CS105
Patrick Young
Stanford University

Appendix A: HTML Reference

Tag Index by Section

The tags are listed in one of nine sections as shown below:

Character Appearance Elements
, <b> Bold
<i> Italic, Emphasis
<q> Quote
<abbr> Abbreviation
<sub> Subscript, <sup> Superscript

Content Structure Elements
<p> Paragraph

 Line Break
<h1>, <h2>, … , <h6> Headings
<header>
<footer>
 Division
<hr /> Horizontal Rule
<!-- --> Comment

Section Elements
<section>
<article>
<nav>
<aside>

File Structure Elements
<!DOCTYPE>
<html>
<head>
<title>
<meta />
<style>
<link />
<body>

Linking
<a> Anchor

Image and Image Maps
<img /> Image
<map>
<area />

List Elements
 Ordered List
 Unordered List
 List Item
<dl> Definition List
<dt> Definition Term
<dd> Definition Description

Table Elements
<table>
<tr> Table Row
<td> Table Data
<th> Table Header
<caption>†

Form and Form Elements
<form>
<input />
<textarea>
<select>
<option>

Common Attributes

There are a number of attributes which may be used with any HTML element.

The `class` and `id` attributes provide support for style sheets. The `id` attribute also allows an element to act as a link destination.

`class=`*`class-identifier`*—The `class` attribute determines the class of the enclosed text for use with style sheets.

`id=`*`id`*—The `id` attribute can be used instead of the `class` attribute, if the element must be uniquely identified.

Here are some more global attributes you may find useful:

`title=`*`text`*—Most web browsers will display the value of the `text` as a tooltip if the user leaves the mouse cursor on top of the element for an extended time.

`lang=`*`language`*—The `lang` attribute tells the web browser what language a particular element is for. For example, we could create two elements, one with lang="en" for English and one with lang="de" for German. Depending on the user's language, we might make one element visible and the other invisible.

Text-Level Elements

The following elements are used to provide information on individual words or phrases.

<i> Italic, Emphasis

Categories: Flow Content, Phrasing Content

May Contain: Phrasing Content

Both i and em elements are typically rendered as italicized text by web browsers. The tag indicates that you want to emphasize a particular word or phrase. The <i> tag may be used for words in a foreign language, ship names, technical terms, or other words or phrases which are commonly italicized.

<b> Bold,

Categories: Flow Content, Phrasing Content

May Contain: Phrasing Content

Both the b and strong elements are typically rendered as bold text by web browsers. In general use of strong is preferred over b, as strong is more semantically focused whereas b is more presentation oriented and is therefore more appropriately handled at the style sheet level.

<q> Quote

Categories: Flow Content, Phrasing Content

May Contain: Phrasing Content

Use this element when you have a quote which is within a larger paragraph. If you want a quote alone by itself use the blockquote element.

<abbr> Abbreviation

Categories: Flow Content, Phrasing Content

May Contain: Phrasing Content

This element is used to represent abbreviations or acronyms.[1] It can be combined with a title attribute (see Common Attribute section above) to give readers a means of determining what the meaning of the abbreviation or acronym is.

<sub> Subscript, <sup> Superscript

Categories: Flow Content, Phrasing Content

May Contain: Phrasing Content

These elements are used to create subscript and superscript text.

Categories: Flow Content, Phrasing Content

May Contain: Phrasing Content

This element is used to group a word or phrase which does not fit in any of the previous text-level element categories. It is typically used in conjunction with a class or id attribute as a means of identifying something for which you wish to provide a style rule.

Content Structure Elements

<p> Paragraph

Categories: Flow Content

May Contain: Phrasing Content

HTML browsers ignore both carriage returns and blank lines in an HTML document. In order to break your document into paragraphs you need to use the <p> paragraph tag.

 Line Break

Categories: Flow Content, Phrasing Content

May Not Contain Elements

[1] HTML4 included a separate acronym element. This element was removed in HTML5 in favor of using the abbr element for both abbreviations and acronyms.

While p elements do have line breaks between them, logically not all line breaks represent paragraphs. The
 tag should be used to force a line break in these non-paragraph cases. There is no corresponding end tag, so we will end all
 tags with a " />".

<h1>, <h2>, … , <h6> Headings

Categories: Flow Content, Headings Content

May Contain: Phrasing Content

The h1 through h6 elements are used to produce headings on the web page. h1 should be used for the most important headers. h6 should be used for the least important headers.

You should have only a single h1 element on your webpage or within a section element (see below).

<header> *HTML5 only*

Categories: Flow Content

May Contain: Flow Content, but may not contain header or footer elements.

Use this to designate content which represents a header for a webpage or a section within a webpage.

<footer> *HTML5 only*

Categories: Flow Content

May Contain: Flow Content, but may not contain header or footer elements.

Use this to designate content which represents a footer for a webpage or a section within a webpage.

Division

Categories: Flow Content

May Contain: Flow Content

The div element is used to create sections within an HTML document if no other element type is appropriate. This element will typically include a class or id attribute to allow the section to be formatted in conjunction with a style sheet.

<hr /> Horizontal Rule

Categories: Flow Content

May Not Contain Elements

The <hr> tag creates a horizontal line or horizontal rule across a webpage. There is no corresponding end tag, so all <hr> tags should end with a " />".

<!-- --> Comment

Categories: None Ignored by Web Browser

An HTML comment allows us to enter text which can be read by other humans, but which is ignored by the computer. If we're writing a particularly complex set of HTML tags, comments allow us to describe in plain English what we're trying to accomplish and how we're accomplishing it, without having these comments show up in the actual webpage.

Any text between the starting <!-- and concluding --> is completely ignored by the browser. Text between these character sequences does *not* appear in the webpage, and any tags between them are ignored.

Section Elements

HTML5 provides several new elements that can be used to create sections within a larger webpage. Each section can include a full-range of headings (h1 through h6) and may contain its own header and footer.

<section> *HTML5 only*

Categories: Flow Content, Sectioning Content

May Contain: Flow Content, but may not contain header or footer elements.

The section element can be used to create any type of generic section within a wider document. Use this if none of the more specialized section elements listed below are appropriate.

<article> *HTML5 only*

Categories: Flow Content, Sectioning Content

May Contain: Flow Content, but may not contain header or footer elements.

Use to designate an article within a larger webpage. This could be used, for example, to designate a blog entry or a news article on a webpage displaying multiple articles.

<nav> *HTML5 only*

Categories: Flow Content, Sectioning Content

May Contain: Flow Content, but may not contain header or footer elements.

This element should be used to indicate that a section is specifically for navigation purposes only. The nav element, for example, would be appropriate for a navigation sidebar.

<aside> *HTML5 only*

Categories: Flow Content, Sectioning Content

May Contain: Flow Content, but may not contain header or footer elements.

Use this to designate content for sidebars, advertising, or other items which are parenthetical to the main webpage content.

File Structure Tags

The following tags serve to structure the HTML file into header and body sections. The tags in this section are introduced in Chapter 3.

<!DOCTYPE>

The <!DOCTYPE> or Document Type tag was originally used to tell the web browser which version of HTML the file is written in. For HTML5 files, the <!DOCTYPE> tag should look like this:

```
<!DOCTYPE html>
```

While the new HTML5 <!DOCTYPE> tag does not include a version number you must include it anyway. Many web browsers will not display a page properly if it does not contain a <!DOCTYPE> tag.

<html>

Categories: None.

May Contain: a head element and a body element

The <html> tag signifies that the enclosed text is HTML. All HTML files should begin with an <html> start tag and end with an </html> end tag.

<head>

Categories: None.

May Contain: Metadata Content

The <head> tag and its corresponding </head> end tag surround the head section at the beginning of the HTML document. Most HTML documents have a head section and a body section. The head section provides information that the browser may need to properly display the document. However, it does not contain any of the actual document text.

<title>

Categories: Metadata Content.

May Contain: Text only

The <title> tag and its corresponding </title> end tag surround the web page title. This title is used as the name of the browser window. The <title> tag should always appear as part of the <head> section of an HTML document. In XHTML a <title> is required.

<meta /> Meta-Information

Categories: Metadata Content.

May Not Contain Elements

The <meta> tag can be used to provide meta-information about a webpage or to provide HTTP header information. There is no corresponding end tag, so <meta> tags must end with a "/>".

The most important use of this tag is to provide the charset.

charset=*name*—The charset attribute tells the web browser what character encoding your webpage is using.

Other information provided by the <meta> tag is commonly used by web search engines. HTTP header information provided by the <meta> tag may be used to automatically redirect browsers to a different website or to cause regular reloads of a news website.

name=*name*—When the <meta> tag is used to provide general meta-information, the name attribute defines what kind of information will be specified in the content attribute.

http-equiv=*name*—When the <meta> tag is used to provide HTTP header information, the http-equiv attribute defines what kind of information will be specified in the content attribute.

content=*data*—The content attribute provides the actual information of the type specified by the name or http-equiv attribute.

<style> Style Sheet

Categories: Metadata Content.

May Contain: Style Rules only

The <style> tag is used to define a style sheet. It should be placed within the head of the HTML file. One or more style rules may be placed between the <style> start tag and the </style> end tag.

<link />

Categories: Metadata Content

May Not Contain Elements

The <link> tag is used to setup the relationship between two files on a webserver. The <link> tag should only appear in the document's head section. There is no corresponding end tag, so <link> tags must end with a "/>".

href=*URL*—The href attribute identifies an external file which is related to the current HTML file.

rel=*relationship*—This attribute determines the nature of the relationship between the files. Use stylesheet to link to a stylesheet.

type=*file-type*—This attribute specifies the type of the external file. For style sheets, set it to "text/css".

<body>

The <body> tag and its corresponding </body> end tag surround the actual document body in the HTML file.

Linking

The <a> anchor tag is introduced in Chapter 5.

<a> Anchor

Categories: Flow Content, Phrasing Content (but only if the a element itself only contains Phrasing Content), Interactive Content

May Contain: Determined by Parent Element.

The a anchor element creates a link to another document or resource.

href=*URL*—When a reader clicks on the text or images between the <a> tag and its associated ending tag, the browser finds the resource specified by the given URL and carries out the appropriate action. The most common URL is an HTTP reference, linking the current web page to another web page.

The a element's target attribute is used to support iframe's or popup windows.

target=*id*—The target attribute tells the browser to open the URL specified in the href attribute in a different window or in an iframe. If the target attribute specifies an id which is not the id of an existing iframe, the web page specified by the href will open in a new window. Future references to the same name will affect the newly opened window.

Image and Image Maps

The following tags are used to place images on the webpage and to create image maps. The <img> tag is first introduced in Chapter 5.

<img /> Image

Categories: Flow Content, Phrasing Content, Embedded Content

May Not Contain Elements

Images are inserted into HTML pages using the <img> tag. There is no corresponding end tag, so <img> tags must end with a "/>".

All <img> tags must have a src attribute and an alt attribute:

src=*URL*—The src source attribute tells the browser where to get the actual image to display. The value of the SRC attribute can either be an absolute reference or a relative reference.

`alt=`*`string`*—The `alt` attribute should be set to a text string describing the image. This string is used as an alternative for browsers which cannot display pictures. Some browsers also display the `alt` string while waiting for the image to download.

In addition to the required `src` and `alt` attributes, the following attributes may be used:

`height=`*`amount`*—The height in pixels of the image displayed.

`width=`*`amount-or-percent`*—Either the width in pixels of the image displayed, or the percent of the browser window width in which the image should be displayed.

Image maps are enabled by using the `usemap` attribute on the `<img>` tag.

`usemap=`*`URL`*—If an image tag includes the `usemap` attribute, the browser knows that the image is an image map. The value of the `usemap` attribute should be the name of a `<map>` tag (see above) preceded by a # sign. For example, if your file contains a `<map>` tag with the name `coolmap`, you could associate an image to that map by adding `usemap="#coolmap"` to your `<img>` tag.

Because the image map is linked, the web browser will place a blue link border around it. To eliminate this border, just set the image's border to zero.

<map>

Categories: Flow Content, Phrasing Content (but only if the a element itself only contains Phrasing Content)

May Contain: Determined by Parent Element.

The `map` element is used in conjunction with `area` elements to define a mapping between areas in an image and webpages to link to. The mapping is defined by the `area` elements (see below) found between the `<map>` start tag and `</map>` end tag.

The `<map>` tag has only one major attribute—`name`. The `name` attribute is required.

`name=`*`string`*—Set the value of the `name` attribute to a string which you will use as the value for the `usemap` attribute in the corresponding `<img>` tag.

<area />

Categories: Flow Content, Phrasing Content

May Not Contain Elements

The `area` element is used to define individual areas within an image and to link those areas to other URLs. There is no corresponding end tag—all the information needed to define an area is included as attributes of the `<area>` tag. `<area>` tags should end with a "`/>`". You can define areas as rectangles, circles, or polygons using the `shape` attribute.

`shape=`*`area-shape`*—The `shape` attribute can be set to `rect`, `circle`, or `poly`.

The `coords` attribute defines the location of the `area`. Its value is dependent on the value of the `shape` attribute.

`coords=`*`coordinates`*—If the `shape` attribute is set to `rect`, the `coords` attribute should specify the upper-left corner and lower-right corner of the area in the following order: X1, Y1, X2, Y2. All coordinates are relative to the upper-left hand corner of the image. So for example, if your rectangle started in the upper-left corner of the image (0,0) and continued to the 100th pixel in width and 200th in height, you would set `coords="0,0,100,200"`.

If your `shape` attribute is set to `circle`, the `coords` attribute should specify the x and y coordinates of the center of your circle and the radius of the circle. For example if you had a circle centered at (40,30) with radius 5, you would set `coords="40,30,5"`.

Finally, if your `shape` attribute is set to `poly`, the `coords` attribute should consist of the X, Y value pairs of all the points forming the polygon. For example, a triangle with vertices (20,20), (40,40), and (0,40) would have `coords="20,20,40,40,0,40"`.

The `href` attribute defines the action taken by the browser when the user clicks in the area. Typically it will be a reference to another web page.

`href=`*`URL`*—The `href` attribute can be set to any valid URL, just like the `href` attribute on an `<a>` anchor tag.

The `alt` attribute on an `<area>` tag is similar to the `alt` attribute on an `<img>` tag. This attribute describes the area for non-graphic web browsers. XHTML requires an `alt` attribute on all `<area>` tags.

`alt=`*`string`*—The `alt` attribute should be set to a text string describing the area. This string is used by browsers which cannot display pictures.

List Tags

The following tags are used to create lists..

 Ordered List

Categories: Flow Content

May Contain: li elements only

The `ol` element is used to create an ordered list. Place `li` elements for each list item between the `<ol>` start and `</ol>` end tags.

Browsers usually number items in an ordered list. Using style sheets you can designate that you'd like the list to use roman numerals or letters instead of numbers.

 Unordered List

Categories: Flow Content

May Contain: li elements only

The `ul` element is used to create an unordered list. Place `li` elements for each list item between the `<ul>` start and `</ul>` end tags.

Browsers will typically display items in the list using bullets '●'. If desired, you can control the symbol used to designate each list item using style sheets.

 List Item

Categories: None

May Contain: Flow Content

The `li` list item element is used to designate list items in both ordered and unordered lists.

`value=`*`integer`*—The `value` attribute can be determines the number associated with an item in an ol ordered list. It should always be set to an integer, even if the list is being displayed using letters or Roman numerals.

<dl> Definition List

Categories: Flow Content

May Contain: dd and dt elements only

The `dl` tag is used to create a definition list. A definition list is a list of terms along with their definitions.

<dt> Definition Term

Categories: None. May only appear in a dl element

May Contain: Phrasing Content

The `dt` element designates a term definition in a definition list.

<dd> Definition Description

Categories: None. May only appear in a dl element

May Contain: Flow Content

The `dd` element is used to designate the description for a corresponding `<dt>` term in a definition list.

Table Tags

The following tags are used to create HTML tables. The table tags are introduced in Chapter 3. Use of table tags for website layout is explored in Chapter 4.

<table>

Categories: Flow Content

May Contain: An optional caption element followed by table rows

All table data and formatting information should be enclosed between the `<table>` and `</table>` tags.

<tr> Table Row

Categories: None

May Contain: td or th elements

Each row in the table should be listed using the <tr> tag.

<td> Table Data

Category: Sectioning Root

May Content: Flow Content

Individual cells within a row are designated using the <td> table data tag or the <th> table header tag discussed below.

Any information between the <td> and either the <td> start and </td> end tag is placed inside of the table cell. While this information is usually text, it can be any legal HTML content. This includes, for example, both images and sub-tables.

<th> Table Header

Category: Sectioning Root

May Content: Flow Content

You may specify that certain cells in a table are table header cells rather than data cells by using the <th> tag. The <th> tag acts exactly the same as the <td> tag except that the contents of the <th> tag are displayed in bold and are centered, rather than left-aligned.

The <th> tag supports all the same attributes as the <td> tag shown above.

<caption>

Categories: None

May Contain: Flow Content, but may not contain Subtables

The <caption> tag can be used to add a caption for an entire table. The caption desired should be placed between the <caption> start and </caption> end tags. The caption tags themselves should be placed immediately after the <table> start tag.

Form and Form Elements

The following tags are used to create forms and form elements. These tags are introduced in Chapter 5. Chapter 13 discusses how to access these elements from JavaScript. Attributes supporting JavaScript events are discussed in Chapter 15.

<form>

Categories: Flow Content

May Contain: Flow Content, but may not contain a a subform.

The <form> tag is used to group together user-interface elements on a web page. In addition to the user-interface elements described below, you can put any other legal HTML inside of a form. The HTML <table> tag, for example, is often used to format the user-input elements contained in a form.

action=*URL*—This attribute determines the action taken by the webserver when the form's contents are submitted. Forms which are not working in conjunction with a webserver should skip this attribute.

<input />

Categories: Flow Content, Phrasing Content, Interactive Content

May Not Contain Elements

The <input> tag is used to create single-line text boxes, buttons, check boxes, and radio buttons. There is no corresponding end tag, so all <input> tags should end with a " />". All <input> tags must be contained within a <form>.

name=*string*—You can provide an input element with a name. The name is used when the form containing the <input> element is submitted. This attribute also makes access to the element easier from JavaScript.

type=*input-type*—The type attribute determines what kind of user-interface element is displayed. Legal types are text, password, checkbox, radio, button, submit, and reset.

value=*string*—The purpose of the value attribute varies depending on the setting of the type attribute. value provides the initial text-field contents for text fields. It provides the label for button, submit, and reset buttons. It can also be used to provide values for transmission to the webserver for both checkbox and radio input elements.

Additional attributes for text input items are:

size=*integer*—The size attribute determines the length of the text field in characters. This is actually just the number of characters which will be visible at any given time. The actual total number of characters which can be typed is determined by the maxlength attribute.

maxlength=*integer*—The maxlength attribute determines the maximum number of characters the user can type into the text field.

Both checkbox and radio input items also support the checked attribute:

checked—If the checked attribute is provided, the radio button or checkbox is initially marked as checked when the web page is first displayed. The checked attribute has no corresponding value so use checked="checked".

The <input> tag's event-related tags also depend on the type. <input> tags of type button, radio, checkbox, submit, or reset support the onclick attribute:

onclick=*script*—The script associated with the onclick attribute will execute when the user clicks on the input element.

<input> tags with type="text" or type="password" support the onchange attribute:

onchange=*script*—The script associated with the onchange attribute will execute when the user changes the text in the text field.

<textarea>

Categories: Flow Content, Phrasing Content, Interactive Content

May Contain: Text Only

While the <input> tag's text input element is used to create a single-line text input item, the <textarea> tag creates a multi-line text area. There is a corresponding required </textarea> end tag.

Attributes for <textarea> include:

name=*string*—As with the <input> tag, providing a name makes the textarea's data easier to access from JavaScript.

rows=*integer*—This attribute determines the number of rows of text visible in the <textarea>. This sets the height of the text area. *In XHTML this attribute is required.*

cols=*integer*—This attribute determines the number of columns of text visible in the <textarea>. This sets the width of the text area. *In XHTML this attribute is required.*

The rows and cols attributes only determine the amount of text displayed at once. The <textarea> provides scrollbar support in case the user decides to provide more information than will fit in the allotted space.

Like the <input> tag with type="text", the <textarea> has an onchange attribute which can be used for JavaScript programming.

onchange=*script*—The script associated with the onchange attribute will execute when the user changes the text in the text area.

<select>

Categories: Flow Content, Phrasing Content, Interactive Content

May Contain: option elements or optgroup elements

The <select> tag can be used to present a list of items for selection to the user. Items in the list are specified by placing <option> tags (described below) between the <select> tag and its corresponding </select> end tag. The list can be presented in a variety of ways depending on the attributes provided.

Attributes include:

name=*string*—Providing the <select> tag with a name makes it easier to access the selection from JavaScript.

`size=`*`integer`*—The value of the `size` attribute determines how many option items are displayed at once. If `size` is one, the select input element will show up as an in-place pull-down menu. Otherwise the select input element will show up as a scrollable list of options.

`multiple`—Providing this tag notifies the browser that the user may select more than one item from the options provided. If `multiple` is set, the select input element will always show up as a scrollable list. The `multiple` attribute has no corresponding value so use `multiple="multiple"`.

The `<select>` element includes an `onchange` attribute for use with JavaScript.

`onchange=`*`script`*—The script associated with the `onchange` attribute will execute when the user changes the current selection.

<option>

Categories: None

May Contain: Text Only

The `<option>` tag is used to provide options for a `<select>` user-input element.

`selected`—If `selected` is set, the option will show up as initially selected in the `<select>` user-input element. The `selected` attribute has no corresponding value so use `selected="selected"`.

`value=`*`string`*—The `value` attribute may be used to define the value sent to a webserver when the option is selected. If no `value` is provided, the text enclosed in the `<option>` tag will be used. This attribute may also be used in conjunction with JavaScript.