Title: Creating Soap Service and consuming it in both php and C#

Author: [behzad](mailto:behzadbabaei69@gmail.com) babaei

Email: behzadbabaei69@gmail.com

Language: English

Platform: PHP,C#

Technology: php soap service,c# windows form

Level: Intermediate

Description: in this article I demonstrate how to create and build a soap service in php and how to consume it both php and c# platform

Section webservices,windows applications

SubSection php sevices,soap services,php client services,windows service consummer

License: CPOL

**Introduction**

SOAP defines the XML-based message format that Web service-enabled [applications](http://searchsoftwarequality.techtarget.com/definition/application) use to communicate and inter-operate with each other over the Web. The heterogeneous environment of the Web demands that applications support a common data encoding protocol and message format. SOAP is a standard for encoding messages in XML that invoke functions in other applications. It is platform and language independent.it was a brief explanation about soap services and how do they work in the real world.

What I intend to do in this article is showing you how to create and build a soap service I in php and how to create a WSDL file for functions within your service and also how to consume it in php and c#.

**Background**

Before starting to read this article I recommend that you get some info about Soap services, php services and also WSDL file which uses a xml schema format in to show the functionality of corresponded service.

It’s better you’ve installed a Xampp or any other type of web server package like easyphp or wamp server.Xampp is a free and open source cross-platform web server solution stack package, consisting mainly of the Apache HTTP Server, MySQL database, and interpreters for scripts written in the PHP and Perl programming languages

Here are some links that might help you to speed your work:

<http://php.net/manual/en/class.soapserver.php>

<http://php.net/manual/en/class.soapclient.php>

http://www.w3schools.com/schema/

**Using the code**

Here is our agenda in this article:

1.Creating a php class expose our functionality

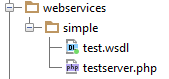
2.Create a soap server by previous class

3.Create a WSDL file for our service

4.Consumming the service in PHP

5. Consumming the service in C# WinForm Applciaction

1.Create a folder in your localhost root folder and name it webservices and create sub folder then name it simple.Then create a php file and name it testserver and also create a test file with **.wsd** **postfix.**



The content of test server file:

class TestClass

{

function hello($someone)

{

return "Hello " . $someone . " Welcome To Soap servers world!";

}

function SumData($a,$b)

{

return $a+$b;

}

function SubData($a,$b)

{

return $a - $b;

}

function MulData($a,$b)

{

return $a \* $b;

}

function DivData($a,$b)

{

return $a / $b;

}

}

As you can see we have declared a few function within our test class which these functions are those function who we decide to expose in web.

2. We instantiated a new soap server. The Soap server can take a few parameter and here we only set our WSDL file address and soap version that we are going to use.

Then we Set our class which respond to the client request. Remember that if you want to add functionality to your service you need to add that function in that class I mean TestClass in our case.

ini\_set("soap.wsdl\_cache\_enabled", "0");

$server = new SoapServer("http://localhost/webservices/simple/test.wsdl",array('soap\_version' => SOAP\_1\_2));

$server->setClass('TestClass'); //this line set our class incharge of responding to the client calls.

$server->handle();//this line handle the requests

Remember that all functions and instantiation of the SoapServer should be in the same page which the name of file is testserver.php

3.Creating WSDL file:

The content of WSDL file

//start of the code

<?xml version="1.0"?>

<wsdl:definitions name="MyDefinition"

targetNamespace="urn:myTns"

xmlns:tns="urn:myTns"

xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"

xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"

xmlns="http://schemas.xmlsoap.org/wsdl/" >

<wsdl:message name="sumincoming">

<wsdl:part name="sumnum1" type="xsd:int"/>

<wsdl:part name="sumnum2" type="xsd:int"/>

</wsdl:message>

<wsdl:message name="sumoutgoing">

<wsdl:part name="anssum" type="xsd:string"/>

</wsdl:message>

<wsdl:message name="subincoming">

<wsdl:part name="subnum1" type="xsd:int"/>

<wsdl:part name="subnum2" type="xsd:int"/>

</wsdl:message>

<wsdl:message name="suboutgoing">

<wsdl:part name="anssub" type="xsd:string"/>

</wsdl:message>

<wsdl:message name="mulincoming">

<wsdl:part name="mulnum1" type="xsd:int"/>

<wsdl:part name="mulnum2" type="xsd:int"/>

</wsdl:message>

<wsdl:message name="muloutgoing">

<wsdl:part name="ansmul" type="xsd:string"/>

</wsdl:message>

<wsdl:message name="divincoming">

<wsdl:part name="divnum1" type="xsd:int"/>

<wsdl:part name="divnum2" type="xsd:int"/>

</wsdl:message>

<wsdl:message name="divoutgoing">

<wsdl:part name="ansdiv" type="xsd:string"/>

</wsdl:message>

<wsdl:message name="helloincoming">

<wsdl:part name="name" type="xsd:string"/>

</wsdl:message>

<wsdl:message name="hellooutgoing">

<wsdl:part name="response" type="xsd:string"/>

</wsdl:message>

<wsdl:portType name="MyPortType">

<wsdl:operation name="SumData">

<wsdl:input message="tns:sumincoming"/>

<wsdl:output message="tns:sumoutgoing"/>

</wsdl:operation>

<wsdl:operation name="SubData">

<wsdl:input message="tns:subincoming"/>

<wsdl:output message="tns:suboutgoing"/>

</wsdl:operation>

<wsdl:operation name="MulData">

<wsdl:input message="tns:mulincoming"/>

<wsdl:output message="tns:muloutgoing"/>

</wsdl:operation>

<wsdl:operation name="DivData">

<wsdl:input message="tns:divincoming"/>

<wsdl:output message="tns:divoutgoing"/>

</wsdl:operation>

<wsdl:operation name="hello">

<wsdl:input message="tns:helloincoming"/>

<wsdl:output message="tns:hellooutgoing"/>

</wsdl:operation>

</wsdl:portType>

<wsdl:binding name="MyBinding" type="tns:MyPortType">

<soap12:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>

<wsdl:operation name="SumData">

<soap12:operation soapAction=""/>

<wsdl:input>

<soap12:body use="literal" namespace="urn:myInputNamespace" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />

</wsdl:input>

<wsdl:output>

<soap12:body use="literal" namespace="urn:myOutputNamespace" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />

</wsdl:output>

</wsdl:operation>

<wsdl:operation name="MulData">

<soap12:operation soapAction=""/>

<wsdl:input>

<soap12:body use="literal" namespace="urn:myInputNamespace" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />

</wsdl:input>

<wsdl:output>

<soap12:body use="literal" namespace="urn:myOutputNamespace" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />

</wsdl:output>

</wsdl:operation>

<wsdl:operation name="SubData">

<soap12:operation soapAction=""/>

<wsdl:input>

<soap12:body use="literal" namespace="urn:myInputNamespace" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />

</wsdl:input>

<wsdl:output>

<soap12:body use="literal" namespace="urn:myOutputNamespace" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />

</wsdl:output>

</wsdl:operation>

<wsdl:operation name="DivData">

<soap12:operation soapAction=""/>

<wsdl:input>

<soap12:body use="literal" namespace="urn:myInputNamespace" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />

</wsdl:input>

<wsdl:output>

<soap12:body use="literal" namespace="urn:myOutputNamespace" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />

</wsdl:output>

</wsdl:operation>

<wsdl:operation name="hello">

<soap12:operation soapAction=""/>

<wsdl:input>

<soap12:body use="literal" namespace="urn:myInputNamespace" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />

</wsdl:input>

<wsdl:output>

<soap12:body use="literal" namespace="urn:myOutputNamespace" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />

</wsdl:output>

</wsdl:operation>

</wsdl:binding>

<wsdl:service name="MyService">

<wsdl:port name="MyPort" binding="tns:MyBinding">

<soap12:address location="http://localhost/webservices/simple/testserver.php"/>

</wsdl:port>

</wsdl:service>

</wsdl:definitions>

///end of the code

The structure of the wsdl file is clear.Each service should consist a WSDL file so that any client could understand what functionality does that service expose.

In general WSDL file contain a some fixed node like below

<definitions>  
  
<types>  
  data type definitions........  
</types>  
  
<message>  
  definition of the data being communicated....  
</message>  
  
<portType>  
  set of operations......  
</portType>  
  
<binding>  
  protocol and data format specification....  
</binding>  
  
</definitions>

4.Consumming the service in PHP

Create one php page anywhere that you desire within your localhost

Create a folder in your localhost and php file within it.



The content of test.php

//start of the code

<?php

function TestAdd()

{

$num1=30;

$num2=24;

$client = new SoapClient("http://localhost/webservices/simple/test.wsdl",array('soap\_version' => SOAP\_1\_2,'trace' => 1 ));

$res=$client->\_\_soapCall( "SumData", array("sumnum1"=>$num1,"sumnum2"=>$num2));

$res1=$client->\_\_soapCall("MulData", array("mulnum1"=>$num1,"mulnum2"=>$num2));

$res2=$client->\_\_soapCall("DivData", array("divnum1"=>$num1,"divnum2"=>$num2));

$res3=$client->\_\_soapCall("SubData", array("subnum1"=>$num1,"subnum2"=>$num2));

echo "this is sum: ".$res."<br>";

echo "this is mul: ".$res1."<br>";

echo "this is div: ".$res2."<br>";

echo "this is sub: ".$res3."<br>";

}

TestAdd();

//

?>

//end of the code

For calling a php service function you need to instantiate sopaclient and pass your WSDL file address and the asoap version that service uses.

Like below:

$client = new SoapClient("http://localhost/webservices/simple/test.wsdl",array('soap\_version' => SOAP\_1\_2,'trace' => 1 ));

And for calling the function of the service you need pass the name of the function the parameter.

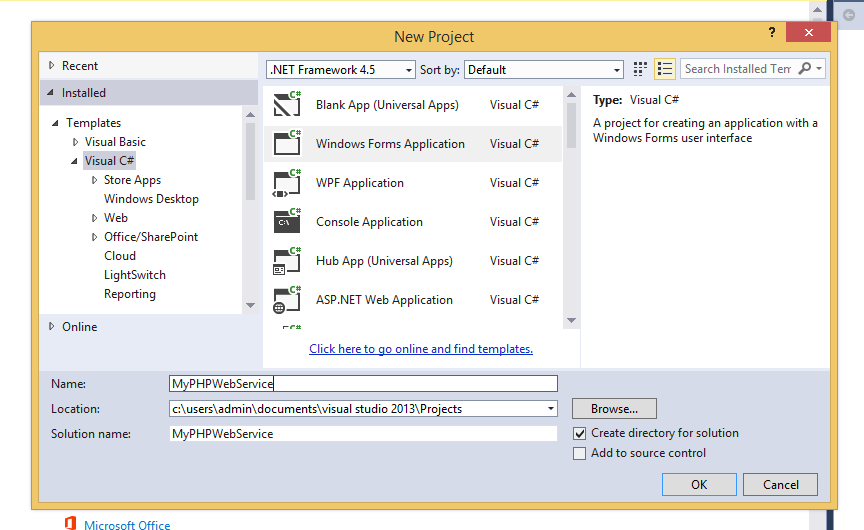
The server return result which I store it in $res variable.

$res=$client->\_\_soapCall( "SumData", array("sumnum1"=>$num1,"sumnum2"=>$num2));

Then I finally show the res by echoing it.

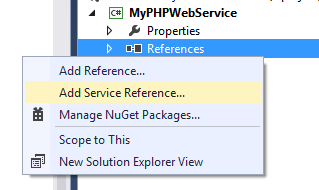
5. Consumming the service in C# WinForm Applciaction

Open up your visual studio and click File->New ->Project

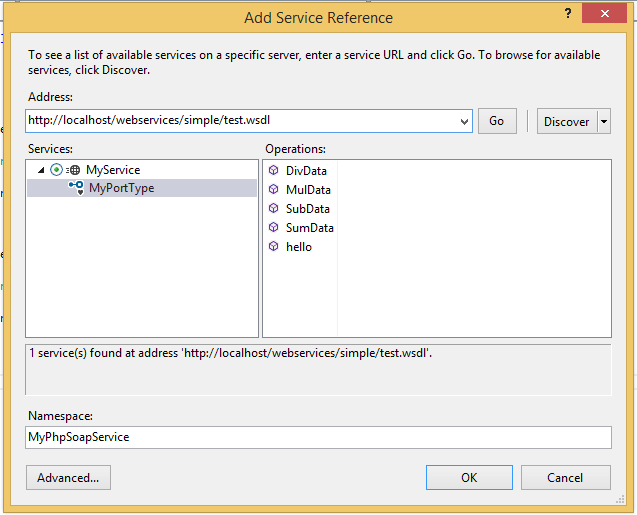


Select Windows Forms Application and name your project what ever you want but in my case I named it MyPHPWebService and click ok

Right click on your Refrences and click add Service Refrences like below

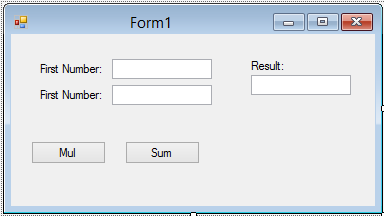


Then in the opend windows,in the address field paste your WSDL file address you can copy it from SoapServer or Client Instance the namespace field type the name of your service name(optional).



As you can see the visual studio has found our server and functions within that service and at the end click ok to submit the changes.

the create a UI for your application like below but you can create your own UI.



And at the end you need to call the service functions.Here are the Codes

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 MyPHPWebService

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void btnMul\_Click(object sender, EventArgs e)

{

//creating the instance of the sevice

MyPhpSoapServer.MyPortTypeClient client = new MyPhpSoapServer.MyPortTypeClient();

//calling the method in the server and get the anser back.

txtRes.Text = client.MulData(int.Parse(txtFirst.Text.Trim()), int.Parse(txtSecond.Text.Trim()));

}

private void btnSum\_Click(object sender, EventArgs e)

{

MyPhpSoapServer.MyPortTypeClient client = new MyPhpSoapServer.MyPortTypeClient();

txtRes.Text = client.SumData(int.Parse(txtFirst.Text.Trim()), int.Parse(txtSecond.Text.Trim()));

}

}}

**History**

The version 1.0 is simple type data and in the further article I explain complex type such as returning class or array.