Reading Notes:

HTML Introduction

Switzerland was first to create “HTML", or "Hyper-Text Markup Language

-HTML provide a language for displaying text and pictures SIMULTANEOUSLY

-also devised the notion of "hyper-text", clickable words which could be used to summon OTHER digital resources, such as HTML pages, pictures, and sound files

What is HTML?

HTML (Hyper-Text Markup Language) is a display language, a text-only code which tells a web browser (such as Internet Explorer or Netscape Communicator) how to assemble a web page of text, pictures, and other multimedia content.

HTML pages are stored as ASCII (text-only) files on a web server. Pictures, stored as GIF or JPEG files, are also stored on a web server. When an HTML page is called up in a web browser, the HTML code tells the browser where on the web server the pictures are located, and where on the page those pictures are supposed to be displayed.

HTML formats text (and everything else, as well) using something called a TAG

<tagname>Content</tagname>

Not all tags have to open and close in this manner, surrounding content. Some tags simply insert formatting information into the flow of HTML text. For instance, in the BR (break) tag inserts a carriage return or line break into text without requiring a closing tag, as in the following example:

<p>Here is a paragraph of text.<br /> Notice how the line break occurs just there?</p>

XHTML (ie. HTML 5)

 Extensible Markup Language, or XML

 The primary motivation for creating XML was the need to send data over the Internet in a universal, structured format.

 Because this new version of HTML would not be based on earlier versions of HTML, but on XML, it was named Extensible Hypertext Markup Language, or XHTML.

-follows strict rules: case-sensitive, file extension should be html

XHTML Elements and Rules

At the very beginning of an XHTML document, you should place the following:

//indicates to the browser the version of XML used for the document’s code  
<?xml version="1.0" encoding="UTF-8"?>

//Next two lines make up the DTD (Document Type Definition) – what type of document this is   
<!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">

xmlns stands for XML namespace, which helps the browser keep track of what all the custom XML tags mean, especially if the same name is used for different tags. A namespace lessens the confusion by providing the browser with information on the meaning of the tags in the document.

we are telling the browser that all tags contained within the <html> element belong to the XHTML namespace, as defined by the W3C and located at the given URL

 XHTML documents must be well-formed

<?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>

NOTE: Exception – frameset page

 XHTML documents must have a <title> element within the header section

<head>  
<title>Steve Rubin's Homepage</title>  
</head>

 Elements must nest and not overlap

<p>The following word is written in italics: <i>italics</i></p> //GOOD

<p>The following word is written in italics: <i>italics</p></i> //BAD p vs i

 All element and attribute names must be in lowercase

In XHTML, although you must write attribute names in lowercase, you can write attribute values in either upper or lowercase. For example, the following is valid in XHTML, even though I don't recommend it.  
  
<body bgcolor="YELLOW">

 All tags must close

In XHTML, you must terminate empty tags such as the <br> tag with a space followed by a forward slash ("/"). For Example: <br />  
  
This requirement to close empty tags also extends to all other tags, which would not normally close in HTML. For example:  
  
<img src="xyz.gif" />  
<meta name="keywords" content="Steve Rubin, web design, critical thinking" />  
<bgsound="xyz.wav" />

 All attributes must have values enclosed within quotes

//GOOD

<body bgcolor="blue">  
<img src="xyz.gif" width="200" height="300" />  
<map name="europe">

//BAD

<body bgcolor=blue>  
<img src=xyz.gif width=200 height=300>  
<map name=europe>

//MORE

<input type="checkbox" name="choice" value="europe" checked>  
  
The above example is incorrect in XHTML. In XHTML, it would be correctly coded as follows:  
  
<input type="checkbox" name="choice" value="europe" checked="checked" />

XHTML Validation

**The Main Reasons are:**

1. Validation helps to ensure your web page looks right, no matter what web browser is being used.
2. Search engines can view and understand the page content. Some errors may prevent your content from being read by the search engines.
3. Pages are more accessible to people with disabilities.

### HTML5:

HTML5 is a standard for structuring and presenting content on the World Wide Web. The new standard incorporates features like video playback and drag-and-drop that have been previously dependent on third-party browser plug-ins such as Adobe Flash, Microsoft Silverlight, andGoogle Gears.

**HTML5 basic format**

<!DOCTYPE html>  
<html lang="en">  
<head>  
<meta charset="utf-8">  
<title></title>  
</head>  
  
<body>  
  
<header>  
</header>  
  
<section>  
</section>  
  
<footer>  
</footer>  
  
</body>  
  
</html>

HTML5 has introduced new tags that actually have real meaning and purpose, much more semantic than simply using divs all over the place.

VIDEO: 10 Things every web dev should know

-semantics are better (more meaningful)

-additional features added

-backwards compatible (you can covert old HTML version to 5 quickly)

-UTF-8 (required for validation) //under meta

-no need for type attribute

Additions to markup:  
-new elements that have better readability (old still work)

-better semantics and structure

-will make IDE’s work better

JavaScript (power behind HTML5)

-geo-location – location awareness for web-apps (user’s coordinates)

-canvas api – 2D drawing surface you can add to your page and add graphics to

without any additional plugins

How does it work with web-pages?

-tons new features that allow pages to behave more like apps

-official language of HTML5

-JavaScript has good performance as it has been improved over the years

Web Storage

-5mbytes

-user can create concurrent threads on web application

3. Should we be using HTML5 now?

-If you have a lot of users using mobile devices that obtain your content then yes.

-If you have a lot of IE users then no (until people move on to new browsers)

4. Care about which browsers implement and API not which specifications it belongs to

-Focus on if browser supports API

NOTE: API – application programming interface (protocol used as an interface by software components to communicate with each other)

NOTE: DOM – Document Object Model; cross-platform and language-independent convenstion for representing and interacting with objects in HTML, XHTML, and XML documents

5. <video> tag

-allows you to put videos right into your page

-add additional controls while video is not playing (etc.. attributes)

-problem: what format of video to use? (encoding world)

.mp4 (iOS)

.ogv

.webm

OR also supply various source tags to allow browser to check what they can or can’t use

6. JSON – King of Content

-used as an alternative to XML to serialize data

-Java Script Object Notation

-takes an object written in JavaScript and turns it into a String

-good because it is more readable than XML and easier to parse

-2 Methods: Stringafy and Parse

-used by many WebServices that use this

-advantage – easy to use and convert / seralizable / unseralizable

-disadvantages: be careful with remote services to not get a malicious object

7. Make it easy on yourself with graceful degradation and feature detection

-largely deals with API

-feature detection – does geolocation exist? do webworkers exist?

-solution: get Modernizr (clean, consistent interface, detect CSS3 features)

8. <canvas>

-more than just for drawing

-alter video source (ie make it black and white)

-JavaScript grabs frames from video and throws it into a canvas then does editing

-use with double buffering techniques

-general purpose display

9. CSS3 supports new features

-CSS replaces common things done with JS

-ie. transition – opacity occurs gradually (fade effect)

-ie. transform – take an element and do something with it (rotate it, scale it)

10. XHTML is dead

-why? people did not like it

-needed to evolve HTML

-However, this is not the end of XHTML 🡪 XHTML5

Why? It does allow you to validate documents and plays well on XML environments (where you need an well structured document)

NOTE: Graphics and Games – SVG (xhtml base extension of the language) – scalable vector graphics

**6 Essential HTML Page Tags**

required tags, their placement, and their usage.

1<html>

2<head>

3<title> </title> \*IMPORTANT\* (most important because search engines use the words in the TITLE as keywords for categorization of your web page in their databases)

HEAD of an HTML page contains mostly invisible, meta, or scripting information.

</head>

4<body>

BODY of an HTML page contains ALL visible content; this includes all text, pictures, headings, lists, animations, sound files, and other multimedia elements.

</body>

</html>

 NO VISIBLE CONTENT may be placed in the HEAD of an HTML page! If it's visible, it goes into the BODY!

**7 Saving and Naming your HTML File**

 must create a file name for your HTML page. This name should always end with the file extension ".htm" or ".html" to indicate that it is an HTML page.

There are four rules governing the naming of HTML pages:

1. NO SPACES IN THE NAMES!
2. Always end the file name with ".htm" or ".html"
3. Only use characters from a-z, A-Z, 0-9, and \_ (underscore). All other characters (including commas, periods (except for the "dot" at the beginning of the dot-extension), tildes, exclamation points, etc) are considered reserved, and are therefore illegal.
4. Always start the file name with a lower-case letter; no numbers or underscore characters are allowed as the FIRST character of the name (although you may use them anywhere else in the name).

**8 Case-Sensitivity and File Names**

myPage.htm" is NOT the same as "myPage.html

URLs are CASE-SENSITIVE

**9 White Space and Other Technical Issues**

Specific HTML coding practices, rules, issues

White Space

Ordinary carriage returns are considered to be "white space". White space is a traditional programming term, used to describe anything which generates blank areas in code. White space includes spaces, carriage returns, tabs, non-breaking-space characters, and anything else which is space-like, carriage-return-like, or tab-like. White space has traditionally been used to break up code into clearer, more easily proof-readable constructions.

-it is used to clarify structure in code

-html does NOT recognize ordinary carriage returns as line breaks for text

-Anywhere that I want a line break I must insert BR tag

<br/> for XHTML

HTML5 is a case-insensitive language, which means that you may use either upper-case or lower-case characters in tag names.

For instance, **<br>** is the same as **<BR>** as far as HTML5 is concerned.

XHTML is a case-sensitive language, with all lower-case tag syntax. In XHTML, all tags must have closing tags.

In XHTML, there is only one acceptable way to code the BR tag: **<br />**

**10 Tag Nesting and the Parent/Child Relationship**

This nesting indicates something called a "parent-child relationship" between the two tags

<p>Here is an <i>italicized</i> word nested within a paragraph.</p>

SIBLINGS - nested tags under the same tag (same parent)

**11 Formatted Lists**

-A series of bullet points on some topic, or a list of steps or tasks to perform, or a sequence of definitions in the dictionary.

NOTE: comments - <!-- end nested list -->

Unordered

Tag: **UL**  
Attribute: **TYPE**  
Values: **disc**, **circle**, **square**

– bullet points (most common)

- UL stands for "Unordered List", while LI stands for "Line Item".

- UL has an attribute: TYPE – used to specify which type of bullets the list will display (3 types: reg, hollow circle, hollow square)

Ordered

Tag: **OL**  
Attribute: **TYPE**  
Values: **1** (default), **A, a, I, I //note: type=”1” is same as default #s**

**A – capital letters, a is lowercase leters, I is uppercase roman numerals,**

**i is lower-case roman numerals**

– numbers / letters

- Ordered Lists are created using the OL and LI tags

You may make your lists start on ANY number (not just 1, A, a, I, or i) by using the START attribute.

Tag: **OL**  
Attribute: **START**  
Value: any number

<ol start="4">

<ol type="I" start="10"> 🡪 starts at X

<ol type="A" start="6"> 🡪 starts at F

Definition – name of word, next line def.

If you wish to support IE5 for Mac, you must NEVER use nested ordered lists. Internet Explorer 5 for Windows does NOT have this problem.

**14 Resources and URLs**

-every digital thing is considered to be a resource

-HTML or text files are resources, pictures are resources, sound files are resources, Flash animation files are resources; any digital media file is a resource. Resource is a catch-all term for digital media or files of any kind.

-The HTML page invokes (or calls up) its various resources using something called a URL

-There are two kinds of URLs: absolute and relative.

-Absolute URLs point to a specific location on the entire Internet. Relative URLs point to a location on a web server which is in relation to the current HTML page.

**15 Absolute URLs**

An absolute URL represents a unique address on the Internet for any piece of digital media (resource).

Absolute URL syntax involves two parts, described in some textbooks as "scheme" and "scheme\_specific\_part", which must be separated by a colon and described in this manner (NO SPACES):

scheme:scheme\_specific\_part

schemes:

**http** (Web address)  
**mailto** (email address)

**ftp** (ftp site address for transfer of files)  
**file** (file address on a hard disk or server)

www.yahoo.com

"com" is the domain name.  
"yahoo" is the sub-domain name (which is commonly called the domain name).  
"www" is the name of the server in the "yahoo.com" domain.

web addresses always begin with the following code:

http://

http://www.yahoo.com/

"http" is the "scheme".  
":" separates the "scheme" from the "scheme\_specific\_part".  
"//" is the required prefix for an absolute web address.  
"www.yahoo.com" represents the server, the sub-domain, and the domain name.

mailto:srubin45@comcast.net

**16 The A (anchor) Tag, Part 1**

The A (anchor) tag allows you to create hyper-text references on an HTML page. Hyper-text, hyper-reference, hot-link, these terms all mean the same thing: a clickable link on a page (usually a word or a picture) which invokes another HTML resource.

Tag: **A**  
Attribute: **HREF**  
Description: The anchor tag allows you to create an anchor point in the flow of HTML text. The HREF attribute assigns a URL to an anchor tag and makes an active hyper-reference out of the word(s) or picture marked by that tag.  
Example:  
**<a href="http://www.yahoo.com/">Link Words</a>**

**17 IMG (image) Tag Basics**

A (anchor) tags are not restricted to invoking only HTML resources.

<a href="http://fog.ccsf.org/~srubin/satthumb.gif">View of Saturn</a>

even though we can invoke an image resource using the A (anchor) tag, we can't INSERT an image into an HTML page this way; we can only create a hot-link or hyper-reference TO an image. To actually insert an image/picture into the flow of an HTML page, we need to use the IMG (image) tag.

Tag: **IMG**  
Attributes: **SRC**, **WIDTH**, **HEIGHT**, **ALT, TITLE**

ALT attribute will allow it's value to pop-up when the user hovers over it, in most browsers except Firefox.

TITLE attribute will pop-up it's value when the user hovers over it in Firefox. I suggest always using both attributes for the IMG tag.

html5

<img src="http://fog.ccsf.org/~srubin/satthumb.gif" width="170" height="129" alt="saturn" title="saturn">

xhtml

<img src="http://fog.ccsf.org/~srubin/satthumb.gif" width="170" height="129" alt="saturn" title="saturn" />

**18 The A (anchor) Tag, Part 2**

**19 Relative URLs**

A relative URL is essentially a set of directions to a resource from your current location

These paths require special prefixes in order to operate correctly, as well as to maneuver through the server's directory structure.

There are three prefixes for relative URLs:

#### Prefix One:

**./**

Conservative syntax for plain relative URL; it means "look for the resource in the same directory."

(NOTE: In most cases the ./ can be omitted and so the above example is virtually the same as)

#### Prefix Two:

**../**

This means: "go up one directory level and start looking for the resource."

#### Prefix Three:

**/**

This means: "go to the root directory for the web server and start looking for the resource." A root directory for a web server is the single folder which has been assigned as the "public" web folder; any resource inside this "root directory" may be served to the outside world via the Web; no resources OUTSIDE this root directory may be served via the Web.

**20 The A (anchor) Tag and the NAME Attribute**

A (anchor) tags can also allow you to create hyper-references that jump to individual points WITHIN a particular HTML page.

you need two A (anchor) tags: One A (anchor) tag to mark the place that you want to jump to, and one regular A (anchor) tag to make the hyper-reference.

These "jump-to" points are called "NAME anchors", and are created using the NAME attribute in the A (anchor) tag.

Tag: **A**  
Attribute: **NAME**  
Description: The NAME attribute turns an A (anchor) tag into a "jump-to" point, which can be referenced by other A (anchor) tags used as hyper-references/hot-links.  
Example: **<a name="fred">Word</a>**

**Note:** Each NAME on an HTML page MUST be unique (no repeats).

"jump-to" points; they are NOT clickable hyper-references, nor do they alter the appearance of marked content.

Once marked in the HTML code, any hyper-reference may jump directly to a NAME-anchored point by calling that name as the URL, preceded by the pound (#) prefix. For instance, "**#fred**" would be an appropriate URL for a hyper-reference jumping to the "**fred**" NAME anchor elsewhere on the same page.

<a href="#fred">Link that jumps to "fred" NAME anchor</a>

<a href="#">This link will jump to the top of the page</a> //jump to top

You may also jump to NAME anchors that exist on OTHER HTML pages, using either absolute or relative URL syntax. Simply take the absolute or relative URL, and append the pound (#) sign, followed by the name of the NAME anchor that you want to jump to, to the end of the URL.

Example:

<a href="./page2.html#bill">Link to "bill" NAME anchor on page2.html</a>

**21 Heading Tags**

Tags: **H1, H2, H3, H4, H5, H6**  
Description: Creates headings and sub-headings for a document. Always closes.

Example:

<h2>Important Stuff</h2>

H1 has the largest font size, H6 has the smallest font size.

Heading tags automatically make text bold, and align to the left of an HTML page, just as paragraphs do. Heading tags automatically create line spacing following themselves; this following space grows smaller as the H tag numbers grow larger. Heading tags don't require extra BR tags to set themselves off from surrounding materials; they do this automatically as well. Heading tags ALWAYS close.

**22 Attributes**

We can use attributes to change the BEHAVIOR of a tag,

We can also use attributes to change the APPEARANCE of a tag.

Tag: **BODY**  
Attribute: **BGCOLOR**  
Values: color values (named colors or hex codes)  
Description: The BGCOLOR attribute in the BODY tag can be used to set the background color for the BODY of an HTML page.

Tag: **BODY**  
Attribute: **TEXT**  
Values: color values (named colors or hex codes)  
Description: The TEXT attribute in the BODY tag can be used to set the text color for all content in the BODY of an HTML page.

attributename="value"

<body bgcolor="black" text="yellow">

You may add as many attributes to a tag as you desire; there is no limit. However, you may NOT repeat the same attribute twice in the same tag!

**23 Coding Style and Tag Syntax**

**24 Comment Tag**

In HTML, a comment is just another tag, although it follows special rules. Here is what an HTML comment looks like in code:

<!-- Here is a comment -->

Incorrect (multiple lines):

<!-- Here is

a comment -->

Correct:

<!--

Here is a comment

Here is another comment

-->

//Note: particularly useful for tables

**25 Special Characters**

Certain characters in HTML are reserved, such as the less-than (**<**) and greater-than (**>**) signs, and the ampersand (**&**) character. If you type characters like these directly into HTML code, you will usually (although not always) get either a broken web page, or gibberish characters on the screen.

you must use character entity references, also known as escape sequences or escape characters.

The syntax for escape characters in HTML is always the same: The ampersand (**&**) character, followed by the character code, and ending with a semi-colon (**;**).

The character codes may be represented either by name (named entity) or by number (numeric entity); named entities are special text abbreviations of the character name, while numeric entities are special number codes representing the character preceded by a pound (**#**) sign.

Examples:

Character: **<** (less-than sign)  
Escape Character: **&lt;** or **&#060;**

**26 PRE Tag**

Tag: **PRE**  
Description: Used to set pre-formatted text. Always closes.

The PRE tag always displays content using the browser-default monospace font (usually Courier 10pt). BR tags are not required for line breaks within the PRE tag; ordinary carriage returns are recognized as line breaks, making the BR tag unnecessary. The PRE tag also respects space characters, displaying all spaces individually, rather than compressing a group of spaces together and displaying them as a single space.

Example (supported/standards-compliant -- use this):

<p>Here is a paragraph of text.</p>

<pre>

</pre>

<p>Here is another paragraph of text.</p>

Displayed:

Here is a paragraph of text.

Here is another paragraph of text.

**27 Physical Style Tags and Content-Based Style Tags**

There are a group of tags, used to mark-up text styles, which split into two types: physical and content-based style tags.

Physical style tags include B (bold), I (italic), SUP (superscript), SUB (subscript), and many more; these tags act physically on text exactly as they claim. Content-based style tags include STRONG (strong), EM (emphasis), CITE (cite/citation), etc; these tags represent what something MEANS, rather than what something looks like.

Whereas physical style tags impart a specific appearance to text that they mark-up, content-based style tags impart information regarding what that text is supposed to BE, or what it is used for. The fact that content-based style tags impart appearance characteristics to text is completely secondary.

They ALL must close.

Tag: **TT**Description: Makes text "teletype", aka the default monospace font (ordinarily, Courier 10pt, but this is dependant entirely on the user's browser settings).

Example:

<p>Mr. CodeHead says: <tt>var myGoober = 10 \* myRadius;</tt></p>

Displayed:

Mr. CodeHead says: var myGoober = 10 \* myRadius;

Content-based style tags are more concerned with what a particular piece of text means or represents, rather than what it looks like. Content-based style tags are probably going to disappear, over time, in favor of custom XML tags; XML tags are much more flexible and expansible in terms of the content they can potentially mark up. Of the content-based tags, only three are in occasional use today: EM (emphasis), STRONG (strong), and CITE (cite/citation).

Tag: **EM**Description: Adds emphasis to a piece of text, usually rendered using italics.

Example:

<p>I <em>really</em> want to do something with my life.</p>

Displayed:

I really want to do something with my life.

If you want to give an address on a web page in place of or in addition to your citation, use the ADDRESS tag (this is not a content-based style, but I think I should mention it anyway).

Tag: **ADDRESS**Description: Marks a mailing address, including email contact, if applicable.

**28 BLOCKQUOTE and HR**

The <blockquote> tag specifies a section that is quoted from another source.

Browsers usually indent <blockquote> elements.

note that the hr tag should self-close in XHTML, i.e., <hr />

Tag: **HR**  
Attribute: **SIZE**  
Value: integer representing thickness of horizontal rule in pixels  
Description: sets HR thickness.

Tag: **HR**  
Attribute: **WIDTH**  
Value: integer representing width of horizontal rule, in pixels. May also be set to percentage value (representing percentage of browser window width).  
Description: sets HR width.

Tag: **HR**  
Attribute: **ALIGN**  
Value: **left, right, center**  
Description: sets the alignment of a horizontal rule on a page.

Horizontal rules are usually inserted to indicate section breaks in a long document, or as a divider or decorative element. HR tags used to be used with such frequency that they came to be associated with unimaginative web page design, and should be used sparingly.

**29 Miscellaneous HTML Tags**

Tag: **BR**  
Attribute: **CLEAR**  
Values: **all**, **left**, **right**  
Description: The BR tag, in combination with an IMG tag, can define how text will break in relation to a left-or-right-aligned image. **clear="all"** forces text to break below all images; **clear="left"** forces text to break below the left-aligned images; **clear="right"** forces text to break below the right-aligned images. "all" is the only commonly used value for the CLEAR attribute.

Example: **<br clear="all">**

**30 META Tag**

The **META** tag is used primarily for two purposes today:

1. It helps robotic search engines (such as Google and Yahoo) better categorize your HTML pages.
2. It automatically moves users from one page to another (as seen on splash pages or forwarding-address pages) using its auto-refresh capabilities.

requires two attributes: **NAME** and **CONTENT**. **NAME** identifies the type or variety of **META** tag in question; **CONTENT** sets the necessary information for that version of the **META** tag. The **NAME** attribute, in some cases, is interchangeable with the **HTTP-EQUIV** attribute; in other words, either **NAME** or **HTTP-EQUIV** will be used as the identifier attribute for the **META** tag, depending on the purpose of that particular **META** tag; the second attribute will always be **CONTENT**.

always goes within the HEAD of an HTML page; it is NEVER placed in the BODY. You may place as many different **META** tags in the HEAD as you like.

An HTML page on the Web is catalogued in a search engine's database with one entry for every pertinent keyword on the page. Each individual word in the TITLE of an HTML document receives a separate keyword entry in the database. Each individual keyword in the **KEYWORDS META** tag receives a separate keyword entry in the database.

1. whether the word is in the TITLE (top priority),
2. whether the word is in the **KEYWORDS META** tag, and
3. whether the word appears in the text in the BODY of the HTML page.

In addition to the position of the keyword on the HTML page as a whole, the search engine "grades" the importance of a given HTML page based on how close that page is located to the root directory of its domain; top level (root directory) pages are given top priority, even if a given keyword does not appear in the TITLE of that document, but merely in the **META** tag or in the BODY.

**DESCRIPTION META** tag allows you to create a brief description of your page or website which will display below the TITLE and URL for your website on a robotic search engine's results page.

If the **DESCRIPTION META** tag has NOT been set, then the user sees the first 10-20 words from the BODY of that HTML page in place of the description.

<meta http-equiv="refresh" content="10; url=http://www.yahoo.com/">

Two attributes of the **META** tag are required to utilize the auto-refresh feature: **HTTP-EQUIV** and **CONTENT**. The **HTTP-EQUIV** attribute must be set equal to **"refresh"**. The **CONTENT** attribute must be set equal to a value following this format: **"secondsOfDelay; url=someURL"**

**UTF-8 and HTML5**

UTF-8 stands for **U**nicode **T**ransformation **F**ormat-**8**. It is an octet (8-bit) lossless encoding of Unicode characters. The following meta statement is highly recommended in HTML5 files.

<meta charset="utf-8">

**31 Document Appearance**

#### What is the difference between the Proportional and Monospace Fonts?

Monospace fonts, such as Courier and Courier New, have characters which are all identical in width. A space character is as wide as an "M", which is as wide as an "i"; all characters in a monospace font are the same width.

Proportional fonts are used for laying out most ordinary body text, whether for publishing on the Web or for print purposes. Most text within the BODY of an HTML document is laid out using the browser's default proportional font.

Macintosh and Windows PC machines will display 12 point text in a browser window at different sizes

It is impossible to create HTML pages which display with identical font sizes cross-platform without the use of CSS. Without CSS, in fact, it is impossible to be certain of the font size under any circumstances, since users may reset browser default font sizes at will.