For additional materials, please see http://www.coreservlets.com/. The JavaScript tutorial section contains complete source code for all examples in the entire tutorial series, plus exercises and exercise solutions for each topic.

For customized training related to JavaScript or Java, email hall@coreservlets.com

Marty is also available for consulting and development support.

Taught by lead author of Core Servlets & JSP, co-author of Core JSF (4th Ed), and this tutorial.

Available at public venues, or custom versions can be held on-site at your organization.

Courses developed and taught by Marty Hall
- JavaScript, jQuery, Ext JS, JSF 2.3, PrimeFaces, Java 8 programming,
- Spring Framework, Spring MVC, Android, GWT, custom mix of topics
- Courses available in any state or country.
- Maryland/DC companies can also choose afternoon/evening courses.

Courses developed and taught by coreservlets.com experts (edited by Marty)
- Hadoop, Hibernate/JPA, HTML5, RESTful Web Services

Contact hall@coreservlets.com for details.
Loading Stylesheets and Applying Styles

For additional materials, please see http://www.coreservlets.com. The JavaScript tutorial section contains complete source code for all examples in the entire tutorial series, plus exercises and exercise solutions for each topic.
Overview

• Many JavaScript developers already know at least the basics of Cascading Style Sheets (CSS)
  – If you are one of them, skip this mini-tutorial entirely
    • This covers barebones syntax basics only, is not advanced CSS, and covers only syntax – it does not discuss CSS design strategies

• But some are new to Web development
  – And jQuery is based in part on CSS, as we will see

• Far more details are available online
  – Any of the online CSS tutorials give more details than this very brief introduction

Loading an External Style Sheet

• Loading the style sheet
  `<head>
    <link href="css/styles.css" rel="stylesheet" type="text/css"/>
    ...
  </head>`

• Locating the style sheet
  – The CSS file goes in normal directory, in location referred to by relative URL above
    • For example, in the above example, the style sheet styles.css is in the “css” subfolder of the current directory
Embedding Styles

• Embedding an internal style sheet
  
  ```html
  <head>
  <style type="text/css">
    p { color: blue; }
    .note { font-weight: bold; background-color: red; }
  </style>
  ...
  </head>
  ```

• Listing CSS styles inline
  
  ```html
  <h1 style="color: red; background-color: blue">...</h1>
  ```

Applying Styles to General Elements

• Styles that apply to elements
  - Apply automatically

• Example CSS
  
  ```css
  h2 { color: blue; font-family: sans-serif }
  ```
  
  - All `<h2>` elements automatically in blue non-serif font

• Example HTML
  
  ```html
  <h2>Hello</h2>
  ```
# Applying Styles to Specific Elements

- **Styles that start with “.”**
  - Must be applied with “class” attribute

## Example CSS
```
.warning { color: red; font-weight: bold; font-size: 120% }
```

## Example HTML
```
<p class="warning">...</p>
<span class="warning">...</span>
```

- **Separate multiple classes with spaces**
```
<h2 class="class1 class2 class3">...</h2>
```
**Big Idea**

• **Selectors are what designate the elements the styles apply to**
  
  ```css
  p { color: red; background-color: yellow }
  ```
  
  • All `<p>` elements have foreground red and background yellow
  
  ```css
  .indented { margin-left: 10px }
  ```
  
  • `<blah class="indented">` elements are indented 10 pixels on the left
  
  ```css
  table.colored td { color: blue }
  ```
  
  • `<td>`’s within `<table class="colored">` have blue foreground

• **Only some of the selectors are shown here**
  
  – For complete list, with examples, see
  
  [http://www.w3.org/TR/css3-selectors/](http://www.w3.org/TR/css3-selectors/)

---

**CSS Selectors: Quick Examples**

```css
#some-id {
  Styles that apply to `<h1 id="some-id">`
}

p {
  Styles that apply to `<p>`
}

.note {
  Styles that apply to `<div class="note">`
}

p span {
  Styles that apply to `<p>...<span>...<span>...<p>`
}

h1, h2, td li {
  Styles that apply to `<h1>` and also to `<h2>` and also to `<td>`...`<li>`...`<li>`...`<td>`
}
```
### Basic Selectors

<table>
<thead>
<tr>
<th>Selector</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>element</code></td>
<td>Matches all elements with given tag name. Could be many matches.</td>
<td><code>li {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>p {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches all li elements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches all p elements</td>
</tr>
<tr>
<td><code>#id</code></td>
<td>Matches the element with given id. Matches 0 or 1 elements.</td>
<td><code>#blah {...}</code></td>
</tr>
<tr>
<td><code>class</code></td>
<td>Matches all elements with given CSS style.</td>
<td><code>.important {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches elements with</td>
</tr>
<tr>
<td></td>
<td></td>
<td>`&lt;... class=&quot;important&quot;&gt;</td>
</tr>
<tr>
<td><code>element.class</code></td>
<td>Matches all elements with given tag name that have given class.</td>
<td><code>div.important {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches elements like</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;div class=&quot;important&quot;&gt;</code></td>
</tr>
<tr>
<td><code>element#id</code></td>
<td>Matches the element that has given tag name and given id. Since ids must be unique, you can omit the element name and get same result.</td>
<td><code>form#blah {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches element with</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;form id=&quot;blah&quot;&gt;</code></td>
</tr>
<tr>
<td><code>*</code></td>
<td>Matches all elements in entire page. Particularly useful for nesting and with the :not selector.</td>
<td><code>* {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>div * {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches all elements.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches all elements that are inside divs</td>
</tr>
</tbody>
</table>

### Hierarchical Selectors

<table>
<thead>
<tr>
<th>Selector</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>s1 s2</code></td>
<td>Elements that match selector s2 and are <em>directly or indirectly</em> inside an element that matches selector s1.</td>
<td><code>div.foo span.bar {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches all <code>&lt;span class=&quot;bar&quot;&gt;</code> elements that are <em>somewhere</em> inside <code>&lt;div class=&quot;foo&quot;&gt;</code>.</td>
</tr>
<tr>
<td><code>s1 &gt; s2</code></td>
<td>Elements that match selector s2 and are <em>directly</em> inside an element that matches s1.</td>
<td><code>div.foo &gt; span.bar {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches all <code>&lt;span class=&quot;bar&quot;&gt;</code> elements that are <em>directly</em> inside <code>&lt;div class=&quot;foo&quot;&gt;</code>.</td>
</tr>
<tr>
<td><code>s1, s2</code></td>
<td>Elements that match either selector.</td>
<td><code>ul,ol,dl.foo {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches all ul, ol, and <code>&lt;dl class=&quot;foo&quot;&gt;</code> elements.</td>
</tr>
<tr>
<td><code>s1 + s2</code></td>
<td>Elements that match s2 and are immediately after a sibling element matching s1.</td>
<td><code>label + input {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches all input elements that are immediately after a label element.</td>
</tr>
<tr>
<td><code>s1 ~ s2</code></td>
<td>Elements that match selector s2 and are somewhere after a sibling element matching s1.</td>
<td><code>label ~ input {...}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches all input elements that have a label element somewhere before them at the same nesting level.</td>
</tr>
</tbody>
</table>
### Attribute Selectors

<table>
<thead>
<tr>
<th>Selector</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>s[att]</td>
<td>Elements that match selector s and also contain attribute named att.</td>
<td>div.blah a[name] {...}</td>
</tr>
<tr>
<td>s[att=val]</td>
<td>Elements that match selector s and also contain attribute named att whose value is (exactly) val.</td>
<td>a[href=&quot;#sect2&quot;] {...}</td>
</tr>
<tr>
<td>s[att^=val]</td>
<td>Elements that match selector s and also contain attribute named att whose value starts with val.</td>
<td>a[href^=&quot;#&quot;] {...}</td>
</tr>
<tr>
<td>s[att$=val]</td>
<td>Elements that match selector s and also contain attribute named att whose value ends with val.</td>
<td>a[href$=jquery.com] {...}</td>
</tr>
<tr>
<td>s[att*=val]</td>
<td>Elements that match selector s and also contain attribute named att whose value contains val.</td>
<td>a[href*=jquery.com] {...}</td>
</tr>
<tr>
<td>s[att!=val]</td>
<td>Elements that match selector s and either do not have the specified attribute, or have a different value.</td>
<td>a[href!=#sect2] {...}</td>
</tr>
<tr>
<td>s:not([…])</td>
<td>Elements that match s but do not match attribute specification.</td>
<td>a:not([href^=http]) {...}</td>
</tr>
</tbody>
</table>

### Positional Selectors

<table>
<thead>
<tr>
<th>Selector</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>s:first</td>
<td>First or last match in page.</td>
<td>ul.foo li:first {...}</td>
</tr>
<tr>
<td>s:last</td>
<td>First or last match in page.</td>
<td>ul.foo li:last {...}</td>
</tr>
<tr>
<td>s:eq(n)</td>
<td>The nth match in the page. Count starts at 0.</td>
<td>p:eq(3) {...}</td>
</tr>
<tr>
<td>s:gt(n), s:lt(n)</td>
<td>Elements after/before the nth.</td>
<td>p:gt(3) {...}</td>
</tr>
<tr>
<td>s:even s:odd</td>
<td>Elements that are even or odd numbered elements in the page. 0-based, so first match is even.</td>
<td>tr:even {...}</td>
</tr>
<tr>
<td>s:first-child s:last-child s:only-child</td>
<td>Elements that are the first or last child of their parents, or that have no siblings.</td>
<td>tr:first-child {...}</td>
</tr>
<tr>
<td>s:nth-child(n)</td>
<td>Elements that are the nth child. First child is nth-child(1), not (0)</td>
<td>tr:nth-child(3) {...}</td>
</tr>
<tr>
<td>s:nth-child(even) s:nth-child(odd)</td>
<td>Elements that are even or odd children of their parent. Count starts at 1, so first match is odd.</td>
<td>tr:nth-child(even) {...}</td>
</tr>
<tr>
<td>s:nth-child(xn+y)</td>
<td>Elements matching formula. You list &quot;n&quot; literally. So, 3n means every third. 3n+1 means entry after every third.</td>
<td>tr:nth-child(4n+2) {...}</td>
</tr>
</tbody>
</table>
## Content Filtering Selectors

<table>
<thead>
<tr>
<th>Selector</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>s:contains(text)</td>
<td>Elements that match s and whose body content contains given text.</td>
<td>.foo li:contains(wow) {...}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Matches li elements that have &quot;wow&quot; in their body text and are inside</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;... class=&quot;foo&quot;&gt;</td>
</tr>
<tr>
<td>s:empty</td>
<td>Elements that have no child elements. Body content counts as a child</td>
<td>div:empty {...}</td>
</tr>
<tr>
<td></td>
<td>element (text node).</td>
<td>Empty divs.</td>
</tr>
<tr>
<td>s:parent</td>
<td>Elements that have child elements.</td>
<td>div:parent {...}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Non-empty divs.</td>
</tr>
<tr>
<td>s1:has(s2)</td>
<td>Elements that match s1 and have directly or indirectly contain elements</td>
<td>table:has(th) {...}</td>
</tr>
<tr>
<td></td>
<td>that match s2.</td>
<td>All tables that have at least one th element inside.</td>
</tr>
</tbody>
</table>

For additional materials, please see http://www.coreservlets.com/. The JavaScript tutorial section contains complete source code for all examples in the entire tutorial series, plus exercises and exercise solutions for each topic.
Big Idea

- **Properties are what you use to style selected elements**
  
  ```
  p { color: red; background-color: yellow }
  ```

  - All `<p>` elements have foreground red and background yellow

  ```
  .indented { margin-left: 10px }
  ```

  - `<blah class="indented">` elements are indented 10 pixels on the left.

- **Only a few properties are shown here**
  - For complete list, with examples, see
    
    http://www.w3.org/TR/CSS21/propidx.html

---

**CSS Properties: Colors and Fonts**
Colors

- **color**
  - Foreground text color
- **background-color**
  - Color of the background
- **border-color**
  - Color of the border (see upcoming “box model” slides)

**Example**
```css
.colored {
  color: red;
  background-color: #00ff00;
  border-color: black;
}
```
- Example usage: `<p class="colored">…</p>`

Color Units

- **Official predefined color name**
  - aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, orange, purple, red, silver, teal, white, yellow
- **X11 color names**
  - See http://www.html-color-names.com/color-chart.php
  - Includes CornSilk, Salmon, PapayaWhip, etc.
    - Universally supported for historical reasons (because the first browser was on OS that used X Windows) but not official
- **RGB spec**
  - `#ff00ff`
  - `rgb(255, 0, 255)`
- **Examples**
  ```css
  .cool { color: red; background-color: #0000ff;
   border-color: papayawhip}
  ```
Fonts

- **font-size**
  
  ```
  { font-size: 25px }
  { font-size: 125% }
  ```

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

- **font-style**
  
  ```
  { font-style: italic }
  ```

- **font-weight**
  
  ```
  { font-weight: bold }
  ```

For additional materials, please see http://www.coreservlets.com/. The JavaScript tutorial section contains complete source code for all examples in the entire tutorial series, plus exercises and exercise solutions for each topic.
**Box Model: Idea**

- **Idea**
  - Block-level elements have four parts: margin, border, padding, and the main content section
    - Size units can be in colors or percentages
    - Color units shown earlier

**Box-Model: Properties**

- **Margin**
  - margin
  - margin-left, margin-right, margin-top, margin-bottom

- **Border**
  - border, border-color
  - border-left, border-left-color, etc.

- **Padding**
  - padding
  - padding-left, padding-right, padding-top, padding-bottom

- **Content**
  - width
  - height
A Few Examples

Centered Headings

- Code
  ```
  h1, h2, h3 {
    text-align: center;
    font-family: Verdana, Arial, Helvetica, sans-serif;
    color: black;
  }
  ```

- Meaning
  - `<h1>`, `<h2>`, and `<h3>` headings are centered and black.
  - They use Verdana font if available, Arial as second choice, Helvetica as third choice, and otherwise use the default non-serif font of the system.

For additional materials, please see http://www.coreservlets.com/. The JavaScript tutorial section contains complete source code for all examples in the entire tutorial series, plus exercises and exercise solutions for each topic.
3D Titles

- **Code**
  ```html
  h1.title {
    display: table;
    margin: auto;
    background-color: #afc4d6;
    border-width: 4px;
    border-style: outset;
    border-color: #9fd1ff;
    padding: 5px 8px;
    letter-spacing: -.025em;
  }
  ```

- **Meaning**
  - `<h1 class="title">Text</h1>` displayed as above

Aligned Textfields and R-Aligned Prompts

- **Code**
  ```html
  .formTable {
    display: table;
  }
  .formTable td:first-child {
    text-align: right;
  }
  ```

- **Meaning**
  - The first `<td>` of each `<tr>` is R-aligned when inside `<table class="formTable">`
    - The prompt is in the first `<td>`, the textfield in the second, and the error message (if any) in the third.
Wrapup

Summary

• Load style sheets
  <link href="css/styles.css" rel="stylesheet"/>

• Apply styles (that start with “.” in the CSS file)
  <tag class="name">

• Selectors
  h1 {...}
  • Applies to all h1 elements
  .foo {...}
  • Applies to <tag class="foo">
  div table.bar td {...}
  • Applies to tds inside <table class="bar" that are inside divs
Questions?

For additional materials, please see http://www.coreservlets.com/. The JavaScript tutorial section contains complete source code for all examples in the entire tutorial series, plus exercises and exercise solutions for each topic.