Introduction to WCF - WCF tutorial | WCF Tutorial - Windows Communication Foundation | WCF Example | WCF Sample code in asp.net 3.5 | Basic WCF Tutorial for Beginners

Introduction:

Here I will explain what WCF (windows communication foundation) is, uses of windows communication foundation and how to create and use windows communication foundation in c#.

Description:

In previous articles explained clearly what webservice is and how to create and consume webservice using asp.net . In another post I explained clearly what windows service is and how to create windows service and sample of windows service using c#. Now in this article I will explain about windows communication foundation. First we will see what a WCF (window communication foundation) is and uses of WCF (windows communication foundation) after that we will see how to create and use WCF in c#.net.

What is WCF (windows communication foundation) Service?

Windows Communication Foundation (Code named Indigo) is a programming platform and runtime system for building, configuring and deploying network-distributed services. It is the latest service oriented technology; Interoperability is the fundamental characteristics of WCF. It is unified programming model provided in .Net Framework 3.0. WCF is a combined feature of Web Service, Remoting, MSMQ and COM+. WCF provides a common platform for all .NET communication.

Advantages of WCF

1) WCF is interoperable with other services when compared to .Net Remoting where the client and service have to be .Net.

2) WCF services provide better reliability and security in compared to ASMX web services.

3) In WCF, there is no need to make much change in code for implementing the security model and changing the binding. Small changes in the configuration will make your requirements.

4) WCF has integrated logging mechanism, changing the configuration file settings will provide this functionality. In other technology developer has to write the code.

Difference between WCF and Web service

Web service is a part of WCF. WCF offers much more flexibility and portability to develop a service when comparing to web service. Still we are having more advantages over Web service; following table provides detailed difference between them.

Features	Web Service	WCF
Hosting	It can be hosted in IIS	It can be hosted in IIS, windows activation service, Self-hosting, Windows service
Programming	[WebService] attribute has to be added to the class	[ServiceContract] attribute has to be added to the class
Model	[WebMethod] attribute represents the method exposed to client	[OperationContract] attribute represents the method exposed to client
Operation	One-way, Request-Response are the different operations supported in web service	One-Way, Request-Response, Duplex are different type of operations supported in WCF
XML	System.Xml.serialization name space is used for serialization	System.Runtime.Serialization namespace is used for serialization
Encoding	XML 1.0, MTOM(Message Transmission Optimization Mechanism), DIME, Custom	XML 1.0, MTOM, Binary, Custom
Transports	Can be accessed through HTTP, TCP, Custom	Can be accessed through HTTP, TCP, Named pipes, MSMQ,P2P, Custom

Protocols	Security	Security, Reliable messaging, Transactions

A WCF Service is composed of three components parts viz,

1) **Service Class** - A WCF service class implements some service as a set of methods.

2) **Host Environment** - A Host environment can be a Console application or a Windows Service or a Windows Forms application or IIS as in case of the normal asmx web service in .NET.

3) **Endpoints** - All communications with the WCF service will happen via the endpoints. The endpoint is composed of 3 parts (collectively called as ABC's of endpoint) as defines below:

Address: The endpoints specify an Address that defines where the endpoint is hosted. It's basically url.

Ex:http://localhost/WCFServiceSample/Service.svc

Binding: The endpoints also define a binding that specifies how a client will communicate with the service and the address where the endpoint is hosted. Various components of the WCF are depicted in the figure below.

- "A" stands for Address: Where is the service?
- "B" stands for Binding: How can we talk to the service?
- "C" stands for Contract: What can the service do for us?

Different bindings supported by WCF

Binding	Description
BasicHttpBinding	Basic Web service communication. No security by default
WSHttpBinding	Web services with WS-* support. Supports transactions
WSDualHttpBinding	Web services with duplex contract and transaction support
WSFederationHttpBinding	Web services with federated security. Supports transactions

MsmqIntegrationBinding	Communication directly with MSMQ applications. Supports transactions
NetMsmqBinding	Communication between WCF applications by using queuing. Supports transactions
NetNamedPipeBinding	Communication between WCF applications on same computer. Supports duplex contracts and transactions
NetPeerTcpBinding	Communication between computers across peer-to-peer services. Supports duplex contracts
NetTcpBinding	Communication between WCF applications across computers. Supports duplex contracts and transactions
BasicHttpBinding	Basic Web service communication. No security by default
WSHttpBinding	Web services with WS-* support. Supports transactions

Contract: The endpoints specify a Contract that defines which methods of the Service class will be accessible via the endpoint; each endpoint may expose a different set of methods.

Different contracts in WCF

Service Contract

Service contracts describe the operation that service can provide. For Eg, a Service provide to know the temperature of the city based on the zip code, this service is called as Service contract. It will be created using Service and Operational Contract attribute.

Data Contract

Data contract describes the custom data type which is exposed to the client. This defines the data types, which are passed to and from service. Data types like int, string are identified by the client because it is already mention in XML schema definition language document, but custom created class or data types cannot be identified by the client e.g. Employee data type. By using DataContract we can make client to be aware of Employee data type that are returning or passing parameter to the method.

Message Contract

Default SOAP message format is provided by the WCF runtime for communication between Client and service. If it is not meeting your requirements then we can create our own message format. This can be achieved by using Message Contract attribute.

Fault Contract

Suppose the service I consumed is not working in the client application. I want to know the real cause of the problem. How I can know the error? For this we are having Fault Contract. Fault Contract provides documented view for error occurred in the service to client. This helps us to easy identity, what error has occurred.

Overall Endpoints will be mentioned in the web.config file for WCF service like this

```
<system.serviceModel>
<services>
<service name="Service" behaviorConfiguration="ServiceBehavior">
<!-- Service Endpoints -->
<endpoint address="http://localhost:8090/MyFirstWcfService/SampleService.svc"
binding="wsHttpBinding" contract="IService">
<identity>
<dns value="localhost"/>
</identity>
</endpoint>
<endpoint address="mex" binding="mexHttpBinding"
contract="IMetadataExchange"/>
</service>
</services>
<behaviors>
<serviceBehaviors>
<behavior name="ServiceBehavior">
<serviceMetadata httpGetEnabled="true"/>
<serviceDebug includeExceptionDetailInFaults="false"/>
```

```
</behavior>
</serviceBehaviors>
</behaviors>
</system.serviceModel>
```

<u>Creating simple application using WCF</u>

First open Visual Studio and click file --> Select New --> Website Under that select WCF Service and give name for WCF Service and click OK

Once you created application you will get default class files including **Service.cs** and **IService.cs**

Here **IService.cs** is an interface it does contain Service contracts and Data Contracts and **Service.cs** is a normal class inherited by **IService** where you can all the methods and other stuff.

Now open **IService.cs** write the following code

```
[ServiceContract]
public interface IService
{
[OperationContract]
string SampleMethod(string Name);
}
```

After that open **Service.cs** class file and write the following code

```
public class Service : IService
{
public string SampleMethod(string Name)
{
return "First WCF Sample Program " + Name;
}
}
```

Here we are using **basicHttpBinding** for that our web.config file system.serviceModel code should be like this and I hope no need to write any code because this code already exists in your web.config file in system.serviceModel

```
<system.serviceModel>
<services>
<service name="Service" behaviorConfiguration="ServiceBehavior">
<!-- Service Endpoints -->
<endpoint address="" binding="wsHttpBinding" contract="IService">
<identity>
<dns value="localhost"/>
</identity>
</endpoint>
<endpoint address="mex" binding="mexHttpBinding"
contract="IMetadataExchange"/>
</service>
</services>
<behaviors>
<serviceBehaviors>
<behavior name="ServiceBehavior">
<serviceMetadata httpGetEnabled="true"/>
<serviceDebug includeExceptionDetailInFaults="false"/>
</behavior>
```

```
</serviceBehaviors>
</behaviors>
</system.serviceModel>
```

Our **WCF** service ready to use with **basicHttpBinding.** Now we can call this WCF Service method console applications

After completion of WCF service creation publish or deploy your WCF Service in your system. If you don't' have idea on deploy check this post publish or deploy website

After completion of deploy webservice now we can see how to use WCF Service in our console application

Calling WCF Service using Console Application

To call WCF service we have many ways like using console app, windows app and web app but here I am going for console application.

Create new console app from visual studio select project type as console application gives some name as you like.

After Creation Console application now we need to add WCF reference to our console application for that right click on your windows application and select **Add Service Reference**

Now one wizard will open in that give your WCF service link and click Go after add your service click OK button.

After completion of adding **WCF** Service write the following code in **Program.cs** class file Main method

```
static void Main(string[] args)
{
ServiceReference1.ServiceClient objService = new ServiceClient();
Console.WriteLine("Please Enter your Name");
string Message = objService.SampleMethod(Console.ReadLine());
Console.WriteLine(Message);
Console.ReadLine();
}
```

After that open your **app.config** file and check your endpoint connection for WCF Service reference that should be like this

```
<endpoint address=" http://localhost/WCFServiceSample/Service.svc"
binding="wsHttpBinding" bindingConfiguration="WSHttpBinding_IService"
contract="ServiceReference1.IService" name="WSHttpBinding_IService">
<identity>
<dns value="localhost" />
</identity>
</endpoint>
```

Now everything is ready run your application that output should be like this