Département Informatique
Master M1 TC 2020/2021

Oran le 29/05/2021

TP3 :
Création de services Web REST

1. Objectifs

L'objectif de ce TP est de créer des services web REST avec JAX-RS, les deployer et écrire les clients qui appellent ces services.

Vous allez dans un premier temps développer un service REST simple ensuite vous allez créer un service Web REST permettant de :

1- Gérer plusieurs listes des Etudiants (M1 SITW)
2- Gérer plusieurs listes de Livres (M1 RSD)
3- Gérer plusieurs listes de Services (M1 IIEP)
4- Gérer plusieurs listes des Banques (M1 ADSI)

Vous manipulerez plusieurs technologies : JAXB (Java Architecture for XML Binding) et JAX-RS (Java API for RESTful Services).

Tutoriaux sur Web service Restful sur netbeans est le suivant :
https://netbeans.apache.org/kb/docs/websvc/rest.html

2- Présentation

Jusqu'à présent, vous avez vu la création et le déploiement de services Web basés sur le protocole SOAP. Mais une autre approche des services Web et qui devient de plus en plus populaire existe: REST. Le concept introduit en 2000 ne fait qu'utiliser les principes fondamentaux du Web. Utilisant le protocole HTTP, il permet l'envoi de messages sans enveloppe SOAP et dans un encodage libre (XML, JSON, binaire, simple texte). Il est actuellement très utilisé par les sites communautaires (ou réseaux sociaux) leur permettant de proposer à leurs clients une API facile à utiliser. Des sites comme Flickr, Facebook, Last.fm, Amazon proposent ainsi de telles API évitant à leurs clients de devoir passer par la case SOAP.

2. Architecture

L'information de base, dans une architecture REST, est appelée ressource. On accède à une ressource (par son URI unique) pour procéder à diverses opérations (GET lecture / POST écriture / PUT modification / DELETE suppression), opérations supportées nativement par HTTP.

Création d'un Service Web REST simple

Les services REST sont spécifiés par le JCP (Java Community Process) sous le nom JAX-RS[1] (Java API for RESTful Services) (https://github.com/jax-rs). Cette spécification précise ce que peut ou doit faire une implémentation, comme pour toutes ces spécifications. L'implémentation de référence que l'on

utilise est Jersey (https://eclipse-ee4j.github.io/jersey/). Jersey est installé en standard dans un serveur JEE (tel que Glassfish ou JBoss), et peut s'installer dans un serveur Tomcat.

Vous allez créer un premier service web REST. Pour ce premier service Web, il s'agit simplement de montrer l'utilisation de celui-ci à partir d'un navigateur Web.

Vous allez suivre les e tapes suivantes :

- Créer un nouveau projet : Java Web→ Web Application que vous nommerez « WebServiceRest », vous choisirez Glassfish comme serveur.
- Créer un nouveau package que vous nommerez comme vous le souhaitez.
- Créer un service web Rest en cliquant sur New→Other→WebService→RESTful WebServices from Pattern. Sélectionner "Simple Root Resource". Dans la fenêtre suivante, il faut rentrer les informations permettant de configurer le service Web REST :
  - ✓ Un nom de package (Resource Package)
  - ✓ Un chemin d'accès qui permet d'accéder au service Web lorsqu'il est déployé (Path) et qui est à ajouter à l'URL donnée lors du déploiement de l'application Web.
  - ✓ Un nom de class (Class Name)
  - ✓ Un type MIME pour le format de réponse du service Web. Modifier MIME Type en sélectionnant « text/plain ».

- Netbeans s'est chargé de la gestion des ressources en générant une sous classe de « javax.ws.rs.core.Application ». Il faut savoir que vous avez également la possibilité de réaliser vous-même la gestion des ressources en configurant Jersey.
- Une nouvelle classe est créée par défaut « GenericResource ». Vous allez supprimer le code par défaut et le remplacer par le code suivant :

```
import javax.ws.rs.core.Context;
import javax.ws.rs.core.UriInfo;
import javax.ws.rs.PathParam;
import javax.ws.rs.Consumes;
import javax.ws.rs.PUT;
import javax.ws.rs.Path;
import javax.ws.rs.GET;
import javax.ws.rs.Produces;

@Path("Mypath")
public class GenericResource {

    @GET
    @Produces("text/plain")
    public String getText() {
        return "Mon premier service REST";
    }
}
```

- Il s'agit d'un **service REST simple** qui invoque la méthode **getText** qui retourne une réponse sous format texte "**Mon premier service REST**".
-

- L'annotation @Path (https://eclipse-ee4j.github.io/jersey/) désigne d'une part l'URI d'accès au service et à ses sous-services, et d'autre part la présence de paramètres dans cette URI.

- Cinq annotations sont définies, qui peuvent annoter certaines méthodes d'un service REST : @GET, @POST, @PUT, @DELETE, et @HEAD. Elles correspondent aux cinq méthodes HTTP qui portent le même nom. On ne peut poser chaque annotation qu'une seule fois sur une unique méthode dans une classe donnée, pour un chemin d'accès donné. La liste des annotations est disponible à l'adresse suivante :

  https://docs.oracle.com/javaee/6/tutorial/doc/gilik.html

Bon courage