**Министерство науки и высшего образования РФ**

Федеральное государственное автономное образовательное учреждение

высшего профессионального образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт – ИШИТР

Отделение – ИТ

Направление – ИСиТ

**Лабораторная работа № 8**

**SOAP сервисы**

по дисциплине: архитектура информационных систем

Выполнил: студент гр. 8И8А \_\_\_\_\_\_\_\_\_\_\_ Санжиев Д.Б

(Номер группы) (Подпись) (Ф.И.О.)

Проверил: доцент ОИТ \_\_\_\_\_\_\_\_\_\_\_ Цапко И.В

(Подпись) (Ф.И.О.)

Дата 19.11.2020

Содержание

C.

Введение 3

Цель работы 3

Результаты работы 4

Вывод 18

Введение

Цель работы

Научиться использовать и создавать SOAP сервисы

**Задание**:

1. Найти открытый SOAP сервис и написать использующее его приложение.

2. Создать SOAP сервис, согласно варианту, имитирующий работу выбранного ранее сервиса.

Результаты работы

**Часть 1. Создание приложения, на основе найденного SOAP сервиса.**

Для выполнения данного пункта был выбран и использован сервис: «http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL». Сервис выводит различную информацию о введенной стране.

Листинг программы:

namespace lb8

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

}

ServiceReference1.CountryInfoServiceSoapTypeClient asp = new ServiceReference1.CountryInfoServiceSoapTypeClient();

private void button1\_Click(object sender, EventArgs e)

{

textBox2.Text = asp.CountryName(textBox1.Text);

textBox5.Text = asp.CapitalCity(textBox1.Text);

}

private void button2\_Click(object sender, EventArgs e)

{

textBox4.Text = asp.CountryISOCode(textBox3.Text);

textBox6.Text = asp.LanguageName(textBox3.Text);

}

private void label3\_Click(object sender, EventArgs e)

{

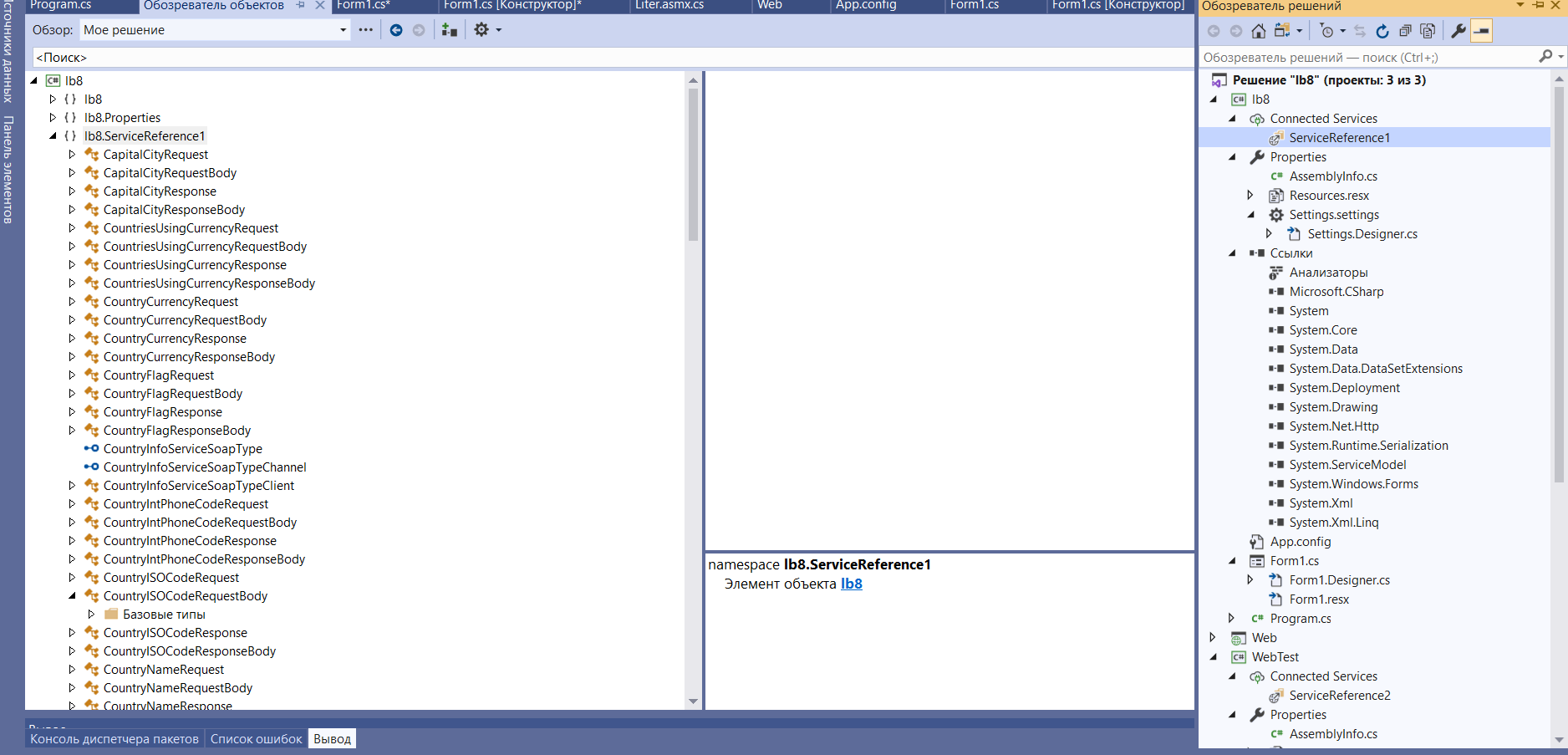
}

}

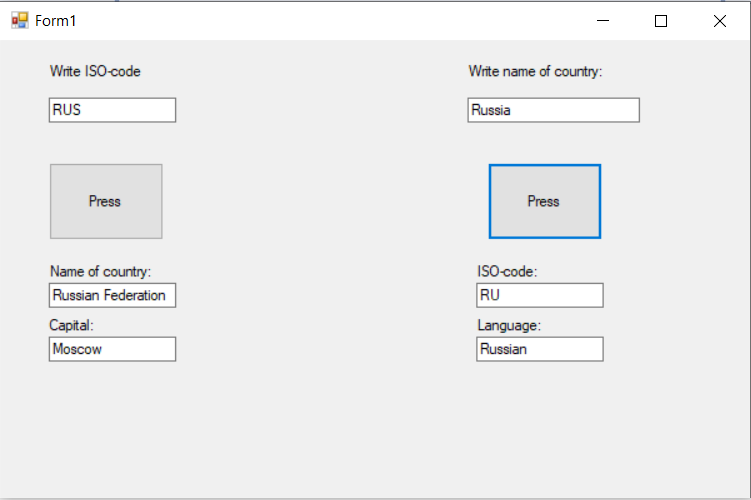
}

Результат работы программы:

Далее показаны результаты работы данной программы (рисунок 1-2).



*Рисунок 1 – Подключенный SOAP сервис*



*Рисунок 2 – Результат работы программы*

**Часть 2.** **Создать SOAP сервис, который конвертирует различные единицы измерения объема, и создать программу на данной SOAP сервисе.**

Листинг SOAP сервиса:

namespace Web

{

/// <summary>

/// Сводное описание для Liter

/// </summary>

[WebService(Namespace = "http://tempuri.org/")]

[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1\_1)]

[System.ComponentModel.ToolboxItem(false)]

// Чтобы разрешить вызывать веб-службу из скрипта с помощью ASP.NET AJAX, раскомментируйте следующую строку.

// [System.Web.Script.Services.ScriptService]

public class Liter : System.Web.Services.WebService

{

[WebMethod]

public string ChangeVolumeWeight(double value, VolumesWeights From, VolumesWeights To )

{

string res="";

switch(From)

{

case VolumesWeights.liter:

res = LiterToSMT(value, To);

break;

case VolumesWeights.deciliter:

res = DeciLiterToSMT(value, To);

break;

case VolumesWeights.centiliter:

res = CentiliterToSMT(value, To);

break;

case VolumesWeights.milliliter:

res = MilliliterToSMT(value, To);

break;

case VolumesWeights.cubicmeter:

res = CubicmeterToSMT(value, To);

break;

case VolumesWeights.cubicdecimeter:

res = CubicdeciToSMT(value, To);

break;

case VolumesWeights.cubiccentimeter:

res = CubiccentiToSMT(value, To);

break;

case VolumesWeights.cubicmillimeter:

res = CubicmilliToSMT(value, To);

break;

case VolumesWeights.hectoliter:

res = hectoliterToSMT(value, To);

break;

case VolumesWeights.decaliter:

res = decaliterToSMT(value, To);

break;

}

return res;

}

public string LiterToSMT(double value, VolumesWeights To)

{

double Res = 0.0;

switch (To)

{

case VolumesWeights.cubicmeter:

Res = value \* 0.001;

break;

case VolumesWeights.cubicdecimeter:

Res = value\*1;

break;

case VolumesWeights.cubiccentimeter:

Res = value \* 1000;

break;

case VolumesWeights.cubicmillimeter:

Res = value \* 1000000;

break;

case VolumesWeights.hectoliter:

Res = value \* 0.01;

break;

case VolumesWeights.decaliter:

Res = value \* 0.1;

break;

case VolumesWeights.liter:

Res = value ;

break;

case VolumesWeights.deciliter:

Res = value \* 10;

break;

case VolumesWeights.centiliter:

Res = value \* 100;

break;

case VolumesWeights.milliliter:

Res = value \* 1000;

break;

}

return Convert.ToString(Res);

}

public string DeciLiterToSMT(double value, VolumesWeights To)

{

double Res = 0.0;

switch (To)

{

case VolumesWeights.cubicmeter:

Res = value \* 0.0001;

break;

case VolumesWeights.cubicdecimeter:

Res = value \* 0.1;

break;

case VolumesWeights.cubiccentimeter:

Res = value \* 100;

break;

case VolumesWeights.cubicmillimeter:

Res = value \* 100000;

break;

case VolumesWeights.hectoliter:

Res = value \* 0.001;

break;

case VolumesWeights.decaliter:

Res = value \* 0.01;

break;

case VolumesWeights.liter:

Res = value\*0.1;

break;

case VolumesWeights.deciliter:

Res = value \* 1;

break;

case VolumesWeights.centiliter:

Res = value \* 10;

break;

case VolumesWeights.milliliter:

Res = value \* 100;

break;

}

return Convert.ToString(Res);

}

public string CentiliterToSMT(double value, VolumesWeights To)

{

double Res = 0.0;

switch (To)

{

case VolumesWeights.cubicmeter:

Res = value \* 0.00001;

break;

case VolumesWeights.cubicdecimeter:

Res = value \* 0.01;

break;

case VolumesWeights.cubiccentimeter:

Res = value \* 10;

break;

case VolumesWeights.cubicmillimeter:

Res = value \* 10000;

break;

case VolumesWeights.hectoliter:

Res = value \* 0.0001;

break;

case VolumesWeights.decaliter:

Res = value \* 0.001;

break;

case VolumesWeights.liter:

Res = value \* 0.01;

break;

case VolumesWeights.deciliter:

Res = value \* 0.1;

break;

case VolumesWeights.centiliter:

Res = value \* 1;

break;

case VolumesWeights.milliliter:

Res = value \* 10;

break;

}

return Convert.ToString(Res);

}

public string MilliliterToSMT(double value, VolumesWeights To)

{

double Res = 0.0;

switch (To)

{

case VolumesWeights.cubicmeter:

Res = value \* 0.000001;

break;

case VolumesWeights.cubicdecimeter:

Res = value \* 0.001;

break;

case VolumesWeights.cubiccentimeter:

Res = value \* 1;

break;

case VolumesWeights.cubicmillimeter:

Res = value \* 1000;

break;

case VolumesWeights.hectoliter:

Res = value \* 0.00001;

break;

case VolumesWeights.decaliter:

Res = value \* 0.0001;

break;

case VolumesWeights.liter:

Res = value \* 0.001;

break;

case VolumesWeights.deciliter:

Res = value \* 0.01;

break;

case VolumesWeights.centiliter:

Res = value \* 0.1;

break;

case VolumesWeights.milliliter:

Res = value \* 1;

break;

}

return Convert.ToString(Res);

}

public string CubicmeterToSMT(double value, VolumesWeights To)

{

double Res = 0.0;

switch (To)

{

case VolumesWeights.cubicmeter:

Res = value \* 1;

break;

case VolumesWeights.cubicdecimeter:

Res = value \* 1000;

break;

case VolumesWeights.cubiccentimeter:

Res = value \* 1000000;

break;

case VolumesWeights.cubicmillimeter:

Res = value \* 1000000000;

break;

case VolumesWeights.hectoliter:

Res = value \* 10;

break;

case VolumesWeights.decaliter:

Res = value \* 100;

break;

case VolumesWeights.liter:

Res = value \* 1000;

break;

case VolumesWeights.deciliter:

Res = value \* 10000;

break;

case VolumesWeights.centiliter:

Res = value \* 100000;

break;

case VolumesWeights.milliliter:

Res = value \* 1000000;

break;

}

return Convert.ToString(Res);

}

public string CubicdeciToSMT(double value, VolumesWeights To)

{

double Res = 0.0;

switch (To)

{

case VolumesWeights.cubicmeter:

Res = value \* 0.001;

break;

case VolumesWeights.cubicdecimeter:

Res = value \* 1;

break;

case VolumesWeights.cubiccentimeter:

Res = value \* 1000;

break;

case VolumesWeights.cubicmillimeter:

Res = value \* 1000000;

break;

case VolumesWeights.hectoliter:

Res = value \* 0.01;

break;

case VolumesWeights.decaliter:

Res = value \* 0.1;

break;

case VolumesWeights.liter:

Res = value \* 1;

break;

case VolumesWeights.deciliter:

Res = value \* 10;

break;

case VolumesWeights.centiliter:

Res = value \* 100;

break;

case VolumesWeights.milliliter:

Res = value \* 1000;

break;

}

return Convert.ToString(Res);

}

public string CubiccentiToSMT(double value, VolumesWeights To)

{

double Res = 0.0;

switch (To)

{

case VolumesWeights.cubicmeter:

Res = value \* 0.000001;

break;

case VolumesWeights.cubicdecimeter:

Res = value \* 0.001;

break;

case VolumesWeights.cubiccentimeter:

Res = value \* 1;

break;

case VolumesWeights.cubicmillimeter:

Res = value \* 1000;

break;

case VolumesWeights.hectoliter:

Res = value \* 0.00001;

break;

case VolumesWeights.decaliter:

Res = value \* 0.0001;

break;

case VolumesWeights.liter:

Res = value \* 0.001;

break;

case VolumesWeights.deciliter:

Res = value \* 0.01;

break;

case VolumesWeights.centiliter:

Res = value \* 0.1;

break;

case VolumesWeights.milliliter:

Res = value \* 1;

break;

}

return Convert.ToString(Res);

}

public string CubicmilliToSMT(double value, VolumesWeights To)

{

double Res = 0.0;

switch (To)

{

case VolumesWeights.cubicmeter:

Res = value \* 0.000000001;

break;

case VolumesWeights.cubicdecimeter:

Res = value \* 0.000001;

break;

case VolumesWeights.cubiccentimeter:

Res = value \* 0.001;

break;

case VolumesWeights.cubicmillimeter:

Res = value \* 1;

break;

case VolumesWeights.hectoliter:

Res = value \* 0.00000001;

break;

case VolumesWeights.decaliter:

Res = value \* 0.0000001;

break;

case VolumesWeights.liter:

Res = value \* 0.000001;

break;

case VolumesWeights.deciliter:

Res = value \* 0.00001;

break;

case VolumesWeights.centiliter:

Res = value \* 0.0001;

break;

case VolumesWeights.milliliter:

Res = value \* 0.001;

break;

}

return Convert.ToString(Res);

}

public string hectoliterToSMT(double value, VolumesWeights To)

{

double Res = 0.0;

switch (To)

{

case VolumesWeights.cubicmeter:

Res = value \* 0.1;

break;

case VolumesWeights.cubicdecimeter:

Res = value \* 100;

break;

case VolumesWeights.cubiccentimeter:

Res = value \* 100000;

break;

case VolumesWeights.cubicmillimeter:

Res = value \* 100000000;

break;

case VolumesWeights.hectoliter:

Res = value \* 1;

break;

case VolumesWeights.decaliter:

Res = value \* 10;

break;

case VolumesWeights.liter:

Res = value \* 100;

break;

case VolumesWeights.deciliter:

Res = value \* 1000;

break;

case VolumesWeights.centiliter:

Res = value \* 10000;

break;

case VolumesWeights.milliliter:

Res = value \* 100000;

break;

}

return Convert.ToString(Res);

}

public string decaliterToSMT(double value, VolumesWeights To)

{

double Res = 0.0;

switch (To)

{

case VolumesWeights.cubicmeter:

Res = value \* 0.01;

break;

case VolumesWeights.cubicdecimeter:

Res = value \* 10;

break;

case VolumesWeights.cubiccentimeter:

Res = value \* 10000;

break;

case VolumesWeights.cubicmillimeter:

Res = value \* 10000000;

break;

case VolumesWeights.hectoliter:

Res = value \* 0.1;

break;

case VolumesWeights.decaliter:

Res = value \* 1;

break;

case VolumesWeights.liter:

Res = value \* 10;

break;

case VolumesWeights.deciliter:

Res = value \* 100;

break;

case VolumesWeights.centiliter:

Res = value \* 1000;

break;

case VolumesWeights.milliliter:

Res = value \* 10000;

break;

}

return Convert.ToString(Res);

}

public enum VolumesWeights

{

cubicmeter,

cubicdecimeter,

cubiccentimeter,

cubicmillimeter,

hectoliter,

decaliter,

liter,

deciliter,

centiliter,

milliliter,

}

}

}

Листинг приложения:

namespace WebTest

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

comboBox1.DataSource = Enum.GetValues(typeof(ServiceReference2.VolumesWeights));

comboBox1.SelectedItem = ServiceReference2.VolumesWeights.liter;

comboBox2.DataSource = Enum.GetValues(typeof(ServiceReference2.VolumesWeights));

comboBox2.SelectedItem = ServiceReference2.VolumesWeights.liter;

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

double value = Convert.ToDouble(textBox1.Text);

ServiceReference2.VolumesWeights from = (ServiceReference2.VolumesWeights)comboBox1.SelectedItem;

ServiceReference2.VolumesWeights to = (ServiceReference2.VolumesWeights)comboBox2.SelectedItem;

string res = soapClient.ChangeVolumeWeight(value, from, to);

textBox2.Text = res;

}

catch

{

textBox2.Text = "Incorrect introduced value";

}

}

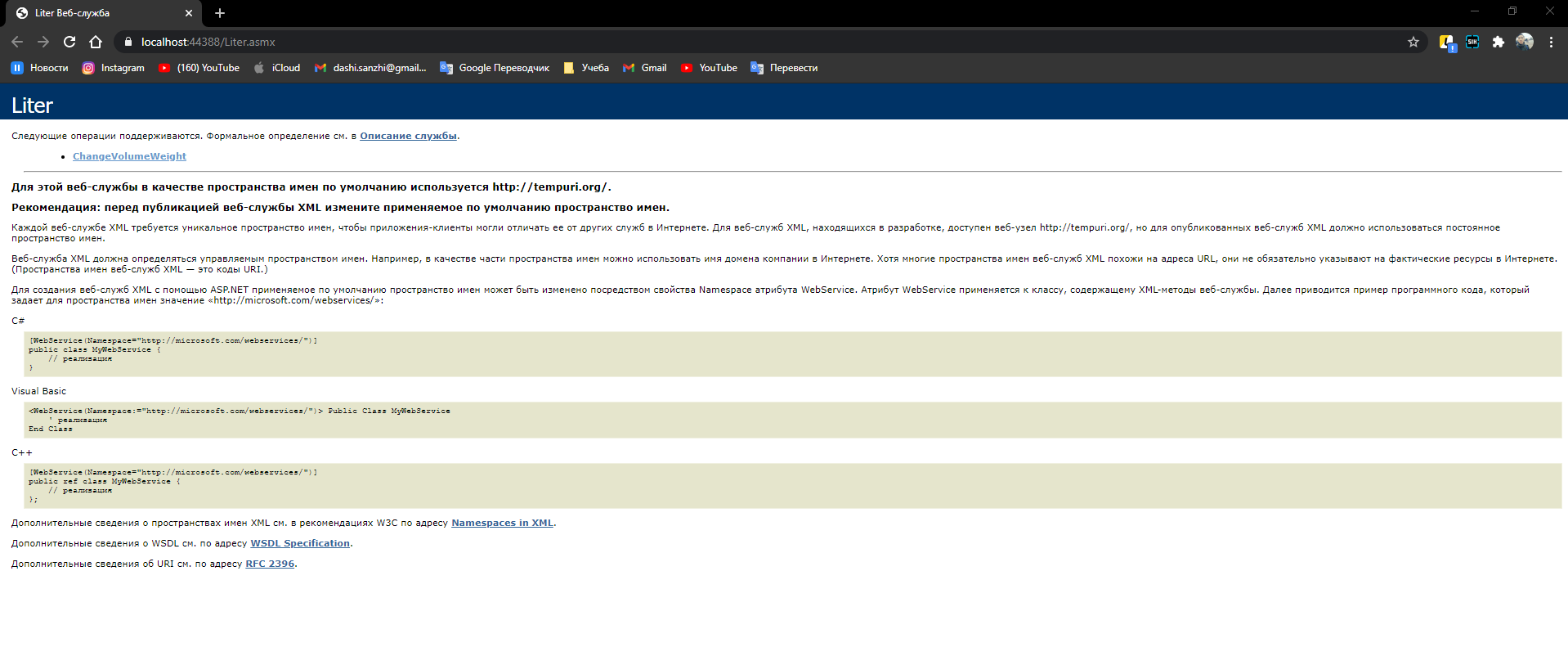
public ServiceReference2.LiterSoapClient soapClient = new ServiceReference2.LiterSoapClient();

}

}

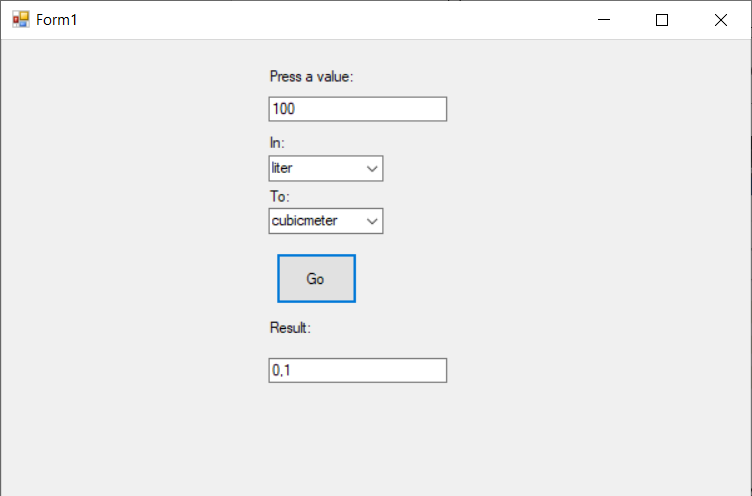
Результаты работы программы:

В ходе запуска программы был выведена следующая страница с данным SOAP сервисом (рисунок 3).



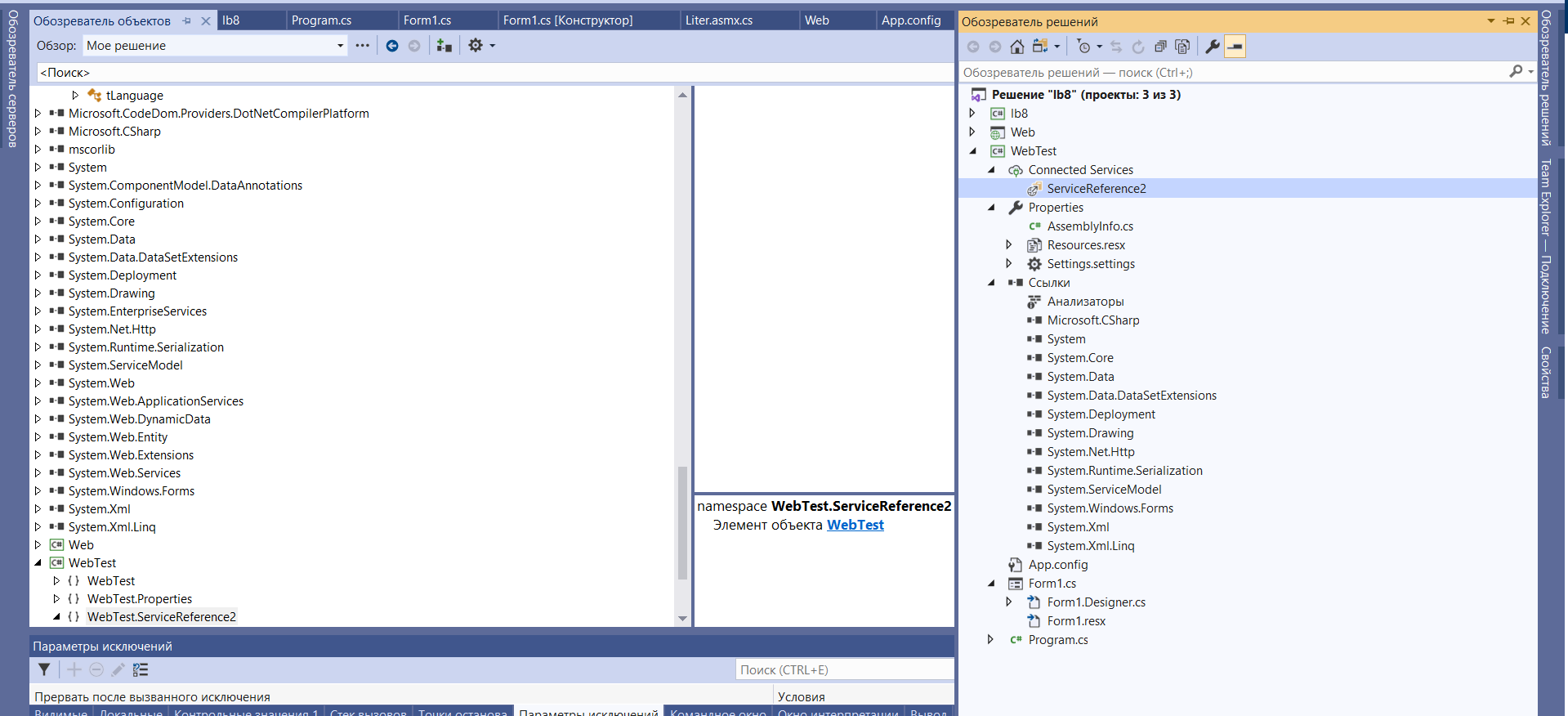
*Рисунок 3 – Подключенный SOAP сервис*

Затем выведена следующая форма приложения в которой используется данный SOAP сервис (рисунок 4).



*Рисунок 4 – Приложение, использующее созданный SOAP сервис*

В качестве подтверждения на рисунке 5 выведено подключенное SOAP приложение.



*Рисунок 5 – Подключенный SOAP сервис*

Вывод

В данной лабораторной работе были изучены основные навыки работы с SOAP сервисами. В ходе этого задания был найден и использован в приложении SOAP сервис, который выводит информацию о введенной стране, а также был создан свой SOAP сервис, который конвертирует различные единицы измерения объема, и создано приложение, реализующее созданный ранее SOAP сервис.