ECOLE NATIONALE SUPÉRIEURE D'INFORMATIQUE

TP N°4 : Réalisation de Web services avec JAX-WS et JAX-RS

Vous allez programmer un Web service qui expose deux opérations :

- Une opération *ajouterLivre* qui prend en paramètre un objet de type String et un ensemble d'objets de type Auteur, sans retour.
- Une opération *listerTousLesLivres* sans paramètres en entrée et qui retourne une liste d'objets de type Livre.

Première partie

Vous allez programmer un Web service SOAP en utilisant l'API JAX-WS.

Etape 1 : Création du Web service SOAP

Dans le projet EJB que vous avez déjà réalisé, créez un nouveau package que vous appellerez *service*. Ensuite, créez une classe Java appelée GestionLivresWS à laquelle vous ajouterez les annotations suivantes : @WebService, @WebMethod et @WebParam.

GestionLivresWS.java

```java
package service;

import java.util.List;
import javax.ejb.EJB;
import javax.ejb.Stateless;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;
import entites.Auteur;
import entites.Livre;
import session.GestionBeanLocal;

@Stateless
@WebService
public class GestionLivresWS {
    @EJB
    private GestionBeanLocal gestion;

    @WebMethod
    public void ajouterLivre(@WebParam(name = "intitule") String intitule,
            @WebParam(name = "auteurs") List<Auteur> auteurs) {
        Livre livre = new Livre(intitule, auteurs);
        gestion.ajouterLivre(livre);
    }

    @WebMethod
    public List<Livre> listerTousLesLivres() {
        return gestion.listerTousLesLivres();
    }
}
```

Etape 2 : Test

Pour tester le bon fonctionnement du Web service, créez un projet Java puis une classe qui contiendra le code ci-après.

ClientWS.java

```
1 import service.GestionLivresWS;
9
10 public class ClientWS {
11
12     public static void main(String[] args) {
13         GestionLivresWS gestion = new GestionLivresWSService().getGestionLivresWSPort();
14         List<Auteur> auteurs = new ArrayList<Auteur>();
15         Auteur auteur = new Auteur();
16         auteurs.add(auteur);
17         gestion.ajouterLivre("java", auteurs);
18         List<Livre> livres = gestion.listerTousLesLivres();
19         for (Livre lv : livres) {
20             System.out.println(lv.getIntitule());
21         }
22
23     }
24
25 }
```

Vous allez donc appeler les méthodes du Web service déjà créé à l'aide d'un Proxy. Ce dernier peut être généré automatiquement en utilisant l'outil *SOAPUI* comme suit :

1. Cliquez sur Tools/JAX-WS Artifacts
2. Précisez les trois champs : WSDL, Target Directory et Source Directory
3. Cliquez Generate.

Désormais, il vous sera possible d'appeler votre Web service au sein de votre client Java.

Deuxième partie

Vous allez programmer un Web service RESTful en utilisant l'API JAX-RS.

Etape 1 : Création du Web service RESTful

GestionLivresRS.java

```
import session.GestionBeanLocal;
import javax.ejb.EJB;
import javax.ejb.Stateless;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;

@Stateless
@Path("GestionLivres")
public class GestionLivresRS {
    @EJB
    private GestionBeanLocal gestion;

    @GET
    @Path("Livre/{intitule}/{code1}/{nom1}/{code2}/{nom2}")
    public void ajouterLivre(@PathParam(value = "intitule") String intitule, @PathParam(value = "code1")
            @PathParam(value = "nom1") String nom1, @PathParam(value = "code2") String code2,
            @PathParam(value = "nom2") String nom2) {
        List<Auteur> auteurs = new ArrayList<Auteur>();
        Auteur auteur1 = new Auteur(code1, nom1);
        auteurs.add(auteur1);
        Auteur auteur2 = new Auteur(code2, nom2);
        auteurs.add(auteur2);
        Livre l = new Livre(intitule, auteurs);
        gestion.ajouterLivre(l);
    }

    @GET
    @Path("Livres")
    @Produces(MediaType.APPLICATION_JSON)
    public List<Livre> listerTousLesLivres() {
        return gestion.listerTousLesLivres();
    }
}
```

Il faudrait aussi ajouter le fichier « ApplicationRS.java » dont le contenu est le suivant :

ApplicationRS.java

```
import javax.ws.rs.ApplicationPath;
import javax.ws.rs.core.Application;

@ApplicationPath("/")
public class ApplicationRS extends Application {

}
```

Etape 2 : Test

- Lancez le navigateur et testez la méthode ajouterLivre en saisissant son URI ainsi que les paramètres d'entrée, par exemple : http://localhost:8080/GestionLivresRS/GestionLivres/Livre/Java/GOS12/Gosling/NAU456/Naughton
- Testez également la méthode ListerTousLesLivres en tapant son URI (http://localhost:8080/GestionLivresRS /GestionLivres/Livres).