

Chapter 8

All About Color Palettes

In this chapter, you'll learn about:

- ♦ Color spaces
- ♦ Color palettes
- ♦ Color palette organization
- ♦ Cross-platform color palette issues
- ♦ System palettes
- ♦ Platform specific palette peculiarities
- ♦ Planning color palettes
- ♦ Creating color palettes
- ♦ Color palette effects
- ♦ Tips for creating effective color palettes
- ♦ Color reduction

There are many elements that influence how an arcade game looks. Of these, color selection is one of the most important. Good color selections can make a game stand out aesthetically, make it more interesting, and enhance the overall perception of the game's quality. Conversely, bad color selection has the potential to make an otherwise good game seem unattractive, boring, and of poor quality.

The purpose of this chapter is to show you how to choose and implement color in your games. In addition, it provides tips and issues to consider during this process.

Color Space

As mentioned previously, color is a very subjective entity and is greatly influenced by the elements of light, culture, and psychology. In order to streamline the identification of color, there has to be an accurate and standardized way to specify and describe the perception of color. This is where the concept of *color space* comes in. A color space is a scientific model that allows us to organize colors along a set of axes so they can be easily communicated between various people, cultures, and more importantly for us, machines.

Computers use what is called the RGB (red-green-blue) color space to specify colors on their displays. The RGB color space describes color by exciting the red, green, and blue phosphors present in your computer monitor's CRT.

The RGB color space is usually visualized by using a 3D cube in which each color (i.e., red, green, and blue) is assigned an axis as shown in Figure 8-1.

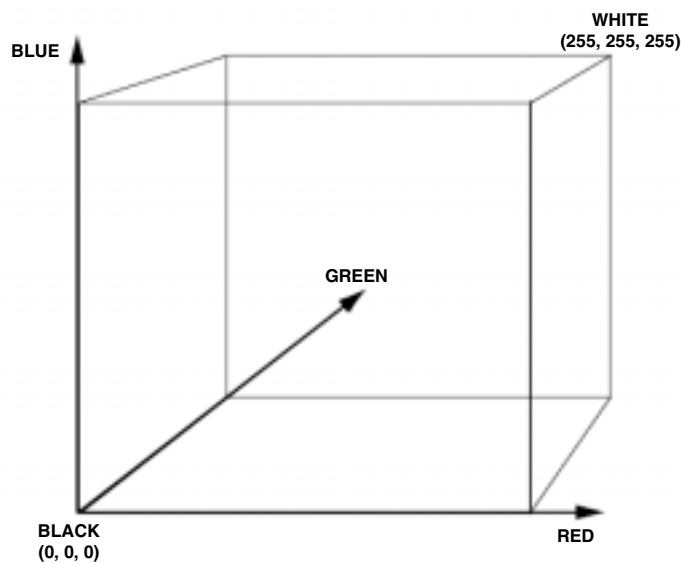


FIGURE 8-1: RGB Color Space

Using this model, three numbers or coordinates are used to identify specific color values. For example, to specify the color black, we would use the coordinates (0,0,0), which effectively means that we are setting all values of the red, green, and blue phosphors to zero. These coordinates are also called *RGB triplets*. A value of zero defines the lowest possible color range of the RGB triplet system but the upper range is actually determined by your computer's graphics hardware.

Every increase to an RGB value indicates one additional level of color brightness. This means, for example, that a computer that has graphics hardware that can support an RGB color range from 0-255 can produce up to 256 unique shades of brightness for a given hue.

HSV, or hue-saturation-value, is another type of computer color space. Under the HSV system, hues are specified using a color wheel as shown in Figure 8-2. The colors on the color wheel are separated by degrees (0 -360). Here, 60 = Yellow/Orange, 120 = Green, 180 = Light Blue/Cyan, 240 = Dark Blue, 300 = Magenta/Purple, and 0 = Red. Meanwhile, saturation is specified using a percentage scale (0-100%) where 0% is a gray value and 100% is full intensity. Value or color brightness is also specified using percentage scales, except in this case 0% represents black and 100% represents white.

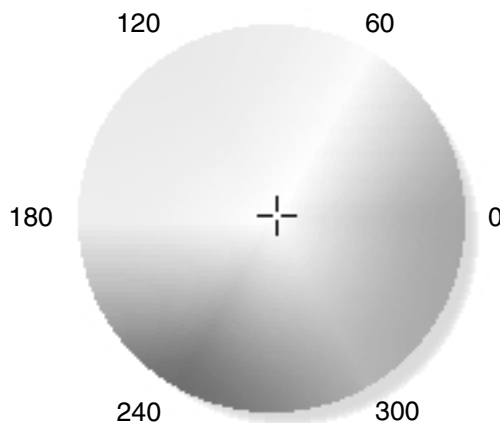


FIGURE 8-2: HSV Color Space

In case you are interested, there are other computer-based color spaces. These include:

- HSL, *Hue Saturation and Lightness*
- HSI, *Hue Saturation and Intensity*
- HSB, *Hue Saturation and Brightness*



NOTE: Most graphics programs allow you to specify colors using one or more of these color space systems. However, because the RGB scheme is the most common, all colors are specified using RGB values in this book.

Color Palettes

A color palette represents the maximum number of colors that can be produced by using all three combinations of red, green, and blue available in the RGB color space. As mentioned in Chapter 2, color palettes come in two flavors, *physical* and *logical*.



Physical palettes contain all of the colors supported by the system's graphics hardware, while logical palettes contain only a fraction of the colors that are available in the physical palette.

When we design game graphics, we select colors from the system's physical color palette but actually render objects using the colors present in the logical palette. Figure 8-3 illustrates the relationship between the two.

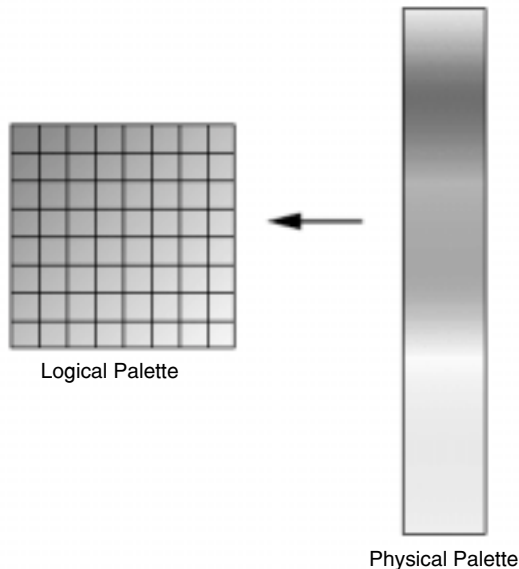


FIGURE 8-3: Physical vs. Logical Color Palette



NOTE: For all intents and purposes, the term *color palette* is really synonymous with a logical palette. Therefore, to make future references more consistent, the term color palette will be used whenever a reference is made to a logical palette for the remainder of the book.

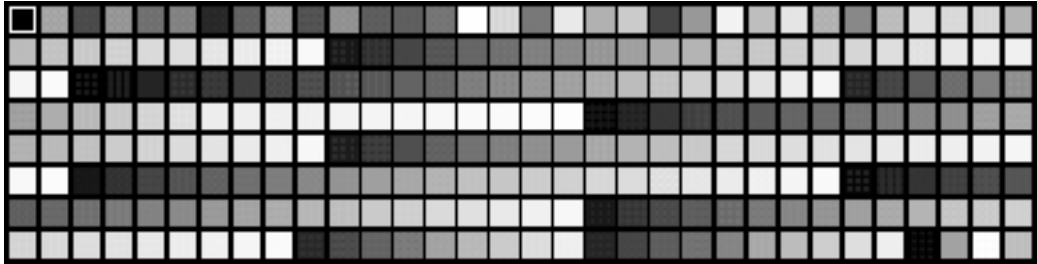


FIGURE 8-4: Color Palette Example

Color palettes are best thought of as “windows” into a larger world of color. Much like a painter’s palette, they are essentially containers that hold the colors with which we paint and draw objects on the screen. In order to make the lives of developers easier, color palettes were developed as a means of managing and specifying small groups of colors within the hardware color space.

As you might expect, color palettes come in different sizes. Larger color palettes can hold more colors and in turn allow graphic objects to be rendered with more color fidelity. Meanwhile, smaller color palettes hold fewer colors and place restrictions on the amount of colors that a given image can contain.

The actual size of a color palette is always determined by the hardware capabilities of the system. However, it’s unusual to find color palettes with more than 256 colors in them since color management quickly becomes unwieldy.

Because they are so convenient for managing and specifying colors, color palettes are ideally suited for all types of arcade graphics development. This stems from the fact that they offer game developers a number of distinct advantages, including:

- **Universal compatibility with all display modes**—Color palettes, especially those that use 16 or 256 colors, are a common denominator among all display modes. For example, images that use 16 or 256 colors are guaranteed to display properly even when shown on display modes that use thousands or even millions of colors. However, the same is not true the other way around.
- **High degree of compression**—Images that use color palettes with small amounts of color (i.e., less than 256) tend to require much less disk space than those that contain thousands or millions of colors.
- **Cross-platform compatibility**—In this day and age, it can be assumed that virtually all platforms can display images with 16 or 256 colors in them. This makes it relatively easy to port palette-based artwork between different hardware platforms. The same can’t be said for images that were created in display modes that support thousands or millions of colors as these display modes are typically only found on higher-end computers manufactured in the last two or three years.

- **Ease of manipulation**—Color palettes are relatively easy to manage and manipulate from both a creative and technical perspective. This isn't true for display modes that use thousands or millions of colors, since direct color access in these modes tends to be much more cumbersome.
- **Good color rendition**—Color palettes can provide a sufficient level of color fidelity for most arcade style games. This makes them useful for rendering most types of objects and images that are often featured in these types of games.
- **Support for special effects**—Color palettes can support powerful special effects such as color cycling and color fades. These effects aren't easily achieved in display modes with thousands or millions of colors without special programming tricks.

NOTE: All of the programs described in Chapter 6 provide the ability to easily manipulate color palettes.

Table 8-1 outlines some common color palette sizes supported by different computer platforms:

TABLE 8-1: Color Palette Sizes by Platform

| Platform | Color Palette Size | Comments |
|-----------|--------------------|---|
| DOS | 16 or 256 | <p>The actual display mode determines the number of available colors in the color palette.</p> <p>Sixteen colors are typically used in older EGA and MCGA display modes. Although these display modes are backward compatible with most modern graphics cards, very few games are written specifically for 16 colors anymore. Virtually all DOS games created since 1991 are written for display modes capable of displaying 256 colors in their color palette.</p> |
| Linux | 16 or 256 | <p>The actual display mode determines the number of available colors in the color palette.</p> <p>Sixteen colors are typically used in older VGA display modes. Sixteen colors are usually the minimum requirement under the various Linux desktop managers. Most Linux-based games use 256 colors or more.</p> |
| Macintosh | 256 | <p>Sixteen-color palettes were quite common on color Macintosh systems until the advent of the PowerMac series in 1994. At that time, game developers were encouraged to use the 256-color palette and the 16-color palette was deprecated.</p> |

| Platform | Color Palette Size | Comments |
|---------------------------------------|--------------------|---|
| Windows 3.1, 95, 98, NT 4.0, and 2000 | 16 or 256 | <p>The actual display mode determines the number of available colors in the color palette.</p> <p>Sixteen colors are typically used in older VGA display modes. Although these display modes are backward compatible with most modern graphics cards, very few games are written for them anymore.</p> <p>Virtually all Windows games since 1994 are written for display modes capable of displaying 256 colors in their color palette.</p> |

NOTE: You may be wondering why there have been no references made to color palettes with thousands or millions of colors in them. Well, they weren't mentioned because these display modes do not actually support color palettes. Rather, they have access to the entire physical palette. On the other hand, color palettes only have access to a relatively small selection of the colors available within the entire physical palette.

NOTE: Support for 16-color palettes has pretty much disappeared on all platforms, and both market forces and the development community at large have long discouraged development of games that use them. Therefore, they aren't covered in this book. Instead, if you design arcade games, it's strongly recommended that you only use 256-color palettes.

Color Palette Organization

To better facilitate the selection of colors, color palettes are organized as an array of *palette entries* or *color indexes*. These are markers that indicate the current position of a particular color within the palette along with its RGB color values.

Figure 8-5 illustrates this concept by showing how a color palette with eight colors might be organized.

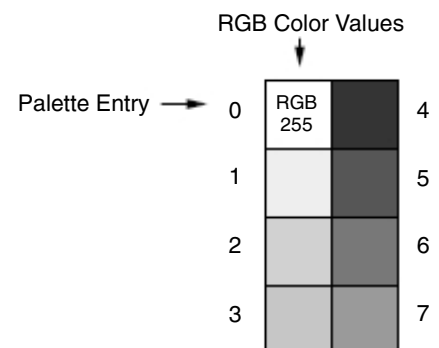


FIGURE 8-5: Color Palette Organization Example

NOTE: Palette entries are always specified starting at 0 and ending at the upper limit of the current palette size. So, for example, if a palette had 256 palette entries, its upper limit would be 255 and its palette entries would be numbered from 0 to 255.

NOTE: Programmers and programming literature often refer to palette entries as *palette registers* or *color registers*. These are just fancier names for the same thing.

It is very important that you understand how color palettes are organized because both graphics programs and programming tools specify individual colors this way. In addition, arrangement also determines the *palette order*, or the particular location of a given color within the palette. Both graphics programs and games come to rely on the palette order in order to determine where certain colors appear and display on the screen.

Although you can usually place colors using any order within a palette, relying on a specific palette order can influence how certain colors appear within an image. For example, if you create an image with the color black in it and then later change the position of black within the palette, all of the objects that should be black will immediately assume some other color value. As you can appreciate, such changes can change the whole character of your artwork.

To better illustrate this concept, consider the color palettes shown in Figure 8-6. Even though color palettes A and B both contain identical color values, they are not the same because their palette order is different. Thus, images that use these colors will look different from each other when displayed using either palette.

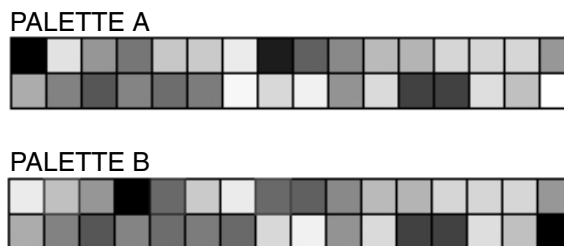


FIGURE 8-6: Color Palette Order

Cross-Platform Color Palette Issues

Each computer platform has its own quirks and peculiarities when it comes to how it supports and implements color palettes. These idiosyncrasies are due to differences in memory architecture and graphics hardware. Of these, one of the prime culprits is the implementation of the DAC chip. All modern computer graphics hardware uses a special chip called a *DAC*, or digital-to-analog converter, to generate the wide range of color supported by the RGB color space. On older, VGA-equipped systems the DAC circuitry is limited to supporting six bits, or 64 levels, of RGB color intensity. This scheme yields 262,144 possible colors (64x64x64) or 64 variations of red, green, and blue. On newer, SVGA-compatible

systems, the DAC circuitry is capable of generating eight bits, or 256 levels, of color intensity. This scheme yields 16,777,216 possible colors (256x256x256).

Please refer to Table 8-2 for more information on the maximum RGB triplet ranges supported by various computer platforms.

TABLE 8-2: RGB Color Value Range Comparisons

| <i>RGB Color Ranges</i> | <i>Maximum Possible Colors</i> | <i>Platforms Supported</i> | <i>Comments</i> |
|----------------------------------|--------------------------------|---|--|
| R = 0-1 G = 0-1 B = 0-1 | 16 | Computers that used CGA-compatible hardware typically used this color range. | Obsolete. The CGA color palette had two states: normal and intense. Half of its palette was simply a higher intensity version of the original color. |
| R = 0-3 G = 0-3 B = 0-3 | 64 | This color range was typically found on computers that used EGA (Enhanced Graphics Adapter) graphic modes (circa 1984). Some early 16-bit arcade game consoles were also limited to this color range. | Seldom used It's interesting to note that EGA-compatible display modes were quite common for DOS-based arcade games up until about 1993. The EGA hardware palette contains only very bright or very dark colors as its DAC chip was incapable of producing subtle shades of the same color. |
| R = 0-5 G = 0-5 B = 0-5 | 216 | Macintosh, Windows, Linux, and Java | Used extensively by Web browsers to render a cross-platform color palette of 216 colors common between the various versions of Windows, Linux, and the Macintosh. |
| R = 0-7 G = 0-7 B = 0-7 | 512 | The Atari ST line of computers (circa 1985) | Seldom used |
| R = 0-15 G = 0-15 B = 0-15 | 4,096 | The Commodore Amiga line (circa 1986) and the Apple IIgs series (circa 1986). Some early color laptop PCs and the later Atari STe/TT computer models also used it. | Seldom used |
| R = 0-31 G = 0-31 B = 0-31 | 32,768 | Some 16- and 32-bit video game systems occasionally used this color range. | Seldom used |

| <i>RGB Color Ranges</i> | <i>Maximum Possible Colors</i> | <i>Platforms Supported</i> | <i>Comments</i> |
|-------------------------------------|--------------------------------|---|--|
| R = 0-63 G = 0-63 B = 0-63 | 262,144 | DOS systems that use MCGA and VGA display hardware. Older versions of Windows using VGA hardware. The lamented Atari Falcon 030 multimedia PC also used this color range. | Very common but rapidly becoming obsolete. All VGA-compatible graphics cards use this color range since the technology was introduced in 1987. Virtually all games that run under DOS are limited to this color range as DOS has trouble “seeing” larger RGB color spaces without special software. |
| R = 0-255 G = 0-255 B = 0-255 | 16,777,216 | DOS with SVGA video hardware, Linux, the Macintosh, and all versions of Windows. | Very common and the current industry standard. The first popular computer to use this color range was the Macintosh II of 1987 vintage. Since that time it has become standard for all Macintosh models. However, it didn’t start becoming common on PCs until the introduction of SVGA-compatible graphics boards and Windows 3.1 in 1992. It can be safely assumed that all PCs shipped since 1996 can support this color range. |

The RGB color ranges supported by both VGA- and SVGA-compatible systems differ from each other only in terms of the color intensities they support. For example, an image created on a system that only supports 64 shades per color will appear slightly darker on a system that supports 256 shades per color. However, the basic color integrity will stay the same, i.e., brown still is brown regardless of whether the color was produced on a system using a 6-bit DAC or an 8-bit DAC.

To compensate for this effect, most graphics software automatically converts the color intensity levels of an image to suit the color capabilities of the currently active graphics chipset.

NOTE: Because of this fact, colors will be specified using a 0-255 RGB color range.

It’s interesting to point out that Windows, Macintosh, and Linux systems internally support the 8-bit DAC range, whether SVGA hardware is available or not.

Table 8-3 shows these differences in RGB color ranges between the most common computer platforms:

TABLE 8-3: RGB Color Value Range Compatibility by Platform

| <i>Platform</i> | <i>RGB Color Range</i> | <i>Supported DAC Resolution</i> | <i>Maximum Possible Colors</i> |
|----------------------------|-------------------------------------|---------------------------------|--------------------------------|
| DOS* | R = 0-63 G = 0-63 B = 0-63 | 6 bits or 8 bits* | 262,144 or 16,777,216* |
| Java | R = 0-255 G = 0-255 B = 0-255 | 8 bits | 16,777,216 |
| Linux+ | R = 0-63+ G = 0-63 B = 0-63 | 6 bits or 8 bits+ | 262,144 or 16,777,216+ |
| Macintosh | R = 0-255 G = 0-255 B = 0-255 | 8 bits | 16,777,216 |
| Windows 3.1/95/98/NT/2000. | R = 0-255 G = 0-255 B = 0-255 | 8 bits | 16,777,216 |

* Denotes that DOS can support 8-bit DAC values if SVGA hardware and the appropriate driver software are present.

+ Denotes that Linux and X Windows can support 6-bit DAC values if running on VGA-compatible hardware.



NOTE: Virtually all systems specify these color ranges starting at zero rather than one. Please remember this, as it'll come in handy when you start creating and editing your own color palettes.



NOTE: All versions of Windows will dither any colors that can't be properly reproduced in the current display mode. For example, this occurs when

you're specifying RGB values within the 0-255 color range on older hardware that's only capable of RGB color ranges of 0-63.

Even with all of this color capability, the average person cannot discern all of the possible colors contained within the RGB space or physical color palette. So, while it's theoretically possible for some platforms to support as many as 16.7 million different RGB values, the likelihood of the average person being able to distinguish between shade 14,056,922 and shade 14,056,923 is practically nil.

System Palettes

All computer platforms in common use have predefined 16- and 256-color palettes. These palettes are better known as *system palettes*. A system palette is the default color scheme that is activated whenever you first turn on or reboot a computer.

Overall, the system palettes across the different computer platforms tend to share much of the same color choices and arrangements. In addition, they all seem to use similar collections of complementary colors. This section describes the implementation of system palettes across the major computer platforms.

DOS System Palette


DOS doesn't have its own system palette per se; rather, it just uses the default color definitions found in the VGA and SVGA hardware's BIOS (basic input/output system) ROM.

The developers who selected and programmed the colors used in the VGA/SVGA palette may have been good engineers but they were definitely lousy designers. By and large, the VGA/SVGA system palette is totally useless for doing any serious graphics work. Here's a synopsis of why:

- One hundred and fifty-two of the 256 colors present in the VGA/SVGA palette lack significant intensity. They are all dark or dim shades and lack sufficient contrast to use them to shade in most types of objects.
- Seventy-two of the 256 colors present in the VGA/SVGA palette are merely rehashes of primary and secondary colors. None of them are particularly good color choices and few have any practical application in arcade game graphics design.
- With the exception of the first 32 palette entries, which do provide some useful shades of gray, there's not sufficient utility or variation in the colors provided.

- All but the first 16 palette entries have color ranges that are too small to produce accurate shading and gradient effects needed to properly render most game objects.


As you can probably gather, the color palette definitions of the DOS system palette are largely unsuitable for serious game graphics work. Therefore, it is strongly recommended that you always take the time to redefine the color palette when working under DOS for game projects.



NOTE: If you're interested in seeing the RGB values for these palettes, they can be found on the book's accompanying CD-ROM under the `PALETTES` directory in the files labeled `DOS16.PAL` and `DOS256.PAL`. These files are in the Microsoft .PAL format and are compatible with *Paintshop Pro* and other programs.

Windows System Palette

All versions of Windows from version 3.1 through Windows 2000 share a common system palette. This palette appears the same regardless of the particular graphics hardware that is installed on the machine on which Windows is running.



NOTE: The Windows 256-color system palette is only available on SVGA hardware as the standard. The VGA hardware specification only provides 16 colors at the 640x480 screen resolution required by Windows. However, for all purposes, the Windows 16-color palette is identical to the DOS 16-color system palette.

Like the DOS 256-color system palette, the Windows 256-color system palette is also poorly designed and it's useful mainly for coloring icons and other GUI screen elements. Here are some of the problems that have been found with it:

- There aren't enough shades of gray present in the palette to realistically depict stone or metallic elements.
- There are too many saturated shades of primary colors (i.e., red, green, and blue) and not enough intermediate or tertiary colors provided. This effectively limits the types of objects this palette can convincingly render.
- The color ranges provided are too small to produce accurate shading and gradient effects.
- Related colors aren't arranged in an accessible order within the palette. This makes the palette very difficult to use in actual projects.

Due to these limitations, I strongly recommend not using the Windows 256-color system palette in your game artwork. It's simply too limiting and restrictive for creating serious game artwork.



NOTE: The only exception to this might be when you're creating artwork or icons for a Windows game that needs to be backward compatible with Windows running on VGA hardware (i.e., 16 colors). In this situation, you must limit yourself to the first 16 palette entries of the Windows 256-color system palette.



NOTE: If you're interested in seeing the RGB values for this palette, they can be found on the book's accompanying CD-ROM under the `PALETTES` directory in the file labeled `WIN256.PAL`.

Linux System Palette

Unlike the other platforms mentioned here, Linux doesn't have an official system palette; rather, it uses whatever color definitions users have defined under their particular desktop manager.

Macintosh System Palette

The Macintosh system palette is interesting in that it shares many of the properties and color arrangements of both the VGA/SVGA (DOS) and Windows 256-color system palettes. Every Macintosh system released since 1990 supports this palette when using an 8-bit color display mode.

Because it shares so much in common with the other system palettes described here, the Macintosh system palette also shares many of the same limitations. Here's a list of them:

- There are too many saturated shades of primary colors (i.e., red, green, and blue) and not enough intermediate or tertiary colors. This effectively limits the types of objects this palette can convincingly render.
- With the exception of the 16 grays in the palette, the Macintosh system palette doesn't provide color ranges large enough to produce realistic-looking shading and gradients.
- Related colors aren't arranged in an easily accessible order within the palette. This makes the palette inconvenient to use in an actual project.
- The Macintosh system palette positions and arranges its colors very differently from how Windows does. This means you won't be able to easily convert images between these palettes without using an image viewer/converter program that supports color remapping.

Again, because of these limitations, I recommend not using the Macintosh system palette for your game artwork. You'll get much better results with a custom color.



NOTE: If you're interested in seeing the RGB values for this palette, they can be found on the book's accompanying CD-ROM under the `PALETTES` directory in the file labeled `MAC256.PAL`.

Java System Palette

As a cross-platform development environment, Java-based arcade games need to display their graphics consistently as much as they have to play consistently on different systems. To facilitate this, Java uses what is called the non-dithering 216-color palette, which is commonly referred to as the Netscape palette.

This palette contains a set of colors that won't dither when displayed inside a browser window regardless of whether it's running on a Macintosh or Windows machine. This fact alone makes this palette useful for cross-platform game projects whether they're written in Java or another development tool.

The Java system palette also borrows many color definitions from other system palettes, namely those from Windows and the Macintosh. Therefore, it too suffers from many of the same issues, including:

- There are too many saturated shades of primary colors (i.e., red, green, and blue) and not enough intermediate or tertiary colors. This effectively limits the type of objects this palette can convincingly render. However, at the same time, these bright colors are actually quite useful in some instances, particularly for creating simple, flat artwork that is commonly seen in many retro-style arcade games, such as *Donkey Kong* or *Bubble Bobble*.
- There are only six shades of any one color. This isn't enough to produce the realistic-looking shading and gradients required by hyper-realistic artwork.
- Related colors aren't arranged in an accessible order within the palette. This makes the palette inconvenient to use in an actual project.



NOTE: If you're interested in seeing the RGB values for this palette, they can be found on the book's accompanying CD-ROM under the `PALETTES` directory in the file labeled `NETSCAPE216.PAL`.



NOTE: Despite its shortcomings, we frequently use the Java/Netscape palette when creating the artwork and animation for ZapSpot's games due to the palettes' special, cross-platform color properties. Examples of where the palette really shines can be found in the games *BullyFrog™* and *FenceOut™*.

Generally speaking, games that use system palettes tend to look sloppy and unprofessional. Using these palettes might have been acceptable during the 1980s but certainly not anymore. With the color capabilities of today's computers, there is simply no excuse not to take advantage of a custom color palette unless you have a very specialized need, such as developing cross-platform or online games. Game developers who don't take full advantage of custom color palettes are just being lazy.

Platform-Specific Palette Peculiarities

This section describes system-specific issues with using color palettes on different computer platforms.

DOS

Table 8-4 shows the various color palettes supported under DOS-based systems.

TABLE 8-4: Color Palettes Supported under DOS

| <i>Display Mode</i> | <i>Color Palette Size</i> | <i>Video Hardware Standard(s)</i> | <i>Comments</i> |
|---------------------|---------------------------|-----------------------------------|--|
| 320x200 | 4 | CGA, EGA, MCGA, VGA, SVGA | Obsolete. Retained only for backward compatibility with ancient game and graphics software. No longer has any use in arcade game graphics. |
| 320x200 | 16 | EGA, MCGA, VGA, SVGA | Seldom used anymore but was once quite popular with developers such as Id and Apogee during the late 1980s and early 1990s. Has only limited use in arcade game graphics. |
| 320x200 | 256 | MCGA, VGA, SVGA | Supported by all arcade games since the late 1980s. Commonly used and popular with game developers. Has limited use in arcade game graphics when compared to Mode X. |

| <i>Display Mode</i> | <i>Color Palette Size</i> | <i>Video Hardware Standard(s)</i> | <i>Comments</i> |
|---------------------|---------------------------|-----------------------------------|--|
| 320x240 (Mode X) | 256 | MCGA, VGA, SVGA | Software modification of the standard 320x200 MCGA/VGA mode. Commonly used and very popular with game developers. Has significant application in arcade game graphics. |
| 640x200 | 2 | CGA, EGA, MCGA, VGA, SVGA | Obsolete. No longer has any use in arcade game graphics. |
| 640x200 | 16 | EGA, MCGA, VGA, SVGA | Obsolete. No longer has any use in arcade game graphics. |
| 640x350 | 16 | EGA, MCGA, VGA, SVGA | Obsolete. Used in some shareware arcade games during the early 1990s. No longer has any use in arcade game graphics. |
| 640x480 | 2 | MCGA, VGA, SVGA | Obsolete. No longer has any use in arcade game graphics. |
| 640x480 | 16 | VGA, SVGA | Obsolete. No longer has any use in arcade game graphics. |
| 640x480 | 256 | SVGA | Commonly used and popular with game developers. Has significant application in arcade game graphics. Usually requires a third-party VESA driver to access. |
| 800x600 | 16 | SVGA | Not commonly used and only has limited use in arcade game graphics. |
| 800x600 | 256 | SVGA | Not commonly used and only has limited application in arcade game graphics. Usually requires a third-party VESA driver to access. |

Palette management under DOS is very liberal. As it happens, DOS isn't restricted to any particular palette order so every one of the 256 color palette entries supported by VGA and SVGA display modes can be redefined. There is

one small catch, however: DOS needs to reserve one of the first 16 colors in the color palette so it can display system text. Therefore, if you redefine or blank out any of these colors, you could potentially render the system font invisible. While there’s certainly nothing preventing you from doing this, please bear in mind that it may cause difficulties with some programming tools and environments. Otherwise, enjoy the freedom DOS provides when it comes to redefining the color palette.

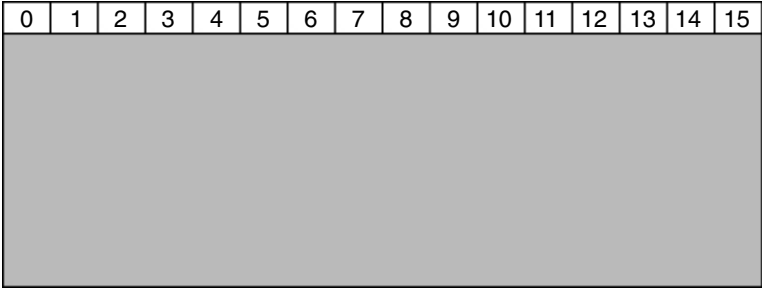


FIGURE 8-7: Diagram of DOS Reserved Colors

Windows

Table 8-5 shows the various color palettes supported under Windows-based systems.

TABLE 8-5: Color Palettes Supported under Windows

| Display Mode | Color Palette Size | Video Hardware Standard(s) | Comments |
|--------------|--------------------|----------------------------|---|
| 320x240 | 256 | VGA, SVGA | Software modification of MCGA/VGA 320x240 display mode supported by DirectX API. |
| 640x480 | 16 | VGA, SVGA | Not supported by Windows 3.1. Obsolete. |
| 640x480 | 256 | SVGA | Has no use in arcade game graphics, although it was commonly used by many Windows games until about 1994. |
| 800x600 | 16 | SVGA | The standard Windows display mode. Commonly used and very popular with game developers. Has significant application in arcade game graphics. Not very commonly used. |

| Display Mode | Color Palette Size | Video Hardware Standard(s) | Comments |
|--------------|--------------------|----------------------------|--|
| 800x600 | 256 | SVGA | A standard Windows display mode. Not as popular as 640x480 with game developers but starting to become much more common in games. |

All versions of Microsoft Windows reserve 20 palette entries for various system purposes such as coloring icons, dialog boxes, and the mouse pointer. Therefore, you shouldn't redefine these colors.

Because of this, only 236 of 256 colors available in the palette can be user-defined. To further complicate matters, Windows does not reserve these colors in one contiguous block; rather, it reserves the first 10 palette entries (entries 0-9) and the last 10 palette entries (entries 246-255) for its own use.

Sixteen of these "reserved" colors are actually composed of standard VGA 16-color palette colors while the remaining four colors are those found in the SVGA 256-color palette. Figure 8-8 shows the organization of the so-called "off-limit" colors under Windows.

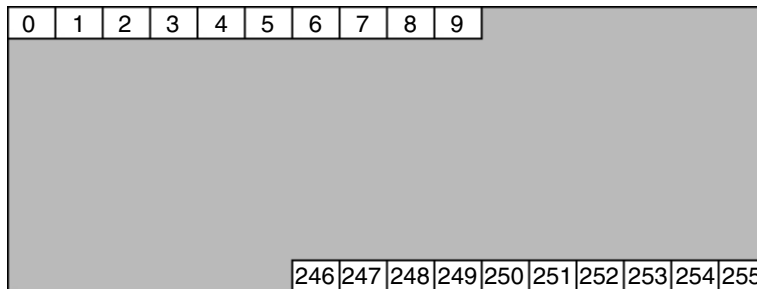


FIGURE 8-8: Diagram of Windows Reserved Colors

TABLE 8-6: Windows Reserved Colors

| Palette Entry | Red Value | Green Value | Blue Value | Color Name |
|---------------|-----------|-------------|------------|--------------|
| 0 | 0 | 0 | 0 | Black |
| 1 | 128 | 0 | 0 | Dark red |
| 2 | 0 | 128 | 0 | Dark green |
| 3 | 128 | 128 | 0 | Brown |
| 4 | 0 | 0 | 128 | Dark blue |
| 5 | 128 | 0 | 128 | Dark magenta |
| 6 | 0 | 128 | 128 | Dark cyan |

| <i>Palette Entry</i> | <i>Red Value</i> | <i>Green Value</i> | <i>Blue Value</i> | <i>Color Name</i> |
|----------------------|------------------|--------------------|-------------------|-------------------|
| 7 | 192 | 192 | 192 | Light gray |
| 8 | 192 | 220 | 192 | Light green |
| 9 | 166 | 202 | 240 | Light blue |
| 246 | 255 | 251 | 240 | Off-white |
| 247 | 160 | 160 | 164 | Medium gray |
| 248 | 128 | 128 | 128 | Dark gray |
| 249 | 255 | 0 | 0 | Light red |
| 250 | 0 | 255 | 0 | Light green |
| 251 | 255 | 255 | 0 | Yellow |
| 252 | 0 | 0 | 255 | Bright blue |
| 253 | 255 | 0 | 255 | Magenta |
| 254 | 0 | 255 | 255 | Cyan |
| 255 | 255 | 255 | 255 | White |

NOTE: Occasionally some Windows video drivers will redefine some of the values presented here. However, the RGB values shown here tend to be the most common implementation by various Windows video card drivers.

NOTE: This restriction only exists in 8-bit color display modes. The 15-, 16- and 24-bit color display modes don't experience this problem.

Unfortunately, this situation leaves Windows game developers with fewer available color choices than the DOS, Linux, or Macintosh platforms provide. Therefore, you'll need to plan your color palettes even more carefully under Windows than on any other system. Not only will you have to make do with fewer colors but also you'll have to take care with regard to how they are arranged.

Linux

Table 8-7 shows the various color palettes supported under Linux-based systems.

TABLE 8-7: Color Palettes Supported under Linux

| <i>Display Mode</i> | <i>Color Palette Size</i> | <i>Video Hardware Standard(s)</i> | <i>Comments</i> |
|---------------------|---------------------------|-----------------------------------|--|
| 640x480 | 16 | VGA, SVGA | The default and minimum video mode for X Windows. Only has limited use in arcade game graphics. |

| <i>Display Mode</i> | <i>Color Palette Size</i> | <i>Video Hardware Standard(s)</i> | <i>Comments</i> |
|---------------------|---------------------------|-----------------------------------|--|
| 640x480 | 256 | SVGA, Macintosh* | The standard Linux display mode. Commonly used and very popular with game developers. Has significant application in arcade game graphics. |
| 800x600 | 16 | VGA, SVGA | The minimum display mode for systems running in an 800x600 resolution. |
| 800x600 | 256 | SVGA, Macintosh* | A standard Linux display mode. Not as popular as 640x480 with game developers but starting to become increasingly common in games. |

* Denotes that Linux can also run on Macintosh hardware.

Most implementations of Linux use a version of the X Windows desktop interface. In an 8-bit display mode, X Windows employs a color reservation system that allows applications, particularly arcade games and graphics packages, to grab as many colors as they need. In order to compensate for this and keep the display legible, X Windows only reserves the first two palette entries, black and white, for system purposes.

Table 8-8 shows their RGB values. These reserved colors include the first and last palette entries. All 254 of the remaining colors are completely user definable. This allows 256-color DOS, Windows, and Macintosh images to be easily ported over to the Linux platform. However, full, 256-color Linux images won't port easily to the Windows platform without a bit of reworking and/or color loss.

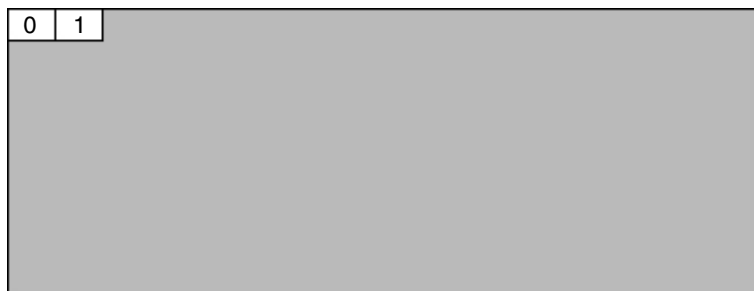


FIGURE 8-9: Diagram of Linux's Reserved Colors

TABLE 8-8: Linux Reserved Colors

| <i>Palette Entry</i> | <i>Red Value</i> | <i>Green Value</i> | <i>Blue Value</i> | <i>Color Name</i> |
|----------------------|------------------|--------------------|-------------------|-------------------|
| 0 | 0 | 0 | 0 | Black |
| 1 | 255 | 255 | 255 | White |

Macintosh

Table 8-9 shows the various color palettes supported under Macintosh-based systems.

TABLE 8-9: Color Palettes Supported by the Macintosh

| <i>Display Mode</i> | <i>Color Palette Size</i> | <i>Comments</i> |
|---------------------|---------------------------|---|
| 640x480 | 16 | Obsolete. Previously was the standard Macintosh display mode. No longer has any use in arcade game graphics, although it was used by many early Macintosh games until about 1994. |
| 640x480 | 256 | The standard Macintosh display mode. Commonly used and very popular with game developers. Has significant application in arcade game graphics. |
| 800x600 | 256 | A standard Macintosh display mode. Not as popular as 640x480 with game developers but starting to become more and more common in games. |

The fact that the Macintosh's early operating systems only supported black and white means that only two palette entries are reserved by the system—black and white.

Table 8-10 shows their RGB values. These reserved colors include the first and last palette entries. All 254 remaining colors are completely user definable. This allows 256-color DOS images to be easily ported to the Macintosh. Unfortunately, 256-color Macintosh images won't port easily to the Windows platform without a bit of reworking and/or color loss.

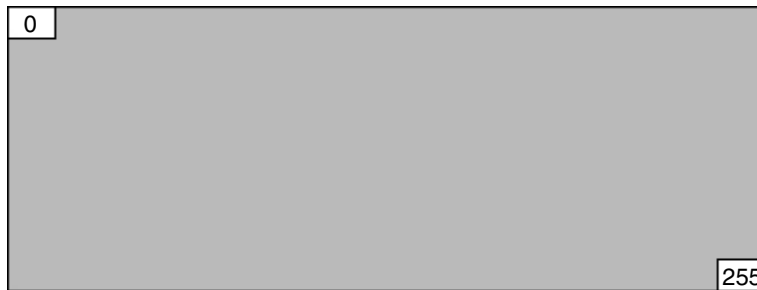


FIGURE 8-10: Diagram of Macintosh Reserved Colors

TABLE 8-10: Macintosh Reserved Colors

| <i>Palette Entry</i> | <i>Red Value</i> | <i>Green Value</i> | <i>Blue Value</i> | <i>Color Name</i> |
|----------------------|------------------|--------------------|-------------------|-------------------|
| 0 | 255 | 255 | 255 | White |
| 255 | 0 | 0 | 0 | Black |

Java

Java color palettes are determined by the platform the Java application is currently running under, and as such, they will inherit the particular palette handling quirks of the host system. So, for example, a Java game that runs under Windows will be subject to how Windows handles reserved colors.



NOTE: You can prevent this issue if you use the Java system palette described earlier in this chapter.

As each platform has its own peculiarities when it comes to the available number of palette entries, it's often easy to forget which restriction applies to which platform. Table 8-11 makes this process easier by compiling this information in one place.

TABLE 8-11: Available Palette Entries in 256-Color Display Modes by Platform

| <i>Platform</i> | <i>Number of Available Colors</i> | <i>Palette Entries Affected</i> | <i>Comments and Restrictions</i> |
|-----------------|-----------------------------------|---------------------------------|---|
| DOS | 256 | 0-15 | <p>All palette entries are open and legal to use but you should be careful not to blank out entries used by the BIOS/DOS text functions.</p> <p>Images can be easily ported over to the Macintosh or Linux systems and vice versa with little or no color loss.</p> |

| Platform | Number of Available Colors | Palette Entries Affected | Comments and Restrictions |
|---------------------------------------|----------------------------|--------------------------|--|
| Windows 3.1, 95, 98, NT 4.0, and 2000 | 236 | 0-9, 246-255 | 20 colors are reserved for system functions and should not be changed. The first 10 and last 10 palette entries are essentially off limits to your palette. Images can be ported to other platforms with little or no color loss but the same does not hold true for images being ported to Windows. Because Windows supports fewer displayable colors, there will be some color loss when moving full-color Macintosh, Linux, and DOS images over to it. |
| Linux | 254 | 0 and 255 | All but two palette entries are free for use. The first and last palette entries are reserved for system use and should not be changed. Images can be ported from DOS, Windows, and the Macintosh platforms with little or no color loss. |
| Macintosh | 254 | 0 and 255 | All but two palette entries are free for use. The first and last palette entries are reserved for system use and should not be changed. Images can be ported from DOS, Linux, and Windows platforms with little or no color loss. |

Creating Color Palettes

There's more to creating a color palette than just picking pretty or interesting colors. Creating an effective color palette actually involves careful planning and implementation. Let's look at the planning aspect first.

Planning a Color Palette

The planning phase happens before you actually define any RGB values and it examines such issues as:

- Game audience
- Game appearance
- Game mood
- Technical restrictions

Game Audience

This looks at who will actually be playing your game. Some things to consider when evaluating your game's audience include age, gender, and geographic/cultural influences of the game's potential players. Therefore, when analyzing the game audience, ask yourself such questions as:

- Is the game targeted towards younger players?
- Is the game targeted towards older players?
- What gender is your game targeted towards?
- Will the game be distributed or marketed overseas or made available for download over the Internet?

Remember that both older and younger users will perceive and react to color schemes differently. Younger audiences will prefer games that have palettes with warmer colors, while older audiences will prefer games with cooler colors.

The same holds true for genders. Be sure to consider this when designing your game's graphics.

Finally, if you're planning to distribute your game overseas or over the Internet, make sure you address any potential cross-cultural color issues that may affect your users.

Game Appearance

This determines the aesthetics of your game. Things to consider here are the desired look and feel you're interested in creating and the degree of realism associated with it. Therefore, as part of this analysis, ask yourself such questions as:

- What look or style are you trying to achieve?
- What colors will you need to produce that particular look?
- How long will your color gradients be?

The "style" you choose for your game can have a significant influence over your palette definitions. Generally, games with realistic design styles tend to emphasize palettes with more intermediate shades while games with retro or cartoon design styles tend to favor palettes with more intense primary colors in them.

Smoother shading is achieved by using longer gradients, so be prepared to reserve sufficient room in your palette if this is something you want to accomplish.

Game Mood

This examines the feelings and symbolism that you're trying to convey in your game. Things to consider when looking at this issue are the types of emotions that

you're attempting to provoke in the player. Ask yourself this question: What mood and emotions are you trying to project to the user?

Remember the discussion in Chapter 7 about the impact that color can have on the user's mood and emotion. While you can always choose to ignore this factor without seriously affecting the quality of your game, why not take advantage of it? By selecting mood-enhancing colors, you can seize the user's emotions and get them fully immersed in your game.

Technical Restrictions

There are a number of technical issues that all games face. Of all of the items discussed so far, this is one of the most complex and frustrating issues to deal with when designing game graphics.

Technical issues can encompass everything from system-specific color capabilities to any special color effects you might want to create. Therefore, when looking at this issue, ask yourself these questions:

- Will the entire game share the same palette or will certain palettes be assigned to specific game levels?
- Will you use separate palettes for menu and title screens?
- What platforms will the game support—DOS? Windows? Linux? Macintosh?

Many games, regardless of their type, contain different levels. Each level can offer the user new and interesting challenges. One of the easiest ways of distinguishing between game levels is to introduce different color schemes. In many cases, this can be done without having to define a brand new palette. However, in some instances it will.

While 256 colors can provide you with many possible color choices, there will be instances where even this many colors can seem restrictive. Because of this, you need to make sure you have completely thought out how your game will handle implementing these level-based color schemes.

The same issue holds true for title and menu screens as well. In addition, let's not forget all of those annoying platform-specific palette differences!



NOTE: If you're unsure about any technical issue raised here, remember to ask your programmer! He or she should be able to fill in any of these gaps for you.

Implementing Your Color Palette

Once you have planned your palette, you will be ready to define its RGB color values and implement it. To do this, you'll need to use the Palette Selection tool that

is specific to your particular painting program. Every program described in Chapter 5 allows you to define your own custom color palettes; however, they all differ somewhat in terms of features and ease of use.

Defining color palettes can be a tricky and tedious process. However, once you grasp the fundamentals, this process will eventually become easier. As with anything else discussed in this book, practice makes perfect.

It is best to break down the process of defining a color palette into several distinct steps. Doing this not only makes the process easier to manage, but it also allows you to back up in case you made a mistake. These steps include:

1. Defining the required color palette components
2. Deciding on a color palette order
3. Adding system colors
4. Selecting your colors
5. Defining your color ranges and gradients
6. Reserving palette entries for programmed effects
7. Testing your color palette
8. Saving your palette

Step 1—Defining the Required Color Palette Components

Each color palette should be broken down into several components. These components include elements such as:

- System colors
- Background color
- Transparent color
- Color ranges/gradients
- Special effect colors

System Colors

As previously mentioned, these are the colors that are reserved by the system for various housekeeping purposes, i.e., coloring windows and icons. Therefore, include them in every palette you define in order to prevent any unpleasant color distortions from occurring during the course of your game.

Background Color

You'll use this color for the background for your game artwork screens. There should only be one background color defined. Make sure that whatever color you choose has sufficient contrast. Very dark backgrounds such as black, dark gray, or dark blue provide excellent contrast when displayed against lighter colors.

NOTE: Traditionally, most programmers assume that palette entry 0 (the first color in the palette) is the background color. However, you are free to use any palette entry as long as you document it and notify your programmer about your selection accordingly.

Transparent Color

Unlike the transparent effects discussed in Chapter 7, transparent colors in the palette are used as *masks*, or stencils, that allow the contents of the foreground to be smoothly blended with the contents of the background. Transparent colors make it possible for irregularly shaped sprites to be used in games and are essential for creating smooth animation. Because of this, no palette should be created without defining a transparent color.

You can use any color at any palette entry as a transparent color as long as you document it and/or tell your programmer about it. However, whatever you do, never make the background color the same as your transparent color when creating a palette. Doing this can potentially cause any object shaded in with the background color (i.e., black) to “show through” when displayed. This effect is depicted in Figure 8-11.

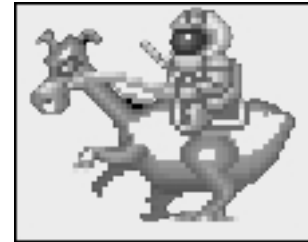


FIGURE 8-11: Background Color Showing through Foreground Object

NOTE: Different painting programs handle transparent colors differently. So, for example, specifying a palette entry as transparent in one painting program won't necessarily be reflected when you import the image into another. However, as long as you record the location of the transparent color in your palette, the transparent effect will be preserved when the image appears in a game.

Color Ranges/Gradients

Color ranges and gradients are colors grouped together for a special purpose such as to render a specific graphic object such as a rock or a tree. Although color ranges don't need to consist of related colors, they should be arranged according to their intensity with colors going from their highest intensity to their lowest intensity. Doing this will make working with your color gradients easier since there will be a visual cue as to which color to use next when shading an object, etc.

As a rule, it's considered good practice to place gradients and ranges that will be used to shade related objects next to each other in the palette. For example, a

gradient that will be used to shade an object representing a tree trunk should be placed next to a gradient that will be used to shade the leaves. Doing this simply makes the process of locating commonly used colors easier and more intuitive.



NOTE: Color ranges and gradients are crucial to constructing a good color palette. Make sure you reserve plenty of room to accommodate them.

Special Effect Colors

Arcade games often make use of several special programmatic effects. These effects can include such items as:

- Register swapping
- Color cycling
- Color fades

Register swapping is a special programming technique that is used to simulate certain visual effects like the flickering of a torch or the flashing of a strobe light. To achieve this effect, the programmer rapidly changes the contents of a single palette entry and swaps in different RGB values at various speeds.



NOTE: As a designer, you really don't have to do anything to produce this effect other than reserving a palette entry for it.

Color cycling, also called *color shifting* or *color indirection*, is the ability to flash a series of colors on the screen in a linear fashion. This technique can be used to create very striking animations of repetitive actions. For example, you can use color cycling to simulate the flowing water in a waterfall, a roving spotlight, or the turning of a wheel.

Figure 8-12 demonstrates how color cycling works. Here, the “X” indicates the direction of the shifting color. To create the illusion of movement, the programmer rotates the contents of palette entries 224 through 231 by incrementing their position within the color palette by one entry at a time.

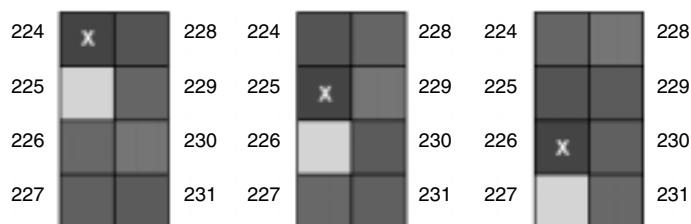



FIGURE 8-12: Color Cycling Mechanics

Although color cycling isn't necessary for most types of arcade games, it can be used to create some very interesting effects.

Traditionally, color cycling effects are handled by the programmer. As a designer, all you are responsible for is planning the color sequence and reserving the necessary palette entries for it.




NOTE: Color cycling was the hallmark of arcade games designed during the late 1980s and early 1990s but has since fallen into disuse. Consequently, very few programs still support the creation of color cycling effects. However, both *Improc* and *Deluxe Paint* support the creation and previewing of this effect.

While there is no minimum number of palette entries required to achieve effective color cycling, the more palette entries you reserve, the smoother the animation effect will be. Table 8-12 provides some guidelines on how many color palette entries to use in order to achieve a particular type of color cycling effect.

TABLE 8-12: Color Cycling Palette Entry Reserve Guidelines

| Number of Color Palette Entries | Comments |
|---------------------------------|--|
| Less than 4 | Produces a very coarse color cycling effect. |
| 4-8 | Produces a relatively coarse color cycling effect. |
| 8-16 | Produces a relatively smooth color cycling effect. |
| Greater than 16 | Produces a smooth color cycling effect. |



NOTE: Due to the limited number of color palette entries, it's advised that you don't use more than 16 palette entries for your color cycling effects unless necessary. In fact, you can usually create effective color cycling with as few as eight palette entries.

Color fading is another common visual used by arcade games. It is used as a means of transitioning in and out of title screens, menu screens, and game screens. Color fading can be smooth or coarse depending on the number of palette entries used to contain the effect.

Color fading works by filling in the contents of the palette with progressively darker RGB color values. When done properly, this has the effect of gradually "blanking" the contents of the screen. Figure 8-13 provides an illustration of the effect. Here, phases 1 through 4 represent the different stages of the palette being filled in with progressively darker colors.

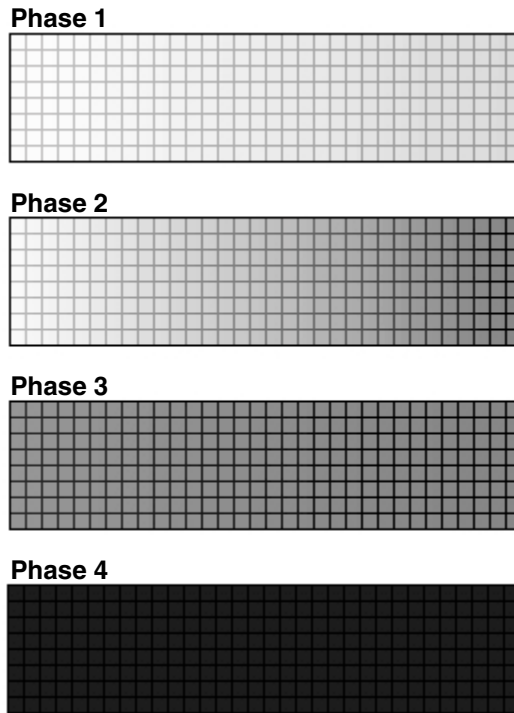


FIGURE 8-13: Color Fading (one box = one palette entry)

The number of palette entries used to create the effect has a direct influence on its quality. Table 8-13 provides some general guidelines for how many color palette entries to reserve in order to manage the quality of a color fading effect.

TABLE 8-13: Color Fading Palette Entry Reserve Guidelines

| <i>Number of Color Palette Entries</i> | <i>Comments</i> |
|--|---|
| Less than 16 | Produces a very coarse color fading effect. |
| 16-32 | Produces a relatively coarse color fading effect. |
| 32-64 | Produces a relatively smooth color fading effect. |
| 64-128 | Produces a smooth color fading effect. |
| Greater than 128 | Produces a very smooth color fading effect. |

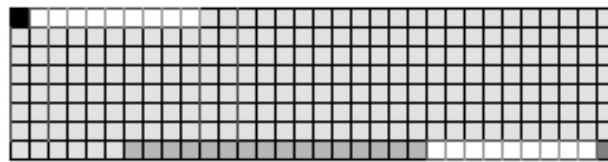


NOTE: Due to the limited number of color palette entries, it's generally recommended not to use more than 128 palette entries for your color fading effects unless absolutely necessary.

NOTE: You are not limited to reserving only one section of your palette for special programmed color effects. In fact, you can reserve as many colors in your palette as you want for these effects. The only real limitations are your imagination and the number of colors you can spare for the effect.

In any case, none of these special color palette effects are necessary to create a good palette or a decent arcade game. Still, when done properly, they can enhance the visual impact of your game and should be seriously considered if you have the time to plan them and the appropriate number of color palette entries available in the palette.

Figure 8-14 illustrates the various color palette components of a typical color palette.



- ☐ System Colors
- ☒ Background Colors
- ☒ Transparent Colors
- ☐ Color Ranges/Gradients
- ☒ Special Effect Colors

FIGURE 8-14: Color Palette Components

Step 2—Deciding on a Color Palette Order

In theory, you don't need to arrange your color palette entries in any particular order. However, in practice, you might want to do this as it offers two important advantages:

- **Makes colors easier to access**—Grouping similar or related colors together in the palette makes them easier to find and access. Doing this will save you time, particularly when working with colors that are used often. For example, if your artwork will contain trees or other vegetation, you might want to place your shades of green alongside shades of brown.
- **Better supports programmed color effects**—Color palette entries that are arranged in a specific and predictable order tend to make it easier for the programmer to implement programmed color effects. In addition to saving the programmer time, it enables these effects to be used consistently within a game.

This being said, it's good practice to spend time mapping out the order of your palette entries. Figure 8-15 shows an example of this.

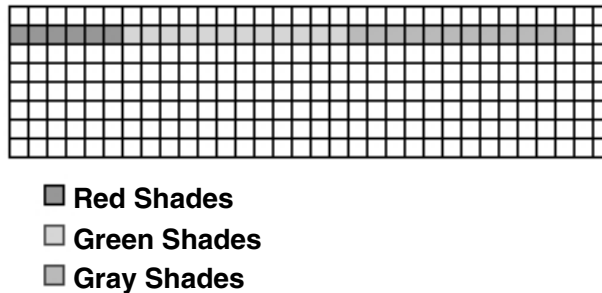


FIGURE 8-15: Color Palette Order

Step 3—Adding System Colors

Always take the time to add your system colors to the color palette for the reasons discussed earlier in this chapter. Failing to do so can cause problems, particularly on Windows systems.

You can either add these colors manually using the RGB values specified in Tables 8-6, 8-8, and 8-10 or you can have your painting program do it for you. Be aware that certain programs such as *NeoPaint for Windows* provide features to do this, as do several of the palette tools offered by programs mentioned in Chapter 6 of this book.

Step 4—Selecting Your Colors

At this stage, you're ready to start picking your own color palette definitions. To do this, bring up the Color Selection tool in your painting program and simply manipulate the RGB (or HSV, HSL, HSB, etc.) color sliders to pick and blend the colors you want.

Moving the color slider down (or to the left in some programs) will cause the RGB value to get darker (closer to pure black) while moving the color slider up (or to the right) will cause the RGB color value to get lighter (closer to pure white).

One of the most challenging aspects of this step is figuring out what the specific RGB values are for a given

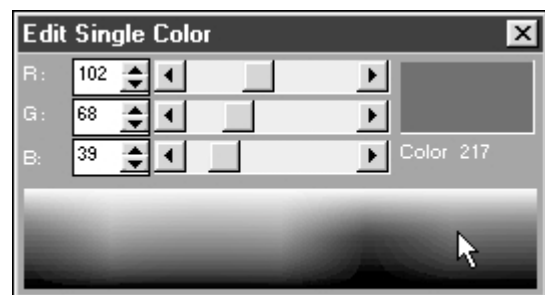


FIGURE 8-16: Color Slider Example

color, particularly for the so-called “real-world” colors that we see and experience every day.

Want to know what color army green is? Or what the RGB color values for a flesh tone might be? Well, look no further. To help jumpstart the color selection process, Table 8-14 includes a list of several dozen “starter” colors.

TABLE 8-14: Starter Colors

| Color Name | RGB Color Values (Windows, Linux, Macintosh, and Java) |
|--------------------|---|
| Aqua | 51 255 204 |
| Army Green | 128 128 0 |
| Avocado | 153 153 0 |
| Banana | 227 207 87 |
| Bauhaus Gray | 66 66 66 |
| Black | 0 0 0 |
| Blue-Green | 0 153 102 |
| Boot Leather | 160 68 0 |
| Brick | 204 102 51 |
| Bright Blue | 0 102 255 |
| Bright Gray | 192 192 192 |
| Bright Green | 0 204 51 |
| Brown | 232 168 144 |
| Burgundy | 153 0 51 |
| Cadet Blue | 96 156 152 |
| Carrot | 237 145 33 |
| Castle Stone | 41 41 41 |
| Chartreuse | 204 255 0 |
| Cobalt | 61 89 171 |
| Coral | 255 127 80 |
| Crimson | 204 0 51 |
| Dark Blue | 0 0 153 |
| Dark Brown | 104 52 0 |
| Dark Gray | 22 22 22 |
| Dark Green | 51 102 51 |
| Dark Periwinkle | 102 102 255 |
| Dark Purplish Blue | 102 0 204 |
| Deep Purple | 128 0 128 |
| Deli Mustard | 255 204 102 |

| Color Name | RGB Color Values (Windows, Linux, Macintosh, and Java) |
|----------------------|---|
| Dull Green | 102 255 153 |
| Eggshell | 252 230 201 |
| Fire Brick | 165 0 33 |
| Flesh | 255 204 153 |
| Florida Gator Orange | 255 102 0 |
| Fuscia | 255 0 255 |
| Gold | 204 204 51 |
| Golden Brown | 153 102 0 |
| Grass | 51 153 102 |
| Gray | 128 128 128 |
| Gray-Blue | 0 102 153 |
| Grayish Brown | 152 140 104 |
| Grayish Purple | 153 153 204 |
| Green | 0 128 0 |
| Gun Metal | 150 150 150 |
| Hot Pink | 255 102 204 |
| Indigo | 8 46 84 |
| Ivory | 255 255 240 |
| Khaki | 240 230 140 |
| Lavender | 153 102 204 |
| Light Beige | 144 132 112 |
| Light Blue #1 | 51 153 255 |
| Light Blue #2 | 0 204 255 |
| Light Brown | 136 96 88 |
| Light Gold | 204 204 102 |
| Light Gray | 178 178 178 |
| Light Orange | 255 153 0 |
| Light Plum | 204 153 204 |
| Light Rose | 255 153 204 |
| Light Yellow | 255 255 102 |
| Lilac | 204 153 255 |
| Lime | 0 255 0 |
| Linen | 250 240 230 |
| Maroon | 128 0 0 |
| Medium Blue | 0 51 204 |

| <i>Color Name</i> | <i>RGB Color Values (Windows, Linux, Macintosh, and Java)</i> |
|-------------------|---|
| Medium Gray | 85 85 85 |
| Medium Green | 0 153 0 |
| Melon | 227 168 105 |
| Milk Chocolate | 144 52 40 |
| Mint | 189 252 201 |
| Navy Blue | 0 0 128 |
| Ocean Blue | 56 176 192 |
| Off White | 227 227 227 |
| Olive Drab | 107 142 35 |
| Periwinkle #1 | 153 0 255 |
| Periwinkle #2 | 102 51 255 |
| Pink | 255 124 128 |
| Pumpkin | 255 153 51 |
| Purple #1 | 102 51 204 |
| Purple #2 | 102 51 153 |
| Raspberry | 135 38 87 |
| Red | 255 0 0 |
| Red-Orange | 204 51 0 |
| Royal Blue | 0 0 255 |
| Royal Purple | 102 0 153 |
| Rust | 112 0 0 |
| Saddle Brown | 112 76 48 |
| Salmon | 255 102 102 |
| Sandy Brown | 120 0 0 |
| Sea Green | 153 204 153 |
| Sienna | 204 102 0 |
| Sky Blue #1 | 128 218 225 |
| Sky Blue #2 | 0 220 255 |
| Snow | 255 250 250 |
| Steel | 102 102 102 |
| Sun Yellow | 255 255 0 |
| Tan | 204 204 153 |
| Taupe | 160 144 128 |
| Teal | 0 128 128 |
| Tomato Soup | 255 102 51 |

| Color Name | RGB Color Values (Windows, Linux, Macintosh, and Java) |
|--------------|---|
| Turquoise | 0 255 255 |
| Ultra Purple | 51 0 51 |
| Wet Cement | 184 156 152 |
| Wheat | 245 222 179 |
| White | 255 255 255 |
| Wild Orchid | 176 48 216 |
| Wine | 204 51 102 |

NOTE: It's interesting to point out that despite the fact that there are literally millions of colors in the visible spectrum, scientists and designers have only succeeded in naming about 7,000 distinct colors.

Step 5—Defining Your Color Ranges and Gradients

Using your painting program's Color Selection tool, define your color ranges and gradients and place them in the areas that you reserved during Step 2 of this process. As mentioned in Chapter 5, most painting programs provide options to do this for you automatically.

Color ranges and gradients are extremely useful as they allow you to choose two different colors and create a bridge of intermediate shades between them. For example, to make a range of flesh tones first create two extremes of color that would best represent flesh. In this case, off-white and medium brown would work. The numbers of color palette entries between these two end colors determines how many colors will actually be generated between them. More palette entries will produce a smooth gradation of color, while fewer palette entries will produce a coarser gradation of color.

Finding the optimal number and arrangement of color ranges and gradients takes time and experience. Expect to spend a fair amount of time experimenting to get the right balance.

If you're short on time or just impatient, consider looking at Table 7-7 for suggestions on how to define and optimize the size of your color gradients.

NOTE: The colors defined in Table 8-15 are good starting points for creating color ranges and gradients. Just make sure to always define a starting color and an ending color, i.e., which color is lighter or darker. If there aren't two intensity extremes between the start and end of the gradient, the effect won't take.

Figure 8-17 illustrates how a color range or gradient might actually look in your palette. Min represents colors with the lowest intensity (darkest) while Max represents colors with the highest intensity (lightest).

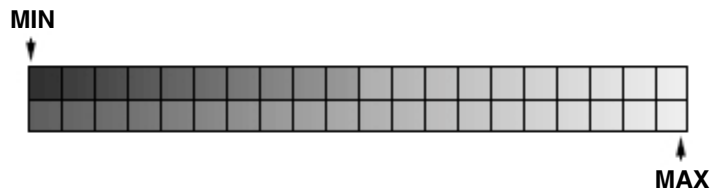


FIGURE 8-17: Example of a Color Gradient in a Color Palette

Every game object, regardless of its function, has some color or set of colors associated with it. For example, apples are red, rocks are often rendered in shades of gray or brown, leaves are green, water is blue, etc. Yet, given the fact that you can choose from thousands if not millions of colors from the hardware palette, it can often be difficult to figure out exactly which colors should go with what object.

To help with this, take a look at Table 8-15 as it provides some general suggestions for which game objects go best with which color gradient definitions.

TABLE 8-15: Color Gradient Usage Suggestions

| <i>Color Gradient</i> | <i>Suggested Use(s)</i> |
|----------------------------|--|
| Reds, yellows, and oranges | Explosions and fire |
| Various shades of blue | Assorted surface structures, sky, ice, and water |
| Various shades of brown | Earth, asteroids, wood, flesh, and rock formations |
| Various shades of gray | Assorted metallic alloys, stone, and smoke |
| Various shades of green | Vegetation |
| Various shades of magenta | Rock formations, assorted flora, and cloth |
| Various shades of pink | Flesh |
| Various shades of red | Blood, lava, and assorted surface structures |
| Various shades of tan | Flesh and earth |
| Various shades of yellow | Sunlight and treasure |

NOTE: To save you time and effort defining your own color gradients, the CD-ROM included with this book features over 50 predefined gradients that address virtually any game related need. These definitions can be found in the `PALETTES` directory in the file labeled `starter_palettes.html`.

Step 6—Reserving Palette Entries for Programmed Effects

If you plan to take advantage of any special programmatic effects such as color cycling or palette fades, this is the point in the palette definition process when you would reserve palette entries for them. The reason this comes up so late in the process is two-fold:

- As programmed effects are not central to your game artwork, your main drawing colors have priority in your palette.
- Programmed effects are often removed from the game at the last minute for various reasons. By adding them last, you can easily reclaim the palette entries they occupied with a minimum of fuss should they be removed later on in the game development cycle.

Step 7—Testing Your Color Palette

After you define your system colors, individual colors, and color gradients, you should always test out your palette before using it in a game. This is an extremely important step since what looks good in your Palette Selection tool won't always look good when in actual use. To see if your palette actually passes muster, try one or more of these procedures:

- The contrast test
- The light/dark test
- The gradient test

The Contrast Test

This particular test evaluates the contrasting qualities of the colors in your palette. As mentioned in Chapter 7, colors without sufficient contrast undermine many of the advantages that color can offer your game.

To determine this, draw some simple shapes on the screen using your current palette definitions, then see how they contrast with each other. Do they look okay? If not, something is wrong with your palette; you'll need to redefine the problematic colors and test your palette again until they do.

The Light/Dark Test

This test evaluates the relative lightness or darkness of the colors you have chosen for your palette.

To perform this test, alternate drawing several test objects against both dark and light backgrounds and see how things look. Do the colors seem too dark? Do they appear to be too light? If so, fix the offending color values in your palette and test again.

The Gradient Test

This test examines how well the colors you've selected blend together. This is important since colors that don't blend well also don't shade well.

To perform this test, create some shapes and fill them with color gradients. Next, take a close look at how the colors within these shapes blend and ask yourself questions like these:

- Are your gradients large enough for the level of shading you want to achieve in your artwork?
- Are they too coarse?
- Is there enough contrast between gradient colors or are they too similar?

If you discover a problem here, go back in your palette and tweak your color selections until the problems you identify in this test vanish completely.

In short, it always pays to play around with your color palette and look for potential problems that might crop up before you start using it in your artwork. Believe me, it's far better to discover a potential problem with the contents or organization of your palette now rather than later when you're in the middle of a major project. By that point, it might be too late to start making wholesale changes to your palette.

Step 8—Saving Your Palette

When you are satisfied with how your palette looks, it's time to save your palette to disk. Whenever possible, try to save your palette in the Microsoft .PAL format, as this will help make your color palette definitions portable across different graphics applications and development environments.



NOTE: Please be aware that several programs mentioned in Chapter 6 and included on the book's CD-ROM save palettes as .PAL files. However, some of these are actually saved in proprietary formats and not in the Microsoft-compatible .PAL format. Always make sure that you save your palettes in the correct .PAL format, especially if you want to be able to share your color palettes with different programs.

A Color Palette Creation Exercise

This section will quickly review the basics of actually defining a color palette. For this example, we'll use *Pro Motion*, an excellent Windows-compatible painting program that is described in more detail in Chapter 6. *Pro Motion* was selected for this exercise because of its powerful color palette manipulation features.

NOTE: Virtually all of the other painting programs mentioned in Chapter 6 offer similar methods for defining the color palette. The only real difference between them and *Pro Motion* is how this operation is actually performed.

1. Start up *Pro Motion* (a demo is available on the book's CD-ROM; see Appendix B for details) and select the **Colors** menu.
2. Select the **Load Palette** option. This brings up a dialog box. Navigate so that you are inside *Pro Motion*'s Palettes directory as shown in Figure 8-18. This directory contains several predefined color palettes that come with the program. The one we're interested in is called *win256.pal*. This palette file contains the definitions for the Windows 256-color system palette described earlier in this chapter. It was chosen for this exercise because it already contains Windows' 20 reserved colors in their proper places.

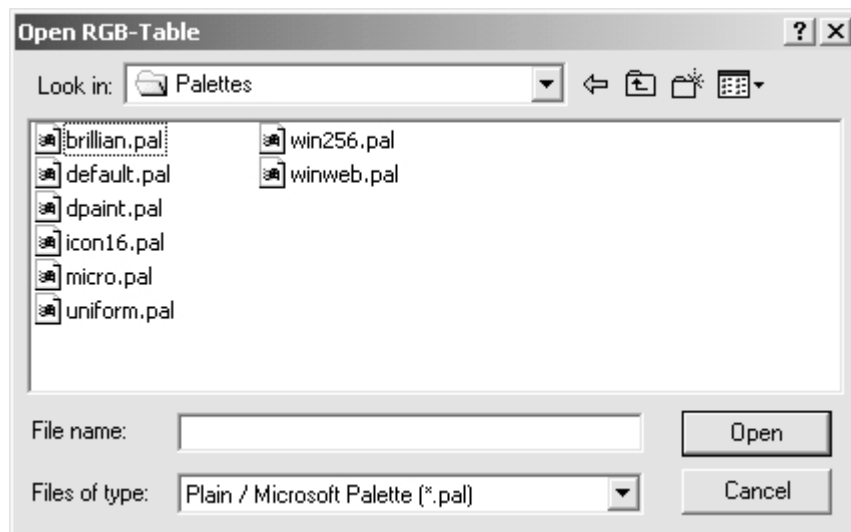


FIGURE 8-18: Palette Load Dialog

3. Select it and choose **Open**. You will notice that color bars at the top left of the screen will change to reflect the new color palette definitions. If they don't, repeat the operation until they do.
4. Go back to the Colors menu and choose the **Edit Palette** option. Doing this brings up *Pro Motion*'s Edit RGB Table dialog (Color Palette tool) as shown in Figure 8-19. This tool presents the current color palette's 256 color definitions in a 16x16 grid. Clicking on any colored box in this dialog will display its RGB color values and its order within the palette. Two colored boxes also appear on the screen and surround two color palette entries. The white box represents

the foreground color, or the start-point of your color gradient. The green color represents the background color, or the endpoint of your color gradient.

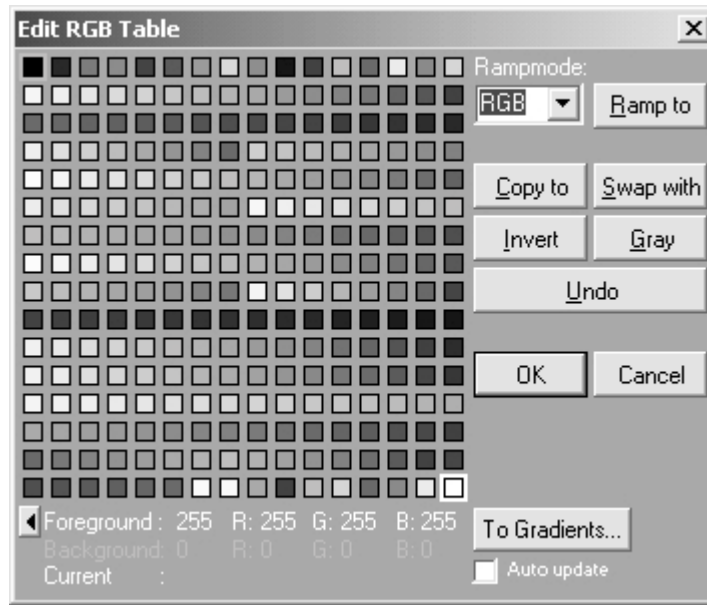


FIGURE 8-19: Edit RGB Table (The Palette Tool)

5. Click on the small left-arrow button located at the bottom of this dialog. This brings up *Pro Motion*'s Edit Single Color dialog as shown by Figure 8-20. From here, you can define the individual colors of the color palette by manipulating the RGB or HSB sliders. In addition, you can visually select colors from the color spectrum located at the bottom of the dialog.

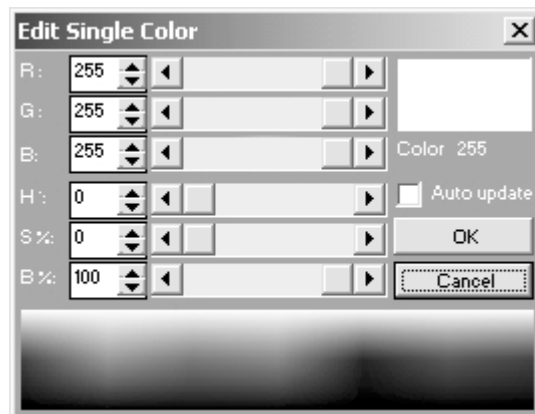


FIGURE 8-20: Edit Single Color Dialog (The Color Editor)

6. To define a color gradient/range, switch to the Edit RGB Table dialog and click on the color palette entry where you would like the gradient to start. As you do this, the white box will surround your selection.
7. Next, go to the Edit Single Color dialog and define a color for the selected palette entry.
8. Now, select a background color by right-clicking the mouse button on the palette entry that marks the end of your gradient and repeat step 7. Just remember the rules on gradients discussed in Chapter 7!

When done, switch back to the Edit RGB Table dialog and click on the **Ramp to** button. The mouse cursor will change to a cross hair. Click on the green box and *Pro Motion* will automatically compute the color shades that fit between your foreground and background color selections as shown in Figure 8-21.

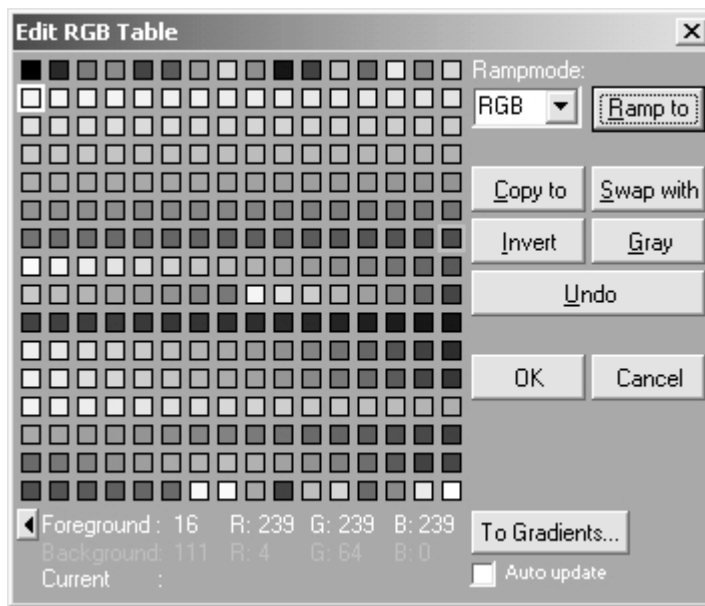


FIGURE 8-21: Successfully Defined Color Gradient

NOTE: *Pro Motion* will only allow you to define gradients that run from the foreground to the background. The opposite won't work in this program; however, some painting programs do support this feature.

9. To define the remainder of your palette, simply repeat steps 6 through 9 until the palette is defined to your liking.
10. You can use the button labeled Copy to to copy single color palette definitions to other areas within the palette. You can use the Swap with option to exchange the positions of individual color palette definitions within the

palette. The Gray button automatically converts the current foreground color to its equivalent RGB gray value. The Invert button replaces the contents of the currently selected foreground color with its mathematical inverse. Finally, the Undo button undoes the last operation or change you made.

11. When done, select the **OK** button. This will cause the program to automatically accept your new palette definitions. If you want to permanently save the palette, go to the Colors menu and choose the Save Palette option. A dialog box will appear as in Figure 8-22. Select **Microsoft Palette** in the Save as type box and choose **Save**.

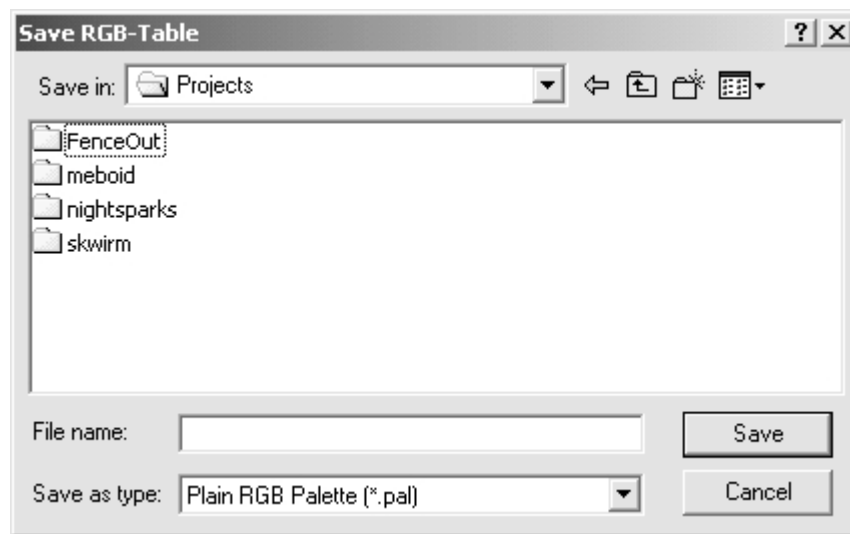


FIGURE 8-22: Save RGB-Table Dialog

Tips for Creating Effective Color Palettes

This section provides some useful tips on creating, defining, and editing your own color palettes.

Gradient Selection

- **Avoid wasting colors**—For many palettes, it's suggested that you arrange your color gradients so that colors go from the highest intensity to their lowest intensity. However, be careful not to waste available color palette entries by using redundant colors (colors that have already been defined) or by filling them in with shades that don't differ substantially in contrast from one other.

- **Build gradients with consistent intensity and saturation**—Gradients that feature consistent intensities and saturations tend to be more flexible than gradients that do not. In addition to keeping your color selections uniform, doing this allows the contents of these gradients to be easily colorized. The Colorize function was discussed in Chapter 6.
- **Conserve colors intelligently**—Remember that you don't always have to use very long color gradients (ones that go from minimum to maximum color intensity) for your palettes. Instead, you can use any gradient length as long as it contains enough shades so that you can render objects that maintain the illusion of realism. Doing this will help you conserve colors in your palette for other purposes.
- **Make logical color selections**—Your color selections should closely mirror the user's expectations. In other words, you don't want to select colors that seem out of context. For example, water should be a shade of blue and not purple. Trees should have brown trunks and not red, etc. Doing this will enhance the character of your artwork and maintain the suspension of disbelief.

Programmed Color Effects

- **Place palette entries adjacently**—In order to work, programmed effects such as color fades and color cycling require that color palette entries be placed adjacently.
- **Test your color cycling**—Whenever possible, test any color cycling effects that you create prior to handing them off to the programmer. You can do this by previewing them with any program that supports color cycling so that you can discover any potential problems with your implementation sooner rather than later.
- **Be conservative**—Don't be greedy. Remember that you will need plenty of colors to render your main game objects so don't reserve too many color palette entries for special effects at the expense of your normal drawing colors. As I said earlier, drawing colors should always have priority in your palette.

Transparent Colors

- **Use two blacks**—As black is commonly used as both a background and transparent color, it's a good idea to reserve a second shade of black (RGB: 0,0,0) somewhere else in your palette. This way, you can use black as much as you want in your artwork without having to worry about the contents of the background showing through your foreground objects.

Miscellaneous Color Palette Tips

- **Borrow color palettes from other games**—If you have trouble coming up with your own color palette definitions, or are like me and just plain lazy, you can always borrow color palettes from other games. To do this, simply use one of the screen capture utilities described in Chapter 6 and capture a game screen that interests you. Next, use a palette tool (also described in Chapter 6) to extract the contents of the image's palette. Doing this can give you excellent insight on how to use color in your games in addition to saving precious time.
- **Use a scanner to obtain “difficult” colors and shades**—Certain colors, such as those used for flesh tones, can be very difficult to accurately reproduce manually when defining a color palette. One of the best ways to determine which colors to use is to let the computer determine the color values for you. You can do this by scanning in a photograph from virtually any magazine into your favorite image editor or painting program and then copy the RGB values it uses to represent areas of flesh into your palette.
- **Try to leave sufficient space at the end of your palette**—You are bound to define color ranges and gradients that don't fit the feel of your artwork as the game evolves. Because of this, you should always leave room at the end of your palette for additional or “replacement” gradients that you can use instead. Doing this will reduce the possibility of you having to scrap any artwork rendered with your existing color palette.

Color Reduction

There will come a time in creating arcade game graphics when you will need to either reduce the amount of colors contained in an image or remap the contents of one palette with the contents of another. Often, you will have to do both.

Color reduction is the process used whenever you need to take an image you made in a high color display mode and want to use it in a display mode that only supports 256 colors (i.e., 8-bit).

Palette remapping is a subset of the color reduction process. It's used when you need to convert an image that was made using colors from one palette and incorporate this image into a different palette. Even if some or all of the colors are the same between the two color palettes, their different palette orders prevent the image from displaying correctly when it is converted over to the new color scheme.

Fortunately, most of the programs mentioned in Chapter 6 can perform color reduction on images; however, their results vary wildly depending on the images used and the options selected.

There are essentially three color reduction methods. These are:

- Palette optimization
- Dithering
- Straight color remapping

Palette Optimization

This color reduction method uses several different algorithms that analyze the contents of both images (the original the new image). It then looks for similarities between the colors contained in both images and weighs them accordingly. The more colors that match in both images, the better the results.



NOTE: There are great disparities between the various palette optimization algorithms in common use. Some work better on certain types of images while some work better on others, etc. In the end, your best bet is to experiment.

Dithering

As mentioned in Chapter 2, dithering is a technique for simulating colors that are missing from an image's palette. By blending pixels of two or more palette colors, a third color can effectively be simulated. In many ways, dithering is very similar to mixing paint and in the proper circumstances, dithering can produce excellent results.

The main drawback to dithering is this: if the unavailable colors differ too much from the originals, the dithering will produce images that look spotty, grainy, and coarse, especially when compared to the original image. See Figures 8-23 and 8-24. Like palette optimization, dithering works best on images that have similar colors in them.

Dithering is frequently used when designing complicated shades for display on screens that do not support large color palettes (i.e., 16-color) and in the hands of a skilled artist, can be used to create fairly convincing and attractive artwork. In this situation, dithering can effectively simulate hundreds of additional shades and colors, making it particularly useful when creating artwork destined for online use (i.e., the Web).



FIGURE 8-23: Original Image



FIGURE 8-24: Dithered Image



NOTE: Dithering is often combined with palette optimization to produce better results, especially when dealing with reducing 24-bit images down to 8-bit color, etc.

Straight Color Remapping

This technique simply does a one-for-one swap of colors between the original image and the target. While this technique is very fast, the results are often less than stellar. This is because straight color remapping works best on images that have similar numbers of colors in them. Figure 8-25 shows the results of color remapping. Compare it to the original image as seen in Figure 8-23 and you will

notice that much of the color fidelity and shading of the original is lost in the remapped version.



FIGURE 8-25: Straight Color Remapped Image