

Cascading Style Sheets, part I

Web Design in a Nutshell, Third Edition

Chapters 16, 17, 18 and 23

Objectives

- Understand CSS style fundamentals
 - Understand the rules
 - Adding styles to a document
 - Use basic selection techniques
 - Font and Text Properties
 - List Properties
-

CSS Fundamentals

- Before Cascading Style Sheets (CSS), web designers were at the mercy of the browser's rendering engine and internal style sheets for the way HTML elements looked in the browser window.
 - CSS is a W3C standard defining the presentation of web documents
 - Cascading style sheets offer much greater control over type characteristics than do the font, bgcolor and other presentational html elements and attributes
 - You can use standard type conventions, such as using point or pixel sizes, setting leading, and specifying indents and alignment
-

Understanding CSS Style Rules

- Style rules are composed of two parts: a selector and a declaration
 - The selector determines the element to which the rule is applied
 - The declaration details the exact property values
-

Selector and Declaration

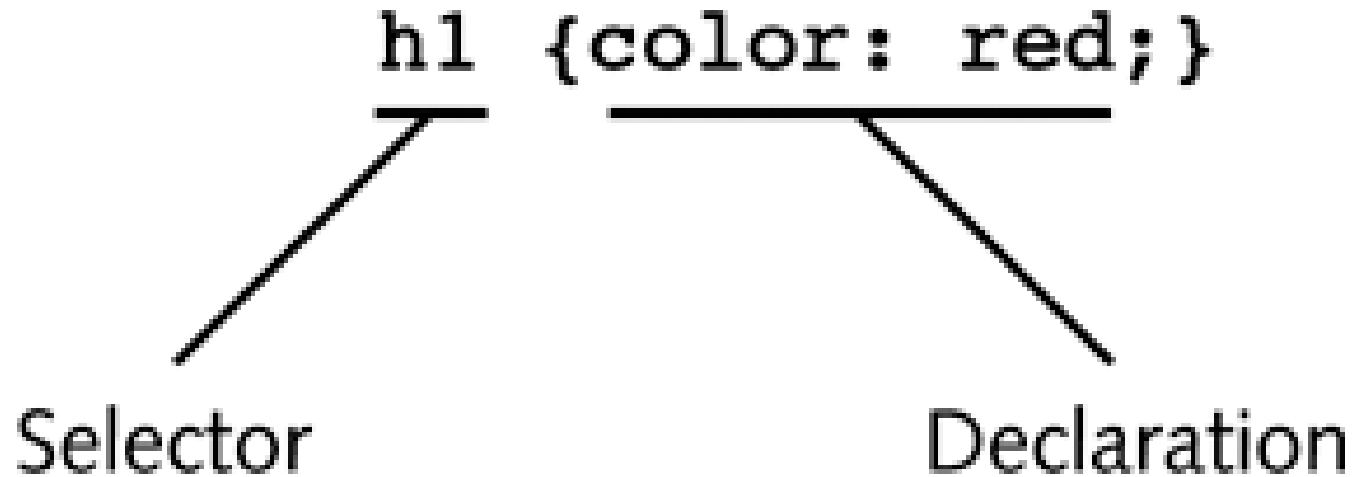


Figure 6-1 Style rule syntax

Understanding CSS Style Rules

- The declaration contains a property and a value
 - The property is a quality or characteristic
 - The precise specification of the property is contained in the value
 - CSS includes over 50 different properties, each with a specific number of values
-

Property and Value

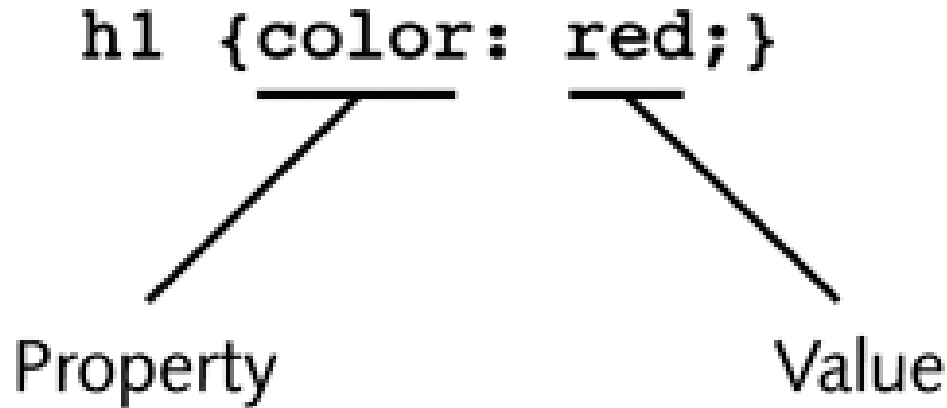


Figure 6-2 Property declaration syntax

CSS Basics – Using CSS

- Three ways to combine CSS rules and XHTML

- The style attribute in a standard xhtml element

```
<p style="text-align: left;">
```

- The <style> element in the head section

```
<style type="text/css">p {text-align: left;}</style>
```

- External style sheet

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

CSS Basics — Combining CSS with XHTML

- The style attribute
 - Defines a style for a single element
 - Generally used to override a style set at a higher level in the document for a single element
 - Only affects one instance of an element in a document
-

CSS Basics — Combining CSS with XHTML

- The `<style>` element
 - Always contained in the `<head>` section of the document
 - Generally used to override a style set at a higher level in the document for a single document
 - Only affects the document in which it resides
-

CSS Basics — Combining CSS with XHTML

- External style sheet
 - Text document that contains style rules
 - Allows specification of rules for multiple HTML documents
 - Does not contain HTML code
-

CSS – Understanding the Cascade

- Understanding the Cascade
 - Cascading mechanism of CSS determines which rules are assigned to document elements by assigning a weight based on four variables:
 1. origin of the rule
 2. specificity of the selector
 3. order of the rule in the style sheet
-

CSS – Understanding the Cascade

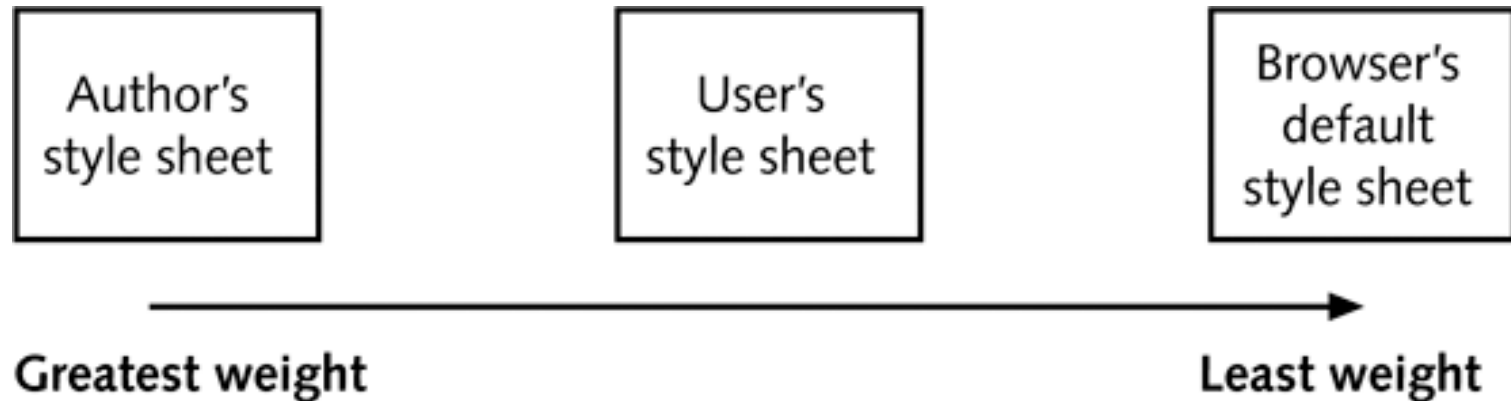


Figure 6-5 The cascading order of precedence

CSS – Understanding the Cascade

- Determining rule weight by specificity
 - Rules with more specific selectors take precedence over rules with less specific selectors
 - Determining rule weight by order
 - Based on order of rule within style sheet
 - Those listed later take precedence over those listed earlier in the style sheet
-

CSS – Understanding Inheritance

- Based on hierarchical structure of documents
 - CSS rules inherit from parent elements to child elements:
 - thus `` elements will inherit style rules from `` elements unless a style rule is specifically set for the `` element
-

CSS – Understanding Inheritance

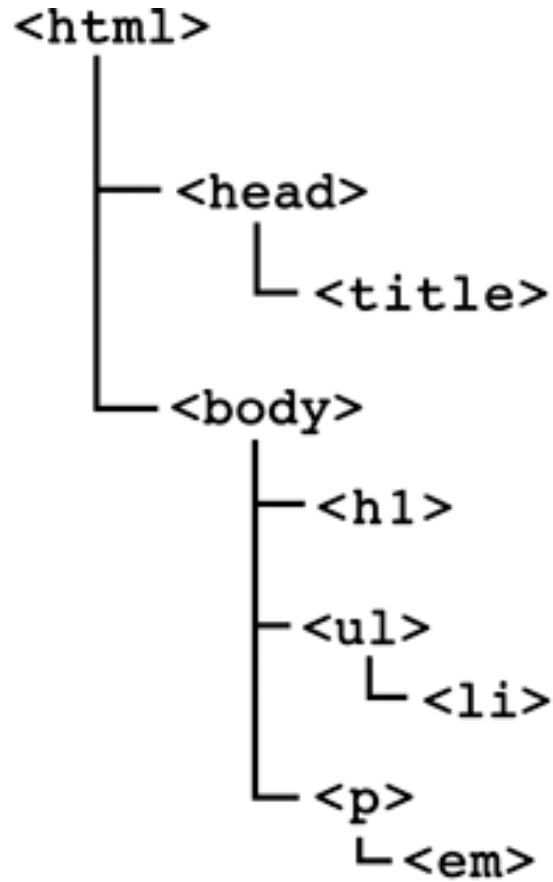


Figure 6-6 HTML document structure

Understanding Selection Techniques

- Using type selectors
 - Grouping selectors
 - Combining declarations
 - Using descendant selectors
-

Using Type Selectors

- The following rule selects the H1 element:



Figure 6-7 Style rule syntax

Grouping Selectors

- The following rule selects the H1 and H2 elements:

```
<style type="text/css">  
  h1, h2 {color: green;}  
</style>
```

Combining Declarations

- The following style rules set the <p> element to 12-point blue text:

```
p {color: blue;}  
p {font-size: 12pt;}
```

These two style rules can be expressed in a simpler way:

```
p {color: blue; font-size: 12pt;}
```

Using Descendant Selectors

- A descendant selector lets you specify the exact context in which a style is applied. To specify that `` elements appear blue only within `<p>` elements, use the following rule:

```
<style type="text/css">  
  p b {color: blue;}  
</style>
```

Understanding Advanced Selection Techniques

- Using the class attribute
 - Using the `<div>` and `` elements
-

Using the class Attribute Selector

- The class attribute lets you write rules and then apply them to groups of elements that you have classified
 - To create a class, declare it within the `<style>` element first. The period (.) flag character indicates that the selector is a class selector.
-

class Attribute Selector

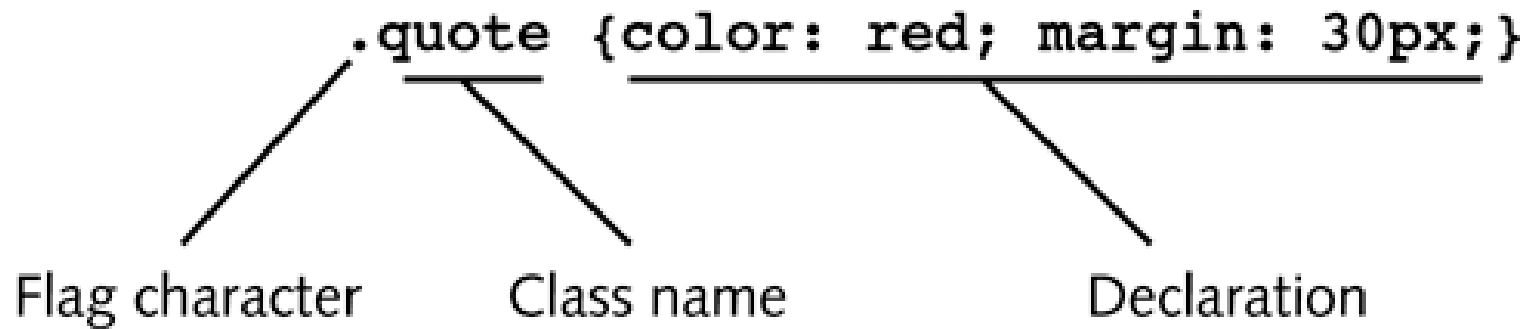


Figure 6-10 Class syntax

class Attribute Selector

```
.special {font-size: 10pt; font-weight:  
        bold;}
```

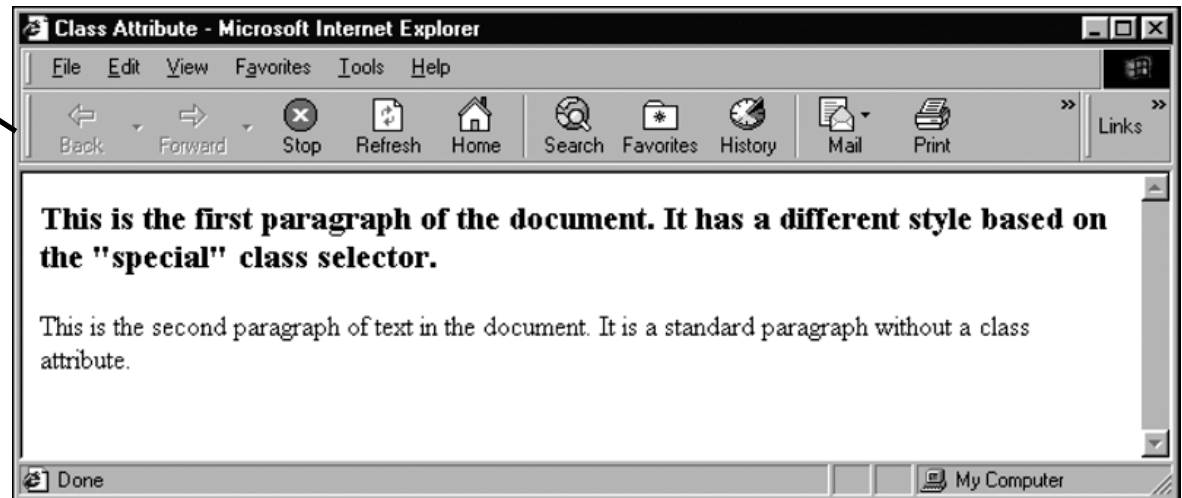


Figure 6-11 Styling with a class attribute

Working with `<div>`

- The `<div>` element lets you specify logical divisions within a document that have their own name and style properties
 - `<div>` is a block-level element. It contains a leading and trailing carriage return.
 - You can use `<div>` with the `class` attribute to create customized block-level elements
-

Working With <div>

- To create a division, declare it within the <style> element first. The following example specifies a division named introduction as the selector for the rule:

```
<style type="text/css">  
div.introduction {color:red;}  
</style>
```

Working With <div>

- Next, specify the <div> element in the document. Then use the class attribute to specify the exact type of division. In the following example, the code defines the <div> element as the special class named “introduction.”

```
<div class="introduction">Some text</div>
```

Working with ``

- The `` element lets you specify inline elements within a document that have their own name and style properties
 - Inline elements go within the line of text, like the `` element
-

Working with ``

- To create a span, declare it within the `<style>` element first. The following example specifies a `` element named “logo” as the selector for the rule:

```
<style type="text/css">  
span.logo {color:red;}  
</style>
```

Working with ``

- Next, specify the `` element in the document. Then use the class attribute to specify the exact type of span. In the following example, the code defines the `` element as the special class named “logo.”

Welcome to the ``Wonder
Software`` Web site.

Working with ``

Welcome to the ``Wonder
Software`` Web site.

Welcome to the **Wonder Software** Web site.

Figure 6-12 Using the `` element

Summary

- CSS rules can be combined with your XHTML code in a number of ways. CSS rules are easy to write and read.
 - CSS uses cascading and inheritance to determine which style rules take precedence. The !important declaration lets users override the author's style rules.
-

Summary

- Basic style rules let you apply style rules based on standard element selectors.
 - You can combine the selectors and declarations to create more powerful style expressions.
 - You can also select elements based on the contextual relationship of elements in the document tree.
-

Summary

- The advanced selection techniques allow you to use the class attribute selector, which is often paired with the `<div>` and `` XHTML elements.
 - These elements have no style of their own, but offer a convenient way of expressing style for any section of a document, whether block-level or inline.
 - Additionally, class allows you to choose a meaningful naming convention for your style rules.
-