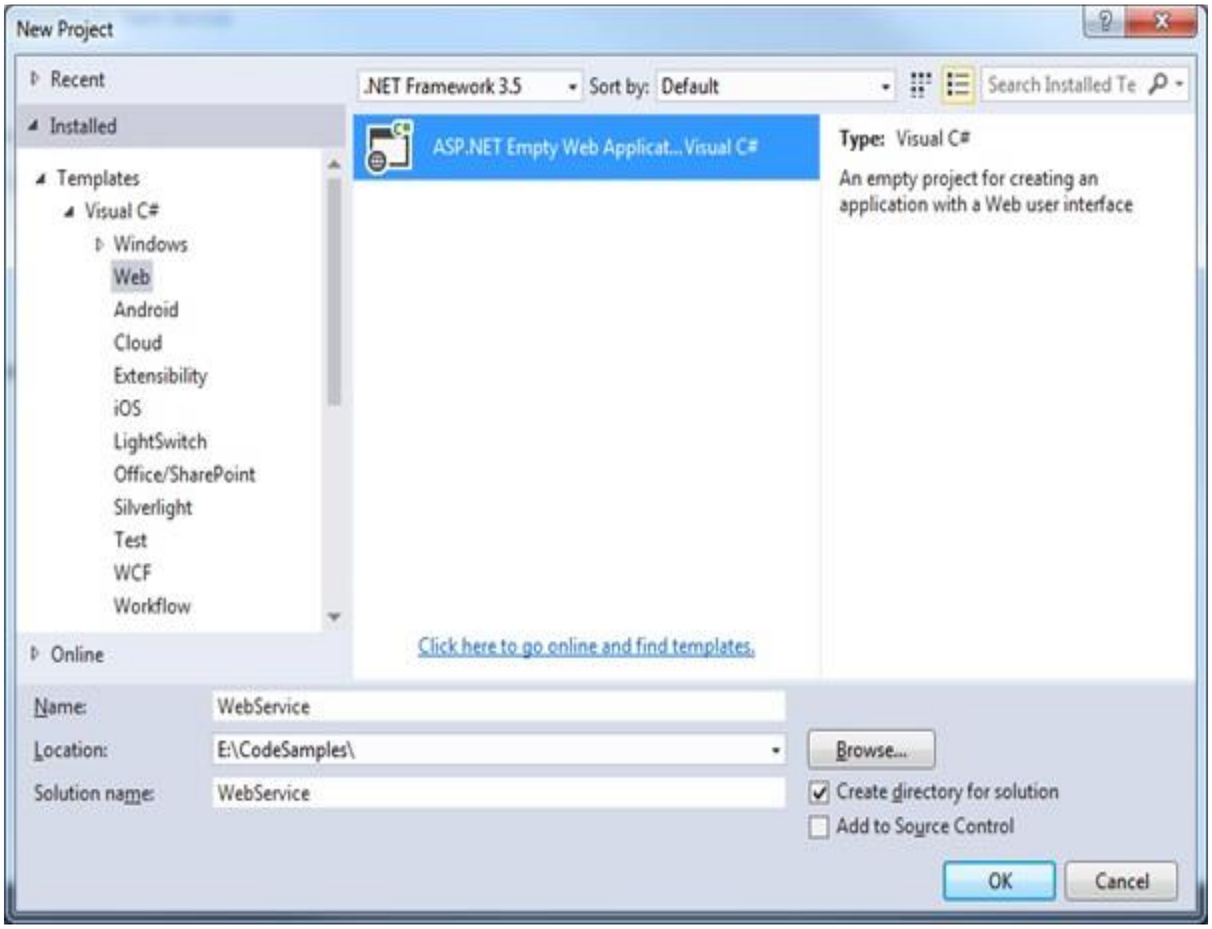
How-To Create and Consume a Web Service in C# and ASP.NET

Create a new Visual Studio project



Type: Visual C#

An empty project for creating an application with a Web user interface

Name: WebService

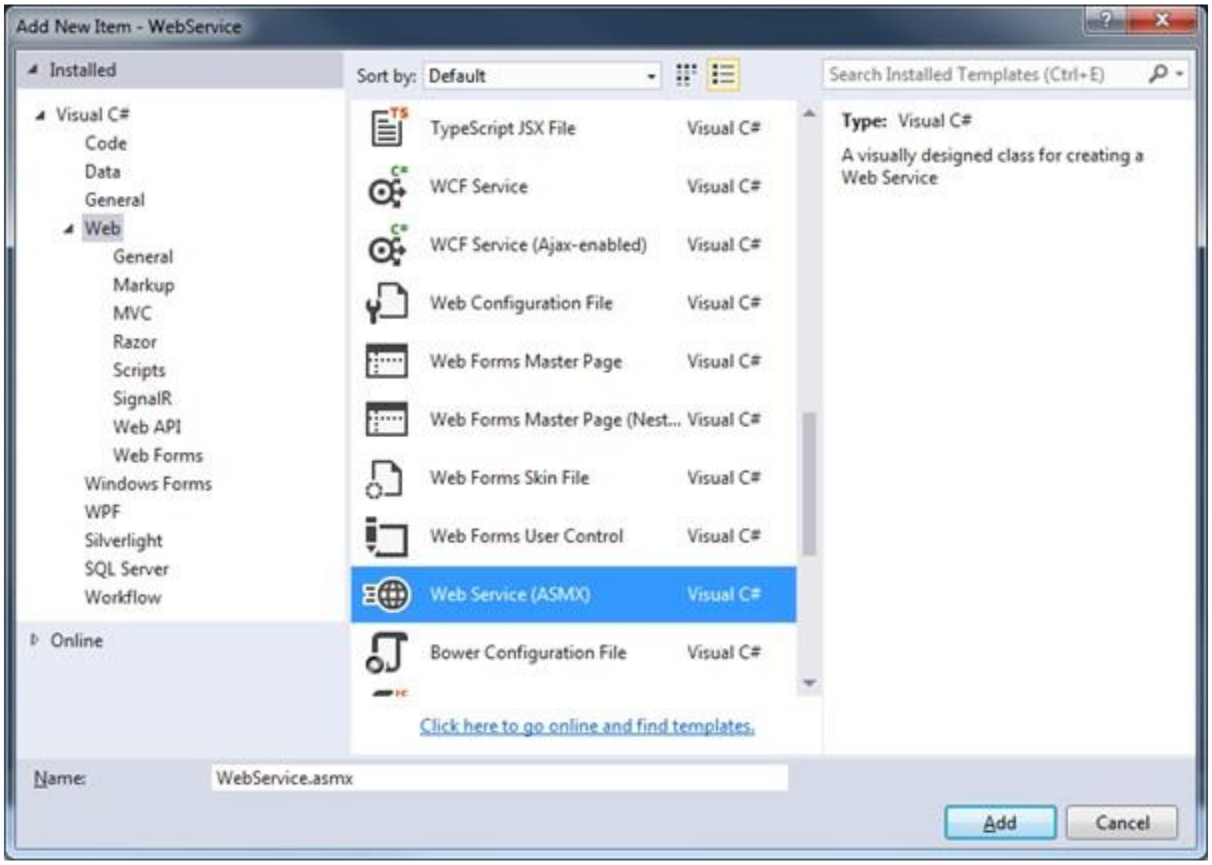
Location: E:\CodeSamples\

Solution name: WebService

☑ Create directory for solution

☐ Add to Source Control

Right click on the project and select Add > New Item > Web Service (ASMX)


Add New Item - WebService
Installed
Visual C#
Code
Data
General
Web
General
Markup
MVC
Razor
Scripts
SignalR
Web API
Web Forms
Windows Forms
WPF
Silverlight
SQL Server
Workflow
Online
Sort by: Default
Search Installed Templates (Ctrl+E)
TypeScript JSX File Visual C#
WCF Service Visual C#
WCF Service (Ajax-enabled) Visual C#
Web Configuration File Visual C#
Web Forms Master Page Visual C#
Web Forms Master Page (Nest... Visual C#
Web Forms Skin File Visual C#
Web Forms User Control Visual C#
Web Service (ASMX) Visual C#
Bower Configuration File Visual C#
Click here to go online and find templates.
Add
Cancel

Type: Visual C#

A visually designed class for creating a Web Service

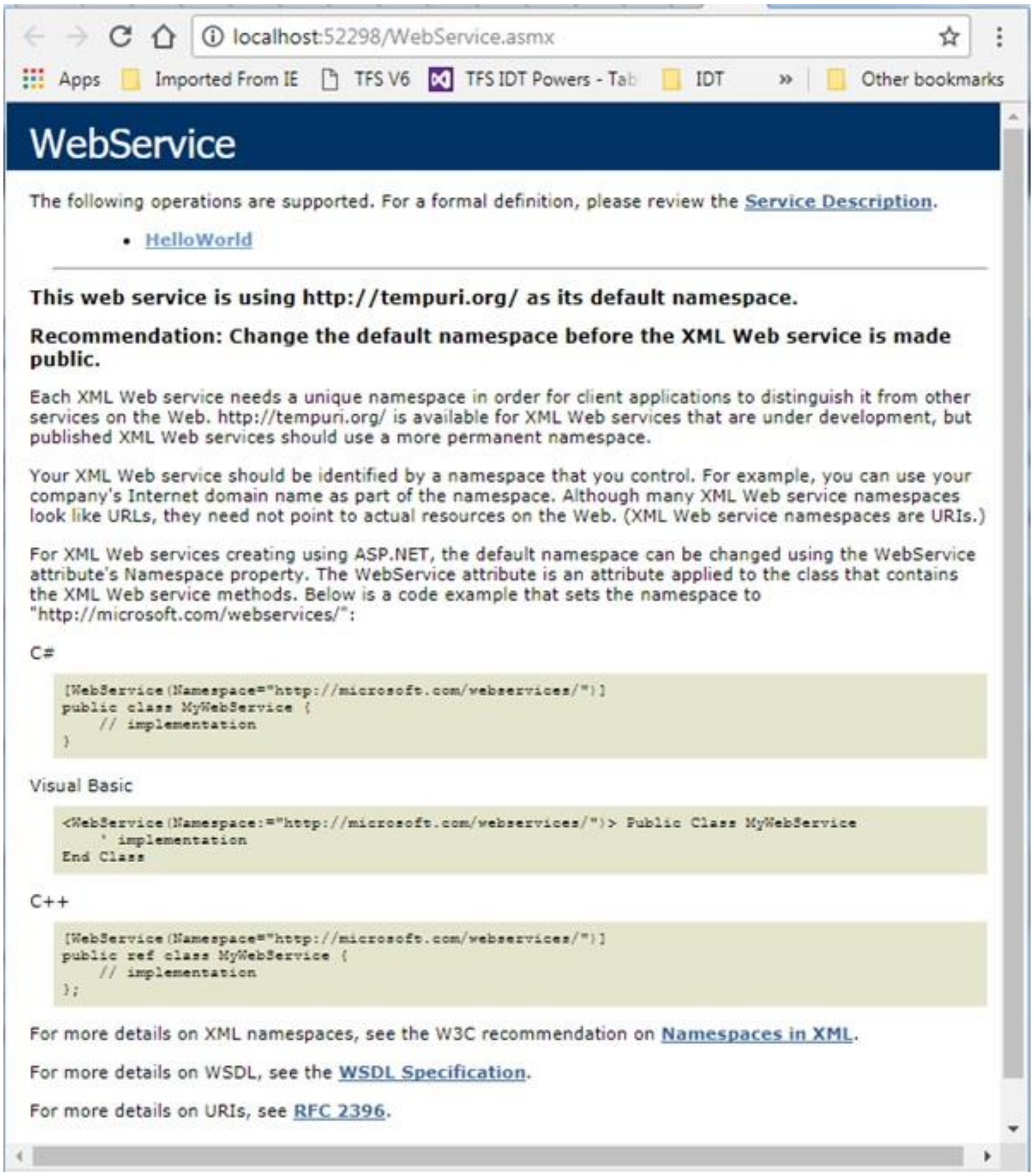
Name: WebService.asmx

See that the following code gets automatically generated in WebService.asmx.cs

WebService.asmx.cs

```
using System.Web.Services;

namespace WebService
{
    /// <summary>
    /// Summary description for WebService
    /// </summary>
    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    // To allow this Web Service to be called from script, using ASP.NET AJAX, uncomme
    // [System.Web.Script.Services.ScriptService]
    public class WebService : System.Web.Services.WebService
    {

        [WebMethod]
        public string HelloWorld()
        {
            return "Hello World";
        }
    }
}
```

On compiling and running the project you are directed to a web browser that points to your service containing the single 'HelloWorld' method:



WebService

The following operations are supported. For a formal definition, please review the Service Description.

- HelloWorld

This web service is using http://tempuri.org/ as its default namespace.

Recommendation: Change the default namespace before the XML Web service is made public.

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on the Web. http://tempuri.org/ is available for XML Web services that are under development, but published XML Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can use your company's Internet domain name as part of the namespace. Although many XML Web service namespaces look like URLs, they need not point to actual resources on the Web. (XML Web service namespaces are URIs.)

For XML Web services creating using ASP.NET, the default namespace can be changed using the WebService attribute's Namespace property. The WebService attribute is an attribute applied to the class that contains the XML Web service methods. Below is a code example that sets the namespace to "http://microsoft.com/webservices/":

C#

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public class MyWebService {
    // implementation
}
```

Visual Basic

```
<WebService(Namespace:="http://microsoft.com/webservices/")> Public Class MyWebService
    ' implementation
End Class
```

C++

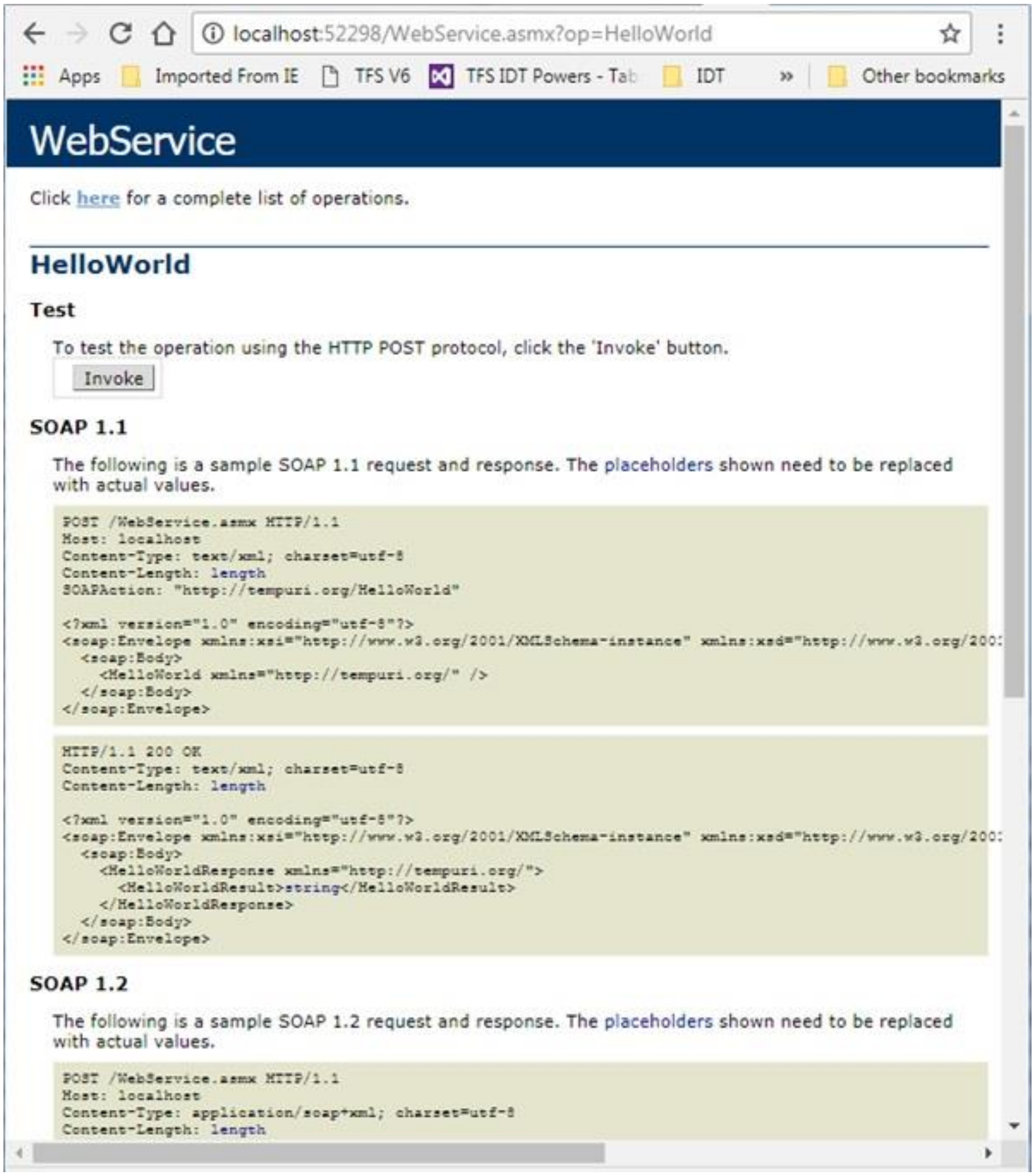
```
[WebService(Namespace="http://microsoft.com/webservices/")]
public ref class MyWebService {
    // implementation
};
```

For more details on XML namespaces, see the W3C recommendation on Namespaces in XML.

For more details on WSDL, see the WSDL Specification.

For more details on URIs, see RFC 2396.

And when you click on the 'HelloWorld' link you are directed to the page to invoke the selected operation:



WebService

Click here for a complete list of operations.

HelloWorld

Test

To test the operation using the HTTP POST protocol, click the 'Invoke' button.

Invoke

SOAP 1.1

The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

```
POST /WebService.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/HelloWorld"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/200
  <soap:Body>
    <HelloWorld xmlns="http://tempuri.org/" />
  </soap:Body>
</soap:Envelope>
```

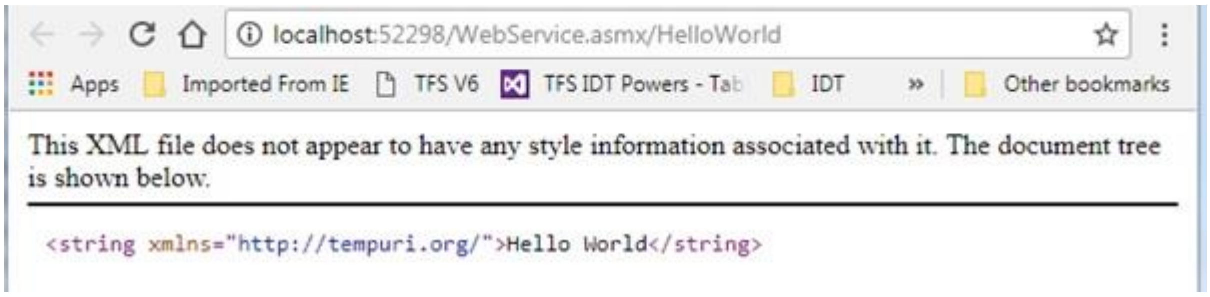
```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/200
  <soap:Body>
    <HelloWorldResponse xmlns="http://tempuri.org/">
      <HelloWorldResult>string</HelloWorldResult>
    </HelloWorldResponse>
  </soap:Body>
</soap:Envelope>
```

SOAP 1.2

The following is a sample SOAP 1.2 request and response. The placeholders shown need to be replaced with actual values.

```
POST /WebService.asmx HTTP/1.1
Host: localhost
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
```

Clicking Invoke then directs you to the XML file:



This XML file does not appear to have any style information associated with it. The document tree is shown below.

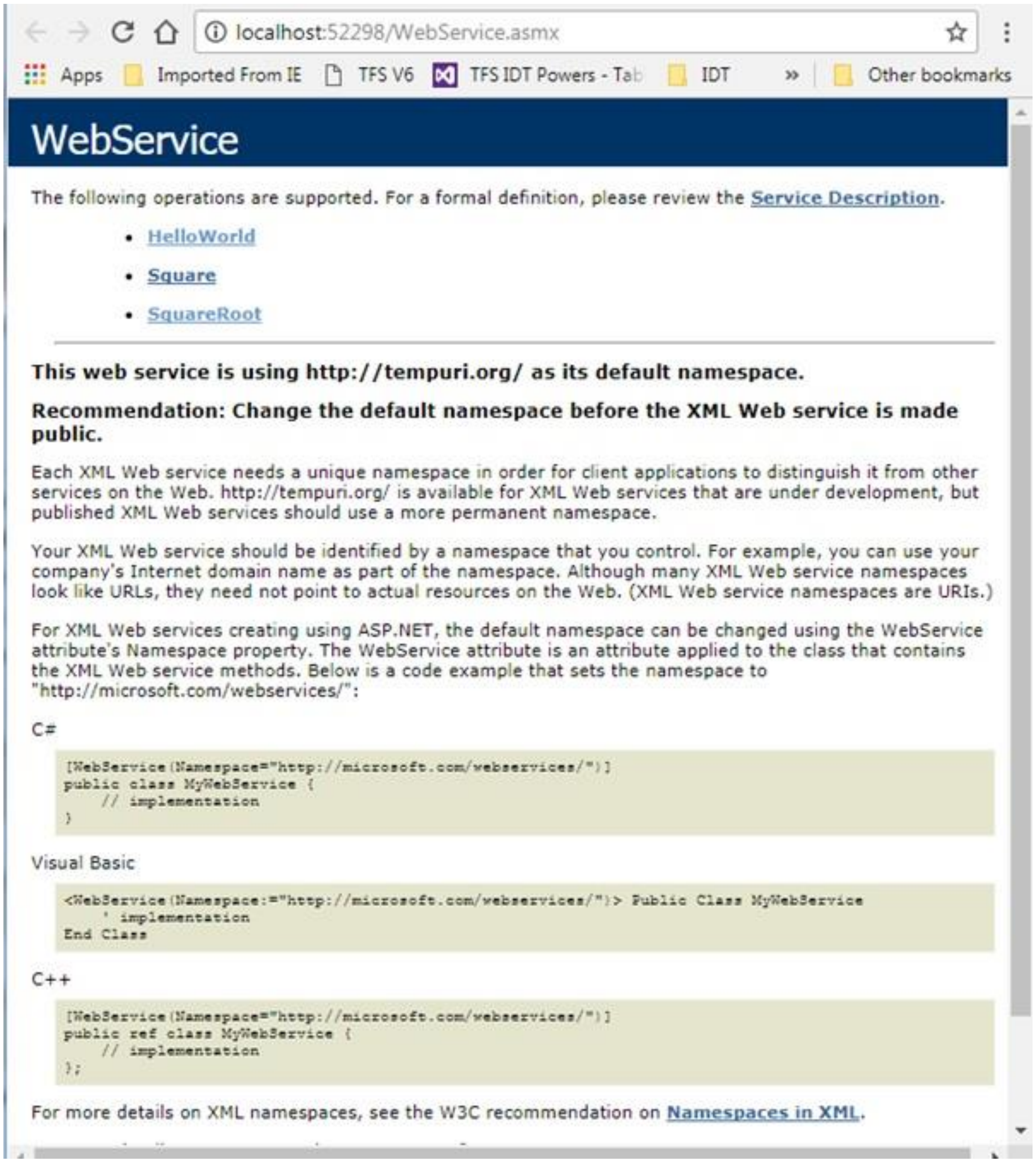
```
<string xmlns="http://tempuri.org/">Hello World</string>
```

Add two additional services, Square and SquareRoot to WebService.asmx.cs:

WebService.asmx.cs

```
using System;
using System.Web.Services;

namespace WebService
{
    /// <summary>
    /// Summary description for WebService
    /// </summary>
    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    // To allow this Web Service to be called from script, using ASP.NET AJAX, uncomme
    // [System.Web.Script.Services.ScriptService]
    public class WebService : System.Web.Services.WebService
    {

        [WebMethod]
        public string HelloWorld()
        {
            return "Hello World";
        }

        [WebMethod]
        public double Square(double x)
        {
            return x * x;
        }

        [WebMethod]
        public double SquareRoot(float x)
        {
            return Math.Sqrt(x);
        }
    }
}
```

Rebuild and re-run your project and notice that two additional services are displayed:



WebService

The following operations are supported. For a formal definition, please review the Service Description.

- HelloWorld
- Square
- SquareRoot

This web service is using http://tempuri.org/ as its default namespace.

Recommendation: Change the default namespace before the XML Web service is made public.

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on the Web. http://tempuri.org/ is available for XML Web services that are under development, but published XML Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can use your company's Internet domain name as part of the namespace. Although many XML Web service namespaces look like URLs, they need not point to actual resources on the Web. (XML Web service namespaces are URIs.)

For XML Web services creating using ASP.NET, the default namespace can be changed using the WebService attribute's Namespace property. The WebService attribute is an attribute applied to the class that contains the XML Web service methods. Below is a code example that sets the namespace to "http://microsoft.com/webservices/":

C#

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public class MyWebService {
    // implementation
}
```

Visual Basic

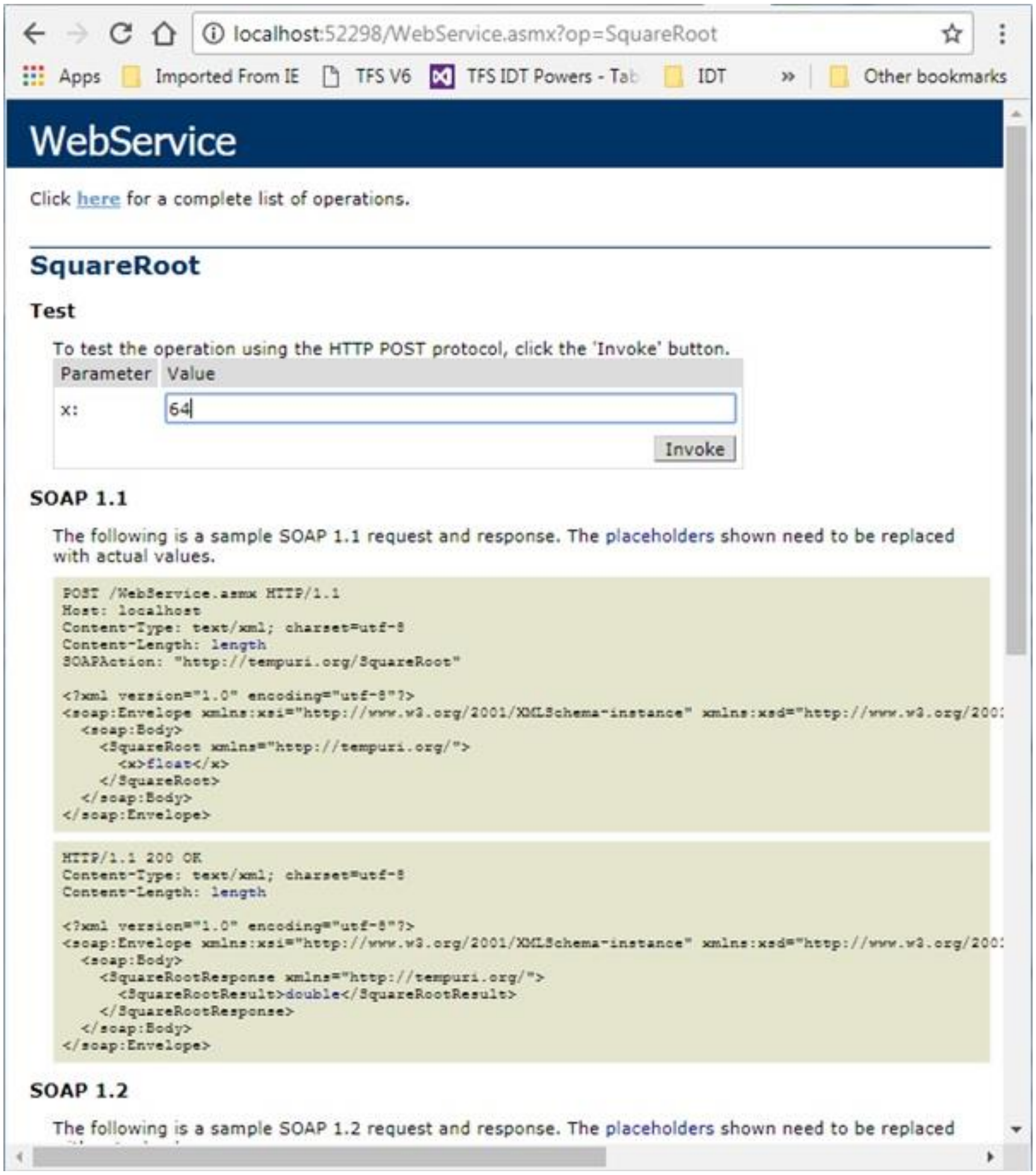
```
<WebService(Namespace:="http://microsoft.com/webservices/")> Public Class MyWebService
    ' implementation
End Class
```

C++

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public ref class MyWebService {
    // implementation
};
```

For more details on XML namespaces, see the W3C recommendation on Namespaces in XML.

Click on one of the new examples, SquareRoot for example, and invoke this:



WebService

Click here for a complete list of operations.

SquareRoot

Test

To test the operation using the HTTP POST protocol, click the 'Invoke' button.

Parameter	Value
x:	64

SOAP 1.1

The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

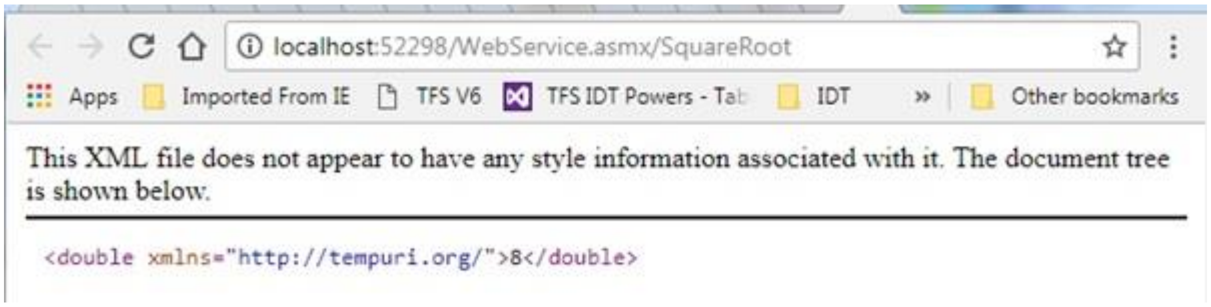
```
POST /WebService.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/SquareRoot"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/200
  <soap:Body>
    <SquareRoot xmlns="http://tempuri.org/">
      <x>float</x>
    </SquareRoot>
  </soap:Body>
</soap:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/200
  <soap:Body>
    <SquareRootResponse xmlns="http://tempuri.org/">
      <SquareRootResult>double</SquareRootResult>
    </SquareRootResponse>
  </soap:Body>
</soap:Envelope>
```

SOAP 1.2

The following is a sample SOAP 1.2 request and response. The placeholders shown need to be replaced

Giving use the answer in XML format as shown:

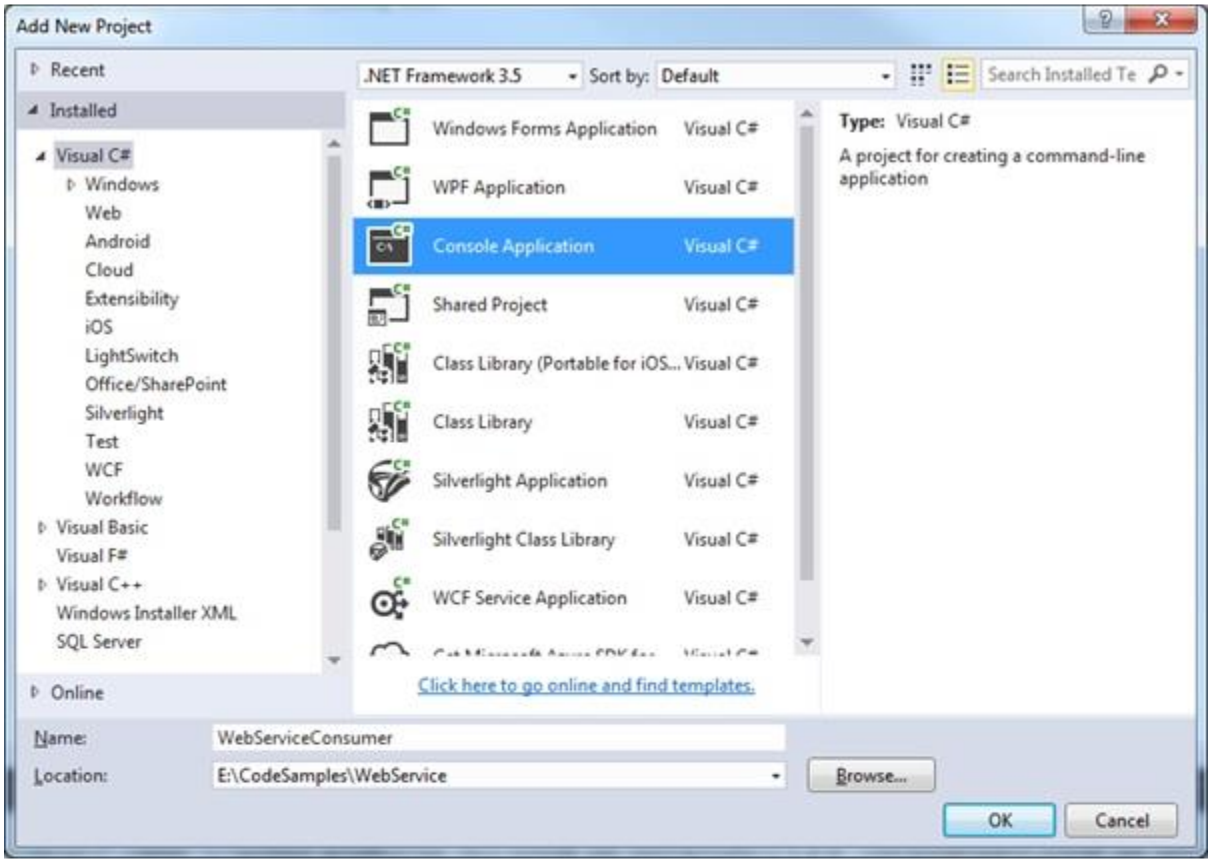


This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<double xmlns="http://tempuri.org/">8</double>
```

Consuming the web service

Create a new Console Application in your solution



Type: Visual C#

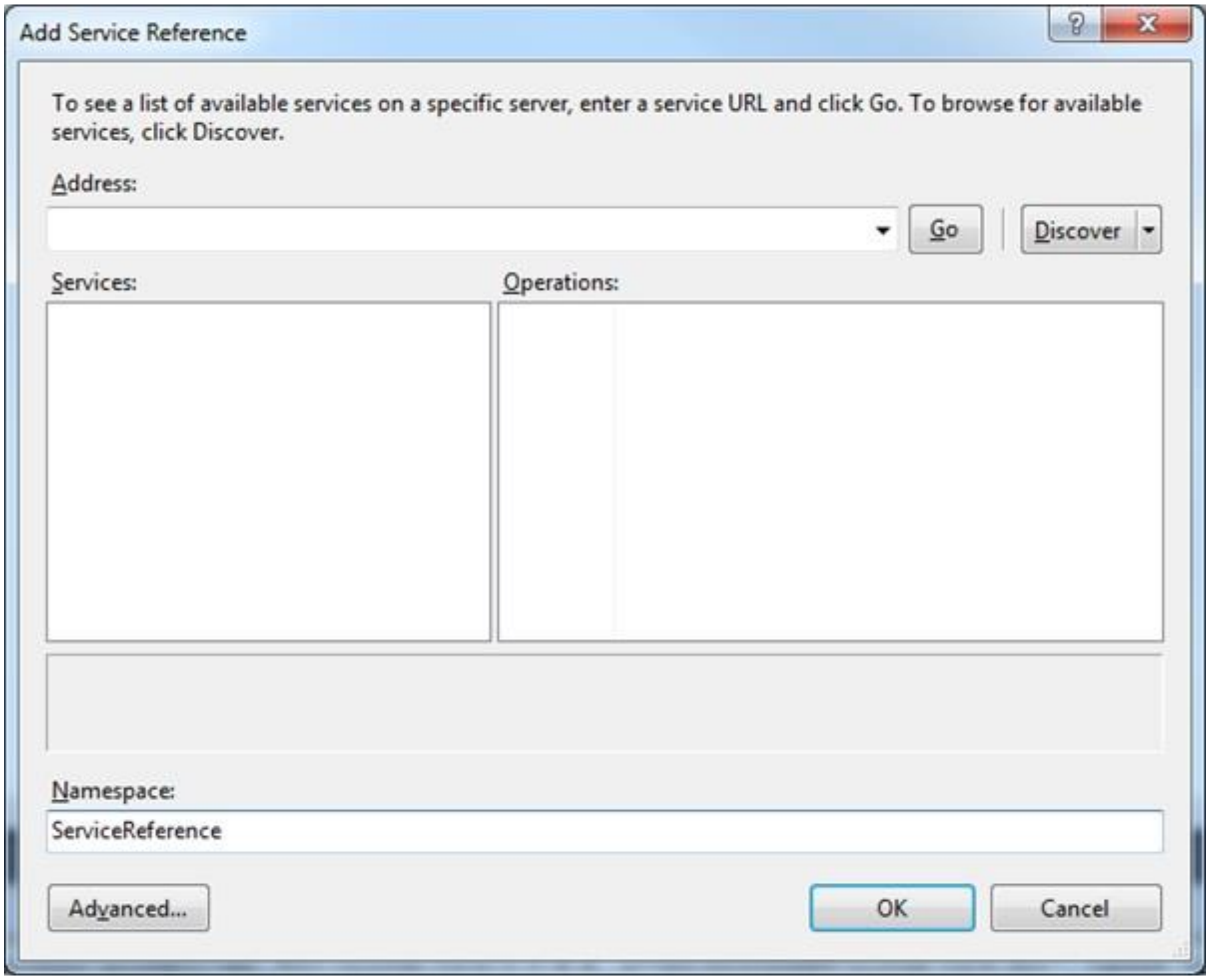
A project for creating a command-line application

Name: WebServiceConsumer

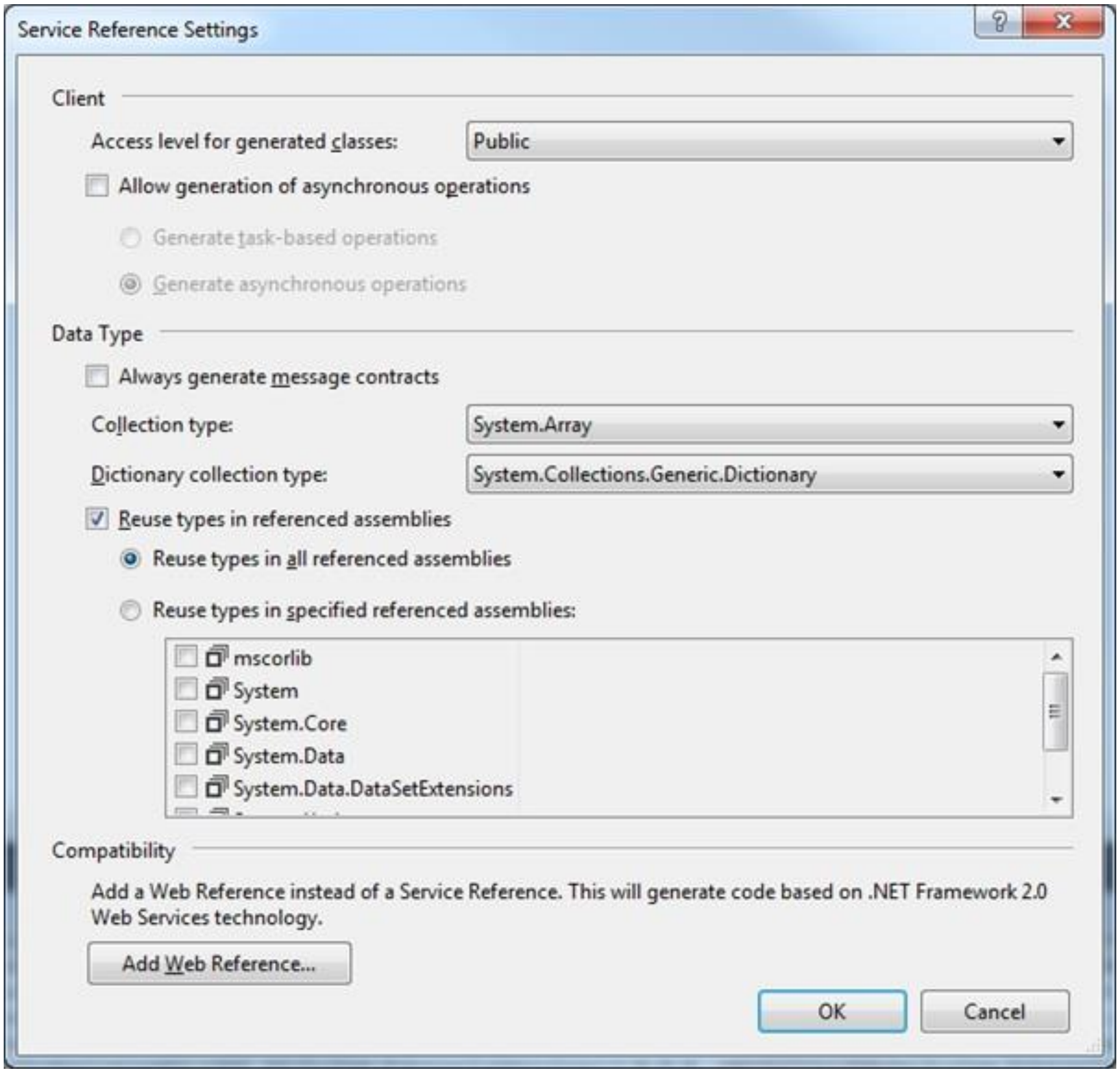
Location: E:\CodeSamples\WebService

Right click your project and select Add > Service Reference...

Rename the namespace if you wish:


Add Service Reference
To see a list of available services on a specific server, enter a service URL and click Go. To browse for available services, click Discover.
Address:
Go
Discover
Services:
Operations:
Namespace:
ServiceReference
Advanced...
OK
Cancel

Click on the Advanced button.



Service Reference Settings

Client

Access level for generated classes: Public

[ ] Allow generation of asynchronous operations

( ) Generate task-based operations

(●) Generate asynchronous operations

Data Type

[ ] Always generate message contracts

Collection type: System.Array

Dictionary collection type: System.Collections.Generic.Dictionary

[✓] Reuse types in referenced assemblies

(●) Reuse types in all referenced assemblies

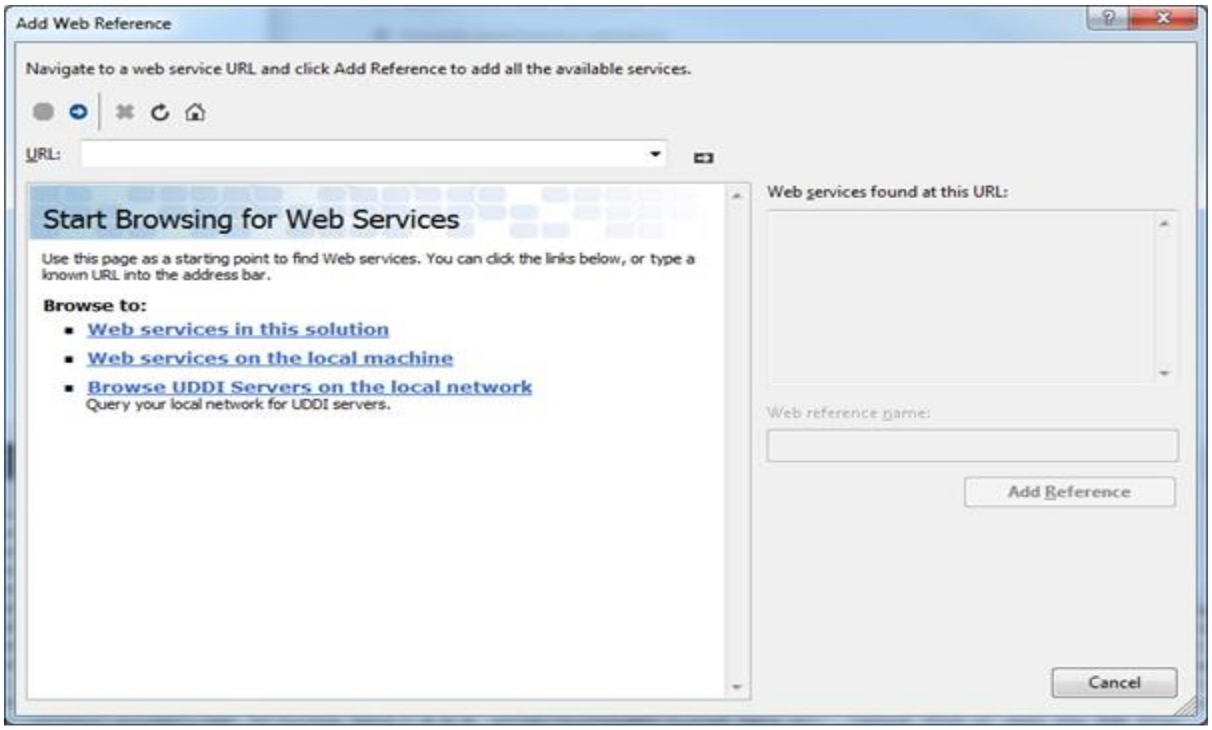
( ) Reuse types in specified referenced assemblies:

[ ] mscorlib
[ ] System
[ ] System.Core
[ ] System.Data
[ ] System.Data.DataSetExtensions

Compatibility

Add a Web Reference instead of a Service Reference. This will generate code based on .NET Framework 2.0 Web Services technology.

Click Add Web Reference


Add Web Reference

Navigate to a web service URL and click Add Reference to add all the available services.

URL:

Start Browsing for Web Services

Use this page as a starting point to find Web services. You can click the links below, or type a known URL into the address bar.

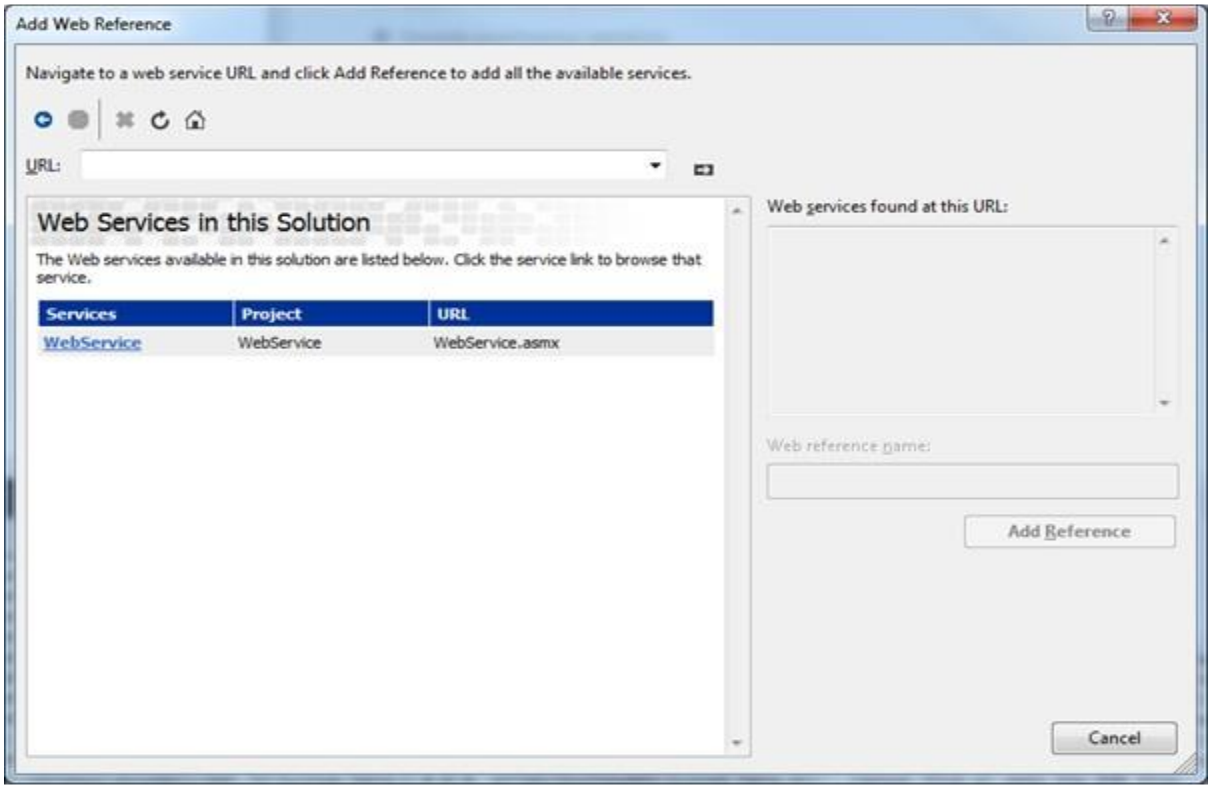
Browse to:

- Web services in this solution
- Web services on the local machine
- Browse UDDI Servers on the local network
  Query your local network for UDDI servers.

Web services found at this URL:

Web reference name:

Click the link that says to browse to services in this solution… Observe that it discovers the web service we earlier created for this solution:


Add Web Reference

Navigate to a web service URL and click Add Reference to add all the available services.

URL:

Web Services in this Solution

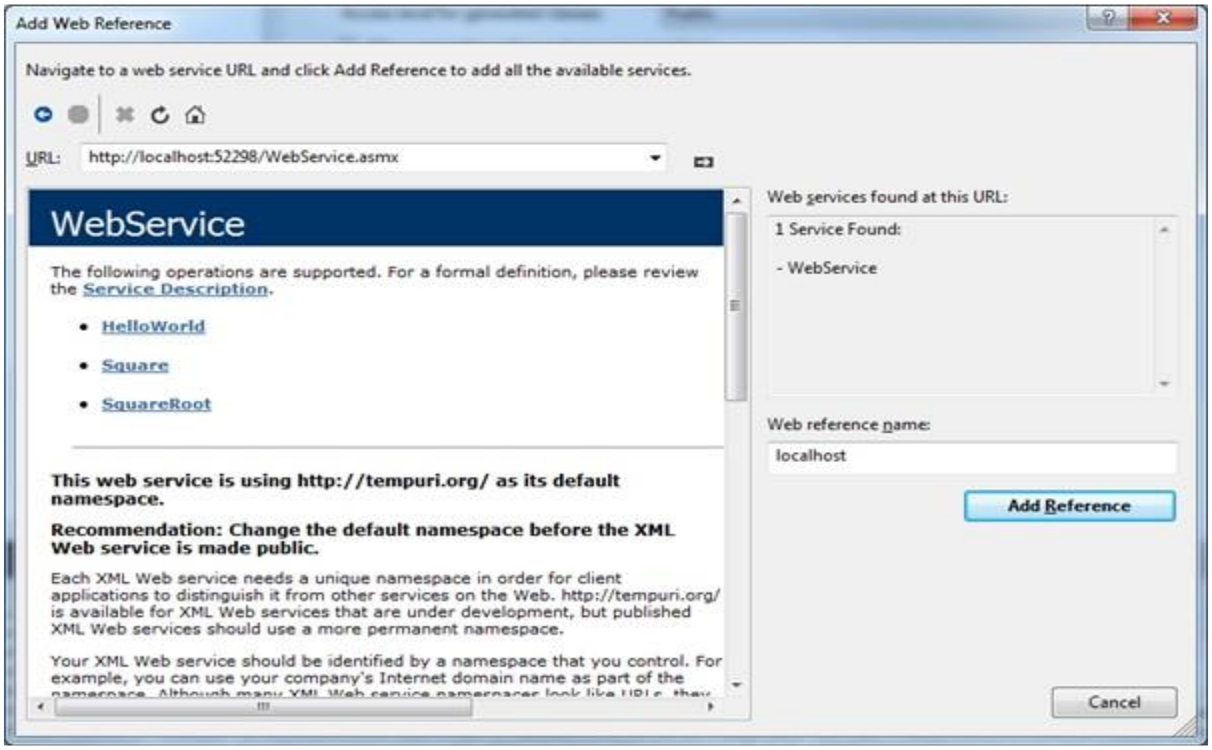
The Web services available in this solution are listed below. Click the service link to browse that service.

Services	Project	URL
WebService	WebService	WebService.asmx

Web services found at this URL:

Web reference name:

Click on this service and see that the range of service methods is displayed:


Add Web Reference

Navigate to a web service URL and click Add Reference to add all the available services.

URL: http://localhost:52298/WebService.asmx

WebService

The following operations are supported. For a formal definition, please review the Service Description.

- HelloWorld
- Square
- SquareRoot

This web service is using http://tempuri.org/ as its default namespace.

Recommendation: Change the default namespace before the XML Web service is made public.

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on the Web. http://tempuri.org/ is available for XML Web services that are under development, but published XML Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can use your company's Internet domain name as part of the

Web services found at this URL:

1 Service Found:

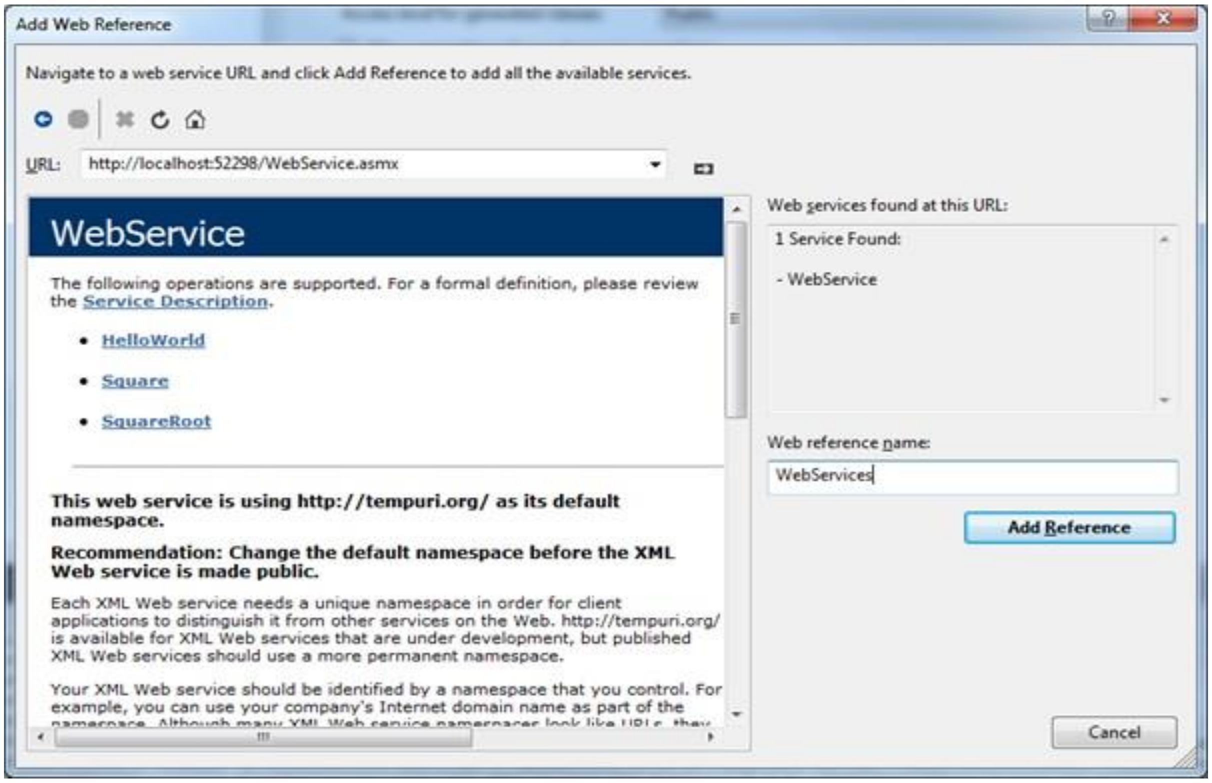
- WebService

Web reference name: localhost

Add Reference

Cancel

Rename the Web reference name to your preference:


Add Web Reference

Navigate to a web service URL and click Add Reference to add all the available services.

URL: http://localhost:52298/WebService.asmx

WebService

The following operations are supported. For a formal definition, please review the Service Description.

- HelloWorld
- Square
- SquareRoot

This web service is using http://tempuri.org/ as its default namespace.

Recommendation: Change the default namespace before the XML Web service is made public.

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on the Web. http://tempuri.org/ is available for XML Web services that are under development, but published XML Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can use your company's Internet domain name as part of the

Web services found at this URL:

1 Service Found:

- WebService

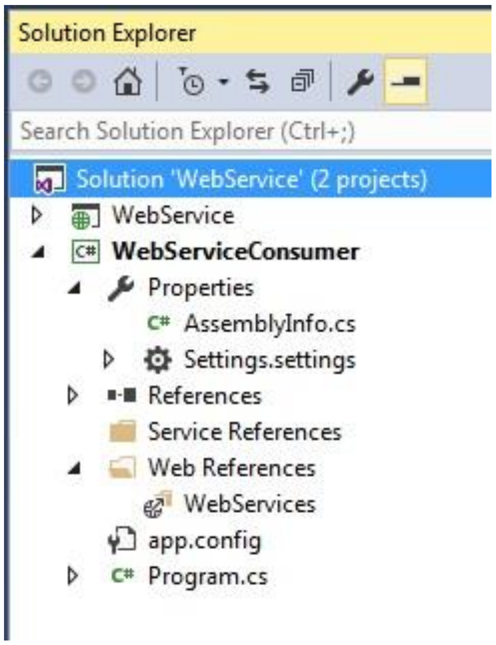
Web reference name: WebServices

Add Reference

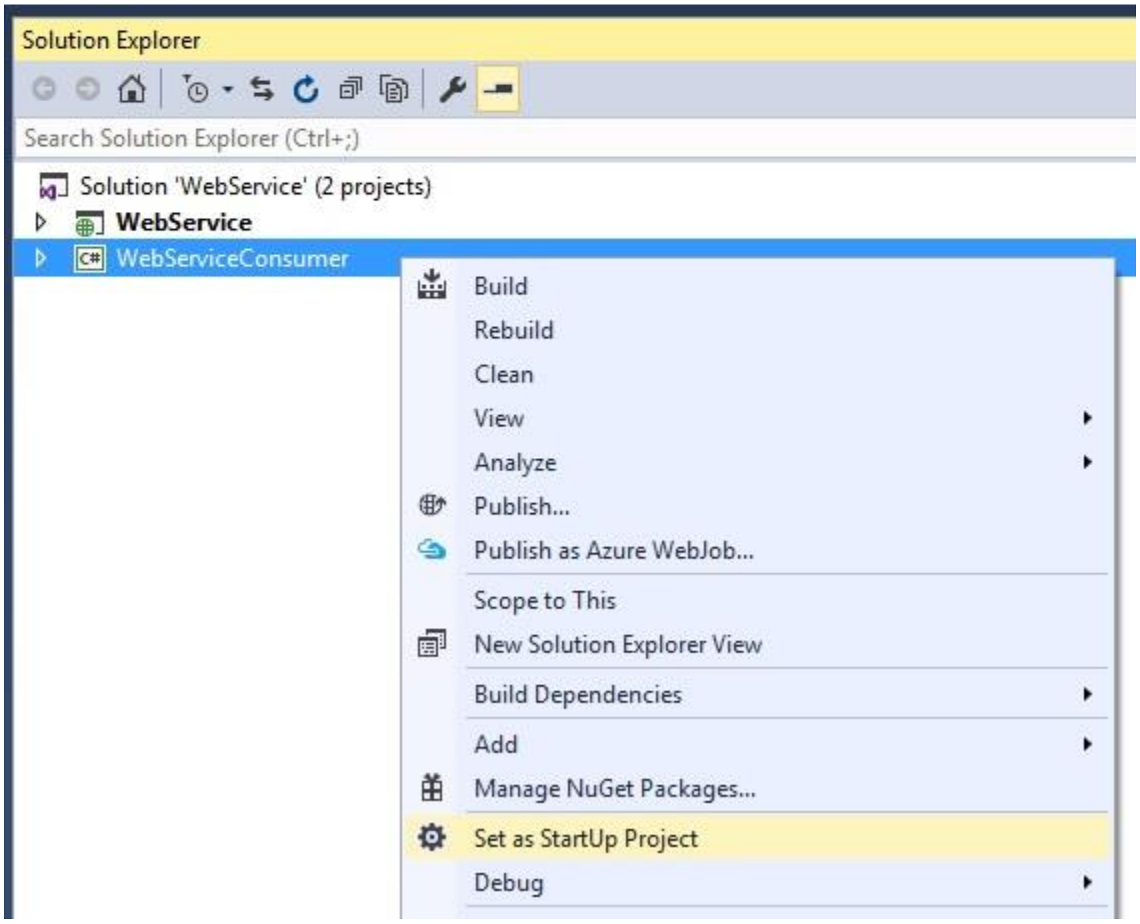
Cancel

Click Add Reference.

Observe the Web References folder containing WebServices is created as shown:



Set the console application as your startup project:



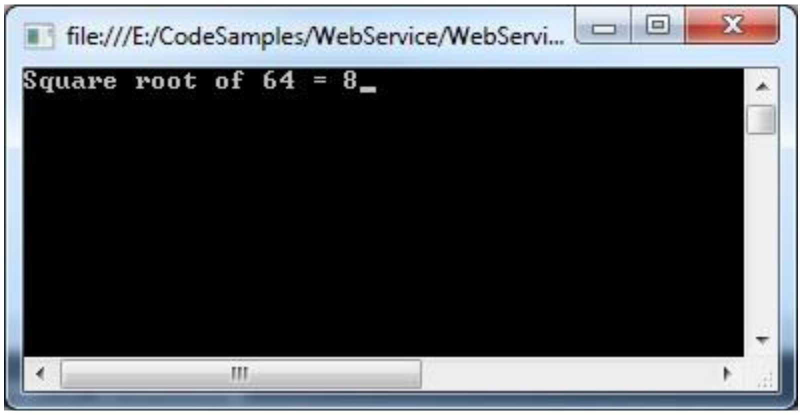
Update Program.cs in your console application to consume the referenced web service.

In this example to find and display the square root of 64:

Program.cs

```
using System;
using WebServiceConsumer.WebServices;

namespace WebServiceConsumer
{
    internal static class Program
    {
        private static void Main(string[] args)
        {
            var service = new WebService();

            Console.Write("Square root of 64 = {0}", service.SquareRoot(64));
        }
    }
}
```

Giving the following console output when run:



```
Square root of 64 = 8_
```

Courtesy: https://www.technical-recipes.com/2017/creating-and-consuming-a-web-service-in-c-net/

Modified: 2021.10.14.7.00.AM

Dököll Solutions, Inc.