Урок 8

[Обзор сервис-ориентированной архитектуры приложений (SOA). Web-сервис asmx. WCF-сервис. Web-api сервис. 1](#_Toc487644868)

[1. Обзор сервис-ориентированной архитектуры приложений (SOA) 1](#_Toc487644869)

[2. Web-сервис 3](#_Toc487644870)

[2.1. Создание веб-сервиса 4](#_Toc487644871)

[2.2. Создание WPF-приложения, потребителя сервиса 8](#_Toc487644872)

[3. WCF-сервис 10](#_Toc487644873)

[3.1. Создание WCF -сервиса 11](#_Toc487644874)

[3.2. Создание WPF-приложения потребителя WCF-сервиса 13](#_Toc487644875)

[4. Web-api сервис 13](#_Toc487644876)

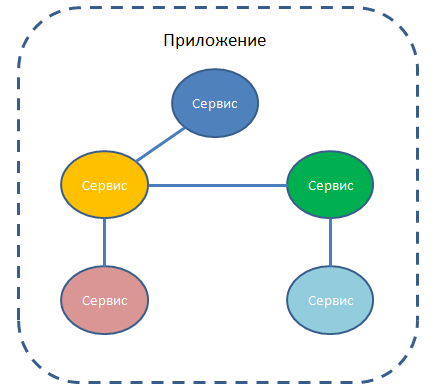
[5. ДЗ 13](#_Toc487644877)

# Обзор сервис-ориентированной архитектуры приложений (SOA). Web-сервис asmx. WCF-сервис. Web-api сервис.

1. Обзор сервис-ориентированной архитектуры приложений (SOA)

<http://masters.donntu.org/2014/fknt/belyy/ind/index.htm>

Сервис – это функциональная единица доступная внешнему миру с помощью стандартного механизма обмена сообщениями. Сервис-ориентированное приложение представляет собой агрегацию сервисов в логически целостное приложение, также как объектно-ориентированное приложение является агрегацией отдельных объектов.



Приложение, также, может выступать как еще один сервис, аналогично тому, как крупный объект может быть композицией более мелких объектов.

Внутри сервиса, разработчики могут по-прежнему использовать специфические языки программирования, технологии и фреймворки. Однако, между сервисами сохраняются стандартные протоколы и сообщения, контракты и обмен метаданными.

Сервисы, входящие в состав отдельного приложения, могут располагаться в одном и том же месте или могут быть распределены по разным узлам локальной сети или Интернет. Отдельные сервисы могут разрабатываться разными компаниями, размещаться на разных технологических платформах и находиться в разных часовых поясах. Все эти аспекты скрыты от клиентских приложений, взаимодействующих с сервисами. Клиентские приложения и сервисы могут взаимодействовать друг с другом путем отправки и получения сообщений. Механизм обмена сообщениями между клиентскими приложениями и сервисами сводит на нет разницу между клиентскими приложениями и сервисами, преобразуя входящие и исходящие сообщения к стандартным протоколам сетевого обмена.

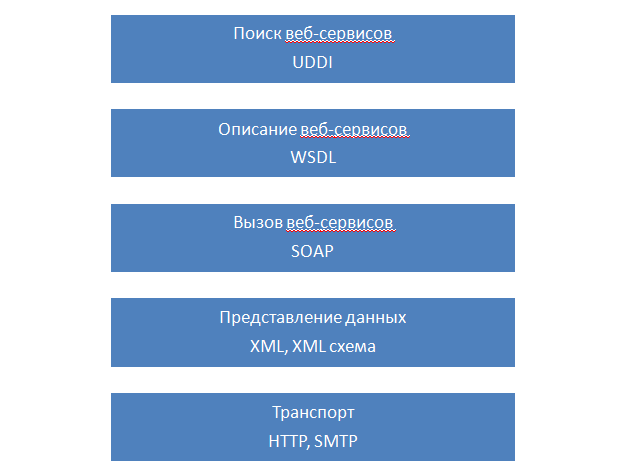
Принципы сервис-ориентированной архитектуры:

* Сервисы должны иметь четкие границы. Любой сервис ограничен возможностями технологии, с помощью которой он реализован, и своим расположением. Эти ограничения не должны проявляться в контрактах сервисов или типах данных, используемых для обмена.
* Сервисы автономны. Сервис, для своей работы, не должен нуждаться в своих клиентах или других сервисах. Он должен функционировать и обновляться независимо от клиентских приложений. Также, сервис должен иметь свой механизм авторизации, не зависящий от уровня доступа пользователя клиентского приложения.
* Сервис предоставляет только контракты взаимодействия с ним и схему данных. Детали реализации и работы веб-сервиса должны быть недоступны извне.
* Совместимость сервисов основана на правилах(policy). Сервисы должны публиковать правила, описывающие, что они могут делать и как клиентские приложения взаимодействуют с ними.

Наиболее распространенные технологии реализации веб-сервисов:

* SOAP (Simple Object Access Protocol). По сути это три стандарта: SOAP/WSDL/UDDI
* REST (Representational State Transfer)

Стандарты, используемые для построения веб-сервиса



* HTTP, Hypertext Transfer Protocol. HTTP – это протокол, реализующий принцип запрос/ответ.Обычно реализуется поверх TCP/IP протокола. HTTP клиент открывет TCP подключение к серверу и отправляет HTTP запрос. Запрос содержит метод GET или POST. GET используется для получения с сервера файла, например HTML страницы. Для передачи данных используется метод POST. Ответ на HTTP запрос содержит возвращаемые данные.
* XML и XML схема. XML - ключевая технология веб сервисов. SOAP использует XML как формат представления данных. WSDL использует XML схему для описания структуры веб сервисов. UDDI использует XML схему для описания структуры репозитария веб сервисов и для взаимодействия с репозитарием.
* SOAP. Протокол для обмена информацией в децентрализованной, распределенной среде.
* WSDL, Web Service Description Language используется для описания интерфейса веб-сервиса.
* UDDI, Universal Description, Discovery, and Integration. Глобальный репозитарий веб сервисов и технология взаимодействия с репозитарием.

1. Web-сервис

Характерные особенности:

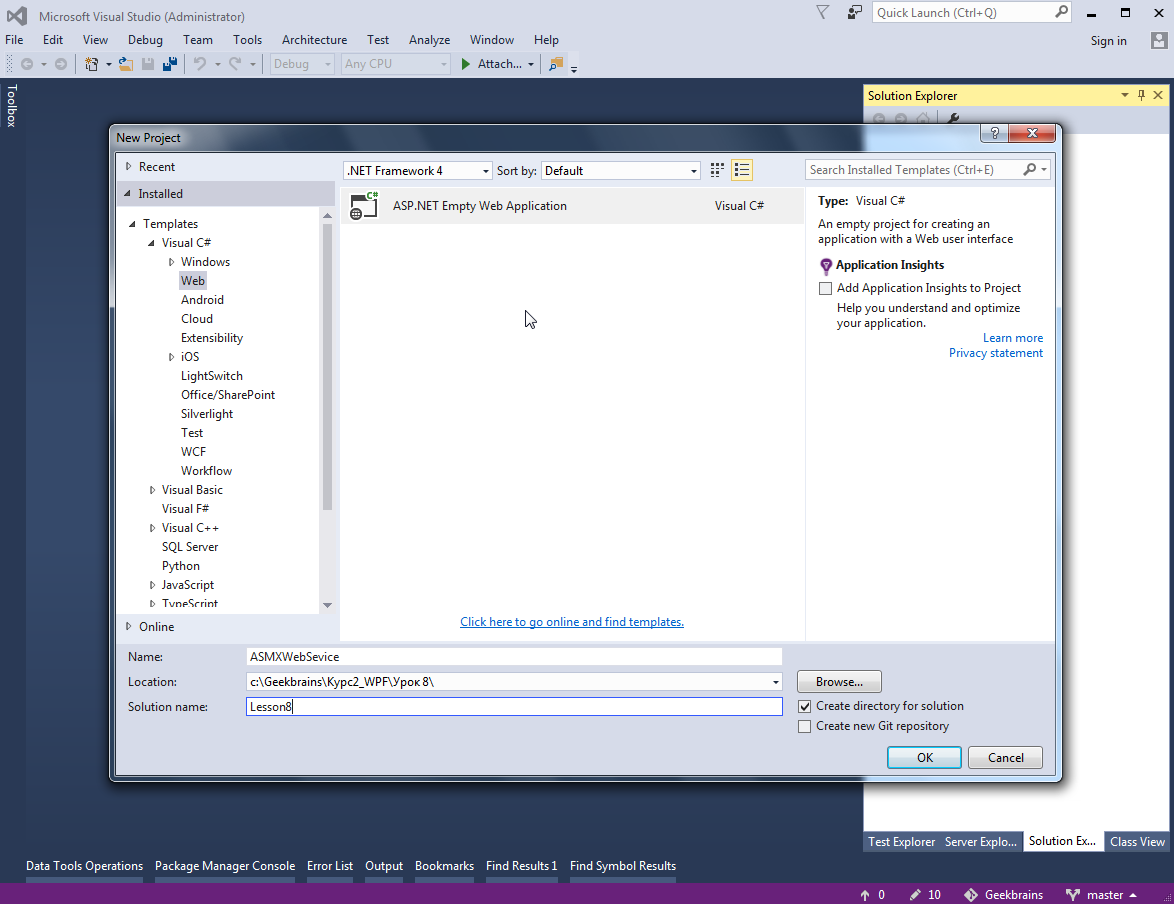
* Основой веб-сервисов является протокол SOAP.
* Формат сериализации данных для веб-сервисов основывается на спецификации XML. Данные, между клиентским приложением и сервисом передаются в XML формате.
* В качестве транспортного протокола используется только HTTP.
* Не является open source технологией, но с сервисом могут работать клиентские приложения, понимающие xml.
* Может быть развернут только на веб сервере Microsoft, IIS(Internet Information Services).
* Для описания веб-сервисов используется WSDL, Web Service Description Language.

Схема использования веб-сериса клиентским приложением:

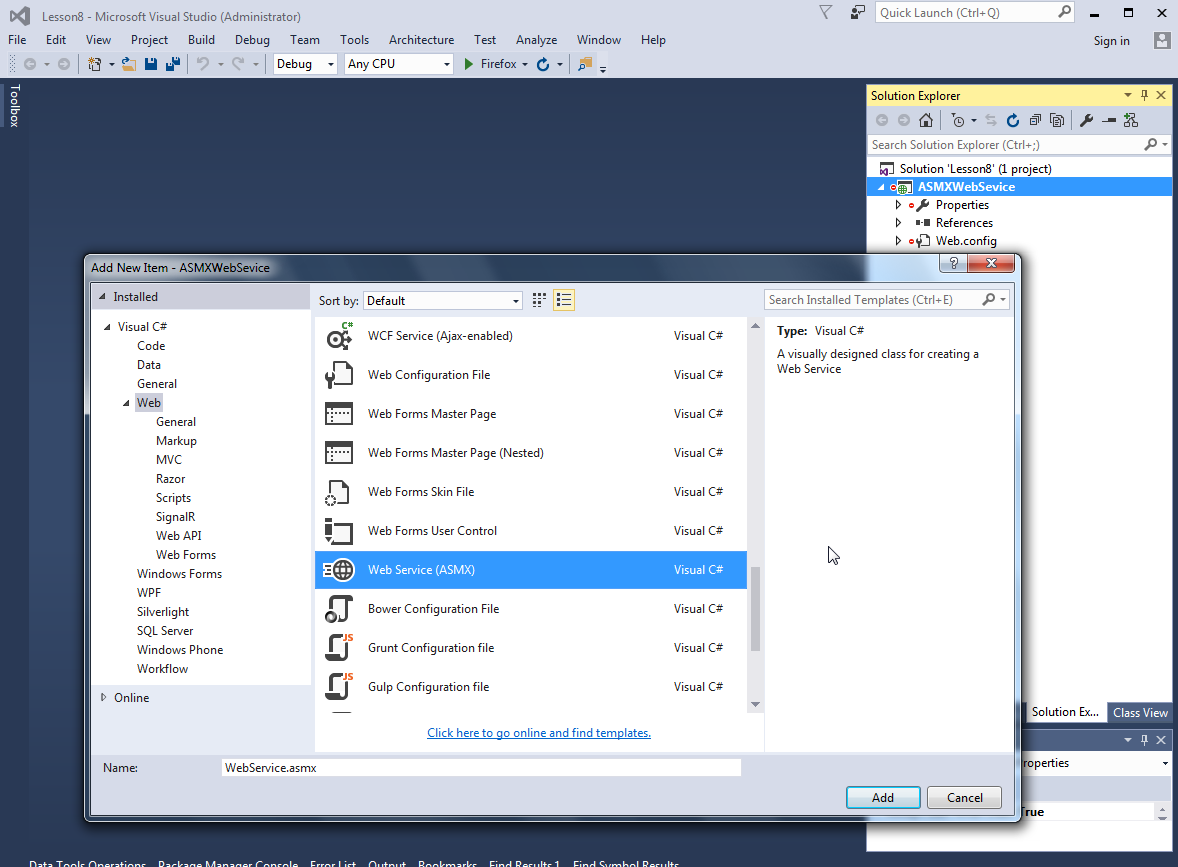


* 1. Создание веб-сервиса

Клиентское приложение WPF выполняет сложение двух чисел, используя веб-сервис. Клиентское приложение и веб-сервис развернуты на одном компьютере.

Создадим веб-сервис, выполняющий сложение двух чисел. Для этого создадим пустой (Empty) ASP.Net-проект.   


После этого добавьте в него проект (Add New Item) Web-service ASMX.

  
Visual Studio cформирует следующий код веб-сервиса:

WebService.asmx.cs

using System.Web.Services;

namespace ASMXWebSevice

{

/// <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, uncomment the following line.

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

public class WebService : System.Web.Services.WebService

{

[WebMethod]

public string HelloWorld()

{

return "Hello World";

}

}

}

В коде веб-сервиса уже реализован метод «HelloWorld». Дополним код веб-сервиса методом для расчета суммы чисел.

WebService.asmx.cs

using System.Web.Services;

namespace ASMXWebSevice

{

/// <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, uncomment the following line.

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

public class WebService : System.Web.Services.WebService

{

[WebMethod]

public string HelloWorld()

{

return "Hello World";

}

[WebMethod]

public string Summ(string number1, string number2)

{

// Входные параметры объявляем типа String, чтобы принимать от пользователя любые символы,

// их анализировать, и при "плохом вводе" сообщать по-русски.

float a, b;

bool flag = float.TryParse(number1, System.Globalization.NumberStyles.Number,

System.Globalization.NumberFormatInfo.CurrentInfo, out a);

if (flag == false) return "В первом поле должно быть число";

flag = float.TryParse(number2, System.Globalization.NumberStyles.Number,

System.Globalization.NumberFormatInfo.CurrentInfo, out b);

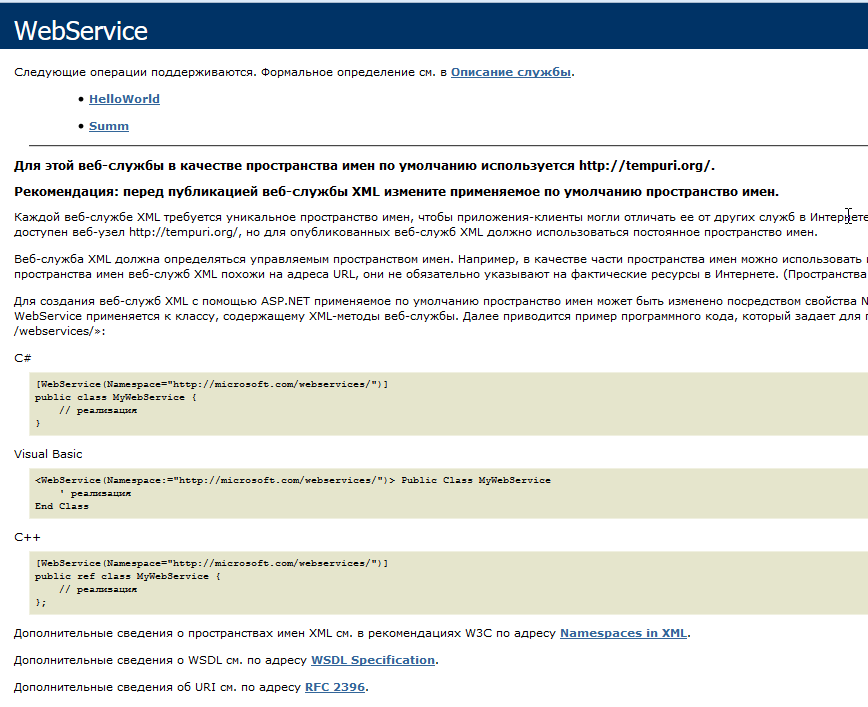
if (flag == false) return "В первом поле должно быть число";

return "Сумма:" + (a + b).ToString();

}

}

}

Выберите в обозревателе решений файл с расширением .asmx и выполните для него команду «Отобразить»,View In Browser . У вас должен будет запуститься браузер и, если всё сделано правильно, появится следующее окно:  


Как можно заметить, нам не потребовалось писать никакой дополнительной логики для реализации веб-сервиса, за исключением указания атрибута [WebMethod].

Атрибут [WebService] обзначает класс, который будет представлен как веб-сервис. Файл asmx может содержать несколько классов, но только один их них может быть представлен как веб-сервис.

Класс веб-сервиса и все его методы, доступные в веб-сервисе должны быть объявлены с модификаторам доступа public. Если добавить атрибут [WebMethod] к приватному методу, то мы не получим сообщения об ошибке. Данный метод просто будет недоступен в веб-сервисе и его будет невозможно вызвать.

* 1. Создание WPF-приложения, потребителя сервиса

Созадим WPF-приложение, потребителя сервиса, ASMXClient.  
MainWindow.xaml

<Window x:Class="ASMXClient.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:ASMXClient"

mc:Ignorable="d"

Title="MainWindow" Height="350" Width="525">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto"/>

<ColumnDefinition Width="Auto"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<TextBox x:Name="textBox1" Grid.Column="0" HorizontalAlignment="Center" Height="23" TextWrapping="Wrap" VerticalAlignment="Center" Width="120" Margin="10"/>

<TextBox x:Name="textBox2" Grid.Column="1" HorizontalAlignment="Center" Height="23" TextWrapping="Wrap" VerticalAlignment="Center" Width="120" Margin="10"/>

<Button x:Name="button" Grid.Row="1" Content="Вычислить" HorizontalAlignment="Center" VerticalAlignment="Center" Width="75" Margin="10" Click="button\_Click"/>

<TextBlock x:Name="textBlock" Grid.Column="1" Grid.Row="1" HorizontalAlignment="Center" TextWrapping="Wrap" VerticalAlignment="Center" Margin="10"/>

</Grid>

</Window>

MainWindow.xaml.cs

using System.Windows;

namespace ASMXClient

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void button\_Click(object sender, RoutedEventArgs e)

{

ServiceReference.WebServiceSoapClient serviceClient = new ServiceReference.WebServiceSoapClient();

string sum = serviceClient.Summ(textBox1.Text, textBox2.Text);

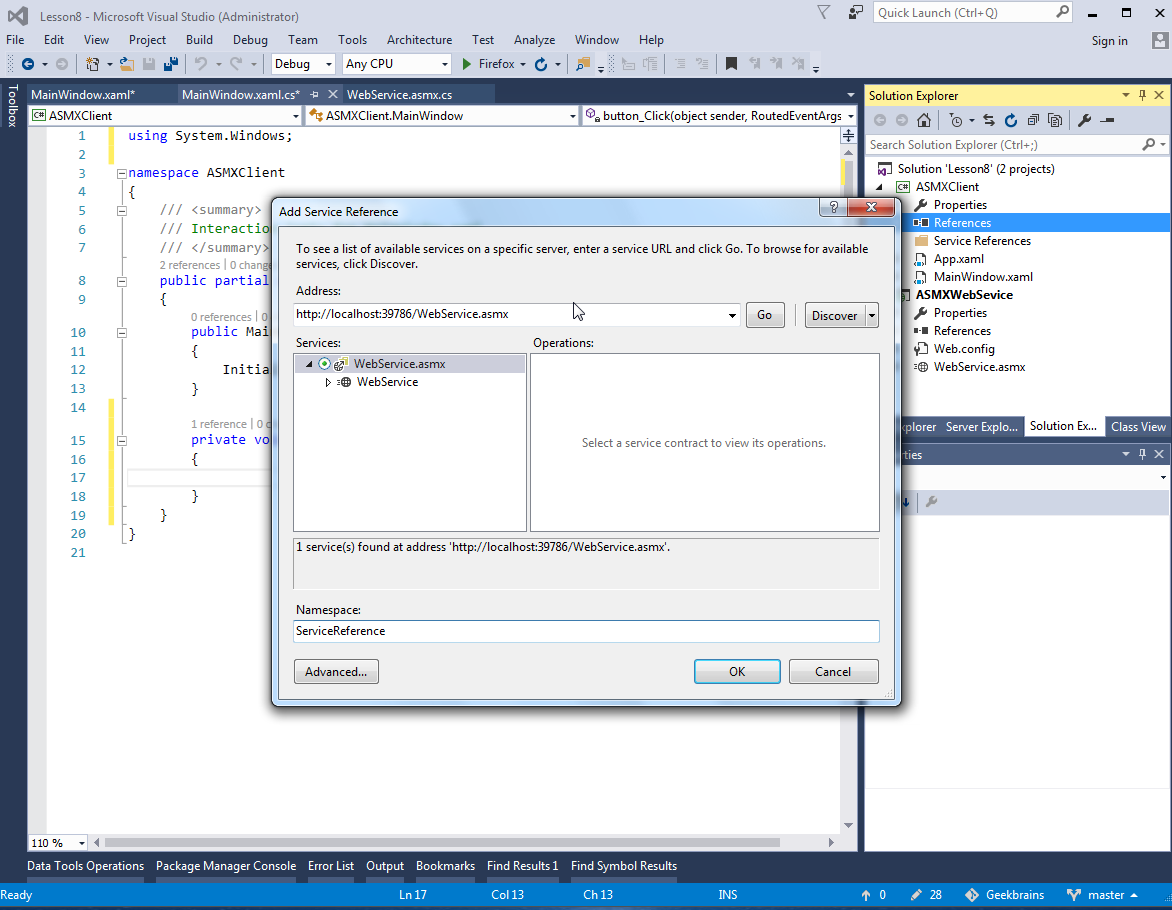
textBlock.Text = sum;

}

}

}

Добавим ссылку на веб-сервис в проект.



1. WCF-сервис

Характерные особенности:

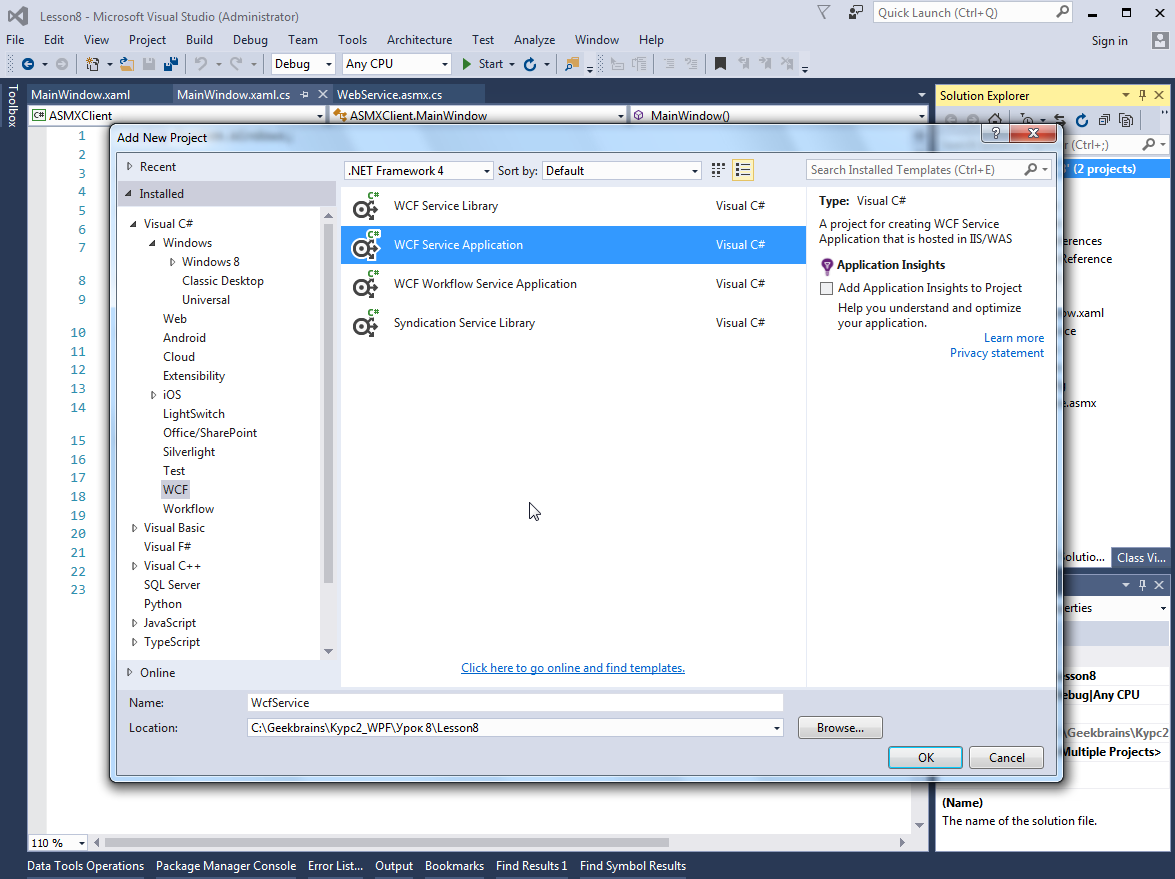
* Основывается на протоколе SOAP.
* Передача данных осуществляется в формате XML.
* Поддерживает различные траспортные протоколы (TCP,HTTP,HTTPS,Named Pipes,MSMQ).
* Не является open source технологией, но с сервисом могут работать клиентские приложения, понимающие xml.
* Может быть развернут на веб сервере IIS, внутри произвольного приложения или Windows сервиса.

Все WCF сервисы предоставляют контракты. Контракт – это платформонезависимый и стандартый способ описания сервисов. Для WCF определены 4 типа контрактов:

* Контракты сервиса [ServiceContract]. Определяют какие операции клиент может запрашивать у сервиса.
* Контракты данных [DataContract]. Определяют какие типы данных передаются от клиента к сервису и обратно. WCF определяет неявные контракты для встроенных типов (int, string и т.п.), но для пользовательских типов нужно указывать явные котракты.
* Контракты ошибок [FaultContract]. Определяют какие ошибки генерируются сервисом и как эти ошибки обрабатываются и передаются клиенту.
* Контракты сообщений [MessageContract]. Определяют контракт сообщения для типа, т. е. определяет сопоставление между типом и конвертом SOAP.
  1. Создание WCF -сервиса

Клиентское приложение WPF, выполняет расчет числа дней от указанной даты, используя веб-сервис. Клиентское приложение и веб-сервис развернуты на одном компьютере.

Создадим WCF-сервис, выполняющий расчет числа дней от указанной даты. Для этого выберем шаблон проекта WCF-service application.



IService.cs содержит пример описания 3 типов контрактов WCF.

[ServiceContract]

public interface IService1

{

[OperationContract]

string GetData(int value);

[OperationContract]

CompositeType GetDataUsingDataContract(CompositeType composite);

// TODO: Add your service operations here

}

// Use a data contract as illustrated in the sample below to add composite types to service operations.

[DataContract]

public class CompositeType

{

bool boolValue = true;

string stringValue = "Hello ";

[DataMember]

public bool BoolValue

{

get { return boolValue; }

set { boolValue = value; }

}

[DataMember]

public string StringValue

{

get { return stringValue; }

set { stringValue = value; }

}

}

Service1.svc.cs содержит реализация интерфейса IService.

Добавим в IServices1.cs описание метода для расчета количества дней с определенной даты.

[OperationContract]

int calculateDays(int day, int month, int year);

Реализуем данный метод в файле Service1.svc.cs.

public class Service1 : IService1

{

public int calculateDays(int day, int Month, int year)

{

DateTime dt = new DateTime(year, Month, day);

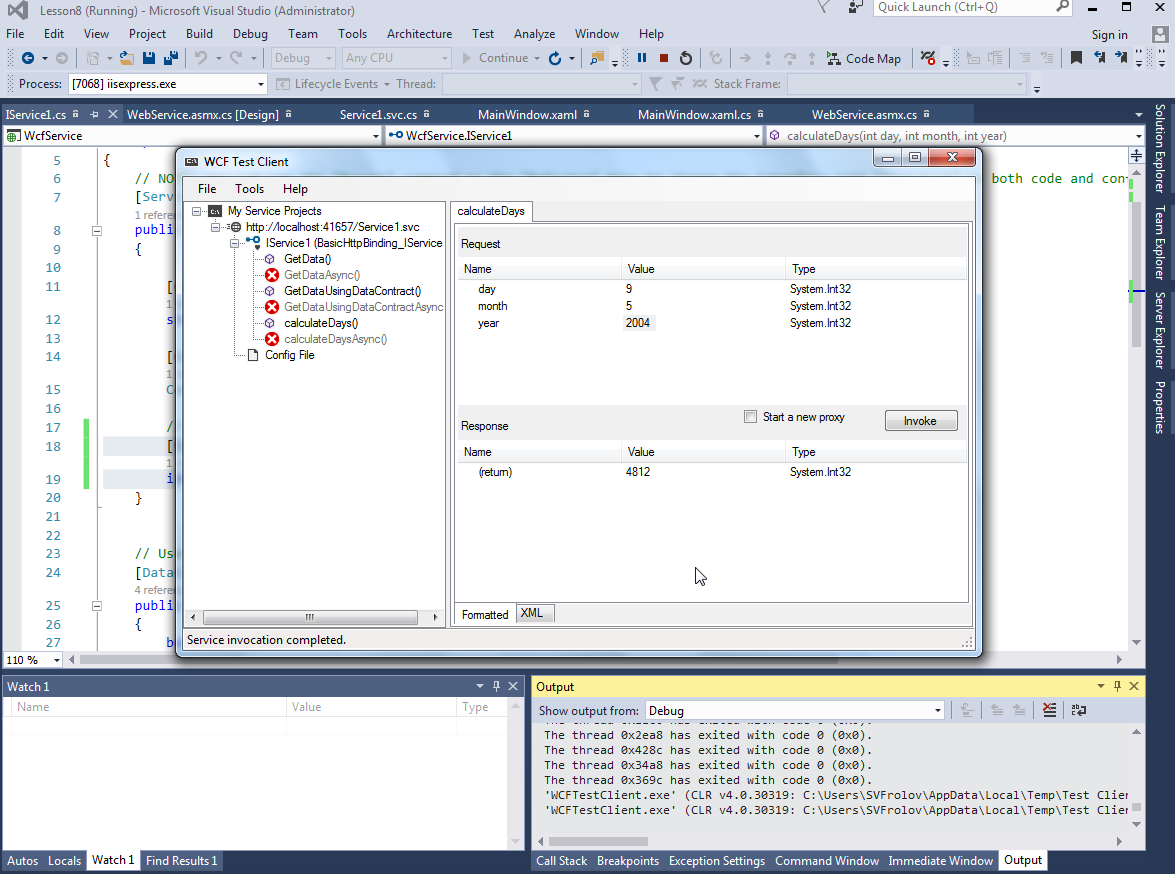
int datetodays = DateTime.Now.Subtract(dt).Days;

return datetodays;

}

…

В дереве проекта выбераем файл Service.svc и нажимаем F5.



В меню «Файл» «Добавить службу» можно скопировать путь к службе.

<http://localhost:41657/Service1.svc>.

* 1. Создание WPF-приложения потребителя WCF-сервиса

1. Web-api сервис

* В качестве транспортного протокола используется только HTTP.
* Open source технология для построения REST-ful сервисов.
* Может быть развернут на веб сервере IIS или внутри произвольного приложения.
* Передача данных может осуществляться в ASCII, XML, JSON или любых других форматах, распознаваемых одновременно и клиентом, и сервером.

1. ДЗ

Измените WPF приложение для ведения списка сотрудников компании, из урока №5, **используя связывание данных, DataGrid и ADO.NET**.  
1. Создайте таблицы Employee и Department в БД MSSQL Server и заполните списки сущностей начальными данными.  
2. Для списка сотрудников и списка департаментов предусмотрите визуализацию (отображение). Это можно сделать, например, с использованием ComboBox или ListView.  
3. Предусмотрите возможность редактирования сотрудников и департаментов. Должна быть возможность изменить департамент у сотрудника. Список департаментов для выбора, можно выводить в ComboBox, это все можно выводить на дополнительной форме.  
4. Предусмотрите возможность создания новых сотрудников и департаментов. Реализуйте данную возможность либо на форме редактирования, либо сделайте новую форму.