Inhibited Colors

The **Palette Manager** recognizes six inhibited usage categories that give you control of which palette entries can and cannot appear on depths of 2, 4, and 8 bits per pixel, on color or gray-scale devices. The categories are specified using these constants:

pmInhibitG2	inhibit on 2-bit gray-scale device
pmInhibitC2	inhibit on 2-bit color device
pmInhibitG4	inhibit on 4-bit gray-scale device
pmInhibitC4	inhibit on 4-bit color device
pmInhibitG8	inhibit on 8-bit gray-scale device
pmInhibitC8	inhibit on 8-bit color device

Here is an example of how these categories can be combined.

myColor8Usage = **SetEntryUsage**(myPalHandle, 300, pmAnimated + pmExplicit +pmInhibitG2 + pmInhibitC2 + pmInhibitG4 + pmInhibitC4 + pmInhibitG8,0);

This sets the usage of entry 300 of the palette specified by myPalHandle to the combined usages of animated and explicit, to be allocated only on color 8-bit devices. (Since 300 is greater than 255, the highest index on an 8-bit device, the index associated with that entry wraps around to 44.)

You should always inhibit tolerant colors on gray-scale devices. **Color QuickDraw** now allows luminance mapping on gray-scale devices. The default CLUT on a gray-scale device is an evenly spaced gray ramp from black to white. Since this is usually the best possible spread on a gray-scale device, you could specify all three inhibited gray-scale categories.

As another example, on a 4-bit device you might want to allocate 14 tolerant colors, while on an 8-bit device there are sufficient indices that you can also use a number of animated colors. By inhibiting the animated entries on 4-bit devices, you ensure that your 14 tolerant colors are allocated. Merely sequencing the palette does not solve this problem, because the animated colors always take precedence over the tolerant colors. (See **Assigning Colors to a Palette** under the section entitled **Creating Palettes** for more information about sequencing.)

Combined Usage Categories

The inhibited usage category is always combined with some other usage category. In addition, the explicit usage category can be combined with the tolerant and animated categories.

The main purpose for using explicit colors is to provide a convenient interface to color table indices. You can select any of these colors for drawing by setting your window's palette to contain as many explicit colors as are in the target device with the greatest number of indices. **PmForeColor** configures the color grafPort to draw with the index of your choice. So that you can easily create effective explicit palettes, two color usage categories can be combined: pmTolerant + pmExplicit and pmAnimated + pmExplicit.

The pmTolerant + pmExplicit combined usage means that you get the color you want at the index you want, across all devices that the window touches. As with pmTolerant, other windows may use those colors in their displays.

The pmAnimated + pmExplicit combined usage means that you get the color you want at the index you want, across all devices that intersect the window, but windows that do not share the palette cannot use that index. The entry can be animated by a call to the **AnimateEntry** procedure.

Since the value of an explicit entry is treated as the entry modulo the bit depth, index collisions can occur between entries of the same usage within a palette. In this case, the lower-numbered entry gets the index. For example, if palette entries 1 and 17 were both pmAnimated + pmExplicit, then on a 4-bit screen, entry 1 would get index 1, and entry 17, although it wraps around to 1, would get nothing.

Unallocated pmTolerant + pmExplicit colors revert to pmTolerant.
Unallocated pmAnimated + pmExplicit colors revert to pmCourteous.