

## Tutorial 1 notes: HTML5

*You will be prompted starting after #5 to try some practice using the handout called **Tag\_Practice Exercise**.*

### Table of Contents

- |  |                                      |
|--|--------------------------------------|
| 0. Background on xHTML and its rules     | 9. sectioning elements               |
| 1. 2-sided tags                          | 10. grouping and text-level elements |
| 2. 1-sided tags                          | 11. div and span                     |
| 3. attributes                            | 12. link to external files           |
| 4. whitespace                            | 13. special characters               |
| 5. evolution of basic parts of html page | 14. image element - img              |
| 6. deprecated defined                    | 15. lists                            |
| 7. meta element                          | 16. web hosting                      |
| 8. title element and comments            |                                      |

### 0. Background on xHTML and its rules

Our textbook (6e) uses the HTML5, version 5 of the HTML language. HTML5 is the version after xHTML. (And prior to xHTML, there was HTML4 and so on, all the way back to the original version, called HTML.) xHTML tightened up some rules that were being neglected and is also code that is compatible with the XML language. And we will strive to follow these rules as well for this course. Here is a review of the more important rules. Read over these rules now and reread them later when they will make more sense to you.

- Rule 1: *Type all code in lowercase. (Revisit after item 1 of these notes.)*
- Rule 2: *Tags must be closed. (Revisit after item 1 of these notes.)* Example: `<li>Apples</li>`. Include both the opening tag and the closing tag of the element. Here `<li>` is the opening tag and `</li>` is the closing tag
- Rule 3: Do not overlap tags in nested tags. *(Revisit after item 1.)*  
Follow: "First Open Last Close" – First tag opened is the last tag closed.  
Correct order:            `<b><i>hello</i></b>`  
Incorrect order:        `<b><i>hello</b></i>`
- Rule 4: *Empty tags must be closed. (Revisit after item 2.)* Example: `<br />` (br is the line break tag.) An empty tag is a single tag. (No pair.) In the HTML standard we wrote it as `<br>`, but now in xHTML we write `<br />`. That is, type "b"- "r"- spacebar - slash. (Note that HTML5 allows the use of either form.)
- Rule 5: *Enclose quotes around each attribute's value. (Revisit after item 3.)*  
And be sure to leave no space before and after the equal sign. In the example given in the book for this rule: **`title="my image"`**, title is an attribute and "my image" is its value.

### Tutorial 1 Notes: HTML5

1. **2-sided tags.** HTML, and its latest iteration HTML5, is a tag language. HTML stands for Hypertext Markup Language. It marks or identifies structures – or elements -- in the web page. It marks a paragraph, a list, a line break and so on.

All the HTML code is typed in lowercase. (In later chapters you deal with formatting and will use the CSS language strictly for that. Formatting includes adding colors, special fonts, layout of page elements, etc..)

You mark elements in the page with tags – usually 2-sided – with the form:

`<element>content</element>` where the word element is replaced with its specific element or tag name.

ex: `<p>This is a paragraph</p>` marks a paragraph, p element, p as in paragraph.

Type tags in lowercase. Tags can be **nested** within other tags. Below is an example. Note that `<i>` stands for italics.

ex: `<p>This is a <i>paragraph</i></p>`

(The tag `<i>` is an exception to the rule that a tag identifies an element or structure that appears in a page. There will be a few other exceptions too. You will learn that it is better to use `<em>` instead of `<i>`.)

2. **1-sided tag** (called **empty tag**). An empty tag is a singleton tag. As contrasted to most all tags, which occur in pairs, the **empty** tag occurs as one single tag. An Empty tag is *empty* of content. It surrounds no content. For example, `<br>` or alternately `<br />` is the line break tag.

In HTML5 you can write the line break either way but to meet xHTML standards it must be written `<br />`. Our textbook will use the `<br />` version. The xHTML tag is said to be “closed” -- by adding a space and a slash after the tag name. Other examples:

- `<hr />` horizontal rule (a separator bar).
- `` (“src=logo.jpg” is an attribute, additional information inside of the tag, and defined next.)

3. **Attributes** can be thought of as *embellishments* to the structure (element). Attributes *give you control over use, behavior or appearance of the element*. You can have more than one attribute in a tag. Attributes are placed inside the opening tag. Each attribute is set equal to a value. This is referred to as an attribute-value pair. As we learn each tag, we will learn some of its common attributes and corresponding values.

form: `<element attrib1="value1" attrib2="value2" ...>content</element>`

ex: `` - This `<img>` tag contains 3 attributes. The first one, src, inserts the image plant.jpg into the webpage and the other two resize the width and height of the image to 200px and 150px, respectively.

**Examples** of some attributes matched with some possible values they can have:

- align -- values possible are left, right and center
- color -- values possible are colors. Colors can be represented by their lcolor names, hexadecimal code or color triplet.

**Example:** `<hr align="center" color="#11fe77" />` `<hr />` is a horizontal rule,

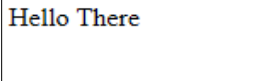
**Example:** `<h1 align="right">Text here</h1>` - h1 stands for the "heading 1"

4. **Whitespace** – is ignored by the browser. In other words, both:

```
<p>Hello      There </p>    and
```

```
<p>  Hello
      There
</p>
```

render in the web page as



## 5. Evolution of basic parts of html page

Three page templates appear below and they correspond to 3 versions of the HTML language. HTML was the original version in 1989, xHTML came out in 2001 and HTML5 in 2012.

### HTML

```
<html>
<head>
</head>
<body>
</body>
</html>
```

### xHTML

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>
<body>
</body>
</html>
```

this first line is optional

Other versions that might appear here are xHTML 1.1, HTML 4.01, etc.

Note how simplified HTML5 is compared to xHTML.

### HTML5

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
</body>
</html>
```

STOP HERE. Find the exercise called **tag\_practice\_steps** in a folder called *Tag\_Practice*, in the Activities Folder. Practice some coding in what you have learned so far. Read the directions and complete steps #1-3.

6. **Deprecated defined.** As the language of HTML of the 90's evolves over time to the present standard of HTML5, old language is phased out (is **deprecated**) and new language replaces it. The World Wide Web Consortium (W3C) guides the web technology including HTML language standards. The deprecated language will still work in older browsers and so therefore if you want your page to be supported by older browsers then you may still want to use the old language. But eventually the language will disappear over time. For example, the align attribute is deprecated. Other such code related to formatting (i.e. making the page look "pretty") are deprecated and we instead should use CSS. The tag attribute align is deprecated and it is preferred that you use the *style* called *align-text* with *center* as its value:

**Example:**

old: `<p align="center" >This is a centered paragraph.</p>`

new: `<p style="text-align: center">This is a centered paragraph.</p>`

(The *new* preferred way above uses a CSS style. CSS is the topic of Tutorial 2.)

7. **meta element.** meta means "about". Details about the page are found in the meta tag which is inserted into the head area of the page. One such detail is the character set that will be used by the web page. It is best to tell the browser, and not let it have to guess, what character set you are using.

Character sets are collections of characters. UTF-8 is a common one for web pages. (Another is Latin-1, also referred to as ISO 8859-1, for most Latin languages.)

Using previous language standard xHTML:

```
<meta http-equiv="Content-Type" content="text/html" charset="character set" />
```

In the newer HTML5 language standard:

```
<meta charset="character_set" />
```

Ex: `<meta charset="UTF-8" />`

Another example of the use of the `<meta>` tag aids the search done by a search engine to find the webpage:

```
<meta name="keywords" content="_, _, _" />
```

where the blanks would be filled in with keywords that describe the webpage or website. Other values for **name** are author and description which would have corresponding **content** values. See page 16.

8. The **title** and **comment elements**. The title element contains the title that you want to appear in the browser's title bar or tab area. The comment element contains any comments you want to include to help yourself and others reading the

code. Comments will NOT appear in the rendered web page. The title element must go in the head part of the page, but comments can go in either the head or body.

ex:

```
<html>

<head>
  <title>My Home Page</title>
</head>

<body>
  <!-- This is a comment. -->

</body>
</html>
```

STOP HERE. Find the exercise called **tag\_practice\_steps** in a folder called *Tag\_Practice*, in the Activities Folder. Practice some coding in what you have learned so far. Pick up where you left off and complete steps #4 and 5.

## 9. Sectioning elements: header, section, article, aside, footer and nav. Also, h1, h2, h3, h4, h5, and h6, and address

ex: <h1>h1 renders in 24 point with line of space above/below it.</h1>

ex of a page template using sectioning elements:

```
<!DOCTYPE html>
<html>

  <head>
  </head>

  <body>
    <header>
    </header>

    <section>
    </section>

    <article>
    </article>

    <aside>
    </aside>

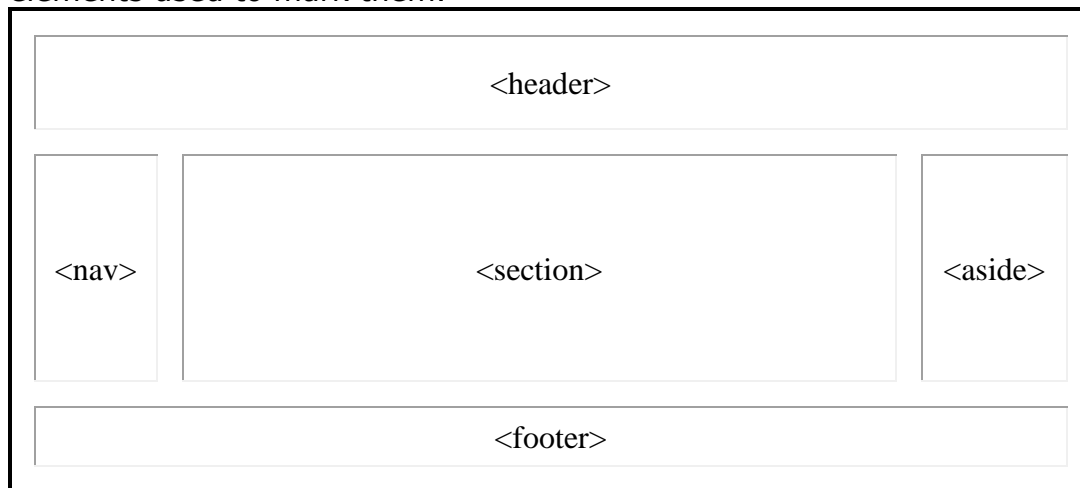
    <footer>
    </footer>
  </body>

</html>
```

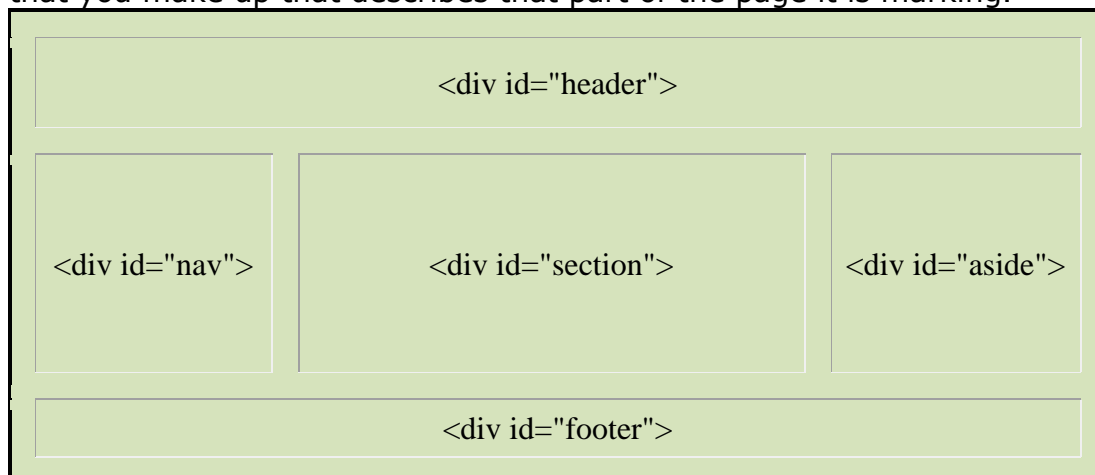
The header, section, article, aside and footer are new elements introduced in HTML5. These are called sectioning (or structural) elements. They identify page divisions. The header element identifies the content that will appear at the top of every page. The section element marks general content. The article element marks self-contained content. The footer element is similar to the header but appears at the bottom of each page. The aside element marks a sidebar.

The h1 through h6 elements mark headings in the page. h1 is for the main heading of a document. If you want to divide the main heading into subheadings you would next use the h2 element. For any further subsections, next use h3, then h4, etc. h1 has the larger font size and from h2 down to h6 they have successively smaller font sizes.

This diagram shows main sections of a webpage and the new HTML5 sectioning elements used to mark them.



This diagram shows how main sections of the web page used to be marked. Prior to HTML5, the parts of your web page were identified using the **div** element. In xHTML the generic div tag together with the id attribute was used to mark divisions or sections of your page. Note that the value of the id attribute is a descriptive name that you make up that describes that part of the page it is marking.



10. **Grouping** and **text-level elements**. There are 3 categories of tags: sectioning, grouping and text-level. Text-level elements should be contained inside of grouping elements, and grouping elements should be contained inside of sectioning elements. **Grouping elements** mark larger amount or block of text, and starts the text on a new line and **text-level elements** marks smaller portions of text. One could say that the text-level element is *inline with* the grouping element's content. For example, a text-level element might mark a word or phrase within a paragraph block. Text-level elements also include those that format text. `<b>` will bold the content, and `<em>` gives "emphasis" to a word(s) and will italicize it.

Some **grouping elements** are `<p>`, `<blockquote>`, `<ol>`, `<ul>`, `<div>`. See chart **p27** for more.

Exs:

- `<p>`p renders regular sized text with line of space above/below it.`</p>`
- `<blockquote>`Quote goes here, and will be indented`</blockquote>`

Some new ones in HTML5 are `<figure>` (which marks a figure/diagram), and `<figcaption>` (which marks a caption for a figure).

Some **text-level elements** are `<a>` (a for anchor -- for creating hyperlinks), `<b>`, `<i>`, `<strong>` (for strong emphasis, i.e. bold), `<em>` (for emphasis i.e. italics), `<sub>` (for subscript), `<sup>` (for superscript), `<span>`. A new HTML5 text-level element is `<mark>`, to mark marked text. See chart **p29** for more.

Ex: `<p><i>`Hamlet`</i>` is a play by William Shakespeare.`</p>` --Hamlet is marked with the `<i>` tag, a text-level element. It's *inside* of the grouping element `p`. Hamlet will render in italics.

A related note: You should favor the **logical and not the physical** interpretation of an element. For example, even though `<i>`, `<cite>`, `<address>` and `<em>` may all *physically* italicize what they mark, you should use the element that better fits logically to the context of the situation. If you mean to emphasize a word, use `<em>`, and if you are marking a citation, use `<cite>`. Thus you should use `<em>` and `<cite>` and not `<i>`.

11. **div** and **span**. The `<div></div>` marks a grouping element(s). `<span></span>` is a marker for a text-level element. You may then insert the id attribute into either one of their opening tag to later format or style that marked element. For example, if you insert the id attribute with the value "mysection" as follows, and surround a section of the web page content with `<div id="mysection"></div>`, then you can reference that section by its id "mysection" and format it, say, in a special way using CSS rules. Note that in `id="id_name"`, you make up the `id_name`.

We now have the new HTML5 sectioning elements (header, footer, section, article, etc.). So instead of surrounding what you had identified as a section of your webpage with `<div id="mysection"></div>` you should now use the newer HTML5

tag `<section></section>` and subsequently the section element can be referenced and formatted using CSS styles.

### Example in 3 parts of use of `<div>`

In the code below, `<div>` here is just being used to enclose the three text-level `<a>` tags as it is a grouping element. The `<div>` tag will not render its marked content in any different way – other than maybe start the block on a new line. It is a “generic” element.

```
<div>
  [ <a href="home.htm">Home</a> ]
  [ <a href="tips.htm">Tips</a> ]
  [ <a href="glossary.htm">Glossary</a> ]
</div>
```

In the code below, is the previous code from above but with the addition of the `id` attribute inserted into the `div` tag -- `id="links"`. This gives one the ability to later apply a CSS style specifying the `id="links"` which will style (format) in a special way the content surrounded by this `div` tag.

```
<div id= "links">
  [ <a href="home.htm">Home</a> ]
  [ <a href="tips.htm">Tips</a> ]
  [ <a href="glossary.htm">Glossary</a> ]
</div>
```

The code below is further improved in that it is using the HTML5 sectioning element `nav` instead -- as the `a` elements listed *are* intended to be a navigation bar of a website. You should use the HTML5 `<nav>` tag instead of a `<div>` tag such as `<div id="links">`. The CSS style will instead be applied to the `nav` element.

```
<nav>
  [ <a href="home.htm">Home</a> ]
  [ <a href="tips.htm">Tips</a> ]
  [ <a href="glossary.htm">Glossary</a> ]
</nav>
```

**Example** of use of the `<span>` tag, where `span` marks an inline (text-level) portion of text within the grouping tag `<p>`:

```
<p>Mary had a <span>little lamb</span>, with white fleece.</p>
```

## 12. Linking to external files.

**Linking to a stylesheet.** HTML is used to identify elements in the web page. CSS stylesheets is used to format or style the web page. CSS stands for Cascading Stylesheets.

There are 3 ways to style the page with CSS. Here we look briefly at one way -- linking the HTML page to an external stylesheet file with the use of the `link` element. You will learn how to create our own stylesheet in Tutorial #2, but for now just know that it contains the style “rules”. The file specifies elements and what styles will be applied to them. If you are given a stylesheet file named *filename.css*, insert the following link element into the head area of the page:

```
<link href="filename.css" rel="stylesheet" />
```



Note that the .css extension of the filename is necessary. Both the current html page and the css file need to be in the same folder, when the file is named as above.

**Linking to a JavaScript file.** Like the previous item, without yet knowing how to write a script (small program) in JavaScript, we can also link up our web page to an external JavaScript (.js) file. Again, there are other ways to insert JavaScript code but here we briefly look at one way – linking the HTML page to an external JavaScript file. JavaScript comes later in the course (Tutorial 9), but for now just know that the .js file contains a JavaScript program that will run in the browser, in the web page to which it is linked. If you are given a JavaScript file named *filename.js*, insert the following into the head area of the page:

```
<script src="filename.js"></script>
```

**13. Special characters.** Refer to demo\_characters.htm demo in the demo folder of html.01 folder if desired. (Recall, html01 is the name of the folder containing the data files for chapter 1, i.e. Tutorial 1.)

There are 2 ways to code a special symbol: with its numeric reference or its character reference. The numeric reference has the form: **&#code;** where code is its number. The character reference has the form: **&char;** where char is its name.

Some examples:

| symbol   | code | char name | Numeric ref       | char ref          |
|----------|------|-----------|-------------------|-------------------|
| ©        | 169  | copy      | <b>&amp;#169;</b> | <b>&amp;copy;</b> |
| ®        | 174  | reg       | <b>&amp;#174;</b> | <b>&amp;reg;</b>  |
| spacebar | 160  | nbsp      | <b>&amp;#160;</b> | <b>&amp;nbsp;</b> |

nbsp stands for "**n**on-**b**reaking **s**pacebar".

Notice the ampersand symbol (&) in front and the semicolon at the end for each expression. The numeric reference also has a # symbol.

STOP HERE. Find the exercise called **tag\_practice\_steps** in a folder called *Tag\_Practice*, in the Activities Folder. Practice some coding in what you have learned so far. Pick up where you left off and complete step #6.

**14. Image element img.** To insert – embed -- a picture into your web page, use the img element. If you are given a picture file named *filename.jpg*, the following tag will embed the picture into the page:

```

```

text is the alternate text that will appear if the browser does not display the picture, and x and y are the number of pixels for the picture's desired width and height, respectively. The "px" units may be dropped off. There are other file extensions for

picture files, such as .png or .gif. The img element is placed "in-line" with the surrounding content, just like a text-level element.

**15. Lists.** A list is a grouping element as it is a distinct block. You should be familiar with the tag names and attributes for the three types of lists:

- (a) `<ol></ol>` tags surround the entire ordered list ("ol" stands for ordered list), and the `<li></li>` tags surround each line ("li" stands for line).
- (b) `<ul></ul>` tags surround the entire unordered list ("ul" stands for unordered list) and `<li></li>` tags surround each line ("li" stands for line). unordered = bulleted list
- (c) A description (or definition) list uses the `<dl>`, `<dt>` and `<dd>` tags. ("dl" stands for definition list, "dt" stands for definition term and "dd" stands for definition.)

**EXAMPLES:**

| ordered list       | Unordered list   | Description (definition) list  |
|--------------------|--|--|
| 1. green<br>2. red | <ul style="list-style-type: none"> <li>• green</li> <li>• red</li> </ul> | Network<br>Computers or devices connected together<br>Parser<br>Interprets the code and renders the page |

Code ordered list:

```
<ol>
  <li>green</li>
  <li>red</li>
</ol>
```

Code for unordered list:

```
<ul>
  <li>green</li>
  <li>red</li>
</ul>
```

Code for description list:

```
<dl>
  <dt>network</dt>
  <dd>computers and devices connected together</dd>
  <dt>parser</dt>
  <dd> Interprets the code and renders the page </dd>
</dl>
```

**16. Web hosting.** You can create your web pages locally on your computer. When you are ready you post (upload) your web pages to a Web server. Consider such things as cost and amount of space on the Web server when choosing a web host. ISP (Internet Service Providers) may times offer you web hosting with domain name, for example, [www.verizon.com/your\\_website\\_name](http://www.verizon.com/your_website_name).

You will also need a **domain name**. Google "domain names" to find a company that sells them. You can do a search at one of these sites to see if the domain name you desire is available. Usual pattern: [your\\_company\\_name.com](http://your_company_name.com) (or .biz or .net)