HTML, XHTML & CSS

Hypertext Markup Language & Cascading Stylesheets

HyperText Markup Language

- What is Hypertext?
  - Text containing links to more text. (see http://www.w3.org/WhatIs.html)
- What is a markup language?
  - A way to embed formatting information within a text file by using tags
  - A way to describe the data or information (metadata) contained in a text file
- What is a tag?
  - Method of identifying metadata within a document
- What is metadata?
  - Data that describes data (ie GPS info in a digital picture)
  - Tags are used to describe the information

HyperText Markup Language

- So: What is HyperText Markup Language?
  - A simple way of marking up documents so that they can shared (and understood) over networks (using HTTP - which uses TCP)
- Why use it?
  - Simple
  - Lightweight (just plain text)
  - An easy way to send information both for users to see, and computers to interpret at the same time.
- Tags?
  - Tags for presentation – www
  - Tags for interpretation – Semantic Web

HyperText Markup Language

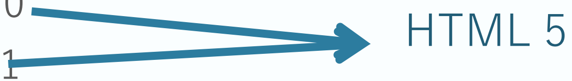
- Why learn it?? Why not just use some GUI??
  - HTML is a *semantic** markup – understanding the underlying semantics will help you make good decisions about how *machines* will interpret your documents (e.g. search engines!)
  - If you do *any* web coding, you'll want to know it
  - If you want to design or develop real-world web applications with a modern user interface, you'll want to be fluent in it
- Use markup tags to build document structure, e.g.

<p>This is a paragraph.</p>

* *the tags can say something meaningful about the content they contain*

HTML Versions

- Recent “HTML” versions:
  - XHTML 1.0
  - HTML 4.01

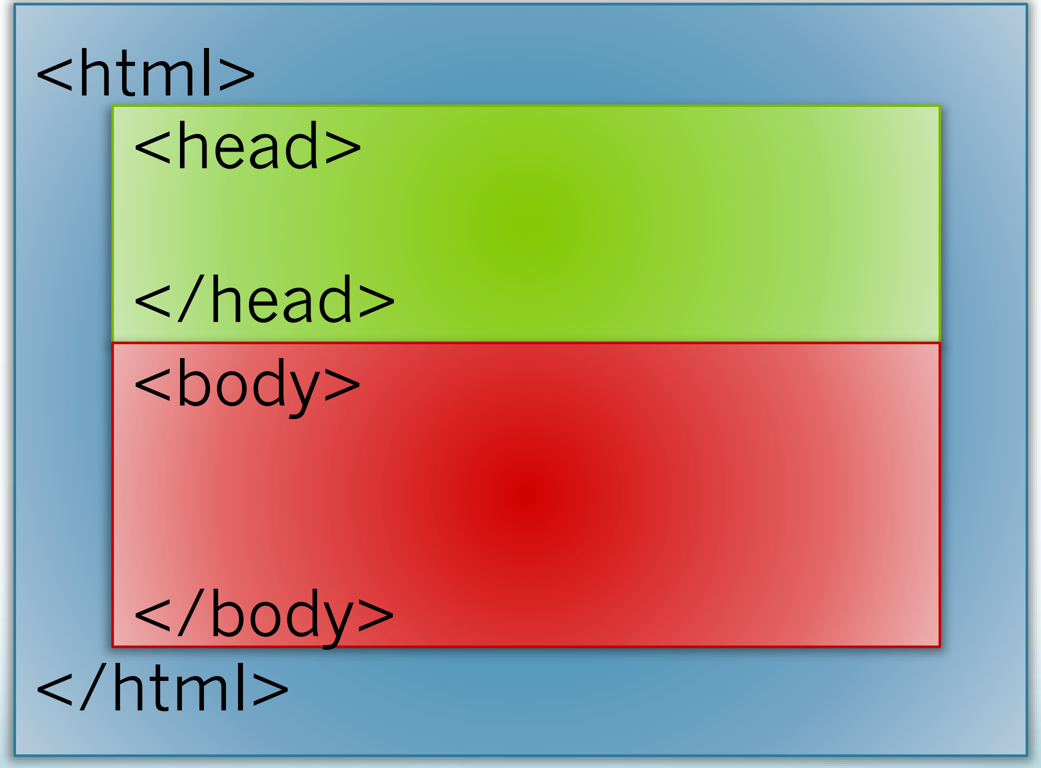


HTML 5

- We will focus on XHTML & HTML 5.x

- Work on HTML 5 is ongoing
  - Became a W3C Recommendation – in Oct 2014
  - 5.1 became a Recommendation in Nov-2016
  - 5.2 is a Recommendation as of Dec 2017
  - 5.3 is a Working Draft as of Aug 2018
  - HTML5, HTML 4.1 and XHTML 1.0 are Obsolete Specifications as of Mar 2018

An HTML Document



<html>

<head>

</head>

<body>

</body>

</html>

Basic HTML Structure (HTML 5)

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>
        a title for the document
    </title>
  </head>
  <body>
        document content goes here
  </body>
</html>
```

<!DOCTYPE> & <html>

- <!DOCTYPE x>
  - is the prologue.
  - tells the interpreter (Browser) what type of document follows.
  - x is the type of HTML that follows
- <html> ··· </html>
  - Is the main tag
  - Tells the Interpreted that what follows is the content of the document and that it is in HTML.
  - Every web page must have this tag
  - It is the root of the document

<head>…</head>

- Contains document header information, e.g. the document title, file includes, meta information, page-level scripts and styles …
- Examples of markup found in the head:
  - <title>Document Title</title>
  - <style type=“text/css”>
  - <link href=“styles.css" rel="stylesheet" type="text/css"/>
  - <script type=“text/javascript”> some javascript here </script>
  - <meta http-equiv="Content-Type"
    content="text/html; charset=UTF-8"/>

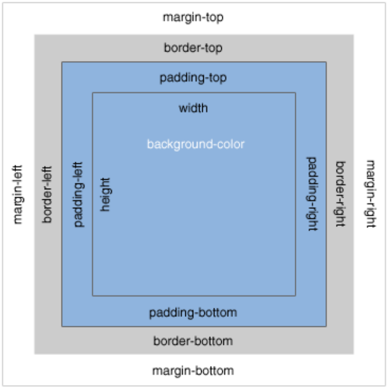
<body>⋯</body>

- Contains document content, e.g. paragraphs of text, images, captions, videos, interactive components, inline scripts …

- Examples of markup found in the body:
  - <h1>This is a First-level Heading</h1>
  - <h2>This is a Second-level Heading</h2>
  - <p>paragraph text</p>
  - <em>emphasized text</em>
  - <div id=“footer”>An arbitrary block named “footer”</div>
  - <img src=“figure1.jpg” width=“500” height=“300” alt=“Figure 1”/>
  - <ul>
    <li>bulleted list item 1</li>
    <li>bulleted list item 2</li>
</ul>

Markup Display Concepts

- HTML Tags have three basic display states:
  1. block
     - block elements can be thought of as a box with breaks before and after, e.g. paragraphs & headings
  2. inline
     - inline elements flow with the content around them and do not break before and after, e.g. *emphasized text*
  3. none
     - elements with display set to none will be hidden in the browser (though their content will still exist in the markup), e.g. scripts



CSS Box Model [2]

XHTML Document

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
      <title>
           a title for the document
      </title>
   </head>
   <body>
      <div id=“content”>
          document content goes here
      </div>
   </body>
</html>
```

XHTML 1 of 2

- eXtensible HyperText Markup Language
- Consists of
  - a DOCTYPE to tell us what html version we are using
  - A <head>, <title>, and <body>
  - elements or “tags”: <p>A paragraph tag</p>
  - attributes: <p class=“golden”>*class* is an attribute on *p*</p>
    - some attributes are mandatory, e.g. the script tag requires a *type:* <script type=“text/javascript”></script>
  - Text nodes (e.g. the text we see in a paragraph)
  - There may be embedded scripts or styles (within <script> and <style> tags.

XHTML 2 of 2

- Rules of the road:
  - elements must be closed <p> closing tag → </p>, <br /> ← also closed
  - attributes must be quoted <p class="shinyGem">is correct</p>
  - elements and attribute names must be lower case
    <p>is correct</p>, <P> is not, <p class="LOVELY">is correct</p>, <p CLASS="isnot">...
  - nesting must be correct <strong><em>is correct</em></strong>
  - DOCTYPE, head, title, and body are mandatory
  - extra white space is ignored by the browser
    ten spaces are the same as one
- **Validate your code.**

CSS

- Cascading Style Sheets – we will focus on CSS 2 (and some CSS 3)
- A CSS document is a list of rules that apply styles to HTML elements
- Consists of
  - Selectors
    - quite literally, things we select for styling
    - can select on tags, IDs, classes, pseudo-classes, and combinations thereof
    - can select children, descendants, parents, ancestors, etc. of an element
    - See http://www.w3.org/TR/CSS2/selector.html
    - See http://css.maxdesign.com.au/selectutorial/
  - Property/value pairs in declaration blocks

CSS Example

```
#footer {
  border-top: 1px solid black;
  padding: 0.5em 1em 1em 1em;
  color: #333;
  background-color: #eee;
}
#footer p {
  margin: 0 0 1em 0;
  padding: 0;
}
#footer p a:hover {
  font-weight: bold;
  font-style: italic;
}
```

```
<div id=“footer”>
  <p>
    <a
href=“somepage.html”>
      This link will be
      styled on a mouse
      hover
    </a>
  </p>
</div>
```

Declaring Styles

- Where to put them? Three ways to do it:
  1. external stylesheets
     - this is best – your stylesheets will be cached by the browser and can be referenced by all your pages
     - ex: `<link href=“mystyle.css" rel="stylesheet" type="text/css"/>`
  2. embedded (in-page) styles
     - useful for a one-off style on a specific page
     - ex: `<head><style type="text/css">…styles go here…</style></head>`
  3. inline styles
     - Good for testing, and *sometimes* helpful when generating sites
     - Generally avoid
     - ex: `<p style="font-weight: bold;">…</p>`

Selectors

HTML like this	*might* be selected like this
ID selectors	
`<div id=“header”></div>`	`#header {}`
`<h2 id=“mainTitle”></h2>`	`h2#mainTitle {}`
class selectors	
`<p class=“green”></p>`	`.green {}`
`<li class=“selected”></li>`	`li.selected {}`
element selectors	
`<p class=“green”></p>`	`p {}`
`<h1></h1>`	`h1 {}`
`<p><strong>hi</strong></p>`	`p strong {}`

See http://www.w3.org/TR/CSS2/selector.html

IDs and Classes

- An "id" is a *unique* identifier for an element
  - For example, <div id="footer"></div>
  - An id of a specific name should only exist once in a document (it must be unique)
  - Ids are useful for naming important blocks
  - *Aside: "footer" actually has its own element defined in HTML5 <footer></footer>*
- A "class" is a *non-unique* identifier for an element
  - For example, <div class="rightCallOut"></div>
  - A class can be placed on many elements

Precedence and the Cascade

- IDs trump classes (by a lot)
- More specific trumps less specific
  - You can set general rules and then override with a class
  - ex:

    p.someclass (more specific) trumps a simple p (less specific)
- Style rules are read in order top to bottom.
  If two style rules call the same selector,
  - Later styles trump earlier styles
  - Inline styles trump embedded styles
  - Embedded styles trump externally declared styles
  - (So again, more specific trumps less specific)

Precedence Example

Given the following HTML and CSS

```
<ul class="menu">
      <li>item 1</li>
      <li>item 2</li>
      <li>item 3</li>
</ul>
```

```
.menu li {
    color: blue;
}

li {
   color: green;
}
```

Will the list elements be rendered in blue or green?

CSS Sizes

- Sizes of fonts, margins, borders, and so forth can be declared using the following units:
  - em – ems are the size of an “M” and scale relative to your font. Very handy. – Why an M?
  - px – pixels … for when you need precision (e.g. for a fixed-width layout)
  - % – percent (e.g. 90%)

  - there are others, but let’s avoid them for now.
- Always put units on sizes
  - with the exception of zero which does not require a unit

CSS Colors

- Red, Green, Blue values used in additive color are commonly represented by hex values, one byte of information per color:

#000000 (black) to #FFFFFF (white)

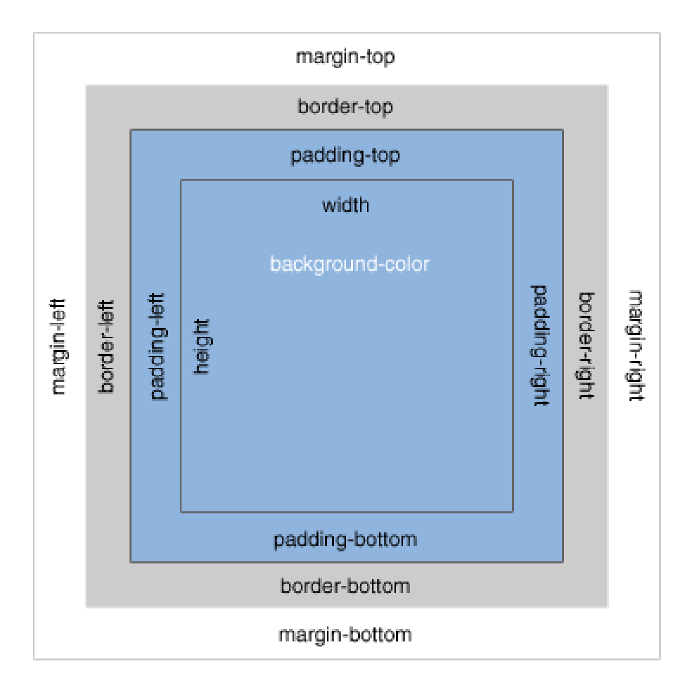
16^6 = 16 million colors (16.8 million, really)

For example, solid red is:

Red	Green	Blue
FF	00	00

Q: How many values of Red are possible?

CSS Box Model



[Untitled graphic of the CSS Box Model, Bitmap].
Retrieved September 10 from
http://www.mandalatv.net/itp/drivebys/css/

CSS Shorthand

- Colors:
  - If Red is #FF0000, we can write shorthand as #F00
- Margins, padding, borders, etc:
  - Properties that have top, right, bottom, and left values can be shorthanded with a single value for all or in various combinations. For this class, either use a single value or write the whole thing out. You can also specify specific top, right, bottom, and left properties
  - Examples:
    - margin: 20px;
    - margin: 1em 2em 3em 2em; /* clockwise from top */
    - margin-top: 200px;

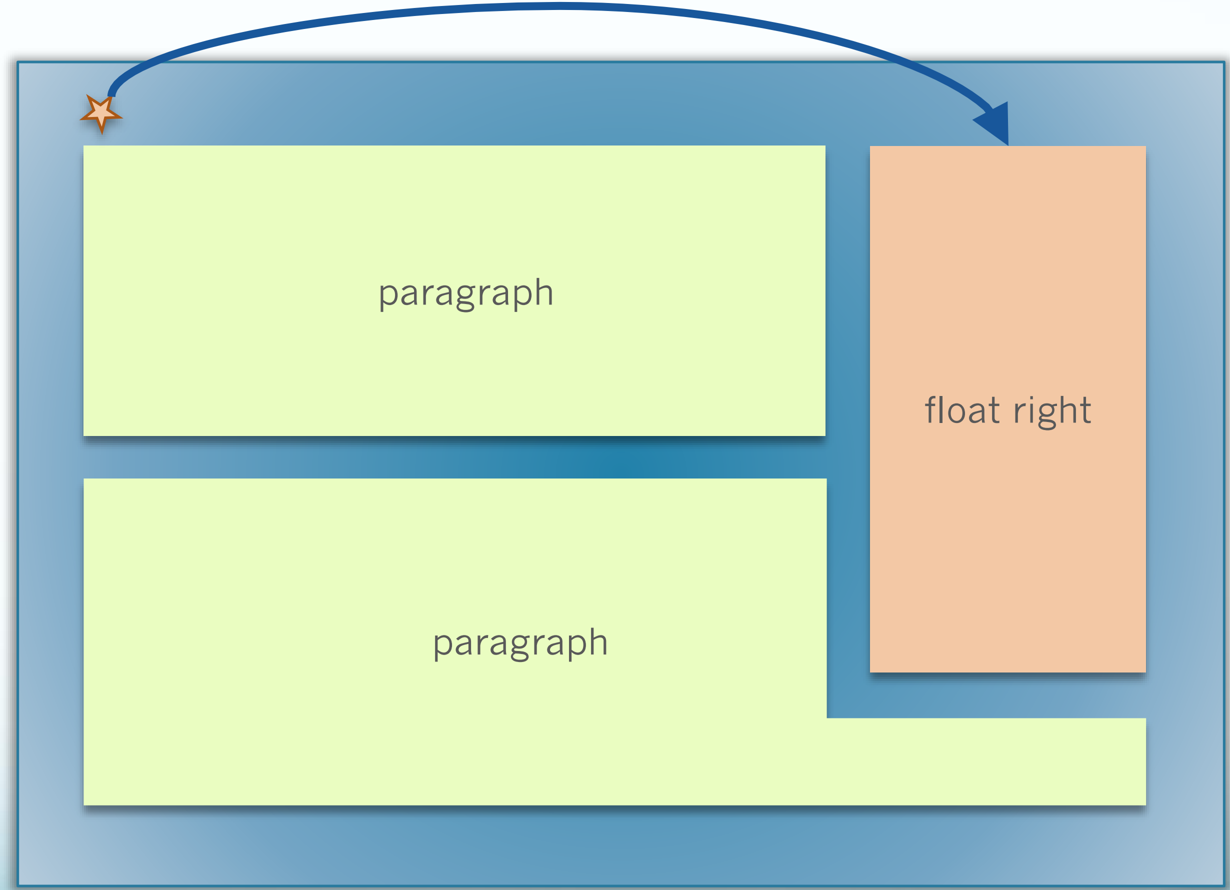
Floating Elements

- An element may be floated left or right (or not at all, which is default)
- Text / elements will flow to the left around items floated to the right. Text will flow to the right around items floated left.

> **“A pull quote is a good example.”**

- The “clear” property allows you to force the flow to break on the left side, right side, or both.

```
.pullQuote {
  float: right;
  clear: right;
}
```


paragraph
float right
paragraph

Where to go for help…

- Search the web! (A search engine is your friend.)
- Try W3Schools:
  - Particularly for its references
  - http://www.w3schools.com/
  - Try other sites too
- View page source! Look at CSS files.
  - The web is open – learn from others

Code Style 1 of 5

- INDENT YOUR WORK. Carefully. Correctly.
  - Many IDEs will do this for you – but you may not use them in this class.
  - Sublime Text has plugins that can help … but many are imperfect, and we won't be using them (much) in this class. DO take advantage of the auto-indent features in good text editors – like VS Code, Atom, Brackets & Sublime
  - Why? To make your life and the lives of those you work with (or who inherit your work) easier.
- Indents should be 2 or 3 *spaces* (not tabs)
- Use white space and comments for readability
- Don't allow single lines to get too long (but for the purposes of this class, we will be flexible).

Code Style 2 of 5

Correct example:

```
<div id=“navigation”>
  <h3>Latest News</h3>
  <ul>
    <li>Item 1</li>
    <li>Item 2</li>
  </ul>
</div>
```

Code Style 3 of 5

Incorrect example (though syntactically correct):

```
  <div
id=  “navigation”>
<h3>
Latest
          News</h3>
<ul><li>Item 1</li><li>
Item 2</li></ul>
                            </div>
```

Code Style 4 of 5

Correct CSS example:

```
ul#header li {
  float: left;
  width: 100px;
  background-color: #F0EEE5;
  color: black;
}

/* highlight the selected tab */
ul#header li.selected {
  background-color: #000;
  color: white;
}
```

Code Style 5 of 5

- In-page CSS is sometimes written like this:

```
p {width: 500px; padding: 0;}
```

- For this class, style all CSS as if it were in an external CSS file, e.g.

```
p {
  width: 500px;
  padding: 0;
}
```

Prior to lab 2 …

- Look over the XHTML tutorial at http://www.w3schools.com/html/html_xhtml.asp

– This covers the basic rules of XHTML which we will be using during Monday's lab.

- Visit W3Schools XHTML tag reference, listed by function http://www.w3schools.com/tags/ref_byfunc.asp
  - look through the tag listing
  - ignore elements marked “deprecated” – they are not to be used
- Look also at the W3Schools CSS reference https://www.w3schools.com/cssref/

Software Overview

- Install Editor
  - Brackets,
  - (alternatives=Sublime or Atom or MS Visual Studio Code, or, or, …)
  - NO IDEs!
- Examine browser tools
  - Chrome
  - Chrome Developer tools

List of Figures