Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Институт информационных технологий

Факультет компьютерных технологий

Лабораторная работа № 5

РАБОТА В СЕТИ

Выполнил: студент гр. 881062 Литасов О.А.

Проверил: Потоцкий Д.С.

Минск 2020

## *Задания*

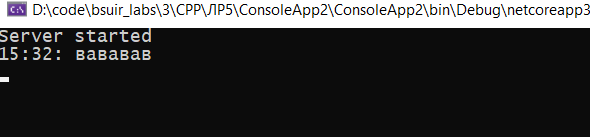
**Вариант 1**. Реализовать клиент-серверное приложение по получению информации о ценах на товары. Название товара передается от клиента.

**Вариант 2**. Создать сетевое приложение для передачи клиенту от сервера файла приложения и запуска его на стороне клиента.

**Вариант 3.** Передать от сервера файл HTML и открыть его браузером на стороне клиента.

**Вариант 4**. Передать сериализованный объект некоторого класса клиенту, открыть его на стороне клиента (десериализовать на стороне клиента) и использовать какой-либо метод.

## *Скриншоты:*





**Код программы**

using System;

using System.IO;

using System.Net;

using System.Net.Sockets;

using System.Xml.Serialization;

using System.Threading;

public class UdpFileServer

{

[Serializable]

public class FileDetails

{

public string FILETYPE = "";

public long FILESIZE = 0;

}

private static FileDetails fileDet = new FileDetails();

private static IPAddress remoteIPAddress;

private const int remotePort = 5002;

private static UdpClient sender = new UdpClient();

private static IPEndPoint endPoint;

private static FileStream fs;

[STAThread]

static void Main(string[] args)

{

try

{

Console.WriteLine("IP-Addres");

remoteIPAddress = IPAddress.Parse(Console.ReadLine().ToString());

endPoint = new IPEndPoint(remoteIPAddress, remotePort);

Console.WriteLine("Enter file");

fs = new FileStream(@Console.ReadLine().ToString(), FileMode.Open, FileAccess.Read);

if (fs.Length > 18192)

{

Console.Write("File size is too big");

sender.Close();

fs.Close();

return;

}

SendFileInfo();

Thread.Sleep(2000);

SendFile();

Console.ReadLine();

}

catch (Exception eR)

{

Console.WriteLine(eR.ToString());

}

}

public static void SendFileInfo()

{

fileDet.FILETYPE = fs.Name.Substring((int)fs.Name.Length - 3, 3);

fileDet.FILESIZE = fs.Length;

XmlSerializer fileSerializer = new XmlSerializer(typeof(FileDetails));

MemoryStream stream = new MemoryStream();

fileSerializer.Serialize(stream, fileDet);

stream.Position = 0;

Byte[] bytes = new Byte[stream.Length];

stream.Read(bytes, 0, Convert.ToInt32(stream.Length));

Console.WriteLine("Sending file...");

sender.Send(bytes, bytes.Length, endPoint);

stream.Close();

}

private static void SendFile()

{

Byte[] bytes = new Byte[fs.Length];

fs.Read(bytes, 0, bytes.Length);

Console.WriteLine("Sending file " + fs.Length + " Byte");

try

{

sender.Send(bytes, bytes.Length, endPoint);

}

catch (Exception eR)

{

Console.WriteLine(eR.ToString());

}

finally

{

fs.Close();

sender.Close();

}

Console.WriteLine("File sent.");

Console.Read();

}

}using System;

using System.IO;

using System.Net;

using System.Diagnostics;

using System.Net.Sockets;

using System.Text;

using System.Xml.Serialization;

public class UdpFileClient

{

[Serializable]

public class FileDetails

{

public string FILETYPE = "";

public long FILESIZE = 0;

}

private static FileDetails fileDet;

private static int localPort = 5002;

private static UdpClient receivingUdpClient = new UdpClient(localPort);

private static IPEndPoint RemoteIpEndPoint = null;

private static FileStream fs;

private static Byte[] receiveBytes = new Byte[0];

[STAThread]

static void Main(string[] args)

{

GetFileDetails();

ReceiveFile();

}

private static void GetFileDetails()

{

try

{

Console.WriteLine("Waitin for file info");

receiveBytes = receivingUdpClient.Receive(ref RemoteIpEndPoint);

Console.WriteLine("file info is gotten");

XmlSerializer fileSerializer = new XmlSerializer(typeof(FileDetails));

MemoryStream stream1 = new MemoryStream();

stream1.Write(receiveBytes, 0, receiveBytes.Length);

stream1.Position = 0;

fileDet = (FileDetails)fileSerializer.Deserialize(stream1);

Console.WriteLine("Gettin file type ." + fileDet.FILETYPE +

" size of " + fileDet.FILESIZE.ToString() + " Byte");

}

catch (Exception eR)

{

Console.WriteLine(eR.ToString());

}

}

public static void ReceiveFile()

{

try

{

Console.WriteLine("Waitin for file");

receiveBytes = receivingUdpClient.Receive(ref RemoteIpEndPoint);

fs = new FileStream("temp." + fileDet.FILETYPE, FileMode.Create, FileAccess.ReadWrite, FileShare.ReadWrite);

fs.Write(receiveBytes, 0, receiveBytes.Length);

Console.WriteLine("File saved");

Console.WriteLine("Openin file");

Process.Start(fs.Name);

}

catch (Exception eR)

{

Console.WriteLine(eR.ToString());

}

finally

{

fs.Close();

receivingUdpClient.Close();

Console.Read();

}

}

}using System;

using System.Text;

using System.Net;

using System.Net.Sockets;

namespace SocketTcpServer

{

class Program

{

static int port = 8005;

static void Main(string[] args)

{

IPEndPoint ipPoint = new IPEndPoint(IPAddress.Parse("127.0.0.1"), port);

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

try

{

listenSocket.Bind(ipPoint);

listenSocket.Listen(10);

Console.WriteLine("Server started");

while (true)

{

Socket handler = listenSocket.Accept();

StringBuilder builder = new StringBuilder();

int bytes = 0;

byte[] data = new byte[256];

do

{

bytes = handler.Receive(data);

builder.Append(Encoding.Unicode.GetString(data, 0, bytes));

}

while (handler.Available > 0);

Console.WriteLine(DateTime.Now.ToShortTimeString() + ": " + builder.ToString());

string message = "Ur message was sent";

data = Encoding.Unicode.GetBytes(message);

handler.Send(data);

handler.Shutdown(SocketShutdown.Both);

handler.Close();

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

}

}using System;

using System.Text;

using System.Net;

using System.Net.Sockets;

namespace SocketTcpClient

{

class Program

{

static int port = 8005;

static string address = "127.0.0.1";

static void Main(string[] args)

{

try

{

IPEndPoint ipPoint = new IPEndPoint(IPAddress.Parse(address), port);

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

socket.Connect(ipPoint);

Console.Write("Enter ur message:");

string message = Console.ReadLine();

byte[] data = Encoding.Unicode.GetBytes(message);

socket.Send(data);

data = new byte[256];

StringBuilder builder = new StringBuilder();

int bytes = 0;

do

{

bytes = socket.Receive(data, data.Length, 0);

builder.Append(Encoding.Unicode.GetString(data, 0, bytes));

}

while (socket.Available > 0);

Console.WriteLine("server response: " + builder.ToString());

socket.Shutdown(SocketShutdown.Both);

socket.Close();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Console.Read();

}

}

}