Adding Images and Other Media

It's like what they say about the perfect picture book. The art and the text stand alone, but together, they create something even better. —Deborah Wiles

This chapter covers

- Embedding an image on a web page
- Working with background images
- Optimizing images for the web
- Adding videos, music, and other media

When you come across a page that's nothing but text, how does it make you feel? It probably makes you feel disappointed or perhaps even sad. And unless the text is absorbing and the typography exceptionally good, it also probably makes you want to click the Back button and look for some place where your sore eyes can catch a break. You don't want people feeling disappointed, sad, or eager to leave your site, so throw them a visual bone or two by sprucing up your pages with images and perhaps even a video once in a while. In this chapter, I show you how it's done.

Adding Images and Other Media

Lesson 6.1: Adding an Image to the Page

Covers: The `img` element

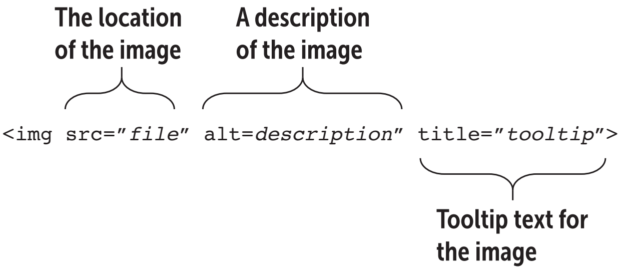
Online: wdpg.io/6-1-0

So far in this book, you've seen that the innards of a web page are text with a few HTML tags and CSS rules sprinkled strategically here and there. So you may be wondering how images fit into this text-only landscape. The short answer is that they don't! Unlike with a word processing document or a presentation, you don't insert images directly into a web page. Instead, you upload the image as a separate file to your website and then insert into your page text a special HTML tag that tells the browser where to locate the image. Then the browser retrieves the file from the server and displays the image on the page in the location you specified.

The special tag that gets the browser to add an image to a web page is the `img` element, which uses the partial syntax shown in Figure 6.1.

FAQ

Do I have to include the alt *attribute? Yes. Your web page won't validate unless every one of your* `<img>` *tags has an* `alt` *attribute present. If you don't want to use alt text for decorative or other nonessential images, you can set the* `alt` *attribute equal to the empty string ("").*



```
<img src="file" alt=description" title="tooltip">
```

The location of the image
A description of the image
Tooltip text for the image

Figure 6.1 You insert an image into a web page by using the `<img>` tag.

You have three attributes to consider here:

- `src`—This attribute (short for *source*) specifies where the image file is located. If the file is on a remote server, use the full URL of the file; if the file is in the same directory as the HTML file, use the name of the file; otherwise, use the image's path and filename. If you've created in your site's main folder a subfolder named `images`, and your image file is `logo.png`, your `src` value would be `/images/logo.png`.
- `alt`—This attribute (short for *alternative*) is a word or short phrase that describes the image and that could be used in place of the image in case the image file can't be displayed. A company logo, for example, might use the alternative text *logo*, preceded by the company name. Alt text is also used by screen readers and Braille apps to give the user some idea of what the image is.
- `title`—You can use this optional attribute to specify tooltip text that appears when the user hovers the mouse pointer over the image, as shown in the example that follows.

REMEMBER

If you don't yet have the image you want to use but know the image's final dimensions, you can insert a placeholder image to occupy the same space on the page until the image is ready to use. You have several ways to do this, but the easiest is to use a placeholder server, such as https://placeholder.com. In the `<img>` *tag, add* `src="https://via.placeholder.com/wxh"`*, where w and h are the width and height, respectively.*

Online: wdpg.io/6-1-2

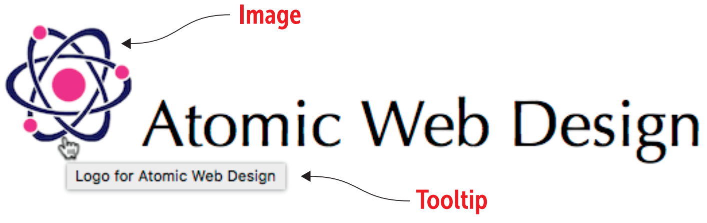
The following example shows an `img` element in action.

▶ *Example*

⇨ **Online:** wdpg.io/6-1-1

This example shows you how to use the `<img>` tag to add an image to a web page.

WEB PAGE



Image

Atomic Web Design

Logo for Atomic Web Design

Tooltip

CSS

```
span {
    font-family: Optima, Verdana, sans-serif;
    font-size: 3em;
}
```

HTML

```
<img
    src="/images/atomic-logo.png"
    alt="Atomic Web Design logo"
    title="Logo for Atomic Web Design">
<span>Atomic Web Design</span>
```

The location of the image file

The alternative text

The tooltip text

> *Your website needs a proper balance between textual and visual content. Awesome images or videos without text will give your visitor little to no useful data, but you might find it hard to engage users with large slabs of plain text.* —Helen Stark

Understanding Image File Formats

In the preceding example, you may have noticed that the image file was named `atomic-logo.png`, meaning that it uses the PNG image file format. That format is common on the web, but it's not the only one you can use. In fact, the web has standardized on four formats that account for almost all web imagery, and I summarize them in Table 6.1.

▶ Table 6.1 Image File Formats

Name	Extension	Description	Uses
GIF	`.gif`	The original web graphics format (the name is short for Graphics Interchange Format and it's pronounced giff or jiff). GIFs are limited to 256 colors, can have transparent backgrounds, and can be combined into short animations.	Use GIFs if you want to combine multiple images into a single animated image.

JPEG	`.jpg`
	`.jpeg`
		This format (which gets its name from Joint Photographic Experts Group and is pronounced jay-peg) supports complex images that have many millions of colors. The main advantage of JPEG files is that they're compressed, so even digitized photographs and other high-quality images can be a reasonably small size for faster downloading. Note, however, that JPEG compression is *lossy*, which means that it makes the image smaller by discarding redundant pixels. The higher the compression, the more pixels are discarded and the less sharp the image appears.	If you have a photo or similarly complex image, JPEG is almost always the best choice because it gives the smallest file size. How small is small enough for the web? You learn about that topic in "Optimizing Images" later in this chapter.

PNG	`.png`	This format (short for Portable Network Graphics and pronounced p-n-g or ping) supports millions of colors. It's a compressed format, but unlike JPEGs, PNGs use *lossless* compression. Images retain sharpness, but the file sizes can get quite big. PNG also supports transparency.	If you have an illustration or icon that uses solid colors, or a photo that contains large areas of near-solid color, PNG is best because it gives you a reasonably small file size while retaining excellent image quality. You can also use PNG if you need transparency effects.
SVG	`.svg`	This format (short for Scalable Vector Graphics) uses vectors rather than pixels to generate an image. These vectors are encoded as a set of instructions in XML format, meaning that the image can be altered in a text editor and can be manipulated to produce animations.	If you have a logo or icon and have a graphics program that can save files as SVG (such as Adobe Illustrator or Inkscape), this format is a good choice because it produces small files that can be scaled to any size without distortion.

Getting Graphics

The text part of a web page is, at least from a production standpoint, a piece of cake for most folks. Graphics, on the other hand, are another kettle of digital fish entirely. Creating a snazzy logo or eye-catching illustration requires a modicum of artistic talent, which is a bit harder to come by than basic typing skills.

LEARN

If you want to join the animated-GIF fun, lots of sites on the web can help. The easiest route is to use an online tool such as GIFCreator (http://gifcreator.me) *or GIFMaker.me* (http://gifmaker.me).

If you have such talent, however, you're laughing: Create the image in your favorite graphics program and save it in JPEG or PNG format. The nonartists in the crowd have to obtain their graphics goodies from other sources. Besides uploading your own photos or scanning your own images, you can find no shortage of other images floating around. Here are some ideas:

- Many programs (including Microsoft Office and most paint and illustration programs) come with clip-art libraries. Clip art is professional-quality artwork that you can incorporate into your own designs. In almost all cases, you're free to use the clip art in your own designs without worrying about copyright.
- Take advantage of the many graphics archives online. Sites all over the web store hundreds and even thousands, of images: stock photos, illustrations, icons, and more. Many of these images are free, but check each site's terms of use.
- Grab an image from a web page. When your browser displays a web page with an image, the corresponding graphics file is stored temporarily on your computer's hard disk. In most browsers, you can right-click the image to save that file permanently. As I elaborate in the note off to the side, however, there are copyright concerns, because you shouldn't use images that you don't own without permission and/or credit.

BEWARE

Don't forget that many images are the property of the people or companies that created them in the first place. Unless you're absolutely sure that a picture is in the public domain (for example, it comes with a Creative Commons license that lets you reuse the image), you need to get permission from the owner before using it. Either way, be sure to give credit to the image owner on your site.

Inserting an HTML5 Figure

Although many of your images are purely decorative or designed to catch a site visitor's eye, you may also use plenty of graphics that tie in with your page text. When you reference an image directly in the text, that image is known as a *figure*. In HTML5, a figure is a semantic page element that you designate with the `figure` element. If the figure has a caption, that caption too is a semantic element that you designate with the `figcaption` element. Here's the basic structure to use:

```
<figure>
    <img src="file" alt="description" title="tooltip">
    <figcaption>Caption text</figcaption>
</figure>
```

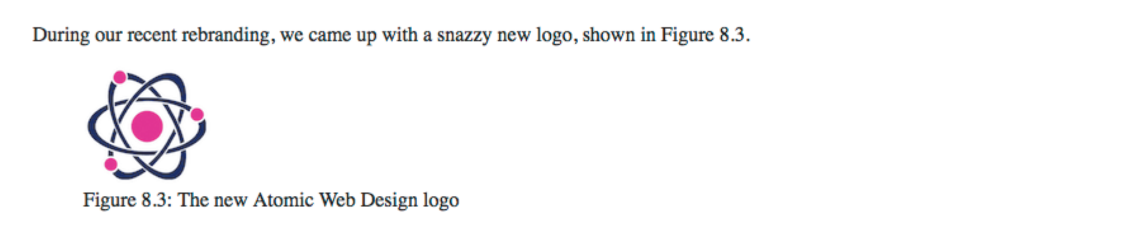
Following is an example.

Example

Online: wdpg.io/6-1-4

This example shows you how to use the `figure` and `figcaption` elements to designate an image as a figure.

WEB PAGE



During our recent rebranding, we came up with a snazzy new logo, shown in Figure 8.3.

Figure 8.3: The new Atomic Web Design logo

HTML

```
<p>
   During our recent rebranding, we came up with a
   snazzy new logo, shown in Figure 8.3.
</p>
<figure>
   <img
      src="/images/atomic-logo.png"
      alt="Atomic Web Design logo"
      title="Logo for Atomic Web Design">
   <figcaption>
      Figure 8.3: The new Atomic Web Design logo
   </figcaption>
</figure>
```

Enclose the caption in the figcaption element.

Enclose the image in the figure element.

Setting Up an Image as a Link

You already know that you can set up a word or phrase as a link, but you can do the same with images. You arrange things in the same way, surrounding the `<img>` tag with the `<a>` and `</a>` tags, like so:

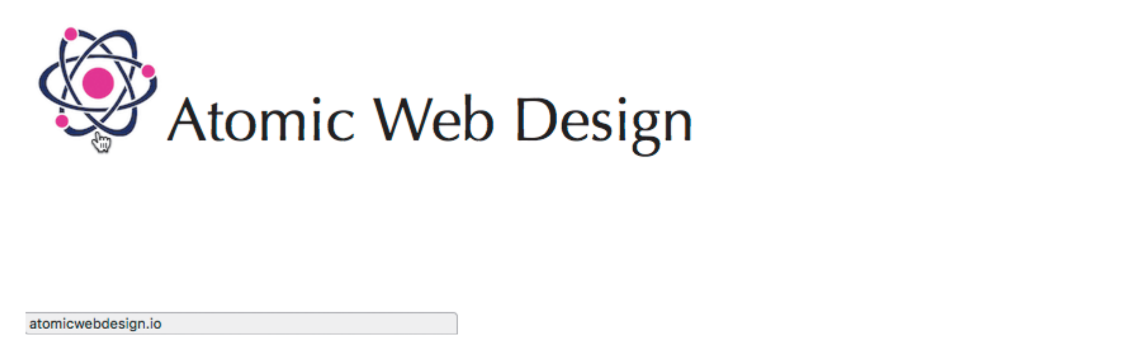
```
<a href="url"><img src="file"></a>
```

Here's an example.

▶ Example ⇨ **Online:** wdpg.io/6-1-6

This example shows you how to use the a element and the `img` *element to turn an image into a link.*

WEB PAGE



HTML

```
<a href="http://atomicwebdesign.io/">
<img src="/images/atomic-logo.png" alt="Atomic Web Design logo"
title="Logo for Atomic Web Design"></a>
<span>Atomic Web Design</span>
```

Enclose the image in the a element.

Using an Image as a Custom Bullet

As you learned in Chapter 2, CSS offers the `list-style-type` property that enables you to specify another type of bullet character to use with an unordered list. You can kick that property up a notch by using the `list-style-image` property to specify an image to use as a custom bullet:

```
ul {
    list-style-image: url(file);
}
```

As with the `<img>` tag, the file value specifies the location of the image file. Note, however, that you don't have to surround the value with quotation marks. Following is an example.

BEWARE

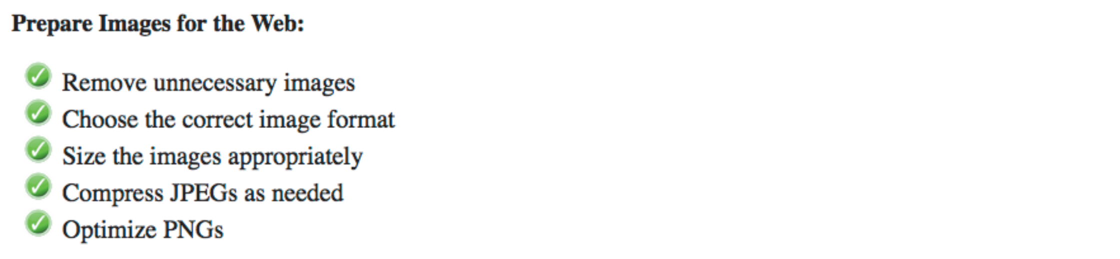
I've shown the code for turning an image into a link on one line for a purpose. If you place these tags on separate lines—particularly the closing `</a>` *tag—you end up with weird artifacts in the text (essentially, underlined carriage returns).*

▶ *Example*

⇨ **Online:** wdpg.io/6-1-7

This example shows you how to use the `list-style-image` *property to specify an image as a custom bullet.*

WEB PAGE



Prepare Images for the Web:

- ✓ Remove unnecessary images
- ✓ Choose the correct image format
- ✓ Size the images appropriately
- ✓ Compress JPEGs as needed
- ✓ Optimize PNGs

CSS

```
ul {
    list-style-image: url(/images/checkmark.png);
}
```

Set the list-style-image property to the image file location.

HTML

```
<h3>
    Prepare Images for the Web:
</h3>
<ul>
    <li>Remove unnecessary images</li>
    <li>Choose the correct image format</li>
    <li>Size the images appropriately</li>
    <li>Compress JPEGs as needed</li>
    <li>Optimize PNGs</li>
</ul>
```

Aligning Images and Text

The `<img>` tag is an inline element, so you can insert it into, say, a paragraph or similar block element, and it will flow along with the rest of the content. By default, the bottom edge of the image aligns with the baseline of the current line, but you can control that vertical alignment by using the `vertical-align` property:

```
element {
    vertical-align: baseline | bottom | middle | top;
}
```

- `baseline`—The bottom of the image is aligned with the baseline of the current line (the default).

- `bottom`—The bottom of the image is aligned with the bottom of the current line (that is, the bottommost extent of descending letters such as y and g).
- `middle`—The middle of the image is aligned with the baseline of the current line, plus one half of the x-height of the current font.
- `top`—The top of the image is aligned with the top of the current line.

The following example shows the `vertical-align` property at work.

MASTER

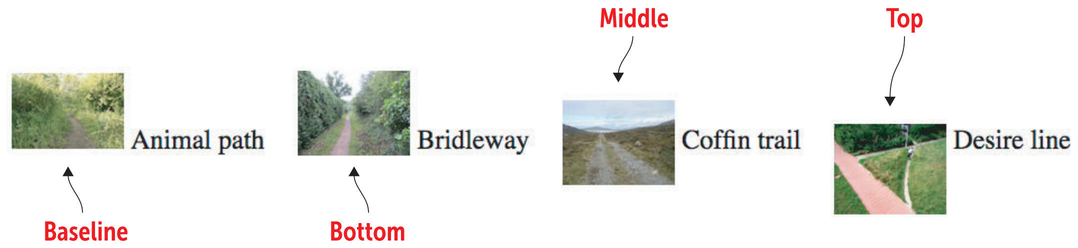
If you need even finer control of the vertical placement of an image, you can specify a length value, in pixels (`px`), for the `vertical-align` property. To move the image up, specify a negative value.

▶ Example

Online: wdpg.io/6-1-10

This example shows you how to use the `vertical-align` property to align an image vertically with surrounding content.

WEB PAGE



HTML

```
<div>
   <img src="/images/animalpath.jpg"
        style="vertical-align: baseline"> Animal path
</div>
<div>
   <img src="/images/bridleway.jpg"
        style="vertical-align: bottom"> Bridleway
</div>
<div>
   <img src="/images/coffintrail.jpg"
       style="vertical-align: middle"> Coffin trail
</div>
<div>
   <img src="/images/desireline.jpg"
       style="vertical-align: top"> Desire line
</div>
```

Applies the baseline alignment

Applies the bottom alignment

Applies the middle alignment

Applies the top alignment

Lesson 6.2: *Working with Background Images*

Covers: `background-image` and related properties

Online: wdpg.io/6-2-0

To add some visual interest to an element, you can use the `background-image` property to specify an image file to use as the background:

```
element {
    background-image: url(file);
}
```

The *`file`* value specifies where the image file is located. If the file is on a remote server, use the full URL of the file; if the file is in the same directory as the HTML file, use the name of the file; otherwise, use the image's path and filename.

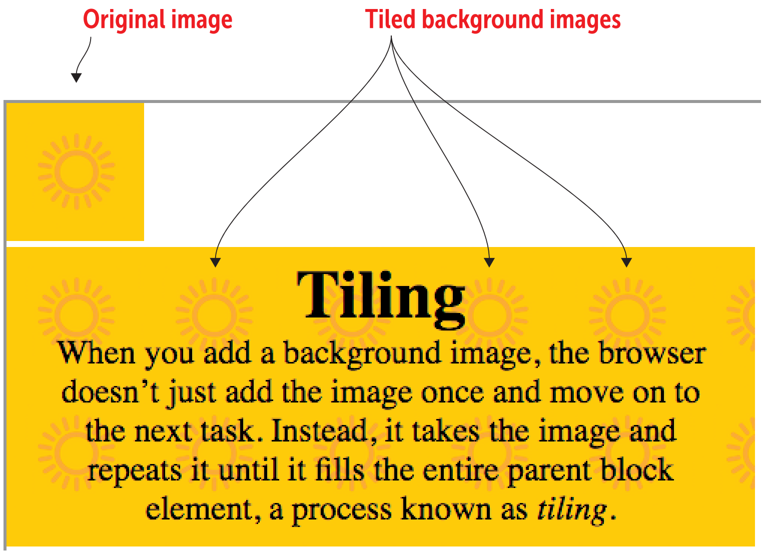
The following example shows this property in action.

▶ *Example*

Online: wdpg.io/6-2-1

This example shows you how to use the `background-image` *property to apply an image as the background of an element.*

WEB PAGE



Tiling

When you add a background image, the browser doesn't just add the image once and move on to the next task. Instead, it takes the image and repeats it until it fills the entire parent block element, a process known as *tiling*.

CSS

```
div {
    background-image: url(/images/bg.png);
    width: 500px;
    height: 200px;
}
```

The div element gets a background image.

HTML

```
<img src="/images/bg.png">
<div>
   <h1>Tiling</h1>
   <p>When you add a background image, the browser doesn't just
add the image once and move on to the next task. Instead, it takes
the image and repeats it until it fills the entire parent block
element, a process known as <i>tiling</i>.
   </p>
</div>
```

The image is displayed on its own for comparison.

When working with background images, you should assume that the image may not load properly for some reason. Therefore, it's always a good idea to specify the `background-color` property with a value that matches the main color of the image. Here's an example:

```
div {
    background-color: #fec72f;
    background-image: url(/images/bg.png);
}
```

Controlling the Background Repeat

You saw in the preceding example that the browser's default behavior for a background image that's smaller than the element is to repeat the image horizontally and vertically until the element is filled. This behavior is called *tiling* the background, and it's usually the behavior you want. However, you can control whether the background repeats horizontally, vertically, or doesn't repeat by using the `background-repeat` property:

```
element {
    background-image: url(file);
    background-repeat: repeat | repeat-x | repeat-y | no-repeat;
}
```

- `repeat`—Tiles the image horizontally and vertically (the default)
- `repeat-x`—Tiles the image only horizontally, as shown in Figure 6.2
- `repeat-y`—Tiles the image only vertically, as shown in Figure 6.3
- `no-repeat`—Displays the image once

BEWARE

A background image can add a nice bit of eye candy to a page, but it leaves a bitter taste if it interferes with the legibility of your page text. Always ensure that you've got lots of contrast between the text and the background.

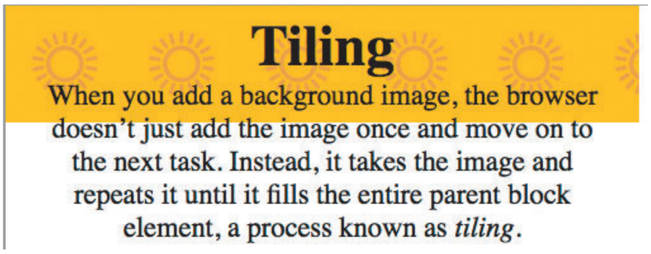
PLAY

You can try out all the `background-repeat` *values interactively in the Web Design Playground.* ⇨ **Online:** wdpg.io/6-2-2

REMEMBER

The `repeat` *value is the default, so declaring* `background-repeat: repeat` *is optional.*

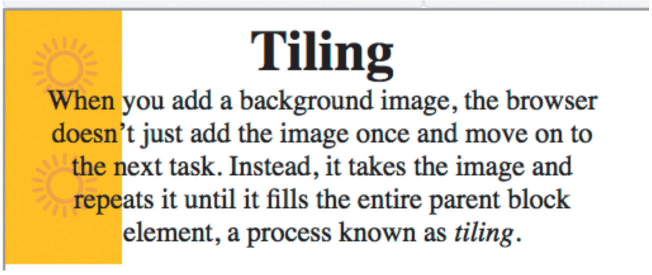
▶ Figure 6.2
With background-repeat: repeat-x, the background image repeats horizontally.



Tiling

When you add a background image, the browser doesn't just add the image once and move on to the next task. Instead, it takes the image and repeats it until it fills the entire parent block element, a process known as *tiling*.

▶ Figure 6.3
With background-repeat: repeat-y, the background image repeats vertically.



Tiling

When you add a background image, the browser doesn't just add the image once and move on to the next task. Instead, it takes the image and repeats it until it fills the entire parent block element, a process known as *tiling*.

Setting the Background Position

By default, the background image tiling starts in the top-left corner of the parent element. You can change that setting by applying the background-position property:

```
element {
    background-image: url(file);
    background-position: horizontal vertical;
}
```

- horizontal—Specifies the starting horizontal position of the background image tiling. You can use the keywords left, center, or right; a percentage; or a pixel value.
- vertical—Specifies the starting vertical position of the background image tiling. You can use the keywords top, center, or bottom; a percentage; or a pixel value.

Figure 6.4 is a composite that shows the nine possible positions when you use the three horizontal keywords (left, center, and right) and three vertical keywords (top, center, and bottom). Note that in each case, I set the background-repeat property to no-repeat.

PLAY

You can try out all the background-position *keywords interactively in the Web Design Playground.*
⇨ Online: wdpg.io/6-2-3

REMEMBER

The left top *value is the default, so declaring* background-position: left top *is optional. Note, too, that this value is equivalent to* background-position: 0px 0px *or* background-position: 0% 0%.

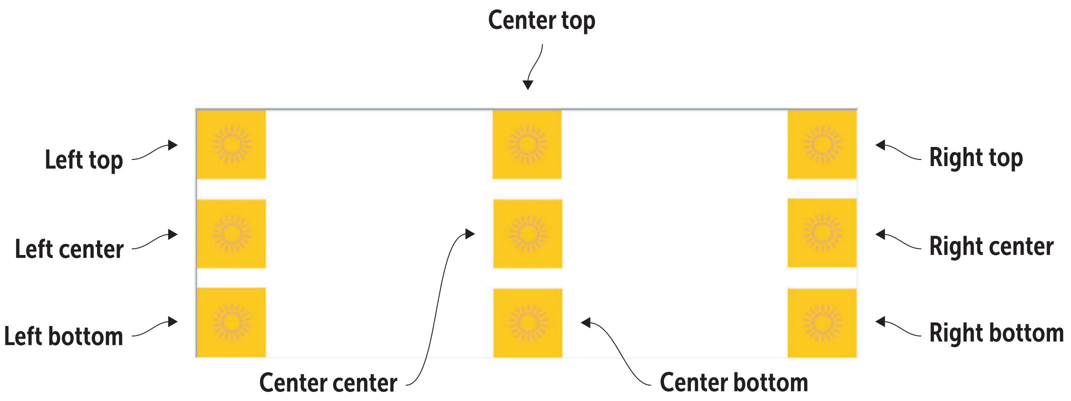


Figure 6.4 The nine possible keyword-based positions for the `background-position` property

Adding a Hero Image

One of the most popular web design trends of the past few years is the *hero image*: an eye-catching photo or illustration that takes up the entire width (and usually the entire height) of the browser window when you first land on a page. Using a hero image is a great way to grab a visitor's attention right off the bat.

To set up a hero image, you need to do the following:

1 Begin the page with a block element (such as a `div`) that's styled to take up the entire browser window:

```
width: 100vw;
height: 100vh;
```

2 For that same block element, add a background image and set its position to `background-position: center center`.

3 Add the declaration `background-size: cover`, which tells the browser to size the image so that it covers the entire background of the block element.

Following is an example.

PLAY

Another way to use an image as a custom bullet is to set the image as the background for the `li` element, which enables you to use `background-position` to control the alignment of the bullet and the item text.

⇨ **Online:** wdpg.io/6-2-4

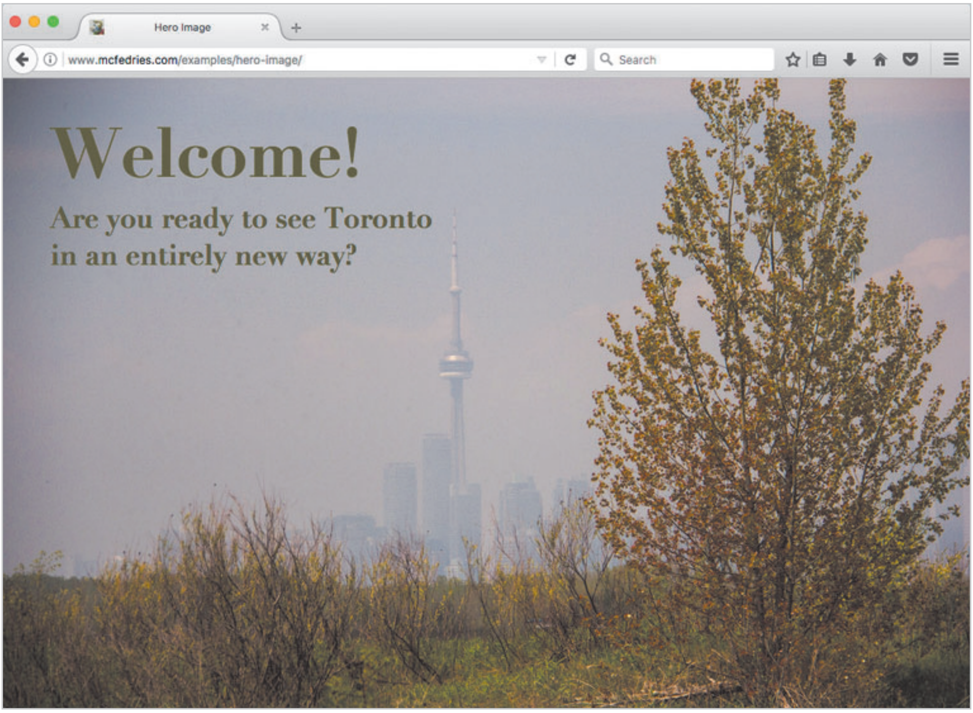
REMEMBER

The `vw` and `vh` units represent one one-hundredth of the browser window's width and height, respectively. For more on these units, see Chapter 7.

Example Online: wdpg.io/6-2-5

This example shows you how to add a hero image to a page.

WEB PAGE



Welcome!

Are you ready to see Toronto in an entirely new way?

CSS

```
.hero-image {
    width: 100vw;
    height: 100vh;
    background-image: url(/images/toronto.jpg);
    background-position: center center;
    background-size: cover;
}
```

Element sized to the browser window

Image is displayed and centered.

Image covers the entire element.

HTML

```
<div class="hero-image">
    <h1>Welcome!</h1>
    <h2>Are you ready to see Toronto in an entirely new way?</h2>
</div>
<div class="main">
    The main page content goes here.
</div>
```

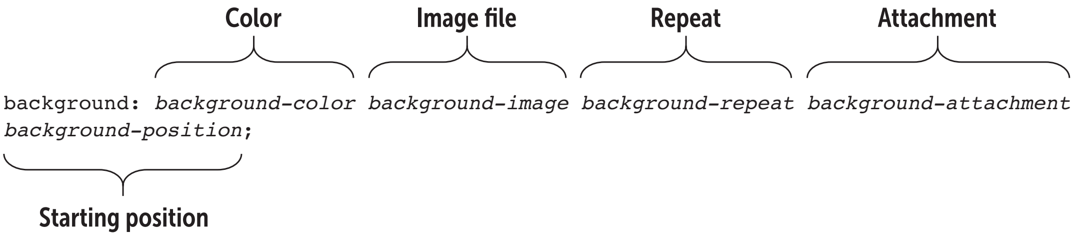
Block element for the hero image

The Background Shorthand Property

CSS has five main background-related components: color, image, repeat, attachment, and position. These components are represented, respectively, by the CSS properties `background-color`, `background-image`, `background-repeat`, `background-attachment`, and `background-position`. Handily, you can apply any or all of these properties with a single statement by using the `background` shorthand property, which takes the syntax shown in Figure 6.5.

PLAY

You can make the background stay in place while you scroll the rest of the page by adding the declaration `background-attachment: fixed`*.* ⇨ **Online:** wdpg.io/6-2-6



```
background: background-color background-image background-repeat background-attachment
background-position;
```

▶ **Figure 6.5** You can apply up to five background properties at the same time by using the `background` property.

This syntax is a straightforward repetition of everything you've learned so far, and you're free to enter the properties in any order you prefer.

PLAY

One of the most surprising aspects of background images is that you can use multiple backgrounds on the same element. You can repeat the same background image in two or more places or use two or more background images (or both!). ⇨ **Online:** wdpg.io/6-2-7

Optimizing Images

There's a common saying in web-design circles: "Color is free on the web." This saying means that you can add colors to text, backgrounds, borders, and other elements without paying a performance price. This is decidedly *not* the case with images, which, thanks to their potential to be huge (particularly those hero images I talked about earlier), can come with high performance costs indeed. To help ensure that your pages aren't bandwidth hogs that take ages to load, here are a few tips to bear in mind for optimizing the images you use:

- *Don't use unnecessary images.* Before adding an image to a page, ask yourself whether the image is needed to convey your message. If so, go for it. If not, leave it behind. Your users will thank you.
- *Watch your image sizes.* Web browsers can resize images as needed, but they shouldn't have to. If you want a 100x100 logo in the top-left corner, don't upload a 2,048x2,048 version of that image and force the browser to resize (by, say, specifying the smaller width and height in your CSS). That bigger file will take a long time to download, which is a waste of bandwidth.
- *Choose your file format wisely.* As a general rule, you should use the image file format that produces the smallest file size while still retaining a satisfactory level of image quality for the job at hand. A hero image should look good, but a tiny thumbnail doesn't have to be high-resolution.

BEWARE

If you plan to overlay text on your hero image, make sure that the image includes an area that's not too busy so that your text will be readable. Also, make sure that you have sufficient contrast between the colors of your image and your text.

- *Take advantage of JPEG compression.* If you're saving your image in the JPEG format, your imaging software allows you to choose a compression level for the file. You'll need to experiment a bit to get the right level, but for most uses, a compression level in the range of 60 to 75 percent is a good place to start. More compression usually leads to poor image quality, and less compression usually results in large file sizes.
- *Optimize PNG images.* When you're working with a PNG image, decide whether you can get away with 8-bit color, which is a mere 256 colors. For a simple logo or icon, 8-bit color may be more than enough, and you'll end up with quite a small file. For more complex images, you'll probably need the full 24-bit palette.

LEARN

If you need to use 24-bit PNGs, software tools are available that can help reduce the size of those files. If you use a Mac, try ImageAlpha (https://pngmini.com); if you run Windows, check out PNGoo (https://pngquant.org).

Adding Video and Audio to the Page

You know that people love their cat videos and podcasts, so you want a piece of the action by adding video or audio content to your own web pages. Great idea! I'll begin with the good news: HTML5 comes with the `<video>` and `<audio>` tags, which offer a somewhat straightforward way to embed media content in a page. Notice that I said *somewhat*. Why the hedge? Ah, that's where the bad news rears its complexifying head. Right now, web media is a crazy quilt of standards, compression algorithms, and file formats. It's borderline absurd, but if you want to serve your visitors sights or sounds, you need to wade into the deep end.

I'll begin by defining two aspects of web media formats:

- *Container*—The file format, called a *container* because it acts like the media equivalent of a zip file—that is, it's an archive that contains multiple items, particularly the media codecs (discussed next) and the media metadata.
- *Codec*—The algorithm used to encode and compress the video or audio in a digital format and to decode and decompress the media for playback. (The word *codec* is a blend of *code/decode* and *compress/decompress*.)

So a web media file that you'd embed in a page comes in a specific media format that uses a particular container, and within that container are all the codecs that the format supports. Sounds simple enough, right? The absurdity comes into play when you understand that there's no such thing as a standard or universal media format.

Web Video Formats

For video, in fact, you have three main formats to worry about:

- *WebM*—This format uses the WebM container, inside which is either the VP8 or VP9 video codec, as well as the Vorbis or Opus audio codec. This format is open source and royalty free. File extension: `.webm`.
- *Ogg*—This format uses the Ogg container, inside which is the Theora video codec, as well as the Vorbis or Opus audio codec. This format is open source and royalty free. File extension: `.ogg` or `.ogv`.
- *MPEG-4*—This format uses the MPEG-4 container, inside which is the H.264 video codec, as well as the AAC audio codec. This format is patented but free for end users. File extension: `.mp4`.

Which one should you use? Most of the time, you can get away with using the MPEG-4 format, which is supported by all major browsers. That support is a bit problematic, however. First, Firefox doesn't support MPEG-4 natively; instead, it relies on the operating system's built-in support for MPEG-4. Second, Google has hinted that it may not support MPEG-4 in future releases of Chrome. It's a good idea to serve your visitors both an MPEG-4 version and a WebM version (which is newer and better supported than Ogg).

LEARN

*Many tools are available to convert videos to formats supported by HTML5. Two online tools that are worth checking out are Zamzar (*https://www.zamzar.com*) and Online-Convert (*https://www.online-convert.com/*).*

Web Audio Formats

For audio, there are even more formats:

- *MP3*—This format is both the container and the audio codec. This format is patented but free for end users. File extension: `.mp3`.
- *WAV*—This format is both the container and the audio codec. File extension: `.wav`.
- *WebM*—This format uses the WebM container, inside which is Vorbis or Opus audio codec. This format is open source and royalty free. File extension: `.webm`.
- *Ogg*—This format uses the Ogg container, inside which is the Vorbis or Opus audio codec. This format is open source and royalty free. File extension: `.ogg`. or `.oga`.
- *MPEG-4*—This format uses the MPEG-4 container, inside which is the AAC audio codec. This format is patented but free for end users. File extension: `.m4a`.

Things are a bit saner in the audio world, where every browser now supports the MP3 format, so you can get away with using the one file type.

LEARN

*The two online tools I mentioned earlier also support the HTML5 web audio formats. You may also want to have a look at media.io (*https://media.io*).*

Lesson 6.3: *Embedding Video in a Web Page*

Covers: The video element

Online: wdpg.io/6-3-0

HTML5's video element offers a no-nonsense way of embedding video content in your web page. Well, *no-nonsense* may be wishful thinking. You can use two syntaxes, depending on the number of video file formats you want to serve.

First, here's the syntax to use if you're offering a single video format:

```
<video src="file"
       poster="file"
       width="value"
       height="value"
       controls
       autoplay
       loop>
</video>
```

- src—Specifies the location of the video file, so it's much the same as the src attribute for the <img> tag
- poster—Specifies the location of an image, such as a title frame or still frame from the video, to display before video playback begins
- width and height—Specify the dimensions of the video playback window
- controls—When included, tells the browser to display the playback controls in the video window
- autoplay—When included, tells the browser to automatically start playing the video as soon as it has downloaded enough of the video file
- loop—When included, tells the browser to begin playback from the beginning each time the video ends

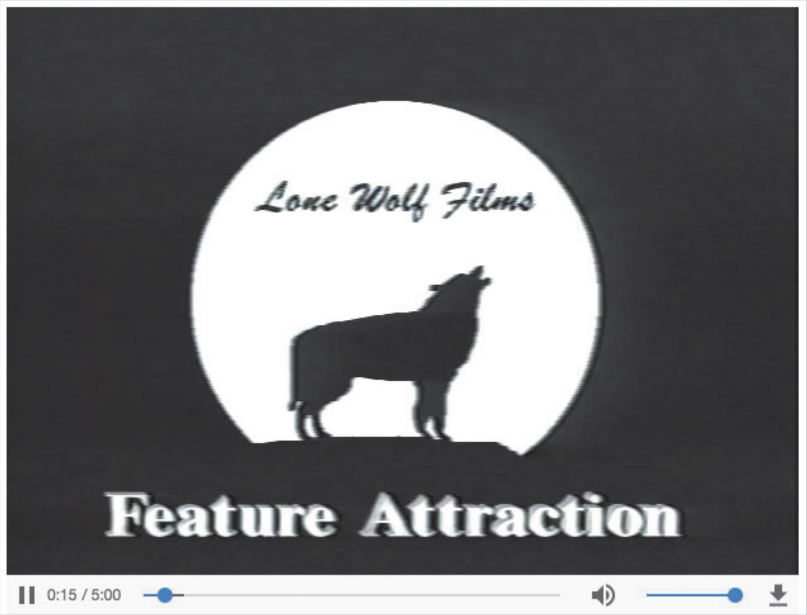
Following is an example.

▶ *Example*

Online: wdpg.io/6-3-1

This example shows you how to embed a single video-file format in a web page.

WEB PAGE



HTML

```
<video src="/videos/movie.mp4"
      poster="/images/movie-poster.jpg"
      width="625"
      height="480"
      controls
      autoplay
      loop>
</video>
```

To offer two or more video formats, you need to remove the src attribute from the <video> tag and replace it with multiple source elements, one for each format you want to offer:

```
<video poster="file"
       width="value"
       height="value"
       controls
       autoplay
       loop>
    <source src="file"
            type='type; codecs="codecs"'>
</video>
```

- src—As before, the src attribute for each <source> tag specifies the name and/or location of the video file.

- `type`—This string (surrounded by single quotation marks) specifies the video format type (as shown earlier in this chapter in the "Web Video Formats" section), a comma-separated and double-quotation-mark-surrounded list of the format's video and audio codecs:
- *MPEG-4*—Use the following:

```
type='video/mp4; codecs="avc1.4D401E, mp4a.40.2"'
```

- *WebM*—Use one of the following:

```
type='video/webm; codecs="vp8, vorbis"'
type='video/webm; codecs="vp9, vorbis"'
type='video/webm; codecs="vp9, opus"'
```

- *Ogg*—Use one of the following:

```
type='video/ogg; codecs="theora, vorbis"'
type='video/ogg; codecs="theora, opus"'
```

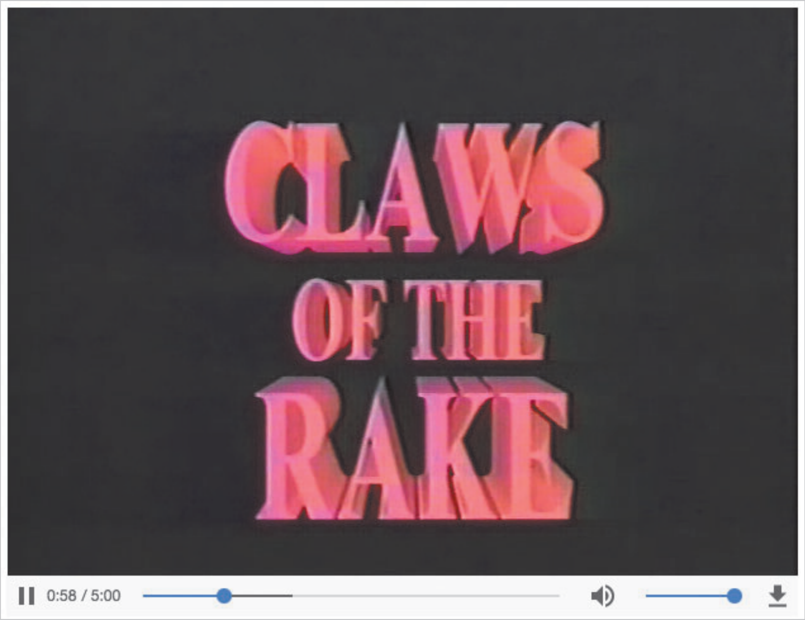
Here's an example.

▶ ***Example***

⇨ **Online:** wdpg.io/6-3-2

This example shows you how to embed multiple video-file formats in a web page.

WEB PAGE



HTML

```
<video poster="/images/movie-poster.jpg"
      width="625"
      height="480"
      controls
      autoplay
      loop>
   <source src="/videos/movie.mp4"
          type='video/mp4; codecs="avc1.4D401E, mp4a.40.2"'>
   <source src="/videos/movie.webm"
          type='video/webm; codecs="vp8, vorbis"'>
   <source src="/videos/movie.ogv"
          type='video/ogg; codecs="theora, vorbis"'>
</video>
```

Lesson 6.4: *Embedding Audio in a Web Page*

Covers: The audio element

Online: wdpg.io/6-4-0

You'll be delighted to hear that embedding audio in a web page is nearly identical to embedding video, because the `<audio>` and `<video>` tags have many of the same attributes.

First, here's the syntax to use if you're offering a single audio format:

```
<audio src="file"
       controls
       autoplay
       loop>
</video>
```

- `src`—Specifies the location of the audio file
- `controls`—When included, tells the browser to display the playback controls in the audio window
- `autoplay`—When included, tells the browser to automatically start playing the audio as soon as it has downloaded enough of the audio file
- `loop`—When included, tells the browser to begin playback from the beginning each time the audio ends

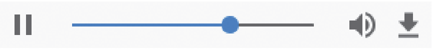
Following is an example.

Example

Online: wdpg.io/6-4-1

This example shows you how to embed a single audio-file format in a web page.

WEB PAGE



HTML

```
<audio src="/audio/music.mp3"
      controls
      autoplay
      loop>
</audio>
```

To offer two or more audio formats, remove the `src` attribute from the `<audio>` tag and replace it with multiple `<source>` tags, one for each format you want to offer:

```
<audio controls
       autoplay
       loop>
    <source src="file"
            type="type">
</audio>
```

- `src`—As before, the `src` attribute for each `<source>` tag specifies the name and/or location of the audio file
- `type`—Specifies the audio format type (as shown earlier in the section “Web Audio Formats”)

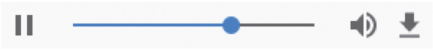
Here's an example.

▶ *Example*

⇨ **Online:** wdpg.io/6-4-2

This example shows you how to embed multiple audio-file formats in a web page.

WEB PAGE



HTML

```
<audio controls
      autoplay
      loop>
   <source src="/audio/music.mp3"
          type="audio/mp3">
   <source src="/audio/music.wav"
          type="audio/wav">
   <source src="/audio/music.webm"
          type="audio/webm">
   <source src="/audio/music.ogg"
          type="audio/ogg">
   <source src="/audio/music.m4a"
          type="audio/mp4">
</audio>
```

Summary

- There are four main image-format types—GIF, JPEG, PNG, and SVG—although most of your pages will use JPEG for photos and complex images, and PNG for illustrations, logos, and icons that use mostly solid colors and/or transparency.
- To add an image to the page, use the `<img>` tag:

  ```
  <img src="file" alt="description" title="tooltip">
  ```

- To infuse your images with HTML5 semantic flavor, surround the `img` element with the `figure` element and, optionally, add a `figcaption` element.
- To make an image do double duty as a link, surround the `img` element with the `a` element.

Adding Images and Other Media

- You can set up an image as an element background by adding the following property to the element's CSS:

  ```
  background-image: url(file);
  ```

- You can control the background image's display by adding one or more of the following properties: `background-repeat`, `background-position`, and `background-attachment`.
- You can set all three of these properties, as well as the `background-color` and `background-image` properties, by using the `background` shorthand property.
- You embed a video in a web page by using the `<video>` tag, and you embed sound in a web page by using the `<audio>` tag.
- With both the `<video>` tag and the `<audio>` tag, you can specify multiple formats by adding a separate `<source>` tag for each.