ПWS

ПОИТ-4,

лабораторная 5

2 часа

**Разработка и исследование простейших WCF-сервиса**

1. Разработайте WCF-сервис с именем **WCFSiplex** помощью технологии ASP.NET, шаблон **библиотека службы WCF**.
2. **WCFSiplex** включает в себя следующие три метода (повторяют методы предыдущей лабораторной работы).

|  |  |  |
| --- | --- | --- |
| Имя метода и его назначение | Параметры метода | Возвращаемый тип |
| **Add**  Возвращает значение суммы двух параметров | int x,  int y | int |
| **Concat**  Возвращает конкатенацию первого и второго параметров | string s  double d, | string |
| **Sum**  Возвращает объект A:  поле s – конкатенация a1.s и a2.s;  поле к – сумма a1.k и a2.k;  поле f – сумма a1.f и a2.f; | A a1,  A a2 | A |

1. Продемонстрируйте работоспособность сервиса с помощью браузера.
2. Разработайте хост для сервиса в форме консольного приложения.
3. Добавьте к сервису дополнительную конечную точку, работающую по протоколу TCP. Используйте привязку **NetTcpBinding**.
4. Продемонстрируйте работоспособность двух конечных точек сервиса с помощью 2 консольных приложений-клиентов (для каждой конечной точки).

Для успешного выполнения лабораторной работы по разработке и исследованию простейшего WCF-сервиса, нужно понять основы технологии WCF, ее применение и способы настройки. Вот ключевая теория, которая вам понадобится:

**Windows Communication Foundation (WCF)**

**WCF** — это платформа Microsoft для создания распределенных приложений. Она позволяет приложениям взаимодействовать друг с другом через различные протоколы связи, такие как HTTP, TCP, MSMQ и другие.

**Основные элементы WCF**

1. **Service (Сервис)**  
   Сервис предоставляет одну или несколько операций (методов), которые могут быть вызваны клиентами.
2. **Endpoint (Конечная точка)**  
   Каждая служба WCF должна иметь хотя бы одну конечную точку, которая указывает, где находится служба, как с ней взаимодействовать и что она предлагает:
   * **Address (Адрес)**: URL-адрес службы.
   * **Binding (Привязка)**: Указывает протокол связи (HTTP, TCP и т.д.).
   * **Contract (Контракт)**: Определяет, какие операции предоставляет служба.
3. **Host (Хост)**  
   Хост — это среда выполнения, где размещается WCF-сервис. Это может быть консольное приложение, IIS или Windows Service.

**Разработка WCF-сервиса**

**1. Создание WCF-сервиса**

1. Создайте проект с типом **"Библиотека службы WCF"** в Visual Studio.
2. Определите интерфейс для методов сервиса:

csharp

Копировать код

[ServiceContract]

public interface IWCFSiplex

{

[OperationContract]

int Add(int x, int y);

[OperationContract]

string Concat(string s, double d);

[OperationContract]

A Sum(A a1, A a2);

}

public class A

{

public string S { get; set; }

public int K { get; set; }

public float F { get; set; }

}

1. Реализуйте интерфейс в классе сервиса:

csharp

Копировать код

public class WCFSiplex : IWCFSiplex

{

public int Add(int x, int y) => x + y;

public string Concat(string s, double d) => $"{s}{d}";

public A Sum(A a1, A a2)

{

return new A

{

S = a1.S + a2.S,

K = a1.K + a2.K,

F = a1.F + a2.F

};

}

}

**2. Настройка сервиса**

Настройте Web.config для добавления службы:

xml

Копировать код

<system.serviceModel>

<services>

<service name="WCFService.WCFSiplex">

<endpoint address="" binding="basicHttpBinding" contract="WCFService.IWCFSiplex" />

</service>

</services>

</system.serviceModel>

**3. Создание хоста**

Создайте консольное приложение для запуска сервиса:

csharp

Копировать код

using System;

using System.ServiceModel;

class Program

{

static void Main(string[] args)

{

using (ServiceHost host = new ServiceHost(typeof(WCFService.WCFSiplex)))

{

host.Open();

Console.WriteLine("Service is running...");

Console.ReadLine();

}

}

}

**Дополнительная конечная точка**

Добавьте в конфигурацию конечную точку для TCP-протокола:

xml

Копировать код

<endpoint address="net.tcp://localhost:8080/WCFSiplex"

binding="netTcpBinding"

contract="WCFService.IWCFSiplex" />

**Создание клиентов**

**Клиент для HTTP**

1. Создайте консольное приложение.
2. Добавьте ссылку на сервис (Service Reference).
3. Используйте вызов методов сервиса:

csharp

Копировать код

var client = new WCFSiplexClient();

Console.WriteLine(client.Add(5, 10));

**Клиент для TCP**

1. Повторите настройку клиента, указав TCP-адрес:

csharp

Копировать код

var binding = new NetTcpBinding();

var endpoint = new EndpointAddress("net.tcp://localhost:8080/WCFSiplex");

var factory = new ChannelFactory<IWCFSiplex>(binding, endpoint);

var client = factory.CreateChannel();

Console.WriteLine(client.Add(5, 10));

**Демонстрация работы**

1. Запустите консольное приложение-хост.
2. Проверьте доступность сервиса через браузер для HTTP-адреса.
3. Создайте два разных клиента для HTTP и TCP и убедитесь, что сервис работает через обе конечные точки.

**Вывод**

В ходе выполнения лабораторной работы вы разработаете WCF-сервис, освоите его развертывание и настройку, а также исследуете работу через несколько конечных точек.