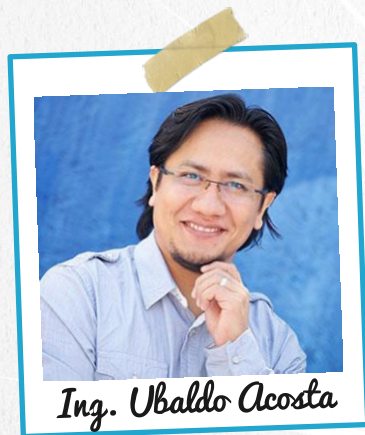


CURSO JAVA EE

HOLA MUNDO CON JAVA WEBSERVICES JAX-WS



Por el experto: Ing. Ubaldo Acosta



Experiencia y Conocimiento para tu vida

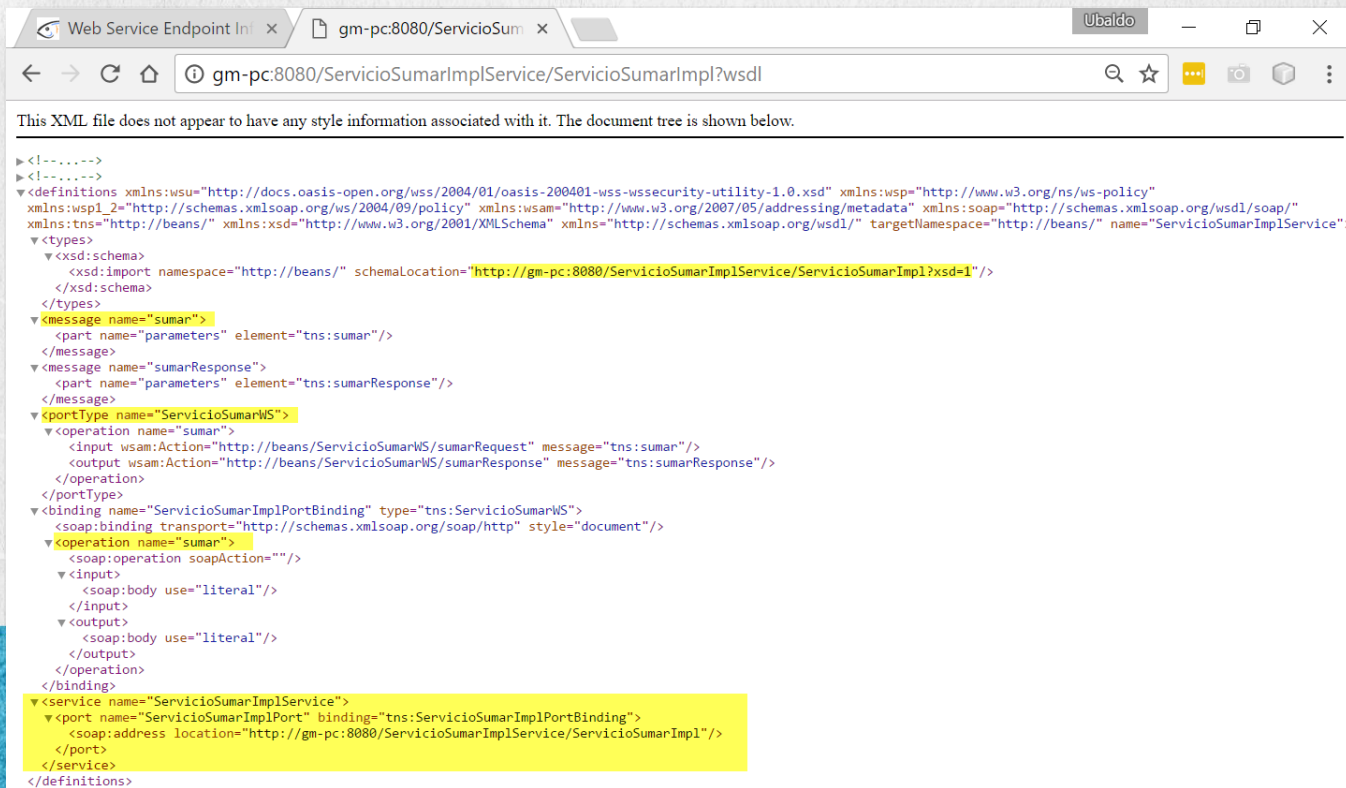


CURSO JAVA EE

www.globalmentoring.com.mx

OBJETIVO DEL EJERCICIO

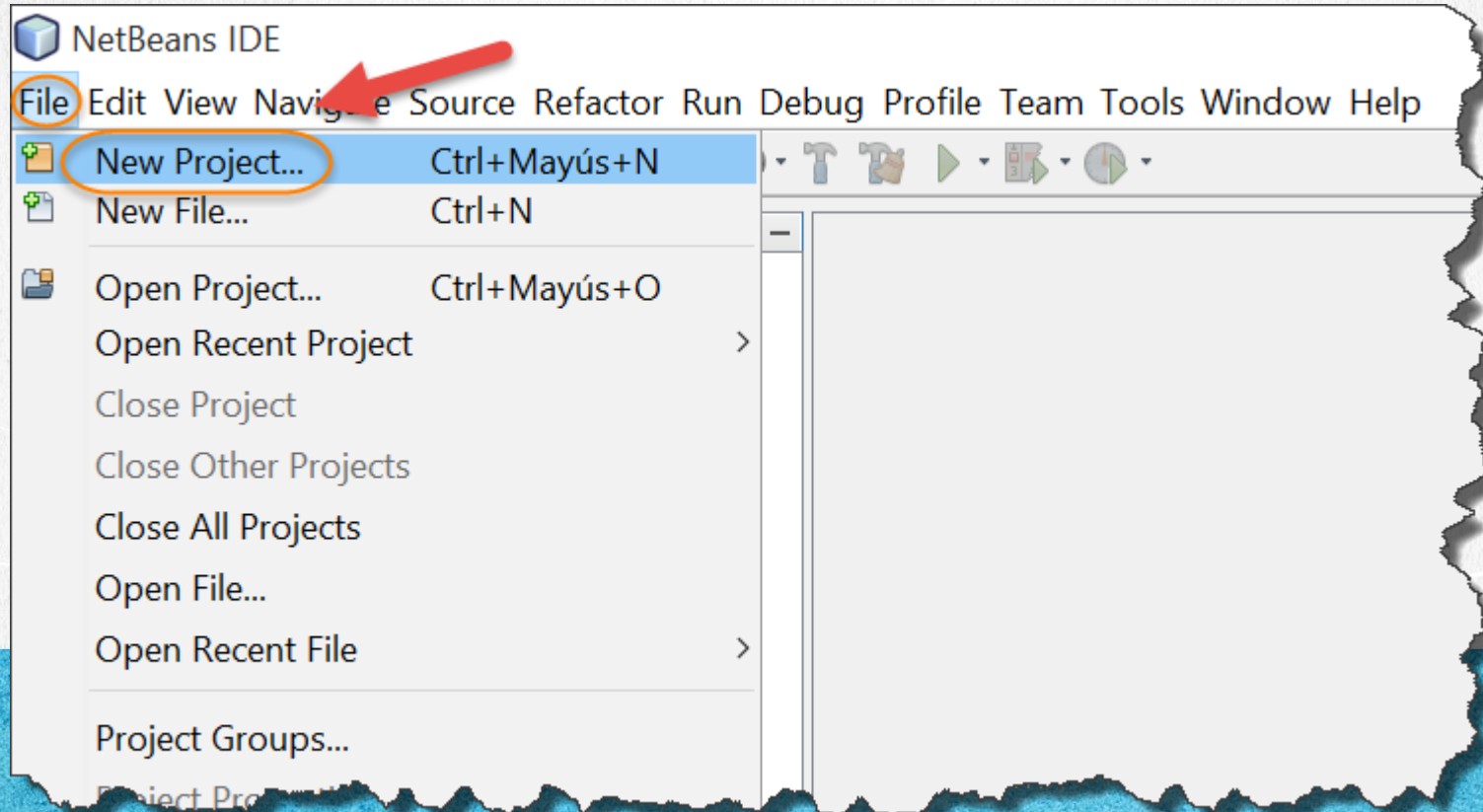
El objetivo del ejercicio crear nuestro primer Web Service con JAX-WS y EJBs de tipo Stateless Session Bean. El resultado se muestra a continuación:



```
<!--...-->
<!--...-->
<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://beans/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http://beans/" name="ServicioSumarImplService">
  <types>
    <xsd:schema>
      <xsd:import namespace="http://beans/" schemaLocation="http://gm-pc:8080/ServicioSumarImplService/ServicioSumarImpl?xsd=1"/>
    </xsd:schema>
  </types>
  <message name="sumar">
    <part name="parameters" element="tns:sumar"/>
  </message>
  <message name="sumarResponse">
    <part name="parameters" element="tns:sumarResponse"/>
  </message>
  <portType name="ServicioSumarWS">
    <operation name="sumar">
      <input wsam:Action="http://beans/ServicioSumarWS/sumarRequest" message="tns:sumar"/>
      <output wsam:Action="http://beans/ServicioSumarWS/sumarResponse" message="tns:sumarResponse"/>
    </operation>
  </portType>
  <binding name="ServicioSumarImplPortBinding" type="tns:ServicioSumarWS">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    <operation name="sumar">
      <soap:operation soapAction="">
        <input>
          <soap:body use="literal"/>
        </input>
        <output>
          <soap:body use="literal"/>
        </output>
      </operation>
    </binding>
  </service name="ServicioSumarImplService">
    <port name="ServicioSumarImplPort" binding="tns:ServicioSumarImplPortBinding">
      <soap:address location="http://gm-pc:8080/ServicioSumarImplService/ServicioSumarImpl"/>
    </port>
  </service>
</definitions>
```

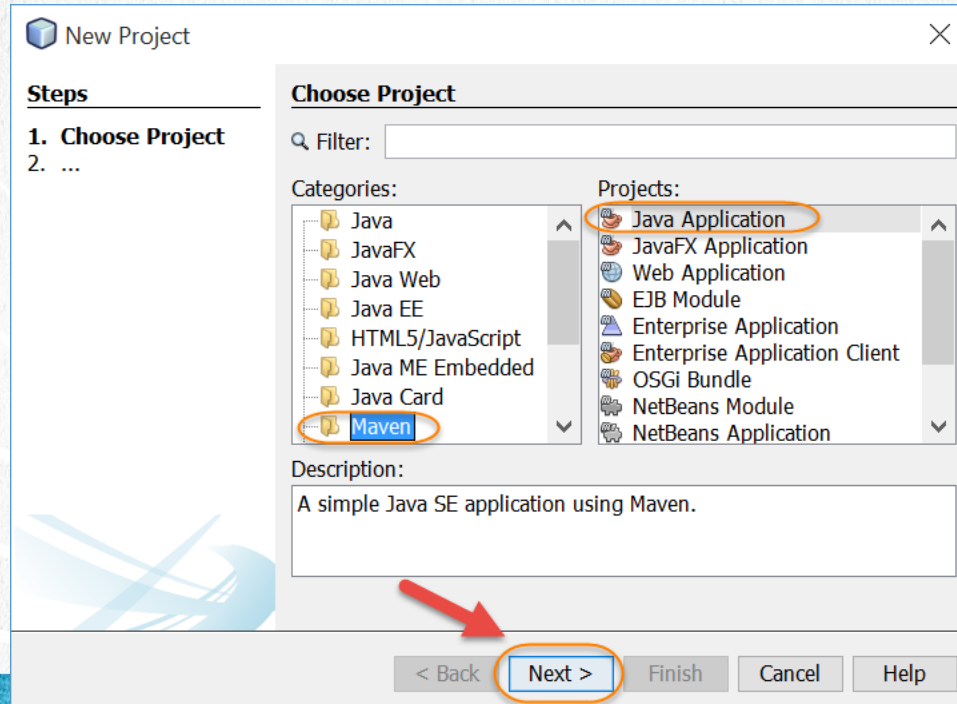
PASO 1. CREACIÓN DEL PROYECTO

Creamos el proyecto sumaws:



PASO 1. CREACIÓN DEL PROYECTO

Creamos el proyecto sumaws como un proyecto de maven:



CURSO JAVA EE

www.globalmentoring.com.mx

PASO 1. CREACIÓN DEL PROYECTO

Creamos el proyecto sumaws como un proyecto de maven:

The screenshot shows the 'New Java Application' dialog box with the following details:

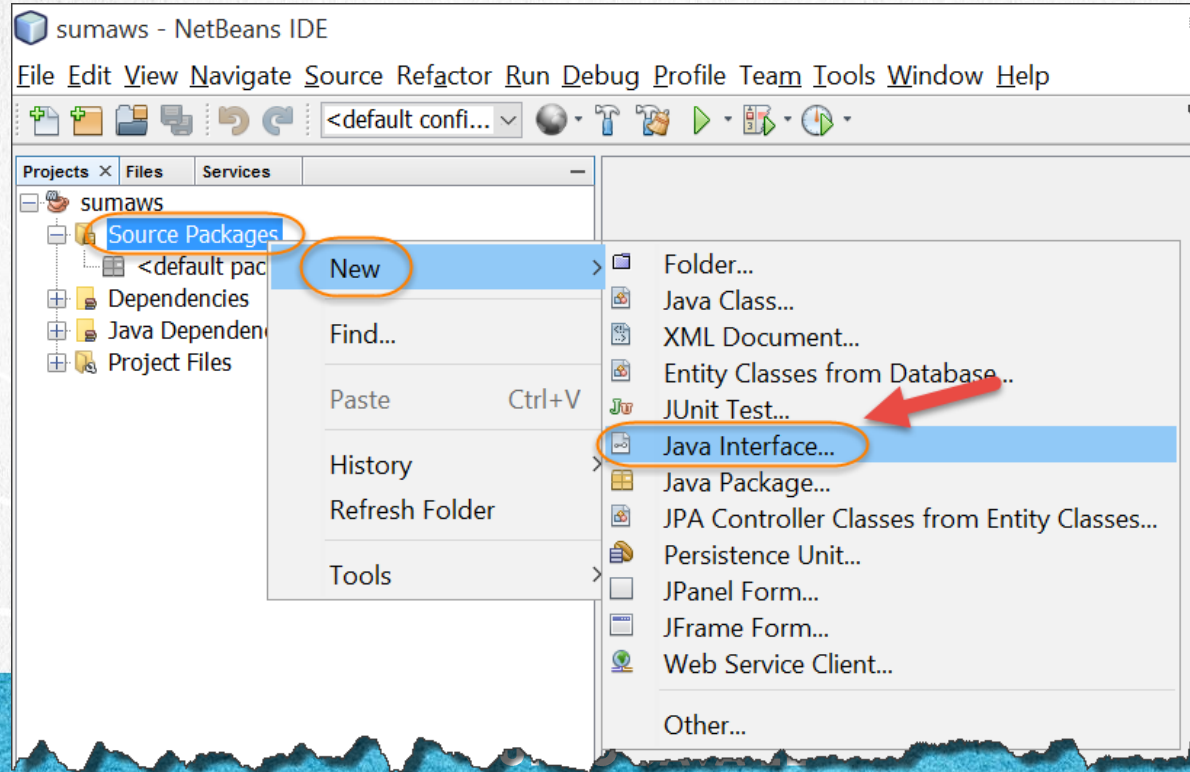
- Steps:**
 1. Choose Project
 2. Name and Location
- Name and Location:**
 - Project Name:** sumaws
 - Project Location:** C:\Cursos\JavaEE\Leccion11 (with a 'Browse...' button)
 - Project Folder:** C:\Cursos\JavaEE\Leccion11\sumaws
 - Artifact Id:** sumaws
 - Group Id:** mx.com.gm
 - Version:** 1.0
 - Package:** (Optional)
- Buttons:** < Back, Next >, **Finish** (highlighted with a red arrow), Cancel, Help

CURSO JAVA EE

www.globalmentoring.com.mx

PASO 2. CREACIÓN INTERFACE JAVA

Creamos la interface ServicioSumarWS.java:



PASO 2. CREACIÓN INTERFACE JAVA

Creamos la interface ServicioSumarWS.java:

New Java Interface

Steps

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

Name and Location

Class Name: ServicioSumarWS

Project: sumaws

Location: Source Packages

Package: beans

Created File: C:\Cursos\JavaEE\Leccion11\sumaws\src\main\java\beans\ServicioSumarWS.java

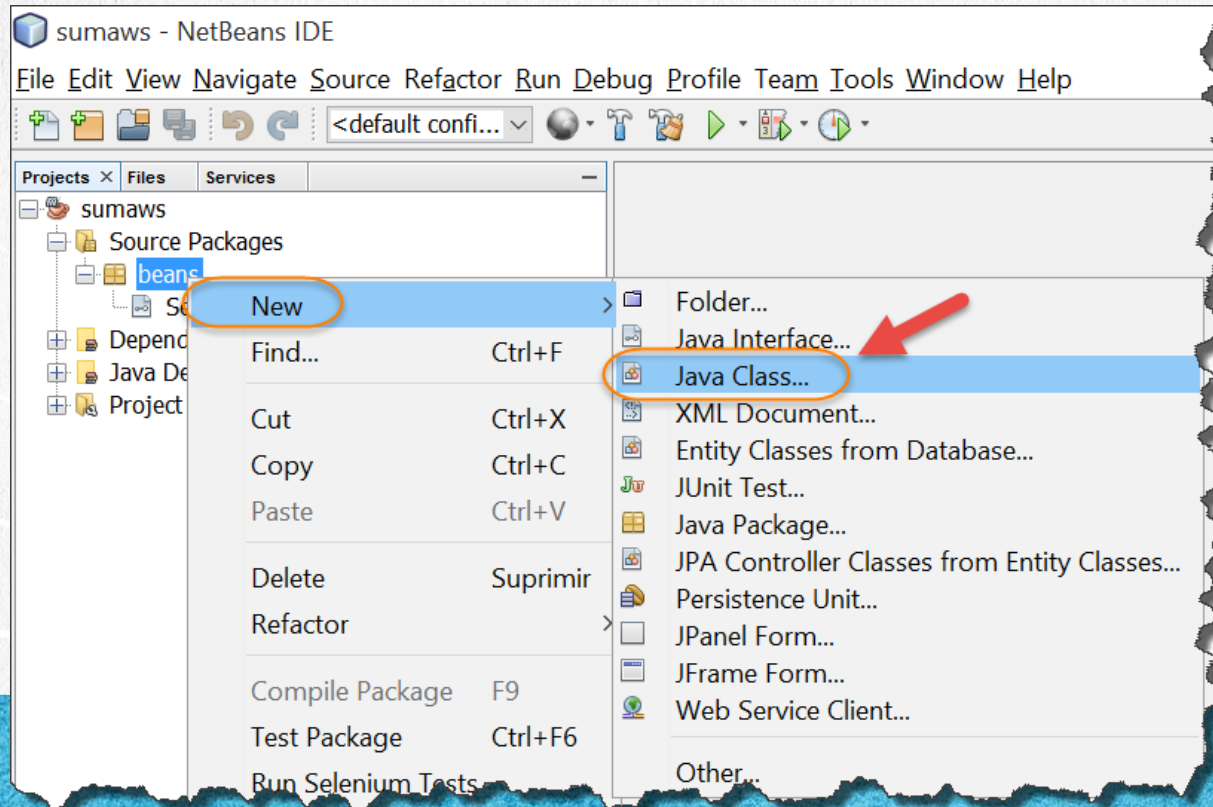
< Back Next > **Finish** Cancel Help

CURSO JAVA EE

www.globalmentoring.com.mx

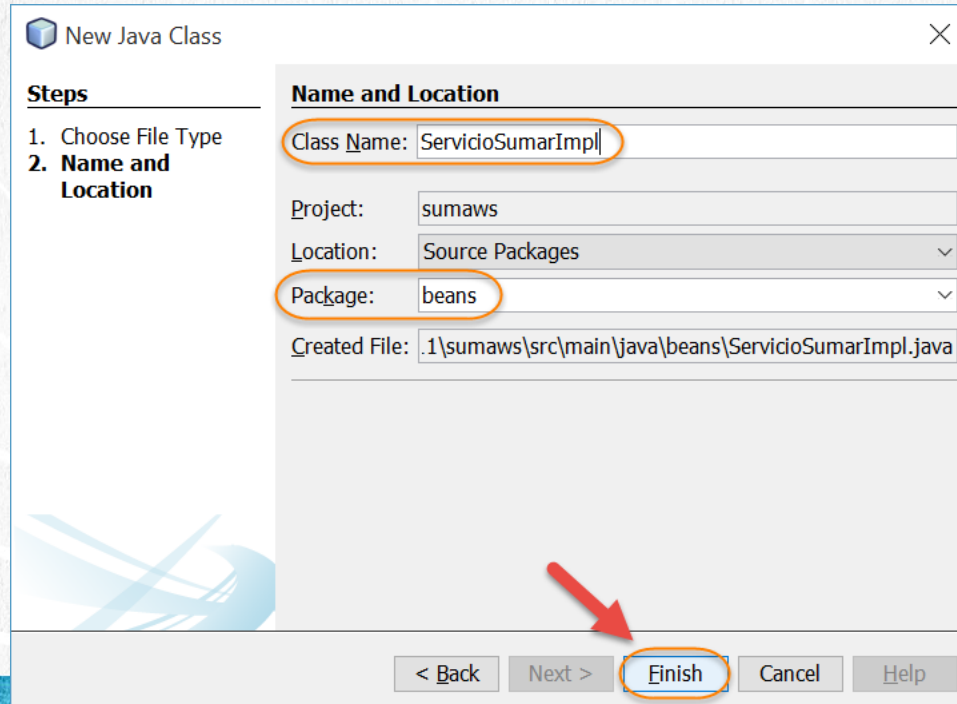
PASO 3. CREACIÓN CLASE JAVA

Creamos la clase ServicioSumarImpl.java:



PASO 3. CREACIÓN CLASE JAVA

Creamos la clase ServicioSumarImpl.java:



New Java Class

Steps

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

Name and Location

Class Name: ServicioSumarImpl

Project: sumaws

Location: Source Packages

Package: beans

Created File: .1\sumaws\src\main\java\beans\ServicioSumarImpl.java

< Back Next > **Finish** Cancel Help

CURSO JAVA EE

www.globalmentoring.com.mx

PASO 4. MODIFICAMOS EL CÓDIGO

Archivo pom.xml:

Dar click para ir al código

```
<?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>sumaws</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>javax</groupId>
      <artifactId>javaee-api</artifactId>
      <version>7.0</version>
      <scope>provided</scope>
    </dependency>
  </dependencies>
</project>
```

PASO 5. MODIFICAMOS EL CÓDIGO

[Archivo ServicioSumarWS.xml:](#)

Dar click para ir al código

```
package beans;

import javax.ws.WebMethod;
import javax.ws.WebService;

@WebService
public interface ServicioSumarWS {

    @WebMethod
    public int sumar(int a, int b);
}
```

CURSO JAVA EE

www.globalmentoring.com.mx

PASO 6. MODIFICAMOS EL CÓDIGO

Archivo ServicioSumarImpl.xml:

Dar click para ir al código

```
package beans;

import javax.ejb.Stateless;
import javax.jws.WebService;

@Stateless
@WebService(endpointInterface = "beans.ServicioSumarWS")
public class ServicioSumarImpl implements ServicioSumarWS {

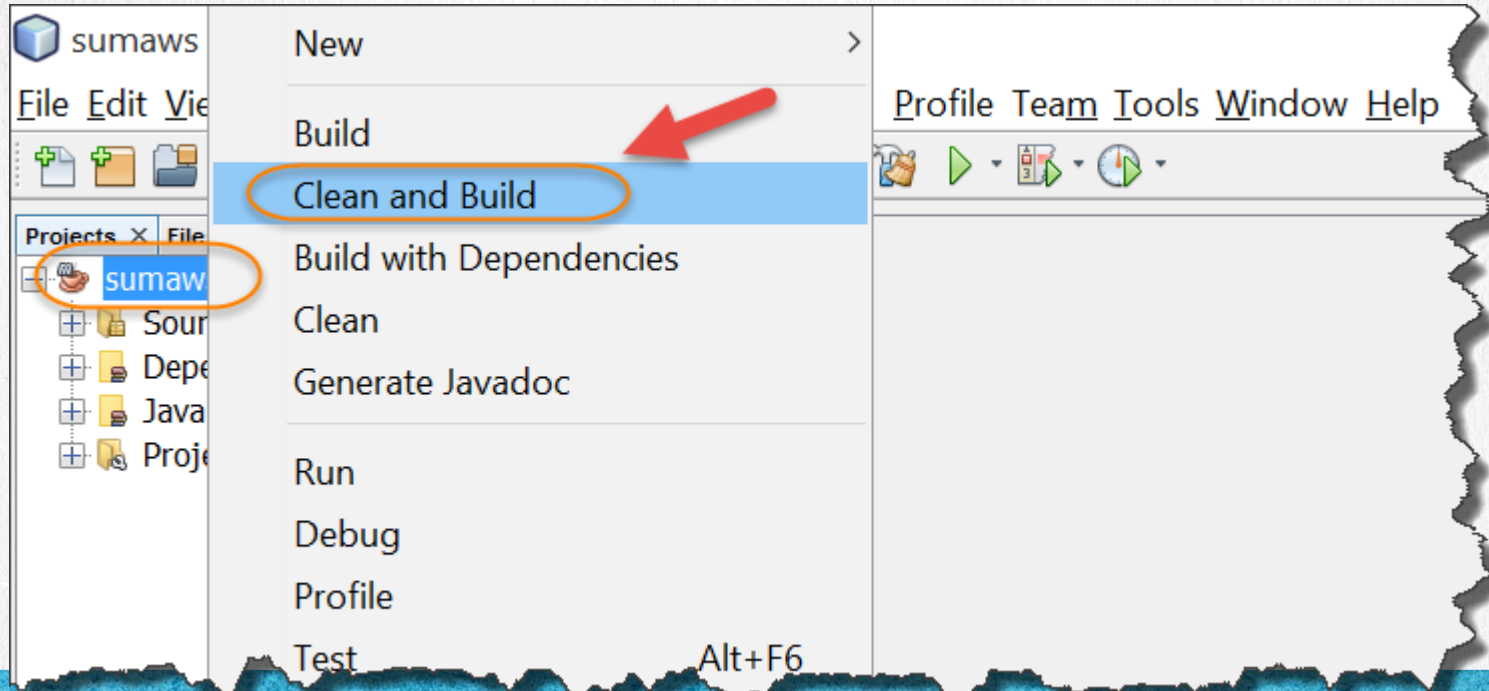
    @Override
    public int sumar(int a, int b) {
        return a + b;
    }
}
```

CURSO JAVA EE

www.globalmentoring.com.mx

PASO 7. CREAMOS EL ARCHIVO .JAR

Hacemos clean & build para crear el archivo .jar:

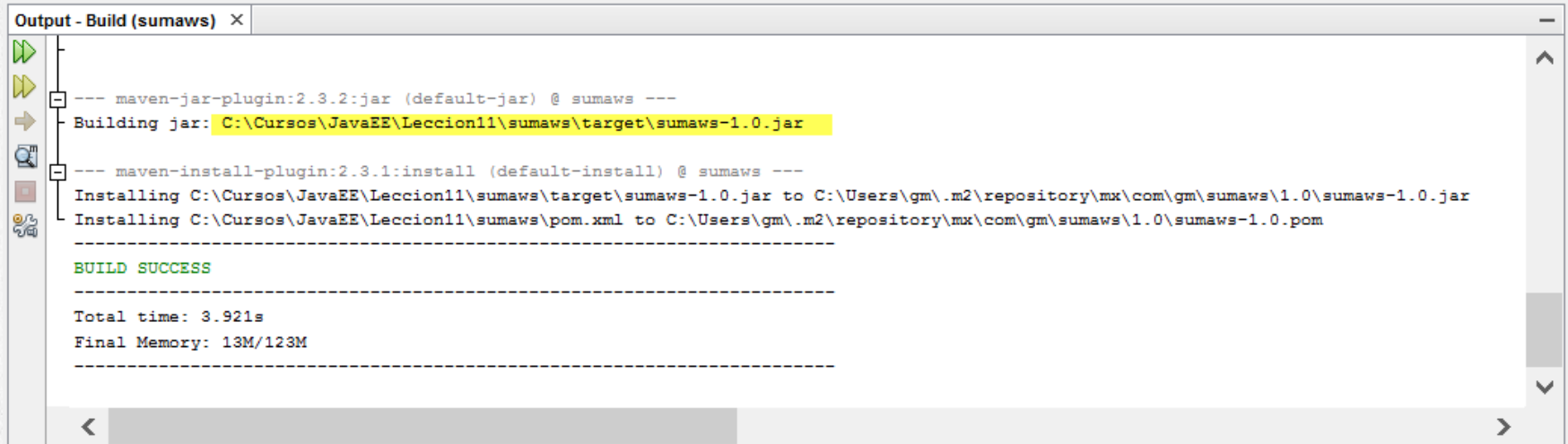


CURSO JAVA EE

www.globalmentoring.com.mx

PASO 7. CREAMOS EL ARCHIVO .JAR

Hacemos clean & build para crear el archivo .jar:



The screenshot shows the 'Output - Build (sumaws)' window in an IDE. The window contains the following text:

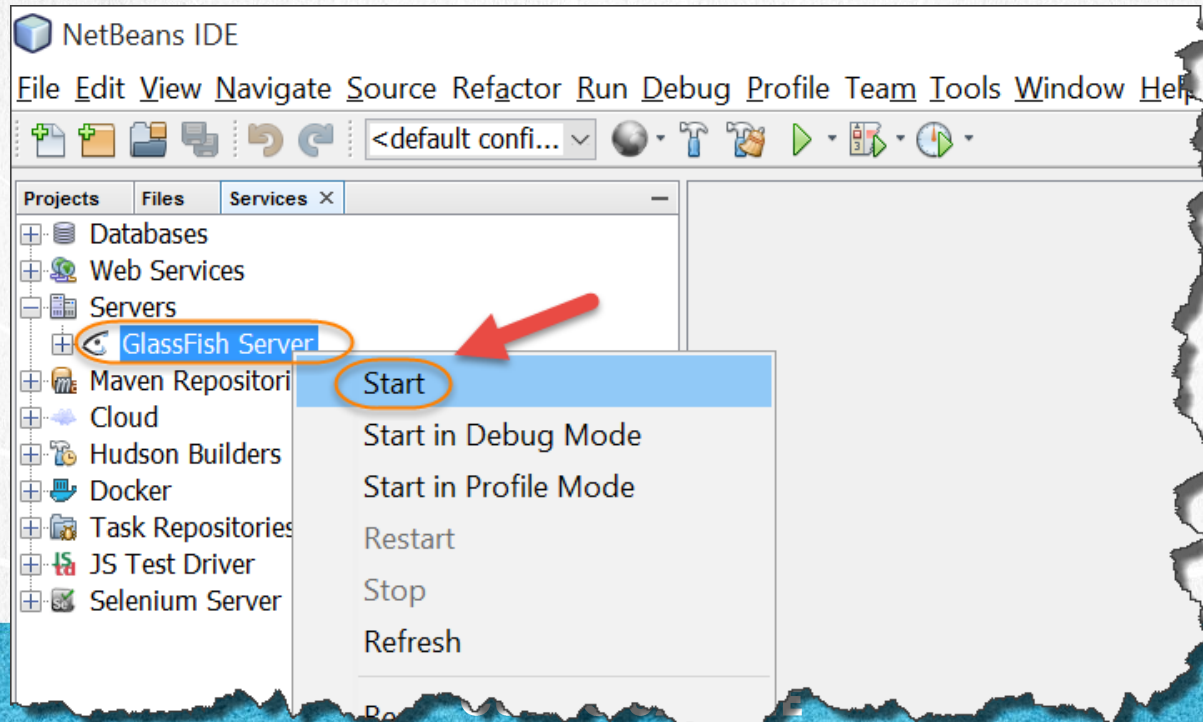
```
--- maven-jar-plugin:2.3.2:jar (default-jar) @ sumaws ---  
Building jar: C:\Cursos\JavaEE\Leccion11\sumaws\target\sumaws-1.0.jar  
--- maven-install-plugin:2.3.1:install (default-install) @ sumaws ---  
Installing C:\Cursos\JavaEE\Leccion11\sumaws\target\sumaws-1.0.jar to C:\Users\gm\.m2\repository\mx\com\gm\sumaws\1.0\sumaws-1.0.jar  
Installing C:\Cursos\JavaEE\Leccion11\sumaws\pom.xml to C:\Users\gm\.m2\repository\mx\com\gm\sumaws\1.0\sumaws-1.0.pom  
  
BUILD SUCCESS  
  
Total time: 3.921s  
Final Memory: 13M/123M
```

CURSO JAVA EE

www.globalmentoring.com.mx

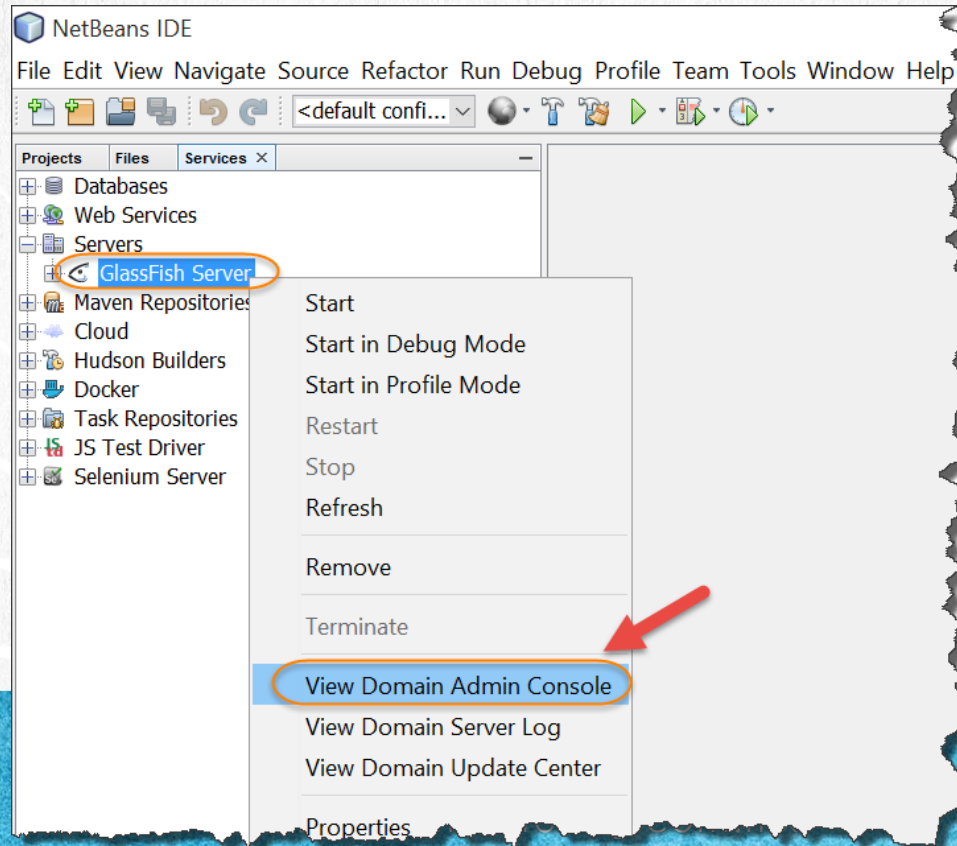
PASO 8. DESPLEGAMOS EL ARCHIVO .JAR

Desplegamos el archivo .jar generado en Glassfish. Levantamos el servidor de Glassfish en caso de que esté apagado:



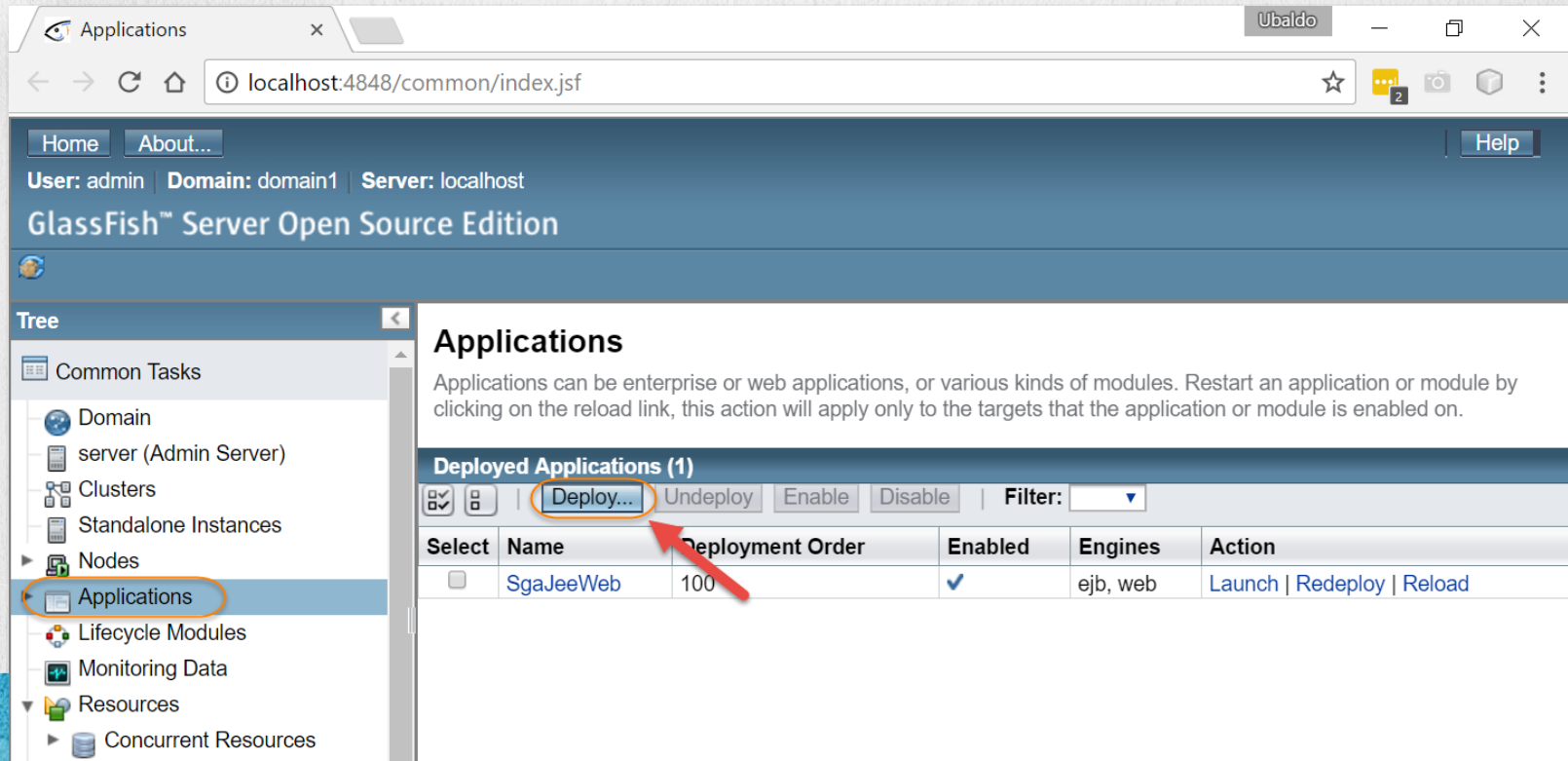
PASO 8. DESPLEGAMOS EL ARCHIVO .JAR

Desplegamos el archivo .jar generado en Glassfish. Entramos a la consola de Glassfish:



PASO 8. DESPLEGAMOS EL ARCHIVO .JAR

Desplegamos el archivo .jar generado en Glassfish.



The screenshot shows the GlassFish Server Open Source Edition web console. The browser address bar indicates the URL is `localhost:4848/common/index.jsf`. The page title is "GlassFish™ Server Open Source Edition". The left sidebar shows a tree view with "Applications" selected. The main content area is titled "Applications" and contains a table of deployed applications. The table has columns for "Select", "Name", "Deployment Order", "Enabled", "Engines", and "Action". One application, "SgaJeeWeb", is listed with a deployment order of 100 and is enabled. The "Deploy..." button is highlighted with a red circle and a red arrow pointing to it.

Applications

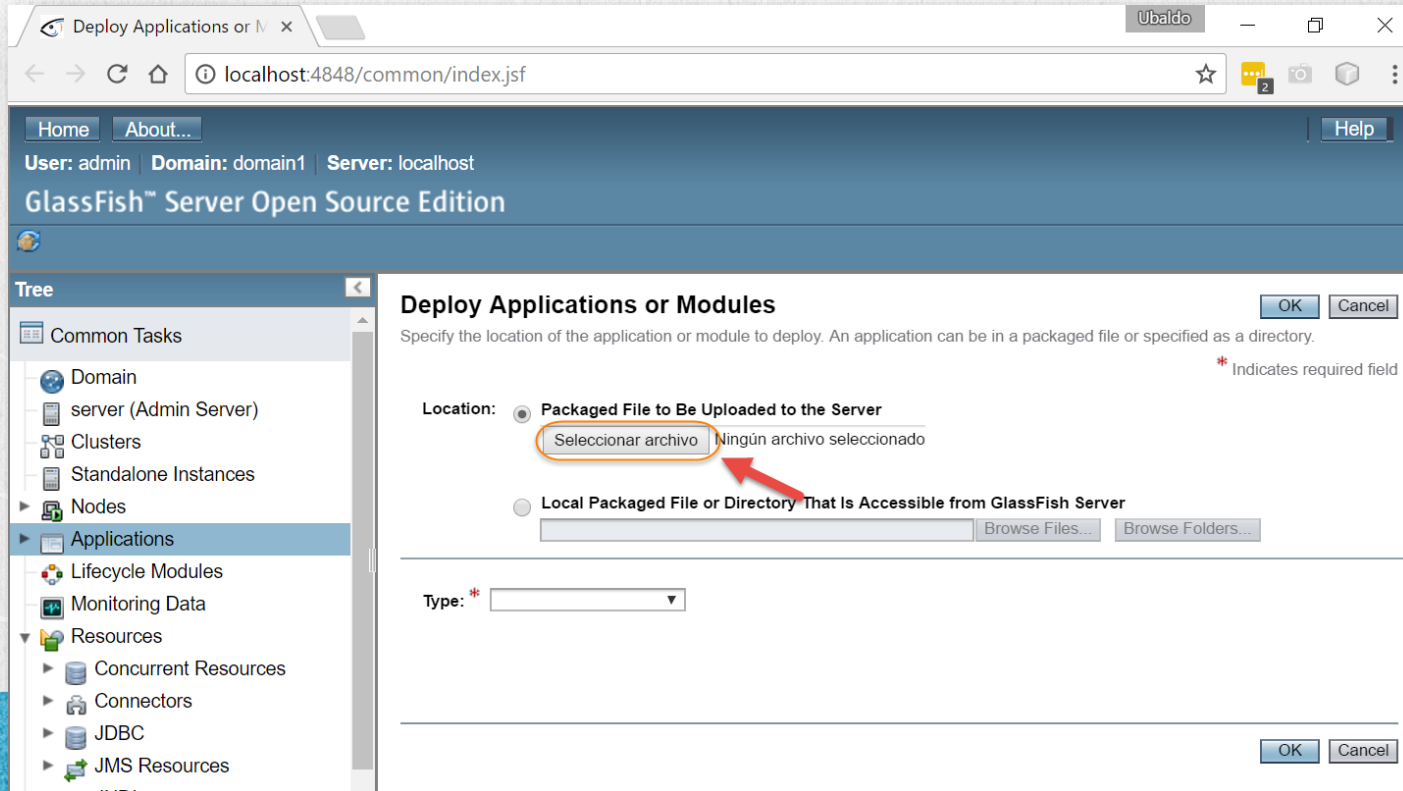
Applications can be enterprise or web applications, or various kinds of modules. Restart an application or module by clicking on the reload link, this action will apply only to the targets that the application or module is enabled on.

Deployed Applications (1)

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

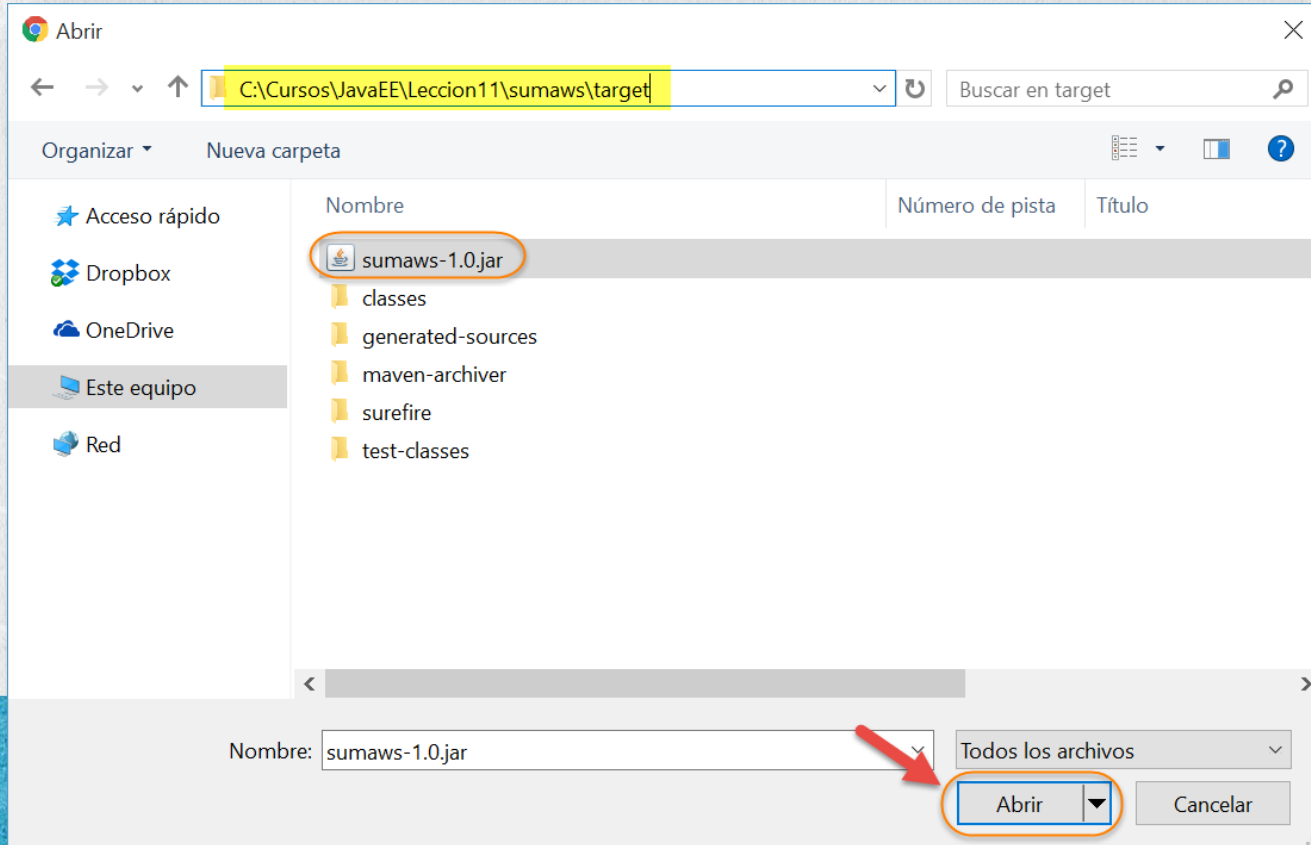
PASO 8. DESPLEGAMOS EL ARCHIVO .JAR

Desplegamos el archivo .jar generado en Glassfish.



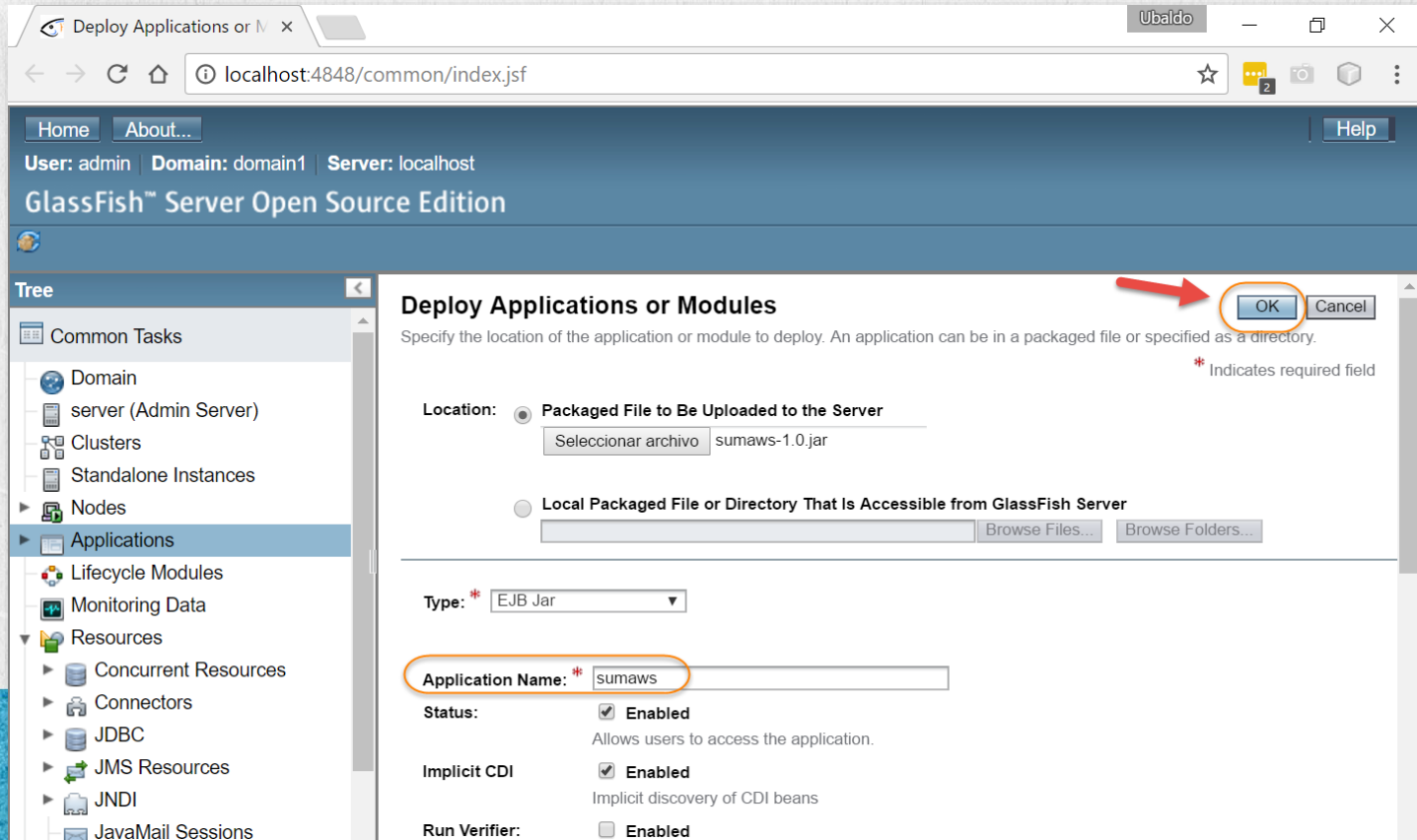
PASO 8. DESPLEGAMOS EL ARCHIVO .JAR

Desplegamos el archivo .jar generado en Glassfish.



PASO 8. DESPLEGAMOS EL ARCHIVO .JAR

Desplegamos el archivo .jar generado en Glassfish.



The screenshot shows the GlassFish Server Administration console. The left sidebar contains a tree view with the following items: Common Tasks, Domain, server (Admin Server), Clusters, Standalone Instances, Nodes, Applications (selected), Lifecycle Modules, Monitoring Data, Resources, Concurrent Resources, Connectors, JDBC, JMS Resources, JNDI, and JavaMail Sessions. The main content area is titled 'Deploy Applications or Modules' and contains the following fields and options:

- Location:** ☒ Packaged File to Be Uploaded to the Server. Below this is a text field 'Seleccionar archivo' with the value 'sumaws-1.0.jar'.
- ☐ Local Packaged File or Directory That Is Accessible from GlassFish Server. Below this are 'Browse Files...' and 'Browse Folders...' buttons.
- Type:** A dropdown menu showing 'EJB Jar'.
- Application Name:** A text field containing 'sumaws'.
- Status:** ☒ Enabled. Below it, the text 'Allows users to access the application.'
- Implicit CDI:** ☒ Enabled. Below it, the text 'Implicit discovery of CDI beans'.
- Run Verifier:** ☐ Enabled.

At the top right of the main content area, there are 'OK' and 'Cancel' buttons. A red arrow points to the 'OK' button. A small asterisk (*) indicates required fields.

PASO 9. VERIFICAMOS EL CLIENTE WS Y EL WSDL

Verificamos el cliente WS y el WSDL generado:

Applications

User: admin | Domain: domain1 | Server: localhost

GlassFish™ Server Open Source Edition

Tree

- Common Tasks
- Domain
 - server (Admin Server)
 - Clusters
 - Standalone Instances
 - Nodes
 - Applications**
 - Lifecycle Modules
 - Monitoring Data
 - Resources
 - Concurrent Resources

Applications

Applications can be enterprise or web applications, or various kinds of modules. Restart an application or module by clicking on the reload link, this action will apply only to the targets that the application or module is enabled on.

Deployed Applications (2)

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

PASO 9. VERIFICAMOS EL CLIENTE WS Y EL WSDL

Verificamos el cliente WS y el WSDL generado:

The screenshot shows the GlassFish Server Open Source Edition administration console. The browser address bar indicates the URL is `localhost:4848/common/index.jsf`. The page title is "GlassFish™ Server Open Source Edition". The user is logged in as "admin" on the "domain1" domain, with the server running on "localhost".

The left sidebar shows a tree view of the server configuration. The "Applications" section is expanded, and the "sumaws" application is selected.

The main content area displays the configuration for the "sumaws" application:

- Name:** sumaws
- Status:** ☒ Enabled
- Implicit CDI:** ☒ Enabled
Implicit discovery of CDI beans
- Location:** `${com.sun.aas.instanceRootURI}/applications/sumaws/`
- Deployment Order:** 100
A number that determines the loading order of the application at server startup. Lower numbers are loaded first. The default is 100.
- Libraries:**
- Description:**

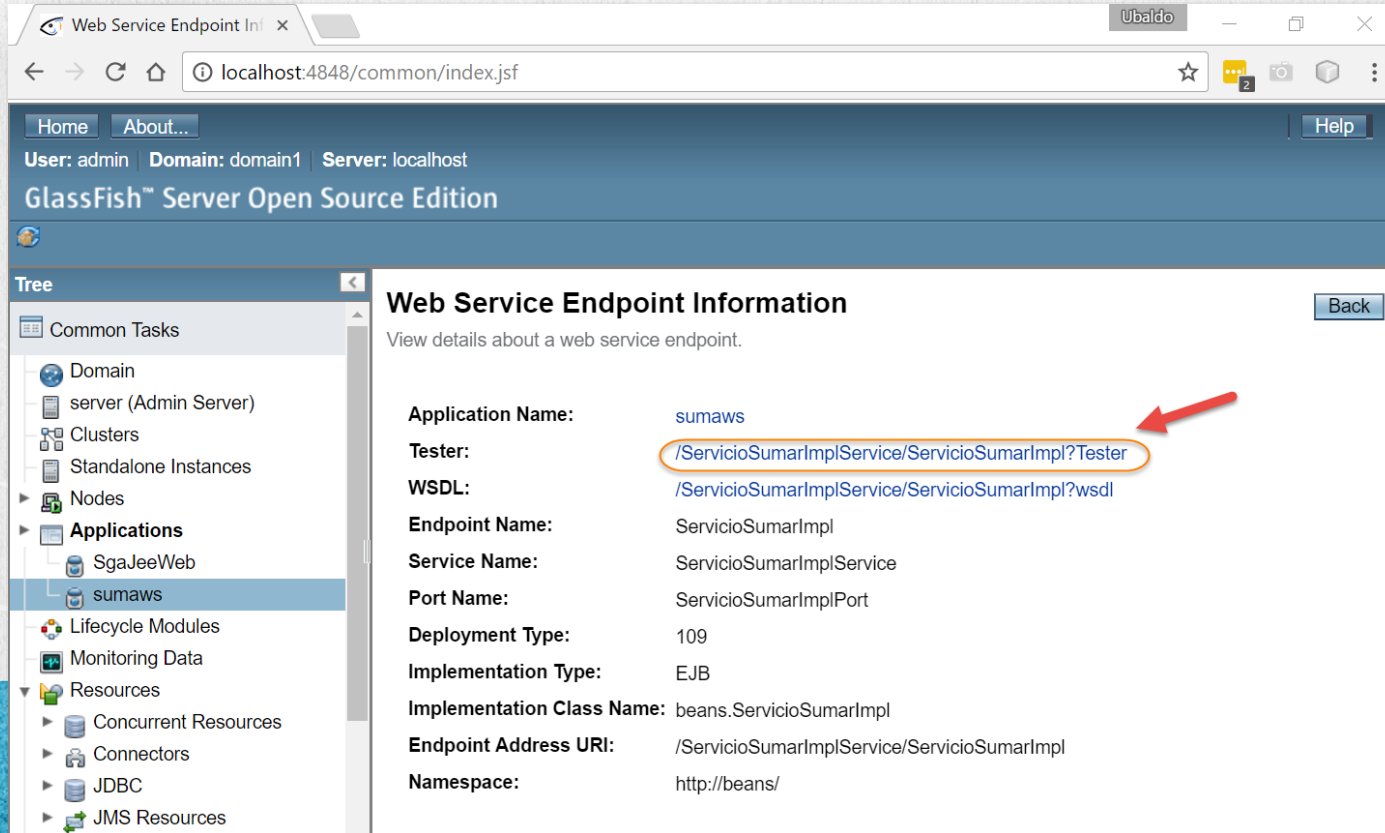
Below the configuration fields is a table titled "Modules and Components (2)":

Module Name	Engines	Component Name	Type	Action
sumaws	[ejb, webservices, weld]	-----	-----	
sumaws		ServicioSumarImpl	StatelessSessionBean	View Endpoint

A red arrow points to the "View Endpoint" link in the "Action" column of the table.

PASO 9. VERIFICAMOS EL CLIENTE WS Y EL WSDL

Verificamos el cliente WS y el WSDL generado:



The screenshot shows the GlassFish Web Service Endpoint Information page. The left sidebar contains a tree view with the following structure:

- Tree
 - Common Tasks
 - Domain
 - server (Admin Server)
 - Clusters
 - Standalone Instances
 - Nodes
 - Applications
 - SgaJeeWeb
 - sumaws
 - Lifecycle Modules
 - Monitoring Data
 - Resources
 - Concurrent Resources
 - Connectors
 - JDBC
 - JMS Resources

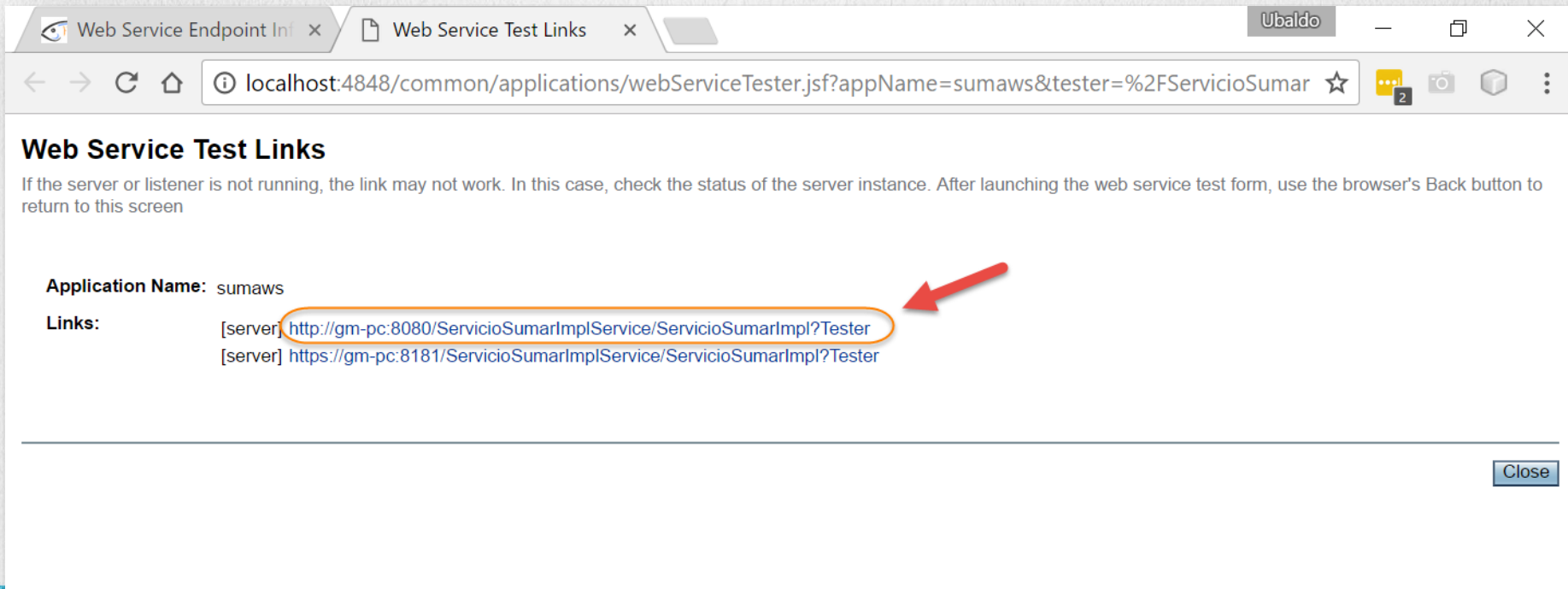
The main content area displays the following information:

Application Name:	sumaws
Tester:	/ServicioSumarImplService/ServicioSumarImpl?Tester
WSDL:	/ServicioSumarImplService/ServicioSumarImpl?wsdl
Endpoint Name:	ServicioSumarImpl
Service Name:	ServicioSumarImplService
Port Name:	ServicioSumarImplPort
Deployment Type:	109
Implementation Type:	EJB
Implementation Class Name:	beans.ServicioSumarImpl
Endpoint Address URI:	/ServicioSumarImplService/ServicioSumarImpl
Namespace:	http://beans/

A red arrow points to the 'Tester' field value.

PASO 9. VERIFICAMOS EL CLIENTE WS Y EL WSDL

Verificamos el cliente WS y el WSDL generado:



Web Service Test Links

If the server or listener is not running, the link may not work. In this case, check the status of the server instance. After launching the web service test form, use the browser's Back button to return to this screen

Application Name: sumaws

Links:

- [server] <http://gm-pc:8080/ServicioSumarImplService/ServicioSumarImpl?Tester>
- [server] <https://gm-pc:8181/ServicioSumarImplService/ServicioSumarImpl?Tester>

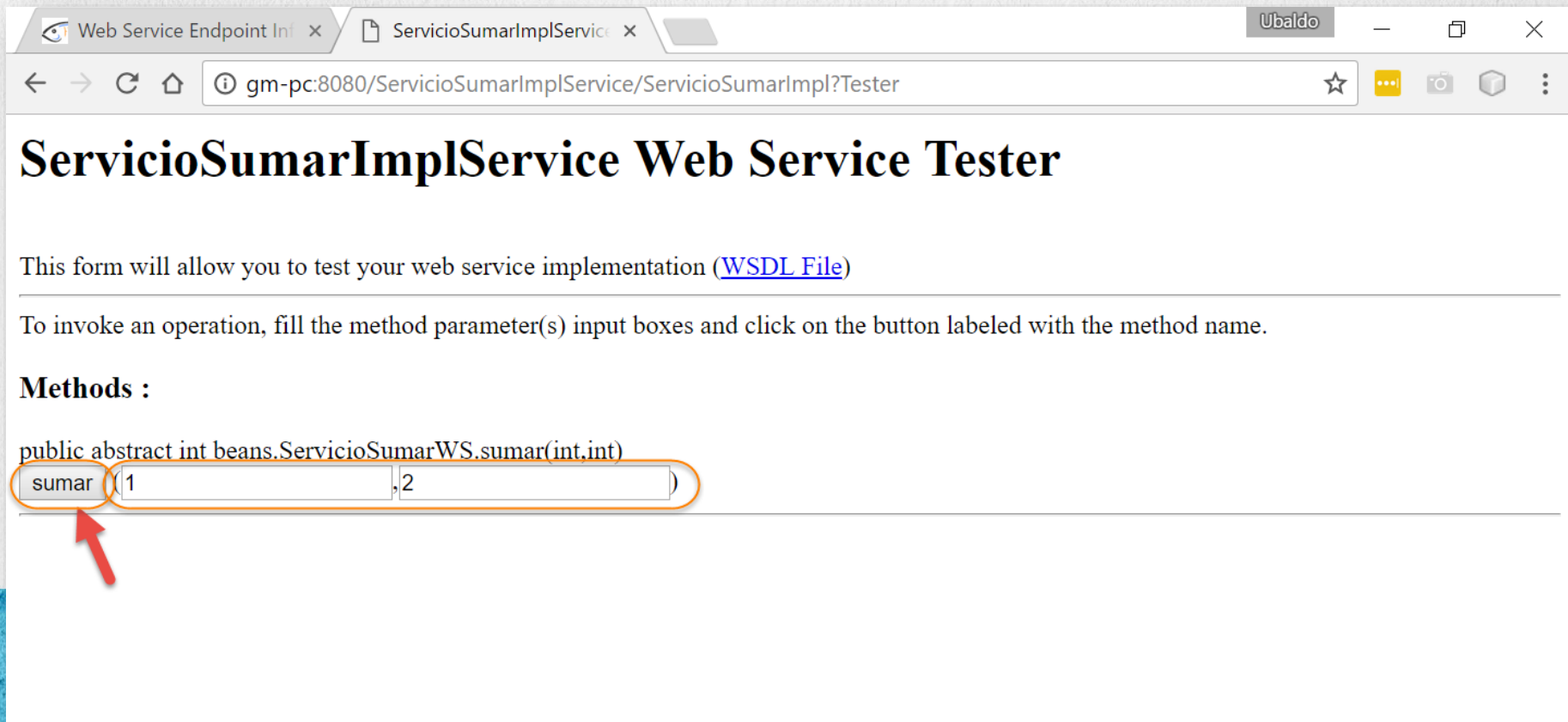
Close

CURSO JAVA EE

www.globalmentoring.com.mx

PASO 9. VERIFICAMOS EL CLIENTE WS Y EL WSDL

Verificamos el cliente WS y el WSDL generado:



Web Service Endpoint Int x ServicioSumarImplService x Ubaldo

gm-pc:8080/ServicioSumarImplService/ServicioSumarImpl?Tester

ServicioSumarImplService Web Service Tester

This form will allow you to test your web service implementation ([WSDL File](#))

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

```
public abstract int beans.ServicioSumarWS.sumar(int,int)
```

sumar (1, 2)

PASO 9. VERIFICAMOS EL CLIENTE WS Y EL WSDL

Verificamos el cliente WS y el WSDL generado:

Web Service Endpoint Int x Method invocation trace x Ubaldo

gm-pc:8080/ServicioSumarImplService/ServicioSumarImpl?Tester

Method parameter(s)

Type	Value
int	1
int	2

Method returned

int : "3"

SOAP Request

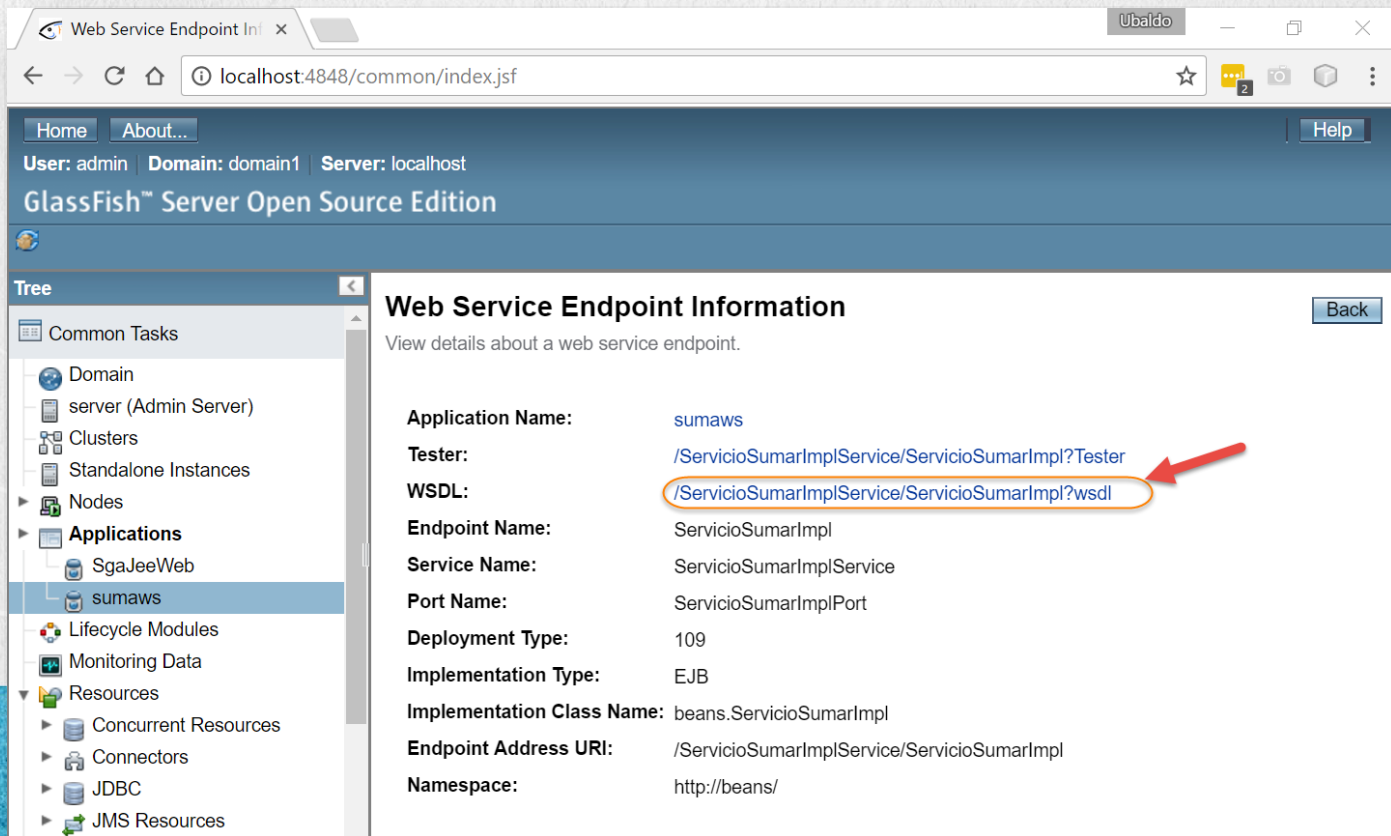
```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <S:Body>
    <ns2:sumar xmlns:ns2="http://beans/">
      <arg0>1</arg0>
      <arg1>2</arg1>
    </ns2:sumar>
  </S:Body>
</S:Envelope>
```

SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <S:Body>
    <ns2:sumarResponse xmlns:ns2="http://beans/">
      <return>3</return>
    </ns2:sumarResponse>
  </S:Body>
</S:Envelope>
```


PASO 9. VERIFICAMOS EL CLIENTE WS Y EL WSDL

Verificamos el cliente WS y el WSDL generado:

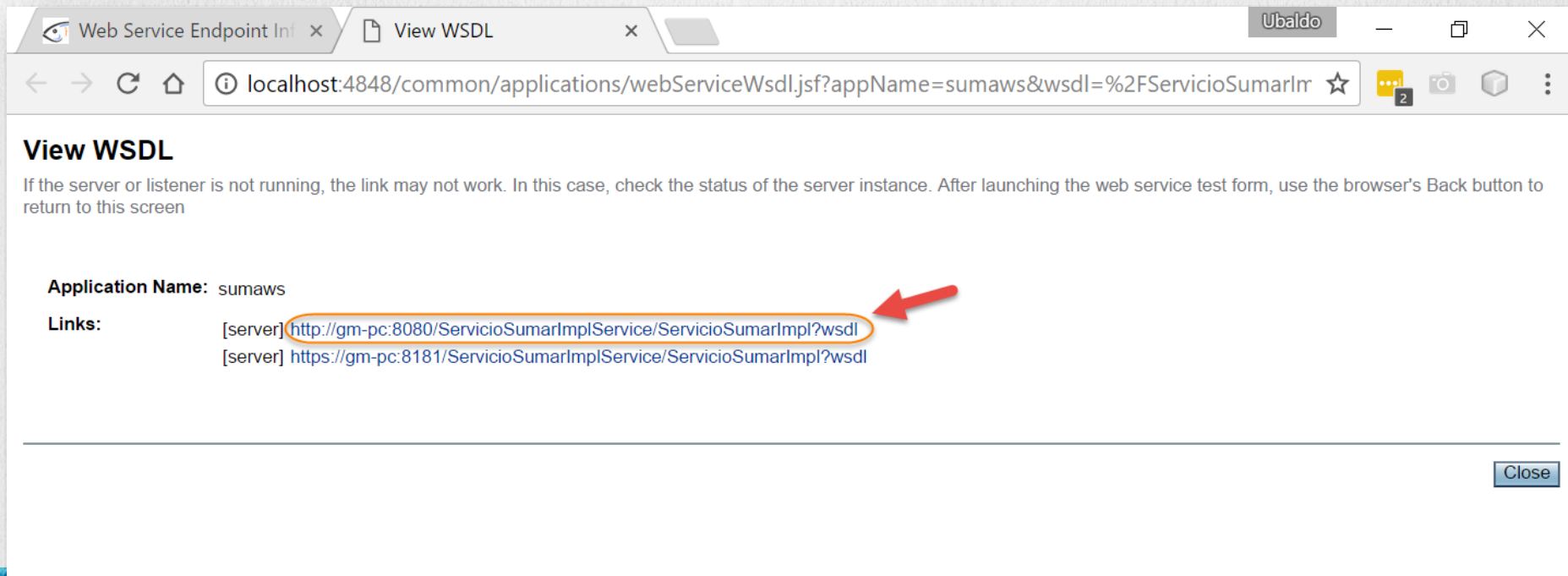


The screenshot shows the GlassFish Web Service Endpoint Information page. The browser address bar displays `localhost:4848/common/index.jsf`. The page header includes navigation links (Home, About..., Help) and user information (User: admin, Domain: domain1, Server: localhost). The left sidebar shows a tree view of the application structure, with `sumaws` selected under `Applications`. The main content area displays the following information:

Web Service Endpoint Information	
View details about a web service endpoint.	
Application Name:	<code>sumaws</code>
Tester:	<code>/ServicioSumarImplService/ServicioSumarImpl?Tester</code>
WSDL:	<code>/ServicioSumarImplService/ServicioSumarImpl?wsdl</code>
Endpoint Name:	<code>ServicioSumarImpl</code>
Service Name:	<code>ServicioSumarImplService</code>
Port Name:	<code>ServicioSumarImplPort</code>
Deployment Type:	<code>109</code>
Implementation Type:	<code>EJB</code>
Implementation Class Name:	<code>beans.ServicioSumarImpl</code>
Endpoint Address URI:	<code>/ServicioSumarImplService/ServicioSumarImpl</code>
Namespace:	<code>http://beans/</code>

PASO 9. VERIFICAMOS EL CLIENTE WS Y EL WSDL

Verificamos el cliente WS y el WSDL generado:



The screenshot shows a web browser window with the title 'Web Service Endpoint Info' and a tab labeled 'View WSDL'. The address bar displays the URL: `localhost:4848/common/applications/webServiceWsdI.jsf?appName=sumaws&wsdl=%2FServicioSumarImr`. The page content includes a heading 'View WSDL' and a warning message: 'If the server or listener is not running, the link may not work. In this case, check the status of the server instance. After launching the web service test form, use the browser's Back button to return to this screen'. Below this, the 'Application Name' is listed as 'sumaws'. Under the 'Links' section, there are two links: '[server] <http://gm-pc:8080/ServicioSumarImplService/ServicioSumarImpl?wsdl>' and '[server] <https://gm-pc:8181/ServicioSumarImplService/ServicioSumarImpl?wsdl>'. The first link is circled in orange, and a red arrow points to it. A 'Close' button is located in the bottom right corner of the page.

View WSDL

If the server or listener is not running, the link may not work. In this case, check the status of the server instance. After launching the web service test form, use the browser's Back button to return to this screen

Application Name: sumaws

Links:

- [server] <http://gm-pc:8080/ServicioSumarImplService/ServicioSumarImpl?wsdl>
- [server] <https://gm-pc:8181/ServicioSumarImplService/ServicioSumarImpl?wsdl>

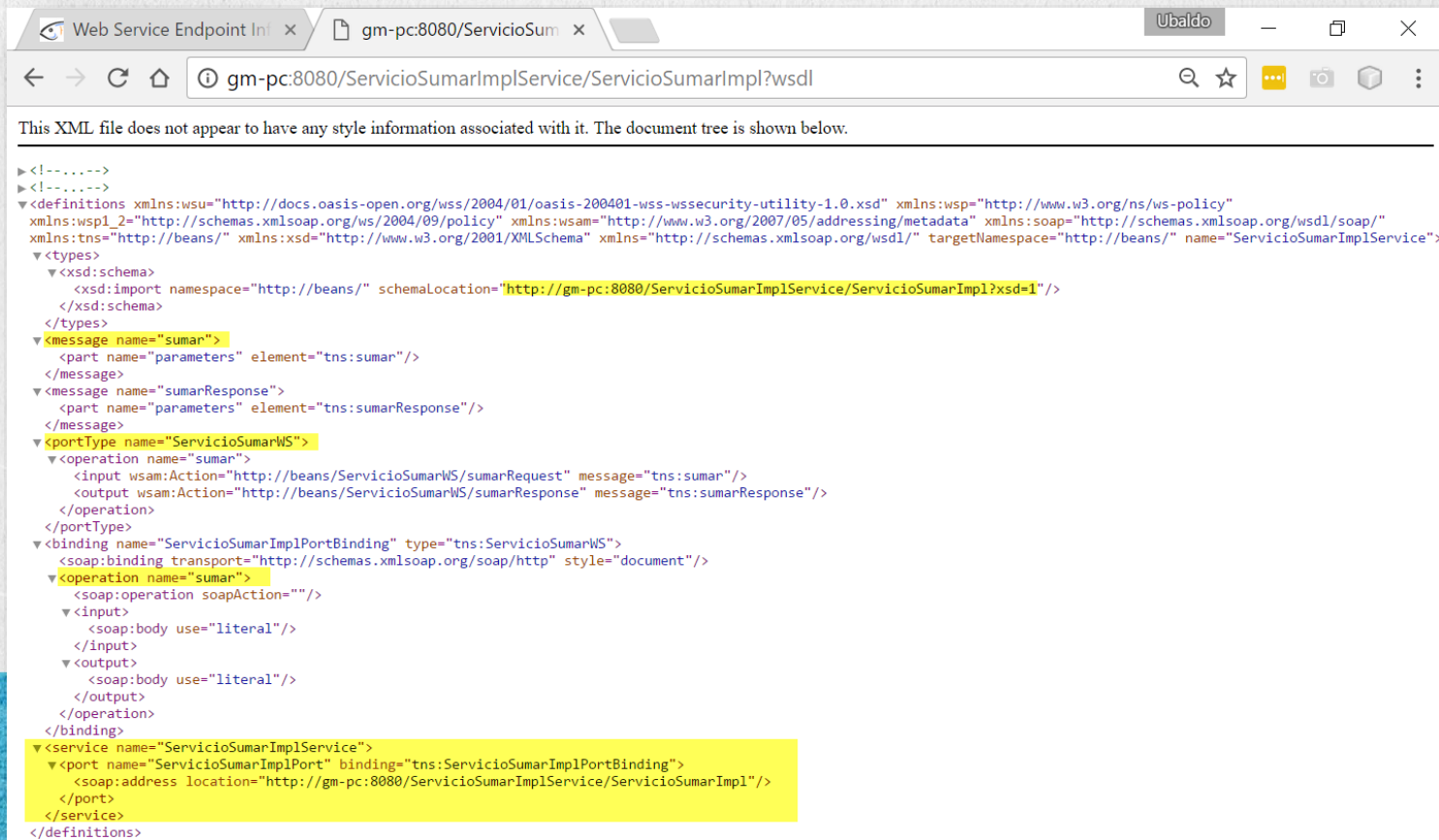
Close

CURSO JAVA EE

www.globalmentoring.com.mx

PASO 9. VERIFICAMOS EL CLIENTE WS Y EL WSDL

Verificamos el cliente WS y el WSDL generado:



The screenshot shows a web browser window with the address bar displaying `gm-pc:8080/ServicioSumarImplService/ServicioSumarImpl?wsdl`. The page content displays the XML WSDL document for the `ServicioSumarImplService`. The XML is structured as follows:

```
<!--...-->
<!--...-->
<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://beans/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http://beans/" name="ServicioSumarImplService">
  <types>
    <xsd:schema>
      <xsd:import namespace="http://beans/" schemalocation="http://gm-pc:8080/ServicioSumarImplService/ServicioSumarImpl?xsd=1"/>
    </xsd:schema>
  </types>
  <message name="sumar">
    <part name="parameters" element="tns:sumar"/>
  </message>
  <message name="sumarResponse">
    <part name="parameters" element="tns:sumarResponse"/>
  </message>
  <portType name="ServicioSumarWS">
    <operation name="sumar">
      <input wsam:Action="http://beans/ServicioSumarWS/sumarRequest" message="tns:sumar"/>
      <output wsam:Action="http://beans/ServicioSumarWS/sumarResponse" message="tns:sumarResponse"/>
    </operation>
  </portType>
  <binding name="ServicioSumarImplPortBinding" type="tns:ServicioSumarWS">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    <operation name="sumar">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
  </binding>
  <service name="ServicioSumarImplService">
    <port name="ServicioSumarImplPort" binding="tns:ServicioSumarImplPortBinding">
      <soap:address location="http://gm-pc:8080/ServicioSumarImplService/ServicioSumarImpl"/>
    </port>
  </service>
</definitions>
```


PASO 10. VERIFICAMOS EL ARCHIVE XSD

Verificamos el archivo xsd:

<http://localhost:8080/ServicioSumarImplService/ServicioSumarImpl?xsd=1>



CURSO JAVA EE

www.globalmentoring.com.mx

CONCLUSIÓN DEL EJERCICIO

Con este ejercicio creado un Servicio Web llamado sumar, el cual recibe dos argumentos enteros, y regresa como resultado la suma de dichos argumentos.

Esto es un ejemplo muy sencillo de un EJB de tipo Stateless que expone uno de sus métodos como servicio Web, pero es el mismo proceso que realizaremos para crear Servicios Web más complejos de cualquier método de un EJB que necesitemos exponer.

En el siguiente ejercicio veremos cómo crear un cliente para consumir el servicio web ya publicado y expuesto en el servidor de Glassfish.

CURSO ONLINE

JAVA EMPRESARIAL JAVA EE

Por: Ing. Ubaldo Acosta



Experiencia y Conocimiento para tu vida



CURSO JAVA EE

www.globalmentoring.com.mx