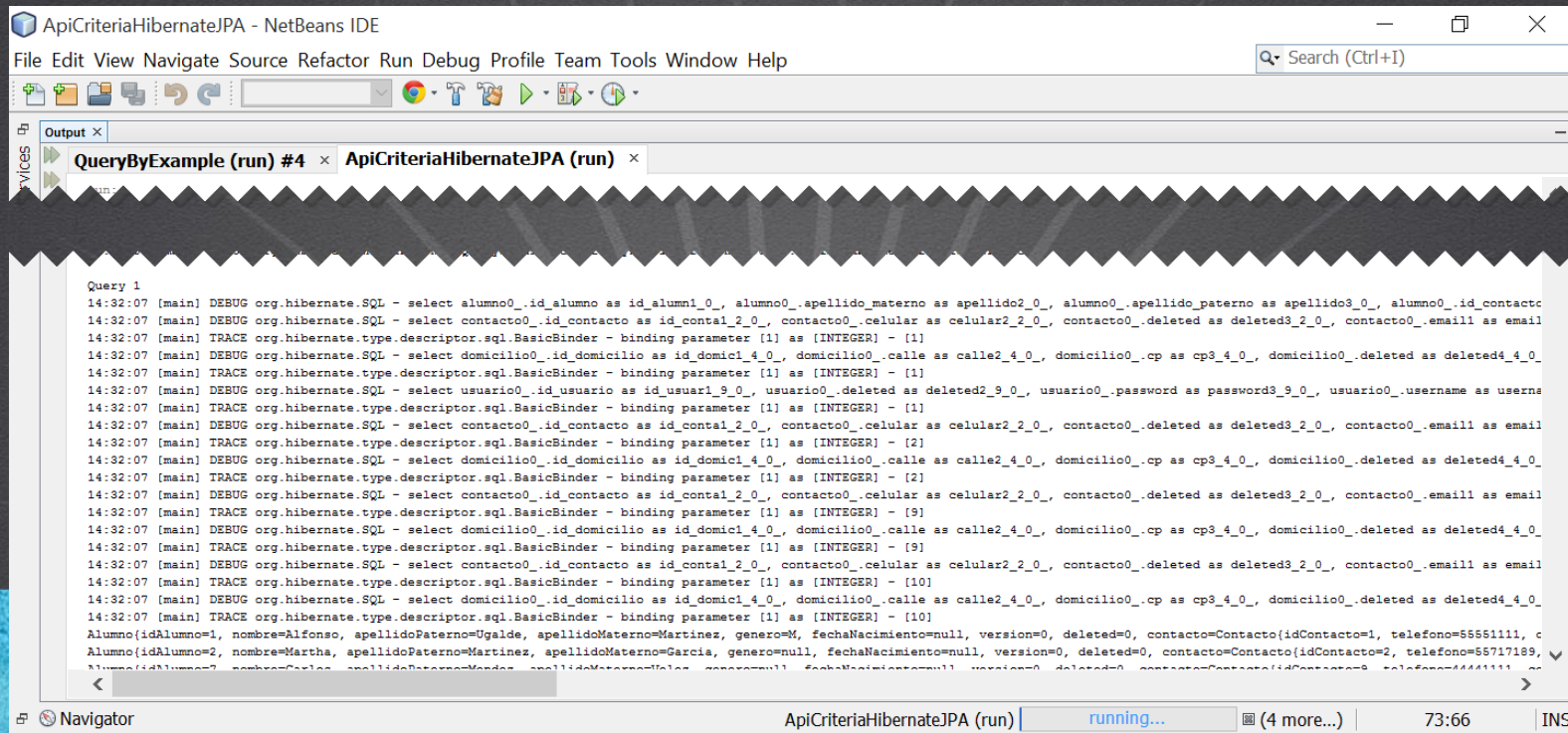
Experiencia y Conocimiento para tu vida

CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

OBJETIVO DEL EJERCICIO

Utilizar el API de Criteria para poner en práctica esta API. Al finalizar deberemos observar lo siguiente:

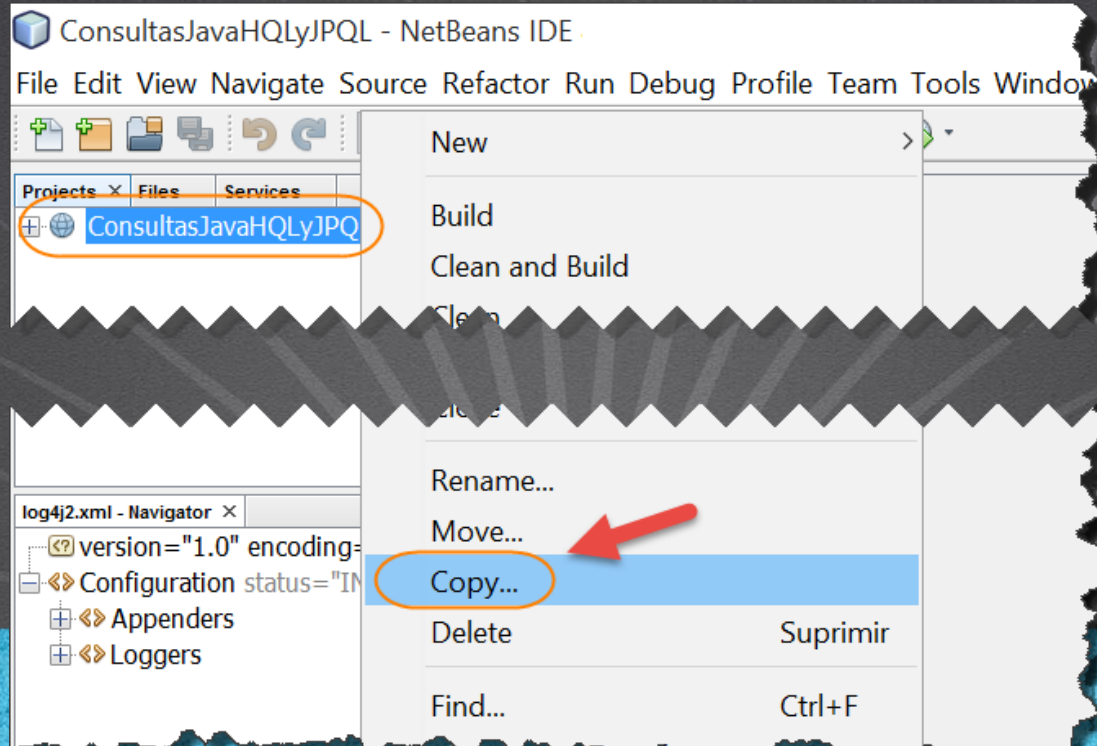


The screenshot shows the NetBeans IDE interface for the project 'ApiCriteriaHibernateJPA'. The 'Output' window is open, displaying the logs for 'QueryByExample (run) #4'. The logs show the execution of a query using the Criteria API, with various SQL statements and parameter bindings. The query results are displayed at the bottom of the output window.

```
Query 1
14:32:07 [main] DEBUG org.hibernate.SQL - select alumno0_.id_alumno as id_alumn1_0_, alumno0_.apellido_materno as apellido2_0_, alumno0_.apellido_paterno as apellido3_0_, alumno0_.id_contacto
14:32:07 [main] DEBUG org.hibernate.SQL - select contacto0_.id_contacto as id_conta1_2_0_, contacto0_.celular as celular2_2_0_, contacto0_.deleted as deleted3_2_0_, contacto0_.email as email
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
14:32:07 [main] DEBUG org.hibernate.SQL - select domicilio0_.id_domicilio as id_domic1_4_0_, domicilio0_.calle as calle2_4_0_, domicilio0_.cp as cp3_4_0_, domicilio0_.deleted as deleted4_4_0_
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
14:32:07 [main] DEBUG org.hibernate.SQL - select usuario0_.id_usuario as id_usuari1_9_0_, usuario0_.deleted as deleted2_9_0_, usuario0_.password as password3_9_0_, usuario0_.username as userns
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
14:32:07 [main] DEBUG org.hibernate.SQL - select contacto0_.id_contacto as id_conta1_2_0_, contacto0_.celular as celular2_2_0_, contacto0_.deleted as deleted3_2_0_, contacto0_.email as email
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [2]
14:32:07 [main] DEBUG org.hibernate.SQL - select domicilio0_.id_domicilio as id_domic1_4_0_, domicilio0_.calle as calle2_4_0_, domicilio0_.cp as cp3_4_0_, domicilio0_.deleted as deleted4_4_0_
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [2]
14:32:07 [main] DEBUG org.hibernate.SQL - select contacto0_.id_contacto as id_conta1_2_0_, contacto0_.celular as celular2_2_0_, contacto0_.deleted as deleted3_2_0_, contacto0_.email as email
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [9]
14:32:07 [main] DEBUG org.hibernate.SQL - select domicilio0_.id_domicilio as id_domic1_4_0_, domicilio0_.calle as calle2_4_0_, domicilio0_.cp as cp3_4_0_, domicilio0_.deleted as deleted4_4_0_
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [9]
14:32:07 [main] DEBUG org.hibernate.SQL - select contacto0_.id_contacto as id_conta1_2_0_, contacto0_.celular as celular2_2_0_, contacto0_.deleted as deleted3_2_0_, contacto0_.email as email
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [10]
14:32:07 [main] DEBUG org.hibernate.SQL - select domicilio0_.id_domicilio as id_domic1_4_0_, domicilio0_.calle as calle2_4_0_, domicilio0_.cp as cp3_4_0_, domicilio0_.deleted as deleted4_4_0_
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [10]
Alumno{idAlumno=1, nombre=Alfonso, apellidoPaterno=Ugalde, apellidoMaterno=Martinez, genero=M, fechaNacimiento=null, version=0, deleted=0, contacto=Contacto{idContacto=1, telefono=55551111, c
Alumno{idAlumno=2, nombre=Martha, apellidoPaterno=Martinez, apellidoMaterno=Garcia, genero=null, fechaNacimiento=null, version=0, deleted=0, contacto=Contacto{idContacto=2, telefono=55717189, c
Alumno{idAlumno=3, nombre=Carlos, apellidoPaterno=Morales, apellidoMaterno=Valdez, genero=null, fechaNacimiento=null, version=0, deleted=0, contacto=Contacto{idContacto=3, telefono=4441111, c
```

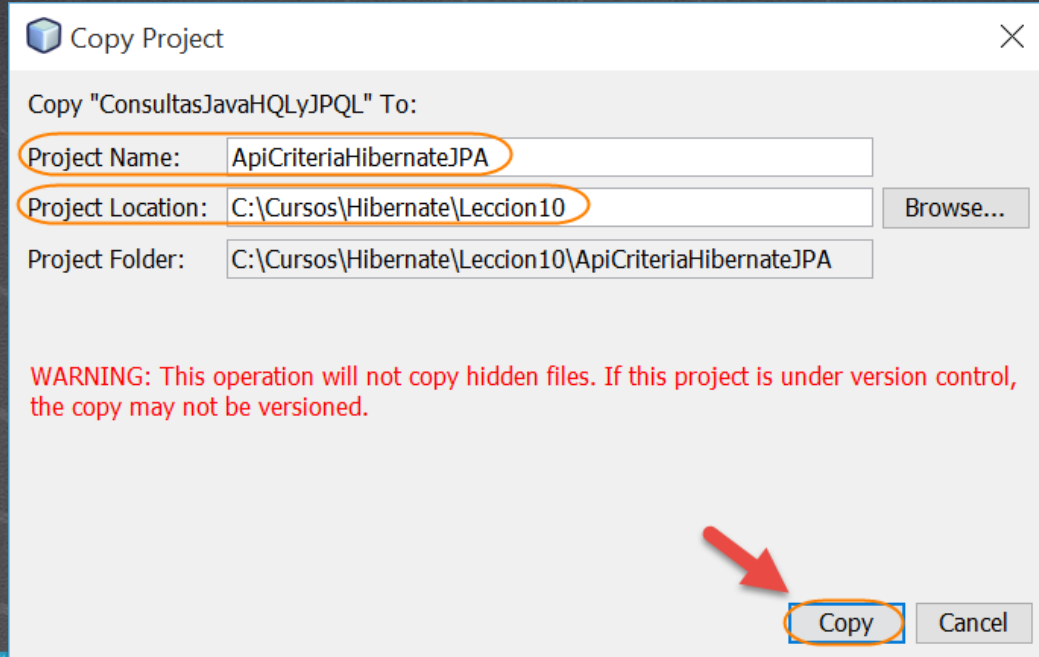

PASO 1. CREACIÓN DEL PROYECTO

Copiamos el proyecto partiendo de ConsultasJavaHQLyJPQL:



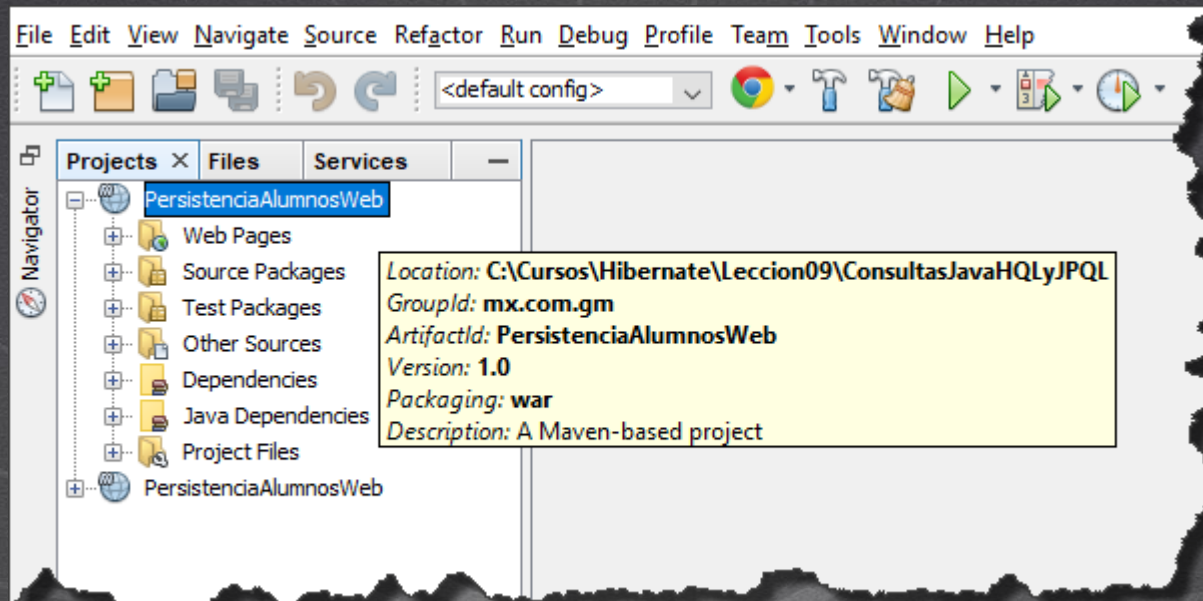
PASO 1. CREACIÓN DEL PROYECTO

Creamos el proyecto ApiCriteriaHibernateJPA:



PASO 2. CERRAMOS EL PROYECTO

Cerramos el proyecto que ya no utilizamos:

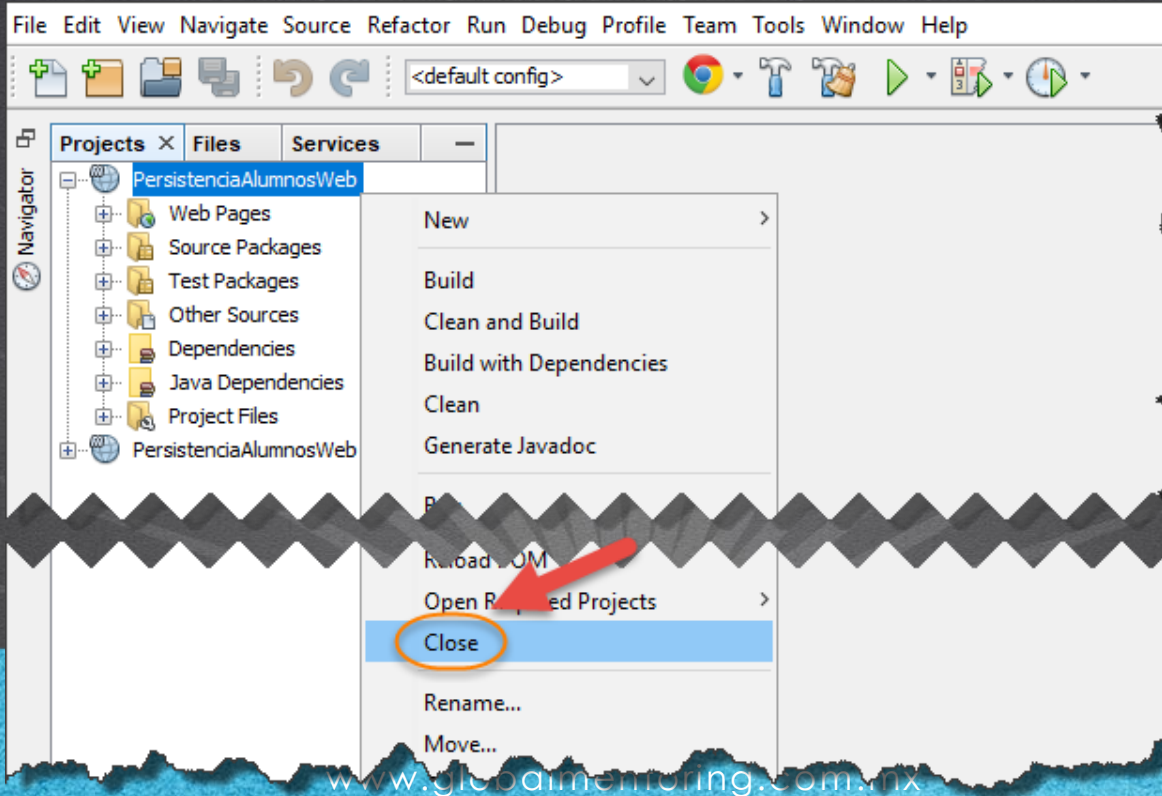


CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

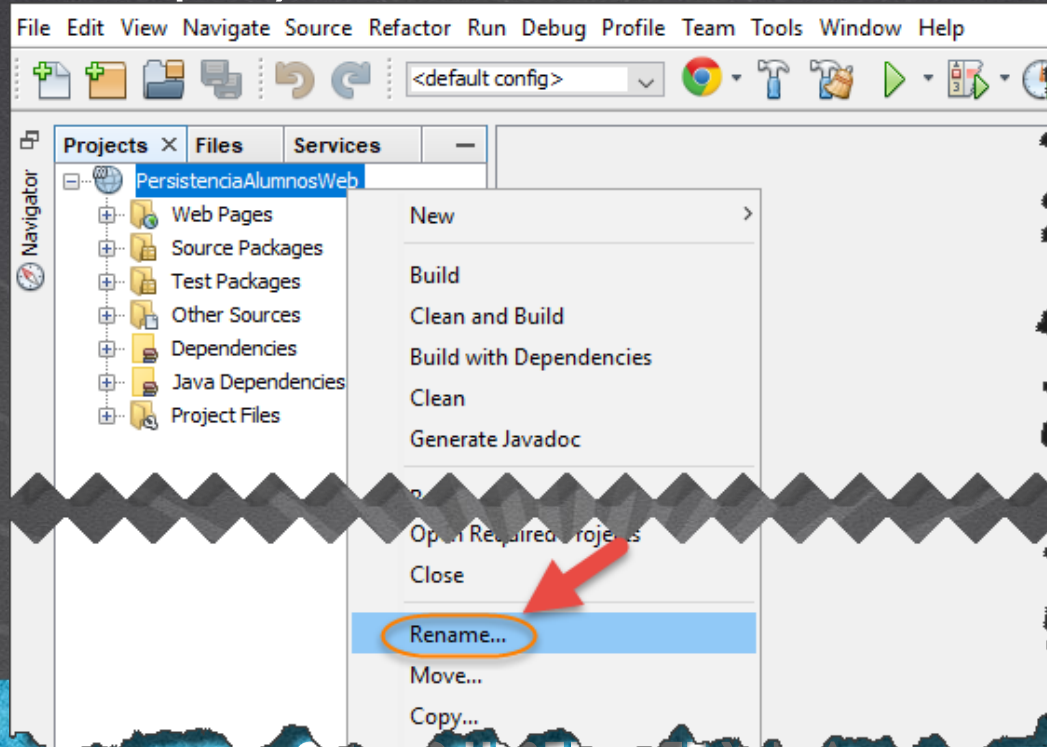
PASO 2. CERRAMOS EL PROYECTO

Cerramos el proyecto que ya no utilizamos:



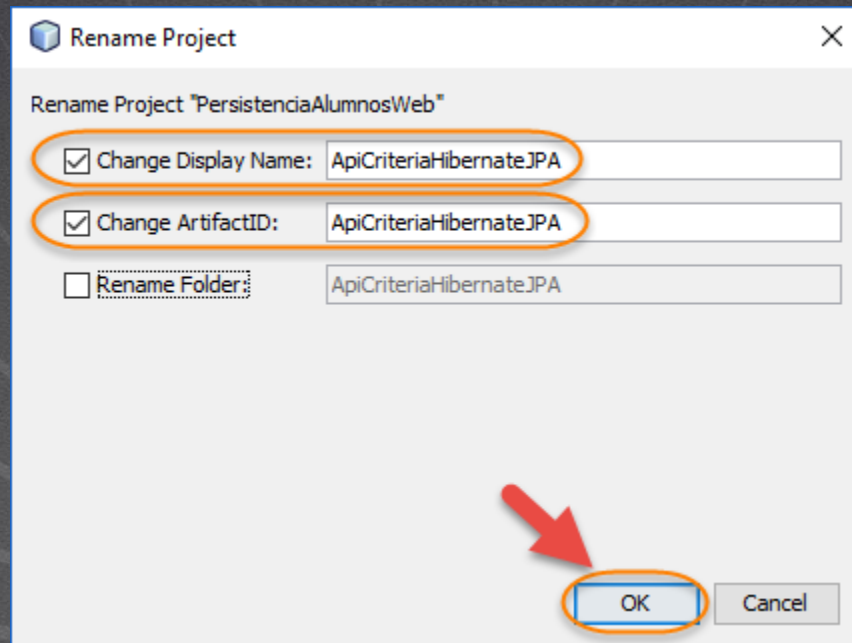
PASO 3. RENOMBRAMOS EL PROYECTO

Renombramos el proyecto:



PASO 3. RENOMBAMOS EL PROYECTO

Renombramos el proyecto:

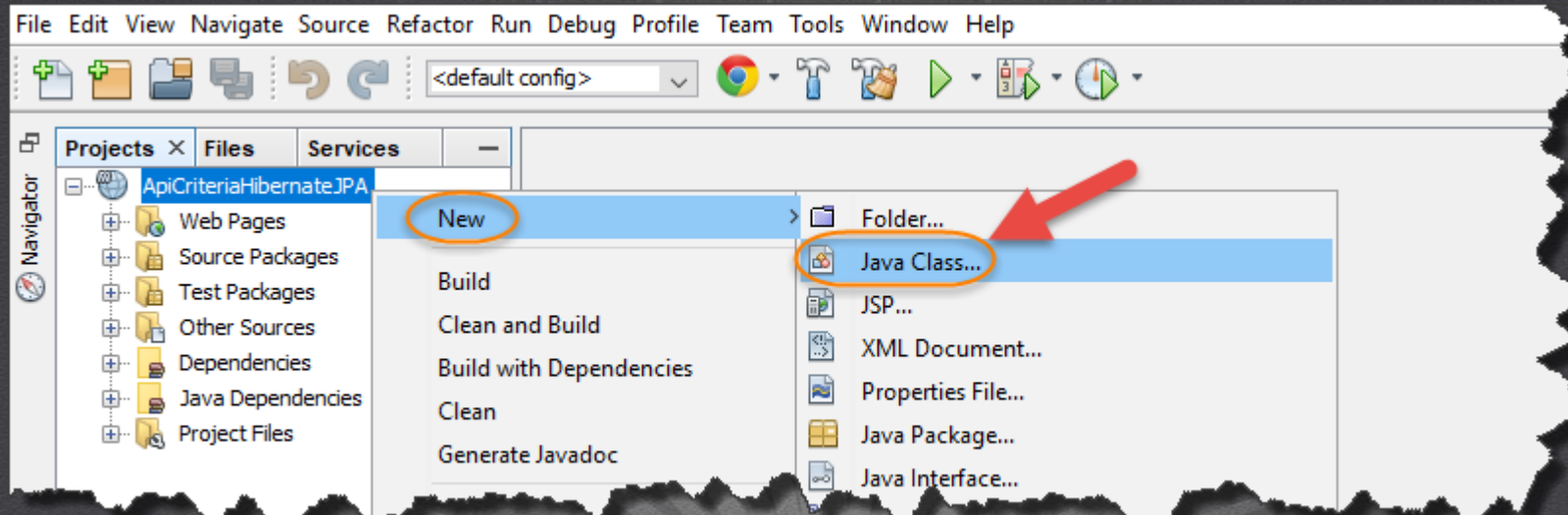


CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 4. CREAR UNA CLASE

Creamos la clase TestApiCriteriaHibernateJPA.java:



PASO 4. CREAR UNA CLASE

Creamos la clase TestApiCriteriaHibernateJPA.java :

New Java Class

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

PASO 5. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

```
package test;

import java.util.Arrays;
import java.util.Collection;
import java.util.List;
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;
import javax.persistence.TypedQuery;
import javax.persistence.criteria.CriteriaBuilder;
import javax.persistence.criteria.CriteriaQuery;
import javax.persistence.criteria.Expression;
import javax.persistence.criteria.ParameterExpression;
import javax.persistence.criteria.Predicate;
import javax.persistence.criteria.Root;
import model.Alumno;

public class TestApiCriteriaHibernateJPA {

    public static void main(String[] args) {
        /*Utilizamos la Unidad de Persistencia de JPA*/
        EntityManagerFactory fabrica = Persistence.createEntityManagerFactory("HibernateJpaPU");
        EntityManager em = fabrica.createEntityManager();

        //Variables de ayuda
        CriteriaBuilder cb = em.getCriteriaBuilder();
        List<Alumno> alumnos = null;
        Alumno alumno = null;
```

PASO 3. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

```
// Query 1
// Consulta de todos los Alumnos
System.out.println("\nQuery 1");
//Se crea el objeto criteria query
CriteriaQuery<Alumno> q1 = cb.createQuery(Alumno.class);
//Establece el root del query
q1.from(Alumno.class);
//Se ejecuta el query
alumnos = em.createQuery(q1).getResultList();
//Imprimimos los alumnos
imprimirAlumnos(alumnos);
```

```
//1.JPQL equivalente
from Alumno
```


PASO 3. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

```
// Query2
// Consulta del Alumno con id = 7
System.out.println("\nQuery 2");
CriteriaQuery<Alumno> q2 = cb.createQuery(Alumno.class);
Root<Alumno> c2 = q2.from(Alumno.class);
ParameterExpression<Integer> pId = cb.parameter(Integer.class);
q2.select(c2).where(cb.equal(c2.get("idAlumno"), pId));
//Ejecutamos el query
TypedQuery<Alumno> query = em.createQuery(q2);
//Establecemos el valor del parámetro
query.setParameter(pId, 7);
alumno = query.getSingleResult();
System.out.println(alumno);
```

```
//2.JPQL equivalente
from Alumno a where a.idAlumno = 7
```

PASO 3. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

```
// Query 3
// Consulta del alumno con nombre
System.out.println("\nQuery 3");
CriteriaQuery<Alumno> q3 = cb.createQuery(Alumno.class);
Root<Alumno> c3 = q3.from(Alumno.class);
ParameterExpression<String> pNombre = cb.parameter(String.class);
q3.select(c3).where(cb.equal(c3.get("nombre"), pNombre));
//Ejecutamos el query
TypedQuery<Alumno> query3 = em.createQuery(q3);
//Establecemos el valor del parámetro
query3.setParameter(pNombre, "Martha");
alumnos = query3.getResultList();
imprimirAlumnos(alumnos);
```

```
//3.JPQL equivalente
from Alumno a where a.nombre = 'Martha'
```


PASO 3. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

```
// Query 4
// Consulta alumnos restringiendo por el idAlumno
System.out.println("\nQuery 4");
CriteriaQuery<Alumno> qb4 = cb.createQuery(Alumno.class);
Root<Alumno> c4 = qb4.from(Alumno.class);
qb4.where(c4.get("idAlumno").in(cb.parameter(Collection.class)));

TypedQuery<Alumno> q4 = em.createQuery(qb4);
Integer[] idAlumnos = {7,8}; //colocar id validos
for (ParameterExpression parameter : qb4.getParameters()) {
    q4.setParameter(parameter, Arrays.asList(idAlumnos));
}
alumnos = q4.getResultList();
imprimirAlumnos(alumnos);
```

```
//4. JPQL equivalente
from Alumno a where a.idAlumno in (7,8)
```

PASO 3. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

```
// Query 5
// Obtiene los alumnos cuyo apeMaterno es no nulo
System.out.println("\nQuery 5");
CriteriaQuery<Alumno> qb5 = cb.createQuery(Alumno.class);
Root<Alumno> c5 = qb5.from(Alumno.class);
qb5.select(c5).where(cb.isNotNull(c5.get("apellidoMaterno")));
//Ejecutamos el query
TypedQuery<Alumno> q5 = em.createQuery(qb5);
alumnos = q5.getResultList();
imprimirAlumnos(alumnos);
```

```
//5. JPQL equivalente
from Alumno a where a.apellidoMaterno is not null
```

PASO 3. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

```
//Query 6
System.out.println("\nQuery 6");
//Obtiene los alumnos cuyo nombre comience con una "m"
CriteriaQuery<Alumno> qb6 = cb.createQuery(Alumno.class);
Root<Alumno> c6 = qb6.from(Alumno.class);

Expression<String> path = c6.get("nombre");
Expression<String> mayusculas = cb.upper(path);
String cadenaBuscar = "" + "m".toUpperCase() + "%";

Predicate predicado = cb.like(mayusculas, cadenaBuscar);
qb6.where(cb.and(predicado));

alumnos = em.createQuery(qb6.select(c6)).getResultList();
imprimirAlumnos(alumnos);
```

```
//6. JPQL equivalente
from Alumno a where a.nombre like 'm%'
```


PASO 3. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

```
//Query 7
System.out.println("\nQuery 7");
//Mismo query que el 6, pero en JPA no existe MatchMode.START por lo que el query queda igual

//Query 8
//Obtiene los alumnos cuyo nombre contenga "s" con ignoreCase
System.out.println("\nQuery 8");
CriteriaQuery<Alumno> qb8 = cb.createQuery(Alumno.class);
Root<Alumno> c8 = qb8.from(Alumno.class);
String cadenaBuscar2 = "%" + "s".toUpperCase() + "%";
qb8.where(cb.like(cb.upper(c8.get("nombre")), cadenaBuscar2));
alumnos = em.createQuery(qb8).getResultList();
imprimirAlumnos(alumnos);
```

```
//8. JPQL equivalente
from Alumno a where upper(a.nombre) = upper('juan')
```

PASO 3. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

```
//Query 9
//Obtiene los alumnos agregando varias restricciones
//se agregan con 'and' por default
System.out.println("\nQuery 9");
CriteriaQuery<Alumno> qb9 = cb.createQuery(Alumno.class);
Root<Alumno> c9 = qb9.from(Alumno.class);
//Creamos las restricciones
Predicate[] restrictions = new Predicate[]{
    cb.equal(c9.get("nombre"), "Carlos"),
    cb.isNull(c9.get("apellidoMaterno"))
};
//Agregamos las restricciones
qb9.where(cb.and(restrictions));
//Ejecutamos el query
alumnos = em.createQuery(qb9).getResultList();
imprimirAlumnos(alumnos);
```

```
//9. JPQL equivalente
from Alumno a where a.nombre = 'Carlos' and apellidoMaterno is not null
```

PASO 3. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

```
//Query 10
//Obtiene los alumnos cuyo nombre empieza con M
//o el apellidoMaterno es no nulo (usando 'or')
System.out.println("\nQuery 10");
CriteriaQuery<Alumno> qb10 = cb.createQuery(Alumno.class);
Root<Alumno> c10 = qb10.from(Alumno.class);
//Creamos las restricciones
String cadenaBuscar3 = "" + "M".toUpperCase() + "%";
Predicate[] restrictions2 = new Predicate[]{
    cb.like(c10.get("nombre"), cadenaBuscar3),
    cb.isNull(c10.get("apellidoMaterno"))
};
//Agregamos las restricciones
qb10.where(cb.or(restrictions2));
//Ejecutamos el query
alumnos = em.createQuery(qb10).getResultList();
imprimirAlumnos(alumnos);
```

```
//10. JPQL equivalente
from Alumno a where a.nombre like 'M%' or apellidoMaterno is not null
```


PASO 3. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

```
//Query 11
//Obtiene los alumnos cuyo apeMaterno es no nulo
//agregando ordenamiento por nombre asc y apellidoPaterno desc
System.out.println("\nQuery 11");
CriteriaQuery<Alumno> qb11 = cb.createQuery(Alumno.class);
Root<Alumno> c11 = qb11.from(Alumno.class);
//Creamos las restricciones
Predicate[] restrictions3 = new Predicate[]{
    cb.equal(c11.get("nombre"), "Carlos"),
    cb.isNull(c11.get("apellidoMaterno"))
};
//Agregamos las restricciones
qb11.where(cb.or(restrictions3));
//Agregamos ordenamiento
qb11.orderBy(cb.asc(c11.get("nombre")), cb.desc(c11.get("apellidoPaterno")));
//Ejecutamos el query
alumnos = em.createQuery(qb11).getResultList();
imprimirAlumnos(alumnos);
```

```
//11. JPQL equivalente
from Alumno a
where a.apellidoPaterno is not null
order by a.nombre asc, a.apellidoPaterno desc
```

PASO 3. MODIFICAMOS EL CÓDIGO

Archivo TestApiCriteriaHibernateJPA.java:

Clic para descargar
el código

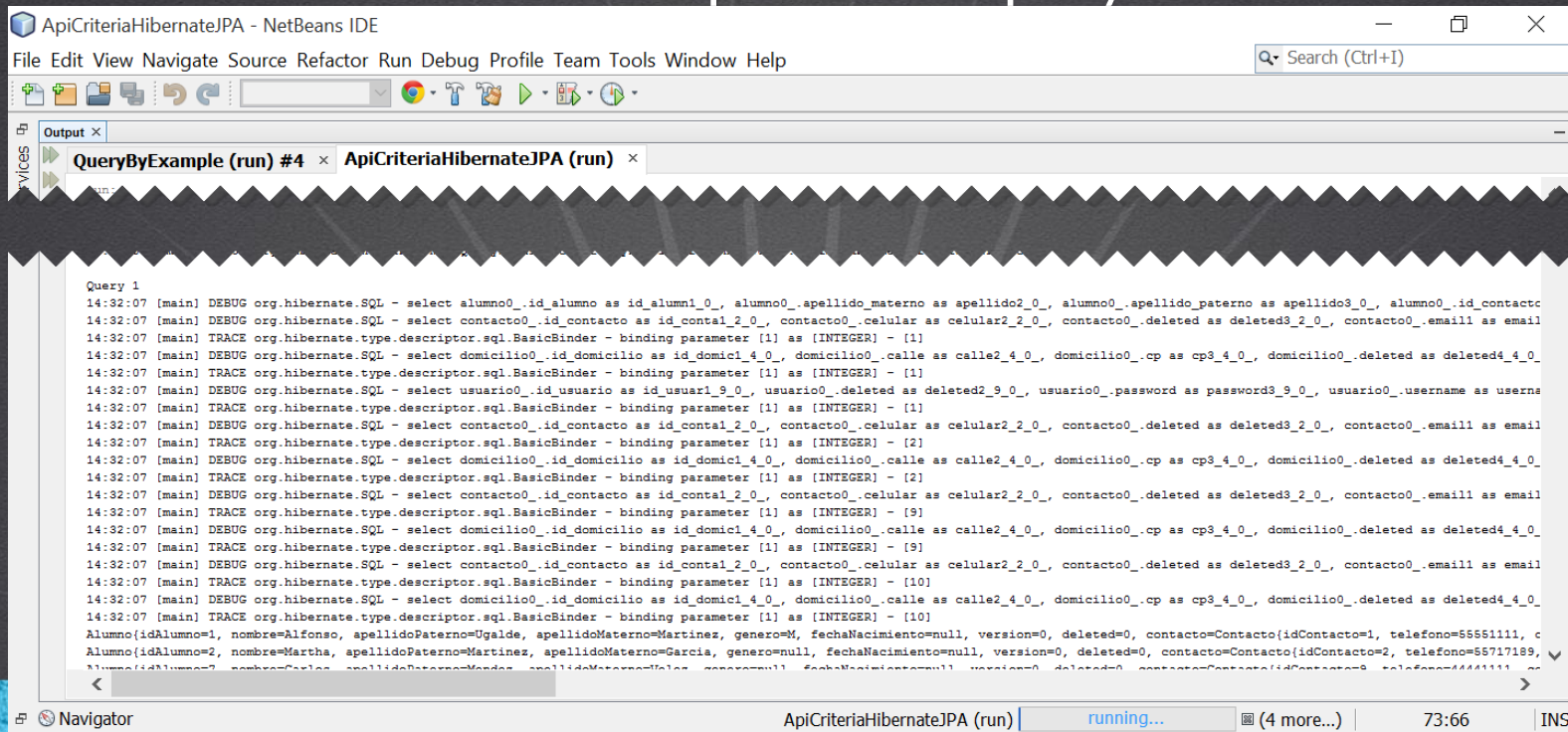
```
private static void imprimirAlumnos(List<Alumno> alumnos) {  
    for (Alumno a : alumnos) {  
        System.out.println(a);  
    }  
}
```

CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 4. EJECUTAMOS EL PROYECTO

Ejecutamos cada una de las queries del proyecto:



The screenshot shows the NetBeans IDE interface for the project 'ApiCriteriaHibernateJPA'. The 'Output' window is open, displaying the execution of 'Query 1'. The log shows a series of SQL queries and Hibernate messages. The final result is a list of three students with their contact information.

```
Query 1
14:32:07 [main] DEBUG org.hibernate.SQL - select alumno0_.id_alumno as id_alumn1_0_, alumno0_.apellido_materno as apellido2_0_, alumno0_.apellido_paterno as apellido3_0_, alumno0_.id_contacto
14:32:07 [main] DEBUG org.hibernate.SQL - select contacto0_.id_contacto as id_contal_2_0_, contacto0_.celular as celular2_2_0_, contacto0_.deleted as deleted3_2_0_, contacto0_.email as email
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
14:32:07 [main] DEBUG org.hibernate.SQL - select domicilio0_.id_domicilio as id_domici_4_0_, domicilio0_.calle as calle2_4_0_, domicilio0_.cp as cp3_4_0_, domicilio0_.deleted as deleted4_4_0_
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
14:32:07 [main] DEBUG org.hibernate.SQL - select usuario0_.id_usuario as id_usuari_9_0_, usuario0_.deleted as deleted2_9_0_, usuario0_.password as password3_9_0_, usuario0_.username as usern
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
14:32:07 [main] DEBUG org.hibernate.SQL - select contacto0_.id_contacto as id_contal_2_0_, contacto0_.celular as celular2_2_0_, contacto0_.deleted as deleted3_2_0_, contacto0_.email as email
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [2]
14:32:07 [main] DEBUG org.hibernate.SQL - select domicilio0_.id_domicilio as id_domici_4_0_, domicilio0_.calle as calle2_4_0_, domicilio0_.cp as cp3_4_0_, domicilio0_.deleted as deleted4_4_0_
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [2]
14:32:07 [main] DEBUG org.hibernate.SQL - select contacto0_.id_contacto as id_contal_2_0_, contacto0_.celular as celular2_2_0_, contacto0_.deleted as deleted3_2_0_, contacto0_.email as email
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [9]
14:32:07 [main] DEBUG org.hibernate.SQL - select domicilio0_.id_domicilio as id_domici_4_0_, domicilio0_.calle as calle2_4_0_, domicilio0_.cp as cp3_4_0_, domicilio0_.deleted as deleted4_4_0_
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [9]
14:32:07 [main] DEBUG org.hibernate.SQL - select contacto0_.id_contacto as id_contal_2_0_, contacto0_.celular as celular2_2_0_, contacto0_.deleted as deleted3_2_0_, contacto0_.email as email
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [10]
14:32:07 [main] DEBUG org.hibernate.SQL - select domicilio0_.id_domicilio as id_domici_4_0_, domicilio0_.calle as calle2_4_0_, domicilio0_.cp as cp3_4_0_, domicilio0_.deleted as deleted4_4_0_
14:32:07 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [10]
Alumno{idAlumno=1, nombre=Alfonso, apellidoPaterno=Ugalde, apellidoMaterno=Martinez, genero=M, fechaNacimiento=null, version=0, deleted=0, contacto=Contacto{idContacto=1, telefono=55551111, c
Alumno{idAlumno=2, nombre=Martha, apellidoPaterno=Martinez, apellidoMaterno=Garcia, genero=null, fechaNacimiento=null, version=0, deleted=0, contacto=Contacto{idContacto=2, telefono=55717189, c
Alumno{idAlumno=3, nombre=Carlos, apellidoPaterno=Morales, apellidoMaterno=Valdez, genero=null, fechaNacimiento=null, version=0, deleted=0, contacto=Contacto{idContacto=3, telefono=4441111, c
```


EJERCICIOS EXTRA

- Ejecutar cada una de las consultas y compararlas con su equivalente de HQL/JPQL para empezar a detectar cuando es más conveniente utilizar una u otra forma de creación de consultas, ya sea con HQL/JPQL o con el API de Criteria de Hibernate/JPA.



Experiencia y Conocimiento para tu vida

CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

CONCLUSIÓN DEL EJERCICIO

- Con este ejercicio hemos ejecutado varias de las consultas con el API de Criteria de Hibernate/JPA.
- Con esto ya podemos comparar y decidir si utilizamos el lenguaje de HQL/JPQL o el API de Criteria.
- Cada uno tiene sus ventajas y desventajas, pero todo dependerá de lo que necesitemos en nuestra aplicación para saber si utilizamos otra solución.
- En general utilizaremos HQL/JPQL cuando las consultas sea más estáticas y utilizaremos el API de Criteria cuando las consultas tiendan a ser más dinámicas, de esta manera evitaremos menos concatenación de cadenas para crear nuestras consultas.

CURSO ONLINE

HIBERNATE & JPA

Por: Ing. Ubaldo Acosta



Experiencia y Conocimiento para tu vida

CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx