IERG4210 Web Programming and Security

User Interface Design
HTML, CSS, and Templating

Dr. Adonis FUNG
Information Engineering, CUHK
Product Security Engineering, Yahoo!
phfung@ie.cuhk.edu.hk

Agenda

- Client-side Languages for User Interface (UI) Design
  - Structure and Content - *HTML*
  - Presentation - *Cascading Style Sheet (CSS)*
  - Behavior - *Javascript (JS)*
- The Best Practices and their Benefits
- Implementations and Examples
- Templating

Ref: Google Code University - HTML, CSS, and Javascript from the Ground Up

How the UI code is/was organized

- Since 1996 - HTML, CSS, Javascript intermixed
- Since 2002 - Separation of presentation from content
- Since 2005 - Clean Separation; More Javascript than static HTML (thanks to the era of AJAX)

(Demo) View the Source Code of this page
Ref: http://www.youtube.com/watch?v=6oO1CJqh8IM

The Best Practices (1/3)

Separation of Content, Presentation and Behavior Code

- Accessibility - Clean Semantic HTML is good for non-visual browsers and crawlers (Search Engine Optimization or *SEO*)
- Portability - A new CSS stylesheet presents the same content in a different way (e.g. mobile webpage)
- Maintainability - CSS by designers, HTML and Javascript by programmers
- Reduced Latency - Separated files of CSS and JS can be cached in browsers and reused across pages

Ref: http://www.youtube.com/watch?v=6oO1CJqh8IM

The Best Practices (2/3)

Graceful Degradation / Progressive Enhancement

- Legacy Browsers may not support new features like HTML 5
- Users may disable CSS and Javascript
- i.e. Make your webpages functional whenever possible

Don't Ignore Errors

- 404 is BAD! Redirect legacy hyperlinks to new pages
- Javascript errors can prohibit page load

Ref: http://www.youtube.com/watch?v=6oO1CJqh8IM

The Best Practices (3/3)

Naming Convention of public URLs for Search Engine Optimization (SEO)

- Keep it Short
- Use Keywords in Foldernames and Filenames
- Avoid Querystrings (e.g. `?page=11` is meaningless to human)
- Hypenated Filename (e.g. `User-Interface-Design.html`)
- i.e. Readable for both Humans and Bots

Good Example: http://web.mit.edu/is/usability/usability-guidelines.html
Ref: http://www.seomoz.org/blog/11-best-practices-for-urls

HTML

Why bother to teach HTML?

> “Why are we building things all from scratch? Why not HTML editor?”

> “We're unfortunately still like teaching stupid machines how to interpret content!!
> What is our future? stronger AI, ...?”

HTML Basics (1/2)

- Defining the Structure and Content:

```
<!-- Some Comments Here -->
<tagName attributeName="attributeValue">Some Content</tagName>
<!-- Closing a content-less tag -->
<tagName attrName="attrVal" />
```

- Avoid styling in HTML (Best Practice):

```
<!-- Some BAD Examples that look the same: -->
<h1 align="center">Hello World!</h1>
<center><font size="7">Hello World!</font></center>

<!-- Good Example:
     - style can be reused and put in a separate file -->
<style>.centered{text-align:center}</style>
<h1 class="centered">Hello World!</h1>
```


HTML Basics (2/2)

- A simple HTML5 Document:

```
<!DOCTYPE html><!-- placed at top to tell what HTML version -->
<html>
  <!-- head tag contains some meta-info tags -->
  <head>
    <!-- To let the browser knows the correct encoding: -->
    <meta charset="utf-8" />
    <title>IERG4210 HTML5 Hello World!</title>
  </head>

  <!-- body tag contains some content -->
  <body>
    <h1>Hello World!</h1>
  </body>
</html>
```

- (Tool) HTML Validator to check syntax: W3C Validator

Ref: http://www.html-5-tutorial.com/doctype.htm

Semantic HTML

HTML5: Every tag/attribute carries a meaning!

Examples:
- v.s. <header>
- <input type="text" /> v.s. <input type="number">

- No special visual effect, they are the same for browsers; BUT they could mean different things to robots.
- To help Google interpret accurately where to index, e.g. <nav> for menu, <article> (but <header>) for content
- Other HTML5 Semantic Tags: <header>, <footer>, <nav>, <section>, <article>, etc...

Ref: https://developer.mozilla.org/en-US/docs/Sections_and_Outlines_of_an_HTML5_document

HTML Headers - <h1>...<h6>

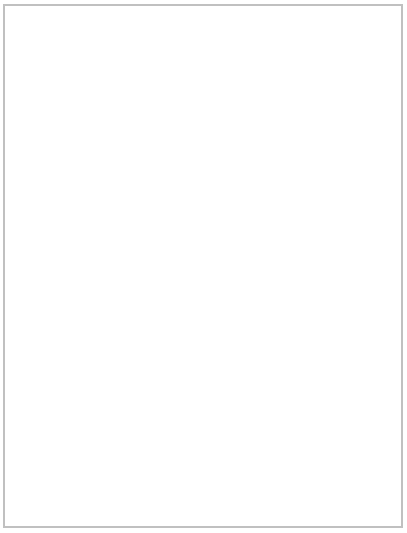
```
<h1>Header 1</h1>
<h2>Header 2</h2>
<h3>Header 3</h3>
...
<h6>Header 6</h6>
```

SEO: <h1> to <h6> are of higher importance than <p>

Live Editor Usage: Edit on LHS, and a "Enter" key triggers update on RHS

HTML Paragraph and Lists - <p>, <ul>,

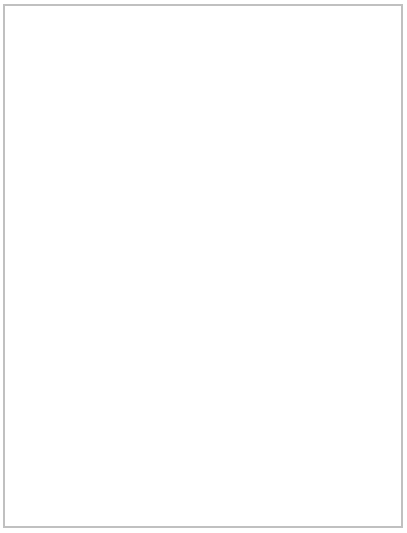
```
<p>Paragraph 1</p>

<p>Unordered List</p>
<ul>
  <li>item 1</li>
  <li>item 2</li>
</ul>

<p>Ordered List</p>
<ol>
  <li>item 1</li>
  <li>item 2</li>
</ol>
```



Note: <p> and <li> both introduce a line break

HTML Strong and Emphasis - ,

```
<p>Below are more semantic!</p>
<strong>Strong</strong>
<em>Emphasis</em>

<p>Below are more stylistic!</p>
<b>bold</b>
<i>italic</i>
```



Note: and are favored according to our best practices

HTML Hyperlink - <a> with Absolute URL

```
<h1>Absolute URLs:</h1>
<a href="http://yahoo.com/">HTTP</a>
<a href="https://yahoo.com/">HTTPS</a>
<!--Follows the Current Protocol:-->
<a href="//yahoo.com/">HTTP/S</a>
```


HTML Hyperlink - <a> with Relative URL

Given the following directory structure:

- incl/
  - cuhk-logo.png
  - test2.html
- test1.html

```
<h1>In test1.html:</h1>
<a href="incl/test2.html">test2.html</a>
```

In test1.html:

test2.html

```
<h1>In incl/test2.html:</h1>
<a href="../test1.html">test1.html</a>
<a
href="/web/tutorials/tutorial01.pdf">T01</a>
```

In incl/test2.html:

test1.html T01

Hover the hyperlinks and see how the relative URLs are translated to full URLs based on the current URL

HTML Image - <img>

Given the same directory structure:

- incl/
  - cuhk-logo.png
  - test2.html
- test1.html

```
<h1>Img in Absolute URL:</h1>
<img
src="https://www.iso.cuhk.edu.hk/english/ima
ges/resource/cuhk-emblem/hor_4c.jpg" />

<h1>Img in Relative URL:</h1>
<img src="incl/cuhk-logo.png" />
<img src="/web/lectures/incl/cuhk-logo.png"
/>
```


HTML Table - <table> (1/2)

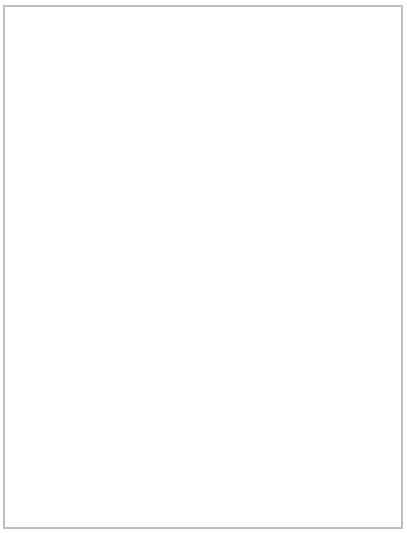
<td> is a general table cell, while <th> stands for a header cell

```
<table>
  <tr><!--table row-->
    <th>First Name</th>
    <th>Last Name</th>
  </tr>
  <tr>
    <td>Alan</td>
    <td>Turing</td>
  </tr>
  <tr>
    <td>Eugene</td>
    <td>Peterson</td>
  </tr>
</table>
```

HTML Table - <table> (2/2)

Multiple rows/columns

```
<table>
  <tr><th>First Name</th>
      <th>Last Name</th></tr>

  <!--Merging the cell in next row-->
  <tr><td rowspan="2">Alan</td>
      <td>Turing</td></tr>
  <tr><td>Tam</td></tr>

  <!--Merging the cell in next column-->
  <tr><td colspan="2">
        Superman!!!!!!!!!!!</td></tr>
</table>
```



HTML + CSS

CSS Basics

- 3 ways to include CSS:

```
<!-- External CSS file can be used across pages -->
<link href="incl/styles.css" rel="stylesheet" type="text/css" />
```

```
<!-- Embedded CSS tag can be used for a specific page-->
<style>p{color:#F00}</style>
```

```
<!-- Inline CSS does not conform to the Best Practice -->
<p>inline <span style="color:#00FF00">CSS</span></p>
```

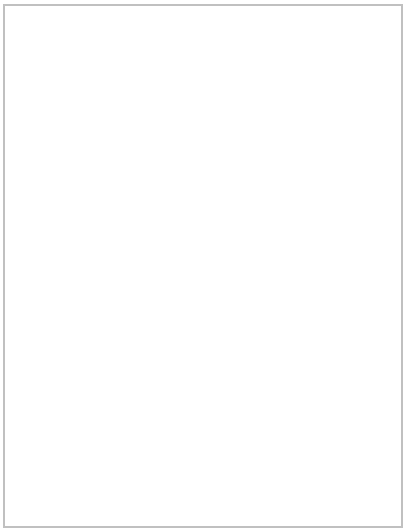
- A CSS rule in External CSS file or Embedded CSS tag:

```
selector1{
  propertyName1:propertyVal1;
  propertyName2:propertyVal2
}
```

Clearly, inline CSS takes only those properties in the braces

CSS Selectors - Rule Precedence - Inheritance

```
<style>
  *,body{color:#00F}
  p{color:#F00}
  p.highlight{background:#FF0}
  p.highlight2{background:#CCC}
  #uniqueId1{font-size:30px;color:#00FF00}
</style>

<h3>inherited the color!</h3>
<p>Oh</p>
<p class="highlight">My</p>
<p class="highlight highlight2">God!</p>
<p id="uniqueId1">overriden the color</p>
```



(Demo) Inspect the output using Browser Developer Tools (e.g., Firebug/Firefox/Chrome)

CSS Selector - Rule Precedence - Specificity

```
<style>
  p{color:#F00}
  p.highlight{background:#FF0}
  #uniq1,#uniq2{color:#00FF00}
  div p.highlight{background:#CCC}
</style>

<p class="highlight">Hello World!</p>
<p class="highlight" id="uniq1">Yo!</p>
<div id="uniq2">
  <p class="highlight">Hello!</p>
</div>
```

Generally, precedence is calculated with a point system: inline > id > class > element

(Midterm/Exam) Rule Precedence

MUST SEE Reference and Examples: http://css-tricks.com/specifics-on-css-specificity/

CSS Selectors for Decendent Elements

Example: CSS Horizontal Menu

```
<style>
  .menu{padding:0;list-style:none}
  .menu li{font-size:9px;display:inline}
</style>
<nav><!-- <nav> is a semantic tag! -->
  <ul class="menu">
    <li><a href="#">Home</a></li>
    <li><a href="#">About Us</a></li>
    <li><a href="#">Contact Us</a></li>
  </ul>
</nav>

<ul>
    <li><a href="#a">Home</a></li>
```

`.menu li` selects every decendent element `<li>` of the element applying `.menu`

CSS Selectors - Link Pseudo-Classes

```
<style>
  .menu a:link{color:#00F}
  .menu a:hover{font-weight:bold}
  .menu a:visited{color:#F00}
</style>
<nav>
  <ul class="menu">
    <li><a href="#a">Home</a></li>
    <li><a href="#b">About Us</a></li>
    <li><a href="#c">Contact Us</a></li>
  </ul>
</nav>
<a href="#">Unaffected!</a>
```

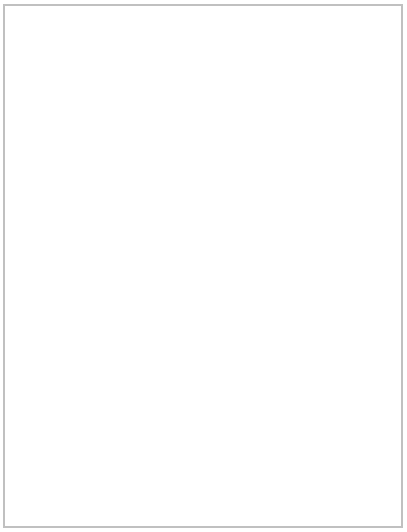

CSS Selectors - User Actions Pseudo-Classes

Example: Mouse-over "MENU" which makes use of `:hover`!

```
<style>
  nav ul{display:none}
  nav:hover ul{display:block}
</style>
<nav>
  <h3>MENU</h3>
  <ul>
    <li><a href="#a">Home</a></li>
    <li><a href="#b">About Us</a></li>
    <li><a href="#c">Contact Us</a></li>
  </ul>
</nav>
Some Content
```

CSS Selectors - A Structural Pseudo-Class

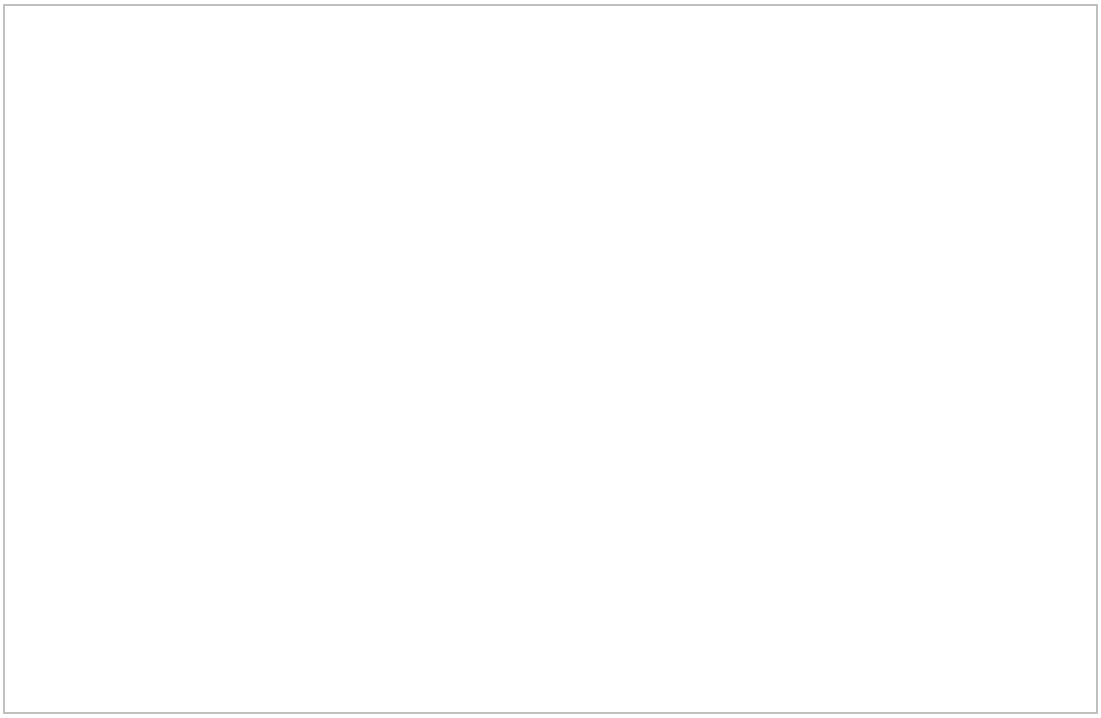
```
<style>
  ul li:nth-child(even){color:#CCC}
  ul li:nth-child(2n){background:#333}
  ul li:nth-child(2n+1){background:#EEE}
</style>

<ul>
  <li>Home</li>
  <li>About Us</li>
  <li>Products</li>
  <li>Contact Us</li>
</ul>
```



n starts at zero and increments by 1 every time

- What will 3n+1 select? (Need a demo?)
- Children list is one-indexed.

More on Selectors



CSS Styles: Font Color, Size and Height

```
<style>
  .para1{color:#0F0;line-height:150%}
  .para2{color:#F00;font-size:150%}
</style>

<p class="para1">Have I not commanded you?
Be strong and courageous. Do not be afraid;
do not be discouraged, for the LORD your God
will be with you wherever you go.</p>

<p class="para2">Have I not commanded you?
Be strong and courageous. Do not be afraid;
do not be discouraged, for the LORD your God
will be with you wherever you go.</p>
```

More: `font-weight:bold`; `font-style:italic`; `text-decoration:underline`

CSS Styles: Text Alignment

```
<style>
  .title{text-align:center}
  .para{text-align:justify;color:#F00}
  .right{text-align:right}
</style>

<h1 class="title">Joshua 1:9</h1>
<p class="para">Have I not commanded you? Be
strong and courageous. Do not be afraid; do
not be discouraged, for the LORD your God
will be with you wherever you go.</p>
<p class="right">Copyright. NIV.</p>
```

CSS Styles: Positioning

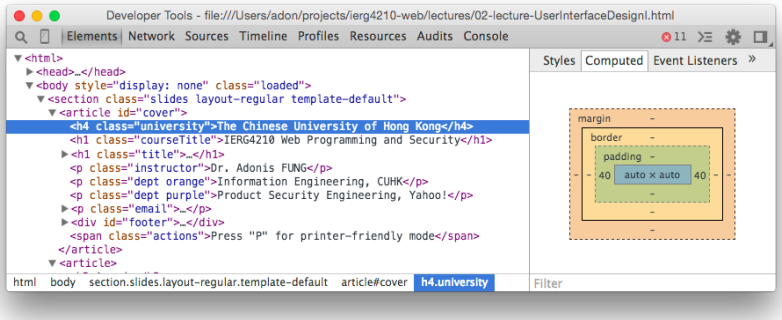
```
<style>
  nav
ul{display:none;position:absolute;margin:-20
px}
  nav:hover ul{display:block}
</style>
<nav>
  <h3>MENU</h3>
  <ul>
    <li><a href="#a">Home</a></li>
    <li><a href="#b">About Us</a></li>
    <li><a href="#c">Contact Us</a></li>
  </ul>
</nav><p>Some Content</p>
```

`position:absolute|relative|fixed|static`

-`fixed` is to avoid being scrolled away

CSS Styles: The Box Model

- The Box Model as displayed in Firebug for the element `ul` :



- position layer: `top`, `right`, `bottom`, `left`
- margin layer: `margin-top`, `margin-right`, `margin-bottom`, `margin-left`
- border layer: `border-top`, `border-right`, `border-bottom`, `border-left`
- padding layer: `padding-top`, `padding-right`, `padding-bottom`, `padding-left`
- Or equiv., `margin:1px 2px 3px 4px;` for top, right, bottom and left direction

CSS Styles: The Box Model (Demo)

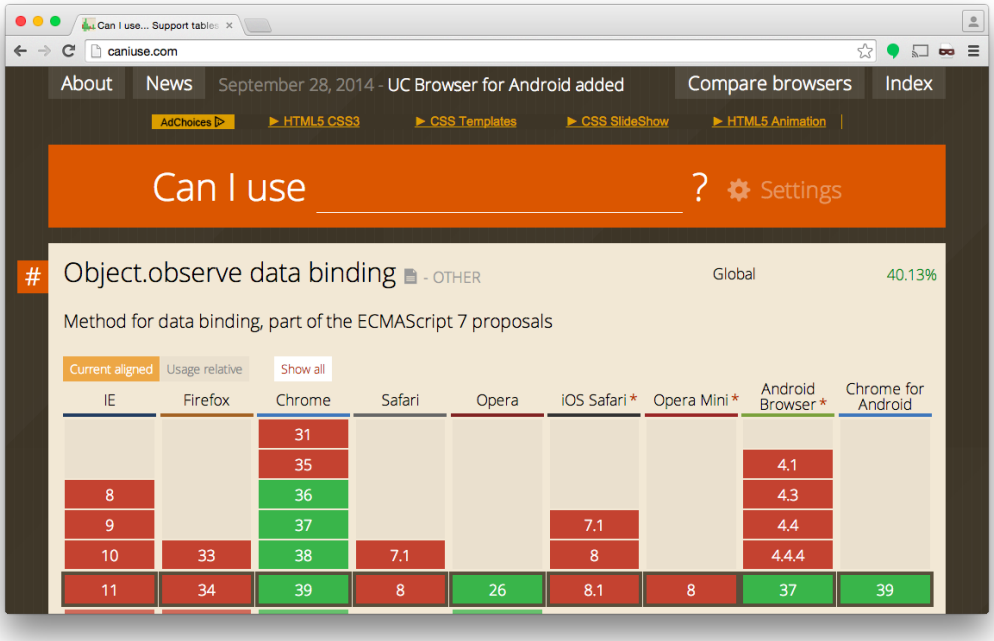
```
<style>
  p.wide{margin:10px;padding:5px}
  p.border,p.wide{border:3px solid #CCC}
  p.LRonly{border-top:0;border-bottom:0}
  p.lifted{margin-top:-50px}
</style>
<p>Content 1</p>
<p class="wide">Content 2</p>
<p class="border">Content 3</p>
<p class="border LRonly">Content 4</p>
<p class="lifted">Content 5</p>
```

Negative Values are accepted.

CSS Styles: Tableless Layout Example

```
<!-- Try resizing the width to 180px -->
<style>
  ul.table{width:240px;height:240px;
    margin:0;padding:0;list-style:none;
    overflow:auto}
  ul.table li{width:70px;height:90px;
    float:left;border:1px solid #CCC}
  .clear{clear:both}
</style>
<ul class="table">
  <li><img src="incl/02-souvenir.jpg" />Product 1</li>
  <li>Product 2</li><li>Product 3</li>
  <li>Product 4</li><li>Product 5</li>
  <li>Product 6</li><li>Product 7</li>
</ul>
<p class="clear">Total: 7</p>
```

Browser Compatibility Issues



More on CSS3

Templating Framework

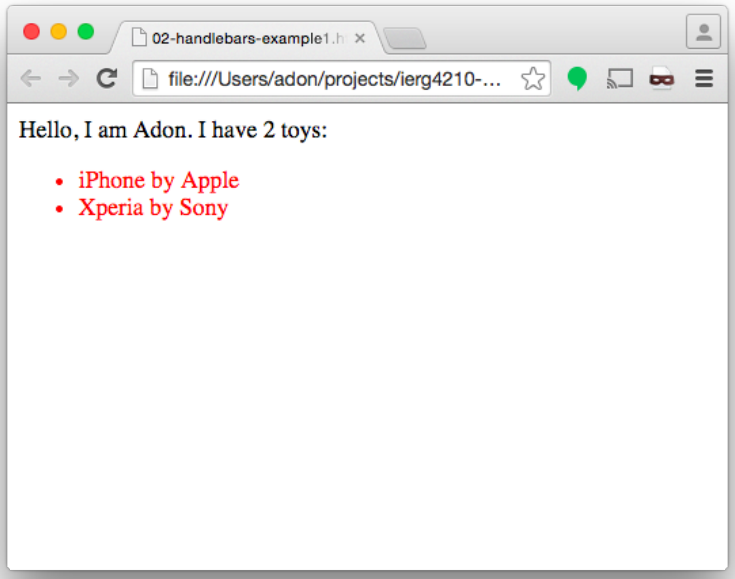
Definition: bind(data in often JSON format, a template in HTML)

- Motivations:
  - **FAST!** when using client-side/JS templating
    - data binding work shifted from server to browser
    - made possible to cache relatively static templates
  - Iterations (data with repeating presentation patterns)
  - Internationalization and localization (i18n)
  - Further separation presentation from content
- Examples: Handlebars, Dust, Angular, React, Mustache, etc...

Handlebars Example

```
<style>li{color:#F00}</style><div id="content"></div>
<script id="tmpl-hello" type="text/x-handlebars-template">
    <p>Hello, I am {{name}}. I have {{toys.length}} toys:</p>
    <ul>{{#toys}}<li>{{model}} by {{make}}</li>{{/toys}}</ul>
</script>
<script src="handlebars.2.0.0.min.js"></script>
<script>
// data in JSON format, possibly fetched over AJAX
var json = {
    "name": "Adon",
    "toys": [  {"model": "iPhone", "make": "Apple"},
               {"model": "Xperia", "make": "Sony"}    ]
};
// compile the template on-the-fly
var tmpl = Handlebars.compile(
        document.getElementById('tmpl-hello').innerHTML);
// bind the data with template, put result back
document.getElementById('content').innerHTML = tmpl(json);
</script>
```

Handlebars Example (Demo)



Demo at jsfiddle

Learn Handlebars

- Core Language Components (inherited from mustache)
  - iterations
  - conditionals
  - partials
- Performance Issues (to be further discussed)
  - Server-side v.s. Client-side data binding
  - Caching templates
  - Pre-compilation v.s. on-the-fly compilation
- Security Issues: (to be later covered)
  - Security: Output Escaping v.s. Unsafe/Raw Output

Some Logistics...

- A quick overview/tutorial on Handlebars: Part 1, Part 2
- Online Quiz ready. Deadline on coming Friday
- Tutorials will start this week
- Assignment Specification for Phase 1
  - to be released on or before Friday