

Re: FOF_NO_UI Constant

Source: <http://www.tech-archive.net/Archive/VB/microsoft.public.vb.winapi/2008-03/msg00074.html>

- *From:* "GeoffG" <geoffcg@xxxxxxxxxxxxx>
 - *Date:* Sat, 15 Mar 2008 09:46:18 -0000
-

If anyone can offer further enlightenment on the question of passing VB strings to the pFrom and pTo members of the SHFILEOPSTRUCT structure, I'd be grateful.

One of my textbooks ("Win32 API Programming with Visual Basic" by Dr Steven Roman, page 458), appears to take no action to terminate the strings with two Null characters, yet as mentioned in the post below, the call to SHFileOperation works.

Any thoughts please?

Regards
Geoff

"GeoffG" <geoffcg@xxxxxxxxxxxxx> wrote in message
news:eJo%23ZSPHIA.2268@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Mike:

Many thanks.

Not being familiar with C, I thought the puzzling code line was probably a bit mask constructed from the other values. Thanks for confirming.

I had wondered what value I should give the FOF_NO_UI constant when declaring it in VBA and it seems 1556 (&H614 or 0x0614) is the value, as in:

```
Private Const FOF_NO_UI = &H614
```

However, I've now chosen not to hard-code the value as explained below. My remaining questions are: is this approach OK and are the strings OK.

I've used two Enums (FileOperations and FileOperationFlags) when declaring the wFunc and fFlags members of the SHFILEOPSTRUCT structure (as shown below). The FileOperationFlags Enum uses its own constants to declare the FOF_NO_UI constant (instead of hard-coding it to &H614). (I think it's neat that you can do this.)

Re: FOF_NO_UI Constant

The Enums work well when giving values to the wFunc and fFlags members, as the choices appear in an intellisense list. (I assume the structure declaration is OK as Enums use long integers and the members would otherwise be declared as longs.)

The other interesting fact is that VB seems automatically to take care of terminating the strings with two Null characters. Is VB doing this when translating from Unicode to ANSI? Or am I just being lucky and sooner or later it'll blow up?

Incidentally, below, I've copied the notes on each member from the Microsoft web site (as I need reminders when I revisit this).

```
' Declare Win32 API function to copy, move,  
' delete, or rename files:  
Private Declare Function SHFileOperation Lib "Shell32.dll" _  
Alias "SHFileOperationA" (lpFileOp As SHFILEOPSTRUCT) As Long
```

```
' Declare the structure to be used with API function:  
Private Type SHFILEOPSTRUCT
```

```
' A window handle to the dialog box to display  
' information about the status of the file operation:  
hwnd As Long
```

```
' A value that indicates which operation to perform;  
' one of the values in the FileOperations Enum:  
wFunc As FileOperations
```

```
' Note: This string must be double-null terminated.  
' A pointer to one or more source file names.  
' These names should be fully-qualified paths to prevent  
' unexpected results. Standard Microsoft MS-DOS wildcard  
' characters, such as "*", are permitted only in the  
' file-name position. Using a wildcard character  
' elsewhere in the string will lead to unpredictable results.  
' Although this member is declared as a single  
' null-terminated string, it is actually a buffer that can  
' hold multiple null-delimited file names. Each file name  
' is terminated by a single NULL character. The last file  
' name is terminated with a double NULL character ("\\0\\0")  
' to indicate the end of the buffer:  
pFrom As String
```

```
' Note: This string must be double-null terminated.  
' A pointer to the destination file or directory name.  
' This parameter must be set to NULL if it is not used.  
' Wildcard characters are not allowed. Their use will
```

Re: FOF_NO_UI Constant

' lead to unpredictable results.
' Like pFrom, the pTo member is also a double-null
' terminated string and is handled in much the same way.
' However, pTo must meet the following specifications:
' – Wildcard characters are not supported.
' – Copy and Move operations can specify destination
' directories that do not exist. In those cases,
' the system attempts to create them and normally
' displays a dialog box to ask the user if they
' want to create the new directory. To suppress
' this dialog box and have the directories created
' silently, set the FOF_NOCONFIRMMKDIR flag in fFlags.
' – For Copy and Move operations, the buffer can contain
' multiple destination file names if the fFlags
' member specifies FOF_MULTIDESTFILES.
' – Pack multiple names into the pTo string in the same
' way as for pFrom.
' – Use fully-qualified paths. Using relative paths is
' not prohibited, but can have unpredictable results.
pTo As String

' Flags that control the file operation;
' the constants that can be used in combination
' in this member are enumerated in the
' FileOperationFlags enum:
fFlags As FileOperationFlags

' When the function returns, this member contains
' TRUE if any file operations were aborted before
' they were completed; otherwise, FALSE.
' An operation can be manually aborted by the user
' through UI or it can be silently aborted by the
' system if the FOF_NOERRORUI or FOF_NOCONFIRMATION
' flags were set:
fAnyOperationsAborted As Long

' When the function returns, this member contains a
' handle to a name mapping object that contains the
' old and new names of the renamed files. This member
' is used only if the fFlags member includes the
' FOF_WANTMAPPINGHANDLE flag. See Remarks for more details:
hNameMappings As Long

' A pointer to the title of a progress dialog box.
' This is a null-terminated string. This member is
' used only if fFlags includes the FOF_SIMPLEPROGRESS
' flag:
lpszProgressTitle As String

End Type

' Declare enumeration of FILE OPERATION constants,
' for use with the wFunc member of the structure:
Private Enum FileOperations

' Move the files specified in pFrom to the
' location specified in pTo:
FO_MOVE = &H1

' Copy the files specified in the pFrom member
' to the location specified in the pTo member:
FO_COPY = &H2

' Delete the files specified in pFrom:
FO_DELETE = &H3

' Rename the file specified in pFrom.
' You cannot use this flag to rename multiple
' files with a single function call. Use
' FO_MOVE instead:
FO_RENAME = &H4

End Enum

' Declare enumeration of FILE OPERATION FLAG constants,
' for use with the fFlags member of the structure;
' flags can be used in combination using OR:
Private Enum FileOperationFlags

' The pTo member specifies multiple destination files
' (one for each source file in pFrom) rather than one
' directory where all source files are to be deposited:
FOF_MULTIDESTFILES = &H1

' Not used:
FOF_CONFIRM_MOUSE = &H2

' Do not display a progress dialog box:
FOF_SILENT = &H4

' Give the file being operated on a new name
' in a move, copy, or rename operation if a file
' with the target name already exists at the destination:
FOF_RENAMEONCOLLISION = &H8

' Respond with Yes to All for any dialog box that is
' displayed:
FOF_NOCONFIRMATION = &H10

Re: FOF_NO_UI Constant

' If FOF_RENAMEONCOLLISION is specified and any files
' were renamed, assign a name mapping object that
' contains their old and new names to the
' hNameMappings member. This object must be freed
' using SHFreeNameMappings when it is no longer needed:
FOF_WANTMAPPINGHANDLE = &H20

' Preserve undo information, if possible. Operations
' can be undone only from the same process that
' performed the original operation. If, despite
' earlier warnings against doing so, pFrom does not
' contain fully-qualified path and file names,
' this flag is ignored:
FOF_ALLOWUNDO = &H40

' Perform the operation only on files (not on folders)
' if a wildcard file name (*.*) is specified:
FOF_FILESONLY = &H80

' Display a progress dialog box but do not show
' individual file names as they are operated on:
FOF_SIMPLEPROGRESS = &H100

' Do not ask the user to confirm the creation of a new
' directory if the operation requires one to be created:
FOF_NOCONFIRMMKDIR = &H200

' Do not display a dialog to the user if an error occurs:
FOF_NOERRORUI = &H400

' Only perform the operation in the local directory.
' Don't operate recursively into subdirectories,
' which is the default behavior:
FOF_NORECURSION = &H1000

' Version 4.71. Do not copy the security attributes
' of the file. The destination file receives the
' security attributes of its new folder:
FOF_NOCOPYSECURITYATTRIBS = &H800

' Version 5.0. Do not move connected files as a group.
' Only move the specified files:
FOF_NO_CONNECTED_ELEMENTS = &H2000

' Version 5.0. Send a warning if a file is being
' permanently destroyed during a delete operation
' rather than recycled. This flag partially overrides
' FOF_NOCONFIRMATION:
FOF_WANTNUKEWARNING = &H4000

Re: FOF_NO_UI Constant

```
' Not used:
FOF_NORECURSEREPARSE = &H8000&

' Version 6.0.6060 (Windows Vista).
' Perform the operation silently, presenting no user
' interface (UI) to the user. This is equivalent to
' FOF_SILENT | FOF_NOCONFIRMATION | FOF_NOERRORUI |
' FOF_NOCONFIRMMKDIR.
FOF_NO_UI = FOF_SILENT Or FOF_NOCONFIRMATION Or FOF_NOERRORUI Or
FOF_NOCONFIRMMKDIR
```

End Enum

Thanks for any further assistance.
Regards
Geoff

"MikeD" <nobody@xxxxxxxxxxxx> wrote in message
[news:O7B\\$e0KhIHA.6032@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:O7B$e0KhIHA.6032@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)

"GeoffG" <geoffcg@xxxxxxxxxxxx> wrote in message
news:%23nTdGrKhIHA.1184@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Karl:
Very many thanks.
The prospect of downloading up to 1000+ Mb on a flakey
broadband
connection didn't fill me with joy, especially as I'm really
only
wanting an updated version of the WIN32API.TXT file. So,
I'm grateful
to you.

Does the following line:

```
#define FOF_NO_UI (FOF_SILENT |
FOF_NOCONFIRMATION
| FOF_NOERRORUI |
FOF_NOCONFIRMMKDIR) // don't
display any UI at all
```

mean that the value of FOF_NO_UI is 0x0614? That is, the
sum of:

Re: FOF_NO_UI Constant

FOF_SILENT (0x0004) + FOF_NOCONFIRMATION
(0x0010) + FOF_NOERRORUI
(0x0400) + FOF_NOCONFIRMMKDIR (0x0200)

Not exactly. Those are bitmasks, so you OR them rather than ADD them.

--

Mike
Microsoft MVP Visual Basic