About the Graphics Overview

This topic presents an overview of Macintosh graphics. It surveys the components and processes of **Color QuickDraw**, the other graphics managers with which it interacts, and how colors flow from your application to the screen.

Read the information given here if your application uses color or gray scales. To use the **Graphics Overview**, you should have a basic understanding of the original **QuickDraw**. The next figure below charts your path into the world of Macintosh graphics.

Macintosh graphics begin with **QuickDraw** and end with glowing phosphor. The early Macintosh systems, with their built-in screens and integral graphics hardware, made well-defined and comparatively limited demands on your understanding of graphics: if you learned **QuickDraw**, you were set.

The Macintosh II computer introduced two features that greatly increased graphics capabilities: slots and color. **Slots** allow the addition of specialized hardware to the system. (In the original Macintosh II, the graphics hardware was always added in NuBus standard slots. Current systems allow a number of hardware expansion modes in addition to NuBus slots.)

With slots, the characteristics of the output device can vary from machine to machine. With color, the extent of variation can be very great: screens not only can vary in horizontal and vertical dimensions, but they can also vary in depth. To the single-bit-per-pixel depth of the original Macintosh systems, the Macintosh II added pixel depths of 2, 4, and 8 bits. Output devices range from black-and-white systems to cards and screens capable of presenting hundreds of colors from palettes of millions. Furthermore, users can combine screens: a user may move your application's window so that it overlaps screens of very different characteristics.

To remove the burden of worrying about output devices (for most applications), **Color QuickDraw** is device-independent. Applications using color can work in an abstract color space defined by three axes of red, green, and blue (RGB). Your application can specify a color as an RGB value, in which each component is defined as a 16-bit integer. **Color QuickDraw** compares such a 48-bit value with the colors actually available on the hardware at execution time and chooses the closest match. Precolor applications and those not concerned with color or gray-scale graphics need not change, and those concerned only with straightforward color usage can ignore the problems of output devices.

When the Macintosh II was introduced, the maximum pixel value was limited to a single byte. Each pixel's byte can specify one of 256 (2^8) different values, and, rather than simply truncating the least significant bits of each component to get a color, **Color QuickDraw** treats such pixel values as indexes into a color table. If your application asks for a 48-bit RGB color, the **Color Manager** examines the colors available in the card. If the video device supports 8 bits per pixel, the card contains a color look-up table (CLUT) with 256 entries, each entry an RGB value. The **Color Manager** determines which RGB value is closest to the requested color and tells **Color QuickDraw** what the index for that color is.

Storage and movement of such indexed color values require a maximum of 8

bits, rather than 48, saving space and time. (The RAM needed to hold a 640-by-480 pixel screen at 8 bits per pixel is about 300,000 bytes.) And because the table is variable-it can be loaded with different colors-applications can display up to 16 million colors, although only 256 different colors can appear at once.

With the addition of direct pixel values, first made available with system software version 6.0.5, the 256 simultaneous color limit has been removed. **Color QuickDraw** can now also process **direct pixel** values, which use 16 or 32 bits to directly represent a color. Using direct color not only removes much of the complexity of the color table mechanism, but it also allows the display of thousands or millions of colors simultaneously, resulting in near-photographic realism.

The device-independence of **Color QuickDraw** is such that on a three-screen system-for example, with displays for gray-scale indexed pixels, color indexed pixels, and direct pixels-the user can move your window to span all three devices and each will show its best representation of your image.

This overview introduces **Color QuickDraw** and the related graphics managers with which it works.

- **Color QuickDraw** calls upon the **Color Manager** to map color requests to the actual colors available. Most applications never need to call the **Color Manager** directly.
- The **Palette Manager** allows your application to specify the set of colors that it needs on a window-by-window basis, and makes the colors available (within application-determined ranges) in a graceful manner.
- The Color Picker Package allows your application to solicit a color choice from the user in a standard way.
- The Picture Utilities Package provides routines with which your application can extract information, such as pixel depth and colors used, from pixel maps and pictures.
- The **Graphics Devices Manager** offers routines for preparing images offscreen, and it manages the data structures that track the characteristics of the graphics hardware of a particular system.
- The **Slot Manager** controls communication with expansion boards of all types, including video cards.

Other topics in the **Graphics Overview**

- introduce the basic graphics components and further defines the differences between indexed and direct pixel images
- present overviews of the important color graphics data structures: the color tables and palettes that hold colors, the pixel maps that hold information about images, the color grafPorts that hold information about windows, and the graphics device records that describe the capabilities of a particular screen

- describe the startup process as it applies to graphics, to show how the data structures are created and initialized
- trace the path of a user's request for a color through the graphics system and onto a screen, in both the indexed and direct pixel systems
- tell you how to determine which version of **QuickDraw** is actually running