13

Blending Pixels Together

IN THIS CHAPTER:

- About Blend Modes
You’ll find that most editing procedures at some point involve putting something new on top of something old. This is true whether you’re applying new paint to an old background, moving a new layer on top of an old layer, or creating a new shape to fit on top of an existing element of a photo. When you put something on top of something else, Photoshop Elements wants to know what it should do to create the result: Does the new stuff replace the old stuff completely? Or does the new stuff blend with existing pixels, and if so, in what way should the pixels be blended? You can answer these fundamental questions in part with the Opacity setting you choose for a tool or a layer; the opacity affects the transparency of the data applied by the tool or on the layer, and thus, how much that data blocks the data on layers below. You can also control how new data interacts with old data by changing a tool or layer’s blend mode. In this chapter, you’ll learn how the various blend modes work.

All the tools that support blend modes have a Mode list on the Options bar from which you can choose the blend mode to use. If you choose Normal, no blending occurs, and the pixels you apply with the tool totally block (overlay) those below. You can also set the blend mode for a layer by choosing the blend mode to use from the Mode list at the top of the Layers palette when that layer is selected.

The best way to describe how blend modes work is to show each one in action. To do this, I assembled two test patterns. One is the source, containing the layer to be copied. The other is the target, on top of which the copied layer will be placed.

The source pattern is made up of four corners, labeled S, K, L, and J. The source’s background is made up of a gradient that fills, left to right, from black to white. At corners K and L, opacity was reduced to 50%; at corners S and J, opacity was left at 100%. Three horizontal gradient stripes appear in the middle of the source image, graduating from red to yellow, cyan to green, and blue to magenta. All colors in these stripes are at full intensity.

The target pattern is Abe’s head from the Lincoln Memorial, set against a gradient background that graduates from top to bottom, blue to green to red. The background colors here are at half intensity.
The source image is a combination of colors and opacities. The target image is a photo with a gradient as a background.

TIP

Some of the tools offer extra “modes” (not shown here, and not really blend modes) that do not apply to copying layers or selections. The Pencil, Brush, Paint Bucket, Clone Stamp, Pattern Stamp, and Gradient tools offer the Behind mode, which paints or copies pixels only on the transparent part of a layer. Using this mode is like painting on the back of transparent areas of a sheet of acetate. The Pencil Brush and Paint Bucket tools also offer the Clear mode, which causes the tool to paint transparency onto an area, as if transparency were a paint. Neither mode works when painting on top of the Background layer, which has no transparent base, so no part of it can be made transparent. The Healing Brush tool offers a Replace mode, which causes the tool to copy pixels from the source, and not blend them with existing pixels, except along the very edges of the brush.

With that setup out of the way, here are the blend modes used by Photoshop Elements. As you read the descriptions, compare the result figure shown next to that particular blend mode with the Normal result shown here.

- **Normal**—Source pixels replace target pixels. The Opacity setting of the source pixels determine the extent to which that replacement is made: totally (Opacity = 100) or partially (Opacity < 100). With Opacity set to less than 100, a percentage of source colors equal to 100 minus the Opacity setting is mixed with target colors. In square K, notice how the half-opaque light pixels brighten Abe’s forehead and the background behind it, and in square L, how the half-opaque dark pixels darken that area. Normal is the default blend mode for all painting and all layer-copying operations.
• **Dissolve**—Gives you the opportunity to apply some visual effects and relies entirely on the **Opacity** setting (set on the **Layers** palette if you’re selecting the **Dissolve** blend mode for a layer, or set on the **Options** bar if you’re selecting the **Dissolve** blend mode for a tool). When you set **Opacity** to less than 100, rather than making the blend color partially transparent, **Dissolve** removes pixels from the source at random locations to let target pixels show through. So if you’re using a brush tool, your tool color might or might not overwrite the target pixel—the lower the **Opacity** value, the less likely your tool will do so. When you’re laying one layer atop another, **Dissolve** covers up some pixels in the next layer entirely, while letting others show through; the higher the **Opacity** setting, the fewer of the lower-layer pixels show through. In the example, notice how the half-opaque squares K and L look fuzzy, while S and J were copied at full opacity.

• **Darken**—Source pixels are blended with target pixels only when they are *darker* than the target pixels. If you’re using a brush tool to apply the blend color, **Darken** adds the color only when the result is *darker* than the existing color. The **Opacity** setting determines the extent of the darkening effect. Notice in the example that bright pixels in squares K and J have little or no effect on the right side of Abe’s head, while squares S and L darken their respective target regions.

• **Multiply**—Compounds the darkness of source pixels with the darkness of target pixels so that the result is always darker. This blend mode most closely simulates the effect of laying one color transparency over another and projecting one light through both. What distinguishes the **Multiply** mode from the **Darken** mode is that, with **Multiply**, all pixels from the source darken all pixels in the target to the extent that they are dark. With **Darken**, when a source pixel isn’t darker than the target to begin with, its effects are discarded; with **Multiply**, if a source pixel is one unit darker than bright white, it darkens the corresponding target pixel by one unit. This is why you can see the faint K and J in the **Multiply** example, while they’re almost invisible in the **Darken** example.
• **Color Burn**—A complex blend mode that focuses on saturation. Dark and highly saturated (richly colored) pixels in the source darken less saturated pixels in the target, increasing their saturation, and transfer their color value to the target. Dark, less saturated pixels in the source do darken corresponding pixels in the target, although without increasing saturation—there’s no saturation in the source that they can lend to the target. Richly saturated pixels in the target, however, are least affected by pixels in the source. Generally, the **Color Burn** blend mode expands areas of rich saturation from surrounding areas, resulting in higher-contrast areas where differing or opposite colors collide with, or come closer to, one another. Notice in the example how the bright pixels in square J darken Abe’s cheek. Up close, you can see how the highlight on his cheekbone in square J is pinker—the saturation there has been increased.

• **Linear Burn**—Applies a similar burn technique as **Color Burn**, but the difference here involves opacity. With **Linear Burn**, the saturation values of all three color channels of source pixels are simply added to those of the target pixels. Meanwhile, the quantity of contrast between source and pixel colors is subtracted from the target, making the target darker by precisely the level of contrast. The effectiveness of the result depends on the **Opacity** level of the source. So in the example, in square S, the dark and unsaturated pixels in the source not only darken Abe’s forehead but add pinks and browns. Although square J has lighter pixels in its source, they too darkened and saturated the target by exactly as much as square S did. But squares K and L, which are 50% opaque, left Abe’s face closer to its original saturation. Meanwhile, the richly saturated color stripes in the source only made a dark, “burned” mark where the source and target colors contrasted most—in square K, where yellow contrasted against blue. Yet while **Color Burn** left Abe’s nose white, **Linear Burn** turned it as blue-green as the center of the source.

• **Lighten**—Applies the source or target color, whichever is lighter, as the result color. Pixels darker than the blend color are replaced, and pixels lighter than the blend color do not change. The result is always the lightest of the two, although its brightness value is unchanged. Notice in the example how none of the dark pixels in squares S and L translated to the target. Square J is mostly lighter than the target, so it mostly occluded the target. Meanwhile, the 50% opaque square K lightened the right side of the target without overwriting it entirely.
• **Screen**—Compounds the lightness of the source pixels with the lightness of the target pixels so that the result color is always lighter. The effect is similar to projecting light from two photographic slides onto the same “screen,” thus the title. With Screen, all pixels from the source lighten all pixels in the target to the extent that they are light. So in the example, even the very slight brightness of the gray pixels in square S brightens pixels near the bridge of Abe’s nose.

• **Color Dodge**—A complex blend mode that involves borrowing color value from surrounding pixels. Light and highly saturated (richly colored) pixels in the source lighten and brighten pixels in the target and transfer their color value to the target. But also, pixels in the source with low saturation (blacks, grays, and whites) but high opacity borrow color data from surrounding pixels and apply that data to the target area, brightening it in the process. Areas of rich saturation are expanded into nearby light areas, introducing new areas of direct and sharp contrast. Notice in the example how the light gray letter L brings in red values from the gradient and from Abe’s face, rendering the formerly green area bright yellow. And notice also how the lighter areas in squares K and J make the right edge of the target bright cyan and yellow, respectively. Meanwhile, the full color value of the three color bands is transferred to the target, leaving white areas white but rendering dark areas bright with color. The dark pixels at the left of the source have no effect on the target.

• **Linear Dodge**—Applies a similar dodge technique as Color Dodge, but the difference here is that Linear Dodge brightens and saturates pixels in the target by as much as the source pixels are both bright and opaque. Dark, opaque pixels have little effect on the target, but opaque midtones brighten the target by the degree of their opacity. This is how the dark-ish letter K in the example makes a light mark on Abe’s hair, even though the half-opaque pixels in the source are actually lighter than the K. This is also why the equally opaque letter S brightens its target area, while its dark surroundings have no effect. Meanwhile, the full-intensity color bands brighten the dark zones in Abe’s face, leaving the white zones white.
About Blend Modes

- **Overlay**—Applies the formula for the **Multiply** mode to dark target pixels and the formula for the **Screen** mode to light target pixels. The result is that lights compound with lights and darks compound with darks. The result includes more stark contrasts with fewer middle values. The **Overlay** mode is one way to create a ghostly image of one layer on top of another, especially at less than full opacity.

- **Soft Light**—A different combination of the **Multiply** and **Screen** modes, with similar results except with less augmented saturation. To compound both brightness and darkness, the source and target pixels are both blended using **Multiply**, then the product is multiplied back into the inverse of the source. Separately, both source and target are blended using **Screen**, with the result multiplied back into the source. The two interim results—darker darks and lighter lights, but with less dramatic saturation—are then added together, resulting in more natural saturation. As you can see in the example, the differences between **Soft Light** and **Overlay** are subtle, but noticeable.

- **Hard Light**—Applies the formula for the **Multiply** mode to dark **source** pixels (as opposed to **Overlay**, which tests the **target** pixels), and the formula for the **Screen** mode to light source pixels. The result with **Hard Light** is that the source (what you’re adding) becomes more prominent than the target; with **Overlay**, the target (what you’re adding to) remains prominent.

- **Vivid Light**—A blend of **Color Dodge** and **Color Burn** modes, the **Vivid Light** mode applies the dodge formula to light target pixels and the burn formula to dark target pixels. The result contains either very highly saturated bright pixels or highly unsaturated dark pixels. Middle gray values in the source make the least impact on the target. Whereas the bright whites in squares K and J in our example leave the right side of the target bright cyan and yellow, the gray values from the source leave the subtle midtones in Abe’s face largely untouched.
• **Linear Light**—A blend of **Linear Burn** and **Linear Dodge** modes, the **Linear Light** mode applies the burn formula to dark pixels in the target and the dodge formula to light pixels in the target. The result has greater impact on the overall lightness of pixels than on their saturation. Notice in the example how the patterns in the color bands are less graduated. For instance, in Abe’s beard, the darkest darks were made more prominent by adding blue (the color from the dark side of the source, if you will); whereas the lightest lights were made more prominent by adding magenta from the light side.

• **Pin Light**—Uses a combination of the **Lighten** and **Darken** formulas. In essence, when pixels from the source are already significantly darker than the target, the **Darken** formula is applied to blend those dark pixels with the target; when pixels from the source are already significantly lighter than the target, the **Lighten** formula blends them with the target. But there’s a notable bias against middle values, such as 50% gray. In the example, notice how the middle gray value on the right side of square S did not darken the center of Abe’s forehead, and the same middle gray value on the left side of square J did not brighten the shadow under Abe’s nose. So middle luminance values in the source are always tossed out of consideration; only extreme values apply.

• **Hard Mix**—Uses simpler mathematics to come up with a psychedelic, posterized result: When the source and target are combined, the hue components of the result are reset either to full strength (255) or no strength (0). The result is comprised only of basic primary colors (red, green, blue), secondary colors (cyan, magenta, yellow), black, and white. In the example, the only reason some middle tones remain in squares K and L is because the opacity for those squares at the source was 50%.

• **Difference**—Subtracts the RGB color values of the source pixels from those of the target pixels. The result is often a completely different color than the original target color. That difference is proportional to the difference between the source and target colors—which, the name of the blend mode. It helps to remember these rules: First, black is considered “zero,” and subtracting zero from anything leaves you with what you had to begin with. Second, subtracting a color from itself results in black. Third, subtracting double a color’s value results in that same value, so medium gray minus white equals medium gray. Notice
in the example how the half-opaque whites in square K ended up making the background of that square entirely middle gray from top to bottom, whereas the bright whites in square J made the background behind Abe’s neck graduate from pink to cyan (as opposed to green to blue).

- **Exclusion**—In some ways the opposite of **Difference**, but with a twist: Unlike **Difference**, **Exclusion** mode *adds* the RGB color values of the source and the target pixels. But to bring the result back to the reality range of 0–255, twice the average of the source and target values are subtracted from the sum. **Exclusion**’s results are similar to **Difference**’s when the differences between the source and target are extreme. But as the example shows, when the differences are more negligible, the result is a middle value. Notice how the middle grays actually blur into one another in the center of Abe’s head with **Exclusion** mode, whereas they created stark contrasts for **Difference** mode.

- **Hue**—Substitutes the hue component of the target pixels with that of the source pixels, leaving saturation and luminance intact. In the example, Abe’s hair, the bridge of his nose, and his beard are all now slightly tinted. But the unsaturated portions of all four squares did create noticeable color noise behind Abe’s head. This is because grayscales are all considered “Hue #0,” so the unsaturated pixels’ hue value is being applied to the target even when there’s no color in those pixels.

- **Saturation**—Substitutes the saturation component of the target pixels with that of the source pixels, leaving hue and luminance intact. In the example, the fully opaque squares S and J removed all saturation from the target, so the result is as unsaturated as the source. The half-opaque squares K and L let some original color show through. But the fully opaque, fully saturated color bands “went to town” with the middle values in Abe’s face. The light pixels look untouched because white is considered fully saturated, while any fully saturated hue ends up translating to white anyway. So the white on the bridge of Abe’s nose, where it was white before, is actually “very light green.”
• **Color**—Substitutes the hue and saturation components of the target pixels with those of the source pixels, leaving the luminance component intact. Result pixels, therefore, are as light or as dark as they were before, but they might be colored. Technically, “color” is achieved by mixing hue and saturation (consider that there is no hue that represents shades such as russet, cobalt, and aquamarine). In the example, squares S and J are now unsaturated, as they were in the source. But as color is added by the color bands, the contrasts in Abe’s face remain as they were.

• **Luminosity**—Substitutes the luminance (or “luminosity”) component of the target pixels with that of the source pixels, leaving the hue and saturation components intact. This leaves target pixels colored as they were before, but made blacker or whiter by the source. Notice in the example how the color bands lent nothing but their brightness to the result.