

Ministerul Educației a Republicii Moldova
Universitatea Tehnică din Moldova
Facultatea Calculatoare, Informatică și Microelectronică

Raport

La disciplina: Programarea în rețea
Lucrarea de laborator nr.5
Tema: “Sockets API”

A efectuat: Vozian T.
studenta grupei TI-161 FR
A verificat: Zgureanu A.

Link la repozitoriu: <https://github.com/taniuskav/Programarea-in-retea>

Obiective:

Proiectarea si realizarea unui protocol de transfer date (mesaje), utilizind protocolul de nivel de transport(TCP).

Executarea lucrării

WebSocket este un protocol care furnizează canale full-duplex de comunicare printr-o singură conexiune TCP. WebSocket este conceput pentru a fi pus în aplicare în browserele web și servere de web, dar poate fi utilizat de către orice client sau server de aplicații. Protocolul WebSocket este un protocol bazat pe TCP.

Server-ul ce utilizează WebSocket la fel ca și în cazul folosirii multicast crează un grup de utilizatori care sunt adăugați într-o listă, mesajele sunt trimise tuturor utilizatorilor înregistrați.

Serverul tratează trei tipuri de evenimente:

- OnOpen – eveniment ce apare la conectare;
- OnClose - eveniment ce apare la conectare;
- OnMessage – evenimentul ce apare la primirea unui mesaj.

Inițial am declarat lista ce va fi folosită pentru adăugarea clienților și adresa serverului (portul). Tratarea evenimentelor va avea loc prin canalul deschis WebSocketServer. În consola serverului la fel vor fi afișate mesajele din log, care cuprind mărimea mesajului primit și transmis și corpul mesajului, pentru a realiza această funcție am folosit Fleck care reprezintă o implimentare WebSocket în limbajul C#.

```
FleckLog.Level = LogLevel.Debug;  
var webSockets = new List<IWebSocketConnection>();  
var server = new WebSocketServer("ws://0.0.0.0:30000");
```

Aplicația folosește propriul protocol și tratează mesajele conform primului caracter primit:

- 1 – conectarea unui utilizator nou;
- 2 – deconectarea unui utilizator;
- / – mesaj privat de la un utilizator către altul.

La conectare are loc adăugarea utilizatorului în lista de utilizatori și formarea mesajului conform protocolului folosit, apoi mesajul este transmis fiecărui participant folosind instrucțiunea ForEach și afișarea mesajului în consolă.

Secvența ce se execută la conectarea unui utilizator poate fi găsită mai jos:

```

case "1":
    users.Add(data[1]);
    temp = "1" + "\0" + data[1] + "\0" + data[2] + "\0" + "Joins the net";
    webSockets.ToList().ForEach(s => s.Send(temp));
    Console.WriteLine(temp);
    break;

```

Pentru transmiterea unui mesaj privat indicăm / și numele utilizatorului cărui adresăm acest mesaj în caz de sunt 3 persoane în chat. În Figura de jos este prezentă partea codului ce execută asta:

```

case "/":
    int index = 0;
    int index1 = 0;
    for (int i = 0; i < users.Count; i++)
    {
        if (users[i].CompareTo(data[1]) == 0)
        {
            index = i;
            Console.WriteLine("Exists" + index);
        }
        else if (users[i].CompareTo(data[2]) == 0)
        {
            index1 = i;
        }
    }

    temp = "/" + "\0" + data[1] + "\0" + data[2] + "\0" + data[3] + "\0" + data[4] + "\0" + "PM from <" + data[2] + ">";
    webSockets.ElementAt(index).Send(temp);
    webSockets.ElementAt(index1).Send(temp);
    Console.WriteLine(temp);
    break;

```

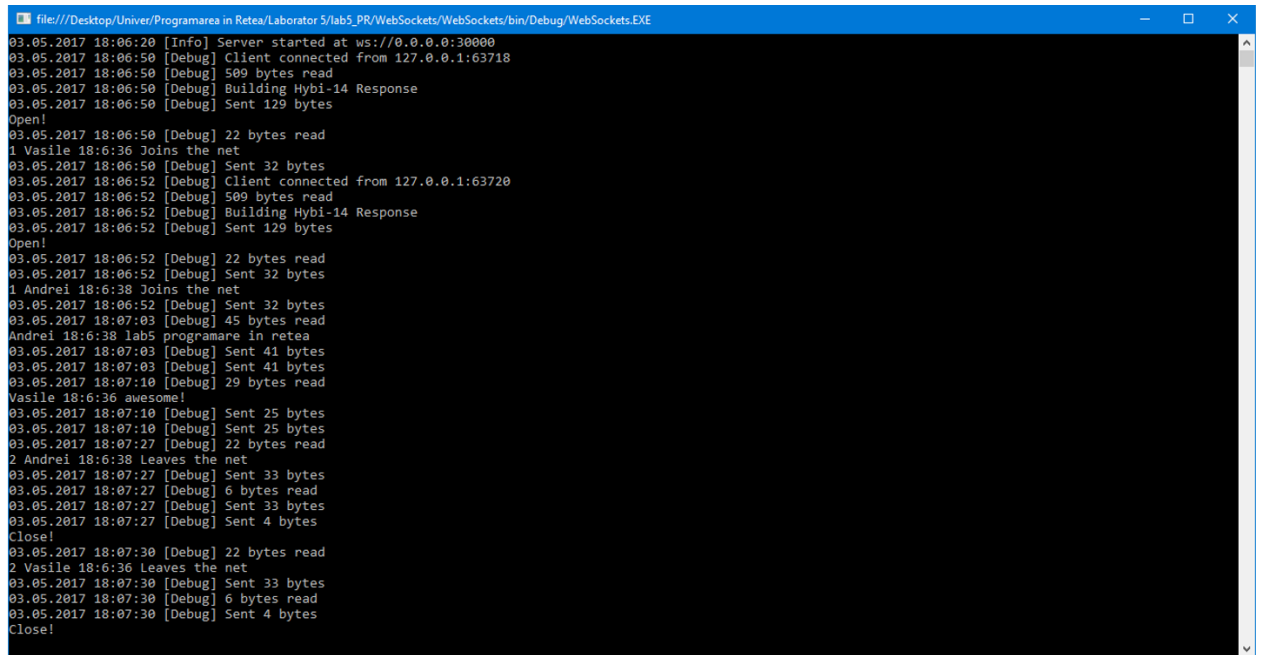
În codul ce urmează este arătată succesiunea de instrucțiuni executate la conectarea unui utilizator. Conform protocolului utilizat se extrage primul caracter în baza căruia se depistează ce fel de mesaj este primit, restul mesajului se împarte în părți și se afișează în chat.

```

ws.onmessage = function(e) {
    var now = new Date();
    var type = e.data.substring(0,1);
    var parts = e.data.split('\0');
    switch(type) {
        // user came online
        case "1":
            list.innerHTML += "<span class='user'>" + parts[1] + " : </span><span class='time'>" +
                parts[2] + " </span>" + parts[3] + "</br>";
            break;

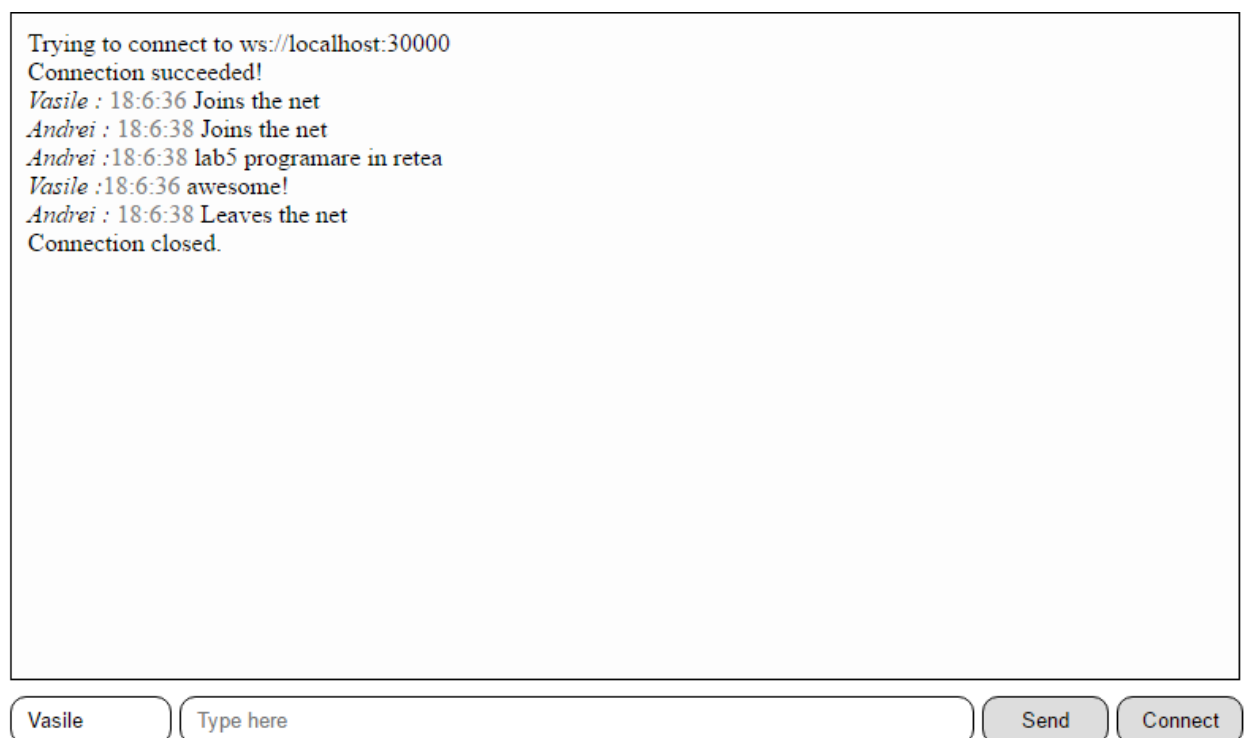
```

Rezultate



```
file:///Desktop/Univer/Programarea in Retea/Laborator 5/lab5_PR/WebSockets/WebSockets/bin/Debug/WebSockets.EXE
03.05.2017 18:06:20 [Info] Server started at ws://0.0.0.0:30000
03.05.2017 18:06:50 [Debug] Client connected from 127.0.0.1:63718
03.05.2017 18:06:50 [Debug] 509 bytes read
03.05.2017 18:06:50 [Debug] Building Hybi-14 Response
03.05.2017 18:06:50 [Debug] Sent 129 bytes
Open!
03.05.2017 18:06:50 [Debug] 22 bytes read
1 Vasile 18:6:36 Joins the net
03.05.2017 18:06:50 [Debug] Sent 32 bytes
03.05.2017 18:06:52 [Debug] Client connected from 127.0.0.1:63720
03.05.2017 18:06:52 [Debug] 509 bytes read
03.05.2017 18:06:52 [Debug] Building Hybi-14 Response
03.05.2017 18:06:52 [Debug] Sent 129 bytes
Open!
03.05.2017 18:06:52 [Debug] 22 bytes read
03.05.2017 18:06:52 [Debug] Sent 32 bytes
1 Andrei 18:6:38 Joins the net
03.05.2017 18:06:52 [Debug] Sent 32 bytes
03.05.2017 18:07:03 [Debug] 45 bytes read
Andrei 18:6:38 lab5 programare in retea
03.05.2017 18:07:03 [Debug] Sent 41 bytes
03.05.2017 18:07:03 [Debug] Sent 41 bytes
03.05.2017 18:07:10 [Debug] 29 bytes read
Vasile 18:6:36 awesome!
03.05.2017 18:07:10 [Debug] Sent 25 bytes
03.05.2017 18:07:10 [Debug] Sent 25 bytes
03.05.2017 18:07:27 [Debug] 22 bytes read
2 Andrei 18:6:38 Leaves the net
03.05.2017 18:07:27 [Debug] Sent 33 bytes
03.05.2017 18:07:27 [Debug] 6 bytes read
03.05.2017 18:07:27 [Debug] Sent 33 bytes
03.05.2017 18:07:27 [Debug] Sent 4 bytes
Close!
03.05.2017 18:07:30 [Debug] 22 bytes read
2 Vasile 18:6:36 Leaves the net
03.05.2017 18:07:30 [Debug] Sent 33 bytes
03.05.2017 18:07:30 [Debug] 6 bytes read
03.05.2017 18:07:30 [Debug] Sent 4 bytes
Close!
```

Figura 1 - Server



```
Trying to connect to ws://localhost:30000
Connection succeeded!
Vasile : 18:6:36 Joins the net
Andrei : 18:6:38 Joins the net
Andrei :18:6:38 lab5 programare in retea
Vasile :18:6:36 awesome!
Andrei : 18:6:38 Leaves the net
Connection closed.
```

Vasile

Type here

Send

Connect

Figura 2 – Chat

Concluzii

În urma realizării acestei lucrări de laborator am studiat noi abordări pentru comunicare în spațiul Web prin intermediul WebSockets. Pentru a realiza sarcina propusă am creat o aplicație client-server care reprezintă un chat simplu cu posibilitatea de conectare, deconectare și trimiterea mesajelor public și private către toți utilizatorii chatului conectați la server.