Push Buttons

-
Alyce
from Liberty BASIC 4 Companion

Button

A push button is a regular button. The user clicks on it to interact with a program. The button on the right in the image below is a push button.

As with all other controls, buttons must be listed in the code before the window is opened. There are two kinds of buttons in Liberty BASIC, (regular) push buttons and BMPBUTTONS. The command to create push buttons must begin with the window handle, then a dot and a unique extension. Although the use of an extension is optional, omitting it isn't recommended, since that limits the program's ability to send commands to the button. In the following example, the window handle is "#1," and the extension for this button is ".button1" Here is the format to include a button:

```
BUTTON #1.button1, "I'm a Button!", [branchLabel], UL, _
    xpos, ypos, width, height
or
BUTTON #1.button1, "I'm a Button!", subName, UL, _
    xpos, ypos, width, height
```

The label that appears on the button is written between the quotation marks.

After the button label, the button command must specify the branch label or sub name in the program where execution should continue when the button is pushed by the user. [branchLabel] or subName

The UL designates a corner of the client area. (Client area is the area of the window that does not include the titlebar, menu or frame - the workspace of the window.) UL stands for upper left. The button appears at a distance specified from the upper left corner. The other possibilities are UR (upper right,) LL (lower left,) and LR (lower right.)

The next number is the X position in relationship to the corner specified. The following number is the Y position.

The last two numbers are optional. Use them to specify the desired width and height of the button for a uniform look to the buttons in a window. If the program does not specify dimensions, the button is sized automatically to the correct width and height for the label and font size chosen. If the font size of a button is changed during program execution, it may be necessary to change the size of the button also with the LOCATE command, since Liberty BASIC does not automatically resize the button.

Default Button

To create a button in a dialog window that will be activated when the user pushes ENTER, give it the extension DEFAULT as in this example:

```
button #1.default, "OK",[do.it],UL,10,300
```

Non-literals in Button Statements

Expressions and variables are also acceptable as parameters in the BUTTON command. Some examples follow.

```
buttonwidth = 2*60

button #1.1, "Open",[open],UL,10,20,buttonwidth,40
button #1.1, "Edit",[edit],UL,200,20,buttonwidth,40

width = 60
height = 25
xOrigin = 10
yOrigin = 10
label$ = "Button"
bitmapFile$ = "run.bmp"
button #main.sized, label$+" Label!", [click], UL,_
    xOrigin+60, yOrigin, width*3, height
bmpbutton #main.run, bitmapFile$, [run], UL, xOrigin, yOrigin
bmpbutton #main.bug, "bug.bmp", [bug], UL, xOrigin, yOrigin * 4
```

EVENT HANDLERS

Both branch labels and subs can be used to handle button events. If a sub is used, the handle of the button is passed into the sub by Liberty BASIC. It is passed in as a string variable. Refer to it by placing the #

character in front of this handle variable, as in the demo below:

```
button #1.b, "Click Me",clickMe,UL,10,10
button #1.c, "Pick Me",[pick],UL,10,60
open "duh" for window as #1

wait

[pick]
'enable the other button
#1.b "!enable"
wait

Sub clickMe handle$
'send a command to this button
'using handle variable
#handle$ "!disable"
End Sub
```

COLORS

The background color of a button is a system color. It cannot be changed by Liberty BASIC. The user's default system color for this is called BUTTONFACE by Liberty BASIC. It is not possible to change the color of the text displayed on the button.

LABEL

The label of a push button can be changed in the course of running the program. Bmpbuttons do not have labels. To change the label on a button, use this command:

```
print #1.button1, "Changed!"

or

caption$="Changed!"
print #1.button1, caption$
```

After the command is carried out, the button caption displays the text supplied in the PRINT command. (Changed!)

FONTS

The default font for controls is Ms Sans Serif 8 point. This is the same font most Windows applications use. A program may change the font that is displayed on buttons. Bmpbuttons cannot accept a FONT command.

```
print #window.bttn, "!font facename width height [attributes]"
```

First designate the font name, then the desired width and height of the font characters in pixels. If the command designates a value of 0 for the width, Liberty BASIC automatically supplies the default width for the chosen font. If a value (or 0) is given for width, the font is sized in pixels. If there is no width parameter specified, the font is sized in points. There are 72 points in an inch.

Possible attributes are bold, underscore, italic and strikeout. They may be used in any combination and listed in any order. Adding attributes is completely optional.

If the program cannot match the font exactly, it provides the closest font matching the one specified. If the desired font name contains blank spaces, you must use an underscore character in place of the spaces when specifying the face name. Both facename and attributes are case insensitive, so "Arial" is the same as "arial" and "ARIAL." "BOLD" is the same as "bolD" and "Bold." Here's the button font command as it appears in a program:

```
print #1.button1, "!font Times_New_Roman 0 20 bold"
```

SETFOCUS

To cause a button to receive the input focus, issue a !SETFOCUS command:

```
print #1.button1, "!setfocus"
```

LOCATE

It is possible to move or resize a button during program execution with the LOCATE statement. After the locate command is sent to the button, always send a REFRESH command to the window itself, so that the screen is redrawn to reflect the change. The format is:

```
print #1.button1, "locate X Y Width Height"
print #1, "refresh"
```

In a program, it looks like this:

```
print #1.button1, "locate 20 400 100 30"
print #1.button2, "locate 20 400 ";buttonwidth;" ";buttonheight
print #1, "refresh"
```

SHOW/HIDE

To hide a button or show a button:

```
print #1.button1, "!show"
print #1.button2, "!hide"
```

ENABLE/DISABLE

A button can be disabled, so that it appears grayed-out and the user cannot click it. It can be enabled again with the ENABLE command:

```
print #1.button1, "!enable"
print #1.button2, "!disable"
```