**Working with Files**

There are two major categories of file manipulation:

* Creating, adding, or removing data, and reading files
* Moving, copying, and deleting files

[Creating Files](javascript:void(0))

There are three ways to create an empty text file (sometimes referred to as a "text stream").

The first way is to use the **CreateTextFile** method. The following example demonstrates how to create a text file using the **CreateTextFile** method.

VBScript

Dim fso, f1

Set fso = CreateObject("Scripting.FileSystemObject")

Set f1 = fso.CreateTextFile("c:\testfile.txt", True)

JScript

var fso, f1;

fso = new ActiveXObject("Scripting.FileSystemObject");

f1 = fso.CreateTextFile("c:\\testfile.txt", true);

The second way to create a text file is to use the **OpenTextFile** method of the **FileSystemObject** object with the **ForWriting** flag set.

VBScript

Dim fso, ts

Const ForWriting = 2

Set fso = CreateObject("Scripting. FileSystemObject")

Set ts = fso.OpenTextFile("c:\test.txt", ForWriting, True)

JScript

var fso, ts;

var ForWriting= 2;

fso = new ActiveXObject("Scripting.FileSystemObject");

ts = fso.OpenTextFile("c:\\test.txt", ForWriting, true);

A third way to create a text file is to use the **OpenAsTextStream** method with the **ForWriting** flag set.

VBScript

Dim fso, f1, ts

Const ForWriting = 2

Set fso = CreateObject("Scripting.FileSystemObject")

fso.CreateTextFile ("c:\test1.txt")

Set f1 = fso.GetFile("c:\test1.txt")

Set ts = f1.OpenAsTextStream(ForWriting, True)

JScript

var fso, f1, ts;

var ForWriting = 2;

fso = new ActiveXObject("Scripting.FileSystemObject");

fso.CreateTextFile ("c:\\test1.txt");

f1 = fso.GetFile("c:\\test1.txt");

ts = f1.OpenAsTextStream(ForWriting, true);

[Adding Data to the File](javascript:void(0))

Once the text file is created, add data to the file using the following three steps:

Open the text file.

Write the data.

Close the file.

To open an existing file, use either the **OpenTextFile** method of the **FileSystemObject** object or the **OpenAsTextStream** method of the **File** object.

To write data to the open text file, use the **Write**, **WriteLine**, or **WriteBlankLines** methods of the [TextStream Object](http://msdn.microsoft.com/en-us/library/312a5kbt(v=vs.84).aspx), according to the tasks outlined in the following table.

|  |  |
| --- | --- |
| **Task** | **Method** |
| Write data to an open text file without a trailing newline character. | Write |
| Write data to an open text file with a trailing newline character. | WriteLine |
| Write one or more blank lines to an open text file. | WriteBlankLines |

To close an open file, use the **Close** method of the **TextStream** object.

|  |
| --- |
| **NoteNote:** |
| The newline character contains a character or characters (depending on the operating system) to advance the cursor to the beginning of the next line (carriage return/line feed). Be aware that the end of some strings may already have such nonprinting characters. |

The following example demonstrates how to open a file, use all three write methods to add data to the file, and then close the file:

VBScript

Sub CreateFile()

Dim fso, tf

Set fso = CreateObject("Scripting.FileSystemObject")

Set tf = fso.CreateTextFile("c:\testfile.txt", True)

' Write a line with a newline character.

tf.WriteLine("Testing 1, 2, 3.")

' Write three newline characters to the file.

tf.WriteBlankLines(3)

' Write a line.

tf.Write ("This is a test.")

tf.Close

End Sub

JScript

function CreateFile()

{

var fso, tf;

fso = new ActiveXObject("Scripting.FileSystemObject");

tf = fso.CreateTextFile("c:\\testfile.txt", true);

// Write a line with a newline character.

tf.WriteLine("Testing 1, 2, 3.") ;

// Write three newline characters to the file.

tf.WriteBlankLines(3) ;

// Write a line.

tf.Write ("This is a test.");

tf.Close();

}

[Reading Files](javascript:void(0))

To read data from a text file, use the **Read**, **ReadLine**, or **ReadAll** method of the [TextStream Object](http://msdn.microsoft.com/en-us/library/312a5kbt(v=vs.84).aspx). The following table describes which method to use for various tasks.

|  |  |
| --- | --- |
| **Task** | **Method** |
| Read a specified number of characters from a file. | Read |
| Read an entire line (up to, but not including, the newline character). | ReadLine |
| Read the entire contents of a text file. | ReadAll |

If you use the **Read** or **ReadLine** method and want to skip to a particular portion of data, use the **Skip** or **SkipLine** method. The resulting text of the read methods is stored in a string which can be displayed in a control, parsed by string functions (such as **Left**, **Right**, and **Mid**), concatenated, and so forth.

Use the **AtEndOfStream** property to check whether a file pointer has reached the end of [TextStream Object](http://msdn.microsoft.com/en-us/library/312a5kbt(v=vs.84).aspx) file.

The following example demonstrates how to open a file, write to it, and then read from it:

VBScript

Sub ReadFiles

Dim fso, f1, ts, s

Const ForReading = 1

Set fso = CreateObject("Scripting.FileSystemObject")

Set f1 = fso.CreateTextFile("c:\testfile.txt", True)

' Write a line.

Response.Write "Writing file <br>"

f1.WriteLine "Hello World"

f1.WriteBlankLines(1)

f1.Close

' Read the contents of the file.

Response.Write "Reading file <br>"

Set ts = fso.OpenTextFile("c:\testfile.txt", ForReading)

s = ts.ReadLine

Response.Write "File contents = '" & s & "'"

ts.Close

End Sub

JScript

function ReadFiles()

{

var fso, f1, ts, s;

var ForReading = 1;

fso = new ActiveXObject("Scripting.FileSystemObject");

f1 = fso.CreateTextFile("c:\\testfile.txt", true);

// Write a line.

Response.Write("Writing file <br>");

f1.WriteLine("Hello World");

f1.WriteBlankLines(1);

f1.Close();

// Read the contents of the file.

Response.Write("Reading file <br>");

ts = fso.OpenTextFile("c:\\testfile.txt", ForReading);

s = ts.ReadLine();

Response.Write("File contents = '" + s + "'");

ts.Close();

}

[Moving, Copying, and Deleting Files](javascript:void(0))

The FSO object model has two methods each for moving, copying, and deleting files, as described in the following table.

|  |  |
| --- | --- |
| **Task** | **Method** |
| Move a file | File.Move or FileSystemObject.MoveFile |
| Copy a file | File.Copy or FileSystemObject.CopyFile |
| Delete a file | File.Delete or FileSystemObject.DeleteFile |

The following example creates a text file in the root directory of drive C, writes some information to it, moves it to a directory called \tmp, makes a copy of it in a directory called \temp, then deletes the copies from both directories.

To run the following example, create directories named \tmp and \temp in the root directory of drive C:

VBScript

Sub ManipFiles

Dim fso, f1, f2, s

Set fso = CreateObject("Scripting.FileSystemObject")

Set f1 = fso.CreateTextFile("c:\testfile.txt", True)

Response.Write "Writing file <br>"

' Write a line.

f1.Write ("This is a test.")

' Close the file to writing.

f1.Close

Response.Write "Moving file to c:\tmp <br>"

' Get a handle to the file in root of C:\.

Set f2 = fso.GetFile("c:\testfile.txt")

' Move the file to \tmp directory.

f2.Move ("c:\tmp\testfile.txt")

Response.Write "Copying file to c:\temp <br>"

' Copy the file to \temp.

f2.Copy ("c:\temp\testfile.txt")

Response.Write "Deleting files <br>"

' Get handles to files' current location.

Set f2 = fso.GetFile("c:\tmp\testfile.txt")

Set f3 = fso.GetFile("c:\temp\testfile.txt")

' Delete the files.

f2.Delete

f3.Delete

Response.Write "All done!"

End Sub

JScript

function ManipFiles()

{

var fso, f1, f2, s;

fso = new ActiveXObject("Scripting.FileSystemObject");

f1 = fso.CreateTextFile("c:\\testfile.txt", true);

Response.Write("Writing file <br>");

// Write a line.

f1.Write("This is a test.");

// Close the file to writing.

f1.Close();

Response.Write("Moving file to c:\\tmp <br>");

// Get a handle to the file in root of C:\.

f2 = fso.GetFile("c:\\testfile.txt");

// Move the file to \tmp directory.

f2.Move ("c:\\tmp\\testfile.txt");

Response.Write("Copying file to c:\\temp <br>");

// Copy the file to \temp.

f2.Copy ("c:\\temp\\testfile.txt");

Response.Write("Deleting files <br>");

// Get handles to files' current location.

f2 = fso.GetFile("c:\\tmp\\testfile.txt");

f3 = fso.GetFile("c:\\temp\\testfile.txt");

// Delete the files.

f2.Delete();

f3.Delete();

Response.Write("All done!");

}