Hướng Dẫn Thực Hành
Tập tin & Thư mục

1 Thư mục:

- Lớp Directory chứa phương thức dành cho việc tạo, di chuyển, duyệt thư mục.

- Lớp DirectoryInfo là lớp tương tự như lớp Directory. Nó cung các tất cả các phương thức mà lớp Directory có đồng thời bổ sung nhiều phương thức hữu ích hơn cho việc duyệt cấu trúc cây thư mục.

Ví dụ: duyệt các thư mục con

```
using System;
using System.IO;
namespace Programming_CSharp
{
   class Tester
   {
      public static void Main()
      {
         Tester t = new Tester();
         string theDirectory = @"D:\luanvan1";
         // duyệt thư mục và hiển thị ngày truy cập gần nhất
         // và tất cả các thư mục con
         DirectoryInfo dir = new DirectoryInfo(theDirectory);
         t.ExploreDirectory(dir);
         // hoàn tất. in ra số lượng thống kê
         Console.WriteLine("\n\n{0} directories found.\n",dirCounter);
      }
      // với mỗi thư mục tìm thấy, nó gọi chính nó
      private void ExploreDirectory(DirectoryInfo dir)
      {
         indentLevel++;     // cấp     độ thư mục
         //            định dạng cho việc trình bày
         for (int i = 0; i < indentLevel; i++)
         Console.Write(" ");            // hai khoảng trắng cho mỗi cấp
         // in thư mục và ngày truy cập gần nhất
         Console.WriteLine("[{0}] {1} [{2}]\n", indentLevel, dir.Name, dir.LastAccessTime);
         // lấy tất cả thư mục con của thư mục hiện tại
         //            đệ quy từng thư mục
         DirectoryInfo[] directories = dir.GetDirectories( );
         foreach (DirectoryInfo newDir        in directories)
         {
            dirCounter++;   //tăng biến          đếm
            ExploreDirectory(newDir);
```

```
        }
        indentLevel--;       // giảm cấp          độ thư mục
      }
      // các biến thành viên tĩnh cho việc thống kê và trình bày
      static int dirCounter = 1;

      static int indentLevel = -1; // so first push = 0
   }
}
```

2 Tập tin

Thư viện .NET cung cấp hai lớp File và FileInfo tương tự như với trường hợp thư mục.
Ví dụ: Chỉnh sửa tập tin

```
using System;
using System.IO;
namespace Programming_CSharp
{
   class Tester
   {
      public static void Main()
      {
         Tester t = new Tester();
         string theDirectory = @"D:\luanvan1";
         DirectoryInfo dir = new DirectoryInfo(theDirectory);
         t.ExploreDirectory(dir);
         Console.WriteLine("\n\n{0} files in {1} directories found.\n", fileCounter, dirCounter);
      }
      private void ExploreDirectory(DirectoryInfo dir)
      {
      indentLevel++;
      for (int i = 0; i < indentLevel; i++)
        Console.Write(" ");
      Console.WriteLine("[{0}] {1} [{2}]\n", indentLevel, dir.Name, dir.LastAccessTime);
      // lấy tất cả các tập tin trong thư mục và
      // in tên, ngày truy cập gần nhất, kích thước của chúng
      FileInfo[] filesInDir = dir.GetFiles( );
      foreach (FileInfo file in filesInDir)
      {
         // lùi vào một khoảng phía dưới thư mục
         // phục vụ việc trình bày
         for (int i = 0; i < indentLevel+1; i++)
            Console.Write(" ");         // hai khoảng trắng cho mỗi cấp
         Console.WriteLine("{0} [{1}] Size: {2} bytes",file.Name, file.LastWriteTime, file.Length);
         fileCounter++;
      }

      DirectoryInfo[] directories = dir.GetDirectories( );
      foreach (DirectoryInfo newDir             in directories)
      {
         dirCounter++;
         ExploreDirectory(newDir);
       }
      indentLevel--;
     }
```

```
    // các biến tĩnh cho việc thống kê và trình bày
    static int dirCounter = 1;
    static int indentLevel = -1;
    static int fileCounter = 0;
  }
}
```

3 Đọc và ghi dữ liệu

Đọc và ghi dữ liệu là nhiệm vụ chính của các luồng(Stream). Stream hỗ trợ cả hai cách đọc ghi đồng bộ hay bất đồng bộ. .NET Framework cung cấp sẵn nhiều lớp thừa kế từ lớp Stream, bao gồm FileStream, MemoryStream và NetworkStream. Ngoài ra còn có lớp BufferedStream cung cấp vùng đệm xuất nhập được dùng thêm với các luồng khác.

Tập tin nhị phân:

Lớp Stream có rất nhiều phương thức nhưng quan trọng nhất là năm phương thức Read(), Write(), BeginRead(), BeginWrite() và Flush().

```
Stream inputStream = File.OpenRead(@"C:\test\source\test1.cs");
Stream outputStream = File.OpenWrite(@"C:\test\source\test1.bak");
```

Ví dụ: Cài đặt việc đọc và ghi tập tin nhị phân

```
using System;
using System.IO;
namespace Programming_CSharp
{
  class Tester
  {
    const      int SizeBuff = 1024;
    public     static    void Main( )
    {
      Tester t = new Tester( );
      t.Run( );
    }
    private    void Run( )
    {
      Stream inputStream = File.OpenRead(@"C:\test\source\test1.cs");
      // ghi vào tập tin này
      Stream outputStream = File.OpenWrite(@"C:\test\source\test1.bak");
      // tạo vùng          đệm chứa dữ liệu
      byte[] buffer = new Byte[SizeBuff];
      int bytesRead;
      // sau khi  đọc dữ liệu xuất chúng ra outputStream
      while ( (bytesRead =inputStream.Read(buffer,0,SizeBuff)) > 0 )
      {
        outputStream.Write(buffer,0,bytesRead);
      }
      //          đóng tập tin trước khi thoát
      inputStream.Close( );
      outputStream.Close( );
```

```
    }
  }
}
```

Luồng có vùng đệm

Lớp BufferedStream là cài đặt cho luồng có vùng đệm.

Để tạo một luồng có vùng đệm trước tiên ta vẫn tạo luồng Stream như trên

```
Stream inputStream = File.OpenRead(@"C:\test\source\folder3.cs");
Stream outputStream = File.OpenWrite(@"C:\test\source\folder3.bak");
```

Sau đó truyền các luồng này cho hàm đựng của BufferedStream

```
BufferedStream bufferedInput = new BufferedStream(inputStream);
BufferedStream bufferedOutput = new BufferedStream(outputStream);
```

Từ đây ta sử dụng bufferedInput và bufferedOutput thay cho inputStream và outputStream. Cách sử dụng là như nhau: cũng dùng phương thức Read() và Write()

```
while((bytesRead = bufferedInput.Read(buffer,0,SIZE_BUFF))>0 )
{
        bufferedOutput.Write(buffer,0,bytesRead);
}
```

Ghi chú: nến dùng hàm Flush() để chắc chắn dữ liệu đã được chuyển từ buffer lên tập tin.

```
bufferedOutput.Flush( );
```

Ví dụ: Cài đặt luồng có vùng đệm

```
using System;
using System.IO;
namespace Programming_CSharp
{
  class Tester
  {
    const int SizeBuff = 1024;
    public static void Main()
    {
      Tester t = new Tester();
      t.Run();
    }
    private void Run()
    {
      // tạo một luồng nhị phân
      Stream inputStream = File.OpenRead(@"C:\test\source\folder3.cs");
      Stream outputStream = File.OpenWrite(@"C:\test\source\folder3.bak");
      // tạo luồng vùng   đệm kết buộc với luồng nhị phân
      BufferedStream bufferedInput = new BufferedStream(inputStream);
      BufferedStream bufferedOutput =new BufferedStream(outputStream);
      byte[] buffer = new Byte[SizeBuff];
      int bytesRead;
      while ((bytesRead = bufferedInput.Read(buffer, 0, SizeBuff)) > 0)
      {
        bufferedOutput.Write(buffer, 0, bytesRead);
      }
```

```
            bufferedOutput.Flush();
            bufferedInput.Close();
            bufferedOutput.Close();
        }
    }
}
```

Làm việc với tập tin văn bản

Đối với các tập tin chỉ chứa văn bản, sử dụng hai luồng StreamReader và StreamWriter cho việc đọc và ghi.

Dùng phương thức OpenText() của lớp FileInfo.

```
FileInfotheSourceFile = new FileInfo (@"C:\test\source\test1.cs");
StreamReader stream = theSourceFile.OpenText( );
```

Sau đó đọc từng dòng văn bản của tập tin

```
do
{
        text = stream.ReadLine( );
} while (text != null);
```

Để tạo đối tượng StreamWriter ta truyền cho hàm đường dẫn tập tin

```
StreamWriter writer = new StreamWriter(@"C:\test\source\folder3.bak",false);
```

tham số thứ hai thuộc kiểu bool, nếu tập tin đã tồn tại, giá trị true sẽ ghi dữ liệu mới vào cuối tập tin, giá trị false sẽ xóa dữ liệu cũ, dữ liệu mới sẽ ghi đè dữ liệu cũ.

Ví dụ: Đọc và ghi tập tin văn bản

```
using System;
using System.IO;
namespace Programming_CSharp
{
   class Tester
   {
      public static void Main()
      {
         Tester t = new Tester();
         t.Run();
      }
      private void Run()
      {
         // mở một tập tin
         FileInfo theSourceFile =     new FileInfo(@"C:\test\source\test.cs");
         // tạo luồng đọc văn bản cho tập tin
         StreamReader reader = theSourceFile.OpenText( );
         // tạo luồng ghi văn bản cho tập tin xuất
         StreamWriter writer =        new StreamWriter( @"C:\test\source\test.bak",false);
         // tạo một biến chuỗi lưu giữ một dòng văn bản
         string text;

         //            đọc toàn bộ tập tin theo từng dòng
         // ghi ra màn hình console và tập tin xuất
         do
         {
            text = reader.ReadLine( );
            writer.WriteLine(text);
```

```
            Console.WriteLine(text);
        } while (text != null);
        //              đóng tập tin
        reader.Close( );
        writer.Close( );
    }
  }
}
```

Khi thực thi chương trình nội dung tập tin nguồn được ghi lên tập tin mới đồng thời xuất ra màn hình console.

Bài tập:

1. Xây dựng chương trình minh họa các câu lệnh sau:

- dir: hiển thị danh sách các tệp tin và thư mục hiện hành(có tùy chọn như /w: hiển thị dạng cột)
- md: tạo thư mục
- rd: xóa thư mục
- type: hiển thị nội dung tệp tin