

Date: 2/12/2005 9:52:18 PM

Reading and Writing files using [Microsoft](#) .NET class libraries

Why write in files ?

You might be thinking that why would someone would write a file when we have all the other databases available in which we can store our data. There are many [applications](#) in which we need to backup the database or to send the data to another user through email. In all those applications you can use any database but the same operations can also be performed using a [file system](#).

Another good reason could be some companies cannot afford to buy expensive databases and their only choice to collect data is in the form of files. Also there could be some operations for which files will be the best option.

Different types of files ?

There are infinite types of files. If I want to define all of them I simply can't since they are infinite. I will only use the most common type file system. This file system includes text files, binary files and [xml files](#). In this article we will see how we can write and read these different type of files.

Writing Text Files :

Text files are the most common type of files. You can make text files in every editor, [notepad](#) is preferably used to make text files. The [extension](#) of the text file is .txt . Lets see what Microsoft .net class libraries provide us for writing and reading text files. Lets first make a simple interface that lets us enter some data in the multi-line [textbox](#).

Button Click Code:

After making the interface let see some of the code. All the code in this example will be written in the Button click event handler since we want the file to be written when the button is clicked.

```
// Don't forget to import System.IO namespace

private void Button1_Click(object sender, System.EventArgs e)
{
    FileStream fs = File.Create(Server.MapPath("test.txt"));
    StreamWriter sw = new StreamWriter(fs);
    sw.Write(TextBox1.Text);
    sw.Close();
    fs.Close();
}
```

As you can see the code is really simple. Lets examine the code and see what's going on.

1. First of all don't forget to import the System.IO namespace.
2. Make the object of FileStream class. The FileStream class is your best option when you don't care about the internal structure of the files with which you're working.
3. Next we make the object of the StreamWriter class. You can think of StreamWriter class as an additional layer of functionality on top of the FileStream class.
4. File.Create method makes a new file. The Server.MapPath means that file will be created in the virtual folder of IIS which will be located at C:\inetpub\wwwroot\YourProjectFolder\
5. Next we write the Text from the TextBox to the file using the StreamWriter Write method.
6. Finally and the most important step is to close the FileStream object explicitly using the Close method.

There are many other ways of writing text to files. One of the way is to use the TextWriter class. For more information about the TextWriter class visit this link <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfsystemiotextwriterclasstopic.asp>

Reading Text Files :

We have seen how we can write the Text Files lets see how we can read from the text files. Reading is also as simple as writing the text files. Lets look at the reading code below:

```
private void Button1_Click(object sender, System.EventArgs e)
{
    FileStream fs = File.OpenRead(Server.MapPath("test.txt"));
    StreamReader sr = new StreamReader(fs);

    while(sr.Peek() > -1)
    {
        Response.Write(sr.ReadLine());
    }
    sr.Close();
    fs.Close();
}
```

Lets see what the code does:

1. First we made the FileStream object and tells it to open the file "test.txt".
2. We make the StreamReader object and tell it what FileStream to read.
3. The while loop checks the end of the file when the end of the file is reached sr.Peek() returns '-1' and the loop breaks.
4. Finally we close the StreamReader and FileStream objects.

Read more on [Reading and Writing Binary Files](#) and [XML File operations](#)