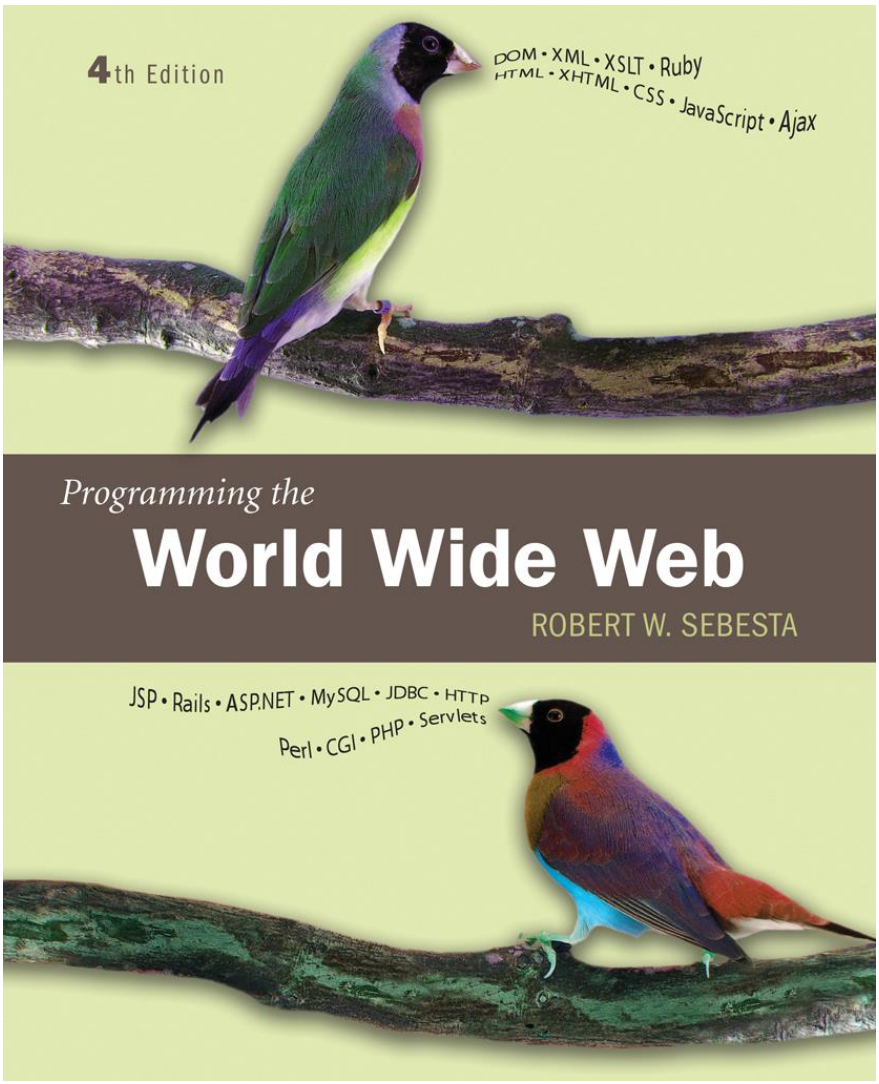


Chapter 3
Cascading Style Sheets

3.1 Introduction

- The CSS1 specification was developed in 1996
- CSS2 was released in 1998
- CSS3 is on its way
- CSSs provide the means to control and change presentation of XHTML documents
- CSS is not technically XHTML, but can be embedded in XHTML documents
- Style sheets allow you to impose a standard style on a whole document, or even a whole collection of documents
- Style is specified for a tag by the values of its properties

3.2 Levels of Style Sheets

- There are three levels of style sheets
  1. **Inline style sheets**- specified for a specific occurrence of a tag and apply only to that tag
     – This is fine-grain style, which defeats the purpose of style sheets - uniform style
  2. **Document-level style sheets** - apply to the whole document in which they appear
  3. **External style sheets** - can be applied to any number of documents
- When more than one style sheet applies to a specific tag in a document, the lowest level style sheet has precedence
  - In a sense, the browser searches for a style property spec, starting with inline, until it finds one (or there isn't one)
- If no style sheet information is specified, the browser default property values are used
- Sometimes the browser may not be capable of using the property values specified in a style sheet

3.2 Levels of Style Sheets (continued)

- Inline style sheets appear in the tag itself
- Document-level style sheets appear in the head of the document
- External style sheets are in separate files, potentially on any server on the Internet
  - Written as text files with the MIME type `text/css`
  - Two types of tags can be used
  - `<link>` tag– which appears only in the `<head>`
  - `@import` tag which appears only at the beginning of the content of the style element

3.2 Linking an External Stylesheet

- A `<link>` tag is used to specify that the browser is to fetch and use an external style sheet file

```
<link rel = "stylesheet"  type = "text/css"
 href = "http://www.wherever.org/termpaper.css">
</link>
```

- External style sheets can be validated

```
http://jigsaw.w3.org/css-validator/
                validator-upload.html
```

3.3 Style Specification Formats

- Format depends on the level of the style sheet
- Inline:
  - Style sheet appears as the value of the `style` attribute
  - General form:

```
style = "property_1: value_1;
         property_2: value_2;
         …
         property_n: value_n;"
```

Example:

```
<h1 style ="font-size:25pt; font-style:italic;">
      This is inline style example</h1>
```

3.3 Format for Document-level

- Style sheet appears as a list of rules that are the content of a `<style>` tag
- The `<style>` tag must include the `type` attribute, set to `"text/css"`
- Comments in the rule list must have a different form - use C comments (`/*…*/`)

- General form:

```
<style type = "text/css">
 <!--
    rule list
 -->
</style>
```

3.3 General Form, Document Level

- Format of the rules:
  selector {list of property/values}
  - Selector indicates the tag or tags affected by the rule
  - Each property/value pair has the form:
    property: value
  - Pairs are separated by semicolons, just as in the value of a `<style>` tag

  Example:

  ```
  <style type="text/css">
  h1, h3 {font-size:25pt; font-style:italic;}
  </style>
  ```

3.3 External style sheets

- These style sheets have the same general form as the document level style sheets
- The link to the style sheet which has the property/value pair appears in the <head> of the html document where those styles are used

Example: The external_style.html contains the following in <head>

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

And the styles.css will contain:

```
h1, h3 {font-size:25pt; font-style:italic;}
```

3.4 Selector Forms:

1. Simple selectors:

- The selector is a tag name or a list of tag names, separated by commas

```
example:
  h1 {font-style: 24pt;}
  h2, h3 {font-size: 20pt;font-style:italic;}
```

- *Contextual (descenant) selectors*
  - Selectors can also specify that the style should apply to elements in certain positions in the document
  - This is done by listing the element hierarchy in the selector, with only whitespace separating the element names.

```
example:
        body b em {font-size: 14pt;}
```

3.4 Selectors Forms (continued)

2. Class selectors:

- Used to allow different occurrences of the same tag to use different style specifications

- A style class has a name, which is attached to a tag name
  - `p.narrow {`property/value list`}`
  - `p.wide {`property/value list`}`

- The class you want on a particular occurrence of a tag is specified with the `class` attribute of the tag

- For example,

```
<p class = "narrow">
...
</p>
...
<p class = "wide">
...
</p>
```

3.4 Selectors Forms (continued)

3. Generic Selectors

- A generic class can be defined if you want a style to apply to more than one kind of tag

- A generic class must be named, and the name must begin with a period

- Example,

```
.sale { property-value list}
```

- Use it as if it were a normal style class

```
<h1 class = "sale"> … </h1>
...
<p class = "sale"> … </p>
```

3.4 Selectors Forms (continued)

4. `id` *Selectors*

- An `id` selector allow the application of a style to one specific element
- General form:

  `#`specific-id `{`property-value list`}`
- Example:

  #section14 {font-size: 20}
- In the tag where we are using this id we should write like

  <h2 id="section14"> This is example of id selector </h2>

3.4 `universal` *Selectors*

- An `universal` selector denoted by an asterisk (*), applies its style to all the elements in the document

- Example:

```
* {color: red;}
```

- Makes all the elements in the document red.

3.4 Pseudo Classes

- Pseudo classes are styles that apply when something happens, rather than because the target element simply exists
- Names begin with colons
- `hover` classes apply when the mouse cursor is over the element
- `focus` classes apply when an element has focus

3.4 Pseudo Class Example

```
<!-- pseudo.html -->
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head> <title> Checkboxes </title>
    <style type = "text/css">
      input:hover {color: red;}
      input:focus {color: green;}
    </style>
  </head>
  <body>
    <form action = "">
      <p>
        Your name:
        <input type = "text" />
      </p>
    </form>
  </body>
 </html>
```

3.5 Properties

- There are 60 different properties in 7 categories:
  - Fonts
  - Lists
  - Alignment of text
  - Margins
  - Colors
  - Backgrounds
  - Borders

3.5 Property Values

- **Keywords - left, small, …**
  - Not case sensitive
- **Length - numbers, maybe with decimal points**
- **Units:**
  - px - pixels
  - in - inches
  - cm - centimeters
  - mm - millimeters
  - pt - points
  - pc - picas (12 points)
  - em - height of the letter ‘m’
  - ex-height - height of the letter ‘x’
  - No space is allowed between the number and the unit specification e.g., 1.5 in is illegal!

3.5 Property Value Forms (continued)

- **Percentage** – used to provide a measure that is relative to the previously used measure for a property value.
  - just a number followed immediately by a percent sign

  Example: if 75%is set for font size, it would take 75% of its previous value and stays there until it is changed again.
  - These can be signed
  - plus sign- percentage is added to the previous value
  - negative- percentage is subtracted from the previous value.
- **URL values**
  - url(tetons.jpg)
  - No space between url and left parenthesis.

3.5 Property Value Forms (continued)

- **Colors**
  - Color names, as six hexadecimal number or in RGB form.
  - rgb(n1, n2, n3)
    - Numbers can be decimal (0-255) or percentages
  - Hex form: #XXXXXX

  For example: powder blue can be specified as

  ```
  powderblue or rgb(176, 224, 230) or #B0E0E6
  ```

- Property values are inherited by all nested tags, unless overridden

  ```
  example: body {font-size: 16pt}
  ```

3.6 Font Properties

1.Font Families

- Used to specify a list of font names - browser uses the first in the list that it supports

```
Example: font-family: Arial, Helvetica,
  Futura
```

- If the browser does not support any of the specified fonts, it will use an alternative of its choosing

3.6 Font Properties (continued)

- Generic fonts can be specified as a font family value

Generic Name	Examples
serif	Times New Roman, Garamond
sans-serif	MS Arial, Helvetica
cursive	Caflisch script, Zapf-Chancery
fantasy	Critter, Cottonwood
Monospace	Courier, Prestige

- Each browser has its font defined for each of these generic names

```
Example: font-family: Arial, Helvetica, Futura, sans-
  serif
```

- If a font name has more than one word, it should be single-quoted `eg: font-family: 'Times New Roman'`

3.6 Font Properties (continued)

2. Font Sizes

- Used to set the font sizes

Example:

```
font-size: 10pt
```

- Relative values can also be defined eg: xx-small, xx-large, x-small, small, medium, large, x-large etc
- Disadvantage of this is: Different browsers can use different values
- If specific size is used then some browsers may not be able to support the values specified.

3. Font Variants

- Default value if `normal` specifies the usual character font.
- Can be set to `small-caps` to specify small capital letters

3.6 Font Properties (continued)

5. Font style

- Most commonly used to specify italic

```
Font style : italic
```

6. Font weights

- Used to specify degree of boldness,

```
Example: font-size: bold
```

- `bolder` and `lighter` can be specified.
- Specific numbers can also be given in multiples of 100 to 900.

7. Font shorthand

- If more than one font property must be specified, the values can be stated in a list as the value of the font property

```
Example: font: bold 14pt 'Times New Roman' Palatino
```

3.6 Font Properties (continued)

- Rule is as follows:
  1. **Font names must be last**
  2. **Font size must be second last**
  3. **Font style, font weight and font variant when they are included can be in any order but precede font size and font name**

8. Text Decoration

- The `text-decoration` property
  - `line-through`, `overline`, `underline`, `none`
  - `letter-spacing` – value is any length property value

    eg: `letter-spacing : 3px`

3.7 *List properties*

- ***list-style-type*** property is used to change both the unordered and ordered list default values.
- *Unordered lists*
  - **Bullet can be a disc (default), a square, or a circle**
  - Set it on either the `<ul>` or `<li>` tag
    - On `<ul>`, it applies to list items

```
<h3> Some Common Single-Engine Aircraft </h3>
  <ul style = "list-style-type: square">
      <li> Cessna Skyhawk </li>
      <li> Beechcraft Bonanza </li>
      <li> Piper Cherokee </li>
  </ul>
```

3.7 *List properties* (continued)

- On `<li>`, `list-style-type` applies to just that item

```
<h3> Some Common Single-Engine Aircraft </h3>
 <ul>
     <li style = "list-style-type: disc">
          Cessna Skyhawk </li>
     <li style = "list-style-type: square">
          Beechcraft Bonanza </li>
     <li style = "list-style-type: circle">
          Piper Cherokee </li>
 </ul>
```

3.7 *List properties* (continued)

- Could use an image for the bullets in an unordered list
  - Example:

```
<li style = "list-style-image: url(bird.jpg)">
```

- *On ordered lists* - `list-style-type` can be used to change the sequence values

Property value	*Sequence type*	*First four*
`Decimal`	Arabic numerals	1, 2, 3, 4
`upper-alpha`	Uc letters	A, B, C, D
`lower-alpha`	Lc letters	a, b, c, d
`upper-roman`	Uc Roman	I, II, III, IV
`lower-roman`	Lc Roman	i, ii, iii, iv

3.8 Colors

- Color is a problem for the Web for two reasons:
  1. Monitors vary widely
  2. Browsers vary widely
- There are three color collections
  1. There is a set of 16 colors that are guaranteed to be displayable by all graphical browsers on all color monitors

black	**000000**	**green**	**008000**
silver	**C0C0C0**	**lime**	**00FF00**
gray	**808080**	**olive**	**808000**
white	**FFFFFF**	**yellow**	**FFFF00**
maroon	**800000**	**navy**	**000080**
red	**FF0000**	**blue**	**0000FF**
purple	**800080**	**teal**	**008080**
fuchia	**FF00FF**	**aqua**	**00FFFF**

  2. There is a much larger set, the Web Palette include 216 colors
  3. When the limitations of browsers and monitors are not considered , 6 digit hex numbers can be used to specify any one of the 16 million colors.

3.8 Colors (continued)

- The `color` property specifies the foreground color of elements

```
<style type = “text/css”>
  th.red {color: red}
  th.orange {color: orange}
</style>
…
<table border = "5">
  <tr>
    <th class = "red"> Apple </th>
    <th class = "orange"> Orange </th>
    <th class = "orange"> Screwdriver </th>
  </tr>
</table>
```

3.8 Colors (continued)

- The `background-color` property specifies the background color of elements

```
<style type = "text/css">
  p.standout{font-size: 24pt; color: blue
          background-color: red}
</style>
.....
<p class = "standout">
  To really make it stand out, use red background!!
</p>
```

3.9 Alignment of Text

- The `text-indent` property allows indentation
  - Takes either a length or a % value

Example:
```
<style type = "text/css">
    p.indent{text-indent: 0.5in}
</style>
```

- The `text-align` property has the possible values, `left` (the default), `center`, `right`, or `justify`

```
p {text-align: right}
```

3.9 Alignment of Text (continued)

- Sometimes we want text to flow around another element - the `float` property
  - The `float` property has the possible values, `left`, `right`, and `none` (the default)
  - If we have an element we want on the right, with text flowing on its left, we use the default `text-align` value (`left`) for the text and the `right` value for `float` on the element we want on the right

3.9 Alignment of Text (continued)

```
<img src = "c210.jpg"
         style = "float: right" />
```

- Some text with the default alignment - `left`

3.10 The Box Model

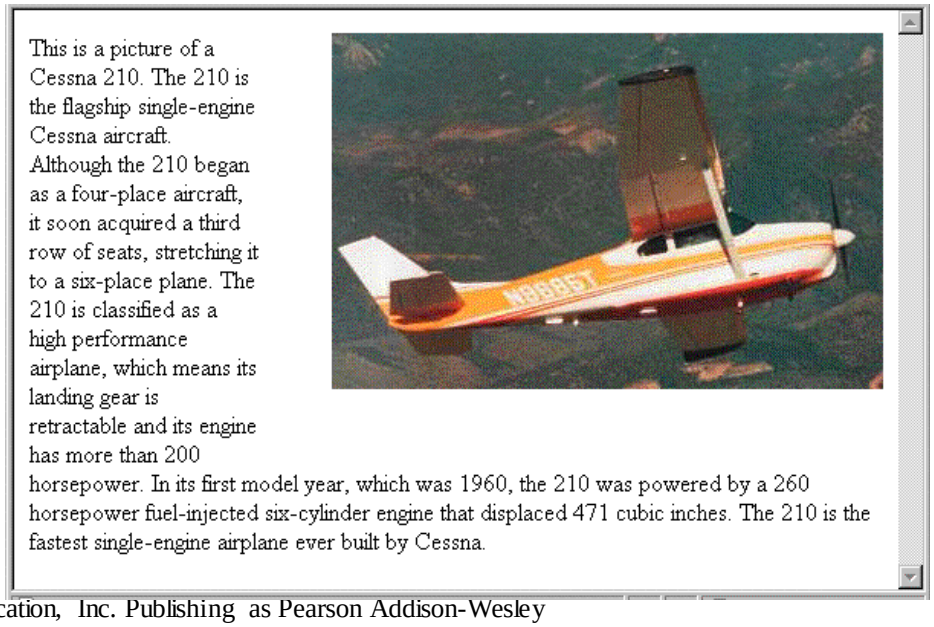
- Borders – every element has a `border-style` property
  - Controls whether the element has a border and if so, the style of the border
  - `border-style` values: `none`, `dotted`, `dashed`, and `double`
  - The styles of one of the four sides of an element can be set using `border-top-style, border-bottom-style, border-left-style` and `border-right-style`
  - `border-width` – `thin`, `medium` (default), `thick`, or a length value in pixels
  - Border width can be specified for any of the four borders (e.g., `border-top-width`)
  - `border-color` – any color can be given for the border
  - Border color can be specified for any of the four borders (e.g., `border-top-color`)

3.10 The Box Model (continued)

- Margin – the space between the border of an element and its neighbor element
- The margins around an element can be set with `margin-left`, etc. - just assign them a length value

```
<img src = "c210.jpg " style = "float: right;
  margin-left: 0.35in;
  margin-bottom: 0.35in" />
```

- If you want to have the same value for all the margins then `margin: value` can be used
- If you want to have the same value for all the padding then `padding: value` can be used



3.10 The Box Model (continued)

- Padding – the distance between the content of an element and its border
  - Controlled by `padding`, `padding-left`, etc.

3.11 Background Images

- The `background-image` property

- Repetition can be controlled
  - `background-repeat` property
  - Possible values: `repeat` (default), `no-repeat`, `repeat-x`, or `repeat-y`
  - `background-position` property
    - Possible values: `top`, `center`, `bottom`, `left`, or `right`

3.12 The and tags

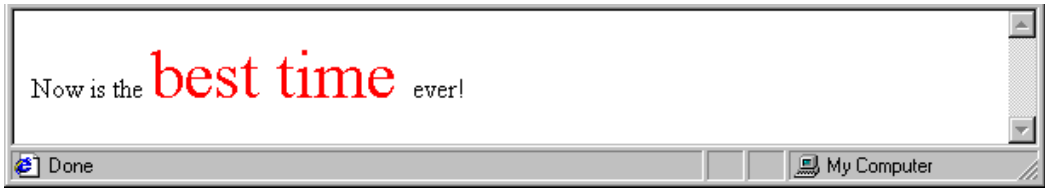
- One problem with the font properties is that they apply to whole elements, which are often too large
  - Solution: a new tag to define an element in the content of a larger element -
  - The default meaning of is to leave the content as it is

```
<p>
Now is the <span> best time </span> ever!
</p>
```

  - Use to apply a document style sheet to its content

```
<style type = "text/css">?
  bigred {font-size: 24pt;
          font-family: Ariel; color: red}
 </style>
 <p>
    Now is the
        <span class = "bigred">
    best time </span> ever!
 </p>
```

3.12 The and tags (continued)



- The tag is similar to other HTML tags, they can be nested and they have id and class attributes
- Another tag that is useful for style specifications:
  - Used to create document sections (or divisions) for which style can be specified
    - e.g., A section of five paragraphs for which you want some particular style

3.13 Conflict Resolution

- **When two or more rules apply to the same tag there are rules for deciding which rule applies**
- **Document level**
  - **In-line style sheets have precedence over document style sheets**
  - **Document style sheets have precedence over external style sheets**
- **Within the same level there can be conflicts**
  - **A tag may be used twice as a selector**
  - **A tag may inherit a property and also be used as a selector**
- **Style sheets can have different sources**
  - **The author of a document may specify styles**
  - **The user, through browser settings, may specify styles**
- **Individual properties can be specified as important**

```
Example: p.special {font-style: italic
  !important; font-size:14pt;}
```

3.13 Precedence Rules

- **From highest to lowest**

1. Important declarations with user origin
2. Important declarations with author origin
3. Normal declarations with author origin
4. Normal declarations with user origin
5. Any declarations with browser (or other user agent) origin

- **Sort by specificity**

1. `Id` selectors
2. Class and pseudo-class selectors
3. Contextual selectors
4. Universal selectors

3.13 Tie-Breakers

- **Specificity**
  1. **id selectors**
  2. **Class and pseudo-class selectors**
  3. **Contextual selectors**
  4. **General selectors**
- **Position**
  - **Essentially, later has precedence over earlier**

Example:

```
p {font-size: 12pt}
P{font-size:10pt}
```

The whole sorting process that is used to resolve style specification conflicts is called the *Cascade.*