Images and Objects

In addition to text content, web pages may include a wide range of multimedia objects, including images, image maps, Java applets, video, Flash movies, even other HTML documents. This chapter focuses on the (X)HTML elements defined for adding images and media objects, including:

`img`	Adds an image
`map`	The map used for an image map
`area`	A geometric region in an image map
`object`	A generic media object
`param`	Specifies values for an object necessary at runtime
`embed`	Embeds media requiring plug-ins *(nonstandard)*
`noembed`	Content displayed if embedded media is not supported *(nonstandard)*
`applet`	Adds an applet *(deprecated)*
`iframe`	A floating frame that displays an external HTML document

Inline Images

Inline images are images that occur in the normal flow of the document's content. As inline elements, they affect the visual display of other elements in the flow, unlike background images, which render behind elements. Images are added to the document with the `img` element. Images are considered to be replaced elements because the actual content resides in external files rather than in the source document.

The HTML 4.01 Recommendation allows images to be added using the generic `object` element, as demonstrated later in this chapter. Because the `object` method is not universally supported, the `img` element is still the primary element used to place images in web documents.

img

```
<img />
```

Attributes

Core (id, class, style, title), *Internationalization, Events*
alt="*text*" *(Required)*
ismap
height="*number*"
longdesc="*URL*"
lowsrc="*URL*"
name="*text*"
src="*URL*" *(Required)*
usemap="*URL*"
width="*number*"

Deprecated attributes

align="absbottom|absmiddle|baseline|bottom|center|left|middle|
right|top"
border="*number*"
hspace="*number*"
vspace="*number*"

Image Formats and Usage

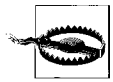
Web images must be in one of the three web-compatible formats: GIF, JPEG, or PNG. Furthermore, the files should be named with the proper suffixes—*.gif*, *.jpeg* or *.jpg*, and *.png*, respectively—so that your web server sends the proper Content-Type—image/gif, image/jpeg, and image/png, respectively—which the browser uses to recognize the image format. These graphic file formats, as well as other requirements for putting images online, are discussed in detail in Part V.

Inline images are used in a variety of ways:

As a simple image
: An image can be used on a web page much as it is used in print—as a static image that adds information, such as a company logo or an illustration.

As a link
: An image can also be used to link to another document as an alternative to text links.

As an image map
: An image map is a single image with multiple "hotspots" that link to other documents. There is nothing special about the image itself; it is an ordinary inline image. Special HTML markup and map files link pointer coordinates with their respective URLs. The upcoming "Image Maps" section of this chapter includes a full explanation of how image maps work and how to create them.



Images (transparent GIFs, in particular) have also been used as spacing devices, but now that we have better control of space and alignment with Cascading Style Sheets, this use of spacer images is outdated and must be avoided in contemporary web design.

With the emergence of standards-driven web design in recent years, there has been a shift away from using inline images for purely decorative purposes. Images that are not part of the content and only contribute to the presentation of the page are commonly placed as background images using CSS instead. Images may be applied to the background of any element (not just `body`) using the `background-image` or shortcut `background` style properties and they *don't* need to tile. Chapter 24 includes examples of several CSS image replacement techniques.

There are several benefits to specifying decorative images only in an external style sheet and keeping them out of the document structure. Not only does it make the document cleaner and more accessible, it also makes it easier to make changes to the look and feel of a site than when presentational elements are interspersed in the content. See Chapter 20 for more details on CSS background images. For inspiration on how visually rich a web page can be with no `img` elements at all, see the CSS Zen Garden site at *www.csszengarden.com*.

Without further ado, it is time to look at an example of `img` element markup.

img Element Syntax

There are over a dozen attributes that can be applied to the `img` element to affect its display, but the only required attributes are `src`, which provides the URL of the image file, and `alt` for providing text for browsers that cannot (or have been asked not to) display images. The syntax for a minimal image element looks like this:

```
<img src="url of image" alt="alternative text" />
```

The URL provided by the `src` attribute may be absolute (including the protocol and domain name) or relative to the current document (using a relative pathname). The conventions for relative pathnames are described in detail in Chapter 4.

Figure 12-1 shows an inline image resulting from this markup.

```
<p>Star light <img src="star.gif" alt="star" /> Star bright.</p>
```

Figure 12-1. An image placed within a line of text

Default presentation

As the example makes clear, because `img` is an inline element, it does not introduce any line breaks or extra space. By default, the bottom of an image aligns with

the baseline of surrounding text. The alignment and position of the image may be changed with style sheet rules as discussed in Chapters 18 and 21. There are also a number of deprecated attributes for controlling the presentation of images that are briefly introduced later in this chapter.

Alternative text

There is no guarantee that an image will be displayed. It may be corrupted or not found, or users may be using a text-only or speech browser that doesn't support images. When an image is not displayed, graphical browsers display a generic broken image icon in its place. Non-graphical browsers generally just write out "[image]." Either of these instances can be a dead end for users and make certain content inaccessible.

The `alt` attribute allows you to specify a string of alternative text to be displayed in place of the image when the image is unavailable. It is also what non-graphical browsers write in place of images. Figure 12-2 shows one possible rendering for this markup if the image file should fail to load.

```
<p>First star <img src="star2.gif" alt="star illustration" /> I see tonight.
</p>
```

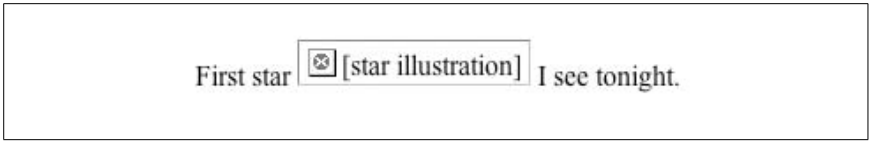


Figure 12-2. Alternative text may be displayed when an image is unavailable

Firefox displays the alternative text as though it were in the text flow. The Safari browser does not display alternative text for missing graphics. Some browsers display alternative text as a pop-up "tool tip" when the mouse rests on the image area, but such behavior is nonstandard is not something to depend on.

The HTML 4.01 specification declared `alt` to be a required attribute within the `img` element (although an image will still display without it). Taking the extra time to provide alternative text for your images is the simplest way to make your page accessible to the greatest number of readers.

The W3C recommends that alternative text be provided only when the image contains content relevant to the document, not when the image is purely decorative. For example, the alternative text "red line" is not useful and only slows down document processing and may be frustrating for users using spoken browsing devices. An `alt` attribute with an empty string (`alt=""`) is recommended instead.

Specifying Width and Height

Although `src` and `alt` are the only required attributes in the `img` element, `width` and `height` are often used because they speed up page display. The `width` and `height` attributes simply indicate the dimension of the image in pixels, such as:

```
<img src="star.gif" width="50" height="50" />
```

With this information, the browser can lay out the page before the images download.

CSS `width` and `height` properties are preferred to the presentational attributes, and will also ensure that the page can be assembled before the images arrive.

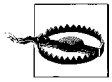
Without `width` and `height` values, the page may be redrawn several times, first without images in place, and again each time new images arrive. It is worthwhile to take the time to include accurate width and height information in every `img` element.

If the `width` and `height` values specified are different than the actual dimensions of the image, the browser resizes the image to match the specified dimensions. If you specify a percentage value for width and height, some later browsers resize the image to the desired proportions.

Although this effect may be convenient and prevent an extra trip to the image editor, in some browsers, it just results in a pixelated, poor quality image, as shown in Figure 12-3.

```
<img src="star.gif" width="50" height="50" />
<img src="star.gif" width="200" height="50" />
```

Figure 12-3. Resizing an image with width and height attributes



Reducing the dimensions of an image with markup is a bad practice and is strongly discouraged. In addition to resulting in poor image quality, it forces an unnecessarily large file download on the user when a much smaller file would do. Changing the image dimensions for the final presentation does not reduce the file size.

Deprecated img Attributes

There are a number of attributes in the Transitional DTDs that have been deprecated because they control presentational aspects of the image. As with most deprecated attributes, they have been replaced with more versatile style sheet properties. This section provides an introduction to these attributes and suggests style sheet alternatives.

If you are authoring using the XHTML Strict or XHTML 1.1 DTDs, using any of these attributes will cause your document to be invalid. Use the style sheet methods listed instead.

Vertical alignment

By default, the bottom of an image aligns with the baseline of the surrounding text (see Figure 12-1), but there are ways to change the vertical alignment. The preferred method is to use style sheets. Using HTML markup alone, the HTML 4.01 Recommendation includes the deprecated `align` attribute with the values `top`, `middle`, and `bottom` for vertical alignment. The HTML 3.2 Recommendation also included the values `texttop`, `absmiddle`, `baseline`, and `absbottom`, but they were dropped from future specifications and are only partially supported by modern browsers. Figure 12-4 shows the effects of these alignment values.

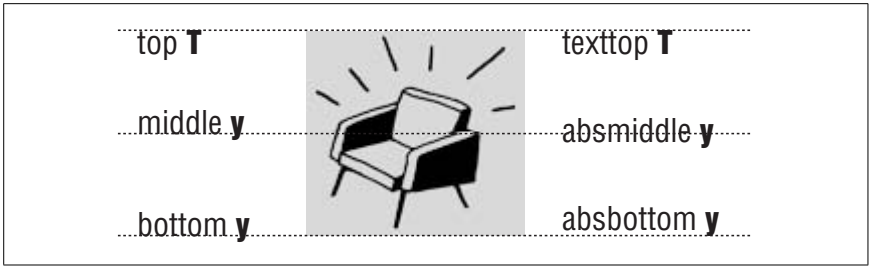


Figure 12-4. Vertical alignment values

The preferred CSS method for specifying vertical alignment is via the `vertical-align` property, which may be used to change the alignment of an image relative to the baseline or the height of the line it occupies. The accepted values are `top`, `text-top`, `bottom`, `text-bottom`, `middle`, `sub`, `super`, and `baseline` (the default), as well as specific length or percentage values. See Chapter 18 for explanations of vertical alignment with CSS.

Horizontal alignment

The `align` attribute is also used to align an image on the left or right margin of the page by using the values `left` or `right`, respectively. What makes left and right alignment special is that, in addition to placing the image on a margin, it allows the text to wrap around it. This is called *floating* the image. Figure 12-5 shows how images are displayed when set to align to the left and right.

```
<p><img src="leaf.gif" alt="" align="left" />An Oak and a Reed were arguing
about their strength...</p>
<p><img src="leaf.gif" alt="" align="right" />An Oak and a Reed were arguing
about their strength...</p>
```

The CSS `float` property is the preferred method for positioning images (or any element) against the right or left edges of the containing block and allowing the following content to wrap around it. Chapter 21 discusses floating elements.

Adding space around aligned images

When text flows around an image, browsers allow it to bump up against the image's edge. Usually, it is preferable to have a little space between the image and the surrounding text. In HTML, you provide this space by using the `vspace` and `hspace` attributes within the `img` element.

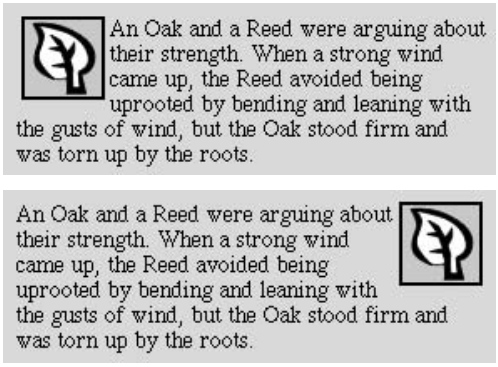


Figure 12-5. Text wraps around floated images

Right Alignment Without Text Wrap

Using the `align="right"` attribute to place an image against the right margin automatically results in text wrapping around the image. If you want to move an image to the right without the wrap, put the image in a paragraph (`p`), and then align the paragraph to the right, as shown:

```
<p align="right"><img src="leaf.gif" /></p>
<p>An Oak and a Reed were arguing...</p>
```

In CSS, to align an element with no text wrap, apply the `text-align` property to a block-level element that contains the image.

The `vspace` (vertical space) attribute holds a specified number of pixels of space above and below an aligned image. Space to the left and the right is added with `hspace` (horizontal space). Note that space is always added symmetrically (both top and bottom, or on both sides), and it is not possible with these attributes to specify an amount of space along a particular side of the image (you can, however, do this with style sheets). Figure 12-6 shows an image aligned with the `hspace` attribute set to 12.

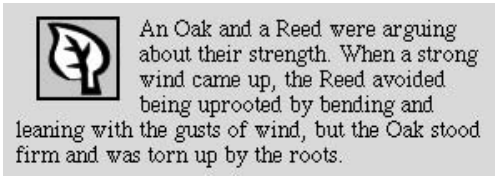


Figure 12-6. Image alignment with horizontal spacing

The preferred CSS method for adding space around the sides of the image is to simply apply an amount of margin around it. The various `margin` properties allow you to apply different amounts of space to each side of the floated image. CSS margins are discussed in Chapter 19.

Stopping text wrap

Text automatically wraps to fill the space along the side of an aligned image (or other inline object). To stop the text from wrapping and start the next line against the margin (instead of against the image), insert a line break (`br`) with the `clear` attribute.

The `clear` attribute gives the browser directions on where to place the new line. It has three possible values: `left`, `right`, and `all`. If an image is aligned right, insert `<br clear="right" />` to begin the text below the image against the right margin. For left-aligned images, use `<br clear="left" />`. The `<br clear="all" />` element starts the text below the image on both margins, so it may be the only value you'll ever need. Figure 12-7 shows the result of this markup.

```
<p><img src="leaf.gif" alt="leaf illustration" align="left" hspace="12" />An
Oak and a Reed were arguing about strength. <br clear="all" />When a strong
wind came up,...
```

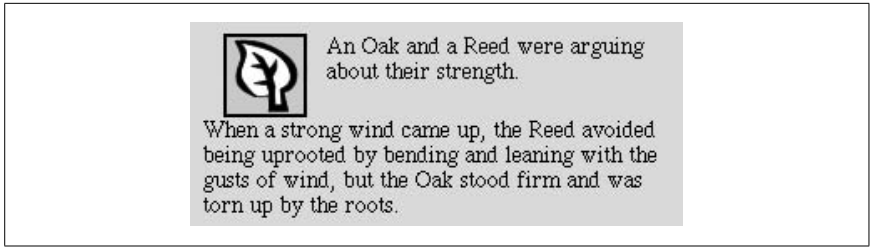


Figure 12-7. The clear attribute starts the next line below an aligned image

The preferred CSS method for preventing the following element from starting next to the floated image is to apply the `clear` property to the following element and specify the side (`left`, `right`, or `both`) that you want to start below any floated objects. Clearing is discussed in Chapter 21.

Borders

By default, when an image is linked, most browsers display a two-pixel-wide border around the image in the same color as the text links on the page (bright blue by default). In most cases, this blue border is visually unappealing, particularly around an image with transparent edges, but it is quite simple to turn it off using the `border` attribute.

The `border` attribute specifies the width of the border in number of pixels. Specifying a value of zero turns the borders off:

```
<a href="document.html"><img src="picture.gif" border="0" /></a>
```