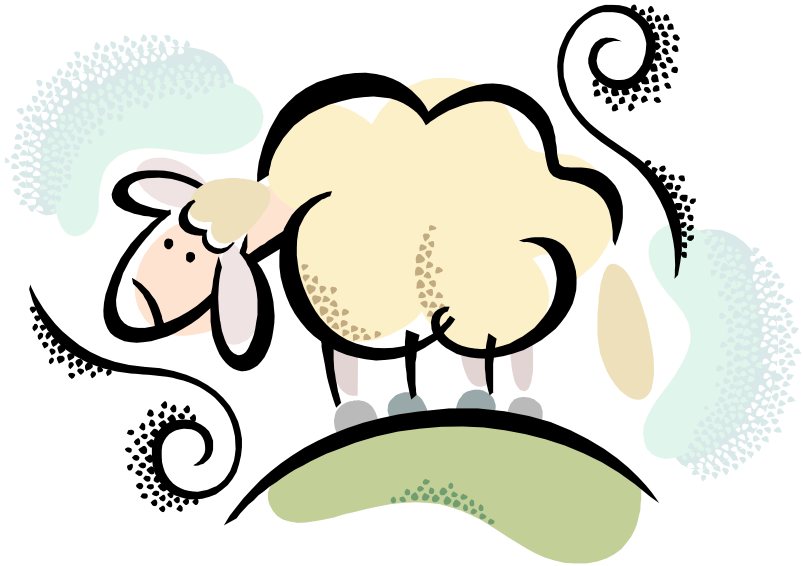

C# Programming IV-6:

System.IO and System.IO.Compression Namespaces

Legal Stuff

This presentation as well as other presentations and documents found on the sheepsqueezers.com website may contain quoted material from outside sources such as books, articles and websites. It is our intention to diligently reference all outside sources. Occasionally, though, a reference may be missed. No copyright infringement whatsoever is intended, and all outside source materials are copyright of their respective author(s).

.NET Lecture Series

sheepsqueezers.com

	C# Programming I: Concepts of OOP	C# Programming II: Beginning C#	C# Programming III: Advanced C#	
C# Programming IV-1: System Namespace	C# Programming IV-2: System.Collections Namespace	C# Programming IV-3: System.Collections. Generic Namespace	C# Programming IV-4A: System.Data Namespace	C# Programming IV-4B: System.Data.Odbc Namespace
C# Programming IV-4C: System.Data.OleDb Namespace	C# Programming IV-4D: Oracle.DataAccess.Client Namespace	C# Programming IV-4E: System.Data.SqlClient Namespace	C# Programming IV-4F: System.Data.SqlTypes Namespace	C# Programming IV-5: System.Drawing/(2D) Namespace
C# Programming IV-6: System.IO Namespace	C# Programming IV-7: System.Numerics	C# Programming IV-8: System.Text and System.Text. RegularExpressions Namespaces	C# Programming V: Introduction to LINQ	

C# Self-Inflicted Project #1 Address Cleaning

C# Self-Inflicted Project #2 Large Intersection Problem

Charting Our Course

- The `System.IO` and `System.IO.Compression` Namespaces
- What Next?

The `System.IO` and `System.IO.Compression` Namespaces

The `System.IO` namespace is defined by Microsoft as follows:

The System.IO namespace contains types that allow reading and writing to files and data streams, and types that provide basic file and directory support.

The System.IO.Compression namespace contains classes that provide basic compression and decompression services for streams.

When writing code using these namespaces, include one or both of the following lines at the top of your C# program:

```
using System.IO;
using System.IO.Compression;
```

Introduction

Introduction

The `System.IO` namespace is comprised of several classes dealing with reading data from and writing data to streams (such as files) as well as providing information on the drives, directories and files available to you.

Now, there are several ways to go when reading/writing data.

1. Reading/Writing pure text characters ➔ Use `StreamReader` and `StreamWriter`
2. Reading/Writing binary data ➔ Use `BinaryReader` and `BinaryWriter`
3. Reading/Writing to and from a String variable ➔ Use `StringReader` and StringWriter
4. Reading/Writing a an in-memory file ➔ Use `MemoryStream`
5. Reading/Writing binary data to a file with Random Access ➔ Use `FileStream` along with its `Seek` method
6. Synchronous/Asynchronous Access to a File ➔ Use `FileStream`

Note that the classes `Stream`, `TextReader` and `TextWriter` are abstract base classes. Both `StringReader` and `StreamReader` are derived from `TextReader` whereas `StringWriter` and `StreamWriter` are derived from `TextReader`.

The `System.IO.Compression` Namespace

- → Classes
  - → `DeflateStream`

DeflateStream

The `DeflateStream` class provides methods and properties for compressing and decompressing streams using the Deflate algorithm.

Constructors

- DeflateStream(Stream, CompressionMode) - Initializes a new instance of the DeflateStream class using the specified stream and CompressionMode value.
- DeflateStream(Stream, CompressionMode, Boolean) - Initializes a new instance of the DeflateStream class using the specified stream and CompressionMode value, and a value that specifies whether to leave the stream open.

Properties

- BaseStream - Gets a reference to the underlying stream.
- CanRead - Gets a value indicating whether the stream supports reading while decompressing a file. (Overrides Stream.CanRead.)
- CanSeek - Gets a value indicating whether the stream supports seeking. (Overrides Stream.CanSeek.)
- CanTimeout - Gets a value that determines whether the current stream can time out. (Inherited from Stream.)
- CanWrite - Gets a value indicating whether the stream supports writing. (Overrides Stream.CanWrite.)
- Length - This property is not supported and always throws a NotSupportedException. (Overrides Stream.Length.)
- Position - This property is not supported and always throws a NotSupportedException. (Overrides Stream.Position.)
- ReadTimeout - Gets or sets a value, in miliseconds, that determines how long the stream will attempt to read before timing out. (Inherited from Stream.)
- WriteTimeout - Gets or sets a value, in miliseconds, that determines how long the stream will attempt to write before timing out. (Inherited from Stream.)

Methods

- BeginRead - Begins an asynchronous read operation. (Overrides Stream.BeginRead(Byte[], Int32, Int32, AsyncCallback, Object).)
- BeginWrite - Begins an asynchronous write operation. (Overrides Stream.BeginWrite(Byte[], Int32, Int32, AsyncCallback, Object).)
- Close - Closes the current stream and releases any resources (such as sockets and file handles) associated with the current stream. (Inherited from Stream.)

Methods (continued)

- CopyTo(Stream) - Reads the bytes from the current stream and writes them to the destination stream. (Inherited from Stream.)
- CopyTo(Stream, Int32) - Reads all the bytes from the current stream and writes them to a destination stream, using a specified buffer size. (Inherited from Stream.)
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- CreateWaitHandle - Obsolete. Allocates a WaitHandle object. (Inherited from Stream.)
- Dispose() - Releases all resources used by the Stream. (Inherited from Stream.)
- Dispose(Boolean) - Releases the unmanaged resources used by the DeflateStream and optionally releases the managed resources. (Overrides Stream.Dispose(Boolean).)
- EndRead - Waits for the pending asynchronous read to complete. (Overrides Stream.EndRead(IAsyncResult).)
- EndWrite - Ends an asynchronous write operation. (Overrides Stream.EndWrite(IAsyncResult).)
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- Flush - Flushes the contents of the internal buffer of the current stream object to the underlying stream. (Overrides Stream.Flush().)
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- ObjectInvariant - Infrastructure. Provides support for a Contract. (Inherited from Stream.)
- Read - Reads a number of decompressed bytes into the specified byte array. (Overrides Stream.Read(Byte[], Int32, Int32).)
- ReadByte - Reads a byte from the stream and advances the position within the stream by one byte, or returns -1 if at the end of the stream. (Inherited from Stream.)
- Seek - This operation is not supported and always throws a NotSupportedException. (Overrides Stream.Seek(Int64, SeekOrigin).)

Methods (continued)

- SetLength - This operation is not supported and always throws a NotSupportedException. (Overrides Stream.SetLength(Int64).)
- ToString - Returns a string that represents the current object. (Inherited from Object.)
- Write - Writes compressed bytes to the underlying stream from the specified byte array. (Overrides Stream.Write(Byte[], Int32, Int32).)
- WriteByte - Writes a byte to the current position in the stream and advances the position within the stream by one byte. (Inherited from Stream.)

The `System.IO.Compression` Namespace

- → Classes
  - → `GZipStream`

GZipStream

The `GZipStream` class provides methods and properties used to compress and decompress streams.

Constructors

- GZipStream(Stream, CompressionMode) - Initializes a new instance of the GZipStream class using the specified stream and CompressionMode value.
- GZipStream(Stream, CompressionMode, Boolean) - Initializes a new instance of the GZipStream class using the specified stream and CompressionMode value, and a value that specifies whether to leave the stream open.

Properties

- BaseStream - Gets a reference to the underlying stream.
- CanRead - Gets a value indicating whether the stream supports reading while decompressing a file. (Overrides Stream.CanRead.)
- CanSeek - Gets a value indicating whether the stream supports seeking. (Overrides Stream.CanSeek.)
- CanTimeout - Gets a value that determines whether the current stream can time out. (Inherited from Stream.)
- CanWrite - Gets a value indicating whether the stream supports writing. (Overrides Stream.CanWrite.)
- Length - This property is not supported and always throws a NotSupportedException. (Overrides Stream.Length.)
- Position - This property is not supported and always throws a NotSupportedException. (Overrides Stream.Position.)
- ReadTimeout - Gets or sets a value, in miliseconds, that determines how long the stream will attempt to read before timing out. (Inherited from Stream.)
- WriteTimeout - Gets or sets a value, in miliseconds, that determines how long the stream will attempt to write before timing out. (Inherited from Stream.)

Methods

- BeginRead - Begins an asynchronous read operation. (Overrides Stream.BeginRead(Byte[], Int32, Int32, AsyncCallback, Object).)
- BeginWrite - Begins an asynchronous write operation. (Overrides Stream.BeginWrite(Byte[], Int32, Int32, AsyncCallback, Object).)
- Close - Closes the current stream and releases any resources (such as sockets and file handles) associated with the current stream. (Inherited from Stream.)

Methods (continued)

- CopyTo(Stream) - Reads the bytes from the current stream and writes them to the destination stream. (Inherited from Stream.)
- CopyTo(Stream, Int32) - Reads all the bytes from the current stream and writes them to a destination stream, using a specified buffer size. (Inherited from Stream.)
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- CreateWaitHandle - Obsolete. Allocates a WaitHandle object. (Inherited from Stream.)
- Dispose() - Releases all resources used by the Stream. (Inherited from Stream.)
- Dispose(Boolean) - Releases the unmanaged resources used by the GZipStream and optionally releases the managed resources. (Overrides Stream.Dispose(Boolean).)
- EndRead - Waits for the pending asynchronous read to complete. (Overrides Stream.EndRead(IAsyncResult).)
- EndWrite - Handles the end of an asynchronous write operation. (Overrides Stream.EndWrite(IAsyncResult).)
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- Flush - Flushes the contents of the internal buffer of the current GZipStream object to the underlying stream. (Overrides Stream.Flush().)
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- Read - Reads a number of decompressed bytes into the specified byte array. (Overrides Stream.Read(Byte[], Int32, Int32).)
- ReadByte - Reads a byte from the stream and advances the position within the stream by one byte, or returns -1 if at the end of the stream. (Inherited from Stream.)
- ToString - Returns a string that represents the current object. (Inherited from Object.)
- Write - Writes compressed bytes to the underlying stream from the specified byte array. (Overrides Stream.Write(Byte[], Int32, Int32).)
- WriteByte - Writes a byte to the current position in the stream and advances the position within the stream by one byte. (Inherited from Stream.)

The `System.IO` Namespace

- → Classes
  - → `FileSystemInfo`

FileSystemInfo

The `FileSystemInfo` class provides the base class for both `FileInfo` and `DirectoryInfo` objects. According to Microsoft's website: *The FileSystemInfo class contains methods that are common to file and directory manipulation. A FileSystemInfo object can represent either a file or a directory, thus serving as the basis for FileInfo or DirectoryInfo objects. Use this base class when parsing a lot of files and directories.*

Constructors

- FileSystemInfo() - Initializes a new instance of the FileSystemInfo class.
- FileSystemInfo(SerializationInfo, StreamingContext) - Initializes a new instance of the FileSystemInfo class with serialized data.

Properties

- Attributes - Gets or sets the attributes for the current file or directory.
- CreationTime - Gets or sets the creation time of the current file or directory.
- CreationTimeUtc - Gets or sets the creation time, in coordinated universal time (UTC), of the current file or directory.
- Exists - Gets a value indicating whether the file or directory exists.
- Extension - Gets the string representing the extension part of the file.
- FullName - Gets the full path of the directory or file.
- LastAccessTime - Gets or sets the time the current file or directory was last accessed.
- LastAccessTimeUtc - Gets or sets the time, in coordinated universal time (UTC), that the current file or directory was last accessed.
- LastWriteTime - Gets or sets the time when the current file or directory was last written to.
- LastWriteTimeUtc - Gets or sets the time, in coordinated universal time (UTC), when the current file or directory was last written to.
- Name - For files, gets the name of the file. For directories, gets the name of the last directory in the hierarchy if a hierarchy exists. Otherwise, the Name property gets the name of the directory.

FileSystemInfo

Methods

- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- Delete - Deletes a file or directory.
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetObjectData - Sets the SerializationInfo object with the file name and additional exception information.
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- Refresh - Refreshes the state of the object.
- ToString - Returns a string that represents the current object. (Inherited from Object.)

Fields

- FullPath - Represents the fully qualified path of the directory or file.
- OriginalPath - The path originally specified by the user, whether relative or absolute.

I am still confused as to why you would use this abstract base class rather than use the derived (concrete) classes `FileInfo` and `DirectoryInfo`!!

The `System.IO` Namespace

- → Classes
  - → `DirectoryInfo`

DirectoryInfo

The `DirectoryInfo` class exposes instance methods for creating, moving, and enumerating through directories and subdirectories.

Constructors

- DirectoryInfo - Initializes a new instance of the DirectoryInfo class on the specified path.

Properties

- Attributes - Gets or sets the attributes for the current file or directory. (Inherited from FileSystemInfo.)
- CreationTime - Gets or sets the creation time of the current file or directory. (Inherited from FileSystemInfo.)
- CreationTimeUtc - Gets or sets the creation time, in coordinated universal time (UTC), of the current file or directory. (Inherited from FileSystemInfo.)
- Exists - Gets a value indicating whether the directory exists. (Overrides FileSystemInfo.Exists.)
- Extension - Gets the string representing the extension part of the file. (Inherited from FileSystemInfo.)
- FullName - Gets the full path of the directory or file. (Inherited from FileSystemInfo.)
- LastAccessTime - Gets or sets the time the current file or directory was last accessed. (Inherited from FileSystemInfo.)
- LastAccessTimeUtc - Gets or sets the time, in coordinated universal time (UTC), that the current file or directory was last accessed. (Inherited from FileSystemInfo.)
- LastWriteTime - Gets or sets the time when the current file or directory was last written to. (Inherited from FileSystemInfo.)
- LastWriteTimeUtc - Gets or sets the time, in coordinated universal time (UTC), when the current file or directory was last written to. (Inherited from FileSystemInfo.)
- Name - Gets the name of this DirectoryInfo instance. (Overrides FileSystemInfo.Name.)
- Parent - Gets the parent directory of a specified subdirectory.
- Root - Gets the root portion of a path.

Methods

- Create() - Creates a directory.
- Create(DirectorySecurity) - Creates a directory using a DirectorySecurity object.
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)

Methods (continued)

- CreateSubdirectory(String) - Creates a subdirectory or subdirectories on the specified path. The specified path can be relative to this instance of the DirectoryInfo class.
- CreateSubdirectory(String, DirectorySecurity) - Creates a subdirectory or subdirectories on the specified path with the specified security. The specified path can be relative to this instance of the DirectoryInfo class.
- Delete() - Deletes this DirectoryInfo if it is empty. (Overrides FileSystemInfo.Delete().)
- Delete(Boolean) - Deletes this instance of a DirectoryInfo, specifying whether to delete subdirectories and files.
- EnumerateDirectories() - Returns an enumerable collection of directory information in the current directory.
- EnumerateDirectories(String) - Returns an enumerable collection of directory information that matches a specified search pattern.
- EnumerateDirectories(String, SearchOption) - Returns an enumerable collection of directory information that matches a specified search pattern and search subdirectory option.
- EnumerateFiles() - Returns an enumerable collection of file information in the current directory.
- EnumerateFiles(String) - Returns an enumerable collection of file information that matches a search pattern.
- EnumerateFiles(String, SearchOption) - Returns an enumerable collection of file information that matches a specified search pattern and search subdirectory option.
- EnumerateFileSystemInfos() - Returns an enumerable collection of file system information in the current directory.
- EnumerateFileSystemInfos(String) - Returns an enumerable collection of file system information that matches a specified search pattern.
- EnumerateFileSystemInfos(String, SearchOption) - Returns an enumerable collection of file system information that matches a specified search pattern and search subdirectory option.
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- GetAccessControl() - Gets a DirectorySecurity object that encapsulates the access control list (ACL) entries for the directory described by the current DirectoryInfo object.
- GetAccessControl(AccessControlSections) - Gets a DirectorySecurity object that encapsulates the specified type of access control list (ACL) entries for the directory described by the current DirectoryInfo object.
- GetDirectories() - Returns the subdirectories of the current directory.
- GetDirectories(String) - Returns an array of directories in the current DirectoryInfo matching the given search criteria.
- GetDirectories(String, SearchOption) - Returns an array of directories in the current DirectoryInfo matching the given search criteria and using a value to determine whether to search subdirectories.

Methods (continued)

- GetFiles() - Returns a file list from the current directory.
- GetFiles(String) - Returns a file list from the current directory matching the given search pattern.
- GetFiles(String, SearchOption) - Returns a file list from the current directory matching the given search pattern and using a value to determine whether to search subdirectories.
- GetFileSystemInfos() - Returns an array of strongly typed FileSystemInfo entries representing all the files and subdirectories in a directory.
- GetFileSystemInfos(String) - Retrieves an array of strongly typed FileSystemInfo objects representing the files and subdirectories that match the specified search criteria.
- GetFileSystemInfos(String, SearchOption) - Retrieves an array of FileSystemInfo objects that represent the files and subdirectories matching the specified search criteria.
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetObjectData - Sets the SerializationInfo object with the file name and additional exception information. (Inherited from FileSystemInfo.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- MoveTo - Moves a DirectoryInfo instance and its contents to a new path.
- Refresh - Refreshes the state of the object. (Inherited from FileSystemInfo.)
- SetAccessControl - Applies access control list (ACL) entries described by a DirectorySecurity object to the directory described by the current DirectoryInfo object.
- ToString - Returns the original path that was passed by the user. (Overrides Object.ToString().)

Fields

- FullPath - Represents the fully qualified path of the directory or file. (Inherited from FileSystemInfo.)
- OriginalPath - The path originally specified by the user, whether relative or absolute. (Inherited from FileSystemInfo.)

DirectoryInfo

Below is an example that determines if a directory exists and, if so, it prints out each file within the directory.

```
using System;
using System.IO;

class MainProgram {

 public static void Main() {

  DirectoryInfo oDI = new DirectoryInfo(@"C:\temp\Flintstones");
  if (oDI.Exists) {

    //List the files in this directory
    Console.WriteLine(@"The following is a list of files that appear in the folder C:\temnp\Flintstones:");
    FileInfo[] oFI = oDI.GetFiles();
    for(Int32 indx=0; indx<oFI.Length; indx++) {
     Console.WriteLine(oFI[indx].Name);
    }

  }
  else {

   Console.WriteLine("Tough luck...the directory does not exist!!");

  }

 }

}
```

The `System.IO` Namespace

- → Classes
  - → `FileInfo`

FileInfo

The `FileInfo` class provides properties and instance methods for the creation, copying, deletion, moving, and opening of files, and aids in the creation of FileStream objects. According to Microsoft's website: *Use the FileInfo class for typical operations such as copying, moving, renaming, creating, opening, deleting, and appending to files. Many of the FileInfo methods return other I/O types when you create or open files. You can use these other types to further manipulate a file. For more information, see specific FileInfo members such as Open, OpenRead, OpenText, CreateText, or Create.*

Constructors

- FileInfo - Initializes a new instance of the FileInfo class, which acts as a wrapper for a file path.

Properties

- Attributes - Gets or sets the attributes for the current file or directory. (Inherited from FileSystemInfo.)
- CreationTime - Gets or sets the creation time of the current file or directory. (Inherited from FileSystemInfo.)
- CreationTimeUtc - Gets or sets the creation time, in coordinated universal time (UTC), of the current file or directory. (Inherited from FileSystemInfo.)
- Directory - Gets an instance of the parent directory.
- DirectoryName - Gets a string representing the directory's full path.
- Exists - Gets a value indicating whether a file exists. (Overrides FileSystemInfo.Exists.)
- Extension - Gets the string representing the extension part of the file. (Inherited from FileSystemInfo.)
- FullName - Gets the full path of the directory or file. (Inherited from FileSystemInfo.)
- IsReadOnly - Gets or sets a value that determines if the current file is read only.
- LastAccessTime - Gets or sets the time the current file or directory was last accessed. (Inherited from FileSystemInfo.)
- LastAccessTimeUtc - Gets or sets the time, in coordinated universal time (UTC), that the current file or directory was last accessed. (Inherited from FileSystemInfo.)
- LastWriteTime - Gets or sets the time when the current file or directory was last written to. (Inherited from FileSystemInfo.)
- LastWriteTimeUtc - Gets or sets the time, in coordinated universal time (UTC), when the current file or directory was last written to. (Inherited from FileSystemInfo.)
- Length - Gets the size, in bytes, of the current file.
- Name - Gets the name of the file. (Overrides FileSystemInfo.Name.)

FileInfo

Methods

- AppendText - Creates a StreamWriter that appends text to the file represented by this instance of the FileInfo.
- CopyTo(String) - Copies an existing file to a new file, disallowing the overwriting of an existing file.
- CopyTo(String, Boolean) - Copies an existing file to a new file, allowing the overwriting of an existing file.
- Create - Creates a file.
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- CreateText - Creates a StreamWriter that writes a new text file.
- Decrypt - Decrypts a file that was encrypted by the current account using the Encrypt method.
- Delete - Permanently deletes a file. (Overrides FileSystemInfo.Delete().)
- Encrypt - Encrypts a file so that only the account used to encrypt the file can decrypt it.
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- GetAccessControl() - Gets a FileSecurity object that encapsulates the access control list (ACL) entries for the file described by the current FileInfo object.
- GetAccessControl(AccessControlSections) - Gets a FileSecurity object that encapsulates the specified type of access control list (ACL) entries for the file described by the current FileInfo object.
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetObjectData - Sets the SerializationInfo object with the file name and additional exception information. (Inherited from FileSystemInfo.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- MoveTo - Moves a specified file to a new location, providing the option to specify a new file name.
- Open(FileMode) - Opens a file in the specified mode.
- Open(FileMode, FileAccess) - Opens a file in the specified mode with read, write, or read/write access.
- Open(FileMode, FileAccess, FileShare) - Opens a file in the specified mode with read, write, or read/write access and the specified sharing option.

FileInfo

Methods

- OpenRead - Creates a read-only FileStream.
- OpenText - Creates a StreamReader with UTF8 encoding that reads from an existing text file.
- OpenWrite - Creates a write-only FileStream.
- Refresh - Refreshes the state of the object. (Inherited from FileSystemInfo.)
- Replace(String, String) - Replaces the contents of a specified file with the file described by the current FileInfo object, deleting the original file, and creating a backup of the replaced file.
- Replace(String, String, Boolean) - Replaces the contents of a specified file with the file described by the current FileInfo object, deleting the original file, and creating a backup of the replaced file. Also specifies whether to ignore merge errors.
- SetAccessControl - Applies access control list (ACL) entries described by a FileSecurity object to the file described by the current FileInfo object.
- ToString - Returns the path as a string. (Overrides Object.ToString().)

Fields

- FullPath - Represents the fully qualified path of the directory or file. (Inherited from FileSystemInfo.)
- OriginalPath - The path originally specified by the user, whether relative or absolute. (Inherited from FileSystemInfo.)

Below is an example that creates a new file and appends text to it.

FileInfo


```
using System;
using System.IO;

class MainProgram {

 public static void Main() {

  FileInfo oFI = new FileInfo(@"C:\temp\Flintstones\Fred.txt");
  if (oFI.Exists && !oFI.IsReadOnly) {

    //Append text to this file
    String sText="Wilma!!!!!!!!!!";
    StreamWriter oSW = oFI.AppendText();
    oSW.WriteLine(sText);
    oSW.Close();

  }
  else {

   Console.WriteLine("ERROR: Cannot write to the file because it either does not exist or is read-only!");

  }

 }

}
```

The `System.IO` Namespace

- → Classes
  - → `DriveInfo`

DriveInfo

The `DriveInfo` class provides access to information on a drive. One nice feature of this class is the static method `GetDrives`.

Constructors

- DriveInfo - Provides access to information on the specified drive.

Properties

- AvailableFreeSpace - Indicates the amount of available free space on a drive.
- DriveFormat - Gets the name of the file system, such as NTFS or FAT32.
- DriveType - Gets the drive type.
- IsReady - Gets a value indicating whether a drive is ready.
- Name - Gets the name of a drive.
- RootDirectory - Gets the root directory of a drive.
- TotalFreeSpace - Gets the total amount of free space available on a drive.
- TotalSize - Gets the total size of storage space on a drive.
- VolumeLabel - Gets or sets the volume label of a drive.

Methods

- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- GetDrives – (STATIC) Retrieves the drive names of all logical drives on a computer.
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- MemberwiseClone - Creates a shallow copy of the current Object. (Inherited from Object.)
- ToString - Returns a drive name as a string. (Overrides Object.ToString().)

DriveInfo


```
using System;
using System.IO;

class MainProgram {

 public static void Main() {

  DriveInfo[] oDrives = DriveInfo.GetDrives();
  for(Int32 indx=0;indx<oDrives.Length; indx++) {
   Console.WriteLine("Drive Available: {0}",oDrives[indx].Name);
  }

 }

}
```

Output is:

```
Drive Available: C:\
Drive Available: D:\
Drive Available: E:\
Drive Available: F:\
```

The `System.IO` Namespace

- → Classes
  - → `Directory`

Directory

The `Directory` class exposes ***static methods*** for creating, moving, and enumerating through directories and subdirectories.

Methods

- CreateDirectory(String) - Creates all directories and subdirectories in the specified path
- CreateDirectory(String, DirectorySecurity) - Creates all the directories in the specified path, applying the specified Windows security
- Delete(String) - Deletes an empty directory from a specified path
- Delete(String, Boolean) - Deletes the specified directory and, if indicated, any subdirectories and files in the directory
- EnumerateDirectories(String) - Returns an enumerable collection of directory names in a specified path
- EnumerateDirectories(String, String) - Returns an enumerable collection of directory names that match a search pattern in a specified path
- EnumerateDirectories(String, String, SearchOption) - Returns an enumerable collection of directory names that match a search pattern in a specified path, and optionally searches subdirectories
- EnumerateFiles(String) - Returns an enumerable collection of file names in a specified path
- EnumerateFiles(String, String) - Returns an enumerable collection of file names that match a search pattern in a specified path
- EnumerateFiles(String, String, SearchOption) - Returns an enumerable collection of file names that match a search pattern in a specified path, and optionally searches subdirectories
- EnumerateFileSystemEntries(String) - Returns an enumerable collection of file-system entries in a specified path
- EnumerateFileSystemEntries(String, String) - Returns an enumerable collection of file-system entries that match a search pattern in a specified path
- EnumerateFileSystemEntries(String, String, SearchOption) - Returns an enumerable collection of file names and directory names that match a search pattern in a specified path, and optionally searches subdirectories
- Exists - Determines whether the given path refers to an existing directory on disk
- GetAccessControl(String) - Gets a DirectorySecurity object that encapsulates the access control list (ACL) entries for a specified directory
- GetAccessControl(String, AccessControlSections) - Gets a DirectorySecurity object that encapsulates the specified type of access control list (ACL) entries for a specified directory
- GetCreationTime - Gets the creation date and time of a directory
- GetCreationTimeUtc - Gets the creation date and time, in Coordinated Universal Time (UTC) format, of a directory
- GetCurrentDirectory - Gets the current working directory of the application

Methods (continued)

- GetDirectories(String) - Gets the names of subdirectories (including their paths) in the specified directory
- GetDirectories(String, String) - Gets an array of directories (including their paths) that match the specified search pattern in the current directory
- GetDirectories(String, String, SearchOption) - Gets the names of the directories (including their paths) that match the specified search pattern in the current directory, and optionally searches subdirectories
- GetDirectoryRoot - Returns the volume information, root information, or both for the specified path
- GetFiles(String) - Returns the names of files (including their paths) in the specified directory
- GetFiles(String, String) - Returns the names of files (including their paths) that match the specified search pattern in the specified directory
- GetFiles(String, String, SearchOption) - Returns the names of files (including their paths) that match the specified search pattern in the specified directory, using a value to determine whether to search subdirectories
- GetFileSystemEntries(String) - Returns the names of all files and subdirectories in the specified directory
- GetFileSystemEntries(String, String) - Returns an array of file system entries that match the specified search criteria
- GetFileSystemEntries(String, String, SearchOption) - Gets an array of all the file names and directory names that match a search pattern in a specified path, and optionally searches subdirectories
- GetLastAccessTime - Returns the date and time the specified file or directory was last accessed
- GetLastAccessTimeUtc - Returns the date and time, in Coordinated Universal Time (UTC) format, that the specified file or directory was last accessed
- GetLastWriteTime - Returns the date and time the specified file or directory was last written to
- GetLastWriteTimeUtc - Returns the date and time, in Coordinated Universal Time (UTC) format, that the specified file or directory was last written to
- GetLogicalDrives - Retrieves the names of the logical drives on this computer in the form "<drive letter>:\"
- GetParent - Retrieves the parent directory of the specified path, including both absolute and relative paths
- Move - Moves a file or a directory and its contents to a new location
- SetAccessControl - Applies access control list (ACL) entries described by a DirectorySecurity object to the specified directory
- SetCreationTime - Sets the creation date and time for the specified file or directory
- SetCreationTimeUtc - Sets the creation date and time, in Coordinated Universal Time (UTC) format, for the specified file or directory
- SetCurrentDirectory - Sets the application's current working directory to the specified directory
- SetLastAccessTime - Sets the date and time the specified file or directory was last accessed
- SetLastAccessTimeUtc - Sets the date and time, in Coordinated Universal Time (UTC) format, that the specified file or directory was last accessed
- SetLastWriteTime - Sets the date and time a directory was last written to
- SetLastWriteTimeUtc - Sets the date and time, in Coordinated Universal Time (UTC) format, that a directory was last written to

Directory

The example below creates a directory if it doesn't already exist and then enumerates all of the sub-directories in my C:\TEMP directory. Note that the static method `EnumerateDirectories` returns a generic `IEnumerable<String>` type!!

```
using System;
using System.Collections.Generic;
using System.IO;

class MainProgram {

 public static void Main() {

  //Create C:\TEMP\HELLO if it does not already exist
  if (!Directory.Exists(@"C:\TEMP\HELLO")) {
   Directory.CreateDirectory(@"C:\TEMP\HELLO");
   Console.WriteLine("Directory Created!!");
  }
  else {
   Console.WriteLine(@"Cannot create C:\TEMP\HELLO because it already exists!!");
  }

  //Enumerate all of the subdirectories under C:\TEMP
  IEnumerable<String> sDirs = Directory.EnumerateDirectories(@"C:\TEMP");
  foreach(String sSubDir in sDirs) {
   Console.WriteLine(sSubDir);
  }

 }

}
```

The `System.IO` Namespace

- → Classes
  - → `File`

File

The `File` class provides ***static methods*** for the creation, copying, deletion, moving, and opening of files, and aids in the creation of `FileStream` objects.

Methods

- AppendAllLines(String, IEnumerable<String>) - Appends lines to a file, and then closes the file
- AppendAllLines(String, IEnumerable<String>, Encoding) - Appends lines to a file by using a specified encoding, and then closes the file
- AppendAllText(String, String) - Opens a file, appends the specified string to the file, and then closes the file. If the file does not exist, this method creates a file, writes the specified string to the file, then closes the file
- AppendAllText(String, String, Encoding) - Appends the specified string to the file, creating the file if it does not already exist
- AppendText - Creates a StreamWriter that appends UTF-8 encoded text to an existing file
- Copy(String, String) - Copies an existing file to a new file. Overwriting a file of the same name is not allowed
- Copy(String, String, Boolean) - Copies an existing file to a new file. Overwriting a file of the same name is allowed
- Create(String) - Creates or overwrites a file in the specified path
- Create(String, Int32) - Creates or overwrites the specified file
- Create(String, Int32, FileOptions) - Creates or overwrites the specified file, specifying a buffer size and a FileOptions value that describes how to create or overwrite the file
- Create(String, Int32, FileOptions, FileSecurity) - Creates or overwrites the specified file with the specified buffer size, file options, and file security
- CreateText - Creates or opens a file for writing UTF-8 encoded text
- Decrypt - Decrypts a file that was encrypted by the current account using the Encrypt method
- Delete - Deletes the specified file
- Encrypt - Encrypts a file so that only the account used to encrypt the file can decrypt it
- Exists - Determines whether the specified file exists
- GetAccessControl(String) - Gets a FileSecurity object that encapsulates the access control list (ACL) entries for a specified file
- GetAccessControl(String, AccessControlSections) - Gets a FileSecurity object that encapsulates the specified type of access control list (ACL) entries for a particular file
- GetAttributes - Gets the FileAttributes of the file on the path
- GetCreationTime - Returns the creation date and time of the specified file or directory
- GetCreationTimeUtc - Returns the creation date and time, in coordinated universal time (UTC), of the specified file or directory
- GetLastAccessTime - Returns the date and time the specified file or directory was last accessed
- GetLastAccessTimeUtc - Returns the date and time, in coordinated universal time (UTC), that the specified file or directory was last accessed

Methods (continued)

- GetLastWriteTime - Returns the date and time the specified file or directory was last written to
- GetLastWriteTimeUtc - Returns the date and time, in coordinated universal time (UTC), that the specified file or directory was last written to
- Move - Moves a specified file to a new location, providing the option to specify a new file name
- Open(String, FileMode) - Opens a FileStream on the specified path with read/write access
- Open(String, FileMode, FileAccess) - Opens a FileStream on the specified path, with the specified mode and access
- Open(String, FileMode, FileAccess, FileShare) - Opens a FileStream on the specified path, having the specified mode with read, write, or read/write access and the specified sharing option
- OpenRead - Opens an existing file for reading
- OpenText - Opens an existing UTF-8 encoded text file for reading
- OpenWrite - Opens an existing file or creates a new file for writing
- ReadAllBytes - Opens a binary file, reads the contents of the file into a byte array, and then closes the file
- ReadAllLines(String) - Opens a text file, reads all lines of the file, and then closes the file
- ReadAllLines(String, Encoding) - Opens a file, reads all lines of the file with the specified encoding, and then closes the file
- ReadAllText(String) - Opens a text file, reads all lines of the file, and then closes the file
- ReadAllText(String, Encoding) - Opens a file, reads all lines of the file with the specified encoding, and then closes the file
- ReadLines(String) - Reads the lines of a file
- ReadLines(String, Encoding) - Read the lines of a file that has a specified encoding
- Replace(String, String, String) - Replaces the contents of a specified file with the contents of another file, deleting the original file, and creating a backup of the replaced file
- Replace(String, String, String, Boolean) - Replaces the contents of a specified file with the contents of another file, deleting the original file, and creating a backup of the replaced file and optionally ignores merge errors
- SetAccessControl - Applies access control list (ACL) entries described by a FileSecurity object to the specified file
- SetAttributes - Sets the specified FileAttributes of the file on the specified path
- SetCreationTime - Sets the date and time the file was created
- SetCreationTimeUtc - Sets the date and time, in coordinated universal time (UTC), that the file was created
- SetLastAccessTime - Sets the date and time the specified file was last accessed
- SetLastAccessTimeUtc - Sets the date and time, in coordinated universal time (UTC), that the specified file was last accessed
- SetLastWriteTime - Sets the date and time that the specified file was last written to
- SetLastWriteTimeUtc - Sets the date and time, in coordinated universal time (UTC), that the specified file was last written to
- WriteAllBytes - Creates a new file, writes the specified byte array to the file, and then closes the file. If the target file already exists, it is overwritten

Methods (continued)

- WriteAllLines(String, IEnumerable<String>) - Creates a new file, writes a collection of strings to the file, and then closes the file
- WriteAllLines(String, String[]) - Creates a new file, write the specified string array to the file, and then closes the file
- WriteAllLines(String, IEnumerable<String>, Encoding) - Creates a new file by using the specified encoding, writes a collection of strings to the file, and then closes the file
- WriteAllLines(String, String[], Encoding) - Creates a new file, writes the specified string array to the file by using the specified encoding, and then closes the file
- WriteAllText(String, String) - Creates a new file, writes the specified string to the file, and then closes the file. If the target file already exists, it is overwritten
- WriteAllText(String, String, Encoding) - Creates a new file, writes the specified string to the file using the specified encoding, and then closes the file. If the target file already exists, it is overwritten

The example below, creates a new file in C:\TEMP\Flintstones called Dino.txt and adds text to it. See next slide.

File


```
using System;
using System.Collections.Generic;
using System.IO;

class MainProgram {

 public static void Main() {

  if (!File.Exists(@"C:\TEMP\Flintstones\Dino.txt")) {

   //Create the file
   FileStream oFS = File.Create(@"C:\TEMP\Flintstones\Dino.txt");
   oFS.Close();

   //Create an array of Strings with data to add to the file
   String[] sMyStmt = {"Now is", "the time", "for all good",
                       "men to come", "to the aid", "of their country!"};

   //Write sMyStmt to the file
   File.WriteAllLines(@"C:\TEMP\Flintstones\Dino.txt",sMyStmt);

  }


 }

}
```

The file Dino.txt looks like this:

```
Now is
the time
for all good
men to come
to the aid
of their country!
```

The `System.IO` Namespace

- → Classes
  - → `Path`

Path

The `Path` class provides performs operations on `String` instances that contain file or directory path information. These operations are performed in a cross-platform manner. All of the methods and fields below are static!

Methods

- ChangeExtension - Changes the extension of a path string
- Combine(String[]) - Combines an array of strings into a path
- Combine(String, String) - Combines two strings into a path
- Combine(String, String, String) - Combines three strings into a path
- Combine(String, String, String, String) - Combines four strings into a path
- GetDirectoryName - Returns the directory information for the specified path string
- GetExtension - Returns the extension of the specified path string
- GetFileName - Returns the file name and extension of the specified path string
- GetFileNameWithoutExtension - Returns the file name of the specified path string without the extension
- GetFullPath - Returns the absolute path for the specified path string
- GetInvalidFileNameChars - Gets an array containing the characters that are not allowed in file names
- GetInvalidPathChars - Gets an array containing the characters that are not allowed in path names
- GetPathRoot - Gets the root directory information of the specified path
- GetRandomFileName - Returns a random folder name or file name
- GetTempFileName - Creates a uniquely named, zero-byte temporary file on disk and returns the full path of that file
- GetTempPath - Returns the path of the current user's temporary folder
- HasExtension - Determines whether a path includes a file name extension
- IsPathRooted - Gets a value indicating whether the specified path string contains a root

Fields

- AltDirectorySeparatorChar - Provides a platform-specific alternate character used to separate directory levels in a path string that reflects a hierarchical file system organization
- DirectorySeparatorChar - Provides a platform-specific character used to separate directory levels in a path string that reflects a hierarchical file system organization
- InvalidPathChars Obsolete. Provides a platform-specific array of characters that cannot be specified in path string arguments passed to members of the Path class.
- PathSeparator - A platform-specific separator character used to separate path strings in environment variables
- VolumeSeparatorChar - Provides a platform-specific volume separator character

Path

Below is an example. Take note that you can easily generate a new temporary file with the `GetTempFileName` static method!!

```
using System;
using System.Collections.Generic;
using System.IO;

class MainProgram {

 public static void Main() {

  String sPath = @"C:\TEMP\Flintstones\Dino.txt";
  Console.WriteLine("GetFullPath={0}",Path.GetFullPath(sPath));
  Console.WriteLine("GetFileName={0}",Path.GetFileName(sPath));
  Console.WriteLine("GetFileNameWithoutExtension={0}",Path.GetFileNameWithoutExtension(sPath));
  Console.WriteLine("GetExtension={0}",Path.GetExtension(sPath));
  Console.WriteLine("GetRandomFileName={0}",Path.GetRandomFileName());
  Console.WriteLine("GetTempFileName={0}",Path.GetTempFileName());

 }

}
```

The output is:

```
GetFullPath=C:\TEMP\Flintstones\Dino.txt
GetFileName=Dino.txt
GetFileNameWithoutExtension=Dino
GetExtension=.txt
GetRandomFileName=c0qepqay.14e
GetTempFileName=C:\Users\Scott\AppData\Local\Temp\tmpA547.tmp
```

The `System.IO` Namespace

- → Classes
  - → `Stream`

Stream

The `Steam` class provides a generic view of a sequence of bytes. `Stream` is the abstract base class of all streams.

Constructors

- Stream - Initializes a new instance of the Stream class.

Properties

- CanRead - When overridden in a derived class, gets a value indicating whether the current stream supports reading
- CanSeek - When overridden in a derived class, gets a value indicating whether the current stream supports seeking
- CanTimeout - Gets a value that determines whether the current stream can time out
- CanWrite - When overridden in a derived class, gets a value indicating whether the current stream supports writing
- Length - When overridden in a derived class, gets the length in bytes of the stream
- Position - When overridden in a derived class, gets or sets the position within the current stream
- ReadTimeout - Gets or sets a value, in miliseconds, that determines how long the stream will attempt to read before timing out
- WriteTimeout - Gets or sets a value, in miliseconds, that determines how long the stream will attempt to write before timing out

Methods

- BeginRead - Begins an asynchronous read operation
- BeginWrite - Begins an asynchronous write operation
- Close - Closes the current stream and releases any resources (such as sockets and file handles) associated with the current stream
- CopyTo(Stream) - Reads the bytes from the current stream and writes them to the destination stream
- CopyTo(Stream, Int32) - Reads all the bytes from the current stream and writes them to a destination stream, using a specified buffer size
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- CreateWaitHandle Obsolete. Allocates a WaitHandle object.
- Dispose() - Releases all resources used by the Stream
- Dispose(Boolean) - Releases the unmanaged resources used by the Stream and optionally releases the managed resources

Methods (continued)

- EndRead - Waits for the pending asynchronous read to complete
- EndWrite - Ends an asynchronous write operation
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- Flush - When overridden in a derived class, clears all buffers for this stream and causes any buffered data to be written to the underlying device
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- ObjectInvariant - Infrastructure. Provides support for a Contract.
- Read - When overridden in a derived class, reads a sequence of bytes from the current stream and advances the position within the stream by the number of bytes read
- ReadByte - Reads a byte from the stream and advances the position within the stream by one byte, or returns -1 if at the end of the stream
- Seek - When overridden in a derived class, sets the position within the current stream
- SetLength - When overridden in a derived class, sets the length of the current stream
- Synchronized - Creates a thread-safe (synchronized) wrapper around the specified Stream object
- ToString - Returns a string that represents the current object. (Inherited from Object.)
- Write - When overridden in a derived class, writes a sequence of bytes to the current stream and advances the current position within this stream by the number of bytes written
- WriteByte - Writes a byte to the current position in the stream and advances the position within the stream by one byte

Fields

- Null - A Stream with no backing store

The `System.IO` Namespace

- → Classes
  - → `TextReader`

TextReader

The `TextReader` class represents a reader that can read a sequential series of characters. According to Microsoft's website: *TextReader is the abstract base class of StreamReader and StringReader, which read characters from streams and strings, respectively. Use these derived classes to open a text file for reading a specified range of characters, or to create a reader based on an existing stream.*

Constructors

- TextReader - Initializes a new instance of the TextReader class.

Methods

- Close - Closes the TextReader and releases any system resources associated with the TextReader.
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- Dispose() - Releases all resources used by the TextReader object.
- Dispose(Boolean) - Releases the unmanaged resources used by the TextReader and optionally releases the managed resources.
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- Peek - Reads the next character without changing the state of the reader or the character source. Returns the next available character without actually reading it from the input stream.

Methods (continued)

- Read() - Reads the next character from the input stream and advances the character position by one character.
- Read(Char[], Int32, Int32) - Reads a maximum of count characters from the current stream and writes the data to buffer, beginning at index.
- ReadBlock - Reads a maximum of count characters from the current stream, and writes the data to buffer, beginning at index.
- ReadLine - Reads a line of characters from the current stream and returns the data as a string.
- ReadToEnd - Reads all characters from the current position to the end of the TextReader and returns them as one string.
- Synchronized - Creates a thread-safe wrapper around the specified TextReader.
- ToString - Returns a string that represents the current object. (Inherited from Object.)

Fields

- Null - Provides a TextReader with no data to read from.

The `System.IO` Namespace

- → Classes
  - → `TextWriter`

TextWriter

The `TextWriter` class represents a writer that can write a sequential series of characters. This class is abstract.

Constructors

- TextWriter() - Initializes a new instance of the TextWriter class
- TextWriter(IFormatProvider) - Initializes a new instance of the TextWriter class with the specified format provider

Properties

- Encoding - When overridden in a derived class, returns the Encoding in which the output is written
- FormatProvider - Gets an object that controls formatting
- NewLine - Gets or sets the line terminator string used by the current TextWriter

Methods

- Close - Closes the current writer and releases any system resources associated with the writer
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- Dispose() - Releases all resources used by the TextWriter object
- Dispose(Boolean) - Releases the unmanaged resources used by the TextWriter and optionally releases the managed resources
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- Flush - Clears all buffers for the current writer and causes any buffered data to be written to the underlying device
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- Synchronized - Creates a thread-safe wrapper around the specified TextWriter
- ToString - Returns a string that represents the current object. (Inherited from Object.)

Methods (continued)

- Write(Boolean) - Writes the text representation of a Boolean value to the text stream
- Write(Char) - Writes a character to the text stream
- Write(Char[]) - Writes a character array to the text stream
- Write(Decimal) - Writes the text representation of a decimal value to the text stream
- Write(Double) - Writes the text representation of an 8-byte floating-point value to the text stream
- Write(Int32) - Writes the text representation of a 4-byte signed integer to the text stream
- Write(Int64) - Writes the text representation of an 8-byte signed integer to the text stream
- Write(Object) - Writes the text representation of an object to the text stream by calling ToString on that object
- Write(Single) - Writes the text representation of a 4-byte floating-point value to the text stream
- Write(String) - Writes a string to the text stream
- Write(UInt32) - Writes the text representation of a 4-byte unsigned integer to the text stream
- Write(UInt64) - Writes the text representation of an 8-byte unsigned integer to the text stream
- Write(String, Object) - Writes out a formatted string, using the same semantics as String.Format
- Write(String, Object[]) - Writes out a formatted string, using the same semantics as String.Format
- Write(Char[], Int32, Int32) - Writes a subarray of characters to the text stream
- Write(String, Object, Object) - Writes out a formatted string, using the same semantics as String.Format
- Write(String, Object, Object, Object) - Writes out a formatted string, using the same semantics as String.Format
- WriteLine() - Writes a line terminator to the text stream
- WriteLine(Boolean) - Writes the text representation of a Boolean followed by a line terminator to the text stream
- WriteLine(Char) - Writes a character followed by a line terminator to the text stream
- WriteLine(Char[]) - Writes an array of characters followed by a line terminator to the text stream
- WriteLine(Decimal) - Writes the text representation of a decimal value followed by a line terminator to the text stream
- WriteLine(Double) - Writes the text representation of a 8-byte floating-point value followed by a line terminator to the text stream
- WriteLine(Int32) - Writes the text representation of a 4-byte signed integer followed by a line terminator to the text stream
- WriteLine(Int64) - Writes the text representation of an 8-byte signed integer followed by a line terminator to the text stream
- WriteLine(Object) - Writes the text representation of an object by calling ToString on this object, followed by a line terminator to the text stream
- WriteLine(Single) - Writes the text representation of a 4-byte floating-point value followed by a line terminator to the text stream
- WriteLine(String) - Writes a string followed by a line terminator to the text stream

Methods (continued)

- WriteLine(UInt32) - Writes the text representation of a 4-byte unsigned integer followed by a line terminator to the text stream
- WriteLine(UInt64) - Writes the text representation of an 8-byte unsigned integer followed by a line terminator to the text stream
- WriteLine(String, Object) - Writes out a formatted string and a new line, using the same semantics as Format
- WriteLine(String, Object[]) - Writes out a formatted string and a new line, using the same semantics as Format
- WriteLine(Char[], Int32, Int32) - Writes a subarray of characters followed by a line terminator to the text stream
- WriteLine(String, Object, Object) - Writes out a formatted string and a new line, using the same semantics as Format
- WriteLine(String, Object, Object, Object) - Writes out a formatted string and a new line, using the same semantics as Format

Fields

- CoreNewLine - Stores the new line characters used for this TextWriter
- Null - Provides a TextWriter with no backing store that can be written to, but not read from

The `System.IO` Namespace

- → Classes
  - → `BinaryReader`

BinaryReader

The `BinaryReader` class reads primitive data types as binary values in a specific encoding.

Constructors

- BinaryReader(Stream) - Initializes a new instance of the BinaryReader class based on the supplied stream and using UTF8Encoding
- BinaryReader(Stream, Encoding) - Initializes a new instance of the BinaryReader class based on the supplied stream and a specific character encoding

Properties

- BaseStream - Exposes access to the underlying stream of the BinaryReader

Methods

- Close - Closes the current reader and the underlying stream
- Dispose() - Releases all resources used by the current instance of the BinaryReader class
- Dispose(Boolean) - Releases the unmanaged resources used by the BinaryReader class and optionally releases the managed resources
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- FillBuffer - Fills the internal buffer with the specified number of bytes read from the stream
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- MemberwiseClone - Creates a shallow copy of the current Object. (Inherited from Object.)
- PeekChar - Returns the next available character and does not advance the byte or character position
- Read() - Reads characters from the underlying stream and advances the current position of the stream in accordance with the Encoding used and the specific character being read from the stream
- Read(Byte[], Int32, Int32) - Reads the specified number of bytes from the stream, starting from a specified point in the byte array
- Read(Char[], Int32, Int32) - Reads the specified number of characters from the stream, starting from a specified point in the character array
- Read7BitEncodedInt - Reads in a 32-bit integer in compressed format

Methods (continued)

- ReadBoolean - Reads a Boolean value from the current stream and advances the current position of the stream by one byte
- ReadByte - Reads the next byte from the current stream and advances the current position of the stream by one byte
- ReadBytes - Reads the specified number of bytes from the current stream into a byte array and advances the current position by that number of bytes
- ReadChar - Reads the next character from the current stream and advances the current position of the stream in accordance with the Encoding used and the specific character being read from the stream
- ReadChars - Reads the specified number of characters from the current stream, returns the data in a character array, and advances the current position in accordance with the Encoding used and the specific character being read from the stream
- ReadDecimal - Reads a decimal value from the current stream and advances the current position of the stream by sixteen bytes
- ReadDouble - Reads an 8-byte floating point value from the current stream and advances the current position of the stream by eight bytes
- ReadInt16 - Reads a 2-byte signed integer from the current stream and advances the current position of the stream by two bytes
- ReadInt32 - Reads a 4-byte signed integer from the current stream and advances the current position of the stream by four bytes
- ReadInt64 - Reads an 8-byte signed integer from the current stream and advances the current position of the stream by eight bytes
- ReadSByte - Reads a signed byte from this stream and advances the current position of the stream by one byte
- ReadSingle - Reads a 4-byte floating point value from the current stream and advances the current position of the stream by four bytes
- ReadString - Reads a string from the current stream. The string is prefixed with the length, encoded as an integer seven bits at a time
- ReadUInt16 - Reads a 2-byte unsigned integer from the current stream using little-endian encoding and advances the position of the stream by two bytes
- ReadUInt32 - Reads a 4-byte unsigned integer from the current stream and advances the position of the stream by four bytes
- ReadUInt64 - Reads an 8-byte unsigned integer from the current stream and advances the position of the stream by eight bytes
- ToString - Returns a string that represents the current object. (Inherited from Object.)

The `System.IO` Namespace

- → Classes
  - → `BinaryWriter`

BinaryWriter

The `BinaryWriter` class writes primitive types in binary to a stream and supports writing strings in a specific encoding.

Constructors

- BinaryWriter() - Initializes a new instance of the BinaryWriter class that writes to a stream
- BinaryWriter(Stream) - Initializes a new instance of the BinaryWriter class based on the supplied stream and using UTF-8 as the encoding for strings
- BinaryWriter(Stream, Encoding) - Initializes a new instance of the BinaryWriter class based on the supplied stream and a specific character encoding

Properties

- BaseStream - Exposes access to the underlying stream of the BinaryWriter

Methods

- Close - Closes the current BinaryWriter and the underlying stream
- Dispose() - Releases all resources used by the current instance of the BinaryWriter class
- Dispose(Boolean) - Releases the unmanaged resources used by the BinaryWriter and optionally releases the managed resources
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- Flush - Clears all buffers for the current writer and causes any buffered data to be written to the underlying device
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- MemberwiseClone - Creates a shallow copy of the current Object. (Inherited from Object.)
- Seek - Sets the position within the current stream
- ToString - Returns a string that represents the current object. (Inherited from Object.)
- Write(Boolean) - Writes a one-byte Boolean value to the current stream, with 0 representing false and 1 representing true
- Write(Byte) - Writes an unsigned byte to the current stream and advances the stream position by one byte
- Write(Byte[]) - Writes a byte array to the underlying stream

Methods (continued)

- Write(Char) - Writes a Unicode character to the current stream and advances the current position of the stream in accordance with the Encoding used and the specific characters being written to the stream
- Write(Char[]) - Writes a character array to the current stream and advances the current position of the stream in accordance with the Encoding used and the specific characters being written to the stream
- Write(Decimal) - Writes a decimal value to the current stream and advances the stream position by sixteen bytes
- Write(Double) - Writes an eight-byte floating-point value to the current stream and advances the stream position by eight bytes
- Write(Int16) - Writes a two-byte signed integer to the current stream and advances the stream position by two bytes
- Write(Int32) - Writes a four-byte signed integer to the current stream and advances the stream position by four bytes
- Write(Int64) - Writes an eight-byte signed integer to the current stream and advances the stream position by eight bytes
- Write(SByte) - Writes a signed byte to the current stream and advances the stream position by one byte
- Write(Single) - Writes a four-byte floating-point value to the current stream and advances the stream position by four bytes
- Write(String) - Writes a length-prefixed string to this stream in the current encoding of the BinaryWriter, and advances the current position of the stream in accordance with the encoding used and the specific characters being written to the stream
- Write(UInt16) - Writes a two-byte unsigned integer to the current stream and advances the stream position by two bytes
- Write(UInt32) - Writes a four-byte unsigned integer to the current stream and advances the stream position by four bytes
- Write(UInt64) - Writes an eight-byte unsigned integer to the current stream and advances the stream position by eight bytes
- Write(Byte[], Int32, Int32) - Writes a region of a byte array to the current stream
- Write(Char[], Int32, Int32) - Writes a section of a character array to the current stream, and advances the current position of the stream in accordance with the Encoding used and perhaps the specific characters being written to the stream
- Write7BitEncodedInt - Writes a 32-bit integer in a compressed format

Fields

- Null - Specifies a BinaryWriter with no backing store
- OutStream - Holds the underlying stream

The `System.IO` Namespace

- → Classes
  - → `StreamReader`

StreamReader

The `StreamReader` class implements a `TextReader` that reads characters from a byte stream in a particular encoding. According to Microsoft's website: *StreamReader is designed for character input in a particular encoding, whereas the Stream class is designed for byte input and output. Use StreamReader for reading lines of information from a standard text file.*

Constructors

- StreamReader(Stream) - Initializes a new instance of the StreamReader class for the specified stream
- StreamReader(String) - Initializes a new instance of the StreamReader class for the specified file name
- StreamReader(Stream, Boolean) - Initializes a new instance of the StreamReader class for the specified stream, with the specified byte order mark detection option
- StreamReader(Stream, Encoding) - Initializes a new instance of the StreamReader class for the specified stream, with the specified character encoding
- StreamReader(String, Boolean) - Initializes a new instance of the StreamReader class for the specified file name, with the specified byte order mark detection option
- StreamReader(String, Encoding) - Initializes a new instance of the StreamReader class for the specified file name, with the specified character encoding
- StreamReader(Stream, Encoding, Boolean) - Initializes a new instance of the StreamReader class for the specified stream, with the specified character encoding and byte order mark detection option
- StreamReader(String, Encoding, Boolean) - Initializes a new instance of the StreamReader class for the specified file name, with the specified character encoding and byte order mark detection option
- StreamReader(Stream, Encoding, Boolean, Int32) - Initializes a new instance of the StreamReader class for the specified stream, with the specified character encoding, byte order mark detection option, and buffer size
- StreamReader(String, Encoding, Boolean, Int32) - Initializes a new instance of the StreamReader class for the specified file name, with the specified character encoding, byte order mark detection option, and buffer size

Properties

- BaseStream - Returns the underlying stream
- CurrentEncoding - Gets the current character encoding that the current StreamReader object is using
- EndOfStream - Gets a value that indicates whether the current stream position is at the end of the stream

StreamReader

Methods

- Close - Closes the StreamReader object and the underlying stream, and releases any system resources associated with the reader. (Overrides TextReader.Close().)
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- DiscardBufferedData - Clears the internal buffer
- Dispose() - Releases all resources used by the TextReader object. (Inherited from TextReader.)
- Dispose(Boolean) - Closes the underlying stream, releases the unmanaged resources used by the StreamReader, and optionally releases the managed resources. (Overrides TextReader.Dispose(Boolean).)
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- Peek - Returns the next available character but does not consume it. (Overrides TextReader.Peek().)
- Read() - Reads the next character from the input stream and advances the character position by one character. (Overrides TextReader.Read().)
- Read(Char[], Int32, Int32) - Reads a specified maximum of characters from the current stream into a buffer, beginning at the specified index. (Overrides TextReader.Read(Char[], Int32, Int32).)
- ReadBlock - Reads a maximum of count characters from the current stream, and writes the data to buffer, beginning at index. (Inherited from TextReader.)
- ReadLine - Reads a line of characters from the current stream and returns the data as a string. (Overrides TextReader.ReadLine().)
- ReadToEnd - Reads the stream from the current position to the end of the stream. (Overrides TextReader.ReadToEnd().)
- ToString - Returns a string that represents the current object. (Inherited from Object.)

Fields

- Null - A StreamReader object around an empty stream

StreamReader

The example below reads in the data from Dino.txt and prints it out.

```
using System;
using System.Collections.Generic;
using System.IO;

class MainProgram {

 public static void Main() {

  String sFile = @"C:\TEMP\Flintstones\Dino.txt";
  String sALine;
  StreamReader oSR = new StreamReader(sFile);
  while(!oSR.EndOfStream) {
   sALine = oSR.ReadLine();
   Console.WriteLine(sALine);
  }

 }

}
```

The output looks like this:

```
Now is
the time
for all good
men to come
to the aid
of their country!
```

The `System.IO` Namespace

- → Classes
  - → `StreamWriter`

StreamWriter

The `StreamWriter` class implements a `TextWriter` for writing characters to a stream in a particular encoding. According to Microsoft's website: *StreamWriter is designed for character output in a particular encoding, whereas classes derived from Stream are designed for byte input and output.*

Constructors

- StreamWriter(Stream) - Initializes a new instance of the StreamWriter class for the specified stream, using UTF-8 encoding and the default buffer size
- StreamWriter(String) - Initializes a new instance of the StreamWriter class for the specified file on the specified path, using the default encoding and buffer size
- StreamWriter(Stream, Encoding) - Initializes a new instance of the StreamWriter class for the specified stream, using the specified encoding and the default buffer size
- StreamWriter(String, Boolean) - Initializes a new instance of the StreamWriter class for the specified file on the specified path, using the default encoding and buffer size. If the file exists, it can be either overwritten or appended to. If the file does not exist, this constructor creates a new file
- StreamWriter(Stream, Encoding, Int32) - Initializes a new instance of the StreamWriter class for the specified stream, using the specified encoding and buffer size
- StreamWriter(String, Boolean, Encoding) - Initializes a new instance of the StreamWriter class for the specified file on the specified path, using the specified encoding and default buffer size. If the file exists, it can be either overwritten or appended to. If the file does not exist, this constructor creates a new file
- StreamWriter(String, Boolean, Encoding, Int32) - Initializes a new instance of the StreamWriter class for the specified file on the specified path, using the specified encoding and buffer size. If the file exists, it can be either overwritten or appended to. If the file does not exist, this constructor creates a new file

Properties

- AutoFlush - Gets or sets a value indicating whether the StreamWriter will flush its buffer to the underlying stream after every call to StreamWriter.Write
- BaseStream - Gets the underlying stream that interfaces with a backing store
- Encoding Gets the Encoding in which the output is written. (Overrides TextWriter.Encoding.)
- FormatProvider Gets an object that controls formatting. (Inherited from TextWriter.)
- NewLine Gets or sets the line terminator string used by the current TextWriter. (Inherited from TextWriter.)

StreamWriter

Methods

- Close - Closes the current StreamWriter object and the underlying stream. (Overrides TextWriter.Close().)
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- Dispose() - Releases all resources used by the TextWriter object. (Inherited from TextWriter.)
- Dispose(Boolean) - Releases the unmanaged resources used by the StreamWriter and optionally releases the managed resources. (Overrides TextWriter.Dispose(Boolean).)
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- Flush - Clears all buffers for the current writer and causes any buffered data to be written to the underlying stream. (Overrides TextWriter.Flush().)
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- ToString - Returns a string that represents the current object. (Inherited from Object.)
- Write(Boolean) - Writes the text representation of a Boolean value to the text stream. (Inherited from TextWriter.)
- Write(Char) - Writes a character to the stream. (Overrides TextWriter.Write(Char).)
- Write(Char[]) - Writes a character array to the stream. (Overrides TextWriter.Write(Char[]).)
- Write(Decimal) - Writes the text representation of a decimal value to the text stream. (Inherited from TextWriter.)
- Write(Double) - Writes the text representation of an 8-byte floating-point value to the text stream. (Inherited from TextWriter.)
- Write(Int32) - Writes the text representation of a 4-byte signed integer to the text stream. (Inherited from TextWriter.)
- Write(Int64) - Writes the text representation of an 8-byte signed integer to the text stream. (Inherited from TextWriter.)
- Write(Object) - Writes the text representation of an object to the text stream by calling ToString on that object. (Inherited from TextWriter.)
- Write(Single) - Writes the text representation of a 4-byte floating-point value to the text stream. (Inherited from TextWriter.)
- Write(String) - Writes a string to the stream. (Overrides TextWriter.Write(String).)
- Write(UInt32) - Writes the text representation of a 4-byte unsigned integer to the text stream. (Inherited from TextWriter.)

Methods(continued)

- Write(UInt64) - Writes the text representation of an 8-byte unsigned integer to the text stream. (Inherited from TextWriter.)
- Write(String, Object) - Writes out a formatted string, using the same semantics as String.Format. (Inherited from TextWriter.)
- Write(String, Object[]) - Writes out a formatted string, using the same semantics as String.Format. (Inherited from TextWriter.)
- Write(Char[], Int32, Int32) - Writes a subarray of characters to the stream. (Overrides TextWriter.Write(Char[], Int32, Int32).)
- Write(String, Object, Object) - Writes out a formatted string, using the same semantics as String.Format. (Inherited from TextWriter.)
- Write(String, Object, Object, Object) - Writes out a formatted string, using the same semantics as String.Format. (Inherited from TextWriter.)
- WriteLine() - Writes a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Boolean) - Writes the text representation of a Boolean followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Char) - Writes a character followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Char[]) - Writes an array of characters followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Decimal) - Writes the text representation of a decimal value followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Double) - Writes the text representation of a 8-byte floating-point value followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Int32) - Writes the text representation of a 4-byte signed integer followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Int64) - Writes the text representation of an 8-byte signed integer followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Object) - Writes the text representation of an object by calling ToString on this object, followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Single) - Writes the text representation of a 4-byte floating-point value followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(String) - Writes a string followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(UInt32) - Writes the text representation of a 4-byte unsigned integer followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(UInt64) - Writes the text representation of an 8-byte unsigned integer followed by a line terminator to the text stream. (Inherited from TextWriter.)

Methods(continued)

- WriteLine(String) - Writes a string followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(UInt32) - Writes the text representation of a 4-byte unsigned integer followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(UInt64) - Writes the text representation of an 8-byte unsigned integer followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(String, Object) - Writes out a formatted string and a new line, using the same semantics as Format. (Inherited from TextWriter.)
- WriteLine(String, Object[]) - Writes out a formatted string and a new line, using the same semantics as Format. (Inherited from TextWriter.)
- WriteLine(Char[], Int32, Int32) - Writes a subarray of characters followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(String, Object, Object) - Writes out a formatted string and a new line, using the same semantics as Format. (Inherited from TextWriter.)
- WriteLine(String, Object, Object, Object) - Writes out a formatted string and a new line

Fields

- CoreNewLine - Stores the new line characters used for this TextWriter. (Inherited from TextWriter.)
- Null - Provides a StreamWriter with no backing store that can be written to, but not read from

The example below opens up Dino.txt and appends a few additional lines of text to it. Note that if you only use a StreamWriter along with a filename, then the text in the file will be overwritten. To avoid this, create a FileStream object and set its file properties to allow for appending of data. See next slide.

StreamWriter


```
using System;
using System.Collections.Generic;
using System.IO;

class MainProgram {

 public static void Main() {

  String sFile = @"C:\TEMP\Flintstones\Dino.txt";
  String sAdditionalText = "...and that's a fact, Jack!";

  //Open up the file Dino.txt for append.
  FileStream oFS = new FileStream(sFile,FileMode.Append,FileAccess.Write,FileShare.None);

  //Write additional text to the Dino.txt file.
  StreamWriter oSW = new StreamWriter(oFS);
  oSW.Write(sAdditionalText);
  oSW.Flush();
  oSW.Close();
  oFS.Close();

  //Now, read that data and print it out.
  String sALine;
  StreamReader oSR = new StreamReader(sFile);
  while(!oSR.EndOfStream) {
   sALine = oSR.ReadLine();
   Console.WriteLine(sALine);
  }

 }

}
```

The `System.IO` Namespace

- → Classes
  - → `StringReader`

StringReader

The `StringReader` class implements a `TextReader` that reads from a string.

Constructors

- StringReader - Initializes a new instance of the StringReader class that reads from the specified string.

Methods

- Close - Closes the StringReader. (Overrides TextReader.Close().)
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- Dispose() - Releases all resources used by the TextReader object. (Inherited from TextReader.)
- Dispose(Boolean) - Releases the unmanaged resources used by the StringReader and optionally releases the managed resources. (Overrides TextReader.Dispose(Boolean).)
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- Peek - Returns the next available character but does not consume it. (Overrides TextReader.Peek().)
- Read() - Reads the next character from the input string and advances the character position by one character. (Overrides TextReader.Read().)
- Read(Char[], Int32, Int32) - Reads a block of characters from the input string and advances the character position by count. (Overrides TextReader.Read(Char[], Int32, Int32).)
- ReadBlock - Reads a maximum of count characters from the current stream, and writes the data to buffer, beginning at index. (Inherited from TextReader.)
- ReadLine - Reads a line from the underlying string. (Overrides TextReader.ReadLine().)
- ReadToEnd - Reads the stream as a string, either in its entirety or from the current position to the end of the stream. (Overrides TextReader.ReadToEnd().)
- ToString - Returns a string that represents the current object. (Inherited from Object.)

The `System.IO` Namespace

- → Classes
  - → `StringWriter`

StringWriter

The `StringWriter` class implements a `TextWriter` for writing information to a string. The information is stored in an underlying `StringBuilder`.

Constructors

- StringWriter() - Initializes a new instance of the StringWriter class
- StringWriter(IFormatProvider) - Initializes a new instance of the StringWriter class with the specified format control
- StringWriter(StringBuilder) - Initializes a new instance of the StringWriter class that writes to the specified StringBuilder
- StringWriter(StringBuilder, IFormatProvider) - Initializes a new instance of the StringWriter class that writes to the specified StringBuilder and has the specified format provider

Properties

- Encoding - Gets the Encoding in which the output is written. (Overrides TextWriter.Encoding.)
- FormatProvider - Gets an object that controls formatting. (Inherited from TextWriter.)
- NewLine - Gets or sets the line terminator string used by the current TextWriter. (Inherited from TextWriter.)

Methods

- Close - Closes the current StringWriter and the underlying stream. (Overrides TextWriter.Close().)
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- Dispose() - Releases all resources used by the TextWriter object. (Inherited from TextWriter.)
- Dispose(Boolean) - Releases the unmanaged resources used by the StringWriter and optionally releases the managed resources. (Overrides TextWriter.Dispose(Boolean).)
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- Flush - Clears all buffers for the current writer and causes any buffered data to be written to the underlying device. (Inherited from TextWriter.)
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetStringBuilder - Returns the underlying StringBuilder
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)

Methods (continued)

- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- ToString - Returns a string containing the characters written to the current StringWriter so far. (Overrides Object.ToString().)
- Write(Boolean) - Writes the text representation of a Boolean value to the text stream. (Inherited from TextWriter.)
- Write(Char) - Writes a character to this instance of the StringWriter. (Overrides TextWriter.Write(Char).)
- Write(Char[]) - Writes a character array to the text stream. (Inherited from TextWriter.)
- Write(Decimal) - Writes the text representation of a decimal value to the text stream. (Inherited from TextWriter.)
- Write(Double) - Writes the text representation of an 8-byte floating-point value to the text stream. (Inherited from TextWriter.)
- Write(Int32) - Writes the text representation of a 4-byte signed integer to the text stream. (Inherited from TextWriter.)
- Write(Int64) - Writes the text representation of an 8-byte signed integer to the text stream. (Inherited from TextWriter.)
- Write(Object) - Writes the text representation of an object to the text stream by calling ToString on that object. (Inherited from TextWriter.)
- Write(Single) - Writes the text representation of a 4-byte floating-point value to the text stream. (Inherited from TextWriter.)
- Write(String) - Writes a string to this instance of the StringWriter. (Overrides TextWriter.Write(String).)
- Write(UInt32) - Writes the text representation of a 4-byte unsigned integer to the text stream. (Inherited from TextWriter.)
- Write(UInt64) - Writes the text representation of an 8-byte unsigned integer to the text stream. (Inherited from TextWriter.)
- Write(String, Object) - Writes out a formatted string, using the same semantics as String.Format. (Inherited from TextWriter.)
- Write(String, Object[]) - Writes out a formatted string, using the same semantics as String.Format. (Inherited from TextWriter.)
- Write(Char[], Int32, Int32) - Writes the specified region of a character array to this instance of the StringWriter. (Overrides TextWriter.Write(Char[], Int32, Int32).)
- Write(String, Object, Object) - Writes out a formatted string, using the same semantics as String.Format. (Inherited from TextWriter.)
- Write(String, Object, Object, Object) - Writes out a formatted string, using the same semantics as String.Format. (Inherited from TextWriter.)
- WriteLine() - Writes a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Boolean) - Writes the text representation of a Boolean followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Char) - Writes a character followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Char[]) - Writes an array of characters followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Decimal) - Writes the text representation of a decimal value followed by a line terminator to the text stream. (Inherited from TextWriter.)

Methods (continued)

- WriteLine(Double) - Writes the text representation of a 8-byte floating-point value followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Int32) - Writes the text representation of a 4-byte signed integer followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Int64) - Writes the text representation of an 8-byte signed integer followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Object) - Writes the text representation of an object by calling ToString on this object, followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(Single) - Writes the text representation of a 4-byte floating-point value followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(String) - Writes a string followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(UInt32) - Writes the text representation of a 4-byte unsigned integer followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(UInt64) - Writes the text representation of an 8-byte unsigned integer followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(String, Object) - Writes out a formatted string and a new line, using the same semantics as Format. (Inherited from TextWriter.)
- WriteLine(String, Object[]) - Writes out a formatted string and a new line, using the same semantics as Format. (Inherited from TextWriter.)
- WriteLine(Char[], Int32, Int32) - Writes a subarray of characters followed by a line terminator to the text stream. (Inherited from TextWriter.)
- WriteLine(String, Object, Object) - Writes out a formatted string and a new line, using the same semantics as Format. (Inherited from TextWriter.)
- WriteLine(String, Object, Object, Object) - Writes out a formatted

Fields

- CoreNewLine - Stores the new line characters used for this TextWriter. (Inherited from TextWriter.)

The `System.IO` Namespace

- → Classes
  - → `FileStream`

FileStream

The `FileStream` class exposes a `Stream` around a file, supporting both synchronous and asynchronous read and write operations. According to Microsoft's website: *Use the FileStream class to read from, write to, open, and close files on a file system, as well as to manipulate other file-related operating system handles including pipes, standard input, and standard output. You can specify read and write operations to be either synchronous or asynchronous. FileStream buffers input and output for better performance. FileStream objects support random access to files using the Seek method. Seek allows the read/write position to be moved to any position within the file. This is done with byte offset reference point parameters. The byte offset is relative to the seek reference point, which can be the beginning, the current position, or the end of the underlying file, as represented by the three properties of the SeekOrigin class.* Be aware the if you are using BeginRead and BeginWrite to perform asynchronous operations, don't forget to set the asynchronous property in the constructor; otherwise, your operations will still be synchronous!!!!

Constructors

- FileStream(SafeFileHandle, FileAccess) - Initializes a new instance of the FileStream class for the specified file handle, with the specified read/write permission
- FileStream(String, FileMode) - Initializes a new instance of the FileStream class with the specified path and creation mode
- FileStream(SafeFileHandle, FileAccess, Int32) - Initializes a new instance of the FileStream class for the specified file handle, with the specified read/write permission, and buffer size
- FileStream(String, FileMode, FileAccess) - Initializes a new instance of the FileStream class with the specified path, creation mode, and read/write permission
- FileStream(SafeFileHandle, FileAccess, Int32, Boolean) - Initializes a new instance of the FileStream class for the specified file handle, with the specified read/write permission, buffer size, and synchronous or asynchronous state
- FileStream(String, FileMode, FileAccess, FileShare) - Initializes a new instance of the FileStream class with the specified path, creation mode, read/write permission, and sharing permission

Constructors (continued)

- FileStream(String, FileMode, FileAccess, FileShare, Int32) - Initializes a new instance of the FileStream class with the specified path, creation mode, read/write and sharing permission, and buffer size
- FileStream(String, FileMode, FileAccess, FileShare, Int32, Boolean) - Initializes a new instance of the FileStream class with the specified path, creation mode, read/write and sharing permission, buffer size, and synchronous or asynchronous state
- FileStream(String, FileMode, FileAccess, FileShare, Int32, FileOptions) - Initializes a new instance of the FileStream class with the specified path, creation mode, read/write and sharing permission, the access other FileStreams can have to the same file, the buffer size, and additional file options
- FileStream(String, FileMode, FileSystemRights, FileShare, Int32, FileOptions) - Initializes a new instance of the FileStream class with the specified path, creation mode, access rights and sharing permission, the buffer size, and additional file options
- FileStream(String, FileMode, FileSystemRights, FileShare, Int32, FileOptions, FileSecurity) - Initializes a new instance of the FileStream class with the specified path, creation mode, access rights and sharing permission, the buffer size, additional file options, access control and audit security

Properties

- CanRead - Gets a value indicating whether the current stream supports reading. (Overrides Stream.CanRead.)
- CanSeek - Gets a value indicating whether the current stream supports seeking. (Overrides Stream.CanSeek.)
- CanTimeout - Gets a value that determines whether the current stream can time out. (Inherited from Stream.)
- CanWrite - Gets a value indicating whether the current stream supports writing. (Overrides Stream.CanWrite.)
- Handle - Obsolete. Gets the operating system file handle for the file that the current FileStream object encapsulates.
- IsAsync - Gets a value indicating whether the FileStream was opened asynchronously or synchronously
- Length - Gets the length in bytes of the stream. (Overrides Stream.Length.)
- Name - Gets the name of the FileStream that was passed to the constructor
- Position - Gets or sets the current position of this stream. (Overrides Stream.Position.)
- ReadTimeout - Gets or sets a value, in miliseconds, that determines how long the stream will attempt to read before timing out. (Inherited from Stream.)
- SafeFileHandle - Gets a SafeFileHandle object that represents the operating system file handle for the file that the current FileStream object encapsulates
- WriteTimeout - Gets or sets a value, in miliseconds, that determines how long the stream will attempt to write before timing out. (Inherited from Stream.)

FileStream

Methods

- BeginRead - Begins an asynchronous read. (Overrides Stream.BeginRead(Byte[], Int32, Int32, AsyncCallback, Object).)
- BeginWrite - Begins an asynchronous write. (Overrides Stream.BeginWrite(Byte[], Int32, Int32, AsyncCallback, Object).)
- Close - Closes the current stream and releases any resources (such as sockets and file handles) associated with the current stream. (Inherited from Stream.)
- CopyTo(Stream) - Reads the bytes from the current stream and writes them to the destination stream. (Inherited from Stream.)
- CopyTo(Stream, Int32) - Reads all the bytes from the current stream and writes them to a destination stream, using a specified buffer size. (Inherited from Stream.)
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- CreateWaitHandle - Obsolete. Allocates a WaitHandle object. (Inherited from Stream.)
- Dispose() - Releases all resources used by the Stream. (Inherited from Stream.)
- Dispose(Boolean) - Releases the unmanaged resources used by the FileStream and optionally releases the managed resources. (Overrides Stream.Dispose(Boolean).)
- EndRead - Waits for the pending asynchronous read to complete. (Overrides Stream.EndRead(IAsyncResult).)
- EndWrite - Ends an asynchronous write, blocking until the I/O operation has completed. (Overrides Stream.EndWrite(IAsyncResult).)
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Ensures that resources are freed and other cleanup operations are performed when the garbage collector reclaims the FileStream. (Overrides Object.Finalize().)
- Flush() - Clears buffers for this stream and causes any buffered data to be written to the file. (Overrides Stream.Flush().)
- Flush(Boolean) - Clears buffers for this stream and causes any buffered data to be written to the file, and also clears all intermediate file buffers
- GetAccessControl - Gets a FileSecurity object that encapsulates the access control list (ACL) entries for the file described by the current FileStream object
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- Lock - Prevents other processes from reading from or writing to the FileStream
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)

FileStream

Methods

- ObjectInvariant - Infrastructure. Provides support for a Contract. (Inherited from Stream.)
- Read - Reads a block of bytes from the stream and writes the data in a given buffer. (Overrides Stream.Read(Byte[], Int32, Int32).)
- ReadByte - Reads a byte from the file and advances the read position one byte. (Overrides Stream.ReadByte().)
- Seek - Sets the current position of this stream to the given value. (Overrides Stream.Seek(Int64, SeekOrigin).)
- SetAccessControl - Applies access control list (ACL) entries described by a FileSecurity object to the file described by the current FileStream object
- SetLength - Sets the length of this stream to the given value. (Overrides Stream.SetLength(Int64).)
- ToString - Returns a string that represents the current object. (Inherited from Object.)
- Unlock - Allows access by other processes to all or part of a file that was previously locked
- Write - Writes a block of bytes to this stream using data from a buffer. (Overrides Stream.Write(Byte[], Int32, Int32).)
- WriteByte - Writes a byte to the current position in the file stream. (Overrides Stream.WriteByte(Byte).)

The `System.IO` Namespace

- → Classes
  - → `MemoryStream`

MemoryStream

The `MemoryStream` class creates a stream whose backing store is memory.

Constructors

- MemoryStream() - Initializes a new instance of the MemoryStream class with an expandable capacity initialized to zero
- MemoryStream(Byte[]) - Initializes a new non-resizable instance of the MemoryStream class based on the specified byte array
- MemoryStream(Int32) - Initializes a new instance of the MemoryStream class with an expandable capacity initialized as specified
- MemoryStream(Byte[], Boolean) - Initializes a new non-resizable instance of the MemoryStream class based on the specified byte array with the CanWrite property set as specified
- MemoryStream(Byte[], Int32, Int32) - Initializes a new non-resizable instance of the MemoryStream class based on the specified region (index) of a byte array
- MemoryStream(Byte[], Int32, Int32, Boolean) - Initializes a new non-resizable instance of the MemoryStream class based on the specified region of a byte array, with the CanWrite property set as specified
- MemoryStream(Byte[], Int32, Int32, Boolean, Boolean) - Initializes a new instance of the MemoryStream class based on the specified region of a byte array, with the CanWrite property set as specified, and the ability to call GetBuffer set as specified

Properties

- CanRead - Gets a value indicating whether the current stream supports reading. (Overrides Stream.CanRead.)
- CanSeek - Gets a value indicating whether the current stream supports seeking. (Overrides Stream.CanSeek.)
- CanTimeout - Gets a value that determines whether the current stream can time out. (Inherited from Stream.)
- CanWrite - Gets a value indicating whether the current stream supports writing. (Overrides Stream.CanWrite.)
- Capacity - Gets or sets the number of bytes allocated for this stream
- Length - Gets the length of the stream in bytes. (Overrides Stream.Length.)
- Position - Gets or sets the current position within the stream. (Overrides Stream.Position.)
- ReadTimeout - Gets or sets a value, in miliseconds, that determines how long the stream will attempt to read before timing out. (Inherited from Stream.)
- WriteTimeout - Gets or sets a value, in miliseconds, that determines how long the stream will attempt to write before timing out. (Inherited from Stream.)

MemoryStream

sheepsqueezers.com

Methods

- BeginRead - Begins an asynchronous read operation. (Inherited from Stream.)
- BeginWrite - Begins an asynchronous write operation. (Inherited from Stream.)
- Close - Closes the current stream and releases any resources (such as sockets and file handles) associated with the current stream. (Inherited from Stream.)
- CopyTo(Stream) - Reads the bytes from the current stream and writes them to the destination stream. (Inherited from Stream.)
- CopyTo(Stream, Int32) - Reads all the bytes from the current stream and writes them to a destination stream, using a specified buffer size. (Inherited from Stream.)
- CreateObjRef - Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
- Dispose() - Releases all resources used by the Stream. (Inherited from Stream.)
- Dispose(Boolean) - Releases the unmanaged resources used by the MemoryStream class and optionally releases the managed resources. (Overrides Stream.Dispose(Boolean).)
- EndRead - Waits for the pending asynchronous read to complete. (Inherited from Stream.)
- EndWrite - Ends an asynchronous write operation. (Inherited from Stream.)
- Equals(Object) - Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
- Finalize - Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object.)
- Flush - Overrides Stream.Flush so that no action is performed. (Overrides Stream.Flush().)
- GetBuffer - Returns the array of unsigned bytes from which this stream was created
- GetHashCode - Serves as a hash function for a particular type. (Inherited from Object.)
- GetLifetimeService - Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- GetType - Gets the Type of the current instance. (Inherited from Object.)
- InitializeLifetimeService - Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject.)
- MemberwiseClone() - Creates a shallow copy of the current Object. (Inherited from Object.)
- MemberwiseClone(Boolean) - Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject.)
- Read - Reads a block of bytes from the current stream and writes the data to a buffer. (Overrides Stream.Read(Byte[], Int32, Int32).)
- ReadByte - Reads a byte from the current stream. (Overrides Stream.ReadByte().) =

MemoryStream

Methods

- Seek - Sets the position within the current stream to the specified value. (Overrides Stream.Seek(Int64, SeekOrigin).)
- SetLength - Sets the length of the current stream to the specified value. (Overrides Stream.SetLength(Int64).)
- ToArray - Writes the stream contents to a byte array, regardless of the Position property
- ToString - Returns a string that represents the current object. (Inherited from Object.)
- Write - Writes a block of bytes to the current stream using data read from a buffer. (Overrides Stream.Write(Byte[], Int32, Int32).)
- WriteByte - Writes a byte to the current stream at the current position. (Overrides Stream.WriteByte(Byte).)
- WriteTo - Writes the entire contents of this memory stream to another stream

The `System.IO` Namespace

→ Attributes

Attributes

Below are the attributes in the `System.IO` namespace.

Attributes

- IODescriptionAttribute-Sets the description visual designers can display when referencing an event, extender, or property.

The `System.IO` Namespace

→ EventArgs

EventArgs

Below are the `EventArgs` in the `System.IO` namespace.

EventArgs

- ErrorEventArgs-Provides data for the Error event.
- FileSystemEventArgs-Provides data for the directory events: Changed, Created, Deleted.
- RenamedEventArgs-Provides data for the Renamed event.

The `System.IO` Namespace

→ Structures

Structures

Below are the structures available in the `System.IO` namespace.

Structures

- WaitForChangedResult-Contains information on the change that occurred.

The `System.IO` Namespace

→ Interfaces

Interfaces

There are no interfaces in the `System.IO` namespace.

The `System.IO` Namespace

→ Delegates

Delegates

Below are the delegates in the `System.IO` namespace.

Delegates

- ErrorEventHandler-Represents the method that will handle the Error event of a FileSystemWatcher object.
- FileSystemEventHandler-Represents the method that will handle the Changed, Created, or Deleted event of a FileSystemWatcher class.
- RenamedEventHandler-Represents the method that will handle the Renamed event of a FileSystemWatcher class.

The `System.IO` and `System.IO.Compression` Namespaces

→ Enumerations

Enumerations

The following are the enumerations available in the `System.IO` namespace.

Enumerations

- DriveType - Defines constants for drive types, including CDRom, Fixed, Network, NoRootDirectory, Ram, Removable, and Unknown.
- FileAccess - Defines constants for read, write, or read/write access to a file.
- FileAttributes - Provides attributes for files and directories.
- FileMode - Specifies how the operating system should open a file.
- FileOptions - Represents additional options for creating a FileStream object.
- FileShare - Contains constants for controlling the kind of access other FileStream objects can have to the same file.
- HandleInheritability - Specifies whether the underlying handle is inheritable by child processes.
- NotifyFilters - Specifies changes to watch for in a file or folder.
- SearchOption - Specifies whether to search the current directory, or the current directory and all subdirectories.
- SeekOrigin - Provides the fields that represent reference points in streams for seeking.
- WatcherChangeTypes - Changes that might occur to a file or directory.

The following are the enumerations available in the `System.IO.Compression` namespace.

Enumerations

- CompressionMode-Specifies whether to compress or decompress the underlying stream.

The `System.IO` Namespace

→ Exceptions

Exceptions

The following are the exceptions available in the `System.IO` namespace.

Exceptions

- DirectoryNotFoundException-The exception that is thrown when part of a file or directory cannot be found.
- DriveNotFoundException-The exception that is thrown when trying to access a drive or share that is not available.
- EndOfStreamException-The exception that is thrown when reading is attempted past the end of a stream.
- FileFormatException-The exception that is thrown when an input file or a data stream that is supposed to conform to a certain file format specification is malformed.
- FileLoadException-The exception that is thrown when a managed assembly is found but cannot be loaded.
- FileNotFoundException-The exception that is thrown when an attempt to access a file that does not exist on disk fails.
- InternalBufferOverflowException-The exception thrown when the internal buffer overflows.
- InvalidDataException-The exception that is thrown when a data stream is in an invalid format.
- IOException-The exception that is thrown when an I/O error occurs.
- PathTooLongException-The exception that is thrown when a path or file name is longer than the system-defined maximum length.
- PipeException-Thrown when an error occurs within a named pipe.

What Next?

In *C# Programming IV-#*, we look at specific classes within specific namespaces such as the System namespace, the System.IO namespace, etc.

References

Click the book titles below to read more about these books on Amazon.com's website.

- Introducing Microsoft LINQ, Paolo Pialorsi and Marco Russo, Microsoft Press, ISBN:9780735623910
- LINQ Pocket Reference, Joseph Albahari and Ben Albahari, O'Reilly Press, ISBN:9780596519247
- Inside C#, Tom Archer and Andrew Whitechapel, Microsoft Press, ISBN:0735616485
- C# 4.0 In a Nutshell, O'Reilly Press, Joseph Albahari and Ben Albahari, ISBN:9780596800956
- The Object Primer, Scott W. Ambler, Cambridge Press, ISBN:0521540186
- CLR via C#, Jeffrey Richter, Microsoft Press, ISBN:9780735621633

Support sheepsqueezers.com

If you found this information helpful, please consider supporting sheepsqueezers.com. There are several ways to support our site:

- ☐ Buy me a cup of coffee by clicking on the following link and donate to my PayPal account: Buy Me A Cup Of Coffee?.
- ☐ Visit my Amazon.com Wish list at the following link and purchase an item: http://amzn.com/w/3OBK1K4EIWIR6

Please let me know if this document was useful by e-mailing me at `comments@sheepsqueezers.com`.