

**CURSO JAVA EE**

# **SGA CON JAVA WEBSERVICES JAX-WS**



Por el experto: Ing. Ubaldo Acosta



Experiencia y Conocimiento para tu vida



**CURSO JAVA EE**

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

# OBJETIVO DEL EJERCICIO

El objetivo del ejercicio exponer el método listarPersonas del EJB del proyecto SGA como un Web Services con ayuda del API JAX-WS. El resultado se muestra a continuación:

← → ↺ ↻ localhost:8080/PersonaServiceImplService/PersonaServiceImpl?test

### listarPersonas Method invocation

Method parameter(s)

Type	Value
------	-------

Method returned

java.util.List : "[mx.com.gm.sga.servicio.Persona@114b08c0, mx.com.gm.sga.servicio.Persona@32a15]

### SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap"
<SOAP-ENV:Header/>
<S:Body>
  <ns2:listarPersonas xmlns:ns2="http://servicio.sga.gm.com.mx"/>
</S:Body>
</S:Envelope>
```

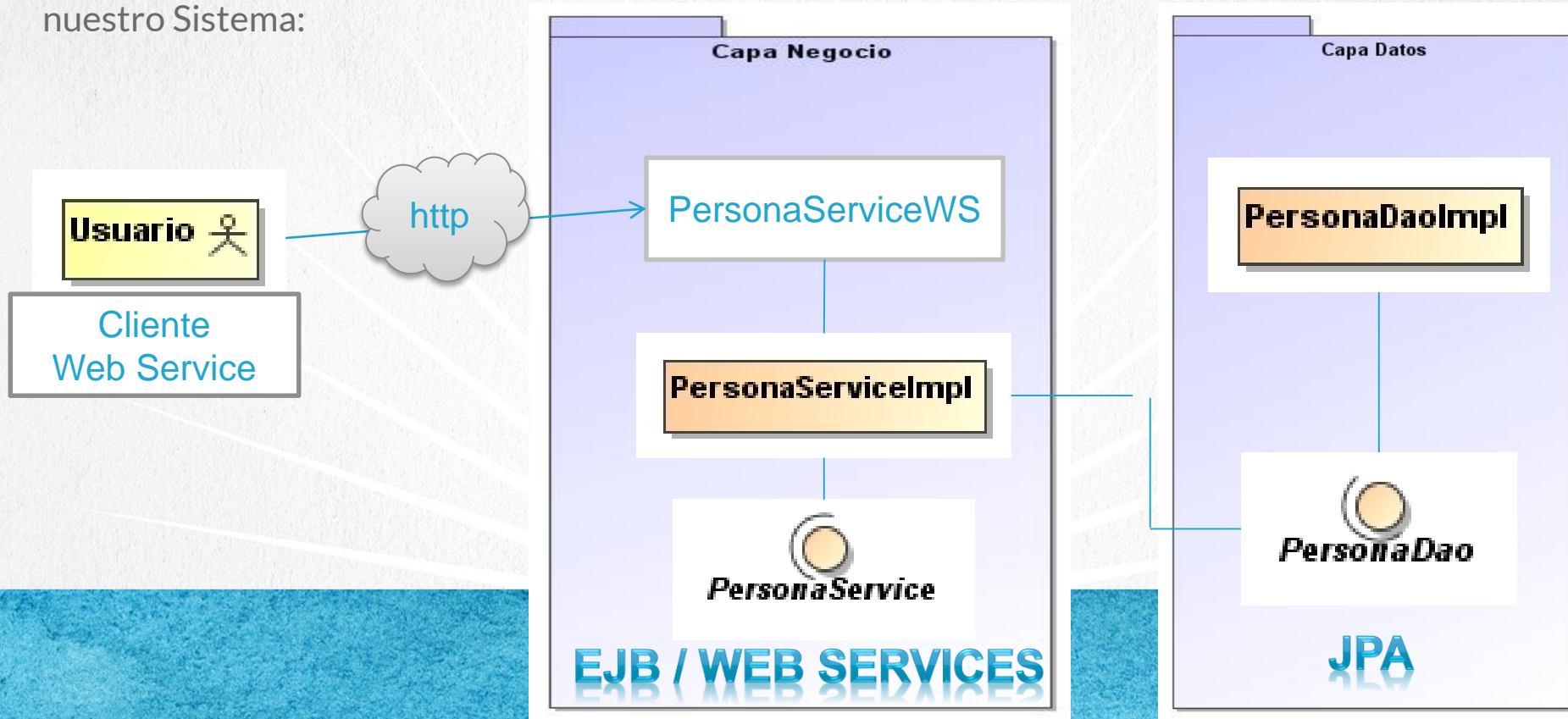
### SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap"
<SOAP-ENV:Header/>
<S:Body>
  <ns2:listarPersonasResponse xmlns:ns2="http://servicio.sga.gm.com.mx">
    <return>
      <idPersona>22</idPersona>
      <nombre>Oscar</nombre>
      <apellidoPaterno>Milan</apellidoPaterno>
      <apellidoMaterno>Lara</apellidoMaterno>
      <email>omilan@mail.com</email>
      <telefono>55112233</telefono>
    </return>
    <return>
      <idPersona>24</idPersona>
      <nombre>Karla</nombre>
      <apellidoPaterno>Lara</apellidoPaterno>
      <apellidoMaterno>Juarez</apellidoMaterno>
      <email>klara@mail.com</email>
      <telefono>44331122</telefono>
    </return>
  </ns2:listarPersonasResponse>
</S:Body>
</S:Envelope>
```



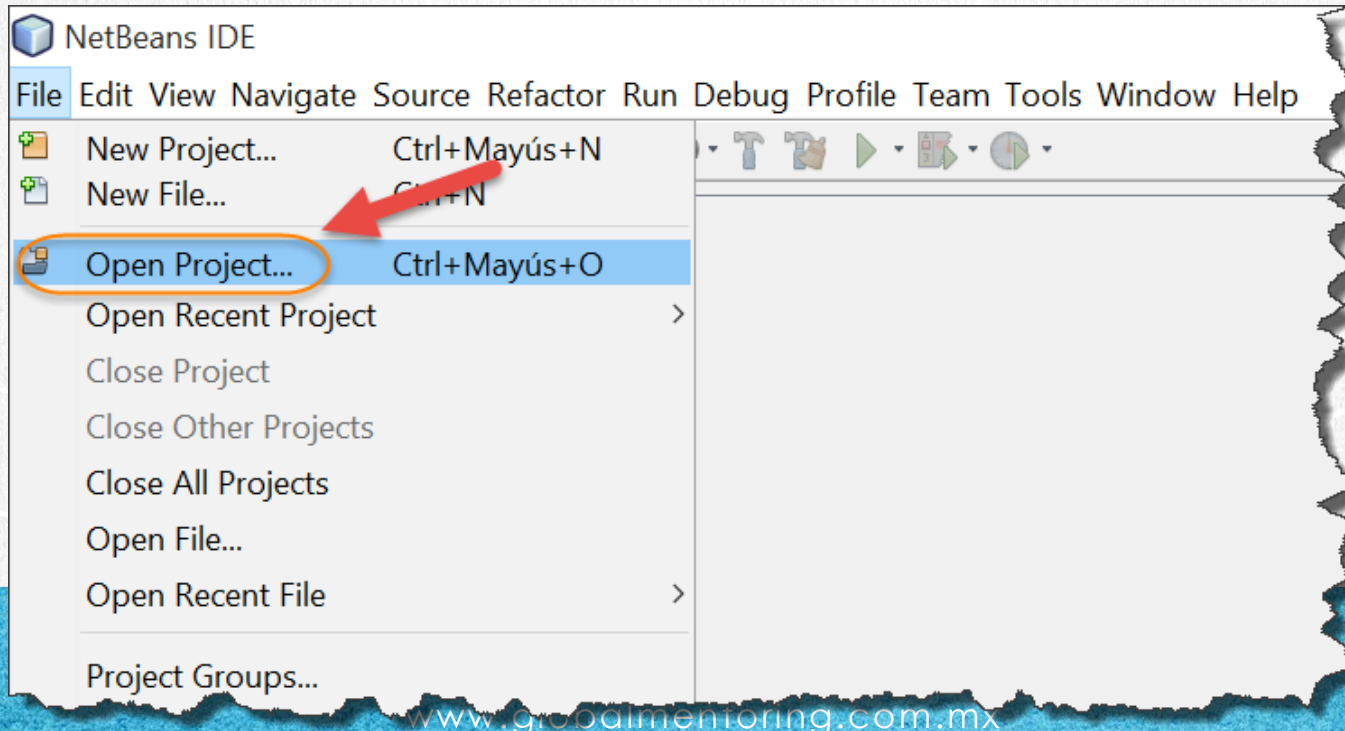
# ARQUITECTURA SGA CON WEB SERVICES

Este es el Diagrama de Clases del Ejercicio, donde se pueden observar la Arquitectura de nuestro Sistema:



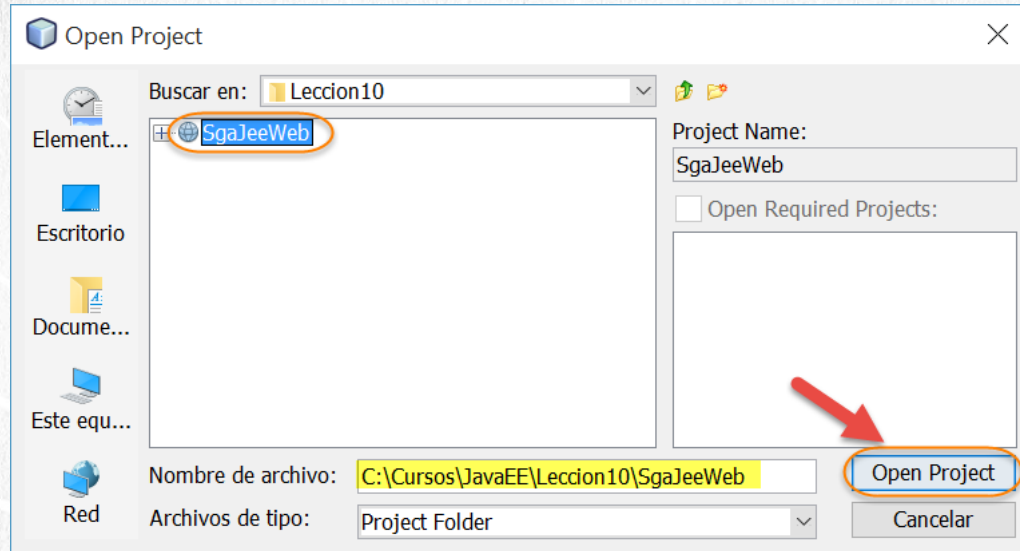
# PASO 1. ABRIMOS EL PROYECTO

En caso que no tengamos abierto el proyecto SgaJeeWeb lo abrimos:



# PASO 1. ABRIMOS EL PROYECTO

En caso que no tengamos abierto el proyecto SgaJeeWeb lo abrimos:



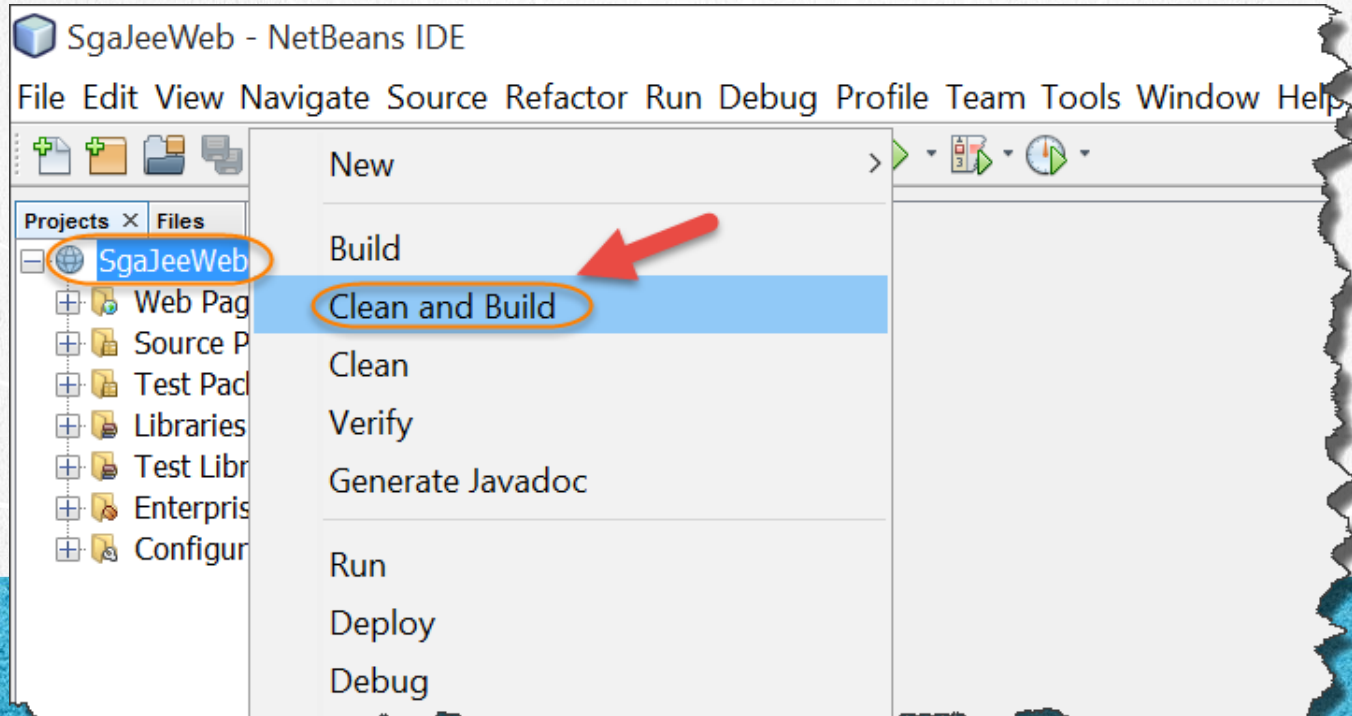
**CURSO JAVA EE**

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



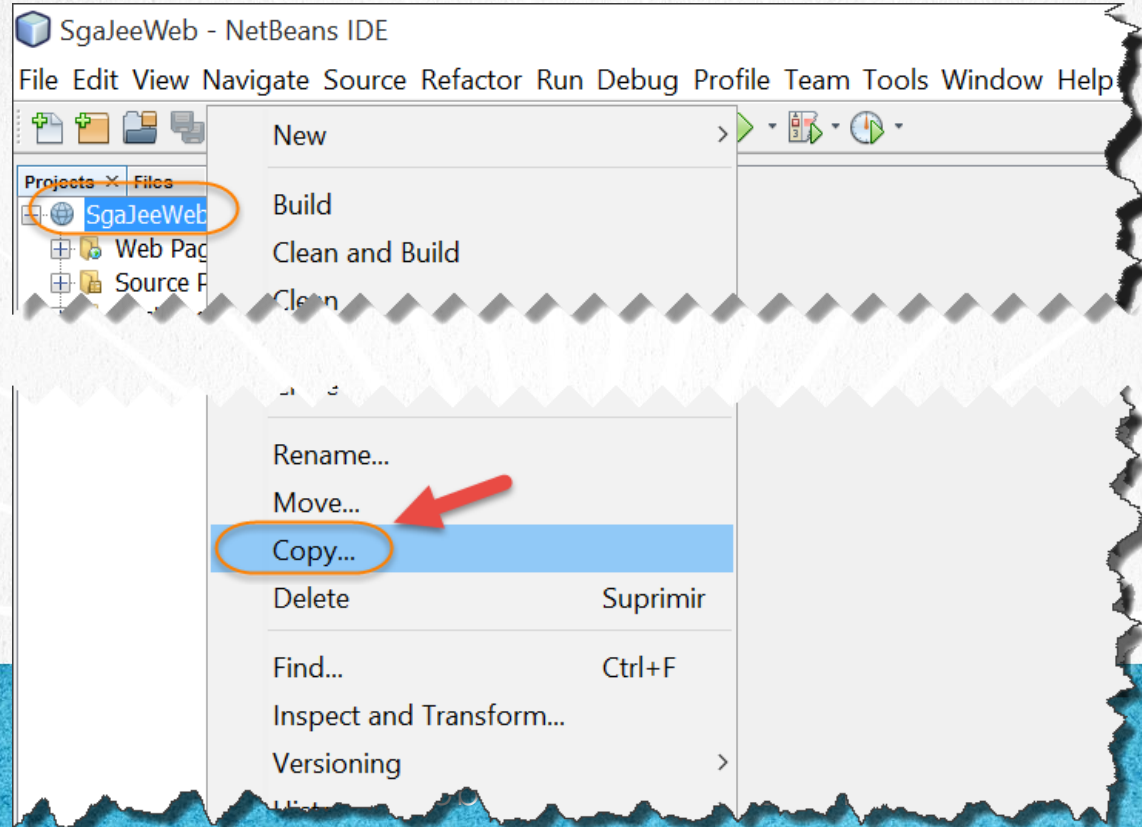
# PASO 1. ABRIMOS EL PROYECTO

Esperamos a que cargue completamente el proyecto. En caso que marque error el proyecto, hacemos un Clean & Build para que se muestren todos los archivos, este paso es opcional:



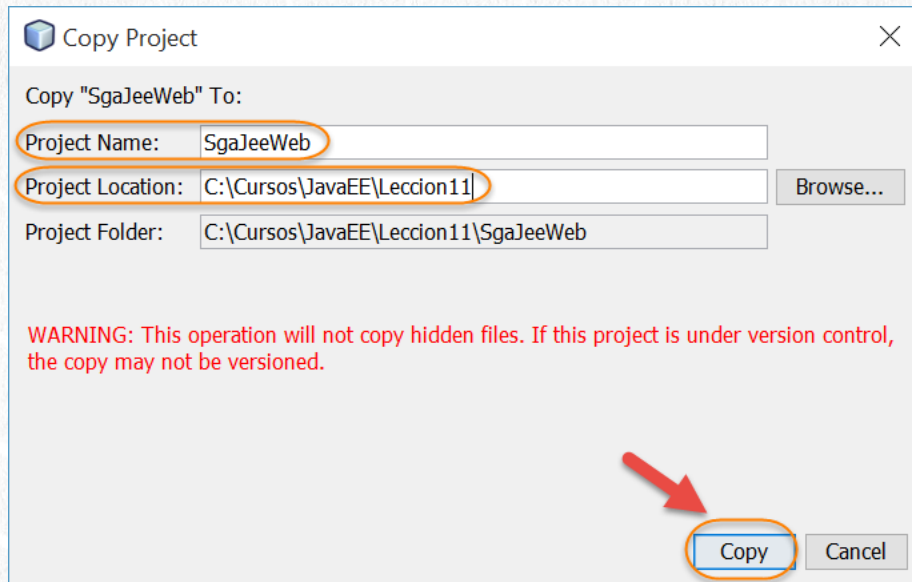
# PASO 2. COPIAMOS EL PROYECTO

Copiamos el proyecto para ponerlo en la nueva ruta:



# PASO 2. COPIAMOS EL PROYECTO

Copiamos el proyecto para ponerlo en la nueva ruta:



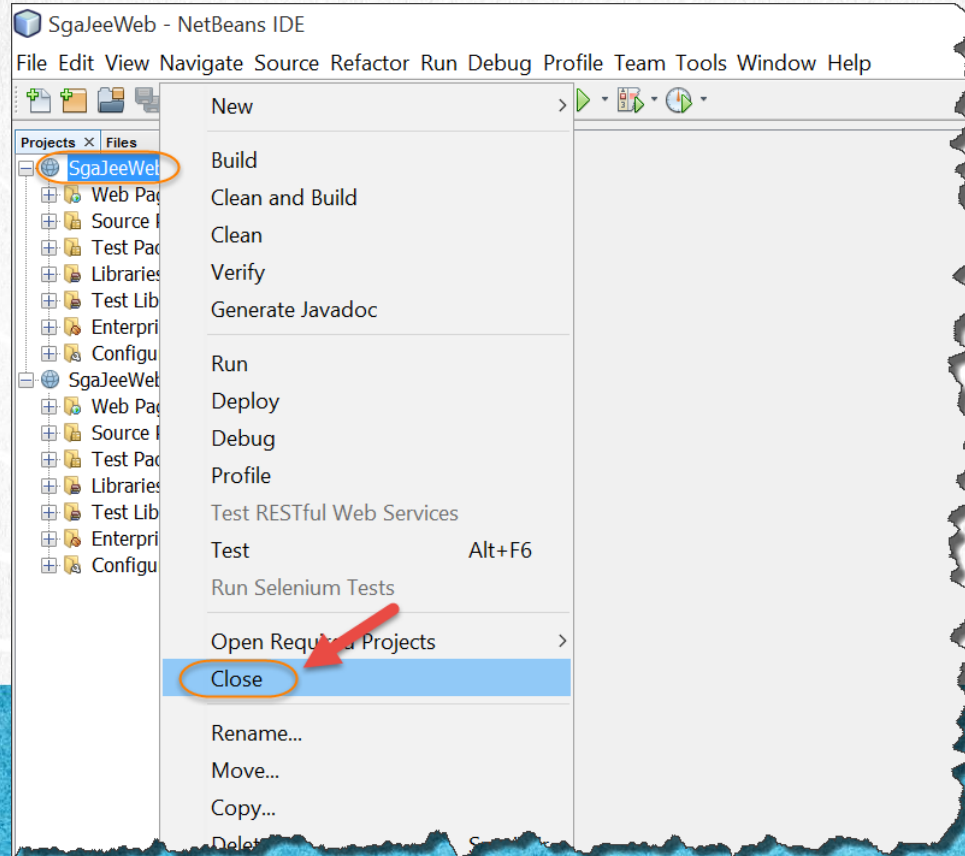
**CURSO JAVA EE**

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



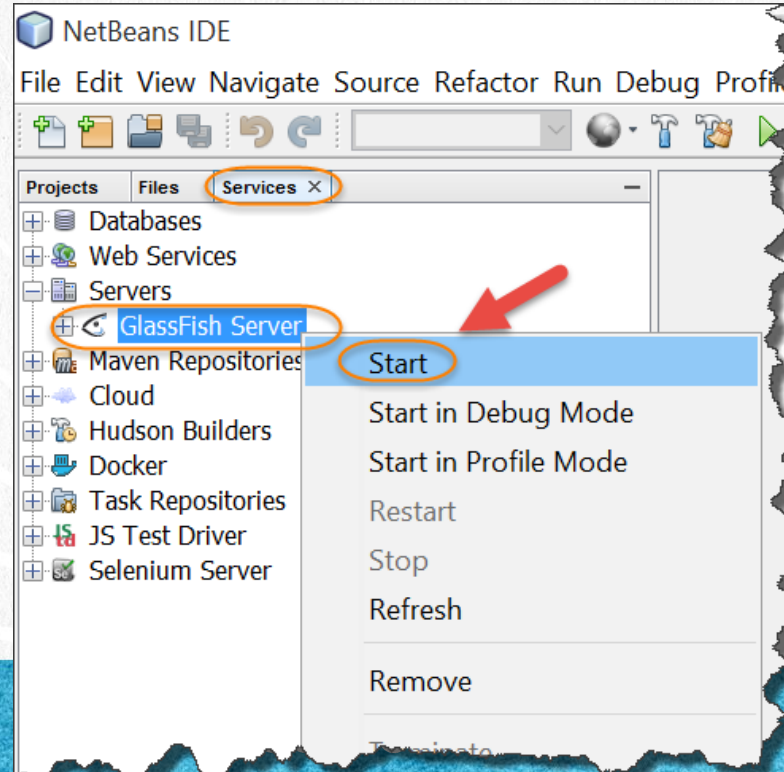
# PASO 2. CERRAMOS EL PROYECTO

Cerramos el proyecto anterior y dejamos solo el nuevo:



# PASO 3. HACEMOS UNDEPLOY EN GLASSFISH

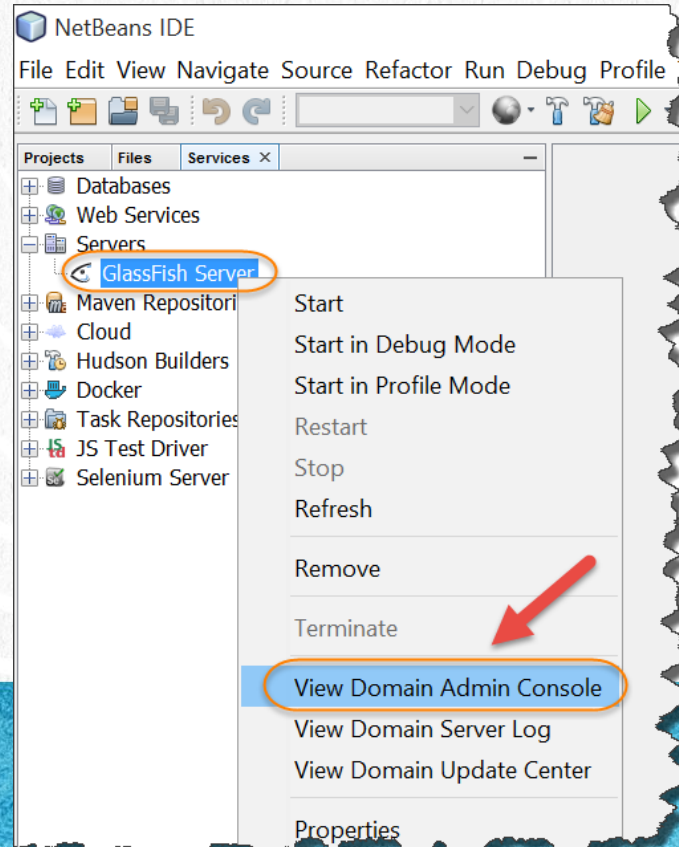
Hacemos undeploy de los proyectos que tengamos en Glassfish:





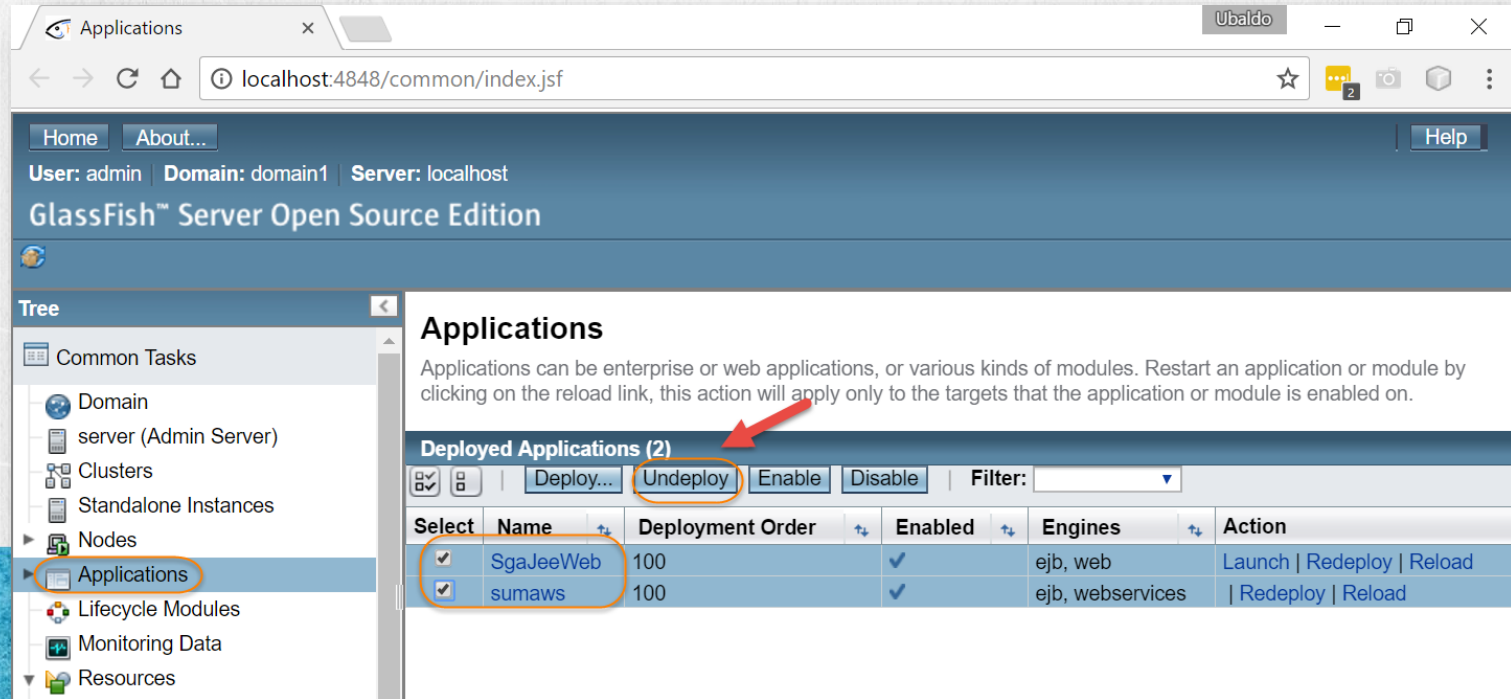
# PASO 3. HACEMOS UNDEPLOY EN GLASSFISH

Hacemos undeploy de los proyectos que tengamos en Glassfish:



# PASO 3. HACEMOS UNDEPLOY EN GLASSFISH

Hacemos undeploy de los proyectos que tengamos en Glassfish. Esto lo hacemos para que los EJB que estén desplegados no choquen con los que vamos a subir y también el log de Glassfish esté lo más limpio posible:



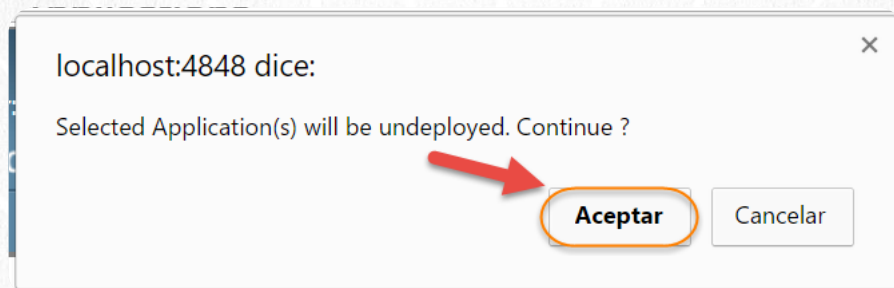
The screenshot shows the GlassFish Server Open Source Edition web console. The browser address bar displays `localhost:4848/common/index.jsf`. The page title is "GlassFish™ Server Open Source Edition". The left sidebar shows a tree view with "Applications" selected and highlighted with a red circle. The main content area is titled "Applications" and contains a table of deployed applications. The "Undeploy" button in the table's toolbar is highlighted with a red circle and a red arrow. The table lists two applications: "SgaJeeWeb" and "sumaws".

Select	Name	Deployment Order	Enabled	Engines	Action
<input checked="" type="checkbox"/>	SgaJeeWeb	100	✓	ejb, web	Launch   Redeploy   Reload
<input checked="" type="checkbox"/>	sumaws	100	✓	ejb, webservices	Redeploy   Reload



# PASO 3. HACEMOS UNDEPLOY EN GLASSFISH

Hacemos undeploy de los proyectos que tengamos en Glassfish.

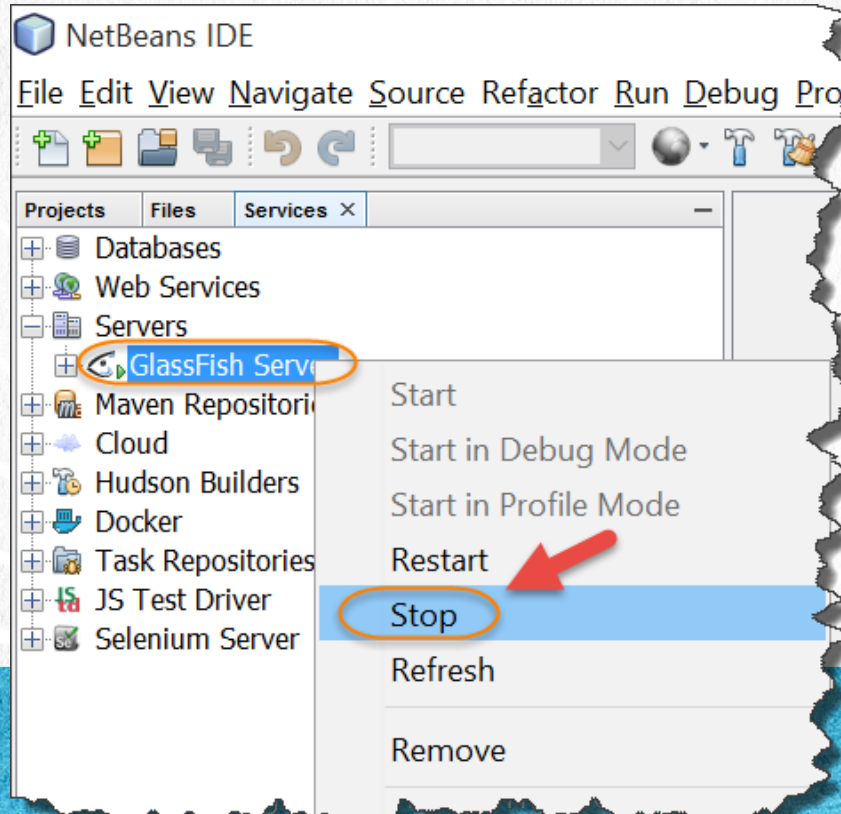


**CURSO JAVA EE**

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

# PASO 4. DETENEMOS GLASSFISH

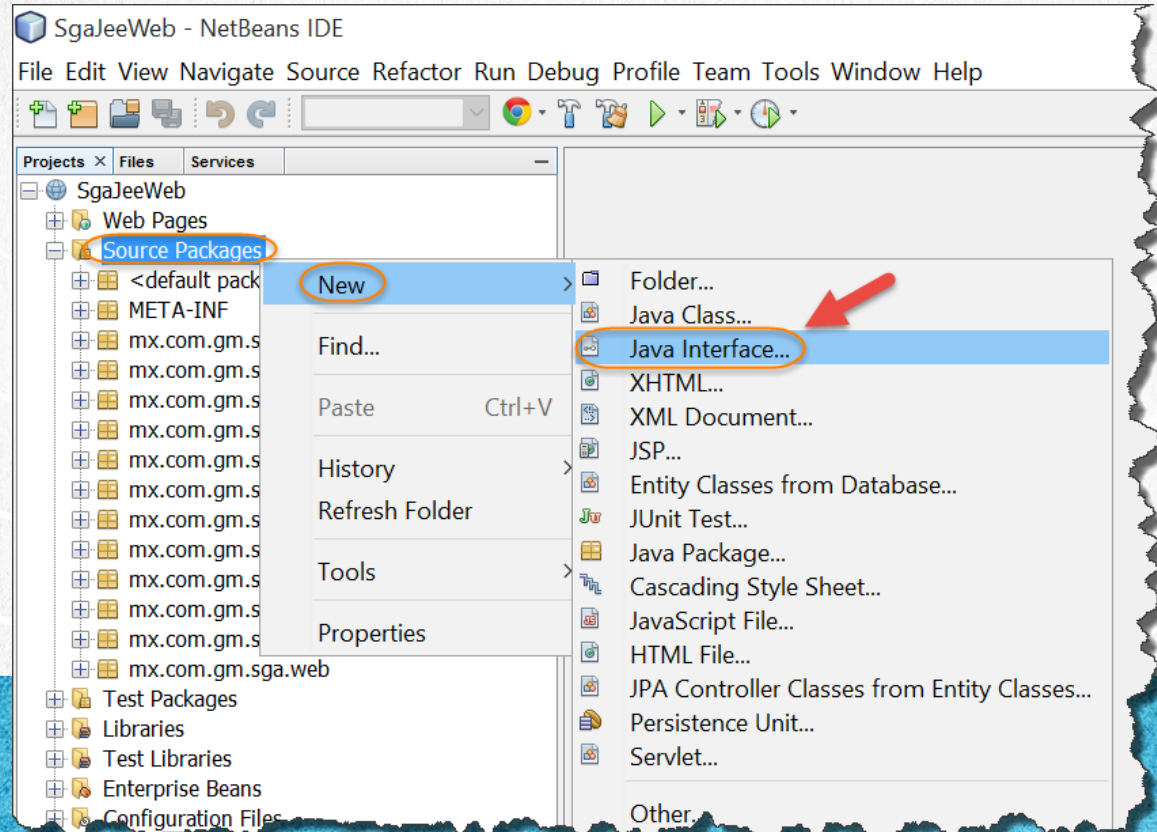
Detenemos el servidor de Glassfish:





# PASO 5. CREAMOS UN ARCHIVO JAVA

Creamos la interface PersonaServiceWS.java:



# PASO 5. CREAMOS UN ARCHIVO JAVA

Creamos la interface PersonaServiceWS.java:

New Java Interface

**Steps**

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

**Name and Location**

Class Name: PersonaServiceWS

Project: SgaJeeWeb

Location: Source Packages

Package: mx.com.gm.sga.servicio

Created File: E:\Leccion11\SgaJeeWeb\src\java\mx\com\gm\sga\servicio\PersonaServiceWS.java

< Back Next > **Finish** Cancel Help

**CURSO JAVA EE**

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



# PASO 6. MODIFICAMOS EL CÓDIGO

[Archivo PersonaServiceWs.java:](#)

Dar click para ir al código

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

import java.util.List;
import javax.jws.WebMethod;
import javax.jws.WebService;
import mx.com.gm.sga.domain.Persona;

@WebService
public interface PersonaServiceWS {

    @WebMethod
    public List<Persona> listarPersonas();
}
```

**CURSO DE JAVA EE**

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

# PASO 7. MODIFICAMOS UN ARCHIVO JAVA

Modificamos la clase PersonaServiceImpl, agregando los siguientes cambios:

1)Agregar la anotación WebService, especificando cual es la interface a utilizar:

```
@WebService(endpointInterface = "mx.com.gm.sga.servicio.PersonaServiceWS")
```

2)Extender de la interface PersonaServiceWS, por lo que la definición de la clase queda así:

```
public class PersonaServiceImpl  
    implements PersonaServiceRemote, PersonaService, PersonaServiceWS
```

3)Agregamos la anotación @Override en el método listarPersonas de la clase PersonaServiceImpl.



# PASO 7. MODIFICAMOS EL CÓDIGO

[Archivo PersonaServiceImpl.java:](#)

Dar click para ir al código

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

import java.util.List;
import javax.annotation.Resource;
import javax.ejb.EJB;
import javax.ejb.SessionContext;
import javax.ejb.Stateless;
import javax.jws.WebService;
import mx.com.gm.sga.domain.Persona;
import mx.com.gm.sga.eis.PersonaDao;

@Stateless
@WebService(endpointInterface = "mx.com.gm.sga.servicio.PersonaServiceWS")
public class PersonaServiceImpl
    implements PersonaServiceRemote, PersonaService, PersonaServiceWS {

    @Resource
    private SessionContext contexto;

    @EJB
    private PersonaDao personaDao;
```

# PASO 7. MODIFICAMOS EL CÓDIGO

[Archivo PersonaServiceImpl.java:](#)

Dar click para ir al código

```
@Override
public List<Persona> listarPersonas() {
    return personaDao.findAllPersonas();
}

@Override
public Persona encontrarPersonaPorId(Persona persona) {
    return personaDao.findPersonaById(persona);
}

@Override
public Persona encontrarPersonaPorEmail(Persona persona) {
    return personaDao.findPersonaByEmail(persona);
}

@Override
public void registrarPersona(Persona persona) {
    personaDao.insertPersona(persona);
}
```

**CURSO DE JAVA EE**

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



# PASO 7. MODIFICAMOS EL CÓDIGO

[Archivo PersonaServiceImpl.java:](#)

Dar click para ir al código

```
@Override
public void modificarPersona(Persona persona) {
    try{
        personaDao.updatePersona(persona);
    }catch(Throwable t){
        contexto.setRollbackOnly();
    }
}

@Override
public void eliminarPersona(Persona persona) {
    personaDao.deletePersona(persona);
}
}
```

**CURSO DE JAVA EE**

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

# PASO 8. MODIFICAMOS UN ARCHIVO JAVA

Modificamos la clase de dominio Persona, agregando los siguientes cambios:

1)Agregar la anotación `@XmlAccessorType` a nivel de la clase, especificando cual es la interface a utilizar: `@XmlAccessorType(XmlAccessType.FIELD)`

*Esta anotación del API de JAXB, significa que cada uno de los atributos de la clase se utilizará en el Web Service, y se validará con el esquema XSD, el cual es parte del mensaje Web Service.*

2)Excluir la relación con la colección de Usuarios. Para esto agregamos la anotación `@XmlTransient` al atributo usuarios de la clase Persona. Esta anotación debe ir antes de la anotación `@OneToMany` de dicho atributo: `@XmlTransient`

*Esta anotación la estamos agregando ya que la clase Persona tiene una relación con muchos Usuarios, y la clase Usuarios tiene una relación con una Persona, esto genera una relación circular, la cual el API de JAXB no soporta, para evitar este problema, indicamos que el atributo `private Set<Usuario> usuarios` no formará parte del envío del mensaje del Web Services. Esto no afecta a la configuración de JPA que también está incluida en la clase de dominio Persona y soporta sin problemas la navegación bidireccional.*



# PASO 8. MODIFICAMOS EL CÓDIGO

## Archivo Persona.java:

Dar click para ir al código

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

import java.io.Serializable;
import java.util.List;
import javax.persistence.Basic;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.OneToOne;
import javax.persistence.Table;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlTransient;
```

**CURSO DE JAVA EE**

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

# PASO 8. MODIFICAMOS EL CÓDIGO

## Archivo Persona.java:

Dar click para ir al código

```
@Entity
@Table(name = "persona")
@NamedQueries({
    @NamedQuery(name = "Persona.findAll", query = "SELECT p FROM Persona p")
    , @NamedQuery(name = "Persona.findByIdPersona", query = "SELECT p FROM Persona p WHERE p.idPersona = :idPersona")
    , @NamedQuery(name = "Persona.findByIdNombre", query = "SELECT p FROM Persona p WHERE p.nombre = :nombre")
    , @NamedQuery(name = "Persona.findByIdApellidoPaterno", query = "SELECT p FROM Persona p WHERE p.apellidoPaterno = :apellidoPaterno")
    , @NamedQuery(name = "Persona.findByIdApellidoMaterno", query = "SELECT p FROM Persona p WHERE p.apellidoMaterno = :apellidoMaterno")
    , @NamedQuery(name = "Persona.findByIdByEmail", query = "SELECT p FROM Persona p WHERE p.email = :email")
    , @NamedQuery(name = "Persona.findByIdByTelefono", query = "SELECT p FROM Persona p WHERE p.telefono = :telefono"))
@XmlAccessorType(XmlAccessType.FIELD)
public class Persona implements Serializable {

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

# PASO 8. MODIFICAMOS EL CÓDIGO

## Archivo Persona.java:

Dar click para ir al código

```
@Basic(optional = false)
@NotNull
@Size(min = 1, max = 45)
@Column(name = "nombre")
private String nombre;

@Basic(optional = false)
@NotNull
@Size(min = 1, max = 45)
@Column(name = "apellido_paterno")
private String apellidoPaterno;

@Size(max = 45)
@Column(name = "apellido_materno")
private String apellidoMaterno;

@Basic(optional = false)
@NotNull
@Size(min = 1, max = 45)
@Column(name = "email")
private String email;

@Size(max = 45)
@Column(name = "telefono")
private String telefono;
```



# PASO 8. MODIFICAMOS EL CÓDIGO

## Archivo Persona.java:

Dar click para ir al código

```
@XmlTransient
@OneToMany(mappedBy = "persona", cascade = CascadeType.ALL)
private List<Usuario> usuarios;

public Persona() {
}

public Persona(Integer idPersona) {
    this.idPersona = idPersona;
}

public Persona(String nombre, String apePaterno, String apeMaterno,
    String email, String telefono) {
    this.nombre = nombre;
    this.apellidoPaterno = apePaterno;
    this.apellidoMaterno = apeMaterno;
    this.email = email;
    this.telefono = telefono;
}

public Persona(Integer idPersona, String nombre, String apellidoPaterno, String email) {
    this.idPersona = idPersona;
    this.nombre = nombre;
    this.apellidoPaterno = apellidoPaterno;
    this.email = email;
}
```

# PASO 8. MODIFICAMOS EL CÓDIGO

## Archivo Persona.java:

Dar click para ir al código

```
public Persona(Integer idPersona, String nombre, String apellidoPaterno, String apellidoMaterno,
               String email, String telefono) {
    this.idPersona = idPersona;
    this.nombre = nombre;
    this.apellidoPaterno = apellidoPaterno;
    this.apellidoMaterno = apellidoMaterno;
    this.email = email;
    this.telefono = telefono;
}

public Integer getIdPersona() {
    return idPersona;
}

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

public String getNombre() {
    return nombre;
}

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

# PASO 8. MODIFICAMOS EL CÓDIGO

## Archivo Persona.java:

Dar click para ir al código

```
public String getApellidoPaterno() {  
    return apellidoPaterno;  
}  
  
public void setApellidoPaterno(String apellidoPaterno) {  
    this.apellidoPaterno = apellidoPaterno;  
}  
  
public String getApellidoMaterno() {  
    return apellidoMaterno;  
}  
  
public void setApellidoMaterno(String apellidoMaterno) {  
    this.apellidoMaterno = apellidoMaterno;  
}  
  
public String getEmail() {  
    return email;  
}  
  
public void setEmail(String email) {  
    this.email = email;  
}
```



# PASO 8. MODIFICAMOS EL CÓDIGO

## Archivo Persona.java:

Dar click para ir al código

```
public String getTelefono() {  
    return telefono;  
}  
  
public void setTelefono(String telefono) {  
    this.telefono = telefono;  
}  
  
public List<Usuario> getUsuarios() {  
    return usuarios;  
}  
  
public void setUsuarios(List<Usuario> usuarios) {  
    this.usuarios = usuarios;  
}  
  
@Override  
public int hashCode() {  
    int hash = 0;  
    hash += (idPersona != null ? idPersona.hashCode() : 0);  
    return hash;  
}
```

# PASO 8. MODIFICAMOS EL CÓDIGO

## Archivo Persona.java:

Dar click para ir al código

```
@Override
public boolean equals(Object object) {
    if (!(object instanceof Persona)) {
        return false;
    }
    Persona other = (Persona) object;
    if ((this.idPersona == null && other.idPersona != null) || (this.idPersona != null &&
!this.idPersona.equals(other.idPersona))) {
        return false;
    }
    return true;
}

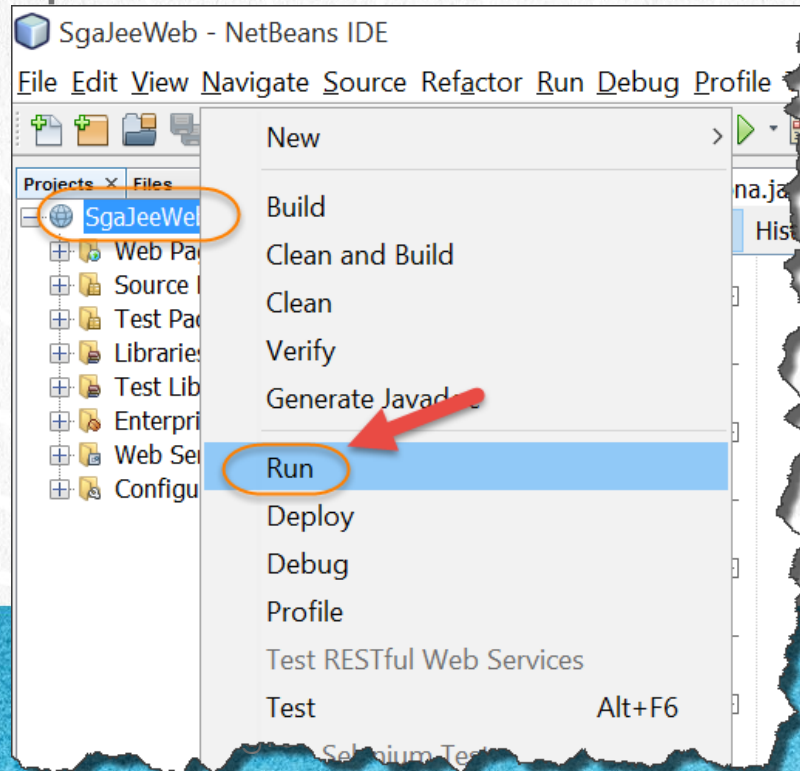
@Override
public String toString() {
    return "Persona{" + "idPersona=" + idPersona + ", nombre=" + nombre + ", apellidoPaterno=" +
apellidoPaterno + ", apellidoMaterno=" + apellidoMaterno + ", email=" + email + ", telefono=" + telefono + '}';
}
}
```

**CURSO DE JAVA EE**

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

# PASO 9. DESPLIEGUE EN GLASSFISH

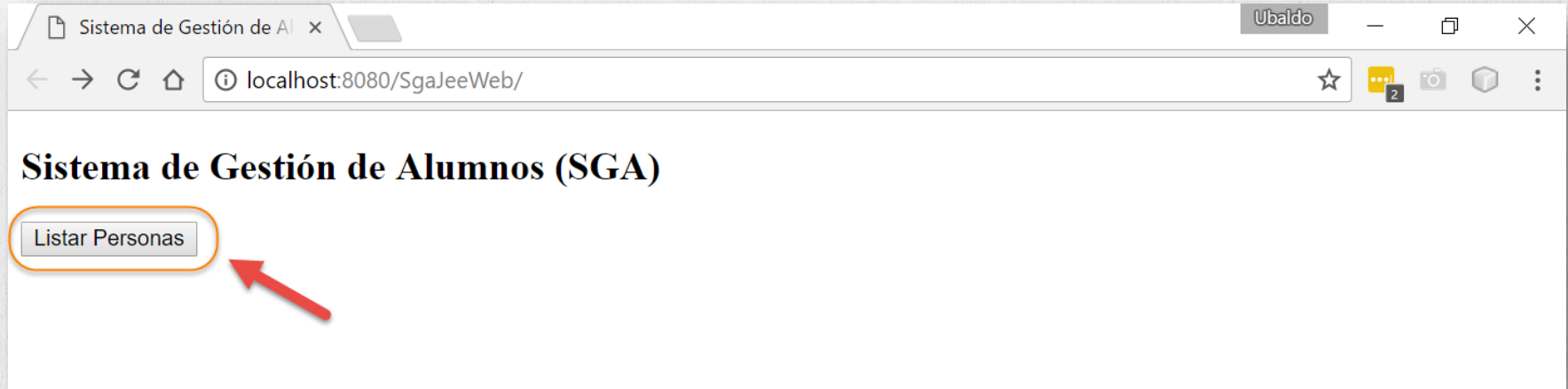
Ejecutamos la aplicación, y esto hará en automático un despliegue de la aplicación:





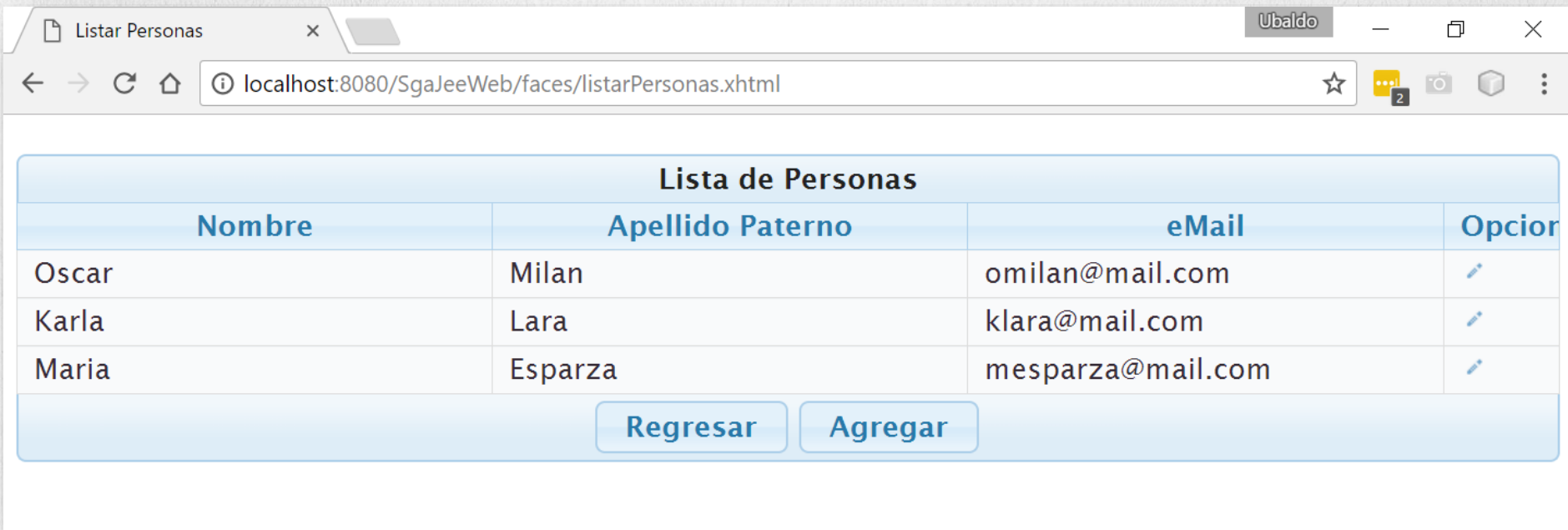
## PASO 9. DESPLIEGUE EN GLASSFISH

Ejecutamos la aplicación. Verificamos el listado de personas solo para conocer los datos que deberemos obtener en el cliente web service:






# PASO 9. DESPLIEGUE EN GLASSFISH

Ejecutamos la aplicación. Verificamos el listado de personas:



The screenshot shows a web browser window with the title 'Listar Personas'. The address bar displays 'localhost:8080/SgaJeeWeb/faces/listarPersonas.xhtml'. The page content features a table titled 'Lista de Personas' with four columns: 'Nombre', 'Apellido Paterno', 'eMail', and 'Opciones'. The table lists three individuals: Oscar Milan (omilan@mail.com), Karla Lara (klara@mail.com), and Maria Esparza (mesparza@mail.com). Each row has a small edit icon in the 'Opciones' column. Below the table are two buttons: 'Regresar' and 'Agregar'.

Lista de Personas			
Nombre	Apellido Paterno	eMail	Opciones
Oscar	Milan	omilan@mail.com	
Karla	Lara	klara@mail.com	
Maria	Esparza	mesparza@mail.com	

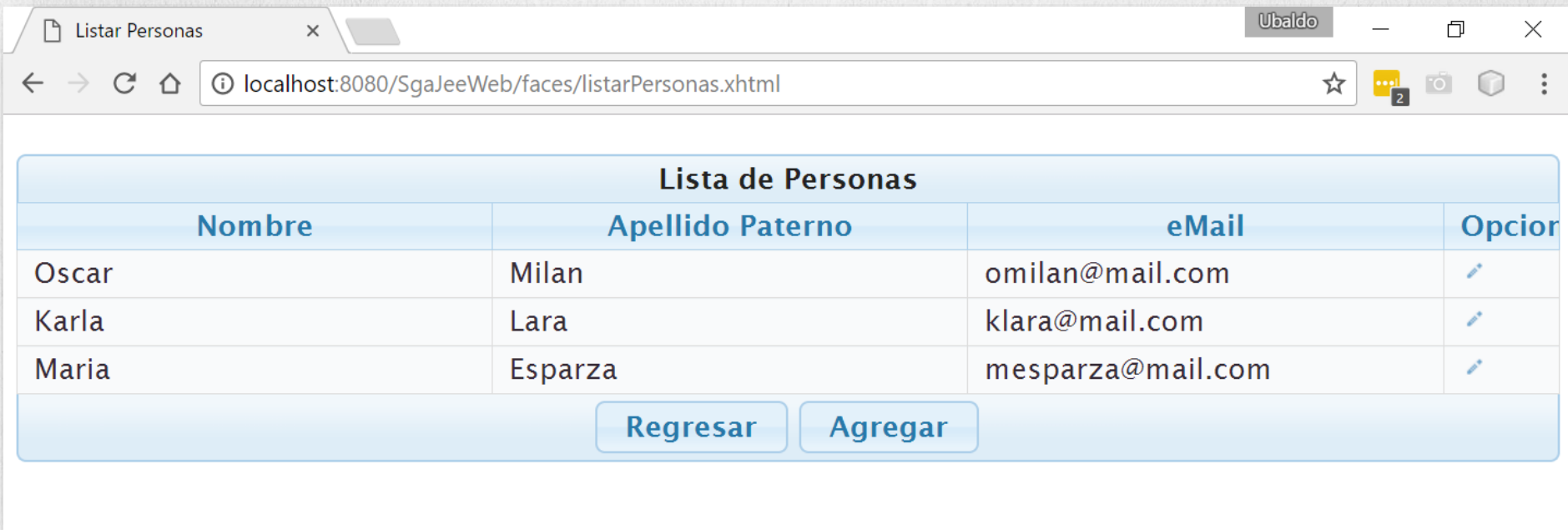
[Regresar](#) [Agregar](#)

**CURSO JAVA EE**




www.globalmentoring.com.mx

# PASO 9. DESPLIEGUE EN GLASSFISH

Ejecutamos la aplicación. Verificamos el listado de personas:



The screenshot shows a web browser window with the title 'Listar Personas'. The address bar displays 'localhost:8080/SgaJeeWeb/faces/listarPersonas.xhtml'. The page content features a table titled 'Lista de Personas' with four columns: 'Nombre', 'Apellido Paterno', 'eMail', and 'Opciones'. The table lists three individuals: Oscar Milan (omilan@mail.com), Karla Lara (klara@mail.com), and Maria Esparza (mesparza@mail.com). Each row has a small edit icon in the 'Opciones' column. Below the table are two buttons: 'Regresar' and 'Agregar'.

Lista de Personas			
Nombre	Apellido Paterno	eMail	Opciones
Oscar	Milan	omilan@mail.com	
Karla	Lara	klara@mail.com	
Maria	Esparza	mesparza@mail.com	

[Regresar](#) [Agregar](#)

**CURSO JAVA EE**

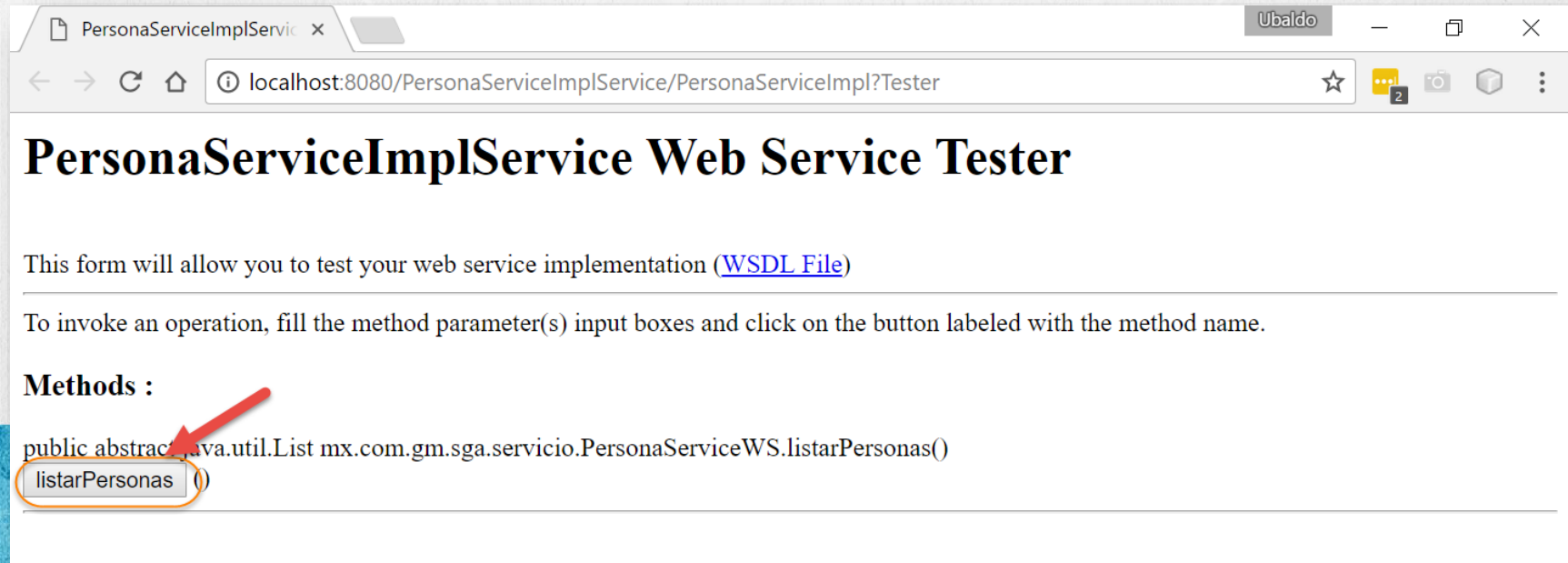
www.globalmentoring.com.mx



# PASO 10. CONSULTA DEL SERVICIO WEB

Una vez desplegada la aplicación en Glassfish, verificamos el servicio web de listadoPersonas publicado. Con la siguiente URLs se puede ejecutar el Test del Web Services. URL del Test:

<http://localhost:8080/PersonaServiceImplService/PersonaServiceImpl?Tester>



# PASO 10. CONSULTA DEL SERVICIO WEB

El resultado debe ser similar al siguiente:

← → ↶ ↷ local host:8080/PersonaServiceImplService/PersonaServiceImpl?testar

---

## listarPersonas Method invocation

---

Method parameter(s)

Type	Value
------	-------

---

Method returned

java.util.List : "[mx.com.gm.sga.servicio.Persona@114b08c0, mx.com.gm.sga.servicio.Persona@32a15

---

### SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap
  <SOAP-ENV:Header/>
  <S:Body>
    <ns2:listarPersonas xmlns:ns2="http://servicio.sga.gm.com.mx"/>
  </S:Body>
</S:Envelope>
```

---

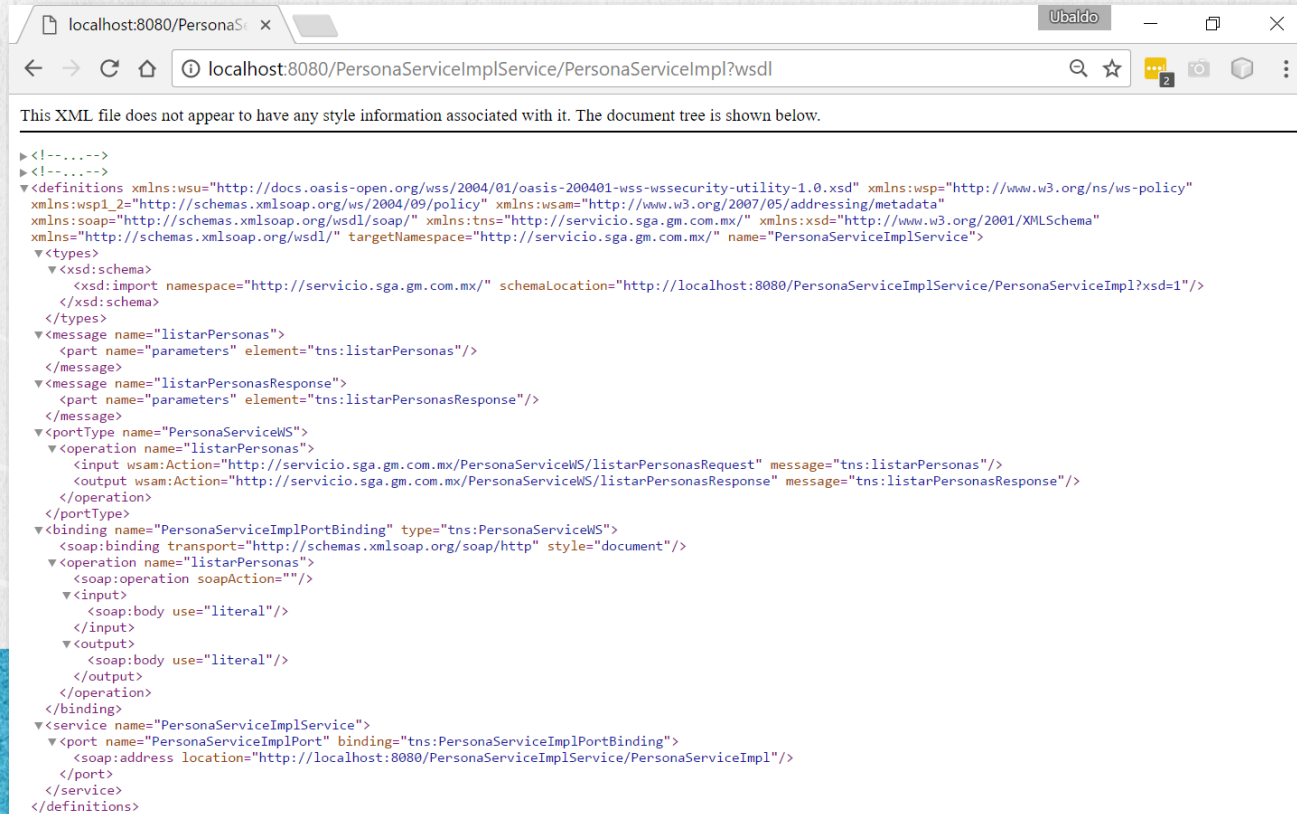
### SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap
  <SOAP-ENV:Header/>
  <S:Body>
    <ns2:listarPersonasResponse xmlns:ns2="http://servicio.sga.gm.com.mx/">
      <return>
        <idPersona>22</idPersona>
        <nombre>Oscar</nombre>
        <apellidoPaterno>Milan</apellidoPaterno>
        <apellidoMaterno>Lara</apellidoMaterno>
        <email>omilan@mail.com</email>
        <telefono>55112233</telefono>
      </return>
    </ns2:listarPersonasResponse>
  </S:Body>
</S:Envelope>
```

# PASO 10. CONSULTA DEL SERVICIO WEB

Revisión del documento wsdl:

<http://localhost:8080/PersonaServiceImplService/PersonaServiceImpl?wsdl>



The screenshot shows a web browser window with the address bar displaying the URL `localhost:8080/PersonaServiceImplService/PersonaServiceImpl?wsdl`. The browser's developer tools are open, showing the XML content of the WSDL file. The XML is a WSDL document for the `PersonaServiceImplService`. It includes imports for XSD and SOAP schemas, defines a `listarPersonas` operation, and specifies the service and port bindings. The XML is displayed with a tree view on the left and the full text on the right.

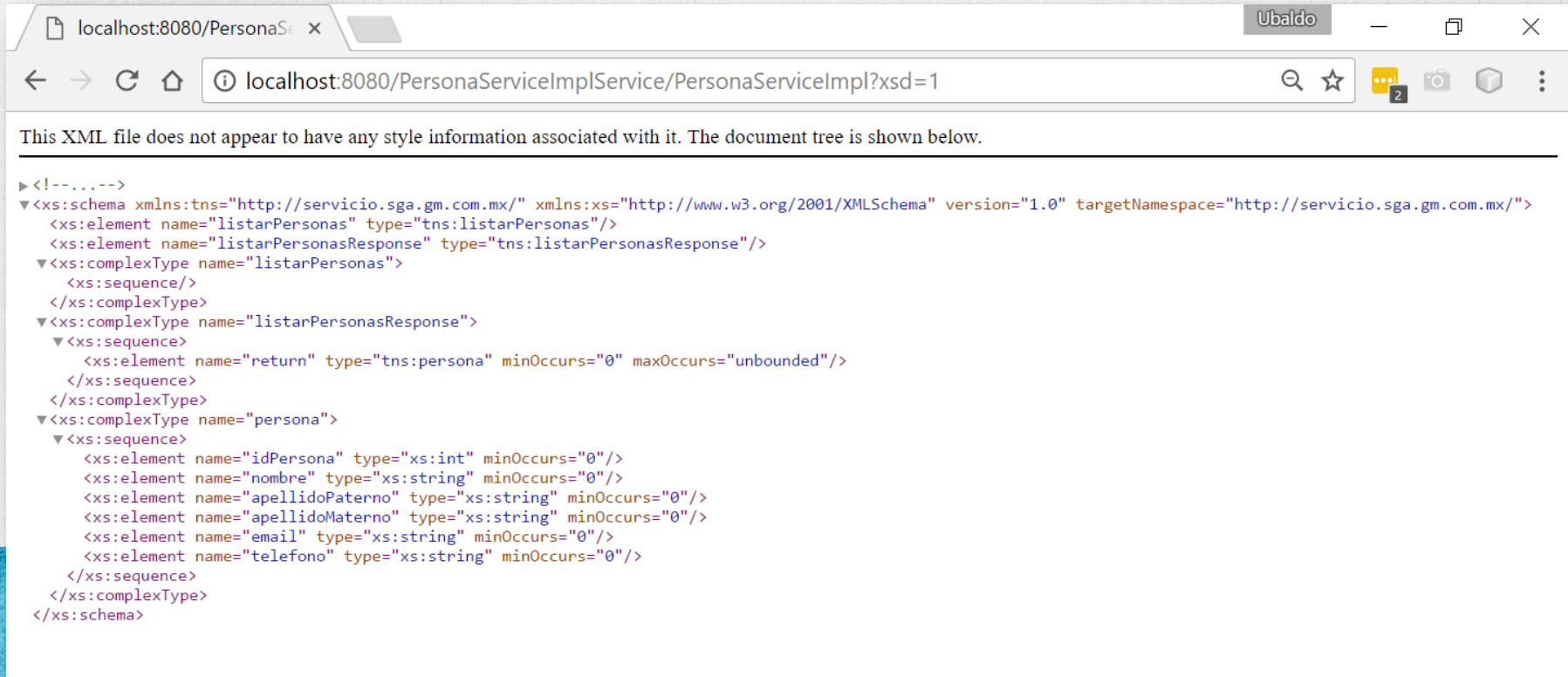
```
<!--...-->
<!--...-->
<?xml version='1.0' encoding='UTF-8'?>
<definitions xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd" xmlns:wsp="http://www.w3.org/ns/ws-policy"
xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy" xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://servicio.sga.gm.com.mx/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http://servicio.sga.gm.com.mx/" name="PersonaServiceImplService">
  <types>
    <xsd:schema>
      <xsd:import namespace="http://servicio.sga.gm.com.mx/" schemaLocation="http://localhost:8080/PersonaServiceImplService/PersonaServiceImpl?xsd=1"/>
    </xsd:schema>
  </types>
  <message name="listarPersonas">
    <part name="parameters" element="tns:listarPersonas"/>
  </message>
  <message name="listarPersonasResponse">
    <part name="parameters" element="tns:listarPersonasResponse"/>
  </message>
  <portType name="PersonaServiceWS">
    <operation name="listarPersonas">
      <input wsam:Action="http://servicio.sga.gm.com.mx/PersonaServiceWS/listarPersonasRequest" message="tns:listarPersonas"/>
      <output wsam:Action="http://servicio.sga.gm.com.mx/PersonaServiceWS/listarPersonasResponse" message="tns:listarPersonasResponse"/>
    </operation>
  </portType>
  <binding name="PersonaServiceImplPortBinding" type="tns:PersonaServiceWS">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    <operation name="listarPersonas">
      <soap:operation soapAction="">
        <input>
          <soap:body use="literal"/>
        </input>
        <output>
          <soap:body use="literal"/>
        </output>
      </operation>
    </binding>
  </service name="PersonaServiceImplService">
    <port name="PersonaServiceImplPort" binding="tns:PersonaServiceImplPortBinding">
      <soap:address location="http://localhost:8080/PersonaServiceImplService/PersonaServiceImpl"/>
    </port>
  </service>
</definitions>
```



# PASO 10. CONSULTA DEL SERVICIO WEB

Revisión del documento xsd:

<http://localhost:8080/PersonaServiceImplService/PersonaServiceImpl?xsd=1>



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<!--...-->
<xs:schema xmlns:tns="http://servicio.sga.gm.com.mx/" xmlns:xs="http://www.w3.org/2001/XMLSchema" version="1.0" targetNamespace="http://servicio.sga.gm.com.mx/">
  <xs:element name="listarPersonas" type="tns:listarPersonas"/>
  <xs:element name="listarPersonasResponse" type="tns:listarPersonasResponse"/>
  <xs:complexType name="listarPersonas">
    <xs:sequence/
  </xs:complexType>
  <xs:complexType name="listarPersonasResponse">
    <xs:sequence>
      <xs:element name="return" type="tns:persona" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="persona">
    <xs:sequence>
      <xs:element name="idPersona" type="xs:int" minOccurs="0"/>
      <xs:element name="nombre" type="xs:string" minOccurs="0"/>
      <xs:element name="apellidoPaterno" type="xs:string" minOccurs="0"/>
      <xs:element name="apellidoMaterno" type="xs:string" minOccurs="0"/>
      <xs:element name="email" type="xs:string" minOccurs="0"/>
      <xs:element name="telefono" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

# CONCLUSIÓN DEL EJERCICIO

Con este ejercicio hemos modificado nuestro proyecto SgaJeeWeb para agregar el servicio web de listado de personas.

Comprobamos que está disponible el WSDL, XSD e hicimos una prueba del servicio web de listado de personas.

En el siguiente ejercicio crearemos el cliente para consumir este servicio web de listado de personas.



Experiencia y Conocimiento para tu vida

**CURSO JAVA EE**

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



**CURSO ONLINE**

# **JAVA EMPRESARIAL JAVA EE**

---

Por: Ing. Ubaldo Acosta



Experiencia y Conocimiento para tu vida



**CURSO JAVA EE**

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