 МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**Комп’ютерний практикум №2**

з дисципліни «[ВЕБ-технології та ВЕБ-дизайн-2. Веб-орієнтована розробка програмного забезпечення](http://wiki.kpi.ua/index.php/%D0%92%D0%95%D0%91-%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%97%20%D1%82%D0%B0%20%D0%92%D0%95%D0%91-%D0%B4%D0%B8%D0%B7%D0%B0%D0%B9%D0%BD-2.%20%D0%92%D0%B5%D0%B1-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B0%20%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B0%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F_(35103510))»

**Виконав:**

студент гр. БС-81

Мумінов А.О.

**Перевірив:**

ас. каф. БМК Матвійчук О. В.  
[ас. каф. БМК Давидько О. Б.](http://rozklad.kpi.ua/Schedules/ViewSchedule.aspx?v=b92c0571-df7d-410d-8013-82552f62f3cf)

Зараховано від \_\_\_.\_\_\_.\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис викладача)

Київ-2020

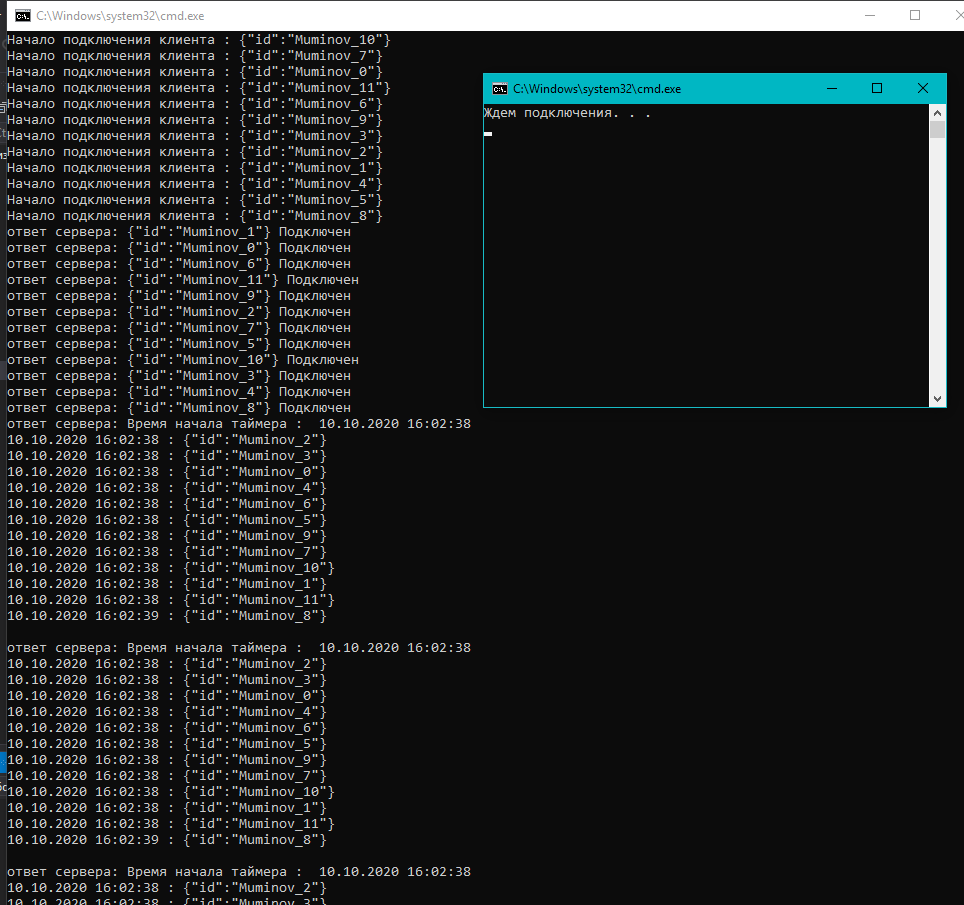
Завдання для практикума:  
Тема: Клієнт-серверна взаємодія

1. Створити веб-сервіс та клієнт до нього.
2. Сервіс та клієнт повинні взаємодіяти шляхом передачі даних по TCP сокетах в форматі Прізвище[0] <= “О” ? JSON : XML
3. Кожен з клієнтів після підключення повинен передати серверу структуру з 1 ключем “id” та значенням “Прізвище\_(#клієнта)”
4. Після підключення першого клієнта сервер має запустити глобальний таймер на #Варіант + 10 секунд
5. По завершенню таймера відправити кожному з клієнтів список з усіма підключеними клієнтами з датами їх підключення, а також дату старту таймера
6. Для отримання змін в сокетах скористатись механізмами для роботи із сокетами, що підтримуються обраною вами мовою. наступним системним інструментом Select()

6.\*) Екстра варіант [epoll/kqueue/I/O Completion Ports] (в залежності від обраної вами системи)

7.) Перевірити роботу системи з (#варіант mod 10) + 10 клієнтами

8.) В процесі роботи не використовувати сторонні бібліотеки для роботи в сокетами, з XML та  JSON по необхідності  
Виконання:



Лістинг:

//Server.cs  
using System;

using System.Collections.Generic;

using System.Text;

using System.Threading.Tasks;

using System.Net;

using System.Net.Sockets;

using Newtonsoft.Json;

using System.Threading;

namespace Server

{

struct Messege\_client

{

public string id;

public Messege\_client(string surname) { id = surname; }

}

struct Massege

{

public string massage;

public Massege(string value\_massage) { massage = value\_massage; }

}

// Класс для збереження даних, потрібних для роботи з підключення

class Serv

{

public static int port = 8005;

public static string address = "127.0.0.1";

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

};

class Server

{

static string list\_client = ""; // Список підключених

static List<Socket> sockets = new List<Socket> { }; // Для зберігання сокетів

static object locker = new object();

//Функція для роботи з підключеними сокетами

static public void Client\_accept(object arg)

{

Socket socket = (Socket)arg; // Отриманий сокет

StringBuilder builder = new StringBuilder();

int buffer\_bytes = 0;

byte[] buffer = new byte[256];

do

{

buffer\_bytes = socket.Receive(buffer);

builder.Append(Encoding.Unicode.GetString(buffer, 0, buffer\_bytes));

}

while (socket.Available > 0);

Messege\_client mess = JsonConvert.DeserializeObject<Messege\_client>(builder.ToString()); // ДиСеріалізація структури у формат json

lock (locker)

{

list\_client = list\_client + DateTime.Now + " : " + builder.ToString() + "\n"; // Занесемо підключенного кліента до списку

}

//Відправляемо відповідь

Massege answ = new Massege(builder.ToString() + " Подключен");

buffer = Encoding.Unicode.GetBytes(JsonConvert.SerializeObject(answ));

socket.Send(buffer);

}

static void Main(string[] args)

{

Console.WriteLine("Ждем подключения. . .");

// Створимо сокет

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

listenSocket.Bind(Serv.ipPoint); // Підклемо до адресси

listenSocket.Listen(10); // Почнемо прослуховування

bool timer\_flag = true;

while (true)

{

if (timer\_flag) // Таймер

{

List<Socket> cond = new List<Socket>() { listenSocket };

Socket.Select(cond, null, null, 100000); // Перевірка на зміни у сокетах

if (cond.Count > 0)

{

// Запускаемо Task - таймер

Task timer = Task.Run(() =>

{

list\_client += "Время начала таймера : " + DateTime.Now + "\n";

Thread.Sleep(TimeSpan.FromSeconds(22));

byte[] buffer = new byte[512];

Massege answ = new Massege(list\_client);

buffer = Encoding.Unicode.GetBytes(JsonConvert.SerializeObject(answ));

for (int i = 0; i < sockets.Count; i++)

sockets[i].Send(buffer);

});

timer\_flag = false;

}

}

// Приймаємо запит

sockets.Add(listenSocket.Accept());

Thread th = new Thread(new ParameterizedThreadStart(Client\_accept)); //Новий потік для роботи зі сокетом

th.Start(sockets[sockets.Count - 1]);

}

}

}

}

//Client.cs

using System;

using System.Text;

using System.Net;

using System.Net.Sockets;

using Newtonsoft.Json;

using System.Threading;

namespace Client

{

struct Messege\_client

{

public string id;

public Messege\_client(string surname) { id = surname; }

}

struct Massege

{

public string massage;

public Massege(string value\_massage) { massage = value\_massage; }

}

// Класс для збереження даних, потрібних для роботи з підключенням

class Serv

{

static public int port = 8005;

static public string address = "127.0.0.1";

//Объект який зв'язує порт та адрессу для підключення сокету

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

};

class Client

{

//Функція для підключення сокету до сервера

static public void client\_connect(object number)

{

//Створимо повідомлення кліента

Messege\_client client = new Messege\_client("Muminov\_" + number.ToString()); // Створимо структуру с даними клиента

string client\_Json = JsonConvert.SerializeObject(client); // Серіалізація структури у формат json

// Створимо tcp сокет

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

Console.WriteLine("Начало подключения клиента : " + client\_Json);

socket.Connect(Serv.ipPoint); // Підключемо сокет до сервера

byte[] buffer = Encoding.Unicode.GetBytes(client\_Json); // Перетворимо повідомлення в набір байтів

socket.Send(buffer); // Відправемо повідомлення через зв'язок сокетів

while (true) //Очікування на відповідь

{

buffer = new byte[512];

StringBuilder builder = new StringBuilder();

int buffer\_bytes = 0;

do

{

buffer\_bytes = socket.Receive(buffer, buffer.Length, 0);

builder.Append(Encoding.Unicode.GetString(buffer, 0, buffer\_bytes));

}

while (socket.Available > 0);

Massege answ = JsonConvert.DeserializeObject<Massege>(builder.ToString());

Console.WriteLine("ответ сервера: " + answ.massage); //Відповідь від сервера

}

}

static void Main(string[] args)

{

for (int i = 0; i <12; i++)

{

//Послідовно створюємо потік для кожного кліента

Thread th = new Thread(new ParameterizedThreadStart(client\_connect));

th.Start(i);

}

}

}

}