

CURSO HIBERNATE Y JPA

EJERCICIO

APLICACIÓN WEB CON HIBERNATE Y JPA



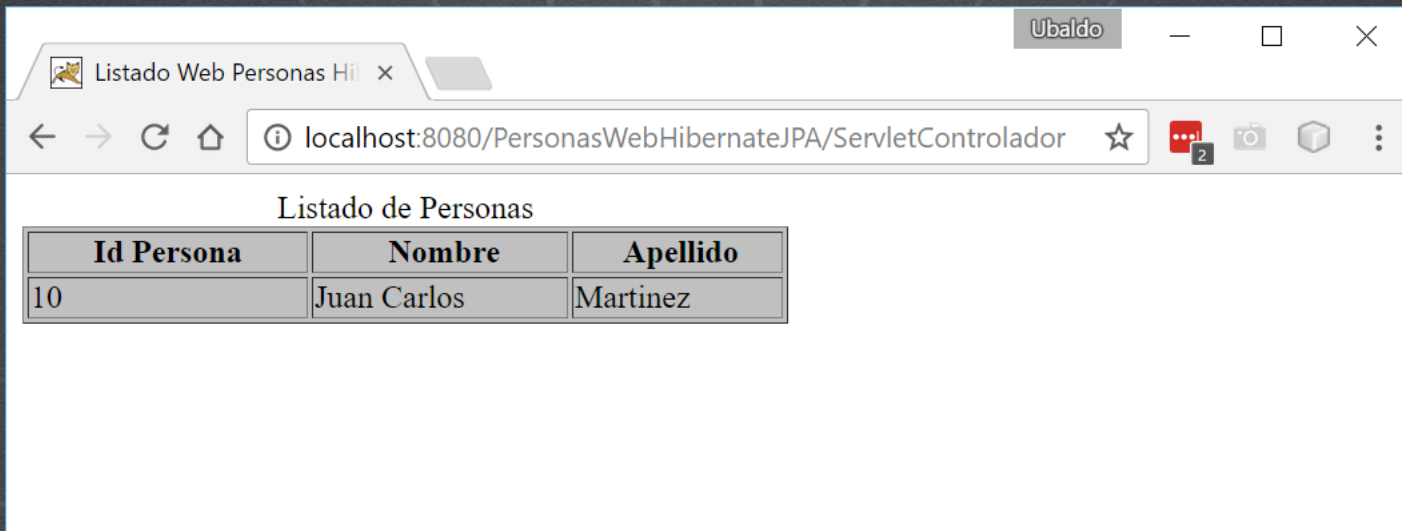
Experiencia y Conocimiento para tu vida

CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

OBJETIVO DEL EJERCICIO

Crear una aplicación Web e integrarla con Hibernate y JPA. El resultado final es como sigue:

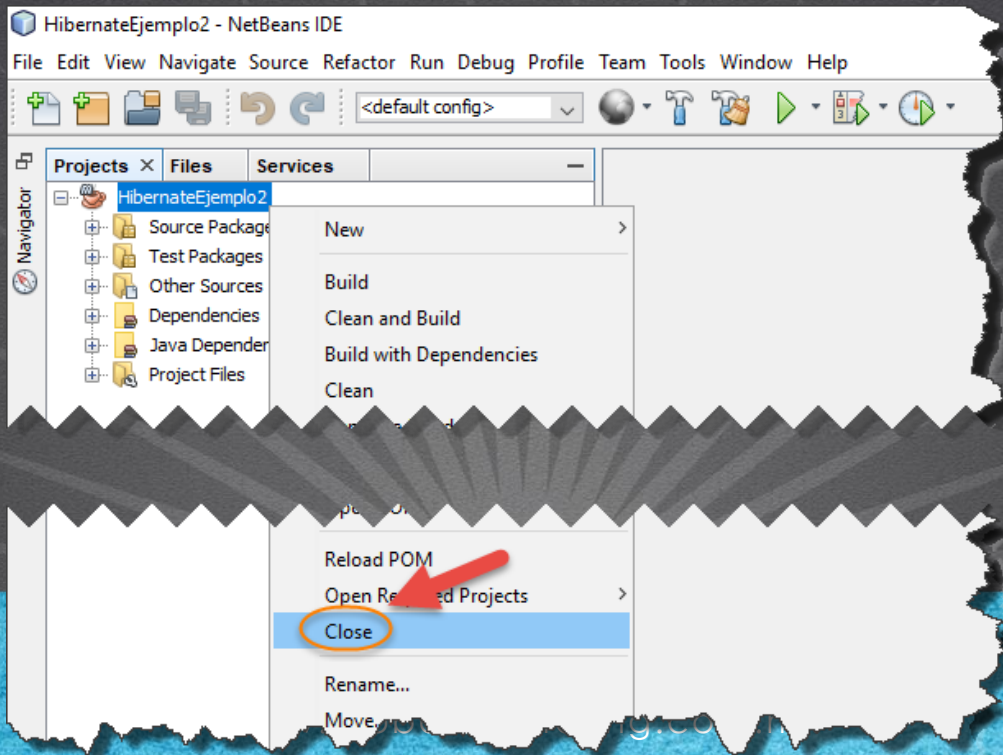


CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

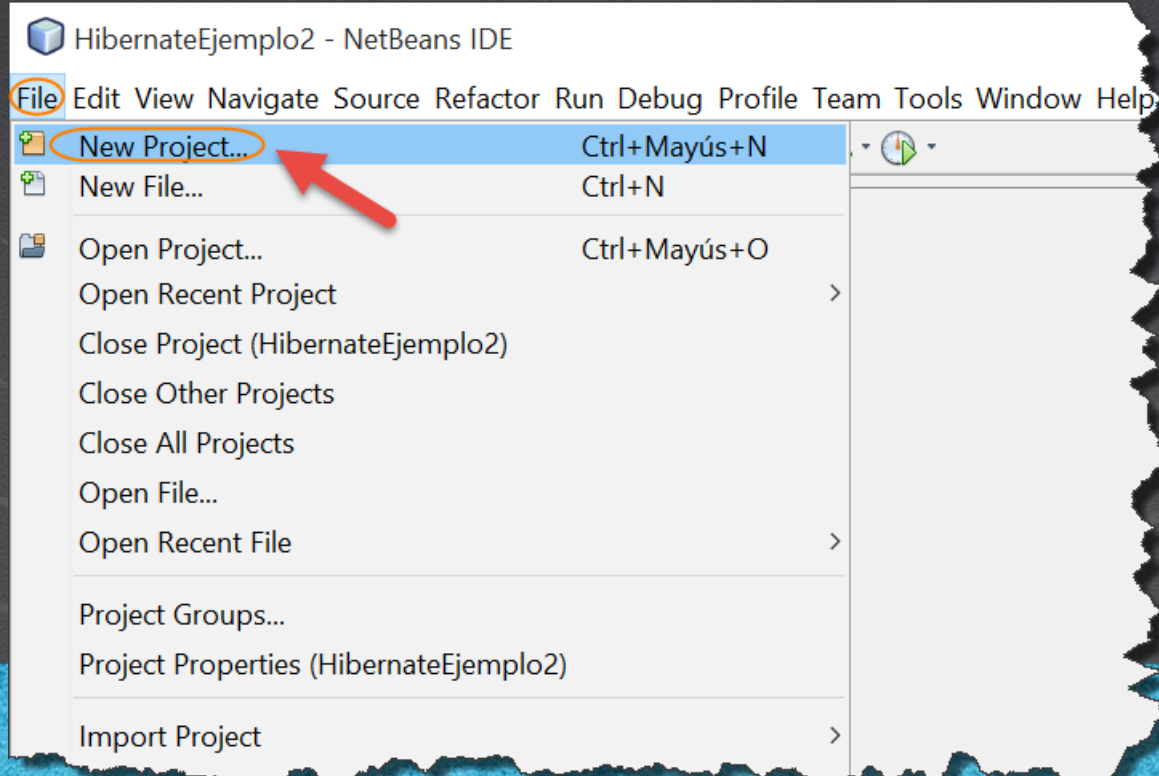
CERRAMOS CUALQUIER OTRO PROYECTO

Cerramos cualquier otro proyecto abierto, para solo trabajar con el proyecto que vamos a crear:



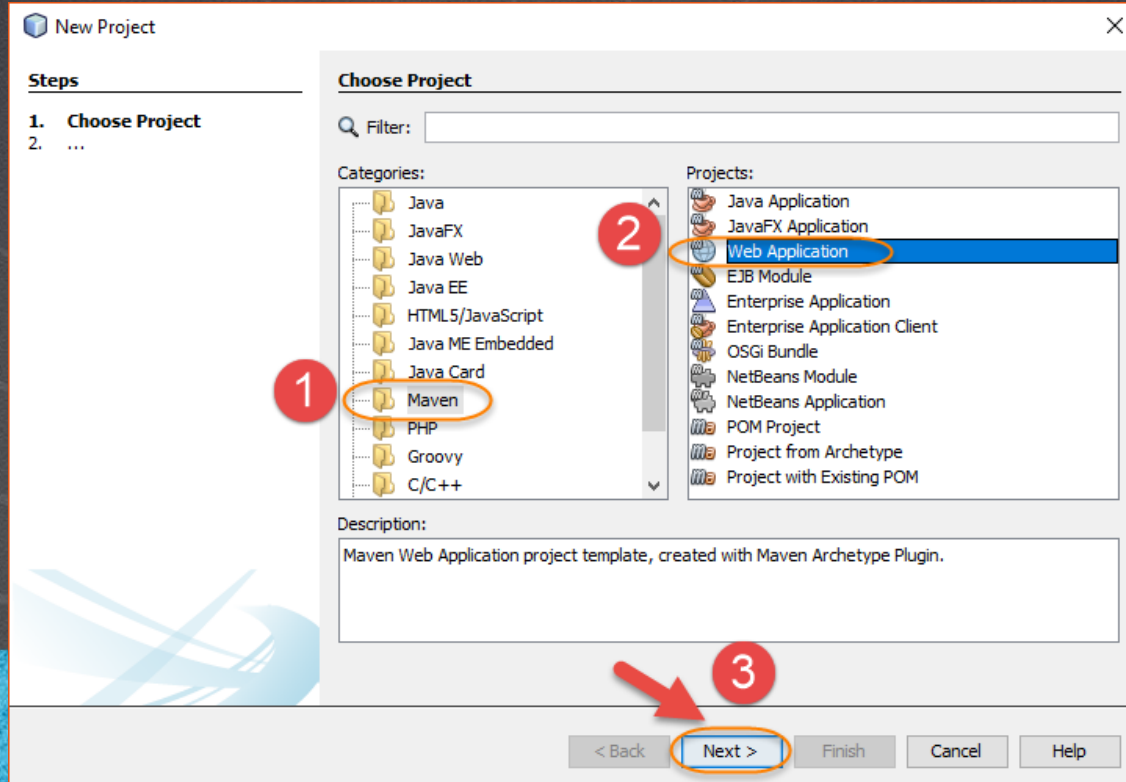
PASO 1. CREACIÓN DEL PROYECTO

Crear un nuevo proyecto Web usando Maven:



PASO 1. CREACIÓN DEL PROYECTO

Crear un nuevo proyecto Web usando Maven:



PASO 1. CREACIÓN DEL PROYECTO

Crear un nuevo proyecto Web usando Maven:

New Web Application

Steps

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

Name and Location

Project Name: PersonasWebHibernateJPA

Project Location: C:\Cursos\Hibernate\Leccion03 Browse...

Project Folder: C:\Cursos\Hibernate\Leccion03\PersonasWebHibernateJPA

Artifact Id: PersonasWebHibernateJPA

Group Id: mx.com.gm

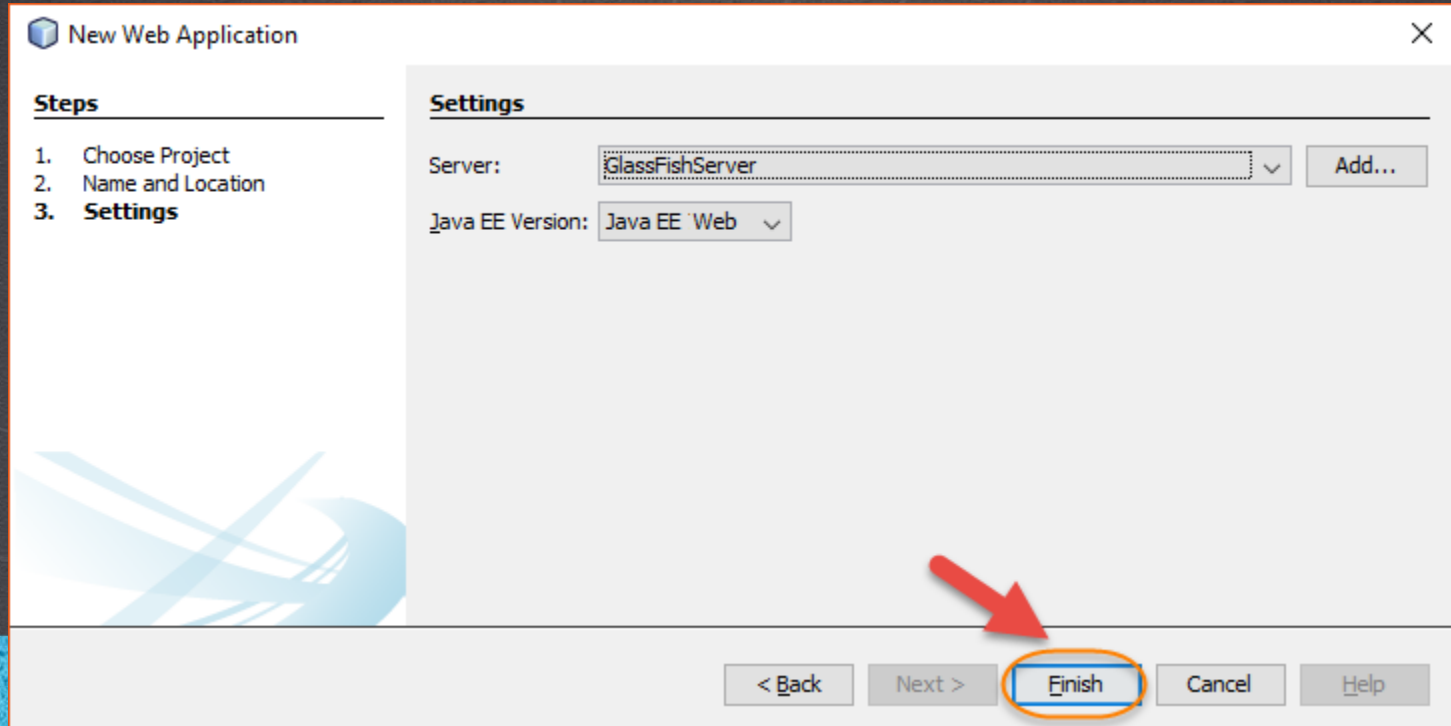
Version: 1.0

Package: (Optional)

< Back **Next >** Finish Cancel Help

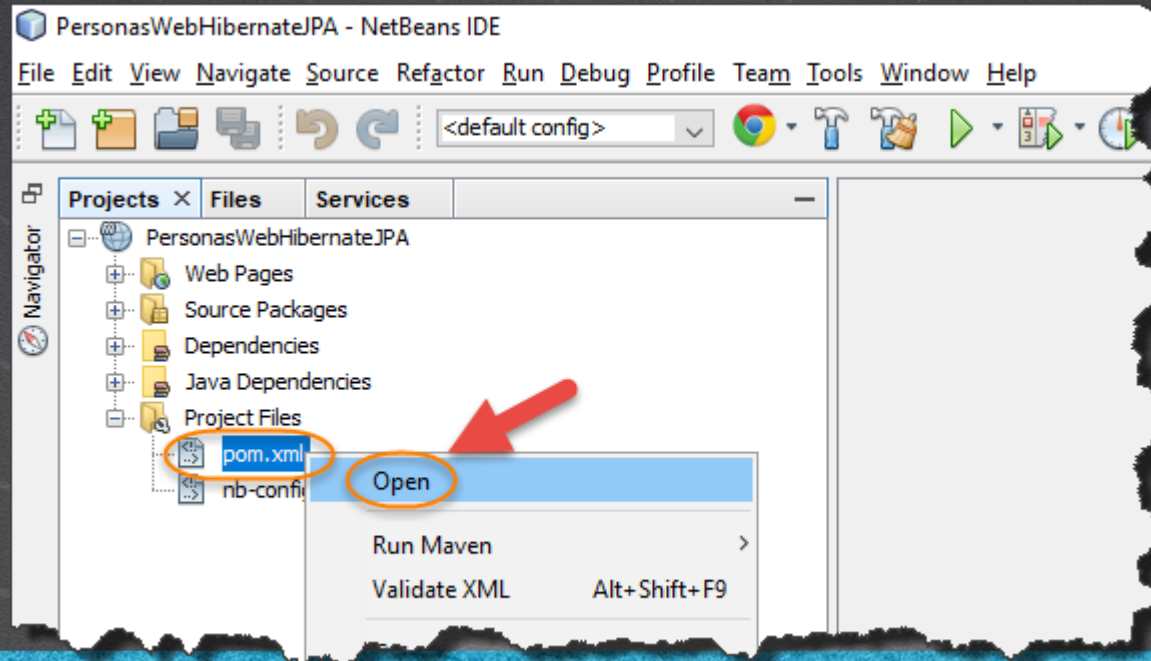
PASO 1. CREACIÓN DEL PROYECTO

Crear un nuevo proyecto Web:



2. ABRIMOS EL ARCHIVO POM.XML DE MAVEN

- El archivo pom.xml de maven administra las librerías Java vamos a utilizar:

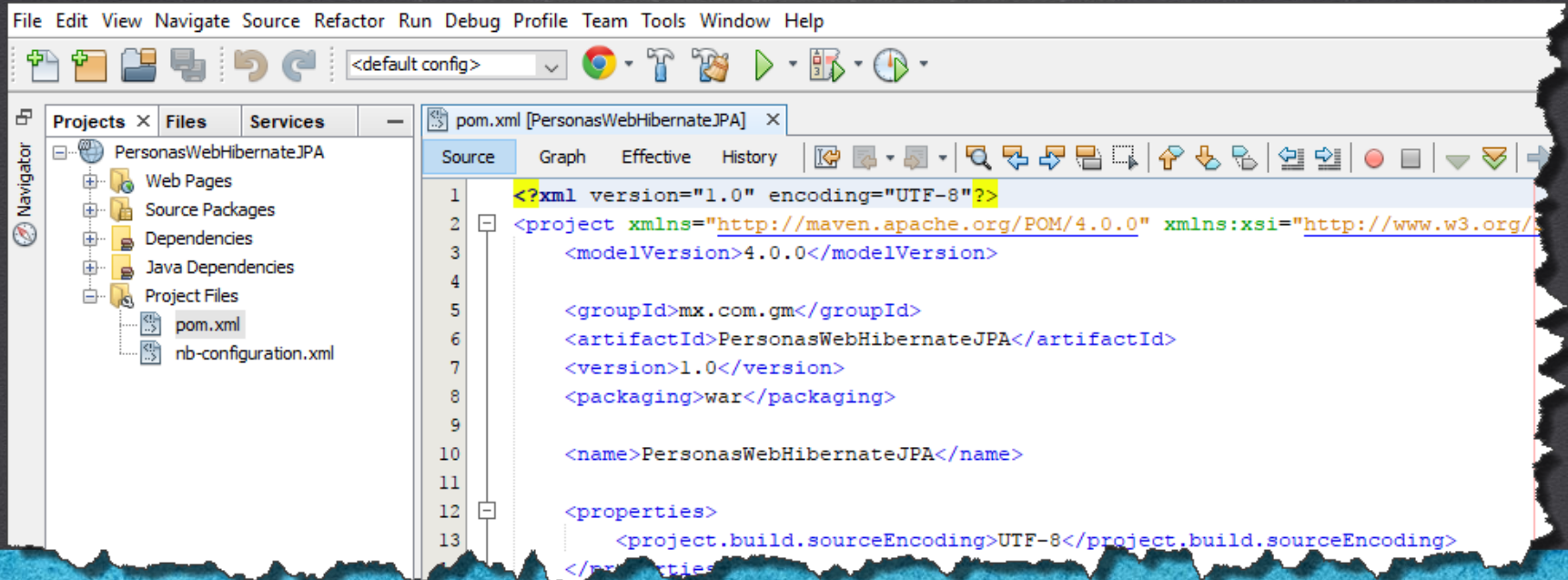


CURSO STRUTS FRAMEWORK

www.globalmentoring.com.mx

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:



CURSO STRUTS FRAMEWORK

www.globalmentoring.com.mx

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>PersonasWebHibernateJPA</artifactId>
  <version>1.0</version>
  <packaging>war</packaging>

  <name>PersonasWebHibernateJPA</name>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>

  <dependencies>
    <dependency>
      <groupId>javax</groupId>
      <artifactId>javaee-web-api</artifactId>
      <version>8.0</version>
      <scope>provided</scope>
    </dependency>
    <dependency>
      <groupId>org.hibernate</groupId>
      <artifactId>hibernate-core</artifactId>
      <version>5.3.0.CR2</version>
    </dependency>
```

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>
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>5.1.42</version>
</dependency>
</dependencies>
```


PASO 3. MODIFICAMOS EL CÓDIGO

Archivo pom.xml:

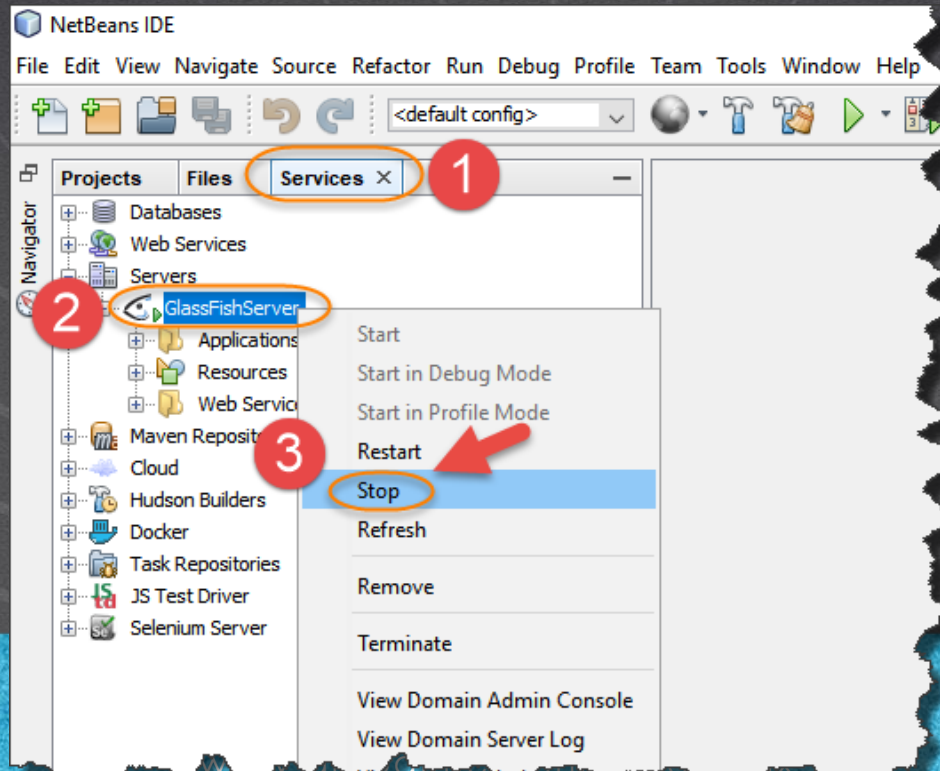
Clic para ver el
archivo

```
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-war-plugin</artifactId>
      <version>2.3</version>
      <configuration>
        <failOnMissingWebXml>>false</failOnMissingWebXml>
      </configuration>
    </plugin>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.7.0</version>
      <configuration>
        <source>1.8</source>
        <target>1.8</target>
      </configuration>
    </plugin>
  </plugins>
</build>

</project>
```

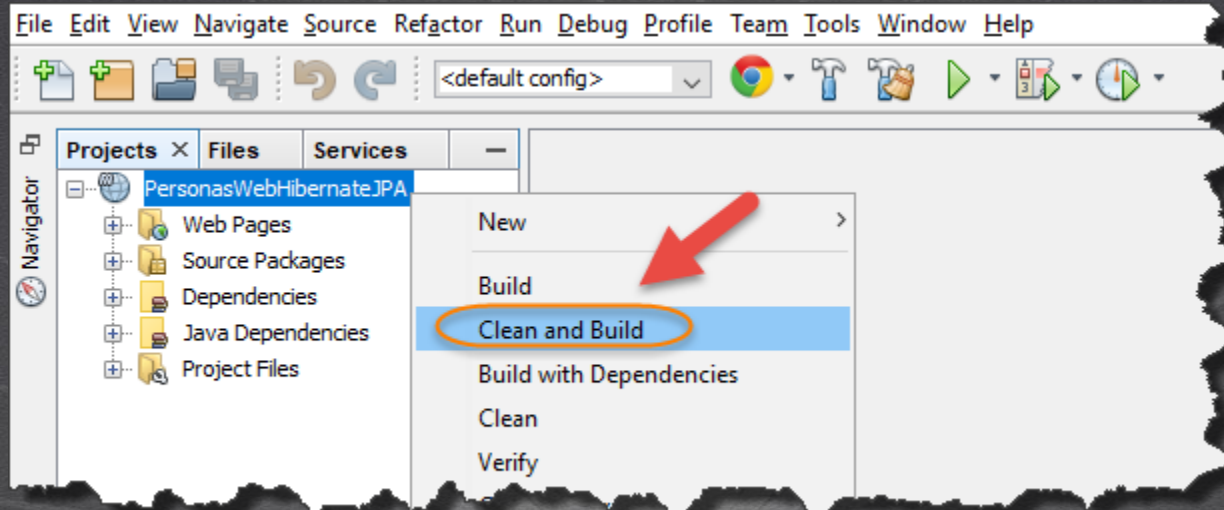
4. DETENEMOS GLASSFISH SI ESTUVIERA INICIADO

- Antes de hacer Clean & Build del proyecto para que descargue las nuevas librerías, verificamos que el servidor de Glassfish no esté iniciado ya que puede haber problemas para hacer el proceso de Clean & build si el servidor está iniciado. Este paso solo es de verificación:



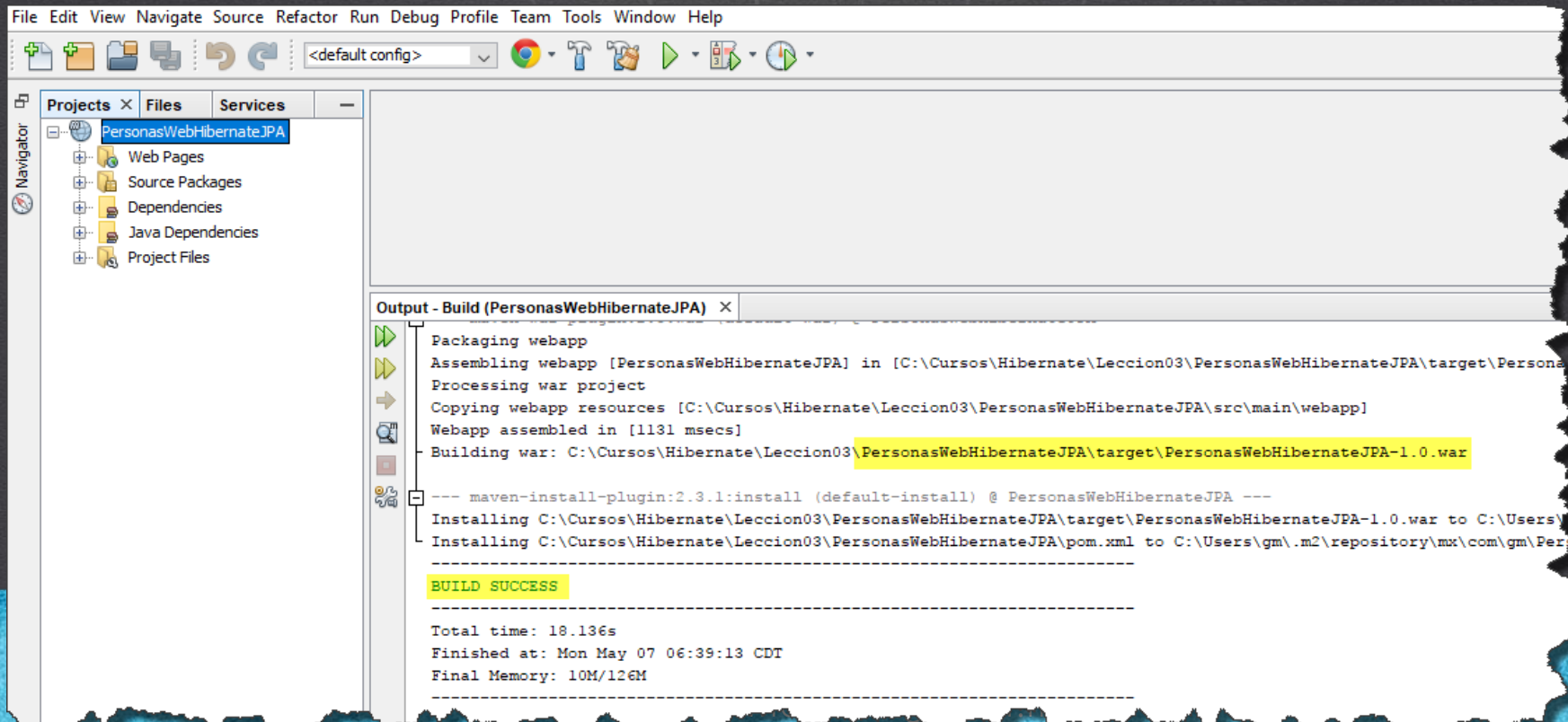
5. HACEMOS CLEAN & BUILD

•Regresamos a la vista de Proyectos. Para que se descarguen las nuevas 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:



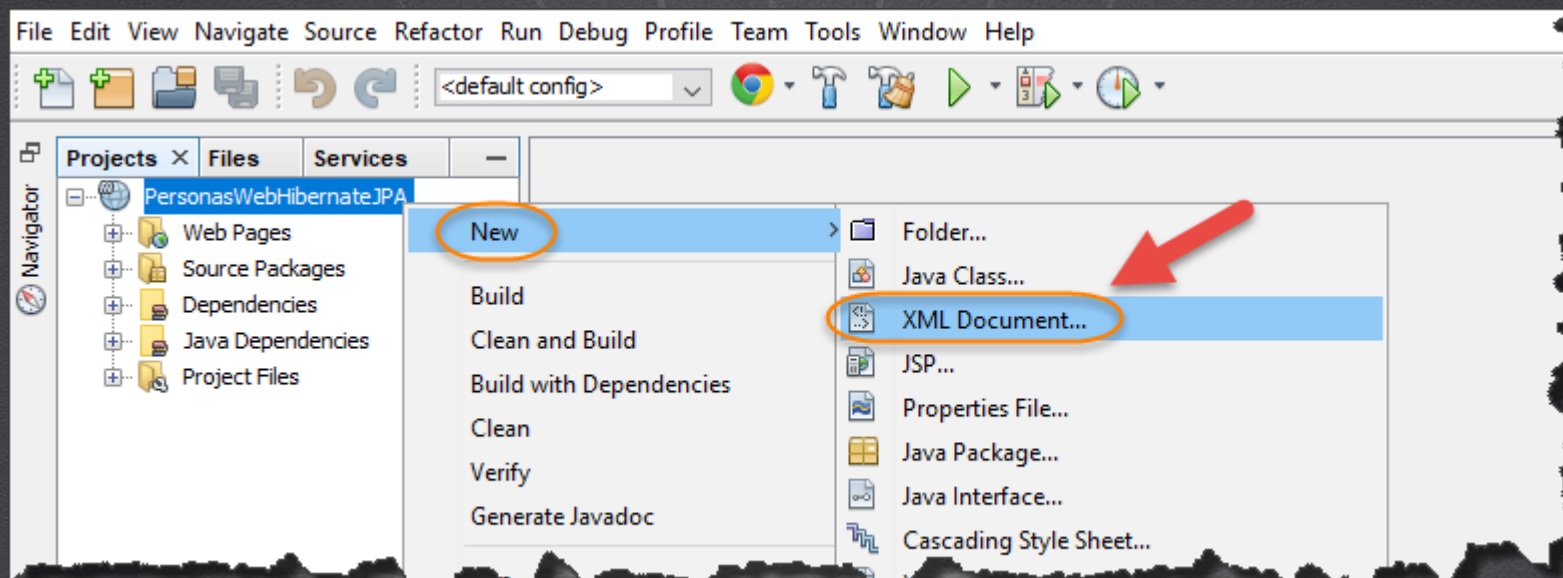
5. HACEMOS CLEAN & BUILD

- Si ya no tuvo que descargar ninguna librería debido a que podría ya tener todas descargadas el proceso es más rápido. Al final deberemos observar lo siguiente:



PASO 6. CREAR UN ARCHIVO XML

Creamos el archivo persistence.xml



CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 6. CREAR UN ARCHIVO XML

Creamos el archivo persistence.xml

New XML Document

Steps

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

Name and Location

File Name: persistence.xml

Project: PersonasWebHibernateJPA

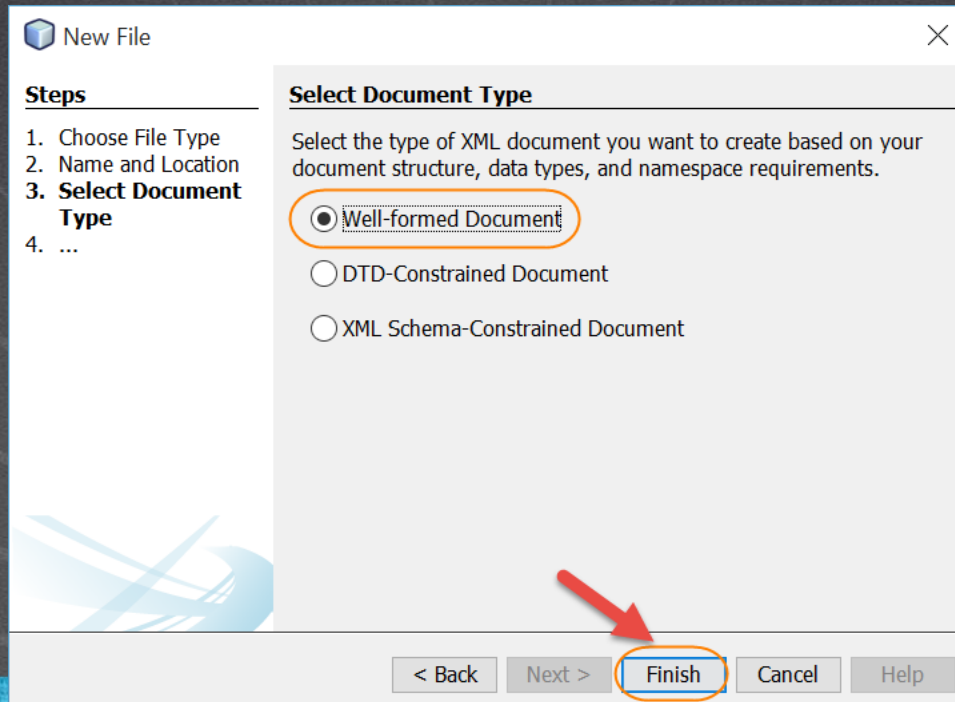
Folder: src/main/resources/META-INF Browse...

Created File: .\eccion03\PersonasWebHibernateJPA\src\main\resources\META-INF\persistence.xml.xml

< Back **Next >** Finish Cancel Help

PASO 6. CREAR UN ARCHIVO XML

Creamos el archivo persistence.xml



PASO 7. MODIFICAMOS EL CÓDIGO

Archivo persistence.xml:

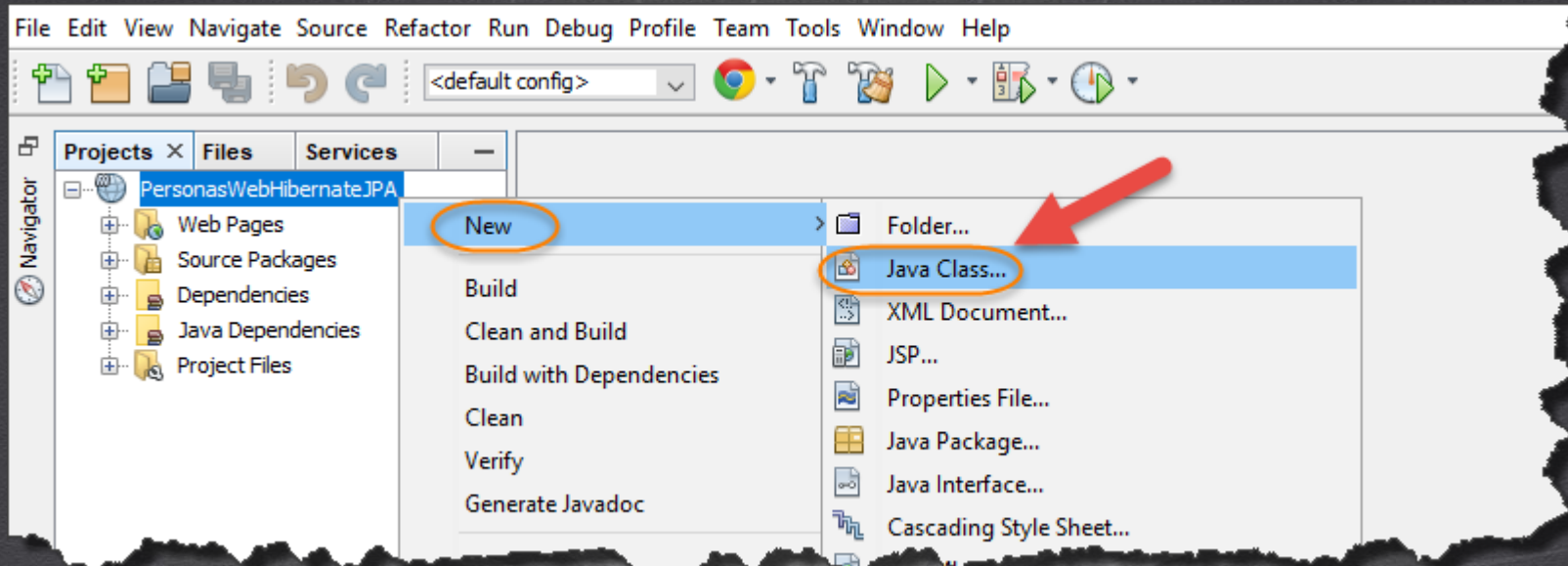
```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
  <persistence-unit name="HibernatePU" transaction-type="RESOURCE_LOCAL">
    <provider>org.hibernate.jpa.HibernatePersistenceProvider</provider>
    <class>mx.com.gm.Persona</class>
    <properties>
      <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/sga"/>
      <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>
```

CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 8. CREAR UNA CLASE

Creamos la clase Persona.java.



PASO 8. CREAR UNA CLASE

Creamos la clase Persona.java.

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 9. MODIFICAMOS EL CÓDIGO

Archivo Persona.java:

```
package mx.com.gm.domain;

import java.io.Serializable;
import javax.persistence.*;

@Entity
public class Persona implements Serializable {

    private static final long serialVersionUID = 1L;

    @Column(name = "id_persona")
    @Id
    private int idPersona;

    private String nombre;

    private String apellido;

    public Persona() {
    }
}
```

CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 9. MODIFICAMOS EL CÓDIGO

Archivo Persona.java:

```
public int getIdPersona() {
    return this.idPersona;
}

public void setIdPersona(int idPersona) {
    this.idPersona = idPersona;
}

public String getNombre() {
    return this.nombre;
}

public void setNombre(String nombre) {
    this.nombre = nombre;
}

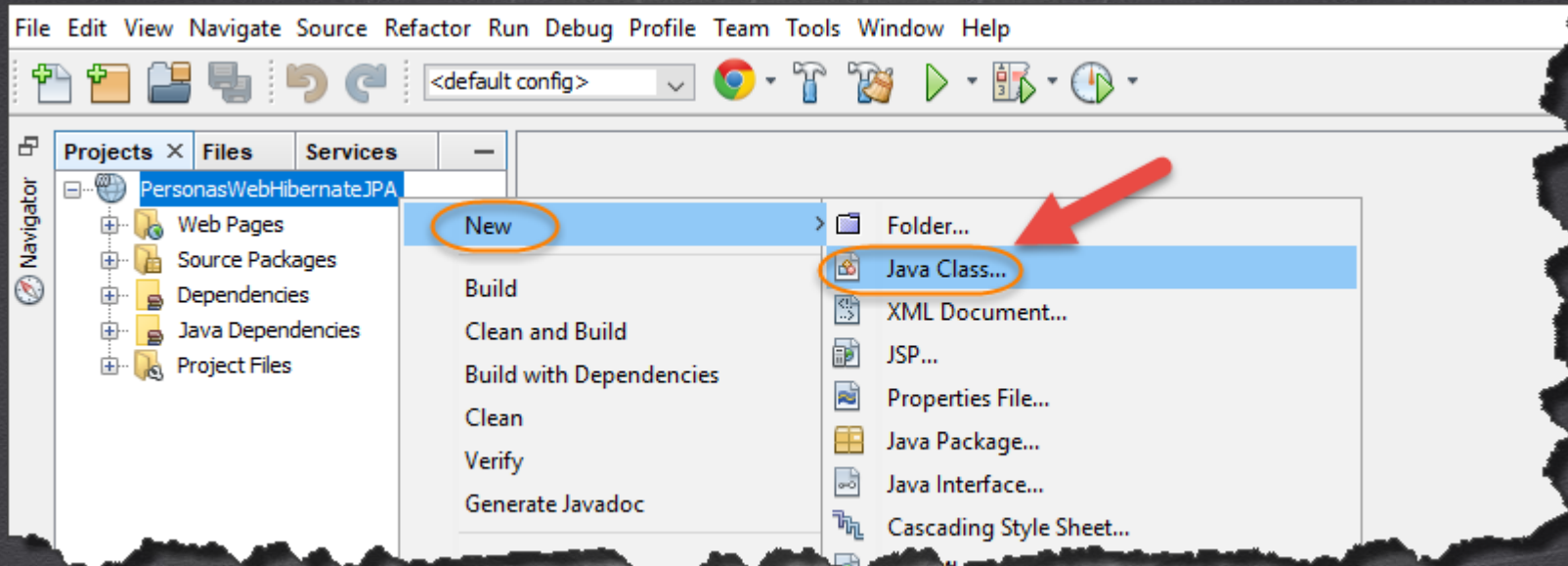
public String getApellido() {
    return this.apellido;
}

public void setApellido(String apellido) {
    this.apellido = apellido;
}

@Override
public String toString() {
    return "Persona [idPersona=" + idPersona + ", nombre=" + nombre + ", apellido=" + apellido + "];"
}
}
```


PASO 10. CREAR UNA CLASE

Creamos la clase PersonaDAO.java.



PASO 10. CREAR UNA CLASE

Creamos la clase PersonaDAO.java.

New Java Class

Steps

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

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

PASO 11. MODIFICAMOS EL CÓDIGO

Archivo PersonaDAO.java:

```
package mx.com.gm.dao;

import java.util.List;
import javax.persistence.*;
import mx.com.gm.domain.Persona;

public class PersonaDAO {
    protected EntityManager em;
    private EntityManagerFactory emf = null;

    public PersonaDAO() {
        // Usamos el persistence unit
        emf = Persistence.createEntityManagerFactory("HibernatePU");
    }

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


PASO 11. MODIFICAMOS EL CÓDIGO

Archivo PersonaDAO.java:

```
public List<Persona> listarPersonas() {  
    // Consulta a ejecutar  
    String hql = "SELECT p FROM Persona p";  
    em = getEntityManager();  
    Query query = em.createQuery(hql);  
    return query.getResultList();  
}  
  
public void insertar(Persona persona) {  
    try {  
        em = getEntityManager();  
        // Iniciamos una transaccion  
        em.getTransaction().begin();  
        // Insertamos la nueva persona  
        em.persist(persona);  
        // Terminamos la transaccion  
        em.getTransaction().commit();  
    } catch (Exception ex) {  
        System.out.println("Error al insertar objeto:" + ex.getMessage());  
        // ex.printStackTrace();  
    } finally {  
        if (em != null) {  
            em.close();  
        }  
    }  
}
```

PASO 11. MODIFICAMOS EL CÓDIGO

Archivo PersonaDAO.java:

```
public void actualizar(Persona persona) {
    try {
        em = getEntityManager();
        // Iniciamos una transaccion
        em.getTransaction().begin();
        // Actualizamos al objeto persona
        em.merge(persona);
        // Terminamos la transaccion
        em.getTransaction().commit();
    } catch (Exception ex) {
        System.out.println("Error al actualizar objeto:" + ex.getMessage());
        // ex.printStackTrace();
    } finally {
        if (em != null) {
            em.close();
        }
    }
}
```

PASO 11. MODIFICAMOS EL CÓDIGO

Archivo PersonaDAO.java:

```
public void eliminar(Persona persona) {  
    try {  
        em = getEntityManager();  
        // Iniciamos una transaccion  
        em.getTransaction().begin();  
        // Sincronizamos y eliminamos a la persona  
        em.remove(em.merge(persona));  
        // Terminamos la transaccion  
        em.getTransaction().commit();  
    } catch (Exception ex) {  
        System.out.println("Error al eliminar objeto:" + ex.getMessage());  
        // ex.printStackTrace();  
    } finally {  
        if (em != null) {  
            em.close();  
        }  
    }  
}
```

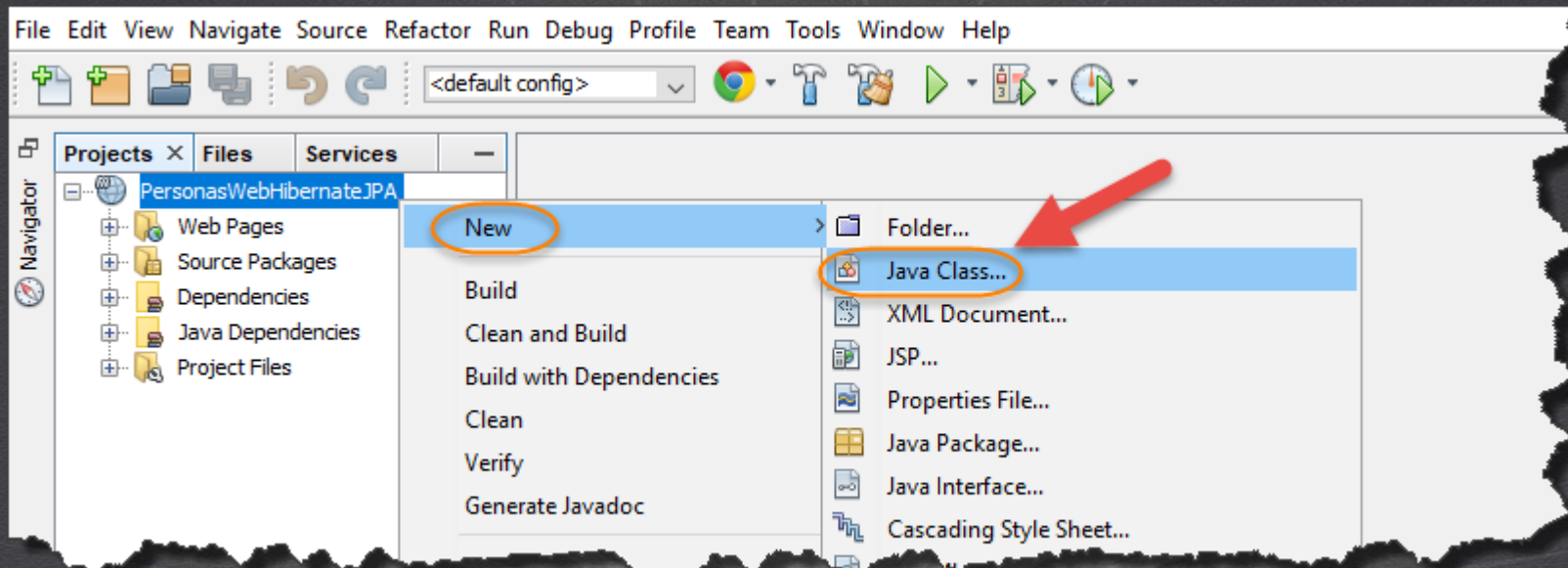

PASO 11. MODIFICAMOS EL CÓDIGO

Archivo PersonaDAO.java:

```
public Persona buscarPorId(Persona p) {  
    em = getEntityManager();  
    return em.find(Persona.class, p.getIdPersona());  
}  
  
private EntityManager getEntityManager() {  
    return emf.createEntityManager();  
}  
}
```

PASO 12. CREAR UNA CLASE

Creamos la clase ServicioPersonas.java:



CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 12. CREAR UNA CLASE

Creamos la clase ServicioPersonas.java:

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. MODIFICAMOS EL CÓDIGO

Archivo ServicioPersonas.java:

```
package mx.com.gm.servicio;

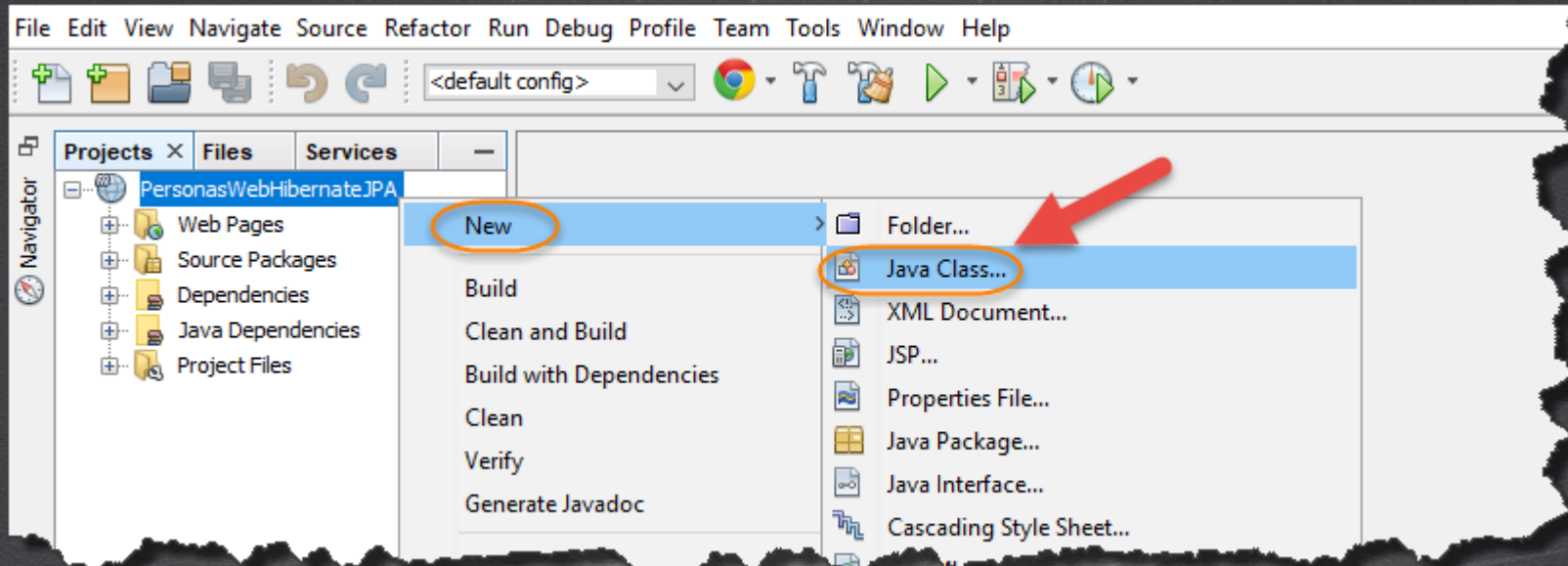
import java.util.List;
import mx.com.gm.dao.PersonaDAO;
import mx.com.gm.domain.Persona;

public class ServicioPersonas {

    public List<Persona> listarPersonas() {
        PersonaDAO personaDao = new PersonaDAO();
        return personaDao.listarPersonas();
    }
}
```

PASO 14. CREAR UNA CLASE

Creamos la clase ServletControlador.java:



CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 14. CREAR UNA CLASE

Creamos la clase ServletControlador.java:

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 15. MODIFICAMOS EL CÓDIGO

Archivo ServletControlador.java:

```
package mx.com.gm.web;

import java.io.IOException;
import java.util.List;
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
import mx.com.gm.domain.Persona;
import mx.com.gm.servicio.ServicioPersonas;

@WebServlet(name = "ServletControlador", urlPatterns = {"/ServletControlador"})
public class ServletControlador extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        try {
            // 1. Nos conectamos a la capa de servicio
            ServicioPersonas servicioPersonas = new ServicioPersonas();

            // 2.Solicitamos el listado de personas
            List<Persona> personas = servicioPersonas.listarPersonas();
```

PASO 15. MODIFICAMOS EL CÓDIGO

Archivo ServletControlador.java:

```
// 3.Compartimos el modelo con la vista
request.setAttribute("personas", personas);
System.out.println("listado:" + personas);

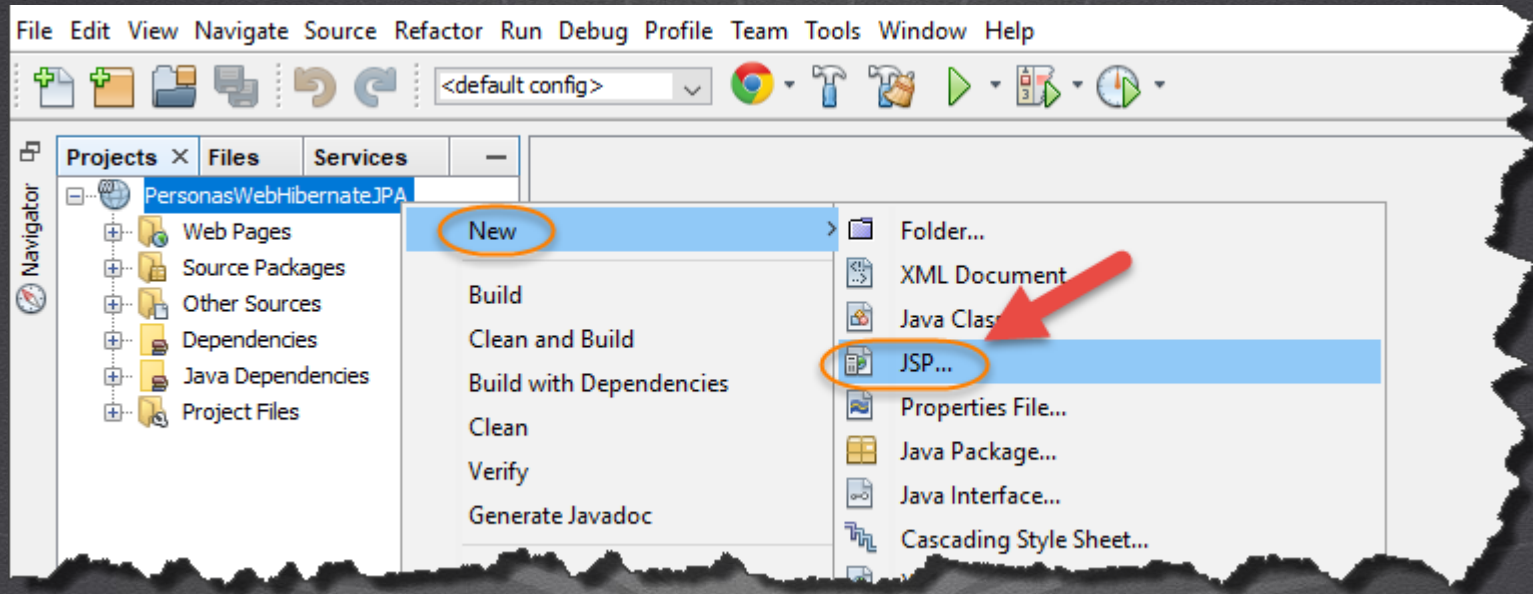
// 4.Redireccionamos a la vista
request.getRequestDispatcher("/WEB-INF/listado.jsp").forward(request, response);

} catch (Exception e) {
    e.printStackTrace();
}

}
```

PASO 16. CREAR UN ARCHIVO JSP

Creamos el archivo index.jsp

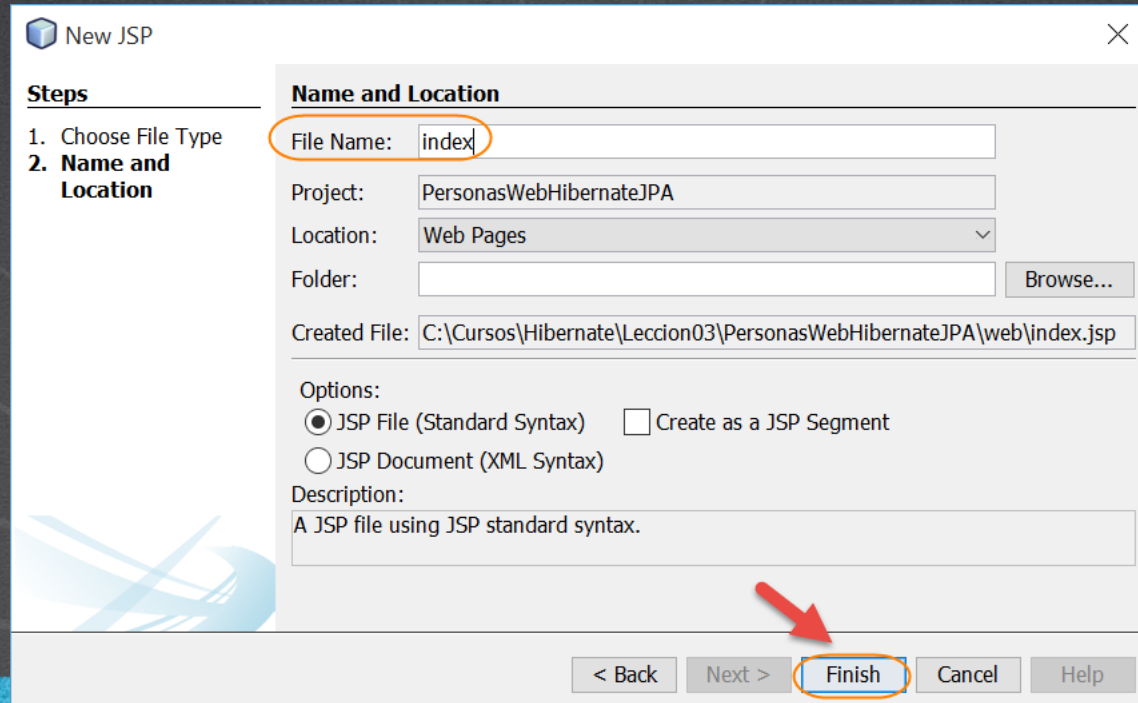


CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 16. CREAR UN ARCHIVO JSP

Creamos el archivo index.jsp



New JSP

Steps

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

Name and Location

File Name:

Project:

Location:

Folder:

Created File:

Options:

☒ JSP File (Standard Syntax) ☐ Create as a JSP Segment

☐ JSP Document (XML Syntax)

Description:

A JSP file using JSP standard syntax.

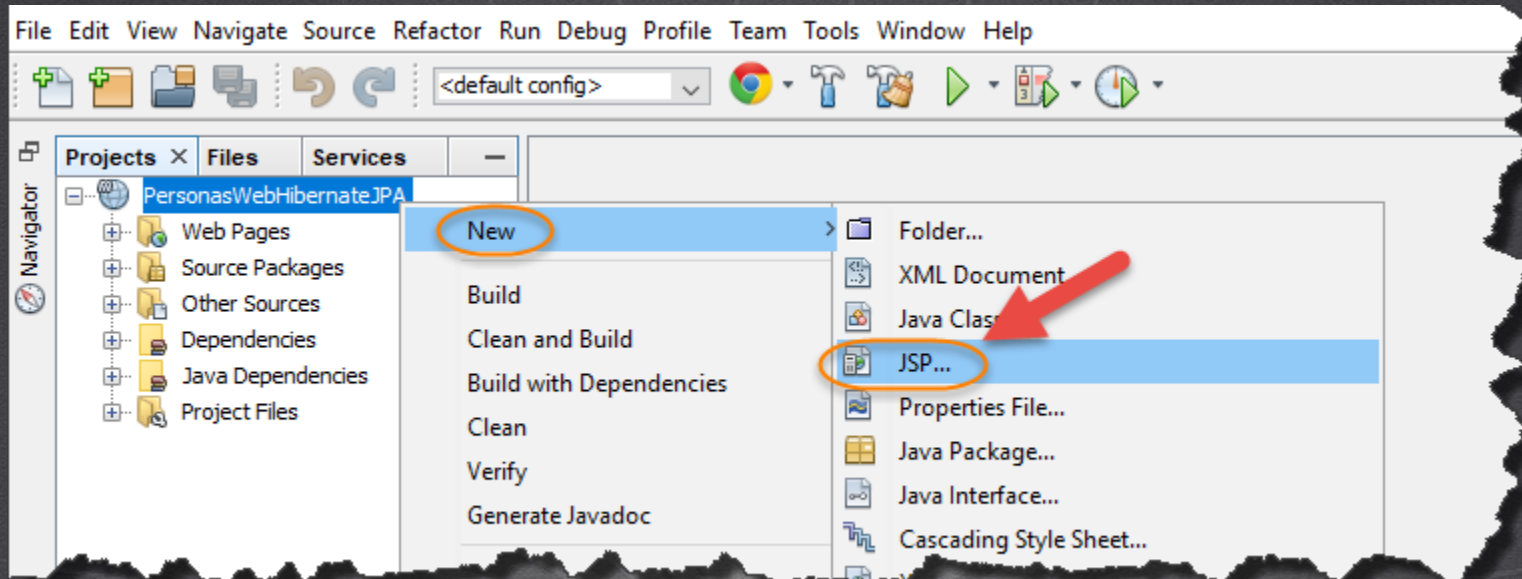
PASO 17. MODIFICAMOS EL CÓDIGO

Archivo index.jsp:

```
<html>
  <head>
    <meta charset="UTF-8">
    <title>Ejemplo Personas Web Hibernate JPA 1</title>
  </head>
  <body>
    <h1>Ejemplo Personas Web Hibernate JPA 1</h1>
    <br>
    <a href="${pageContext.request.contextPath}/ServletControlador">
      Listar Personas con Hibernate y JPA
    </a>
  </body>
</html>
```

PASO 18. CREAR UN ARCHIVO JSP

Creamos el archivo listado.jsp



CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 18. CREAR UN ARCHIVO JSP

Creamos el archivo listado.jsp

New JSP

Steps

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

Name and Location

File Name: listado

Project: PersonasWebHibernateJPA

Location: Web Pages

Folder: WEB-INF

Browse...

Created File: ;\Hibernate\Leccion03\PersonasWebHibernateJPA\src\main\webapp\WEB-INF\listado.jsp

Options:

☒ JSP File (Standard Syntax) ☐ Create as a JSP Segment

☐ JSP Document (XML Syntax)

Description:

A JSP file using JSP standard syntax.

< Back Next > **Finish** Cancel Help

PASO 19. MODIFICAMOS EL CÓDIGO

Archivo listado.jsp:

```
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Listado Web Personas Hibernate JPA 1</title>
  </head>
  <body>
    <table width="383" border="1" bgcolor="#c0c0c0" height="76">
      <tbody>
        <caption>Listado de Personas</caption>
        <tr>
          <th>
            &nbsp;Id Persona
          <br>
          </th>
          <th>
            &nbsp;Nombre
          </th>
          <th>
            &nbsp;Apellido
          </th>
        </tr>
```

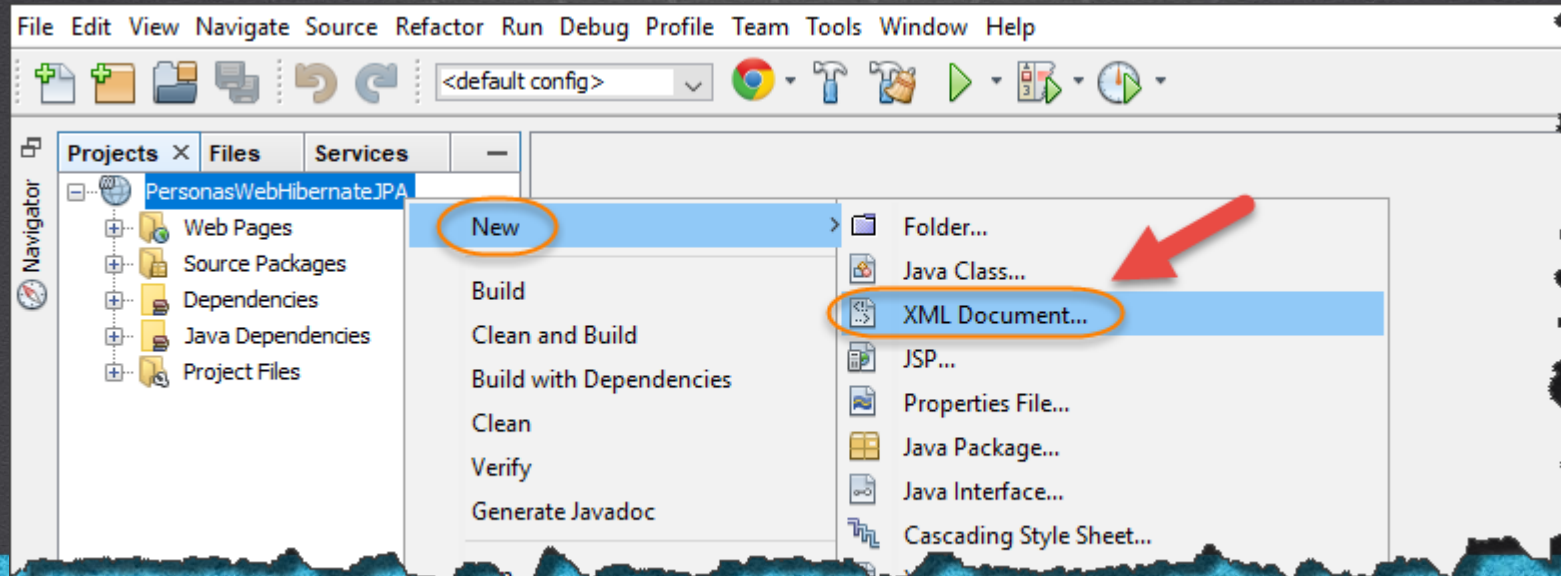
PASO 19. MODIFICAMOS EL CÓDIGO

Archivo listado.jsp:

```
<!--Iteramos los elementos de la lista de personas --%>
<c:forEach var="persona" items="${personas}" >
    <tr>
        <td>
            ${persona.idPersona}
        </td>
        <td>
            ${persona.nombre}
        </td>
        <td>
            ${persona.apellido}
        </td>
    </tr>
</c:forEach>
</tbody>
</table>
</body>
</html>
```


PASO 20. CREAR UN ARCHIVO XML

Creamos un archivo log4j2.xml. Este archivo se usa para el manejo de logging en Java:



CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

PASO 20. CREAR UN ARCHIVO XML

Creamos un 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:

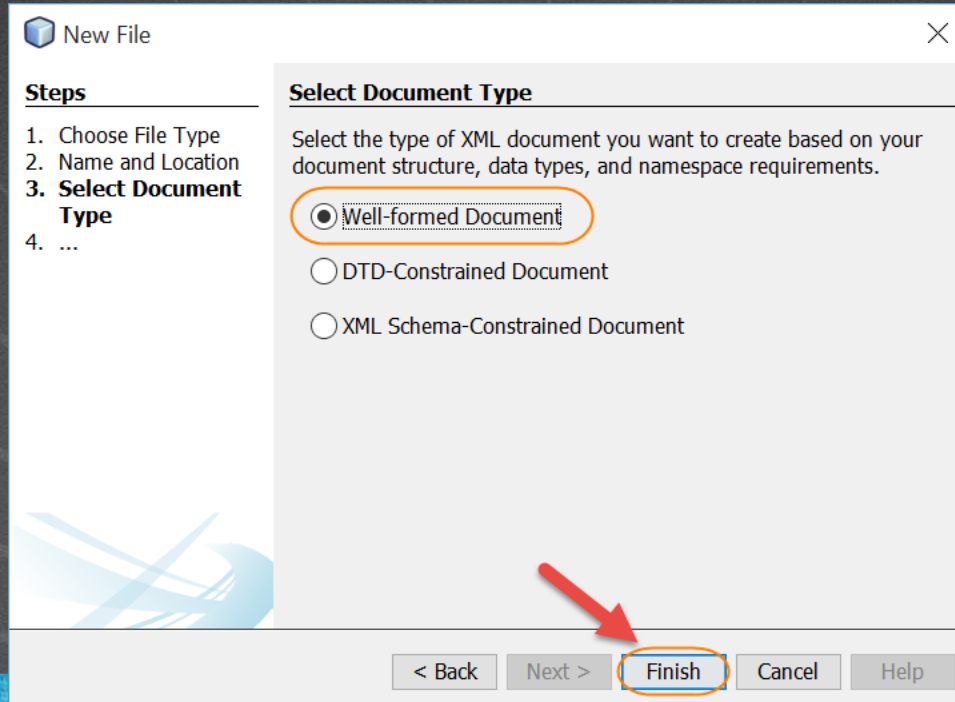
Project:

Folder:

Created File:

PASO 20. CREAR UN ARCHIVO XML

Creamos un archivo log4j2.xml:



PASO 21. MODIFICAMOS EL CÓDIGO

Archivo log4j2.xml:

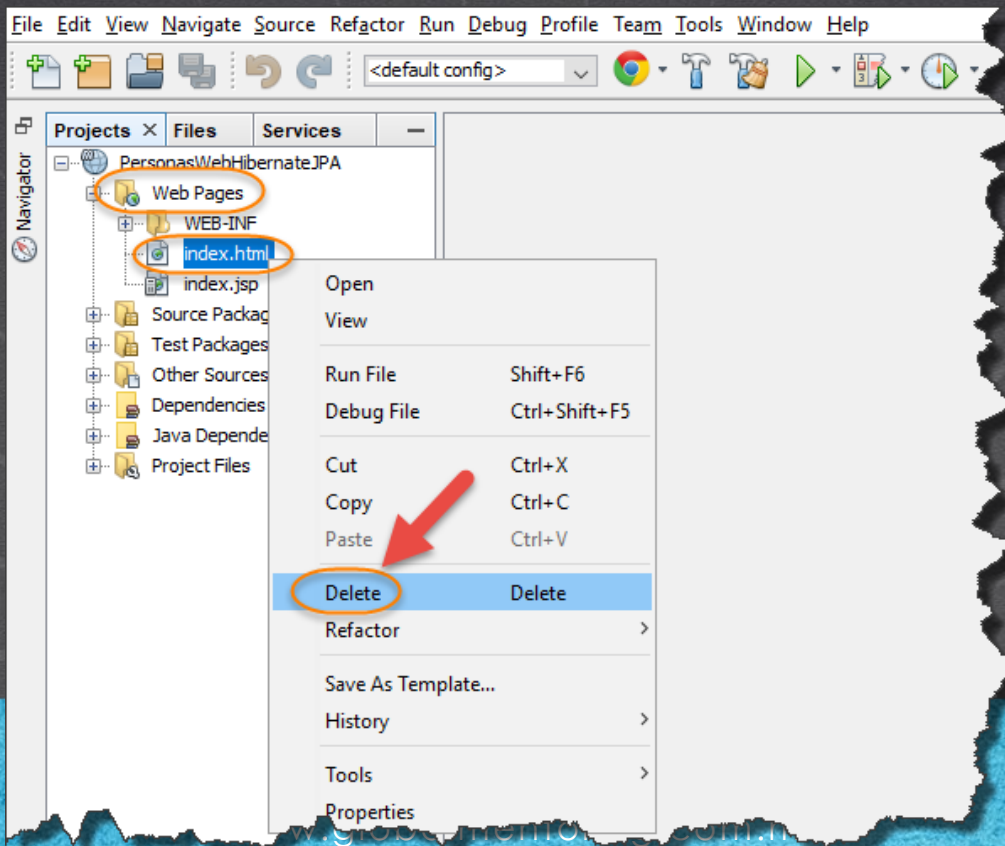
```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="INFO">
  <Appenders>
    <Console name="Console" target="SYSTEM_OUT">
      <PatternLayout pattern="%d{HH:mm:ss} [%t] %-5level %logger{36} - %msg%n" />
    </Console>
  </Appenders>
  <Loggers>
    <Logger name="org.hibernate.SQL" level="debug" 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 HIBERNATE Y JPA

www.globalmentoring.com.mx

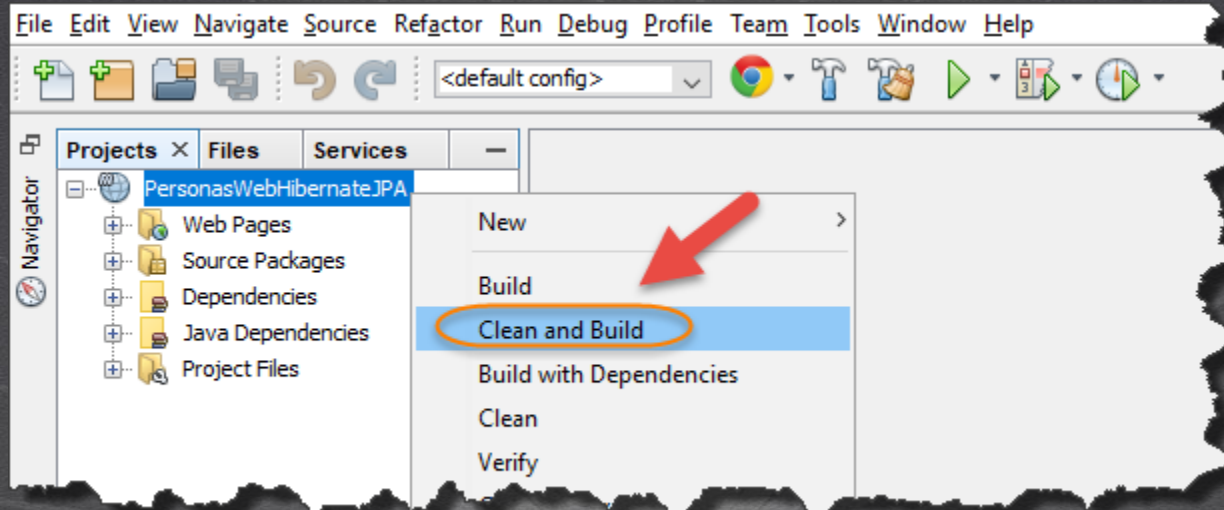
22. ELIMINAMOS EL ARCHIVO INDEX.HTML

- Eliminamos el archivo index.html si existe:



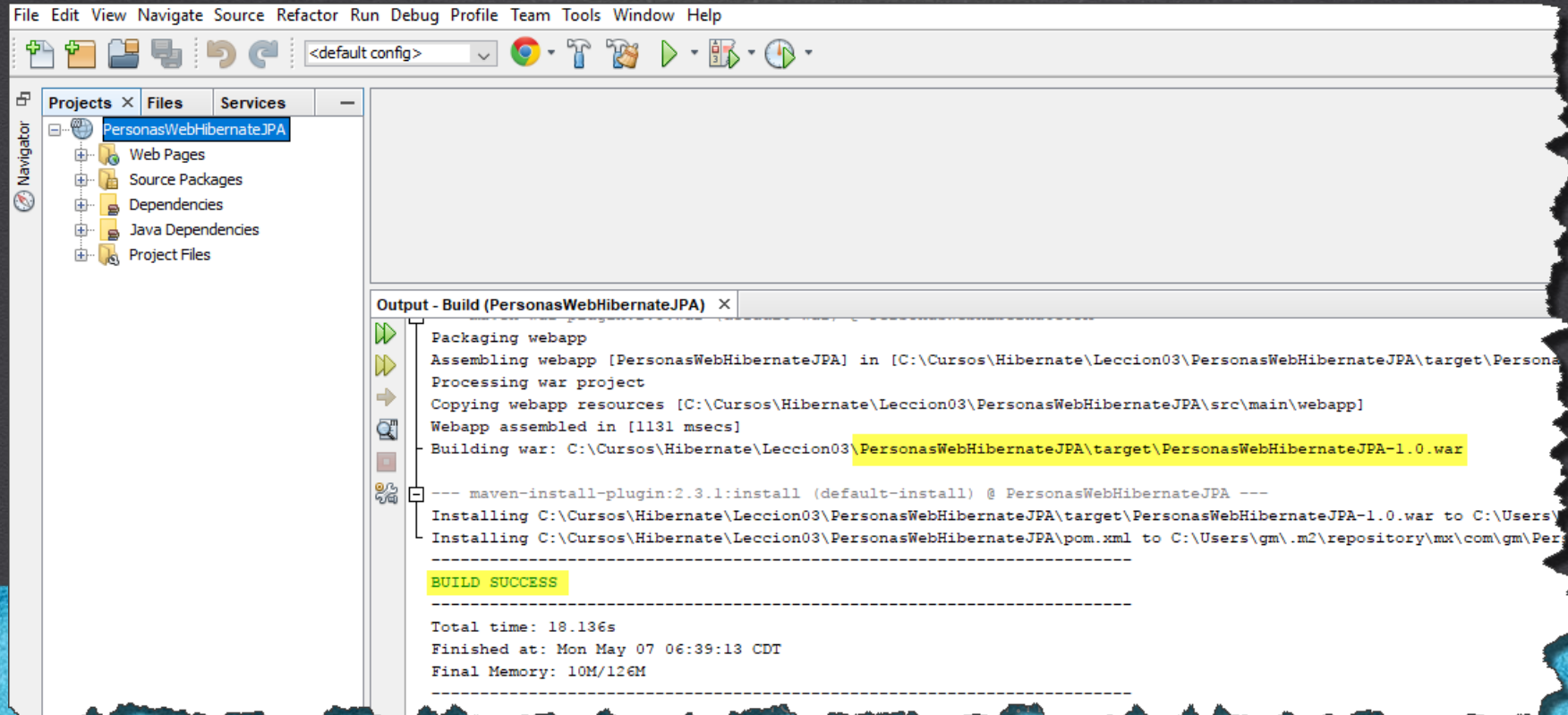
23. HACEMOS CLEAN & BUILD

•Regresamos a la vista de Proyectos. Para que se descarguen las nuevas 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:



23. HACEMOS CLEAN & BUILD

- Si ya no tuvo que descargar ninguna librería debido a que podría ya tener todas descargadas el proceso es más rápido. Al final deberemos observar lo siguiente:



The screenshot shows an IDE window with the 'Output - Build (PersonasWebHibernateJPA)' tab selected. The build log displays the following steps:

```
Packaging webapp
Assembling webapp [PersonasWebHibernateJPA] in [C:\Cursos\Hibernate\Leccion03\PersonasWebHibernateJPA\target\PersonasWebHibernateJPA]
Processing war project
Copying webapp resources [C:\Cursos\Hibernate\Leccion03\PersonasWebHibernateJPA\src\main\webapp]
Webapp assembled in [1131 msecs]
Building war: C:\Cursos\Hibernate\Leccion03\PersonasWebHibernateJPA\target\PersonasWebHibernateJPA-1.0.war

--- maven-install-plugin:2.3.1:install (default-install) @ PersonasWebHibernateJPA ---
Installing C:\Cursos\Hibernate\Leccion03\PersonasWebHibernateJPA\target\PersonasWebHibernateJPA-1.0.war to C:\Users\gm\repository\mx\com\gm\PersonasWebHibernateJPA-1.0.war
Installing C:\Cursos\Hibernate\Leccion03\PersonasWebHibernateJPA\pom.xml to C:\Users\gm\repository\mx\com\gm\PersonasWebHibernateJPA-1.0.war

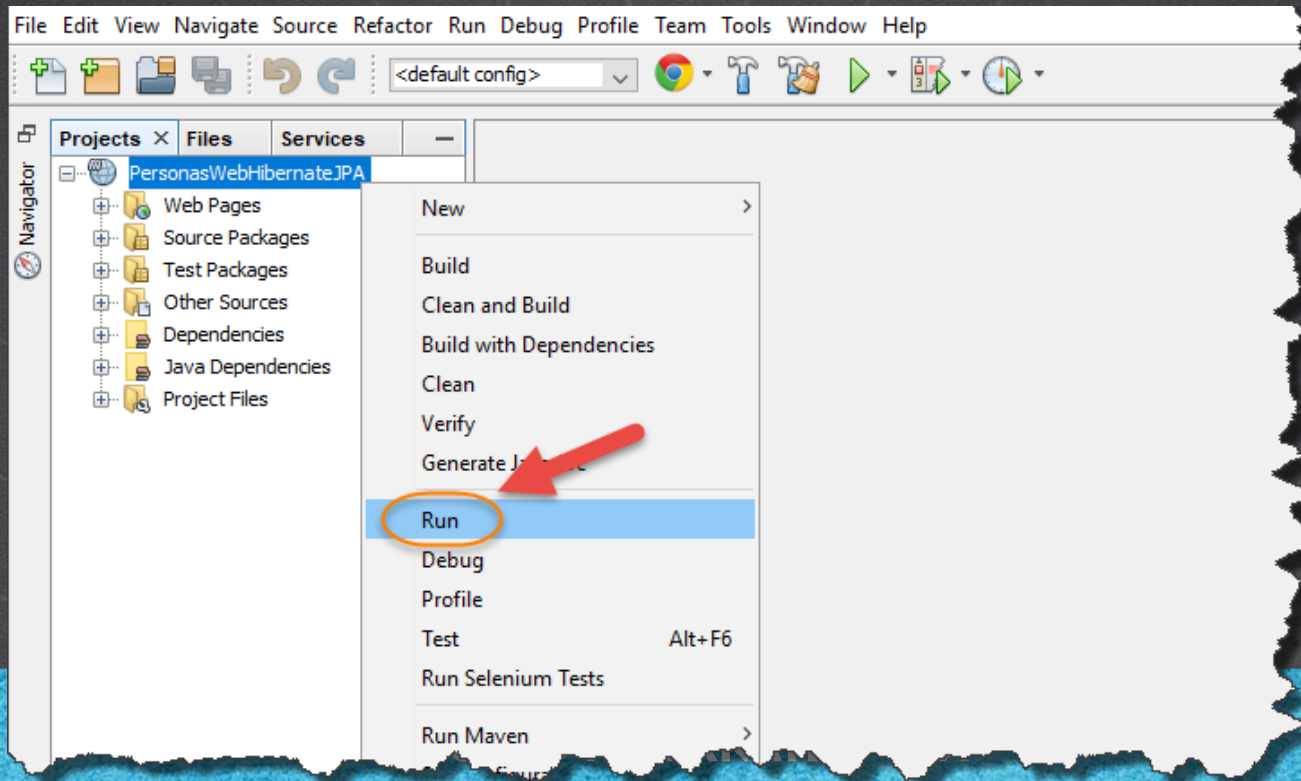
BUILD SUCCESS

Total time: 18.136s
Finished at: Mon May 07 06:39:13 CDT
Final Memory: 10M/126M
```

The IDE interface includes a menu bar (File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help), a toolbar with icons for file operations and running, and a left sidebar with a 'Navigator' pane showing the project structure: PersonasWebHibernateJPA, Web Pages, Source Packages, Dependencies, Java Dependencies, and Project Files.

PASO 24. EJECUTAMOS EL PROYECTO

Ejecutamos el proyecto:



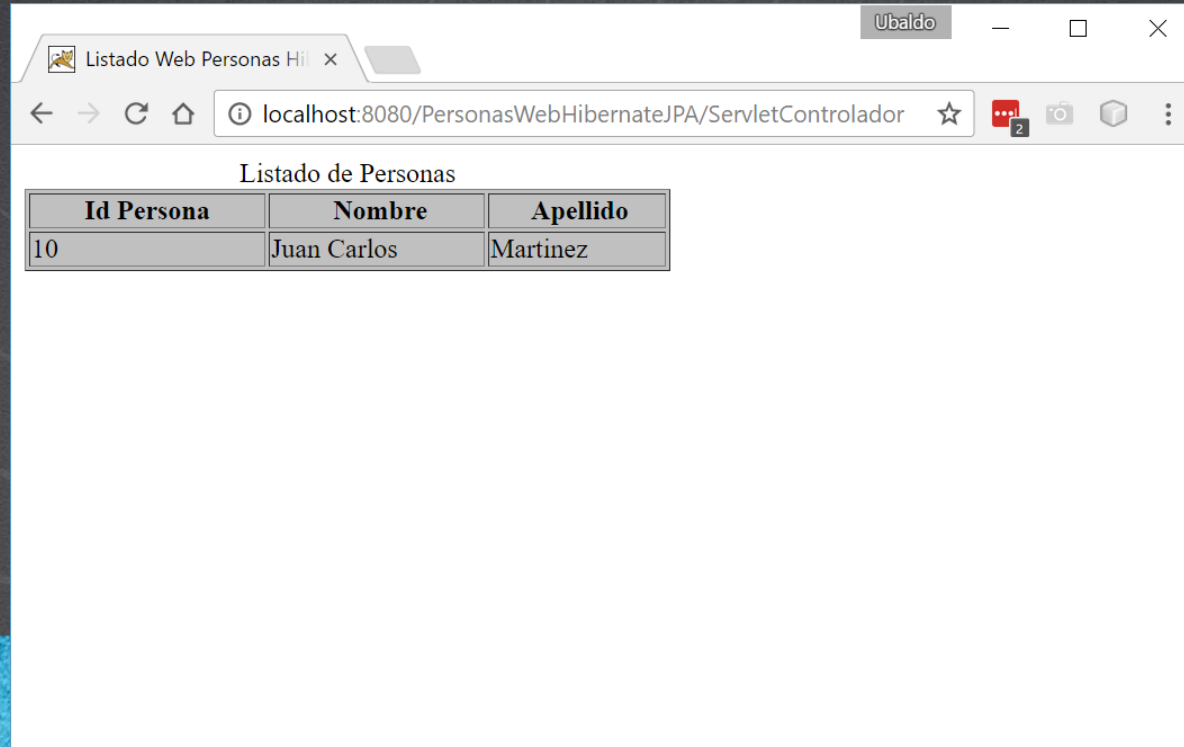
PASO 24. EJECUTAMOS EL PROYECTO (CONT)

Entramos al listado de personas:



PASO 24. EJECUTAMOS EL PROYECTO (CONT)

Entramos al listado de personas:



CONCLUSIÓN DEL EJERCICIO

- Con este ejercicio hemos integrado una aplicación Web con Hibernate y JPA utilizando Glassfish como servidor de aplicaciones.
- Aplicamos varios patrones de diseño como MVC, DAO y DTO para en su conjunto poder desplegar el listado de personas, todo esto con ayuda de la capa de persistencia apoyados de Hibernate y JPA.



Experiencia y Conocimiento para tu vida

CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx

CURSO ONLINE

HIBERNATE Y JPA

Por: Ing. Ubaldo Acosta



Experiencia y Conocimiento para tu vida

CURSO HIBERNATE Y JPA

www.globalmentoring.com.mx