THE ULTIMATE

HTML

REFERENCE

Ian Lloyd

sitepoint®

The Ultimate HTML Reference (Chapter 7)

Thank you for downloading this sample chapter from *The Ultimate HTML Reference*, by Ian Lloyd.

This excerpt encapsulates the Summary of Contents, Information about the Author and SitePoint, Table of Contents, and Chapter 7: *Image and Media Elements.*

We hope you find this information useful in evaluating the book.

For more information, visit sitepoint.com

Summary of Contents of this Excerpt

Summary of Additional Book Contents

THE ULTIMATE HTML REFERENCE

BY **IAN LLOYD**

The Ultimate HTML Reference

by Ian Lloyd

Notice of Rights

Notice of Liability

Trademark Notice

About the Author

Ian Lloyd runs Accessify.com, a web accessibility site that he started in 2002, and has written or co-written a number of books on the topic of web standards and development, including SitePoint's best-selling beginners' title, *Build Your Own Web Site The Right Way using HTML & CSS.* Ian was previously a member of the Web Standards Project and is a regular speaker at web development conferences, including the highly regarded South By Southwest (SXSW) and @media events. He lives in Swindon, UK, with wife Manda and lively terrier Fraggle, and has a bit of a thing for classic Volkswagen camper vans.

About the Expert Reviewers

Lachlan Hunt (http://lachy.id.au/) worked as a front-end web developer, primarily developing with HTML, CSS, and JavaScript, for four years before joining Opera Software in late 2007.

As a developer and advocate of web standards, he has participated in the WHATWG (http://www.whatwg.org/) and various W3C working groups, including Web API, Web Application Formats, and HTML Working Groups, where he actively contributes to the work on HTML5.

Tommy Olsson is a pragmatic evangelist for web standards and accessibility who lives in the outback of central Sweden. Visit his blog at http://www.autisticcuckoo.net/.

About the Technical Editor

Toby Somerville is a serial webologist who caught the programming bug back in 2000. For his sins, he has been a pilot, a blacksmith, a web applications architect, and a freelance web developer. In his spare time he likes to kite buggy and climb stuff.

About the Technical Director

As Technical Director for SitePoint, **Kevin Yank** oversees all of its technical publications—books, articles, newsletters, and blogs. He has written over 50 articles for SitePoint, but is best known for his book, *Build Your Own Database Driven*

Website Using PHP & MySQL. Kevin lives in Melbourne, Australia, and enjoys performing improvised comedy theater and flying light aircraft.

About SitePoint

SitePoint specializes in publishing fun, practical, and easy-to-understand content for web professionals. Visit http://www.sitepoint.com/ to access our books, newsletters, articles, and community forums.

The Online Reference

The online version of this reference is located at http://reference.sitepoint.com/html. The online version contains everything in this book, and is fully hyperlinked and searchable. The site also allows you to add your own notes to the content and to view those added by others. You can use these user-contributed notes to help us keep the reference up to date, to clarify ambiguities, and to correct any errors.

Your Feedback

If you wish to contact us, for whatever reason, please feel free to email us at books@sitepoint.com. We have a well-staffed email support system set up to track your inquiries. Suggestions for improvement are especially welcome.

Reprint Permissions

Do you want to license parts of this book for photocopying, email distribution, intranet or extranet posting, or for inclusion in a coursepack? Please go to Copyright.com and type in this book's name or ISBN to purchase a reproduction license.

Table of Contents

Chapter 9   Frame and Window Elements

Chapter

7

Image and Media Elements

Once upon a time, in the dim, distant past, the Web was little more than a bunch of boring academic documents that were exciting to the handful of scientists and physicists who were technically capable of putting such documents together and able to configure a cranky old dial-up modem. Then along came the `img` (p. 331) element, and the Web started its transformation into the form we know and love today.

Yes, that's a *slight* oversimplification of how we got from A to B, but in all seriousness, until the elements detailed in *this* section were included in the W3C specs, and then in web browsers (or sometimes the other way around), the Web didn't have much to offer in the way of eye candy.

The elements in this section provide many opportunities to liven up a web page, whether that be to add a few images (using the `img` element), perhaps add a Flash movie or some other multimedia item using the `object` (p. 355) or `embed` (p. 330) elements, or create interactive maps that users can navigate through using the `map` (p. 352) and `area` (p. 318) elements.

`<···>` applet

```
<applet archive="uri" code="uri" codebase="uri"
height=" { number | percentage } "width=" { number
| percentage } ">
</applet>
```

SPEC		
deprecated	empty	version
YES	NO	HTML 3.2
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
PARTIAL	PARTIAL	PARTIAL	PARTIAL

The `applet` element's purpose is to embed small Java applications (or applets—little apps) into the flow of a page. If any `param` (p. 376) elements are specified in the `applet` tag, they must be placed before all other content. This element has been replaced by the much more flexible, and non-Java-specific `object` (p. 355) element.

For more information, see http://reference.sitepoint.com/html/applet/.

Other Relevant Stuff

`<···>` object (p. 355)

specifies a generic, multipurpose container for a media object

`<···>` area

```
<area alt="string" coords="coordinates" href="uri"
shape=" { circle | default | poly | rect } "/>
```

SPEC		
deprecated	empty	version
NO	YES	HTML 3.2
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

This code identifies one `area` element within an image `map`:

```
<map name="bigthings" id="bigthings">
  <area shape="rect" coords="35,4,205,108"
      href="http://en.wikipedia.org/wiki/Australia's_Big_Things"
      alt="Australia's Big Things (on Wikipedia)"/>
    ⋮
</map>
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
    border="0" usemap="#bigthings"/></p>
```

We use the `area` element to define each hotspot that's clickable or actionable and will act as a link within a client-side image `map`. The behavior of a hotspot is similar to a normal link (the `a` (p. 144) element) in that it can be activated with a mouse click or the keyboard, and it's used to link to another web page or resource; it's also similar to the image element because, as it defines a specific area of a larger image, it requires us to set an `alt` (p. 321) attribute for those using non-visual browsers.

The hit, or actionable, area of the image is defined using a combination of the `shape` (p. 326) and `coords` (p. 323) attributes, which create the hit area boundaries as a sequence of *x,y* coordinate pairings.

Use This For …

This element is used to create clickable or actionable areas on an image. For example, you might use it to create a graphically rich navigation mechanism such as a map on which the different countries or regions are the clickable areas.

Note that not every `area` needs to be actionable. It's perfectly acceptable to use `area` *without* an `href` attribute for the purposes of outlining a given area, and for providing additional help in the form of a tooltip created using the `title` (p. 504) attribute. However, this technique is inaccessible to people who use anything other than a mouse to navigate the web page, since there will be no mouse hover event for those users—the usual trigger for the tooltip display.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

Other Relevant Stuff

`map` (p. 352)

defines a client-side image map

"..." accesskey *for <area>*

```
accesskey="key"
```

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

Here, the first `area` has an `accesskey` of "b" defined:

```
<map name="bigthings" id="bigthings">
  <area shape="rect" coords="35,4,205,108"
      href="http://en.wikipedia.org/wiki/Australia's_Big_Things"
      alt="Australia's Big Things (on Wikipedia)" accesskey="b"/>
    ⋮
</map>
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
    border="0" usemap="#bigthings"/></p>
```

The `accesskey` attribute allows the user to activate a control on a page using a keyboard shortcut. This may save time for users who would otherwise need to tab through a series of form controls or move the mouse to get to the desired link. The key combination that activates the link to which the `accesskey` is applied varies depending on the platform and browser combination. For IE/Windows, users press Alt + `accesskey`, while Firefox/Windows users press Alt + Shift + `accesskey`; users of most Mac browsers press Ctrl + `accesskey`; in Opera, pressing Shift + Esc displays a list of links for which `accesskey` attributes are defined, allowing users to choose the key they want to use.

Generally speaking, browsers don't indicate that an `accesskey` attribute is defined for a form control, and this lack of discoverability is a problem. The most common method for indicating the `accesskey` value is to place it in a `title` (p. 504) attribute of the element to which it's applied. However, for this approach to work, the user must mouse over the element in question. You may want to state the `accesskey` value in some other way—for example:

```
<p>Press <kbd>b</kbd> to go to Australia's Big Things (on
Wikipedia).</p>
```

Image and Media Elements

Value

This attribute takes as its value a single character, which can be numerical, alphabetical, or even a symbol.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

There's some variety in the way that the `accesskey` is activated, but generally speaking, it can work well. The downside of using this attribute is that keystrokes defined may clash with those of other technologies. For example, an assistive device such as a screen reader or magnifier may have designated for certain purposes the keystrokes that you've defined in the `accesskey` attribute. In addition, different language browsers use different "accelerator keys" for their own menu options, which may also clash with those you've defined. As a result of these clashes, the `accesskey` may not work as expected for all users. However, it may be very useful for controlled environments such as intranets, where you know exactly what browsers and languages are in use.

"..." alt *for <area>*

```
alt="string"
```

SPEC		
deprecated	required	version
NO	YES	HTML 3.2
BROWSER SUPPORT		
IE7	FF2	Saf3	Op9.5+
NONE	NONE	NONE	FULL

Example

This `alt` attribute explains the link destination—a Wikipedia entry:

```
<map name="bigthings" id="bigthings">
  <area shape="rect" coords="35,4,205,108"
      href="http://en.wikipedia.org/wiki/Australia's_Big_Things"
      alt="Australia's Big Things (on Wikipedia)"/>
  ⋮
</map>
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
border="0" usemap="#bigthings"/></p>
```

In the event that the user can't view the image—perhaps because he or she is accessing your page over a very slow connection, because an incorrect `src` (p. 346) attribute has been defined, or because the user is visually impaired and is accessing the content using a screen reader—the `alt` attribute provides alternative content that can be displayed instead of the image.

Thus, an `alt` attribute applied to the `area` element will render over that `area` if the image isn't displayed.

Value

This attribute takes as its value text that's equivalent to the purpose or destination of the link (as defined by the `href` (p. 324) attribute). For a full rundown of how best to handle content inside the `alt` attribute, refer to the `alt` (p. 335) element type.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
None	None	None	None	None	None	None	None	None	None	Full

Unlike the `img`'s `alt` attribute (p. 335), support for the `alt` attribute's application to the `area` element is poor. In cases where the image was unavailable, only the `img`'s `alt` attribute was displayed by the tested browsers. The alternative text for the clickable areas defined by the `area` elements' `alt` attributes wasn't rendered by any browser except Opera 9.5.

"..." **coords** *for <area>*

coords="*coordinates*"

Image and Media Elements

Example

This coords attribute defines the top-left and bottom-right coordinates for a rectangular shape:

```
<map name="bigthings" id="bigthings">
  <area shape="rect" coords="35,4,205,108"
      href="http://en.wikipedia.org/wiki/Australia's_Big_Things"
      alt="Australia's Big Things (on Wikipedia)"/>
  ⋮
</map>
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
border="0" usemap="#bigthings"/></p>
```

The coords attribute tells the browser the shape of the hotspot. Its use depends on the type of shape (p. 326) that's being applied, as detailed below.

Value

The values that are used in the coords attribute are as follows:

■ For rectangular shapes ("rect"), the coords attribute will take four values: $x1$, $y1$, $x2$, and $y2$. These values define the top-left corner of the rectangle (how many pixels along and down from the image's top-left corner the boundary will appear), and the bottom-right corner (how many pixels along and up from the image's bottom-right corner the boundary will appear).

■ For circular shapes ("circ"), three values are required: x, y, and r. The x and y coordinates tell the browser where the circle's center point is, while the r value specifies the radius of the circle.

■ Polygonal shapes ("poly"), which are almost always created using a WYSIWYG HTML editor such as Dreamweaver, are defined by a series of x, y coordinates, each of which relates to a point on the polygon's outline.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

"..." **href** *for <area>*

```
href="uri"
```

SPEC		
deprecated	required	version
NO	NO	HTML 3.2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The `href` below defines a link to a page on Wikipedia:

```
<map name="bigthings" id="bigthings">
  <area shape="rect" coords="35,4,205,108"
      href="http://en.wikipedia.org/wiki/Australia's_Big_Things"
      alt="Australia's Big Things (on Wikipedia)"/>
  ⋮
</map>
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
border="0" usemap="#bigthings"/></p>
```

The `href` defines the destination of the link for this `area`. It may be a web page in the same directory, a page somewhere else on the same server, a location within the current page, or a document stored on another server.

For a full description of the `href` attribute, refer to the entry for the `a` element's `href` (p. 151)—its usage and syntax is exactly the same when applied to an `area`.

Value

This attribute takes as its value the location of the destination document relative to the referencing document, relative to the server root, or as a complete URI containing the `http://` protocol, the server name, and the path to the document on that server. It may also contain reference to the `ftp:` or `mailto:` protocols.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

nohref *for <area>*

```
nohref="nohref"
```

SPEC		
deprecated	required	version
NO	NO	HTML 2
BROWSER SUPPORT		
IE7	FF2	Saf3	Op9.5
NONE	NONE	NONE	NONE

Example

The nohref attribute is used here in place of an href:

```
<map name="bigthings" id="bigthings">
  <area shape="rect" coords="35,4,205,108"
      nohref="nohref"
      alt="The giant prawn, another of Australia's Big Things"/>
  ⋮
</map>
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
border="0" usemap="#bigthings"/></p>
```

The nohref attribute is intended to inform the browser that an href attribute isn't present, when in fact you could more easily do that by, well, not including an href!

The example shows the specification of the nohref in XHTML-compliant syntax, with an attribute and value pairing, but it can be used as an attribute on its own in HTML, as shown below:

```
<map name="bigthings" id="bigthings">
  <area shape="rect" coords="35,4,205,108" nohref
      alt="The giant prawn, another of Australia's Big Things"/>
  ⋮
</map>
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
border="0" usemap="#bigthings"/></p>
```

Image and Media Elements

Value

"`nohref`" is the only value this attribute can take.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
None	None	None	None	None	None	None	None	None	None	None

The HTML specifications provide no indication as to what browsers should do with this attribute, so it's difficult to say which browsers are and aren't compatible. All the tested browsers have been described as providing no support for this attribute, although given that they have no guide on what they should be doing to support it, this seems unfair.

In essence, this attribute is entirely useless, and you have little to gain from using it.

shape *for <area>*

```
shape=" { circle | default | poly | rect } "
```

SPEC		
deprecated	required	version
NO	NO	HTML 3.2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

This `shape` attribute is set to "`rect`", for rectangle:

```
<map name="bigthings" id="bigthings">
  <area shape="rect" coords="35,4,205,108"
      href="http://en.wikipedia.org/wiki/Australia's_Big_Things"
      alt="Australia's Big Things (on Wikipedia)"/>
  ⋮
</map>
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
border="0" usemap="#bigthings"/></p>
```

The `shape` attribute allows the author to define some simple area hotspots, using "`rect`" or "`default`" for rectangles, and "`circle`" or "`poly`" for more complex polygon shapes. The `coords` attribute will differ depending on the type of shape

that's specified; `"circle"` is the simplest, as it requires just three values, while `"poly"` is the most complex, as any number of coordinates may be specified for it.

Value

`"circle"`, `"default"`, `"poly"`, and `"rect"` are the W3C-approved attribute values, but some browsers will also recognize variants of these, namely the abbreviated `"circ"`, and the expanded `"polygon"` and `"rectangle"`. It's best to stick to the approved attribute values, though.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

tabindex *for <area>*

`tabindex="number"`

SPEC		
deprecated	required	version
NO	NO	HTML 3.2
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The `tabindex` is set to `"3"` for the link in the `area` below:

```
<map name="bigthings" id="bigthings">
  <area shape="rect" coords="35,4,205,108" tabindex="3"
      href="http://en.wikipedia.org/wiki/Australia's_Big_Things"
      alt="Australia's Big Things (on Wikipedia)"/>
  ⋮
</map>
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
border="0" usemap="#bigthings"/></p>
```

The `tabindex` is used to define a sequence that users follow when they use the Tab key to navigate through a page. By default, the natural tabbing order will match the source order in the markup. In certain circumstances it may be necessary to override

the default tabbing order, but it's strongly recommended that you craft a page in a logical flow and let the browser work through it in the default order—an approach that negates the need for the `tabindex` attribute.

A `tabindex` can start at 0 and increment in any value. As such, the sequence 1, 2, 3, 4, 5 would be fine, as would 10, 20, 30, 40, 50. If you need to introduce a `tabindex`, it's advisable to use a sequence that contains intervals (like the second example provided), as this will give you the opportunity to inject other controls later if need be (for example, 10, 15, 20) without having to reindex all the `tabindex` values on the page. Should a given `tabindex` value be applied to more than one element in error, the tabbing order of those affected elements will be as per the source markup order.

If a `tabindex` is set anywhere on a page—even if it's the hundredth link or the fiftieth form control—the tab order will start at the element with the lowest `tabindex` value, and work through the increments. Only *then* will the tab order take in the remaining elements for which no `tabindex` has been set. As such, great care must be taken to ensure that adding a `tabindex` doesn't harm the usability of the page as a whole.

If the `disabled` attribute is set on an element which has a `tabindex`, that `tabindex` will be ignored.

Value

This attribute takes a number value.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

"..." **target** *for <area>*

SPEC		
deprecated	required	version
YES	NO	HTML 3.2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

```
target=" { _blank | frame name | _parent | _self |
_top } "
```

Example

The `target` attribute for this `area` is set to "_top":

```
<map name="bigthings" id="bigthings">
  <area shape="rect" coords="35,4,205,108" target="_top"
      href="http://en.wikipedia.org/wiki/Australia's_Big_Things"
      alt="Australia's Big Things (on Wikipedia)"/>
  ⋮
</map>
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
border="0" usemap="#bigthings"/></p>
```

The `target` attribute is deprecated and its use as a layout mechanism, like that of `frameset` (p. 492), is no longer common. However, if you do find yourself having to maintain a frameset-based web site, you may need to open links defined in `area` elements in frames or windows other than the one in which the image map resides.

Value

This attribute can take the following values:

"_blank"   sends the results to a completely new window

"frame name"   sends the results to a frame with a custom name

"_parent"   sends the results to the parent `frameset` for the current `frame`

"_self"   displays the form's submission results in the same frame (This attribute isn't normally required, as this is the default behavior unless the `base` (p. 75) element specifies otherwise. In that case, you'd need to override the specification using "_self", for example <base **target="searchresults"** />.)

"**_top**"   sends the results to the absolute top-level `frameset` (in effect, the whole browser window), no matter how many nested levels down the current `frame` is located

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

---

bgsound

```
<bgsound balance="number" loop="number" src="uri"
volume="number">
```

SPEC		
deprecated	empty	version
NO	YES	N/A
BROWSER SUPPORT		
IE5.5+	FF2	Saf3	Op9.5
FULL	NONE	NONE	NONE

The `bgsound` element is used to play an audio file when the page loads, and has a handful of attributes to control that file.

This is a nonstandard element (it was never defined in *any* standard), so there's little point in writing it to be XHTML-compliant, as it will never validate.

For more information, see http://reference.sitepoint.com/html/bgsound/.

---

embed

```
<embed alt="string" height=" { number |
percentage } " hidden=" { true | false } "
pluginspage="uri" src="uri" type="MIME type"
width="number">
</embed>
```

SPEC		
deprecated	empty	version
YES	YES	N/A
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
PARTIAL	PARTIAL	PARTIAL	PARTIAL

The `embed` element is a nonstandard but well-supported element that is used to embed multimedia content, including media types that might not normally be natively supported by the browser (it can also be used for embedding media that *are* supported, for example images in **.jpg**, **.gif**, or **.png** format).

For more information, visit http://reference.sitepoint.com/html/embed/.

Other Relevant Stuff

`<···>` object (p. 355)

specifies a generic, multipurpose container for a media object

`<···>` noembed (p. 355)

alternative content for browsers that do not support embed

`<···>` # img

```
<img alt="string" height=" { number | percentage } "
src="uri" width=" { number | percentage } " />
```

SPEC		
deprecated	empty	version
NO	YES	HTML 2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+ Op9.2+
FULL	FULL	FULL FULL

Example

Here's an `img` element for which only the required attributes are specified (`src` (p. 346) and `alt` (p. 335)):

```
<img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"/>
```

The `img` element provides a means for embedding an image in the document, which can be used for as many different purposes as your imagination allows. With just the couple of required attributes shown in the example, the `img` provides a reference to the image file to display, and a text alternative should the image not be available for whatever reason. A number of optional, and deprecated, attributes are covered below in detail.

As it's an empty element, the `img` element requires a trailing slash if it's to be XHTML-compliant, but it can be expressed in HTML as follows:

```
<img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina">
```

As an image is an inline element, a break isn't created before or after it, so the following HTML would render with the text appearing on either side of the image:

```
<p>Driving along, we spotted a giant prawn <img
    src="giant-prawn.jpg" alt="The Giant Prawn at Ballina">,
    so we had to stop and take a closer look.</p>
```

The somewhat unsightly effect of this markup can be seen in Figure 7.1.

Driving along, we spotted a giant prawn   , so we had to stop
and take a closer look.

Figure 7.1: The image failing to create a break in the text

However, with CSS you can achieve great control over the `img` element, creating wrapping text with margins (or gutters, to use the print analogy), border styles, and more.

Some presentational attributes, which control alignment and dimensions, are covered below, but these effects are best controlled using CSS.

Use This For …

This element is used to place illustrative images—pictures that convey some important information. It's not used for purely decorative images that don't offer any information, and which could easily be removed from the page without detriment to its content. Such noncritical, decorative images may be better implemented using the CSS `background-image` property.

This element can be used for photographs, charts and graphs, and maps. Even when it's used for images that may be difficult to accurately describe in words (for instance,

Image and Media Elements

a trend may be seen easily on a graph, but its meaning may be impossible to understand in the absence of the image, for whatever reason), it's important to ensure that an alternative description is available. See the sections on the `alt` (p. 335) and `longdesc` (p. 344) attributes for more information.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

Other Relevant Stuff

<···>   `input` (p. 251)

defines the input control for a form

align *for <img>*

```
align=" { bottom | left | middle | right | top } "
```

SPEC		
deprecated	required	version
YES	NO	HTML 2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The `align` attribute for this `img` is set to `"right"`:

```
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
    align="right"/>Driving along, we spotted a giant prawn, so had to
    stop and take a closer look.</p>
```

The `align` attribute is used to specify how the image sits on the page in relation to surrounding text and other elements. It is a fairly rudimentary attribute—and a very presentational one—that allows you to move an image to the right or left, with text wrapping around it accordingly (although without surrounding whitespace, which generally produces quite an ugly result). You can also alter the adjacent text

alignment so that the first line of the text aligns with the top, middle, or bottom of the image.

The example markup shown with a right-aligned image would appear as shown in Figure 7.2.

Figure 7.2: A right-aligned image

If the value were changed to `"top"`, the effect would be very different, as Figure 7.3 shows.

Figure 7.3: An image for which `align` is set to `"top"`

As the two examples show, the `align` attribute is no precision instrument when it comes to layout!

If an `img` is aligned `"left"` or `"right"`, text will continue to wrap around it until it encounters either:

- a `br` (p. 40) element with a `clear` (p. 42) attribute
- any other element whose CSS `clear` property is set to `"left"`, `"right"`, or `"both"`

Value

`"bottom"`, `"left"`, `"middle"`, `"right"`, and `"top"` are the values that this attribute may take.

Nonstandard, proprietary attributes (which are still supported by Internet Explorer) that are acceptable include `"absbottom"`, `"absmiddle"`, `"baseline"`, and `"texttop"`.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

This attribute is now deprecated, and is highly presentational in its nature. The desired visual effects can all be achieved with CSS and, as such, this attribute shouldn't be used—it's presented here for informational purposes only.

"..." `alt` *for <img>*

`alt="string"`

SPEC		
deprecated	required	version
NO	YES	HTML 2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The `alt` attribute clearly explains what the image contains:

```
<img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina" />
```

In the event that the user can't view the image—perhaps because he or she is accessing your page over a very slow connection, because an incorrect `src` (p. 346)

attribute has been defined, or because the user is visually impaired and is accessing the content using a screen reader—the `alt` attribute provides alternative content that can be displayed instead of the image.

Value

This attribute takes as its value text that's equivalent to the content or purpose of the image:

- For an image that conveys important information, describe the image.
- For an image that's contained inside an `a` (p. 144) element, and is thus being used as a link, use an `alt` attribute that explains where the link goes, or what activating the link will do.
- For purely decorative images that offer no additional information to the page content, use an empty `alt` attribute:

```
alt=""
```

  Don't simply omit the `alt` attribute—it's required, and absence of an `alt` attribute can cause problems for screen readers, which, in an effort to provide information about the image, may read out the image's filename, for example.

- If an image is supplementary to the surrounding text, but isn't purely decorative, don't simply replicate the content from the surrounding text in the image's `alt` attribute. In this instance, you should use an empty `alt` attribute.
- If the image is a graphic, for example a chart or graph, and if the pattern or process illustrated in the image is explained alongside or nearby in text, apply an empty `alt` attribute to the image.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

Most browsers don't display the content of the `alt` attribute unless the image is actually missing—it is, after all, supposed to be an *alternative* to the image. Internet Explorer, however, will display the value for the `alt` in the form of a tooltip when the user mouses over the `img` element in question. This isn't the correct behavior, so be mindful of the fact that the other browsers are operating according to the specification, and it's Internet Explorer's interpretation that's slightly wrong. If you *do* want to create a tooltip effect on an image, use the `title` (p. 504) attribute.

Note that Internet Explorer and Opera don't deal with missing images very well, as Figure 7.4 shows (that screenshot was taken using IE6 on Windows XP).

> ☒ The Giant Prawn at Ballina Driving along, we spotted a giant prawn, so had to stop and take a closer look.
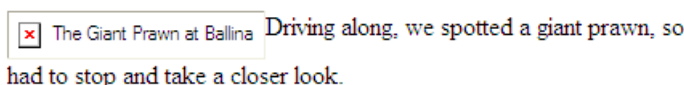
Figure 7.4: Alternative text showing for a missing image in IE

Despite the fact that the image is absent, both of these browsers honor the `height` and `width` attributes of the missing image (if they're set) and display a placeholder frame. If the `alt` text takes up more space than the image's dimensions, neither shows the alternative text properly—it's a little like looking through a letterbox. Safari is even worse, displaying a question mark where the image should appear, and no alternative text. Arguably the best of the bunch is Firefox, which doesn't display a placeholder frame, and allows whatever space is required for the alternative text, making it a much more usable implementation.

“…” **border** *for <img>*

`border="number"`

SPEC		
deprecated	required	version
YES	NO	HTML 2
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

This `border` attribute is set to `"5"`:

```
<img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
    border="5"/>
```

By default, an image will display without a border (unless it's contained inside an `a` (p. 144) element and CSS has *not* been used to remove the border). The `border` attribute rectifies this, and allows you to specify a border in pixels. Depending on the browser, the border will either be black, or will match the color of the document's text.

Figure 7.5: An image with a border thickness of five pixels

Value

This attribute takes as its value a number that represents the width of the border in pixels.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

Although support for this attribute is good, it's now deprecated, and is highly presentational in its nature. The desired visual effects can all be achieved with CSS and, as such, this attribute shouldn't be used—it's presented here for informational purposes only.

`height` *for `<img>`*

```
height=" { number | percentage } "
```

SPEC		
deprecated	required	version
NO	NO	HTML 2
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

Here, the `height` attribute is set to `"30"`:

```
<img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
    height="30"/>
```

An image doesn't require a `height` attribute, but it has its uses. The main use for specifying the `height` (and `width` (p. 351)) is to improve the user experience while a page is loading. If the dimensions are specified in the markup, as the page is loaded, the space required for the images is reserved by the browser; without this information, the browser doesn't know how big the image is, and can't allocate the appropriate space to it. On a slow-loading page, the effect can be quite disturbing, as content is constantly reflowed as each new image appears on the page.

The downside of specifying a `height` (and `width`) is that if you later decide to update an image that's used site-wide—for example, a company logo—you'd need to change the dimension attributes for each page of the site. Depending on how your web site is managed (manually, in a template-driven way, via a CMS, or through server-side includes), this may either be a minor niggle, or a real issue for you. It's a case of weighing up the pros and cons in each situation.

If the `height` attribute is set by itself, but no `width` attribute is set, the image will be rescaled proportionally, as shown by the three images (set to `"30"`, `"100"`, and `"200"` pixel widths, respectively) in Figure 7.6.

Figure 7.6: Rescaling an image by setting the `height` attribute alone

If, however, the correct `width` attribute is set, but the `height` differs, the image will appear stretched, as shown in Figure 7.7.

Figure 7.7: Stretched images resulting from the application of an incorrect `height` attribute

Note that it's not a good practice to rescale images in your markup. If you need an image with dimensions of 200×200 pixels, *don't* drop in an image of 1000×1000

pixels and use HTML attributes to rescale it. Not only does this approach force the user to download a large image that's rendered at a much smaller size on the web page, but the result is usually quite untidy. The correct method is to rescale the image in a graphics application first, and then place the correctly sized image on the page.

Value

This attribute takes a number representing the height of the image in pixels, or as a percentage of the containing element.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

"..." hspace *for <img>*

`hspace="number"`

SPEC		
deprecated	required	version
YES	NO	HTML 2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

Here, the `hspace` attribute is set to `"10"` pixels around a left-aligned image:

```
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
    align="left" hspace="10"/>Driving along, we spotted a giant prawn,
    so we had to stop and take a closer look.</p>
```

When an image is aligned (p. 333) left or right, text will flow around the image, but no space will appear between the text and the image edge. The `hspace` (and related `vspace` (p. 350)) attribute provides a little breathing space, but it applies space on both sides of the image, which is not entirely flexible, as Figure 7.8 shows.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nulla sapien neque, vulputate a, cursus consequat, ultricies eu, mi. Etiam est nibh, interdum ut, dapibus at, adipiscing eu, quam.

Driving along, we spotted a giant prawn, so we had to stop and take a closer look.

Quisque in eros ultrices risus gravida vestibulum. Maecenas adipiscing, lacus vel laoreet hendrerit, tortor diam viverra velit, id consectetuer orci purus id tortor.

Figure 7.8: An image to which an hspace of 10 is applied

Value

This attribute takes as its value a number that represents the width of spacing on either side of the image in pixels.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

Although support for this attribute is good, it's now deprecated, and is highly presentational in its nature. The desired visual effects can all be achieved with CSS and, as such, this attribute shouldn't be used—it's presented here for informational purposes only.

"…" **ismap** *for <img>*

```
ismap="ismap"
```

SPEC		
deprecated	required	version
NO	NO	HTML 2
BROWSER SUPPORT		
IE5.5+ FF1+	Saf1.3+	Op9.2+
FULL FULL	FULL	FULL

Example

The `ismap` attribute is set as follows:

```
<img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
    ismap="ismap"/>
```

The `ismap` attribute is rarely seen, but when it is, it's used to indicate that the user's mouse actions over the image should be processed using a server-side image map. When the user clicks on an area of the image using the mouse (it can't be activated by keyboard controls), the coordinates are sent back to the server in the form of a query string in the URI.

It's far more common to see this behavior handled on the client side with the `usemap` (p. 348) attribute.

Value

`"ismap"` is the only value that this attribute can take.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

longdesc *for <img>*

```
longdesc="uri"
```

SPEC		
deprecated	required	version
NO	NO	HTML 2
BROWSER SUPPORT		
IE5.5+ FF1+ Saf1.3+ Op9.2+		
PARTIAL PARTIAL PARTIAL PARTIAL		

Example

This `longdesc` attribute refers to a text file "prawn.txt":

```
<img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
     longdesc="prawn.txt"/>
```

The `alt` (p. 335) attribute is intended to be a *short* alternative for the image, and shouldn't be used for lengthy descriptions of the image. The attribute that's used to provide a pointer to further information is `longdesc`. Unfortunately, it's so poorly supported that it's almost unusable (see below for more information).

Value

This attribute takes as its value the URL for a file that contains the extra descriptive text, most likely a simple **.txt** file.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial

The `longdesc` attribute has almost no practical use, even with today's good, standards-aware browsers. Despite the best intentions, no browser on the support charts makes it clear to the user when extra information is available for the image in the form of a descriptive text file, and this level of indifference toward the attribute is likely to continue. Even the technology that would benefit the most from the presence of this attribute—assistive technology such as screen readers—is oblivious to the presence of a `longdesc`. Only Firefox appears to show a basic level of awareness of the attribute: if you right-click on the image and choose **Properties**,

the `longdesc`'s file location is visible next to the **Description** title, as shown in Figure 7.9.
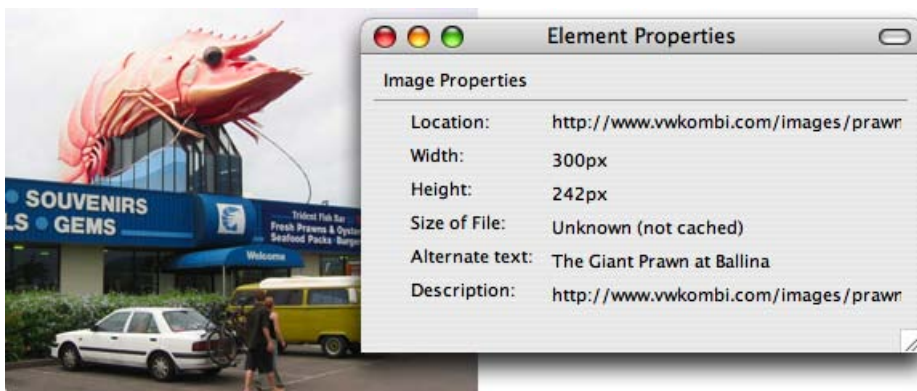
Figure 7.9: The `longdesc` shows in Firefox's contextual menu as the **Description**

A much safer option is to avoid this attribute altogether, and simply to create a link that anyone can access or see, perhaps linking from picture caption text.

name *for <img>*

`name="string"`

SPEC		
deprecated	required	version
NO	NO	HTML 2
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The `name` attribute provides a means by which the image may be referenced:

```
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
    name="giantprawn"/></p>
```

The `name` attribute is one method for referencing an `img` element using JavaScript (the alternative, more forward-thinking method is to reference it using the `id` (p. 499) attribute). It has historically been used for techniques such as image swaps for rollovers (which have since largely been replaced using CSS techniques).

Value

This attribute takes as its value any name that the developer chooses, as long as it doesn't contain spaces or special characters.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

src *for <img>*

```
src="uri"
```

SPEC		
deprecated	required	version
NO	YES	HTML 2
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The src attribute for this image shows that the image is located in the same directory as the web page:

```
<img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"/>
```

The src attribute instructs the browser where on the server it should look for the image that's to be presented to the user. This may be an image in the same directory, an image somewhere else on the same server, or an image stored on another server.

The example refers to an image that's located in the same directory as the web page that's calling it, but if the image was stored in a directory that was one level higher than the referencing document, the syntax would be as follows:

```
<img src="../giant-prawn.jpg" alt="The Giant Prawn at Ballina"/>
```

Here, ../ equates to "move up one directory in the hierarchy."

You can also reference an image relative to the web site's root (that is, any file or folder after the domain name):

```
<img src="/giant-prawn.jpg" alt="The Giant Prawn at Ballina"/>
```

This basically means "display the image **giant-prawn.jpg** that can be found in **www.example.com/**." This is a very handy way of referencing the file, as you could move the document that referenced the image to any location on the file system without breaking the link.

If you're referencing an image that's held on another server, you'd express the `src` using a complete URI, as follows:

```
<img src="http://www.example.com/giant-prawn.jpg"
     alt="The Giant Prawn at Ballina"/>
```

Value

This attribute takes as its value the location of the image relative to the referencing document, relative to the server root, or as a complete URI containing the `http://` protocol, the server name, and the path to the document on that server.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

"..."   **usemap** *for <img>*

```
usemap="#map name"
```

SPEC		
deprecated	required	version
NO	NO	HTML 2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

Here, the `usemap` attribute references a map named `"bigthings"`:

```
<map name="bigthings" id="bigthings">
  <area shape="rect" coords="35,4,205,108"
      href="http://en.wikipedia.org/wiki/Australia's_Big_Things"
      alt="Australia's Big Things (on Wikipedia)"/>
  <area shape="rect" coords="136,163,255,230"
      href="http://vwkombi.com/"
      alt="The VW Kombi, another Aussie icon"/>
</map>
⋮
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
usemap="#bigthings"/></p>
```

The `usemap` attribute tells the browser which `map` (p. 352) element in the document it should refer to. Then one or more hotspots (specified in the `area` (p. 318) element) are defined, to act as overlays on the image. These areas are similar to links (the `a` (p. 144) element) in that they allow the user to click to go to the page identified in the `area`'s `href` attribute (they're also keyboard-navigable).

When an image map is defined in this way, the image displays no hint that the area is actionable. The image must, in itself, hint at the clickable areas, perhaps by containing button-like features, or by the presence of instructional text near to the image. The clickable areas are visible at design time in some web authoring applications, such as Dreamweaver, which shows the clickable areas as depicted in Figure 7.10.

Image and Media Elements

Figure 7.10: Dreamweaver shows the clickable areas on the image in Design View

Value

This attribute takes as its value a reference to the map's `name` attribute in the form of a hash character (`"#"`) plus the `"name"`, like so:

```
usemap="#bigthings"
```

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

"..." **vspace** *for <img>*

`vspace="number"`

SPEC		
deprecated	required	version
YES	NO	HTML 2
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

This `vspace` attribute is set to `"10"` pixels around a left-aligned image:

```
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
    align="left" vspace="10"/>Driving along, we spotted a giant prawn,
    so we had to stop and take a closer look.</p>
```

The `vspace` attribute is similar to `hspace` (p. 341) and provides a bit of breathing space above and below an image, although it's not exactly a precision design tool. Figure 7.11 shows the problem that occurs when an image is placed between paragraphs (the spacing appears to be unequal because the `p` (p. 70) element already has a bottom margin of its own, which exists in addition to the `vspace` of `"10"`).

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nulla sapien neque, vulputate a, cursus consequat, ultricies eu, mi. Etiam est nibh, interdum ut, dapibus at, adipiscing eu, quam.

Driving along, we spotted a giant prawn, so we had to stop and take a closer look.

Quisque in eros ultrices risus gravida vestibulum. Maecenas adipiscing, lacus vel laoreet hendrerit, tortor diam viverra velit, id consectetuer orci purus id tortor.

Figure 7.11: Image to which a `vspace` of 10 is applied (note the space above and below image)

Value

This attribute takes as its value a number representing the amount of space to appear above and below the image in pixels.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

Although support for this attribute is good, it's now deprecated, and is highly presentational in its nature. The desired visual effects can all be achieved with CSS and, as such, this attribute shouldn't be used—it's presented here for informational purposes only.

"…" width *for <img>*

```
width=" { number | percentage } "
```

SPEC		
deprecated	required	version
NO	NO	HTML 2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

Here, the width attribute is set to "300":

```
<img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
    width="300"/>
```

An image doesn't require a width attribute but, like the height (p. 339) attribute, width has its uses (refer to the advantages set out in the height attribute reference).

If the width attribute is set by itself, but no height attribute is set, the image will be rescaled proportionally. In every other respect, the width attribute is identical in usage and behavior to the height attribute.

Value

This attribute takes as its value a number representing the width of the image in pixels, or a percentage of the containing element.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

<···>
map

```
<map name="string">
</map>
```

SPEC		
deprecated	empty	version
NO	NO	HTML 3.2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

This code defines an image `map` named `"bigthings"`:

```
<map name="bigthings">
  <area shape="rect" coords="35,4,205,108"
      href="http://en.wikipedia.org/wiki/Australia's_Big_Things"
      alt="Australia's Big Things (on Wikipedia)"/>
  <area shape="rect" coords="136,163,255,230"
      href="http://vwkombi.com/" alt="The VW Kombi,
      another Aussie icon"/>
</map>
```

The `map` element is a container for a number of `area` (p. 318) elements that define specific areas within an image. The map has one required attribute, the `name` attribute (p. 354), which is associated with the image's `usemap` attribute (p. 348) to create a relationship between the image and the `map`.

Note that not every `area` needs to be actionable. It's perfectly acceptable to use `area` *without* an `href` attribute for the purposes of outlining a given area, and for providing additional help in the form of a tooltip created using the `title` (p. 504) attribute.

map 353

However, this technique is inaccessible to people who use anything other than a mouse to navigate the web page, since there will be no mouse hover event for those users—the usual trigger for the tooltip display.

Use This For ...

This element is used to create specific clickable hotspots within a single larger image, and provides an alternative to the process whereby a larger image is sliced into smaller images, and numerous links are created using the `a` (p. 144) element.

In the past, the `map` and `area` elements have often been used to create navigation bars and the like. However, that approach is used with decreasing frequency, as CSS support is excellent, and in most cases provides a more suitable mechanism for creating navigation blocks.

Other examples for the usage of the `map` element include:

- identifying regions on a map, be they countries on a world map or areas on regional maps
- outlining locations in a floor plan—for example, a shopping mall or theme park
- creating an overlay for a photo whose different components or features need identification—for example, highlighting the parts of the inside of a laptop computer for the purposes of repair

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

Other Relevant Stuff

area (p. 318)

defines a hotspot within a client-side image map

name *for <map>*

```
name="string"
```

SPEC		
deprecated	required	version
NO	YES	HTML 3.2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The name attribute provides a means for referencing the map:

```
<map name="bigthings">
  <area shape="rect" coords="35,4,205,108"
      href="http://en.wikipedia.org/wiki/Australia's_Big_Things"
      alt="Australia's Big Things (on Wikipedia)"/>
  <area shape="rect" coords="136,163,255,230"
      href="http://vwkombi.com/" alt="The VW Kombi,
      another Aussie icon"/>
</map>
⋮
<p><img src="giant-prawn.jpg" alt="The Giant Prawn at Ballina"
border="0" usemap="#bigthings"/></p>
```

The name attribute is required to provide a reference for the map element. The image element refers to this attribute (through its usemap attribute (p. 348)) using a combination of the "#" character and the map's name attribute, as the example HTML shows.

Value

This attribute can take as its value any name that the developer chooses, so long as it doesn't contain spaces or special characters.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

noembed

```
<embed>
</embed>
```

SPEC		
deprecated	empty	version
YES	NO	N/A
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

The noembed element is used to provide alternative content for browsers that do not support the embed (p. 330) element. It is not defined by any standard (it was introduced by early Netscape browsers), and as such there are no hard guidelines as to what it may or may not contain.

For more information, visit http://reference.sitepoint.com/html/noembed/.

Other Relevant Stuff

embed (p. 330)

specifies a generic container for a media object

object

```
<object archive="uri" border=" { number |
percentage } " classid="class ID" codebase="uri"
codetype="MIME type" data="uri" height=" { number |
percentage } " type="MIME type" width=" { number |
percentage } ">
</object>
```

SPEC		
deprecated	empty	version
NO	NO	HTML 4
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
PARTIAL	PARTIAL	PARTIAL	PARTIAL

Example

In this example, a super-simple object element is used to display an image:

```
<object data="giant-dog.jpg">
</object>
```

The object element's purpose is to embed into a document a variety of different kinds of media files. Historically, it was used primarily for placing ActiveX controls

onto a page, but it can also be used to embed images (**.gif**, **.jpg**, and so on), movie files and applets, video files, PDF documents, Flash, and even HTML.

While this element is specified in the HTML 4 recommendation, and thus constitutes valid markup, it's often shunned in favor of the better supported, but nonstandard `embed` (p. 330) element.

To embed Flash using `object`, rather than going down the nonstandard `embed` route, use the following markup:

```
<object data="movie.swf"
    type="application/x-shockwave-flash"
    width="200" height="100">
  <param name="movie" value="movie.swf">
  <param name="wmode" value="opaque">
</object>
```

The `param` (p. 376) element with the `name` of `"movie"` helps Internet Explorer to load the Flash file. The `"wmode"` encourages IE to play nicely with the `z-index` property, allowing other elements to be placed on top of the Flash movie.

Use This For ...

This element is used for media files, applets, and ActiveX objects. For images, it's currently better practice to use the completely supported `img` (p. 331) element, rather than take a risk using `object`; `img` is also far less clunky to use.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial

This element is not completely supported, although it has to be said that the support that is available very much depends on the type of object that's being embedded. Internet Explorer displays simple images poorly, with horizontal and vertical scrollbars, but is able to display an embedded Word document, albeit with a warning in the form of an ActiveX alert. Safari and Firefox, on the other hand, behaves impeccably with images, yet no amount of coaxing or installing of third-party plugins will allow the Word document to display—even with Microsoft Office installed on

the host machine. For this reason, before you apply it, you should consider the purpose for which you need to use the `object` element. If it's to display images, you'll likely be better off to use the `img` element.

Other Relevant Stuff

`<···>` embed (p. 330)

specifies a generic container for a media object

`<···>` img (p. 331)

specifies an inline image element

`"..."` # align *for <object>*

```
align=" { bottom | left | middle | right | top } "
```

SPEC		
deprecated	required	version
YES	NO	HTML 4
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The `align` attribute for this `object` is set to `"right"`:

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
    codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/
➥swflash.cab#version=9,0,28,0" align="right" width="320"
    height="285" title="Flash tester">
  <param name="movie" value="flash-test.swf"/>
  <param name="quality" value="high"/>
</object>
```

The `align` attribute is used to specify how the object sits on the page in relation to surrounding text and other elements. It is a fairly rudimentary attribute—and a very presentational one—that allows you to move an object to the right or left, with text wrapping around the object accordingly (although generally, whitespace isn't included, which tends to result in quite an ugly result). You can also change the way that adjacent text aligns with the object so that the first line of the text aligns with the top, middle, or bottom of the object.

The example markup shown would render as illustrated in Figure 7.12.

Figure 7.12: A right-aligned object containing a Flash video

Aside from aligning `"left"` or `"right"`, the `align` attribute isn't very flexible when it's applied to an `object`.

If an `object` is aligned `"left"` or `"right"`, text will continue to wrap around it until either:

■ a `br` (p. 40) element with a `clear` (p. 42) attribute is encountered

■ any other element whose CSS `clear` property is set to `"left"`, `"right"`, or `"both"` is encountered

Value

Possible values for this attribute include `"bottom"`, `"left"`, `"middle"`, `"right"`, and `"top"`.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

This attribute is now deprecated, and is highly presentational in its nature. The desired visual effects can all be achieved with CSS and, as such, this attribute shouldn't be used—it's presented here for informational purposes only.

"..." archive *for <object>*

```
archive="uri"
```

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
PARTIAL	PARTIAL	PARTIAL	PARTIAL

Example

In this code, the `archive` attribute obtains supporting classes from **giant-dog.jar**:

```
<object classid="java:giant-dog.class" archive="giant-dog.jar">
</object>
```

The `archive` attribute allows the author to define a number of files that are required for the `object` content to render or run correctly, effectively preloading the necessary resources.

Value
This attribute takes as its value a space-separated list of URLs of the files required.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial

This attribute's compatibility is dependent on the type of object with which it's used.

"..." **border** *for <object>*

```
border="number"
```

SPEC		
deprecated	required	version
YES	NO	HTML 4
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

This border attribute is set to "10":

```
<object data="giant-dog.jpg" border="10"></object>
```

By default, an object will not have a border around it. The border attribute rectifies this, allowing you to set a border with a width specified in pixels. Depending on the browser, the border will either be black, or will match the color of the text attribute of the body (p. 37) element. An example is shown in Figure 7.13.

Figure 7.13: An object displaying with a border thickness of ten pixels

Value

This attribute takes a number representing the width of the border in pixels.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

This attribute is poorly supported and highly presentational. CSS should be used to control appearance instead.

"..." `classid` *for <object>*

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT			

IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

```
classid="class ID"
```

Example

This example shows the definition of a `classid` attribute for a Flash movie:

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
    codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/
➥swflash.cab#version=9,0,28,0" width="320" height="285" title="Flash
    tester">
  <param name="movie" value="flash-test.swf"/>
  <param name="quality" value="high"/>
  ⋮
</object>
```

The `classid` attribute provides a reference that the browser can use to understand how the object should be implemented. It's usually used to ensure that the browser has the correct version of the control.

Value
This attribute takes as its value the URI of a document on the Web, or an internal reference in the form of `"classid:object-id"`, as shown in the example above (`"classid:D27CDB6E-AE6D-11cf-96B8-444553540000"`).

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

The compatibility of this attribute depends upon the type of object with which it is used.

codebase *for <object>*

```
codebase="uri"
```

SPEC		
deprecated	required	version
NO	NO	HTML 2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

Here, `codebase` defines the base URL of `"http://bestjavaappsever.com/classes/"`:

```
<object classid="calendar.class"
    codebase="http://bestjavaappsever.com/classes/"></object>
```

The `codebase` attribute is used to set the base URL for the value specified in the `classid` attribute, overriding any base URL set in the `head` (p. 61) of the document.

In the example shown, the `codebase` is identified as `"http://bestjavaappsever.com/classes/"` (the complete path suggests that the resources are held on another server), while the `classid` refers to a file entitled `"calendar.class"`. The full address for the calendar code would therefore be interpreted as **http://bestjavaappsever.com/classes/calendar.class**.

Value

This attribute takes a URI, which may be a complete path on another server (for example, `"http://bestjavaappsever.com/classes/"`, or could be a different folder on the same server, as follows:

```
<object classid="calendar.class"
    codebase="/code/java-classes/"></object>
```

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

Every browser listed supports this attribute.

"…" **codetype** *for <object>*

```
codetype="MIME type"
```

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

This `codetype` attribute identifies an embedded Word document:

```
<object data="Hello.doc" codetype="application/msword"></object>
```

The `codetype` attribute defines the MIME type[5] of the embedded `object`, as specified in the `data` (p. 364) attribute. This shouldn't be confused with the `type` (p. 372) attribute, which is used to specify the MIME type of data that the `object` consumes.

Value

This attribute takes as its value a MIME type in the format type/subtype; for example, `"text/html"`, `"image/x-rgb"`, or `"application/java"`.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

Every browser listed supports this attribute.

---

[5]   http://reference.sitepoint.com/html/mime-types/

data *for <object>*

`data="uri"`

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT			

IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The `data` attribute for this object refers to a **.jpg** file:

```
<object data="giant-dog.jpg"
    type="image/jpeg" height="225"></object>
```

The `data` attribute tells the browser where it can find the necessary data or file for the object. In the case of images, it's roughly equivalent to `src` (p. 346), but may point to any number of different file or data types (for example, video files, audio files, or Microsoft Office documents).

Value

This attribute takes as its value the location of the data—the image, video, or audio file, and so on—relative to the referencing document, relative to the server root, or as a complete URI containing the `http://` protocol, the server name, and the path to the document on that server.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

"…" # declare *for <object>*

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT		
IE5.5+	FF2	Saf3	Op9.2+
FULL	NONE	NONE	FULL

```
declare="declare"
```

Example

Here's the declare attribute, specified in XHTML-compliant markup:

```
<object data="giant-dog.jpg" declare="declare"
    type="image/jpeg" height="225"></object>
```

The declare attribute declares an object, but doesn't instantiate it, which may be useful for the purposes of deferring the object's download until it's actually needed. In HTML, it's not necessary to include the attribute and value pairing—all we need is the attribute on its own, as shown here:

```
<object data="giant-dog.jpg" declare
    type="image/jpeg" height="225"></object>
```

Value

"declare" is the only value this attribute can take.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	None	None	None	None	None	None	Full	Full

This element type isn't particularly well supported and has limited practical value.

height *for* *<object>*

```
height=" { number | percentage } "
```

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT			

IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

Here, the `height` attribute is set to `"100"`:

```
<object data="giant-dog.jpg" height="100">
</object>
```

An `object` doesn't require a `height` attribute, but it has its uses. The main reason for specifying the `height` (and `width` (p. 375)) is to improve the user experience while a page is loading. If the dimensions are specified in the markup, the space required for the object is reserved by the browser as the page loads. Without this information, the browser doesn't know how big the object is, and can't allocate the necessary space to it. On a slow-loading page, the effect can be quite unsightly, as content is constantly reflowed as each new object appears on the page.

The downside of specifying a `height` (and `width`) is that if you later decide to update an object that's used site-wide, you'll need to change the dimension attributes for each page of the site. Depending on how your web site's managed (manually, in a template-driven way, via a CMS, or through server-side includes), this may either be a minor niggle, or a real issue for you. It's a case of weighing up the pros and cons in each situation.

If the `height` attribute is set by itself, but no `width` attribute is set, the image will be rescaled proportionally. The results of this approach vary depending on the type of the object in question, and the browser rendering it. If nonproportional dimensions are specified (for instance, a 200x200-pixel object is set to take a `height` of 100 and a `width` of 300 pixels), the results vary: image objects can be stretched or squashed just as they can when different `width` and `height`s are applied using the `img` (p. 331)id element, but not all multimedia objects distort in the same way.

Value

This attribute takes as its value a number that represents the height of the object in pixels, or a percentage of the containing element.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

The compatibility of this attribute is dependent on the type of the object. For objects of `type` image, there is excellent cross-browser support for the `height` attribute.

"…" hspace *for <object>*

`hspace="number"`

SPEC		
deprecated	required	version
YES	NO	HTML 4
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

This `hspace` attribute is set to 30 pixels around a left-aligned image object:

```
<object data="giant-dog.jpg" height="225" width="300" align="left"
    hspace="30"></object>
```

When an object is aligned (p. 357) left or right, text will flow around the image, but no space will appear between it and the image. The `hspace` (and related `vspace` (p. 374)) attribute provides a little breathing space, but it will apply space on both sides of the object, which isn't entirely flexible, as Figure 7.14 reveals.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nulla sapien neque, vulputate a, cursus consequat, ultricies eu, mi. Etiam est nibh, interdum ut, dapibus at, adipiscing eu, quam.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nulla sapien neque, vulputate a, cursus consequat, ultricies eu, mi. Etiam est nibh, interdum ut, dapibus at, adipiscing eu, quam.

Figure 7.14: An image object to which an `hspace` of `"30"` is applied

Value

This attribute takes a number representing the pixel width of the spacing to be applied on either side of the image.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

Although support for this attribute is good, it's now deprecated, and is highly presentational in its nature. The desired visual effects can all be achieved with CSS and, as such, this attribute shouldn't be used—it's presented here for informational purposes only.

"..." name *for <object>*

```
name="string"
```

SPEC		
deprecated	required	version
YES	NO	HTML 4
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The `name` attribute provides a means for referencing the object:

```
<object data="giant-dog.jpg" height="225" width="300"
    align="left" vspace="30" name="dog-information-kiosk">
</object>
```

The `name` attribute is one method for referencing an `object` element in JavaScript (the alternative, forward-thinking method is to reference it by its `id` (p. 499) attribute). This attribute has historically been used for techniques such as image swaps for rollovers, which have since largely been replaced using CSS techniques.

Value

This attribute takes as its value any name that the developer chooses, so long as it doesn't contain spaces or special characters.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

standby *for <object>*

`standby="string"`

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT			
IE7	FF2	Saf3	Op9.5
NONE	NONE	NONE	NONE

Example

Here, the `standby` attribute is used to warn users of a download delay:

```
<object data="Kuata-Joinup.jpg" width="4466" height="535"
    standby="This may take a while. Go put the kettle on.">
</object>
```

For large media files that are pulled into a page using the `object` element, it's preferable not to have what appears to be a large space where the content should be, making the page appear, to all intents and purposes to, be broken. The `standby` attribute allows the author to display a message that will remain on screen only while the `object`'s content is being loaded.

Value

This attribute takes as its value a suitable warning message, as shown in the example.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
None	None	None	None	None	None	None	None	None	None	None

Poor support is provided for this attribute. It's probably better to use a JavaScript technique to create a custom content-is-loading message in the style of your web page or site, which can subsequently be replaced or removed once the `object`'s content has loaded.

tabindex *for <object>*

`tabindex="number"`

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

This code sets the `tabindex` for both form controls:

```
<object data="giant-dog.jpg" height="225" tabindex="1"></object>
```

The `tabindex` is used to define a sequence that users follow when they use the Tab key to navigate through a page. By default, the natural tabbing order will match the source order in the markup. In certain circumstances it may be necessary to override the default tabbing order, but it's strongly recommended that you craft a page in a logical flow and let the browser work through it in the default order—an approach that negates the need for the `tabindex` attribute.

A `tabindex` can start at 0 and increment in any value. As such, the sequence 1, 2, 3, 4, 5 would be fine, as would 10, 20, 30, 40, 50. If you need to introduce a `tabindex`, it's advisable to use a sequence that contains intervals (like the second example provided), as this will give you the opportunity to inject other controls later if need be (for example, 10, 15, 20) without having to reindex all the `tabindex` values on the page. Should a given `tabindex` value be applied to more than one element in error, the tabbing order of those affected elements will be as per the source markup order.

If a `tabindex` is set anywhere on a page—even if it's the hundredth link or the fiftieth form control—the tab order will start at the element with the lowest `tabindex` value, and work through the increments. Only *then* will the tab order take in the remaining elements for which no `tabindex` has been set. As such, great care must be taken to ensure that adding a `tabindex` doesn't harm the usability of the page as a whole.

Value

This attribute takes a number value.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

"..."   # type *for <object>*

```
type="MIME type"
```

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

This `type` attribute is set to `"image/gif"` to display a **.gif** image:

```
<object data="logo.gif" type="image/gif"></object>
```

The `type` attribute lets the author define the MIME type[6] of the data used in the `object`—the file that's specified in the `data` (p. 364) attribute. This is slightly different from the `codetype` (p. 363) attribute, which is used to specify the MIME type of the object itself. If the server sends data with the appropriate MIME type, this attribute may be omitted.

Value

This attribute takes a MIME type in the format type/subtype, for example, `"text/html"`, `"image/x-rgb"`, or `"application/java"`.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

Every browser listed supports this attribute.

---

[6] http://reference.sitepoint.com/html/mime-types/

usemap *for <object>*

`usemap="#map name"`

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

This `usemap` attribute references a map named `"dogmap"`:

```
<p>
  <object data="giant-dog.jpg" usemap="#dogmap"
      height="225" width="300" border="0"></object>
  <map name="dogmap" id="dogmap">
    <area shape="circle" coords="216,120,24"
        href="http://tirauinfo.homestead.com/TheBigDog.html"
        alt="Click on the nose for more info about this big dog!"/>
  </map>
</p>
```

The `usemap` attribute tells the browser which of the `map` (p. 352) elements in the document it should refer to. The hotspots defined (using the `area` (p. 318) element) act as overlays on the object. These areas are similar to links (created with the `a` (p. 144) element), and allow the user to click to go to the page identified in the respective `area`'s `href` attribute. They're also keyboard-navigable.

The `usemap` attribute is only used in the `object` element when the type of object is an image (**.gif**, **.jpg**, or **.png**).

Value

This attribute takes as its value a reference to the map's `name` attribute in form of an `"#"` character plus the `"name"`, like so:

`usemap="#dogmap"`

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

It causes no compatibility issues, and has excellent support across all tested browsers.

vspace *for <object>*

```
vspace="number"
```

SPEC		
deprecated	required	version
YES	NO	HTML 4
BROWSER SUPPORT			

IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The `vspace` attribute is set to `"30"` pixels around a left-aligned image object:

```
<object data="giant-dog.jpg" height="225" width="300" align="left"
    vspace="30"></object>
```

The `vspace` attribute is similar to the `hspace` (p. 367) and provides a bit of breathing space above and below an object, although it's not exactly a precision design tool, as Figure 7.15 shows.

Figure 7.15: Image object to which `vspace` of `"30"` is applied

Value

This attribute takes a number representing the amount of spacing to appear above and below the image in pixels.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

Although support for this attribute is good, it's now deprecated, and is highly presentational in its nature. The desired visual effects can all be achieved with CSS and, as such, this attribute shouldn't be used—it's presented here for informational purposes only.

"..." width *for <object>*

```
width=" { number | percentage } "
```

SPEC		
deprecated	required	version
NO	NO	HTML 4
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

Here, the width attribute is set to "200":

```
<object data="giant-dog.jpg" height="100" width="200">
</object>
```

An object doesn't require a width attribute but, like the height (p. 366) attribute, it has its uses (refer to the advantages set out in the height attribute reference).

If the width attribute is set by itself, but no height attribute is set, the object will be rescaled proportionally (assuming that object is a simple image).

In every other respect, the width attribute is identical in usage and behavior to the height attribute.

Value

This attribute takes as its value a number representing the width of the object in pixels.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

This attribute's compatibility is dependent on the type of object to which it's applied. For objects of `type` image, excellent cross-browser support is provided for the `height` attribute.

param

```
<param name="string" type="MIME type"
value="value"valuetype=" { data | object | ref } ">
```

SPEC		
deprecated	empty	version
NO	YES	HTML 3.2
BROWSER SUPPORT		
IE5.5+ FF1+ Saf1.3+ Op9.2+		
FULL FULL FULL FULL		

Example

Here's an example of an `object` (p. 355) element that contains two `param` elements:

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
    codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/
➥swflash.cab#version=9,0,28,0"
    width="320" height="285" title="Flash tester">
  <param name="movie" value="flash-test.swf"/>
  <param name="quality" value="high"/>
</object>
```

The `param` element is used in conjunction with the `applet` (p. 318) and `object` (p. 355) elements to provide parameters or variables to the parent element.

Use This For ...

A typical use for the `param` element is demonstrated in the example above, where it's used to pass to the object information regarding an embedded movie clip's quality and filename. When it's used with an `applet`, the `param` might be used to pass variables to a function that the applet uses, for example, instructing the applet to draw five polygons on the page, and to make them purple:

```
<applet code="draw.class">
  <param name="shape" value="triangle"/>
  <param name="amount" value="5"/>
  <param name="color" value="purple"/>
</applet>
```

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

Every browser listed supports this element type.

Other Relevant Stuff

<···> applet (p. 318)

specifies a Java applet (a mini application) for insertion into the document

<···> object (p. 355)

specifies a generic, multipurpose container for a media object

"..." name *for <param>*

SPEC		
deprecated	required	version
NO	YES	HTML 3.2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

```
name="string"
```

Example

The name of the parameter here is "shape":

```
<param name="shape" value="triangle"/>
```

This required attribute defines the data that's being passed to the object or applet in the parameter.

Value

The value of the `param`'s `name` attribute very much depends on the type of object that's being embedded into the web page. Whatever parameter name is being used, it's understood that the object is able to make sense of the information being passed to it. In the example above, the `param` is named `"shape"`, so the Java applet would presumably have an interface with a function that accepts various values, one of which is a `"shape"` variable. Therefore, it should understand that a shape is being passed in, and that it will need to act on the value passed to it in the `value` attribute.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

Every browser listed supports this attribute.

type *for <param>*

	SPEC	
deprecated	required	version
NO	NO	3.2

```
type="MIME type"
```

Example

This code shows a `param` element whose `type` is set to `"video/mp4"`:

```
<param type="video/mp4"/>
```

The `type` attribute lets the author define the MIME type[7] of the `applet` or `object` content. In many cases, though, this attribute is not required, as the browser can determine the MIME type on the basis of the URL or the header sent by the server for the embedded content.

---

[7]   http://reference.sitepoint.com/html/mime-types/

Value

This attribute takes a MIME type in the format type/subtype, for instance, `"text/html"`, or `"application/x-shockwave-flash"`, or `"video/mp4"`.

"..." **value** *for <param>*

`value="value"`

SPEC		
deprecated	required	version
NO	NO	HTML 3.2
BROWSER SUPPORT			
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The value of this `"shape"` parameter is `"triangle"`:

```
<param name="shape" value="triangle"/>
```

The `value` attribute works in conjunction with the `name` attribute (p. 377) to pass the necessary parameter or variable information to the parent `object` or `applet` element. According to the HTML specifications, the `value` attribute isn't a required attribute (only `name` is required), but it's rare to see the `name` attribute specified on its own.

Value

The value of the `value` attribute very much depends on the `name` of the parameter, so there's no fixed list of possible values.

Compatibility

Internet Explorer			Firefox			Safari			Opera	
5.5	6.0	7.0	1.0	1.5	2.0	1.3	2.0	3.0	9.2	9.5
Full	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

Every browser listed supports this attribute.

valuetype *for <param>*

```
valuetype=" { data | object | ref } "
```

SPEC		
deprecated	required	version
NO	NO	HTML 3.2
BROWSER SUPPORT		
IE5.5+	FF1+	Saf1.3+	Op9.2+
FULL	FULL	FULL	FULL

Example

The valuetype of the "shape" parameter here is "ref", as the value is a *ref*erence to a URL:

```
<param name="shape" value="poly.txt" valuetype="ref"/>
```

The valuetype provides a mechanism for defining exactly what *type* of data is being passed to the parent object. If a valuetype isn't specified, the browser assumes the default value of "data". You only really need to bring the valuetype attribute into play if the content inside the value is a URL which points to a file that contains the necessary information. In the example above, the reference is to a text file poly.txt which would, presumably, contain the set of rules or coordinates required to draw a polygonal shape, for which you'd specify a valuetype of "ref". If "object" is specified as the valuetype, the value attribute should correlate to another object on the page, which is referenced by that object's id:

```
<param name="handler" value="mplayer" valuetype="object"/>
   ⋮
<object id="mplayer">
    ⋮
</object>
```

Value

"data", "ref", and "object" are the only possible values for valuetype.

What's Next?

If like what you've seen in this chapter from *The Ultimate HTML Reference*, why not order yourself a copy?

This book is perfect anyone wanting an in-depth, accurate, and beautifully presented HTML reference at their fingertips. You'll not find a more up-to-date HTML reference that includes browser compatibility information, working examples and easy-to-read descriptions.

Covering the entire HTML language *The Ultimate HTML Reference* contains all the HTML knowledge you'll ever need.

Purchase Options

Depending on your preference, you've got the convenience, durability, and usability of a hard cover printed version, a transportable off-line version (PDF), as well as our freely accessible online version. Choose one or choose them all.

Order a copy today.

Appendix C

Alphabetic Element Index

This is a complete, alphabetical list of the HTML elements contained in this reference:

Alphabetic Element Index