# How to Read and Write a Text File in C#?

Last Updated : 01 Apr, 2020



Termination of a program leads to the deletion of all data related to it. Therefore, we need to store the data somewhere. Files are used for permanently storing and sharing data. C# can be used to retrieve and manipulate data stored in text files.

**Reading a Text file:** The file class in C# defines two static methods to read a text file namely *File.ReadAllText()* and *File.ReadAllLines()*.

- The File.ReadAllText() reads the entire file at once and returns a string. We need to store this string in a variable and use it to display the contents onto the screen.
- The File.ReadAllLines() reads a file one line at a time and returns that line in string format. We need an array of strings to store each line. We display the contents of the file using the same string array.

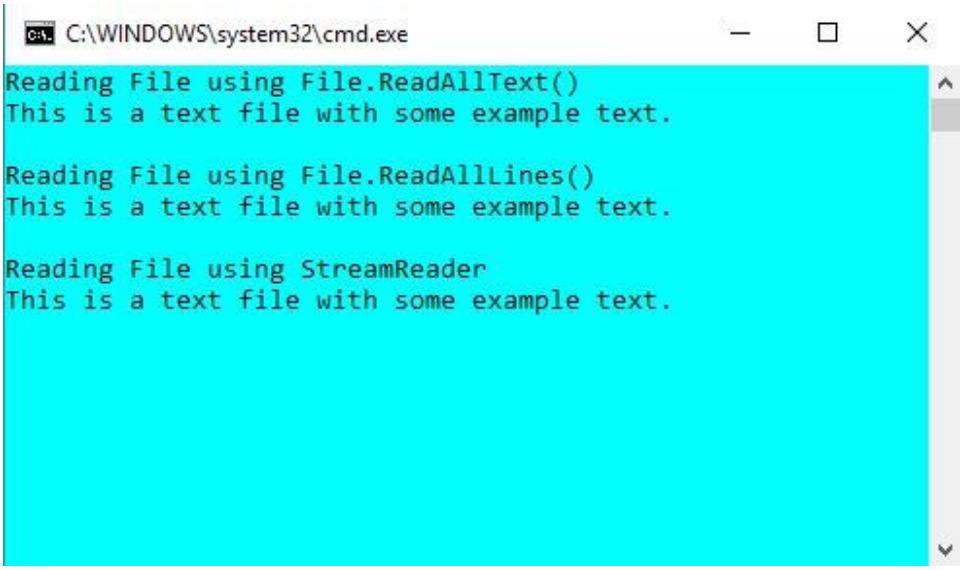
There is another way to read a file and that is by using a StreamReader object. The StreamReader also reads one line at a time and returns a string. All of the above-mentioned ways to read a file are illustrated in the example code given below.

```
// C# program to illustrate how
// to read a file in C#
using System;
using System.IO;

class Program {
    static void Main(string[] args)
    {
        // Store the path of the textfile in your system
        string file = @"M:\Documents\Textfile.txt";
```

```
        if (File.Exists(file)) {
            // Read all the content in one string
            // and display the string
            string str = File.ReadAllText(file);
            Console.WriteLine(str);
        }
        Console.WriteLine();

        Console.WriteLine("Reading File using File.ReadAllLines()");

        // To read a text file line by line
        if (File.Exists(file)) {
            // Store each line in array of strings
            string[] lines = File.ReadAllLines(file);

            foreach(string ln in lines)
                Console.WriteLine(ln);
        }
        Console.WriteLine();

        Console.WriteLine("Reading File using StreamReader");

        // By using StreamReader
        if (File.Exists(file)) {
            // Reads file line by line
            StreamReader Textfile = new StreamReader(file);
            string line;

            while ((line = Textfile.ReadLine()) != null) {
                Console.WriteLine(line);
            }

            Textfile.Close();

            Console.ReadKey();
        }
        Console.WriteLine();
    }
}
```

To run this program, save the file with *.cs* extension and then can execute using *csc filename.cs* command on cmd. Or you can use the Visual Studio. Here, we have a text file named as *Textfile.txt* which have the content shown in the output.

**Output:**




**Writing a Text File:** The File class in C# defines two static methods to write a text file namely *File.WriteAllText()* and *File.WriteAllLines()*.

- The File.WriteAllText() writes the entire file at once. It takes two arguments, the path of the file and the text that has to be written.
- The File.WriteAllLines() writes a file one line at a time. It takes two arguments, the path of the file and the text that has to be written, which is a string array.

There is another way to write to a file and that is by using a StreamWriter object. The StreamWriter also writes one line at a time. All of the three writing methods create a new file if the file doesn't exist, but if the file is already present in that specified location then it is overwritten. All of the above-mentioned ways to write to a text file are illustrated in the example code given below.

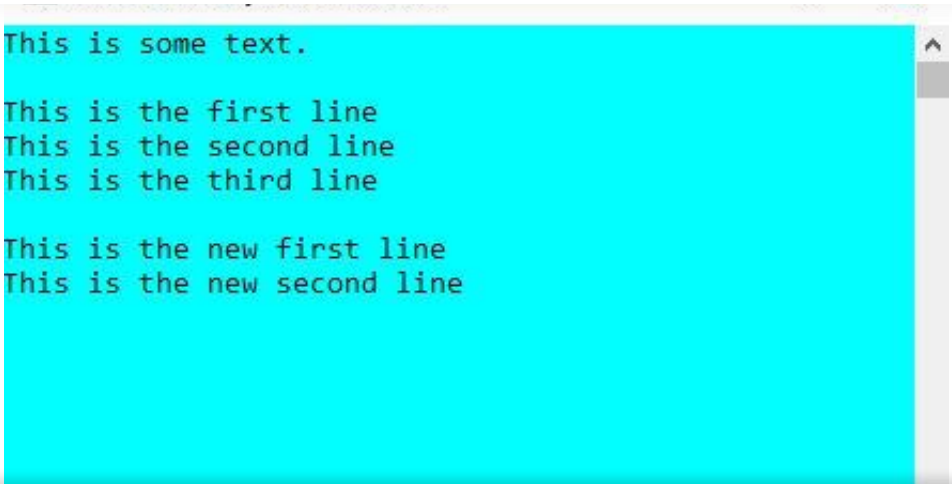
```
// C# program to illustrate how
// to write a file in C#
using System;
using System.IO;

class Program {
    static void Main(string[] args)
    {
        // Store the path of the textfile in your system
        string file = @"M:\Documents\Textfile.txt";

        // To write all of the text to the file
        string text = "This is some text.";
        File.WriteAllText(file, text);
```

```
        // To write text to file line by line
        string[] textLines1 = { "This is the first line",
                                "This is the second line",
                               "This is the third line" };

        File.WriteAllLines(file, textLines1);

        // To display current contents of the file
        Console.WriteLine(File.ReadAllText(file));

        // To write to a file using StreamWriter
        // Writes line by line
        string[] textLines2 = { "This is the new first line",
                              "This is the new second line" };

        using(StreamWriter writer = new StreamWriter(file))
        {
            foreach(string ln in textLines2)
            {
                writer.WriteLine(ln);
            }
        }
        // To display current contents of the file
        Console.WriteLine(File.ReadAllText(file));

        Console.ReadKey();
    }
}
```

To run this program, save the file with *.cs* extension and then can execute using *csc filename.cs* command on cmd. Or you can use the Visual Studio.

**Output:**

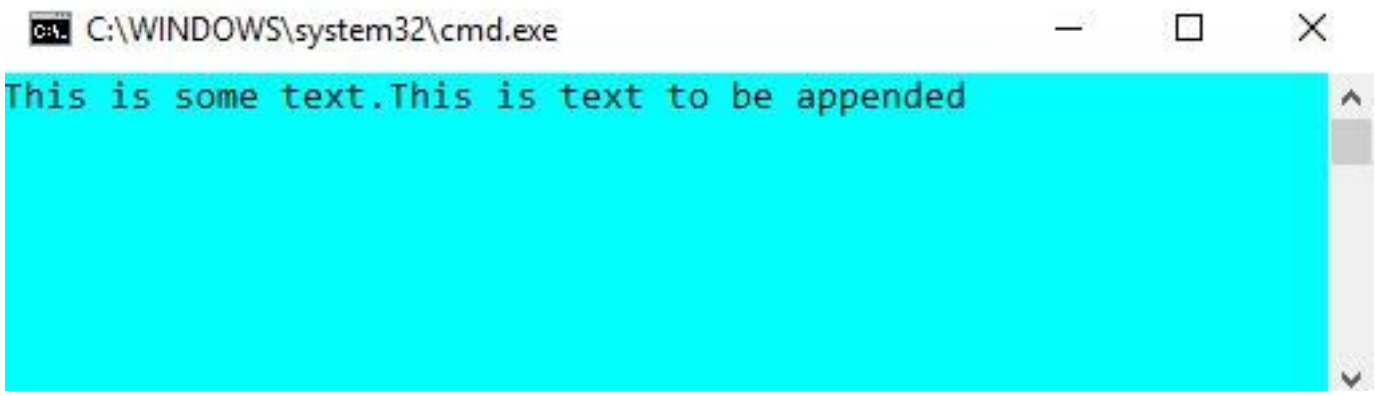




In case you want to add more text to an existing file without overwriting the data already stored in it, you can use the append methods provided by the File class of System.IO.

```
using System;
using System.IO;

class Program {
    static void Main(string[] args)
    {
        // Store the path of the textfile in your system
        string file = @"M:\Documents\Textfile.txt";

        // To write all of the text to the file
        string text1 = "This is some text.";
        File.WriteAllText(file, text1);

        // To append text to a file
        string text2 = "This is text to be appended";
        File.AppendAllText(file, text2);

        // To display current contents of the file
        Console.WriteLine(File.ReadAllText(file));
        Console.ReadKey();
    }
}
```

**Output:**




## Related Articles

## Start Your Coding Journey Now!

1. C# Program to Read and Write a Byte Array to File using FileStream Class
2. C# Program To Copy Content Of One File To Another File By Overwriting Same File Name
3. C# - Reading Lines From a File Until the End of File is Reached
4. C# Program to Get File Time Using File Class
5. C# - Copying the Contents From One File to Another File
6. Basic CRUD (Create, Read, Update, Delete) in ASP.NET MVC Using C# and Entity Framework
7. Difference between Console.Read and Console.ReadLine in C#
8. C# Program to Read a String and Find the Sum of all Digits
9. C# | Check if the StringCollection is read-only
10. C# | Check if the ArrayList is read-only



Previous Next

**Article Contributed By :**

## Vote for difficulty

Easy | Normal | Medium | Hard | Expert

**Article Tags :** CSharp-File-Handling, C#

Improve Article | Report Issue

GeeksforGeeks
A-143, 9th Floor, Sovereign Corporate Tower,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org