

Chapter 2: Developing WCF Service Application and Client

- Hosting WCF Service in IIS/ASP.NET Development Server
- Using a Service in the client application
- Understand Contracts in the service.
 - ServiceContract.
 - OperationContract.
 - DataContract.
- Understand Instancing behavior in the service.
 - Single
 - PerCall
 - PerSession
- Building WCF Library based Host and Client application.

Walkthrough 1: IIS / ASP.NET as Hosting Environment

In Server

1. File → New → Website → WCF Service, Location = File System, Path=d:\WCF\HostInIIS\
2. Delete existing Service.svc, IService.cs, Service.cs
3. Add → New Item → WCF Service → MathService
4. Add to the Project “Complex” class as below

```
[DataContract]
public class Complex
{
    private int _Real, _Imag;

    [DataMember]
    public int Real
    {
        get { return _Real; }
        set { _Real = value; }
    }

    [DataMember]
    public int Imag
    {
        get { return _Imag; }
        set { _Imag = value; }
    }
}
```

5. Modify the IMathService and MathService as below

```
[ServiceContract]
public interface IMathService
{
    [OperationContract]
    int Add(int a,int b);

    [OperationContract]
    Complex AddComplex(Complex c1,Complex c2);

    [OperationContract]
    int GetCounter();
}

public class MathService : IMathService
{
    public int Add(int a, int b)
    {
        Count++;
        return a+b;;
    }

    public Complex AddComplex(Complex c1, Complex c2)
    {
        Count++;
        Complex c = new Complex();
    }
}
```

```
        c.Real = c1.Real + c2.Real;
        c.Imag = c1.Imag + c2.Imag;
        return c;
    }
    int Count;
    public int GetCounter()
    {
        return Count;
    }
}
```

6. Run the MathService (Ctrl+F5). It Opens in the browser. Copy the URL.

In Client:

1. Start New Instance of VS.NET
2. Add Service Reference → Use the URL from above (Step 6)
3. Add Button and handle its Click event with the following code.

```
localhost.MathServiceClient ds = new localhost.MathServiceClient();
MessageBox.Show(ds.Add(100,200).ToString());
```

Demo of InstanceContextMode – Instancing behavior

In Server

1. In IEmpService interface add a method called as GetCount()
2. In EmpService declare a variable “Count” and in every method increment its value.
3. In EmpService Implement GetCount to return “Count”
4. For the EmpService add the attribute “ServiceBehaviour” with following
 - a. **InstanceContextMode = PerSession** – For every client a new Service instance is created and same is used for all the methods called by that client.
 - b. InstanceContextMode = **PerCall** – New Instance of Service is created for every method called by client
 - c. InstanceContextMode = **Single** – Only one Service instance is created and the same is used by all the clients for their methods called.
5. For every “InstanceContextMode”, Update Service Reference in the client application and view return value of “GetCount()” method

Walkthrough 2: Console Application as Hosting Environment

Server Application

1. Create a WCF Library Project (WCFLibrary) → File → New Project → Visual C# → WCF → WCF Service Library.

Project Name = WCFLibrary and SolutionName = WCFDemo

2. Delete from that the default Service1 and also delete the App.Config file of the project.
3. Add to the project a new WCF Service (MathService) which generates the following interface IDemoService and the DemoService.

```
using System;
using System.Runtime.Serialization;
using System.ServiceModel;
namespace WCFLibrary
{
    [ServiceContract]
    public interface IMathService
    {
        [OperationContract]
        int Add(int a, int b);
        [OperationContract]
        int Sub(int a, int b);
    }
    public class MathService : IMathService
    {
        public int Add(int a, int b)
        {
            return a + b;
        }
        public int Sub(int a, int b)
        {
            return a - b;
        }
    }
}
```

4. Add to the Solution a new Console Application (WCFServer)
5. Add reference to System.ServiceModel (Net Tab) and WCFLibrary (Projects Tab)
6. Move (Drag and Drop) the App.Config from ClassLibrary to Console Application Project and delete from WCFLibrary Project.

7. Edit the Autogenerated Main method in Program.cs

```
using System.ServiceModel
```

```
class Program
```

```
{
```

```
    static void Main()
```

```
    {
```

```
        ServiceHost sh;
```

```
        sh = new ServiceHost(typeof(WCFLibrary.MathService));
```

```
        sh.Open();
```

```
        Console.WriteLine("Server Started");
```

```
        Console.WriteLine("Press any key to close the Server");
```

```
        System.Threading.Thread.CurrentThread.Join();
```

```
    }
```

```
}
```

Client Application:

1. Create a Windows Application (MathServiceClient)
2. Right Click on project → select Add Service Reference
3. Handle Add Click and Sub Click as follows:

```
MathServiceClient ms = new MathService()
```

```
private void btnAdd_Click(object sender, EventArgs e)
```

```
{
```

```
    int res = dc.Add(int.Parse(txtN1.Text), int.Parse(txtN2.Text));
```

```
    txtResult.Text = res.ToString();
```

```
}
```

```
private void btnSub_Click(object sender, EventArgs e)
```

```
{
```

```
    int res = dc.Sub(int.Parse(txtN1.Text), int.Parse(txtN2.Text));
```

```
    txtResult.Text = res.ToString();
```

```
}
```