INSTITUT SUPERIEUR DES ETUDES TECHNOLOGIQUES DE SFAX

Département Technologies de Informatique

Atelier SOA -TP04
Tester un service web avec un client JAVA

Objectifs

Tester un Web Service développé avec l'API JAX-WS

1. Tester les méthodes du WS via un client JAVA

- Utiliser la bibliothèque JAX-WS pour générer les classes de souche
- Appeler des méthodes du service web

A. Invoquer le SW à partir d'une application JAVA (client)

1. Créer un simple projet **JAVA** dans **IntelliJ IDEA** nommé **jax-ws-client** ayant les caractéristiques suivantes :

   - groupId : **org.soa.tp4**
   - artifactId : **jax-ws-client**
   - version : **1.0-SNAPSHOT**

2. Ajouter dans le fichier POM la définition du plugin de jaxws (directement à l'intérieur de la balise <**project**>) :

```
<build>

  <plugins>

    <plugin>
      <groupId>com.sun.xml.ws</groupId>
      <artifactId>jaxws-maven-plugin</artifactId>
      <version>4.0.1</version>
      <configuration>
        <wsdlUrls>
          <wsdlUrl>http://192.168.1.6:8585/BanqueWS?wsdl</wsdlUrl>
        </wsdlUrls>
```

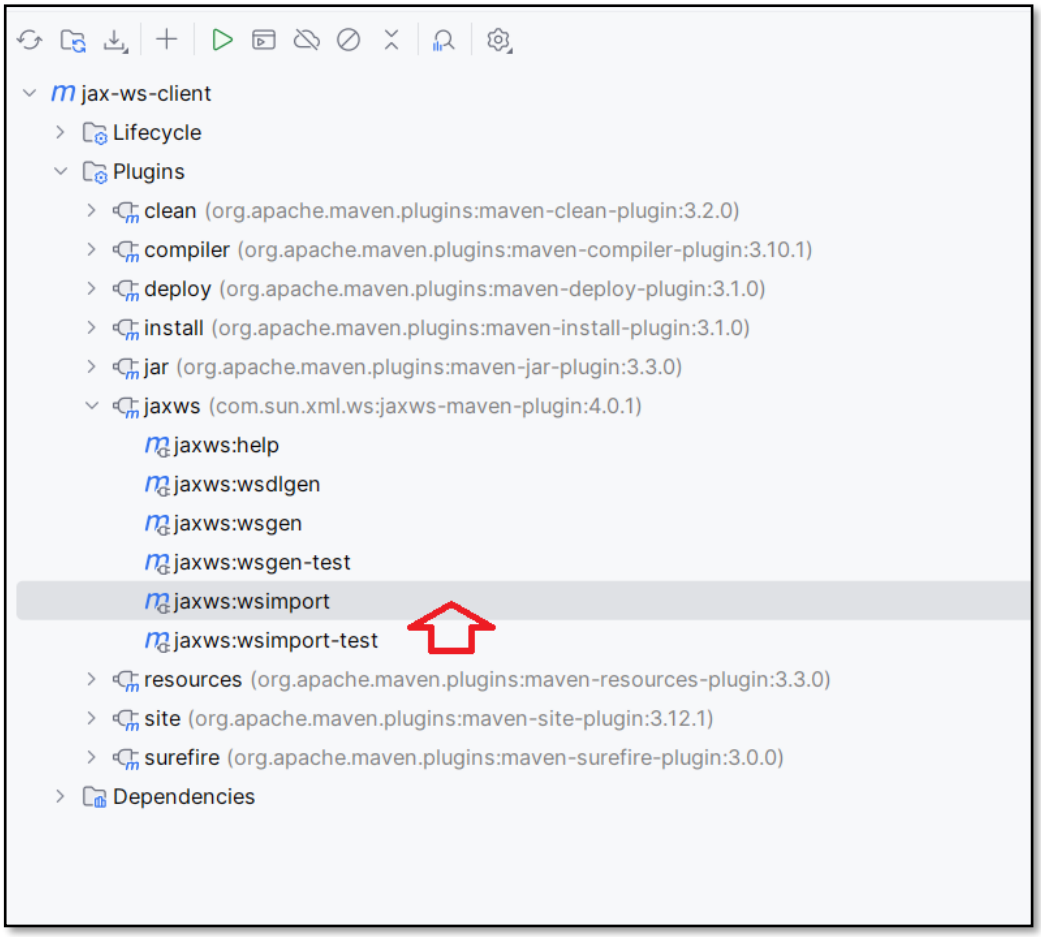
```
        <packageName>org.soa.tp2</packageName>
        <sourceDestDir>
          ${project.build.sourceDirectory}/
        </sourceDestDir>
      </configuration>

    </plugin>
  </plugins>

</build>
```

NB : Spécifier l'adresse convenable à votre serveur dans la balise <wsdlUrls>

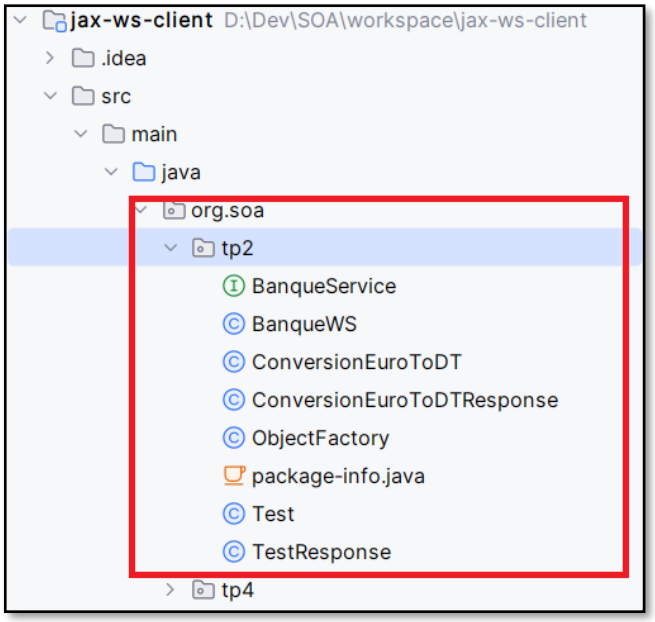
3. Ajouter les deux dépendances suivantes :

```
<dependency>
  <groupId>jakarta.xml.ws</groupId>
  <artifactId>jakarta.xml.ws-api</artifactId>
  <version>4.0.0</version>
</dependency>


<dependency>
  <groupId>com.sun.xml.ws</groupId>
  <artifactId>rt</artifactId>
  <version>4.0.1</version>
</dependency>
```

4. Mettre à jour le projet
5. Lancer la génération des proxys en double cliquant sur **jaxws :wsimport** comme suit :



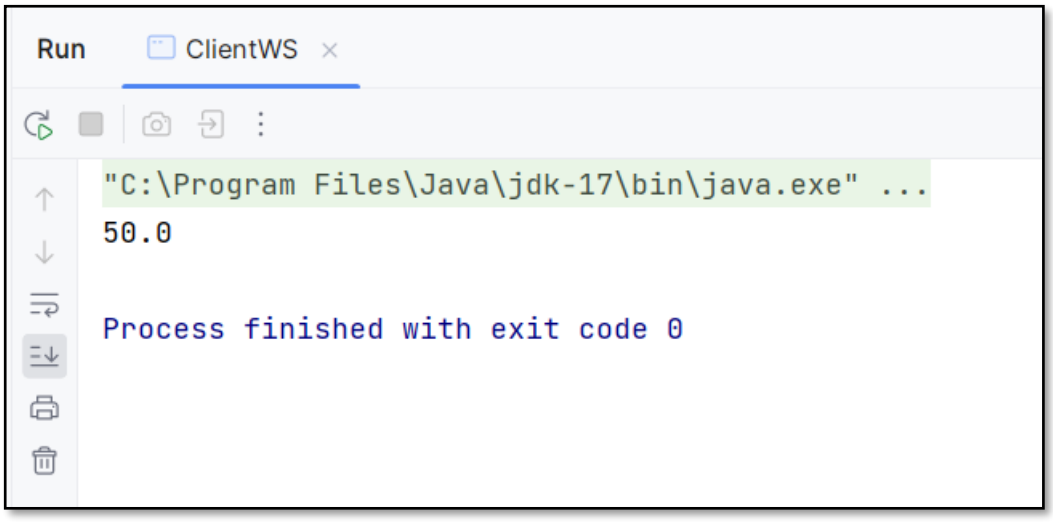
6. Remarquer la génération du code source dans un package « **org.soa.tp2** » :



7. Dans un package « **org.soa.tp4** », créer la classe « **ClientWS** » qui présente l'application cliente :

```
package org.soa.tp4;
import org.soa.tp2.*;

public class ClientWS {

    public static void main(String[] args)
    {
        BanqueService stub = new
BanqueWS().getBanqueServicePort();

    System.out.println(stub.conversionEuroToDT
(20.0));

    }
}
```

8. Exécuter la classe « ClientWs » et remarquer le résultat



B. Gérer les objets à travers un service web

1. Reprendre les étapes précédentes (vues dans TP_02 et Tp_04) pour créer , déployer et invoquer un web service nommé « **GestionWS** » qui manipule des objets de type « **Compte** » :

   - Premièrement, écrire dans le projet « **jax-ws-serveur** » le code de la classe « **GestionService** »

```
package org.soa.tp2;

import java.util.ArrayList;
import java.util.List;

public class GestionService
{

    public Compte getCompte( int codeCompte)
    {
    return new Compte (codeCompte, Math.random()*9000);
    }

    public List<Compte> getComptes()
    {
        List<Compte> cptes = new ArrayList<Compte>();
        for (int i=0; i<5;i++)
        {
            cptes.add(new Compte(i, Math.random()*6000));
        }
        return cptes;
    }
}
```

   - Le code de la classe « **Compte** » est le suivant :

```
package org.soa.tp2;

public class Compte
{
    private int code;
    private double solde;
    public int getCode() {
        return code;
    }
    public void setCode(int code) {
        this.code = code;
```

```
	}
	public double getSolde() {
		return solde;
	}
	public void setSolde(double solde) {
		this.solde = solde;
	}
	public Compte(int code, double solde) {
		super();
		this.code = code;
		this.solde = solde;
	}
	public Compte()
	{
		System.out.println("Nouveau compte..");
	}
	public String toString() {
		return "Compte [code=" + code + ", solde=" + solde + "]";
	}
}
```

- Ajouter les annotations « **JAX-WS** » pour :

  a. Définir un service web nommé « **GestionWS** »
  b. Définir les méthodes à publier
  c. Donner le nom « **code** » pour l'argument de la méthode « **getCompte** »

2. Invoquer le service web dans une application client du projet « **jax-ws-client** »