Architectures distribuées TP 1 et 2

Création d'un service RESTFUL

Arnaud Saval
arnaud.saval@gmail.com

Clément Caron
caron.clement@gmail.com

I) Introduction

Dans cette introduction aux travaux pratiques d'architectures distribuées nous allons commencer par l'appel de services web. Dans un premier temps, nous allons voir comment récupérer des informations fournies par un service web géographique qui retrouve la ville / le bâtiment le plus proche en fonction des coordonnées géographiques : Geonames.
La documentation de ce service est disponible à cette adresse :

http://www.geonames.org/export/web-services.html#findNearby

Un exemple d'appel à ce service est disponible sur cette page :

http://jsfiddle.net/6opts5de/

a) Faites un Fork de cet exemple

b) Remplacez le web service appelé par sa version JSON et affichez la position exacte du bâtiment / de la ville renvoyée (sur la carte pour des points supplémentaires)

c) Ajoutez le web service wolfram|alpha pour obtenir plus d'information sur la ville/bâtiment

http://products.wolframalpha.com/api/explorer.html

2) Création d'un service

Dans Eclipse, créez un projet java et recopiez le code ci-dessous. Montrez le résultat obtenu à l'enseignant et expliquez ce que vous avez compris de l'exercice.

```
import java.io.StringReader;
import javax.xml.transform.Source;
import javax.xml.transform.stream.StreamSource;
import javax.xml.ws.*;

@WebServiceProvider
@ServiceMode(value = Service.Mode.PAYLOAD)
public class REST implements Provider<Source> {
	public Source invoke(Source source) {
		String replyElement = new String("<p>hello world</p>");
		StreamSource reply = new StreamSource(new StringReader(replyElement));
		return reply;
	}

	public static void main(String args[]) {
		Endpoint e = Endpoint.create(HTTPBinding.HTTP_BINDING, new REST());
		e.publish("http://127.0.0.1:8084/hello/world");
	}
}
```

a) Affichez le message **« Université de Rouen »** à l'adresse :

http://127.0.0.1:8090/test

b) Affichez le message **« Réponse du service REST »** à l'adresse :

`http://127.0.0.1:8084/hello/world`

3) Création d'un service RESTFUL

Forkez le projet disponible à l'URL https://gitlab.com/ad2020/TP-RESTFUL et importez-le comme un nouveau projet Java. Il contient les bases d'un service et d'un client RESTFUL que vous devez compléter.
Le service offre actuellement les fonctions suivantes :

MÉTHODE	URL	BODY	DESCRIPTION
GET	/animals		Retourne l'ensemble des animaux du centre
POST	/animals	Animal	Ajoute un animal dans votre centre
GET	/animals/{animal_id}		Retourne l'animal identifié par {animal_id}

Vous devez au moins fournir les méthodes suivantes :

MÉTHODE	URL	BODY	DESCRIPTION
PUT	/animals	??????	Modifie l'ensemble des animaux
DELETE	/animals		Supprime l'ensemble des animaux
POST	/animals/{animal_id}	Animal	Crée l'animal identifié par {animal_id}
PUT	/animals/{animal_id}	Animal	Modifie l'animal identifié par {animal_id}
DELETE	/animals/{animal_id}		Supprime l'animal identifié par {animal_id}
GET	/find/byName/{name}		Recherche d'un animal par son nom
GET	/find/at/{position}		Recherche d'un animal par position
GET	/find/near/{position}		Recherche des animaux près d'une position
GET	/animals/{animal_id}/wolf		Récupération des info. Wolfram d'un animal
GET	/center/journey/from/{position}		Récupération des info. Du trajet depuis une position GPS jusqu'à votre centre en utilisant le service Graphhopper.

Il est également possible d'améliorer davantage le projet, toute bonne idée implémentée sera récompensée.

Vous aurez à rendre ce projet en binôme pour le 26 Mars 2020. Tout retard sera pénalisé suivant une règle de calcul exponentielle (1 point, 3 points, 6 points, etc.).

Le projet est à rendre par email sous la forme d'une adresse vers un projet **GITLAB** privé. Vous ajouterez vos encadrants (les comptes sont **ccaron** et **asaval**) pour qu'ils puissent avoir accès à votre projet en vous assurant que celui-ci contienne :

1. Les noms et prénoms de votre binôme dans un fichier **AUTEURS** à la racine.
2. Le **lien** de votre JS Fiddle
3. Le **code** de votre service RESTFUL
4. Une documentation de votre service dans un fichier **README** à la racine.
5. Un **rapport** sur les problèmes rencontrés et les solutions que vous y avez apportés.
6. Un **exemple d'utilisation** de votre service déroulant les différentes actions utilisateurs définies dans le **scénario**,

Toute configuration « en dur » (de type *C:\Users\Bobby\etc*), est à proscrire, et sera pénalisé, de même que l'absence de commentaires dans votre code.