BÁO CÁO THỰC HÀNH

**Môn học: Lập trình mạng căn bản**

**Kỳ báo cáo: Buổi 03 (Session 03)**

**Tên chủ đề: Lập trình Sockets trong C#**

*GVHD: Nghi Hoàng Khoa*

*Ngày báo cáo: 17/11/2022*

**Nhóm: ĐN (ghi số thứ tự nhóm)**

1. **THÔNG TIN CHUNG:**

*(Liệt kê tất cả các thành viên trong nhóm)*

Lớp: NT106.M21.ATCL

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Email** |
| 1 | Hoàng Văn Anh Đức | 20520890 | 20520890@gm.uit.edu.vn |
| 2 | Bùi Thị Trúc Nhạn | 20521692 | 20521692@gm.uit.edu.vn |

1. **NỘI DUNG THỰC HIỆN:[[1]](#footnote-1)**

|  |  |  |
| --- | --- | --- |
| **STT** | **Công việc** | **Kết quả tự đánh giá** |
| 1 | Bài 1: Kết nối UDP | 100% |
| 2 | Bài 2: Kết nối TCP | 100% |
| 3 | Bài 3: TCP Client – TCP Listener | 100% |
| 4 | Bài 4: TCP Client – TCP Listener | 100% |

**Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.**

BÁO CÁO CHI TIẾT

\*Chú ý : Các dòng code đã được chú thích trong ảnh chụp

## Bài 1: Kết nối UDP

Viết ứng dụng thực hiện gởi và nhận dữ liệu dữ liệu giữa hai bên sử dụng giao thức UDP (UDP Client và UDP Server). Người dùng ở Client sẽ chỉ định IP, port cần kết nối và thông điệp gửi đến Server. Tại Server sẽ nhận được thông điệp gửi từ Client

***Bước 1:*** Thiết kế phần giao diện

Graphical user interface, application

Description automatically generated

Giao diện của Server

- Trong Server: Giao diện gồm có

* 2 Label có tên “Port” và “Received message”

“Port”: Nhập “port” ở “textBox2” mà server sẽ lắng nghe và nhận tin nhắn thông qua “port ” đó từ client

“Received message”: Tin nhắn server nhận được từ Client sẽ được hiển thị trên “ListView1”

* 1 button có tên “Listen”: để “lắng nghe” tin nhắn của Client thông qua “Port” mà ta nhập vào ở “textBox2

+ Tạo hàm **serverThread** để đón nhận dữ liệu từ bên Client gởi sang và hiện lên **ListView1**

Text

Description automatically generated

+ Trong **button1** tạo một **thread** để chạy hàm nhận dữ liệu từ bên gởi là Client

Text

Description automatically generated

- Thư viện cần thêm trong Server:

+ using System.Net

+ using System.Net.Sockets

+ using System.Threading

Text

Description automatically generated

**Thư viện** cần dùng trong **Server**

***Bước 2:*** Tạo Client

- Giao diện của Client có :

* 3 Label có tên là “IP RemoteHost”, “Port” và “Message”
* 3 textBox
* “txbHost”: để nhập địa chỉ IP
* “txbPort”: để nhập vào “Port” mà tin nhắn sẽ gửi qua port này đến server ( 2 port của Server và Client phải giống nhau )
* “txbMessage”: Nhập tin nhắn mà muốn gửi đến server
* 1 button có tên là “Send”: để gửi “Message” vừa nhập ở txbMessage sang cho server

Graphical user interface, application

Description automatically generated

Giao diện của **Client**

- **Button** “**Send**”

Text

Description automatically generated

Sự kiện cho **nút Send**

**-** Thư viện cần có:

Text

Description automatically generated

Thư viện trong Client

Demo :

Graphical user interface, application

Description automatically generated

**Nhập Port** của server và **nhấn Listen** để lắng nghe client

Graphical user interface

Description automatically generated

**Nhập IP, Port** và **Message** muốn gửi cho Server và **nhấn Send** để gửi cho Server

Kết quả :

Graphical user interface, application

Description automatically generated

**Server** nhận dược tin nhắn và hiển thị lên **listView1**

## Bài 2: Kết nối TCP

## Viết chương trình lắng nghe dữ liệu từ dịch vụ Telnet sử dụng kết nối TCP (sử dụng lớp Socket) với mô tả sau:

1. Chạy chương trình
2. Nhấn nút Listen
3. Mở CMD gõ lệnh: telnet <IP của máy> 8080
4. Vào màn hình telnet, gõ thông điệp tuỳ ý, chương tình sẽ nhận và hiện lên form
5. Xem hình mẫu

Graphical user interface, application

Description automatically generated

***Bước 1***: Tạo TCP Server lắng nghe kết nối

- Giao diện cần có: 1 button có tên là “btnListen” và 1 ListView “lsvMessage”

Graphical user interface, application

Description automatically generated

Giao diện bài 2

- Để telnet có thể đến địa chỉ IP của máy tại cổng 8080, ta phải mở port 8080 lắng nghe kết nối TCP đến port 8080, do đó khi nhấn nút “btnListen” có nghĩa là ta đang thực hiện lắng nghe kết nối tại địa chỉ IP của máy và cổng là 8080

Text

Description automatically generated

Sự kiện cho **btnListen**

Text

Description automatically generatedText

Description automatically generated

Hàm **StartUnsafeThread()**

- ***Bước 2***: Chạy chương trình

+ Chạy server và nhấn “btnListen” để lắng nghe dịch vụ telnet

Graphical user interface

Description automatically generated

+ Mở cmd để kết nối đến Server

Background pattern

Description automatically generated

Kết nối thành công

Graphical user interface, application

Description automatically generated

Gửi tin nhắn đến Server

Text

Description automatically generated

Server nhận được tin nhắn

Graphical user interface

Description automatically generated

- Thư viện cần dùng:

+ using System.Net;

+ using System.Net.Sockets;

+ using System.Threading;

Text

Description automatically generated

## Bài 3: TCP Client – TCP Listener

Viết ứng dụng thực hiện gởi và nhận dữ liệu sử dụng giao thức TCP (TCP Client và TCP Listener). Server lắng nghe kết nối và thông điệp từ Client

1. Chạy Server
2. Nhấn nút Listen
3. Khởi tạo Client
4. Gửi thông điệp từ Client đến Server
5. Server nhận thông điệp và hiện lên form
6. Xem hình mẫu:

Graphical user interface

Description automatically generated

* ***Bước 1:*** Thiết kế giao diện cho Server và client

Graphical user interface, application

Description automatically generated Graphical user interface, application, PowerPoint

Description automatically generated

Tạo Form **Server** để lắng nghe những gì của **Client**

* ***Bước 2:*** Tiến hành code

+ Giao diện Client cần có: 1 button có tên “btnSend” để gửi thông điệp từ Client đến Server

Graphical user interface, application

Description automatically generated

Giao diện form **Form1** (Client)

+ Cách tạo Client :

* Đầu tiên tạo 1 đối tượng TCP Client

Text

Description automatically generated

Tạo đối tượng

+ Tạo hàm **ServerThread()** để :

1. Kết nối đến Server với địa chỉ và port xác định với phương thức **tcpClient.Connect()**
2. Lấy luồng (stream) giao tiếp bằng phương thức **tcpClient.GetStream()**
3. Thực hiện giao tiếp với server

Text

Description automatically generated

Hàm **ServerThread()**

+ Trong nút **button** “btnSend” tạo Thread để thực thi hàm **ServerThread()** :

Text

Description automatically generated

Sự kiện nút “**btnSend”**

* Thư viện trong **Client** cần có:

+ using System.Net.Sockets;

+ using System.Threading.Tasks;

Text

Description automatically generated

* ***Bước 4:*** Tạo  **Server**

+ Giao diện cần có: **1 button** có tên “btnListen” để Server lắng nghe thông điệp từ **Client** và hiện lên **lsvMessage**

Graphical user interface, application, Word

Description automatically generated

Hình 35: Giao diện form Server

Text

Description automatically generated

Thông báo server có interfaces với ip là : 127.0.0.1 và port là : 8080

+ Hàm xử lý ( StartUnsafeThread() )

Text

Description automatically generated

Hàm **StartUnsafeThread()**

Graphical user interface, text

Description automatically generated

Hàm **AddMessage** để thêm string vào **lsvMessage**

Text

Description automatically generated

Sự kiện nút **button**

* Thư viện cần có:

+ using System.Net.Sockets;

+ using System.Threading.Task;

Text

Description automatically generated

Thư viện form **Server**

***Bước 5 :*** Thực thi

+Chạy Server và nhấn “btnListen” : Server sẽ bắt đầu lắng nghe với IPEndPoint là 127.0.0.1 và port là 8080

Graphical user interface, application, Word

Description automatically generated

+Chạy Client và nhấn “btnSend” liên tục để có thể gửi nhiều tin nhắn đến server

Graphical user interface, application

Description automatically generated

+Và đây là kết quả :

Graphical user interface, application, Word

Description automatically generated

## Bài 4: TCD Client – TCP Listener

Viết chương trình Chat Room/ Gửi và nhận dữ liệu sử dụng TCP Client và TCP Listener. Mỗi người dùng sẽ có một tài khoản, khi một người dùng gửi tin nhắn thì tất cả mọi người còn lại đều sẽ nhận được tin nhắn đó

* ***Bước 1:*** Thiết kế giao diện của Client và Server

Graphical user interface, application, Word

Description automatically generated Graphical user interface, application, Word

Description automatically generated

* ***Bước 2:*** Tạo **Client**

+ Giao diện cần có:

* 1 listView để hiện tin nhắn của **Server** và các **client** khác
* 2 Label có tên “Your name” và “Message”
* 2 textbox ứng với 2 label : “txbName” để xác định tên của người gửi , “txbMessage” để nhập nội dung cần gửi
* 1 button có tên “btnSend” để nhấn gửi đến Server và tất cả các client

* Thư viện cần dùng:

+ using System.Net.Sockets;

+ using System.Threading.Tasks;

+ using System.Runtime.Serialization.Formatters.Binary;

+ using System.IO;

Text

Description automatically generated

Khi chạy chương trình của **Client** thì **Client** sẽ cố gắng kết nối với **Server** :

Text

Description automatically generated

Text

Description automatically generated

Tạo 1 **Socket client** có ip : 127.0.0.1 và port :8080 để kết nối tới server

Kết nối bằng **client.Connect()**

Trong hàm **Connect()** còn có cả **Thread(Receive)** để tạo luồng lắng nghe khi vừa kết nối tới **Server .**

Text

Description automatically generated

Hàm **Receive()** để nhận các thông tin từ **Server** hoặc các **Client** khác, khi các **Server** hoặc **Client** khác gửi tin thì sẽ có hàm **Serialize()** để phân mảnh dữ liệu từ 1 mảng byte sang 1 object và hàm **Deserialize()** sẽ gom mảnh lại thành 1 chuỗi byte trước khi **Client** nhận được

A screenshot of a computer

Description automatically generated with medium confidence

Phân mảnh 1 object thành 1 chuỗi byte để có thể gửi cho **Server** và các **Client** khác

Text

Description automatically generated

Hàm **Deserialize()** sẽ thực hiện gom mảnh các object đã đc phân mảnh lại thành 1 chuỗi byte để **Server** và các **Client** khác có thể nhận được

Text

Description automatically generated

Hàm **AddMessage()** để gửi thông tin từ các Server hoặc client khác lên trên **lsvMessage**

Text

Description automatically generated

Ta có sự kiện **btnSend()** để gửi dữ liệu đến các **Client** và **Server** khác

Text

Description automatically generated

Hàm **Send()** sẽ kiểm tra xem “**txbMessage”** , nếu không rỗng thì khi nhấp btnSend thì dữ liệu sẽ được gửi đến các **Client** và **Server** khác

***Bước 3***:Tạo Server

Text

Description automatically generated

Khi chạy **Server** thì **Server** cũng cố gắng kết nối tới các **Client**

Text

Description automatically generated

Text

Description automatically generated

Ở phần **Server** chúng ta phải tạo 1 List các **Client :**Sau khi connect đến **Client** thì sẽ sẽ bắt đầu lắng nghe tất cả các **Client .** Nếu “lắng nghe” thành công thì sẽ thêm Client vào **ClientList .**Trong khi đó thì ta cũng sẽ tạo luồng lắng nghe xem các **Client** có gửi gì đến không .Ta dùng Try catch để khi kết nối đến **client** mà có 1 **client** disconnect thì **server** sẽ chạy vòng lặp while liên tục để chương trình ko bị crash

Text

Description automatically generated

Ta dùng try catch để Remove các thành client phát sinh lỗi khi **Server** đang lắng nghe thông tin từ các **Client** .Đầu tiên ta khai báo 1 **Socket** để nhận dữ liệu => Khởi tạo mảng byte để **Socket** đó nhận dữ liệu => Chuyển dữ liệu đó sang dạng string

Ta dùng Forech để gửi **Message** nhận được này đến các **Client** và hiển thị message nhận được lên trên **lsvMessage**

Hàm **Serialize()** và **Deserialize()** của **Server** tương tự như của **Client**

***Mở rộng :***

Text

Description automatically generated

Sự kiện **btnSend** để gửi thông tin cho các **Client** trong **ClientList .** Duyệt từng phần tử trong **ClientList** và gửi thông tin nhập vào trong **“txbMessage”** của **Server** và gửi nó đến các **Client**

Text

Description automatically generated

Trong hàm **Send()** của Server thì tham số truyền vào sẽ là **Socket** tên **Client** .Kiểm tra nếu Socket đó khác null và **txbMessage** khác rỗng thì phân mảnh và gửi đến các **Client**

***DEMO:***

Graphical user interface, application

Description automatically generated

Khi chạy **Server** thì **Server** sẽ lắng nghe xem có **Client** nào kết nối tới nó không .Khi ta chạy 2 **client** thì bên **Server** có thông báo kết nối thành công

Graphical user interface, application

Description automatically generated

Khi **Client** đầu tiên gửi tin nhắn thì cả **Server** và các **client** khác cũng nhận được thông qua việc **Client** này gửi đến **Server** và **Server** sẽ gửi tới các **Client** khác (**Client** còn lại).Khi gửi thì message sẽ được phân mảnh và tới đích sẽ được gom mảnh lại .

Graphical user interface, application

Description automatically generated

Khi Client còn lại gửi thì cả Server và Client đầu tiên cũng nhận đc

***Mở rộng :***

Graphical user interface, application

Description automatically generated

Khi **Server** gửi thì các **Client** cũng sẽ nhận được tin nhắn

***Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này***

20521692 - Hoàng Văn Anh Đức

20521692 - Bùi Thị Trúc Nhạn

# **YÊU CẦU CHUNG**

* Sinh viên tìm hiểu và thực hành theo hướng dẫn.
* Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
* Sinh viên báo cáo kết quả thực hiện và nộp bài.

**Báo cáo:**

* File .PDF. Tập trung vào nội dung, không mô tả lý thuyết.
* Nội dung trình bày bằng Font chữ Times New Romans/ hoặc font chữ của mẫu báo cáo này (UTM Neo Sans Intel/UTM Viet Sach)– cỡ chữ 13. Canh đều (Justify) cho văn bản. Canh giữa (Center) cho ảnh chụp.
* Đặt tên theo định dạng: [Mã lớp]-SessionX\_GroupY. (trong đó X là Thứ tự buổi Thực hành, Y là số thứ tự Nhóm Thực hành đã đăng ký với GVHD-TH).

*Ví dụ: [*NT101.K11.ANTT*]-Session1\_Group3.*

* Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
* Không đặt tên đúng định dạng – yêu cầu, sẽ **KHÔNG** chấm điểm bài Lab.
* Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

**Đánh giá**: Sinh viên hiểu và tự thực hiện được bài thực hành. Khuyến khích:

* Chuẩn bị tốt.
* Có nội dung mở rộng, ứng dụng trong kịch bản phức tạp hơn, có đóng góp xây dựng bài thực hành.

*Bài sao chép, trễ, … sẽ được xử lý tùy mức độ vi phạm.*

**HẾT**

1. Ghi nội dung công việc, các kịch bản trong bài Thực hành [↑](#footnote-ref-1)