

Forum:[Hack Forums](#)

Thread: [Webcam \[without clipboard\]](#)

KingDeath360 - February 1, 2015, 5:49 am

I had this on my hdd a while, it may come in useful

Code Option Explicit On

Option Strict On

Public Class iCam

#Region "Api/constants"

Private Const WS_CHILD As Integer = &H4000000

Private Const WS_VISIBLE As Integer = &H1000000

Private Const SWP_NOMOVE As Short = &H2S

Private Const SWP_NOZORDER As Short = &H4S

Private Const WM_USER As Short = &H400S

Private Const WM_CAP_DRIVER_CONNECT As Integer = WM_USER + 10

Private Const WM_CAP_DRIVER_DISCONNECT As Integer = WM_USER + 11

Private Const WM_CAP_SET_VIDEOFORMAT As Integer = WM_USER + 45

Private Const WM_CAP_SET_PREVIEW As Integer = WM_USER + 50

Private Const WM_CAP_SET_PREVIEWRATE As Integer = WM_USER + 52

Private Const WM_CAP_GET_FRAME As Long = 1084

Private Const WM_CAP_COPY As Long = 1054

Private Const WM_CAP_START As Long = WM_USER

Private Const WM_CAP_STOP As Long = (WM_CAP_START + 68)

Private Const WM_CAP_SEQUENCE As Long = (WM_CAP_START + 62)

Private Const WM_CAP_SET_SEQUENCE_SETUP As Long = (WM_CAP_START + 64)

Private Const WM_CAP_FILE_SET_CAPTURE_FILEA As Long = (WM_CAP_START + 20)

Private Declare Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal hwnd As Integer, ByVal wParam As Integer, ByVal wParam As Short, ByVal lParam As String) As Integer

Private Declare Function capCreateCaptureWindowA Lib "avicap32.dll" (ByVal lpszWindowName As String, ByVal dwStyle As Integer, ByVal x As Integer, ByVal y As Integer, ByVal nWidth As Integer, ByVal nHeight As Short, ByVal hWndParent As Integer, ByVal nID As Integer) As Integer

Private Declare Function capGetDriverDescriptionA Lib "avicap32.dll" (ByVal wDriver As Short, ByVal lpszName As String, ByVal cbName As Integer, ByVal lpszVer As String, ByVal cbVer As Integer) As Boolean

Private Declare Function BitBlt Lib "GDI32.DLL" (ByVal hdcDest As IntPtr, ByVal nXDest As Integer, ByVal nYDest As Integer, ByVal nWidth As Integer, ByVal nHeight As Integer, ByVal hdcSrc As IntPtr, ByVal nXSrc As Integer, ByVal nYSrc As Integer, ByVal dwRop As Int32) As Boolean

#End Region

Private iDevice As String

Private hHwnd As Integer

Private lwndC As Integer

Public iRunning As Boolean

```
Private CamFrameRate As Integer = 15
Private OutputHeight As Integer = 350
Private OutputWidth As Integer = 550
```

```
Public Sub resetCam()
    'resets the camera after setting change
    If iRunning Then
        closeCam()
        Application.DoEvents()

        If setCam() = False Then
            MessageBox.Show("Error Setting/Re-Setting Camera")
        End If
    End If
End Sub
```

```
Public Sub New(ByVal parentH As Integer)
    'Gets the handle and initiates camera setup
    If Me.iRunning = True Then
        MessageBox.Show("Camera Is Already Running")
    Exit Sub
    Else
```

```
        hHwnd = capCreateCaptureWindowA(iDevice, WS_VISIBLE Or WS_CHILD, 0, 0, OutputWidth,
        CShort(OutputHeight), parentH, 0)
```

```
        SendMessage(hHwnd, WM_CAP_DRIVER_CONNECT, CShort(iDevice), CType(0, String))
        Do While setCam() = False
            Threading.Thread.Sleep(1)
        Loop
```

```
    End If
End Sub
```

```
Public Sub setFrameRate(ByVal iRate As Long)
    'sets the frame rate of the camera
    CamFrameRate = CInt(1000 / iRate)

    resetCam()
End Sub
```

```
Private Function setCam() As Boolean
    'Sets all the camera up
    If SendMessage(hHwnd, WM_CAP_DRIVER_CONNECT, CShort(iDevice), CType(0, String)) =
1 Then
        SendMessage(hHwnd, WM_CAP_SET_PREVIEWRATE, CShort(CamFrameRate), CType(0,
String))
        SendMessage(hHwnd, WM_CAP_SET_PREVIEW, 1, CType(0, String))
        Me.iRunning = True
        Return True
    Else
```

```
Me.iRunning = False
Return False
End If
End Function
```

```
Public Function closeCam() As Boolean
'Closes the camera
If Me.iRunning Then
closeCam = CBool(SendMessage(hHwnd, WM_CAP_DRIVER_DISCONNECT, 0, CType(0,
String)))
Me.iRunning = False
End If
End Function
```

```
Public Function copyFrame(ByVal src As PictureBox, ByVal rect As RectangleF) As Bitmap
If iRunning Then
Dim srcPic As Graphics = src.CreateGraphics
Dim srcBmp As New Bitmap(src.Width, src.Height, srcPic)
Dim srcMem As Graphics = Graphics.FromImage(srcBmp)

Dim HDC1 As IntPtr = srcPic.GetHdc
Dim HDC2 As IntPtr = srcMem.GetHdc
```

```
BitBlt(HDC2, 0, 0, CInt(rect.Width), _
CInt(rect.Height), HDC1, CInt(rect.X), CInt(rect.Y), 13369376)
```

```
copyFrame = CType(srcBmp.Clone(), Bitmap)
```

```
'Clean Up
srcPic.ReleaseHdc(HDC1)
srcMem.ReleaseHdc(HDC2)
srcPic.Dispose()
srcMem.Dispose()
Else
MessageBox.Show("Camera Is Not Running!")
End If
End Function
```

```
Public Function FPS() As Integer
Return CInt(1000 / (CamFrameRate))
End Function
```

```
End Class
Code Dim dfdsf As New iCam(Me.Handle.ToInt64)
dfdsf.setFrameRate(24)
```

```
Spoiler \(Click to View\)Code Imports System
Imports System.IO
Imports System.Runtime.InteropServices
Imports Microsoft.Win32
```

```
Public Class Form1
Inherits System.Windows.Forms.Form
```

```
Const WM_CAP_START = &H400S
```

```

Const WS_CHILD = &H4000000
Const WS_VISIBLE = &H1000000
Const WM_CAP_DRIVER_CONNECT = WM_CAP_START + 10
Const WM_CAP_DRIVER_DISCONNECT = WM_CAP_START + 11
Const WM_CAP_EDIT_COPY = WM_CAP_START + 30
Const WM_CAP_SEQUENCE = WM_CAP_START + 62
Const WM_CAP_FILE_SAVEAS = WM_CAP_START + 23
Const WM_CAP_SET_SCALE = WM_CAP_START + 53
Const WM_CAP_SET_PREVIEWRATE = WM_CAP_START + 52
Const WM_CAP_SET_PREVIEW = WM_CAP_START + 50
Const SWP_NOMOVE = &H2S
Const SWP_NOSIZE = 1
Const SWP_NOZORDER = &H4S
Const HWND_BOTTOM = 1
'--The capGetDriverDescription function retrieves the version
' description of the capture driver--
Declare Function capGetDriverDescriptionA Lib "avicap32.dll" _
    (ByVal wDriverIndex As Short, _
    ByVal lpszName As String, ByVal cbName As Integer, _
    ByVal lpszVer As String, _
    ByVal cbVer As Integer) As Boolean

'--The capCreateCaptureWindow function creates a capture window--
Declare Function capCreateCaptureWindowA Lib "avicap32.dll" _
    (ByVal lpszWindowName As String, ByVal dwStyle As Integer, _
    ByVal x As Integer, ByVal y As Integer, ByVal nWidth As Integer, _
    ByVal nHeight As Short, ByVal hWnd As Integer, _
    ByVal nID As Integer) As Integer

'--This function sends the specified message to a window or windows--
Declare Function SendMessage Lib "user32" Alias "SendMessageA" _
    (ByVal hwnd As Integer, ByVal Msg As Integer, _
    ByVal wParam As Integer, _
    <MarshalAs(UnmanagedType.AsAny)> ByVal lParam As Object) As Integer

'--Sets the position of the window relative to the screen buffer--
Declare Function SetWindowPos Lib "user32" Alias "SetWindowPos" _
    (ByVal hwnd As Integer, _
    ByVal hWndInsertAfter As Integer, ByVal x As Integer, _
    ByVal y As Integer, _
    ByVal cx As Integer, ByVal cy As Integer, _
    ByVal wFlags As Integer) As Integer

'--This function destroys the specified window--
Declare Function DestroyWindow Lib "user32" _
    (ByVal hwnd As Integer) As Boolean

'---used to identify the video source---
Dim CamSource As Integer

'---used as a window handle---
Dim hWnd As Integer

```

```

Private Sub cameraSource()
    Dim DriverName As String = Space(80)
    Dim DriverVersion As String = Space(80)
    For i As Integer = 0 To 9
        If capGetDriverDescriptionA(i, DriverName, 80, _
            DriverVersion, 80) Then
            ListBox1.Items.Add(DriverName.Trim)
        End If
    Next
End Sub

```

```

Private Sub previewCamera(ByVal pbCtrl As PictureBox)
    hWnd = capCreateCaptureWindowA(CamSource, _
        WS_VISIBLE Or WS_CHILD, 0, 0, 0, _
        0, pbCtrl.Handle.ToInt32, 0)
    Do While (SendMessage(hWnd, WM_CAP_DRIVER_CONNECT, CamSource, 0) = 0)
        Threading.Thread.Sleep(1)
    Loop

```

```

'---set the preview scale---
SendMessage(hWnd, WM_CAP_SET_SCALE, True, 0)
'---set the preview rate (ms)---
SendMessage(hWnd, WM_CAP_SET_PREVIEWRATE, 30, 0)
'---start previewing the image---
SendMessage(hWnd, WM_CAP_SET_PREVIEW, True, 0)
'---resize window to fit in PictureBox control---
SetWindowPos(hWnd, HWND_BOTTOM, 0, 0, _
    pbCtrl.Width, pbCtrl.Height, _
    SWP_NOMOVE Or SWP_NOZORDER)

```

```
End Sub
```

```

Private Sub stopPreviewCamera()
    SendMessage(hWnd, WM_CAP_DRIVER_DISCONNECT, CamSource, 0)
    DestroyWindow(hWnd)
End Sub

```

```

Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
    cameraSource()
    Button4.Enabled = False
    Button1.Enabled = False
    Button2.Enabled = False
    Button3.Enabled = False
End Sub

```

```

Private Sub ListBox1_DoubleClick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ListBox1.DoubleClick
    Label1.Visible = False
    previewCamera(PictureBox1)
    Button1.Enabled = True
    Button4.Enabled = False
    Button2.Enabled = True

```

End Sub

Private Sub ListBox1_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ListBox1.SelectedIndexChanged

CamSource = ListBox1.SelectedIndex

'---preview the selected video source

End Sub

'stop preview

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

stopPreviewCamera()

Button4.Enabled = True

Button1.Enabled = False

End Sub

' recording

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

Button3.Enabled = True

Button2.Enabled = False

SendMessage(hWnd, WM_CAP_SEQUENCE, 0, 0)

End Sub

' stop recording and ask to save video

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click

Dim save As Integer

save = MsgBox("Do you want to save your recording video", MsgBoxStyle.YesNo + MsgBoxStyle.Information, "Recording Video")

If (save = MsgBoxResult.Yes) Then

Dim saveName As New SaveFileDialog

saveName.Filter = "Avi file(*.avi)|*.avi"

If saveName.ShowDialog = Windows.Forms.DialogResult.OK Then

' SendMessage(hWnd, WM_CAP_FILE_SAVEAS, 0, "C:\RecordedVideo.avi")

SendMessage(hWnd, WM_CAP_FILE_SAVEAS, 0, saveName.FileName)

End If

End If

Me.Cursor = System.Windows.Forms.Cursors.Default

Button2.Enabled = True

Button3.Enabled = False

End Sub

' preview

Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click

CamSource = ListBox1.SelectedIndex

previewCamera(PictureBox1)

Button4.Enabled = False

Button1.Enabled = True

End Sub

End ClassCode Imports System.Runtime.InteropServices

Public Class Form1

Private m_Width As Integer = 320

Private m_Height As Integer = 240

Private mCapHwnd As Integer

Public Const WM_USER As Integer = 1024

Public Const WM_CAP_CONNECT As Integer = 1034

Public Const WM_CAP_DISCONNECT As Integer = 1035

Public Const WM_CAP_GET_FRAME As Integer = 1084

Public Const WM_CAP_COPY As Integer = 1054

Public Const WM_CAP_START As Integer = WM_USER

Public Const WM_CAP_DLG_VIDEOFORMAT As Integer = WM_CAP_START + 41

Public Const WM_CAP_DLG_VIDEOSOURCE As Integer = WM_CAP_START + 42

Public Const WM_CAP_DLG_VIDEODISPLAY As Integer = WM_CAP_START + 43

Public Const WM_CAP_GET_VIDEOFORMAT As Integer = WM_CAP_START + 44

Public Const WM_CAP_SET_VIDEOFORMAT As Integer = WM_CAP_START + 45

Public Const WM_CAP_DLG_VIDEOCOMPRESSION As Integer = WM_CAP_START + 46

Public Const WM_CAP_SET_PREVIEW As Integer = WM_CAP_START + 50

<DllImport("user32", EntryPoint:="SendMessage")> Public Shared Function SendMessage(hWnd As Integer, Msg As UInteger, wParam As Integer, lParam As Integer) As Integer
End Function

<DllImport("avicap32.dll", EntryPoint:="capCreateCaptureWindowA")> Public Shared Function
capCreateCaptureWindowA(lpszWindowName As String, dwStyle As Integer, X As Integer, Y As
Integer, nWidth As Integer, nHeight As Integer, _
hwndParent As Integer, nID As Integer) As Integer
End Function

Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
mCapHwnd = capCreateCaptureWindowA("WebCap", WS_VISIBLE | WS_CHILD, 0, 0,
pictureBox1.Width, pictureBox1.Height, pictureBox1.Handle.ToInt32(), 0);
Application.DoEvents()
Threading.Thread.Sleep(9900)
If SendMessage(mCapHwnd, WM_CAP_CONNECT, 0, 0) <> 0 Then
Call SendMessage(mCapHwnd, WM_CAP_CONNECT, 0, 0)
Application.DoEvents()
Call SendMessage(mCapHwnd, WM_CAP_SET_PREVIEW, 0, 0)
Else
Call SendMessage(mCapHwnd, WM_CAP_CONNECT, 0, 0)
Application.DoEvents()
Call SendMessage(mCapHwnd, WM_CAP_SET_PREVIEW, 0, 0)
MsgBox("No Camera Detected")
SendMessage(mCapHwnd, WM_CAP_GET_FRAME, 0, 0)
End If
End Sub
End Class

Not mine