**Leer y escribir en un archivo de datos recién creado**

Las clases [BinaryWriter](http://msdn.microsoft.com/es-es/library/system.io.binarywriter(v=VS.80).aspx) y [BinaryReader](http://msdn.microsoft.com/es-es/library/system.io.binaryreader(v=VS.80).aspx) se usan para escribir y leer datos, en lugar de cadenas de caracteres.

En el siguiente ejemplo de código se muestra cómo se escriben y se leen datos en una nueva secuencia de archivos vacía (Test.data).

Después de crear el archivo de datos en el directorio actual, se crean las clases **BinaryWriter** y **BinaryReader** asociadas, y **BinaryWriter** se usa para escribir los enteros de 0 a 10 en Test.data, que deja el puntero de archivo al final del archivo. Después de volver a establecer el puntero de archivo en el origen, **BinaryReader** lee el contenido especificado.

**Ejemplo**

using System;

using System.IO;

class MyStream

{

private const string FILE\_NAME = "Test.data";

public static void Main(String[] args)

{

// Create the new, empty data file.

if (File.Exists(FILE\_NAME))

{

Console.WriteLine("{0} already exists!", FILE\_NAME);

return;

}

FileStream fs = new FileStream(FILE\_NAME, FileMode.CreateNew);

// Create the writer for data.

BinaryWriter w = new BinaryWriter(fs);

// Write data to Test.data.

for (int i = 0; i < 11; i++)

{

w.Write( (int) i);

}

w.Close();

fs.Close();

// Create the reader for data.

fs = new FileStream(FILE\_NAME, FileMode.Open, FileAccess.Read);

BinaryReader r = new BinaryReader(fs);

// Read data from Test.data.

for (int i = 0; i < 11; i++)

{

Console.WriteLine(r.ReadInt32());

}

r.Close();

fs.Close();

}

}

**Cómo: Abrir y anexar a un archivo de registro**

[StreamWriter](http://msdn.microsoft.com/es-es/library/system.io.streamwriter(v=VS.80).aspx) y [StreamReader](http://msdn.microsoft.com/es-es/library/system.io.streamreader(v=VS.80).aspx) escriben y leen caracteres de secuencias. En el siguiente ejemplo de código, se abre el archivo log.txt para escribir en él, o se crea el archivo si todavía no existe y se agrega información al final del archivo. Seguidamente, se escribe el contenido del archivo en la salida estándar para su presentación. Como alternativa a este ejemplo,la información se podría almacenar como una única cadena o matriz de cadenas, y el método [WriteAllText](http://msdn.microsoft.com/es-es/library/system.io.file.writealltext(v=VS.80).aspx) o [WriteAllLines](http://msdn.microsoft.com/es-es/library/system.io.file.writealllines(v=VS.80).aspx) se podría emplear para conseguir la misma funcionalidad.

|  |
| --- |
| **NoteNota** |
| Los usuarios de Visual Basic pueden elegir utilizar los métodos y propiedades proporcionados por los objetos **My.Application.Log** o **My.Computer.FileSystem** para crear archivos de registro o escribir en ellos. Para obtener más información, vea [My.Application.Log (Objeto)](http://msdn.microsoft.com/es-es/library/5049kcd2(v=VS.80).aspx) y [My.Computer.FileSystem (Objeto)](http://msdn.microsoft.com/es-es/library/0b485hf7(v=VS.80).aspx). |

**Ejemplo**

[VB](javascript:%20CodeSnippet_SetLanguage('Visual%20Basic');)

C#

[C++](javascript:%20CodeSnippet_SetLanguage('Visual%20C++');)

[F#](javascript:%20CodeSnippet_SetLanguage('F#');)

[JScript](javascript:%20CodeSnippet_SetLanguage('JScript');)

[Copiar](javascript:CodeSnippet_CopyCode('CodeSnippetContainerCode0');)

using System;

using System.IO;

class DirAppend

{

public static void Main(String[] args)

{

using (StreamWriter w = File.AppendText("log.txt"))

{

Log ("Test1", w);

Log ("Test2", w);

// Close the writer and underlying file.

w.Close();

}

// Open and read the file.

using (StreamReader r = File.OpenText("log.txt"))

{

DumpLog (r);

}

}

public static void Log (String logMessage, TextWriter w)

{

w.Write("\r\nLog Entry : ");

w.WriteLine("{0} {1}", DateTime.Now.ToLongTimeString(),

DateTime.Now.ToLongDateString());

w.WriteLine(" :");

w.WriteLine(" :{0}", logMessage);

w.WriteLine ("-------------------------------");

// Update the underlying file.

w.Flush();

}

public static void DumpLog (StreamReader r)

{

// While not at the end of the file, read and write lines.

String line;

while ((line=r.ReadLine())!=null)

{

Console.WriteLine(line);

}

r.Close();

}

}

<http://msdn.microsoft.com/es-es/library/z4kzt0dd(v=VS.80).aspx>