Конструювання клієнт-серверної взаємодії за допомогою WCF-сервісу.

**Мета:** Засвоїти методику та виробити практичні навички роботи та створення клієнт-серверних програм за допомогою сервісу WCF.

**Теоретичні відомості.**

**WCF**

WCF (**Windows Communication Foundation**) – це програмний фреймворк, який використовується для обміну даними між додатками, що входять до складу. NET Framework.

WCF робить можливим побудову безпечних і надійних транзакційних систем через спрощену уніфіковану програмну модель міжплатформенної взаємодії. Комбінуючи функціональність існуючих технологій .NET по розробці розподілених додатків, WCF надає єдину інфраструктуру розробки, а при вмілому застосуванні підвищує продуктивність і знижує витрати на створення безпечних, надійних і транзакційних Web-служб нового покоління.

Клас служби WCF не може існувати самостійно. Кожна служба WCF повинна перебувати під управлінням деякого процесу Windows, так званого хостового процесу. Існують кілька варіантів хостингу:

* автохостинг (тобто хост-процесом є, наприклад, консольний або графічний Windows додаток);
* хостинг в одній із служб Windows;
* хостинг з використанням IIS (Internet Information Server) або WAS (Windows Activation Services).

**Приклад створення служби WCF і тестування**

1. У меню *Файл* потрібно вибрати пункт *Створити*, а потім команду *Проект*.
2. У діалоговому вікні *Новий проект,* розгорнувши вузол Visual C #, потрібно вибрати пункт *WCF*, потім *Бібліотека служби WCF*. Натисніть кнопку *ОК* для відкриття проекту.
3. У браузері рішень двічі клацніть на файл IService1.cs і знайдіть наступний рядок:

[OperationContract]

string GetData(int value);

Змініть тип параметра value на String:

[OperationContract]

string GetData(string value);

1. Атрибути <OperationContract()> і [OperationContract] у наведеному вище коді. Ці атрибути обов'язкові для будь-якого методу, наданого службою.
2. У браузері рішень двічі клацніть файл Service1.cs і знайдіть наступний рядок:

public string GetData(int value)

{

return string.Format("You entered: {0}", value);

}

1. Змініть тип параметра значення на String:

public string GetData(string value)

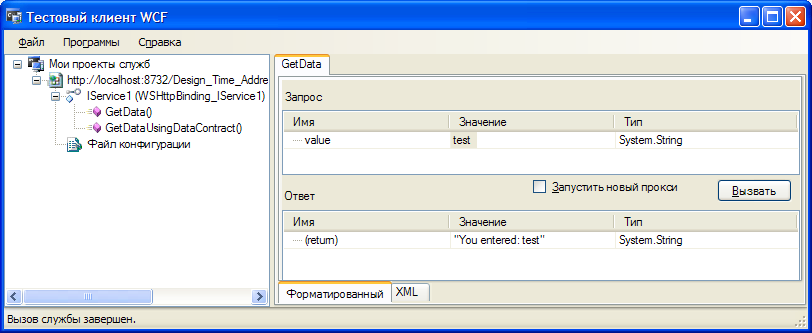
{

return string.Format("You entered: {0}", value);

}

Тестування служби WCF:

1. Щоб запустити службу, натисніть клавішу *F5*. Форма *Клієнт тесту* *WCF* з'явиться на екрані і завантажить службу.
2. У формі *Клієнт тесту* *WCF* двічі клацніть метод *GetData* () під вузлом *IService1*. З'явиться вкладка *GetData*.
3. В області *Запит* виберіть поле *Значення* і введіть будь-яку стрічку.
4. Натисніть кнопку *Виклик*. Якщо з'явиться діалогове вікно *Попередження системи безпеки*, натисніть кнопку *Так*. Результат буде виведений в області *Відповідь.*



1. Щоб закрити цю тестову форму, в меню *Файл* клацніть *Вихід*.

**Доступ до служби WCF**

1. У діалоговому вікні *Новий проект* розгорніть вузол *Visual C #,* виберіть пункт *Windows* і виберіть елемент *Додаток Windows Forms*. Натисніть кнопку *ОК* для відкриття проекту.
2. Клацніть правою кнопкою миші *WindowsApplication1* і виберіть *Додати посилання на службу*.
3. У діалоговому вікні *Додати посилання на службу* виберіть *Знайти.*
4. В області Служби з'явиться *Service1*. Якщо не з’явиться, то з Тестового клієнту скопіюйте посилання на службу.
5. Натисніть кнопку *ОК*, щоб додати це посилання на службу.
6. Якщо конструктор *Windows Forms* ще не відкрито, двічі клацніть файл *Form1.cs* в оглядачі рішень, щоб відкрити його.
7. З *Панелі елементів* перетягніть у форму елементи керування *TextBox, Label і Button.*
8. Двічі клацніть *Button* і додайте в обробник подій *Click* наступний код:

private void button1\_Click(System.Object sender, System.EventArgs e)

{

ServiceReference1.Service1Client client = new

ServiceReference1.Service1Client();

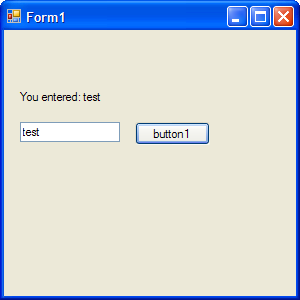
string returnString;

returnString = client.GetData(textBox1.Text);

label1.Text = returnString;

}

1. Щоб запустити проект, натисніть клавішу *F5*. Введіть якийсь текст і натисніть кнопку *Button1*. У label відображається повідомлення "Ви ввели:" і введений вами текст.



**Клієнт-серверна архітектура**

[Архітектура](http://uk.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0) клієнт-сервера є однією із [архітектурних шаблонів програмного забезпечення](http://uk.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%BD%D1%96_%D1%88%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) та є домінуючою концепцією у створенні розподілених мережних застосувань і передбачає взаємодію та обмін даними між ними. Вона передбачає такі основні компоненти:

* набір [серверів](http://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80), які надають інформацію або інші послуги програмам, які звертаються до них;
* набір [клієнтів](http://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82), які використовують сервіси, що надаються серверами;
* [мережа](http://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%80%D0%B5%D0%B6%D0%B0), яка забезпечує взаємодію між клієнтами та серверами.

Сервери є незалежними один від одного. Клієнти також функціонують паралельно і незалежно один від одного. Немає жорсткої прив'язки клієнтів до серверів. Більш ніж типовою є ситуація, коли один сервер одночасно обробляє запити від різних клієнтів; з другого боку, клієнт може звертатися то до одного сервера, то до іншого. Клієнти мають знати про доступні сервери, але можуть не мати жодного уявлення про існування інших клієнтів.

Для реалізації клієнт-серверної взаємодії в мові C#, є ряд класів:

* TcpClient - забезпечує клієнтські підключення для мережевих служб TCP.
* TcpListener - чекає підключення від tcp-клієнтів мережі.

Для створення клієнт-серверної програми важливу роль відіграє порт передачі даних та ІР-адреса комп’ютера. Якщо комп’ютер не в мережі, то використовується операційною системою локальна ІР-адреса: 127.0.0.1

Розглянемо приклад коду створення простої клієнт-серверної програми.

Створюємо два проекти, один з яких відповідає за сервер:

*//*

*/\* Server Program \*/*

using System;

using System.Text;

using System.Net;

using System.Net.Sockets;

public class serv {

public static void Main() {

try {

IPAddress ipAd = IPAddress.Parse("172.21.5.99");

*// use local m/c IP address, and*

*// use the same in the client*

*/\* Initializes the Listener \*/*

TcpListener myList=new TcpListener(ipAd,8001);

*/\* Start Listeneting at the specified port \*/*

myList.Start();

Console.WriteLine("The server is running at port 8001...");

Console.WriteLine("The local End point is :" +

myList.LocalEndpoint );

Console.WriteLine("Waiting for a connection.....");

Socket s=myList.AcceptSocket();

Console.WriteLine("Connection accepted from " + s.RemoteEndPoint);

byte[] b=new byte[100];

int k=s.Receive(b);

Console.WriteLine("Recieved...");

for (int i=0;i<k;i++)

Console.Write(Convert.ToChar(b[i]));

ASCIIEncoding asen=new ASCIIEncoding();

s.Send(asen.GetBytes("The string was recieved by the server."));

Console.WriteLine("\nSent Acknowledgement");

*/\* clean up \*/*

s.Close();

myList.Stop();

}

catch (Exception e) {

Console.WriteLine("Error..... " + e.StackTrace);

}

}

}

Другий проект відповідає за клієнтську частину:

*/\* Client Program \*/*

using System;

using System.IO;

using System.Net;

using System.Text;

using System.Net.Sockets;

public class clnt {

public static void Main() {

try {

TcpClient tcpclnt = new TcpClient();

Console.WriteLine("Connecting.....");

tcpclnt.Connect("172.21.5.99",8001);

*// use the ipaddress as in the server program*

Console.WriteLine("Connected");

Console.Write("Enter the string to be transmitted : ");

String str=Console.ReadLine();

Stream stm = tcpclnt.GetStream();

ASCIIEncoding asen= new ASCIIEncoding();

byte[] ba=asen.GetBytes(str);

Console.WriteLine("Transmitting.....");

stm.Write(ba,0,ba.Length);

byte[] bb=new byte[100];

int k=stm.Read(bb,0,100);

for (int i=0;i<k;i++)

Console.Write(Convert.ToChar(bb[i]));

tcpclnt.Close();

}

catch (Exception e) {

Console.WriteLine("Error..... " + e.StackTrace);

}

}

}

**Завдання**

Розробити клієнт-серверну програму на основі попередньої лабораторної роботи з ADO.NET використанням WCF служби.

Реалізувати:

* Серверну програму, яка буде отримувати запит від клієнта у форматі XML запити для стандартних дій з БД(кількість записів, пошук по всіх параметрів, додавання параметрів та знищення) та у відповідь відправляти результат у вигляді XML.
* Клієнську програму, яка отримує запити від користувача, перетворює в XML формат і передає на сервер на опрацювання, у відповідь отримує результат у форматі XML який обробляє і виводить інформацію інтуїтивно зрозумілу користувачу