5. Lớp luồng NetworkStream

NetworkStream là một lớp con của không gian System.Net.Sockets. Nó làm việc qua các socket.

Luồng NetworkStream có bản chất khác biệt với luồng FileStream. Luồng FileStream cho phép truy xuất ngẫu nhiên, trong khi luồng NetworkStream chỉ được phép truy xuất tuần tự.

Luồng NetworkStream không hỗ trợ bộ đệm để làm việc với các ứng dụng có lượng dữ liệu trung gian lớn trong khi truyền gửi. Cần kết hợp thêm với lớp luồng bộ đệm.

Tiết học này sẽ giới thiệu lớp luồng NetworkStream nhưng dành chút thời gian để giới thiệu về lớp hỗ trợ bộ đệm BufferedStream.

Lớp luồng BufferedStream

Bộ đệm là một không gian nhớ dành để lưu trữ tạm thời các dữ liệu nhằm mục đích cải thiện tốc độ làm việc của ứng dụng. Nó còn để đồng bộ hóa giữa các thiết bị có các tốc độ truyền nhận khác nhau.

Khi tạo mới một đối tượng bộ đệm, kích thước nhầm định của nó là 4096 byte. Tuy nhiên ra có thể thay đổi kích thước bộ đệm bằng nhiều phương thức.

Phương thức sau đọc dữ liệu từ luồng Stream và lưu vào bộ đệm. Sau cùng, nó đọc bộ đệm và in ra màn hình. Phhương thức này có bao gói riêng một luồng Stream có tên là st.

```
// Đọc BufferedStream
public static void readBufStream(Stream st)
{
   // Chuẩn bị bộ đệm BufferedStream
   BufferedStream bf = new BufferedStream(st);
   byte[] inData = new Byte[st.Length];
   // Đọc và hiện thị dữ liệu đệm
   bf.Read(inData, 0, Convert.ToInt32(st.Length));
   Console.WriteLine(Encoding.ASCII.GetString(inData));
}
```

Lớp luồng NetworkStream

Lớp luồng NetworkStream có một số thuộc tính quan trọng sau đây.

Thuộc tính	Mô tả
DataAvailable	Giá trị Boolean true được trả về báo hiệu dữ liệu trong luồng đã sẵn sàng để đọc.
Readable	Giá trị Boolean dùng để truy vấn hay thiết lập quyền đọc luồng. Nó hoạt động giống như thuộc tính CanRead trong các luồng khác.
Socket	Chio biết socket đang sử dụng cho kết nối
Writeable	Giá trị Boolean dùng để truy vấn luồng có cho phép ghi hay không. Nó hoạt động giống như thuộc tính CanWrite trong các luồng khác

Mỗi constructor NetworkStream cần có ít nhất một socket. Ngoài ra, có thể sử dụng một giá trị Boolean để chỉ thị về quyền sở hữu luồng, và có thể thêm một giá trị nữa của hàm liệt kê FileAccess để kiểm soát đọc/ghi luồng.

Đoạn mã sau đây minh họa về lớp luồng mạng NetworkStream trong một ứng dụng TCP listener đơn giản. Lớp TCP lấy trong không gian System.Net.Sockets.

```
using System;
using System.IO;
using System.Text;
using System.Net;
using System.Net.Sockets;

class TCPListenerDemo
{
   public static void Main()
   {
      try
      {
// Tạo mới TCP listener
        TcpListener listener = new TcpListener(5001);
        listener.Start();
        TcpClient tc = listener.AcceptClient();
        NetworkStream stm = tc.GetStream();
```

Ở đoạn mã trên, trước hết, ta tạo đối tượng TcpListener để theo dõi cổng 5001. Đối tượng này được khởi động công việc bằng phương thức Start(). Đối tượng này thi hành phương thức

AcceptClient() để chấp nhận một kết nối đến từ phía client bằng cách trả về cho TcpClient. Sau cùng, phương thức GetStream() để tạo luồng NetworkStream cho đối tượng tc của TcpClient với tên luồng là stm. Từ bây giờ có thể sử dụng phương thức Read() để đọc dữ liệu giống như các luồng khác:

```
         byte[] readBuf = new byte[100];
         stm.Read(readBuf,0,100);
         //hiển thị dữ liệu
         Console.WriteLine(Encoding.ASCII.GetString(readBuf));
         stm.Close();
      }
      catch (Exception e) { Console.WriteLine(e.ToString()); }
   }
}
```

Ứng dụng TcpListener trên đây cần thi hành cùng với một ứng dụng client mà nó sẽ gửi dữ liệu tới. Sau đây là mã chương trình TcpClient.

```
using System;
using System.IO;
using System.Text;
using System.Net;
using System.Net.Sockets;
class TcpClient Example
{
   static void Main(string[] args)
   {
      try
      {
        // tạo mới TCP Client
        TcpClient client = new TcpClient();
        //Kết nối với hostname và port: host cục bộ và cổng 5001
        client.Connect ("localhost", 5001);
        //Tạo luồng NetworkStream để gửi dữ liệu
        NetworkStream stm = client.GetStream();
        byte[] sendBytes = Encoding.ASCII.GetBytes("This data has come from" +
                                                   " another place!!!");
        stm.Write (sendBytes, 0, sendBytes.Length);
         client.Close();
```

```
      }
      catch (Exception e)
      {
         Console.WriteLine(e.ToString());
         Console.WriteLine("The listener has probably not started");
      }
   }
}
```

Ghi nhớ: chạy ứng dụng Listener trước khi chạy ứng dụng client