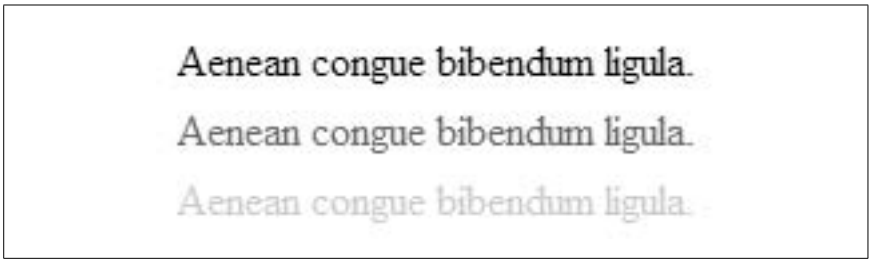

Figure 20-1. Changing the foreground color

- It is valid to add a foreground color to images. The content of the image won't be affected by it, of course, but the color will be used for the image border if one is specified.
- If there is both a foreground color and a border color property applied to an element, the `border-color` property always overrides `color` for the border color.
- If you want to change the color of all the text in a document, apply the `color` property to the `body` element. Color may be assigned globally to the `html` element or by using the universal selector (*) as well, but this is less common due to irregularities in inheritance and problems with form elements in some browsers. Be aware, however, that on some older browsers, table elements do not properly inherit properties from the `body`, so text within tables would go back to the default text color. To be on the safe side, you can make a color declaration for `body` and the relevant table elements, like this:

  ```
  body, table, td, th { color: fuschia; } /* ok, maybe not fuschia */
  ```

- You can apply the `color` property to form input elements like buttons and pull-down menus. Although it's valid use of CSS, it is not supported consistently across browsers. Make sure that your design is legible even if your chosen form input colors do not display the way you intended.

Background Color

It's been common practice to add a background color to a page using the `bgcolor` attribute in the `body` element in HTML. With CSS, not only can you provide a background color for a whole page, but for any element in the document, both block-level and inline. Boxes of color anywhere you want them…and no tables required!

Background color is declared with the (no surprise here) `background-color` attribute.

background-color

Values: `<color> | transparent | inherit`

Initial Value: `transparent`

Applies to: All elements

Inherited: No

Background properties are applied to the "canvas" behind an element. With regard to the box model, background colors fill the content area, the padding area, and extend behind the border to its outer edge. This means that if the border has gaps, the background color will show through.

Background properties are not inherited, but because the default value is `transparent`, the parent's background color shows through its child elements. Figure 20-2 shows an example of the `background-color` property. Note how a little padding added to the element gives the content a little breathing room inside the resulting rectangular colored box.

```
p {padding: 5px;}
p.a {background-color: #333333;}
p.b {background-color: #666666;}
p.c {background-color: #CCCCCC;}

<p class="a">Fusce rhoncus facilisis sapien.</p>
<p class="b">Fusce rhoncus facilisis sapien.</p>
<p class="c">Fusce rhoncus facilisis sapien.</p>
```

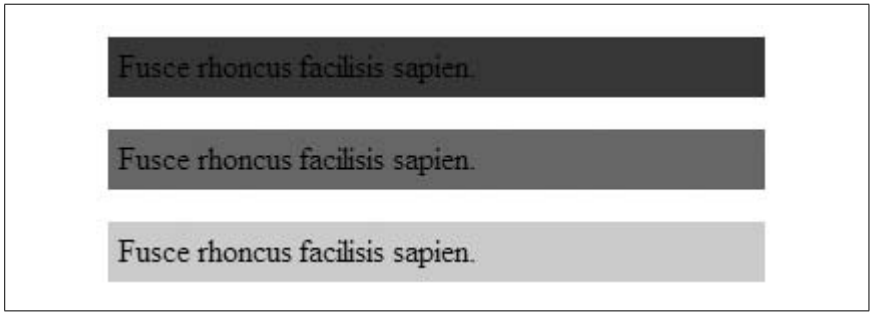

Figure 20-2. The background-color property

Background Images

Once again, CSS beats HTML hands down in the background department (but then, HTML was never intended to be fussing around with things like background images). With CSS, you're not stuck with a repeating tile pattern, and you can position a background image wherever you like. You can also apply a background image to any element in the document.

This section covers the CSS properties for adding and manipulating background images, with the basic `background-image` property as a starting point and moving on to more advanced background image behaviors such as controlling repeating patterns, positioning the image within the element, and preventing the image from scrolling off the page.

Background Image Tips

When working with background images, keep these guidelines and tips in mind:

- Use an image that won't interfere with the legibility of the text over it.
- As usual for the Web, it is important to keep the file size as small as possible for background images, which may lag behind the display of the rest of the page.
- Provide a `background-color` that matches the primary color of the image in the background. If the background image fails to display, at least the overall design of the page will be similar. This is particularly important if the text color would be illegible against the default white (or light gray) browser background.

background-image

Values: `<uri> | none | inherit`

Initial value: `none`

Applies to: All elements

Inherited: No

`background-image` is the basic property for adding an image to the background (the "canvas") of an element. When applied to the `body` element, it functions just like the `background` attribute, causing the image to tile horizontally and vertically until it fills the browser window. Unlike the `background` attribute, the `background-image` property can be applied to any element in the document, both block and inline.

Figure 20-3 shows background images applied to a whole page and to an individual paragraph using these style rules.

```
body {background-image: url(stripes.gif);}
p.highlight {background-image: url(dots.gif);}
```

The `background-image` property is not inherited (in fact, none of the background properties are). Instead, the pattern merely shows through the descendant elements because their background colors are transparent by default. If tiling images were inherited, the result would be a mess in which a new tiling pattern would begin in the top-left corner of each new element on the page.

If a `background-color` property is also specified, the image is overlaid on top of the color. Always provide a similar background color for an element when you add a background image. That way, if the image fails to load, the text and foreground elements maintain a readable contrast against the background.

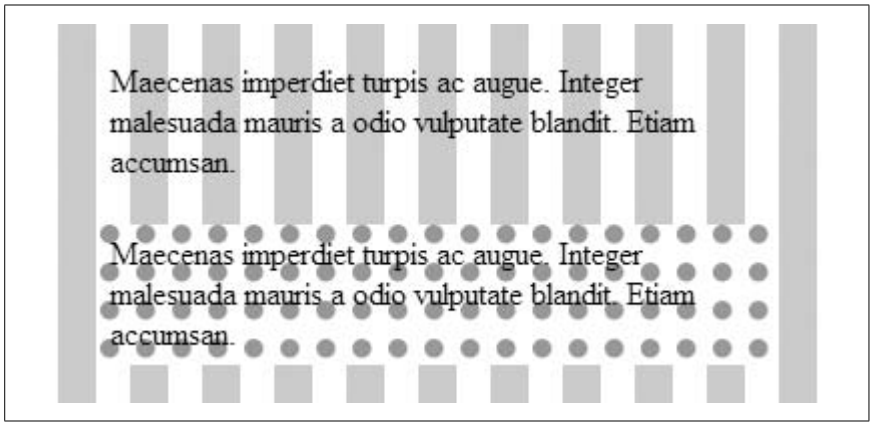

Figure 20-3. The background-image property applied to an entire page and a single paragraph

Background Tiling (Repeat)

Use the `background-repeat` property to prevent the background image from tiling (repeating) or to make it tile in one direction only.

background-repeat

Values: `repeat | repeat-x | repeat-y | no-repeat | inherit`

Initial value: `repeat`

Applies to: All elements

Inherited: No

By default, background images tile both horizontally and vertically. You can turn this behavior off and make the image appear just once by using the `no-repeat` keyword value as shown in Figure 20-4.

```
div.ringo {background-image: url(starr.gif); background-repeat: no-repeat}
```

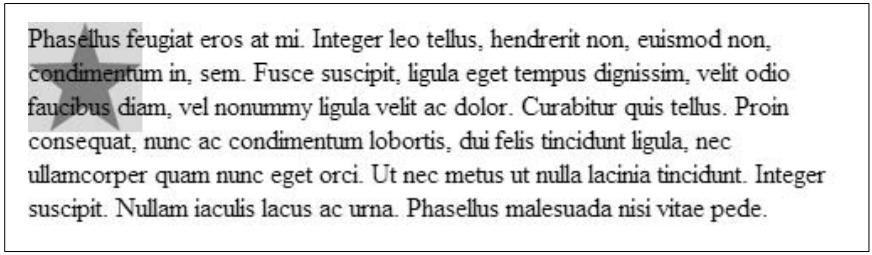

Figure 20-4. Turning off tiling with no-repeat

repeat-x allows the image to repeat only horizontally. Similarly, repeat-y allows the image to repeat only on the vertical axis. Examples of both are shown in Figure 20-5.

```
div.horiz {background-image: url(starr.gif); background-repeat: repeat-x;}
div.vert {background-image: url(starr.gif); background-repeat: repeat-y;}
```


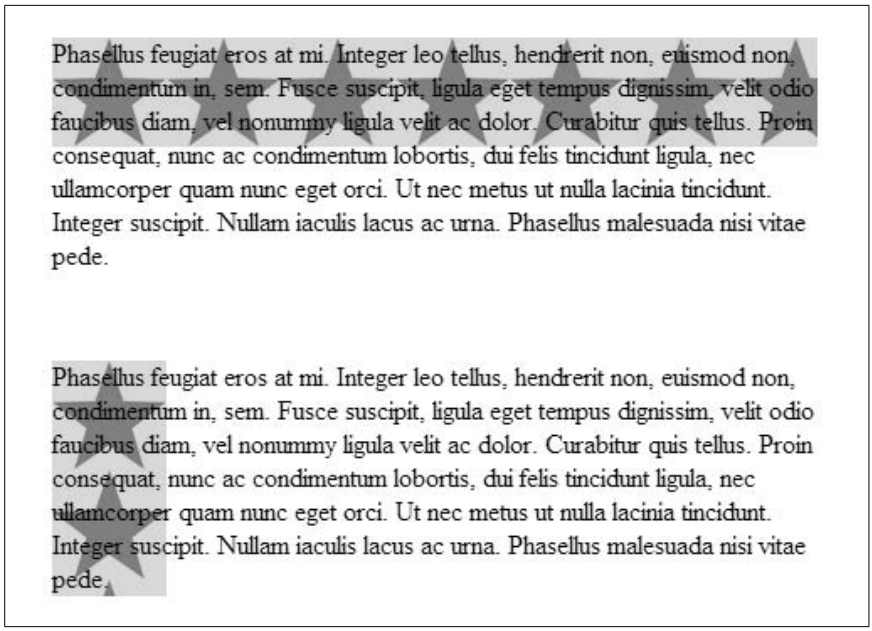

Figure 20-5. Horizontal and vertical tiling

Notice that in the examples in Figure 20-4 and Figure 20-5, the tiling begins in the top-left corner of the viewing area (in most cases, the browser window). But the background image doesn't necessarily need to start there, as discussed next.

Background Position

The background-position property specifies the position of the *origin image* in the background of the element. You can think of the origin image as the first image that is placed in the background. It's also the starting point from which repeated (tiling) images extend.

background-position

Values: [[<percentage> | <length> | left | center | right] [<percentage> |<length> | top | center | bottom]? } | [[left | center | right] || [top | center | bottom]] | inherit

Initial value: 0% 0% /* same as left top */

Applies to: All elements

Inherited: No

The background-position property specifies the initial position of the origin image. Measurements are relative to the top-left corner of the padding area for the element (the default position). It is not placed behind the border, although if the image is set to repeat, the repeating images will extend and show through the border area when the border style has gaps.

Figure 20-6 shows a simple example of the background-position property, The background-repeat property has been set to no-repeat to make the position of the origin image clear.

```
body { background-image: url(bigstar.gif);
       background-position: top center;
       background-repeat: no-repeat; }
```

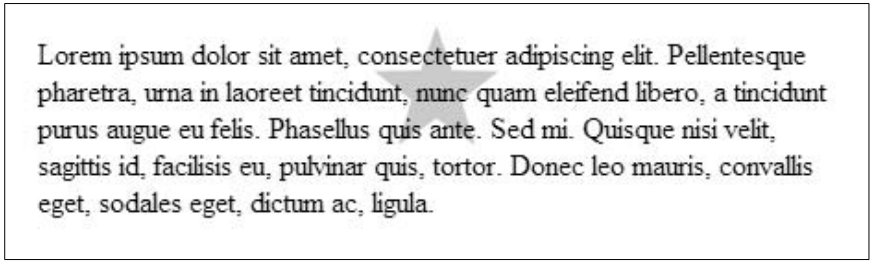

Figure 20-6. The background-position property

There are a number of methods for specifying the value of background-position. The options after "Values" above may look like gobbledy-gook, but it boils down to three general systems: keywords, lengths, and percentages.

Keyword positioning

The *keyword values* for positioning are left, right, top, bottom, and center. Each value (except center) places the specified edge of the image all the way to the respective edge of the element. For example, the left value pushes the left edge of the image all the way to the left edge of the background area. The center value places the center of the image in the center of the element. And so on.

Keywords are usually used in pairs, as in these examples:

```
{background-position: left top;}
{background-position: right center;}
{background-position: center bottom;}
```

Each of these positions is demonstrated in Figure 20-7.

The order of the keywords is not important according to the CSS 2 Recommendation, but Netscape 6 and related browsers require that the horizontal measurement be provided first, so it's good practice to provide them in horizontal/vertical order just to be safe.

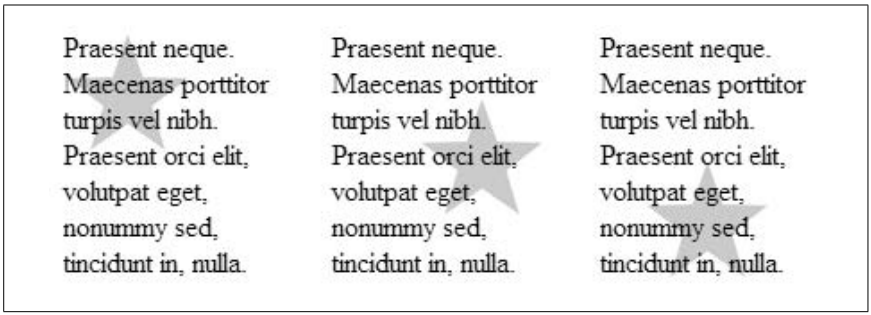

Figure 20-7. Positioning with keywords

If you only provide one keyword, the missing keyword is assumed to be `center`. Therefore the second and third previous examples could also be written like this:

```
{background-position: right;}
{background-position: bottom;}
```

Length measurements

It is also possible to specify the position of the origin image in units of length. When length units are provided, they are interpreted as the distance from the top-left corner of the padding area to the top-left corner of the image. Length values must be provided with the horizontal measurement first.

In this example, the top-left corner of the image will start 150 pixels from the left edge and 15 pixels from the top of the intro paragraph, as shown in Figure 20-8.

```
p.intro { background-image: url(something.gif);
          background-position: 150px 15px;
           background-repeat: no-repeat;}
```

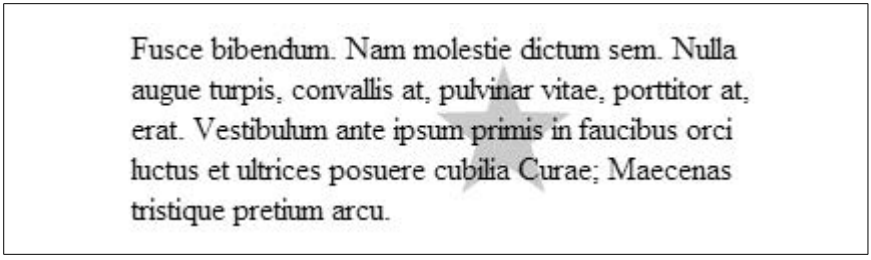

Figure 20-8. Positioning with length measurements

It is valid CSS to specify negative length measurements, thus pulling the image out of the visible background area of the element. Not all browsers currently support negative background image values, so be sure to test on your targeted browsers.

Percentage values

Percentage values follow the same basic positioning model as keywords, but they provide a more fine-tuned control over the image placement. Percentage values

are given in horizontal/vertical pairs, with a default value of 0% 0%, which places the upper-left corner of the image in the upper-left corner of the element.

Each percentage value specified applies to both the background canvas area and the image itself. A few simple examples should make this clear.

- The percentage values 50% 50% place the center of the image in the center of the element.
- The percentage values 100% 100% place the bottom-right corner of the image in the bottom-right corner of the element.
- The percentage values 10% 25% match a point that is 10% from the left and 25% from the top edge of the image with the same point in the element.

As for keywords, when only one percentage value is provided, the other is assumed to be 50%.

It is fine to mix length and percentage values, which makes it easy to specify that an image should be centered horizontally in the element but appear exactly 25 pixels from its top edge. CSS 2.1 also allows length and keywords to be combined, but not all browsers support that combination as of this writing.

Positioning repeating images

In the previous examples, the background-repeat property was set to no-repeat for the sake of clarity. The principles of positioning do not change when the image is allowed to tile. When both properties are provided, the positioned origin image functions as the starting point for the repeating pattern.

It is significant to note that the tile pattern extends in both directions from the origin image. Therefore, if an image is positioned in the center of the element and the repeat is set to horizontal, the tiles will repeat on both the left and right of the centered image. Similarly, a vertical pattern extends both up and down from the origin image. There is currently no way to make the repeat go in one direction only in CSS 2.1, but that functionality may be added to a later specification.

In Figure 20-9, both the background-position and background-repeat properties are used to guarantee that one image is always centered in the browser window.

```
body { background-image: url(something.gif);
       background-position: center;
       background-repeat: repeat-x; }
```

Background Attachment

The default behavior for a background image in CSS is to scroll along with the document when the document scrolls, as though it is stuck to the element. This is the also the way background images applied with the body element function.

CSS provides the background-attachment property that frees the background image from the content and allows it to stay in a fixed position when the content of the document scrolls. In effect, it disconnects the image from the content flow and attaches it to the viewing area (typically a browser window).

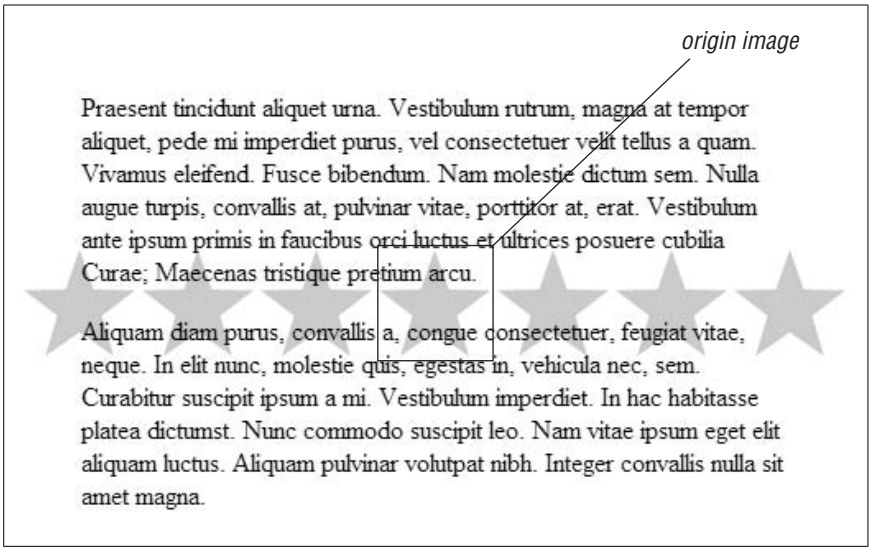

Figure 20-9. Combining position and tiling

background-attachment

Values: `scroll | fixed | inherit`

Initial value: `scroll`

Applies to: All elements

Inherited: No

The default value is `scroll`, so the origin image will scroll if you do not specify the `background-attachment` property. The other alternative is `fixed`, which fixes the image in one place relative to the viewing area.

In this example, the background image is fixed, as demonstrated in Figure 20-10.

```
body { background-image: url(img/star.gif);
       background-position: top;
       background-repeat: no-repeat;
       background-attachment: fixed; }
```

The other primary difference between a fixed origin image and a scrolling one is that for fixed images, the values of `background-position` are relative to the top-left corner of the viewing area, not the element itself.

This creates an interesting effect when a fixed background pattern is applied to an element other than `body`. The image stays in the same place and the element's containing box reveals a rectangular slice of the background at a time. Unfortunately, Internet Explorer for Windows Versions 6 and earlier do not support fixed background images on elements other than `body`. Non-`body` support is promised in Version 7, in beta as of this writing.

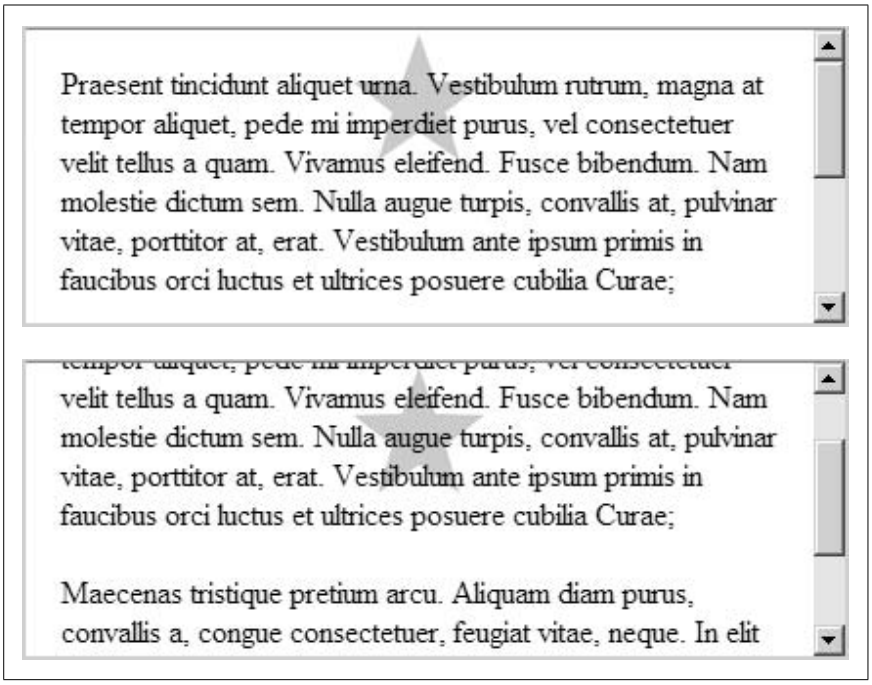

Figure 20-10. Preventing scrolling with the background-attachment property

Eric Meyer demonstrates some interesting effects using fixed images as backgrounds for several elements on a page on his page *www.meyerweb.com/eric/css/edge/complexspiral/glassy.html*. To see the full effect, make sure you are using a standards-compliant browser other than Internet Explorer.

Combining Background Properties

CSS provides a handy `background` shorthand property that allows all the background properties to be combined in one style rule, similar to the `font` shorthand property (see Chapter 18).

background

Values: `[<'background-color'>||<'background-image'>||<'background-repeat'>||<'background-attachment'>||<'background-position'>]|inherit`

Applies to: All elements

Inherited: No

The background shorthand property takes a value from any of the background-related properties. There are no required values, and the values may appear in any order. The only restriction is that if two values are provided for background-position they must appear together and with the horizontal value first, followed immediately by vertical.

The following are valid examples of the background shorthand property:

```
body {background: url(superstar.gif) fixed top center no-repeat; }
div.intro {background: repeat-x url(topborder.gif) red; }
p {background: #336600; }
```


Watch for Accidental Overrides

Bear in mind that because background is a shorthand property, values that are omitted will be reset to the default for those properties. That combined with the fact that later rules in a style sheet override previous rules makes it easy to accidentally override previously declared background properties with the defaults. In this example, the background image *dots.gif* will not be applied to h3 elements, because by omitting a value for background-image, it essentially set that value to none.

```
h1, h2, h3 { background: red url(dots.gif) repeat-x;}
h3 {background: blue; }
```

To override particular properties, be sure to use the specific background property you intend to change (background-color would be appropriate for the h3 in the example). When using the background (or any shorthand) property, pay attention to related rules earlier in the style sheet, or be sure that every property is specified.