Rapport TP SOAP

Partie 1 : Programme

Partie 1.1 : Application.java

```java
import javax.xml.ws.Endpoint;
public class Application {
    public static void main(String[] args) {
        System.out.println("Début de déploiement de mon service");
        String url= "http://localhost:8888/";
        Endpoint.publish(url, new MonserviceWeb());
        System.out.println("Le service web est déployé");
    }
}
```

Partie 1.1.1 : Rôle

C'est votre classe principale. Celle que vous exécuterez pour lancer votre WebService SOAP.

Partie 1.1.2 : Explication du programme

Le programme commence par afficher la phrase « Début de déploiement de mon service » sur la console. Une fois sa tâche complété il y affichera également « Le service web est déployé ».

Il va ensuite utiliser Endpoint.publish(String URL, Object WebService) démarre un serveur HTTP sur l'URL spécifié utilisant le WebService spécifié. Les arguments utilisés sont http://localhost:8888/ et Monserviceweb().

Partie 2 : Monserviceweb.java

```java
import javax.jws.WebParam;
import javax.jws.WebService;
import javax.jws.WebMethod;
@WebService(targetNamespace="http://www.polytech.fr")  1 usage
public class MonserviceWeb {
    @WebMethod(operationName="convertir")  no usages
    public double conversion(double mt) { return mt * 0.9; }
    public double somme (@WebParam(name="param1") double a, double b) { return a*b; }

    public Etudiant getEtudiant(int identifiant){  no usages
        return new Etudiant( identifiant: 1,  nom: "Mario",  moyenne: 19);
    }
```

Partie 1.1.1 : Rôle

C'est la classe détaillant les méthodes utilisées par votre WebService SOAP.

Partie 1.1.2 : Explication du programme

Un site web peut utiliser plusieurs WebService SOAP. Rien n'empêche deux service de posséder deux méthodes portant le même nom et le même nombre et types de variables.

C'est pour éviter la confusion entre ces services que l'on précise @WebService(targetNamespace= String Url).

Cette classe possède trois méthodes :

1) Conversion : Prend un Double en paramètre et retourne la multiplication de celui-ci et 0,9.
2) Somme : Prend deux Double en paramètre et retourne leur somme.
3) GetEtudiant : Prend un Int en paramètre et retourne un objet de classe Etudiant. Celui-ci a comme identifiant 1, comme nom « Mario » et comme moyenne 19.

La mention de @WebMethod(operationName= String Name) avant la méthode « conversion » permet de renommer celle-ci en « convertir » aux yeux du client SOAP et du WSDL.
La mention de @WebParam(name= String Name) avant le paramètre « a » dans la méthode somme permet de renommer celui-ci en « param1 » aux yeux du client SOAP et du WSDL.

Partie 3 : Etudiant.java

```java
import javax.xml.bind.annotation.XmlRootElement;
import java.io.Serializable;

@XmlRootElement  2 usages
public class Etudiant {
    private int identifiant;  3 usages
    private String nom;  3 usages
    private double moyenne;  3 usages

    public Etudiant(){}  no usages

    public Etudiant(int identifiant, String nom, double moyenne){  1 usage
        this.identifiant=identifiant;
        this.nom=nom;
        this.moyenne=moyenne;
    }

    public int getIdentifiant() { return identifiant; }

    public void setIdentifiant(int identifiant) { this.identifiant = identifiant; }

    public String getNom() { return nom; }

    public void setNom(String nom) { this.nom = nom; }

    public double getMoyenne() { return moyenne; }

    public void setMoyenne(double moyenne) { this.moyenne = moyenne; }
}
```

Partie 1.1.1 : Rôle

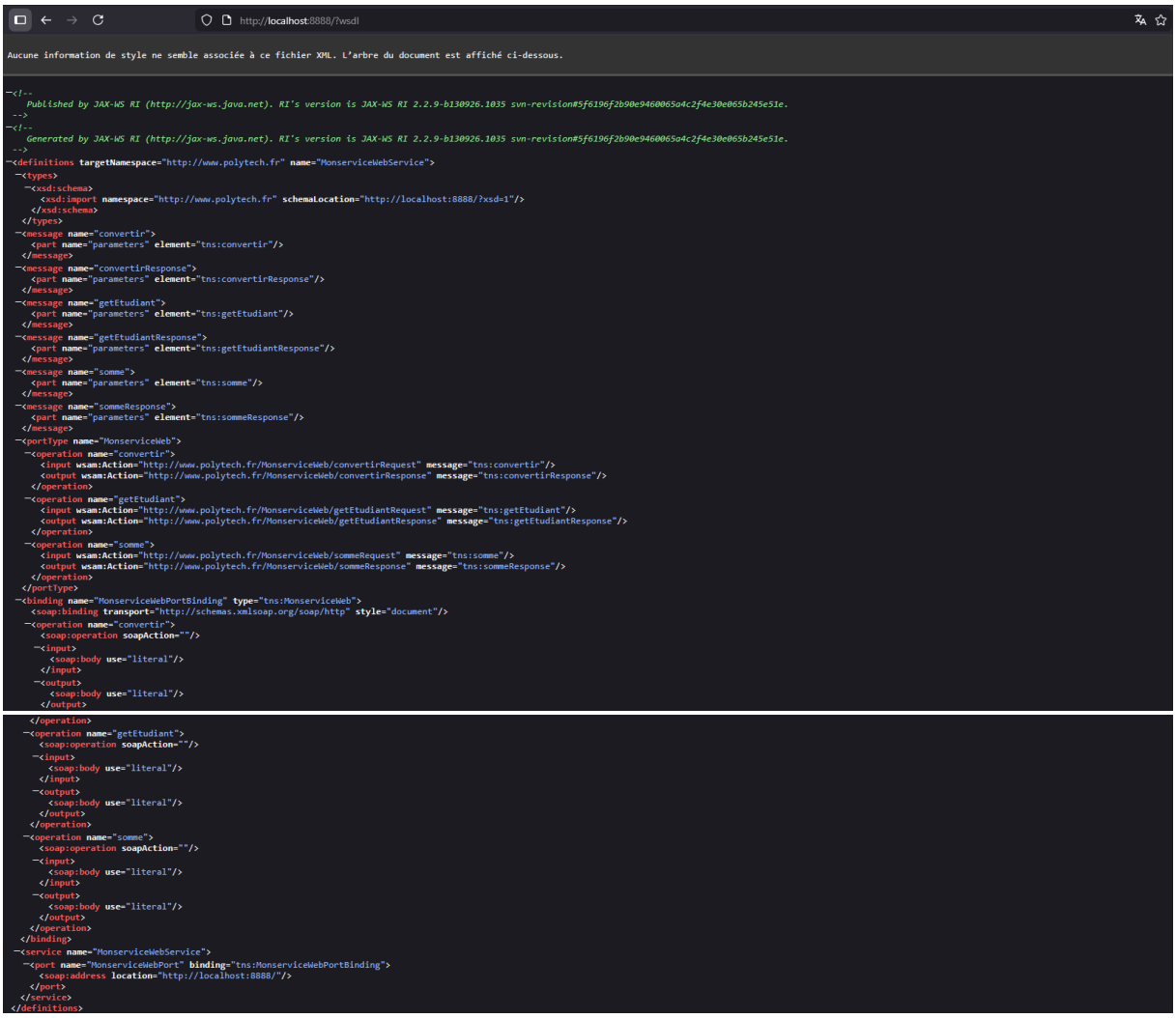
C'est la classe détaillant ce qu'est un étudiant.

Partie 1.1.2 : Explication du programme

Un étudiant possède un identifiant (Int), un nom (String) et une moyenne (Double).
Cette classe possède un constructeur demandant un Int, un String et un Double.
Les méthodes suivantes sont de simples Getters et Setters, un pour chacun des paramètres.
On crée un constructeur vide qui sera utile plus tard pour les injections de dépendance.

Partie 2 : Page Web

En tapant alors http://localhost:8888/?wsdl vous devrez avoir une page ressemblant à celle-ci.



Partie 3 : Utilisation avec Postman

Utiliser la méthode POST avec comme argument http://127.0.0.1/8888/.

Dans les Headers, préciser que :

1) Content-Type = text/xml=utf-8
2) SOAPAction = ""

Enfin dans le body, utiliser les arguments suivants pour utiliser la méthode getEtudiant avec l'argument 1.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:pol="http://www.polytech.fr">
    <soapenv:Header/>
    <soapenv:Body>
        <pol:getEtudiant>
            <arg1>1</arg1>
        </pol:getEtudiant>
    </soapenv:Body>
</soapenv:Envelope>
```