42

System.IO

Introducing System.IO namespace

1. System.IO.FileInfo
2. System.IO.DirectoryInfo
3. System.IO.Directory
4. System.IO.File
5. System.IO.DriveInfo
6. System.IO.FileStream
7. System.IO.StreamWriter
8. System.IO.StreamReader
9. System.IO.BinaryWriter
10. System.IO.BinaryReader
11. System.Runtime.Serialization.Formatters.Binary.BinaryFormatter
12. System.Xml.Serialization.XmlSerializer
13. Newtonsoft.Json.JsonConvert

› This namespace contains classes to perform File I/O operations.

System.IO.FileInfo

Class that represents a file on the disk and performs manipulations on files.

System.IO.DirectoryInfo

Class that represents a directory (folder) on the disk and performs manipulations on directories.

System.IO.Directory

Static class that manipulates directory (folder).

System.IO.File

Static class that manipulates file.

System.IO.DriveInfo

Class that represents a drive and performs manipulations on drives.

System.IO.FileStream

Class that performs file I/O operations.

System.IO.StreamWriter

Class that writes text data into the file.

System.IO.StreamReader

Class that reads text data from the file.

System.IO.BinaryWriter

Class that writes binary data into the file.

System.IO.BinaryReader

Class that reads binary data from the file.

System.Runtime.Serialization.Formatters.Binary.BinaryFormatter

Class that serializes (converts) an object-state into binary format into and stores in binary file.

System.Xml.Serialization.XmlSerializer

Class that serializes (converts) an object-state into XML format and stores in XML file.

Newtonsoft.Json.JsonConvert

Class that serializes (converts) an object-state into JSON format.

FIleInfo

- Represents a file and provides methods to manipulate the file.

FileInfo

```
FileInfo referenceVariable = new FileInfo( "Your File Path Here");
```

Constructors

Constructor	Description
FileInfo(string)	It initializes the file path which needs to be manipulated.

Properties

Property	Description
bool Exists	Determines whether the file exists in the disk or not.
string FullName	Represents full path (including file name and extension) of the file.
string Name	Represents only name of the file (without path).
string DirectoryName	Represents only path of the file (without file name).
string Extension	Represents only file extension (without file name).
DateTime CreationTime	Represents date and time of file creation.
DateTime LastWriteTime	Represents date and time of last modification of the file.
DateTime LastAccessTime	Represents date and time of last access of the file.
long Length	Represents file size (in the form of no. of bytes).

Methods

#	Method
1	FileStream Create( )
	Creates the file at specified path & creates and returns an object of FileStream class, which can write data to the same file.
2	void Delete( )
	Deletes the file permanently.
3	FileStream Open(FileMode mode, FileAccess access)
	Opens the file at specified file mode and specified file access permission & creates and returns an object of FileStream class that can write / read the file data.
4	FileStream OpenRead( )
	Opens the file in read mode & creates and returns an object of FileStream class that can read the file data.

#	Method
5	StreamWriter AppendText( )
	Opens the file in append mode & creates and returns an object of StreamWriter class that can write the appended text to the file.
6	StreamWriter AppendText( )
	Opens the file in append mode & creates and returns an object of StreamWriter class that can write the appended text to the file.
7	FileInfo CopyTo(string destFilePath)
	Copies the file into the new destination path.
8	void MoveTo(string destFilePath)
	Moves the file into the new destination path.

DirectoryInfo

› Class that represents a directory (folder) on the disk and performs manipulations on directories.

DirectoryInfo
DirectoryInfo referenceVariable = new DirectoryInfo("Your Directory Path Here");

Constructors

Constructor	Description
DirectoryInfo(string)	It initializes the directory path which needs to be manipulated.

Properties

Property	Description
bool Exists	Determines whether the directory exists in the disk or not.
string FullName	Represents full path of the directory
string Name	Represents only name of the directory (without path).
DirectoryInfo Parent	Represents parent directory of the current directory.
DirectoryInfo Root	Represents root (drive) of the directory.
DateTime CreationTime	Represents date and time of directory creation.
DateTime LastWriteTime	Represents date and time of last modification of the directory.
DateTime LastAccessTime	Represents date and time of last access of the directory.

Methods

#	Method
1	void Create( )
	Create the directory at the current path.
2	DirectoryInfo CreateSubDirectory (string path)
	Creates a sub directory at the current path with specified name.
3	void Delete(bool recursive)
	true: Deletes the directory including sub directories. false: Deletes the directory only if it is empty.
4	DirectoryInfo[ ] GetDirectories()
	Returns DirectoryInfo[ ] that represents sub directories of current directory.
5	DirectoryInfo[ ] GetDirectories (string searchPattern)
	Returns DirectoryInfo[ ] that represents sub directories that matches with specified search pattern.

#	Method
6	FileInfo[ ] GetFiles( )
	Returns FileInfo[ ] that represents files of current directory.
7	FileInfo[ ] GetFiles(string searchPattern)
	Returns FileInfo[ ] that represents files that matches with specified search pattern.
8	void MoveTo(string destDirName)
	Moves the current directory to the specified location in the same drive.

Directory

- Static class that manipulates directory (folder).
- All methods of this class are static methods.

Methods

#	Method
1	static DirectoryInfo CreateDirectory(string path)
	Create a directory at the specified path.
2	static void Delete (string path, bool recursive)
	true: Deletes the directory including sub directories. false: Deletes the directory only if it is empty.
3	static bool Exists(string path)
	Determines whether the directory exists in the disk or not.
4	static DirectoryInfo[ ] GetDirectories(string path)
	Returns DirectoryInfo[ ] that represents sub directories of specified directory.
5	static DirectoryInfo[ ] GetDirectories(string path, string searchPattern)
	Returns DirectoryInfo[ ] that represents sub directories that matches with specified search pattern.

#	Method
6	static FileInfo[ ] GetFiles(string path)
	Returns FileInfo[ ] that represents files of specified directory.
7	static FileInfo[ ] GetFiles(string path, string searchPattern)
	Returns FileInfo[ ] that represents files that matches with specified search pattern.
8	static void Move(string sourceDir, string destDir)
	Moves the source directory to the specified destination location in the same drive.

File

- Static class that manipulates file.
- All methods of this class are static methods.

Methods

#	Method
1	static FileStream Create(string path)
	Create a file at the specified path.
2	static void Delete (string path)
	Deletes the specified file permanently.
3	static bool Exists(string path)
	Determines whether the file exists in the disk or not.
4	static FileStream Open(string path)
	Opens the file in specified file mode with specified file access permission.
5	static FileStream OpenRead(string path)
	Opens the file in read mode and returns FileStream.
6	static FileStream OpenWrite(string path)
	Opens the file in write mode and returns FileStream.

#	Method
7	static string[ ] ReadAllLines(string path)
	Reads file content as text and returns all lines.
8	static string ReadAllText(string path)
	Reads file content as text and returns the same.
9	static void WriteAllLines(string path, IEnumerable<string> contents)
	Writes specified lines of content to the file.
10	static void WriteAllText(string path, string contents)
	Writes specified content to the file.
11	static void Copy(string sourceFile, string destFile)
	Copies the source file to the destination location.
12	static void Move(string sourceFile, string destFile)
	Moves the source file to the destination location.

DriveInfo

- Class that represents a drive and performs manipulations on drives.
- You can read both fixed / removable drives.

DriveInfo
DriveInfo referenceVariable = new DriveInfo("Your Drive Name Here");

Constructors

Constructor	Description
DriveInfo(string)	It initializes the drive name which needs to be manipulated.

Properties

Property	Description
string Name	Represents name of the drive.
string DriveType	Represents type of drive either 'Fixed' or 'Removable'
string VolumeLabel	Represents label of the drive (set by user).
DirectoryInfo RootDirectory	Represents root directory of the drive.
long TotalSize	Represents total size (bytes) of the drive.
long AvailableFreeSpace	Represents total free space (bytes) of the drive.

FileStream

- Class that performs file I/O operations.
- Writes / reads data in byte[] format.

FileStream
FileStream referenceVariable = new FileStream("File Path", FileMode.Create, FileAccess.Write);

Constructors

#	Constructor
1	FileStream(string filePath, FileMode mode, FileAccess access)
	It initializes opens the file at specified mode and specified access permission. FileMode: CreateNew, Create, Open, OpenOrCreate, Append FileAccess: Read, Write, ReadWrite

Methods

#	Method
1	void Write(byte[] array, int offset, int count)
	Writes the specified no. of bytes specified by count, based on the byte[] specified by 'array' into the file, after the no. of bytes specified by 'offset'.
2	int Read(byte[] array, int offset, int count)
	Read the specified no. of bytes specified by count, into the byte[] specified by 'array' from the file, after the no. of bytes specified by 'offset'.
3	void Close( )
	Closes the file.

StreamWriter

- Class that writes text data into the file.
- Internally uses FileStream.

StreamWriter
StreamWriter referenceVariable = new StreamWriter("File Path");

Constructors

Constructor	Description
StreamWriter(string path)	It initializes the StreamWriter for the specified file path.
StreamWriter(FileStream stream)	It initializes the StreamWriter based on the specified FileStream.

Methods

#	Method
1	void Write(string content)
	Writes the specified string content to the file.
2	void Close( )
	Close the file.

StreamReader

- Class that reads text data from the file.
- Internally uses FileStream.

StreamReader
StreamReader referenceVariable = new StreamReader("File Path");

Constructors

Constructor	Description
StreamReader(string path)	It initializes the StreamReader for the specified file path.
StreamReader(FileStream stream)	It initializes the StreamReader based on the specified FileStream.

Methods

#	Method
1	string ReadToEnd( )
	Reads complete content of the file in text format and returns the same as a string.
2	void Close( )
	Close the file.
3	string ReadLine( )
	Reads a line from the file in text format and returns the same as a string.
4	int Read( )
	Reads a single next character from the file and returns its ASCII value. It returns -1 if no more characters are available.

BinaryWriter

- Class that writes binary data into the file.
- Internally uses FileStream.
- For example, int x = 56 is written as 00111000 00000000 00000000 00000000.

BinaryWriter
BinaryWriter referenceVariable = new BinaryWriter(fileStream);

Constructors

Constructor	Description
BinaryWriter(FileStream stream)	It initializes the BinaryWriter based on the specified FileStream.

Methods

#	Method
1	void Write(...)
	Writes any type of value (only primitive types / string) into the file.
2	void Close( )
	Close the file.

BinaryReader

- Class that reads binary data from the file.
- Internally uses FileStream.

BinaryReader
`BinaryReader referenceVariable = new BinaryReader( fileStream );`

Constructors

Constructor	Description
BinaryReader(FileStream stream)	It initializes the BinaryReader based on the specified FileStream.

Methods

#	Method
1	byte ReadByte( )
	Reads and returns a byte value from the file.
2	ReadSByte( ), ReadInt16( ), ReadUInt16( ), ReadInt32( ), ReadUInt32( ), ReadInt64( ), ReadUInt64( ), ReadSingle( ), ReadDouble( ), ReadDecimal( ), ReadChar( ), ReadString( ), ReadBoolean( )
3	string Close( )
	Closes the file.

BinaryFormatter

- Class that serializes (converts) an object-state into binary format into and stores in binary file.
- It can also read existing object-state from the binary file.
- **Full Path:** System.Runtime.Serialization.Formatters.Binary.BinaryFormatter

BinaryFormatter
`BinaryFormatter referenceVariable = new BinaryFormatter();`

Methods

#	Method
1	void Serialize(FileStream FileStream, object data)
	Converts the object-state into the binary file, using the specified FileStream (with Write access).
2	object Deserialize(FileStream FileStream)
	Reads the existing object-state from the binary file, using the specified FileStream (with Read access).

XmlSerializer

- Class that serializes (converts) an object-state into XML format and stores in XML file.
- It can also read existing object-state from the XML file.
- Full Path: System.Xml.Serialization.XmlSerializer

XmlSerializer

```
XmlSerializer referenceVariable = new XmlSerializer( typeof(ClassName) );
```

Methods

#	Method
1	void Serialize(FileStream FileStream, object data)
	Converts the object-state into the xml file, using the specified FileStream (with Write access).
2	object Deserialize(FileStream FileStream)
	Reads the existing object-state from the xml file, using the specified FileStream (with Read access).

JsonConvert

- Class that serializes (converts) an object-state into JSON format.
- It can also convert JSON data into object of any class.
- It is available in a NuGet package: Newtonsoft.Json
  - Install-Package Newtonsoft.Json
- Full Path: Newtonsoft.Json.JsonConvert

JsonConvert

```
string Newtonsoft.Json.JsonConvert.SerializeObject( YourObject ); //serialization

T Newtonsoft.Json.JsonConvert.DeserializeObject<T>( string jsonData ); //deserialization
```