File Utility v1.0

by Richard King

HAVOC Software ©1997

Read Register.doc

This Dynamic Link Library (DLL) was created in Visual Basic 4.0 Enterprise Edition by Richard King of HAVOC Software. The Files Utility DLL is a class that makes the headiest task of doing everyday file functions easy and less time consuming. This DLL holds 12 file functions which are described in more detail below in examples, but first I will tell you how to register the file utility DLL.

INSTRUCTIONS to REGISTER

...\regsvr32 hsfileutil32.dll

CONSTANTS:

```
Global Const HS0x_FNAME = &O12
Global Const HS0x_PATH = &O13
Global Const HS0x_EXT = &O11
Global Const HS0x_SFNAME = &O14
Global Const HS0x_LNGFORMAT = &O20
Global Const HS0x_SHRTFORMAT = &O21
Global Const HS0s_COMPNAME = "CompanyName"
Global Const HS0s_VERNUM = "FileVersion"
Global Const HS0s_FDISC = "FileDescription"
Global Const HS0s_COPYRIGHT = "LegalCopyright"
Global Const HS0x_LONGPATH = &O30
Global Const HS0x_SHRTPATH = &O31
```

USAGE:

You must define the new object class. ClsFileUtils is the name of the class to be called by for example dowb below:

```
Global <ObjName> As New clsFileUtils
```

FUNCTION USAGE:

CopyFile: This function uses the true Windows 95 copy procedure, and copies a file from a source destination to a target destination. It returns a long value; hFilehandle.

```
<ObjName>.CopyFile(source,target)

lngHnd& = <ObjName>.CopyFile("C:\autoexec.bat","D:\backup\autoexec.bak")
```

DeleteFile: This function uses the true Windows 95 delete procedure, and will delete the specified file given within the prameters of the function. It returns a long value; hFile handle.

<ObjectName>.DeleteFile(delFile)

lngHnd& = <ObjName>.DeleteFile("C:\autoexec.bat")

Exists: This function will take the specified file and check if it exists in the specified directory or drive. This function uses the Windows 95 search algorithm to locate the file. This function returns a boolean value. True if found False if not.
<ObjName>.Exists(srchFile)

If <ObjName>.Exists("C:\boot.txt") = True Then Found = True

FileInfo: This function has many useful features by passing on a text ID or a Constant value defined in your module in the parameters. FileInfo has four different internal functions to choose from, CompanyName, FileDescription, FileVersion and LegalCopyright. The names of the parameter constants exsplain them selves. This function returns a string.
<ObjName>.FileInfo(InfoState,File)

Constants
HS0s_COMPNAME = "CompanyName"
HS0s_VERNUM = "FileVersion"
HS0s_FDISC = "FileDescription"
HS0s_COPYRIGHT = "LegalCopyright"

CompInfo$ = <ObjName>.FileInfo(HS0s_COMPNAME,"C:\vb.exe")
**This would return Microsoft

FileToBytes: This function returns the byte size of a given file in two different formats. Long format or Short format. This function also has constant parameters to use,
HS0x_LNGFORMAT or &020:=LongFormat
HS0x_SHRTFORMAT or &021:=ShortFormat
This function returns as a variant.
<ObjName>.FileToBytes(FormatState,File)

vSize = <ObjName>.FileToBytes(HS0x_SHRTFORMAT,"C:\win386.com")

FormatPathAs: This function reads and formats a path to either a long path name (Win95) or a short path name (DOS, Win 3.xx). You can either enter a short path and it will return the long path or file name or vis versa. This function has two parameter constants,
HS0x_LONGPATH = &O30
HS0x_SHRTPATH = &O31
to determin which process to proceed. This function returns a string value.
<ObjName>.FormatPathAs(FormatState,PathName)

<ObjName>.FormatPathAs(HS0x_LONGPATH,"C:\window~1\HomeSh~1.txt")

***Would return something like this:C:\windows 95NT\HomeShoping.txt

GetFileType: This function returns the kind of file the specified file is. This function can only be used under win95. This function returns a string value.

<ObjName>.GetFileType(File)

StrKind$=<ObjName>.GetFileType("C:\File1.txt")
***Could return 'Text Document'

MoveFile: This function uses the true Windows 95 move procedure, and moves a file from a source destination to a target destination. It returns a long value; hFile handle.

<ObjName>.MoveFile(source,target)

lngHnd& =<ObjName>.MoveFile("C:\autoexec.bat","D:\backup\autoexec.bat")

RenameFile: This function uses the true Windows 95 renaming procedure, and renames a file to the new name specified. It returns a long value; hFile handle.

<ObjName>.RenameFile(OldName,NewName)

lngHnd& = <ObjName>.RenameFile("C:\fun.zop","C:\fun.zoo")

SendToRecycle: This function takes a file or file(s) and does not delete them but uses the new Recycle process that comes with Win95. You can pass one file threw or many like 'c:*.*' or 'C:*.bak' threw the function to have it work. This function returns a unidentified value (long value).

<ObjName>.SendToRecycle(Param array)

<ObjName>.SendToRecycle("C:\windows\system*.oca")

StripFileOf: This function has many options to strip up a file name. This function has four parameters to do each task.

HS0x_FNAME or &O12: Strips away the filename "C:\win\"
HS0x_PATH or &O13: Strips away the path "text1.text"
HS0x_EXT or &O11: Strips away the extension "text1"
HS0x_SFNAME or &O14: Strips away just the name of the file ".text"
This function returns a string value
<ObjName>.StripFileOf(state,file)

Ext$=<ObjName>.StripFileOf(HS0x_SFNAME,"C:\windows\system.ini")
***Returns 'ini'

Truncate: This function takes a file path with root and truncates it to the users length in choice. It cuts up the path in sections depending on the "\" directory slash. If a certain directory is over the max then it is cut off at the end with the character specified by the user.

<ObjName>.Truncate(intLngth, strInhrtParm, strPathName)

intLngth:=Is the trigger length to truncate. If the input to the functions file path is over this amount, the Truncate procedure is processed.

strInhrtParm:=This holds the truncation character to be replaced for a directory that is truncated.
EX: C:\Windows\...\ {...} is the strInhrtParm

strPathName:=Is the string value of the file path.

<ObjName>.Truncate(40, "...", "C:\windows 95\setup program\TheDispresent.com")

These are all the functions in the version 1.0 File Utilities DLL for Visual Basic 4/5 32bit. All code and was written by Richard King of HAVOC Software. **If you have any suggestions or any ideas or bug reports contact me Richard King at:**

havocsoftware@technologist.com
DalNet #HAVOC1997