

Disabling Flash Right-Click Context Menu

To be quite frank, after 3 months of ASP.NET development, the opportunity to do some down and dirty API programming really appealed to me today. So, the question (as indicated by the title) involved a member wanting to disable the right-click context menu from the Flash ActiveX component when being used in a WinForms application. (See previous blog entries for examples of Flash in VB.NET and communication between the two.

Add items to Windows Explorer Shell context (right-click) menu easily – How to add them ?

Add entries to Windows Explorer Shell context menu easily with Windows Explorer Shell Context Menu. This powerful .Net component for custom items appending to Windows Explorer Shell context menu will add all your custom items to Explorer Shell context menu. It , C++ and VB.NET support include detailed C# and VB.NET samples, tutorials and support all you may need :

- Add all your items to Windows Explorer Shell context menu to be shown on any Windows operating system (all operating systems are supported – Windows XP, Vista, x64 , etc.)
- Add all your items to Windows Explorer Shell context menu to be shown in any way - with custom caption and your custom icon, as separator or sub-menu
- Add items to Windows Explorer Shell context menu to be shown for all types of files or shown only for files of particular type (for example, only for .PDF .TXT , .MP3,.WMA,.AAC , .AVI media files)
- Add items to Explorer Shell context menu, sub-menus, sub-menus of unlimited depth and even more

Explorer Shell Context Menu - is a .Net component that support all you may need to add all your items to the Explorer Shell context menu - in a fast and a very easy way. Add your application entries to Explorer context menu right now – add items to context menu fast , easy and exactly as you prefer :

The simple answer is to capture the right-click of the Flash Window itself and prevent it from being processed. A little play with the special software quickly came up with the answers and the rest was all about good old fashioned subclassing. Because this method works only for Windows 95 / Windows 98 (not on XP, Vista, x64 - 64-bit Windows), to add items to Windows Explorer Shell context menu you should use, according to Microsoft guidelines, appropriate .Net component - Windows Explorer Shell Context Menu. I must admit it pinched the API declarations and smartened them up a tad and that is pretty much it.

For the record, this is something I've been meaning to do for a while + I can leave work not feeling like some CSS floozy :)

Note: to recreate, create one form, one button and add one Flash Active X control. Copy and Paste as necessary. Tested in VS2005. If you get the dreaded 'Failed to import the ActiveX control' message, check out this workaround

```
Imports System.Runtime.InteropServices
```

```
Public Class Form1
```

```
#Region "API Routines"
```

```

    <DllImport("User32.dll")> _
    Private Shared Function EnumChildWindows _
        (ByVal WindowHandle As IntPtr, ByVal Callback As EnumWindowProcess, _
        ByVal lParam As IntPtr) As Boolean
    End Function

    <DllImport("user32.dll", CharSet:=CharSet.Auto)> _
    Private Shared Sub GetClassName(ByVal hWnd As System.IntPtr, _
        ByVal lpClassName As System.Text.StringBuilder, ByVal nMaxCount As Integer)
    End Sub

    <DllImport("user32.dll", CharSet:=CharSet.Auto)> _

```

```
Private Overloads Shared Function SetWindowLong(ByVal hWnd As IntPtr, ByVal
nIndex As Integer, ByVal dwNewLong As Integer) As Integer
End Function
```

```
<DllImport("user32.dll", CharSet:=CharSet.Auto)> _
Private Overloads Shared Function SetWindowLong(ByVal hWnd As IntPtr, ByVal
nIndex As Integer, ByVal dwNewLong As FlashCaptureRoutine) As Integer
End Function
```

```
<DllImport("user32.dll", CharSet:=CharSet.Auto)> _
Private Shared Function CallWindowProc(ByVal lpPrevWndFunc As Integer, ByVal
hWnd As IntPtr, ByVal Msg As Integer, ByVal wParam As Integer, ByVal lParam As Integer) As Integer
End Function
```

```
#End Region
```

```
#Region "Delegates"
```

```
' Used for processing Flash Window messages
Public Delegate Function FlashCaptureRoutine(ByVal hwnd As Integer, ByVal
Msg As Integer, ByVal wParam As Integer, ByVal lParam As Integer) As Integer
```

```
' Used for enumerating child windows
Public Delegate Function EnumWindowProcess(ByVal Handle As IntPtr, ByVal Parameter As IntPtr) As Boolean
```

```
#End Region
```

```
' Constants
Public Const GWL_WNDPROC As Integer = (-4)
Public Const WM_ENTERMENULOOP As Integer = &H211
Public Const WM_RBUTTONDOWN As Integer = &H204
Public Const WM_INITMENU = &H116
```

```
' Private vars
Private mFlashWindowHandle As IntPtr
Private mPreviousHandle As Integer
Private isCapturing As Boolean
```

```
Private Sub Form1_FormClosing(ByVal sender As Object, ByVal
e As System.Windows.Forms.FormClosingEventArgs) Handles Me.FormClosing
' Cleanup if appropriate
If isCapturing Then CleanupCapture()
End Sub
```

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles MyBase.Load
' Load the test movie...
Me.AxShockwaveFlash1.Movie = "MyTestMovie.swf"
Me.AxShockwaveFlash1.Menu = True
End Sub
```

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button1.Click
' Capture / Reset Flash Window Processing as appropriate

' React as appropriate
If isCapturing Then
' Resume
CleanupCapture()
Else
' Capture events
CaptureRightClick()
End If
' Switch over the flag
isCapturing = Not isCapturing
```

```
End Sub

Private Sub CaptureRightClick()
    ' Attempt to obtain the Flash Window
    EnumChildWindows(Me.Handle,
AddressOf EnumWindow, IntPtr.Zero)
    ' Subclass using new routine storing the old result
    mPreviousHandle = SetWindowLong(mFlashWindowHandle, GWL_WNDPROC, AddressOf FlashWindowCapture)
End Sub

Private Sub CleanupCapture()
    ' Reset back to the original handle...
    SetWindowLong(mFlashWindowHandle, GWL_WNDPROC, mPreviousHandle)
End Sub

Private Function FlashWindowCapture(ByVal hwnd As Integer, ByVal Msg As Integer,
ByVal wParam As Integer, ByVal lParam As Integer) As Integer

    ' Capture the appropriate message and prevent it from being processed
    Select Case Msg
        Case WM_RBUTTONDOWN
            ' Exit
            Exit Function
    End Select

    ' Carry on...
    Return CallWindowProc(mPreviousHandle, hwnd, Msg, wParam, lParam)
End Function

Private Function EnumWindow(ByVal Handle As IntPtr, ByVal Parameter As IntPtr) As Boolean
    ' Obtain the class name of the child window
    Dim ClassName As New System.Text.StringBuilder("", 255)
    GetClassName(Handle, ClassName, ClassName.MaxCapacity)

    ' Is this the Flash Window?
    If ClassName.ToString = "MacromediaFlashPlayerActiveX" Then
        mFlashWindowHandle = Handle
    End If
    Return True
End Function

End Class
```