

2022-2023

**Département Mathématiques et Informatique**  
**GLSID 3 – S5**

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

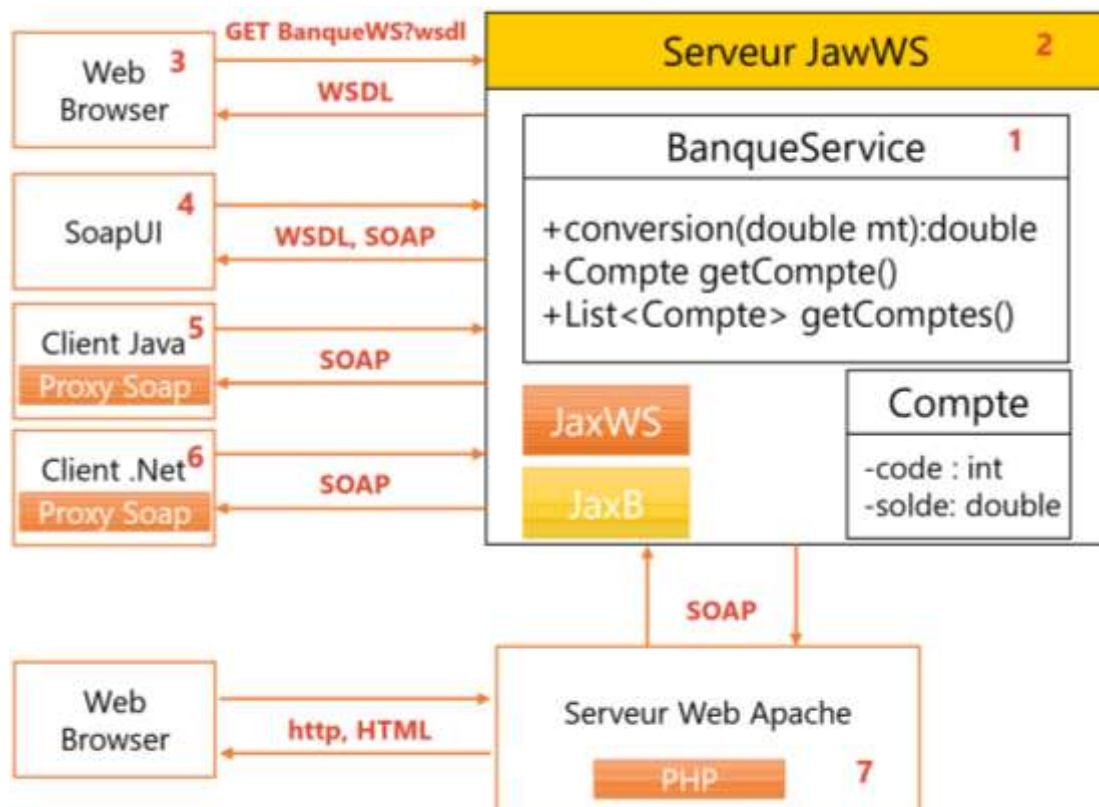
Rapport de devoir 1

**Préparé par : Fatima Zahra HASBI**

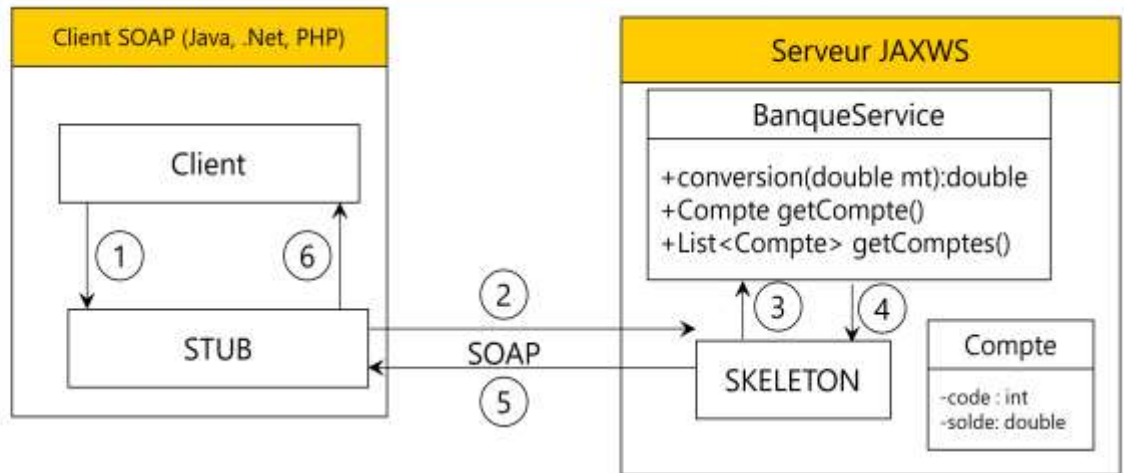
**Encadré par : Mr Mohamed YOUSSEFI**

## Application

1. Créer un Web service qui permet de :
  - Convertir un montant de l'auro en DH
  - Consulter un Compte
  - Consulter une Liste de comptes
2. Déployer le Web service avec un simple Serveur JaxWS
3. Consulter et analyser le WSDL avec un Browser HTTP
4. Tester les opérations du web service avec un outil comme SoapUI ou Oxygen
5. Créer un Client SOAP Java
6. Créer un Client SOAP Dot Net
7. Créer un Client SOAP PHP
8. Déployer le Web Service dans un Projet Spring Boot



## Architecture 1



- 1 Le client demande au stub de faire appel à la méthode conversion(12)
- 2 Le Stub se connecte au Skeleton et lui envoie une requête SOAP
- 3 Le Skeleton fait appel à la méthode du web service
- 4 Le web service retourne le résultat au Skeleton
- 5 Le Skeleton envoie le résultat dans une la réponse SOAP au Stub
- 6 Le Stub fournit le résultat au client



## La classe Compte:

```
package ws;

import java.util.Date;

public class Compte {
    private int code;
    private double solde;
    private Date dateCreation;

    public Compte() {}
    public Compte(int code, double solde, Date dateCreation) {
        this.code = code;
        this.solde = solde;
        this.dateCreation = dateCreation;
    }

    public Date getDateCreation() {
        return dateCreation;
    }

    public double getSolde() {
        return solde;
    }

    public int getCode() {
        return code;
    }

    public void setCode(int code) {
```

```

        this.code = code;
    }

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

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

```

## La classe BanqueService qui contient les méthodes :

```

package ws;

import jakarta.jws.WebMethod;
import jakarta.jws.WebParam;
import jakarta.jws.WebService;

import java.util.Date;
import java.util.List;

@WebService(serviceName = "BanqueWS")
public class BanqueService {

    @WebMethod(operationName = "Convert")
    public double conversion(@WebParam(name = "montant") double mt){
        return mt*11;
    }

    @WebMethod
    public Compte getCompte(@WebParam(name = "code")int code){
        return new Compte(code, Math.random()*54000, new Date());
    }

    @WebMethod
    public List<Compte> listComptes(){
        return List.of(
            new Compte(1, Math.random()*54000, new Date()),
            new Compte(2, Math.random()*54000, new Date()),
            new Compte(3, Math.random()*54000, new Date())
        );
    }
}

```

## La classe ServerJWS

```

import jakarta.xml.ws.Endpoint;
import ws.BanqueService;

public class ServerJWS {
    public static void main(String[] args) {
        Endpoint.publish("http://0.0.0.0:9191/", new BanqueService());
        System.out.println("Web service déployé sur http://0.0.0.0:9191/");
    }
}

```


Sur le browser :


Web Services	
Endpoint	Information
Service Name: {http://ws/}BanqueWS	Address: http://localhost:9191/
Port Name: {http://ws/}BanqueSEServicePort	WSDL: <a href="http://localhost:9191/?wsdl">http://localhost:9191/?wsdl</a>
	Implementation class: ws.BanqueSEService

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

```
<!-- Published by XML-WS Runtime (https://github.com/eclipse-ee4j/metro-jax-ws). Runtime's version is XML-WS-2.3.0
<!-- Generated by XML-WS Runtime (https://github.com/eclipse-ee4j/metro-jax-ws). Runtime's version is XML-WS-2.3.0
▼<definitions xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy" xmlns:wsam="http://www.w3.org/2007/05/addressing"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http://ws/"
▼<types>
▼<xsd:schema>
<xsd:import namespace="http://ws/" schemaLocation="http://localhost:9191/?xsd=1"/>
</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="BanqueSEService">
▼<operation name="Convert">
<input wsam:Action="http://ws/BanqueSEService/ConvertRequest" message="tns:Convert"/>
<output wsam:Action="http://ws/BanqueSEService/ConvertResponse" message="tns:ConvertResponse"/>
</operation>
▼<operation name="getCompte">
<input wsam:Action="http://ws/BanqueSEService/getCompteRequest" message="tns:getCompte"/>
<output wsam:Action="http://ws/BanqueSEService/getCompteResponse" message="tns:getCompteResponse"/>
</operation>
```

## Test sur SOAP UI

 New SOAP Project ✕

**New SOAP Project** 

Creates a WSDL/SOAP based Project in this workspace


Project Name:


Initial WSDL:

Create Requests: ☒ Create sample requests for all operations?

Create TestSuite: ☐ Creates a TestSuite for the imported WSDL

Relative Paths: ☐ Stores all file paths in project relatively to project file (requires save)




SOAP Request 1  http://localhost:9191/

**Raw XML**

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <ws:Convert>
      <montant>102030</montant>
    </ws:Convert>
  </soapenv:Body>
</soapenv:Envelope>
```

**Raw XML**

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:ConvertResponse xmlns:ns2="http://ws/">
      <return>1122330.0</return>
    </ns2:ConvertResponse>
  </S:Body>
</S:Envelope>
```

SOAP Request 1  http://localhost:9191/

**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>
        <code>1</code>
        <dateCreation>2022-10-24T12:48:27.853+01:00</dateCreation>
        <solde>49935.457627903605</solde>
      </return>
    </ns2:getCompteResponse>
  </S:Body>
</S:Envelope>
```

SOAP Request 1  http://localhost:9191/

**Raw XML**

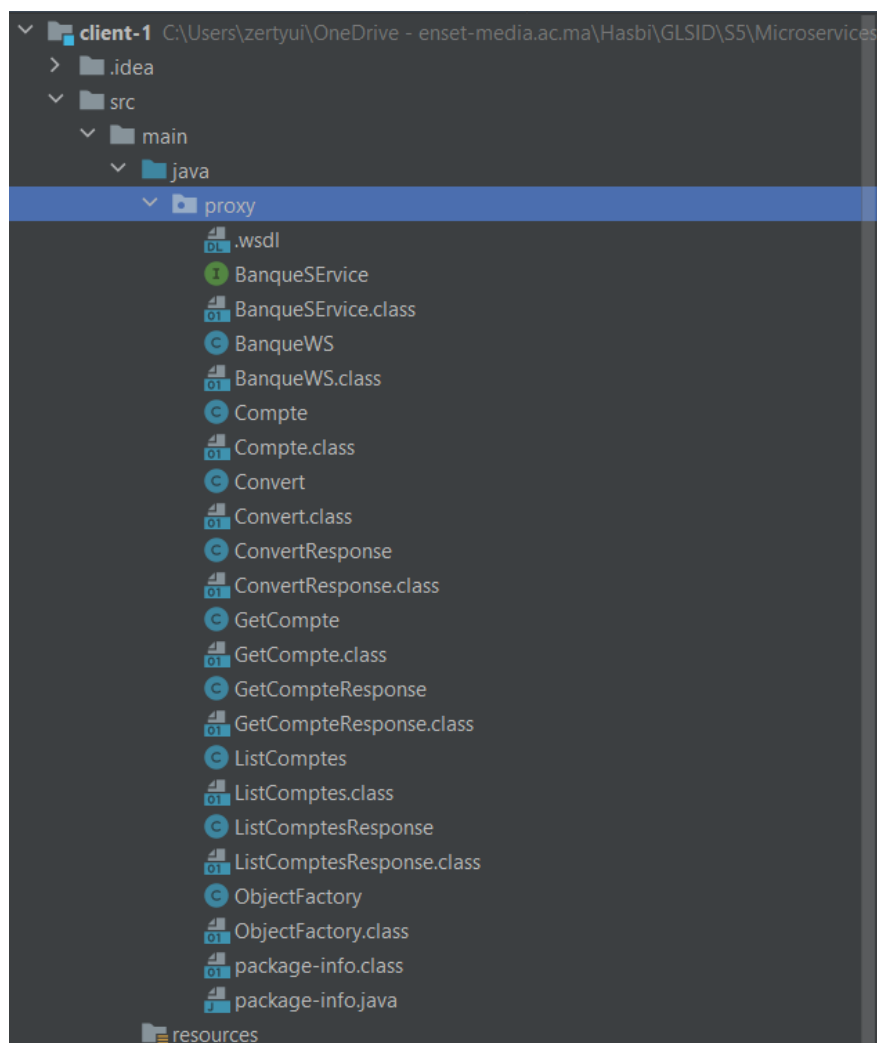
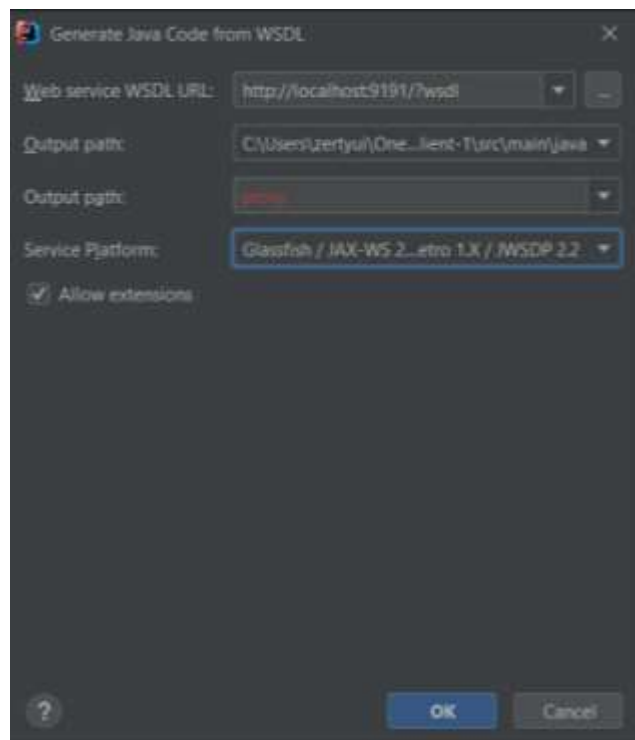
```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <ws:listComptes>
    </ws:listComptes>
  </soapenv:Body>
</soapenv:Envelope>
```

**Raw XML**

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:listComptesResponse xmlns:ns2="http://ws/">
      <return>
        <code>1</code>
        <dateCreation>2022-10-24T12:50:19.670+01:00</dateCreation>
        <solde>1094.3834811914023</solde>
      </return>
      <return>
        <code>2</code>
        <dateCreation>2022-10-24T12:50:19.670+01:00</dateCreation>
        <solde>34507.77874868649</solde>
      </return>
      <return>
        <code>3</code>
        <dateCreation>2022-10-24T12:50:19.670+01:00</dateCreation>
        <solde>27244.635509954278</solde>
      </return>
    </ns2:listComptesResponse>
  </S:Body>
</S:Envelope>
```

Client :

## Génération de proxy





```

import proxy.BanqueService;
import proxy.BanqueWS;
import proxy.Compte;

public class Client {
    public static void main(String[] args) {
        BanqueService stub = new BanqueWS().getBanqueServicePort();
        System.out.println(stub.convert( 5600));
        Compte cp=stub.getCompte(3);
        System.out.println(cp.getCode());
        System.out.println(cp.getSolde());
    }
}

```

The screenshot shows an IDE with a project named 'client-1'. The file 'Client.java' is open, displaying the same code as the first block. The 'Run' tab at the bottom shows the execution output for 'Client'.

```

Run: Client
"C:\Program Files\Java\jdk-17\bin\java.exe" ...
61600.0
3
18317.108187605132
Process finished with exit code 0

```