Reviewing the Stylebits Parameters

The four parameters of stylebits are AddBit, RemoveBit, AddExtendedBit, RemoveExtendedBit. For a review of these four parameters, and an introduction to Stylebits in general, please view Stylebits - Windows.

Stylebits and a Default Button

A default button is the button that will be triggered with the Enter key, as long as the dialog window has focus. Without stylebits, a default button is only possible in a Dialog Window and then only if the button has the extension *default*. Brent Thorn posted code using Stylebits _BS_DEFPUSHBUTTON to assign a default button in a regular window with any extension. Thanks, Brent! Be sure to give focus to the dialog window by setting focus to one of the (non-button) controls prior to the activation of the default button. A simple Setfocus to a textbox (any textbox) will accomplish that. Here is a modification of Brent's code.

```
'Using a Default Button in a Regular Window
'Thanks to Brent Thorn
' http://libertybasic.conforums.com/index.cgi?board=novice&action=disp
lay&num=1134254812
'Modifications by Janet = Menu and Setfocus to #main.tbx1

    WindowWidth = 300
    WindowHeight = 350
    UpperLeftX = Int((DisplayWidth - WindowWidth)/2)
    UpperLeftY = Int((DisplayHeight - WindowHeight)/2)

    Menu #main, "&Options", "E&xit", EndDemoMenu
    Textbox #main.tbx1, 150, 50, 120, 30
    Textbox #main.tbx2, 150, 100, 120, 30
    Textbox #main.tbx3, 150, 150, 120, 30
    Textbox #main.tbx4, 150, 200, 120, 30
    Stylebits #main.default, _BS_DEFPUSHBUTTON, 0, 0, 0
    Button #main.default, "", DefaultButton, UL, -100, -100

    Open "Default Button in a Regular Window" for Window as #main
    #main "Trapclose EndDemo"
    #main.textbox "!Setfocus"

    Wait

    Sub DefaultButton handle$
        #main.tbx1 "!Contents? text1$"
        #main.tbx2 "!Contents? text2$"
```

```
        #main.tbx3 "!Contents? text3$"
        #main.tbx4 "!Contents? text4$"
        Notice "The textboxes read";Chr$(13);Chr$(13) + _
            text1$;Chr$(13);text2$;Chr$(13);text3$;Chr$(13);text4$
    End Sub

    Sub EndDemoMenu
        Call EndDemo "#main"
    End Sub

    Sub EndDemo handle$
        Close #main
        End
    End Sub
```

Stylebits and Button Labels

Many of the stylebits statictext border effects can be obtained with buttons as well. Use the same windows style _WS_ prefixes seen in changing window and other control edges. You can even add a titlebar to a button. These stylebits give a bit of variety to the button appearance and may enhance the overall 3D effect.

```
_WS_BORDER, 0, 0, 0
_WS_DLGFRAME
_WS_CAPTION, _WS_MAXIMIZEBOX
0, 0, _WS_EX_CLIENTEDGE, 0
0, 0, _WS_EX_CLIENTEDGE or _WS_EX_DLGMODALFRAME, 0
```

This demo shows some of the border effects that can be achieved with stylebits

```
    WindowWidth = 600
    WindowHeight = 560
    UpperLeftX = Int((DisplayWidth-WindowWidth)/2)
    UpperLeftY = Int((DisplayHeight-WindowHeight)/3)

    Nomainwin

    Button #demo.bttn1, "One", Click, UL, 20, 60, 80, 40
    bttnText1$ = "0, 0, 0, 0 'No Stylebits"
    Statictext #demo, bttnText1$, 180, 70, 400, 30
    Button #demo.bttn2, "Two", Click, UL, 20, 120, 80, 40
```

```
    bttnText2$ = "_WS_BORDER, 0, 0, 0"
    Stylebits #demo.bttn2, _WS_BORDER, 0, 0, 0
    Statictext #demo, bttnText2$, 180, 130, 400, 30
    Button #demo.bttn3, "Three", Click, UL, 20, 180, 80, 40
    bttnText3$ = "_WS_DLGFRAME, 0, 0, 0"
    Stylebits #demo.bttn3, _WS_DLGFRAME, 0, 0, 0
    Statictext #demo, bttnText3$, 180, 190, 400, 30
    Button #demo.bttn4, "Four", Click, UL, 20, 240, 80, 100
    bttnText4$ = "_WS_CAPTION, _WS_MAXIMIZEBOX, 0, 0";Chr$(13) + _

"Remove maximize ability or double clicking caption will cause " + _
        "button to expand to size of window."
    Stylebits #demo.bttn4, _WS_CAPTION, _WS_MAXIMIZEBOX, 0, 0
    Statictext #demo, bttnText4$, 180, 250, 400, 90
    Button #demo.bttn5, "Five", Click, UL, 20, 360, 80, 40
    bttnText5$ = "0, 0, _WS_EX_CLIENTEDGE, 0"
    Stylebits #demo.bttn5, 0, 0, _WS_EX_CLIENTEDGE, 0
    Statictext #demo, bttnText5$, 180, 370, 400, 30
    Button #demo.bttn6, "Six", Click, UL, 20, 420, 80, 40
    bttnText6$ = "0, 0, _WS_EX_CLIENTEDGE or _WS_EX_DLGMODALFRAME, 0"
    Stylebits #demo.bttn6, 0, 0, _WS_EX_CLIENTEDGE or
 _WS_EX_DLGMODALFRAME, 0
    Statictext #demo, bttnText6$, 180, 420, 400, 60
    msg$ =
"These are only a few examples of button borders available with " + _
        "stylebits.  Experiment to find others."
    Statictext #demo, msg$, 20, 480, 560, 60

    Open "Button Effects with Stylebits" for Window as #demo
    #demo "Trapclose EndDemo"
    #demo "Font Times_New_Roman 14 Bold"
    Wait

Sub EndDemo handle$
    Close #demo
    End
End Sub

Sub Click handle$
    nButton$ = Right$(handle$, 1)
    Notice "Button #";nButton$;" Clicked"
End Sub
```

Stylebits and Tabbing

Pressing the TAB key can cycle focus from one control to the next. The cycling order is the same order as that which the controls are created. Sometimes it may be desirable to not include a control in the tabbed cycle. The programmer may not want the QUIT button to be included in the TAB Cycle so that the user doesn't inadvertently exit a program prematurely. Assigning the stylebit _WS_TABSTOP to the RemoveBit (second) stylebits parameter will keep that button from gaining focus in the TAB sequence. The button does remain active and will receive focus when mouseclicked.

```
    WindowWidth = 600
    WindowHeight = 560
    UpperLeftX = Int((DisplayWidth-WindowWidth)/2)
    UpperLeftY = Int((DisplayHeight-WindowHeight)/3)

    Nomainwin

    Button #demo.bttn1, "Button 1", Click, UL, 100, 100, 120, 40
    Stylebits #demo.bttn1, 0, 0, 0, 0
    Button #demo.bttn2, "Button 2", Click, UL, 100, 200, 120, 40
    Stylebits #demo.bttn2, 0, 0, 0, 0
    Button #demo.bttn3, "Button 3", Click, UL, 320, 100, 120, 40
    Stylebits #demo.bttn3, 0, 0, 0, 0
    Button #demo.bttn4, "Button 4", Click, UL, 320, 200, 120, 40
    Stylebits #demo.bttn4, 0, 0, 0, 0
    Button #demo.bttn5, "Exit", endDemo, UL, 100, 300, 120, 40
    Statictext #demo, "0, _WS_TABSTOP, 0, 0", 40, 350, 200, 30
    Stylebits #demo.bttn5, 0, _WS_TABSTOP, 0, 0
    Textbox #demo.txtbx, 320, 300, 120, 40
    Stylebits #demo.txtbx, 0, _WS_TABSTOP, 0, 0
    Statictext #demo, "0, _WS_TABSTOP, 0, 0", 320, 350, 200, 30
    msg$ =
"Use the TAB key to maneuver from one button to the next.  " + _

"The button with focus has a darkened outline.  The removal of " + _

"_WS_TABSTOP stylebits prevents the EXIT button and the textbox " + _

"from being part of the tabbed cycle.  Note that one button must " + _
        "receive focus before the TAB key can be recognized."
    StaticText #demo.txt5, msg$, 20, 390, 560, 120

    Open "Button Effects with Stylebits" for Window as #demo
    #demo "Trapclose EndDemo"
    #demo "Font Times_New_Roman 14 Bold"
    #demo.bttn4 "!Setfocus"

    Wait
```

```
Sub EndDemo handle$
    Close #demo
    End
End Sub

Sub Click handle$
    nButton$ = Right$(handle$, 1)
    Notice "Button #";nButton$;" Clicked"
End Sub
```

Stylebits and Formatted Text Labels

The stylebits commands to format button text labels are similar to those _ES_ textbox and statictext formatting commands. Button formatting style commands begin with _BS_. The default style is text that remains on one line centered both vertically and horizontally. With stylebits, you can left justify, right justify, center, place text at the top of the button and place text at the bottom of the button. The most useful of these stylebits is the _BS_MULTILINE which allows word wrapping and multiline button labels.

```
_BS_LEFT
_BS_CENTER 'The Default
_BS_RIGHT
_BS_TOP
_BS_BOTTOM
_BS_MULTLINE
```

This third demo demonstrates button text formatting with stylebits

```
    WindowWidth = 600
    WindowHeight = 560
    UpperLeftX = Int((DisplayWidth-WindowWidth)/2)
    UpperLeftY = Int((DisplayHeight-WindowHeight)/3)

    Nomainwin
    Button #demo.bttn1, "Default", Click, UL, 20, 80, 120, 80
    Statictext #demo, "0, 0, 0, 0", 44, 165, 260, 60
    Button #demo.bttn2,
"Multiline Text on a Button", Click, UL, 20, 230, 120, 80
    Stylebits #demo.bttn2, _BS_MULTILINE, 0, 0, 0
    Statictext #demo, "_BS_MULTILINE, 0, 0, 0", 20, 315, 260, 60
    Button #demo.bttn3, "Text@Bottom", Click, UL, 300, 80, 120, 80
    Stylebits #demo.bttn3, _BS_BOTTOM, 0, 0, 0
```

```
    Statictext #demo, "_BS_BOTTOM, 0, 0, 0", 300, 165, 260, 260
    Button #demo.bttn4, "Text@Top", Click, UL, 300, 230, 120, 80
    Stylebits #demo.bttn4, _BS_TOP, 0, 0, 0
    Statictext #demo, "_BS_TOP, 0, 0, 0", 300, 315, 260, 60

    msg$ = "See API Corner - Easy BMPButtons by Alyce Watson " + _

"in the Liberty BASIC Newsletter Issue #123 to learn how the " + _
        "Stylebit _BS_BITMAP is used to create a more Windows " + _
        "conforming look to your buttons. "
    StaticText #demo.txt5, msg$, 20, 370, 560, 120

    Open "Button Formatting with Stylebits" for Window as #demo
    #demo "Trapclose EndDemo"
    #demo "Font Times_New_Roman 14 Bold"
    #demo.bttn4 "!Setfocus"
    Wait

Sub EndDemo handle$
    Close #demo
    End
End Sub

Sub Click handle$
    nButton$ = Right$(handle$, 1)
    Notice "Button #";nButton$;" Clicked"
End Sub
```

Stylebits and API Calls

Stylebits allow buttons to display images, but only when accompanied by an API call to user32.dll (SendMessageA). For an indepth explanation of placing images on button controls, see API Corner - Easy BmpButtons (Liberty BASIC Newsletter, May, 2004) by Alyce Watson. In this referenced article, Alyce also explains how using Stylebits with BmpButtons preserves the intended recessed look of the button when clicked, rather than the inverse color effect on the non-Stylebits native Liberty Basic BmpButton. If you use the _BS_BITMAP stylebit, be sure to include the height and width of the bitmap as the height and width of the button. Otherwise, the button size defaults to that of the null character of the chosen font. As with statictext, you can combine _BS_BITMAP with border styles to achieve interesting effects. Try making colorful buttons in your program without having to preload bitmap images.

```
' Demonstrates colorful buttons without first
' having to load bitmaps
    WindowWidth = 600
```

```
    WindowHeight = 560
    UpperLeftX = Int((DisplayWidth-WindowWidth)/2)
    UpperLeftY = Int((DisplayHeight-WindowHeight)/3)

    Nomainwin

    Open "Quick Draw" for Graphics as #1
    #1 "Trapclose EndDemo"
    #1 "Down"
    hBmp1 = solidButton()
    hBmp2 = varigatedButton()
    hBmp3 = stripedButton()
    Close #1

    Button #demo.b1, "", Click, UL, 20, 20, 50, 50
    Stylebits #demo.b1, _BS_BITMAP, 0,
 _WS_EX_CLIENTEDGE OR _WS_EX_DLGMODALFRAME, 0
    Button #demo.b2, "", Click, UL, 20, 100, 50, 50
    Stylebits #demo.b2, _BS_BITMAP OR _WS_DLGFRAME, 0, 0, 0
    Button #demo.b3, "", Click, UL, 20, 180, 50, 50
    Stylebits #demo.b3, _BS_BITMAP OR _WS_BORDER, 0, 0, 0

    Open "Colorful Buttons" for Window as #demo
    #demo "Trapclose EndDemo"
    hButton1 = hWnd(#demo.b1)
    Call CreateBMPButton hButton1, hBmp1
    hButton2 = hWnd(#demo.b2)
    Call CreateBMPButton hButton2, hBmp2
    hButton3 = hWnd(#demo.b3)
    Call CreateBMPButton hButton3, hBmp3
    Wait

Sub EndDemo handle$
    Close #demo
    End
End Sub

Sub Click handle$
    nButton$ = Right$(handle$, 1)
    Notice "Button #";nButton$;" Clicked"
End Sub


Sub CreateBMPButton hButton, hBmp
    CallDLL #user32, "SendMessageA",_
        hButton As uLong, _
```

```
        _BM_SETIMAGE As Long,_
        _IMAGE_BITMAP as long,_
        hBmp As uLong,_
        re As Long
End Sub

Function solidButton()
    hue$ = "128 0 128"
    #1 "Backcolor ";hue$
    #1 "Color ";hue$
    #1 "Place 0 0"
    #1 "Boxfilled 50 50"
    #1 "Getbmp button1 1 1 50 50"
    solidButton = hBmp("button1")
End Function

Function varigatedButton()
    c1 = 51: c2 = 101
    blueHue = 232
    For i = 1 to 25
        #1 "Color 0 0 ";blueHue
        #1 "Place ";c1;" ";c1
        #1 "Box ";c2;" ";c2
        c1 = c1+1: c2 = c2-1
        blueHue = blueHue-8
    Next i
    #1 "Flush"
    #1 "Getbmp button2 51 51 50 50"
    varigatedButton = hBmp("button2")
End Function

Function stripedButton()
    hueStrand$ = "Red White Blue White Red"
    For i = 0 to 4
        hue$ = Word$(hueStrand$, i+1)
        #1, "Color ";hue$
        For j = 1 to 10
            #1, "Line ";50+i*10+j;" 0 ";50+i*10+j;" 50"
        Next j
    Next i
    #1 "Flush"
    #1 "Getbmp button3 50 0 50 50"
    stripedButton = hBmp("button3")
End Function
```

These are just some examples of what you can do with stylebits and buttons. With experimentation, you

may find more.

A List of Stylebits

You can get a list of all dwStyles and dwExStyles available with the Stylebits command at the MSDN Library - Button Styles.