

LAB 05

NAME : Rathod Gautam Harshadkumar

Roll No. : CE009

ID. : 20CEUTG136

Lab : 05

Subject : WSD

✓ IserviceClass.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;

namespace CalculatorService
{
    // NOTE: You can use the "Rename" command on the "Refactor" menu to change
    the interface name "IService1" in both code and config file together.
    [ServiceContract]
    public interface IServiceClass
    {

        [OperationContract]
        ComplexType Addition(ComplexType num1, ComplexType num2);

        [OperationContract]
        ComplexType Subtraction(ComplexType num1, ComplexType num2);

    }

    [DataContract]
    public class ComplexType
    {
        [DataMember]
        public double RealNumber;
        [DataMember]
        public double ImaginaryNumber;

        public ComplexType(double num1, double num2)
        {
            this.RealNumber = num1;
            this.ImaginaryNumber = num2;
        }
    }
}
```

```

    }
}

```

✓ **ServiceClass.cs of service**

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;

namespace CalculatorService
{
    // NOTE: You can use the "Rename" command on the "Refactor" menu to change
    the class name "Service1" in both code and config file together.
    public class ServiceClass : IServiceClass
    {
        public ComplexType Addition(ComplexType cn1, ComplexType cn2)
        {
            return new ComplexType(cn1.RealNumber + cn2.RealNumber,
cn1.ImaginaryNumber + cn2.ImaginaryNumber);
        }

        public ComplexType Subtraction(ComplexType cn1, ComplexType cn2)
        {
            return new ComplexType(cn1.RealNumber - cn2.RealNumber,
cn1.ImaginaryNumber - cn2.ImaginaryNumber);
        }
    }
}

```

✓ **App.config of host(CalculatorServiceHost) - define the endpoint in host application**

```

<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>

  <system.serviceModel>
    <services>
      <service name="CalculatorService.ServiceClass">
        <!-- <endpoint address="" binding="basicHttpBinding"
contract="CalculatorService.IServiceClass">
          <identity>
            <dns value="localhost" />
          </identity>
        </endpoint> -->
        <!-- <endpoint address="" binding="wsHttpBinding"
contract="CalculatorService.IServiceClass"/> -->
        <endpoint address="" binding="netTcpBinding"
contract="CalculatorService.IServiceClass" />
        <endpoint address="tcpmex" binding="mexTcpBinding"
contract="IMetadataExchange" />

```

```

        <!-- <endpoint address="namedpipemex"
binding="mexNamedPipeBinding" contract="IMetadataExchange"/> -->
        <host>
            <baseAddresses>
                <!-- <add
baseAddress="http://localhost:8080/CalculatorService"/> -->
                <!-- <add
baseAddress="net.pipe://localhost/CalculatorService"/> -->
                <add
baseAddress="net.tcp://localhost:8000/CalculatorService"/>
            </baseAddresses>
        </host>
    </service>
</services>
<behaviors>
    <serviceBehaviors>
        <behavior>

            <serviceMetadata httpGetEnabled="false"
httpsGetEnabled="false"/>

            <serviceDebug
includeExceptionDetailInFaults="False" />
        </behavior>
    </serviceBehaviors>
</behaviors>
</system.serviceModel>
</configuration>

```

✓ Host form1.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.ServiceModel;

namespace CalculatorServiceHost
{
    public partial class Form1 : Form
    {
        ServiceHost sh = null;

        private void Form1_Load(object sender, EventArgs e)
        {
            sh = new ServiceHost(typeof(CalculatorService.ServiceClass));
            sh.Open();
            label1.Text = "Service Running";
        }

        private void Form1_FormClosing(object sender, FormClosingEventArgs e)
        {
            sh.Close();
        }
    }
}

```

```

        public Form1()
        {
            InitializeComponent();
        }
    }
}

```

✓ Client form1.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.ServiceModel;

namespace CalculatorServiceClient
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
        }

        private void button1_Click(object sender, EventArgs e)
        {
            TCP.ServiceClassClient client = new
CalculatorServiceClient.TCP.ServiceClassClient("NetTcpBinding_IServiceClass");
            CalculatorServiceClient.TCP.ComplexType Val1 = new
CalculatorServiceClient.TCP.ComplexType();
            Val1.RealNumber = double.Parse(textBox1.Text);
            Val1.ImaginaryNumber = double.Parse(textBox3.Text);

            CalculatorServiceClient.TCP.ComplexType Val2 = new
CalculatorServiceClient.TCP.ComplexType();
            Val2.RealNumber = double.Parse(textBox2.Text);
            Val2.ImaginaryNumber = double.Parse(textBox4.Text);

            CalculatorServiceClient.TCP.ComplexType result =
client.Addition(Val1, Val2);

            label3.Text = result.RealNumber.ToString();
            label4.Text = result.ImaginaryNumber.ToString();
            client.Close();
        }

        private void button2_Click(object sender, EventArgs e)
        {

```

```

        TCP.ServiceClassClient client = new
CalculatorServiceClient.TCP.ServiceClassClient("NetTcpBinding_IServiceClass");
        CalculatorServiceClient.TCP.ComplexType Val1 = new
CalculatorServiceClient.TCP.ComplexType();
        Val1.RealNumber = double.Parse(textBox1.Text);
        Val1.ImaginaryNumber = double.Parse(textBox3.Text);

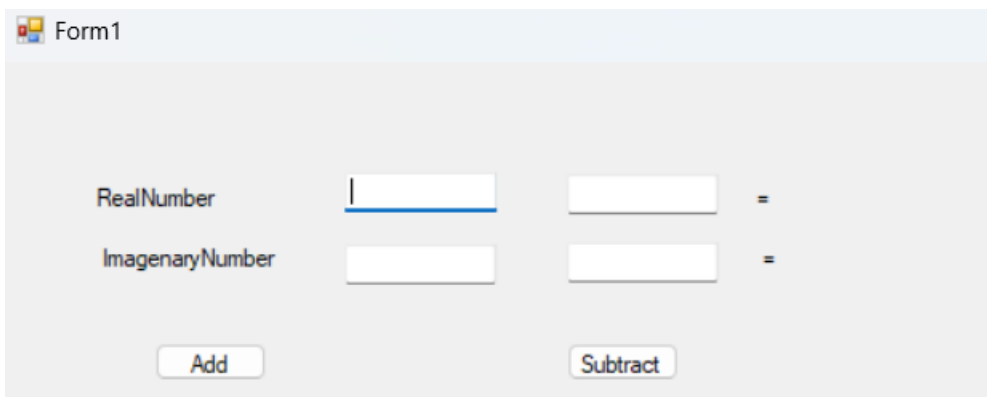
        CalculatorServiceClient.TCP.ComplexType Val2 = new
CalculatorServiceClient.TCP.ComplexType();
        Val2.RealNumber = double.Parse(textBox2.Text);
        Val2.ImaginaryNumber = double.Parse(textBox4.Text);

        CalculatorServiceClient.TCP.ComplexType result =
client.Subtraction(Val1, Val2);

        label3.Text = result.RealNumber.ToString();
        label4.Text = result.ImaginaryNumber.ToString();
        client.Close();
    }
}

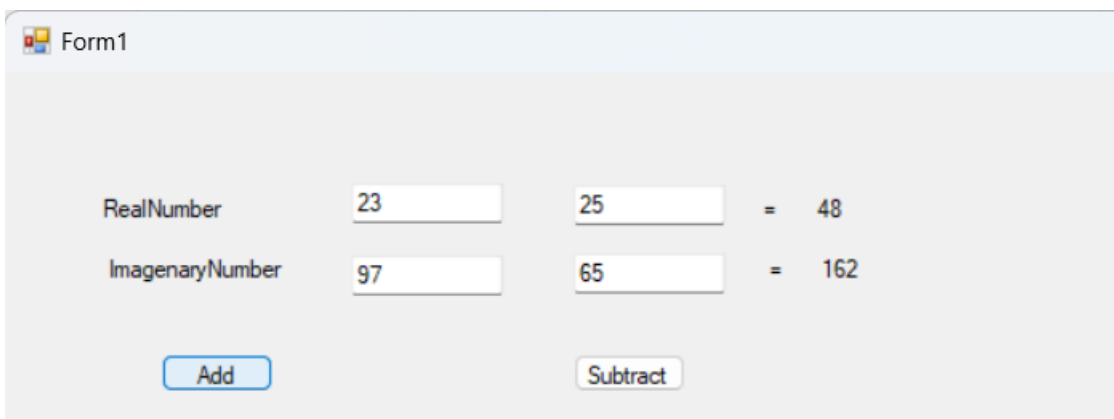
```

✓ Output Screen-shots



Form1

RealNumber	<input type="text"/>	<input type="text"/>	=
ImaginaryNumber	<input type="text"/>	<input type="text"/>	=



Form1

RealNumber	<input type="text" value="23"/>	<input type="text" value="25"/>	=	48
ImaginaryNumber	<input type="text" value="97"/>	<input type="text" value="65"/>	=	162

Form1

RealNumber 23 25 = -2

ImaginaryNumber 97 65 = 32

Add Subtract

✓ App.config of MsgContractSvc

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>

  <appSettings>
    <add key="aspnet:UseTaskFriendlySynchronizationContext" value="true" />
  </appSettings>
  <system.web>
    <compilation debug="true" />
  </system.web>
  <!-- When deploying the service library project, the content of the config file
must be added to the host's
  app.config file. System.Configuration does not support config files for
  libraries. -->
  <system.serviceModel>
    <services>
      <service name="MsgContractSvc.MsgContractClass">
        <host>
          <baseAddresses>
            <add baseAddress =
"http://localhost:8733/Design_Time_Addresses/MsgContractSvc/Service1/" />
          </baseAddresses>
        </host>
        <!-- Service Endpoints -->
        <!-- Unless fully qualified, address is relative to base address supplied
above -->
        <endpoint address="" binding="basicHttpBinding"
contract="MsgContractSvc.IServiceClass">
          <!--
            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>
        <!-- Metadata Endpoints -->
```

```

        <!-- The Metadata Exchange endpoint is used by the service to describe
        itself to clients. -->
        <!-- This endpoint does not use a secure binding and should be secured or
        removed before deployment -->
        <endpoint address="mex" binding="mexHttpBinding"
contract="IMetadataExchange"/>
    </service>
</services>
<behaviors>
    <serviceBehaviors>
        <behavior>
            <!-- To avoid disclosing metadata information,
            set the values below to false before deployment -->
            <serviceMetadata httpGetEnabled="True" httpsGetEnabled="True"/>
            <!-- To receive exception details in faults for debugging purposes,
            set the value below to true. Set to false before deployment
            to avoid disclosing exception information -->
            <serviceDebug includeExceptionDetailInFaults="False" />
        </behavior>
    </serviceBehaviors>
</behaviors>
</system.serviceModel>
</configuration>

```

✓ IserviceClass.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.ServiceModel;
using System.Text;
using System.Threading.Tasks;

namespace MsgContractSvc
{
    [ServiceContract]
    public interface IServiceClass
    {
        [OperationContract]
        Composite Addition(Composite number);

        [OperationContract]
        Composite Subtraction(Composite number);
    }

    [MessageContract]
    public class Composite
    {
        private string client_id;
        private double complex1;
        private double complex2;
        private double complex3;

        public Composite() { }
        public Composite(Composite number)
        {
            this.client_id = number.client_id;

```

```

        this.complex1 = number.complex1;
        this.complex2 = number.complex2;
        this.complex3 = number.complex3;
    }

    [MessageHeader]
    public string Client_id
    {
        get { return client_id; }
        set { client_id = value; }
    }

    [MessageBodyMember]
    public double Complex1
    {
        get { return complex1; }
        set { complex1 = value; }
    }

    [MessageBodyMember]
    public double Complex2
    {
        get { return complex2; }
        set { complex2 = value; }
    }

    [MessageBodyMember]
    public double Complex3
    {
        get { return complex3; }
        set { complex3 = value; }
    }
}
}

```

✓ MsgContractClass.cs of service

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace MsgContractSvc
{
    public class MsgContractClass : IServiceClass
    {
        public Composite Addition(Composite request)
        {
            Composite response = new Composite(request);
            response.Complex3 = request.Complex1 + request.Complex2;
            response.Client_id = "CL1234";
            return response;
        }
        public Composite Subtraction(Composite request)
    }
}

```



```

        {
            Composite response = new Composite(request);
            response.Complex3 = request.Complex1 - request.Complex2;
            response.Client_id = "CL1234";
            return response;
        }
    }
}

```

✓ Form1.cs Host

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.ServiceModel;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace MsgContractSvcHost
{
    public partial class Form1 : Form
    {
        ServiceHost sh = null;
        public Form1()
        {
            InitializeComponent();

            private void Form1_Load(object sender, EventArgs e)
            {
                sh = new ServiceHost(typeof(MsgContractSvc.MsgContractClass));
                sh.Open();
                label1.Text = "Service Running";
            }
        }
    }
}

```

✓ Host App.config - define endpoints

```

<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2"
    />
    </startup>

    <system.serviceModel>
        <services>
            <service name="MsgContractSvc.MsgContractClass"
behaviorConfiguration="metadataSupport">

```

```

        <endpoint address="" binding="wsHttpBinding"
contract="MsgContractSvc.IServiceClass"/>
        <endpoint address="tcpmex" binding="mexTcpBinding"
contract="IMetadataExchange" />
        <endpoint address="namedpipemex"
binding="mexNamedPipeBinding" contract="IMetadataExchange"/>

        <host>
            <baseAddresses>
                <add
baseAddress="http://localhost:9000/MsgContractSvc"/>
                <add
baseAddress="net.pipe://localhost/MsgContractSvc"/>
                <add
baseAddress="net.tcp://localhost:8000/MsgContractSvc"/>
            </baseAddresses>
        </host>
    </service>
</services>
<behaviors>
    <serviceBehaviors>
        <behavior name="metadataSupport">

            <serviceMetadata httpGetEnabled="false"
httpsGetEnabled="false" httpGetUrl="" />

            <serviceDebug
includeExceptionDetailInFaults="False" />
        </behavior>
    </serviceBehaviors>
</behaviors>
</system.serviceModel>
</configuration>

```

✓ client form1.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace MsgContractSvcClient
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {

```

```

        TCP.ServiceClassClient client = new
MsgContractSvcClient.TCP.ServiceClassClient("WSHttpBinding_IServiceClass");
        MsgContractSvcClient.TCP.Composite Val1 = new
MsgContractSvcClient.TCP.Composite();
        Val1.Complex1 = double.Parse(textBox1.Text);
        Val1.Complex2 = double.Parse(textBox2.Text);

        MsgContractSvcClient.TCP.Composite reply =
((TCP.IServiceClass)client).Addition(Val1);
        textBox3.Text = reply.Complex3.ToString();
        textBox4.Text = reply.Client_id;
        client.Close();

    }

    private void button2_Click(object sender, EventArgs e)
    {
        TCP.ServiceClassClient client = new
MsgContractSvcClient.TCP.ServiceClassClient("WSHttpBinding_IServiceClass");
        MsgContractSvcClient.TCP.Composite Val1 = new
MsgContractSvcClient.TCP.Composite();
        Val1.Complex1 = double.Parse(textBox1.Text);
        Val1.Complex2 = double.Parse(textBox2.Text);

        MsgContractSvcClient.TCP.Composite reply =
((TCP.IServiceClass)client).Subtraction(Val1);
        textBox3.Text = reply.Complex3.ToString();
        textBox4.Text = reply.Client_id;
        client.Close();

    }

    private void Form1_Load(object sender, EventArgs e)
    {

    }

}

```

✓ soap code of request

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <Action s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">http://tempuri.org/IServiceClass/Subtraction</Action>
    <h:Client_id i:nil="true" xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns:h="http://tempuri.org/" />
  </s:Header>
  <s:Body>
    <Composite xmlns="http://tempuri.org/">
      <Complex1>20</Complex1>

```

```

        <Complex2>30</Complex2>
        <Complex3>0</Complex3>
    </Composite>
</s:Body>
</s:Envelope>

```

✓ soap code response

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <h:Client_id xmlns:h="http://tempuri.org/">CL1234</h:Client_id>
  </s:Header>
  <s:Body>
    <Composite xmlns="http://tempuri.org/">
      <Complex1>20</Complex1>
      <Complex2>30</Complex2>
      <Complex3>-10</Complex3>
    </Composite>
  </s:Body>
</s:Envelope>

```

✓ Output Screen-Shots

Form1

Complex1

Complex2

Complex3

Client_id

Add Subtract

Enter Complex1 and Complex2 and press button

client_id and output in Complex3 is generated by

Form1

Complex1	12
Complex2	55
Complex3	67
Client_id	CL1234

Add Subtract

Form1

Complex1	12
Complex2	55
Complex3	-43
Client_id	CL1234

Add Subtract

File Tools Help

My Service Projects

- http://localhost:8733/Design_Time_Addresses/MyServiceClass (BasicHttpBinding_IServiceClass)
 - Addition()
 - AdditionAsync()
 - Subtraction()
 - SubtractionAsync()
- Config File

Addition Subtraction

Request		
Name	Value	Type
request	Composite	Composite
Client_id	(null)	System.String
Complex1	0	System.Double
Complex2	0	System.Double
Complex3	0	System.Double

Response ☐ Start a new proxy

Name	Value	Type
------	-------	------

File Tools Help

My Service Projects

- http://localhost:8733/Design_Time_Addresses/MyServiceClass (BasicHttpBinding IServiceClass)
- Addition()
- AdditionAsync()
- Subtraction()
- SubtractionAsync()
- Config File

Addition

Request

Name	Value	Type
request	Composite	Composite
Client_id	(null)	System.String
Complex1	12	System.Double
Complex2	13	System.Double
Complex3	0	System.Double

Response

☐ Start a new proxy

Name	Value	Type
(return)	Composite	Composite
Client_id	"CL1234"	System.String
Complex1	12	System.Double
Complex2	13	System.Double
Complex3	25	System.Double

File Tools Help

My Service Projects

- http://localhost:8733/Design_Time_Addresses/MyServiceClass (BasicHttpBinding IServiceClass)
- Addition()
- AdditionAsync()
- Subtraction()
- SubtractionAsync()
- Config File

Addition Subtraction

Request

Name	Value	Type
request	Composite	Composite
Client_id	(null)	System.String
Complex1	20	System.Double
Complex2	30	System.Double
Complex3	0	System.Double

Response

☐ Start a new proxy

Name	Value	Type
(return)	Composite	Composite
Client_id	"CL1234"	System.String
Complex1	20	System.Double
Complex2	30	System.Double
Complex3	-10	System.Double