GetLongPath_Legacy Function: A GetLongPathName Equivalent For Older Systems

Dave Liske, September 1999

One of the most useful functions in the Windows API is GetShortPathName. This function accepts a long path name and converts it into its 8.3 equivalent. However, the complement to this, GetLongPathName, has been non-existent for developers. The API call does exist now in kernel32.dll for Windows 98 and Windows 2000 (see the latest Platform SDK), but there's still a need for Visual Basic developers targeting Windows 95 and NT4 systems.

I discovered a need for this function when attempting to retrieve the shdocvw.dll path from the registry to find out which version of Internet Explorer was installed. The path was in the short path name format, and FindFirstFile, which I called to ensure the file was where it was supposed to be before calling GetFileVersionInfo, doesn't like short path names. I had to have a way to accurately convert this to the long path name.

Looking through MSDN, I found a description of how one programmer had parsed the short path name into its component parts, then sent each part to FindFirstFile. The cFileName component of the returned WIN32_FIND_DATA struct subsequently contains the long path name equivalent as found on the hard drive. The individual resulting long path name components are then concatenated into the complete long path name. This is exactly how the following code works ... but in Visual basic instead of Visual C++.

Using this method, developers don't need to test whether or not the original path is short or long. If a valid long path name is passed to GetLongPath_Legacy, the same path is returned.

Also, if the operating system is Windows 98 or 2000, the actual GetLongPathName API call is used instead. This results in a slightly shorter processing time. One stickler here is the fact that the cFileName component of the returned WIN32_FIND_DATA struct is returned with a large number of trailing nulls. An additional procedure, StripNulls, takes care of this problem. Also dealt with is whether or not the supplied path has quotes around it on its own, as is the case with the registered Iexplore.exe path.

First, we have a bunch of declarations to declare.

```
' Constants for GetLongPath_Legacy
Private Const SINGLE_QUOTE = """"
Private Const INVALID_HANDLE_VALUE = -1

' Constants for determining OS
Private Const VER_PLATFORM_WIN32s = 0
Private Const VER_PLATFORM_WIN32_WINDOWS = 1
Private Const VER_PLATFORM_WIN32_NT = 2

Private Const UNKNOWN_OS = 0
Private Const WINDOWS_NT_3_51 = 1
Private Const WINDOWS_95 = 2
Private Const WINDOWS_NT_4 = 3
Private Const WINDOWS_98 = 4
Private Const WINDOWS_2000 = 5
```

```
' UDT for determining OS
Private Type OSVERSIONINFO
  dwOSVersionInfoSize As Long
  dwMajorVersion As Long
  dwMinorVersion As Long
  dwBuildNumber As Long
  dwPlatformId As Long
  szCSDVersion As String * 128
End Type

Private Type WIN32_FIND_DATA
  dwFileAttributes As Long
  ftCreationTime As FILETIME
  ftLastAccessTime As FILETIME
  ftLastWriteTime As FILETIME
  nFileSizeHigh As Long
  nFileSizeLow As Long
  dwReserved0 As Long
  dwReserved1 As Long
  cFileName As String * MAX_PATH
  cAlternate As String * 14
End Type

' Call to translate short file path to long file path
' (Win98 and Win2k and above only - see comments for the
' GetLongFilePath_Legacy procedure)
Private Declare Function GetLongPathName Lib "kernel32" _
    (ByRef pszShortPath As String, _
    ByRef lpszLongPath As String, _
    ByVal cchBuffer As Long) As Long
```

The GetLongPath_Legacy function is set up to return just the resulting long path name.

```
Private Function GetLongPath_Legacy(ByVal strShortName As String) As String

  On Error GoTo ErrHandler

  Dim strLongName As String
  Dim lngResult As Long
  Dim strTempLongName As String
  Dim strTemp As String
  Dim strPathTemp As String
  Dim intLength As Integer
  Dim intPosition As Integer
  Dim intStart As Integer
  Dim lngHandle As Long
  Dim lpFindFileData As WIN32_FIND_DATA
```

First, if it's already a long file name, will skip the whole thing.

```
  ' If it's already a long name, don't worry about it.
  ' The GetLongPathName API call will hiccup if we don't
  ' do this.
  If InStr(strShortName, "~") = 0 Then
    GetLongPath_Legacy = strShortName
    Exit Function
  End If
```

We'll then check which operating system we're running on, and if we can do it, we'll use the GetLongPathName API call and exit. This is a rather simple API call, and we can use it with no trouble.

```
  If GetWindowsVersion >= WINDOWS_98 Then
    ' For Windows 98 and later, and Windows 2000
    ' and later, use the following API call:
```

```
  Call GetLongPathName(strShortName, _
      strLongName, 256)
  GetLongPath_Legacy = strLongName
  Exit Function
End If
```

We'll get rid of any quotation marks, which will likely be there if the path name came from the registry. Then, we'll ensure there's a backslash on the name so the InStr function won't fail.

```
GetLongPath_Legacy = 0

' If stored in the registry, in some cases it's
' enclosed in double-quotes, so we need to delete them
If Left$(strShortName, 1) = SINGLE_QUOTE Then _
    strShortName = Right$(strShortName, Len(strShortName) - 1)
If Right$(strShortName, 1) = SINGLE_QUOTE Then _
    strShortName = Left$(strShortName, Len(strShortName) - 1)

' Add \ to short name to prevent Instr from failing
If Right$(strShortName, 1) <> "\" Then _
    strShortName = Left$(strShortName, Len(strShortName) & "\")
```

Next, we'll snag just the drive letter and the colon for later use. After that, we'll get rid of it on our working string and locate the first backslash in this string.

```
' Save the drive letter for later
strLongName = Left(strShortName, 2)

' Strip the drive letter off the temporary string
strPathTemp = Right(strShortName, Len(strShortName) - 3)

' Find the first backslash
intPosition = InStr(strPathTemp, "\")
```

We can now pick the path name apart piece-by-piece, verify the individual components of it, and build the long file path from the returned paths. This isn't a complicated procedure, as it get's the individual component of the path name, tacks it to what's already built, verifies it, builds the long path as it's verified, then loops till it runs out of stuff to look at.

```
Do While intPosition <> 0

  ' Get the individual component of the path name
  strTemp = Left(strPathTemp, intPosition - 1)

  ' Translate the short path component into its
  ' actual long path component as found by FindFirstFile
  lngHandle = FindFirstFile(strLongName & "\" & strTemp, lpFindFileData)

  If (lngHandle) = INVALID_HANDLE_VALUE Then

    ' The folder or file does not exist
    Exit Function

  End If

  ' Get rid of any null characters retrieved if
  ' from the registry
  strTempLongName = StripNulls(lpFindFileData.cFileName)

  ' Build the long path name, starting with the
  ' previously-saved drive letter
  strLongName = strLongName & "\" & strTempLongName

  ' Delete the short path component we just used
```

```
    strPathTemp = Right(strPathTemp, Len(strPathTemp) - (Len(strTemp) + 1))

    ' Find the next backslash
    intPosition = InStr(strPathTemp, "\")

  Loop
```

Once the path is built, we can tack the file name itself onto it and spit the whole thing out.

```
  ' Add the remainder, which is the name of the file
  GetLongPath_Legacy = strLongName & "\" & strPathTemp

  Exit Function
```

Sometimes, an error will occur if an empty string is passed to this function. We'll take care of this in the error handler.

```
ErrHandler:
  Select Case Err.Number
  Case 52
    strLongName = ""
    Exit Function
  Case Else
    Resume Next
  End Select

End Function
```

The StripNulls function gets rid of any trailing nulls that might be present when dealing with paths retrieved from the registry. This is done in the loop to catch any possible problems.

```
Private Function StripNulls(OriginalStr As String) As String

' Strips any trailing nulls from path names retrieved
' from the registry. This function is found in the
' following Microsoft(r) knowledge base articles:
' Q183009 "HOWTO: Enumerate Windows Using the WIN32 API"
' Q185476 "HOWTO: Search Directories to Find or List Files"
' Q190218 "HOWTO: Retrieve Settings From a Printer Driver"

  If (InStr(OriginalStr, Chr(0)) > 0) Then
    OriginalStr = Left(OriginalStr, _
        InStr(OriginalStr, Chr(0)) - 1)
  End If

  StripNulls = OriginalStr

End Function
```

Finally, the GetWindowsVersion function let's us know when we can use the GetLongPathName API call.

```
Private Function GetWindowsVersion() As Long

  Dim osinfo As OSVERSIONINFO
  Dim retvalue As Integer

  osinfo.dwOSVersionInfoSize = 148
  osinfo.szCSDVersion = Space$(128)
  retvalue = GetVersionExA(osinfo)

  With osinfo
    Select Case .dwPlatformId
```

```
    Case VER_PLATFORM_WIN32_WINDOWS
      If .dwMinorVersion = 0 Then
        GetWindowsVersion = WINDOWS_95
      ElseIf .dwMinorVersion = 10 Then
        GetWindowsVersion = WINDOWS_98
      End If
    Case VER_PLATFORM_WIN32_NT
      If .dwMajorVersion = 3 Then
        GetWindowsVersion = WINDOWS_NT_3_51
      ElseIf .dwMajorVersion = 4 Then
        GetWindowsVersion = WINDOWS_NT_4
      ElseIf .dwMajorVersion = 5 Then
        GetWindowsVersion = WINDOWS_2000
      End If
    Case Else
      GetWindowsVersion = UNKNOWN_OS
    End Select
  End With

End Function
```