Services Web de type REST
Développement sous C# .Net

1 Introduction

1.1 Bref rappel du Cours « du Web au Web services » (SI3)

Les services web de type REST sont les plus simples et les plus intuitifs qui soient pour peu que l'on ait pratiqué la gestion de pages web dynamiques avec un serveur Web et le protocole HTTP.

En effet, dans le cas de pages web dynamiques les données de réponse à un GET ou à un POST, sont générées à la volée côté serveur (par un cgi-bin par exemple). Elles sont destinées à une famille de clients bien définies : les browsers web (ex. Chrome, Firefox, IExplorer, …). Les formats des données renvoyées par le serveur sont donc contraintes par les capacité des clients à les lire, parser et afficher.

Dans le cas des services Web, les clients peuvent être toute sorte d'applications et sont seulement contraintes par le protocole d'échange utilisé pour communiquer avec le serveur Web (souvent du JSON, de l'XML ou du CSV). C'est ainsi que les Services Web peuvent se retrouver au cœur du développement d'applications réparties.

1.2 Les 5 principes REST

REST (Representational State Transfer) ou RESTful est donc un style d'architecture permettant de construire des applications réparties. Il s'agit d'un ensemble de conventions et de bonnes pratiques à respecter et non d'une technologie à part entière. L'architecture REST utilise les spécifications originelles du protocole HTTP, plutôt que de réinventer une surcouche (comme nous le verrons avec les web services WS-SOAP).

Principe n°1 : l'URI est l'identifiant des ressources

Principe n°2 : les verbes HTTP comme opérations sur les ressources :

- Créer (create) => POST
- Afficher (read) => GET
- Mettre à jour (update) => PUT
- Supprimer (delete) => DELETE

Principe n°3 : les réponses HTTP comme représentation des ressources (contenu des ressources codé en JSON, XML, CSV ou autre)

Et parfois les principes n°4 et n°5 avec des liens comme relation entre les ressources et un paramètre comme jeton d'authentification pour les accès contrôlé à certain services web (ex. Google API).

1.3 Utilitaire cURL de Test :

Dans la suite nous utiliserons Curl (outil en ligne de commande) qui permet d'accéder à toutes les commandes HTTP et qui est donc un outil idéal pour tester les API REST.

Si besoin, vous pouvez le télécharger sur http://curl.haxx.se/download.html, en particulier pour Windows.

Voici quelques exemples d'utilisation de cURL :

Pour déposer un document, envoyer un PUT au « EndPoint » identifié par l'URI du document comme paramètre. Une extension de fichier .xml, .json, ou .txt indique son type. Vous pouvez remplacer ce dernier par un paramètre contentType.
Par exemple, nous déposons ici un document au format JSON à l'URI /afternoon-drink.json :

```
curl -X PUT -d '{"name": "Iced Mocha", "size": "Grandé", "tasty": true }' 'http://myhost/store?uri=/afternoon-drink.json'
```

Pour obtenir un document à partir de son URI, vous pouvez utiliser une requête GET:

```
curl -X GET 'http://myhost/store?uri=/afternoon-drink.json'
```

Pour mettre à jour un document, vous pouvez envoyer une requête POST :

```
curl -X POST -d '{"name": "Iced Tea", "size": "Jumbo", "refreshing": true }' 'http://myhost/store?uri=/afternoon-drink.json'
```

Pour supprimer un document, vous pouvez envoyer une requête DELETE :

```
curl -X DELETE 'http://myhost/store?uri=/afternoon-drink.json'
```

2 Un premier Client REST en C#

2.1 Les services REST et l'Open Data

L'open data ou donnée ouverte est une donnée numérique dont l'accès et l'usage sont laissés libres aux usagers. Elle peut être d'origine publique ou privée, produite notamment par une collectivité, un service public (éventuellement délégué) ou une entreprise. Elle est diffusée de manière structurée selon une méthode et une licence ouverte garantissant son libre accès et sa réutilisation par tous, sans restriction technique, juridique ou financière.

L'ouverture des données (open data) est à la fois un mouvement, une philosophie d'accès à l'information et une pratique de publication de données librement accessibles et exploitables.

Elle s'inscrit dans une tendance qui considère l'information publique comme un bien commun (tel que défini par Elinor Ostrom) dont la diffusion est d'intérêt public et général.

Au Sommet du G8 2013, les chefs d'État du G8 ont signé une « Charte du G8 pour l'ouverture des données publiques ». En Europe et dans certains pays, des directives et lois imposent aux collectivités de publier certaines données publiques sous forme numérique.

2.2 La Gestion des Vélibs de la ville de Paris …

Un grand nombre de communes françaises se sont regroupées dans une association OpenDataFrance. Elle a pour but de regrouper et soutenir les collectivités territoriales françaises engagées activement dans une démarche d'ouverture des données publiques et de favoriser les démarches entreprises par ces collectivités pour la promotion et le développement de l'open data en France

A l'instar de nombreux organismes publics, la ville de Paris s'inscrit dans la démarche de l'Open Data.

Nous allons utiliser dans la première partie de ce TD un service web fourni par la ville de Paris pour la gestion des Vélibs.

L'application présentée permet de visualiser les disponibilités des vélos et des emplacements de parking à la disposition des parisiens dans le cadre du service Vélib géré par la ville de Paris :

a) La liste des points d'accès Vélib et leurs localisations

b) La dispo des Vélibs en temps réel à chaque point d'accès

question 1 : A l'aide de Curl récupérer la liste des points d'accès Vélib à l'URI http://www.velib.paris.fr/service/carto. Quel est le langage de description du contenu de la ressource ?

Les URI de la forme http://www.velib.paris.fr/service/stationdetails/<number> fournissent les disponibilités des points d'accès. Sur chaque point d'accès <number>, le champ <available> donne le nombre de vélos disponibles, le champ <free> le nombre d'emplacements libres, le champ <total> donne le nombre d'emplacements, le champ <ticket> à 1 si le paiement par carte bleue est accepté. Les trois champs restants sont non documentés.

question 2 : A l'aide de Curl, récupérer les informations sur le point d'accès MICHEL CHASLES-GARE DE LYON. Combien de Vélibs sont disponibles ? Combien de dépôts sont en panne ?

2.3 Client de Service Web en C# .Net

question 3 : En reprenant les informations précédentes, écrire un client qui vous donne le nombre de Vélibs disponibles à un point d'accès dont vous indiquerez le nom complet. Le point d'accès sera identifié grâce à une sous-chaine présente dans son nom <name>. En cas de plusieurs points d'accès identifiés, prenez le premier trouvé.
Vous utiliserez pour cela la classe HttpWebRequest de System.Net (https://msdn.microsoft.com/fr-fr/library/system.net.httpwebrequest(v=vs.110).aspx).

Pour vous aider voici un exemple de programme client simple :

```
using System;
using System.IO;
using System.Net;
using System.Text;

namespace Examples.System.Net
{
    public class WebRequestGetExample
    {
        public static void Main ()
        {
            // Create a request for the URL.
            WebRequest request = WebRequest.Create (
              "http://www.contoso.com/default.html");
            // If required by the server, set the credentials.
            // request.Credentials = CredentialCache.DefaultCredentials;
            // Get the response.
            WebResponse response = request.GetResponse ();
            // Display the status.
            Console.WriteLine (((HttpWebResponse)response).StatusDescription);
            // Get the stream containing content returned by the server.
            Stream dataStream = response.GetResponseStream ();
            // Open the stream using a StreamReader for easy access.
            StreamReader reader = new StreamReader (dataStream);
            // Read the content.
            string responseFromServer = reader.ReadToEnd ();
            // Display the content.
            Console.WriteLine (responseFromServer);
            // Clean up the streams and the response.
            reader.Close ();
            response.Close ();
        }
    }
}
```