Tutorial 1: *Getting Started with HMTL5*

Logical and Physical Tags

What is the difference between *logical tags* and *physical tags*?

In HTML there are both *logical tags* and *physical tags*. The differences between logical and physical tags is one of the fundamental concepts in HTML that, when understood, can have a huge impact on a web designer's way of doing things.

Logical Tags:

Logical tags are structural tags that describe the nature and 'type' of the content they enclose. Logical tags represent the text's function on the page. They do not determine how the text will appear. The Web browser is free to set the presentation. The advantage of using the logical rather than the physical tags is that your meaning is more precisely conveyed.

Logical tags represent the structure and meaning of a document, with only suggested renderings for their appearance which may or may not be followed by various browsers under various system configurations. The way they're displayed is up to the browser, which means they're platform-independent. The original HTML specifications contained logical tags almost exclusively. HTML was originally a structural language, not a design and screen-layout language.

There are many logical tags, some of which are listed in the following table. Note that these are all two sided-tags.

Logical Tags		
Tag Name	**Tag Description**	**Your Web Browser Displays The Tag Like This**
	Strong Emphasis	**I am strong inside a tag**
	Emphasis	*I am emphasized inside a tag*
<abbr>	Abbreviation	I am an abbreviation inside a <abbr> tag
<cite>	Citation	*I am a citation inside a <cite> tag*
<code>	Code	I am programming code inside a <code> tag
<dfn>	Definition	*I am a definition inside a <dfn> tag*
<kbd>	Keyboard	I am quite like keyboard strokes inside a <kbd> tag
<samp>	Sample	I am a sample inside a <samp> tag
<var>	Programming Variable	*I am a programming variable inside a <var> tag*
	Neutral Inline Container	The tag is used to group inline-elements in a document.
	Neutral Block Element	The tag defines a division/section in a document
<h3>	Heading 4	**I am a fourth level heading in an <h4> tag**

An example of a logical tag is the tag. By placing text in between these tags you are telling the browser that the text has some greater importance. By default all browsers make the text appear bold when in between the and tags, but the point to take away from this is that the strong tag implies that importance of that text. The <b> tag does not tell the browser anything about the text except that is should display it in bold.

A second example of logical tag use is a text reader for the visually impaired may speak words inside of an tag louder or with a different tone of voice ("acoustic emphasis") than it would text inside of a <i> tag. A speech browser has no "italic font" and is therefore likely to ignore the <i> tag altogether.

A third example is that search engines like Google look for such logical tags to help figure out what the page is about. Physical tags simply do not describe the nature and 'type' of the content it encloses.

The importance of using logical markup reaches far beyond the problem of accessibility for the disabled; in fact, it is the best method to ensure the general accessibility, processibility, and, in the bottom line, longevity of your information. When the computer (not only your human reader) is aware of the logical structure of your data, the chances of its being able to successfully convert it into another format or media are very much improved. And accessibility implies, for the most part, exactly this: the ease of transforming the document into another media or presentation mode, in order to be able to communicate it to people with disabilities.

The fact that the mass-market browsers don't actually do anything with logical and physical tag distinctions like and <i> at the moment is no reason to let them slide; other software can still take advantage of them when indexing or abstracting your documents. Not to mention stylesheets. A site full of hard-coded physical markup will be much harder to convert to stylesheet use than a site with a clean, simple logical structure.

Try to use the logical constructs whenever they fit the meaning you're trying to convey. When you want a header at the top of a page, the <h1> tag is a good choice. The alternative of using a physical <font size="+3"> tag has the negative effect of not logically connoting a header; any program that attempts to create a structural outline of your document from its headers will be frustrated when you don't mark them as such. A text-mode browser like Lynx, and a reader program for the blind, may have a manner of its own to signal a header to the user, and this will be inoperative on your document when you signal that with a font change that's meaningless on these other devices.

Physical Tags:

Physical tags define how the text should be displayed in the browser. They control the Physical characteristics of the text.

The physical tags represent specific visual effects which are intended to be reproduced in a precise manner, and carry no connotation as to their semantic meaning. Physical tags exist only to create a specific visual effect. Physical tags don't provide meaning beyond physical appearance. As HTML was augmented by Web browser-makers Netscape and Microsoft, a large

number of physical tags were added, with very little augmentation of the logical tag set. That's why there are so many physical tags in existence today.

In a nutshell, physical tags were invented to add style to HTML pages because cascading style sheets were not around, though the original intention of HTML was to not have physical tags. Why? Well physical tags are just messy, tedious, and are more trouble than they're worth (for the most part). Rather than use physical tags to style your HTML pages, you should be using style sheets (also know as cascading style sheets or CSS).

There are many physical tags, some of which are listed in the following table. Note that these are all two sided-tags.

Physical Tags		
Tag Name	**Tag Description**	**Your Web Browser Displays The Tag Like This**
<i>	Italics	*I am in italics nside an <i> tag*
<b>	Bold	**I am bold inside a <b> tag**
<u>	Underline	I am underlined inside a <u> tag
<strike>	Strikethrough	~~I am in strikethrough inside a <strike> tag~~
<sup>	Superscript	I am superscript inside a <sup> tag
<sub>	Subscript	I am subscript inside a <sub> tag
<tt>	Keyboard	I am in typewriter form in a <tt> tag
<big>	Biger Font	I am bigger font inside a <big> tag
<small>	Smaller Font	I am bigger font inside a <small> tag
<s>	Strikethrough Alternative	~~I am a strikethrough alternative inside an <s> tag~~
<font>	Font Face	I am a inside an <font face="arial"> tag

Summary:

Logical tags, as mentioned above, have default ways in which browsers (like IE, Opera, or Fierefox) render them. But it is understood that style sheets, also called Cascading Style Sheets or CSS, should be used to give them their style, or in other words their 'look'.

The use of physical tags creates unnecessary limitations and problems for developers. To overcome these limitations and problems Web developers now use Cascading Style Sheets instead of physical tags to change the visual appearance of Web pages.

The idea is that logical tags should describe the content of the text and style sheets should modify the appearance of the text.

Physical tags are an old weak tool from the past. Cascading Style Sheets provide more formatting flexibility and power than any physical tag could possibly dream of.