THỰC HÀNH TUẦN 2

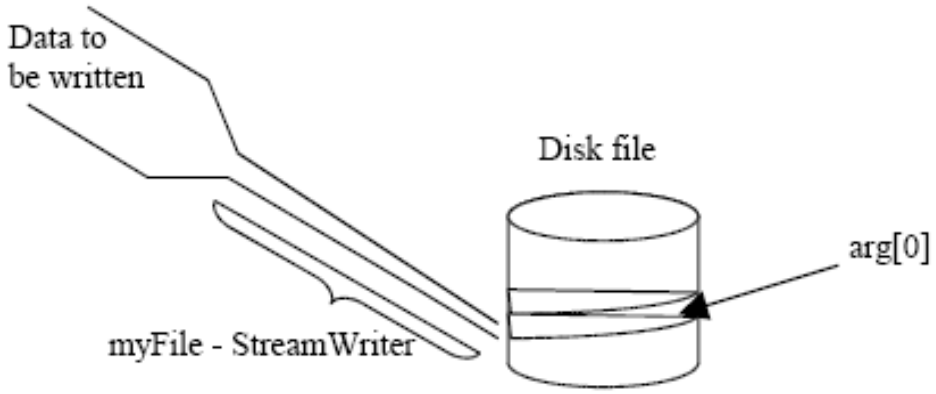
I/O STREAM

I. MỤC ĐÍCH

- Cung cấp khả năng khởi tạo, đọc, viết và khả năng cập nhật File.
- Hiểu được luồng thông tin (Stream) trong C#.
- Có thể sử dụng được các lớp FileStream, lớp StreamReader và lớp BinaryFormatter để đọc và viết các đối tượng vào trong các File.

II. NỘI DUNG

1. **Luồng (Stream)** là luồng của thông tin, chứa thông tin sẽ được chuyển qua, còn **tập tin (File)** thì để lưu trữ thông tin, dữ liệu; thậm chí còn có thể giữ lại dữ liệu sau khi chương trình kết thúc.



Hình 1 : Mô tả thực hiện tạo tập tin và đưa dữ liệu vào

Dữ liệu được truyền theo hai hướng

- Đọc dữ liệu : đọc dữ liệu từ bên ngoài vào chương trình.
- Ghi dữ liệu: đưa dữ liệu từ chương trình ra bên ngoài.

Có 3 đối tượng Stream:

- Console.In: Trả về đối tượng Stream đưa vào chuẩn.
- Console. Out: Trả về đối tượng Stream lấy ra chuẩn.
- Console. Error: Trả về đối tượng Stream thông báo lỗi chuẩn

2. **Thứ tự của việc đọc/ghi một tập tin**

Khi đọc hay viết một tập tin, cần thiết phải theo một trình tự xác định. Đầu tiên là phải thực hiện công việc mở tập tin. Nếu như tạo mới tập tin, thì việc mở tập tin cùng lúc với việc tạo ra tập tin đó. Khi một tập tin đã mở, cần thiết phải tạo cho nó một luồng để đặt thông tin vào trong một tập tin hay là lấy thông tin ra từ tập tin. Khi tạo một luồng, cần thiết phải chỉ ra thông tin trực tiếp sẽ được đi qua luồng. Sau khi tạo một luồng gắn với một tập tin, thì lúc này chúng ta có thể thực hiện việc đọc ghi các dữ liệu trên tập tin. Khi thực hiện việc đọc thông tin từ một tập tin, chúng ta cần thiết phải kiểm tra xem con trỏ tập tin đã chỉ tới cuối tập tin chưa, tức là chúng ta đã đọc đến cuối tập tin hay chưa. Khi hoàn thành việc đọc ghi thông tin trên tập tin thì tập tin cần phải được đóng lại.

Tóm lại các bước cơ bản để làm việc với một tậo tin là:

- Bước 1: Mở hay tạo mới tập tin
- Bước 2: Thiết lập một luồng ghi hay đọc từ tập tin
- Bước 3: Đọc hay ghi dữ liệu lên tập tin
- Bước 4: Đóng lập tin lại

Có rất nhiều luồng (stream) khác nhau. Chúng ta sẽ sử dụng những luồng khác nhau và những phương thức khác nhau phụ thuộc vào kiểu dữ liệu bên trong của tập tin. Trong phần này, việc đọc/ghi sẽ được thực hiện trên tập tin văn bản và trên tập tin nhị phân. Thông tin nhị phân bao hàm khả năng mạnh lưu trữ giá trị số và bất cứ kiểu dữ liệu nào khác.

3. **FileStream** : Là một lớp dẫn xuất từ Stream, được sử dụng đọc và viết dữ liệu vào một file hay đọc và viết dữ liệu từ 1 file. Ví dụ:

```
FileStream fs = new FileStream(FileName, mode);
```

Trong đó:

- FileName: tập tin mà chúng ta muốn truy xuất đến.
- Mode: chế độ mở file như thế nào (Append, Create, CreateNew, Open, OpenOrCreate…)

  Ex: `FileStream fs = new FileStream("thuchanh.txt", FileMode.CreateNew);`

4. **StreamReader**: là 1 lớp dẫn xuất từ Stream, là luồng đọc tập tin. Để đọc file ta dùng lớp StreamReader. Để ghi file ta dùng lớp StreamWriter. Đây là lớp được dùng để viết và ghi 1 tập tin dạng văn bản

StreamReader sr = new StreamReader(FileStream fileName);

StreamWriter sw = new StreamReader(FileStream fileName);

Ví dụ:

StreamReader sr = new StreamReader(fs);

5. **BinaryStream:**

Nếu chúng ta sử dụng một tập tin văn bản, thì khi chúng ta lưu dữ liệu kiểu số thì phải thực hiện việc chuyển đổi sang dạng chuỗi ký tự để lưu vào trong tập tin văn bản và khi lấy ra ta cũng lấy được giá trị chuỗi ký tự do đó ta phải chuyển sang dạng số. Đôi khi chúng ta muốn có cách thức nào đó tốt hơn để lưu trực tiếp giá trị vào trong tập tin và sau đó đọc trực tiếp giá trị ra từ tập tin.

Ví dụ: khi viết một số lượng lớn các số integer vào trong tập tin như là những số nguyên, thì khi đó ta có thể đọc các giá trị này ra như là số integer. Trường hợp nếu chúng được viết vào tập tin với dạng văn bản, thì khi đọc ra ta phải đọc ra văn bản và phải chuyển mỗi giá trị từ một chuỗi đến các số integer. Tốt hơn việc phải thực hiện thêm các bước chuyển đổi, ta có thể gắn một kiểu luồng nhị phân **BinaryStream** vào trong một tập tin, rồi sau đó đọc và ghi thông tin nhị phân từ luồng này.

Ghi chú: Thông tin nhị phân là thông tin đã được định dạng kiểu lưu trữ dữ liệu.

Ví dụ:

```
FileStream fs = new FileStream(sfd.FileName,FileMode.CreateNew);
 BinaryWriter bw = new BinaryWriter(fs);
```

6. **BinaryFormatter:** sử dụng 2 phương thức Serialize và Deserialize để viết và đọc đối tượng từ trong luồng:

- ❑ ***Serialize***: chuyển đổi một đối tượng sang một định dạng, và có thể được viết vào File mà không mất dữ liệu.
- ❑ ***Deserialize***: đọc dữ liệu đã định dạng từ một File và chuyển nó về dạng ban đầu

Ví dụ:

Serialize

```
BinaryFormatter binaryFormatter = new BinaryFormatter();
FileStream fileName = File.Create("..\\student.txt");
binaryFormatter.Serialize(fileName, st);
```

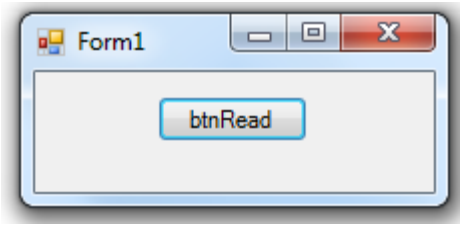
Deserialize

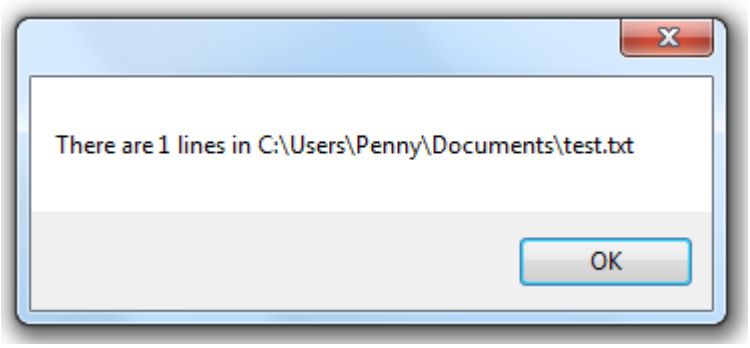
```
BinaryFormatter bf = new BinaryFormatter();
FileStream fs = File.OpenRead("..\\student.txt");
Student student = (Student)bf.Deserialize(fs);
```

III. BÀI TẬP MẪU

Bài 1

Viết chương trình đọc file như sau: Khi nhấn vào button btnRead sẽ đếm và thông báo số dòng có trong một tập tin bất kỳ. Hình minh họa.



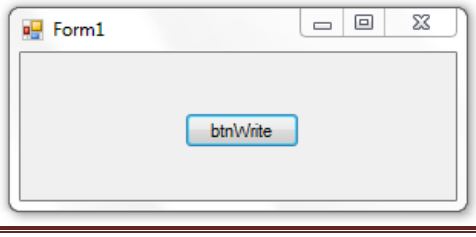


Gợi ý: bắt sự kiện cho nút btnRead, sử dụng lớp StreamReader.

```
private void btnRead_Click(object sender, System.EventArgs e)
        {

            OpenFileDialog ofd = new OpenFileDialog();
            ofd.ShowDialog();
           FileStream fs = new FileStream(ofd.FileName,
           FileMode.OpenOrCreate);
            StreamReader sr = new StreamReader(fs);
            int lineCount = 0;
            while (sr.ReadLine() != null)
            {
                lineCount++;
            }
            fs.Close();
           MessageBox.Show("There are " + lineCount + " lines in " +
           ofd.FileName);
        }
```

Bài 2

Viết chương trình ghi thành file nhị phân bất kỳ. Hình minh họa



Gợi ý: bắt sự kiện cho nút btnWrite, sử dụng BinaryWriter

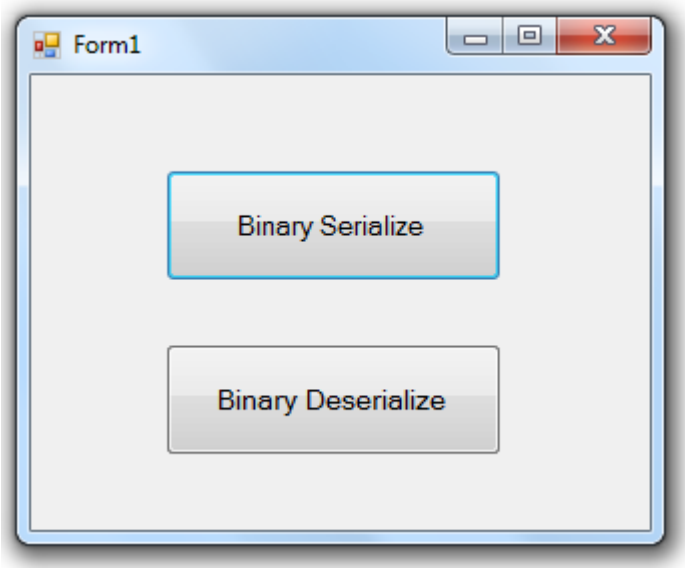
```
private void btnWrite_Click(object sender, EventArgs e)
        {
            SaveFileDialog sfd = new SaveFileDialog();
            sfd.ShowDialog();
           FileStream fs = new FileStream(sfd.FileName,
           FileMode.CreateNew);

            BinaryWriter bw = new BinaryWriter(fs);
            int[] myArray = new int[1000];
            for (int i = 0; i < 1000; i++)
            {
                myArray[i] = i;
                bw.Write(myArray[i]);
            }
            bw.Close();
        }
```

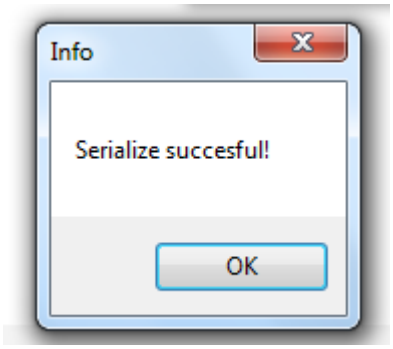
Bài 3:

Tạo Class Student với các thành phần sau:

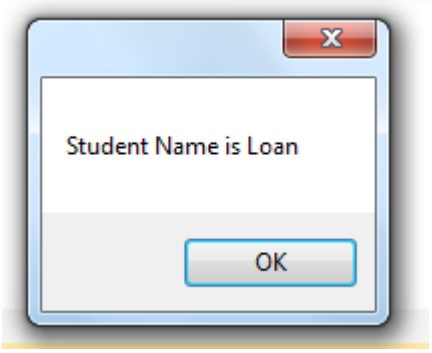
```
public class Student{
            public String lastName; // Họ
            public String firstName; // Tên
            public int     age;     // Tuổi
        }
```

Viết chương trình chuyển đổi đối tượng Student trên sang định dạng nhị phân, ghi vào tập tin student.txt khi nhấn nút BinarySerialize như hình mẫu. Sau đó đọc dữ liệu đã định dạng từ tập tin student.txt và chuyển nó về dạng ban đầu.





Serialize



Deserialize

Gợi ý: Sử dụng BinaryFormater

```
private void button1_Click(object sender, EventArgs e)
        {
            Student st = new Student();
            st.lastName = "Tôn";
            st.firstName = "Loan";
            st.age = 22;
            BinaryFormatter binaryFormatter = new BinaryFormatter();
            FileStream fileName = File.Create("..\\student.txt");
            binaryFormatter.Serialize(fileName, st);
            fileName.Close();
            MessageBox.Show("Serialize succesful!", "Info");
        }

        private void button2_Click(object sender, EventArgs e)
        {
            BinaryFormatter bf = new BinaryFormatter();
            FileStream fs = File.OpenRead("..\\student.txt");
            Student student = (Student)bf.Deserialize(fs);
            fs.Close();
            MessageBox.Show("Student Name is " + student.firstName);
        }
        [Serializable()]
        public class Student{
            public String lastName;
            public String firstName;
            public int     age;
        }
```

Lưu ý:

- Phải đặt `[Serializable()]` trước đối tượng cần chuyển đổi.
- Tập tin student.txt mặc định sẽ được lưu vào thư mục *bin* của project.

Bài 4:

Tạo một form đăng nhập có tên đăng nhập và mật khẩu, nếu không check vào ô Ghi nhớ, nhấn đăng nhập dữ liệu ghi vào file .txt là 0; nếu có check nó sẽ lưu user, pass và số 1 vào 3 dòng lien tiếp nhau.

Hướng dẫn:

// Trong hàm load Form lên

```
        {
            FileStream fs;
            if (!File.Exists("D://Pass.txt”))            {
                fs = new FileStream("D://Pass.txt", FileMode.Create);
                StreamWriter sWriter = new StreamWriter(fs, Encoding.UTF8);

                sWriter.WriteLine("Hello World!");
                sWriter.Flush();
                fs.Close();
            }
```

//Trong hàm click button

```
        {
            FileStream fs = new FileStream("D://Pass.txt", FileMode.Create);
            StreamWriter writeFile = new StreamWriter(fs, Encoding.UTF8);//dùng
streamwriter để ghi file
            if (RememberCheck.Checked == true)//nếu checkbox được checked thì nhớ tên và
mật khẩu
            {
                writeFile.WriteLine(txtUser.Text);
                writeFile.WriteLine(txtPass.Text);
                writeFile.WriteLine("1");//dòng "1" để kiểm tra có checked hay không
                writeFile.Flush();//ghi từng dòng vào file Pass.txt
            }
            else writeFile.WriteLine("0");//dòng "0" là không checked vào checkbox
            writeFile.Close();
        }
```

IV. BÀI TẬP THỰC HÀNH

TẤT CẢ CÁC BÀI TẬP DƯỚI ĐÂY ĐỀU SỬ DỤNG WINDOWS FORM APPLICATION

Bài 5:

Nhập nhiều đoạn văn bản vào 1 Textbox và ghi xuống file "input.txt'. Đọc nội dung file "input.txt" và xuất ra màn hình.

Bài 6:

Đọc nội dung từ file "input.txt" với nội dung theo định dạng (4 dòng, mỗi dòng 1 số nguyên), sau đó thực hiện các phép tính và ghi kết quả xuống file "output.txt".

Ví dụ : *Nội dung file "input.txt" :*

A = 10

B = 1

C = 4

D = 2

Nội dung file "output.txt"

A+B = 11

C-D= 2

A*B = 10

C/D = 2

Bài 7:

Viết chương trình sử dụng `BinaryFormatter` cho phép :

Nhập 1 mảng các nhân viên (chú ý khi nhập không nhập giá trị của TONGLUONG) và ghi xuống file "input.txt". Cấu trúc của Học Viên như sau :

- MANV : String
- HOTEN : String
- SDT : String
- LUONGCUNG : double
- PHUCAP : double
- TONGLUONG : double

- Đọc thông tin mảng NHAN Viên từ file “input.txt” và tính điểm tổng lương cho từng nhân viên sau đó ghi xuống file “output.txt” và xuất ra màn hình

Ví dụ:

Cấu trúc file “input.txt”

IH811
NguyenVanA
1234567890
10.000.000
5.500.000

IH812
NguyenVanB
1234567891
8.200.000
3.500.000

Cấu trúc file “output.txt”

IH811
NguyenVanA
1234567890
10.000.000
5.500.000
15.500.000

IH812
NguyenVanB
1234567891
8.200.000
3.500.000
11.700.000