INTRODUCTION TO HTML AND CSS:
CLASS 1
HTML BASICS

Instructor: Mignonne Maxwell

Learning Objectives:

- The history of HTML and who uses it now
- HTML vs. CSS
- Elements and Tags
- Files and folders
- Organizing your code.

HTML: The Bare Bones

```
<!DOCTYPE html>
<html>
        <head>
                <title>My web page</title>
        </head>
        <body>
                <h1>Hello World!</h1>
                <p>Meet your new web designer!</p>
        </body>
</html>
```

History

90s

- Invented by Tim Berners-Lee
- Created "hypertext" to share scientific papers
- First web page August 6, 1991
- No layout, No styling.
- A great example! http://www.w3.org/People/Raggett/book4/ch02.html
- HTML 4 in 1997-1998
    - Standardized by w3 Consortium (pack of super nerds)
    - Pages still had very little styling options that would work in every browser.
    - Plenty of table-based layouts around!

2000s

- XHTML in 2000
    - Combined XML and HTML
    - Introduced stricter syntax
- HTML 5 in 2008-2009
    - Adopted over XHTML 2.0
    - Still HTML, but with many new features

Who Uses HTML

Web designers:
Plan, design and create (usually) small-scale websites.
Front-end web developers:
Involved in programming dynamic web applications or large sites.

- Responsible for the user experience on large sites (client-side).
- Differ from Back-end web developers, who are responsible for handling and storing data and server file structure (server-side).

Tools of the trade

Browsers and their debuggers

- Chrome — Chrome Developer Tool (comes with)
- Firefox — Firebug (add-on)
- Internet Explorer — IE Developer Tool (comes with)
- Safari — Developer Tool (comes with and must be enabled)

Text Editors

- TextWrangler - Mac
- Notepad ++ - Windows
- Sublime Text - Linux, Mac or Windows
- Just plain old Notepad!

Web Development Applications and Integration Development Environments (IDEs)

- BBEdit - Mac
- DreamWeaver - Windows or Mac
- Microsoft Expression
- Microsoft Visual Studio

HTML and CSS

HTML:

- Came first (about 7 years before CSS).
- Provides logical structure to content (words and images) by organizing it into paragraphs, headings, tables, etc.
- Browser's default "styles", like spacing and font size, are just enough to make the structure apparent (if not appealing).
- Before CSS, some additional "styles" were applied with HTML. *This is now considered very bad coding!*

CSS:

- Stands for Cascading Style Sheets.
- Invented to remove all styles from HTML and keep them separate.
- Very flexible and powerful language that allows websites to look the way they do today.
- Describes how you want your site look — presentation-wise.

HTML Element

An element is usually composed of content (words, images, numbers, or even other elements), and HTML tags.

We create elements by "wrapping" chunks of content inside an opening *tag* and a matching closing *tag*.

```
<p>Words within a paragraph.</p>
```

HTML is *case-insensitive* but the accepted convention is to use lower-case (except for the doctype element).

Container Elements

Contain content along with an *opening* and a *closing* tag.

Commonly used container elements:

- <p> (paragraph)
- <h1> (heading levels 1 - 6)
- <table> (table)
- (unordered list)
- (ordered list)
- (list item)
- <a> (link)

Empty Elements

If the element does *not* contain content, it is said to be an **empty element**.

"
" is an **empty element** that tells the browser to insert a line break in a sentence.

It can be written three different ways:

-
</br> (open and close tag, no content)
-
 *(*self-closing tag*)*
-
 (just an opening tag)

Commonly used empty elements:

-
 (break tag)
- <img /> (image tag)
- <input /> (form input)
- <button /> (form button)
- <hr /> (horizontal rule)

Nesting

All elements "nest" inside other elements...except the HTML element! (everything else nests inside *it*)

In our bare-bones example: the "p" element nests inside the "body" element, which nests inside the "html" element.

Whichever element OPENS first CLOSES last!

Doctype Element:

The first element on an HTML page. It tells the browser which version of HTML the page is using.

Here's the old way of writing it:

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

Now we can write it this way:

> <!DOCTYPE html>

HTML Element

After <DOCTYPE>, the very next element on every page is the html element. Its opening and closing <html> tags wrap around all the rest.

```
<!DOCTYPE html>
<html>
  Everything else goes in here!
  ...
</html>
```

Head and Body Elements

Head: Contains the title, meta information, embedded styles, and often scripts.

- Meta information is not visible to the user, but tells search engines about your page, who created it, and other info.
- The contents of the head element are not visible on the web page...except the title!

Body: Contains the actual content of the page. This is the part of your page that visitors see and interact with.

Heading Elements

<h1>Heading 1</h1>

<h2>Heading 2</h2>

<h3>Heading 3</h3>

<h4>Heading 4</h4>

<h5>Heading 5</h5>

<h6>Heading 6</h6>

Heading number indicates hierarchy, not size. Important for accessibility

Paragraphs

- This is a one-sentence paragraph

Line Breaks

Imagine there's no Heaven
It's easy if you try
No hell below us
Above us only sky

Lists

Unordered list (bullets)

```
<ul>
  <li>List Item</li>
  <li>Another List Item</li>
</ul>
```

Ordered list (sequence)

```
<ol>
  <li>List Item</li>
  <li>Another List Item</li>
</ol>
```

Tables

Tables present data (information) in a grid format. They're built row by row, so they're very hard to edit/update.

```
<table>
  <tr>
    <th>Column Heading</th>
    <th>Column Heading</th>
  </tr>
  <tr>
    <td>data</td>
    <td>data</td>
  </tr>
</table>
```

Block-level vs. Inline Elements

Block-level:

- Block level elements begin on a new line, and their default width is usually the width of the browser!
- Browsers give them default padding on top and bottom.

Commonly used block-level elements:

<h1> thru <h6> (headings)
 and <ul> (lists)
 (list items)
<table> (tables)
<form> (forms)

Inline:

- Inline elements do not start on a new line and their default width is only as wide as their contents.
- They must be nested inside a block-level element.

Commonly used inline elements:

<img> (images)
<a> (links or "anchors")
 (emphasize)
 (make strong)
 (has no effect by itself)

Deprecated" (obsolete) elements:

Deprecated elements are elements that have been phased out and will eventually no longer be supported by browsers.

Examples:
<i> (italicize)
<b> (bold)

<i> and <b> both "style" the content so they are discouraged in favor of using CSS.

Span Element

- has no other purpose than to provide a "hook" to text that can't be otherwise targeted.
- Most often used for styling or scripting.
- By itself, has no visible or interactive affect on content.

Attributes

Two important elements of web pages — links and images — require *attributes*.

Attributes are components of an elements (just like eyes are components of a human).

You describe an attribute by using a *value* (like saying "Her eyes are brown").

think ~ person: eyes = "brown"

Links require an href attribute to tell where they link to (href stands for "hypertext reference").

Here's how that looks:

think ~ person: address = "123 Main Street"

Attributes are always placed inside an opening tag, before the right angle bracket.

Links

The <a> (anchor) tag surrounds text or images to turn them into links.

Links have two mandatory components:

- tag: <a>
- href attribute: "http://www.girldevelopit.com"
- A third component, the title attribute, should only be used if the link's destination isn't obvious (like clicking on an image)

Using *target="_blank"* causes the link to open in a new window/tab.

example: Link Text

Inserting *mailto:some_email_address.com* into the href attribute causes the link to open the default mail client.

example: E-mail us!

Image Element

<img> is an empty element. It is also an inline element.

Image elements have three components

- Tag: <img/>
- Src attribute: "http://girldevelopit.com/assets/pink-logo.png"
- Alt attribute: "Girl Develop It logo"

Relative vs. Absolute paths for links & images

Absolute:

- Refer to a specific location of a file on a server
  src = "http://www.girldevelopit.com/chapters/detroit"
- Typically used when pointing to a link that is not within your own domain.
- Easy to use. There's no need for a starting point.
- Think ~ searching for an address on MapQuest.

Relative:

- Refer to a local file in your site root folder
  src = "images/myimage.jpg"
- Describes the location of the file relative to the file you're in.
- Think ~ using the "directions" feature on MapQuest.

Writing Clean Code

1. Nest your tags properly!
2. Make good use of white space!
3. Leave yourself notes!

You can add comments to your code. The browser ignores them, but *you* (or another coder) can see them.

<!-- Comment goes here -->

Use them to organize your code:

- <!-- Beginning of header -->
- Header Content
- <!-- End of header -->

Or 'comment out' code (to hide it from the browser):

- <!--
-
- List Item
- Another List Item
-
- -->