|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра теоретической и прикладной информатики | | |
| Лабораторная работа № 2 | | |
| по дисциплине «Компьютерные сети» | | |
| **Технология клиент-сервер: эхо-повтор.** | | |
|  | | |
|  | Бригада 11 | Затолоцкая юлия |
| Группа ПМ-91 | константинова анастасия |
|  |  |
|  |  |
| Преподаватели | Сивак Мария Алексеевна |
|  | Кобылянский Валерий Григорьевич |
| Дата | 04.09.2021 |
|  |  |
| Новосибирск, 2021 | | |

1. **Цель работы**

Изучить основные принципы разработки клиент-серверных приложений на примере простейшей однопользовательской программы.

1. **Задание**
2. Написать простейшее приложение с одним сервером и одним клиентом, используя API-интерфейс низкого уровня.

Запуск сервера происходит с указанием номера порта протокола TCP или UDP из диапазона возможных номеров. Каждая бригада создает свой сервер с номером порта, определяемым по правилу: номер бригады + 2000. Сервер начинает свою работу с ожидания запроса от клиента на соединение. Запуск клиента происходит с указанием ему IP-адреса и номера порта сервера. Клиент устанавливает связь с сервером и посылает набор данных, введенный пользователем, на сервер. Сервер получает набор данных, выполняет указанные в варианте действия, возвращает клиенту результат и снова переходит в состояние ожидания запроса на соединение. Клиент, получив ответ с сервера, выводит его на экран и прекращает свою работу.

**Вариант:**

3. Клиент пересылает серверу данные (строки текста). Сервер в полученном тексте в конец каждого предложения вставляет свой IP-адрес и номер порта и возвращает в таком виде данные клиенту.

1. **Код программы**

**Сервер:**

using System;

using System.Net;

using System.Net.Sockets;

using System.Text;

namespace server

{

class Program

{

static void Main(string[] args)

{

const int buff\_s = 1000;

// Создание сокета

Socket server = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

try

{

Int32 port = 2011; // порт сервера

string adr1 = "0.0.0.0";//;

IPAddress localhost = IPAddress.Parse(adr1);

IPEndPoint address = new IPEndPoint(localhost, port);

server.Bind(address);

Byte[] bytes = new Byte[buff\_s];

String data;

while (true)

{

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

server.Listen(1);

Socket new\_s = server.Accept();

Console.WriteLine("Connected!");

new\_s.Receive(bytes);

int i = 0;

for (; bytes[i] != 0;)

i++;

if (i > 0)

{

IPEndPoint endPoint = new\_s.LocalEndPoint as IPEndPoint;

string localIP = endPoint.Address.ToString();

data = Encoding.ASCII.GetString(bytes, 0, i);

Console.WriteLine("Received: {0}", data);

string answer\_message = data + " "+ localIP + " " + port;

byte[] msg = Encoding.ASCII.GetBytes(answer\_message);

new\_s.Send(msg, msg.Length, SocketFlags.None);

Console.WriteLine("Sent: {0}", answer\_message);

}

new\_s.Close();

}

}

catch (SocketException expt)

{

Console.WriteLine("SocketException: {0}", expt);

}

finally

{

server.Dispose();

}

}

}

}

**Клиент:**

using System;

using System.Text;

using System.Net.Sockets;

namespace client

{

class Program

{

static void Main(string[] args)

{

try

{

Console.WriteLine("Enter message to server:");

string message = Console.ReadLine(); // строка, которую пошлем серверу

Console.WriteLine("Enter ip of server:");

string localhost = Console.ReadLine();

Console.WriteLine("Enter port of server: ");

Int32 port = Convert.ToInt32(Console.ReadLine());

Byte[] data = System.Text.Encoding.ASCII.GetBytes(message);

Socket client = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

Console.WriteLine("Sent: {0}", message);//печатаем то, что отправили

client.Connect(localhost, port);

client.Send(data, data.Length, SocketFlags.None);

// буффер для приема сообщений

data = new Byte[1000];

// строка для приема сообщений сервера

client.Receive(data, SocketFlags.None);

int i = 0;

for (; data[i] != 0;)

i++;

String responseData = Encoding.ASCII.GetString(data, 0, i);

//печатаем то, что получили

Console.WriteLine("Received: {0}", responseData);

// закрываем соединение

client.Close();

}

catch (ArgumentNullException expt)

{

Console.WriteLine("ArgumentNullException: {0}", expt);

}

catch (SocketException expt)

{

Console.WriteLine("SocketException: {0}", expt);

}

Console.WriteLine("\n Press Enter to continue...");

Console.Read();

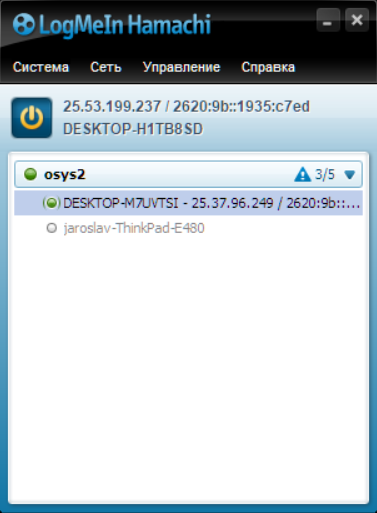
}

}

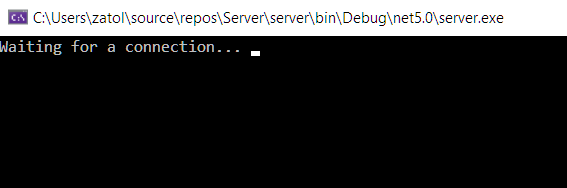
}

1. **Результаты тестирования**

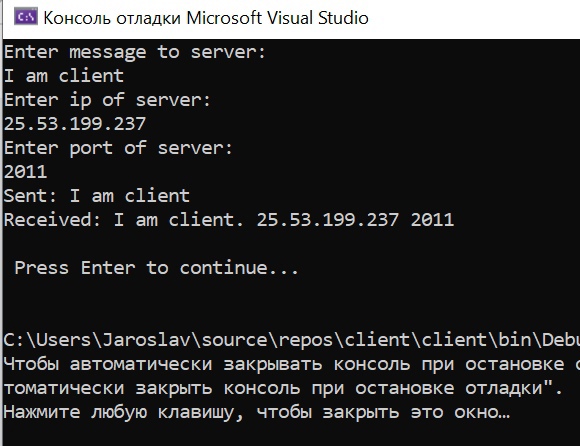
Для тестирования создадим виртуальную сеть на основе бесплатной VPN-технологии Hamachi, которая является надстройкой над сетью Интернет и может использоваться для объединения в единую защищенную сеть компьютеров с серыми IP-адресами.



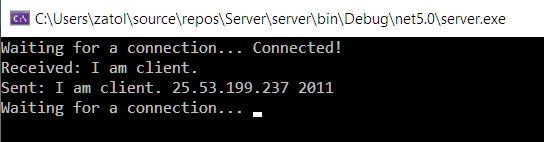
Запустим сервер.



Запустим клиента на другом компьютере и подключимся к серверу, используя виртуальный IP адрес.



Сервер получил сообщение, ответ клиенту отправлен успешно. Сервер ожидает следущего запроса на соединение. Клиент завершил работу.



1. **Выводы**

Контрольные вопросы проработаны.