

UNIVERSITY OF CALIFORNIA, SANTA CRUZ EXTENSION
Winter 2011

**20816 HTML/XHTML 101:
BUILDING BLOCKS FOR WEB DEVELOPMENT**
X400.467 CMPS

4 Days

Wednesday's:
January 19th and 26th, 2011
10:30am – 5:30pm

Saturday's:
January 22nd and 29th, 2011
10:30 am - 5:30 pm

Audrey Blumeneau, Instructor
ablumene@ucsc.edu



University of California Santa Cruz Extension
20816 HTML/XHTML: BUILDING BLOCKS FOR WEB DEVELOPMENT

Instructor: Audrey Blumeneau

Course Description:

In this hands-on course, students create structurally sound, valid Web pages. Learn the history, power and limitations of basic Web page development using the Web's foundation languages, Hypertext Markup Language (HTML) and Extensible Hypertext Markup Language (XHTML) as specified by the official Web standards body. The course includes an introduction to HTML5, a new version of HTML currently under development.

Students learn to hand-code and validate HTML/XHTML pages, including how to use certain attributes for increased accessibility and search engine optimization. The course also covers incorporating images and creating absolute, relative and email hyperlinks, complex image maps, and data tables. Students will code a page that contains a form for surveys. The course also examines ways in which HTML5 changes current standards.

This course is for students who have not coded Web pages before and serves as a review for students who have some HTML coding knowledge and want to build on that knowledge as the foundation for learning XHTML or HTML5.

Course Objectives:

At the conclusion of the course, participants should be able to:

- Understand the structure of basic web page coding with HTML/XHTML
- Generate well-formed online documentation sets and understand why they are necessary

Performance Evaluation:

Through class projects students will show/demonstrate their knowledge of HTML and XHTML concepts covered in the **4-day class**. At the end of class each student will have created their own website that conforms to HTML and XHTML standards.

◆ 25 point for each class attended:	25x4 = 100points
◆ 30 points for each homework/in-class assignment:	3x30 = 90points
◆ 20 points for each Quiz:	3x20 = 60points
◆ 50 points for the Final Project	<u>50 points</u>
	300 points total

Grades:

- 300 – 286 = A
- 285 – 270 = A-
- 266 – 269 = B+
- 252 – 265 = B
- 240 – 251 = B-
- 231 – 239 = C+
- 222 – 230 = C
- 210 – 221 = C-

Assignments will be submitted via the online UCSC class web portal and uploaded to a web server. There will be a short review **quiz** at the beginning of each class.

Suggested Reading/Required and-or Recommended Texts (optional):

HTML, XHTML, and CSS, Sixth Edition (Visual Quickstart Guide), by Elizabeth Castro
ISBN-10: 0321430840; **Publisher:** Peachpit Press; 6 edition (August 26, 2006).

Head First HTML with CSS & XHTML, by Eric Freeman, Elisabeth Freeman
ISBN-10: 059610197X; **Publisher:** O'Reilly Media, Inc. (December 8, 2005)

Sams Teach Yourself HTML and CSS in 24 Hours (8th Ed.); by Julie Meloni, Michael Morrison
ISBN-10: 0672328410; **Publisher:** Sams; 8 edition (December 20, 2009).

UCSC Extension Examination and Homework Return Policy

For applicable homework and examinations that you wish to receive back, please bring a self-addressed stamped envelope to the instructor on the final day of class or send one with your submitted project/exam.

Course Outline

Class 1:

Introductions/Syllabus

- History of HTML
 - What is it? How has it evolved?
- Tools and Setting up your first page
 - HTML Editors
 - Basic HTML Elements
 - Saving your pages – Naming Conventions
- The Doctype
- Head Tags
 - Title elements
 - Meta elements
- Body Tag
 - Block Level Elements (headings and paragraphs)
 - Inline Elements (br, emphasis and strong)
- Validating your page
- Phrase Elements
- Class assignment / Wrap up

Class 2:

- Warm Up / Quiz
- Lists
 - Ordered and Unordered
 - Nested Lists
 - Definition Lists
- Horizontal Rule
- Character Entities
- Inserting Images
- Inserting Links
 - Relative, Absolute, Email, Anchor
- Images and Linking:
 - Image Mapping
 - Using Images as links
 - Linking Thumbnails to Larger Images
- Class assignment / Wrap up

Class 3:

- Warm up / quiz
- Tables
- Forms
- In class assignment / Wrap up

Class 4:

- Warm Up / Quiz
- Framesets
- iframes (Floating Frames)
- XHTML vs HTML
- HTML5
- Resources
- Final Web Site Project / Wrap up

UCSC Extension Policies:

Academic Integrity Policy:

UCSC Extension, as a unit of the University of California Santa Cruz, takes academic integrity very seriously. All forms of academic misconduct, including but not limited to, cheating, fabrication, plagiarism, or facilitating academic dishonesty are grounds for student discipline. Unless otherwise indicated by the course instructor, assignments must be individual efforts. It is not acceptable to copy (verbatim or even with minor changes) whole sections of a book, article or Internet resource, and submit them as one's own work. References should be listed and direct quotes indicated as such, with the author cited.

Grade Options

When students enroll, letter grading (**A, B, C, D or F**), is the default. The Pass/No Pass (P/NP) or Not for Credit (NC) options are available only to students in good academic standing. Students may elect to take courses for a letter grade, Pass/No Pass (P/NP) or Not for Credit (NC). However, requests for P/NP or NC grades must be submitted before the last scheduled day of the course. A passing letter-grade is required in order for a course to be applicable to a certificate.

If a student does not intend to, or for any reason cannot, complete a course s/he enrolled in, it is the student's obligation to formally notify the instructor and UCSC Extension of this change in standing before the last day of class. All withdrawal requests must be submitted using the form on the Web <http://www.ucsc-extension.edu/student-services/forms/withdrawal>. Failure to follow this policy and associated guidelines will result in the entry of a default grade of "F" on the student's permanent record. Please see Student Services webpage for links to the complete description of grading options <http://www.ucsc-extension.edu/student-services/grading>

Agenda for Wednesday, January 19th, 2010

- **Introductions/Syllabus**
- **History of HTML**
 - What is it?
 - How has it evolved?
- **Tools and Setting up your first page**
 - HTML Editors
 - Basic HTML Elements
 - Saving your pages – Naming Conventions

Lunch Break (12:30 – 1:30)

- **The Doctype**
- **Head Tags**
 - Title elements
 - Meta elements
- **Body Tag**
 - Block Level Elements (headings and paragraphs)
 - Inline Elements (br, emphasis and strong)

Break (3:00 – 3:15)

- **Validating your page**
- **Uploading (FTP)**
- **Attributes and Phrase Elements**
 - Background color
 - Text Color
 - Text formatting
 - Phrase elements

Break (4:45 – 5:00)

- **In class assignment**
- **Wrap up**

History of HTML

THE INTERNET

The “**Internet**” is defined as simply being *a large-scale network of millions of computers*.

The **Internet** dates back to the 1960’s during the cold war between the United States and the former Soviet Union so that communications between military sites could be preserved in the event of nuclear war. It came to include universities so that the military could keep in contact with the world’s top scientists. Initial sites were **.gov** and **.edu**. The **Advanced Research Projects Agency Network (ARPANET)** developed by ARPA of the U.S. Department of Defense was the world’s first operational packet switching network, and the progenitor of the global Internet.

The fathers of the Internet, **Vint Cerf** and **Bob Kahn** defined the "Internet Protocol" (IP) by which packets are sent on from one computer to another until they reach their destination.

The mode in which we communicate through these different computers is called a “**protocol**”.

Common Internet protocols that we use today include:

- **HTTP (Hyper Text Transfer Protocol)** – allow us to access the World Wide Web
- **FTP (File Transfer Protocol)** – used to upload or download web pages to and from a server)
- **E-Mail (Electronic mail)** - Allows us to send and receive email
- **IRC (Internet Relay Chat)** - allows you to connect to a chosen channel and talk in real-time to people with the same interests as you.
- **USENET (Unix User Network)** - a system of bulletin boards where you and anyone else can post messages and people will read and reply to them.

THE WORLD WIDE WEB (WWW)

The **World Wide Web** (or “the Web” or “WWW”) is defined as being *a collection of documents (text, graphics, animations, sounds or video’s) that we can access* via a protocol. The Web has its own protocol called **Hyper Text Transfer Protocol** or **HTTP**. What makes the Web unique is that you can link (via hypertext) one document to many other documents, thereby forming a “web” of information.

The **WWW** began as a networked information project at **CERN (European Organization for Nuclear Research)** also known as the **European Particle Physics Laboratory** in Switzerland in 1989/1990 (see links below)! A computer specialist, **Tim Berners-Lee**, first proposed a system of information management that used the *hypertext process* to link documents over a network. Tim Berners-Lee wrote, in 1990, the first WWW client (a browser-editor running under NeXTStep), and called it "**WorldWideWeb**". It ran on a NeXT computer along with the first WWW server along with most of

the communications software, defining URLs, HTTP and HTML. For the first few years, web pages were text based only.

In 1993, team members of **National Center for Supercomputing Applications (NCSA)** along with graduate students from University of Illinois, Champagne-Urbana, created the first graphical browser called **NCSA Mosaic**. Among the programmers was a graduate student by the name of Marc Andreessen. After graduating he moved out to Silicon Valley and together with a friend of his, started a company called Mosaic. They later changed its name to Netscape Communications Corp. in November, 1994. This changed forever the way we view information on the web.

Version 1.0 of Microsoft Corp.'s Internet Explorer browser was announced in 1995 after Bill Gates took notice of Netscape's success. Microsoft's browser competed with Netscape's browser, and each evolved its own HTML features which provided us with what is now termed as the "browser wars"!

The browser wars are what sparked Tim Berners-Lee to set up the **World Wide Web Consortium (W3C)**, which he continues to this day to lead as acting director. The W3C mission is to set the standards by which browsers are to interpret HTML and all other languages.

What does HTML stand for and what is it?

Hyper**t**ext **M**arkup **L**anguage - HTML only *marks up* (describes) data. HTML does provide a few attributes so that functionality may be added to a document, but the actual functionality is not accomplished by HTML. Therefore, writing HTML is not programming. It simply provides **structure** of the document.

The Evolution of HTML:

Read: Chapter 2: A History of HTML:

<http://www.w3.org/People/Raggett/book4/ch02.html>

Since it's inception back in the early 1990's, we now have many versions of HTML:

- **HTML 1.0** – this was the first release of HTML. Very, very basic.
- **HTML 2.0** – July 1994 version 2.0 was released. It was the standard until about 1997.
- **HTML 3.0** – In 1995 this version was suppose to appease the cries of "designers" who wanted more control over the look of a web site. Netscape was creating what was called Netscape Extension tags, but when other browsers tried to emulate, they didn't work the same and designers got frustrated. **Dave Ragget** came up with a solution by introducing HTML 3.0, which included many new tags, but browsers were slow to implement them. It was not successful.

- **HTML 3.2** – In 1997 the World Wide Consortium endorsed HTML version 3.2 which did not include many of these extension tags. Instead, this version became the standard by which the browsers and designers would start to implement.
- **HTML 4.0** - HTML 4.0 was recommended by the W3C in December 1997 and became the official standard in April 1998. Browser support was undertaken surprisingly earnestly by Microsoft with Internet Explorer browser, and the market-leading IE5 had good support for almost all of the new tags and attributes. In comparison, Netscape’s terribly flawed Navigator 4.7 was terrible when using HTML 4.0 and basic CSS. Navigator 6 was a vast improvement.
- **HTML 4.01** - Once HTML 4.0 had been out for a little while, the documentation was revised and corrected in a few minor ways and was entitled HTML 4.01; this final version of the specification is the one that will use as it is the base for everything that we will be learning and using.
- **XML** – XML stands for eXtensible Markup Language. It is language that lets us create markup languages. It can provide the basic structure and set of rules to which any markup language must adhere. It lays the ground rules for organizing information in a consistent manner. XML is broader than the Web as it can be used to represent any kind of information on any kind of computer – mobile phones, handhelds, televisions, radio etc. XML works together with HTML to clean up HTML and adhere to stricter rules and structure. To accomplish this, a compliant version of HTML was created called XHTML.
- **XHTML 1.0** – At the end of 1999 the W3C issued their specifications of XHTML 1.0 as a *recommendation*. Since January 26, 2000 it stands as joint-standard with HTML 4.01. XHTML marks a departure from the way new specs have worked — it is an entirely new branch of HTML, taking in ideas from XML. There aren’t many new or deprecated tags and attributes in this version of HTML, but there are things that have changed with a view of increased accessibility and functionality. It’s mainly just a **new set of coding rules**. The W3C was working on XHTML 2.0 but officially closed all work on this version in early 2009 in response to the next version of HTML – see next item!
- **HTML 5.0** – the future! Still being worked on – the W3C published a Draft version in August of 2009. The W3C states that: “*HTML 5 defines the fifth major revision of the core language of the World Wide Web, HTML... HTML 5...can be written in HTML syntax and in XML syntax*”. New versions of Firefox, Opera, Safari support HTML 5... Internet Explorer 8.0 does not. So as it stands, web designers won’t fully use this and yet, those cutting edge designers are pushing the envelope and there are pages out there using the newest version of HTML. For more information: <http://www.w3.org/TR/html5-diff/>

What's the difference between a programming language, a scripting language, and a markup language?

The primary difference between a "programming language" (C, C++, VB etc.) and a "scripting language" (PHP, ASP, JSP, JavaScript, VBScript) is that code written in a programming language needs to be compiled before it is run. Once it is compiled, it can be run any number of times.

Scripting languages, on the other hand, are interpreted at run-time. This means that every time you want to run the program, a separate program needs to read the code, interpret it, and then follow the instructions in the code.

Compiled code has already been interpreted into machine language, so it will typically execute faster because the conversion into machine language has already been done.

Markup languages (HTML, XML) are somewhat different from both of the others. A markup language is simply a set of tags that are used to "mark up" text documents so that sections of text can be logically arranged and labeled. These documents can be viewed as plain text, or, more commonly, are viewed through a browser. The browser parses the document, looking for markup tags, and it then arranges the text and/or formats it according to the values in the tags.

Although both called "languages", programmatic languages and markup languages differ immensely.

Programming languages:

- are dynamic
- process data through calculations such as sort, query, input, output, and render
- they use the CPU cycles of the computer to input information, act upon it and produce results

Markup languages:

- are static – a document with markup can do nothing by itself
- do not process information

A programming language can easily process the information in markup for different uses.

Markup languages identify similar units of information and aids the way in which an application can read and process the document.

(from: <http://cs.senecac.on.ca/~atworkow/11lecture/11lecture-6.html>)

Tools and Setting up your first page

Software to write HTML

You don't need to go out and buy any fancy software to write HTML/XHTML code. In fact, you don't need to spend a penny on software to create web pages! Well, that's not completely true...but for what we are going to do, we can use what comes with our computers – or better yet, get free software!

If we were using a Windows Operating System (Windows 2000, XP or Vista) we would use **Notepad** or **WordPad** which are free text editing tools that come with the Windows operating system. Mac's Operating System, OS X+ has the equivalent called **TextEdit**.

It's not to say we can't go out and download other text editors. In fact if you are using Windows, I STRONGLY recommend getting a better version of Notepad called **Notepad++**. It is better because it *colorizes*, *indents* and *numbers* your code and makes it easier to read. Another program is called **TextPad** and it too colorizes and indents your text. Both of these programs can be used for other hand-coding projects including Java and JavaScript.

HTML Editors:

Comes with your Operating System:

- Mac: **TextEdit**
- Windows: **Notepad** or **WordPad**

Free Downloads:

- **Notepad++** (Windows Only) <http://notepad-plus.sourceforge.net/uk/site.htm>
- **Programmer's Notepad** (Windows Only) <http://www.pnotepad.org/>
- **TextWrangler** (Mac Only) <http://www.barebones.com/products/textwrangler/>

Purchase:

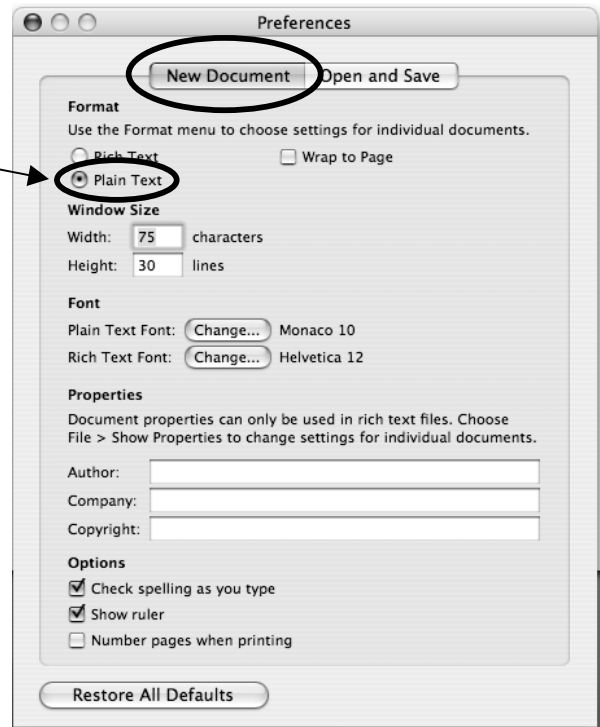
- **TopStyle** (created by same person who created Homesite)
<http://www.newsgator.com/Individuals/TopStyle/Default.aspx>
- **BBEdit** - (BareBones Software) (Mac Only). <http://www.barebones.com/>
- **SubEthaEdit** (Coding Monkeys) (Mac Only)

For this class, we are going to use **Programmer's Notepad**. However I will go over a few issues/preference settings with other editors in case you prefer to use these at home/work.

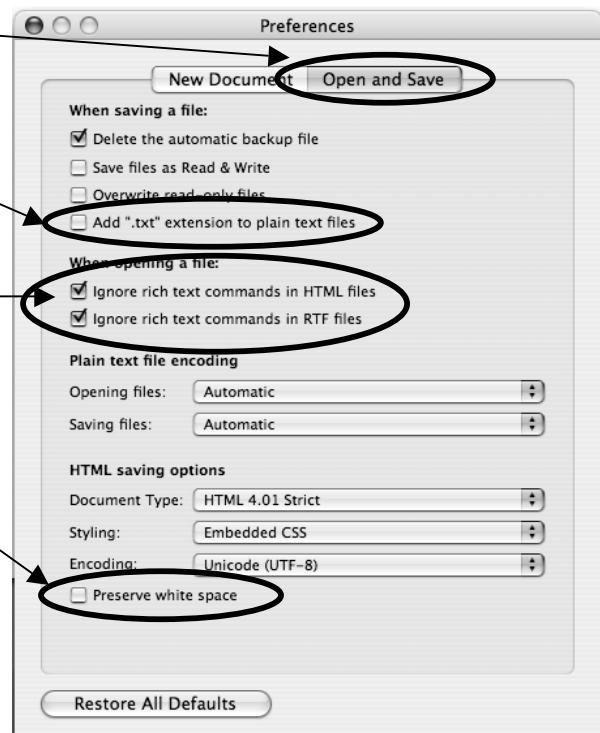
Using TextEdit on Mac OS 10.4:

There are a few settings for TextEdit (which comes with any MAC OS) as we wish to view the HTML code and edit it later.

1. Open TextEdit. (Click on the **Hard Drive > Applications > TextEdit**)
2. Choose: **TextEdit > Preferences.**
3. In **New Document**, make sure "**Format**" is set to "**Plain Text**"



4. Go to **Open and Save.**
5. **Uncheck** to box under **When Saving a file:** "**Add \".txt\" extension to plain text files**".
6. Check the both boxes for "**When opening a file:**" "**Ignore rich text commands in HTML files**".

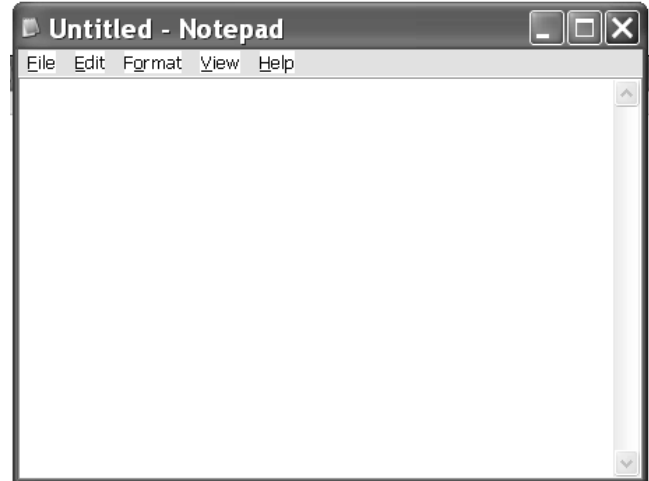


7. **Uncheck** "**Preserve White Space**"
8. **Close Preferences** (click on red button on upper right corner to close window).

Open a **New file** and **Save As:** someName.html

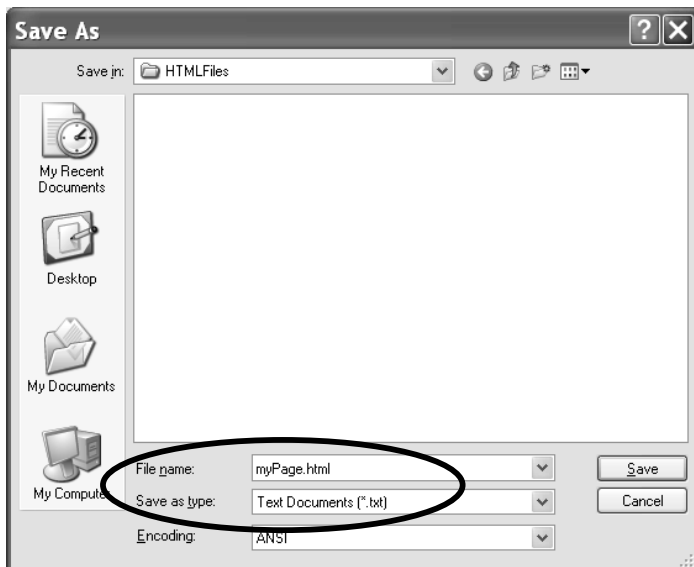
Using NOTEPAD (Windows Operating System):

- 1) Open a new document – **File > New**.
- 2) Choose **File > Save as** to save it as “someName.html”.



All documents will be saved with an extension of **.html**

When returning to Notepad, and you want to open a saved html document, make sure that in the category called **Files of Type:** that “**all files**” has been selected – this way you will be able to “see” all your html documents.



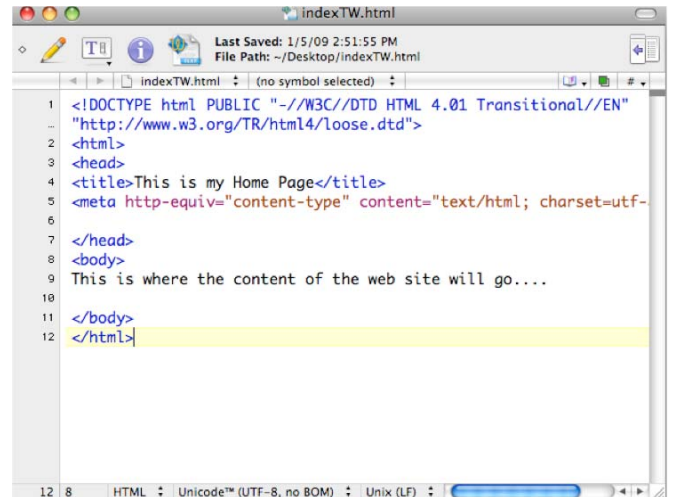
Freeware - for the Mac:

Text Wrangler

To increase the font size, go to **TextWrangler > Preferences;**

Choose: **Editor Default** and Set Default Font button.

To wrap text, choose the small arrow on the "T" icon shown at top of image and choose "**Soft Wrap Text**".



The screenshot shows the TextWrangler editor window titled 'indexTW.html'. The menu bar includes 'File Path: ~/Desktop/indexTW.html'. The code in the editor is as follows:

```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  - "http://www.w3.org/TR/html4/loose.dtd">
2 <html>
3 <head>
4 <title>This is my Home Page</title>
5 <meta http-equiv="content-type" content="text/html; charset=utf-
6
7 </head>
8 <body>
9 This is where the content of the web site will go...
10
11 </body>
12 </html>
```

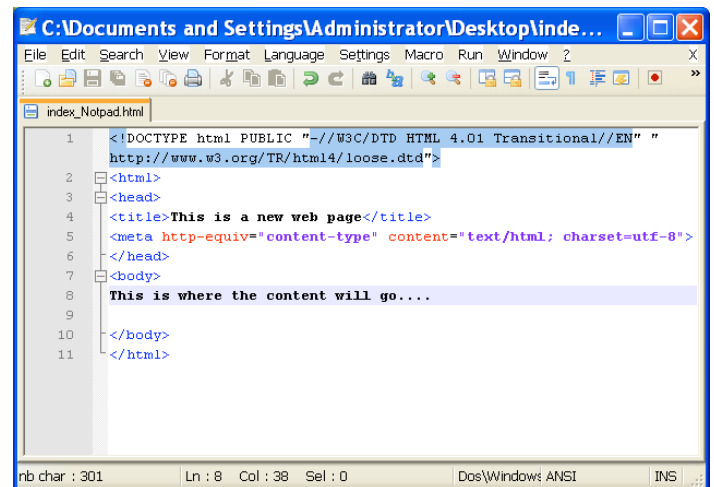
Freeware - for Windows:

Notepad++

Easy download and really no set up on this at all.

Set **Word Wrap** with icon on far right (under the word Window on the menu).

What is nice about Notepad++ is that you can collapse section of code (see little boxes next to <html>, <head> and <body>).



The screenshot shows the Notepad++ editor window titled 'C:\Documents and Settings\Administrator\Desktop\inde...'. The menu bar includes 'File Edit Search View Format Language Settings Macro Run Window ?'. The code in the editor is as follows:

```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "
  http://www.w3.org/TR/html4/loose.dtd">
2 <html>
3 <head>
4 <title>This is a new web page</title>
5 <meta http-equiv="content-type" content="text/html; charset=utf-8">
6 </head>
7 <body>
8 This is where the content will go...
9
10 </body>
11 </html>
```

The Dissection of HTML:

- **Tags:**

Elements that are written between < and > (known as angle brackets) are HTML **tags**.

<body> is a tag and the word body is an element. Sometimes you will hear or read about tags or elements. The tags tell a browser how to display your images and text. Tags usually come in pairs. There are starting and closing tags for most tags – a starting <body> tag. A closing tag has a / (forward slash) inside the tag: </body>. Not all tags have a closing (or “end”) tag as we will see later; we will also see how XHTML deals with these types of non-closing tags.

- **Attributes:**

These add options to tags and are placed inside the opening tag. If I want the background color of a web page or table to be white, I will use an **attribute** called **bgcolor** to tell the browser that I want to change the default color to something I have choose.

- **Values:**

Attributes have values. If I want to use the **bgcolor** attribute to change the background color of a page or table, the value of the attribute will make it be so: **bgcolor="#ffffff"**

- **Quotation Marks:**

Values should be enclosed in "**straight**" **Quotation Marks** (not “**curly**” ones). These are mandatory in XHTML. In order to get into good coding practices, consider them mandatory in HTML.

Basic overall Page Structure – these tags give the document structure.

<html>	Opens up the html document
<head>	Opens content that is not visible to the user such as: scripts (javascript), css, meta tags and title tags.
</head>	Closes the content not visible to the user.
<body>	Opens the content which is visible to the user.
All Content (images, text, links) goes between the body tags.	
</body>	Closes the visible area to the user.
</html>	Closes the html document

How to Name Files and Directories

Often our Web files sit on a Unix server, so Unix file and directory (folder) naming conventions apply. This means that file and directory names can contain **ONLY** the following:

- Uppercase letters A-Z.
- Lowercase letters a-z.
- Numbers 0-9.
- A hyphen – use sparingly.
- An underscore.
- A dot (a period) but only between the name and the extension.

In other words: **no** other punctuation, **no** special characters, and **NO SPACES**.

Repeat: **NO SPACES** in file or directory names!

Additionally:

- File and directory names should not start with a dot.
- File and directory names are case-sensitive.



File name extensions for Web pages

Web pages should always be named with the following extensions:

- **.html**
- **.htm**

Your Home Page:

The home page of any web site shall be named:

- **index.html**
- **index.htm**

DOCTYPE Declaration (DTD)

HTML documents actually begin with a **DOCTYPE declaration (DTD)** that identifies the version of the HTML page to the browser or validation service.

The **doctype** tells the browser what to expect from the HTML contained in the document and thus how to render it. Without the **doctype** the browser pretty much assumes that this is a legacy document as opposed to a standards compliant document where the developer knew exactly what she/he was doing.

There are three different **DOCTYPE** declarations for HTML 4.01 (almost the same for XHTML):

<http://www.w3.org/QA/2002/04/valid-dtd-list.html>

• **Strict HTML 4.01 DOCTYPE**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

• **Transitional HTML 4.01 DOCTYPE**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

• **Frameset HTML 4.01 DOCTYPE**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
```

An **XHTML doctype for transitional** would look like this:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

A document that does not use any deprecated or presentational tags or attributes will pass “**strict**” validation. If the document uses **deprecated** elements (such as fonts and alignments) then a “**Transitional DOCTYPE**” would be used.

Deprecated: means that a tag or attribute is being phased out and should be replaced with Cascading Style Sheets. They are generally “presentational” tags/attributes that Transitional will allow. I will color in **red** any code that is considered “deprecated”. The reason I include them is because you will see them everywhere and it is important to know what they are.

These links discuss the importance of using the DTD in your document.

- **Fix Your Site With the Right DOCTYPE!: A List Apart**
<http://www.alistapart.com/articles/doctype/>
- **Browser News: Doctypes**
http://www.upsdell.com/BrowserNews/res_doctype.htm

I will talk about HTML 5’s DOCTYPE later in the course so as not to confuse, as we won’t use it yet.

The HTML tag <html> ... </html>

Once the <!DOCTYPE> has been entered, you can now begin your HTML page with the opening <html> tag. This tag tells your browser that this is the start of an HTML document. Place the closing </html> tag at the bottom of your page to tell the browser this is the end. **Everything** you do from this point goes in-between these two tags.

The <head> Tag

The next tag, after the <html> of your HTML document should be:

```
<head>
```

The information contained in the <head> section of your document provides information to browsers and search engines, but is not displayed to the person viewing the page. These include meta tags, scripts and title tags.

This tag **must** have a closing tag, which is:

```
</head>
```

The <title> Tag

The **title tag** goes between your opening <head> and closing </head>.

```
<title> This is the title of my page. </title>
```

The text between the open and close <title> tags will:

- 1) Appears on the **Browser Window Title Bar** (see top of the browser window)
- 2) be used by **search engine's** looking for key words to match a request and
- 3) be used as the text in your favorites/bookmarks when you **bookmark the site**.,



As a result, you should keep these things in mind when writing a title for your web page (each web page gets a unique title):

- **Keep the <title> information short.** Title information should not be more than 60 characters or 80 characters. This includes spaces, commas etc. Keep it short and sweet and use keywords that describe the page (i.e. if on an About the Company page, the title should reflect this).
- **Keep the <title> meaningful.** When coming up with a title, don't say something like "Welcome to our site". Give the name of the company or your name. "**Ted Smith's Art Portfolio**" is far better than "**Welcome to Ted's Site**", both for search engines but also for people who are going to bookmark your site.
- **It's a required tag:** HTML 4.01 specification states that all documents must contain a meaningful title within the <head> of the document. Titles should contain only ASCII characters (letters, numbers, base punctuation). Special characters such as "&" should be referred to by their character entities, in this case "&#amp;";

Declaring the Encoding:

All text documents are saved with a “**Character Encoding**”. This makes it easier for browsers on systems with different default encodings to view the characters in your pages correctly.

```
<meta http-equiv="content-type" content="text/html; charset=utf-8">
```

More on Character Sets:

<http://www.w3.org/TR/html401/charset.html#h-5.2>

Character Encodings:

<http://www.w3.org/International/O-charset.en.php>

Encoding on the Internet (Penn State) – Excellent source of information:

<http://tlit.its.psu.edu/suggestions/international/web/encoding/index.html>

Other <meta> Tags:

Between your <head> & </head> also goes the <meta> tag(s). These are used to present information to search engines, robots and spiders, not to your readers.

```
<meta name="description" content="this is a page about stuff.">  
<meta name="keywords" content="page, nothing, programming">
```

These are two of the more common <meta> tags for assisting search engines in indexing your site.

There are no closing <meta> tags because it is not a container tag.

If using an **XHTML <!DOCTYPE>** you must close the meta tag with a space and a forward slash at the end.

```
<meta name="keyword" content="stuff." />
```

The <body> Tag

The <body> tag immediately follows the </head> tag and defines the beginning and end of the document that contains the content that will be visible or displayed to the user on the browser window.

```
<body>      </body>
```

Block Level Elements

Block level elements include html elements such as headings, paragraph and list tags. What this means is that by default on the browser window, there will be line space before and after the element. The next element will automatically be set on a new line – so every time one uses a heading or paragraph element, the next one will automatically be placed on a new line below the previous one. Imagine a “box” around each of these elements – the browser determines, via browser styles, the distance set between headings and paragraphs.

The Heading tag <h1> </h1>

Headings are displayed in bold text with space above and below. There are 6 levels of headings <h1> to <h6>, with <h1> being the largest and <h6> being the smallest font displayed on the browser. This means that <h5> and <h6> headings are generally displayed the same size or smaller than the default font size. Legal HTML syntax requires that headings appear in order. For example, <h3> cannot precede <h2>. The attribute for this tag is:

```
align="center" / "left" / "right"
```

PLEASE NOTE: The “align” attribute is **deprecated**. You will still find it used by many sites.

```
<h1 align="center">This is a Heading Level 1 that is centered</h1>
```

The <p> Tag

Browsers ignore carriage returns, so you must tell the browser using an HTML tag to create a new paragraph or line break. To create a paragraph you use the <p> tag. Although technically the closing tag is optional, please get into the habit of closing this tag. In the future, XHTML standards will mandate this tag be closed – it will no longer be optional.

The attribute for the <p> tag is the “align” attribute, which is a **deprecated** attribute. It aligns text with the tags to the left, right, or center of the page.

```
<p>Here is some text</p>  
<p>And here is the next paragraph</p>
```

This paragraph now has vertical spacing that has been created by adding paragraphs.

Inline Elements

Inline elements are those elements that are placed “*in the line*” of that particular heading or paragraph. For instance if we wanted to place *emphasis* on a word we would use the element in front of the text we want to emphasize and close it after the word or words we wanted to emphasize. Another useful “Inline element” is the
 tag, which gives us a line break rather than a paragraph break.

The Tag

NOTE: This is a DEPRECATED element.

The tag is an inline element used to specify **size**, **color**, and **font face** for the enclosed text. It was created as an extension to HTML to give designers more control over the display of text. This tag is deprecated. The W3C has phased out this tag and all its attributes in favor of using CSS.

Please note and understand that I am including this tag only because you will see it in old web sites and you should know what it is and how it works; if you are building a new web site – you would NEVER use this in a professional job or web design company!

Specifying Font Type ``

The “**face**” attribute was introduced by IE 1.0 and later adopted by NN in Version 3.0 (which of course means that NN 2.0 and below do not recognize this attribute). Also, earlier versions of HTML did not use this tag. It was used beginning with HTML 3.2.

The tag does not guarantee that the user will see your specified font because it still depends on whether you user has that font installed on their system. You can consider it rather as a recommendation!

```
<font face="Arial, Helvetica, sans-serif">  
this text is in Arial or Helvetica depending on your computers' operating system.  
</font>
```

- All text between the `` & `` will be displayed in the indicated font.
- Place commas between font names.

Note about Fonts types: If the computer does not have any of the fonts you listed, it will use its default font face. The browser will look at the string of font names until it finds one that is installed on the system. If no fonts are found, the default font is used instead. This is why you should include a **generic font family** – sans-serif, serif, mono-space, cursive or fantasy.

Another Note about Validation issues: When adding the `` tag, because it is an *inline* element, **you must place it within block level elements** such as `<p>` or `<h1>` tags. This means that for every paragraph, you have to put in a new font tag. Yes, this gets to be a pain and it adds tons of code to your page. Remember, the font tag is considered presentational code so therefore web designers will use CSS to control the type of font, the size and the color of text. In the end, we will NOT USE THIS TAG!

Specifying size with ``

The “**size**” attribute is used within the `` tag to adjust the size of the type. (It has been supported since version 1.1 of NN and IE!). Browsers measure the size of HTML type on a relative scale from 1 to 7 where 3 is the default and will be displayed at the size specified by the user’s preferences. These sizes are “relative” meaning that they don’t signify actual pixel or point adjustments. Each size is about 20% smaller or larger then the default size 3.

```
<font size="3"> Text is set to default size</font>
```

Size can be specified as an absolute value or relative value. This means that when a relative value is given, the default value is increased or decreased by that relative amount (+1 - +7).

You set the following attributes this way:

- **size="2"** is absolute to the browser and thus smaller.
- **size="5"** absolute to the browser and thus larger.

So a tag would look something like this:

```
<p><font face="Verdana, Helvetica, Arial, sans-serif"
color="#ff0000" size="4">
```

Here is some text that should show up red and be slightly larger than the default set on your browser.

```
</font></p>
```

The
 Tag -

If you want to break a line by not adding any extra space, insert a line break with the
 tag. The line break does not have a closing tag. It can be placed in the flow of the text where you want the line break to occur.

```
<p>Here is some text
<br>and here is the next line</p>
```

Web Standards Project:

What is it all about:

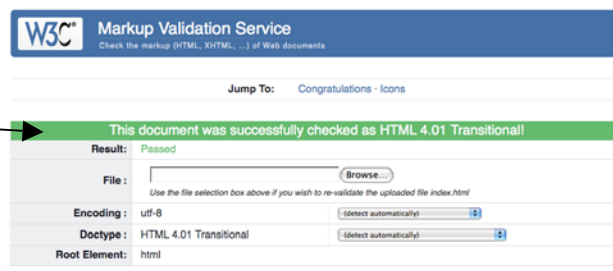
<http://webstandards.org/learn/faq/>

Validating your Web Page:

It has become mandatory for web designers to use valid HTML/XHTML. Any web design firm will expect you to know and understand what it means to write "valid code".

We will be using the W3C's validation service although there are others.

1. Go to the **W3C – HTML Validator**: <http://validator.w3.org/>
2. Select the tab: "**Validate by File Upload**". We can check our pages from our computer by "uploading" the page.
3. Click the "Browse" button and locate the HTML file you wish to validate.
4. If your page validates – you get green!
5. If your page does not validate (a red bar shows), make the corrections and validate until you get a message that your page validates. **TIP: correct only the first listed problem**; save your document and try again. Sometimes fixing the first problem fixes the rest of them.
6. Follow the instructions to put the Validate Icon on your page.



(you also have a choice of a blue icon!).

Do not get discouraged!

Don't get discouraged about finding tons of "errors". The top mistakes are:

- **Typos** – watch out for inverted letters in your tags (my personal most common error!)
- **Missing quotation marks** (only the opening or the closing quote was placed).
- **Missing equal sign (=)** between attributes and values
- **Differentiate between colons (:) and semi-colons (;)** – They look the same, so sometimes I find that I need to zoom in to read my page better (poor eyesight...).
- **The DOCTYPE and the META tag** for charset are often another area of problems...these two need to be written EXACTLY as shown – uppercase and lower case sensitivity included!

Enhancing a simple page and Phrase Elements

Background Colors and Text Colors:

You can optionally set the following attributes within the `<body>` tag:

Note these are deprecated attributes that you will replace using Cascading Style Sheets as they are considered presentational and not structural. Again, I am showing you these deprecated tags because you will still see plenty of web sites that still incorporate these attributes and you need to know what they are, what they do and why/how to replace them with CSS. I know that many people want their pages to look “pretty” right away. We are building the structure of our web pages. Here is how in the old fashioned HTML ways (including XHTML Transitional). If we were using a **Strict doctype** for either HTML or XHTML, these pages would not validate if we were to use these attributes. The **Transitional doctype** considers these valid code.

- `bgcolor="#FFFFFF"` makes the background color white (**deprecated**)
- `text="#000000"` text will be black (**deprecated**)

A `<body>` tag that specifies the documents' background color and text color would be written like this (notice only spaces separate the attributes – no commas):

```
<body bgcolor="#ffffff" text="#000000">
```

These “colors” are written in Hexadecimal code although you can write color names (i.e. white, black). However, the W3C HTML/XHTML and CSS standards have listed only 16 valid color names:

aqua, black, blue, fuchsia, gray, green, lime, maroon,

navy, olive, purple, red, silver, teal, white, and yellow.

If you want other colors and valid HTML/XHTML or CSS then you must use the HEX values (i.e. #660099) instead.

To learn more about Hexadecimal color code or name code visit these web sites below:

<http://www.lynda.com/hex.asp>

<http://html-color-codes.com/>

http://www.w3schools.com/html/html_colornames.asp

Text Formatting Tags

The `<pre>` tag:

```
This is
      preformatted text.
It preserves   both spaces
      and line breaks.
Great for poetry or
      Or for computer code!
```

Essentially the `<pre>` element simply preserves any line breaks and spaces.

Bold `` `` or Strong `` ``

Enclosed text is rendered **bold** on the browser.

Italic `<i>` `</i>` or Emphasis `` ``

Enclosed text is rendered in *italics* on the browser.

As you see there are two tags each for making text bold or italicized, so which one should you use?

Appearance vs. functionality (bold vs. strong and italic vs. emphasis?):

`<u>`, `` or `<i>` are **plainly decorative**, and are only meant to give an element some specific appearance. There should be a **reason** for highlighting those words. If you want to “emphasize” a word, then `` is far better to use than `<i>`.

`` and `` are used by screen readers to do what they describe. In fact, the voice inflection actually changes depending on this tag.

When working with an older document:

- ◆ Replace any tag with `` to `` **This is strongly suggested text** ``
- ◆ Replace any `<i>` tag with `` *This is emphasized text* ``

The Underline element: `<u>` `</u>` NOTE: This tag has been **DEPRECATED**

Enclosed text is rendered with an underline on the browser. This is a tag that really should never be used, plus it is **deprecated**. I show it only so that you know that it exists....and you can be smart about not using it!

"Computer Output" Tags

The <code> tag:

This tag not only formats the text between this element in a specific way it also provides information to screen readers about what they are reading (that the following text is computer code).

```
<code>Computer code</code>
```

This code will render as a **monospaced** font on a browser: `Computer code`

The <kbd> tag:

```
<kbd>Keyboard input</kbd>
```

This code will render as a **monospaced** font on a browser: `Keyboard input`

It signifies that text surrounded by this tag is supposed to be entered by the reader on their keyboard

The <tt> tag:

(Note: This tag has been removed from HTML 5 specifications)

```
<tt>Teletype text</tt>
```

Makes the text 'typewriter text', and sets it in a **monospaced** font: `Teletype text`

The <samp> tag:

```
<samp>Sample text</samp>
```

This code will render as a **monospaced** font on a browser: `Sample text`

It simply stands for sample, and is used when you're using an example, usually of an output from a program (for computer geeks!).

The <var> tag:

```
<var>Computer variable</var>
```

This code will render as a monospaced font on a browser: `Computer variable`

This is used to denote variables in equations

Citations, Quotations, and Definition Tags

These elements are meant to provide “meaning” to a page; they can be considered “logical” elements in that they give structure to the page by describing what they contain.

The `<address>` tag:

```
<address>
Mr and Mrs. Jones<br>
P.O. Box 555<br>
SomeCity, SomeState, USA <br>
</address>
```

This will render like this

Mr and Mrs. Jones
P.O. Box 555
SomeCity, SomeState, USA

The `<abbr>` and `<acronym>` tags:

```
<abbr title="California">CA</abbr>

<acronym title="World Wide Web">WWW</acronym>
```

What is the difference?

This is a rather heated debate...amplified more so by the fact that Internet Explorer doesn't support the `<abbr>` tag. When I say doesn't support, it means that it doesn't apply the default styling and tool tips that one gets with other browsers. Hence the reason one might want to use the `<acronym>` tag. There is definite confusion out there about when to use which one but here is what the W3C says about the topic:

ABBR:

Indicates an abbreviated form (e.g., WWW, HTTP, URI, Mass., etc.).

ACRONYM:

Indicates an acronym (e.g., WAC, radar, etc.).

Here is what the *Merriam-Webster's Collegiate Dictionary* states:

- an **acronym** as "a word (as NATO, radar, or snafu) formed from the initial letter or letters of each of the successive parts or major parts of a compound term;"
- an **abbreviation** as "a shortened form of a written word or phrase used in place of the whole."

So you can see right from here, how the two get this seemingly simple concept confused. The general consensus seems to be that you wrap everything up as an **abbreviation** as that is more semantically correct.

The `title` attribute is used to show the spelled-out version when holding the mouse pointer over the acronym or abbreviation. This only works for the acronym element in IE 5.

1. IE6 does **not** support `<abbr>`
2. **HTML5 does not support `<acronym>`** and because of this... we will use `<abbr>`

More information/discussion:

- ◆ http://www.netmechanic.com/news/vol8/usability_no1.htm
- ◆ http://www.evolt.org/article/HTML_is_not_an_acronym/17/35750/

Quotation Elements:

Long Quotation: This is how a long quotation is written in HTML:

```
<blockquote  
cite="http://www.sfmoma.org/eliasson/data/index.html">  
  I think the principle question about taking your time is also about generosity.<br>  
  Olafur Elliasson.  
</blockquote>
```

The long quotation will look like this the browser:

```
I think the principle question about taking your time is also about generosity.  
Olafur Elliasson.
```

You can use the `cite` attribute to cite where you got the quotation from. Notice that you won't get the quotation marks using the `<blockquote>`.

Also note: You can add `<p>` tags *within* a `<blockquote>`, **but don't put a `<blockquote>` within `<p>` tags** as this will not validate.

Short Quotation: This is how a short quotation is written in HTML:

```
<q>I think the principle question about taking your time is also about generosity.</q><br>  
Olafur Elliasson
```

This is how the short quotation looks on the browser:

```
"I think the principle question about taking your time is also about generosity."  
Olafur Elliasson
```

With the `<blockquote>` element, the browser inserts line breaks and margins, but the `<q>` element does not render as anything special.

```
As <cite>Harry S. Truman</cite> said,  
<q lang="en-us">The buck stops here.</q>
```

Superscripts and Subscripts

Using a **superscript** for this sentence:

```
This is the 4th time!
```

We would be written like this in X/HTML:

```
This is the 4<sup>th</sup> time!
```

To use the **subscript**, we could write something like:

```
Did you know that H2O was water!
```

The X/HTML code would look like this:

```
Did you know that H<sub>2</sub>O was water!
```

Delete and Insert Tags:

Here is one lawyers and bloggers use to make edits to copy on a web site so that one can mark changes to the text.

Marking **newly inserted text** use `<ins> </ins>`

Marking **text that is to be deleted** use ` `

This two tags are meant to work together. What will show on the browser is that any inserted text will be underlined and any deleted text will show a strikethrough. You could use the underline tag or even a strike through tag, but this clearly identifies what is happening (text is being inserted rather than just underlined for some unknown reason).

Example:

The soccer game is going to be `Friday <ins>Monday</ins>`.

On the browser it will look like this:

The soccer game is going to be ~~Friday~~ Monday.

For more about **Phrase Elements** see the W3C:

<http://www.w3.org/TR/html401/struct/text.html#h-9.2.1>

Comment Tags:

These are great for adding comments either for yourself or for other people who are working with you. The user who is viewing your web page can't see the comment. You can only see comments within the code itself.

```
<!-- This is a comment tag -->
```

Homework Exercise:

1) Create an "About Me" web page:

a) Using an editor of your choice (Notepad++, Smultron or TextWrangler) create a new HTML document and save it "YourLastName_Aboutme.html".

b) This page must display your full name.

c) This page must include at least two paragraphs

- Information about you (e.g.: a synopsis of your education, jobs, travels, etc.).
- What are your personal goals (a degree in what? Dream job doing what?)?
- Include basic tags like **Headings** and **Paragraphs**. Add some **Phrase Elements** (doesn't have to include every single one in the handout, but include tags like ``, ``, `<sub>`, `<sup>`, `<blockquote>` as these are fairly commonly used.
- Include copyright information at the bottom of the page.

d) This page's `<title>` must indicate that this is your page (therefore your name).

e) Each assignment will have appropriate and well thought out `<meta>` tags for **description** and **keywords**.

f) The page must contain the **W3C's HTML Validator** icon -
`http://validator.w3.org/#validate_by_upload`

g) **Save this file as "YourLastName-Aboutme.html"**. Remember: no spaces! (i.e. Blumeneau-Aboutme.html).

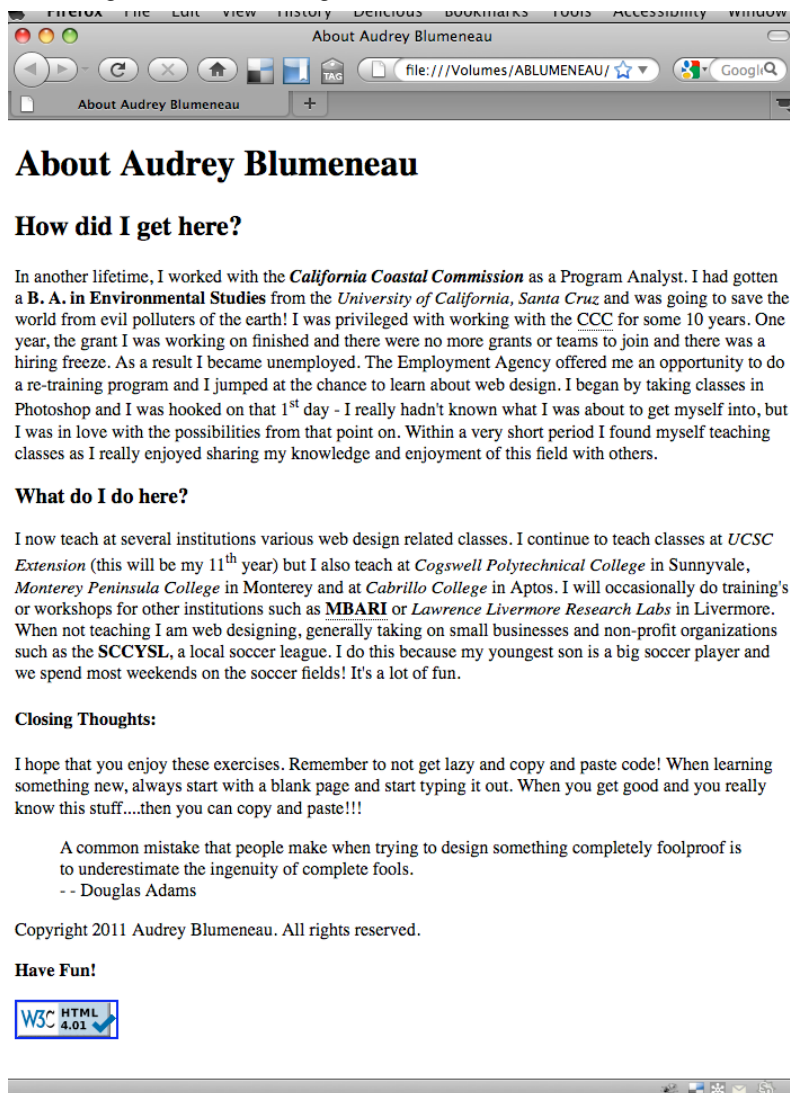
2) Homework will be graded based on the following criteria (total points: 30):

A: Includes tags as asked for above plus tags discussed in class or handouts for this section; the page looks good, and it **Validates with no errors**. (Validation Icon is included).

B: Includes many of the required tags, but not all. Page does validate but with warnings.

C: Work has minimal tags and/or does not validate.

An example of what you should be able to do for your homework. This page includes three headings (**h1**, **h2**, and **h3**), **paragraphs**, **strong** and **emphasis** elements, **abbreviations**, **sup** and a **blockquote**.



About Audrey Blumeneau

How did I get here?

In another lifetime, I worked with the *California Coastal Commission* as a Program Analyst. I had gotten a **B. A. in Environmental Studies** from the *University of California, Santa Cruz* and was going to save the world from evil polluters of the earth! I was privileged with working with the *CCC* for some 10 years. One year, the grant I was working on finished and there were no more grants or teams to join and there was a hiring freeze. As a result I became unemployed. The Employment Agency offered me an opportunity to do a re-training program and I jumped at the chance to learn about web design. I began by taking classes in Photoshop and I was hooked on that 1st day - I really hadn't known what I was about to get myself into, but I was in love with the possibilities from that point on. Within a very short period I found myself teaching classes as I really enjoyed sharing my knowledge and enjoyment of this field with others.

What do I do here?

I now teach at several institutions various web design related classes. I continue to teach classes at *UCSC Extension* (this will be my 11th year) but I also teach at *Cogswell Polytechnical College* in Sunnyvale, *Monterey Peninsula College* in Monterey and at *Cabrillo College* in Aptos. I will occasionally do training's or workshops for other institutions such as *MBARI* or *Lawrence Livermore Research Labs* in Livermore. When not teaching I am web designing, generally taking on small businesses and non-profit organizations such as the *SCCYSL*, a local soccer league. I do this because my youngest son is a big soccer player and we spend most weekends on the soccer fields! It's a lot of fun.

Closing Thoughts:

I hope that you enjoy these exercises. Remember to not get lazy and copy and paste code! When learning something new, always start with a blank page and start typing it out. When you get good and you really know this stuff...then you can copy and paste!!!

A common mistake that people make when trying to design something completely foolproof is to underestimate the ingenuity of complete fools.
-- Douglas Adams

Copyright 2011 Audrey Blumeneau. All rights reserved.

Have Fun!

