CSE 154

LECTURE 3: YET MORE HTML AND CSS

HTML Character Entities

a way of representing any Unicode character within a web page

character(s)	entity
< >	< >
é è ñ	é è ñ
™ ©	™ ©
π δ Δ	π δ Δ
И	И
" &	" &

- Complete list of HTML entities

Nesting tags

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

- 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

HTML

- (how would we get the above effect in a valid way?)

Unordered list: <ul>,

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

- No shoes
- No shirt
- No problem!

output

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

More about unordered lists

```
<ul>
   <li>Harry Potter characters:
      <ul>
         <li>Harry Potter</li>
         <li>Hermione</li>
         <li>Ron</li>
      </ul>
   </li>
   <li>LOTR characters:
      <ul>
         <li>Frodo</li>
         <li>Bilbo</li>
         <li>Sam</li>
      </ul>
   </li>
</ul>
```

HTML

- Harry Potter characters:
  - Harry Potter
  - Hermione
  - Ron
- LOTR characters:
  - Frodo
  - Bilbo
  - Sam

output

Ordered list

```
<p>Apple business model:</p>
<ol>
<li>Beat Microsoft</li>
<li>Beat Google</li>
<li>Conquer the world!</li>
</ol>
```

HTML

Apple business model:

1. Beat Microsoft
2. Beat Google
3. Conquer the world

output

- **ol** represents a numbered list of items
- we can make lists with letters or Roman numerals using CSS (later)

Definition list <dl>, <dt>, <dd>

```HTML
<dl>
   <dt>newbie</dt> <dd>one who does not have mad skills</dd>
   <dt>own</dt> <dd>to soundly defeat (e.g. I owned that newbie!)</dd>
   <dt>frag</dt> <dd>a kill in a shooting game</dd>
</dl>
```

HTML

```output
newbie
    one who does not have mad skills
own
    to soundly defeat (e.g. I owned that newbie!)
frag
    a kill in a shooting game
```

output

- **dl** represents a list of definitions of terms
- **dt** represents each term, and **dd** its definition

Web page metadata: <meta>

information about your page (for a browser, search engine, etc.)

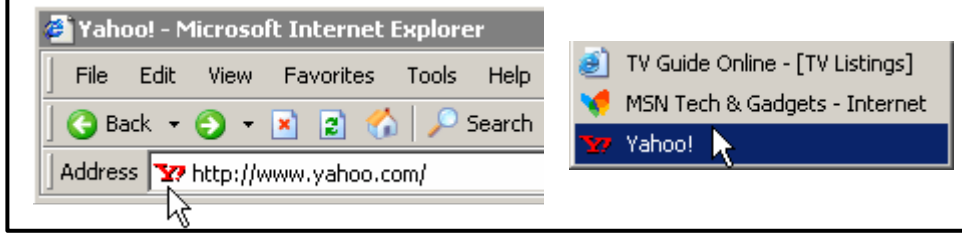
```
<meta charset="utf-8" />
<meta name="description“ content="Authors' web site for Building Java Programs." />
<meta name="keywords" content="java, textbook" />
```
HTML

- placed **in the head** section of your HTML page
- meta tags often have both the name and content attributes
  - some meta tags use the http-equiv attribute instead of name
  - the meta tag with charset attribute indicates language/character encodings
- using a meta tag Content-Type stops validator "tentatively valid" warnings

Favorites icon ("favicon")

```
<link href="filename" type="MIME type" rel="shortcut icon" />
```
HTML

```
<link href="yahoo.gif" type="image/gif" rel="shortcut icon" />
```
HTML



output

- the link tag, placed in the head section, attaches another file to the page
  - in this case, an icon to be placed in the browser title bar and bookmarks
- IE6: Doesn't work; must put a file favicon.ico in the root of the web server

Web Standards

It is important to write proper HTML code and follow proper syntax.

Why use valid HTML 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 HTML Validator

```
<p>
        <a href="http://validator.w3.org/check/referer">
                <img src="http://webster.cs.washington.edu/w3c-html.png" alt="Validate" />
        </a>
</p>
```

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

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>
```

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

Embedding style sheets: <style> (BAD!)

```
<head>
    <style type="text/css">
        p { font-family: sans-serif; color: red; }
        h2 { background-color: yellow; }
    </style>
</head>
```

HTML

- CSS code can be embedded within the head of an HTML page
- this is *bad style*; DO NOT DO THIS (why?)

Inline styles: the style attribute (BAD!)

```
<p style="font-family: sans-serif; color: red;">
This is a paragraph</p>
```

HTML

This is a paragraph

output

- higher precedence than embedded or linked styles
- used for one-time overrides and styling a particular element
- this is *bad style*; DO NOT DO THIS (why?)

Cascading Style Sheets (CSS): <link>

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

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

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

- a CSS file consists of one or more rules
- a rule's selector specifies HTML element(s) and applies style properties
- a selector of * selects all elements

CSS comments: /* ... */

```css
/* This is a comment.
   It can span many lines in the CSS file. */
p {
    color: red;
    background-color: aqua;
}
```

CSS

- CSS (like HTML) is usually not commented as much as code such as Java
- the // single-line comment style is NOT supported in CSS
- the <!-- ... --> HTML comment style is also NOT supported in CSS

W3C CSS Validator

```
<p>
   <a href="http://jigsaw.w3.org/css-validator/check/referer">
     <img src="http://jigsaw.w3.org/css-validator/images/vcss"
       alt="Valid CSS!" /></a>
</p>
```

HTML

output

- jigsaw.w3.org/css-validator/
- checks your CSS to make sure it meets the official CSS specifications
- more picky than the web browser, which may render malformed CSS correctly

CSS properties for fonts

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-size

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

This paragraph uses the style above.

- 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%

font-family

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

This paragraph uses the first style above.

This h2 uses the second style above.

- enclose multi-word font names in quotes

More about font-family

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

This paragraph uses the above style.

- can specify multiple fonts from highest to lowest priority
- generic font names:

  serif, sans-serif, cursive, fantasy, monospace

font-weight, font-style

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

This paragraph uses the style above.

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

Grouping styles

```
p, h1, h2 {
color: green;
}
h2 {
background-color: yellow;
}
```

This paragraph uses the above style.

This h2 uses the above styles.

output

- A style can select multiple elements separated by commas
- The individual elements can also have their own styles

The list-style-type property

```css
ol { list-style-type: lower-roman; }
```

CSS

Possible values:

i. `none` : No marker

ii. `disc` (default), `circle, square`

iii. `Decimal`: 1, 2, 3, etc.

iv. `decimal-leading-zero`: 01, 02, 03, etc.

v. `lower-roman`: i, ii, iii, iv, v, etc.

vi. `upper-roman`: I, II, III, IV, V, etc.

vii. `lower-alpha`: a, b, c, d, e, etc.

viii. `upper-alpha`: A, B, C, D, E, etc.

x. `lower-greek`: alpha, beta, gamma, etc.

others: hebrew, armenian, georgian, cjk-ideographic, hiragana…

The visibility property

```css
p.secret {
  visibility: hidden;
}
```

CSS

output

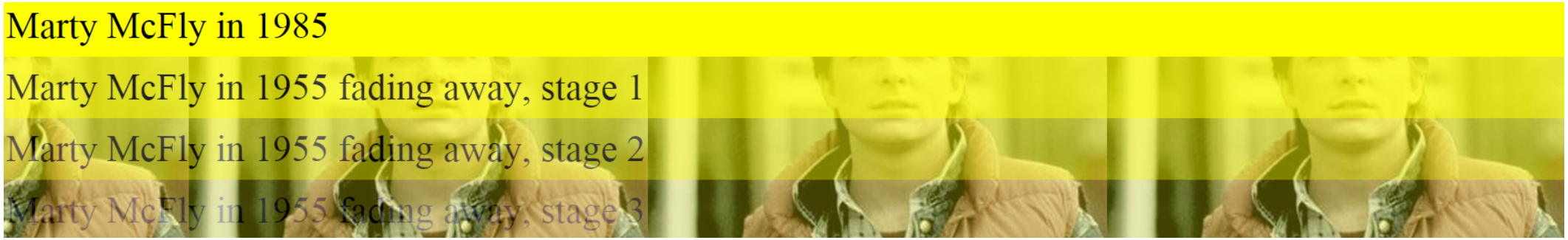
property	description
visibility	sets whether an element should be shown onscreen; can be visible (default) or hidden

- `hidden` elements will still take up space onscreen, but will not be shown
  - to make it not take up any space, set `display` to `none` instead
- can be used to show/hide dynamic HTML content on the page in response to events

The opacity property

```
body  { background-image: url("images/marty-mcfly.jpg");
background-repeat: repeat; }
p  { background-color: yellow;}
p.mcfly1 { opacity: 0.75; }
p.mcfly2 { opacity: 0.50; }
p.mcfly3 { opacity: 0.25; }
```

CSS

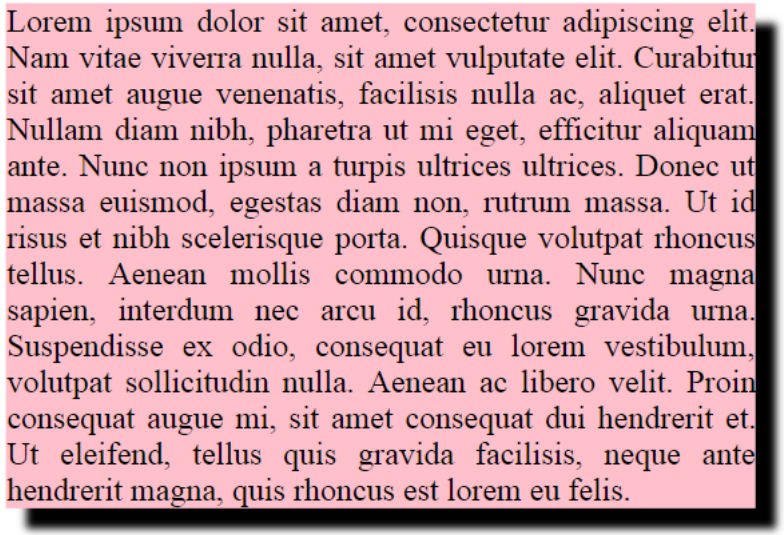


property	description
opacity	how not-transparent the element is; value ranges from 1.0 (opaque) to 0.0 (transparent)

box-shadow

```
box-shadow: h-shadow v-shadow blur;
```
CSS

```
box-shadow: 10px 10px 5px;
```
CSS



Styles that conflict

```
p, h1, h2 { color: blue; font-style: italic; }
h2 { color: red; background-color: yellow; }
```

CSS

This paragraph uses the first style above.

This heading uses both styles above.

output

- when two styles set conflicting values for the same property, the latter style takes precedence

Inheriting styles (explanation)

```
body { font-family: sans-serif; background-color: yellow; }
p { color: red; background-color: aqua; }
a { text-decoration: underline; }
h2 { font-weight: bold; text-align: center; }
```
CSS

This is a heading

A styled paragraph. Previous slides are available on the website.

- A bulleted list

output

- when multiple styles apply to an element, they are inherited
- a more tightly matching rule can override a more general inherited rule
- not all properties are inherited (notice link's color above)