EAST ASIA UNIVERSITY OF TECHNOLOGY

LẬP TRÌNH MẠNG
(Network Programming)

Managed I/O - Streams, Readers, and Writers

Nguyễn Anh Thơ
natho5578@gmail.com

Nội dung

Tuần	Nội dung	
01	Streams là gì?	
02	Các kiểu Stream	
03	Phương thức và thuộc tính stream	
04	Các dạng dữ liệu stream	
05	Đọc ghi với các dạng dữ liệu stream	
06	Bài tập	

Streams (Net) là gì?

- Streams sử dụng truy cập vào tài nguyên của hệ thống như: file, bộ nhớ, tài nguyên mạng
- Streams là một lớp trong .NET Framework sử dụng quản lý IO
- Streams cung cấp phương thức đọc ghi dữ liệu vào ổ địa lưu trữ

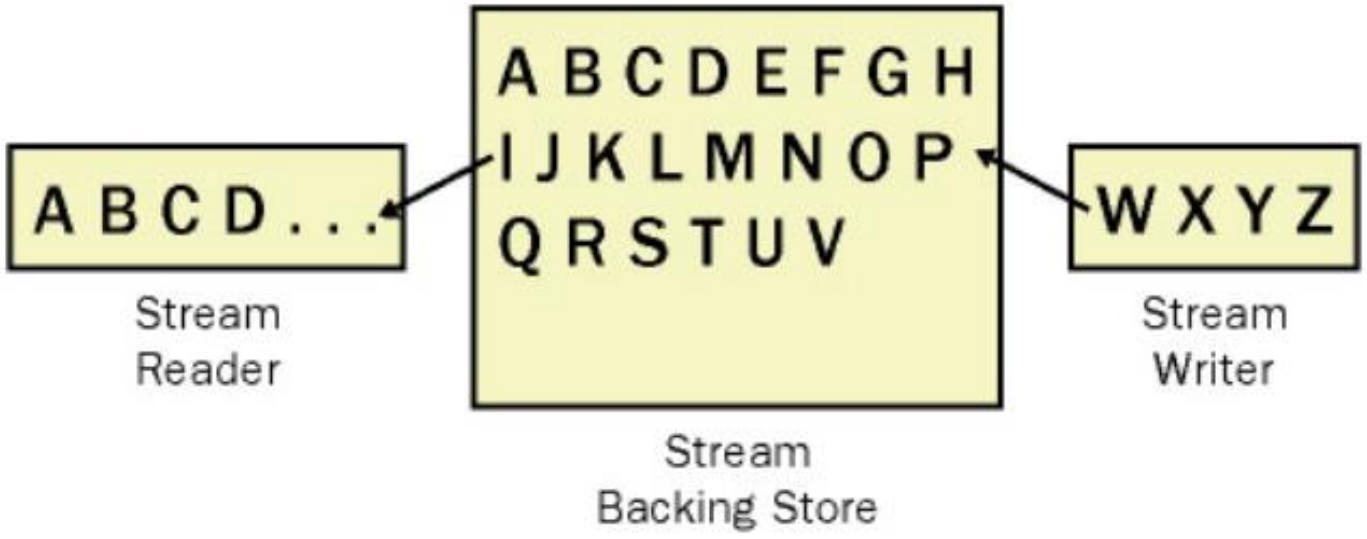

Figure 2-1: The stream I/O process

Stream Types

- **Base streams** are **streams that work directly with a backing store such as a file.**
- **Composable streams** are **streams that operate on top of other streams**

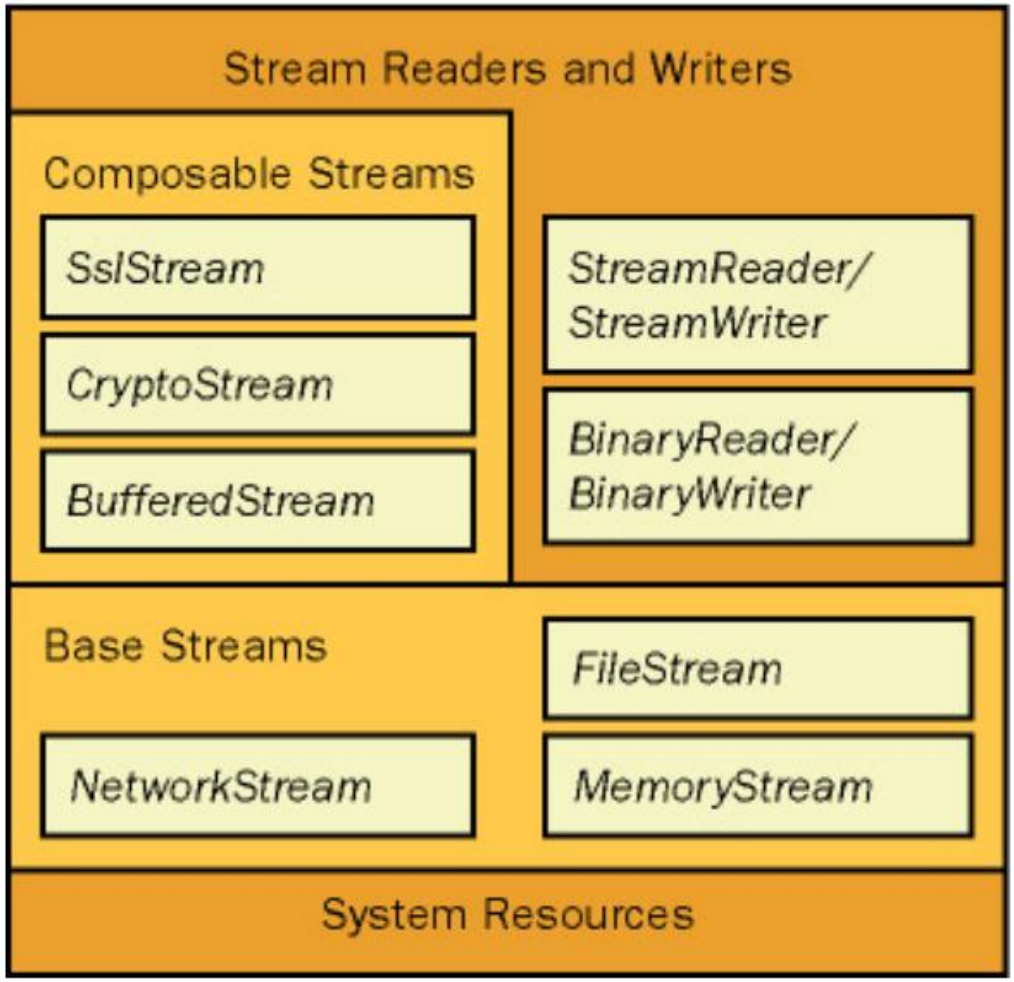

Figure 2-2: Stream relationship diagram

Stream Methods & properties

Table 2-1: Basic Stream Methods

Method	Description
BeginRead()	Allows data to be read asynchronously from a data store.
BeginWrite()	Allows data to be written asynchronously to a data store.
Close()	Closes a stream for further I/O operations and releases any operating system resources associated with a stream.
EndRead()	Completes asynchronous read operations started from BeginRead().
EndWrite()	Completes asynchronous write operations started from BeginWrite().
Flush()	Forces any buffered data associated with a stream to be written to the backing store.
Read()	Allows one or more data bytes to be read from a backing store.
ReadByte()	Allows one byte to be read from a backing store.
Seek()	Allows the Position property of the stream to be set.
SetLength()	Allows the size of the backing store in bytes to be controlled.
Write()	Allows bytes to be written to a backing store.
WriteByte()	Allows one byte to be written to a backing store.

Table 2-2: Basic Stream Properties

Property	Description
CanRead	Determines if the stream can be read.
CanSeek	Determines if the stream allows you to change the Position property.
CanWrite	Determines if the stream can be written to.
Length	Reports the size in bytes of the backing store.
Position	Controls the location in a stream where the next byte will be read or written to a backing store.

Dạng dữ liệu Stream

- **Stream cho phép xử lý các dạng dữ liệu sau:**
  - **byte-type arrays**
  - **text data**
  - **binary data types**

Các bước làm việc với file

- Bước 1. Khai báo biến file
- Bước 2. Mở file
- Bước 3. Khai báo dữ liệu cần đọc/ghì lên file
- Bước 4. Ghi/đọc dữ liệu từ file
- Bước 5. Đóng file

Chú ý: Quản lý vị trí bởi lệnh

<filename>.Seek(0, SeekOrigin.Begin);

Stream – Read/write data

- // Tạo và ghi dữ liệu lên file

```
byte[] data = new byte[10];
FileStream fileStream = null;
// Mở file để đọc và ghi
try {
    fileStream = new FileStream("data.dat", FileMode.Create, FileAccess.ReadWrite);
    // Gán dữ liệu cho buyte
    for (int i = 0; i < data.Length; i++)
    {
        data[i] = 1;
    }

    fileStream.Write(data, 0, data.Length);

    Console.WriteLine("Tao file thanh cong");
} catch (Exception ex)
{
    Console.WriteLine("Loi tao file:" + ex.Message);
}
```

Stream – Read/write data

- // Đọc dữ liệu trên file

```
// Trường hợp đọc ghi file đã tồn tại ta nên sử dụng hàm "Seek
// để xác định lại vị trí bắt đầu nghi
fileStream.Seek(0, SeekOrigin.Begin);

// Đọc file
byte[]  bf= new byte[1];
while(true)
{
    int ReadByte;

    ReadByte  = fileStream.Read(bf, 0, bf.Length);
    Console.WriteLine("Doc file thanh cong");
    if (ReadByte == 0)
    {
        Console.WriteLine("File da doc xong");
        break;
    }
    Console.WriteLine("Read byte -> " + bf[0].ToString());
}
```

Bài tập 1

- **Ghi/đọc file dữ liệu dạng text**

```
// Ghi dữ liệu lên file text
StreamWriter MyStreamWriter = null;
MyStreamWriter = new StreamWriter(".\\dtText.txt");
MyStreamWriter.WriteLine("Nội dung file cần ghi!");

// Đọc dữ liệu lên file Text
StreamReader MyStreamReader = null;
MyStreamReader = new StreamReader(".\\dtText.txt");
string FileData = null;
Do { FileData = MyStreamReader.ReadLine();
     if (FileData != null)     {
          Console.WriteLine("We read -> " + FileData);   }
} while (FileData != null);
```

Bài tập 2 – Đọc ghi file nhị phân

1. Ghi file nhị phân

```
FileStream MyFileStream = null;
MyFileStream = new FileStream(".\\dataBi.dat", FileMode.Create,
FileAccess.ReadWrite);
BinaryWriter MyBinaryWriter = null;
MyBinaryWriter.Write(459);
MyBinaryWriter.Flush();
```

2. Đọc file nhị phân

```
BinaryReader MyBinaryReader = null;
MyBinaryReader = new BinaryReader(MyFileStream);
while (true) {int Number;
Number = MyBinaryReader.ReadInt32();
Console.WriteLine("We read number -> " + Number.ToString());
```

Bài tập 3 – Ghi dữ liệu bộ nhớ

1. Khai báo dạng dữ liệu bộ nhớ

```
MemoryStream MyMemoryStream = null;
MyMemoryStream = new MemoryStream();
Console.WriteLine("The memory stream has allocated " +
MyMemoryStream.Capacity.ToString() + " bytes.");
```

2. Khai báo mảng byte

```
byte[ ] MyByteArray = new byte[4000];
for (int i = 0; i < MyByteArray.Length; i++)  {  MyByteArray[i] = 1;  }
int TotalBytesWritten = 0;
for (int j = 0; j < 9; j++) {// Ghi dữ liệu vào bộ nhớ
MyMemoryStream.Write(MyByteArray, 0, MyByteArray.Length);
TotalBytesWritten = TotalBytesWritten + MyByteArray.Length;
```

Đọc dữ liệu byte từ bộ nhớ

```
// Khai báo mảng lưu dữ liệu
byte[ ] MyReadBuffer = new byte[1];
// Đọc dữ liệu từ file
int ByteTotal = 0;
while (true)  {
   int BytesRead;
      BytesRead = MyMemoryStream.Read(MyReadBuffer, 0,
      MyReadBuffer.Length);
if (BytesRead == 0) {    break;  }
 ByteTotal = ByteTotal + BytesRead;
Console.WriteLine("We read " + ByteTotal.ToString() + " bytes from the
memory stream.");
```

QUESTION ?