

Appeler un service WCF depuis Silverlight

par Ludovic Lefort ([Site web](#)) ([Blog](#))

Date de publication :

Dernière mise à jour :

J'ai lu dans plusieurs forums que pas mal de personnes éprouvaient quelques difficultés pour appeler un service WCF depuis une application Silverlight. C'est la raison pour laquelle j'ai décidé d'écrire cet article.

0 - Introduction.....	3
1 - Pré-requis.....	4
2 - Première étape : création d'un site web.....	5
3 - Le projet Silverlight.....	6
4 - Le projet WCF.....	9
5 - Ajouter le service WCF au projet Silverlight.....	11
6 - Appeler un webservice hébergé sur un autre site.....	14
7 - Liens.....	15

0 - Introduction

Dans sa version actuelle (2 beta 1) silverlight ne permet de référencer un autre projet (class library).

La seule manière de le faire est de passer par une couche de Web service.

Cela peut se faire par des service web asp.net 2.0 ou grâce à Windows Communication Foundation (WCF) ajoutés dans le framework .net 3.0

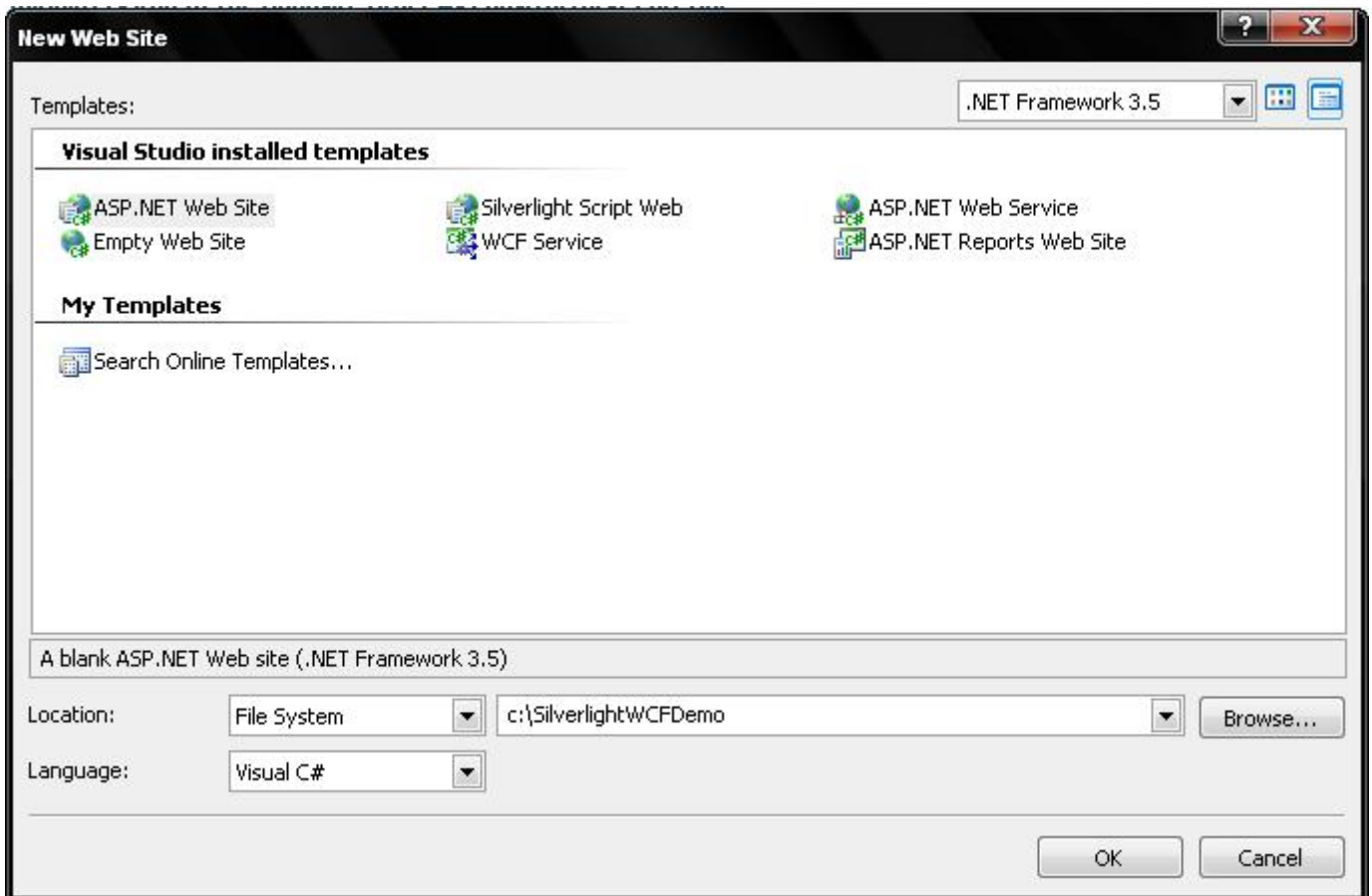
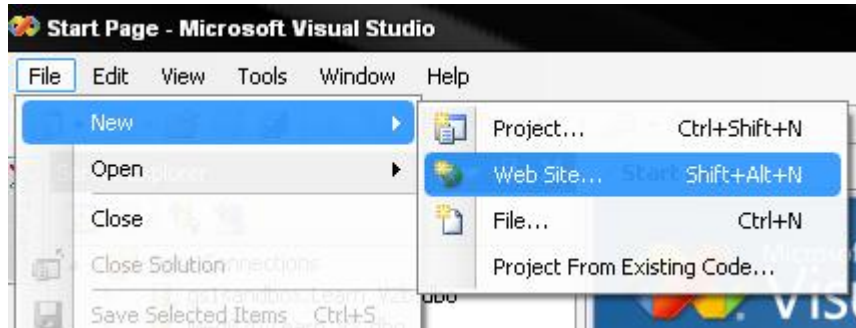
Microsoft recommande l'utilisation de WCF. J'ai donc choisi cette technologie pour cet article.

1 - Pré-requis

- Microsoft Visual Studio 2008 (**Virtual pc**)
- **Microsoft Silverlight Tools Beta 1 for Visual Studio 2008**

2 - Première étape : création d'un site web

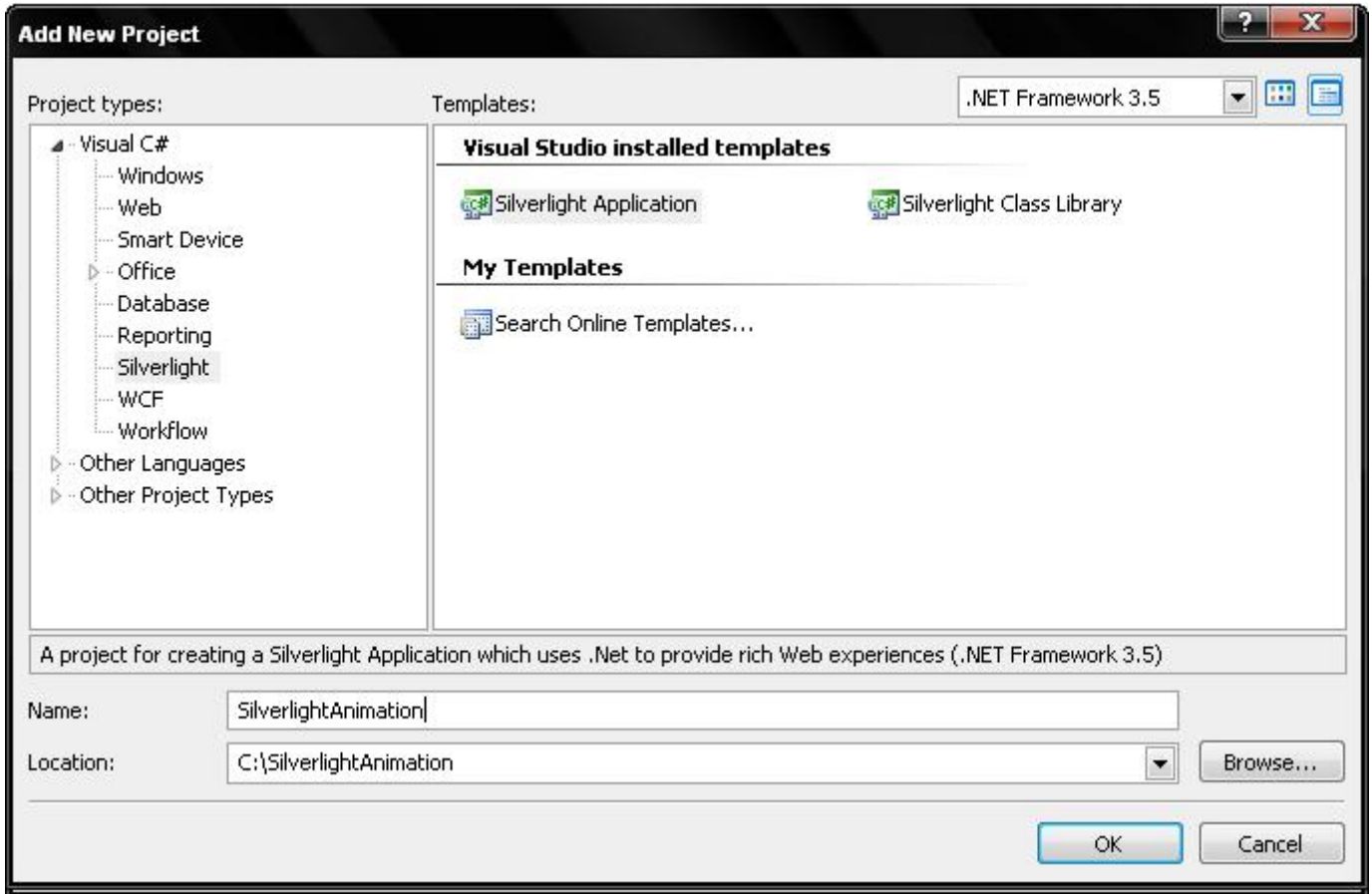
Pour commencer nous allons créer un nouveau site web asp.net avec Visual Studio 2008, appelons le SilverlightWCFDemo :



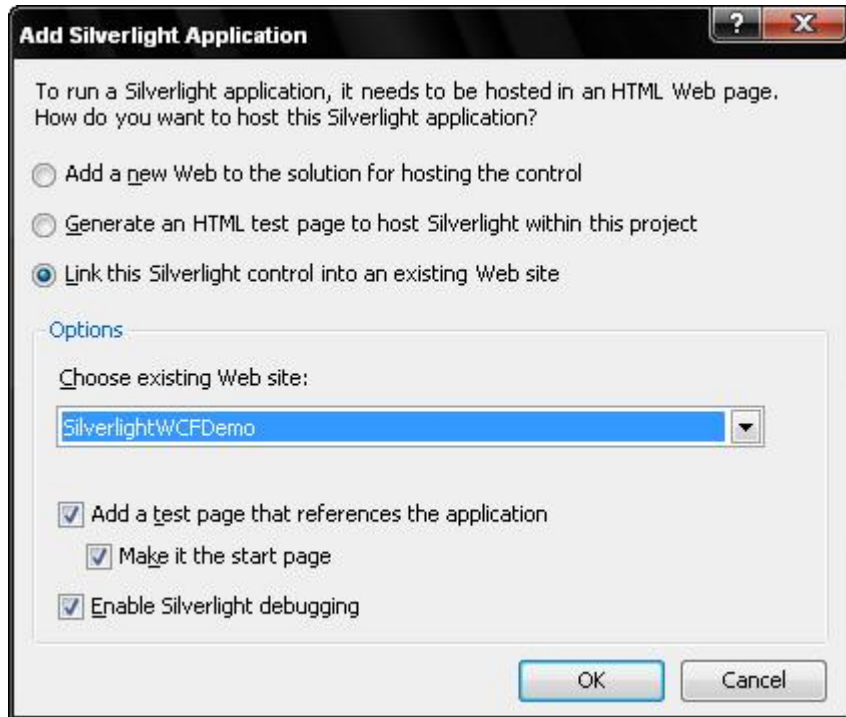
3 - Le projet Silverlight

Il nous faut à présent ajouter un nouveau projet à notre solution qui contiendra l'animation silverlight.
Clic droit sur la solution -> add -> new project.

Dans la rubrique Silverlight, choisissez Silverlight Application et nommée la SilverlightAnimation :



Après avoir cliqué sur OK Visual Studio va vous demander de lier ce projet à un site web, cliquez sur OK :



Normalement visual studio a du vous ajouter une page nommée **SilverlightAnimationTestPage.aspx** contenant votre animation silverlight et qui sera executée au démarrage de votre site web.

Pour tous ceux qui ont eu la joie de tester la version 1 de Silverlight : quel progrès :)

Modifions à présent le fichier Page.xml (animation silverlight) qui actuellement vierge.

Pour le design de mon animation j'utilise Microsoft Expresion Blend mais vous pouvez également travailler directement en xaml depuis Visual Studio.

Voici le contenu de notre animation pour cet exemple :

```
<UserControl x:Class="SilverlightAnimation.Page"
  xmlns="http://schemas.microsoft.com/client/2007"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  Width="400" Height="300">
<UserControl.Resources>
<Storyboard x:Name="AnimResult">

  <DoubleAnimationUsingKeyFrames Storyboard.TargetName="textBlock" Storyboard.TargetProperty="(TextBlock.FontSize)
  <SplineDoubleKeyFrame KeyTime="00:00:02" Value="72"/>
  </DoubleAnimationUsingKeyFrames>

  <DoubleAnimationUsingKeyFrames Storyboard.TargetName="textBlock" Storyboard.TargetProperty="(UIElement.RenderTr
  (TransformGroup.Children)[0].(ScaleTransform.ScaleX)" BeginTime="00:00:00">
  <SplineDoubleKeyFrame KeyTime="00:00:02" Value="9.107"/>
  </DoubleAnimationUsingKeyFrames>

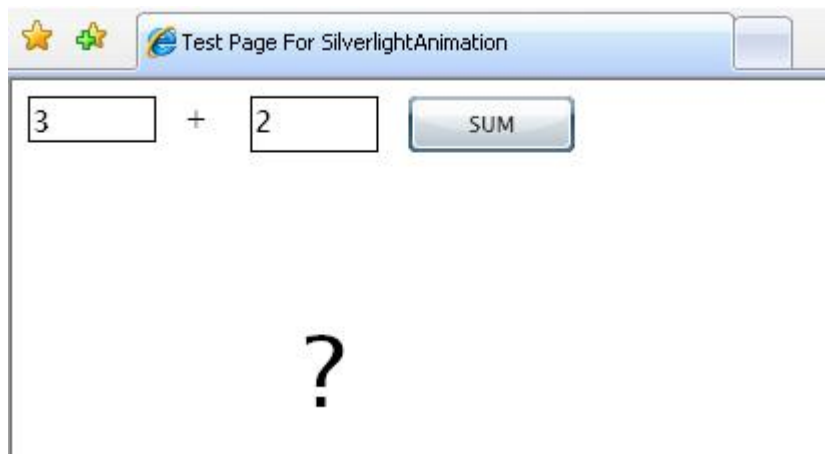
  <DoubleAnimationUsingKeyFrames Storyboard.TargetName="textBlock" Storyboard.TargetProperty="(UIElement.RenderTr
  (TransformGroup.Children)[0].(ScaleTransform.ScaleY)" BeginTime="00:00:00">
  <SplineDoubleKeyFrame KeyTime="00:00:02" Value="2.242"/>
  </DoubleAnimationUsingKeyFrames>

  <DoubleAnimationUsingKeyFrames Storyboard.TargetName="textBlock" Storyboard.TargetProperty="(UIElement.RenderTr
  (TransformGroup.Children)[3].(TranslateTransform.X)" BeginTime="00:00:00">
  <SplineDoubleKeyFrame KeyTime="00:00:02" Value="54.789"/>
  </DoubleAnimationUsingKeyFrames>

  <DoubleAnimationUsingKeyFrames Storyboard.TargetName="textBlock" Storyboard.TargetProperty="(UIElement.RenderTr
  (TransformGroup.Children)[3].(TranslateTransform.Y)" BeginTime="00:00:00">
  <SplineDoubleKeyFrame KeyTime="00:00:02" Value="39.732"/>
  </DoubleAnimationUsingKeyFrames>
</Storyboard>
</UserControl.Resources>
```

```
<Grid x:Name="LayoutRoot" Background="White">
<TextBox x:Name="txtX" Height="23" Margin="8,8,0,0" VerticalAlignment="Top" Text="2" Width="64" HorizontalAlignm
<TextBlock Height="32" HorizontalAlignment="Left" Margin="86,8,0,0" VerticalAlignment="Top" Width="20" Text="+
<TextBox x:Name="txtY" Height="28" Margin="119,8,0,0" VerticalAlignment="Top" Text="3" Width="64" HorizontalAli
<TextBlock HorizontalAlignment="Left" Margin="146,110,0,128" VerticalAlignment="Stretch" Width="28" Text="?" Te
    <TextBlock.RenderTransform>
        <TransformGroup>
            <ScaleTransform/>
            <SkewTransform/>
            <RotateTransform/>
            <TranslateTransform/>
        </TransformGroup>
    </TextBlock.RenderTransform>
</TextBlock>
<Button x:Name="btnSubmit" Height="28" Margin="198,8,119,0" VerticalAlignment="Top" Content="SUM"/>
</Grid>
</UserControl>
```

Vous devriez obtenir ceci :



Vous l'aurez sûrement déjà compris un clic sur le bouton "Sum" devra nous donner le résultat de la somme des deux TextBox et déclencher le storyboard "AnimResult".

Ajoutons donc le code nécessaire à l'exécution du storyboard. Ce se fait dans le fichier page.xml.cs :

```
public partial class Page : UserControl
{
    public Page()
    {
        InitializeComponent();
        this.btnSubmit.MouseLeftButtonUp += new
        MouseButtonEventHandler(btnSubmit_MouseLeftButtonUp);
    }

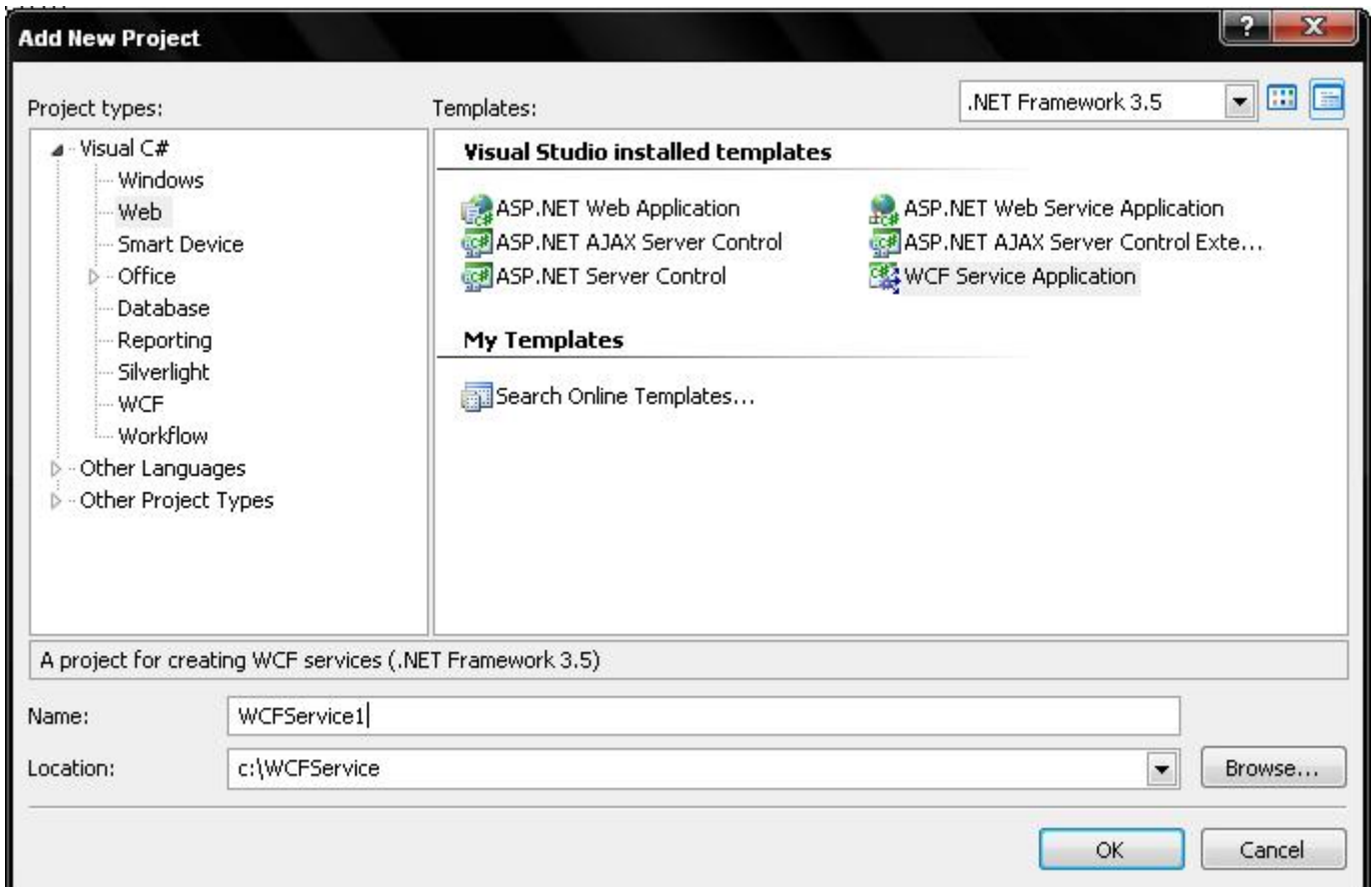
    void btnSubmit_MouseLeftButtonUp(object sender, MouseButtonEventArgs e)
    {
        this.AnimResult.Begin();
    }
}
```

Il nous reste à présent à faire la somme des deux textbox pour cela nous allons utiliser WCF.

4 - Le projet WCF

Comme pour le projet silverlight : clic droit sur la solution --> add --> new projet.

Dans la partie Web choisissez WCF Service Application et nommé le WCFService1 :



Parmi les fichiers créés automatiquement par le template, deux doivent attiré notre attention : **IService1.cs** et **Service1.cs**.

IService1.cs est une interface qui ne contiendra donc que les signatures des méthodes alors que **Service1.cs** permettra son implémentation.

Ces fichiers contiennent des exemples de codes que nous pouvons supprimer et remplacer par notre méthode :
IService1.cs :

```
namespace WCFService1
{
    [ServiceContract]
    public interface IService1
    {
        [OperationContract]
        int getSum(int x, int y);
    }
}
```

Service1.cs :

```
namespace WCFService1
{
```

```
public class Service1 : IService1
{
    public int getSum(int x,int y)
    {
        return x + y;
    }
}
```

Jettons maintenant un oeil sur le fichier web.config de notre service, plus précisément sur la section `system.serviceModel` :

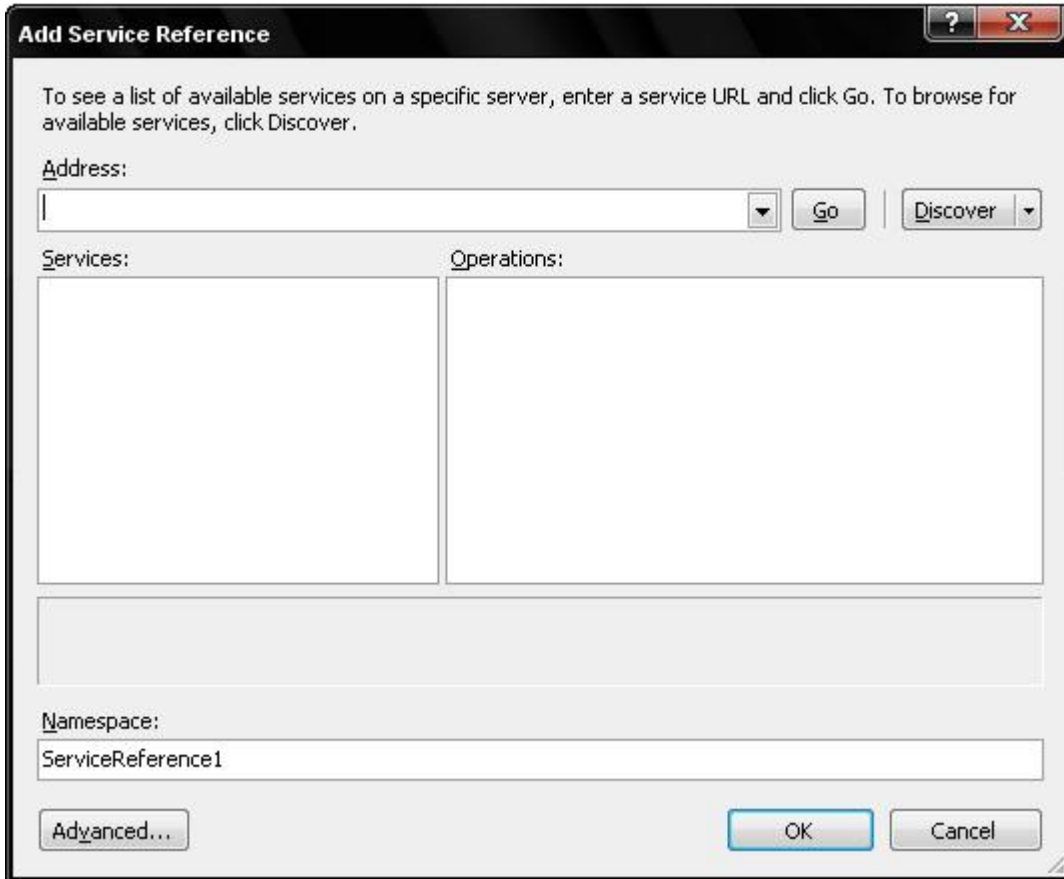
```
<system.serviceModel>
  <services>
    <service name="WCFService1.Service1" behaviorConfiguration="WCFService1.Service1Behavior">
      <!-- Service Endpoints -->
      <endpoint address="" binding="basicHttpBinding" contract="WCFService1.IService1">
        <!--
          Upon deployment, the following identity element should be removed or replaced to reflect the
          identity under which the deployed service runs.  If removed, WCF will infer an appropriate identity
          automatically.
        -->
        <identity>
          <dns value="localhost"/>
        </identity>
      </endpoint>
      <endpoint address="mex" binding="mexHttpBinding" contract="IMetadataExchange"/>
    </service>
  </services>
  <behaviors>
    <serviceBehaviors>
      <behavior name="WCFService1.Service1Behavior">
        <!-- To avoid disclosing metadata information, set the value below to false and remove the metadata endpoint ab
        >
        <serviceMetadata httpGetEnabled="true"/>
        <!-- To receive exception details in faults for debugging purposes, set the value below to true.  Set to false
        >
        <serviceDebug includeExceptionDetailInFaults="false"/>
      </behavior>
    </serviceBehaviors>
  </behaviors>
</system.serviceModel>
```

Silverlight n'est compatible avec le service qu'en **basicHttpBinding**. Cette ligne est donc très importante :

```
<endpoint address="" binding="basicHttpBinding" contract="WCFService1.IService1">
```

5 - Ajouter le service WCF au projet Silverlight

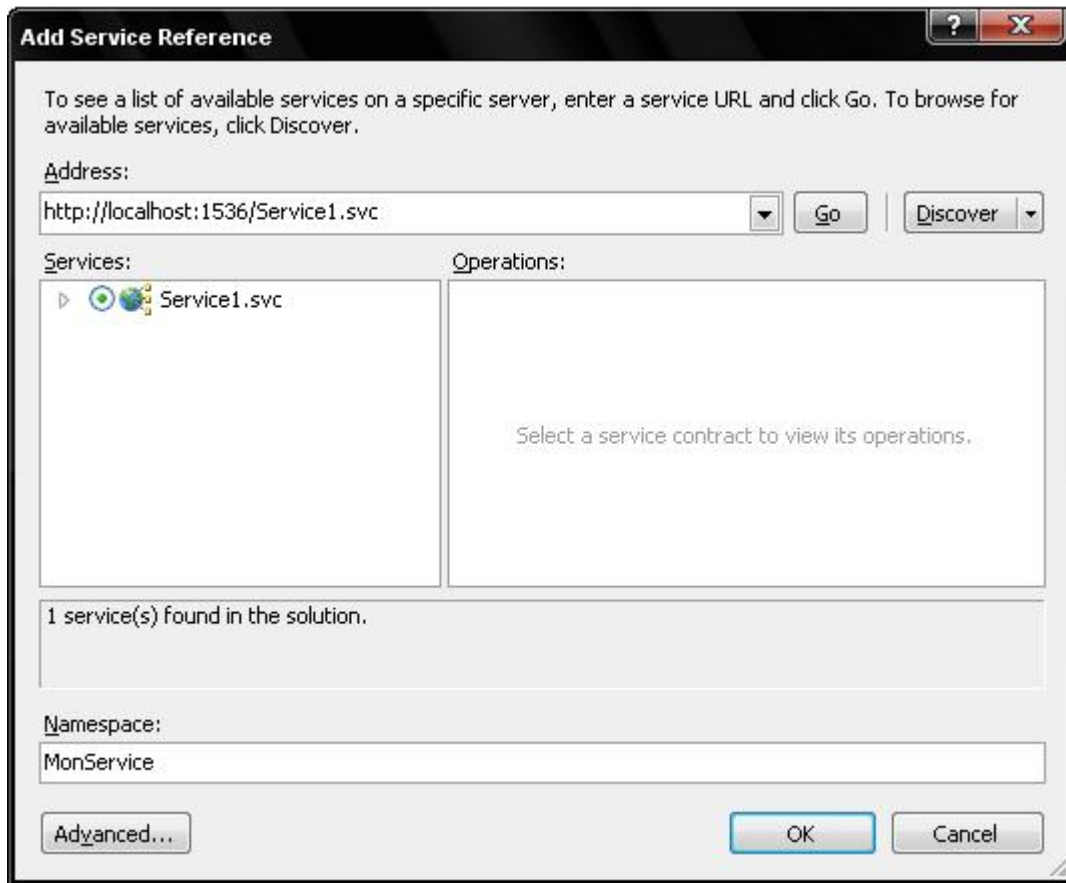
Nous avons créé une animation Silverlight et un service WCF, reste maintenant à lier les deux.
Pour cela : clic droit sur le projet silverlight --> add Service Reference.
Vous obtenez alors une fenêtre permettant de sélectionner un service :



Ouvrez ensuite le menu discover et choisissez "Service in solution" par ajouter un service présent dans la solution :



Choisissez Service1 dans la liste et appelez le **MonService** :



Reste ensuite à utiliser le proxy que nous venons de créer. Modifions donc notre code dans le fichier page.xaml.cs :

```
namespace SilverlightAnimation
{
    public partial class Page : UserControl
    {
        public Page()
        {
            InitializeComponent();
            this.btnSubmit.MouseLeftButtonUp += new
            MouseButtonEventHandler(btnSubmit_MouseLeftButtonUp);
        }

        void btnSubmit_MouseLeftButtonUp(object sender, MouseButtonEventArgs e)
        {
            MonService.Service1Client client = new SilverlightAnimation.MonService.Service1Client();
            client.getSumCompleted += new
            EventHandler<SilverlightAnimation.MonService.getSumCompletedEventArgs>(client_getSumCompleted);
            client.getSumAsync(Convert.ToInt32(txtX.Text), Convert.ToInt32(txtY.Text));
        }

        void client_getSumCompleted(object sender,
            SilverlightAnimation.MonService.getSumCompletedEventArgs e)
        {
            int result = e.Result;
            this.textBlock.Text = result.ToString();
            this.AnimResult.Begin();
        }
    }
}
```

En Silverlight il n'est possible d'appeler un service que de manière asynchrone c'est donc la raison pour laquelle nous commençons par attacher une méthode à l'événement **getSumCompleted** de notre client.

Le résultat de la méthode est récupérée dans le paramètre **e** passé à la méthode **client_getSumCompleted**.

6 - Appeler un webservice hébergé sur un autre site

Par défaut Silverlight refuse d'utiliser un webservice hébergé sur un autre site que l'application pour contourner ce problème et autoriser le "Cross Domain" vous devez ajouter le fichier **crossdomain.xml** à la racine de votre serveur web.

Voici son contenu :

```
<access-policy>
  <cross-domain-access>
    <policy>
      <allow-from>
        <domain uri="*" />
      </allow-from>
      <grant-to>
        <resource path="/" include-subpaths="true" />
      </grant-to>
    </policy>
  </cross-domain-access>
</access-policy>
```

La section **grant-to** permet de définir les répertoires pouvant être appelés par l'application.

La section **allow-from** quand à elle défini les domaines pouvant utiliser le web service.

7 - Liens

Mes articles : <http://lefortludovic.developpez.com>

Mon blog : <http://blogs.ezos.com/blog/le>