A method and medium for a computer readable input area. The input area is created by a computer program on a display capable of receiving touch-screen input. The computer on which the input area in used is at least a 32-bit system. The input area may contain a keyboard which is an image map. External programming may selectively access the input area through a dynamic link library. The input area has no task bar and may not be minimized, maximized, or deleted. Therefore, the input area becomes an integral component and provides the user with a constant and reliable method of inputting information into the computer program.

10 Claims, 2 Drawing Sheets
METHOD AND MEDIUM FOR COMPUTER READABLE KEYBOARD DISPLAY INCAPABLE OF USER TERMINATION

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates generally to a method and medium for inputting data, and more particularly, to a keyboard of constant size and shape present on the screen of a touch-screen style computer whenever user input may be desired. The keyboard display may be used by any number of computer software programs, including any known operating system in which a touch-sensitive computer display may be incorporated. Additionally, the present invention may be used in conjunction with any individual computer, network and/or Internet based system.

2. Problem in the Art

Computers with touch-screen displays, allowing a user to simply press on a desired location to obtain a desired input, have been around for some time. For example, a pen-based computer, such as the Fujitsu Model Point 1600, allows a user to press on the screen using the attached pen or other stylus, and thereby provide user input. The use of such a pen-based computer allows a user to enter all necessary data without the need for an external keyboard, mouse or other input device. The use of an on-screen keyboard in such a computer allows a user to input data without the need for additional handwriting recognition software. Handwriting recognition software, while constantly improving, is often inaccurate and cumbersome. Further, such handwriting recognition software is often processor intensive.

Currently, on-screen keyboards allow a user to maximize, minimize, or simply remove the keyboard on the display. Further, the shape and size of the keyboard may be altered. Often, such alterations or terminations are accidental and returning a keyboard to a usable size and shape wastes valuable time. In a medical setting, for example, it is highly undesirable to have a care provider attempting to recover from an accidental keyboard alteration when the care provider should be attending to and recording information on patients. It is therefore desirable to provide an on-screen keyboard which is incapable of alteration or termination by a user.

More and more applications are being developed for pen-based or touch-screen based computers. These applications will typically require a user to input data at a specific location on the screen. An on-screen keyboard may be necessary to provide the desired input. However, current on-screen keyboards may be moved by the user and therefore placed in undesirable locations which may block necessary text input fields or instructions. Further, current on-screen keyboard include a task bar having minimizing and maximizing buttons which allow a user to enlarge or reduce the window in which the keyboard appears. Often, such keyboards also include a close button which allows the user to terminate the keyboard. Upon pressing these buttons, many computer novices have difficulty launching another instance of the keyboard or recovering the keyboard to a usable state. It is therefore desirable to have an on-screen keyboard which is capable of permanent placement on a computer display.

Computer programs may require input only randomly. Many ask for user input and then present the results. As it would clearly hamper the presentation of results, data or other information to have an on-screen keyboard present at all times, it is desirable to provide an on-screen keyboard which may be selectively called up as a subroutine or subprogram by a variety of programming.

There is therefore a need to have an on-screen keyboard which solves these and other problems in the art.

FEATURES OF THE INVENTION

A general feature of the present invention is the provision of an input area which overcomes the problems found in the prior art.

A further feature of the present invention is the provision of an input area which may be used in conjunction with touch-sensitive displays.

Another feature of the present invention is the provision of an input area which is immutable.

A further feature of the present invention is the provision of an input area which may not be moved.

A still further feature of the present invention is the provision of an input area which allows a user to input data without the need for handwriting recognition software.

An additional feature of the present invention is the provision of an input area which may not be maximized.

Another feature of the present invention is the provision of an input area which may not be minimized.

A still further feature of the present invention is the provision of an input area which contains a keyboard.

Another feature of the present invention is the provision of an input area which may be selectively used by a computer program.

A still further feature of the present invention is the provision of an input area which provides an easy to use and reliable method of inputting information into a computer system regardless of the level of computer skill possessed by the user.

These, as well as other features and advantages of the present invention will become apparent from the following specification and claims.

SUMMARY OF THE INVENTION

The present invention generally comprises an immutable keyboard display. In a preferred embodiment, the present invention includes a software application that provides a keyboard display which may not be minimized, maximized, closed, or deleted. Further, the keyboard display allows a user to input information as desired via a touch-screen based or pen-based computer.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a pictorial representation of a display of a pen-based computer incorporating the keyboard display of the present invention.

FIG. 2 is a close-up view of the keyboard display of the present invention.

DETAILED DESCRIPTION OF THE EMBODIMENT

The present invention will be described as it applies to its preferred embodiment. It is not intended that the present invention be limited to the described embodiment. It is intended that the invention cover all modifications and
alternatives which may be included within the spirit and scope of the invention.

As shown in FIG. 1, a pen-based computer 10, such as the Fujitsu Model Point 1600, includes a touch-sensitive display 12. On the display 12 is shown the user interface for a software application 14 which may be running from or accessed by the computer 10. It is to be understood that the computer 10 could be a stand-alone computer or a part of any network or Internet based system. However, the computer 10 preferably provides a 32 bit environment. Computer 10 may access any type of software application through any number of known drives or via a network or web server. Once accessed, the user will see the application as it appears on the display 12 of the computer 10. The application may ask for user input at various locations through the use of text boxes 16 or other fields. The user may provide the desired input by holding the pen 18 or any other known input device which may include the user’s finger, and pressing on the display 12 of the computer 10 so as to strike a desired key 22 of the keyboard 20.

The keyboard 20 is preferably an image map or active map incorporated at a set location on the display 12. The keyboard 20 may not be moved, maximized, or minimized. Therefore, the keyboard 20 provides the user with a constant input area to which the user may become accustomed and becomes an integral component.

The keyboard 20, as shown in FIG. 2, contains a plurality of keys 22. The keys 22 may include all those currently found on any standard typewriter or computer keyboard, or may be application-specific. For instance, if the software in which user input is desired is primarily financial software, the keyboard 20 may include only numbers. Further, if the software requires the user to input names or words, the keyboard 20 may include one key 22 for every letter of the alphabet and any necessary punctuation or function keys. Further, the keys 22 may be programmed to represent any symbol or accentuated letter to allow the keyboard to be used in applications in which input may be required in various languages.

The keyboard 20 is preferably the result of a software application written in Visual Basic or C++, though various software programming languages may be used. The keyboard has all task bars removed and may not be minimized, maximized, deleted, closed or resized and is therefore immutable. Preferably, the keyboard application is a subroutine or subprogram which is made available for use by external software applications. The keyboard application is preferably part of the operating system running on the computer 10. Incorporating the keyboard application into the computer 10 allows the keyboard application to be available to any external software application capable of running on the computer 10. The keyboard application may include a dynamic link library (dll) application. The dll application allows the external software to selectively use the keyboard and either have the keyboard in or out. This allows the software to use the entirety of the screen when necessary for displaying information or results.

An example of the keyboard application programming as it would appear in Visual Basic is:

```
Option Explicit
Public Sub Shift_Down()
    cmdLeftShift.Caption="LowerCase"
    cmdRightShift.Caption="LowerCase"
    cmdLeftShift.Tag="OFF"
    cmdRightShift.Tag="OFF"
    Caps_OFF
```

```
Commandl(26).Caption="0"
Commandl(27).Caption="1"
Commandl(28).Caption="2"
Commandl(29).Caption="3"
Commandl(30).Caption="4"
Commandl(31).Caption="5"
Commandl(32).Caption="6"
Commandl(33).Caption="7"
Commandl(34).Caption="8"
Commandl(35).Caption="9"
Commandl(36).Caption=";"
Commandl(37).Caption="="
Commandl(38).Caption="/"
Commandl(39).Caption="\"
Commandl(40).Caption="`
Commandl(41).Caption="[
Commandl(42).Caption="]
Commandl(43).Caption="`n"
Commandl(44).Caption="`r"
Commandl(45).Caption="`t"
Commandl(46).Caption="`u"
Commandl(47).Caption="`v"
Commandl(48).Caption="`w"
Commandl(49).Caption="`x"
Commandl(50).Caption="`y"
Commandl(51).Caption="`z"
Commandl(52).Caption="`a"
Commandl(53).Caption="`b"
Commandl(54).Caption="`c"
Commandl(55).Caption="`d"
Commandl(56).Caption="`e"
Commandl(57).Caption="`f"
Commandl(58).Caption="`g"
Commandl(59).Caption="`h"
Commandl(60).Caption="`i"
Commandl(61).Caption="`j"
Commandl(62).Caption="`k"
Commandl(63).Caption="`l"
Commandl(64).Caption="`m"
Commandl(65).Caption="`n"
Commandl(66).Caption="`o"
Commandl(67).Caption="`p"
Commandl(68).Caption="`q"
Commandl(69).Caption="`r"
Commandl(70).Caption="`s"
Commandl(71).Caption="`t"
Commandl(72).Caption="`u"
Commandl(73).Caption="`v"
Commandl(74).Caption="`w"
Commandl(75).Caption="`x"
Commandl(76).Caption="`y"
Commandl(77).Caption="`z"
```

```
Commandl(20).Caption="F1"
Commandl(21).Caption="F2"
Commandl(22).Caption="F3"
Commandl(23).Caption="F4"
Commandl(24).Caption="F5"
Commandl(25).Caption="F6"
Commandl(26).Caption="F7"
Commandl(27).Caption="F8"
Commandl(28).Caption="F9"
Commandl(29).Caption="F10"
Commandl(30).Caption="F11"
Commandl(31).Caption="F12"
Commandl(32).Caption="Enter"
Commandl(33).Caption="Space"
Commandl(34).Caption="Esc"
Commandl(35).Caption="Ins"
Commandl(36).Caption="Del"
Commandl(37).Caption="Home"
Commandl(38).Caption="End"
Commandl(39).Caption="PgUp"
Commandl(40).Caption="PgDn"
```

```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
Public Sub ShifLUp()

    cmdLeftShift.Caption = "UpperCase"
    cmdRightShift.Caption = "UpperCase"
    cmdLeftShift.Tag = "ON"
    cmdRightShift.Tag = "ON"

Caps_On
Command(26).Caption = "A"
Command(27).Caption = "B"
Command(28).Caption = "C"
Command(29).Caption = "D"

End Sub

Public Sub Shift_Up( )

    cmdLeftShift.Caption = "UpperCase"
    cmdRightShift.Caption = "UpperCase"
    cmdLeftShift.Tag = "ON"
    cmdRightShift.Tag = "ON"

Caps_On
Command(41).Caption = "A"
Command(42).Caption = "B"
Command(43).Caption = "C"
Command(44).Caption = "D"

End Sub
Public Sub Key_Layout1()

US 6,784,873 B1

End Sub

Public Sub Set_Caps_Lock( )

If cmdCapsLock.Tag="OFF" Then
    cmdCapsLock.Caption="Caps On"
    cmdCapsLock.Tag="ON"
Else
    cmdCapsLock.Caption="Caps Off"
    cmdCapsLock.Tag="OFF"
End If

End Sub

Public Sub Caps_OFF( )

Public Sub Caps_ON( )

If cmdCapsLock.Tag="OFF" Then
    cmdCapsLock.Caption="Caps On"
    cmdCapsLock.Tag="ON"
Else
    cmdCapsLock.Caption="Caps Off"
    cmdCapsLock.Tag="OFF"
End If

End Sub

Public Sub Caps_Lock( )

Public Sub Caps_Off( )

Public Sub Caps_On( )

If cmdCapsLock.Tag="OFF" Then
    cmdCapsLock.Caption="Caps On"
    cmdCapsLock.Tag="ON"
Else
    cmdCapsLock.Caption="Caps Off"
    cmdCapsLock.Tag="OFF"
End If

End Sub

End Sub
Command(33).Left=3000*lngPcnt
Command(33).Width=intLetterWidth
Command(33).Height=intHeight

'7 Key
Command(44).Font.Size=intFontSize
Command(44).Top=intRow2Top
Command(44).Left=3000*lngPcnt
Command(44).Width=intLetterWidth
Command(44).Height=intHeight

'8 Key
Command(34).Font.Size=intFontSize
Command(34).Top=intRow2Top
Command(34).Left=3420*lngPcnt
Command(34).Width=intLetterWidth
Command(34).Height=intHeight

'9 Key
Command(35).Font.Size=intFontSize
Command(35).Top=intRow2Top
Command(35).Left=3840*lngPcnt
Command(35).Width=intLetterWidth
Command(35).Height=intHeight

0 Key
Command(26).Font.Size=intFontSize
Command(26).Top=intRow2Top
Command(26).Left=4260*lngPcnt
Command(26).Width=intLetterWidth
Command(26).Height=intHeight

' Key
Command(46).Font.Size=intFontSize
Command(46).Top=intRow2Top
Command(46).Left=4680*lngPcnt
Command(46).Width=intLetterWidth
Command(46).Height=intHeight

' Key
Command(47).Font.Size=intFontSize
Command(47).Top=intRow2Top
Command(47).Left=5100*lngPcnt
Command(47).Width=intLetterWidth
Command(47).Height=intHeight

Backspace Key
cmdBackspace.Font.Size=dblFontSize
cmdBackspace.Top=intRow2Top
cmdBackspace.Left=5520*lngPcnt
cmdBackspace.Width=intRightBorder
cmdBackspace.Left
cmdBackspace.Height=intHeight

Row 3
TAB Key
cmdTabFont.Size=dblFontSize
cmdTabTop=intRow3Top
cmdTabLeft=60*lngPcnt
cmdTabWidth=650*lngPcnt
cmdTabHeight=intHeight

'Q Key
Command(16).Font.Size=intFontSize
Command(16).Top=intRow3Top
Command(16).Left=780*lngPcnt

Command(16).Width=intLetterWidth
Command(16).Height=intHeight

'W Key
Command(22).Font.Size=intFontSize
Command(22).Top=intRow3Top
Command(22).Left=1200*lngPcnt
Command(22).Width=intLetterWidth
Command(22).Height=intHeight

'E Key
Command(4).Font.Size=intFontSize
Command(4).Top=intRow3Top
Command(4).Left=1620*lngPcnt
Command(4).Width=intLetterWidth
Command(4).Height=intHeight

'R Key
Command(17).Font.Size=intFontSize
Command(17).Top=intRow3Top
Command(17).Left=2040*lngPcnt
Command(17).Width=intLetterWidth
Command(17).Height=intHeight

'T Key
Command(19).Font.Size=intFontSize
Command(19).Top=intRow3Top
Command(19).Left=2460*lngPcnt
Command(19).Width=intLetterWidth
Command(19).Height=intHeight

'Y Key
Command(24).Font.Size=intFontSize
Command(24).Top=intRow3Top
Command(24).Left=2880*lngPcnt
Command(24).Width=intLetterWidth
Command(24).Height=intHeight

'U Key
Command(20).Font.Size=intFontSize
Command(20).Top=intRow3Top
Command(20).Left=3300*lngPcnt
Command(20).Width=intLetterWidth
Command(20).Height=intHeight

'I Key
Command(8).Font.Size=intFontSize
Command(8).Top=intRow3Top
Command(8).Left=3720*lngPcnt
Command(8).Width=intLetterWidth
Command(8).Height=intHeight

'O Key
Command(14).Font.Size=intFontSize
Command(14).Top=intRow3Top
Command(14).Left=4140*lngPcnt
Command(14).Width=intLetterWidth
Command(14).Height=intHeight

'P Key
Command(15).Font.Size=intFontSize
Command(15).Top=intRow3Top
Command(15).Left=4560*lngPcnt
Command(15).Width=intLetterWidth
Command(15).Height=intHeight
[ Key
CommandI(41).FontSize=intFontSize
CommandI(41).Top=intRow3Top
CommandI(41).Left=4980*lngPcnt
CommandI(41).Width=intLetterWidth
CommandI(41).Height=intHeight
] Key
CommandI(42).FontSize=intFontSize
CommandI(42).Top=intRow3Top
CommandI(42).Left=5400*lngPcnt
CommandI(42).Width=intLetterWidth
CommandI(42).Height=intHeight
\ Key
CommandI(43).FontSize=intFontSize
CommandI(43).Top=intRow3Top
CommandI(43).Left=5820*lngPcnt
CommandI(43).Width=intLetterWidth
CommandI(43).Height=intHeight
Delete
cmdDelete.Font.Size=dblFontSize
cmdDelete.Top=intRow3Top
cmdDelete.Left=6240*lngPcnt
cmdDelete.Width=intRightBorder-cmdDelete.Left
cmdDelete.Height=intHeight
' Row 4
' Caps Lock Key
cmdCapsLock.Font.Size=dblFontSize
cmdCapsLock.Top=intRow4Top
cmdCapsLock.Left=60*lngPcnt
cmdCapsLock.Width=1070*lngPcnt
cmdCapsLock.Height=intHeight
' A Key
CommandI(0).FontSize=intFontSize
CommandI(0).Top=intRow4Top
CommandI(0).Left=1200*lngPcnt
CommandI(0).Width=intLetterWidth
CommandI(0).Height=intHeight
' S Key
CommandI(18).FontSize=intFontSize
CommandI(18).Top=intRow4Top
CommandI(18).Left=1620*lngPcnt
CommandI(18).Width=intLetterWidth
CommandI(18).Height=intHeight
'D Key
CommandI(3).FontSize=intFontSize
CommandI(3).Top=intRow4Top
CommandI(3).Left=2040*lngPcnt
CommandI(3).Width=intLetterWidth
CommandI(3).Height=intHeight
' F Key
CommandI(5).FontSize=intFontSize
CommandI(5).Top=intRow4Top
CommandI(5).Left=2460*lngPcnt
CommandI(5).Width=intLetterWidth
CommandI(5).Height=intHeight
' G Key
CommandI(6).FontSize=intFontSize
CommandI(6).Top=intRow4Top
CommandI(6).Left=2880*lngPcnt
CommandI(6).Width=intLetterWidth
CommandI(6).Height=intHeight
'H Key
CommandI(7).FontSize=intFontSize
CommandI(7).Top=intRow4Top
CommandI(7).Left=3300*lngPcnt
CommandI(7).Width=intLetterWidth
CommandI(7).Height=intHeight
' J Key
CommandI(9).FontSize=intFontSize
CommandI(9).Top=intRow4Top
CommandI(9).Left=3720*lngPcnt
CommandI(9).Width=intLetterWidth
CommandI(9).Height=intHeight
' K Key
CommandI(10).FontSize=intFontSize
CommandI(10).Top=intRow4Top
CommandI(10).Left=4140*lngPcnt
CommandI(10).Width=intLetterWidth
CommandI(10).Height=intHeight
' L Key
CommandI(11).FontSize=intFontSize
CommandI(11).Top=intRow4Top
CommandI(11).Left=4560*lngPcnt
CommandI(11).Width=intLetterWidth
CommandI(11).Height=intHeight
'; Key
CommandI(39).FontSize=intFontSize
CommandI(39).Top=intRow4Top
CommandI(39).Left=4980*lngPcnt
CommandI(39).Width=intLetterWidth
CommandI(39).Height=intHeight
' Row 5
' Left Shift Key cmdLeftShift.Font.Size=dblFontSize
cmdLeftShift.Top=intRow5Top
cmdLeftShift.Left=60*lngPcnt
cmdLeftShift.Width=1293*lngPcnt
cmdLeftShift.Height=intHeight
' Z Key
CommandI(25).FontSize=intFontSize
CommandI(25).Top=intRow5Top
CommandI(25).Left=1423*lngPcnt
CommandI(25).Width=intLetterWidth
Private Sub cmdAlt_Click()
    If cmdAlt.Tag = "OFF" Then
        cmdAlt.Tag = "ON"
        cmdAlt.Caption = "Alt On"
    Else
        cmdAlt.Tag = "OFF"
        cmdAlt.Caption = "Alt Off"
    End If
End Sub

Private Sub cmdAlt2_Click()
    cmdAlt Click
End Sub

Private Sub cmdBackspace_Click()
    strKeys = "{BKSP}"
    SendVKeys (strKeys)
End Sub

Private Sub cmdCntrl_Click()
    If cmdCntrl.Tag = "OFF" Then
        cmdCntrl.Tag = "ON"
        cmdCntrl.Caption = "Ctrl On"
        cmdCntrl2.Tag = "ON"
        cmdCntrl2.Caption = "Ctrl On"
    Else
        cmdCntrl.Tag = "OFF"
        cmdCntrl.Caption = "Ctrl Off"
        cmdCntrl2.Tag = "OFF"
        cmdCntrl2.Caption = "Ctrl Off"
    End If
End Sub

Private Sub cmdCntrl2_Click()
    cmdCntrl Click
End Sub

Private Sub cmdDelete_Click()
    strKeys = " "
    strKeys = strKeys & "{DEL}"
    SendVKeys (strKeys)
End Sub

Private Sub cmdEnter_Click()
    strKeys = 
    strKeys = strKeys & "{ENTER}"
    SendVKeys (strKeys)
End Sub

Private Sub cmdEscape_Click()
    strKeys = 
    SendVKeys (strKeys)
End Sub

Private Sub cmdExitKeyboard_Click()
End
End Sub

Private Sub cmdLeftShift_Click()
    If cmdLeftShift.Tag = "OFF" Then
        If cmdCapsLock.Tag = "OFF" Then
            Set_Caps_Lock
            Shift_Up
        Else
            Shift_Up
        End If
    Else
        Set_Caps_Lock
        Shift_Up
    End If
End Sub

Private Sub cmdMoveLeft_Click()
    strKeys = 
    strKeys = strKeys & "{LEFT}"
    SendVKeys (strKeys)
End Sub

Private Sub cmdMoveRight_Click()
    strKeys = 
    strKeys = strKeys & "{RIGHT}"
    SendVKeys (strKeys)
End Sub

Private Sub cmdRightShift_Click()
    cmdLeftShift_Click
End Sub

Private Sub cmdSpaceBar_Click()
    strKeys = " "
    SendVKeys (strKeys)
End Sub

Private Sub cmdTab_Click()
    strKeys = 
    If cmdLeftShift.Tag = "ON" Then
        strKeys = strKeys & "+"
    End If
    SendVKeys (strKeys)
End Sub

Private Sub Form_Activate()
    Dim dl&
    "KeyboardWindow=GetForegroundWindow dl&=
    SetWindowPos(hwnd, -1, 4905, 7965, 6975, 2475, &H1 Or &H2)
End Sub

Private Sub Form_GotFocus()
    If Me.WindowState <> 0 Then
        Me.WindowState = 0
        Me.Width = 7000
    End If
End Sub

Private Sub Form_Resize()
    If Me.WindowState <> 0 Then
        Me.WindowState = 0
        Me.Width = 7000
    End If
End Sub

Private Sub Key_Layout()
End Sub

Private Sub cmdCapsLock_Click()
    If cmdCapsLock.Tag = "OFF" Then
        cmdCapsLock.Caption = "Caps On"
        cmdCapsLock.Tag = "ON"
        Caps_On
    Else
        cmdCapsLock.Tag = "OFF"
        cmdCapsLock.Caption = "Caps Off"
        Caps_Off
    End If
End Sub

Private Sub cmdCmd1_Click(Index As Integer)
    If cmdCmd1.Tag = "OFF" Then
        cmdCmd1.Tag = "ON"
        cmdCmd1.Caption = "Cmd On"
    Else
        cmdCmd1.Tag = "OFF"
        cmdCmd1.Caption = "Cmd Off"
    End If
End Sub

Private Sub cmdCmd2_Click()
    cmdCmd2 Click
End Sub

Private Sub cmdCmd3_Click()
    cmdCmd3 Click
End Sub

Private Sub cmdCmd4_Click()
    cmdCmd4 Click
End Sub
If cmdLeftShift.Tag="ON" Then
    strKeys=strKeys & "+"
End If
If cmdAlt.Tag="ON" Then
    strKeys=strKeys & "%"
End If
If cmdCntrl.Tag="ON" Then
    strKeys=strKeys & "A"
End If
SendVKeys (strKeys)

Private Sub Form_Load()
    Dim hSysMenu As Long
    Dim nCnt As Long
    ' First, show the form
    Me.Show
    ' Get handle to our form's system menu
    ' (Restore, Maximize, Move, close etc.)
    SysMenu=GetSystemMenu(Me.hwnd, False)
    If hSysMenu Then
        nCnt=GetMenuItemCount(hSysMenu)
        If nCnt Then
            ' Menu count is based on 0 (0, 1, 2, 3 ...)
            RemoveMenu hSysMenu, nCnt-1, _
            MF_BYPOSITION Or MF_REMOVE
            RemoveMenu hSysMenu, nCnt-2, _
            MF_BYPOSITION Or MF_REMOVE ' Remove the separator
            DrawMenuBar Me.hwnd
            ' Force caption bar's refresh. Disabling X button
            Me.Caption="GeniSus Keyboard"
        End If
    End If
End Sub

Private Sub Unload(Cancel As Integer)
End Sub

Public Sub DeactivateClose()
End Sub

An example of an accompanying dynamic link library, .dll application, through which external applications may access the keyboard application is:

```cpp
#include <windows.h>
#include <winuser.h>
#pragma data_seg("SHARDATA")
static int hWndActive=0;
static int hWndSelf=0;
static HHOOK hHook=0;
#pragma data_seg()

BOOL APIENTRY DllMain( HANDLE hModule, 
DWORD ul_reason_for_call, 
LPVOID lpReserved)
{
    switch (ul_reason_for_call)
    {
        case DLL_PROCESS_ATTACH:
        case DLL_THREAD_ATTACH:
        case DLL_THREAD_DETACH:
        case DLL_PROCESS_DETACH: break;
    }
    return TRUE;
}

long CALLBACK CBTProc( 
    int nCode, // hook code
    WPARAM wParam, // current-process flag
    LPARAM lParam // message data
    )
{
    if (nCode==HCBT_ACTIVATE &
        (int)wParam != hWndSelf)
    {
        hWndActive=(int)wParam;
    }
    return CallNextHookEx(hHook, nCode, wParam, lParam);
}
void_stdcall HookMsg(int hwnd)
{
    HINSTANCE hModule;
    hModule=GetModuleHandle("vKeyHook.dll");
    hHook=SetWindowsHookEx(WH_CBT, CBTProc, hModule, 0);
    hWndSelf=hwnd;
}
void_stdcall UnHookMsg( )
{
    UnhookWindowsHookEx(hHook);
}
int_stdcall GetActiveWnd( )
{
    return hWndActive;
    These codes are preferably executed in conjunction with a Windows 98® operating system. These codes may be executed in any type of system, including, but not limited to, a web based system, a computer network, or any personal computer, personal digital assistant or other device.

    As can clearly be seen in FIGS. 1 and 2, there are no minimizing, maximizing, or close options available for the user. Therefore, a user can input data by selecting keys 22 on the keyboard 20 as necessary.

    A general description of the present invention as well as a preferred embodiment of the present invention has been set forth above. Those skilled in the art to which the present invention pertains will recognize and be able to practice additional variations in the methods and systems described which fall within the teachings of this invention. Accordingly, all such modifications and additions are deemed to be within the scope of the invention which is to be limited only by the claims appended hereto.

What is claimed is:
1. A method of entering data on a touch screen display, the method comprising:
   - invoking a computer program in which user input is sought;
   - invoking an input area, including a plurality of data input fields;
   - invoking a graphical keyboard area incapable of user termination independent of termination of the input area, the graphical keyboard area having a plurality of keys on the display;
   - selecting keys on the keyboard to provide the desired input; and
automatically terminating the graphical keyboard area
after the desired input is received in the input area.
2. The method of entering data on a touch screen display
of claim 1 wherein the input area is created by an executable
code.
3. The method of entering data on a touch screen display
of claim 2 wherein the executable code is compiled visual
basic code.
4. The method of entering data on a touch screen display
of claim 1 wherein the computer program invokes the input
area.
5. The method of entering data on a touch screen display
of claim 4 wherein the computer program accesses a
dynamic link library file in order to invoke the input area.
6. The method of entering data on a touch screen display
of claim 5 wherein the dynamic link library file is a C++
program.
7. The method of entering data on a touch screen display
of claim 1 wherein the computer program is executing on a
personal computer.
8. The method of entering data on a touch screen display
of claim 1 wherein the computer program is executing on a
pen-based computer.
9. The method of entering data on a touch screen display
of claim 1 wherein the computer program is executing on a
computer with a touch-screen display.
10. A method of providing a user interface for receiving
information from a user using a user immutable graphical
keyboard linked to an input area, comprising:
invoking the input area;
determining that input from the user using the graphical
keyboard is needed within the input area;
invoking the graphical keyboard on a touch screen display
to receive input from a user, the graphical keyboard
placed in a set position;
persistently maintaining the graphical keyboard on the
touch screen display such that the user cannot move,
resize, remove, or close the graphical keyboard through
the user interface while the input area remains and
requires input;
receiving input within the input area from the user
through the graphical keyboard;
determining that further input from the user is no longer
needed in the input area; and
removing the graphical keyboard.