FrameButton 1.2

by: Chip Burkhead

Introduction

I remember last summer when I took a Macintosh development class, the instructor said that almost all Mac programmers at some time (when they know enough to figure out how to do it), write a procedure to put a graphic frame around the OK button (or whatever the default button should be). At the time I thought that this didn't apply to me because I didn't see myself ever wanting to write such a thing. Well, here is my button framing routine. I guess you can't always be sure of what you might or might not do in the future. I wrote two previous versions of this routine, one that only worked on dialogs and the other that added window support. This version will handle those as well as allowing you to erase the frame when desired. Note: if most of this text seems familiar, it is. I hate documenting things so I re-use pre-written text as much as possible.

What it works on

This routine is meant for use with dialogs and windows. It can be used with modal or modeless dialogs. If you do not know the differences between these types of dialogs, I suggest the dialogs section in volume 2 of Macintosh Revealed for an explanation.

How it works

This is actually a pretty simple routine. When called it looks at the dialog item number and the control reference number that you give it. It then determines if the item is in fact a button and in the case your control is in a window whether the control reference number exists. If so, it finds out how big the button is (it calculates the slope). The slope is then put into a formula that will determine (more or less) how round the corners of the frame need to be. Do not ask how this formula was derived, I didn't do it, my mathematically gifted housemate is responsible for it. Then it builds the rectangle for the frame based on the button rectangle, and finally draws the frame. The frame is drawn in basic black. For you people with color you can probably modify the code to draw the frame in any color. Since I do most of my work on Mac Plus' and SE's (Yes, there are a few of us left.) I didn't worry about color. If you specify FALSE as the last parmeter then the frame is drawn, but this is meant to erase a pre-existing frame. If you just need to frame a button then put TRUE in as the last parameter.

How you make it work

When you call this routine all it needs is the DialogPtr or WindowPtr of the dialog/window with the button you wish to frame, the control reference number of the actually control or the item number for the button. Sound simple? Well, there are a couple of other things you need to do to get the desired results. In regards to the item number or control reference number, you only need one of these to get this to work. The control reference number is for dealing with windows. I needed a way to figure out what control to frame so you need to assign a unique number to the button(s) you will want to frame. This is just like assigning a reference number to a window. There is a field in the control record for this information. The item number is for dialogs. It is simply the button to frame. Every time you call this routine one of these numbers should be 0 (zero).

EX:

```
    FrameButton(dialog, 0, OK, TRUE);  (This is a call for a dialog)
    FrameButton(wind, 1000, 0, TRUE); (This is a call for a window)
```

Also you want to call FrameButton after you make a ShowWindow call (assuming that the dialog/window is invisible) and after a DrawControls call for windows. If you take this code and put it in a library or another project that you can include in other programs you write, you will want to make sure the number you pass for the control reference number is a long integer. I have found this particularly important when making a dialog call. Just putting 0 for the control reference number won't cut it, you need 0L or something similar.

Also, you need to consider drawing it again if a DA or another window covers all or part of the frame. In a modal dialog this is not a concern (unless you are doing something really strange). In a modeless dialog make a call to FrameButton either before or after DialogSelect (after you have determined that the event is related to your dialog with IsDialogEvent). You need only call Frame Button if the event is an Update or an Activate, all else shouldn't affect the frame. For windows I suggest calling FrameButton anytime you get an update or activate for the window, it makes things easier.

Conclusion

I hope that you find this routine useful in your own programs. I am asking no money for it since it wasn't that hard to write; and since I wouldn't know what to charge for something this small. You may distribute it as freely as you wish. If you upload it to another BBS or modify the code all I ask is that you leave my name in the header so I can be recognized as the original author. If you find something that needs to be changed, have an enhancement, or have questions, please send me some E-mail on GEnie. My address is C.BURKHEAD.