-------------------------------------------------------Học mạng máy tính-----------------------------------------------------

+, Kiến thức cần có: API Protocal.

+, Kiến thức về các giao thức mạng.

+, Mục tiêu môn học xây dựng một API.

+, Ứng dụng sử dụng Socket với các giao thức.

---Mục tiêu học buổi 1-----

+, Tìm hiểu thư viện System.IO.

Thư viện System.IO trong .NET là một thư viện chuẩn được cung cấp bởi Microsoft, cung cấp các lớp và phương thức để làm việc với hệ thống tệp và đường ống (stream) trong ứng dụng .NET.

Các lớp trong System.IO cho phép bạn đọc, ghi, sao chép và xóa các tệp và thư mục, truy cập các thông tin tệp và thư mục (như tên, đường dẫn, kích thước và thời gian tạo), đọc và ghi các đối tượng dạng văn bản hoặc nhị phân vào các đường ống (stream), và thực hiện các thao tác I/O khác.

Một số lớp chính trong System.IO bao gồm:

File: Cho phép thao tác với các tệp trên hệ thống tệp.

Directory: Cho phép thao tác với các thư mục trên hệ thống tệp.

Path: Cung cấp các phương thức để xử lý các chuỗi đường dẫn tệp và thư mục.

Stream: Cung cấp các lớp để đọc và ghi dữ liệu theo dạng byte.

StreamReader và StreamWriter: Cung cấp các lớp để đọc và ghi dữ liệu theo dạng văn bản.

System.IO là một thư viện quan trọng trong lập trình .NET và được sử dụng rộng rãi trong các ứng dụng desktop, web và mobile.

+, Tìm hiểu 2 địa chỉ mạng có trong chat GPT.

0.0.0.0:9000

Địa chỉ 0.0.0.0:9000 là một địa chỉ IP và cổng kết nối mạng, thường được sử dụng để chỉ định một máy tính có thể lắng nghe kết nối từ mọi địa chỉ IP khác nhau trên cùng một cổng. Điều này có nghĩa là máy tính được cấu hình để chấp nhận kết nối đến cổng 9000 từ bất kỳ địa chỉ IP nào trên mạng.

Router mạng tính dẫn đường trong nó có các thuật toán định tuyến, giúp tìm được đi ngắn nhất.

Hork có từ lọa holk 2laye có interface.

Máy tính đang sử dụng mô hình TCP được tham chiếu từ OSI

Địa chỉ IP 0.0.0.0 thường được sử dụng để đại diện cho tất cả các địa chỉ IP có thể trên mạng, vì vậy khi một máy tính lắng nghe kết nối đến địa chỉ này trên một cổng nhất định, nó sẽ lắng nghe kết nối từ tất cả các địa chỉ IP trên mạng.

Ví dụ, nếu một ứng dụng web được cấu hình để lắng nghe kết nối đến địa chỉ 0.0.0.0:9000, nó sẽ chấp nhận kết nối từ bất kỳ máy tính nào trên mạng có thể kết nối đến địa chỉ IP của máy tính này trên cổng 9000.

Giải thích cho tôi địa chỉ mạng: 127.0.0.1

Địa chỉ mạng 127.0.0.1 là địa chỉ IP loopback, hay còn được gọi là địa chỉ loopback hoặc localhost. Đây là một địa chỉ đặc biệt được sử dụng để gửi và nhận dữ liệu trên cùng một máy tính mà không cần kết nối tới bất kỳ mạng ngoài nào.

Khi một ứng dụng hoặc dịch vụ trên máy tính yêu cầu kết nối đến chính nó (ví dụ: khi bạn chạy một server web trên máy tính của mình), thì ứng dụng đó có thể sử dụng địa chỉ mạng 127.0.0.1 để thực hiện kết nối với chính nó.

Khi bạn gõ "http://127.0.0.1" hoặc "http://localhost" vào trình duyệt web của mình, nó sẽ được chuyển đến máy tính của bạn thông qua địa chỉ loopback, và bạn sẽ nhìn thấy trang web được chạy trên máy tính của bạn.

Địa chỉ mạng 127.0.0.1 là một địa chỉ mạng cục bộ và chỉ có thể được sử dụng trên máy tính của bạn. Không thể sử dụng địa chỉ này để kết nối đến một máy tính khác trên mạng.

Tìm hiểu IIS là gì.

**Tìm hiểu luồng.**

Luồng (hay còn gọi là thread) là một khái niệm trong lập trình và hệ điều hành, nó đề cập đến cách thức thực thi đa nhiệm trong hệ thống máy tính. Luồng là một đơn vị nhỏ nhất của thực thi mà có thể được lên lịch để chạy trên một bộ xử lý (processor) độc lập, với khả năng chia sẻ tài nguyên của hệ thống.

---------------------------------------------------------Giao thức tcp--------------------------------------------------------

Giao thức TCP (Transmission Control Protocol) là một trong hai giao thức cơ bản trong giao thức Internet (IP) để truyền tải dữ liệu qua mạng. Giao thức TCP cung cấp một kênh truyền tin cậy giữa các ứng dụng trên các thiết bị mạng khác nhau.

Giao thức TCP hoạt động bằng cách chia các dữ liệu thành các gói tin (packet) và gửi chúng qua mạng đến đích. Trong quá trình này, TCP đảm bảo rằng các gói tin được gửi đến đúng địa chỉ và đúng thứ tự, đồng thời kiểm tra tính toàn vẹn và độ tin cậy của dữ liệu bằng cách sử dụng các thông số bảo mật.

Giao thức TCP sử dụng một số cơ chế để đảm bảo tính tin cậy của dữ liệu, bao gồm:

Kiểm tra tổng kiểm tra (checksum): TCP sử dụng thuật toán kiểm tra tổng kiểm tra để đảm bảo tính toàn vẹn của các gói tin được gửi đi.

Cơ chế tái truyền (retransmission): Nếu một gói tin không được nhận đúng cách hoặc bị mất trong quá trình truyền tải, TCP sẽ tái gửi lại gói tin đó cho đến khi nó được nhận đúng cách.

Xác định số thứ tự (sequence numbering): TCP sử dụng số thứ tự để đảm bảo rằng các gói tin được gửi đi theo đúng thứ tự và không bị trùng lặp.

Quản lý luồng (flow control): TCP sử dụng cơ chế quản lý luồng để điều chỉnh tốc độ truyền dữ liệu giữa các máy tính, đảm bảo rằng các máy tính có tốc độ truyền dữ liệu tương đương

Tóm lại, giao thức TCP là một giao thức truyền tải dữ liệu tin cậy, đảm bảo tính toàn vẹn, tin cậy và đúng thứ tự của các gói tin được truyền tải qua mạng.

--------------------------------------------------------Giao thức UDP----------------------------------------------------------

Giao thức UDP (User Datagram Protocol) là một trong hai giao thức chính trong mô hình truyền thông mạng TCP/IP, bên cạnh giao thức TCP (Transmission Control Protocol). UDP là giao thức truyền dữ liệu không đồng bộ và không đảm bảo tính toàn vẹn dữ liệu, không đảm bảo đến được hoặc không đảm bảo thứ tự đến của các gói tin.

UDP được sử dụng chủ yếu trong các ứng dụng truyền thông mạng như streaming video, trò chơi trực tuyến, VoIP, và các ứng dụng tương tự, nơi mà một chút mất mát dữ liệu có thể chấp nhận được và tính năng truyền dữ liệu nhanh là ưu tiên hơn tính toàn vẹn dữ liệu.

UDP không sử dụng các bước xác thực, xác nhận, kiểm soát lỗi hay phục hồi lỗi như TCP, điều này giúp giảm độ trễ và độ trễ dao động trong quá trình truyền dữ liệu. Tuy nhiên, điều này đồng nghĩa với việc nếu có lỗi xảy ra trong quá trình truyền, dữ liệu sẽ bị mất mát hoặc sai sót. Do đó, việc sử dụng UDP hay TCP phụ thuộc vào tính chất và yêu cầu của ứng dụng cụ thể.

---------------------------------------------------------------CODE--------------------------------------------------------------------

data = arr.SelectMany(BitConverter.GetBytes).ToArray(); Đoạn này giúp chuyển một arr thành một mảng byte

int[] arr = new int[data.Length / 4];//// Mảng int có chiều dài là 1/4 chiều dài của mảng byte

Buffer.BlockCopy(data, 0, arr, 0, data.Length); Giúp chuyển mảng byte thành arr,

Hoành thành ôn tập sử dụng tcp.

----------------------------------------------------------Sử dụng UDP-------------------------------------------------------------

Console.OutputEncoding = Encoding.Unicode;

Console.InputEncoding = Encoding.Unicode;

IPAddress iPAddress = IPAddress.Any;

IPEndPoint iPEndPoint = new IPEndPoint(IPAddress.Any, 2023);

Socket socket\_server = new Socket(AddressFamily.InterNetwork, SocketType.Dgram, ProtocolType.Udp);

socket\_server.Bind(iPEndPoint);

Console.WriteLine("Server {0} waiting...", iPEndPoint);

IPEndPoint remoteIpendpoint = new IPEndPoint(IPAddress.Any, 0);

EndPoint endpoint = (EndPoint)remoteIpendpoint;

byte[] data = new byte[1024];

int recv = socket\_server.ReceiveFrom(data, ref endpoint);

string s = Encoding.Unicode.GetString(data, 0, recv);

// in Thông Báo địa chỉ của clent kết nối

Console.WriteLine("Nhận Kết Nối Từ: {0}", s);

data = Encoding.Unicode.GetBytes("Xin chào client");

socket\_server.SendTo(data, endpoint);

while (true)

{

data = new byte[1024];

recv = socket\_server.ReceiveFrom(data, ref endpoint);

s = Encoding.Unicode.GetString(data, 0, recv);

if (s.ToUpper().Equals("QUIT"))

{

break;

}

Console.WriteLine(s);

data = new byte[1024];

data = Encoding.Unicode.GetBytes(s.ToUpper());

socket\_server.SendTo(data, 0, data.Length, SocketFlags.None, endpoint);

}

Console.ReadKey();

socket\_server.Close();

Client: Console.InputEncoding = Encoding.UTF8;

Console.OutputEncoding = Encoding.UTF8;

IPAddress iPAddress = IPAddress.Parse("127.0.0.1");

IPEndPoint iPEndPoint = new IPEndPoint(iPAddress, 2023);

Socket socket\_client = new Socket(AddressFamily.InterNetwork, SocketType.Dgram, ProtocolType.Udp);

string s = "Hello Server!...";

// tao endpoint nhan du lieu tu server

EndPoint endPoint = (EndPoint)iPEndPoint;

byte[] data = new byte[1024];

data = Encoding.Unicode.GetBytes(s);

socket\_client.SendTo(data, iPEndPoint);

// tao endpoint nhan data tu server

EndPoint endpoint = (EndPoint)iPEndPoint;

data = new byte[1024];

int rec = socket\_client.ReceiveFrom(data, ref endPoint);

s = Encoding.Unicode.GetString(data, 0, rec);

Console.WriteLine("Nhan Ve Tu SerVer: " + s);

while(true)

{

s = Console.ReadLine();

data = new byte[1024];

data = Encoding.Unicode.GetBytes(s);

socket\_client.SendTo(data, endpoint);

if (s.ToUpper().Equals("QUIT"))

{

break;

}

data = new byte[1024];

rec = socket\_client.ReceiveFrom(data, ref endpoint);

s = Encoding.Unicode.GetString(data, 0, rec);

Console.WriteLine(s);

}

Console.ReadKey();

socket\_client.Close();

Thằng này khác là tcp là, ở client không cần trực tiếp connect. Và gửi dữ liệu là Sendto(data, ipendpoint), nhận dữ liệu là ReceiveFrom(<mảng byte nhận>, ref end Point);

Mã đọc data ra sẽ là Unicode. Còn lại là giống.