**Lesson 04 - Working with File using C#**

Objective:

In this lesson, we will learn how to do read-write operation in a file using C#. At the end of this lesson, we will learn how to work with a CSV file by using a third party library.

Prerequisite:

Participants must have familiarization with basic C# syntax and must basic concept on Windows file system.

Topics to be discussed:

* What is a file to a programmer?
* What is Stream?
* How to do read-write operation in a file
* How to work with a third party DLL in your own project

Pretest & Posttest questions:

1. What are File and Stream?
2. What will be the problem if the stream is not closed?
3. What is the advance of CSV file?
4. What is DLL?

* **Working with Text File in C#**

**What are file and stream?**

A file is a collection of data stored in a computer disk with a specific name and a directory path. When a file is opened for reading or writing, it becomes a stream.

The stream is the sequence of bytes passing through the communication path. There are two main streams: the input stream and the output stream. The input stream is used for reading data from file (read operation) and the output stream is used for writing into the file (write operation). Now we will work (read-write) with a text file and then more theory will be discussed.

**Walkthrough: Make an application by which user can keep some person’s name in a file and can see these when (s)he wants.**

UI layout should be as follows:



1. Create a desktop project. And Design the UI as above. At first, we have to define where the file will be created and what will be name. So, write the following line inside your Form class:

private string fileLocation = @"studentname.txt";

So, this file will be created (if not exists) in application path (where the exe of this application is created) and its name will be studentname.txt

1. Write the following code inside saveButton\_click method.

FileStream aFileStream = new FileStream(fileLocation, FileMode.Append);

StreamWriter aStreamWriter = new StreamWriter(aFileStream);

aStreamWriter.Write(nameTextBox.Text);

aStreamWriter.WriteLine();

aStreamWriter.Close();

We will discuss about FileStream, StreamWriter and FileMode later.

1. Now, write the following code inside showButton\_click method.

FileStream aFileStream = new FileStream(fileLocation, FileMode.Open);

StreamReader aStreamReader = new StreamReader(aFileStream);

namesListBox.Items.Clear();

while (!aStreamReader.EndOfStream)

{

string aLine = aStreamReader.ReadLine();

namesListBox.Items.Add(aLine);

}

aStreamReader.Close();

You see that there is a line inside condition checking part of while loop. It is:

!aStreamReader.EndOfStream

This line ensures that while loop will be continued till end of file. Also you will find aStreamReader.Close() just after while loop. Without this line opened file will not be closed and so other applications couldn’t open it later.

1. Now run your project and see it works.

Download the solution from attachment.

Let’s discuss about several topics on file related operations.

**C# I/O Classes**

The System.IO namespace has various class that are used for performing various operation with files, like creating and deleting files, reading from or writing to a file, closing a file etc.

The following table shows some commonly used classes in the System.IO namespace:

|  |  |
| --- | --- |
| *I/O Class* | *Description* |
| BinaryReader | Reads primitive data from a binary stream. |
| BinaryWriter | Writes primitive data in binary format. |
| BufferedStream | A temporary storage for a stream of bytes. |
| Directory | Helps in manipulating a directory structure. |
| DirectoryInfo | Used for performing operations on directories. |
| DriveInfo | Provides information for the drives. |
| File | Helps in manipulating files. |
| FileInfo | Used for performing operations on files. |
| FileStream | Used to read from and write to any location in a file. |
| MemoryStream | Used for random access to streamed data stored in memory. |
| Path | Performs operations on path information. |
| StreamReader | Used for reading characters from a byte stream. |
| StreamWriter | Is used for writing characters to a stream. |
| StringReader | Is used for reading from a string buffer. |
| StringWriter | Is used for writing into a string buffer. |

**FileStream**

FileStream class in the System.IO namespace helps in reading from, writing to and closing files.

You need to create a FileStream object to create a new file or open an existing file.

**FileMode**

There are several modes for a file to be opened. FileMode enumerator defines various methods for opening files. The members of the FileMode enumerator are:

*Append*: It opens an existing file and puts cursor at the end of file, or creates the file, if the file does not exist.

*Create*: It creates a new file.

*CreateNew*: It specifies to the operating system, that it should create a new file.

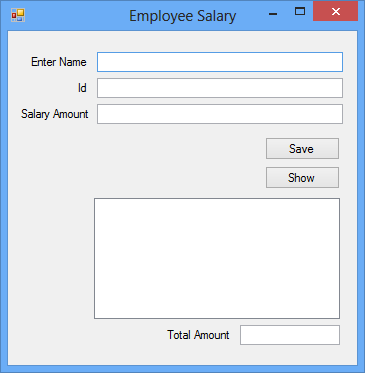
*Open*: It opens an existing file.

*OpenOrCreate*: It specifies to the operating system that it should open a file if it exists, otherwise it should create a new file.

*Truncate*: It opens an existing file and truncates its size to zero bytes.

So far, we have worked with text file only. But from above discussion you have got an idea to work with [binary file](http://en.wikipedia.org/wiki/Binary_file) also. See details [here](http://www.tutorialspoint.com/csharp/csharp_binary_files.htm)

**Practice: Make an application by which user will save employees name, id and salary. See the figure below:**



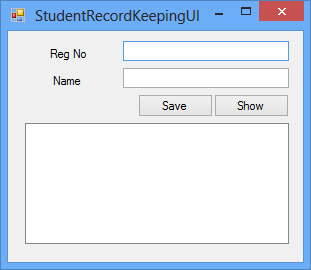
When show button will be clicked. All information will be shown in a listbox and total will be shown in a textbox. Download the source from attachment.

**Working with CSV file using a third party library**

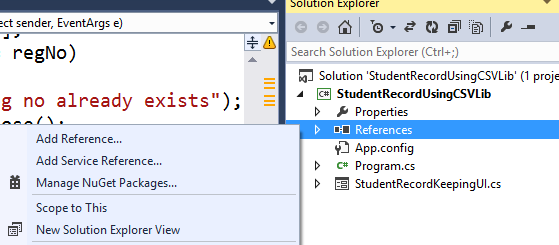
A **Comma-Separated Values** (**CSV**) (also sometimes called *character-separated values,* because the separator character does not have to be a comma) file stores [tabular](http://en.wikipedia.org/wiki/Tabular) data (numbers and text) in plain-text form. [Plain text](http://en.wikipedia.org/wiki/Plain_text) means that the file is a sequence of [characters](http://en.wikipedia.org/wiki/Character_(computing)), with no data that has to be interpreted instead, as binary numbers. A CSV file consists of any number of [records](http://en.wikipedia.org/wiki/Record_(computer_science)), separated by line breaks of some kind; each record consists of [fields](http://en.wikipedia.org/wiki/Field_(computer_science)), separated by some other character or string, most commonly a literal comma or [tab](http://en.wikipedia.org/wiki/Tab_character#Tab_characters). Usually, all records have an identical sequence of fields. – Source Wikipedia.

Now we will work with CSV file using C#. In this application, we will use a third party library to work with. There are lots of third party libraries in C# to work with CSV file. But we will use the library customized by us (main library is here: <http://csvfile.codeplex.com>). Download CSVLib from the attachment and find a DLL file there.

**Walkthrough: Make an application where user will keep as many student information as he wants. See UI below. Note that registration number must be unique.**



1. Create a desktop application and design UI as above. Now, our first job is to add reference in our application. Right click on References of your project as picture below. Click on Add Reference on popup. Reference Manager Window will be displayed soon. Browse your downloaded dll (from CVSLib) and add it. Press ok button. That all about adding reference in a project.



1. Now keep file location string your form file.

private string fileLocation = @"studentrecord.csv";

1. Now add the following code inside saveButton\_click method:

FileStream aStream = new FileStream(fileLocation, FileMode.Append);

CsvFileWriter aWriter = new CsvFileWriter(aStream);

List<string> aStudentRecord = new List<string>();

aStudentRecord.Add(regNoTextBox.Text);

aStudentRecord.Add(nameTextBox.Text);

aWriter.WriteRow(aStudentRecord);

aStream.Close();

Here, we use the object of CsvFileWriter to write something in our CSV file. We use its method WriteRow for writing purpose.

1. Finally, add the following code inside showButton\_click method:

FileStream aStream = new FileStream(fileLocation, FileMode.Open);

CsvFileReader aReader = new CsvFileReader(aStream);

List<string> aStudentRecord = new List<string>();

studentListBox.Items.Clear();

while (aReader.ReadRow(aStudentRecord))

{

string regNo = aStudentRecord[0];

string name = aStudentRecord[1];

studentListBox.Items.Add(regNo + " " + name);

}

aStream.Close();

Download the complete code from attachment where you will also find how we check existing reg no saved in our file to avoid duplication.

**Practice: Make an address book as describe below:**

* Make an address book where user will save person’s name, email address, personal contact number, home contact number, home address. Where personal contact number will be unique. Also note that all information will be saved in a CSV file
* When show all button will be clicked, all saved information will be shown in listview.
* There’s should be a search option where user will search addresses using name, email or personal contact number.

No solution is given for this practice.