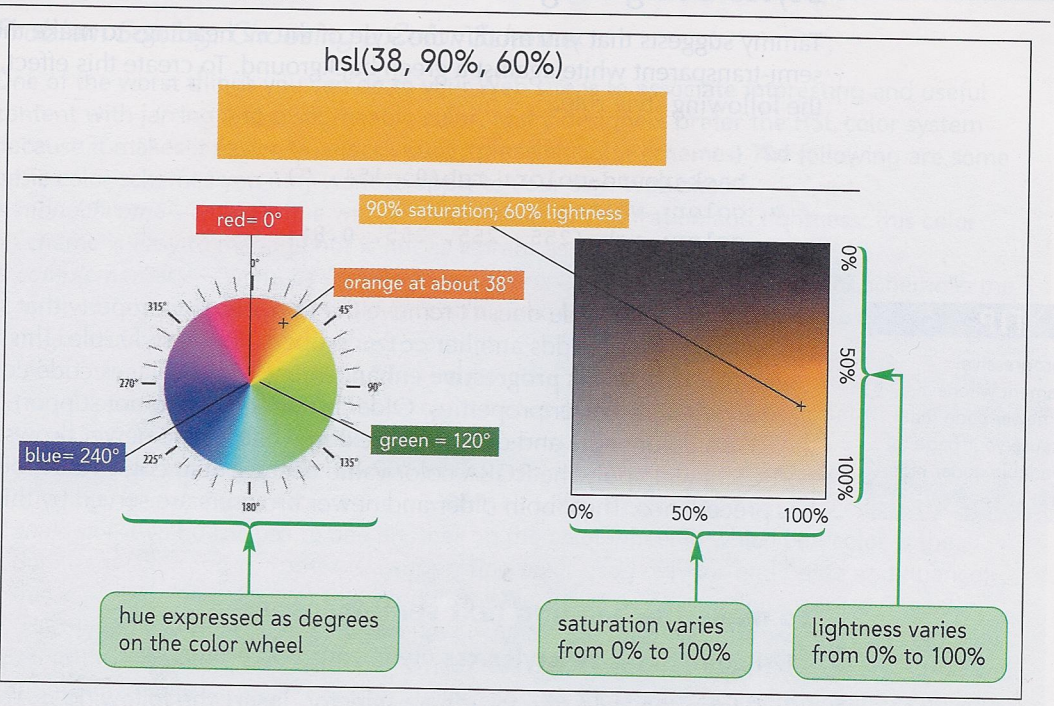
Figure 3-14 HSL color saturation model



Graphic designers consider HSL easier to use because it allows you to guess at an initial color based on hue and then tweak the saturation and lightness values to fine-tune the final color. This is more difficult in the RGB model because you have to balance three completely different colors to achieve the right mix. For example, the RGB equivalent to the color in Figure 3-14 would be the color value rgb(245, 177, 61); however, it's not immediately apparent why that mixture of red, green, and blue would result in that particular shade of orange.

Opacity Values in CSS3

CSS3 also allows page designers to augment RGB and HSL color values by specifying a color's opacity. Opacity defines how much of the colors below the surface of the current object show through to affect its appearance. The opacity of a color can be specified using either of the following `rgba` and `hsla` color values

```
rgba(red, green, blue, opacity)
hsla(hue, saturation, lightness, opacity)
```

where *opacity* sets the transparency of the color as a decimal ranging from 0 (completely transparent) up to 1.0 (completely opaque). For example, the following style displays the text of `h1` headings in a medium shade of orange at 70% opacity:

```
hsla(38, 90%, 60%, 0.7)
```

TIP

The a in rgba and hsla stands for *alpha* and refers to the alpha channel, a color concept developed in the 1970s to add transparency to the color model.

With semi-transparent colors, the final color rendered by a browser depends on the background color of the parent element. Displayed against a white background, this medium orange color would appear in a lighter shade of orange, while displayed against a black background it would appear as very dark orange. The advantage of using semi-transparent colors is that it makes it easier to create a color theme in which similarly tinted colors are used throughout the page.