File IO

- File is considered as "Unformatted Uniform Stream of Bytes".
- When a file is opened for reading or writing, it becomes a stream.
- The stream is basically the sequence of bytes passing through the communication path. There are two main streams: the input stream and the output stream.
- The input stream is used for reading data from file (read operation) and the output stream is used for writing into the file (write operation).
- The System.IO namespace has various classes that are used for performing numerous operations with files, such as creating and deleting files, reading from or writing to a file, closing a file etc.
- In C# we can perform Byte oriented, String oriented and Binary file IO depends on the applications requirements.

Below applications demonstrates different ways inn which we can perform File IO in C#

Byte Oriented File IO

In this type of file IO we have to perform file operations in terms of bytes.

```
using System;
using System.IO;

class Marvellous
{
	static void Main(string[] args)
	{
		FileStream F = new FileStream("Marvellous.dat", FileMode.OpenOrCreate,
		                                          FileAccess.ReadWrite);

		for (int i = 1; i <= 50; i++)
		{
			F.WriteByte((byte)i);
		}

		F.Position = 0;

		Console.WriteLine("Contents from file are : ");

		for (int i = 0; i <= 50; i++)
		{
			Console.Write(F.ReadByte() + " ");
		}

		F.Close();
	}
}
```

String Oriented File IO

In this type of file IO we have to read and write the data in terms of string.

```
using System;
using System.IO;

class Marvellous
{
	static void Main(string[] args)
	{
		string[] Batches = new string[] {"PPA", "Logic Building", "Angular", "C# .Net", "Industrial Project Development", "UNIX Internals", "Python with Automation"};

		using (StreamWriter sw = new StreamWriter("MarvellousBatches.txt"))
		{
			foreach (string s in Batches)
			{
				sw.WriteLine(s);
			}
		}

		string line = "";
		using (StreamReader sr = new StreamReader("MarvellousBatches.txt"))
		{
			while ((line = sr.ReadLine()) != null)
			{
				Console.WriteLine(line);
			}
		}
	}
}
```

Binary File IO

The BinaryReader and BinaryWriter classes are used for reading from and writing to a binary file.
The BinaryReader class is used to read binary data from a file. A BinaryReader object is created by passing a FileStream object to its constructor.
The BinaryWriter class is used to write binary data to a stream. A BinaryWriter object is created by passing a FileStream object to its constructor.

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

```
namespace BinaryReadWrite
{
    class Program
    {
        static void Main(string[] args)
        {
            BinaryWriter bw;
            BinaryReader br;

            int i = 65;
            double d = 3.14157;
            bool b = true;
            string s = "Marvellous Infosystems";

            //create the file
            try
            {
                bw = new BinaryWriter(new FileStream("MarvellousBinary",
FileMode.Create));
            }
            catch (IOException e)
            {
                Console.WriteLine(e.Message + "\n Cannot create file.");
                return;
            }

            //writing into the file
            try
            {
                bw.Write(i);
                bw.Write(d);
                bw.Write(b);
                bw.Write(s);
            }
            catch (IOException e)
            {
                Console.WriteLine(e.Message + "\n Cannot write to file.");
                return;
            }
            bw.Close();


            //reading from the file
            try
            {
                br = new BinaryReader(new FileStream("MarvellousBinary", FileMode.Open));
            }
            catch (IOException e)
            {
                Console.WriteLine(e.Message + "\n Cannot open file.");
                return;
            }

            try
            {
                i = br.ReadInt32();
                Console.WriteLine("Integer data: {0}", i);

                d = br.ReadDouble();
                Console.WriteLine("Double data: {0}", d);

                b = br.ReadBoolean();
                Console.WriteLine("Boolean data: {0}", b);

                s = br.ReadString();
                Console.WriteLine("String data: {0}", s);
            }
            catch (IOException e)
            {
                Console.WriteLine(e.Message + "\n Cannot read from file.");
                return;
            }
            br.Close();
        }
    }
}
```