CSS for JSF Developers: A Very Quick Overview

Origins of slides and source code for examples: http://www.coreservlets.com/JSF-Tutorial/jsf2/

Also see the PrimeFaces tutorial – http://www.coreservlets.com/JSF-Tutorial/primefaces/
and customized JSF2 and PrimeFaces training courses – http://courses.coreservlets.com/jsf-training.html

Customized Java EE Training: http://courses.coreservlets.com/
Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at your location.

For live training on JSF 2, PrimeFaces, or other Java EE topics, email hall@coreservlets.com
Marty is also available for consulting and development support

Taught by the author of Core Servlets and JSP, this tutorial, and JSF 2.2 version of Core JSF. Available at public venues, or customized versions can be held on-site at your organization.

- Courses developed and taught by Marty Hall
  - JSF 2, PrimeFaces, Ajax, jQuery, Spring MVC, JSP, Android, general Java, Java 8 lambdas/streams, GWT, custom topic mix
  - Courses available in any location worldwide. Maryland/DC area companies can also choose afternoon/evening courses.
- Courses developed and taught by coreservlets.com experts (edited by Marty)
  - Hadoop, Spring, Hibernate/JPA, RESTful Web Services
Contact hall@coreservlets.com for details
Overview

• Most JSF developers already know at least the basics of 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
  – I have been asked by several JSF students to give some introductory material on HTML (earlier section) and CSS (this section)
• Far more details are available online
  – Any of the online CSS tutorials give more details than this ultra-brief introduction

Topics in This Section

• Loading and using style sheets
  – Standard HTML
  – JSF-specific
• CSS selectors
  – Applying styles in standard HTML elements
  – Applying styles in JSF elements
• CSS properties
• Examples
• References
Loading and Using Style Sheets

Customized Java EE Training: http://courses.coreservlets.com/
Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at your location.

Loading an External Style Sheet: Standard HTML

• 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.

  [The type attribute is officially optional in HTML 5. In other HTML versions, although it is ignored by browsers, it should be included for HTML validators.]
Loading an External Style Sheet: JSF-Specific Approach

• Loading the style sheet
  
  \begin{verbatim}
  <h:head>
    <h:outputStylesheet name="styles.css" library="css"/>
    ...
  </h:head>
  \end{verbatim}

• Locating the style sheet
  – The CSS file goes in the folder referred to by “library”, relative to the “resources” folder.
    • For example, in above example, styles.css is in
      \begin{verbatim}
      ../../resources/css/ (WebContent/resources/css/ in Eclipse)
      \end{verbatim}

• Motivation
  – So that you can refer to style sheets when writing a page or a piece of a page when you do not know the URL
    • Examples and details of this approach are covered in tutorial sections on JSF page templating and on composite components

Embedding Styles (HTML and JSF)

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

• Listing CSS styles inline
  – \begin{verbatim}<h1 style="color: red; background-color: blue">…</h1>\end{verbatim}
Using Styles from Style Sheets

• **Styles that apply to elements**
  – Apply automatically
    • h2 { color: blue; font-family: sans-serif }
      – All `<h2>` elements automatically in blue non-serif font

• **Styles that start with “.”**
  – Must be applied with “class” (HTML) or “styleClass” (JSF) attributes
    • .warning { color: red; font-weight: bold; font-size: 120% }
  – Standard HTML elements
    • `<p class="warning">…</p>`
    • `<span class="warning">…</span>`
  – JSF elements
    • `<h:message styleClass="warning" …/>`
    • `<h:outputText styleClass="warning"…/>`

CSS Selectors
Big Idea

• Selectors are what designate which elements the styles apply to
  – 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. For JSF elements, you would use <h:blahTag styleClass="indented">
  – 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/
## Basic Selectors

<table>
<thead>
<tr>
<th>Selector</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>element</td>
<td>Matches all elements with given tag name. Could be many matches.</td>
<td>li {...}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>p {...}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#id</td>
<td>Matches the element with given id. Matches 0 or 1 elements.</td>
<td>#blah {...}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.class</td>
<td>Matches all elements with given CSS style.</td>
<td>.important {...}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>element.class</td>
<td>Matches all elements with given tag name that have given class.</td>
<td>div.important {...}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>element#id</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>form#blah {...}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>Matches all elements in entire page. Particularly useful for nesting and with the :not selector.</td>
<td>* {...}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>div * {...}</td>
</tr>
</tbody>
</table>

## Hierarchical Selectors

<table>
<thead>
<tr>
<th>Selector</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1 s2</td>
<td>Elements that match selector s2 and are directly or indirectly inside an element that matches selector s1.</td>
<td>div.foo span.bar {...}</td>
</tr>
<tr>
<td>s1 &gt; s2</td>
<td>Elements that match selector s2 and are directly inside an element that matches s1.</td>
<td>div.foo &gt; span.bar {...}</td>
</tr>
<tr>
<td>s1, s2</td>
<td>Elements that match either selector.</td>
<td>ul,ol,dl.foo {...}</td>
</tr>
<tr>
<td>s1 + s2</td>
<td>Elements that match s2 and are immediately after a sibling element matching s1.</td>
<td>label + input {...}</td>
</tr>
<tr>
<td>s1 ~ s2</td>
<td>Elements that match selector s2 and are somewhere after a sibling element matching s1.</td>
<td>label ~ input {...}</td>
</tr>
</tbody>
</table>
### Attribute Selectors

<table>
<thead>
<tr>
<th>Selector</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
</table>
| s[att]  | Elements that match selector s and also contain attribute named att. | div.blah a[name] {...} \(
|          |         | Matches all `<a name="...">` elements that are inside `<div class="blah">` |
| s[att=val]  | Elements that match selector s and also contain attribute named att whose value is (exactly) val. | a[href=#sect2] {...} \(
|          |         | Matches all `<a href="#sect2">` elements |
| s[att^=val]  | Elements that match selector s and also contain attribute named att whose value starts with val. | a[href^=#] {...} \(
|          |         | Matches all internal hyperlinks |
| s[att$=val]  | Elements that match selector s and also contain attribute named att whose value ends with val. | a[href$=jquery.com] {...} \(
|          |         | Matches all hyperlinks pointing to `blah.jquery.com` home page (not subpages) |
| s[att*=val]  | Elements that match selector s and also contain attribute named att whose value contains val. | a[href*=jquery.com] {...} \(
|          |         | Matches all hyperlinks pointing to any page at `blah.jquery.com` |
| s[att!=val]  | Elements that match selector s and either do not have the specified attribute, or have a different value. | a[href!=#sect2] {...} \(
|          |         | Matches all hyperlinks except `<a href="#sect2">` elements |
| s:not([...])  | Elements that match s but do not match attribute specification. | a:not([href^=http]) {...} \(
|          |         | Matches hyperlinks that do not start with http… |

### Positional Selectors

<table>
<thead>
<tr>
<th>Selector</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
</table>
| s:first  | First or last match in page. | ul.foo li:first {...} \(
| s:last   |         | Matches first li element that is inside `<ul class="foo">` |
| s:eq(n)  | The nth match in the page. Count starts at 0. | p:eq(3) {...} \(
|          |         | Fourth p element in page. |
| s:gt(n), s:lt(n)  | Elements after/before the nth. | p:gt(3) {...} \(
|          |         | 5th and following p elements. |
| s:even   | Elements that are even or odd numbered elements in the page. 0-based, so first match is even. | tr:even {...} \(
| s:odd    |         | Finds all table rows, then matches the even numbered ones from that overall list. |
| s:first-child | Elements that are the first or last child of their parents, or that have no siblings. | tr:first-child {...} \(
| s:last-child |         | Matches the first row of every table. |
| s:only-child |         | |
| s:nth-child(n)  | Elements that are the nth child. First child is nth-child(1), not (0) | tr:nth-child(3) {...} \(
|          |         | Matches the third row of each table |
| s:nth-child(even) | Elements that are even or odd children of their parent. Count starts at 1, so first match is odd. | tr:nth-child(even) {...} \(
| s:nth-child(odd)  |         | Matches rows that are even numbered rows of their own table. |
| s:nth-child(xn+y)  | Elements matching formula. You list “n” literally. So, 3n means every third. 3n+1 means entry after every third. | tr:nth-child(4n+2) {...} \(
|          |         | Matches row 6, 10, 14, … of each table. |
# Content Filtering Selectors

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

© 2015 Marty Hall

Customized Java EE Training: [http://courses.coreservlets.com/](http://courses.coreservlets.com/)
Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at your location.
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. For JSF elements, you would use <h:blahTag styleClass="indented">

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

Colors

• color
  – Foreground text color
• background-color
  – Color of the background
• border-color
  – Color of the border (see upcoming “box model” slide)

• Example
  .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.
  - Supported for historical reasons, because the first browser was on OS that used X Windows

- **RGB spec**
  - #ff00ff
  - rgb(255, 0, 255)

- **Examples**
  - .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 }
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**

```css
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.
3D Titles

• Code

```css
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

Forms with Aligned Textfields and R-Aligned Prompts

```
.formTable {
  display: table;
}
.formTable td:first-child {
  text-align: right;
}
.formTable td:nth-child(3) {
  color: red;
  font-weight: bold;
}
```

• Meaning
  – The first `<td>` of each `<tr>` is R-aligned when inside `<table class="formTable">` (or h:panelGrid)
  • The prompt is in the first `<td>`, the textfield in the second, and the red/bold error message (if any) in the third.
Wrapup

References

• **Books**
  – *CSS Cookbook* by Christopher Schmitt
  – *CSS3: Visual Quickstart Guide*
    by Jason Cranford Teague
  – *HTML5 and CSS3* by Brian P. Hogan

• **Online References**
  – All official CSS specs: http://www.w3.org/Style/CSS/specs.en.html
  – CSS tutorial: http://www.w3schools.com/css/
  – Search on Google for “CSS tutorial[s]"
  – Cool demo: http://www.csszengarden.com/
    • Examples of how you can totally revamp a page by changing only the CSS.
Summary

- Load style sheets
  - HTML and JSF
    - `<link href="css/styles.css" rel="stylesheet" type="text/css"/>
  - JSF only
    - `<h:outputStylesheet name="styles.css" library="css"/>

- Apply styles (that start with “.”)
  - HTML elements: `<tag class="name">
  - JSF elements: `<h:tag styleClass="name">

- Selectors
  - `h1 { ...}
    - Applies to all h1 elements
  - `.foo {...}
    - Applies to `<tag class="foo"> or `<h:tag styleClass="foo">
  - `<table class="bar">td {...}
    - Applies to tds inside `<table class="bar"> that are inside divs