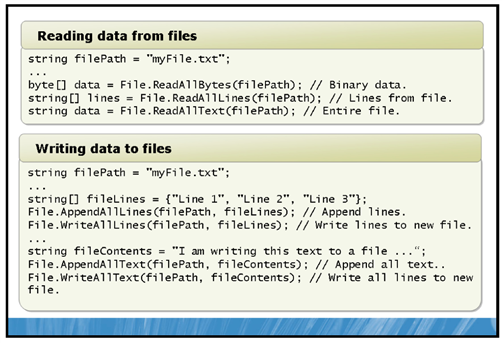
**File Class**

| **Method** | **Description** | **Code example** |
| --- | --- | --- |
| **AppendAllText** | Enables you to open anexisting file, append text tothat file, and then close thefile, all in a single operation. | string filePath =  "...";  string fileContents =  "...";  File.AppendAllText(  filePath,  fileContents); |
| **Copy** | Enables you to copy anexisting file to a newlocation. | string sourceFile =  "...";  string destFile = "...";  bool overwrite = false;  File.Copy(sourceFile,  destFile,  overwrite); |
| **Create** | Enables you to create a newfile on the Windows® filesystem. The **Create** methodreturns a **FileStream**objectthat enables you to interactwith the file by using thestreaming model. Streamsare covered in the nextlesson. | string filePath =  "...";  int bufferSize = 128;  FileStream file =  File.Create(  filePath,  bufferSize,  FileOptions.None); |
| **Delete** | Enables you to delete a filefrom the Windows filesystem. | string filePath =  "...";  File.Delete(filePath); |
| **Exists** | Enables you to determinewhether a file exists. | string filePath = "...";  bool exists =  File.Exists(  filePath); |
| **GetCreationTime** | Enables you to get the   creation time of a file. | string filePath =  "...";  DateTime time =  File.  GetCreationTime(  filePath); |
| **GetLastAccessTime** | Enables you to get the   last access time of a file. | string filePath =  "...";  DateTime time =  File.  GetLastAccessTime(  filePath); |
| **Move** | Enables you to move a file toa new location. You can alsouse this method to rename afile. | string sourceFile =  "...";  string destFile = "...";  File.Move(sourceFile,  destFile); |
| **ReadAllText** | Enables you to read all of thetext from a file into a stringvariable. | string filePath = "...";  string fileContents =  File.ReadAllText(  filePath); |
| **SetCreationTime** | Enables you to set thecreation time of a file. | string filePath =  "...";  File.SetCreationTime(  filePath,  DateTime.Now); |
| **SetLastAccessTime** | Enables you to set the lastaccess time of a file. | string filePath =  "...";  File.SetLastAccessTime(  filePath,  DateTime.Now); |
| **WriteAllText** | Enables you to create a newfile, write text to that file,and then close the file, all ina single operation. | string filePath =  "...";  string fileContents =  "...";  File.WriteAllText(  filePath,  fileContents); |

**The FileInfo Class**

| **Member** | **Description** | **Code example** |
| --- | --- | --- |
| **CreationTime**(property) | Enables you to get or set thecreation time for a particularfile. | string filePath = "...";  FileInfo file = new  FileInfo(filePath);  file.CreationTime =  DateTime.Now;  ...  DateTime time =  file.CreationTime; |
| **CopyTo**(method) | Enables you to copy the fileto a new location on the filesystem. | string filePath = "...";  FileInfo file = new  FileInfo(filePath);  string destPath = "...";  file.CopyTo(destPath); |
| **Delete**(method) | Enables you to delete a file. | string filePath = "...";  FileInfo file = new  FileInfo(filePath);  file.Delete(); |
| **DirectoryName**  (property) | Enables you to get the   directory path to the file. | string filePath = "...";  FileInfo file = new  FileInfo(filePath);  string dirPath =  file.DirectoryName |
| **Exists**(property) | Enables you to determinewhether the file exists. | string filePath = "...";  FileInfo file = new  FileInfo(filePath);  bool exists = file.Exists; |
| **Extension**(property) | Enables you to get the   extension of the file. | string filePath = "...";  FileInfo file = new  FileInfo(filePath);  string ext =  file.Extension; |
| **Length**(property) | Enables you to get the   length of the file in bytes. | string filePath = "...";  FileInfo file = new  FileInfo(filePath);  long length =  file.Length; |
| **Name**(property) | Enables you to get the   name of the file. | string filePath = "...";  FileInfo file = new  FileInfo(filePath);  string name = file.Name; |
| **Open**(method) | Enables you to open a file onthe Windows file system.The **Open** method returns a**FileStream**object thatenables you to interact withthe file by using thestreaming model. Streamsare covered in the nextlesson. | string filePath = "...";  FileInfo file = new  FileInfo(filePath);  FileStream stream =  file.Open(  FileMode.OpenOrCreate); |



### The Directory Class

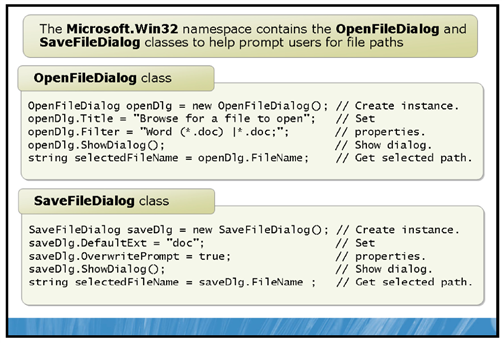
| **Method** | **Description** | **Code example** |
| --- | --- | --- |
| **CreateDirectory** | Enables you to create all of thedirectories that are specified inthe path that don’t already exist. | string dirPath =  @"C:\NewFolder\SubFolder";  Directory.CreateDirectory(  dirPath); |
| **DeleteDirectory** | Enables you to delete one ormore directories from the filesystem. | string dirPath =  @"C:\Users\Student\"  + "MyDirectory";  bool deleteSubFolders = true;  Directory.Delete(  dirPath,  deleteSubFolders); |
| **GetDirectories** | Enables you to get all of   the subdirectories in thespecified path. | string dirPath = "...";  string[] dirs =  Directory.GetDirectories(  dirPath); |
| **GetFiles** | Enables you to get all of   the files in the specified path. | string dirPath = "...";  string[] files =  Directory.GetFiles(  dirPath); |
| **Exists** | Enables you to determinewhether a directory exists at thespecified path. | string dirPath = "...";  bool dirExists =  Directory.Exists(  dirPath); |
| **Move** | Enables you to move adirectory. You cannot use the**Move** method to movedirectories to different drives. | string sourcePath = "...";  string destPath = "...";  Directory.Move(  sourcePath, destPath); |

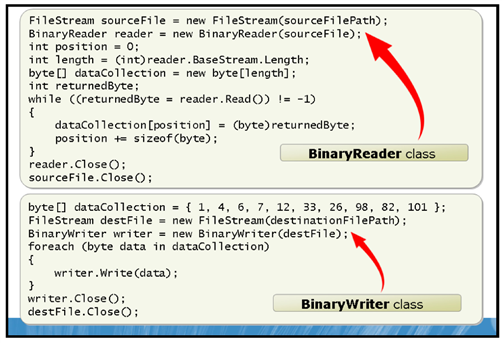
### The DirectoryInfo Class

| **Member** | **Description** | **Code example** |
| --- | --- | --- |
| **Create**(method) | Enables you to create thedirectories in the path specified.If the directory already exists, itis ignored. | string dirPath = "...";  DirectoryInfo dir = new  DirectoryInfo(dirPath);  dir.Create(); |
| **Delete**(method) | Enables you to delete severaldirectories. If the directorycannot be found, a  **DirectoryNotFoundEx** **ception**exception is thrown. | string dirPath = "...";  DirectoryInfo dir = new  DirectoryInfo(dirPath);  dir.Delete(); |
| **Exists**(property) | Enables you to determinewhether a directory exists at thespecified path. | string dirPath = "...";  DirectoryInfo dir = new  DirectoryInfo(dirPath);  bool exists = dir.Exists; |
| **FullName** (property) | Enables you to get the full pathof the directory. | string dirPath = "...";  DirectoryInfo dir = new  DirectoryInfo(dirPath);  string fullName = dir.FullName; |
| **GetDirectories**(method) | Enables you to get all of   the subdirectories in thespecified path. This methodreturns a **DirectoryInfo** array,which enables you to use eachof the **DirectoryInfo** memberson all subdirectories. | string dirPath = "...";  DirectoryInfo dir = new  DirectoryInfo(dirPath);  DirectoryInfo[] dirs =  dir.GetDirectories(); |
| **GetFiles**  (method) | Enables you to get all of   the files in the specified path.This method returns a **FileInfo**array, which enables you to useeach of the **FileInfo** memberson all of the files in thedirectory. | string dirPath = "...";  DirectoryInfo dir = new  DirectoryInfo(dirPath);  FileInfo[] files =  dir.GetFiles(); |
| **MoveTo** (method) | Enables you to move adirectory. You cannot use the**MoveTo** method to movedirectories to different drives. | string dirPath = "...";  DirectoryInfo dir = new  DirectoryInfo(dirPath);  string destPath = "...";  dir.MoveTo(destPath); |
| **Name**(property) | Enables you to get the name ofthe directory. | string dirPath = "...";  DirectoryInfo dir = new  DirectoryInfo(dirPath);  string dirName =  dir.Name; |
| **Parent**  (property) | Enables you to get the parentdirectory. | string dirPath =  @"C:\Users\Student\Music\";  DirectoryInfo dir = new  DirectoryInfo(dirPath);  DirectoryInfo parentDir =  dir.Parent; |

### The Path Class

| **Method** | **Description** | **Code example** |
| --- | --- | --- |
| **GetDirectory   Name** | Enables you to get all of thedirectories in the path. | string path = @"C:\Temp\SubFolder\MyFile.txt";  string dirs =  Path.GetDirectoryName(path); |
| **GetExtension** | Enables you to get theextension of the specifiedfile. | string path = @"C:\Temp\SubFolder\MyFile.txt";  string ext =  Path.GetExtension(path); |
| **GetFileName** | Enables you to get the filename including the extensionfrom the specified path. | string path = @"C:\Temp\SubFolder\MyFile.txt";  string fileName =  Path.GetFileName(path); |
| **GetFileName  Without   Extension** | Enables you to get the filename without the extensionfrom the specified path. | string path = @"C:\Temp\SubFolder\MyFile.txt";  string fileName =  Path.  GetFileNameWithoutExtension(  path); |
| **GetRandom   FileName** | Enables you to generate arandom folder or file name. | string fileName =  Path.GetRandomFileName(); |
| **GetTempFile   Name** | Enables you to create a newtemp file in your localWindows temp folder. Thismethod then returns theabsolute path to that file. | string tempFilePath =  Path.GetTempFileName(); |
| **GetTempPath** | Enables you to get the path tothe local Windows tempfolder. | string tempPath =  Path.GetTempPath(); |



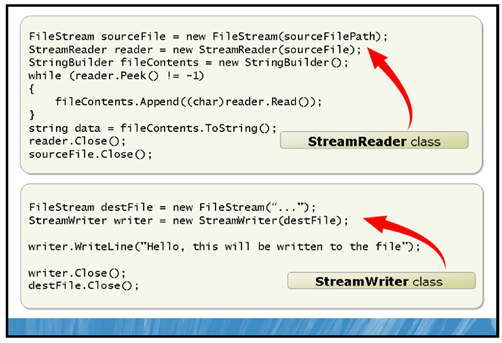


BinaryReader

| **Member** | **Description** |
| --- | --- |
| **BaseStream**(property) | Enables you to access the underlying stream that the **BinaryReader** object uses. |
| **Close**(method) | Enables you to close the **BinaryReader** object and the underlying stream. |
| **Read**(method) | Enables you to read the number of remaining bytes in the stream from a particular index. |
| **ReadByte** (method) | Enables you to read the next byte from the stream, and advance the stream to the next   byte. |
| **ReadBytes**(method) | Enables you to read a specified number of bytes into a byte array. |

BinaryWriter

| **Member** | **Description** |
| --- | --- |
| **BaseStream**(property) | Enables you to access the underlying stream that the **BinaryWriter** object uses. |
| **Close**(method) | Enables you to close the **BinaryWriter** object and the underlying stream. Any data in the buffer will be flushed to theunderlying stream. |
| **Flush**(method) | Enables you to explicitly flush any data in the current buffer to the underlying stream. |
| **Seek**(method) | Enables you to set your position in the current stream, thus writing to a specific byte. |
| **Write**(method) | Enables you to write your data to the stream, and advance the stream. The **Write**method provides several overloads thatenable you to write all primitive data types to a   stream. |



### The StreamReader Class

| **Member** | **Description** |
| --- | --- |
| **Close**(method) | Enables you to close the **StreamReader** object and the underlying stream. |
| **EndOfStream**(property) | Enables you to determine whether you have reached the end of the stream. |
| **Peek**(method) | Enables you to get the next available character in the stream, but does not consume it. |
| **Read**(method) | Enables you to get and consume the next available character in the stream. This method returns an **int**variable thatrepresents the binary of the character, which you may need to explicitly convert. |
| **ReadBlock**(method) | Enables you to read an entire block of characters from a specific index from the stream. |
| **ReadLine** (method) | Enables you to read an entire line of characters from the stream. |
| **ReadToEnd**(method) | Enables you to read all characters from the current position in the stream. |

### The StreamWriter Class

| **Member** | **Description** |
| --- | --- |
| **AutoFlush**(property) | Enables you to instruct the **StreamWriter** object to flush data to the underlying stream   after every write call. |
| **Close**(method) | Enables you to close the **StreamWriter** object and the underlying stream. |
| **Flush**(method) | Enables you to explicitly flush any data in the current buffer to the underlying stream. |
| **NewLine** (property) | Enables you to get or set the characters that are used for new line breaks. |
| **Write**(method) | Enables you to write your data to the stream, and advance the stream. |
| **WriteLine** (method) | Enables you to write your data to the stream followed by a new line break, and then advance the stream. |

**// Source file path.**

**string sourceFilePath =**

**@"C:\Users\Student\Documents\PrimitiveDataTypeFile.txt";**

**// Create a FileStream object so that you can interact with the file**

**// system.**

**FileStream sourceFile = new FileStream(**

**sourceFilePath, // Pass in the source file path.**

**FileMode.Open, // Open an existing file.**

**FileAccess.Read);// Read an existing file.**

**// Create a BinaryWriter object passing in the FileStream object.**

**BinaryReader reader = new BinaryReader(sourceFile);**

**bool boolValue = reader.ReadBoolean();**

**byte byteValue = reader.ReadByte();**

**byte[] byteArrayValue = reader.ReadBytes(4);**

**char charValue = reader.ReadChar();**

**char[] charArrayValue = reader.ReadChars(4);**

**decimal decimalValue = reader.ReadDecimal();**

**double doubleValue = reader.ReadDouble();**

**float floatValue = reader.ReadSingle();**

**int intValue = reader.ReadInt32();**

**long longValue = reader.ReadInt64();**

**sbyte sbyteValue = reader.ReadSByte();**

**short shortValue = reader.ReadInt16();**

**string stringValue = reader.ReadString();**

**uint unintValue = reader.ReadUInt32();**

**ulong ulongValue = reader.ReadUInt64();**

**ushort ushortValue = reader.ReadUInt16();**

**// Close the streams to release any file handles.**

**reader.Close();**

**sourceFile.Close();**

byte byteValue = 1;

writer.Write(byteValue);

byte[] byteArrayValue = { 1, 4, 6, 8 };

writer.Write(byteArrayValue);

char charValue = 'a';

writer.Write(charValue);

char[] charArrayValue = {'a', 'b', 'c', 'd'};

writer.Write(charArrayValue);

decimal decimalValue = 1.00m;

writer.Write(decimalValue);

double doubleValue = 2.5;

writer.Write(doubleValue);

float floatValue = 4.5f;

writer.Write(floatValue);