Applications Réparties TD 3
Web Services .NET

Ce TD vous enseignera les bases de l'utilisation des Web Services en .NET avec Visual Studio .NET

1 Introduction

Vos applications .NET, quelles soient Windows Forms ou Web Forms, peuvent avoir recours, dans le but de vous simplifier la vie, à des **Web Services**.

Pour faire simple, vous pouvez imaginer ces Web Services comme des fonctions, stockées sur un serveur Web, et mise à la disposition de tout le monde. Vous pouvez donc, à partir de n'importe quel ordinateur disposant d'une connexion Internet, avoir accès à ces méthodes, mais vous ne pouvez pas les modifier ou bien même les visualiser : vous passer un ou plusieurs paramètres au Web Service, celui-ci effectue le traitement et vous renvoi ce qu'il faut.

En théorie, cela peut s'avérer un peu complexe d'où ce TD.

2 Création d'un Client à un Web Service existant

2.1 Ajout de la référence Web

Avant de pouvoir travailler avec un Web Service, il faut ajouter sa référence Web dans votre projet. Pour cela, vous devez connaître l'adresse Web du Web Service.

Nous utiliserons le site http://www.xmethods.com/ qui contient un bon nombre de Web Services et qui vous offre la possibilité de pouvoir les tester en ligne.

Une fois l'adresse du Web Service trouvée, faîtes un clic droit sur "Références" dans votre projet et choisissez « **Ajouter une référence Web** ».

Vous arrivez alors sur l'écran qui vous permet de saisir l'adresse du Web Service, puis vous cliquer sur le bouton « **Aller à** ».

A ce moment là, vous verrez une barre de progression s'afficher à l'écran : elle vous indiquera que vous vous connectez au Web Service, afin de pourvoir récupérer la liste des méthodes disponibles. Une fois cette liste récupérée, elle sera affichée à l'écran. Observez les biens.

Ensuite, vous n'avez plus qu'à cliquer sur le bouton « **Ajouter la référence** » pour pouvoir ajouter ce Web Service en tant que référence à votre projet. Une fois que la référence est correctement ajoutée, vous pouvez l'apercevoir dans l'explorateur de solutions.

Une fois cette partie terminée, vous pouvez passer à la partie codage à proprement parler.

2.2 Codage avec le Web Service

Pour utiliser les méthodes de votre Web Service, vous devez, comme avec le reste de la programmation orientée objet, créer un objet. Ce qui nous donne quelque chose comme ceci (cela peut varier en fonction du nom de la référence de votre Web Service <WS>):

```
<WS>.com.dataaccess.www.NumberConversion ob ;
ob = new WS.com.dataaccess.www.NumberConversion();
```

Vous pouvez maintenant appeler les méthodes de votre Web Service, méthode dont vous avez eu le nom

- sur le site Web du Web Service
- lors de la connexion au Web Service, avant l'ajout de la référence au projet

Applications Réparties TD 3
Web Services .NET

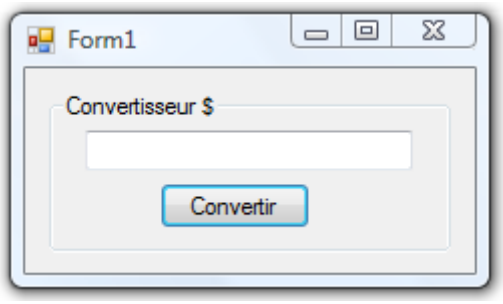
Dans notre cas, la méthode de notre Web Service prend en paramètre un entier qui est le nombre à convertir en dollars. Le résultat retourné est une chaîne de caractère (regardez l'aide sur NumberToDollars)

2.3 Application

Construisez une application qui prendra un nombre dans une textbox et le convertira en une somme en dollars après activation d'un bouton, en utilisant le Web Service :

http://www.dataaccess.com/webservicesserver/numberconversion.wso

Par exemple « 25 » donnera « twenty five dollars » dans votre application.



3 Création d'un Web Service

3.1 Une première version du Web Service

- Dans Visual Studio .NET, créer un nouveau projet de Service Web
  - Fichier > Nouveau > Projet
  - Types de projets : choisir « Projets Visual C# »
  - Modèles : choisir « Application de Service Web ASP.NET »
  - Nom du projet : « http://localhost/WebService1 »

- Regarder les fichiers qui ont été créés. Vous pouvez trouver :
  - App_WebRéférences : ne correspond pas à un fichier mais aux références sur les composants dont pourra avoir besoin votre Web Service
  - Le répertoire « bin » qui va contenir la version compilée en DLL de votre Web Service
  - « Service1.asmx » : votre Web Service avec son code source et ses ressources
  - « Web.config » ou « PrecompiledApp.config » : le fichier de configuration XML de votre application
  - « WebService1.vsdisco » : le fichier DISCO d'inspection du Web Service

- Afficher le code source du Web Service
  - Dans le panneau « Explorateur de solutions »
  - Sélectionner le fichier « Service1.asmx »
  - Cliquer sur l'icône « Voir le source »
  - Dans la zone centrale, vous avez le code source du fichier « Service1.asmx »
  - Prendre le temps de consulter ce code

- Créer une « WebMethod », remplacer la méthode d'exemple par :

Applications Réparties TD 3
Web Services .NET

```
[WebMethod]
public double EuroVersFranc(double Euro)
{
        return Euro * 6.55957;
}
```

Seul l'attribut [WebMethod] est nécessaire pour faire d'une méthode normale une méthode accessible par votre Web Service.

- Tester la solution (Ctrl+F5)
- Visual Studio a appelé Internet Explorer pour y afficher l'url de votre Web Service : http://localhost/Service1/Service1.asmx

La page web que vous voyez est automatiquement générée par le Framework .Net et vous permet de tester votre Web Service en utilisant votre navigateur. Elle contient la liste des méthodes de votre Web Service, donc ici uniquement la méthode « EuroVersFranc ».

Remarquez le message vous indiquant que le namespace par défaut n'a pas été redéfini.

- En cliquant sur le lien « Description du service » vous verrez s'afficher le document WSDL correspondant à votre Web Service.
- Dans la première page, cliquer sur le lien « EuroVersFranc » :

Vous accédez à une page web (également générée automatiquement par le Framework .NET) qui vous permet d'appeler le Web Service en donnant ses arguments. Vous voyez également quelles seraient les données envoyées au Web Service suivant trois méthodes possibles : SOAP 1.1, SOAP 1.2, HTTP POST.

Cette page permet de tester directement un Web Service en utilisant la méthode GET. Celle-ci ne peut fonctionner que pour les services dont les paramètres d'entrées sont une liste de types simples (ni tableaux, ni classes).

- Rentrer un prix en Euro et cliquez sur le bouton « Invoquer »
    - Le navigateur reçoit et affiche telle qu'elle la réponse XML du Web Service

- Modifier le « namespace » (espace de nom XML) de votre Web Service : Ajouter la ligne d'attribut suivante avant la ligne de déclaration de la classe « Service1 »

```
[WebService(Namespace="http://www.polytech.unice.fr/")]
public class Service1 : System.Web.Services.WebService
{
        public Service1()
…
```

- Compiler et tester à nouveau, le message sur le namespace par défaut disparaît

3.2 Ajout d'une seconde méthode Web

- Ajouter une méthode de conversion « FrancVersEuro »

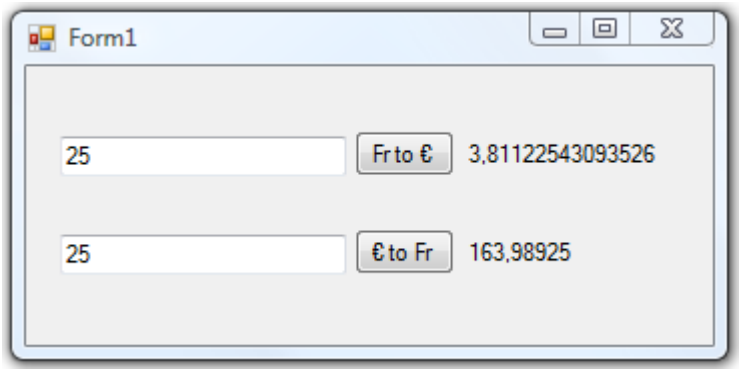
Applications Réparties TD 3
Web Services .NET

```
[WebMethod]
public double FrancVersEuro(double Franc)
{
        return Franc / 6.55957;
}
```

- Compiler puis Tester en navigant sur la page « Service1.asmx »: Vous avez cette fois-ci deux méthodes dans la page de présentation de votre Web Service
- Tester l'importance de l'attribut « WebMethod » ; Mettre en commentaire la ligne définissant l'attribut « WebMethod » sur la méthode « FrancVersEuro ». Vous avez alors :

```
//[WebMethod]
public double FrancVersEuro(double Franc)
{
        return Franc / 6.55957;
}
```

- Recompiler puis tester : Seule la méthode « EuroVersFranc » apparaît dans la liste des méthodes supportées par le Web Service
- Enlever le commentaire pour que l'on puisse utiliser cette méthode dans la suite des exercices
- Retourner voir votre projet SiteWeb de l'exercice 2 et utilisez cette nouvelle méthode (en ajoutant Ajouter 1 TextBox, 1 Label et 1 Bouton)



3.3 Un Web Service avec des paramètres de type complexe

Nous allons revenir sur le premier Web Service. Vous allez ajouter une méthode qui prend en paramètre un utilisateur (une classe simple avec trois champs : Prénom et Nom et Francs) pour convertir sa somme d'argent en francs en euros.
Essayer de lui donner le même nom que FrancVersEuro.
Tester cette nouvelle Web méthode.

4 Web Services Avancés

Le but est de créer et consommer des Services Web plus complexes : gestion des erreurs, retour de types complexes…

4.1 Gestion des erreurs d'un Service Web

- Dans Visual Studio .NET, créer un nouveau projet de Service Web
  - Fichier > Nouveau > Projet
  - Types de projets : choisir « Projets Visual C# »

Applications Réparties TD 3
Web Services .NET

- Modèles : choisir « Service Web ASP.NET »
- Nom du projet : « http://localhost/ServiceWebAvance »

- Dans le service « Service1.asmx », ajouter :

```
using System.Web.Services.Protocols;
```

- Créer une nouvelle WebMethod appelée « Division ». En cas de division par 0, la méthode lèvera une exception SOAP.
- Créer une application Windows C# .NET « ConsommationServiceWebAvance » pour accéder à ce service Web avec 2 TextBox et un Bouton. Le résultat s'affichera dans une MessageBox.
- Insérer un « Try… Catch » afin de gérer l'exception SOAP en cas de division par zéro et Tester.

4.2 Transmettre un flux XML

- Ajouter une WebMethod à votre Service Web qui retourne le fichier Web.Config du Service Web à l'application appelante.
- Dans l'application Windows cliente, ajouter un nouveau bouton pour appeler cette méthode. Le résultat s'affichera dans une MessageBox.

4.3 Transmettre une image

- Copier le fichier « Trafic.bmp ci-joint dans le répertoire « C:\Inetpub\wwwroot\ServiceWebAvance »
- Ajouter la référence « .NET », « System.Drawing.dll » à votre projet.
- Ajouter une nouvelle WebMethod au Service Web qui retournera le bitmap de cette image.
- Dans l'application Windows cliente, mettre ajouter un Bouton et un PictureBox pour tester cette méthode.
- N'oubliez pas de définir la propriété « SizeMode » du contrôle PictureBox à « StretchImage » afin que l'image se mette automatiquement à la taille du contrôle PictureBox.

5 Pour aller plus loin

Refaire la même application mais cette fois ci le transfert de binaire doit ce faire en attachement à un message SOAP avec DIME
http://msdn.microsoft.com/en-us/library/ms824597.aspx