

Web Programming Step by Step

Lecture 2

HTML/CSS Basics

Reading: Ch. 2, 3.1

Except where otherwise noted, the contents of this presentation are Copyright 2009 Marty Stepp and Jessica Miller.



Basic HTML and Elements

- 2.1: Basic HTML
- 2.2: More HTML Elements
- 2.3: Web Standards

Hypertext Markup Language (HTML) (2.1.1)

- describes the *content* and structure of information on a web page
 - not the same as the *presentation* (appearance on screen)
- surrounds text content with opening and closing **tags**
- each tag's name is called an **element**
 - syntax: `<element> content </element>`
 - example: `<p>This is a paragraph</p>`
- most whitespace is insignificant in HTML (ignored or collapsed to a single space)
- we will use a stricter, more standard version called **XHTML**

Structure of an XHTML page (2.1.2)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    information about the page
  </head>

  <body>
    page contents
  </body>
</html>
```

HTML

- the **header** describes the page and the **body** contains the page's contents
- an HTML page is saved into a file ending with extension `.html`

Page title: <title>

describes the title of the web page

```
<title>Chapter 2: HTML Basics</title>
```

HTML

- placed within the head of the page
- displayed in the web browser's title bar and when bookmarking the page

Paragraph: <p> (2.1.3)

paragraphs of text (block)

```
<p>You're not your job.  
You're not how much money you have in the bank.  
You're not the car you drive. You're not the contents  
of your wallet. You're not your khakis. You're  
the all-singing, all-dancing crap of the world.</p>
```

HTML

You're not your job. You're not how much money you have in the bank. You're not the car you drive. You're not the contents of your wallet. You're not your khakis. You're the all-singing, all-dancing crap of the world.

output

- placed within the body of the page
- [more paragraph examples](#)

Headings: `<h1>`, `<h2>`, ..., `<h6>`

headings to separate major areas of the page (block)

```
<h1>University of Whoville</h1>  
<h2>Department of Computer Science</h2>  
<h3>Sponsored by Micro$oft</h3>
```

HTML

University of Whoville

Department of Computer Science

Sponsored by Micro\$oft

output

- [More heading examples](#)

Horizontal rule: `<hr>`

a horizontal line to visually separate sections of a page (block)

```
<p>First paragraph</p>  
<hr />  
<p>Second paragraph</p>
```

HTML

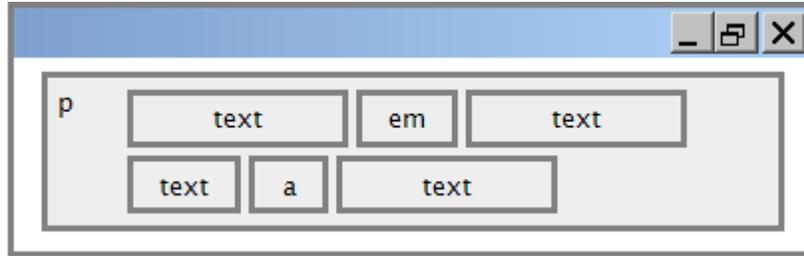
First paragraph

Second paragraph

output

- should be immediately closed with `/>`

Block and inline elements (explanation)



- **block** elements contain an entire large region of content
 - examples: paragraphs, lists, table cells
 - the browser places a margin of whitespace between block elements for separation
- **inline** elements affect a small amount of content
 - examples: bold text, code fragments, images
 - the browser allows many inline elements to appear on the same line
 - must be nested inside a block element

More about HTML tags

- some tags can contain additional information called **attributes**
 - syntax: `<element attribute="value" attribute="value"> content </element>`
 - example: `Next page`
- some tags don't contain content; can be opened and closed in one tag
 - syntax: `<element attribute="value" attribute="value" />`
 - example: `<hr />`
 - example: ``

Links: <a> (2.1.4)

links, or "anchors", to other pages (inline)

```
<p>  
  Search  
  <a href="http://www.google.com/">Google</a> or our  
  <a href="lectures.html">Lecture Notes</a>.  
</p>
```

HTML

Search [Google](http://www.google.com/) or our [Lecture Notes](#).

output

- uses the `href` attribute to specify the destination URL
 - can be **absolute** (to another web site) or **relative** (to another page on this site)
- anchors are inline elements; must be placed in a block element such as `p` or `h1`

Images:

inserts a graphical image into the page (inline)

```

```

HTML



output

- the `src` attribute specifies the image URL
- XHTML also requires an `alt` attribute describing the image

More about images

```
<a href="http://theonering.net/">
  
</a>
```

HTML



output

- if placed inside an a anchor, the image will become a link
- the title attribute specifies an optional tooltip

Line break: `
`

forces a line break in the middle of a block element (inline)

```
<p>Teddy said it was a hat, <br /> So I put it on.</p>
<p>Now Daddy's sayin', <br /> Where
the heck's the toilet plunger gone?</p>
```

HTML

Teddy said it was a hat,
So I put it on.

Now Daddy's sayin',
Where the heck's the toilet plunger gone?

output

- br should be immediately closed with `</>`
- br should not be used to separate paragraphs or used multiple times in a row to create spacing

Comments: `<!-- ... -->`

comments to document your HTML file or "comment out" text

```
<!-- My web page, by Suzy Student  
CSE 190 D, Spring 2048 -->  
<p>CSE courses are <!-- NOT --> a lot of fun!</p>
```

HTML

CSE courses are a lot of fun!

output

- many web pages are not thoroughly commented (or at all)
- comments are still useful for disabling sections of a page
- comments cannot be nested and cannot contain a `--`

Phrase elements : ``, ``

em: emphasized text (usually rendered in italic)

strong: strongly emphasized text (usually rendered in bold)

```
<p>  
HTML is <em>really</em>,  
<strong>REALLY</strong> fun!  
</p>
```

HTML

HTML is *really*, **REALLY** fun!

output

- as usual, the tags must be properly nested for a valid page

Nesting tags

Bad:

```
<p>
  HTML is <em>really,
  <strong>REALLY</em> lots of</strong> fun!
</p>
```

HTML

- tags must be correctly nested
 - (a closing tag must match the most recently opened tag)
- the browser may render it correctly anyway, but it is invalid XHTML

Unordered list: ``, `` (2.2.1)

ul represents a bulleted list of items (block)
li represents a single item within the list (block)

```
<ul>
  <li>No shoes</li>
  <li>No shirt</li>
  <li>No problem!</li>
</ul>
```

HTML

- No shoes
- No shirt
- No problem!

output

More about unordered lists

- a list can contain other lists:

```
<ul>
  <li>Simpsons:
    <ul>
      <li>Homer</li>
      <li>Marge</li>
    </ul>
  </li>
  <li>Family Guy:
    <ul>
      <li>Peter</li>
      <li>Lois</li>
    </ul>
  </li>
</ul>
```

HTML

- Simpsons:
 - Homer
 - Marge
- Family Guy:
 - Peter
 - Lois

output

Ordered list:

ol represents a numbered list of items (block)

```
<p>RIAA business model:</p>
<ol>
  <li>Sue customers for copying music</li>
  <li>??</li>
  <li>Profit!</li>
</ol>
```

HTML

RIAA business model:

1. Sue customers for copying music
2. ???
3. Profit!

output

- we can make lists with letters or Roman numerals using CSS (later)

Web Standards

- 2.1: Basic HTML
- 2.2: More HTML Elements
- **2.3: Web Standards**

Web Standards (2.3.1)

- It is important to write proper XHTML code and follow proper syntax.
- Why use XHTML and web standards?
 - more rigid and structured language
 - more interoperable across different web browsers
 - more likely that our pages will display correctly in the future
 - can be interchanged with other XML data: [SVG](#) (graphics), [MathML](#), [MusicML](#), etc.

W3C XHTML Validator (2.3.2)

```
<p>  
<a href="http://validator.w3.org/check/referer">  
    
</a>  
</p>
```

HTML



output

- validator.w3.org
- checks your HTML code to make sure it follows the official XHTML syntax
- more picky than the browser, which may render bad XHTML correctly

3.1: Basic CSS

- 3.1: Basic CSS
- 3.2: More CSS

The bad way to produce styles

```
<p>
  <font face="Arial">Welcome to Greasy Joe's.</font>
  You will <b>never</b>, <i>ever</i>, <u>EVER</u> beat
  <font size="+4" color="red">OUR</font> prices!
</p>
```

HTML

Welcome to Greasy Joe's. You will **never**, *ever*, EVER beat **OUR** prices!

output

- tags such as `b`, `i`, `u`, and `font` are discouraged in strict XHTML
 - Why is this bad?

Cascading Style Sheets (CSS): `<link>` (3.1.2)

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

HTML

```
<link href="style.css" type="text/css" rel="stylesheet" />
```

HTML

- **CSS** describes the appearance and layout of information on a web page
 - (as opposed to HTML, which describes the content of the page)
- can be embedded in HTML or placed into separate `.css` file (preferred)

Basic CSS rule syntax (3.1.1)

```
selector {  
  property: value;  
  property: value;  
  ...  
  property: value;  
}
```

CSS

```
p {  
  font-family: sans-serif;  
  color: red;  
}
```

CSS

- a CSS file consists of one or more **rules**
- each rule starts with a **selector** that specifies an HTML element(s) and then applies style **properties** to them
 - a selector of * selects all elements

CSS properties for colors (3.1.3)

```
p {  
  color: red;  
  background-color: yellow;  
}
```

CSS

This paragraph uses the style above.

output

property	description
color	color of the element's text
background-color	color that will appear behind the element

Specifying colors

```
p { color: red; }  
h2 { color: rgb(128, 0, 196); }  
h4 { color: #FF8800; }
```

CSS

This paragraph uses the first style above.

This h2 uses the second style above.

This h4 uses the third style above.

output

- color names: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, (white), yellow
- RGB codes: red, green, and blue values from 0 (none) to 255 (full)
- hex codes: RGB values in base-16 from 00 (0, none) to FF (255, full)

CSS properties for fonts (3.1.5)

property	description
font-family	which font will be used
font-size	how large the letters will be drawn
font-style	used to enable/disable italic style
font-weight	used to enable/disable bold style
Complete list of font properties	

font-family

```
p {  
  font-family: Georgia;  
}  
h2 {  
  font-family: "Courier New";  
}
```

CSS

This paragraph uses the first style above.

This h2 uses the second style above.

output

- enclose multi-word font names in quotes

More about font-family

```
p {  
  font-family: Garamond, "Times New Roman", serif;  
}
```

CSS

This paragraph uses the above style.

output

- can specify multiple fonts from highest to lowest priority
- **generic font names:**
serif, sans-serif, cursive, fantasy, monospace
- if the first font is not found on the user's computer, the next is tried
- generally should specify similar fonts
- placing a generic font name at the end of your font-family value ensures that every computer will use a valid font

font-size

```
p {  
  font-size: 14pt;  
}
```

CSS

This paragraph uses the style above.

output

- units: pixels (px) vs. point (pt) vs. m-size (em)
16px, 16pt, 1.16em
- vague font sizes: xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger
- percentage font sizes, e.g.: 90%, 120%
- pt specifies number of *point*, where a point is 1/72 of an inch onscreen
- px specifies a number of pixels on the screen
- em specifies number of *m-widths*, where 1 em is equal to the font's current size

font-weight, font-style

```
p {  
  font-weight: bold;  
  font-style: italic;  
}
```

CSS

This paragraph uses the style above.

output

- either of the above can be set to normal to turn them off (e.g. headings)