Playing with NTFS File Streams

CalvinH 26 Sep 2006 2:11 AM 3

I was browsing MSDN, and I came across this article: A Programmer's Perspective on NTFS 2000 Part 1: Stream and Hard Link

So I copied the code from Calling the Windows APIs for Large Files, modified it a little, and came up with a sample program (Fox and VB versions below).

A file can contain data, which is a single sequence of bytes. A file on an NTFS volume can contain multiple streams, which means multiple sequences of data. Programmers usually think of a file as containing only one stream of data. In fact, the OS doesn't have very much support for streams, as the article says. For example, Windows Explorer doesn't show the disk space taken by a non-mainstream stream.

The VB sample has a routine that creates a temporary file, and writes some text into it. Then it shows the file size and reads the text back from the file.

The routine is called twice: once with an empty stream name, the 2nd time with a stream name. The 2nd time, the file size shows as having 0 bytes, even though the bytes can be read back from the stream!

The Fox version goes further: it demonstrates that the file can be a table used by VFP at the same time the program reads and writes to the stream. You can embed lots of hidden data into the table!

As the title says, this only works on NTFS volumes.

```
#DEFINE CREATE_NEW                  1
#DEFINE CREATE_ALWAYS               2
#DEFINE OPEN_EXISTING               3
#DEFINE FILE_ATTRIBUTE_NORMAL       128
#DEFINE GENERIC_READ          2147483648  && 0x80000000
#DEFINE GENERIC_WRITE         1073741824  && 0x40000000
#DEFINE GENERIC_ALL           268435456  && 0x10000000
#DEFINE MAXIMUM_ALLOWED         33554432  && 0x02000000
#DEFINE STANDARD_RIGHTS_ALL      2031616  && 0x001F0000
#DEFINE FILE_SHARE_READ             1
#DEFINE FILE_SHARE_WRITE            2
#DEFINE FILE_SHARE_DELETE           4
#DEFINE INVALID_HANDLE_VALUE        -1
#DEFINE  FILE_BEGIN  0
#DEFINE  FILE_CURRENT  1
#DEFINE  FILE_END  2
#DEFINE  MAXDWORD 4294967295

CLEAR
CLOSE DATABASES all

ox=CREATEOBJECT("TestStream")
DEFINE CLASS TestStream as Custom
        PROCEDURE init
                DECLARE INTEGER CreateFile IN kernel32 STRING    lpFileName,;
                      INTEGER   dwDesiredAccess, INTEGER   dwShareMode,;
                      INTEGER   lpSecurityAttr, INTEGER   dwCreationDisp,;
                      INTEGER   dwFlagsAndAttrs, INTEGER   hTemplateFile

                DECLARE INTEGER CloseHandle IN kernel32 INTEGER hObject
                DECLARE long GetLastError in win32api
                DECLARE INTEGER WriteFile IN WIN32API integer hFile, ;
                                        string lpBuffer, ;
                                        integer nBytes, ;
                                        integer @nWritten, ;
                                        integer lpOverlapped
                DECLARE INTEGER ReadFile IN WIN32API integer hFile, ;
                                        string lpBuffer, ;
                                        integer nBytes, ;
                                        integer @nRead, ;
                                        integer lpOverlapped

                DECLARE long SetFilePointer IN kernel32;
                   long hnd,;
                   long lDistanceToMove,;
                   long @lDistanceToMoveH,;
                   long dwMoveMethod
```

```
DECLARE long SetFilePointerEx IN kernel32;
    long hnd,;
    long lDistanceToMoveL,;
    long lDistanceToMoveH,;
    string @lpNewFilePointer,;
    long dwMoveMethod
DECLARE integer GetLastError IN win32api
SET COMPATIBLE ON && so FSIZE() reports file size
cFileName = "t1.dbf"
cStreamName=":stream1"
ERASE (cFileName)
CREATE TABLE (cFileName) (name c(10), address c(20))
INSERT INTO (DBF()) VALUES ("Fred","123 Rock St")
INSERT INTO (DBF()) VALUES ("Barney","125 Stone St")
USE
?"File size before writing stream = "+TRANSFORM(FSIZE(cFilename))+" bytes"
fhStream = this.CreateNewFileHandle(cFileName,cStreamName,CREATE_NEW)
this.WriteToHandle(fhStream,REPLICATE("Four score and seven years ago ",2))
CloseHandle(fhStream)
?"File size after  writing stream = "+TRANSFORM(FSIZE(cFilename))+" bytes"
fhStream = this.CreateNewFileHandle(cFileName,cStreamName,OPEN_EXISTING)
?this.ReadFromHandle(fhStream)
USE (cFileName)       && We can even use the file while the stream is open
LIST
this.WriteToHandle(fhStream,REPLICATE("More Text ",2))          && write to the
stream while VFP has it open too!
LIST
SetFilePointer(fhStream,0, 0, 0)     && go to stream BOF()
?this.ReadFromHandle(fhStream)
LIST
CloseHandle(fhStream)
USE
?"File size after  writing stream again = "+TRANSFORM(FSIZE(cFilename))+" bytes"
PROCEDURE WriteToHandle(fh as Integer,cTestString as String) as Integer
LOCAL nWritten
nWritten=0
writefile(fh,cTestString,LEN(cTestString),@nWritten,0)
RETURN nWritten

PROCEDURE ReadFromHandle(fh as Integer) as String
LOCAL nRead,buf
nRead=0
buf=SPACE(1000)
ReadFile(fh,@buf, LEN(buf),@nread,0)
buf=LEFT(buf,nRead)
RETURN buf

PROCEDURE CreateNewFileHandle(cFileName as String, cStreamName as String, nOpenMode
as Integer)  as Integer
LOCAL fh
fh = CreateFile(cFileName+cStreamName  , ;
          GENERIC_WRITE + GENERIC_READ,;
          FILE_SHARE_WRITE + FILE_SHARE_READ,;
          0,;
          nOpenMode,;
          FILE_ATTRIBUTE_NORMAL,;
          0)
 IF fh = INVALID_HANDLE_VALUE
        ?"Could not open "+cFileName+cStreamName
        SUSPEND
        RETURN -1
 ENDIF
 RETURN fh
ENDDEFINE
```

Now the VB version:

```
' create a new VB Console application
Imports System.Runtime.InteropServices
Imports System.io
Module Module1
```

```
    Dim cFileName As String = "c:\t.txt"
    Dim cTestString As String = "Four score and seven years ago"
    Declare Function ReadFile Lib "kernel32" (ByVal hFile As IntPtr, ByVal Buffer
As IntPtr) As Integer
    Const CREATE_NEW = 1
    Const CREATE_ALWAYS = 2
    Const OPEN_EXISTING = 3
    Const FILE_ATTRIBUTE_NORMAL = 128
    Const GENERIC_READ = &H80000000
    Const GENERIC_WRITE = 1073741824  '&& 0x40000000
    Const GENERIC_ALL = 268435456  '&& 0x10000000
    Const MAXIMUM_ALLOWED = 33554432  '&& 0x02000000
    Const STANDARD_RIGHTS_ALL = 2031616  '&& 0x001F0000
    Const FILE_SHARE_READ = 1
    Const FILE_SHARE_WRITE = 2
    Const FILE_SHARE_DELETE = 4
    Const INVALID_HANDLE_VALUE = -1
    Const FILE_BEGIN = 0
    Const FILE_CURRENT = 1
    Const FILE_END = 2
    Const MAXDWORD = 4294967295

    <DllImport("kernel32.dll", SetLastError:=True)> _
    Private Function CreateFile(ByVal lpFileName As String, ByVal dwDesiredAccess
As IntPtr, _
        ByVal dwShareMode As IntPtr, _
        ByVal lpSecurityAttributes As IntPtr, _
        ByVal dwCreationDisposition As IntPtr, _
        ByVal dwFlagsAndAttributes As IntPtr, ByVal hTemplateFile As IntPtr) As
IntPtr
    End Function
    <DllImport("kernel32.dll", SetLastError:=True)> _
    Private Function ReadFile(ByVal hFile As IntPtr, ByVal aBytes As Byte(), _
        ByVal nBytes As IntPtr, _
        ByRef nRead As IntPtr, _
        ByVal lpOverlapped As IntPtr) As IntPtr
    End Function
    <DllImport("kernel32.dll", SetLastError:=True)> _
    Private Function WriteFile(ByVal hFile As IntPtr, ByVal aBytes As Byte(), _
        ByVal nBytes As IntPtr, _
        ByRef nWritten As IntPtr, _
        ByVal lpOverlapped As IntPtr) As IntPtr
    End Function
    <DllImport("kernel32.dll", SetLastError:=True)> _
    Private Function CloseHandle(ByVal hFile As IntPtr) As IntPtr
    End Function

    Sub Main()
        Dim cStreamName As String = ""
        Console.WriteLine("First: normal read and write of a string to file")
        TestString(cFileName, cStreamName, cTestString)

        cStreamName = ":stream1"
        Console.WriteLine("Now using a stream. Note the file length is 0")
        TestString(cFileName, cStreamName, cTestString)
        Console.ReadLine()
    End Sub
    Sub TestString(ByVal cFileName As String, ByVal cStreamName As String, ByVal
cTestString As String)
        Dim encA As New System.Text.ASCIIEncoding
        If File.Exists(cFileName) Then
            File.Delete(cFileName)
        End If
        If True Then
            Dim hf As Integer = CreateFile(cFileName + cStreamName, GENERIC_WRITE,
FILE_SHARE_WRITE, 0, CREATE_NEW, 0, 0)
            If hf = INVALID_HANDLE_VALUE Then
                MsgBox("error creating file")
            Else
                Dim nWritten As Integer = 0
                WriteFile(hf, encA.GetBytes(cTestString), cTestString.Length,
nWritten, 0)
                CloseHandle(hf)
                Dim fi As New FileInfo(cFileName)
                Console.WriteLine("# Bytes written= " & nWritten & " File size= " &
Int(fi.Length))
                Console.WriteLine("Now read from file:")
                hf = CreateFile(cFileName + cStreamName, GENERIC_READ,
```

```
FILE_SHARE_READ, 0, OPEN_EXISTING, 0, 0)
                Dim cBuf(1000) As Byte
                Dim nRead As Integer = 0
                ReadFile(hf, cBuf, cBuf.Length - 1, nRead, 0)
                CloseHandle(hf)
                ReDim Preserve cBuf(nRead)
                fi = New FileInfo(cFileName)
                Console.WriteLine("File Size=" & Int(fi.Length) & " # Bytes Read="
& nRead & " " & encA.GetString(cBuf))
                '               MsgBox("read a string: " + encA.GetString(cBuf))
                Console.WriteLine("" + vbCrLf)
            End If

        Else
            Dim fh As FileStream = File.Create(cFileName + ":stream1")  ' this
approach doesn't work: illegal file name
            fh.Write(encA.GetBytes(cTestString), 0, cTestString.Length)
            fh.Close()
        End If
    End Sub

End Module
```

Comments

Vassilis 26 Sep 2006 9:51 AM #

*Great great great post Calvin! Thank you!!!

wOOdy 26 Sep 2006 11:44 AM #

So why can't we use filestreams with FOPEN(), FWRITE() etc ? Would be a nice enhancement for Sedna ;)

余啊雷 8 Jun 2009 6:10 PM #

PingBack from http://insomniacuresite.info/story.php?id=2097