You can modify the appearance and function of buttons with the STYLEBITS command. The syntax is as follows:

stylebits #win.button, addbits, removebits, addexbits, removeexbits

This tip explains how to put bits into the addbits argument to modify buttons.

_BS_DEFPUSHBUTTON
This creates a button similar to a regular button, but with a heavier border. In a dialog window, the user can activate this button by pressing the ENTER key.
Example:

```
stylebits #win.button, _BS_DEFPUSHBUTTON,0,0,0
button #win.button, "Default Button", [quit],UL,10,10,120,30
open "test" for dialog as #win
#win "trapclose [quit]"
wait
[quit] close #win:end
```

_BS_BOTTOM, _BS_TOP, _BS_LEFT, and _BS_RIGHT
These styles create a button with text aligned to the specified direction.
Example:

```
stylebits #win.button, _BS_BOTTOM,0,0,0
button #win.button, "Bottom Text", [quit],UL,10,10,120,80
open "test" for window as #win
#win "trapclose [quit]"
wait
[quit] close #win:end
```

_BS_FLAT
This creates a non-3D button.
Example:

```
stylebits #win.button, _BS_FLAT,0,0,0
button #win.button, "Flat Appearance", [quit],UL,10,10,120,30
open "test" for window as #win
#win "trapclose [quit]"
wait
[quit] close #win:end
```

_BS_PUSHLIKE
This style modifies a checkbox or radiobutton so that it looks like a regular button. It looks normal when it

is unchecked, and depressed when it is checked.
Example for checkbox:

```
stylebits #win.button, _BS_PUSHLIKE,0,0,0
checkbox #win.button, "Pushlike", [set], [reset], 5, 5, 75, 30
open "test" for window as #win
#win "trapclose [quit]"
wait
[quit] close #win:end

[set]
print "set"
wait
[reset]
print "reset"
wait
```

Example for radiobutton:

```
stylebits #win.button1, _BS_PUSHLIKE,0,0,0
stylebits #win.button2, _BS_PUSHLIKE,0,0,0
radiobutton #win.button1, "Pushlike", [set], [set], 5, 5, 75, 30
radiobutton #win.button2, "Pushlike", [set], [set], 80, 5, 75, 30
button #win.check, "Check State",[check],UL,10,100
open "test" for window as #win
#win "trapclose [quit]"
wait
[quit] close #win:end

[set]
wait

[check]
#win.button1 "value? res$"
if res$ = "set" then
print "button 1 set"
else
print "button 1 not set"
end if

#win.button2 "value? res$"
if res$ = "set" then
print "button 2 set"
```

```
else
print "button 2 not set"
end if
wait
```

_BS_MULTILINE
If the text is too long to fit on one line, it will form more lines as necessary. See multilinebutton.