

DEPARTEMENT MATHÉMATIQUES ET INFORMATIQUE

Compte-rendu Activité Pratique N° 3

Filière :

« Génie du Logiciel et des Systèmes Informatiques Distribués »

GLSID

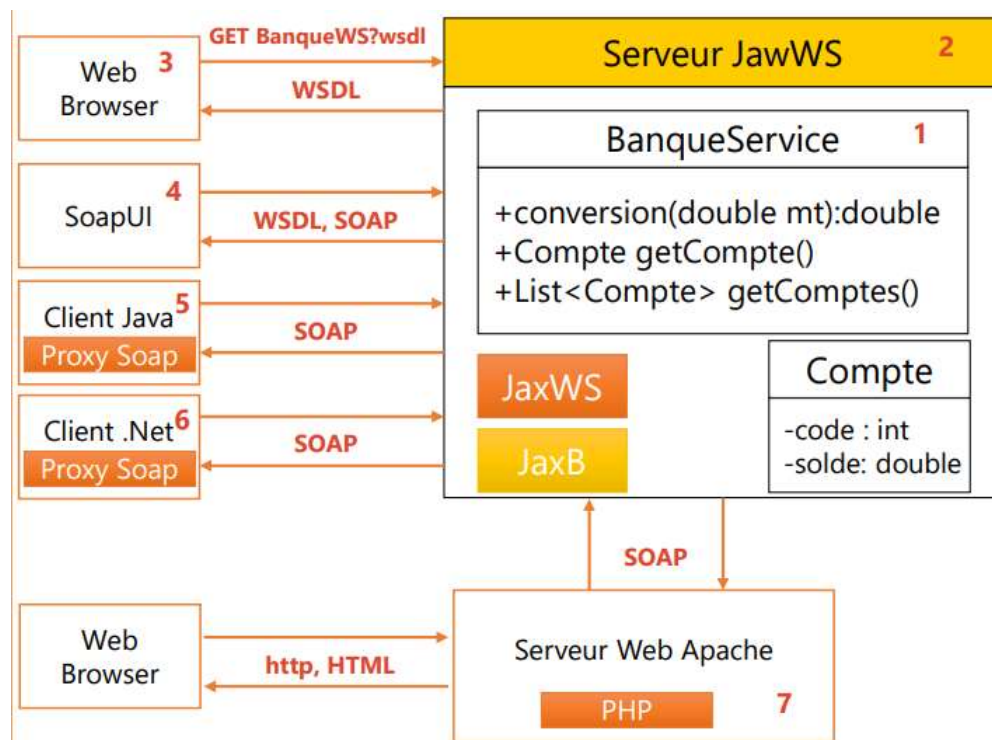
**Web services SOAP, WSDL, UDDI avec
JAXWS**

Préparé par : Hajar TAJAYOUTI

Année Universitaire : 2022-2023

Objectif : Créer un Web service basé sur SOAP qui permet de :

- Convertir un montant de l'auro en DH
- Consulter un Compte
- Consulter une Liste de comptes



Etape **1** :

Classe BanqueService

```
@WebService(name = "BanqueWS")
public class BanqueService {

    @WebMethod(operationName = "Convert")
    public double Conversion(@WebParam(name = "montant") double montant)
    {
        return montant*16.54;
    }

    @WebMethod
    public Compte getCompte(@WebParam(name = "code") int id)
    {
        return new Compte(id, solde: Math.random()*988, new Date());
    }

    @WebMethod
    public List<Compte> listComptes()
    {
        return List.of(
            new Compte(id: 1, solde: Math.random()*988, new Date()),
            new Compte(id: 2, solde: Math.random()*988, new Date()),
            new Compte(id: 3, solde: Math.random()*988, new Date())
        );
    }
}
```

Class Compte

```
public class Compte {
    //Attributes
    private int id;
    //Attributes
    private double solde;
    //Attributes
    private Date dateCreation;

    public Compte() {}

    //Attributes
    public Compte(int id, double solde, Date dateCreation) {...}

    public int getId() { return id; }

    public void setId(int id) { this.id = id; }

    public double getSolde() { return solde; }

    public void setSolde(double solde) { this.solde = solde; }

    public Date getDateCreation() { return dateCreation; }

    public void setDateCreation(Date dateCreation) { this.dateCreation = dateCreation; }
}
```

Etape 2 : Déployer le Web service avec un simple Serveur JaxWS

```
public class ServerJaxWS {
    public static void main(String[] args) {
        //demarrer un serveur HTTP use port 9191 pour consulter le web service banqueService
        Endpoint.publish( address: "http://0.0.0.0:9191/",new BanqueService());
        System.out.println("web service deployé sur http://0.0.0.0:9191");
    }
}
```

Etape 3 : visualiser le WSDL avec un browser HTTP

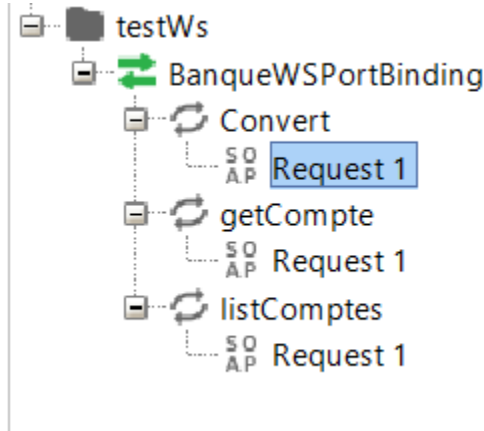


This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://ws/BanqueWS" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" targetNamespace="http://ws/BanqueWS">
  <import namespace="http://ws/" schemaLocation="http://localhost:9191/BanqueWS?wsdl"/>
  <types>
    <xsd:schema>
      <import namespace="http://ws/" schemaLocation="http://localhost:9191/BanqueWS?wsdl"/>
    </xsd:schema>
  </types>
  <message name="Convert">
    <part name="parameters" element="tns:Convert"/>
  </message>
  <message name="ConvertResponse">
    <part name="parameters" element="tns:ConvertResponse"/>
  </message>
  <message name="getCompte">
    <part name="parameters" element="tns:getCompte"/>
  </message>
  <message name="getCompteResponse">
    <part name="parameters" element="tns:getCompteResponse"/>
  </message>
  <message name="listComptes">
    <part name="parameters" element="tns:listComptes"/>
  </message>
  <message name="listComptesResponse">
    <part name="parameters" element="tns:listComptesResponse"/>
  </message>
  <portType name="BanqueWS">
    <operation name="Convert">
      <input wsam:Action="http://ws/BanqueWS/ConvertRequest" message="tns:Convert"/>
      <output wsam:Action="http://ws/BanqueWS/ConvertResponse" message="tns:ConvertResponse"/>
    </operation>
    <operation name="getCompte">
      <input wsam:Action="http://ws/BanqueWS/getCompteRequest" message="tns:getCompte"/>
      <output wsam:Action="http://ws/BanqueWS/getCompteResponse" message="tns:getCompteResponse"/>
    </operation>
    <operation name="listComptes">
      <input wsam:Action="http://ws/BanqueWS/listComptesRequest" message="tns:listComptes"/>
      <output wsam:Action="http://ws/BanqueWS/listComptesResponse" message="tns:listComptesResponse"/>
    </operation>
  </portType>
  <binding name="BanqueWSBinding" type="tns:BanqueWS">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
  </binding>
</definitions>
```

```
▼<portType name="BanqueWS">
  ▼<operation name="Convert">
    <input wsam:Action="http://ws/BanqueWS/ConvertRequest" message="tns:Convert"/>
    <output wsam:Action="http://ws/BanqueWS/ConvertResponse" message="tns:ConvertResponse"/>
  </operation>
```

Etape 4 : tester les opérations du web service avec l'outil SoapUI



Tester la méthode **Convert** qui permet de convertir un montant de l'auro en DH

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <ws:Convert>
      <montant>10</montant>
    </ws:Convert>
  </soapenv:Body>
</soapenv:Envelope>
```

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:ConvertResponse xmlns:ns2="http://ws/">
      <return>1.0</return>
    </ns2:ConvertResponse>
  </S:Body>
</S:Envelope>
```

Tester la méthode **getCompte** qui permet de consulter un Compte

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <ws:getCompte>
      <code>1</code>
    </ws:getCompte>
  </soapenv:Body>
</soapenv:Envelope>
```

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getCompteResponse xmlns:ns2="http://ws/">
      <return>
        <dateCreation>2022-11-03T23:09:35.601+01:00</dateCreation>
        <id>1</id>
        <solde>130.0242483445283</solde>
      </return>
    </ns2:getCompteResponse>
  </S:Body>
</S:Envelope>
```

Tester la méthode listComptes qui permet de consulter une Liste de comptes

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <ws:listComptes/>
  </soapenv:Body>
</soapenv:Envelope>
```

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:listComptesResponse xmlns:ns2="http://ws/">
      <return>
        <dateCreation>2022-11-03T23:11:03.366+01:00</dateCreation>
        <id>1</id>
        <solde>534.8745993991175</solde>
      </return>
      <return>
        <dateCreation>2022-11-03T23:11:03.366+01:00</dateCreation>
        <id>2</id>
        <solde>774.2204023687365</solde>
      </return>
      <return>
        <dateCreation>2022-11-03T23:11:03.366+01:00</dateCreation>
        <id>3</id>
        <solde>609.5326467565233</solde>
      </return>
    </ns2:listComptesResponse>
  </S:Body>
</S:Envelope>
```

- ✓ **JaxWS** est besoin d'une librairie pour **convertir** un objet **java** en **xml** alors on utilise **JaxB** pour faire le **Mapping Objet XML**

⇒ Annotation JaxB pour ignorer ce champ lors de Mapping

```
@XmlTransient
private Date dateCreation;
```

⇒ Pour appliquer les annotations sur les champs et pas sur les Getters et Setters

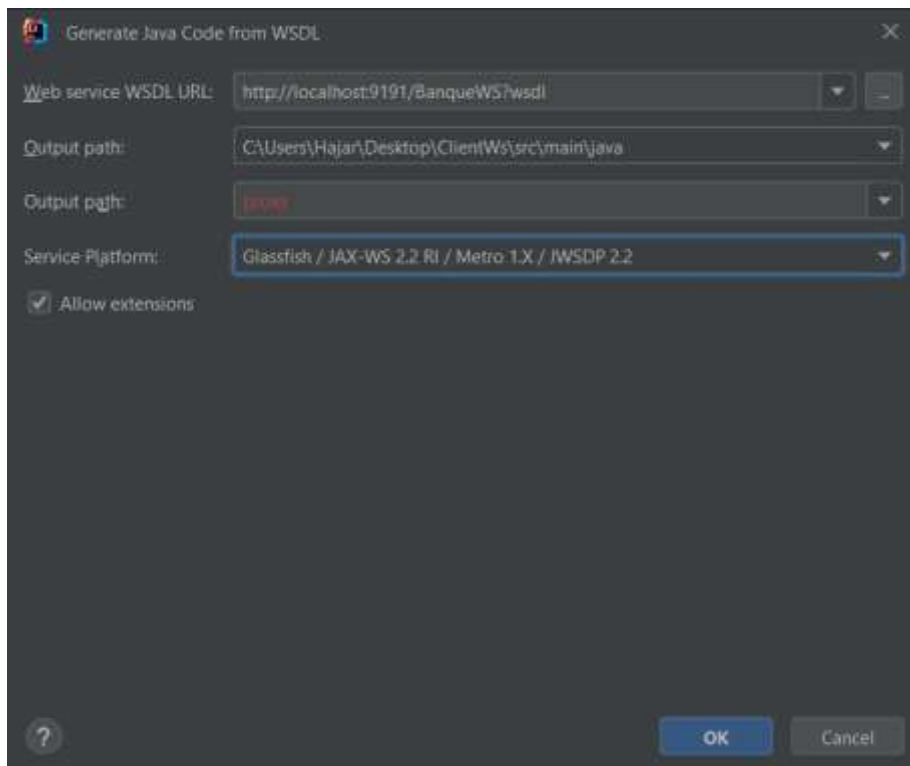
```
@XmlAccessorType(XmlAccessType.FIELD)
```

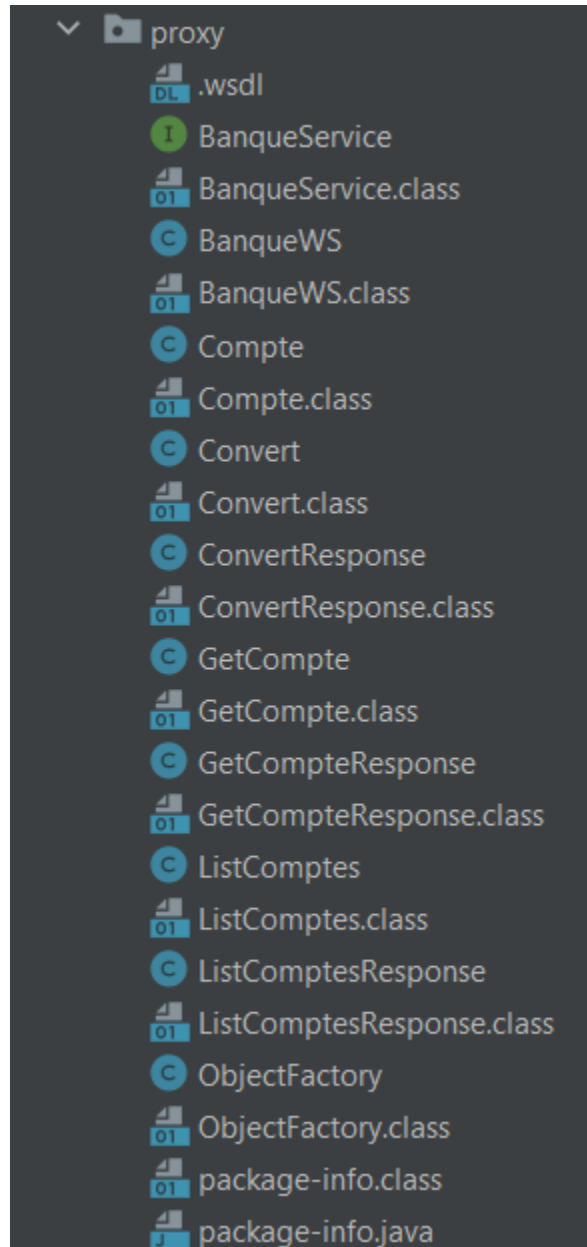
```
<Raw XML>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <ws:getCompte>
      <code>1</code>
    </ws:getCompte>
  </soapenv:Body>
</soapenv:Envelope>
```

```
<Raw XML>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getCompteResponse xmlns:ns2="http://ws/">
      <return>
        <id>1</id>
        <solde>406.37113801389404</solde>
      </return>
    </ns2:getCompteResponse>
  </S:Body>
</S:Envelope>
```

Etape 5 : création d'un **client java** qui peut consommer ce web service

La génération des classes java qui permettent de communiquer avec le web service utilisant WSDL





nous avons créé un **Middleware** (intermediere) «stub» qui permet a un objet qui se trouve dans une machine de se connecter a un objet qui se trouve dans une autre machine


```
public class ClientWS {  
    public static void main(String[] args) {  
        BanqueService stub=new BanqueWS().getBanqueServicePort();  
        System.out.println(stub.convert( montant: 100));  
        Compte compte=stub.getCompte( code: 1);  
        System.out.println(compte.getId());  
        System.out.println(compte.getSolde());  
    }  
}
```

```
"C:\Program Files\Java\jdk-17.0.3\bin\java.exe" ...  
10.0  
1  
469.1514056629838  
  
Process finished with exit code 0
```