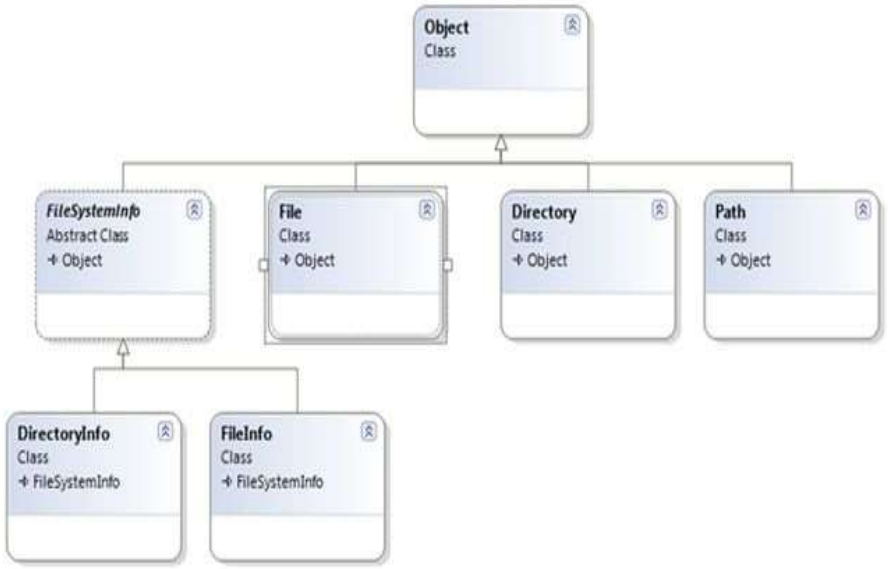
File Handling

Arctech Info Private Limited

The Anatomy of the File System

- The System.IO namespace provides four classes that allow you to manipulate individual files, as well as interact with a machine directory structure. The Directory and File directly extends System.Object and supports the creation, copying, moving and deletion of files using various static methods. They only contain static methods and are never instantiated. The FileInfo and DirecotryInfo types are derived from the abstract class FileSystemInfo type and they are typically, employed for obtaining the full details of a file or directory because their members tend to return strongly typed objects. They implement roughly the same public methods as a Directory and a File but the members of these classes are not static.



DriveInfo

- The System.IO provides a class DriveInfo to manipulate the system drive related tasks. The DriveInfo class provides numerous details such as the total number of drives, calculation of total hard disk space, available space, drive name, ready status, types and so on. Consider the following program that shows the total disk drives:

```
DriveInfo[] di = DriveInfo.GetDrives();
Console.WriteLine("Total Partitions");

foreach(DriveInfo items in di)
{
   Console.WriteLine(items.Name);
}
```

Working with Directories – DirectoryInfo

- DirectoryInfo Class
  - The DirectoryInfo class contains a set of members for the creation, deletion, moving and enumeration over directories and subdirectories.
  - Syntax –
  - DirectoryInfo di=new DirectoryInfo(@"D:\temp\xyz");
- Typically, we make the assumption that the path passed in the constructor of the DirectoryInfo class physically exists. However, if you attempt to interact with a nonexistent directory then the exception will be thrown
- We can also programmatically extends a directory structure using the CreateSubdirectory() method.

Working with Directories – Directory

- The Directory class provides nearly the same functionality as DirecotryInfo. The Directory class typically returns string data rather than strongly typed DirectoryInfo objects.

Reading a file

Reading a Text file

The file class in C# defines two static methods to read a text file namely

- File.ReadAllText()
- File.ReadAllLines().

The **File.ReadAllText()** reads the entire file at once and returns a string. We need to store this string in a variable and use it to display the contents onto the screen.

The **File.ReadAllLines()** reads a file one line at a time and returns that line in string format. We need an array of strings to store each line. We display the contents of the file using the same string array.

Writing a text file

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

Stream

- The .NET provides many objects such as FileStream, StreamReader/Writer, BinaryReader/Writer to read from and write data to a file. A stream basically represents a chunk of data flowing between a source and a destination. Stream provides a common way to interact with a sequence of bytes regardless of what kind of devices store or display the bytes.

Methods	Description
Read()/ ReadByte()	Read a sequence of bytes from the current stream.
Write()/WriteByte()	Write a sequence of bytes to the current stream.
Seek()	Sets the position in the current stream.
Position()	Determine the current position in the current stream.
Length()	Return the length of the stream in bytes.
Flush()	Updates the underlying data source with the current state of the buffer and then clears the buffer.
Close()	Closes the current stream and releases any associated stream resources.

FileStream

- A FileStream instance is used to read or write data to or from a file. In order to construct a FileStream, first we need a file that we want to access. Second, the mode that indicates how we want to open the file. Third, the access that indicates how we want to access a file. And finally, the share access that specifies whether you want exclusive access to the file.

Enumeration	Values
FileMode	Create, Append, Open, CreateNew, Truncate, OpenOrCreate
FileAccess	Read, Write, ReadWrite
FileShare	Inheritable, Read, None, Write, ReadWrite

StreamWriter

- The StreamWriter class implements TextWriter for writing character to stream in a particular format.

Method	Description
Close()	Closes the current StreamWriter object and stream associate with it.
Flush()	Clears all the data from the buffer and write it in the stream associate with it.
Write()	Write data to the stream. It has different overloads for different data types to write in stream.
WriteLine()	It is same as Write() but it adds the newline character at the end of the data.

StreamReader

- The StreamReader class implements TextReader for reading character from the stream in a particular format.

Method	Description
Close()	Closes the current StreamReader object and stream associate with it.
Peek()	Returns the next available character but does not consume it.
Read()	Reads the next character in input stream and increment characters position by one in the stream
ReadLine()	Reads a line from the input stream and return the data in form of string
Seek()	It is use to read/write at the specific location from a file

BinaryReader and BinaryWriter

- The BinaryReader and Writer class allows you to read and write discrete data types to an underlying stream in a compact binary format. The BinaryWriter class defines a highly overloaded Write method to place a data type in the underlying stream.

Members	Description	Class
Write	Write the value to current stream	BinaryWriter
Seek	Set the position in the current stream	BinaryWriter
Close	Close the binary reader	BinaryWriter
Flush	Flush the binary stream	BinaryWriter
PeekChar	Return the next available character without advancing the position in the stream	BinaryReader
Read	Read a specified set of bytes or characters and store them in the incoming array.	BinaryReader