U5. Colors

UtilIt supports several commands to help programmers get and set colors, plus a single command to switch the color palette associated with a program. The need for help with getting and setting colors stems from the fact that the color environments available on the Mac range from B & W w/o Color QuickDraw to the latest version of 32-bit QuickDraw running on 24-bit deep screens. By using UtilIt commands to get and set colors, drawing can be done without needing to be aware of the current color environment.

GetFgC 178 a,b,c,d,uRGB,uResult,uMenuID, uMenuItem,ul2

GetBkC 179 a,b,c,d,uRGB,uResult,uMenuID, uMenuItem,ul2

Returns the foreground, GetFgC, or background, GetBkC, color from the designated port or color table. When getting a color in a color table, the only difference between GetFgC and GetBkC is that the former returns black and the latter white when a color cannot be found. All forms of GetFgC and GetBkC work in all color & non-color environments.

◦ if b = 0 then...
a = source window or port
0 = front modal or active modeless window
1 = current port
other = WindowPtr or GrafPtr
c & d are not used (pass 0)

◦ if b = -1 then...
c = clut ID or handle
d = clut index (1-based)
a is not used (pass 0)

◦ if b = -2 then...
c = clut ID or handle
d = clut value (partID)
a is not used (pass 0)

where the unusual use of parameter b to determine the type of action follows that seen in the SetFgC/BkC commands.

For both color and non-color ports, the color is returned in both uRGB and in the three 4-byte integer variables uResult, uMenuID, and uMenuItem, where the latter either contain an old-style color constant in uResult and -1 in uMenuID and uMenuItem, or the three RGB color components in the low word of each variable.

When getting a color by index from a clut, ul2 returns the entry's partID. When getting a color by partID, and the partID cannot be found in the clut, then UtilIt will get the color from clut 1111 which contains a copy of the standard control colors.

UtilIt includes clut 1110 which contains the 8 old-style colors as RGB colors with part IDs equal to the old-style color constants. This facilitates finding the RGB equivalent of an old-style color by calling GetFgC with b = -2, c = 1110, and d = old-style color constant (= the partID).

SetFgC 180 a,b,c,d,uRGB,uResult

SetBkC 181 a,b,c,d,uRGB,uResult

Resets the foreground, SetFgC, or background, SetBkC, color-related fields of the designated port's (C)GrafPort record, where parameters b, c, and d specify the color. All forms of SetFgC and SetBkC will work in all color and non-color environments. Black (foreground) or white (background) is used as the default color if the specified color is not supported in the current environment. If the port's color is changed, UtilIt returns uResult ≠ 0.

a = target window or port
0 = front modal or active modeless window

1 = current port
other = WindowPtr or GrafPtr
◦ if b = c = d = -1 then...
uRGB is used as source of color
◦ if b = -1 then...
c = clut ID or handle
d = clut index (1-based)
◦ if b = -2 then...
c = clut ID or handle
d = clut value (partID)
◦ if b > 0 and c = d = -1 then...
b = address of RBG color OR old-style color constant:
33 = black, 30 = white, 205 = red, 341 = green,
409 = blue, 273 = cyan, 137 = magenta, 69 = yellow
◦ if b > 0, c > 0, d > 0 then...
b = red component (in low word, 0 to 65535)
c = green component (in low word, 0 to 65535)
d = blue component (in low word, 0 to 65535)
where the unusual use of parameter b arose from efforts to maintain backward compatibility with older versions.
When searching for a "partID" (typically used by control drivers to set control part colors), and the partID cannot be found in the designated clut, then UtilIt will try getting the color from clut 1111 which contains a copy of the standard control colors. This makes it easy for control drivers to set control part colors without having to worry about the color environment in use.
To help understand the use of b, c, and d, consider the following examples which all specify the color white:
b = c = d = -1, uRGB.red = green = blue = $FFFF
b = -1, c = 1111, d = 2 (the 2nd color in clut 1111)
b = 30, c = d = $FFFFFFFF = -1
b = c = d = $0000FFFF = 65535
NOTE: *When creating complex pictures or pixmaps with thousands of different colors, it is usually not advisable to use SetFgC or SetBkC to set colors since they are more time-consuming than the corresponding toolbox calls.*

SetPal2 185 a,b,c,d,uResult

Resets the current program-wide color palette or palette characteristics according to the parameters a, b, c, and d. SetPal2 is ignored if Color QuickDraw is not supported, and uResult returns a negative value if an error occurs.
a = scope or type of palette change
-1 = apply to all palette entries
0 to 255 = palette entry number
> 255 = clut handle in memory
< -1 = - ID of clut resource or System color table
b = palette entry usage
0 = pmCourteous
2 = pmTolerant
4 = pmAnimated
8 = pmExplicit
10 = pmTolerant + pmExplicit (requires 32-bit QD)
12 = pmAnimated + pmExplicit (requires 32-bit QD)
c = palette entry tolerance
d = target palette
0 = the program-wide palette
other = a palette handle

If the scope of changes applies to the entire palette (a < 0 or a > 255), then all visible windows in all screens are erased and invalidated. In cases where your program keeps control after calling SetPal2, you may want to call DoUpdt2 to force FaceIt to redraw all of the invalidated windows.