

CURSO DE HIBERNATE Y JPA

EJERCICIO

ASOCIACIONES EN HIBERNATE/JPA



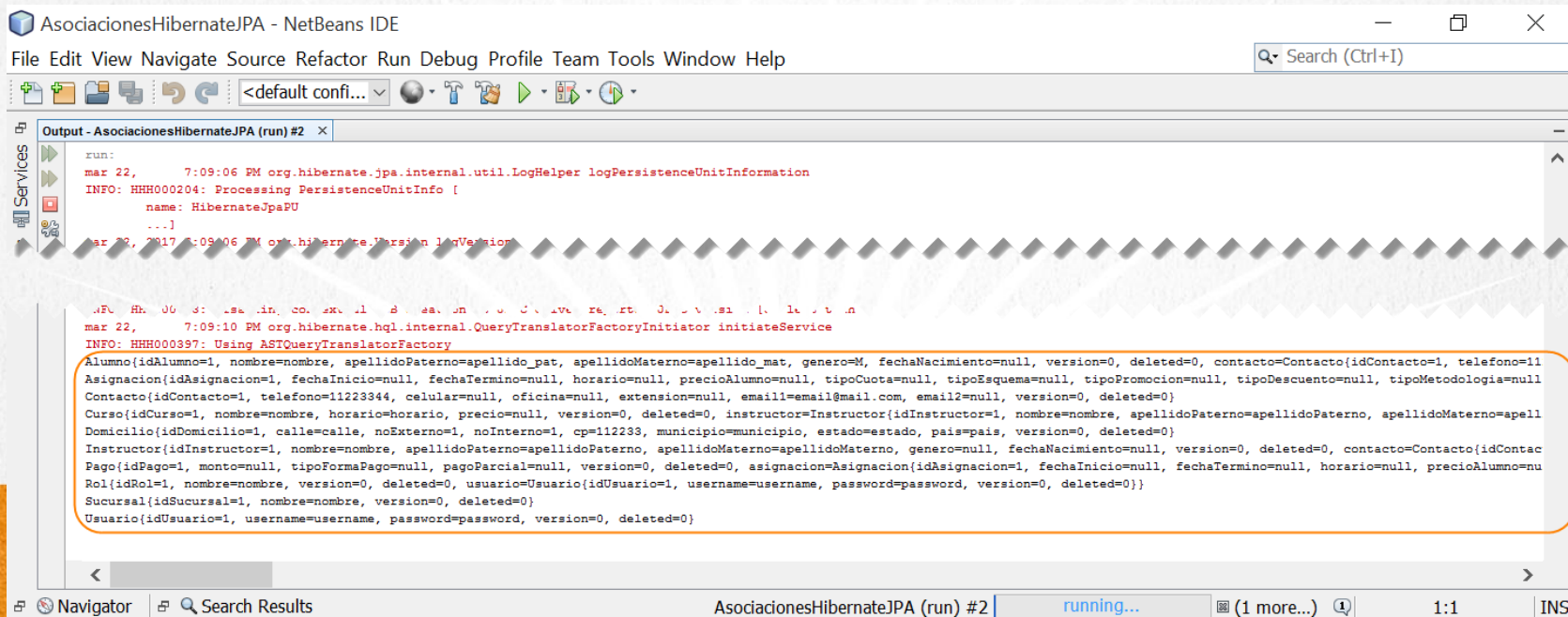
Experiencia y Conocimiento para tu vida

CURSO DE HIBERNATE Y JPA

www.globalmentoring.com.mx

OBJETIVO DEL EJERCICIO

Crear un programa para practicar el mapeo de relaciones con Hibernate y/o JPA. Al finalizar deberemos observar lo siguiente:



AsociacionesHibernateJPA - NetBeans IDE

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

Search (Ctrl+I)

Output - AsociacionesHibernateJPA (run) #2

```
run:
mar 22, 7:09:06 PM org.hibernate.jpa.internal.util.LogHelper logPersistenceUnitInformation
INFO: HHH000204: Processing PersistenceUnitInfo [
    name: HibernateJpaPU
    ...]
mar 22, 2017 7:09:06 PM org.hibernate.Version logVersion
INFO: HHH000039: Using ASTQueryTranslatorFactory

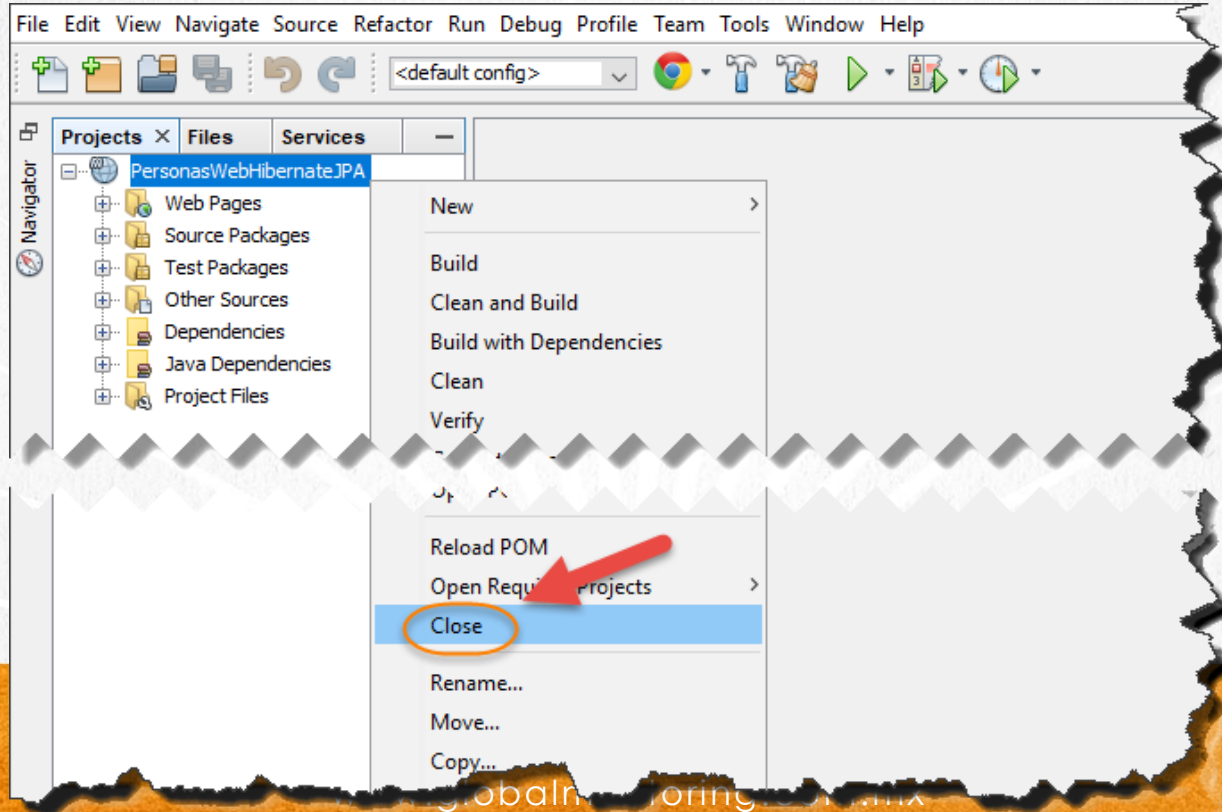
mar 22, 7:09:10 PM org.hibernate.hql.internal.QueryTranslatorFactoryInitiator initiateService
INFO: HHH000039: Using ASTQueryTranslatorFactory

Alumno{idAlumno=1, nombre=nombre, apellidoPaterno=apellido_pat, apellidoMaterno=apellido_mat, genero=M, fechaNacimiento=null, version=0, deleted=0, contacto=Contacto{idContacto=1, telefono=11
Asignacion{idAsignacion=1, fechaInicio=null, fechaTermino=null, horario=null, precioAlumno=null, tipoCuota=null, tipoEsquema=null, tipoPromocion=null, tipoDescuento=null, tipoMetodologia=null
Contacto{idContacto=1, telefono=11223344, celular=null, oficina=null, extension=null, email1=email@mail.com, email2=null, version=0, deleted=0}
Curso{idCurso=1, nombre=nombre, horario=horario, precio=null, version=0, deleted=0, instructor=Instructor{idInstructor=1, nombre=nombre, apellidoPaterno=apellidoPaterno, apellidoMaterno=apell
Domicilio{idDomicilio=1, calle=calle, noExterno=1, noInterno=1, cpe=112233, municipio=municipio, estado=estado, pais=pais, version=0, deleted=0}
Instructor{idInstructor=1, nombre=nombre, apellidoPaterno=apellidoPaterno, apellidoMaterno=apellidoMaterno, genero=null, fechaNacimiento=null, version=0, deleted=0, contacto=Contacto{idContac
Pago{idPago=1, monto=null, tipoFormaPago=null, pagoParcial=null, version=0, deleted=0, asignacion=Asignacion{idAsignacion=1, fechaInicio=null, fechaTermino=null, horario=null, precioAlumno=nu
Rol{idRol=1, nombre=nombre, version=0, deleted=0, usuario=Usuario{idUsuario=1, username=username, password=password, version=0, deleted=0}}
Sucursal{idSucursal=1, nombre=nombre, version=0, deleted=0}
Usuario{idUsuario=1, username=username, password=password, version=0, deleted=0}
```

Navigator Search Results AsociacionesHibernateJPA (run) #2 running... (1 more...) 1:1 INS

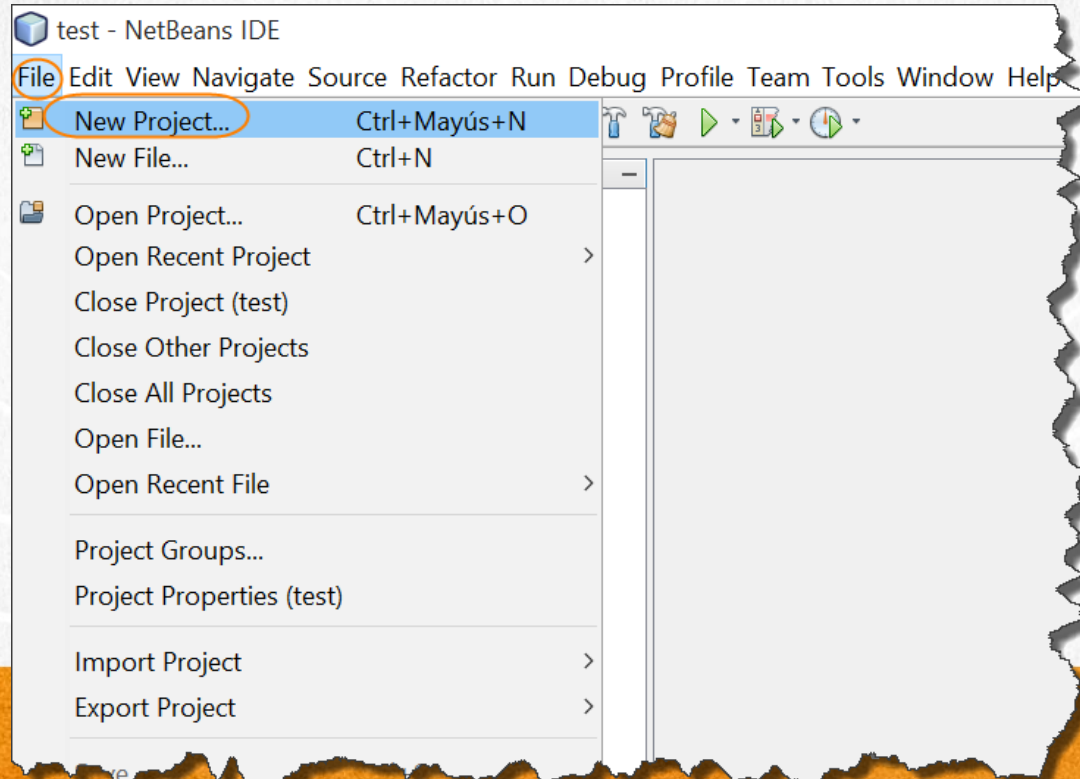
CERRAMOS CUALQUIER PROYECTO ABIERTO

Cerramos cualquier otro proyecto abierto:



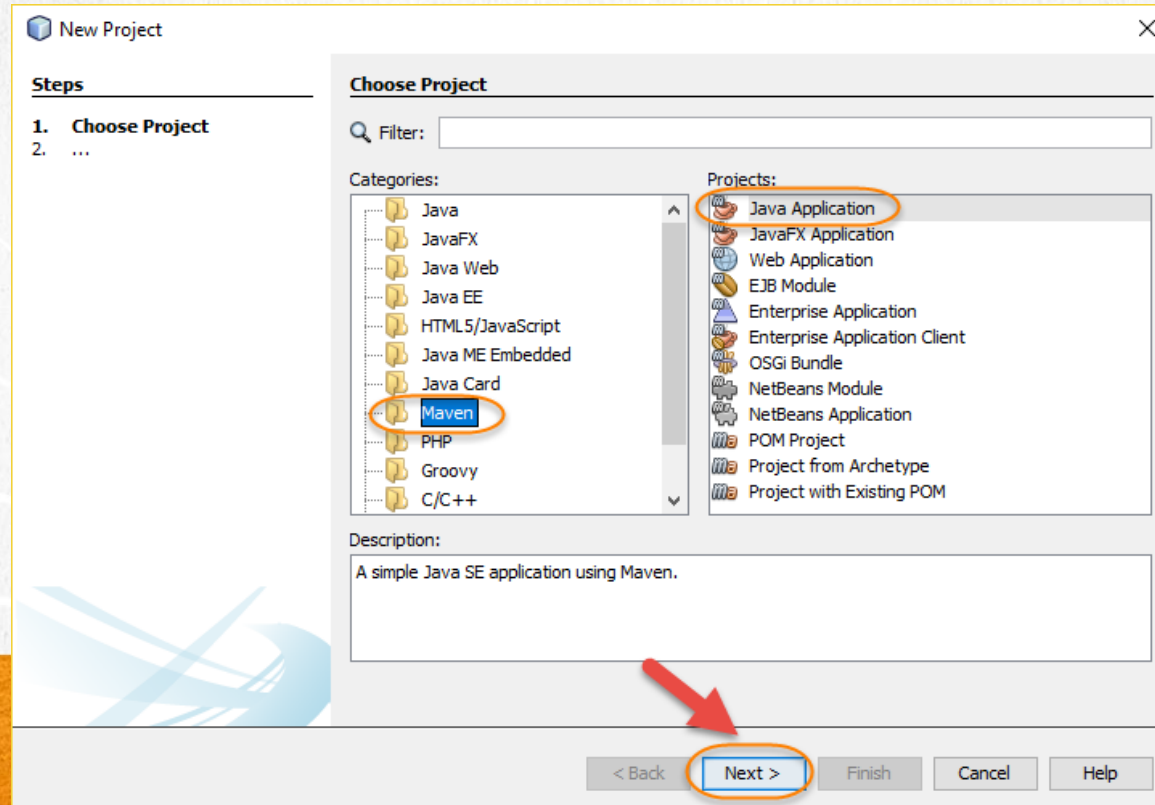
PASO 1. CREACIÓN DEL PROYECTO

Vamos a crear el proyecto:



PASO 1. CREACIÓN DEL PROYECTO

Vamos a crear el proyecto:



PASO 1. CREACIÓN DEL PROYECTO

Vamos a crear el proyecto:

New Java Application [X]

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name: AsociacionesHibernate.JPA

Project Location: C:\Cursos\Hibernate\Leccion04 [Browse...]

Project Folder: C:\Cursos\Hibernate\Leccion04\AsociacionesHibernate.JPA

Artifact Id: AsociacionesHibernate.JPA

Group Id: mx.com.gm

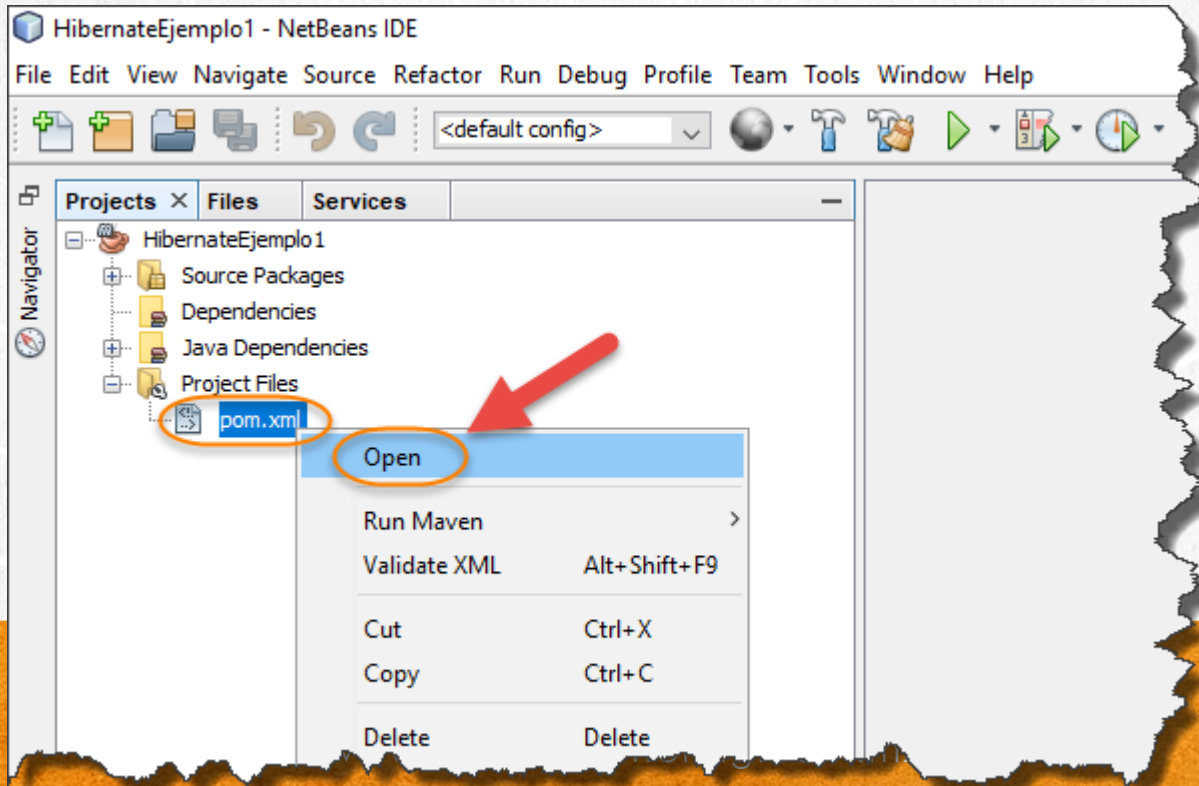
Version: 1.0

Package: [] (Optional)

< Back Next > **Finish** Cancel Help

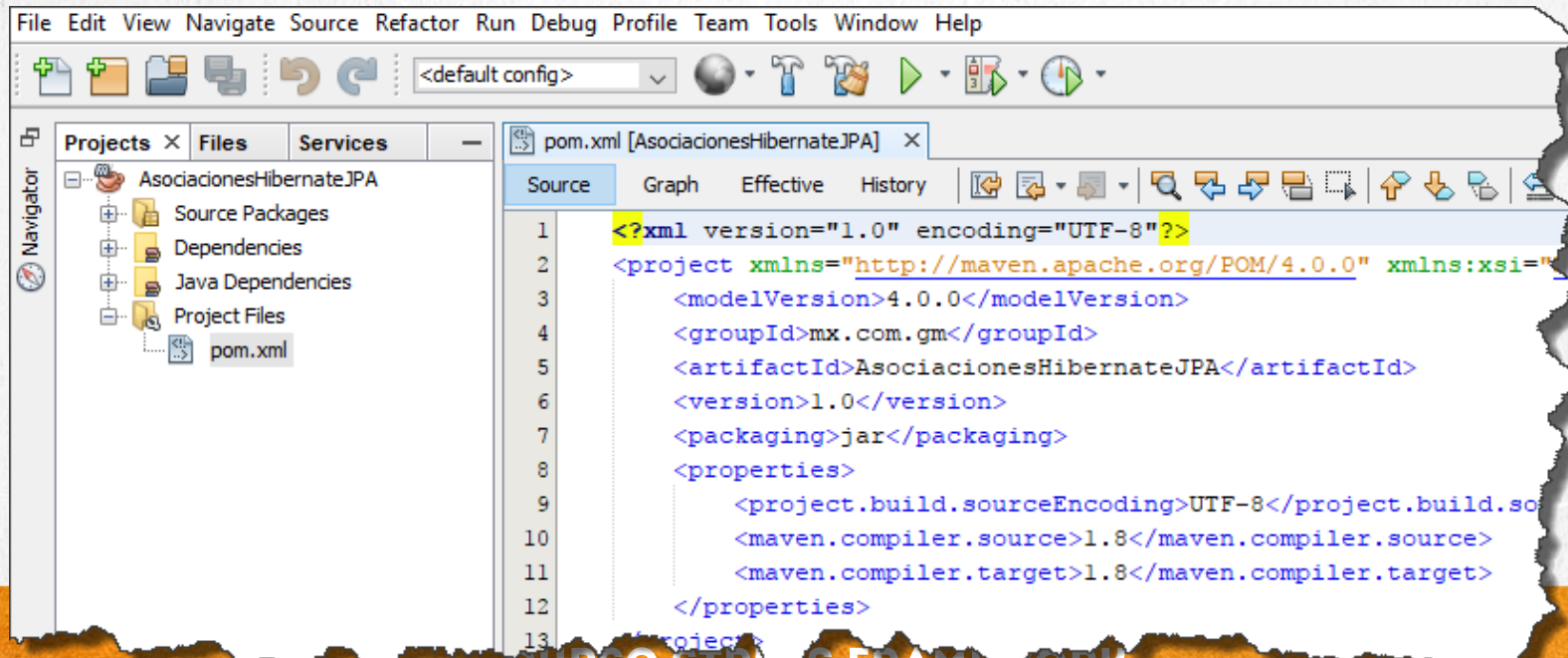
2. ABRIMOS EL ARCHIVO POM.XML DE MAVEN

- El archivo pom.xml de maven administra las librerías Java vamos a utilizar. Abrimos el archivo pom.xml para modificarlo con el código siguiente:



2. ABRIMOS EL ARCHIVO POM.XML DE MAVEN

- Una vez abierto, vamos a modificar la información por completo de este archivo, con la información proporcionada a continuación:



PASO 3. MODIFICAMOS EL CÓDIGO

Archivo pom.xml:

Clic para ver el
archivo

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>mx.com.gm</groupId>
  <artifactId>AsociacionesHibernateJPA</artifactId>
  <version>1.0</version>
  <packaging>jar</packaging>
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
  <dependencies>
    <dependency>
      <groupId>org.hibernate</groupId>
      <artifactId>hibernate-core</artifactId>
      <version>5.3.0.CR2</version>
    </dependency>
    <dependency>
      <groupId>mysql</groupId>
      <artifactId>mysql-connector-java</artifactId>
      <version>5.1.46</version>
    </dependency>
  </dependencies>
</project>
```

PASO 3. MODIFICAMOS EL CÓDIGO

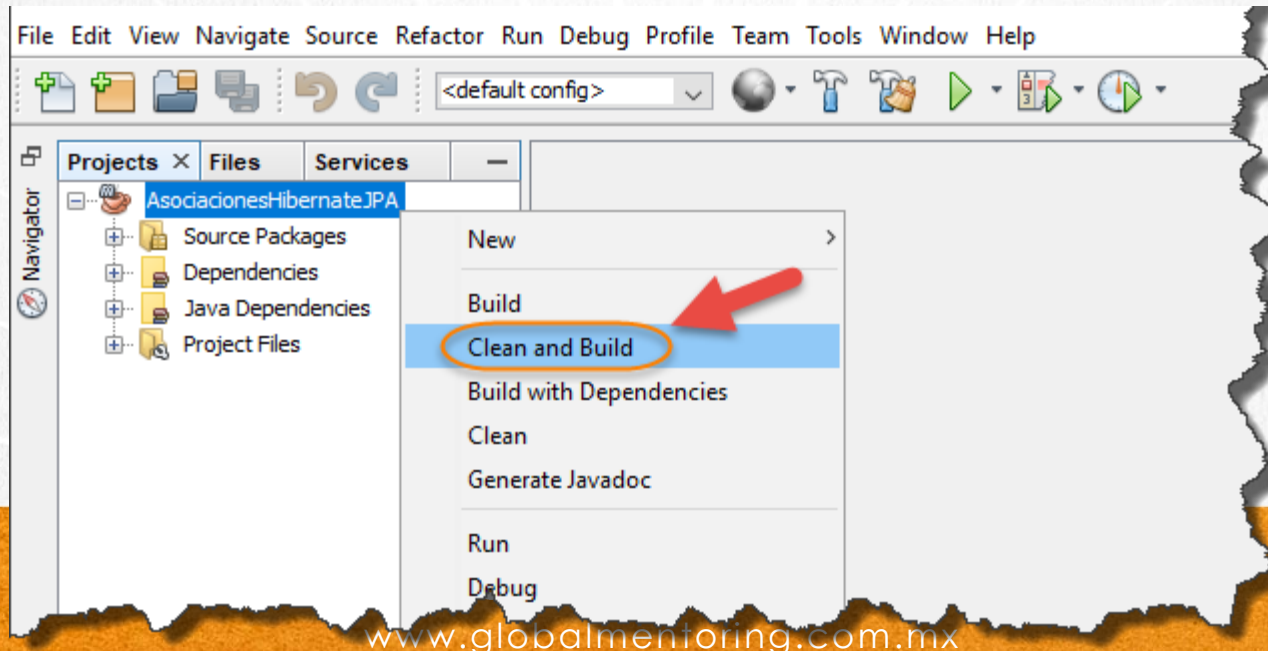
Archivo pom.xml:

Clic para ver el
archivo

```
<dependency>
  <groupId>org.apache.logging.log4j</groupId>
  <artifactId>log4j-api</artifactId>
  <version>2.10.0</version>
</dependency>
<dependency>
  <groupId>org.apache.logging.log4j</groupId>
  <artifactId>log4j-core</artifactId>
  <version>2.10.0</version>
</dependency>
</dependencies>
</project>
```

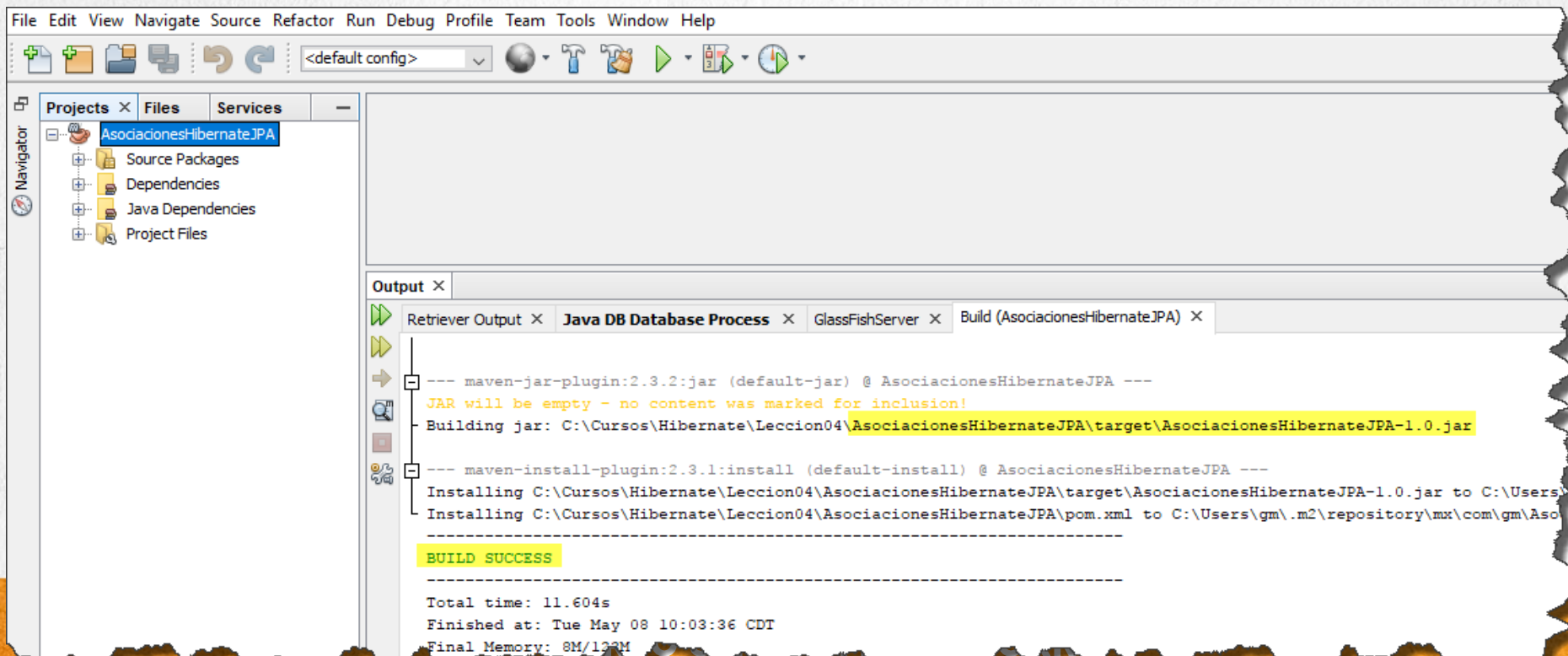
4. HACEMOS CLEAN & BUILD

• Para que se descarguen las librerías, hacemos Clean & Build al proyecto. Si por alguna razón este proceso falla, se debe desactivar cualquier software como antivirus, Windows defender o firewall durante este proceso para que no se impida la descarga de archivos .jar de Java. Una vez terminado se pueden volver a activar estos servicios. Este proceso puede demorar varios minutos dependiendo de su velocidad de internet:



4. HACEMOS CLEAN & BUILD

- Una vez terminado el proceso se debe mostrar una salida similar a la siguiente:



The screenshot shows an IDE interface with the 'Output' window open. The 'Output' window has several tabs, and the active tab is 'Build (AsociacionesHibernateJPA)'. The output text shows the Maven build process, including the execution of the 'maven-jar-plugin' and 'maven-install-plugin'. The build is successful, and the output ends with 'BUILD SUCCESS'.

```
Retriever Output x Java DB Database Process x GlassFishServer x Build (AsociacionesHibernateJPA) x

--- maven-jar-plugin:2.3.2:jar (default-jar) @ AsociacionesHibernateJPA ---
JAR will be empty - no content was marked for inclusion!
Building jar: C:\Cursos\Hibernate\Leccion04\AsociacionesHibernateJPA\target\AsociacionesHibernateJPA-1.0.jar

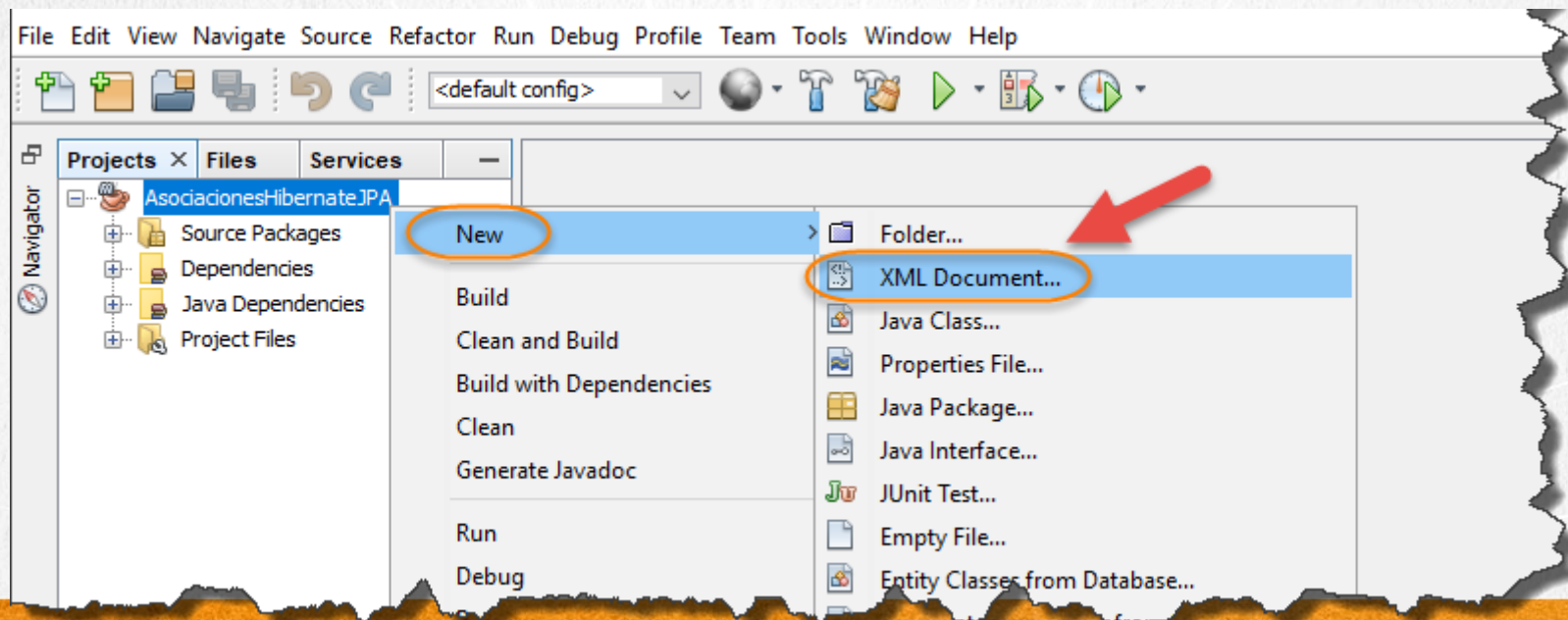
--- maven-install-plugin:2.3.1:install (default-install) @ AsociacionesHibernateJPA ---
Installing C:\Cursos\Hibernate\Leccion04\AsociacionesHibernateJPA\target\AsociacionesHibernateJPA-1.0.jar to C:\Users\
Installing C:\Cursos\Hibernate\Leccion04\AsociacionesHibernateJPA\pom.xml to C:\Users\gm\.m2\repository\mx\com\gm\AsociacionesHibernateJPA\pom.xml

BUILD SUCCESS

Total time: 11.604s
Finished at: Tue May 08 10:03:36 CDT
Final Memory: 8M/128M
```

5. CREAR UN ARCHIVO XML

- Creamos el archivo persistence.xml. El archivo persistence.xml tiene la configuración de conexión a la base de datos, en este caso mysql, entre otros valores, como las clases de entidad que vamos a utilizar en el proyecto, y el proveedor del Entity Manager:



CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

5. CREAR UN ARCHIVO XML

- Creamos el archivo persistence.xml. Lo depositamos en la ruta indicada:

New XML Document

Steps

1. Choose File Type
- 2. Name and Location**
3. Select Document Type
4. ...

Name and Location

File Name:

Project:

Folder:

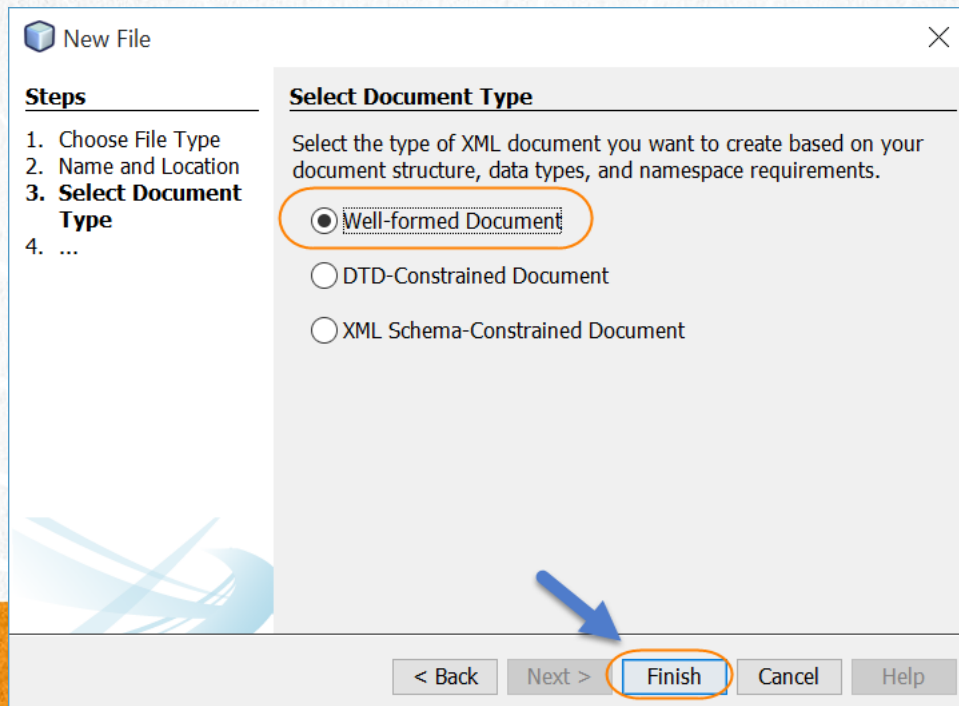
Created File:

CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

5. CREAR UN ARCHIVO XML

- En este paso seleccionamos cualquier opción, no es importante ya que vamos a sobrescribir el archivo:



PASO 6. MODIFICAMOS EL CÓDIGO

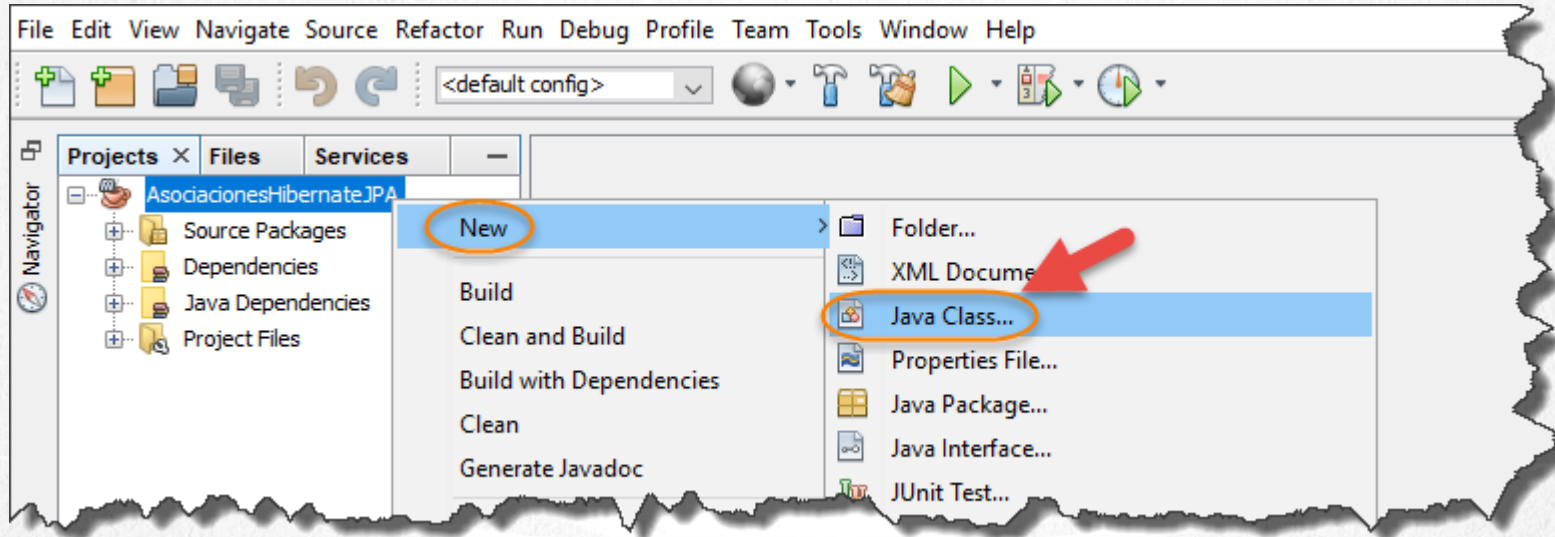
Archivo persistence.xml:

Clic para descargar código

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
  <persistence-unit name="HibernateJpaPU" transaction-type="RESOURCE_LOCAL">
    <provider>org.hibernate.jpa.HibernatePersistenceProvider</provider>
    <class>model.Rol</class>
    <class>model.Sucursal</class>
    <class>model.Curso</class>
    <class>model.Alumno</class>
    <class>model.Usuario</class>
    <class>model.Contacto</class>
    <class>model.Pago</class>
    <class>model.Domicilio</class>
    <class>model.Asignacion</class>
    <class>model.Instructor</class>
    <properties>
      <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/sga_db"/>
      <property name="javax.persistence.jdbc.user" value="root"/>
      <property name="javax.persistence.jdbc.password" value="admin"/>
      <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver"/>
    </properties>
  </persistence-unit>
</persistence>
```

PASO 7. CREACIÓN DEL CLASES DE ENTIDAD

Crear cada una de las clases de Entidad siguientes. El procedimiento para crear una clase es el mismo para todas:



CURSO DE HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 8. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File: `rsos\Hibernate\Leccion04\AsociacionesHibernateJPA\src\main\java\model\Domicilio.java`

PASO 9. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File: C:\Users\user\workspace\hibernate\Leccion04\AsociacionesHibernateJPA\src\main\java\model\Contacto.java

< Back Next > **Finish** Cancel Help

PASO 10. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

PASO 11. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 12. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 13. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

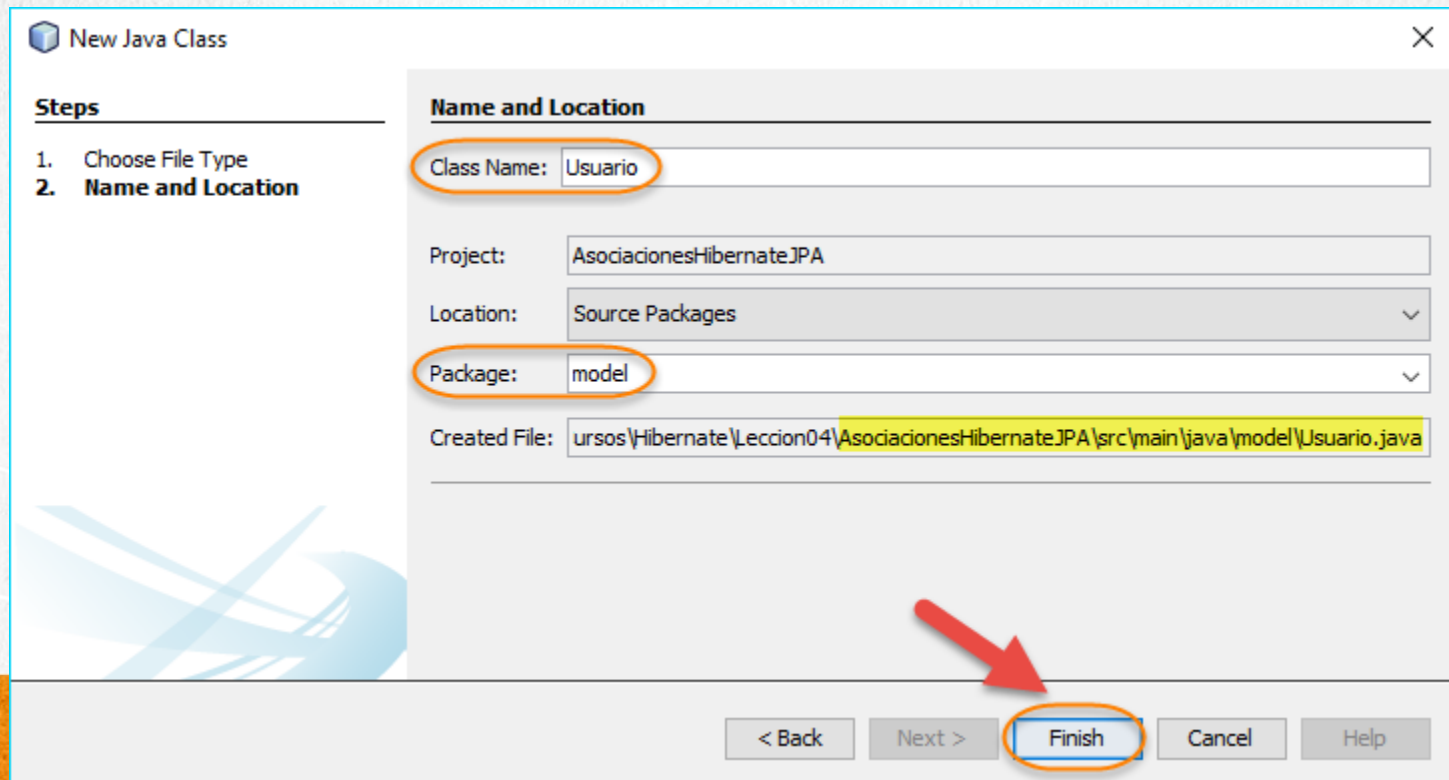
Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 14. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:



New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File: `ursosos\hibernate\Leccion04\AsociacionesHibernateJPA\src\main\java\model\Usuario.java`

< Back Next > **Finish** Cancel Help

PASO 15. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 16. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File: `sos\Hibernate\Leccion04\AsociacionesHibernateJPA\src\main\java\model\Instructor.java`

< Back Next > **Finish** Cancel Help

PASO 17. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 18. MODIFICAMOS EL CÓDIGO

Archivo Domicilio.java:

Clic para descargar código

```
@Entity
@Table(name = "domicilio")
@NamedQueries({
    @NamedQuery(name = "Domicilio.findAll", query = "SELECT d FROM Domicilio d")})
public class Domicilio implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id_domicilio")
    private Integer idDomicilio;

    @Basic(optional = false)
    @Column(name = "calle")
    private String calle;

    @Column(name = "no_externo")
    private String noExterno;
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 19. MODIFICAMOS EL CÓDIGO

Archivo Contacto.java:

Clic para descargar código

```
@Entity
@Table(name = "contacto")
@NamedQueries({
    @NamedQuery(name = "Contacto.findAll", query = "SELECT c FROM Contacto c")})
public class Contacto implements Serializable {
```

```
    private static final long serialVersionUID = 1L;
```

```
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id_contacto")
    private Integer idContacto;
```

```
    @Column(name = "telefono")
    private String telefono;
```

```
    @Column(name = "celular")
    private String celular;
```

```
    @Column(name = "oficina")
    private String oficina;
```

```
//Dar click en el link para ver el archivo completo y descargar el código
```

PASO 20. MODIFICAMOS EL CÓDIGO

Archivo Curso.java:

Clic para descargar código

```
@Entity
@Table(name = "curso")
@NamedQueries({
    @NamedQuery(name = "Curso.findAll", query = "SELECT c FROM Curso c")})
public class Curso implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id_curso")
    private Integer idCurso;

    @Basic(optional = false)
    @Column(name = "nombre")
    private String nombre;

    @Column(name = "horario")
    private String horario;
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 21. MODIFICAMOS EL CÓDIGO

Archivo Sucursal.java:

Clic para descargar código

```
@Entity
@Table(name = "sucursal")
@NamedQueries({
    @NamedQuery(name = "Sucursal.findAll", query = "SELECT s FROM Sucursal s")})
public class Sucursal implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id_sucursal")
    private Integer idSucursal;

    @Basic(optional = false)
    @Column(name = "nombre")
    private String nombre;

    @Basic(optional = false)
    @Column(name = "version")
    private int version;
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 22. MODIFICAMOS EL CÓDIGO

Archivo Pago.java:

Clic para descargar código

```
@Entity
@Table(name = "pago")
@NamedQueries({
    @NamedQuery(name = "Pago.findAll", query = "SELECT p FROM Pago p")})
public class Pago implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id_pago")
    private Integer idPago;

    @Column(name = "monto")
    private Float monto;

    @Column(name = "tipo_forma_pago")
    private String tipoFormaPago;
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 23. MODIFICAMOS EL CÓDIGO

Archivo Rol.java:

Clic para descargar código

```
@Entity
@Table(name = "rol")
@NamedQueries({
    @NamedQuery(name = "Rol.findAll", query = "SELECT r FROM Rol r")})
public class Rol implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id_rol")
    private Integer idRol;

    @Basic(optional = false)
    @Column(name = "nombre")
    private String nombre;

    @Basic(optional = false)
    @Column(name = "version")
    private int version;
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 24. MODIFICAMOS EL CÓDIGO

Archivo Usuario.java:

Clic para descargar código

```
@Entity
@Table(name = "usuario")
@NamedQueries({
    @NamedQuery(name = "Usuario.findAll", query = "SELECT u FROM Usuario u")})
public class Usuario implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id_usuario")
    private Integer idUsuario;

    @Basic(optional = false)
    @Column(name = "username")
    private String username;

    @Basic(optional = false)
    @Column(name = "password")
    private String password;
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 25. MODIFICAMOS EL CÓDIGO

Archivo Alumno.java:

Clic para descargar código

```
@Entity
@Table(name = "alumno")
@NamedQueries({
    @NamedQuery(name = "Alumno.findAll", query = "SELECT a FROM Alumno a")})
public class Alumno implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id_alumno")
    private Integer idAlumno;

    @Basic(optional = false)
    @Column(name = "nombre")
    private String nombre;

    @Basic(optional = false)
    @Column(name = "apellido_paterno")
    private String apellidoPaterno;
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 26. MODIFICAMOS EL CÓDIGO

[Archivo Instructor.java:](#)

Clic para descargar código

```
@Entity
@Table(name = "instructor")
@NamedQueries({
    @NamedQuery(name = "Instructor.findAll", query = "SELECT i FROM Instructor i")})
public class Instructor implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id_instructor")
    private Integer idInstructor;

    @Basic(optional = false)
    @Column(name = "nombre")
    private String nombre;

    @Basic(optional = false)
    @Column(name = "apellido_paterno")
    private String apellidoPaterno;
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 27. MODIFICAMOS EL CÓDIGO

[Archivo Asignacion.java:](#)

Clic para descargar código

```
@Entity
@Table(name = "asignacion")
@NamedQueries({
    @NamedQuery(name = "Asignacion.findAll", query = "SELECT a FROM Asignacion a")})
public class Asignacion implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id_asignacion")
    private Integer idAsignacion;

    @Column(name = "fecha_inicio")
    @Temporal(TemporalType.TIMESTAMP)
    private Date fechaInicio;

    @Column(name = "fecha_termino")
    @Temporal(TemporalType.TIMESTAMP)
    private Date fechaTermino;
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 28. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 29. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name: DomicilioDAO

Project: AsociacionesHibernateJPA

Location: Source Packages

Package: dao

Created File: os\Hibernate\Leccion04\AsociacionesHibernateJPA\src\main\java\dao\DomicilioDAO.java

< Back Next > **Finish** Cancel Help

PASO 30. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File: `is\Hibernate\Leccion04\AsociacionesHibernateJPA\src\main\java\dao\ContactoDAO.java`

PASO 31. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 32. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name: SucursalDAO

Project: AsociacionesHibernateJPA

Location: Source Packages

Package: dao

Created File: os\Hibernate\Leccion04\AsociacionesHibernateJPA\src\main\java\dao\SucursalDAO.java

< Back Next > **Finish** Cancel Help

PASO 33. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 34. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

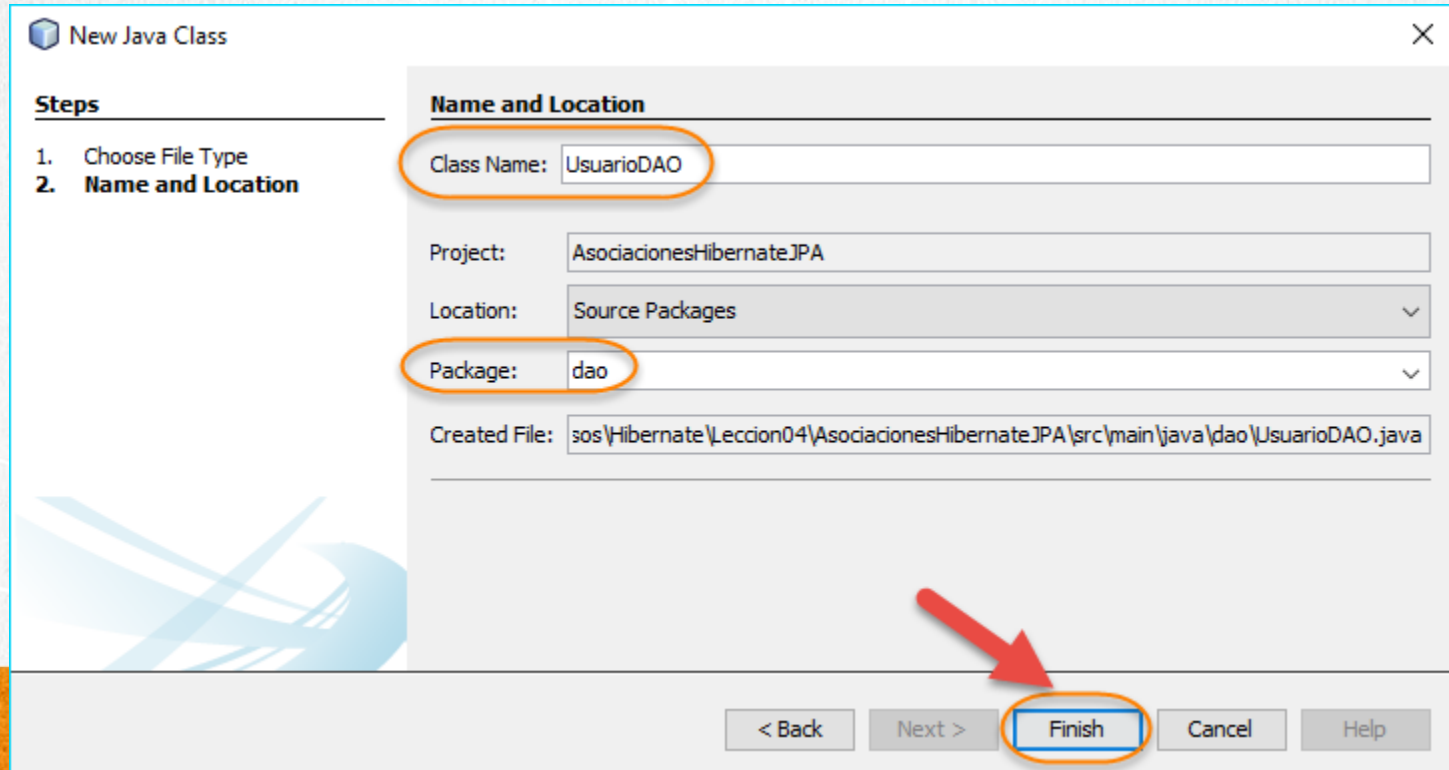
Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 35. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:



New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

PASO 36. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 37. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 38. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 39. AGREGAMOS UNA CLASE

Agregamos una clase al proyecto:

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 40. MODIFICAMOS EL CÓDIGO

Archivo GenericDAO.java:

Clic para descargar código

```
public abstract class GenericDAO {

    protected static EntityManager em;
    private static EntityManagerFactory emf;
    private static final String PERSISTENCE_UNIT_NAME = "HibernateJpaPU";

    public GenericDAO() {
        if (emf == null) {
            emf = Persistence.createEntityManagerFactory(PERSISTENCE_UNIT_NAME);
        }
    }

    protected EntityManager getEntityManager() {
        if (em == null) {
            em = emf.createEntityManager();
        }
        return em;
    }
}
```

PASO 41. MODIFICAMOS EL CÓDIGO

[Archivo DomicilioDAO.java:](#)

Clic para descargar código

```
public class DomicilioDAO extends GenericDAO{
    public void listar() {
        // Consulta a ejecutar
        // No necesitamos crear una nueva transaccion
        String hql = "SELECT d FROM Domicilio d";
        em = getEntityManager();
        Query query = em.createQuery(hql);
        List<Domicilio> list = query.getResultList();
        for (Domicilio domicilio : list) {
            System.out.println(domicilio);
        }
    }
}
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 42. MODIFICAMOS EL CÓDIGO

Archivo ContactoDAO.java:

Clic para descargar código

```
public class ContactoDAO extends GenericDAO{
    public void listar() {
        // Consulta a ejecutar
        // No necesitamos crear una nueva transaccion
        String hql = "SELECT c FROM Contacto c";
        em = getEntityManager();
        Query query = em.createQuery(hql);
        List<Contacto> list = query.getResultList();
        for (Contacto c : list) {
            System.out.println(c);
        }
    }
}
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 43. MODIFICAMOS EL CÓDIGO

Archivo CursoDAO.java:

Clic para descargar código

```
public class CursoDAO extends GenericDAO{
    public void listar() {
        // Consulta a ejecutar
        // No necesitamos crear una nueva transaccion
        String hql = "SELECT c FROM Curso c";
        em = getEntityManager();
        Query query = em.createQuery(hql);
        List<Curso> list = query.getResultList();
        for (Curso c : list) {
            System.out.println(c);
        }
    }
}
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 44. MODIFICAMOS EL CÓDIGO

Archivo SucursalDAO.java:

Clic para descargar código

```
public class SucursalDAO extends GenericDAO{
    public void listar() {
        // Consulta a ejecutar
        // No necesitamos crear una nueva transaccion
        String hql = "SELECT s FROM Sucursal s";
        em = getEntityManager();
        Query query = em.createQuery(hql);
        List<Sucursal> list = query.getResultList();
        for (Sucursal sucursal : list) {
            System.out.println(sucursal);
        }
    }
}
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 45. MODIFICAMOS EL CÓDIGO

Archivo PagoDAO.java:

Clic para descargar código

```
public class PagoDAO extends GenericDAO{

    public void listar() {
        // Consulta a ejecutar
        // No necesitamos crear una nueva transaccion
        String hql = "SELECT p FROM Pago p";
        em = getEntityManager();
        Query query = em.createQuery(hql);
        List<Pago> list = query.getResultList();
        for (Pago pago : list) {
            System.out.println(pago);
        }
    }
}
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 46. MODIFICAMOS EL CÓDIGO

Archivo RolDAO.java:

Clic para descargar código

```
public class RolDAO extends GenericDAO{
    public void listar() {
        // Consulta a ejecutar
        // No necesitamos crear una nueva transaccion
        String hql = "SELECT r FROM Rol r";
        em = getEntityManager();
        Query query = em.createQuery(hql);
        List<Rol> list = query.getResultList();
        for (Rol rol : list) {
            System.out.println(rol);
        }
    }
}
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 47. MODIFICAMOS EL CÓDIGO

Archivo UsuarioDAO.java:

Clic para descargar código

```
public class UsuarioDAO extends GenericDAO{
    public void listar() {
        // Consulta a ejecutar
        // No necesitamos crear una nueva transaccion
        String hql = "SELECT s FROM Usuario s";
        em = getEntityManager();
        Query query = em.createQuery(hql);
        List<Usuario> list = query.getResultList();
        for (Usuario usuario : list) {
            System.out.println(usuario);
        }
    }
}
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 48. MODIFICAMOS EL CÓDIGO

Archivo AlumnoDAO.java:

Clic para descargar código

```
public class AlumnoDAO extends GenericDAO {
```

```
    public void listar() {  
        // Consulta a ejecutar  
        // No necesitamos crear una nueva transaccion  
        String hql = "SELECT a FROM Alumno a";  
        em = getEntityManager();  
        Query query = em.createQuery(hql);  
        List<Alumno> list = query.getResultList();  
        for (Alumno a : list) {  
            System.out.println(a);  
        }  
    }  
}
```

//Dar click en el link para ver el archivo completo y descargar el código

CURSO DE HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 49. MODIFICAMOS EL CÓDIGO

[Archivo AsignacionDAO.java:](#)

Clic para descargar código

```
public class AsignacionDAO extends GenericDAO{
    public void listar() {
        // Consulta a ejecutar
        // No necesitamos crear una nueva transaccion
        String hql = "SELECT a FROM Asignacion a";
        em = getEntityManager();
        Query query = em.createQuery(hql);
        List<Asignacion> list = query.getResultList();
        for (Asignacion a : list) {
            System.out.println(a);
        }
    }
}
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 50. MODIFICAMOS EL CÓDIGO

Archivo InstructorDAO.java:

Clic para descargar código

```
public class InstructorDAO extends GenericDAO{
    public void listar() {
        // Consulta a ejecutar
        // No necesitamos crear una nueva transaccion
        String hql = "SELECT i FROM Instructor i";
        em = getEntityManager();
        Query query = em.createQuery(hql);
        List<Instructor> list = query.getResultList();
        for (Instructor instructor : list) {
            System.out.println(instructor);
        }
    }
}
```

//Dar click en el link para ver el archivo completo y descargar el código

PASO 51. MODIFICAMOS EL CÓDIGO

Archivo TestDAOs.java:

Clic para descargar código

```
public class TestDAOs {
```

```
    public static void main(String[] args) {  
        AlumnoDAO alumnoDao = new AlumnoDAO();  
        alumnoDao.listar();
```

```
  
        AsignacionDAO asignacionDao = new AsignacionDAO();  
        asignacionDao.listar();
```

```
  
        ContactoDAO contactoDao = new ContactoDAO();  
        contactoDao.listar();
```

```
  
        CursoDAO cursoDao = new CursoDAO();  
        cursoDao.listar();
```

```
  
        DomicilioDAO domicilioDao = new DomicilioDAO();  
        domicilioDao.listar();
```

```
  
        InstructorDAO instructorDao = new InstructorDAO();  
        instructorDao.listar();
```

```
  
        //Dar click en el link para ver el archivo completo y descargar el código
```

52. CREAR UN ARCHIVO XML

Creamos un archivo log4j2.xml. El API de log4j nos permite manejar el log o bitácora de una aplicación Java de manera más simple.

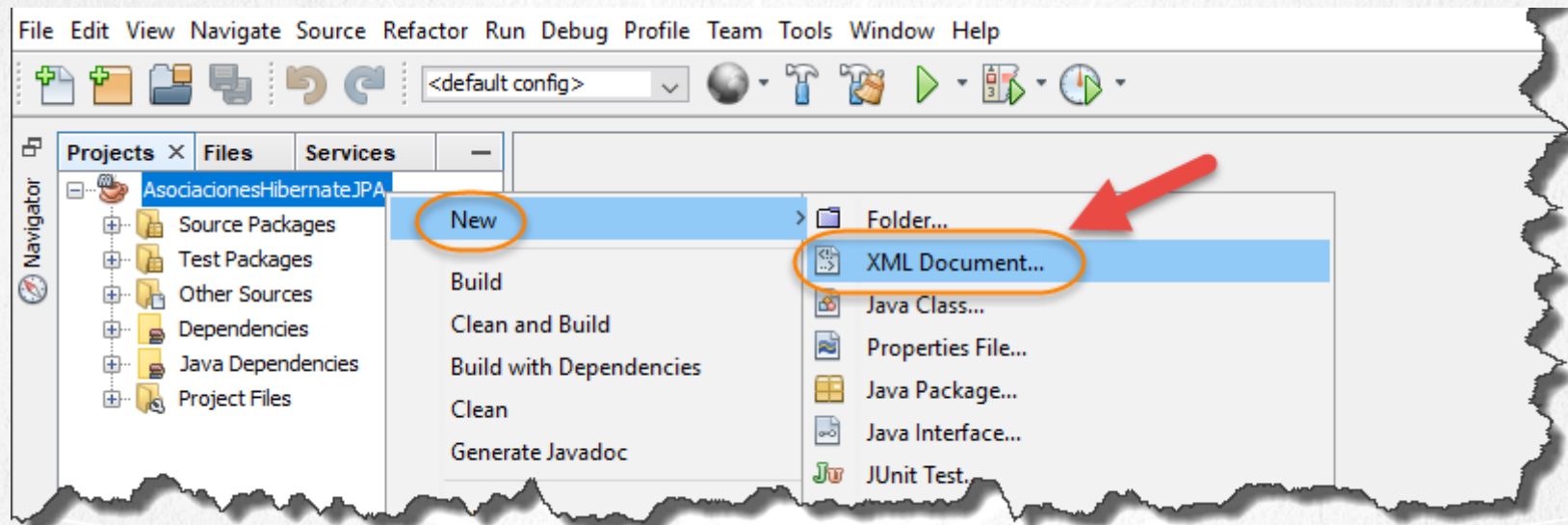
Para poder utilizar este API solo se requiere agregar las librerías de log4j las cuales ya han sido agregadas al archivo pom.xml de maven, y el archivo log4j2.xml en algún lugar que reconozca el classpath, por ejemplo en la carpeta de resources del proyecto.

Con esto estaremos listos para especificar que información queremos que se envíe a la consola u otros lugares, como un archivo. Para más información de esta API consultar:

<https://logging.apache.org/log4j/2.x/>

52. CREAR UN ARCHIVO XML

- Creamos el archivo log4j2.xml:



52. CREAR UN ARCHIVO XML

- Creamos el archivo log4j2.xml:

New XML Document

Steps

1. Choose File Type
- 2. Name and Location**
3. Select Document Type
4. ...

Name and Location

File Name:

Project:

Folder:

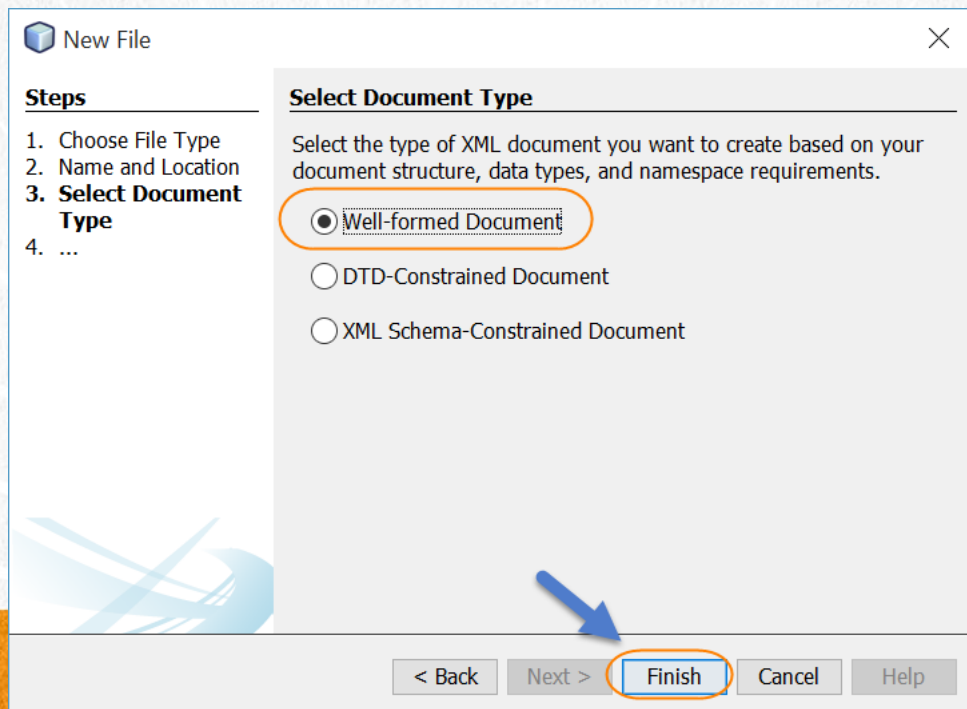
Created File:

CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

52. CREAR UN ARCHIVO XML

- Creamos el archivo log4j2.xml. En este paso seleccionamos cualquier opción, no es importante ya que vamos a sobrescribir el archivo:



PASO 53. MODIFICAMOS EL CÓDIGO

Archivo log4j2.xml:

Clic para ver el
archivo

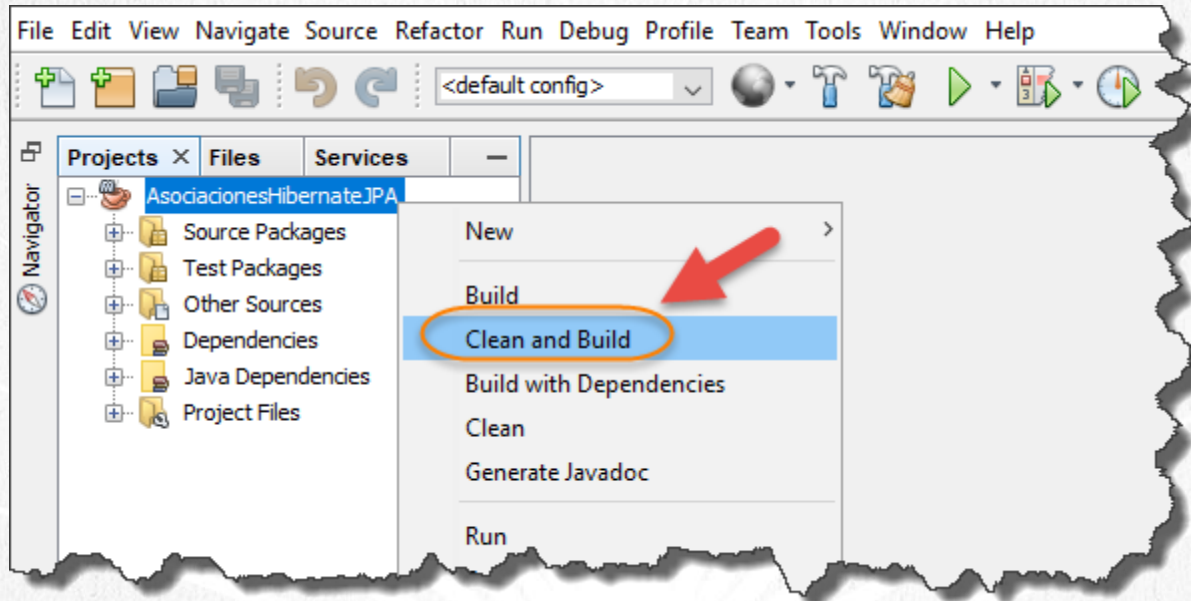
```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="INFO">
  <Appenders>
    <Console name="Console" target="SYSTEM_OUT">
      <PatternLayout pattern="%d{HH:mm:ss} [%t] %-5level %logger{36} - %msg%n" />
    </Console>
  </Appenders>
  <Loggers>
    <Logger name="org.hibernate.SQL" level="debug" additivity="false">
      <AppenderRef ref="Console"/>
    </Logger>
    <logger name="org.hibernate.type.descriptor.sql.BasicBinder" level="trace" additivity="false">
      <AppenderRef ref="Console"/>
    </logger>
    <Root level="info">
      <AppenderRef ref="Console" />
    </Root>
  </Loggers>
</Configuration>
```

CURSO DE JAVA CON JDBC

www.globalmentoring.com.mx

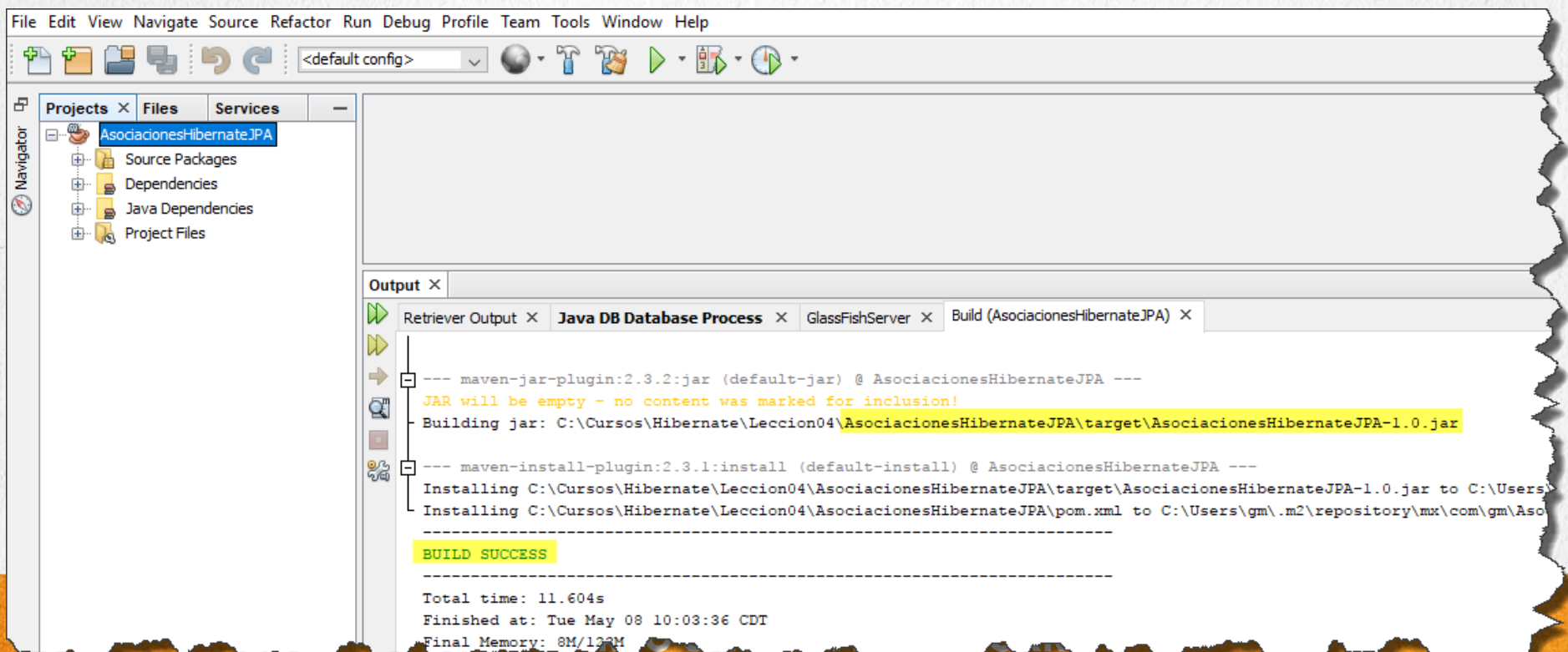
54. HACEMOS CLEAN & BUILD

- Para que tener la última versión de cada archivo, ejecutamos un Clean & Build:



54. HACEMOS CLEAN & BUILD

- Una vez terminado el proceso se debe mostrar una salida similar a la siguiente:



The screenshot shows an IDE window with the 'Output' tab selected. The 'Output' tab contains several sub-tabs, with 'Build (AsociacionesHibernateJPA)' being the active one. The output text shows the Maven build process, including the execution of the maven-jar-plugin and maven-install-plugin. The build is successful, and the output ends with 'BUILD SUCCESS'.

```
Retriever Output x Java DB Database Process x GlassFishServer x Build (AsociacionesHibernateJPA) x

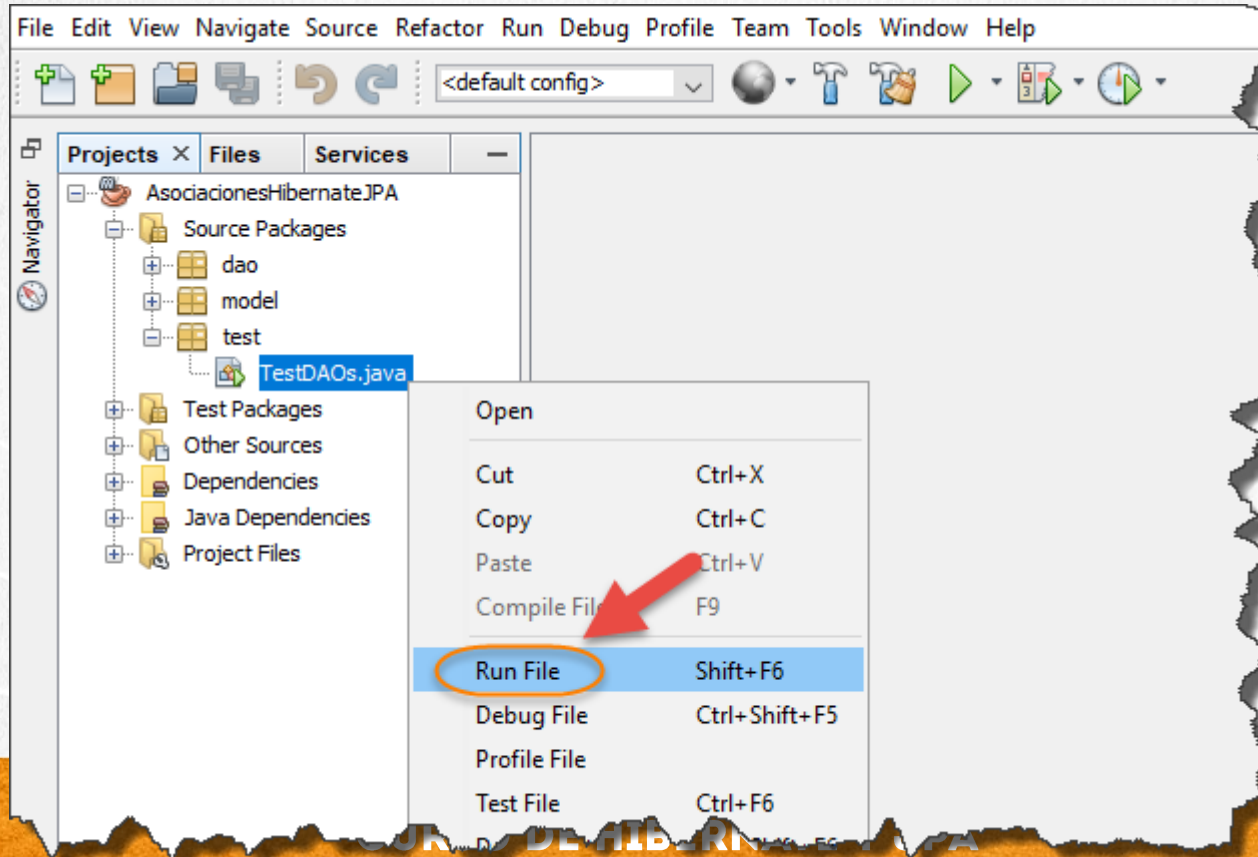
--- maven-jar-plugin:2.3.2:jar (default-jar) @ AsociacionesHibernateJPA ---
JAR will be empty - no content was marked for inclusion!
Building jar: C:\Cursos\Hibernate\Leccion04\AsociacionesHibernateJPA\target\AsociacionesHibernateJPA-1.0.jar

--- maven-install-plugin:2.3.1:install (default-install) @ AsociacionesHibernateJPA ---
Installing C:\Cursos\Hibernate\Leccion04\AsociacionesHibernateJPA\target\AsociacionesHibernateJPA-1.0.jar to C:\Users\
Installing C:\Cursos\Hibernate\Leccion04\AsociacionesHibernateJPA\pom.xml to C:\Users\gm\.m2\repository\mx\com\gm\AsociacionesHibernateJPA\pom.xml

BUILD SUCCESS

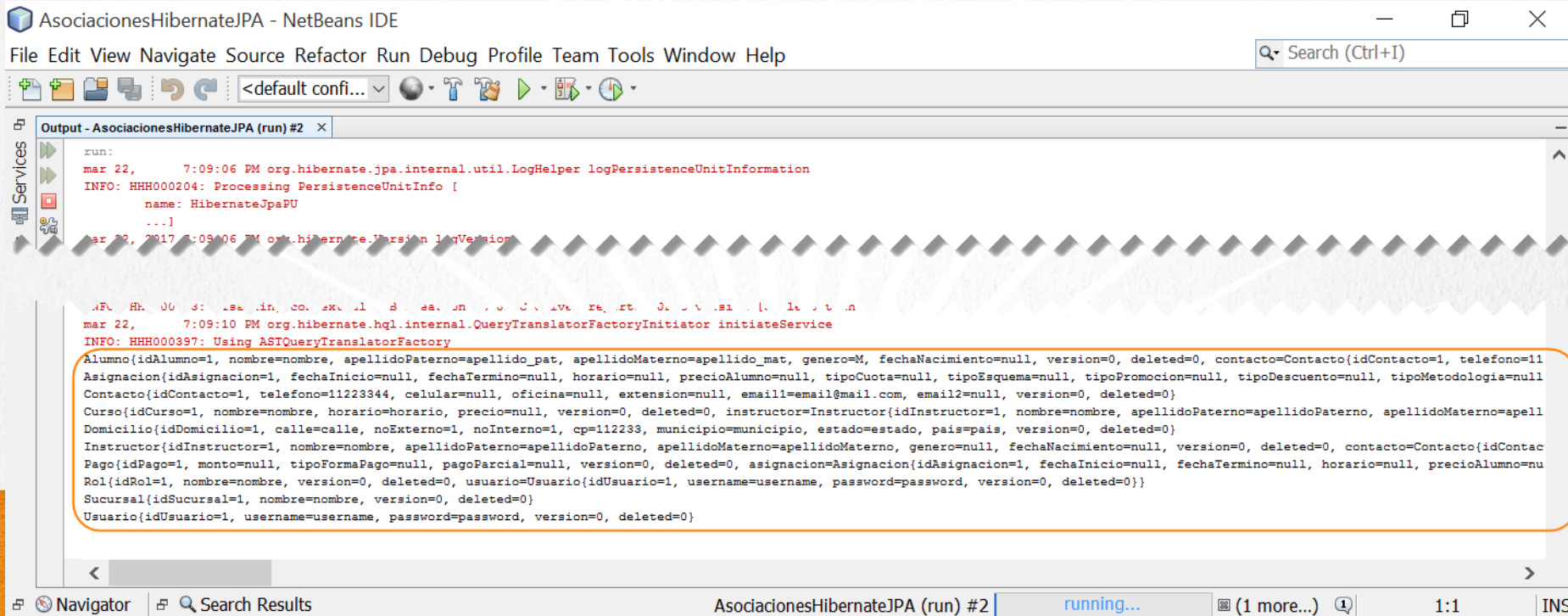
Total time: 11.604s
Finished at: Tue May 08 10:03:36 CDT
Final Memory: 8M/123M
```


PASO 55. EJECUTAMOS EL PROYECTO



PASO 55. EJECUTAMOS EL PROYECTO

- Observamos el siguiente resultado de ejecutar una consulta con cada uno de los DAO's creados para cada tabla. El resultado puede variar según los datos que se tengan en la base de datos:



```
run:
mar 22, 7:09:06 PM org.hibernate.jpa.internal.util.LogHelper logPersistenceUnitInformation
INFO: HHH000204: Processing PersistenceUnitInfo [
    name: HibernateJpaPU
    ...]
mar 22, 2017 7:09:06 PM org.hibernate.Version logVersion
INFO: HHH000397: Using ASTQueryTranslatorFactory

mar 22, 7:09:10 PM org.hibernate.hql.internal.QueryTranslatorFactoryInitiator initiateService
INFO: HHH000397: Using ASTQueryTranslatorFactory

Alumno{idAlumno=1, nombre=nombre, apellidoPaterno=apellido_pat, apellidoMaterno=apellido_mat, genero=M, fechaNacimiento=null, version=0, deleted=0, contacto=Contacto{idContacto=1, telefono=11
Asignacion{idAsignacion=1, fechaInicio=null, fechaTermino=null, horario=null, precioAlumno=null, tipoCuota=null, tipoEsquema=null, tipoPromocion=null, tipoDescuento=null, tipoMetodologia=null
Contacto{idContacto=1, telefono=11223344, celular=null, oficina=null, extension=null, email1=email@mail.com, email2=null, version=0, deleted=0}
Curso{idCurso=1, nombre=nombre, horario=horario, precio=null, version=0, deleted=0, instructor=Instructor{idInstructor=1, nombre=nombre, apellidoPaterno=apellidoPaterno, apellidoMaterno=apell
Domicilio{idDomicilio=1, calle=calle, noExterno=1, noInterno=1, cp=112233, municipio=municipio, estado=estado, pais=pais, version=0, deleted=0}
Instructor{idInstructor=1, nombre=nombre, apellidoPaterno=apellidoPaterno, apellidoMaterno=apellidoMaterno, genero=null, fechaNacimiento=null, version=0, deleted=0, contacto=Contacto{idContac
Pago{idPago=1, monto=null, tipoFormaPago=null, pagoParcial=null, version=0, deleted=0, asignacion=Asignacion{idAsignacion=1, fechaInicio=null, fechaTermino=null, horario=null, precioAlumno=nu
Rol{idRol=1, nombre=nombre, version=0, deleted=0, usuario=Usuario{idUsuario=1, username=username, password=password, version=0, deleted=0}}
Sucursal{idSucursal=1, nombre=nombre, version=0, deleted=0}
Usuario{idUsuario=1, username=username, password=password, version=0, deleted=0}
```

CONCLUSIÓN DEL EJERCICIO

Como este ejercicio hemos creado la base de nuestro proyecto que estaremos utilizando a lo largo del curso.

Creamos cada una de las clases de Entidad, con sus relaciones incluidas según el esquema Entidad – Relación descrito anteriormente.

Además las clases DAO base que nos permitirán interactuar con cada una de las clases de Entidad utilizando Hibernate/JPA.



Experiencia y Conocimiento para tu vida

CURSO DE HIBERNATE Y JPA

www.globalmentoring.com.mx

CURSO ONLINE

HIBERNATE & JPA

Por: Ing. Ubaldo Acosta



Experiencia y Conocimiento para tu vida

CURSO DE HIBERNATE Y JPA

www.globalmentoring.com.mx