

Blend Modes



- [What is a blend mode?](#)
 - [Browser compatibility](#)
 - [mix-blend-mode](#)
 - [background-blend-mode](#)
 - [Separable blend modes](#)
 - [Normal](#)
 - [Multiply](#)
 - [Screen](#)
 - [Overlay](#)
 - [Darken](#)
 - [Lighten](#)
 - [Color dodge](#)
 - [Color burn](#)
 - [Hard light](#)
 - [Soft light](#)
 - [Difference](#)
 - [Exclusion](#)
 - [Non-separable blend modes](#)
 - [Hue](#)
 - [Saturation](#)
 - [Color](#)
 - [Luminosity](#)
 - [The isolation property](#)
-

[Duotone](#) is a popular color treatment for photography which makes an image look like it is only made up of two contrasting colors: one for highlights and the other for lowlights. How do you do this with CSS though?

```
{% Codepen { user: 'web-dot-dev', id: 'bGgvYMG' } %}
```

Using blend modes—and other techniques you have learned about, such as [filters](#) and [pseudo-elements](#)—you can apply this effect to any image.

What is a blend mode?

Blend modes are commonly used in design tools such as Photoshop to create a compositional effect by mixing colors from two or more layers. By changing how colors mix, you can achieve really interesting visual effects. You can also use blend modes as a utility, such as isolating an image that has a white background, so it appears to have a transparent background.

```
{% Codepen { user: 'web-dot-dev', id: 'dyNmJor' } %}
```

You can use most of the blend modes available in a design tool with CSS, using the `mix-blend-mode` or the `background-blend-mode` properties. The `mix-blend-mode` applies blending to a whole element and the `background-blend-mode` applies blending to the background of an element.

You use `background-blend-mode` when you have multiple backgrounds on an element and want them all to blend into each other.

```
{% Codepen { user: 'web-dot-dev', id: 'XWpEVGO' } %}
```

The `mix-blend-mode` affects the entire element, including its pseudo-elements. One use-case is in the initial example of a duotone image, which has color layers applied to the element through its pseudo-elements.

```
{% Codepen { user: 'web-dot-dev', id: 'bGgvYMG' } %}
```

Blend modes fall into two categories: separable and non-separable. A separable blend mode considers each color component, such as RGB, individually. A non-separable blend mode considers all color components equally.

Browser compatibility

`mix-blend-mode`

```
{% BrowserCompat 'css.properties.mix-blend-mode' %}
```

`background-blend-mode`

```
{% BrowserCompat 'css.properties.background-blend-mode' %}
```

Separable blend modes

Normal

This is the default blend mode and changes nothing about how an element blends with others.

Multiply

The `multiply` blend mode is like stacking multiple transparencies on top of each other. White pixels will appear transparent, and black pixels will appear black. Anything in between will multiply its luminosity (light) values. This means lights get much lighter and darks, darker—most often producing a darker result.

```
.my-element {  
  mix-blend-mode: multiply;  
}
```

{% Codepen { user: 'web-dot-dev', id: 'vYgRdOy', height: 450 } %}

Screen

Using `screen` multiplies the *light* values—an inverse effect to `multiply`, and will most often produce a brighter result.

```
.my-element {  
  mix-blend-mode: screen;  
}
```

{% Codepen { user: 'web-dot-dev', id: 'JjELpYo', height: 450 } %}

Overlay

This blend mode—`overlay`—combines `multiply` and `screen`. Base dark colors become darker and base light colors become lighter. Mid-range colors, such as a 50% gray, are unaffected.

```
.my-element {  
  mix-blend-mode: overlay;
```

```
}
```

```
{% Codepen { user: 'web-dot-dev', id: 'BaprYom', height: 450 } %}
```

Darken

The `darken` blend mode compares the source image and backdrop image's dark color luminosity and selects the darkest of the two. It does this by comparing rgb values instead of luminosity (like `multiply` and `screen` would do), for each color channel. With `darken` and `lighten`, new color values are often created from this comparison process.

```
.my-element {  
  mix-blend-mode: darken;  
}
```

```
{% Codepen { user: 'web-dot-dev', id: 'dyNmdGM', height: 450 } %}
```

Lighten

Using `lighten` does the exact opposite of `darken`.

```
.my-element {  
  mix-blend-mode: lighten;  
}
```

```
{% Codepen { user: 'web-dot-dev', id: 'OJWvQNO', height: 450 } %}
```

Color dodge

If you use `color-dodge`, it lightens the background color to reflect the source color. Pure black colors see no effect from this mode.

```
.my-element {  
  mix-blend-mode: color-dodge;  
}
```

```
{% Codepen { user: 'web-dot-dev', id: 'abpYqpz', height: 450 } %}
```

Color burn

The `color-burn` blend mode is very similar to the `multiply` blend mode, but increases contrast, resulting in more saturated mid-tones and reduced highlights.

```
.my-element {  
  mix-blend-mode: color-burn;  
}
```

{% Codepen { user: 'web-dot-dev', id: 'gOgevmG', height: 450 } %}

Hard light

Using `hard-light` creates a vivid contrast. This blend mode either screens or multiplies luminosity values. If the pixel value is lighter than 50% gray, the image is lightened, as if it were screened. If it is darker, it's multiplied.

```
.my-element {  
  mix-blend-mode: hard-light;  
}
```

{% Codepen { user: 'web-dot-dev', id: 'ZELxreN', height: 450 } %}

Soft light

The `soft-light` blend mode is a less-harsh version of `overlay`. It works in very much the same way with less contrast.

```
.my-element {  
  mix-blend-mode: soft-light;  
}
```

{% Codepen { user: 'web-dot-dev', id: 'OJWvQmQ', height: 450 } %}

Difference

A good way to picture how `difference` works is a bit like how a photo negative works. The `difference` blend mode takes the difference value of each pixel, inverting light colors. If the color values are identical, they become black. Differences in the values will invert.

```
.my-element {  
  mix-blend-mode: difference;  
}
```

```
}
```

```
{% Codepen { user: 'web-dot-dev', id: 'mdRxXwM', height: 450 } %}
```

Exclusion

Using `exclusion` is very similar to `difference`, but instead of returning black for identical pixels, it will return 50% gray, resulting in a softer output with less contrast.

```
.my-element {  
  mix-blend-mode: exclusion;  
}
```

```
{% Codepen { user: 'web-dot-dev', id: 'JjELpmb', height: 450 } %}
```

Non-separable blend modes

You can think of these blend modes like HSL [color](#) components. Each takes a specific component value from the active layer and mixes it with other component values.

Hue

The `hue` blend mode takes the hue of the source color and applies it to the saturation and luminosity of the backdrop color.

```
.my-element {  
  mix-blend-mode: hue;  
}
```

```
{% Codepen { user: 'web-dot-dev', id: 'BaprYGO' } %}
```

Saturation

This works like `hue`, but using `saturation` as the blend mode applies the saturation of the source color to the hue and luminosity of the backdrop color.

```
.my-element {  
  mix-blend-mode: saturation;  
}
```

```
}
```

```
{% Codepen { user: 'web-dot-dev', id: 'QWdmQoP' } %}
```

Color

The `color` blend mode will create a color from the hue and saturation of the source color and the luminosity of the backdrop color.

```
.my-element {  
  mix-blend-mode: color;  
}
```

```
{% Codepen { user: 'web-dot-dev', id: 'jOyzZRo' } %}
```

Luminosity

Lastly, `luminosity` is the inverse of `color`. It creates a color with the luminosity of the source color and the hue and saturation of the backdrop color.

```
.my-element {  
  mix-blend-mode: luminosity;  
}
```

```
{% Codepen { user: 'web-dot-dev', id: 'zYNWWOK' } %}
```

The `isolation` property

```
{% BrowserCompat 'css.properties.isolation' %}
```

If you set the `isolation` property to have a value of `isolate`, it will create a new stacking context, which will prevent it from blending with a backdrop layer. As you learned in the [z-index module](#), when you create a new stacking context, that layer becomes the base layer. This means that parent-level blend modes will no longer apply, but elements inside of an element with `isolation: isolate` set can *still* blend.

Note that this doesn't work with `background-blend-mode` because the background property is already isolated.

```
{% Codepen { user: 'web-dot-dev', id: 'JjELLXy' } %}
```

{% Assessment 'blend-modes' %}

0000 #2

0000 0000 2026-10-31 22:45:16 UTC

0000 0000 2026-10-31 22:45:17 UTC