Text Files in VB .NET

To be able to open a file and read the data from a storage unit of a computer, such as a hard drive and able to save the data into the storage unit are important functions of a computer program. In fact, the ability to store, retrieve and modify data makes a computer a powerful tool in database management.

In this lesson, we will learn how to manage data that is stored as a text file. Visual Basic allows the user to create a text file, save the text file as well as read the text file.

There is a very useful object in VB.NET called System.IO (the IO stands for Input and Output). You can use this object to read and write to text files.

For now, let's just see how to open up a text file using the System.IO object.

First, here's an explanation of just what we mean by "text file".

What is a Text File?

Text files have an extension that ends in .txt. The Windows operating system gives you a good, basic Text Editor in Notepad. The Notepad programme allows you to save files with the .txt extension. In other words, as Text Files. These Text Files can then be opened by a wide variety of programmes.

Advantages with text files

Portability

One of the best things about plain text is that it is a portable format between almost any operating system. You can use plain text files on Windows, Mac OS, iOS, Android, Windows Phone, Linux, etc. All of these operating systems have ways of natively showing you the contents of a text file as well and also allowing you to edit its contents.

Easy To use

Plain text files are at the peak ease of use. There isn’t really anything to learn; you just start typing text into a blank file. That’s it. No keyboard shortcuts to learn, or complicated menu structures.

No lock-in

Another great reason that with plain text is that there is no vendor lock-in. This goes hand-in-hand with the portability reason mentioned above. There is no “special app” that only supports text files. There is no “compatibility issues” that you need to deal with.

How to Open a Text File in VB .NET

The ability to open up a text file and read its contents can be very useful to you in your programming life. You might have a text file containing quiz questions and answers, for example. You could read the questions and answers from a text file and create your own "Who wants to be a Millionaire" game. Or you might want to save some data associated with your programme, and then open it up again when the programme starts. Well see how to open up a text file in VB .NET right now. In a later section, you'll learn how to save data to a text file.

Create a StreamReader

To open up a text file, you need to create something called a "StreamReader". This, as its name suggests, reads streams of text. The StreamReader is an object available to System.IO. You create a StreamReader like this:

```
Dim FILE_NAME As String = "E:\test.txt"

Dim objReader As New System.IO.StreamReader (FILE_NAME)
```

The first line just sets up a string variable called FILE_NAME. We store the path and name of our text file inside of the string variable:

```
= "E:\test.txt"
```

We're saying that there is a text file called test which is at the location (path) "E:\".

You set up the StreamReader to be a variable, just like a String or Integer variable. But we're setting up this variable differently:

```
Dim objReader As New System.IO.StreamReader( FILE_NAME )
```

We've called the variable objReader. Then, after the "As" word comes "New". This means "Create a New Object". The type of object we want to create is a StreamReader object:

```
System.IO.StreamReader
```

System is the main object. IO is an object within System. And StreamReader is an object within IO.

StreamReader needs the name of a file to Read. This goes between a pair of round brackets:

```
System.IO.StreamReader( FILE_NAME )
```

VB will then assign all of this to the variable called objReader. So instead of assigning say 10 to an Integer variable, you are assigning a StreamReader to a variable.

```
Read To End
```

But this won't do you any good. We haven't actually opened the text file yet. We've just told VB where the text file is and what object to open it with. You do the opening like this:

```
TextBox1.Text = objReader.ReadToEnd
```

Now that objReader is an object variable, it has its own properties and methods available for use (in the same way that the textbox has a Text property).

One of the Methods available to our new StreamReader variable is the ReadToEnd method. This will read the whole of your text, right to the end. We're then popping this in a textbox.

Do the following:

Start a new project

Add a textbox to your new form, and just leave it on the default Name of Textbox1

Set its MultiLine property to True

Add a Button to your form

Double click the button and add the following code for it:

```
Dim FILE_NAME As String = "E:\test.txt"

Dim objReader As New System.IO.StreamReader( FILE_NAME )

TextBox1.Text = objReader.ReadToEnd

objReader.Close()
```

The last line closes the StreamReader we set up. You have to close your stream objects after you've used them, otherwise you'll get errors messages.

When you're done, run your programme and click your Button.

Unless you already have a file called test.txt at the location specified you'll get this error message popping up like this one: (You may see this error message in a different format, depending on which version of Visual Studio you have.)

FileNotFound Error message in VB NET

The last line spells it out clearly: Could not find file "C:\Users\Owner\Documents\test.txt". So we were trying to read a text file that doesn't exist.

Does the File Exist?

You can, though, test to see if the file exists. If it does, you can open it; if not, you can display an error message. Amend your code to this (the new lines are in bold):

```
Dim FILE_NAME As String = "C:\Users\Owner\Documents\test.txt"

If System.IO.File.Exists(FILE_NAME) = True Then

Dim objReader As New System.IO.StreamReader(FILE_NAME)

TextBox1.Text = objReader.ReadToEnd

objReader.Close()

Else

MessageBox.Show("File Does Not Exist")

End If
```

We've now wrapped up our code in an If Statement. The first line of the If Statement is this:

```
If System.IO.File.Exists( FILE_NAME ) = True Then
```

This tests to see whether or not a file exists. Again, you start with System.IO. Then you access another object of System.IO - the File object. This has a method called Exists. In between the round brackets, you type the name (or variable) of the file you want to check. The value returned will either be True (if it does exist), or False (if it doesn't).

If the file exists then we can go ahead and create our StreamReader; If it doesn't, we can display an error message for the user.

When you have done that, run your programme again. Click the button once more, and you should see the text from your file appear in the textbox. (If you get the error message again, it means you haven't copied the file to the right place.)

The StreamReader Class

The StreamReader class also inherits from the abstract base class TextReader that represents a reader for reading series of characters.

```
Imports System.IO

Module fileProg

  Sub Main()

    Try

      ' Create an instance of StreamReader to read from a file.

      ' The using statement also closes the StreamReader.

      Using sr As StreamReader = New StreamReader("e:\test.txt")

        Dim line As String

        ' Read and display lines from the file until the end of the file is reached.

        line = sr.ReadLine()

        While (line <> Nothing)

          Console.WriteLine(line)

          line = sr.ReadLine()

        End While

      End Using

    Catch e As Exception

      ' Let the user know what went wrong.

      Console.WriteLine("The file could not be read:")

      Console.WriteLine(e.Message)

    End Try

    Console.ReadKey()

  End Sub

End Module
```

The StreamWriter Class

The StreamWriter class inherits from the abstract class TextWriter that represents a writer, which can write a series of character.

```
Imports System.IO
Module fileProg
  Sub Main()
    Dim names As String() = New String() {"Sadhana", “Hyndavi", "Akshra", "Akshaya"}
    Dim s As String
    Using sw As StreamWriter = New StreamWriter("names.txt")
      For Each s In names
        sw.WriteLine(s)
      Next s
    End Using
    ' Read and show each line from the file.
    Dim line As String
    Using sr As StreamReader = New StreamReader("names.txt")
      line = sr.ReadLine()
      While (line <> Nothing)
        Console.WriteLine(line)
        line = sr.ReadLine()
      End While
    End Using
    Console.ReadKey()
  End Sub
End Module
```