



Universidad Tecnológica de Durango

T.S.U en Tecnologías de la Información  
Área Desarrollo de Software  
Multiplataforma

Aplicaciones Web Orientado a Servicios

*Practicas*

Alumno: Moran Vázquez Miguel Ángel

4°B

Profesor: M.T.I Dagoberto Fiscal Gurrola

Durango, Dgo a 2 de octubre del 2022



Tabla de Ilustraciones

Ilustración 1.Crear el proyecto JAVA..... 3

Ilustración 2.Nombre del proyecto JAXWS ..... 3

Ilustración 3.Crear un Package en el proyecto..... 4

Ilustración 4.Nombre del Package ..... 4

Ilustración 5.Calculadora\_Original ..... 4

Ilustración 6.Calculadora Implementar..... 5

Ilustración 7.Calculadora Publish ..... 5

Ilustración 8.Abrir cmd..... 6

Ilustración 9.Calculadora Implementar Service ..... 6

Ilustración 10.Calculadora Consumer ..... 7

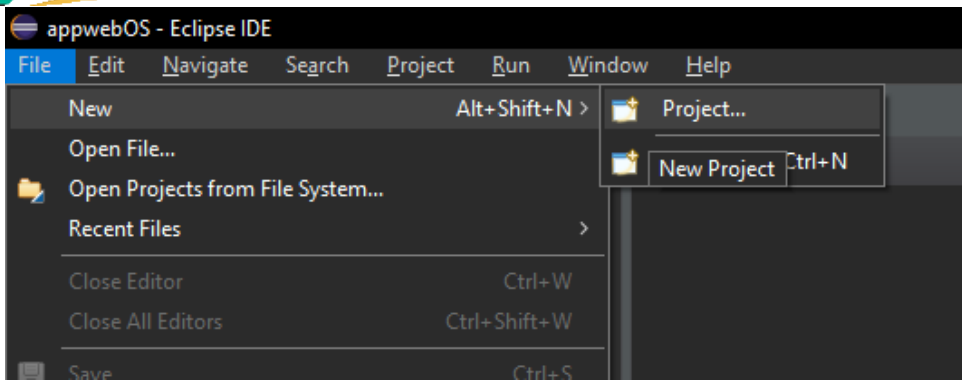


Ilustración 1. Crear el proyecto JAVA

Vamos a crear un proyecto en Eclipse accediendo a File, new y por último en Java Project, si no aparece entra en other y busca Java Project.

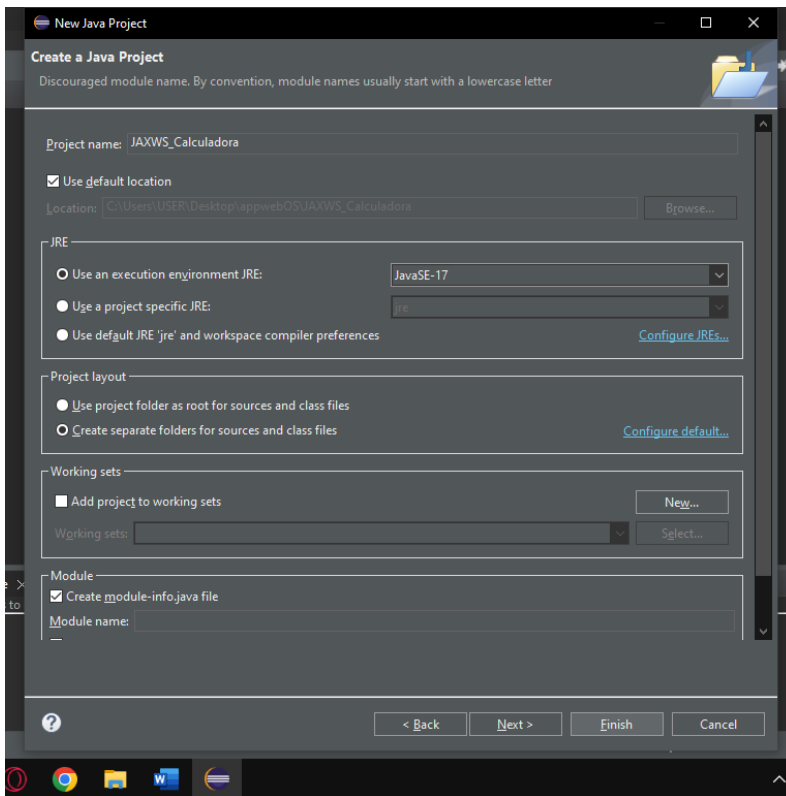


Ilustración 2. Nombre del proyecto JAXWS

Con el nombre de JAXWS\_Calculadora.

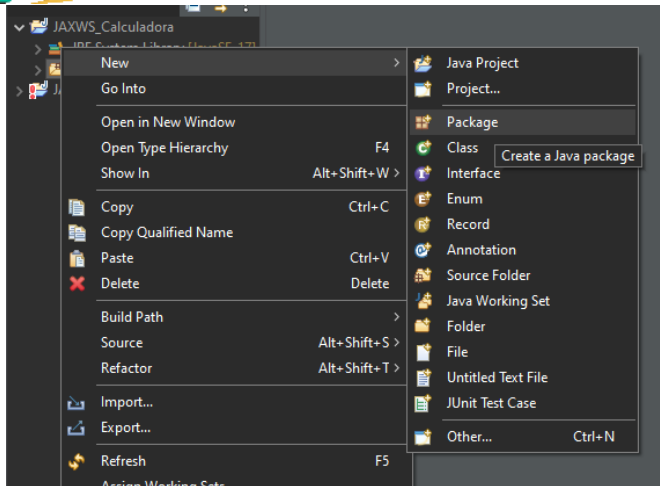


Ilustración 3. Crear un Package en el proyecto

Se creará un package para administrar nuestros archivos.

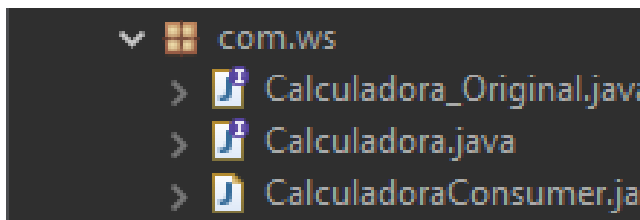


Ilustración 4. Nombre del Package

Con el nombre com.ws

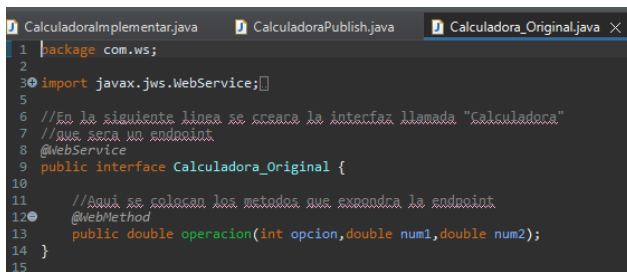


Ilustración 5. Calculadora Original

Este es el primer archivo que vamos a crear que es la interfaz de la calculadora que será un endpoint donde utilizamos la librería `javax.jws.WebService`, se colocó un método que expone el endpoint.



```
CalculadoraImplementar.java x CalculadoraPublish.java Calculadora_Original.java
1 package com.ws;
2
3 * Esta clase permite implementar la interfaz Calculadora
4
5 import javax.jws.WebService;
6
7 //La siguiente línea se especifica cual es el endpoint de la
8 //interfaz a implementar
9 @WebService(endpointInterface="com.ws.Calculadora")
10 public class CalculadoraImplementar implements Calculadora{
11
12     @Override
13     public double operacion(int opcion,double num1,double num2) {
14
15         double resultado=0;
16
17         switch(opcion)
18         {
19             case 1:resultado=num1+num2;break;
20             case 2:resultado=num1-num2;break;
21             case 3:resultado=num1*num2;break;
22             case 4:resultado=num1/num2;break;
23             case 5:resultado=num1*num2/2;break;
24             default:break;
25         }
26         return resultado;
27     }
28 }
29
30
31
32
```

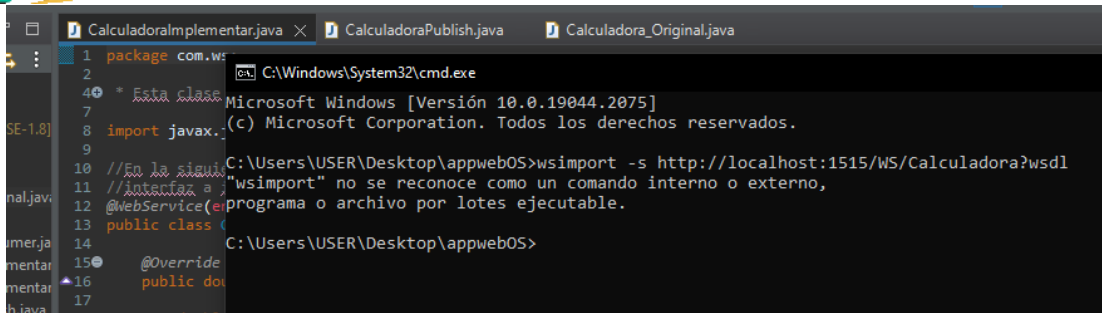
Ilustración 6.Calculadora Implementar

Esta es la clase que nos va a permitir implementar la interfaz de la calculadora igualmente se utilizara la librería, esto igual en todo el proyecto, se esta creando un switch que contendrá 5 casos que son los métodos de operación de la calculadora.

```
Calculadora.java CalculadoraImplementar.java CalculadoraPublish.java x
1 package com.ws;
2
3 //Crear una clase para publicar el WS en el
4 import javax.xml.ws.Endpoint;
5
6 public class CalculadoraPublish {
7
8     public static void main(String[] args) {
9         // TODO Auto-generated method stub
10
11         Endpoint.publish("http://127.0.0.1:1515/WS/Calculadora", new CalculadoraImplementar());
12         System.out.println("Se publico correctamente el Web Service en el servidor");
13     }
14 }
15
16
17
```

Ilustración 7.Calculadora Publish

En esta clase se está publicando el WS en el servidor que genera la computadora como servidor local, se creara una clase dentro del mismo paquete que es el método main para que el usuario pueda manipularlo. El endpoint estará colocado en localhost en el puerto 15:15, iremos a nuestro navegador y colocaremos esta dirección: <http://localhost:1515/WS/Calculadora?wsdl> para ver si funciona correctamente.



```
1 package com.ws;
2
3
4 * Esta clase
5
6
7
8 import javax.xml
9
10 //En la sigui
11 //instanci
12 @WebService(
13 public class
14
15 @Override
16 public do
17
```

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Versión 10.0.19044.2075]
(c) Microsoft Corporation. Todos los derechos reservados.

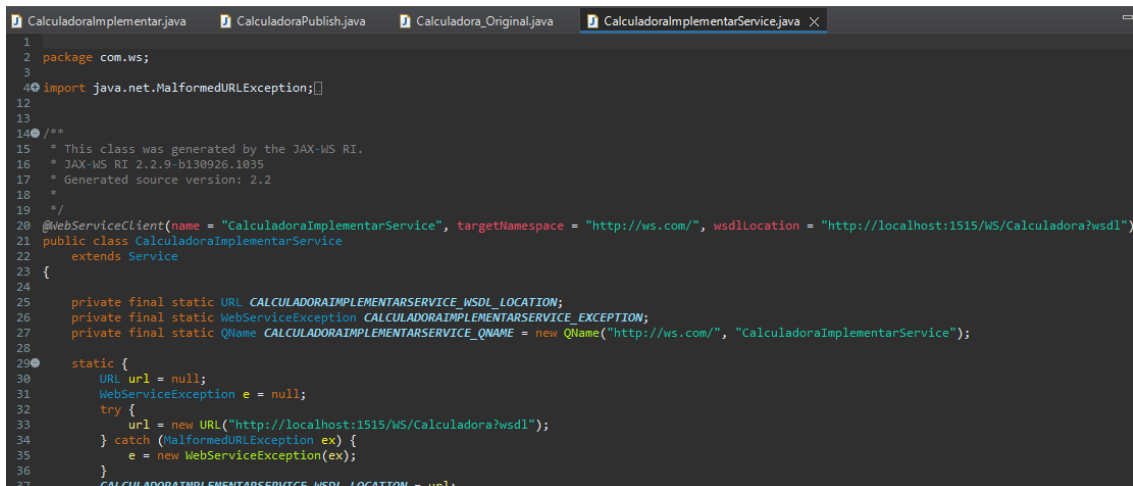
C:\Users\USER\Desktop\appwebOS>wsimport -s http://localhost:1515/WS/Calculadora?wsdl
"wsimport" no se reconoce como un comando interno o externo,
programa o archivo por lotes ejecutable.

C:\Users\USER\Desktop\appwebOS>
```

Ilustración 8. Abrir cmd

Entraremos a la carpeta donde esta ubicado nuestro proyecto, damos click en la url, borramos la url y escribimos cmd para acceder a línea de comandos con la dirección de nuestra carpeta.

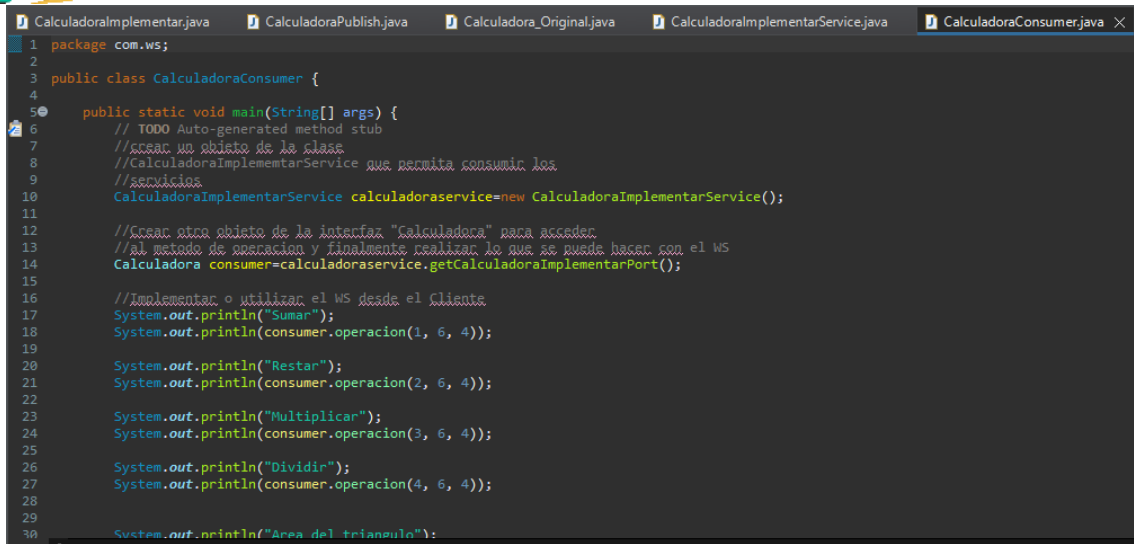
Dentro del cmd escribiremos wsimport -s http://localhost:1515/WS/Calculadora?wsdl esto nos dará un generador de código para encontrar nuestro servidor.



```
1 package com.ws;
2
3
4 import java.net.MalformedURLException;
5
6
7
8
9
10
11
12
13
14 /**
15  * This class was generated by the JAX-WS RI.
16  * JAX-WS RI 2.2.9-b130926.1035
17  * Generated source version: 2.2
18  */
19
20 @WebServiceClient(name = "CalculadoraImplementarService", targetNamespace = "http://ws.com/", wsdlLocation = "http://localhost:1515/WS/Calculadora?wsdl")
21 public class CalculadoraImplementarService
22     extends Service
23 {
24
25     private final static URL CALCULADORAIMPLEMENTARSERVICE_WSDL_LOCATION;
26     private final static WebServiceException CALCULADORAIMPLEMENTARSERVICE_EXCEPTION;
27     private final static QName CALCULADORAIMPLEMENTARSERVICE_QNAME = new QName("http://ws.com/", "CalculadoraImplementarService");
28
29     static {
30         URL url = null;
31         WebServiceException e = null;
32         try {
33             url = new URL("http://localhost:1515/WS/Calculadora?wsdl");
34         } catch (MalformedURLException ex) {
35             e = new WebServiceException(ex);
36         }
37         CALCULADORAIMPLEMENTARSERVICE_WSDL_LOCATION = url;
```

Ilustración 9. Calculadora Implementar Service

Después de ingresar al cmd se genera un código como en cmd y dentro de nuestra carpeta de eclipse, es para que la calculadora funcione correctamente al momento de generar una cuenta aritmética.



```
1 package com.ws;
2
3 public class CalculadoraConsumer {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         // crear un objeto de la clase
8         // CalculadoraImplementarService que permita consumir los
9         // servicios
10        CalculadoraImplementarService calculadoraservice=new CalculadoraImplementarService();
11
12        // crear otro objeto de la interfaz "Calculadora" para acceder
13        // al metodo de operacion y finalmente realizar lo que se pueda hacer con el WS
14        Calculadora consumer=calculadoraservice.getCalculadoraImplementarPort();
15
16        // Implementar o utilizar el WS desde el cliente
17        System.out.println("Sumar");
18        System.out.println(consumer.operacion(1, 6, 4));
19
20        System.out.println("Restar");
21        System.out.println(consumer.operacion(2, 6, 4));
22
23        System.out.println("Multiplicar");
24        System.out.println(consumer.operacion(3, 6, 4));
25
26        System.out.println("Dividir");
27        System.out.println(consumer.operacion(4, 6, 4));
28
29        System.out.println("Area del triángulo");
30    }
31}
```

Ilustración 10. Calculadora Consumer

Con esta clase se utiliza el WS desde el cliente y se pueden realizar las operaciones de manera correcta.

### Retroalimentación

Esta interesante como es que se publica un proyecto java en el servidor que genera nuestra computadora, varios de nosotros batallamos para realizar esta práctica porque realmente no teníamos un previo conocimiento sobre este tema de servicios web y es totalmente nuevo para nosotros.

Si lo aplicaría en el ambiente laboral ya que estamos brindando un servicio y automatizando una tarea.