



Sign in

xorrior / RandomPS-Scripts Public

Notifications

Fork 86

Star 315

[Code](#) [Issues 1](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#)

RandomPS-Scripts / Get-DXWebcamVideo.ps1



273 lines (217 loc) · 106 KB

Code Blame

Raw



```
1 function Get-DXAudioInput
2 {
3     <#
4     .SYNOPSIS
5     List audio input options for DirectX
6
7     Author: Justin Warner (@sixdub)
8     License: BSD 3-Clause
9
10    .DESCRIPTION
11    This function will list all audio input options for DirectX
12
13    #>
14    $filters = New-Object DirectX.Capture.Filters
15    if ($filters.AudioInputDevices -ne $null)
16    {
17        $filters.AudioInputDevices
18    }
19    else
20    {
21        Write-Verbose "[!] There are no audio inputs"
22    }
23 }
24
25 function Get-DXVideoInput
26 {
```

```
27     <#
28     .SYNOPSIS
29     List video input options for DirectX
30
31     Author: Justin Warner (@sixdub)
32     License: BSD 3-Clause
33
34     .DESCRIPTION
35     This function will list all video input options for DirectX
36
37     #>
38     $filters = New-Object DirectX.Capture.Filters
39     if ($filters.VideoInputDevices -ne $null)
40     {
41         $filters.VideoInputDevices
42     }
43     else
44     {
45         Write-Verbose "[!] There are no video inputs"
46     }
47 }
48
49 function Get-DXAudioCompression
50 {
51     <#
52     .SYNOPSIS
53     List audio compression options for DirectX
54
55     Author: Justin Warner (@sixdub)
56     License: BSD 3-Clause
57
58     .DESCRIPTION
59     This function will list all audio compression options for DirectX
60
61     #>
62     $filters = New-Object DirectX.Capture.Filters
63     if ($filters.AudioCompressors -ne $null)
64     {
65         $filters.AudioCompressors
66     }
67     else
68     {
69         Write-Verbose "[!] Audio compression not available"
70     }
71 }
72
```

```
73     function Get-DXVideoCompression
74     {
75         <#
76         .SYNOPSIS
77         List video compression options for DirectX
78
79         Author: Justin Warner (@sixdub)
80         License: BSD 3-Clause
81
82         .DESCRIPTION
83         This function will list all video compression options for DirectX
84
85         #>
86
87         $filters = New-Object DirectX.Capture.Filters
88         if ($filters.VideoCompressors -ne $null)
89         {
90             $filters.VideoCompressors
91         }
92         else
93         {
94             Write-Verbose "[!] Video Compression not available"
95         }
96     }
97
98     function Get-DXWebcamVideo
99     {
100         <#
101         .SYNOPSIS
102         This function utilizes the DirectX and DShowNET assemblies to record video from the host's webcam
103
104         Author: Chris Ross (@xorrior)
105         License: BSD 3-Clause
106
107         .DESCRIPTION
108         This function will capture video output from the hosts webcam. It will by default choose the f
109         Compression can be specified by naming pattern and the first compression method matching that pat
110
111         .PARAMETER RecordTime
112         Amount of time to record in seconds. It takes 1-2 seconds for the video to open. Defaults to 5.
113
114         .PARAMETER Path
115         File path to save the recorded output. Defaults to the current users APPDATA directory. The output
116
117         .PARAMETER VideoInputIndex
118         The index of the input device to use. To find this you can use Get-DXVideoInput. Default = 0 (fi
```

440 the index of the input device to use. To find this, you can use `Get-DeviceInput`. `$Device = 0` (first

```
200         $VideoCapture.VideoCompressor = $VidCompression
201     }
202     catch [System.Exception] {
203         Write-Error $_
204         break
205     }
206 }
207 if ($PSBoundParameters['AudioCompressorPattern']) {
208     try {
209         $AudCompression = Get-DXAudioCompression | ?{$_.Name -like $AudioCompressorPattern} | S
```

```
210         Write-Verbose "[+] Selected the Audio compression $($AudCompression.Name)"
211         Write-Verbose "[+] Setting Audio Compression"
212         $VideoCapture.AudioCompressor = $AudCompression
213     }
214     catch [System.Exception] {
215         Write-Error $_
216         break
217     }
218 }
219
220 #Set the framerate to help control size
221 $VideoCapture.FrameRate = $FrameRate
222 Write-Verbose "[+] Framerate Set to $FrameRate"
223
224 #Start the video capture
225 Write-Verbose "[+] Starting Webcam video capture"
226 try{
227     $VideoCapture.Start()
228 }
229 catch [System.Exception]{
230     $VideoCapture.Stop()
231     Write-Error $_
232     break
233 }
234
235 #Pause while the recording goes
236 Write-Verbose "[+] Capture Started. Sleeping $Recordtime Seconds..."
237 Start-Sleep -seconds $RecordTime
238
239 $VideoCapture.stop()
240
241 Write-Verbose "[+] Webcam video capture completed"
242
243 Get-ChildItem -Path $Path
244 }
245
246 ##### LOAD ASSEMBLIES USED BY ALL CMDLETS #####
247
248 #ALL CREDIT FOR THE FOLLOWING .NET ASSEMBLIES GOES TO THE ORIGINAL AUTHORS. THE ASSEMBLIES WERE NOT
249 #DirectX Capture Class Library:
250 #Author: Brian Low
251 #CodeProject User: @Brian-Low
252 #Link: http://www.codeproject.com/Articles/3566/DirectX-Capture-Class-Library
253 #License: Public Domain
254
255 #DirectShowNet:
```

```
256     #Author: Unknown
257     #http://directshownet.sourceforge.net/
258     #License: GNU Lesser General Public License
259
260     #Merged the DirectX and DShowNET assemblies
261     $encMergedAssembly = 'TVqQAAMAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
262
263     #Convert the base64 encoded assembly to raw bytes.
264     $bytes = [Convert]::FromBase64String($encMergedAssembly)
265     try
266     {
267         $null = [System.Reflection.Assembly]::Load($bytes)
268     }
269     catch [Exception]
270     {
271         Write-Error $_
272         break
273     }
```