Introduction à Silverlight 1.1

par Benjamin Roux (Retour aux articles)

Date de publication : 27/09/2007

Dernière mise à jour : 29/10/2007

Cet article est une introduction à Silverlight 1.1

1 - Introduction

Silverlight est l'une des nouvelles technologies nées de chez Microsoft.

Avec, il est possible de créer des applications web riches, des animations, des jeux#

Voici ce qu'il est, entre autres, capable de réaliser

http://www.tafiti.com/

Des exemples sont également disponibles sur le site officiel

http://www.silverlight.net

Silverlight se décline actuellement en 2 versions :

- ***1.0*** qui permet de faire des applications côté client, tout est en Javascript.
- ***2.0*** (anciennement 1.1) qui intègre une version du Framework .NET avec un CLR et qui permet donc d'exécuter du code C# ou VB.NET.

Dans cet article nous apprendrons donc comment utiliser la version 1.1.

Quelques outils sont requis pour développer en Silverlight 1.1.

Les outils requis :

- **Microsoft Silverlight 1.1 Alpha Refresh.**
- **Microsoft Visual Studio 2008 Beta 2.**
- **Microsoft Silverlight Tools Alpha for Visual Studio 2008 Beta 2.**

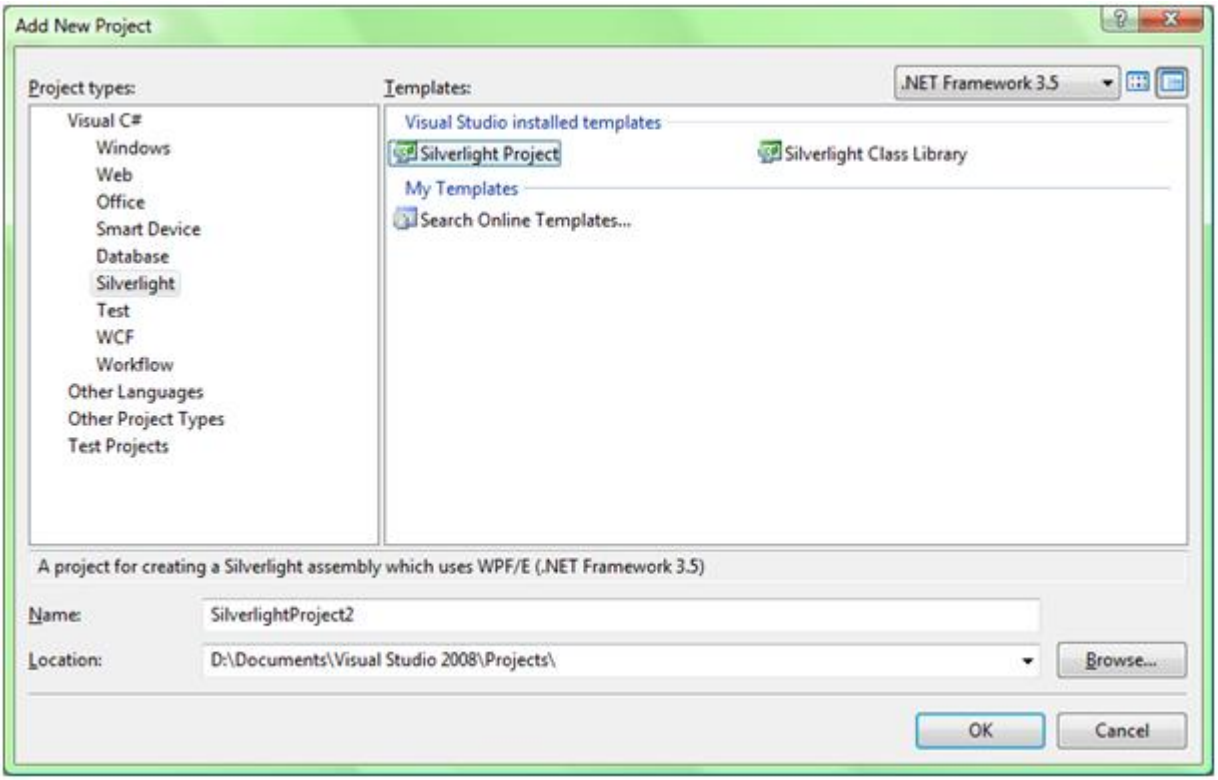
Une fois tous ces outils installés nous pouvons commencer.

Suite à l'écriture de cet article, j'ai décidé d'en écrire un deuxième pour illustrer ce que nous allons voir.

Réalisation d'un chat en Silverlight 1.1

2 - Création d'un projet Silverlight

Pour créer, il suffit d'ouvrir Visual Studio 2008 et de choisir un nouveau projet Silverlight.



Nouveau projet Silverlight

Plusieurs fichiers sont créés par défaut :

- *Page.xaml* : contient le code XAML de votre interface
- *Page.xaml.cs* : contient le code behind de votre interface (événements#)
- *TestPage.html* : une page HTML contenant un contrôle Silverlight
- *TestPage.html.js* : contient le code Javascript permettant de créer le contrôle Silverlight
- *Silverlight.js* : contient un code Javascript vérifiant entre autre si Silverlight est installé sur la machine cliente.

Pour lancer votre projet, c'est simple, clic droit sur *TestPage.html* puis clic sur **View in Browser**, ou bien **F5**.

3 - Intéragir avec les éléments HTML d'une page Web

Dans cette partie nous allons voir comment interagir avec les éléments HTML présents sur notre page, via notre code managé.

Créez un projet Silverlight (voir première partie) et dans le fichier *TestPage.html*, rajoutez 2 éléments HTML, un bouton et un textbox.

```
<div>
    <input type="text" id="name" />
    <input type="button" id="send" value="Send" />
</div>
```

Dans le fichier *Page.xaml.cs* remplacez ceci :

```
public partial class Page : Canvas
{
    public void Page_Loaded(object o, EventArgs e)
    {
        // Required to initialize variables
        InitializeComponent();
    }
}
```

Par ceci :

```
public partial class Page : Canvas
{
    public void Page_Loaded(object o, EventArgs e)
    {
        // Required to initialize variables
        InitializeComponent();

        // Get the HTML Document
        HtmlDocument document = HtmlPage.Document;

        // Get the 2 HTML elements by their ID
        HtmlElement textbox = document.GetElementByID("name");
        HtmlElement button = document.GetElementByID("send");

        if (textbox != null)
        {
            // we set the value attribute to DVP
            textbox.SetAttribute("value", "DVP");
        }
        if (button != null)
        {
            // we add an event to the button
            // when the button will be clicked the event onJump will be called
            button.AttachEvent("onclick", new EventHandler<HtmlEventArgs>(this.OnJump));
        }
    }

    private void OnJump(object sender, EventArgs e)
    {
        // we send the user to this URI
        HtmlPage.Navigate("http://broux.developpez.com");
    }
}
```

Maintenant nous pouvons lancer notre page, et essayer notre bouton.

Un clic sur le bouton doit vous envoyer sur **http://broux.developpez.com**

Comme vous venez de le voir, interagir avec les éléments HTML de notre page est des plus simples.
Nous pouvons modifier les attributs de nos éléments, ajouter des événements, en enlever et ce d'une manière tout de même assez simple.

Pour aller un peu plus loin, vous pouvez regarder les différentes Propriétés de la classe ***HtmlPage***.

4 - Communiquer avec d'autres applications Web

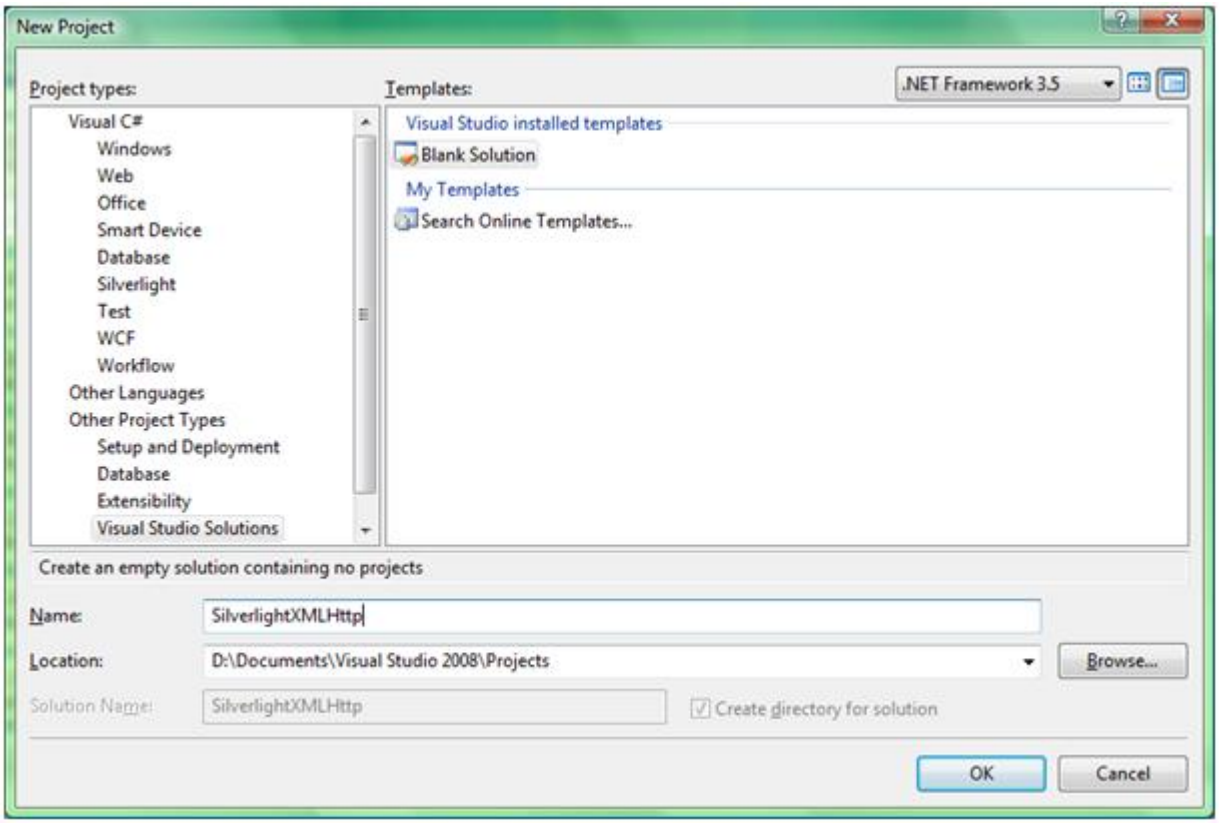
Comme dit plus haut, Silverlight 1.1 peut exécuter les langages .NET, on peut donc désormais communiquer avec d'autres applications Web. En revanche, il est impossible de communiquer avec des applications situées sur d'autres serveurs tout du moins directement et avec l'objet ***HttpWebRequest***.

Le plus simple reste de passer par un Web Service tournant sur le même serveur que votre application Silverlight.

Voici 2 méthodes :

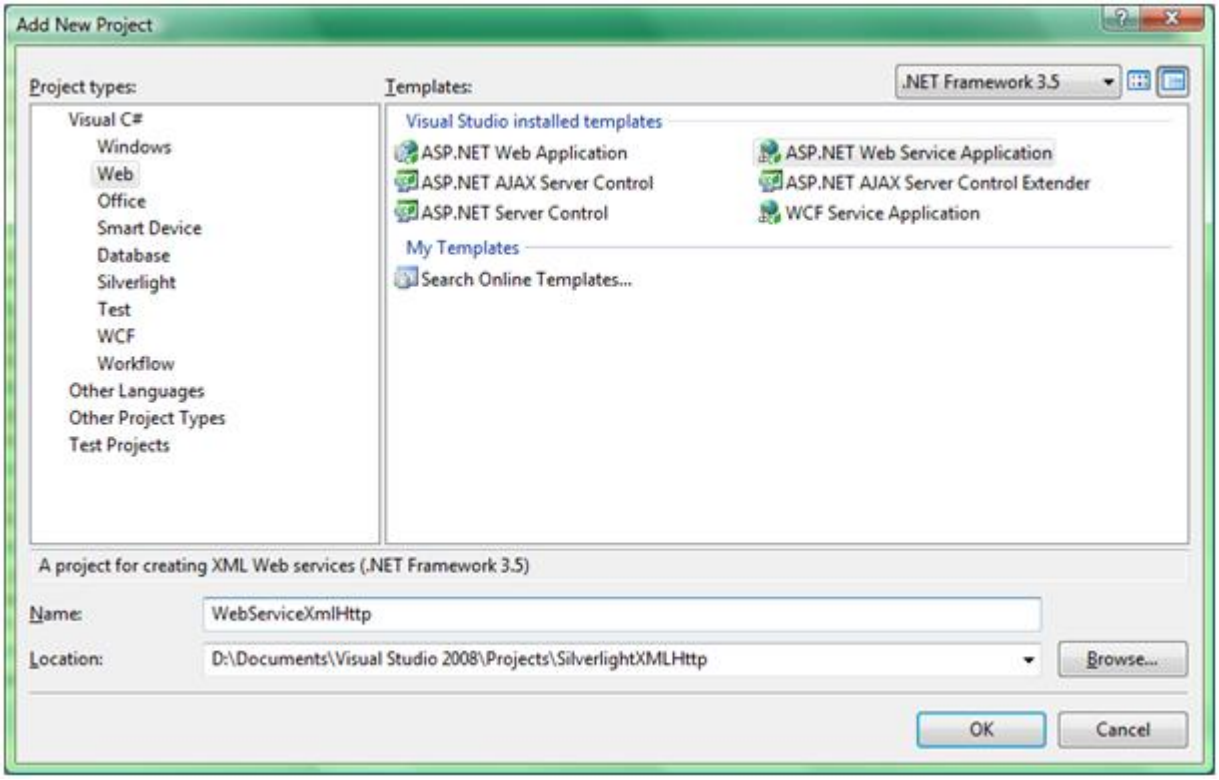
4-1 - Recevoir des réponses XML à travers HTTP

Tout d'abord il vous faut créer une solution vide dans VS 2008.



Solution vide

Une fois la solution créée, il faut créer un projet de type Web Service dans la solution.



Nouveau Web Service

Dans le fichier *Service1.asmx* crée, remplacez la méthode ***HelloWorld*** par celle-ci

```
[WebMethod]
[System.Web.Script.Services.ScriptMethod(UseHttpGet=true)]
public string HelloWorld(string input)
{
    if (input == null) input = "Anonym";
    return "Hello World " + input;
}
```

Rajouter ceci dans le web.config (avant la balise ***<httpHandlers>*** par exemple)

```
<webServices>
    <protocols>
        <add name="HttpGet"/>
        <add name="HttpPost"/>
    </protocols>
</webServices>
```

Il doit désormais être possible d'appeler votre méthode en GET et lui donner une valeur.

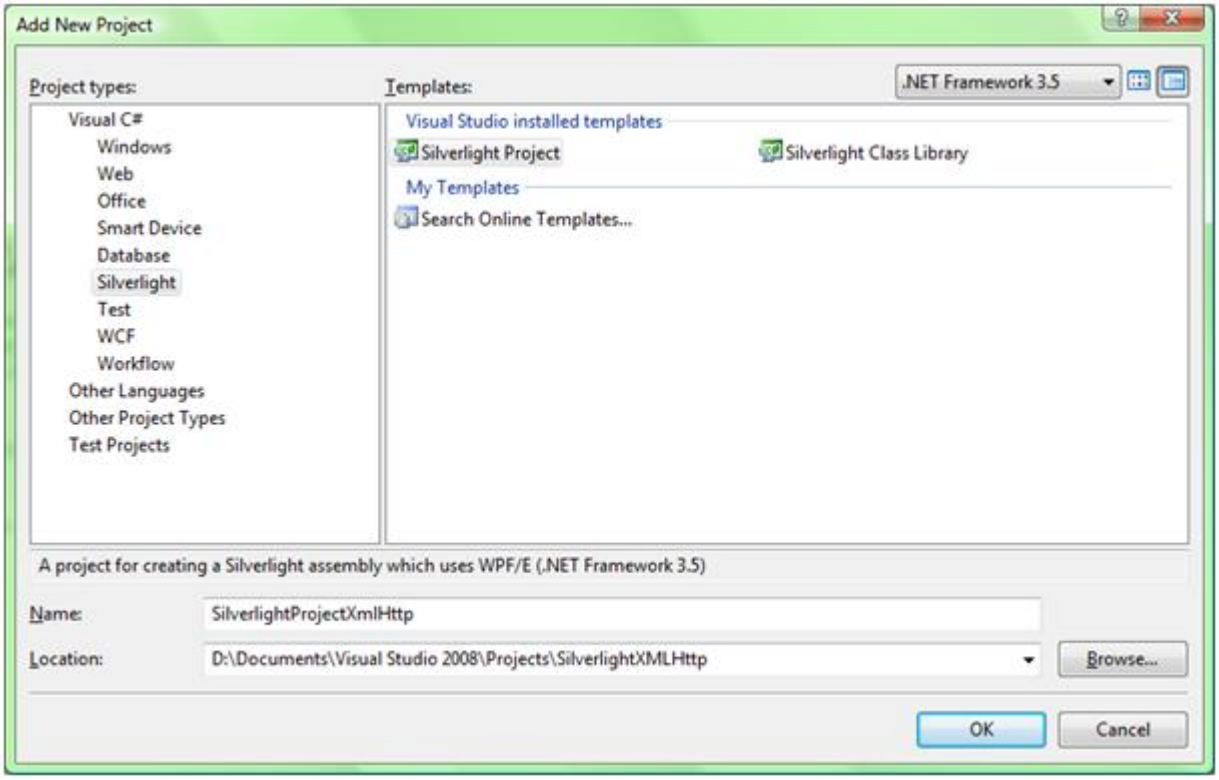
Essayez : **F5** et tapez dans le navigateur (remplacer le port par le vôtre)
http://localhost:50436/Service1.asmx/HelloWorld?input=Sky

Vous devriez obtenir une réponse de votre Web Service dans ce genre

```
<?xml version="1.0" encoding="utf-8" ?>
<string xmlns="http://tempuri.org/">Hello World Sky</string>
```

Une fois que vous avez ça, il ne nous reste plus qu'à développer notre application Silverlight.

Ajoutez donc un nouveau projet Silverlight à votre solution.



Nouveau projet Silverlight

Rajoutez ceci dans la balise ***<Canvas>*** du fichier *Page.xaml* généré.

```
<TextBlock x:Name="helloWorld" Name="helloWorld"></TextBlock>
```

Ceci rajoute seulement un *TextBlock* (Label) dans votre canevas, il nous permettra d'afficher le résultat du Web Service.

Ensuite, ouvrez le fichier *Page.xaml.cs* et remplacez ceci

```
public partial class Page : Canvas
{
    public void Page_Loaded(object o, EventArgs e)
    {
        // Required to initialize variables
        InitializeComponent();
    }
}
```

Par cela :

```
public partial class Page : Canvas
{
    // an BrowserHttpRequest for execute our HTTP Request
    private BrowserHttpWebRequest mRequest;

    public void Page_Loaded(object o, EventArgs e)
    {
        // Required to initialize variables
        InitializeComponent();

        try
        {
            // replace the port by yours
            mRequest = new BrowserHttpWebRequest(new Uri("http://localhost:50436/Service1.asmx/
HelloWorld?input=Sky"));
```

```
            // Get the response to the request
            // You can call it asynchronously too
            HttpWebResponse response = (HttpWebResponse)mRequest.GetResponse();

            // Read response
            StreamReader responseReader = new StreamReader(response.GetResponseStream());
            // Read all the responde (ie : the XML text returned)
            string allResponse = responseReader.ReadToEnd();
            // We create en XML Reader with the XML text returned
            XmlReader xr = XmlReader.Create(new StringReader(allResponse));
            // Method return string so we search the string markup
            xr.ReadToFollowing("string");
            // Get following text node
            xr.Read();

            // we display the result in the TextBlock
            helloWorld.Text = xr.Value;

            // Close th XmlReader and the HTTP Request
            xr.Close();
            mRequest.Close();
        }
        catch (Exception ex)
        {
            // if error, we display the exception's message in the textblock
            helloWorld.Text = ex.Message;
        }
    }
}
```

Le code est commenté pour une meilleure compréhension.

Là on se dit, c'est bon ça fonctionne. Mais essayez donc.
Clic droit sur *TestPage.html* puis **View in Browser**.

Normalement vous vous retrouvez avec ce message dans le TextBlock
Cross domain calls are not supported by BrowserHttpWebRequest

Et oui, votre projet Silverlight et votre Web Service ne sont pas exécutés sur le même serveur, enfin si mais pas sur le même port.

Pour résoudre ça, il faut suivre cette méthode.

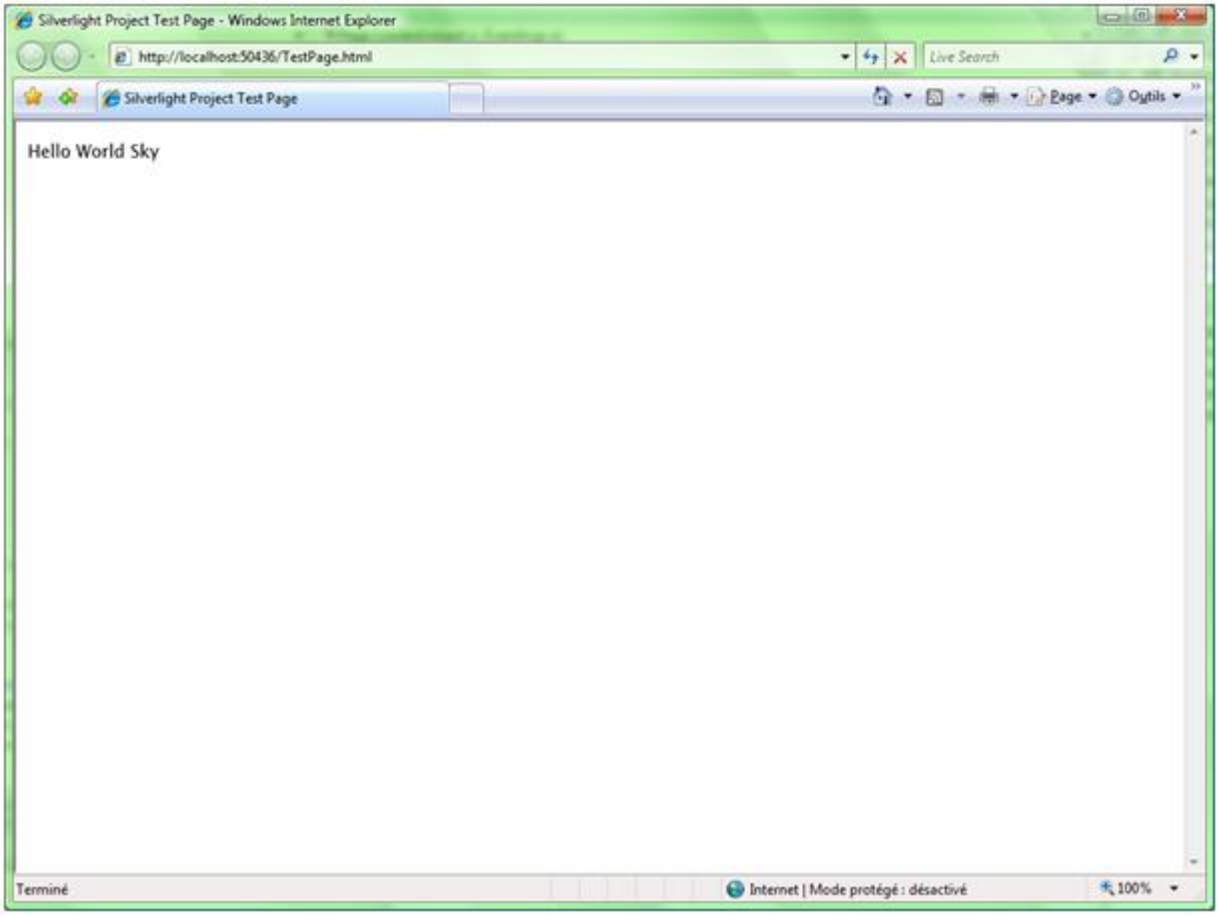
Clic droit sur votre Web Service puis clic sur **Add Silverlight Link#**, dans la fenêtre qui s'affiche, sélectionnez votre projet Silverlight puis OK.

Ceci rajoute une référence dans votre projet vers le projet Silverlight, il copie aussi le fichier *Page.xaml* (le code behind se trouvant dans la dll ajoutée en référence).

Une fois ceci fait, il vous faut copier *TestPage.html*, *TestPage.html.js* ainsi que *Silverlight.js* dans votre projet Web Service. Vous pouvez les supprimer de votre projet Silverlight après ça.

Maintenant réessayez (clic droit sur *TestPage.html* dans votre projet Web Service, puis **View in Browser**).

Et voilà.



Ça fonctionne !

Passons maintenant à la partie 2, qui va vous montrer comment utiliser un Web Service pleinement et facilement avec Silverlight.

4-2 - Silverlight et les Web Services

Pour cette partie, recréez une solution vide dans VS 2008, ainsi qu'un Web Service dans cette solution.

À la différence de l'autre partie, il faut dé commenter une ligne, afin de permettre l'appel de notre Web Service par notre application Silverlight.

```
// To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following line.
//[System.Web.Script.Services.ScriptService]
```

Devient donc tout simplement :

```
// To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following line.
[System.Web.Script.Services.ScriptService]
```

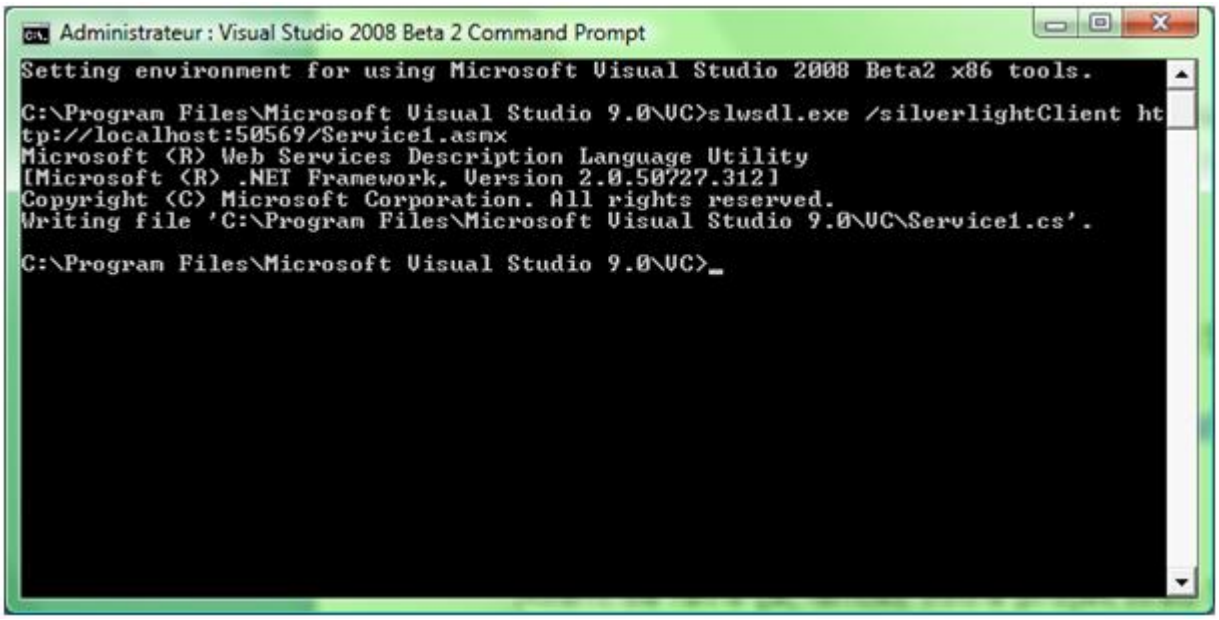
Ensuite créez votre projet Silverlight, rajoutez également un *TextBlock* dans le Canvas de votre *Page.xaml*.

```
<TextBlock x:Name="helloWorld" Name="helloWorld"></TextBlock>
```

Maintenant nous allons créer un proxy pour accéder à notre Web Service via notre application Silverlight, et ceux d'une façon simple.

Pour cela, 2 méthodes s'offrent à vous :

La première à l'aide d'un outil en ligne de commande
Pour ce faire ouvrez une console Visual Studio et tapez :
slwsdl.exe /silverlightClient http://localhost:50569/Service1.asmx



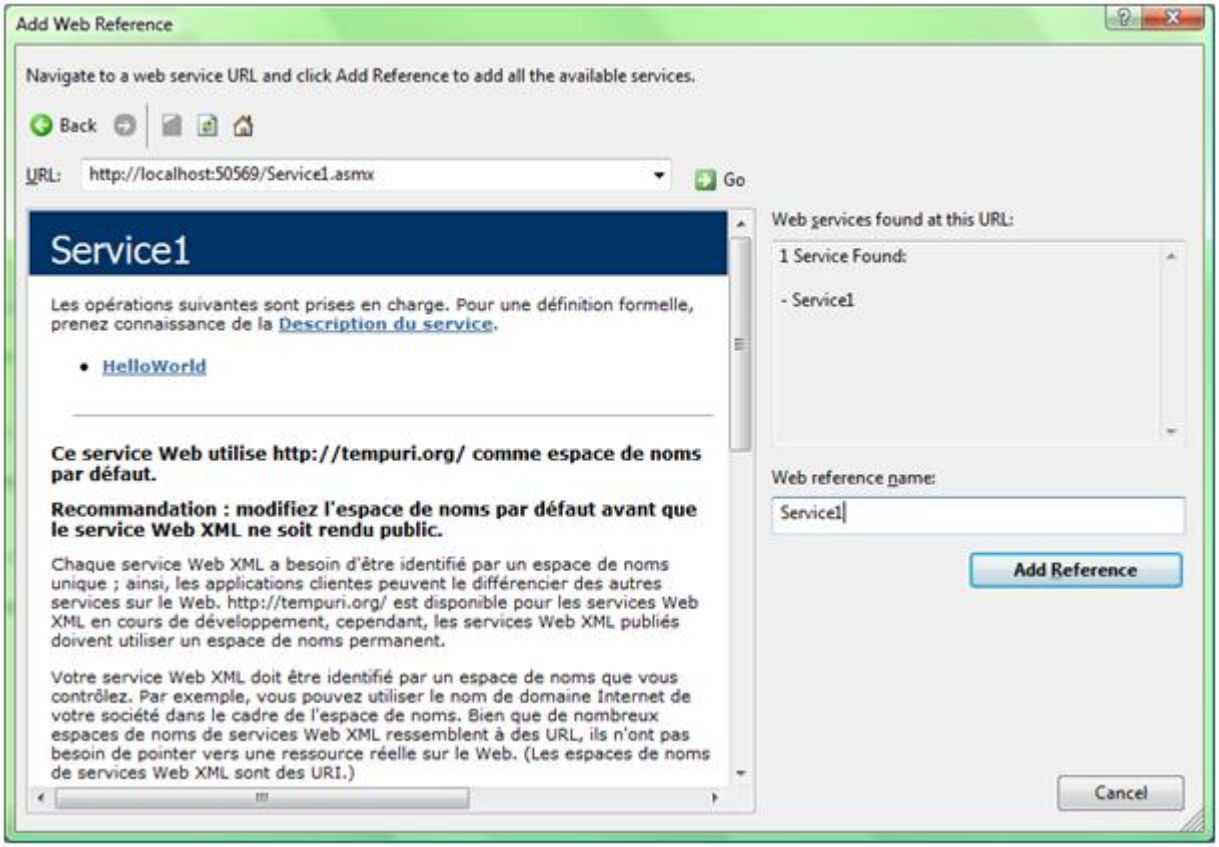
slwsdl.exe

Avant de faire ça, lancez votre projet Web Service pour qu'un serveur de développement soit lancé, remplacez ensuite le port par le vôtre.

Cette commande crée un fichier : *Service1.cs* dans le répertoire à partir duquel vous avez lancé la commande.

Allez le récupérer et ajoutez le à votre projet Silverlight.

La seconde méthode est d'utiliser Visual Studio.
Faites clic droit sur votre projet Silverlight puis **Add Web Reference**.
Dans URL vous tapez :
http://localhost:50569/Service1.asmx
Puis vous cliquez sur **Add Reference**



Add Reference

C'est tout.

Une fois l'une des 2 méthodes effectuées, vous pouvez utiliser votre Web Service comme une simple classe dans votre projet Silverlight.

Ouvrez le fichier *Page.xaml.cs* et comme tout à l'heure remplacez ceci :

```
public partial class Page : Canvas
{
    public void Page_Loaded(object o, EventArgs e)
    {
        // Required to initialize variables
        InitializeComponent();
    }
}
```

Par ça

```
public partial class Page : Canvas
{
    Service1.Service1 mService;

    public void Page_Loaded(object o, EventArgs e)
    {
        // Required to initialize variables
        InitializeComponent();

        // we instanciate our proxy
        mService = new Service1.Service1();

        // we call our method asynchronously
        // when the result will be available the event OnIHaveMyResponse will be called
        IAsyncResult iar = mService.BeginHelloWorld(new AsyncCallback(OnIHaveMyResponse), mService);
```

```
    }

    public void OnIHaveMyResponse(IAsyncResult iar)
    {
        // the method is finish we get our result (which is a string)
        string res = ((Service1.Service1)iar.AsyncState).EndHelloWorld(iar);

        // we display it on our TextBlock
        helloWorld.Text = res;
    }
}
```

Comme tout à l'heure, il faut ajouter un lien vers notre projet Silverlight dans notre projet Web Service.

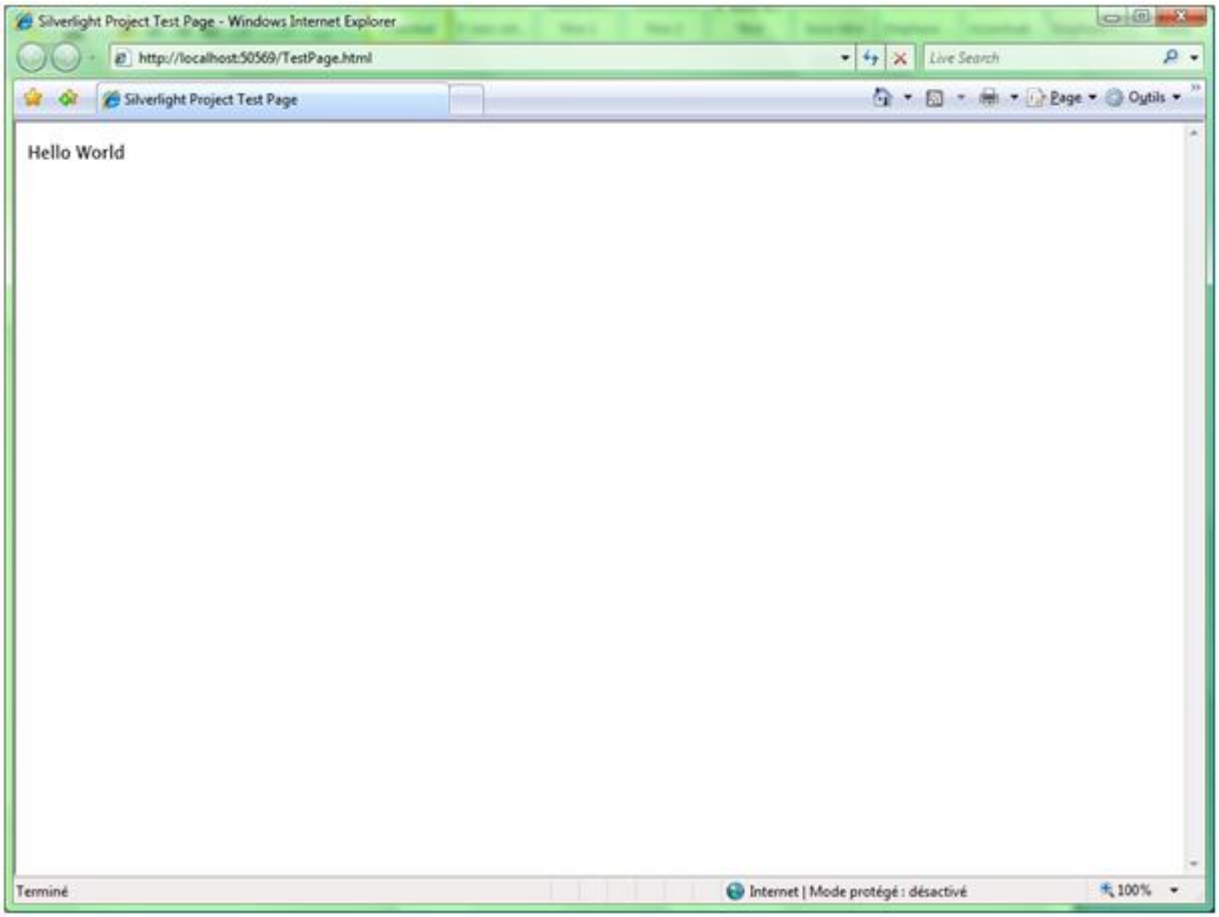
Clic droit sur notre Web Service puis **Add Silverlight Link#**, on choisit notre projet Silverlight puis OK.

On copie ensuite les fichiers *TestPage.html*, *TestPage.html.js* et *Silverlight.js* vers notre projet Web Service (on peut les supprimer ensuite de notre projet Silverlight).

C'est bon tout devrait fonctionner.

On essaye : Clic droit sur *TestPage.html* puis **View in Brower**.

Et voilà.



Ça fonctionne !

5 - Conclusion

Comme nous venons de le voir, Silverlight dans sa version 1.1 permet un grand nombre de choses, du simple au difficile, de l'inutile au puissant. Un très grand nombre de choix s'offrent ainsi au développeur en termes de développement Silverlight.

Dans un prochain article nous verrons comment manipuler du XML, comment utiliser un ***Isolated Storage*** et tout ça en Silverlight.

Suite à l'écriture de cet article, j'ai décidé d'en écrire un deuxième pour illustrer ce que nous avons vu.

Réalisation d'un chat en Silverlight 1.1

6 - Remerciements

Je tiens à remercier Aspic pour sa relecture et ses corrections.