

Forum:[Hack Forums](#)

Thread: [Hotkey control?](#)

Vrekt - October 17, 2015, 3:31 pm

Is there any sort-of control for changing hotkeys?

For example, I click this 'textbox' or the control, and I press a key or click a mouse button. It then sets the key, or mouse click to a variable and I'm able to use it with GetAsyncKeyState Lib?

GetAsyncKeyState won't work with keycodes anymore. It requires a 'Keys.KeyHere' (Keys.A,etc).

I've looked on google, and most don't seem fit.

Let's be honest, who wants to programmatically add all the hotkeys in one big IF statement.

nathan72389 - October 17, 2015, 4:28 pm

(Oct 17, 2015 03:31 PM)Vrekt Wrote: [\[CITE\]](#) GetAsyncKeyState won't work with keycodes anymore. lol?

its never been changed. where did you hear that from?

Vrekt - October 17, 2015, 5:32 pm

(Oct 17, 2015 04:28 PM)nathan72389 Wrote: [\[CITE\]](#) lol?

its never been changed. where did you hear that from?

I tried using keycodes, would not work. Only with Keys.Key

Jackie Chun - October 17, 2015, 5:32 pm

use RegisterHotKey

```
Code Public Const WM_HOTKEY As Integer = &H312
Public Const NULL As Integer = &H0 'See
https://msdn.microsoft.com/en-us/library/ms646309%28v=vs.85%29.aspx
Public Declare Function RegisterHotKey Lib "user32" (ByVal hwnd As IntPtr, ByVal id As Integer,
ByVal fsModifiers As Integer, ByVal vk As Integer) As Integer
Public Declare Function UnregisterHotKey Lib "user32" (ByVal hwnd As IntPtr, ByVal id As
Integer) As Integer
Private Sub MainForm_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
    Call RegisterHotKey(Me.Handle, 1, NULL, Keys.F1)
    Call RegisterHotKey(Me.Handle, 2, NULL, Keys.F2)
    FontComboBox1.Populate(True)
End Sub
Private Sub MainForm_FormClosing(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.FormClosingEventArgs) Handles MyBase.FormClosing
    Call UnregisterHotKey(Me.Handle, 1)
    Call UnregisterHotKey(Me.Handle, 2)
    Application.ExitThread()
End Sub
Protected Overrides Sub WndProc(ByRef m As System.Windows.Forms.Message)
    If m.Msg = WM_HOTKEY Then
        Dim id As IntPtr = m.WParam
        Select Case (id.ToString)
```

```

        Case "1"
        Case "2"
    End Select
End If
MyBase.WndProc(m)
End Subp

```

Vrekt - October 18, 2015, 7:14 am

(Oct 17, 2015 05:32 PM)Jackie Chun Wrote: [\[CITE\]](#) use RegisterHotKey

```

Code Public Const WM_HOTKEY As Integer = &H312
    Public Const NULL As Integer = &H0 'See
https://msdn.microsoft.com/en-us/library/ms646309%28v=vs.85%29.aspx
    Public Declare Function RegisterHotKey Lib "user32" (ByVal hwnd As IntPtr, ByVal id As Integer,
ByVal fsModifiers As Integer, ByVal vk As Integer) As Integer
    Public Declare Function UnregisterHotKey Lib "user32" (ByVal hwnd As IntPtr, ByVal id As
Integer) As Integer
    Private Sub MainForm_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        Call RegisterHotKey(Me.Handle, 1, NULL, Keys.F1)
        Call RegisterHotKey(Me.Handle, 2, NULL, Keys.F2)
        FontComboBox1.Populate(True)
    End Sub
    Private Sub MainForm_FormClosing(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.FormClosingEventArgs) Handles MyBase.FormClosing
        Call UnregisterHotKey(Me.Handle, 1)
        Call UnregisterHotKey(Me.Handle, 2)
        Application.ExitThread()
    End Sub
Protected Overrides Sub WndProc(ByRef m As System.Windows.Forms.Message)
    If m.Msg = WM_HOTKEY Then
        Dim id As IntPtr = m.WParam
        Select Case (id.ToString)
            Case "1"
            Case "2"
        End Select
    End If
    MyBase.WndProc(m)
End Subp

```

This will work perfectly thanks. Way better than the approach I was going to take.