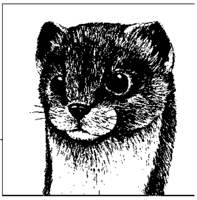


Text Elements

This chapter gets to the real meat and potatoes of document markup: elements used to structure text content. It's no surprise that nearly half of all the elements in the (X)HTML Recommendation are introduced in this chapter. The elements and discussions are organized as follows:

Block elements		**Generic elements**	
`h#`	Heading	`div`	Block division
`p`	Paragraph	`span`	Span of inline content
`pre`	Preformatted text	**Lists**	
`address`	Contact information	`ul`	Unordered list
`blockquote`	Lengthy quotation	`ol`	Ordered list
Inline elements		`li`	List item
`abbr`	Abbreviation	`dl`	Definition list
`acronym`	Acronym	`dt`	Term
`cite`	Citation or reference	`dd`	Definition
`code`	Code fragment	`menu`	Menu list
`dfn`	Defining term	`dir`	Directory
`em`	Emphasized text	**Presentational elements**	
`q`	Short inline quotation	`b`	Bold
`strong`	Strongly emphasized	`big`	Big text
`samp`	Sample output	`i`	Italic
`kbd`	Text entered by a user	`s`	Strike-through
`var`	Variable or program argument	`strike`	Strike-through
`sub`	Subscript	`tt`	Teletype
`sup`	Superscript	`u`	Underlined

Line breaks		**Presentational elements (*continued*)**	
`br`	Inserts line break	`font`	Font face, color, and size
Edit notation		`basefont`	Sets default font face
`ins`	Inserted text	`nobr`	No break
`del`	Deleted text	`wbr`	Word break
		`hr`	Horizontal rule

Choosing Text Elements

This chapter, jam-packed as it is with text elements, is a good opportunity for a reminder about the importance of well-structured and meaningful (semantic) markup.

In the early years of web design, it was common to choose elements based on their default formatting in the browser. Don't like the size of the `h1`? Hey, use an `h4` instead. Don't like bullets on your list? Make something list-like using `br` elements. Need indents? Blockquote it! Those days are over and gone.

Now we have Cascading Style Sheets (CSS) to visually format any element any way we like, at last liberating us from the browsers' default rendering styles. That means you must choose elements that accurately describe your content. If you don't like how it looks, change it with a style sheet. If you don't see an HTML element that fits, use a generic `div` or `span` element to add appropriate structure and meaning.

Additional tips on good markup are listed in Chapter 8.

A Word on Deprecated Elements

Many elements and attributes in this book are marked as "deprecated," which means they are being phased out of HTML and are discouraged from use. Most of the deprecated elements and attributes are presentational and have analogous style sheet properties that should be used instead. Others are simply obsolete or poorly supported.

The W3C needed a way to get the HTML specification back on track while acknowledging legacy browser capabilities and the authoring methods that catered to them. Rather than yanking them all at once, causing virtually every site in the world to be invalid, they put the deprecated elements and attributes in a "transitional" DTD that is available while browsers get up to speed with standards and web authors (and authoring tools) change their markup practices.

Now that style sheet alternatives to presentational HTML are widely supported, it is time to start phasing deprecated elements out of your documents as well.

The Building Blocks of Content

Text elements fall into two broad categories: inline and block. *Inline elements* occur in the flow of text and do not cause line breaks by default (they are covered later in this chapter). *Block-level elements*, on the other hand, have a default presentation that starts a new line and tends to stack up like blocks in the normal flow of the document. Block elements make up the main components of document structure.

Compared to inline elements, there are relatively few block elements. This section looks at heading levels, paragraphs, blockquotes, preformatted text, and addresses. Lists and list items are also block elements, and they are discussed later in this chapter, as is the generic `div` element used for defining custom block elements. The other block-level elements are tables and forms, which are treated in their own respective chapters.

Headings

Headings are used to introduce ideas or sections of text. (X)HTML defines six levels of headings, from `h1` to `h6`, in order from most to least important.

h1 through h6

`<hn>...</hn>`

Attributes

Core (`id`, `class`, `style`, `title`), *Internationalization, Events*

Deprecated attributes

`align="center|left|right"`

This example defines the element as a first-level heading.

```
<h1>Camp Sunny-Side Up</h1>
```

HTML syntax requires that headings appear in order (for example, an `h2` should not precede an `h1`) for proper document structure. Doing so not only improves accessibility, but aids in search engine optimization (information in higher heading levels is given more weight). Using heading levels consistently throughout a site—using `h1` for all article titles, for example—is also recommended.

Browsers generally render headings in bold text in decreasing size, but style rules may be applied to easily change their presentation.

Paragraphs

Paragraphs are the most rudimentary elements of a text document. They are indicated by the `p` element.

p

`<p>...</p>`

Attributes

Core (`id`, `class`, `style`, `title`), *Internationalization, Events*

Deprecated attributes

`align="center|left|right"`

Paragraphs may contain text and inline elements, but they may not contain other block elements, including other paragraphs. The following is an example of a paragraph marked up as a `p` element.

```
<p>Paragraphs are the most rudimentary elements of a text
document. They are indicated by the p element.</p>
```

Because paragraphs are block elements, they always start a new line. Most browsers also add margins above and below block elements. Text is formatted flush-left, ragged right for left-to-right reading languages (and flush-right for right-to-left reading languages). Style sheets may be used to override any default browser rendering.

HTML 4.01 allows the end `</p>` tag to be omitted, leaving user agents to parse the beginning of a new block element as the end of the previous paragraph. In XHTML, however, all elements must be terminated, and omitting end tags will cause the document to be invalid. For reasons of forward compatibility, it is recommended that you close paragraphs and all elements regardless of the markup language you are using.

Quotations (blockquote)

Use the `blockquote` element for lengthy quotations, particularly those that span several paragraphs and require line breaks.

blockquote

`<blockquote>...</blockquote>`

Attributes

Core (`id`, `class`, `style`, `title`), *Internationalization, Events*
`cite="`*`URL`*`"`

It is recommended that content within a blockquote be contained in other block-level elements, such as paragraphs, headings, lists, and so on, as shown in this markup example.

```
<blockquote cite="http://www.jenandtheneverendingstory.com">
<p>This is the beginning of a lengthy quotation (text continues...) </p>
<p>And it's still going on and on (text continues...) </p>
</blockquote>
```

The cite attribute is intended to be used to provide information about the source from which the quotation was borrowed, but it has very limited browser UI support (only Netscape 6+ as of this writing) and is not currently in common use.

The HTML specification recommends that blockquotes be displayed as indented text, which, in fact, they usually are. The blockquote element should not be used merely to achieve indents.

Preformatted Text

Preformatted text is used when it is necessary to preserve the whitespace in the source (character spaces and line breaks) when the document is displayed. This may be useful for code or poetry where spacing and alignment is important for meaning. Preformatted text is indicated with the pre element.

pre

```
<pre>...</pre>
```

Attributes

Core (id, class, style, title), *Internationalization, Events*

Deprecated Attributes

width="*number*"

Preformatted text is unique in that it displays exactly as it is typed in the HTML source code—including all line returns and multiple character spaces. Long lines of text stay intact and are not reflowed. The pre element in this example displays as shown in Figure 10-1. The second part of the figure shows the same content marked up as a p element for comparison.

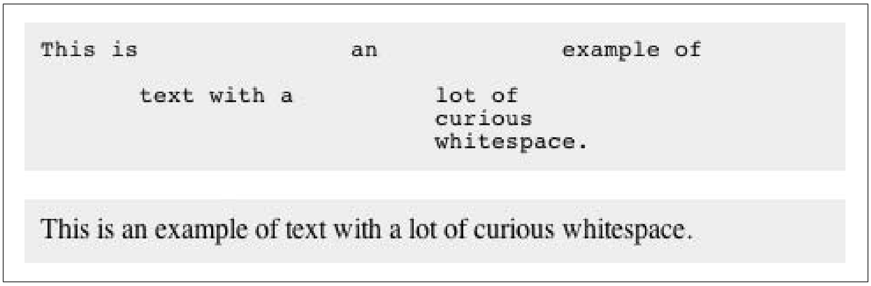
```
<pre>
This is               an             example of

       text with a          lot of
                            curious
                            whitespace.
</pre>

<p>
This is               an             example of

       text with a          lot of
                            curious
                            whitespace.
</p>
```

Preformatted text is meant to be displayed in a fixed-width font to preserve the alignment of columns of characters. Authors are discouraged from changing the font face and whitespace settings with style sheets. Preformatted elements may include any inline element with the exception of img, object, big, small, sub, sup, and font, all of which would disrupt the column alignment of the fixed-width font.



```
This is                 an              example of

      text with a          lot of
                           curious
                           whitespace.
```

This is an example of text with a lot of curious whitespace.

Figure 10-1. Preformatted text compared to a paragraph

Addresses

The address element is used to provide contact information for the author or maintainer of the document. It is not appropriate for all address listings. It is generally placed at the beginning or end of the document, or associated with a large section of content (such as a form).

address

`<address>...</address>`

Attributes

Core (id, class, style, title), *Internationalization, Events*

An address might be used as shown in this markup example.

```
<address>
Contributed by <a href="../authors/robbins/">Jennifer Robbins</a>,
<a href="http://www.oreilly.com/">O'Reilly Media</a>
</address>
```

Inline Elements

Most text elements are *inline* elements (spans of characters within the flow of text). Inline elements by default do not add line breaks or extra space.

This section introduces the semantic text elements that describe the enclosed text's meaning, context, or usage. These elements leave the specific rendering of the element to style sheets, either the author's or the browser's default rendering. There are other inline elements in the XHTML specification that are concerned with presentation (for example, the b element for bold text). They are briefly discussed at the end this chapter.

Phrase Elements

HTML 4.01 and XHTML 1.0 and 1.1 define a collection of *phrase* elements (also called *logical* elements) for adding structure and meaning to inline text. Because phrase elements share syntax and attributes, they are aggregated into one element listing here.

abbr, acronym, cite, code, dfn, em, kbd, samp, strong, var

`<abbr>...</abbr>, <acronym>...</acronym>, etc.`

Attributes

Core (`id`, `class`, `style`, `title`), *Internationalization, Events*

Phrase elements may contain other inline elements. The meaning and use of each element is listed here. When elements have a standardized presentation in browsers (for example, `em` elements universally display in an italic font), it is also noted. Authors are reminded, however, to choose elements based on meaning, not a desired rendering effect.

`em`
: Indicates emphasized text. `em` elements are nearly always rendered in italics.

`strong`
: Denotes strongly emphasized text. Strong elements are nearly always rendered in bold text.

`abbr`
: Indicates an abbreviated form.

`acronym`
: Indicates an acronym.

`cite`
: Denotes a citation: a reference to another document, especially books, magazines, articles, and so on. `cite`s are commonly rendered in italics.

`dfn`
: Indicates the defining instance or first occurrence of the enclosed term. It can be used to call attention to the introduction of special terms and phrases. Defining terms are often rendered in italics.

`code`
: Denotes a program code sample. By default, code is rendered in the browser's specified fixed-width font (usually Courier).

`kbd`
: Stands for "keyboard" and indicates text entered by the user. It may be useful for technical documents. Keyboard text is typically rendered in a fixed-width font.

`samp`
: Indicates sample output from programs, scripts, etc. It may be useful for technical documents. Sample text is usually rendered in a fixed-width font by default.

`var`
: Indicates the instance of a variable or program argument. This is another element that will be most useful for technical documents. Variables usually render in italics.

Indicating emphasis

The em and strong elements are used for indicating emphasis and even stronger emphasis, as demonstrated in this example.

```
<p>We <em>really</em> need to leave <strong>right now</strong>!</p>
```

Although emphasized text renders reliably in italics, it is not always an appropriate substitute for the i element. For example, if you want to italicize the title of a book, the cite element is the better choice. If there is no good match, create your own meaningful element using a generic span element and apply italics with the font-style style property. To use another example, it is a convention to display words from another language in italics, but that doesn't necessarily mean that those words are emphasized.

A good rule of thumb is to consider how your document would sound if it were read aloud (as it might be). Do you want the italic words to be read louder or at a different pitch from the rest of the sentence? If the answer is no, then it is probably best to find an alternative to em. The same logic applies to the strong element.

Acronyms and abbreviations

The abbr element indicates that text is an abbreviation: a shortened form of a word ending in a period, such as Mass., Inc., or etc. Acronyms (indicated with the acronym element) are abbreviations formed from the initial letters or groups of letters of words in a phrase, such as WWW and USA. An acronym may be pronounced as a word (NATO) or letter by letter (FBI).

The title attribute may be added to either element to provide the full name or longer form. The value of the title attribute may be displayed as a "tool tip" by visual browsers, or read aloud by a speech device.

```
<acronym title="National Aeronautics and Space Administration">NASA
   </acronym>
<abbr title="Tablespoons">Tbs.</abbr>
```

Marking up shorthand terms such as abbreviations and acronyms provides useful information on how they should be interpreted by user agents such as spellcheckers, aural devices, and search-engine indexers. It also improves the accessibility of the content.

> The CSS 2.1 specification provides the informative speak aural property that allows authors to specify whether an acronym should be read as a word or spoken letter by letter, as shown here:
>
> ```
> acronym#FBI {speak: spell-out;}
> ```
>
> The speak property is documented in Appendix B.

Short Quotations

HTML 4 introduced the q element for indicating short inline quotations, such as "To be, or not to be." Longer quotations should use the blockquote element listed earlier.

q

```
<q>...</q>
```

Attributes

Core (id, class, style, title), *Internationalization, Events*
cite="*url*"

The HTML Recommendation suggests that user agents should automatically insert quotation marks before and after q elements, therefore, authors are advised to omit them in the source. As of this writing, Internet Explorer 5 for Macintosh, Netscape 6, and Opera do insert generic double quotation marks, but Internet Explorer 6 for Windows does not.

```
As mother always said, <q>a guest is no one to criticize.</q>
```

Ideally, when used with the lang (language) attribute, the browser may insert language-specific quotation marks. Contextual quotation marks will be better handled with CSS-based generated text, as described in Chapter 23, once browser support improves.

The cite attribute is intended to provide a link to additional information about the source of the quote, but it is not well supported as of this writing. Netscape 6.1 makes the cite link available in a contextual menu accessed by right-clicking the quotation.

Deleted and Inserted Text

The ins and del elements are used to mark up changes to the text and indicate parts of a document that have been inserted or deleted (respectively). They may be useful for legal documents and any instance where edits need to be tracked.

As HTML elements, ins and del are unusual in that they may be used to indicate both block-level and inline elements. They may contain one or more words in a paragraph or one or more elements like paragraphs, lists, and tables. When ins and del are used as inline elements (as in within a p), they may not contain block-level elements because that violates the allowable content of the paragraph.

del, ins

```
<del>...</del>, <ins>...</ins>
```

Attributes

Core (id, class, style, title), *Internationalization, Events*
cite="*URL*"
datetime="*YYYY-MM-DDThh:mm:ssTZD*"

The following markup indicates that one name has been deleted and another one inserted in its place.

```
Chief Executive Officer: <del title="retired">Peter Pan</del> <ins>Pippi
Longstockings</ins>
```