

# **CURSO DE HIBERNATE Y JPA**

## **EJERCICIO**

### **CICLO DE VIDA EN HIBERNATE/JPA**



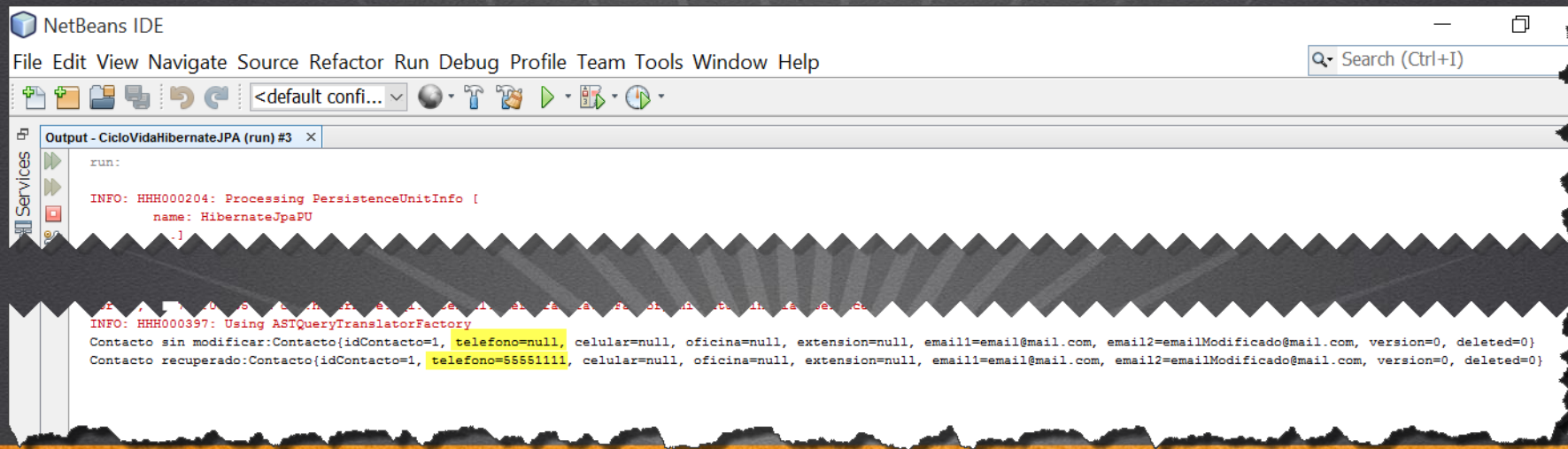
Experiencia y Conocimiento para tu vida

**CURSO DE HIBERNATE Y JPA**

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

# OBJETIVO DEL EJERCICIO

Poner en práctica el Ciclo de Vida de los objetos de entidad en Hibernate/JPA. Al finalizar veremos:



```
NetBeans IDE
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)

Output - CicloVidaHibernateJPA (run) #3 x
run:
INFO: HHH000204: Processing PersistenceUnitInfo [
    name: HibernateJpaPU
    .]
INFO: HHH000397: Using ASTQueryTranslatorFactory
Contacto sin modificar:Contacto{idContacto=1, telefono=null, celular=null, oficina=null, extension=null, email1=email@mail.com, email2=emailModificado@mail.com, version=0, deleted=0}
Contacto recuperado:Contacto{idContacto=1, telefono=55551111, celular=null, oficina=null, extension=null, email1=email@mail.com, email2=emailModificado@mail.com, version=0, deleted=0}
```

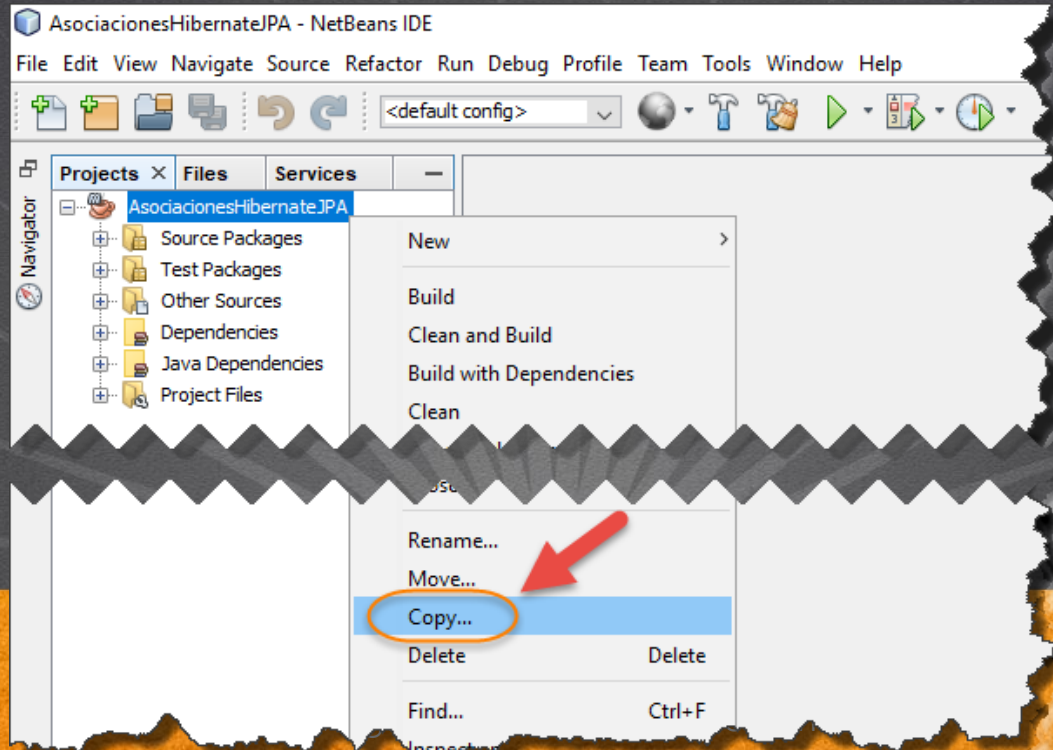
**CURSO DE HIBERNATE Y JPA**

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



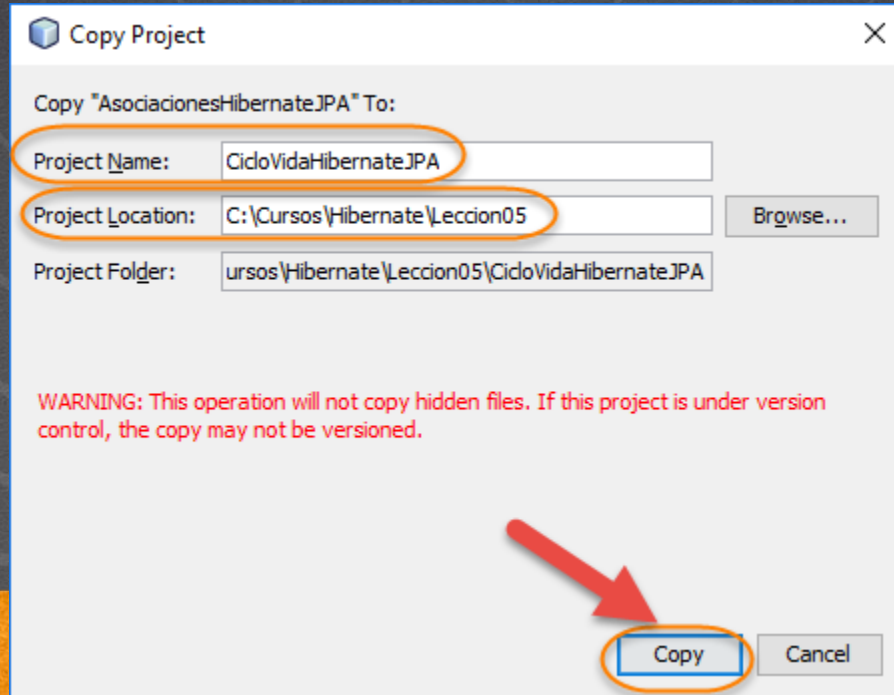
# PASO 1. CREAMOS EL PROYECTO

Copiamos y pegamos el proyecto anterior  
AsociacionesHibernateJPA:



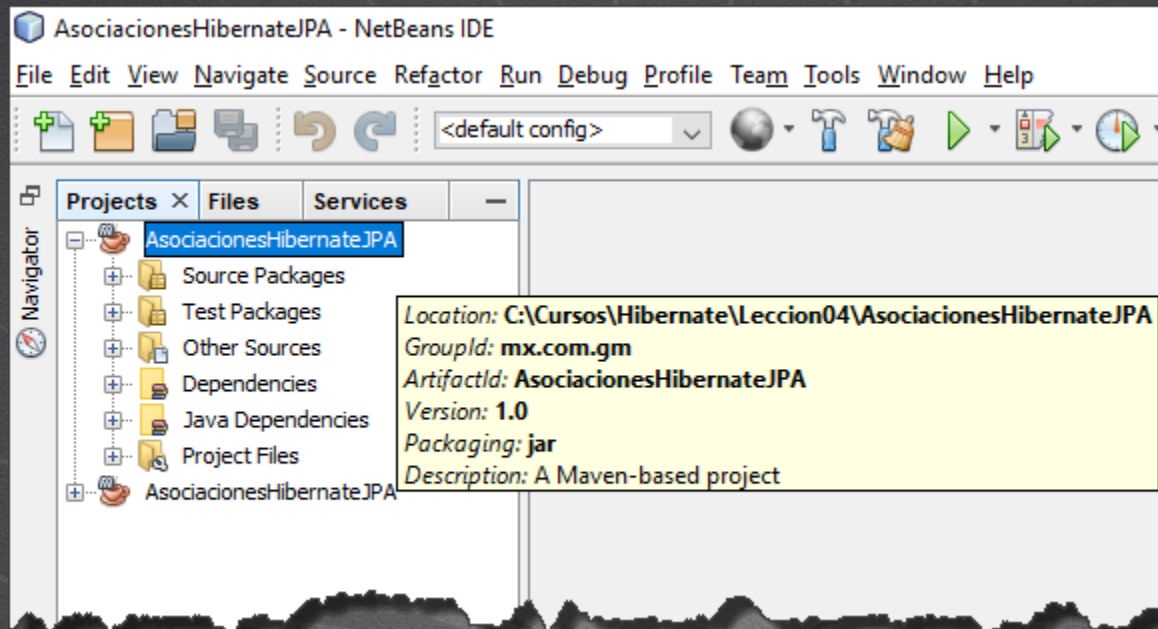
# PASO 1. CREAMOS EL PROYECTO

Copiamos y pegamos el proyecto anterior  
AsociacionesHibernateJPA:



# PASO 2. CERRAMOS EL PROYECTO

Revisamos el proyecto que deseamos cerrar:



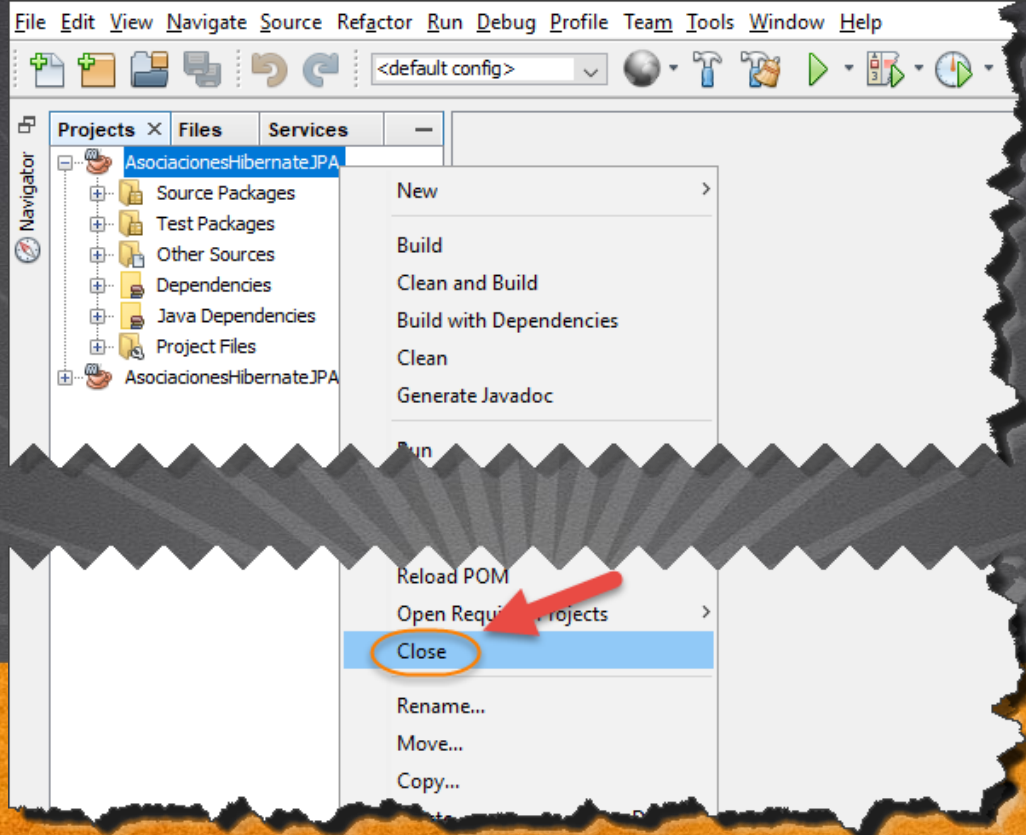
**CURSO DE HIBERNATE Y JPA**

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



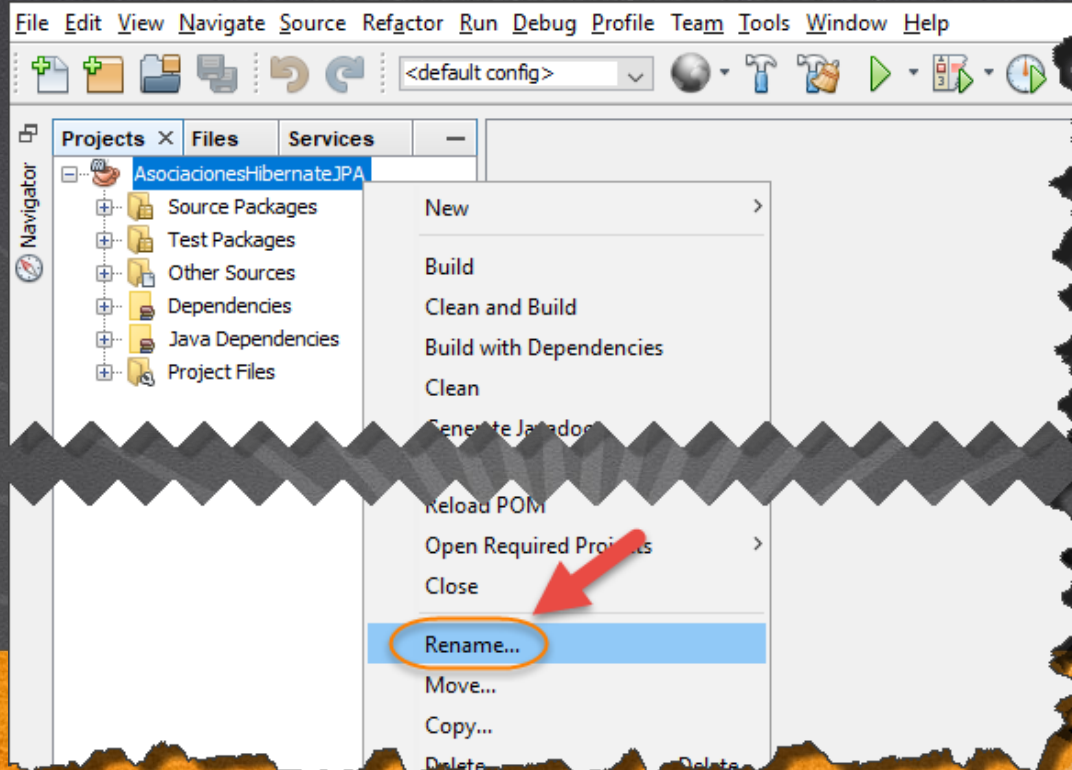
# PASO 2. CERRAMOS EL PROYECTO

Cerramos el proyecto que ya no utilizamos:



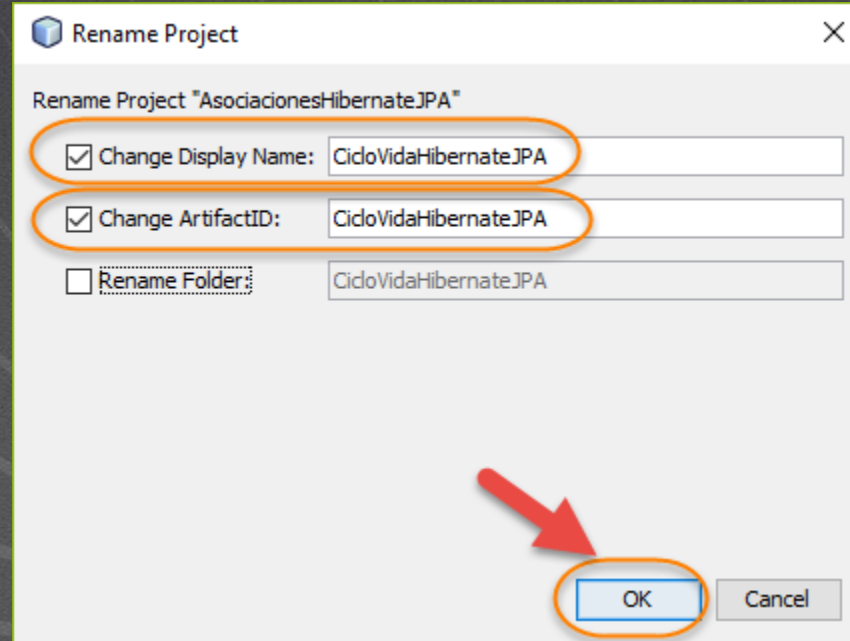
# PASO 3. RENOMBRAMOS EL PROYECTO

Renombramos el proyecto :



# PASO 3. RENOMBAMOS EL PROYECTO

Renombramos el proyecto :



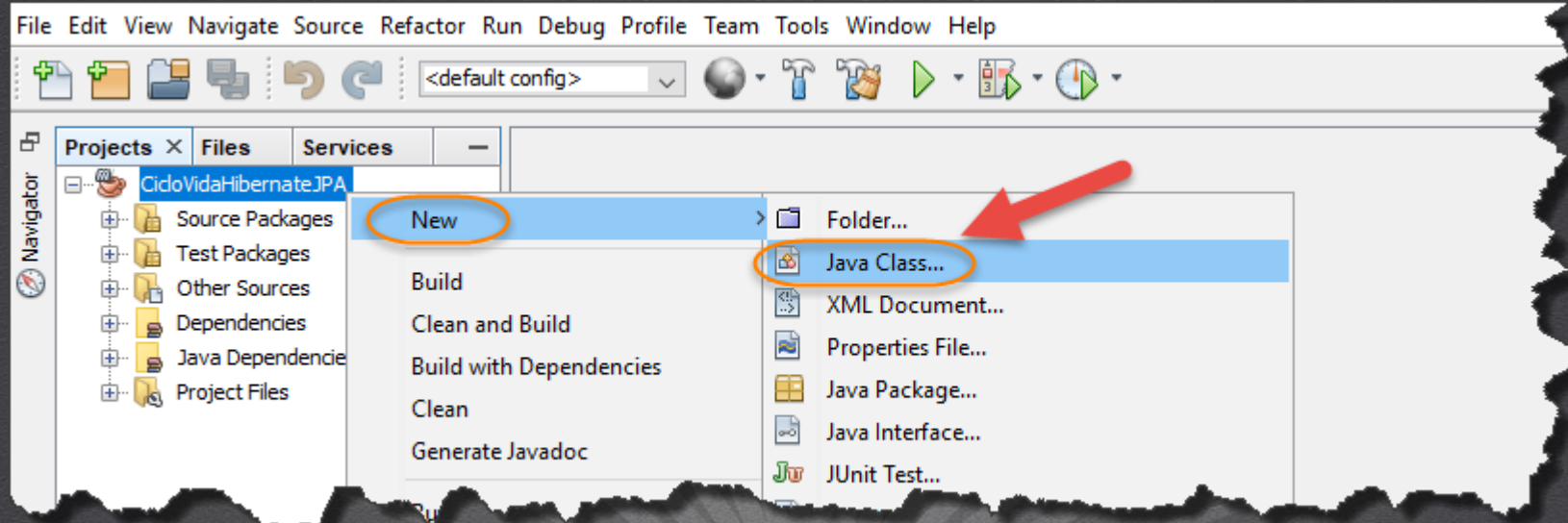
**CURSO DE HIBERNATE Y JPA**

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



# PASO 4. CREAMOS UNA CLASE JAVA

Creamos una clase Java:



**CURSO DE HIBERNATE Y JPA**

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

# PASO 4. CREAMOS UNA CLASE JAVA

Creamos una clase Java:

**New Java Class**

**Steps**

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

**Name and Location**

Class Name: Estado1Persistido

Project: AsociacionesHibernateJPA

Location: Source Packages

Package: test.cidovida

Created File: ste\Leccion05\CidoVidaHibernateJPA\src\main\java\test\cidovida\Estado1Persistido.java

< Back Next > **Finish** Cancel Help

# PASO 5. MODIFICAMOS EL CÓDIGO

Archivo Estado1Persistido.java:

Dar click para ir al código

```
package test.ciclovida;

import javax.persistence.*;
import model.Contacto;

public class Estado1Persistido {

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

        //1. Creamos el objeto (estado new o transitivo)
        Contacto contacto = new Contacto();
        contacto.setEmail1("email2@dominio.com");
        contacto.setVersion(0);
        contacto.setDeleted(0);
    }
}
```



# PASO 5. MODIFICAMOS EL CÓDIGO

Archivo Estado1Persistido.java:

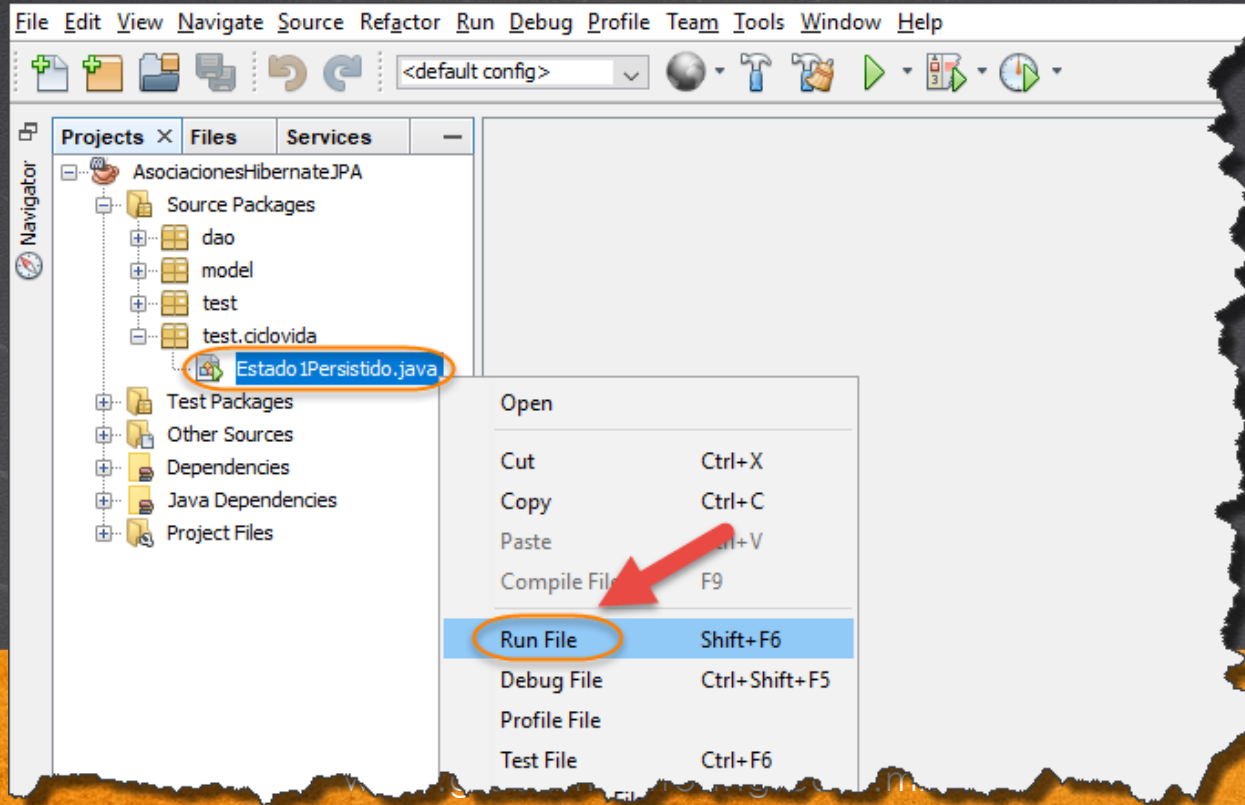
Dar click para ir al código

```
//2. Persistimos el objeto (cambia de estado a persistente)
//Iniciamos la transaccion
//No utilizaremos las clases DAO en estos ejercicios
try {
    // Iniciamos una transaccion
    em.getTransaction().begin();
    // Insertamos la nueva persona
    em.persist(contacto);
    // Terminamos la transaccion
    em.getTransaction().commit();
} catch (Exception ex) {
    System.out.println("Error al insetar objeto:" + ex.getMessage());
    // ex.printStackTrace();
} finally {
    if (em != null) {
        em.close();
    }
}

//El objeto de contacto cambia de edo. persistente a detached
//al cerrar la sesion
System.out.println("Nuevo contacto:" + contacto);
}
```

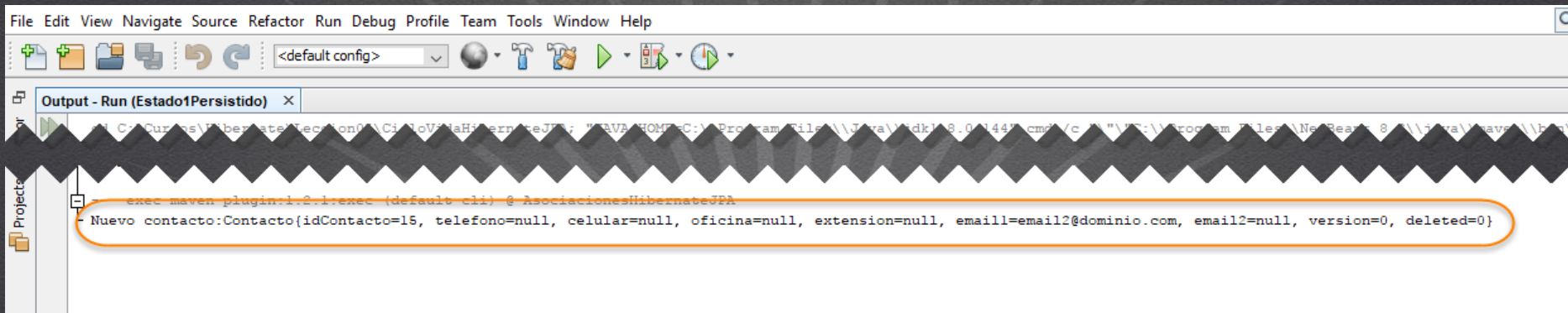
# PASO 6. EJECUTAMOS EL CÓDIGO

Ejecutamos la clase:



## PASO 6. EJECUTAMOS EL CÓDIGO

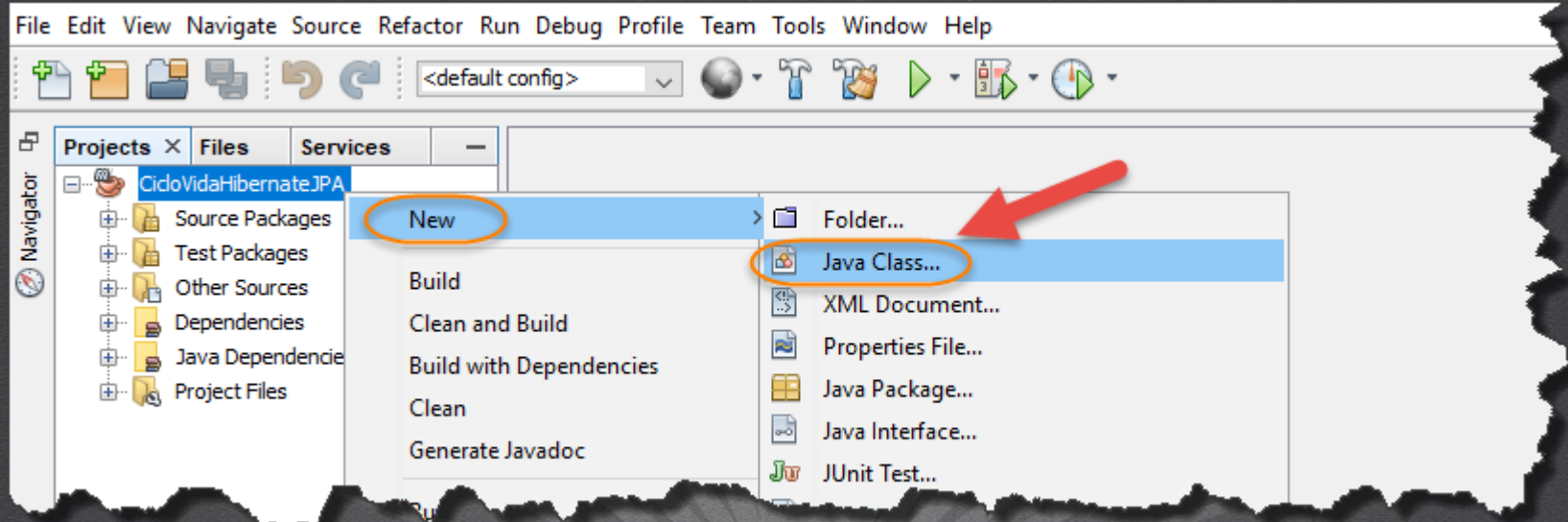
Observamos el resultado de insertar un nuevo objeto de tipo Contacto. El valor de idContacto puede variar según los registros que se tengan en la base de datos en la tabla de contacto:





# PASO 7. CREAMOS UNA CLASE JAVA

Creamos una clase Java:



**CURSO DE HIBERNATE Y JPA**

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

# PASO 7. CREAMOS UNA CLASE JAVA

Creamos una clase Java:

**New Java Class**

**Steps**

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

**Name and Location**

Class Name: Estado2RecuperarObjetoPersistente

Project: CidoVidaHibernateJPA

Location: Source Packages

Package: test.cidovida

Created File: VidaHibernateJPA\src\main\java\test\cidovida\Estado2RecuperarObjetoPersistente.java

< Back   Next >   **Finish**   Cancel   Help

# PASO 8. MODIFICAMOS EL CÓDIGO

[Archivo Estado2RecuperarObjetoPersistente.java:](#)

Dar click para ir al código

```
package test.ciclovida;

import javax.persistence.*;
import model.Contacto;

public class Estado2RecuperarObjetoPersistente {

    public static void main(String[] args) {
        /*Utilizamos la Unidad de Persistencia de JPA*/
        EntityManagerFactory fabrica = Persistence.createEntityManagerFactory("HibernateJpaPU");
        EntityManager em = fabrica.createEntityManager();
        /**
         * Objetivo: Recuperar un objeto persistente con Hibernate apoyandonos de JPA
         */
        //Existen 2 maneras de realizar esta accion
        //1. Metodo get, regresa null si no encuentra el objeto
        Contacto contacto1 = null;

        try {
            em.getTransaction().begin();
            //Proporcionamos el ID a recuperar, en JPA se usa find en lugar de get
            contacto1 = (Contacto) em.find(Contacto.class, 2);
            System.out.println("Contacto recuperado con find:" + contacto1);
            em.getTransaction().commit(); //hacemos flush
        } catch (Exception e) {
            em.getTransaction().rollback();
            e.printStackTrace();
        }
    }
}
```



# PASO 8. MODIFICAMOS EL CÓDIGO

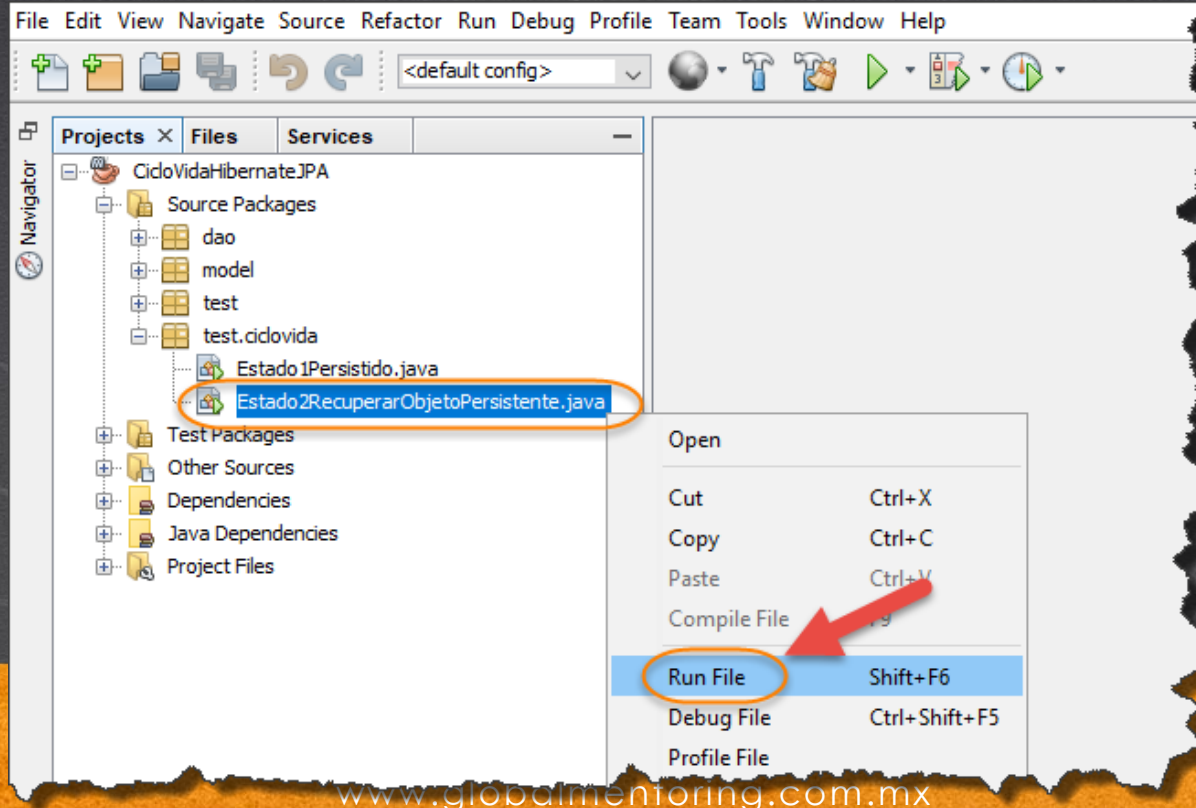
[Archivo Estado2RecuperarObjetoPersistente.java:](#)

Dar click para ir al código

```
//El objeto de contacto1 cambia a edo. detached
//al cerrar la sesion
//2. Metodo load, regresa ObjectNotFoundException
//si no encuentra el id proporcionado
Contacto contacto2 = null;
//Creamos una segunda transaccion
try {
    em.getTransaction().begin();
    //Proporcionamos el id a recuperar, no levanta las relaciones (lazy)
    //En JPA tenemos getReference como otra opcion
    contacto2 = (Contacto) em.getReference(Contacto.class, 2);
    System.out.println("Contacto recuperado con getReference:" + contacto2);
    em.getTransaction().commit();
} catch (Exception e) {
    em.getTransaction().rollback();
    e.printStackTrace();
} finally {
    if (em != null) {
        em.close();
    }
}
//El objeto contacto2 cambia a edo. detached
//al cerrar la transaccion 2
}
```

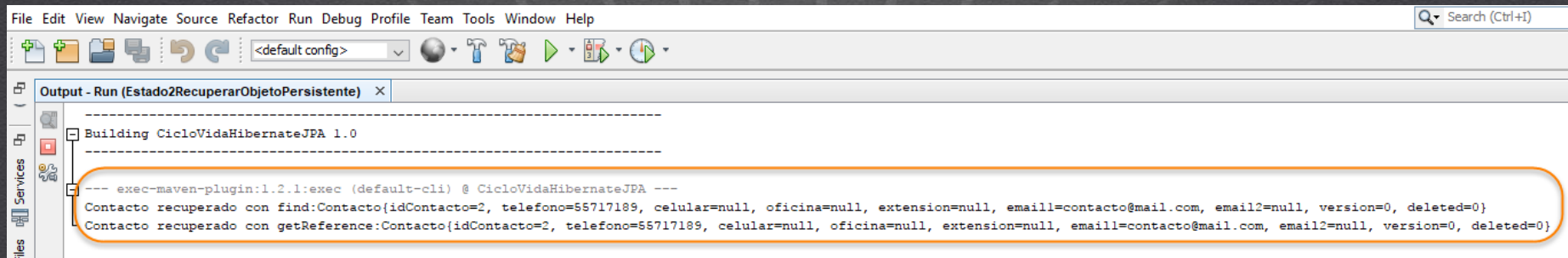
# PASO 9. EJECUTAMOS EL CÓDIGO

Ejecutamos la clase:



# PASO 9. EJECUTAMOS EL CÓDIGO

Observamos el resultado de ejecutar nuestra clase. El idContacto a buscar podría no existir, por lo que se debe adecuar el valor a un valor existente.

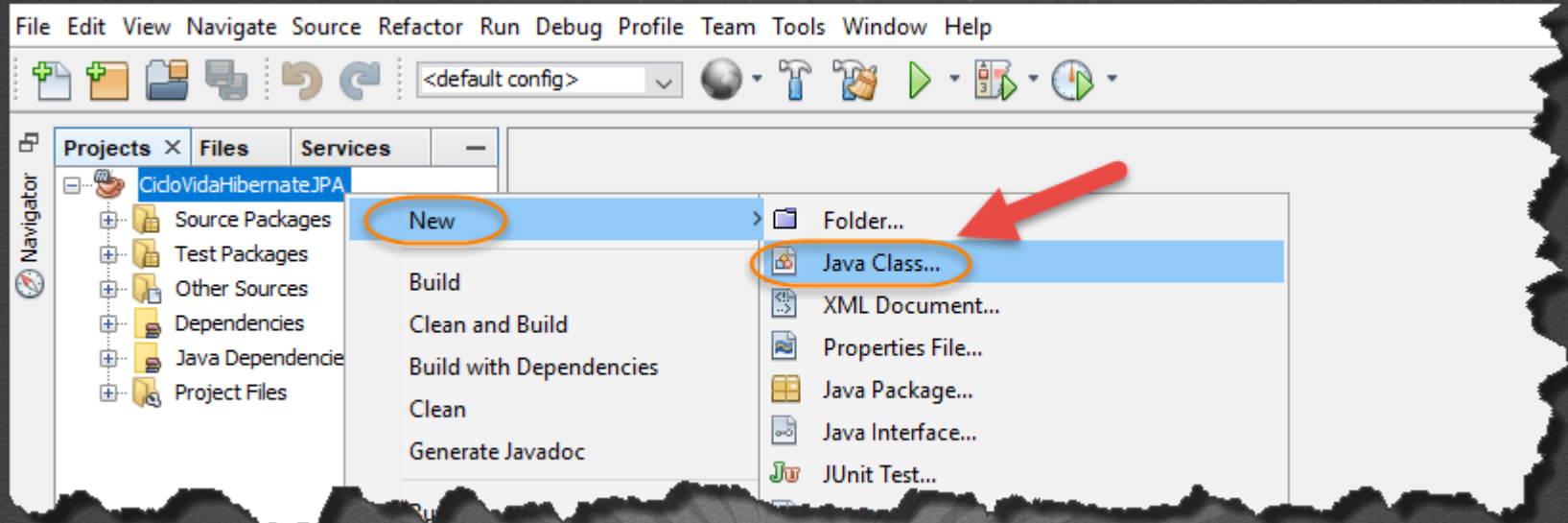


```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config>
Output - Run (Estado2RecuperarObjetoPersistente) X
-----
Building CicloVidaHibernateJPA 1.0
-----
--- exec-maven-plugin:1.2.1:exec (default-cli) @ CicloVidaHibernateJPA ---
Contacto recuperado con find:Contacto{idContacto=2, telefono=55717189, celular=null, oficina=null, extension=null, email1=contacto@mail.com, email2=null, version=0, deleted=0}
Contacto recuperado con getReference:Contacto{idContacto=2, telefono=55717189, celular=null, oficina=null, extension=null, email1=contacto@mail.com, email2=null, version=0, deleted=0}
```



# PASO 10. CREAMOS UNA CLASE JAVA

Creamos una clase Java:



**CURSO DE HIBERNATE Y JPA**

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

# PASO 10. CREAMOS UNA CLASE JAVA

Creamos una clase Java:

**New Java Class**

**Steps**

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

**Name and Location**

Class Name: Estado3ModificarObjetoPersistente

Project: CidoVidaHibernateJPA

Location: Source Packages

Package: test.cidovida

Created File: oVidaHibernateJPA\src\main\java\test\cidovida\Estado3ModificarObjetoPersistente.java

< Back Next > **Finish** Cancel Help

# PASO 11. MODIFICAMOS EL CÓDIGO

[Archivo Estado3ModificarObjetoPersistente.java:](#)

Dar click para ir al código

```
package test.ciclovida;

import javax.persistence.*;
import model.Contacto;

public class Estado3ModificarObjetoPersistente {

    public static void main(String[] args) {
        /*Utilizamos la Unidad de Persistencia de JPA*/
        EntityManagerFactory fabrica = Persistence.createEntityManagerFactory("HibernateJpaPU");
        EntityManager em = fabrica.createEntityManager();
        /**
         * Objetivo: Modificar un objeto persistente (reattaching) Pasar de un
         * estado detached a persistente
         */
        //Recuperamos un objeto persistente
        Contacto contacto = null;
        try {
            em.getTransaction().begin();
            //Identificador a recuperar, también se puede usar merge para recuperar un objeto
            contacto = (Contacto) em.find(Contacto.class, 1);
            em.getTransaction().commit(); //hacemos flush
        } catch (Exception e) {
            em.getTransaction().rollback();
            e.printStackTrace(System.out);
        }
    }
}
```



# PASO 11. MODIFICAMOS EL CÓDIGO

Archivo Estado3ModificarObjetoPersistente.java:

Dar click para ir al código

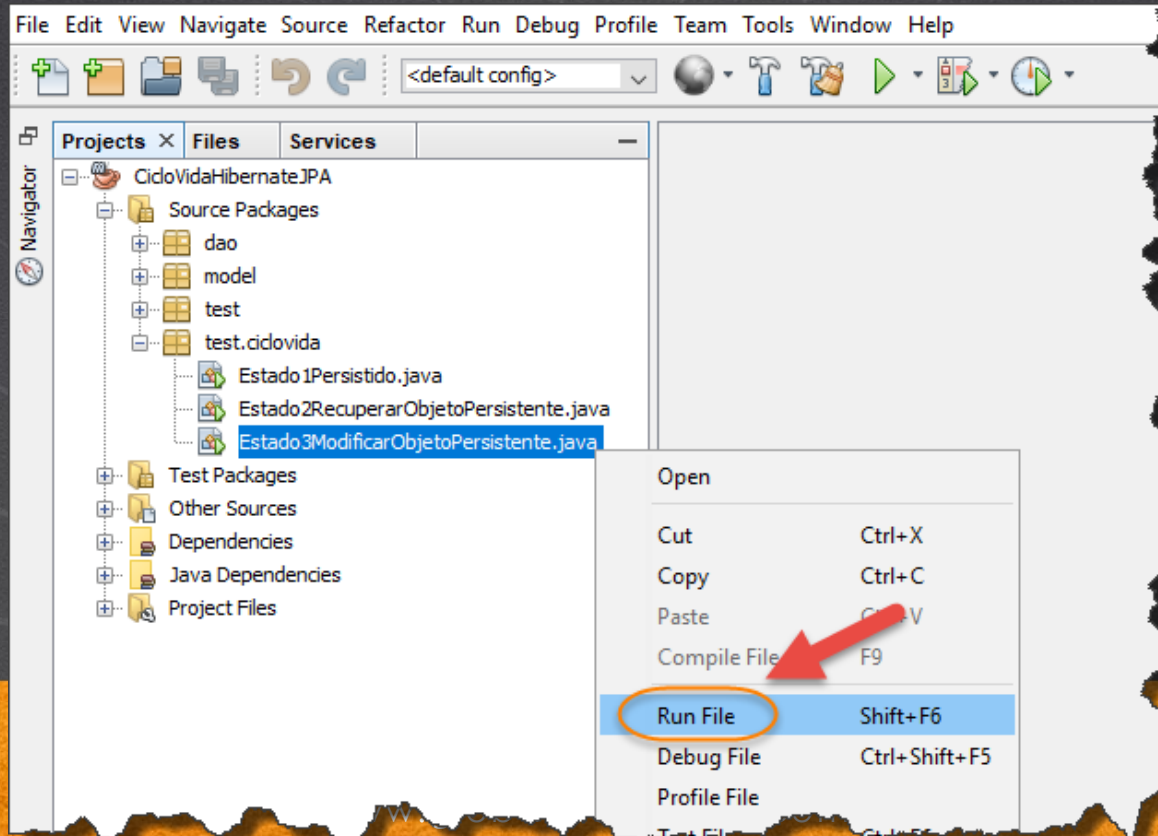
```
//Imprimimos los valores del objeto recuperado
//estado detached
System.out.println("Contacto recuperado:" + contacto);

//Modificamos el objeto fuera de una transaccion
if(contacto != null)
    contacto.setEmail2("emailModificado@mail.com");

//Volvemos a guardar el objeto en una nueva transaccion
//Cambiamos el estado de detached a persistente (reattaching)
try {
    em.getTransaction().begin();
    em.merge(contacto);
    em.getTransaction().commit(); //hacemos flush
} catch (Exception e) {
    em.getTransaction().rollback();
    e.printStackTrace(System.out);
} finally {
    if (em != null) {
        em.close();
    }
}
System.out.println("Objeto contacto modificado:" + contacto);
//El objeto de contacto cambia a edo. detached al cerrar session
}
```

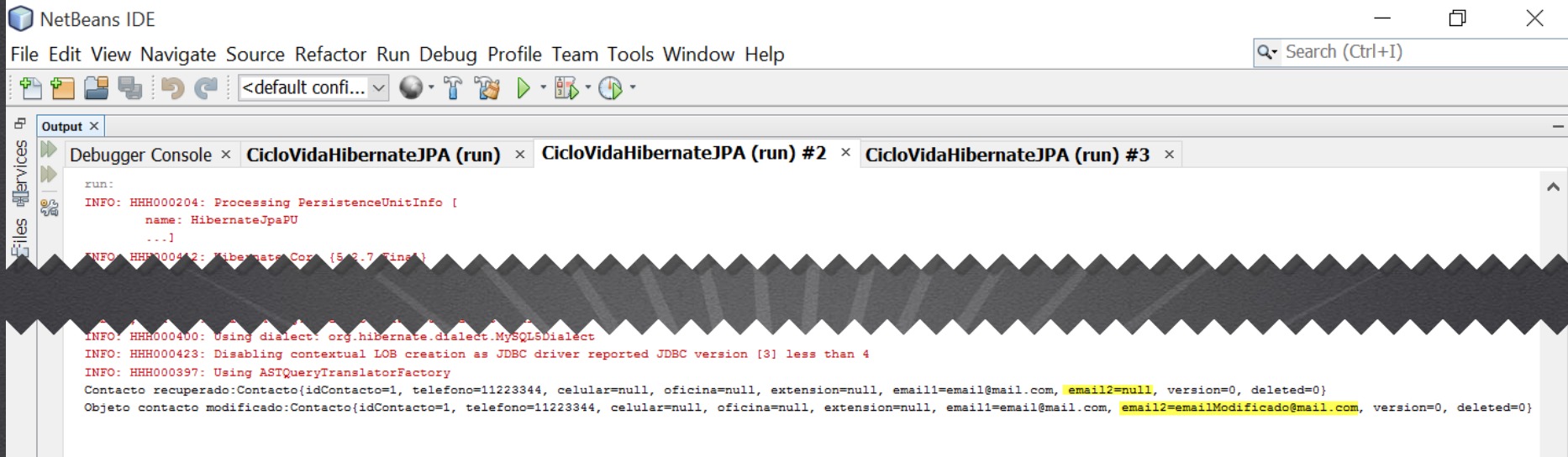
# PASO 12. EJECUTAMOS EL CÓDIGO

Ejecutamos la clase:



# PASO 12. EJECUTAMOS EL CÓDIGO

Ejecutamos la clase:



NetBeans IDE

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+I)

Output x

Debugger Console x CicloVidaHibernateJPA (run) x CicloVidaHibernateJPA (run) #2 x CicloVidaHibernateJPA (run) #3 x

```
run:
INFO: HHH000204: Processing PersistenceUnitInfo [
    name: HibernateJpaPU
    ...]
INFO: HHH000412: Hibernate Core (5.2.7.Final)

INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL5Dialect
INFO: HHH000423: Disabling contextual LOB creation as JDBC driver reported JDBC version [3] less than 4
INFO: HHH000397: Using ASTQueryTranslatorFactory
Contacto recuperado:Contacto{idContacto=1, telefono=11223344, celular=null, oficina=null, extension=null, email1=email@mail.com, email2=null, version=0, deleted=0}
Objeto contacto modificado:Contacto{idContacto=1, telefono=11223344, celular=null, oficina=null, extension=null, email1=email@mail.com, email2=emailModificado@mail.com, version=0, deleted=0}
```

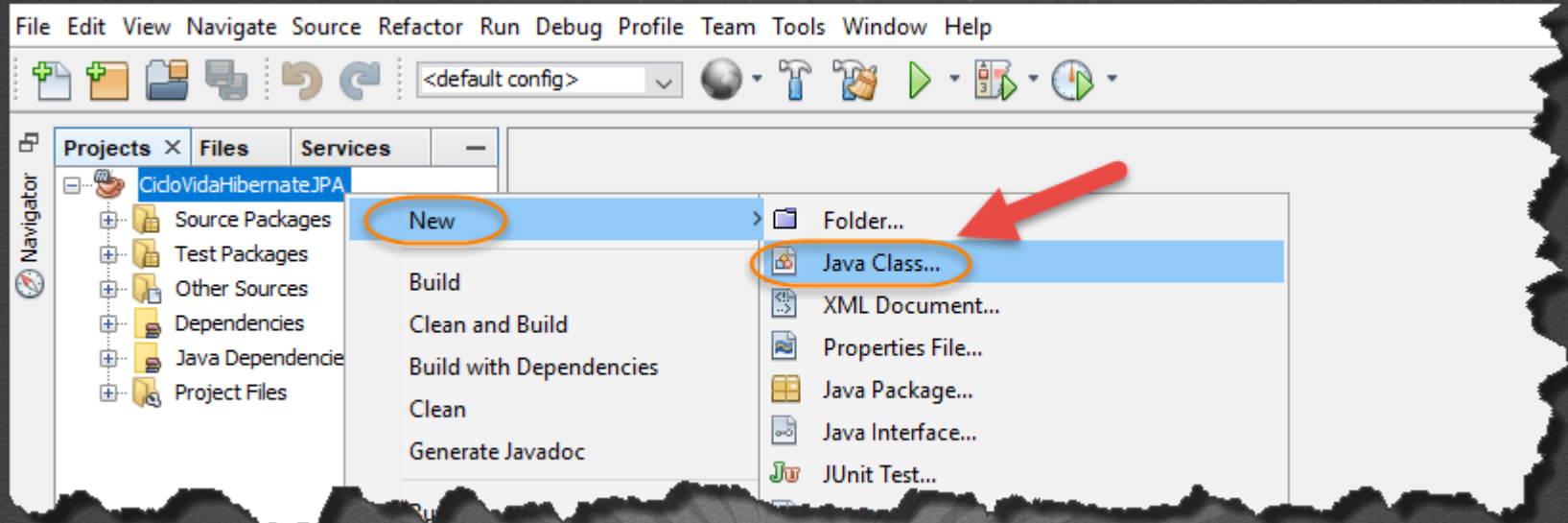
**CURSO DE HIBERNATE Y JPA**

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



# PASO 13. CREAMOS UNA CLASE JAVA

Creamos una clase Java:



**CURSO DE HIBERNATE Y JPA**

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

# PASO 13. CREAMOS UNA CLASE JAVA

Creamos una clase Java:

**New Java Class**

**Steps**

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

**Name and Location**

Class Name: Estado3ModificarObjetoPersistente

Project: CidoVidaHibernateJPA

Location: Source Packages

Package: test.cidovida

Created File: oVidaHibernateJPA\src\main\java\test\cidovida\Estado3ModificarObjetoPersistente.java

< Back Next > **Finish** Cancel Help

# PASO 14. MODIFICAMOS EL CÓDIGO

[Archivo Estado4ModificarObjetoSesionLarga.java:](#)

Dar click para ir al código

```
package test.ciclovida;

import javax.persistence.*;
import model.Contacto;

public class Estado4ModificarObjetoSesionLarga {

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

        //Recuperamos un objeto persistente
        Contacto contacto = null;
```



# PASO 14. MODIFICAMOS EL CÓDIGO

[Archivo Estado4ModificarObjetoSesionLarga.java:](#)

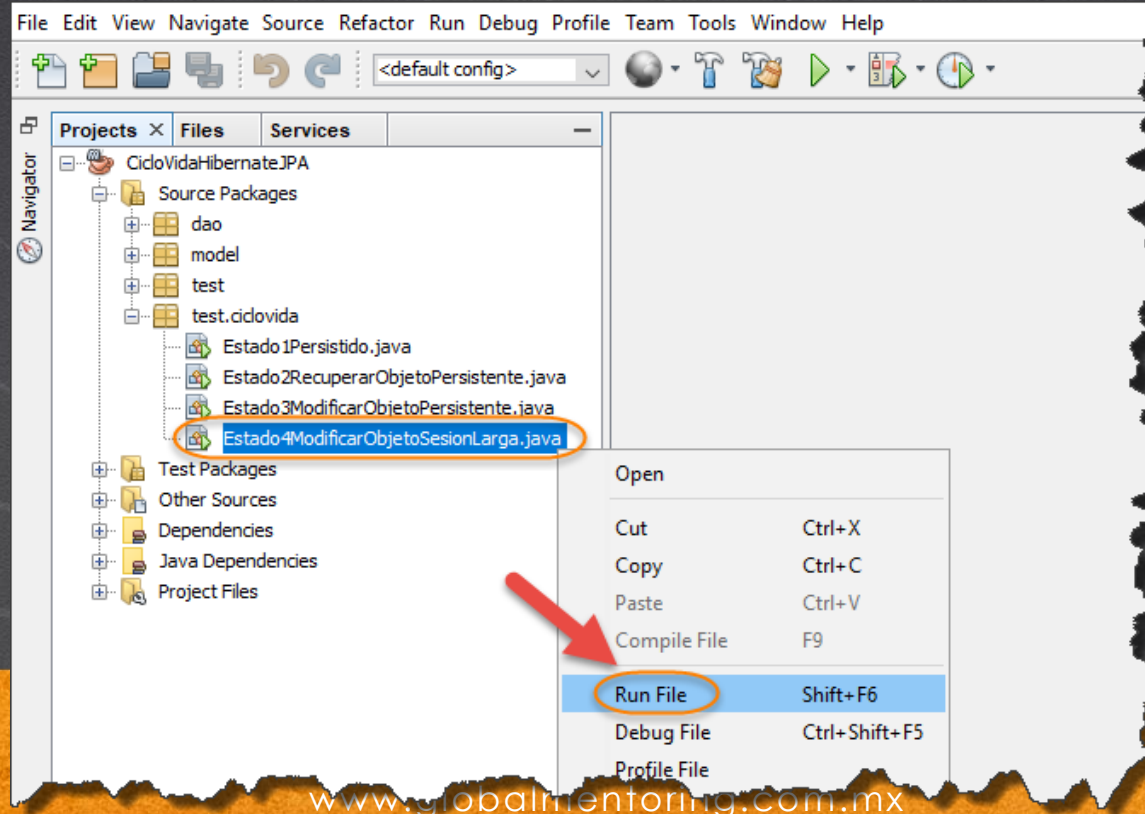
Dar click para ir al código

```
try {
    em.getTransaction().begin();
    //Identificador a recuperar, tambien se puede usar merge para recuperar un objeto
    contacto = (Contacto) em.find(Contacto.class, 1);
    System.out.println("Contacto sin modificar:" + contacto);
    //Modificamos el objeto en la misma transaccion
    contacto.setTelefono("55551111");
    em.getTransaction().commit(); //hacemos flush
} catch (Exception e) {
    em.getTransaction().rollback();
    e.printStackTrace(System.out);
} finally {
    if (em != null) {
        em.close();
    }
}

//Imprimimos los valores (estado detached)
System.out.println("Contacto recuperado:" + contacto);
}
```

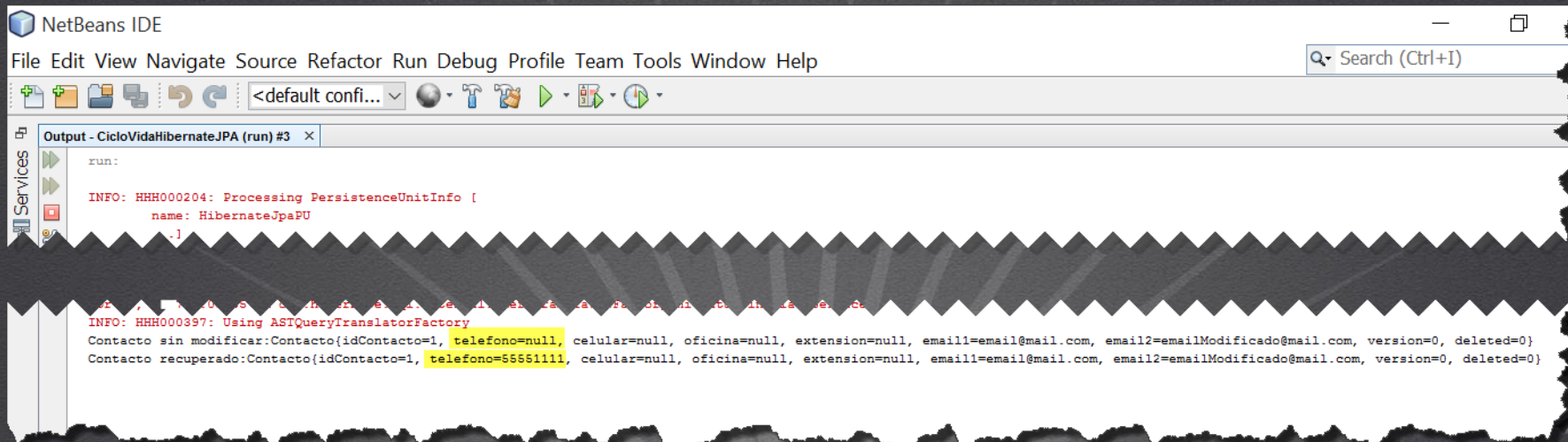
# PASO 15. EJECUTAMOS EL CÓDIGO

Ejecutamos la clase:



# PASO 15. EJECUTAMOS EL CÓDIGO

Ejecutamos la clase:



The screenshot shows the NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. Below the menu is a toolbar with icons for file operations and running code. The main window displays the 'Output - CicloVidaHibernateJPA (run) #3' console. The output text is as follows:

```
run:
INFO: HHH000204: Processing PersistenceUnitInfo [
    name: HibernateJpaPU
    ...
]
INFO: HHH000397: Using ASTQueryTranslatorFactory
Contacto sin modificar:Contacto{idContacto=1, telefono=null, celular=null, oficina=null, extension=null, email1=email@mail.com, email2=emailModificado@mail.com, version=0, deleted=0}
Contacto recuperado:Contacto{idContacto=1, telefono=55551111, celular=null, oficina=null, extension=null, email1=email@mail.com, email2=emailModificado@mail.com, version=0, deleted=0}
```

**CURSO DE HIBERNATE Y JPA**

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



# CONCLUSIÓN DEL EJERCICIO

- Con este ejercicio hemos visto el ciclo de vida de los objetos de entidad apoyándonos del API de JPA.
- El ciclo de vida de los objetos de Entidad en Hibernate / JPA es muy importante conocerlo, ya que de eso depende que podamos realizar correctamente las operaciones sobre nuestros objetos de Entidad y al final sobre nuestros registros en la tabla de base de datos respectiva.
- Con esto concluimos el tema de ciclo de vida en Hibernate / JPA.

**CURSO ONLINE**

# **HIBERNATE & JPA**

Por: Ing. Ubaldo Acosta



Experiencia y Conocimiento para tu vida

**CURSO DE HIBERNATE Y JPA**

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