

1 Création de services WCF avec C

1.1 Création de Service WCF de base en C #. Net

WCF (Windows Communication Foundation) est une partie de. NET. Nous utiliserons Visual Studio 2012 ou une version postérieur pour ce TD.

Le projet à créer est un New Project -> WCF -> WCFClassLibrary projet comme indiqué ci-dessous

New Project			? ×
Project types:	Templates:	.NET Framework 3.5	▼ 000 tit:
Visual Basic Database Projects Other Languages Visual C# Windows Web Smart Device Office Database Reporting SSIS_ScriptComponent SSIS_ScriptTask Test WCF Workflow	Visual Studio installed templates Visual Studio installed templates Sequential State Machine Syndication Workflow S Service Library My Templates Search Online Templates	Service rary	
A project for creating a WCF service on Name: MathsLibrary	ass library (.dl) (.NET Framework 3.5)		
		ОК	Cancel

Donner le nom MathsLibrary à votre projet.

Le projet contient un échantillon des fichiers Service1.cs & IService.cs.

Nous allons les effacer pour ajouter notre propre service.

Faites un clic droit sur MathsLibrary -> Ajouter -> New Item -> Select Class1.cs.

Renommez-le MathsOperations.cs

Cela va créer un fichier de classe simple. Ouvrez le fichier et rendez la classe publique.

De la même façon ajouter une classe IMathsOperations.cs qui sera une interface, qui fournira une liste de toutes les opérations que le service WCF propose.

Ouvrez IMathsOperations.cs et changer pour public IMathsOperations

Ajoutez le code ci-dessous.



Applications Réparties

Services WCF (Windows Communication Foundation)

```
namespace MathsLibrary
{
    public interface IMathsOperations
    {
        int Add(int num1, int num2);
        int Multiply(int num1, int num2);
    }
}
```

Pour que IMathsOperations devienne un contrat de service WCF, ajouter un attribut [ServiceContract].

Ainsi toute opération que vous souhaitez rendre visible pour le client doit être décoré avec l'attribut [OperationContract].

[ServiceContract] et [OperationContract] sont inclus dans l'espace de nom System.ServiceModel

```
] using System;
 using System.Collections.Generic;
 using System.Ling;
 using System.Text;
 using System.ServiceModel;
] namespace MathsLibrary
 {
     [ServiceContract]
     public interface IMathsOperations
Ξ
     ŧ
          [OperationContract]
         int Add(int num1, int num2);
         [OperationContract]
         int Multiply(int num1, int num2);
     }
- }
```

Maintenant, une fois le contrat fixé, nous pouvons implémenter cette interface dans notre service comme cidessous



Applications Réparties

Services WCF (Windows Communication Foundation)

```
using System;
using System.Collections.Generic;
using System.Ling;
using System.Text;
namespace MathsLibrary
    public class MathsOperations : IMathsOperations
    ł
        public int Add(int num1, int num2)
        £.
            return num1 + num2;
        ъ
        public int Multiply(int num1, int num2)
        £
            return num1 * num2;
        )
    )
```

Générez le projet une fois que vous avez terminé à cela.

Ajoutez ou modifier App.config dans la solution. App.config contient des détails sur les endpoints, ce qui comprend les paramètres ABC (address et binding principalement).

- Address est l'adresse Où le service peut être trouvé.
- Binding est le Comment accéder aux services
- Contrat est ce que (Quoi) contient le service contient.

Maintenant, nous allons modifier App.config. Faites un clic droit sur App.config, cliquez sur Modifier la configuration WCF



4

Applications Réparties Services WCF (Windows Communication Foundation)



Le popup ci-dessous doit apparaître

<u>Eile H</u> elp	
Configuration	Services
Services B-X MathsLibrary.Service1 B-C Client Bindinas	Create a New Service
⊕- ☐ Diagnostics ⊕- ☐ Advanced	Service: MathsLibrary.Service1 Create a New Service Endpoint
	Endpoint: (Empty Name) Delete Address: Binding: wsHttpBinding Binding configuration: IDefault Click to Create Contract: MathsLibrary.IService1
Tasks 🛞 Create a New Service Create a New Client	Endpoint: (<u>Empty Name)</u> <u>Delete</u> Address: mex Binding: mexHtpBinding Binding configuration: <u>(Default) Click to Create</u> Contract: IMetadataExchange

Sélectionnez le Servicei du panneau de gauche, la fenêtre suivante doit apparaître

Ele ttep Configuration	Service: MathsLibrary.	Service1
Services MathsLibrary.Service1 Gient Gient Gient Gindpoints Gindpoints Diagnostics Advanced	E [General] BehaviorConfiguration Name	MathsLibrary.Service1Behavior MathsLibrary.Service1

Contributeurs par ordre alphabétique : Jean-Yves Tigli

TD inspiré de "WCF Service Creation With C#" par TheDhruv



Cliquez sur le bouton "points de suspension" et trouvez MathsLibrary.dll.

Cliquez sur ce lien, il vous donnera le nom du service qu'elle contient.



De même aller sur des endpoints, et sélectionnez un contrat approprié avec les mêmes étapes que ci-dessus.

E- 🔁 Services	General Identity Headers	
	(Configuration) Name	
Endpoints (Empty Name)	Endpoint Properties	
(Empty Name)	Address	
E- 🛄 Client	BehaviorConfiguration	
- 🛅 Bindings	Binding	wsHttpBinding
Diagnostics	BindingConfiguration	
Advanced	BindingName	
	BindingNamespace	
	Contract	MathsLibrary.IService1
	ListenUri	
	ListenUniMode	Explicit

Nous pouvons aussi changer le binding comme ci-dessous ...

Configuration	Service Endpoint	
Services Athena ibrary Mathe@nerations	General Identity Headers	
Host	(Configuration) Name Endopsion Properties	
Empty Name)	Address BehaviorConfiguration	
Client Gindings Jagnostics	BindingConfiguration BindingName	DasicHttpBinding
🗈 🧰 Advanced	BindingNamespace Contract	MathsLibrary.IMathsOperations
	ListenUn ListenUnMode	Explicit

Sélectionnez "Host" sur le côté gauche et vous verrez l'adresse de base. Vous pouvez la modifier vers l'adresse que vous voulez.



Configuration	Host	
	Timeouts settings	
MathsLibrary.MathsOperations	🗆 (General)	
Host	CloseTimeout 00:00:10	
E-Endpoints	OpenTimeout 00:01:00	
(Empty Name)		
(Empty Name)	CloseTimeout	
	The time interval defines the maximum time a service is a	llowed to take to gracefully
- Bindings	ebut down. After the time limit runs out, the service is form	H to shut down.
Diagnostics Diagnostics		
Advanced Base address:	http://localhost:8700/MathsService/	
	OK Cancel	

Une fois que cela est fait, générez le projet.

Visual Studio founit son propre preocessus d'accueil d'hébergement de services. Nous pouvons également héberger le service comme une application console ou un Service Windows, ou encore dans IIS.

2 Création d'un client WCF avec C

Maintenant, nous allons créer une application cliente. Ajouter un projet "application console" (ou tout autre type de projet) dans la même solution ou une solution différente.

Ajouter une référence au service MathsService que nous avons créé. Il chargera automatiquement toutes les DLL nécessaires.

Add Service Reference			<u>? ×</u>
To see a list of available services on a spe available services, click Discover.	cific server, enter a service	URL and click Go. To brov	vse for
Address:			
http://localhost:8701/MathsService/		<u> </u>	over 🔻
Services:	Operations:		
IMathsOperations IMathsOperations	=∲Add =∲Multiply		
1 service(s) found at address 'http://localhost:8701/MathsService/'. Namespace: MathsService			
,			
Ad <u>v</u> anced		ОК Са	ancel

Puis, avec le code ci-dessous, nous pouvons appeler le service :



```
public class Program
{
    public static void Main(string[] args)
    {
        MathsOperationsClient client = new MathsOperationsClient();
        Console.WriteLine(client.Add(100,101));
        Console.WriteLine(client.Multiply(100, 101));
        Console.ReadLine();
    }
}
```

Cela vous donnera dessous de la sortie



4 Exploration de différents Bindings

Sans modifier vos codes sources, modifiez et testez de nouveaux bindings entre le client et le serveur.

Créer pour cela plusieurs endpoints avec des bindings différents et testez-les.

Attention dans ce cas de passer le nom du endpoint que vous voulez adresser lors d'une instanciation de la classe proxy sur le service.

Exemple MathsOperationsClient client = new MathsOperationsClient("BasicHTTP");

Dans le cas :

</endpoint>

Contributeurs par ordre alphabétique : Jean-Yves Tigli

TD inspiré de "WCF Service Creation With C#" par TheDhruv