

Forum:[Hack Forums](#)

Thread: [Hook To Disable Win Shortcuts C# \(.NET\)](#)

Volim te - June 6, 2010, 9:28 pm

C# (.NET)

OK I Will make this as fast as i can

1st We Need To Import System.Diagnostics And
System.Runtime.InteropServices

```
Code using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Runtime.InteropServices;
using System.Diagnostics;
```

```
namespace KyREADER
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
    }
}
```

Now assign callback function each time keyboard process

```
Code public Form1()
{
    InitializeComponent();
    ProcessModule objCurrentModule =
    Process.GetCurrentProcess().MainModule;

    objKeyboardProcess = new LowLevelKeyboardProc(captureKey);

    ptrHook = SetWindowsHookEx(13, objKeyboardProcess,
    GetModuleHandle(objCurrentModule.ModuleName), 0);
}
```

Now Under that we need

Code [StructLayout(LayoutKind.Sequential)]

```
private struct KBDLLHOOKSTRUCT
{
```

```

public Keys key;

public int scanCode;

public int flags;

public int time;

public IntPtr extra;

```

}Time For The DLL imports

```
Code private delegate IntPtr LowLevelKeyboardProc(int nCode, IntPtr wParam, IntPtr lParam);
```

```

[DllImport("user32.dll", CharSet = CharSet.Auto, SetLastError = true)]
private static extern IntPtr SetWindowsHookEx(int id, LowLevelKeyboardProc callback, IntPtr
hMod, uint dwThreadId);

```

```

[DllImport("user32.dll", CharSet = CharSet.Auto, SetLastError = true)]
private static extern bool UnhookWindowsHookEx(IntPtr hook);

```

```

[DllImport("user32.dll", CharSet = CharSet.Auto, SetLastError = true)]
private static extern IntPtr CallNextHookEx(IntPtr hook, int nCode, IntPtr wp, IntPtr lp);

```

```

[DllImport("kernel32.dll", CharSet = CharSet.Auto, SetLastError = true)]
private static extern IntPtr GetModuleHandle(string name);

```

```
[DllImport("user32.dll", CharSet = CharSet.Auto)]
```

Last Bit

```

Code private static extern short GetAsyncKeyState(Keys key);
private IntPtr ptrHook;
private LowLevelKeyboardProc objKeyboardProcess;
private IntPtr captureKey(int nCode, IntPtr wp, IntPtr lp)
{
    if (nCode >= 0)
    {
        KBDLLHOOKSTRUCT objKeyInfo = (KBDLLHOOKSTRUCT)Marshal.PtrToStructure(lp,
typeof(KBDLLHOOKSTRUCT));
        //DISABLES THE KEYS HERE>>>>>>!'

        //NOTE!! DOES --NOT-- Disable ctrl + Alt + Del!
        if (objKeyInfo.key == Keys.Alt || objKeyInfo.key == Keys.F4 || objKeyInfo.key ==
Keys.Escape || objKeyInfo.key == Keys.RControlKey || objKeyInfo.key == Keys.RWin
        || objKeyInfo.key == Keys.LWin)
        {

            return (IntPtr)1;

        }
    }
}

return CallNextHookEx(ptrHook, nCode, wp, lp);

```

```
}  
Full Code
```

||NOTE THIS WORKS 4 MOST KEYS AND SHORTCUTS BUT NOT CTRL & ALT & DEL (WORKS FOR EACH KEY CTRL ,DEL AND AIT JUST NOT THE SHORTCUT!

```
Code using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Windows.Forms;  
using System.Runtime.InteropServices;  
using System.Diagnostics;
```

```
namespace KyREADER  
{  
    public partial class Form1 : Form  
    {  
        public Form1()  
        {  
            InitializeComponent();  
            ProcessModule objCurrentModule = Process.GetCurrentProcess().MainModule;  
  
            objKeyboardProcess = new LowLevelKeyboardProc(captureKey);  
  
            ptrHook = SetWindowsHookEx(13, objKeyboardProcess,  
GetModuleHandle(objCurrentModule.ModuleName), 0);  
        }  
  
        [StructLayout(LayoutKind.Sequential)]  
  
        private struct KBDLLHOOKSTRUCT  
        {  
            public Keys key;  
  
            public int scanCode;  
  
            public int flags;  
  
            public int time;  
  
            public IntPtr extra;  
        }  
  
        // DLL IMPORTS FOR KEYBOARD HOOK!>>>>>  
        private delegate IntPtr LowLevelKeyboardProc(int nCode, IntPtr wParam, IntPtr lParam);  
  
        [DllImport("user32.dll", CharSet = CharSet.Auto, SetLastError = true)]  
        private static extern IntPtr SetWindowsHookEx(int id, LowLevelKeyboardProc callback, IntPtr
```

```
hMod, uint dwThreadId);
```

```
[DllImport("user32.dll", CharSet = CharSet.Auto, SetLastError = true)]
```

```
private static extern bool UnhookWindowsHookEx(IntPtr hook);
```

```
[DllImport("user32.dll", CharSet = CharSet.Auto, SetLastError = true)]
```

```
private static extern IntPtr CallNextHookEx(IntPtr hook, int nCode, IntPtr wp, IntPtr lp);
```

```
[DllImport("kernel32.dll", CharSet = CharSet.Auto, SetLastError = true)]
```

```
private static extern IntPtr GetModuleHandle(string name);
```

```
[DllImport("user32.dll", CharSet = CharSet.Auto)]
```

```
private static extern short GetAsyncKeyState(Keys key);
```

```
private IntPtr ptrHook;
```

```
private LowLevelKeyboardProc objKeyboardProcess;
```

```
private IntPtr captureKey(int nCode, IntPtr wp, IntPtr lp)
```

```
{
```

```
    if (nCode >= 0)
```

```
    {
```

```
        KBDLLHOOKSTRUCT objKeyInfo = (KBDLLHOOKSTRUCT)Marshal.PtrToStructure(lp,
        typeof(KBDLLHOOKSTRUCT));
```

```
        //DISABLES THE KEYS HERE>>>>>>!
```

```
        //NOTE!! DOES --NOT-- Disable ctrl + Alt + Del!
```

```
        if (objKeyInfo.key == Keys.Alt || objKeyInfo.key == Keys.F4 || objKeyInfo.key ==
        Keys.Escape || objKeyInfo.key == Keys.RControlKey || objKeyInfo.key == Keys.RWin
        || objKeyInfo.key == Keys.LWin)
```

```
        {
```

```
            return (IntPtr)1;
```

```
        }
```

```
    }
```

```
    return CallNextHookEx(ptrHook, nCode, wp, lp);
```

```
}
```

```
private void Form1_Load(object sender, EventArgs e)
```

```
{
```

```
}
```

The Ostrich - November 6, 2010, 3:46 pm

how do you re-enable them?

Volim te - November 6, 2010, 6:08 pm

The hook only works when the process is running so to

Enable them the program has to be ended/killed

The Ostrich - November 7, 2010, 3:00 pm

i get the error "'InitializeComponent' does not exist in the current context" how do i fix that?

Volim te - November 7, 2010, 6:46 pm

you ,Porlly put this part in the wrong place

```
Code ProcessModule objCurrentModule =  
Process.GetCurrentProcess().MainModule;
```

```
objKeyboardProcess = new LowLevelKeyboardProc(captureKey);
```

```
ptrHook = SetWindowsHookEx(13, objKeyboardProcess,  
GetModuleHandle(objCurrentModule.ModuleName), 0);you ,do not make a new void for ^ that  
it go's here (its under your project's namespace)
```

```
Code public partial class Form1 : Form
```

```
{  
public Form1()  
{  
InitializeComponent();
```

```
// here
```

```
}PM me or post if you keep getting that error and i will upload a project of it
```

spyware - January 18, 2013, 12:20 pm

Thanks for the share, it works perfectly :)

Got many keys disabled :)