Khái niệm về stream

Một luồng (stream) là một đối tượng được sử dụng để truyền dữ liệu. Khi dữ liệu truyền từ các nguồn bên ngoài vào ứng dụng ta gọi đó là đọc stream, và khi dữ liệu truyền từ chương trình ra nguồn bên ngoài ta gọi nó là ghi stream.

**Nguồn bên ngoài** thường là các file, tuy nhiên tổng quát thì nó có thể từ nhiều loại như đọc/ghi dữ liệu thông qua một giao thức mạng để trao đổi dữ liệu với một máy remote khác, đọc/ghi vào một nhớ ...

Một số stream chỉ cho đọc, một số chỉ cho ghi, một số lại cho phép truy cập nhẫu nhiên chứ không chỉ truy cập tuần tự (cho phép thay đổi vị trí con trỏ đọc dữ liệu trong luồng - ví dụ dịch chuyển vào giữa luồng dữ liệu để đọc dữ liệu ở khoảng nào đó)

Thư viện .NET cung cấp lớp cơ sở System.IO.Stream để hỗ trợ làm việc đọc ghi các byte dữ liệu với các stream, từ lớp cơ sở này một loạt lớp kế thừa cho những stream đặc thù như: FileStream, BufferStream, MemoryStream ...

**Lấy thông tin về stream** - khi có đối tượng lớp System.IO.Stream có một số thuộc tính để tra cứu thông tin về stream

|  |  |
| --- | --- |
| **Thuộc tính** | **Ý nghĩa** |
| CanRead | Cho biết stream hỗ trợ việc đọc hay không |
| CanWrite | Cho biết stream hỗ trợ việc ghi hay không |
| CanSeek | Cho biết stream hỗ trợ dịch chuyển con trỏ hay không |
| CanTimeout | Cho biết stream có đặt được time out |
| Length | Cho biết kích thước (byte) của stream |
| Position | Đọc hoặc thiết lập vị trí đọc (thiết lập thì stream phải hỗ trợ Seek) |
| ReadTimeout | Đọc hoặc thiết lập giá trị (mili giây) danh cho tác vụ đọc stream trước khi time out phát sinh |
| WriteTimeout | Đọc hoặc thiết lập giá trị (mili giây) danh cho tác vụ ghi stream trước khi time out phát sinh |

**Một số phương thức cho đối tượng Stream**

|  |  |
| --- | --- |
| **Phương thức** | **Ý nghĩa** |
| ReadByte | Đọc từng byte, trả về int (cast 1 byte) hoặc -1 nếu cuối file. |
| Read | Đọc một số byte, từ vị trí nào đó, kết quả đọc lưu vào mảng byte. Trả về số lượng byte đọc được, 0 nếu cuối stream.  // Tạo một stream và lưu vào đó một số byte  Stream stream = new MemoryStream ();  for (int i = 0; i < 122; i++) {  stream.WriteByte ((byte) i);  }  // Thiết lập vị trí là điểm bắt đầu  stream.Position = 0;  // Đọc từng block 5 bytes  byte[] buffer = new byte[10] { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 }; // mảng chứa 10 byte để làm bộ nhớ lưu byte đọc được  int offset = 0; // vị trí (index) trong buffer - nơi ghi byte đầu tiên đọc được  int count = 5; // đọc 5 byte  int numberbyte = stream.Read (buffer, 0, 2); // bắt đầu đọc  while (numberbyte != 0) {  Console.Write ($"{numberbyte} bytes: ");  for (int i = 1; i <= numberbyte; i++) {  byte b = buffer[i - 1];  Console.Write (string.Format ("{0, 5}", b));  }  numberbyte = stream.Read (buffer, offset, count); // đọc 5 byte tiếp theo  Console.WriteLine ();  } |
| WriteByte | Lưu 1 byte vào stream |
| Write | Lưu mảng bytes vào stream  stream.Read(buffer, offset, count); |
| Seek | Thiết lập vị trí trong stream |
| Flush | Giải phóng các bộ đêm |

## Làm việc với FileStream

Lớp FileStream tạo ra các đối tượng để đọc và ghi dữ liệu ra file. Do stream là tài nguyên không quản lý bởi GC, nên cần đưa nó vào cấu trúc using để tự động gọi giải phóng tài nguyên (Dispose) khi hết khối lệnh.

string filepath = "/home/data/data.txt";

using (var stream = new FileStream(path:filepath, mode: FileMode.Open, access: FileAccess.Read, share: FileShare.Read))

{

// code sử dụng stream (System.IO.Stream)

}

Như vậy, để tạo ra một stream file, để trao đổi dữ liệu cần 4 thông tin:

* **path** đường dẫn đến file
* **mode** kiểu liệt kê FileMode, nó cho biết bạn muốn mở file như thế nào, như:
  + FileMode.CreateNew tạo file mới
  + FileMode.Create tạo mới, nếu file đang có bị ghi đè
  + FileMode.Open mở file đang tồn tại
  + FileMode.OpenOrCreate mở file đang tồn tại, tạo mới nếu không có
  + FileMode.Truncate mở file đang tồn tại và làm rỗng file
  + FileMode.Append mở file đang tồn tại và tới cuối file, hoặc tạo mới
* **access** kiểu liệt kê FileAccess, cho biết muốn truy cập file như thế nào
  + FileAccess.Read chỉ đọc
  + FileAccess.Write chỉ ghi
  + FileAccess.ReadWrite đọc và ghi
* **share** kiểu liệt kê FileShare, cho phép thiết lập chia sẻ truy cập file
  + FileShare.None không chia sẻ - tiến trình khác truy cập file sẽ lỗi cho đến khi tiến trình mở file đóng nó lại.
  + FileShare.Read cho tiến trình khác mở đọc file.
  + FileShare.Write cho tiến trình khác mở ghi file.
  + FileShare.ReadWrite cho tiến trình khác mở đọc ghi file.
  + FileShare.Delete cho tiến trình khác xóa file.

Lớp File hỗ trợ tạo FileStream. File.OpenRead(filename) tạo stream để đọc, File.OpenWrite(filename) tạo stream để ghi.

**Lấy thông tin về stream** - khi có đối tượng lớp System.IO.Stream có một số thuộc tính để tra cứu thông tin về stream

string filepath = "/mycode/1.txt";

using (var stream = new FileStream( path:filepath, mode: FileMode.Open, access: FileAccess.ReadWrite, share: FileShare.Read))

{

Console.WriteLine(stream.Name);

Console.WriteLine($"Kích thước stream {stream.Length} bytes / Vị trí {stream.Position}");

Console.WriteLine($"Stream có thể : Đọc {stream.CanRead} - Ghi {stream.CanWrite} - Seek {stream.CanSeek} - Timeout {stream.CanTimeout} ");

}

// /mycode/1.txt

// Kích thước stream 289 bytes / Vị trí 0

// Stream có thể : Đọc True - Ghi True - Seek True - Timeout False

**Lấy thông tin encoding của file Text**

Khi đọc các file text (không phải file nhị phân), đầu tiên cần xác định encoding của nó (UTF8, Unicode, UTF32 ...). Thông tin về encoding được lưu ở vài byte đầu tiên của file nó gọi là BOM - Preamble (xem thêm [BOM](https://en.wikipedia.org/wiki/Byte_order_mark)). Tùy thuộc vào giá trị của khoảng 5 byte đầu tiên này mà xác định được encoding.

Thường bạn chỉ việc 5 byte đầu tiên, để xác định encoding như sau:

using System;

using System.IO;

using System.Text;

namespace CS016\_Stream\_FileStream {

public class UtilsEncoding {

public static Encoding GetEncoding (Stream stream) {

byte[] BOMBytes = new byte[4]; // mảng chứa 4 byte để làm bộ nhớ lưu byte đọc được

int offset = 0; // vị trí (index) trong buffer - nơi ghi byte đầu tiên đọc được

int count = 4; // đọc 4 byte

int numberbyte = stream.Read (BOMBytes, offset, count); // bắt đầu đọc 4 đầu tiên lưu vào buffer

if (BOMBytes[0] == 0xfe && BOMBytes[1] == 0xff) {

stream.Seek (2, SeekOrigin.Begin); // Di chuyển về vị trí bắt đầu của dữ liệu (đã trừ BOM)

return Encoding.BigEndianUnicode;

}

if (BOMBytes[0] == 0xff && BOMBytes[1] == 0xfe) {

stream.Seek (2, SeekOrigin.Begin); // Di chuyển về vị trí bắt đầu của dữ liệu (đã trừ BOM)

return Encoding.Unicode;

}

if (BOMBytes[0] == 0xef && BOMBytes[1] == 0xbb && BOMBytes[2] == 0xbf) {

stream.Seek (3, SeekOrigin.Begin);

return Encoding.UTF8;

}

if (BOMBytes[0] == 0x2b && BOMBytes[1] == 0x2f && BOMBytes[2] == 0x76) {

stream.Seek (3, SeekOrigin.Begin);

return Encoding.UTF7;

}

if (BOMBytes[0] == 0xff && BOMBytes[1] == 0xfe && BOMBytes[2] == 0 && BOMBytes[3] == 0) {

stream.Seek (4, SeekOrigin.Begin);

return Encoding.UTF32;

}

if (BOMBytes[0] == 0 && BOMBytes[1] == 0 && BOMBytes[2] == 0xfe && BOMBytes[3] == 0xff) {

stream.Seek (4, SeekOrigin.Begin);

return Encoding.GetEncoding (12001);

}

stream.Seek (0, SeekOrigin.Begin);

return Encoding.Default;

}

}

}

Với ASCII, 7 bit biểu diễn các ký tự - nó đủ biểu diễn bảng chữ cái tiếng Anh (in hoa, thường, số ký tự đặc biệt) - ASCII 1 byte biểu diễn 1 ký tự. UTF-16 thì dùng 2 byte biểu diễn 1 ký tự. UTF-32 dùng 4 byte biểu diễn 1 ký tự. Với UTF-8 (dùng nhiều ngày nay) - nó dùng biến để xác định bao nhiêu byte cho mỗi ký tự cụ thể, Mỗi ký tự có thể từ 1 - 6 byte

**Ghi file text bằng stream**

Ghi file text (tạo mới, ghi đè) - chọn file có Encoding UTF8, đầu tiên phải ghi các bytes BOM, lấy mảng bytes DOM bằng cách gọi encoding.GetPreamble(), sau đó chuyển chuỗi thành mảng bytes (đã encoding) - rồi lưu ra stream bằng Write

string filepath = "/mycode/2.txt";

using (var stream = new FileStream( path:filepath, mode: FileMode.Create, access: FileAccess.Write, share: FileShare.None))

{

//Write BOM - UTF8

Encoding encoding = Encoding.UTF8;

byte[] bom = encoding.GetPreamble();

stream.Write(bom, 0, bom.Length);

string s1 = "Xuanthulab.net - Xin chào các bạn! \n";

string s2 = "Ví dụ - ghi file text bằng stream";

// Encode chuỗi - lưu vào mảng bytes

byte[] buffer = encoding.GetBytes(s1);

stream.Write(buffer, 0, buffer.Length); // lưu vào stream

buffer = encoding.GetBytes(s2);

stream.Write(buffer, 0, buffer.Length); // lưu vào stream

}

**Đọc file text bằng stream**

Đầu tiên xác định Encoding của file text, sau đó đọc từng khối byte vào buffer (mảng byte), rồi thực hiện Encoding để xác định chuỗi.

string filepath = "/mycode/1.txt";

int SIZEBUFFER = 256;

using (var stream = new FileStream( path:filepath, mode: FileMode.Open, access: FileAccess.ReadWrite, share: FileShare.Read))

{

Encoding encoding = GetEncoding(stream);

Console.WriteLine(encoding.ToString());

byte[] buffer = new byte[SIZEBUFFER];

bool endread = false;

do

{

int numberRead = stream.Read(buffer, 0, SIZEBUFFER);

if (numberRead == 0) endread = true;

if (numberRead < SIZEBUFFER)

{

Array.Clear(buffer, numberRead, SIZEBUFFER - numberRead);

}

string s = encoding.GetString(buffer, 0, numberRead);

Console.WriteLine(s);

} while (!endread);

}

**Copy file stream**

Tạo 2 stream, một để đọc - một để lưu

string filepath\_src = "/mycode/1.txt";

string filepath\_des = "/mycode/3.txt";

int SIZEBUFFER = 5; // tăng lên đọc sẽ nhanh

using (var streamwrite = File.OpenWrite(filepath\_des))

using (var streamread = File.OpenRead(filepath\_src))

{

byte[] buffer = new byte[SIZEBUFFER];

bool endread = false;

do

{

int numberRead = streamread.Read(buffer, 0, SIZEBUFFER);

if (numberRead == 0) endread = true;

else {

streamwrite.Write(buffer, 0, numberRead);

}

} while (!endread);

}