

# HIBERNATE & JPA COURSE

## EXERCISE

### CRITERIA API LAZY & EAGER LOADING



HIBERNATE & JPA COURSE

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)

# EXERCISE OBJECTIVE

Use the Criteria API to implement the concept of Lazy and Eager Loading. At the end we should observe the following:

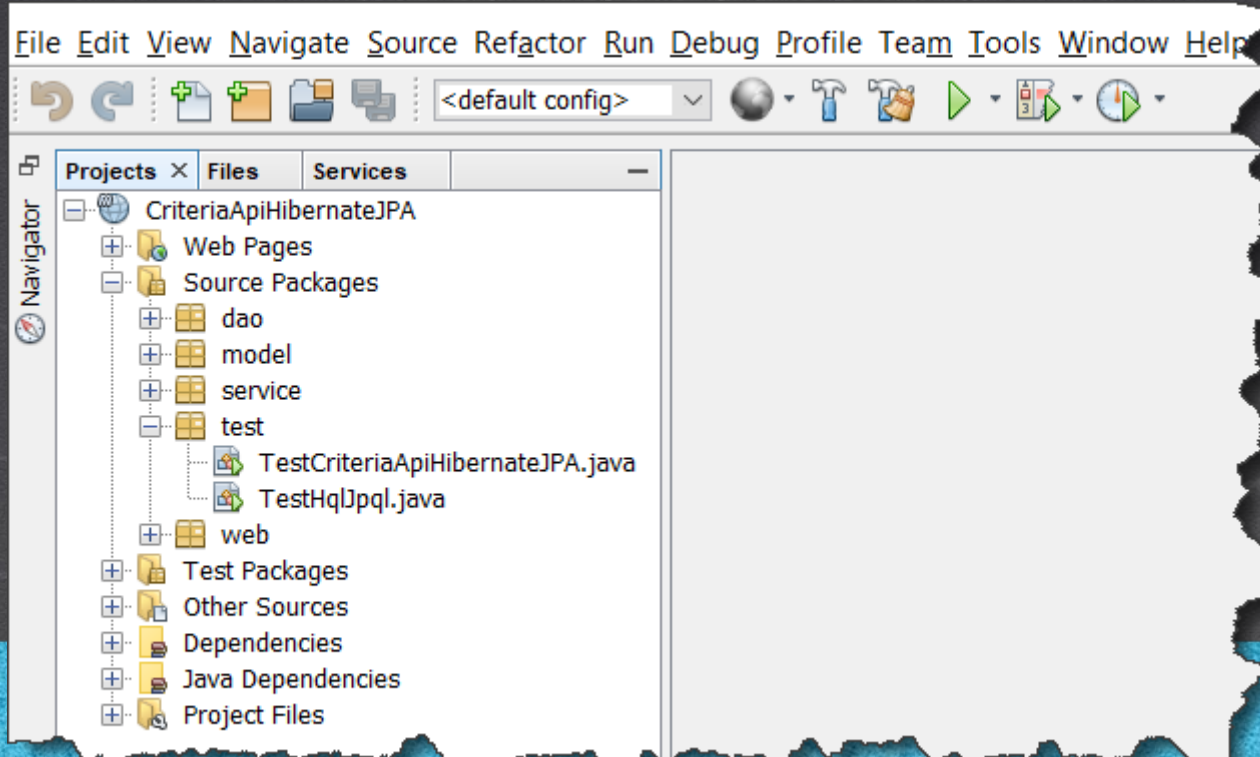
```
Output - Run (TestCriteriaApiLazyEagerLoading) X
13:08:46 [main] INFO org.hibernate.hql.internal.QueryTranslatorFactoryInitiator - HHH000397: Using ASTQueryTranslatorFactory

Query 1
13:08:46 [main] DEBUG org.hibernate.SQL - select student0_.id_student as id_studel_3_, student0_.id_address as id_addre5_3_, student0_.deleted as deleted2_3_, student0_.r
Hibernate: select student0_.id_student as id_studel_3_, student0_.id_address as id_addre5_3_, student0_.deleted as deleted2_3_, student0_.name as name3_3_, student0_.id_u
13:08:46 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.
Hibernate: select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_,
13:08:46 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
13:08:46 [main] DEBUG org.hibernate.SQL - select user0_.id_user as id_user1_4_0_, user0_.deleted as deleted2_4_0_, user0_.password as password3_4_0_, user0_.username as u
Hibernate: select user0_.id_user as id_user1_4_0_, user0_.deleted as deleted2_4_0_, user0_.password as password3_4_0_, user0_.username as username4_4_0_, user0_.version a
13:08:46 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
13:08:46 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.
Hibernate: select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_,
13:08:46 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [3]
13:08:46 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.
Hibernate: select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_,
13:08:46 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [5]
13:08:46 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.
Hibernate: select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_,
13:08:46 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [6]
```



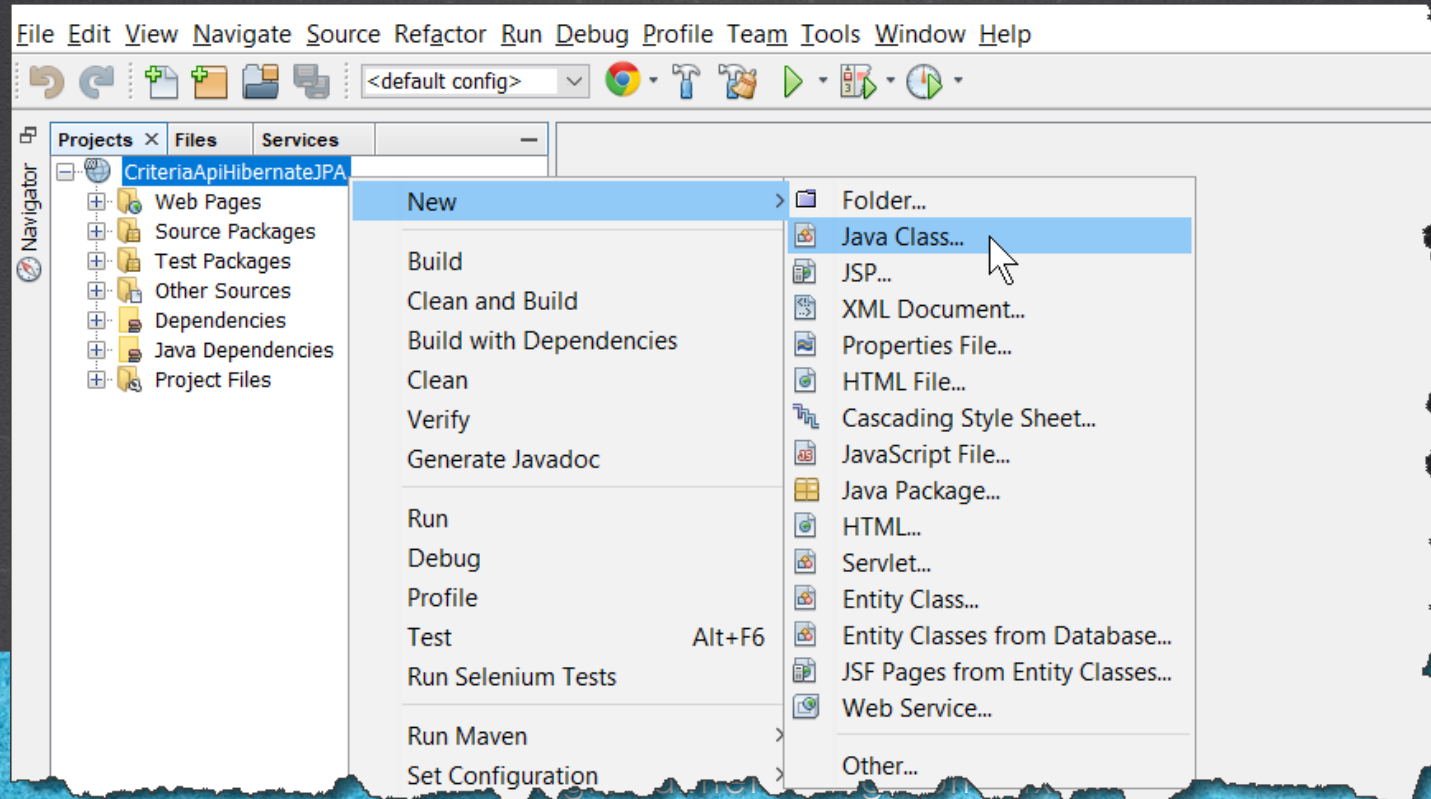
# 1. USE THE PROJECT

We open the CriteriaApiHibernateJPA project:



## 2. CREATE A CLASS

We create the class TestCriteriaApiLazyEagerLoading.java:



## 2. CREATE A CLASS

We create the class TestCriteriaApiLazyEagerLoading.java:

New Java Class

**Steps**

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

**Name and Location**

Class Name: TestCriteriaApiLazyEagerLoading

Project: CriteriaApiHibernateJPA

Location: Source Packages

Package: test

Created File: C:\Courses\Hibernate\Lesson10\CriteriaApiHibernateJPA\src\main\java\test\TestCriteriaApiLazyEagerLoading.java

< Back Next > **Finish** Cancel Help

# 3. MODIFY THE CODE

## TestCriteriaApiLazyEagerLoading.java:

Click to download

```
package test;

import java.util.*;
import javax.persistence.*;
import javax.persistence.criteria.*;
import model.*;

public class TestCriteriaApiLazyEagerLoading {

    public static void main(String[] args) {
        EntityManagerFactory fabrica = Persistence.createEntityManagerFactory("HibernateJpaPU");
        EntityManager em = fabrica.createEntityManager();

        //Help variables
        CriteriaBuilder cb = em.getCriteriaBuilder();
        List<Student> students = null;

        //By default the queries are lazy type, that is, they do not
        //recover the Left Join relations with the Criteria API
        //Query 1
        System.out.println("\nQuery 1");
        CriteriaQuery<Student> qbl = cb.createQuery(Student.class);
        Root<Student> c1 = qbl.from(Student.class);
        c1.join("address", JoinType.LEFT);

        students = em.createQuery(qbl).getResultList();
        printStudents(students);
    }
}
```



# 3. MODIFY THE CODE

## TestCriteriaApiLazyEagerLoading.java:

Click to download

```
//Query 2
System.out.println("\nQuery 2");
//We define the query
CriteriaQuery<Student> qb2 = cb.createQuery(Student.class);
//We define the query root
Root<Student> c2 = qb2.from(Student.class);
//We specify the join
Join<Student, Address> add = c2.join("address");
//Optionally add the restriction using the join
qb2.where(cb.equal(add.<Integer>get("idAddress"), 1));
//We define an Entity Graph to specify the Fetch join
EntityGraph<Student> fetchGraph = em.createEntityGraph(Student.class);
//We specify the relationship to be raised in an eager way (anticipated)
fetchGraph.addSubgraph("address");
//loadgraph adds the definition plus what is already specified in the
//entity class. fetchgraph ignores what is defined in the entity class
//and adds only the new
em.createQuery(qb2).setHint("javax.persistence.loadgraph", fetchGraph);

TypedQuery<Student> q2 = em.createQuery(qb2);
students = q2.getResultList();
printStudents(students);
```

**HIBERNATE & JPA COURSE**

www.globalmentoring.com.mx

# 3. MODIFY THE CODE

## TestCriteriaApiLazyEagerLoading.java:

Click to download

```
// Query 3
System.out.println("\nQuery 3");
CriteriaQuery<Student> qb3 = cb.createQuery(Student.class);
Root<Student> c3 = qb3.from(Student.class);
Join<Student, Address> address = c3.join("address");
List<Predicate> conditions = new ArrayList();
Integer idStudent = 1;
conditions.add(cb.equal(c3.get("idStudent"), idStudent));
conditions.add(cb.isNull(address.get("streetNumber")));

TypedQuery<Student> q3 = em.createQuery(
    qb3
    .select(c3)
    .where(conditions.toArray(new Predicate[]{}))
    .orderBy(cb.asc(c3.get("idStudent")))
    .distinct(true)
);

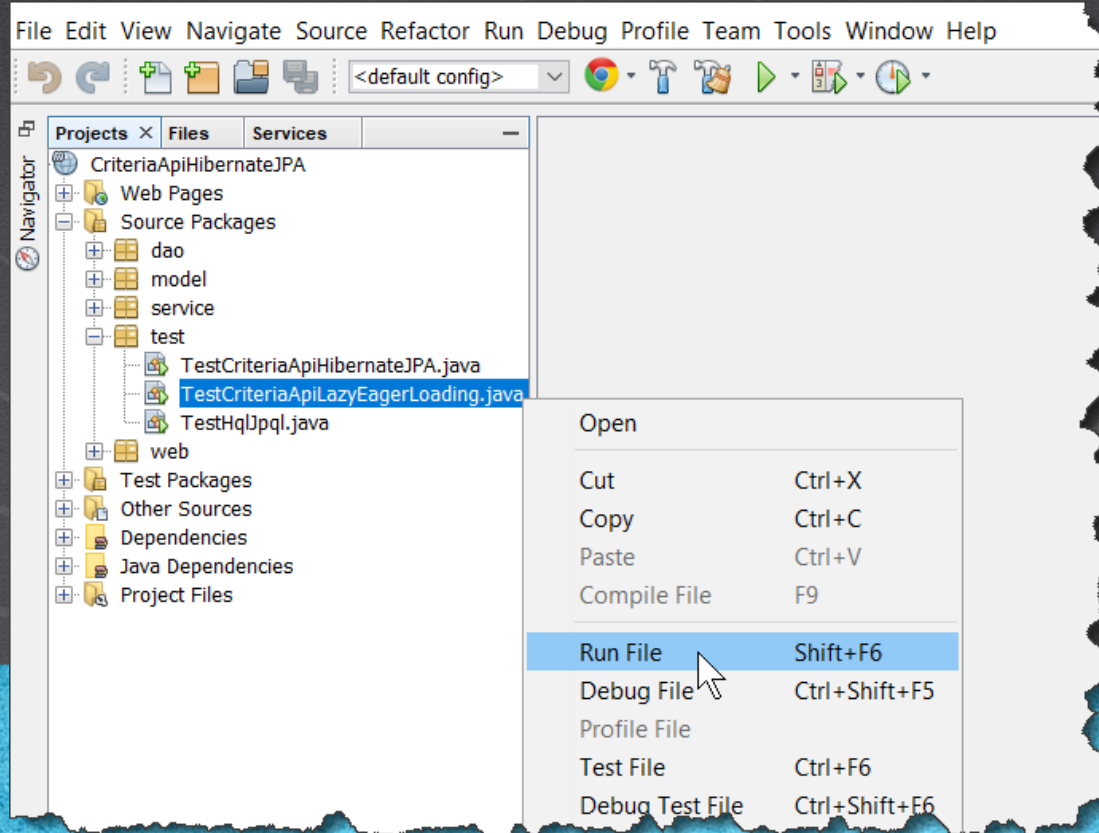
printStudents(q3.getResultList());
}

private static void printStudents(List<Student> students) {
    for (Student a : students) {
        System.out.println(a);
    }
}
}
```



# 4.EXECUTE THE PROJECT

We execute each of the queries of the project:



# 4. EXECUTE THE PROJECT

We execute each of the queries of the project:

```
Output - Run (TestCriteriaApiLazyEagerLoading) X
13:08:46 [main] INFO org.hibernate.hql.internal.QueryTranslatorFactoryInitiator - HHH0000397: Using ASTQueryTranslatorFactory

Query 1
13:08:46 [main] DEBUG org.hibernate.SQL - select student0_.id_student as id_studel_3_, student0_.id_address as id_addre5_3_, student0_.deleted as deleted2_3_, student0_.name as name3_3_, student0_.id_user as id_user1_4_0_, student0_.password as password3_4_0_, student0_.username as username4_4_0_, student0_.version as version4_4_0_ from student0_ where student0_.deleted=0
Hibernate: select student0_.id_student as id_studel_3_, student0_.id_address as id_addre5_3_, student0_.deleted as deleted2_3_, student0_.name as name3_3_, student0_.id_user as id_user1_4_0_, student0_.password as password3_4_0_, student0_.username as username4_4_0_, student0_.version as version4_4_0_ from student0_ where student0_.deleted=0
13:08:46 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_ from address0_ where address0_.deleted=0
Hibernate: select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_ from address0_ where address0_.deleted=0
13:08:46 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
13:08:46 [main] DEBUG org.hibernate.SQL - select user0_.id_user as id_user1_4_0_, user0_.deleted as deleted2_4_0_, user0_.password as password3_4_0_, user0_.username as username4_4_0_, user0_.version as version4_4_0_ from user0_ where user0_.deleted=0
Hibernate: select user0_.id_user as id_user1_4_0_, user0_.deleted as deleted2_4_0_, user0_.password as password3_4_0_, user0_.username as username4_4_0_, user0_.version as version4_4_0_ from user0_ where user0_.deleted=0
13:08:46 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
13:08:46 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_ from address0_ where address0_.deleted=0
Hibernate: select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_ from address0_ where address0_.deleted=0
13:08:46 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [3]
13:08:46 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_ from address0_ where address0_.deleted=0
Hibernate: select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_ from address0_ where address0_.deleted=0
13:08:46 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [5]
13:08:46 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_ from address0_ where address0_.deleted=0
Hibernate: select address0_.id_address as id_addrel_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.street_name as street_n4_0_0_ from address0_ where address0_.deleted=0
13:08:46 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [6]
```



# EXERCISE CONCLUSION

- With this exercise we have executed several of the queries with the Criteria API of Hibernate / JPA applying the concept of Fetch or Eager Loading.
- By default the queries are Lazy type in Hibernate / JPA, however we have seen how to load the relationships in advance (eager) by applying the concept of Fetch to the queries.
- With this we can already compare and decide if we use the HQL / JPQL language or the Criteria API.
- Each one has its advantages and disadvantages, but everything will depend on what we need in our application to know if we use another solution.



**ONLINE COURSE**

# **HIBERNATE & JPA**

By: Eng. Ubaldo Acosta



**HIBERNATE & JPA COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)