Practical 2: Creating a Web Service with ASP.NET

Objective:

- ❑ Create a Web Service
- ❑ Create a web application to consume the web service created

Creating a Web Application

First, we create a web application to host the web service.

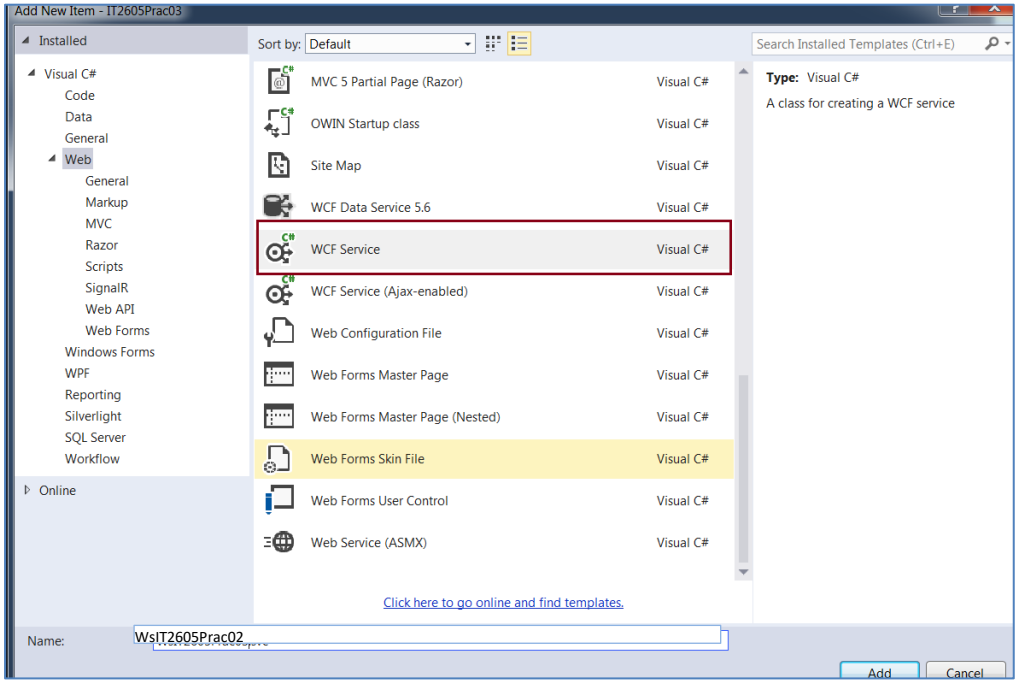
1. Start Visual Studio.
2. Create a new ASP.NET Empty Web Application project using C# template.
    a. For Language, choose "Visual C#".
    b. Named the project as "IT2605Prac02"
    c. For the Location, specify the location of this project. Click Ok button.
    d. Select an **empty** template, click OK to create project.

Exercise 1 : Create a simple Web Service

Add a web service to the web application.

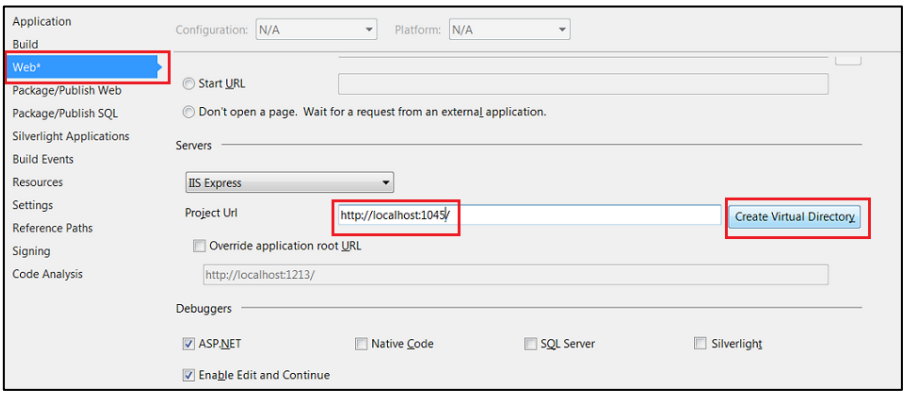
3. Right-click your project in the *Solution Explorer* and click *Add New Item*.
4. In the *Add New Item* dialog (See Figure 1):
    a. Select Visual C# under Installed Templates.
    b. Select ***WCF Service***. (Note: **NOT** Web Service.)
    c. Change the filename to *WsIT2605Prac02.svc*.
    d. Click *Add* button.

Figure 1



e. Set the port number of your project.

- In Solution Explorer, right click the Project folder and then select Properties.
- Click the Web tab.
- In the Servers section, use IIS Express, in the Project URL box change the port number any number you like. In this case the project use 1045.
- To the right of the Project URL box, click Create Virtual Directory, and then click OK.
- In the File menu, click Save Selected Items.



5. Windows Communication Foundation (WCF) is a framework for building service-oriented application. WCF supports several communication protocols including HTTP, TCP, UDP, MSMQ. The communication data message can be SOAP or JSON.

6. Examine Solution Explorer and note the new files added to your project (See Figure 2).

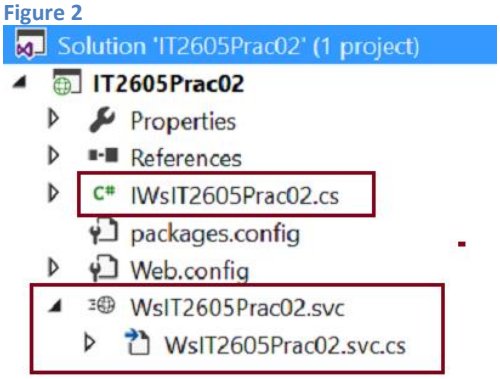
Q: How many new files are created?

A: _____

Q: What is the difference between the names of the 2 *.cs* files?

B: _____

Figure 2



Understanding the Structure ASP.NET WCF Service .cs File

7. ***IWsIT2605Prac02.cs*** is the contract file, it contains a collection of operations. Operations are the methods exposed to the client.
8. Open *IWsIT2605Prac02.cs*(see Figure 3):
    a. There is '[ServiceContract]' that declares the Interface as a Service.
    a. Above the DoWork() method, there is '[OperationContract]'. This defines the DoWork() method as an Operation of the Service.

Figure 3

```
using System.ServiceModel;
using System.Text;

namespace IT2605Prac02
{
    // NOTE: You can use the "Rename" command on the "Refactor
    [ServiceContract]
    1 reference
    public interface IWsIT2605Prac02
    {
        [OperationContract]
        1 reference
        void DoWork( );

    }
}
```

9. *WsIT2605Prac02.svc implements the contract* (see Figure 4):
    a. Open WsIT2605Prac02.svc.cs, an empty DoWork() method is declared. Delete DoWork() from BOTH WsIT2605Prac02.svc.cs and IWsIT2605Prac02.cs files.

Figure 4

```
namespace IT2605Prac02
{
    // NOTE: You can use the "Rename" command on the "Refacto
    // NOTE: In order to launch WCF Test Client for testing t
    0 references
    public class WsIT2605Prac02 : IWsIT2605Prac02
    {
        1 reference
        public void DoWork()
        {
        }

    }
}
```

10. Declare a new HelloWorld() method to accept the user name and return a welcome message as follow sentence

Hello, *user name* ! Welcome to the world of Web Services

11. Open *WsIT2605Prac02.svc.cs,* declare a new HelloWorld() method (see Figure 5):
    a. This method returns a string.
    b. It requires an input s*tring* parameter named *username*.
    c. Complete your code below

Figure 5

```
namespace IT2605Prac02
{
    // NOTE: You can use the "Rename" command on the "Refactor" menu to change the clas
    // NOTE: In order to launch WCF Test Client for testing this service, please select
    0 references
    public class WsIT2605Prac02 : IWsIT2605Prac02
    {
        1 reference
        public _____ HelloWorld( ____  _______ )
        {
            return "Hello " + username + ", Welcome to the world of the Web Services";
        }

    }
}
```

12. Save your work.

13. Next, create the *HelloWorld()* operation in *IWsIT2605Prac02.cs.* Open *IWsIT2605Prac02.cs* and declare the return type, method name and the parameter as shown in **Figure 6.**

Figure 6

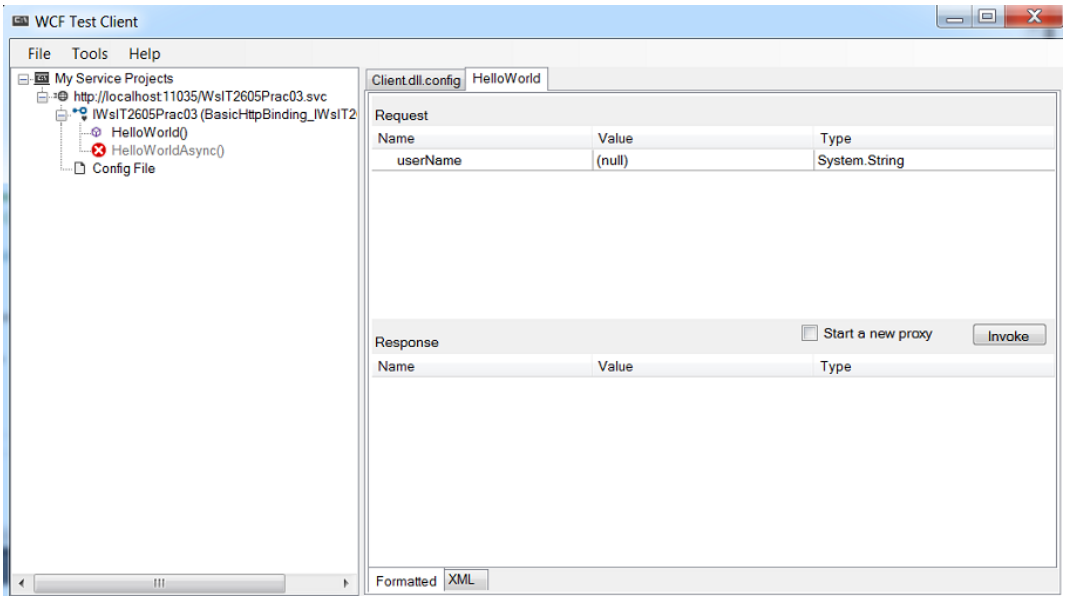
```
namespace IT2605Prac02
{
    // NOTE: You can use the "Rename" command on the
    [ServiceContract]
    public interface IWsIT2605Prac02
    {
            [OperationContract]
            string HelloWorld(string username);

     }
}
```

Test the Web Service

Visual Studio provides a test tool to ensure that your web service operations work correctly.

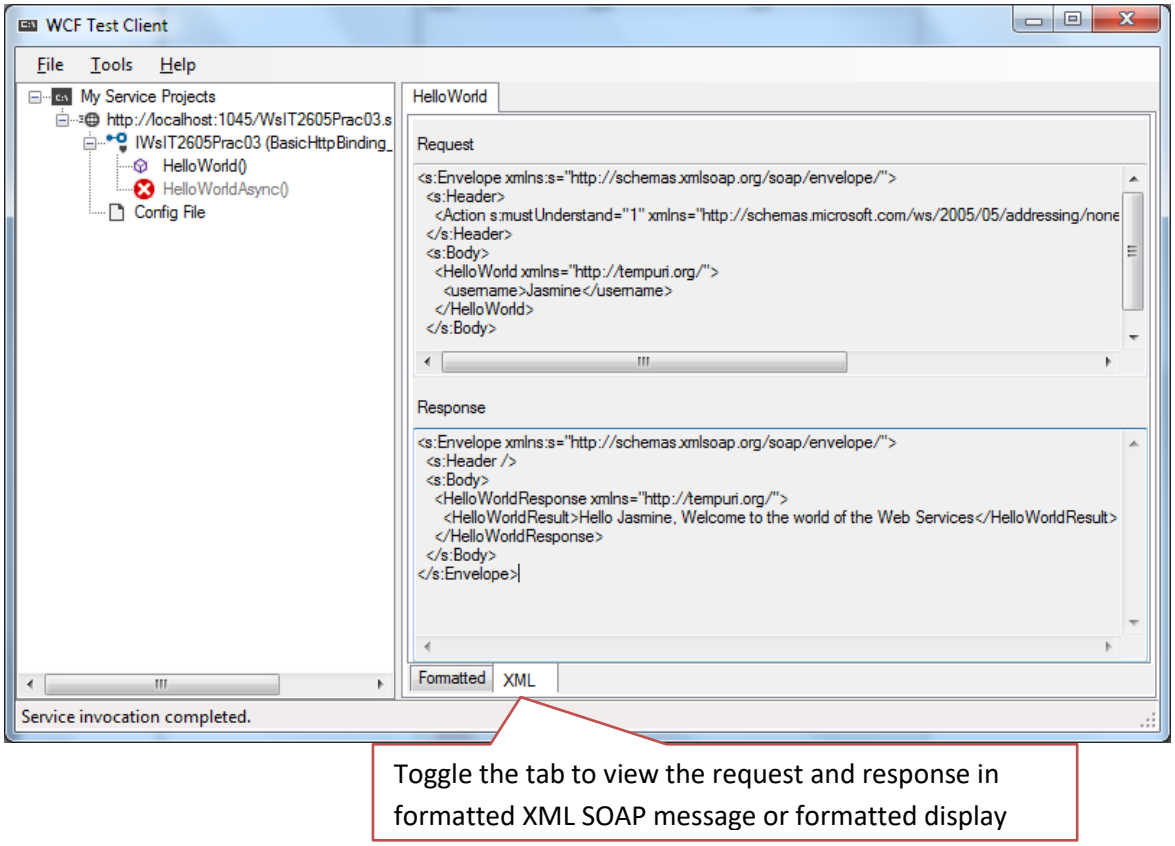
1. Right click the service ***WsIT2605Prac02.svc*** in Solution Explorer and click set as Start Page.
2. Click the run button at the menu or press CTRL F5
3. The WCF Test Client appears, as shown in Figure 7.

Figure 7



4. In the left pane, double click the HelloWorld() operation.
5. Enter any value in username, and then click Invoke
6. If security prompt dialog box appears, click OK to confirm that you will send information to the service,
7. The operation result appears as in Figure 8.

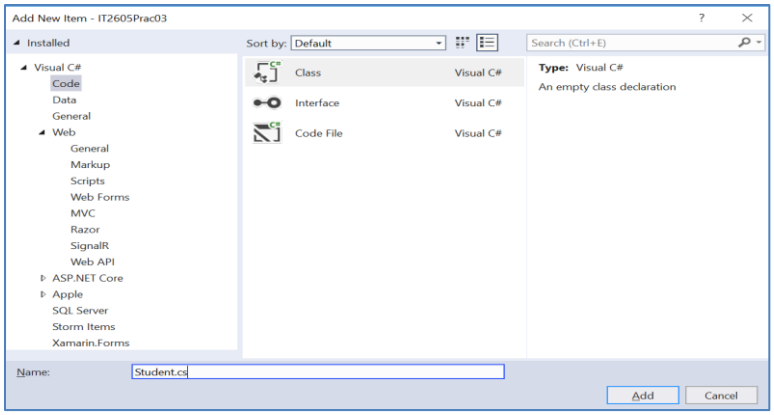
Figure 8



Exercise 2 : Create a web method returning complex data type

The operation contract in the earlier exercise return a simple data type. But usually, the web service is required to create a more complex data such as object. Now, we will add a new operation contract GetStudent which takes in an integer value as student id and return a Student object. Student object has four properties: StudentID, StudentName, CourseStudy and StudentGPA.

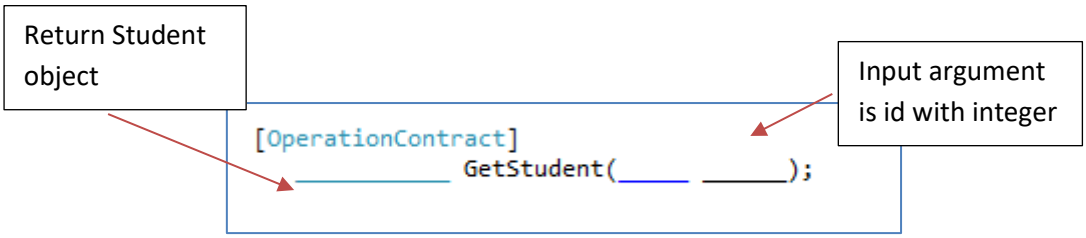
1. Create user defined data using data contract.
    - Right click project in solution explorer, select add new item then select Class
    - Named the class as Student.cs



- In Student class, add [DataContract] annotation above the class. Declared four properties and annotate each property with [DataMember]. Make sure using System.Runtime.Serialization is added.

```
using System.Runtime.Serialization;

namespace IT2605Prac02
{
    public class Student
    {
        [DataMember]
        public int StudentID { get; set; }

        [DataMember]
        public string StudentName { get; set; }

        [DataMember]
        public string CourseStudy { get; set; }

        [DataMember]
        public decimal StudentGPA { get; set; }
    }
}
```

2. Open IWsIT2605Prac02.cs, declare a new operation contract name GetStudent. Based on the requirement stated earlier, complete the following code snippet within the interface.

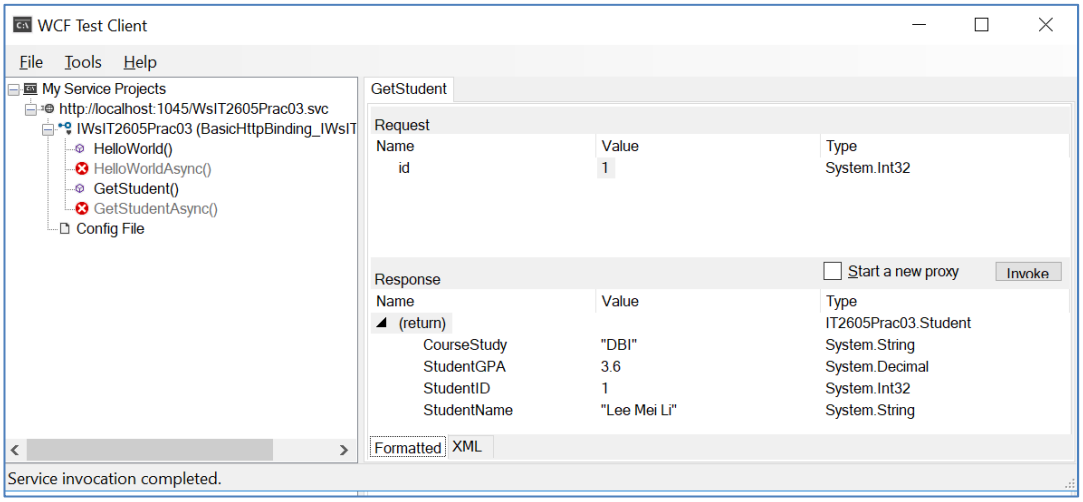


3. Open *WsIT2605Prac02.svc.cs to complete the snippet t*o implement back end logic of GetStudent operation contract. This time, we simply hard code the value to the property of all the DataMember and return the instance to the client.

   The following three lines of codes will help you to get started. Complete the snippet to assign value to all the instant properties.

```
public Student GetStudent(int id)
{
    Student me = new Student();
    me.StudentID = id;
    me.StudentName = "Lee Mei Li";


    return me;
}
```

5. Press F5 to test the web service.
6. From the WCF test client, you can find a new GetStudent operation in the left pane.
7. Double click the GetStudent() operation.
8. Enter any integer value for id, and click Invoke, you will obtain the formatted operation result. Toggle to XML, you will obtain the result represented in SOAP format.



9. No matter what input you enter in the id, the operation result will always be the same. This is because we have not added dynamic retrieval from data entity yet.
10. Back up the practical, you are going to consume this web service in the next practical.

~ End ~

Checkpoint

By the end of this practical, you should be able to:

- ✓ Learn ASP.NET WCF framework
- ✓ Able to create a web service