Summary

The Read a text file section of this article describes how to use the `StreamReader` class to read a text file. The Write a text file (example 1) and the Write a text file (example 2) sections describe how to use the `StreamWriter` class to write text to a file.

Read a text file

The following code uses the `StreamReader` class to open, to read, and to close the text file. You can pass the path of a text file to the `StreamReader` constructor to open the file automatically. The `ReadLine` method reads each line of text, and increments the file pointer to the next line as it reads. When the `ReadLine` method reaches the end of the file, it returns a null reference. For more information, see StreamReader Class.

1. Create a sample text file in Notepad. Follow these steps:
   a. Paste the *hello world* text in Notepad.
   b. Save the file as *Sample.txt*.

2. Start Microsoft Visual Studio.

3. On the **File** menu, point to **New**, and then select **Project**.

4. Select **Visual C# Projects** under **Project Types**, and then select **Console Application** under **Templates**.

5. Add the following code at the beginning of the *Class1.cs* file:

```
C#                                                    Copy

using System.IO;
```

6. Add the following code to the `Main` method:

```
C#                                                    Copy

String line;
try
{
    //Pass the file path and file name to the StreamReader constructor
    StreamReader sr = new StreamReader("C:\\Sample.txt");
    //Read the first line of text
    line = sr.ReadLine();
    //Continue to read until you reach end of file
    while (line != null)
    {
        //write the line to console window
        Console.WriteLine(line);
```

```
        //Read the next line
        line = sr.ReadLine();
    }
    //close the file
    sr.Close();
    Console.ReadLine();
}
catch(Exception e)
{
    Console.WriteLine("Exception: " + e.Message);
}
finally
{
    Console.WriteLine("Executing finally block.");
}
```

7. On the **Debug** menu, select **Start** to compile and to run the application. Press ENTER to close the **Console** window. The **Console** window displays the contents of the *Sample.txt* file:

```
Console
```

Copy

```
Hello world
```

Write a text file (example 1)

The following code uses the `StreamWriter` class to open, to write, and to close the text file. In a similar way to the `StreamReader` class, you can pass the path of a text file to the `StreamWriter` constructor to open the file automatically. The `WriteLine` method writes a complete line of text to the text file.

1. Start Visual Studio.
2. On the **File** menu, point to **New**, and then select **Project**.
3. Select **Visual C# Projects** under **Project Types**, and then select **Console Application** under **Templates**.
4. Add the following code at the beginning of the *Class1.cs* file:

```
C#
```

Copy

```
using System.IO;
```

5. Add the following code to the `Main` method:

C# Copy

```
try
{
    //Pass the filepath and filename to the StreamWriter Constructor
    StreamWriter sw = new StreamWriter("C:\\Test.txt");
    //Write a line of text
    sw.WriteLine("Hello World!!");
    //Write a second line of text
    sw.WriteLine("From the StreamWriter class");
    //Close the file
    sw.Close();
}
catch(Exception e)
{
    Console.WriteLine("Exception: " + e.Message);
}
finally
{
    Console.WriteLine("Executing finally block.");
}
```

6. On the **Debug** menu, select **Start** to compile and to run the application. This code creates a file that is named *Test.txt* on drive C. Open *Test.txt* in a text editor such as Notepad. *Test.txt* contains two lines of text:

Console Copy

```
Hello World!!
From the StreamWriter class
```

Write a text file (example 2)

The following code uses the `StreamWriter` class to open, to write, and to close the text file. Unlike the previous example, this code passes two additional parameters to the constructor. The first parameter is the file path and the file name of the file. The second parameter, `true`, specifies that the file is opened in append mode. If you specify `false` for the second parameter, the contents of the file are overwritten each time you run the code. The third parameter specifies `Unicode`, so that `StreamWriter` encodes the file in Unicode format. You can also specify the following encoding methods for the third parameter:

- ASC11
- Unicode
- UTF7

- UTF8

The `Write` method is similar to the `WriteLine` method, except that the `Write` method doesn't automatically embed a carriage return or line feed (CR/LF) character combination. It's useful when you want to write one character at a time.

1. Start Visual Studio.
2. On the **File** menu, point to **New**, and then click **Project**.
3. Click **Visual C# Projects** under **Project Types**, and then click **Console Application** under **Templates**.
4. Add the following code at the beginning of the *Class1.cs* file:

C# Copy

```
using System.IO;
using System.Text;
```

5. Add the following code to the `Main` method:

C# Copy

```
Int64 x;
try
{
    //Open the File
    StreamWriter sw = new StreamWriter("C:\\Test1.txt", true,
Encoding.ASCII);

    //Write out the numbers 1 to 10 on the same line.
    for(x=0; x < 10; x++)
    {
    sw.Write(x);
    }

    //close the file
    sw.Close();
}
catch(Exception e)
{
    Console.WriteLine("Exception: " + e.Message);
}
finally
{
    Console.WriteLine("Executing finally block.");
}
```

6. On the **Debug** menu, select **Start** to compile and to run the application. This code

creates a file that is named *Test1.txt* on drive C. Open *Test1.txt* in a text editor such as Notepad. *Test1.txt* contains a single line of text: *0123456789*.

Complete code listing for how to read a text file

C# Copy

```
//Read a Text File
using System;
using System.IO;
namespace readwriteapp
{
    class Class1
    {
        [STAThread]
        static void Main(string[] args)
        {
            String line;
            try
            {
                //Pass the file path and file name to the StreamReader con-
structor
                StreamReader sr = new StreamReader("C:\\Sample.txt");
                //Read the first line of text
                line = sr.ReadLine();
                //Continue to read until you reach end of file
                while (line != null)
                {
                    //write the lie to console window
                    Console.WriteLine(line);
                    //Read the next line
                    line = sr.ReadLine();
                }
                //close the file
                sr.Close();
                Console.ReadLine();
            }
            catch(Exception e)
            {
                Console.WriteLine("Exception: " + e.Message);
            }
            finally
            {
                Console.WriteLine("Executing finally block.");
            }
        }
    }
}
```

Complete code listing for how to write a text file (version 1)

C# Copy

```
//Write a text file - Version-1
using System;
using System.IO;
namespace readwriteapp
{
    class Class1
    {
        [STAThread]
        static void Main(string[] args)
        {
            try
            {
                //Pass the filepath and filename to the StreamWriter
Constructor
                StreamWriter sw = new StreamWriter("C:\\Test.txt");
                //Write a line of text
                sw.WriteLine("Hello World!!");
                //Write a second line of text
                sw.WriteLine("From the StreamWriter class");
                //Close the file
                sw.Close();
            }
            catch(Exception e)
            {
                Console.WriteLine("Exception: " + e.Message);
            }
            finally
            {
                Console.WriteLine("Executing finally block.");
            }
        }
    }
}
```

Complete code listing for how to write a text file (version 2)

C# Copy

```
//Write a text file - Version 2
using System;
using System.IO;
using System.Text;
```

```
namespace readwriteapp
{
    class Class1
    {
        [STAThread]
        static void Main(string[] args)
        {
            Int64 x;
            try
            {
                //Open the File
                StreamWriter sw = new StreamWriter("C:\\Test1.txt", true,
Encoding.ASCII);
                //Writeout the numbers 1 to 10 on the same line.
                for(x=0; x < 10; x++)
                {
                    sw.Write(x);
                }
                //close the file
                sw.Close();
            }
            catch(Exception e)
            {
                Console.WriteLine("Exception: " + e.Message);
            }
            finally
            {
                Console.WriteLine("Executing finally block.");
            }
        }
    }
}
```

Troubleshoot

For all file manipulations, it's good programming practice to wrap the code inside a `try-catch-finally` block to handle errors and exceptions. Specifically, you may want to release handles to the file in the final block so that the file isn't locked indefinitely. Some possible errors include a file that doesn't exist, or a file that is already in use.

Recommended content

How to read a text file one line at a time - C# Programming Guide

Learn how to read a text file one line at a time. See a code example and view additional available resources.

How to create a file or folder - C# Programming Guide

Learn how to create a file or folder programmatically. You can create a folder, a subfolder, a file in the subfolder, and write data to that file.

How to read from a text file - C# Programming Guide

Learn how to read from a text file using static methods from the File class. See a code example and view additional available resources.

File.AppendText(String) Method (System.IO)

Creates a StreamWriter that appends UTF-8 encoded text to an existing file, or to a new file if the specified file does not exist.

Show more