Objectives

- Explore the history and theory of CSS
- Define a style rule
- Study style precedence and inheritance
- Apply color using CSS
- Explore CSS3 color extensions
- Use contextual selectors
Objectives

• Work with attribute selectors
• Apply text and font styles
• Install a Web font
• Define list styles
• Use pseudo-classes and pseudo-elements
• Create a rollover effect
Style Sheets and Color

The Sunny Acres Farm Shop aims to offer the highest quality fresh produce. You can pick your own or buy it in our shop. Set amidst acres of outstanding natural beauty on the beautiful rolling hills northeast of Council Bluffs, the Farm Shop is easily reached on Highway G, with easy access from Interstate 570.

The Farm Shop was established over 20 years ago with great success. Our products have won numerous awards at local festivals and fairs. We also cater to local supermarkets in the Council Bluffs/Omaha area. Look for our products every Saturday morning from May to October at the Council Bluffs Farmers’ Market.

**Hours**
- Monday - Friday: 9 am - 5 pm
- Saturday: 9 am - 3 pm
- Pick Your Own Produce is available from May 15 - October 22
- The Farm Shop is open year-round

**Products**
- Freshly baked breads and quilts
- High-quality meats
- Cheese and other dairy products
- Freshly-picked fruits and vegetables (in season)
- Canned goods and preserves

New Perspectives on HTML, CSS, and Dynamic HTML 5th Edition
Introducing CSS

To render a document, the device displaying the page needs a style sheet that specifies the appearance of each page element.

The style sheet language used on the Web is the Cascading Style Sheets language, also known as CSS.

Versions include CSS1, CSS2, CSS 2.1, and CSS3.

With CSS, as with HTML, Web page designers need to be aware of compatibility issues that arise not just among different versions of the language, but also among different versions of the same browser.
Defining a Style Rule

The general syntax of a CSS style rule is

```
selector {  
    property1: value1;
    property2: value2;
    property3: value3;
...
}
```
Applying a Style Sheet

The design you apply to a Web site is usually a combination of several style sheets.

Figure 3-4  Order in which style sheets are interpreted

- browser’s internal style sheet
- user-defined styles
- author’s external style sheet
- author’s embedded style sheet
- author’s inline styles

accessed last
accessed first
User-Defined Styles

Almost all browsers allow users to modify the default settings of the internal style sheet.
External Style Sheets

Figure 3-6  Linking to an external style sheet

```html
<meta charset="UTF-8" />
<title>Sunny_Acres</title>
<script src="modernizr-1.5.js"></script>
<link href="sa_layout.css" rel="stylesheet" type="text/css" />
</head>
```
Embedded Style Sheets

Another type of style sheet created by a Web page author is an embedded style sheet, in which the styles are inserted directly within the head element of an HTML document using the style element:

```html
<style type="text/css">
    styles
</style>
```

The exact order in which external style sheets and embedded style sheets are processed by the browser depends on the order in which they are listed within the HTML file.
Inline Styles

The very last styles to be interpreted by the browser are inline styles, which are styles applied directly to specific elements using the `style` attribute

```html
<element style="style rules"> ...
</element>
```

It is clear exactly what page element is being formatted

Not recommended in most cases and considered inefficient
Exploring the Style Cascade

As a general rule of thumb, all other things being equal, the more specific style is applied instead of the more general.

An additional factor in applying a style sheet is that properties are passed from a parent element to its children in a process known as style inheritance.

```css
body {color: blue;}

h1 {text-align: center;}
```
If you need browsers to enforce a style, you can append the `!important` keyword to the style property, using the syntax

```markdown
property: value !important;
```

The `!important` keyword is often necessary for visually impaired users who require their pages rendered with large, clear text and highly contrasting colors.
Writing Style Comments

**Figure 3-8** Entering style sheet comments

```css
/*
Sunny Acres Style Sheet
Author: Tammy Nielsen
Date:  3/1/2014
*/
```
Defining Color in CSS

A color value is a numerical expression that describes the properties of a color.

CSS represents these intensities mathematically as a set of numbers called an RGB triplet, which has the format

```
rgb(red, green, blue)
```

CSS also allows RGB values to be entered as hexadecimal numbers

```
#redgreenblue
```
Defining Color in CSS

### Figure 3-10: The 16 basic CSS2 color names

<table>
<thead>
<tr>
<th>Color Name</th>
<th>RGB Triplet</th>
<th>Hexadecimal</th>
<th>Color Name</th>
<th>RGB Triplet</th>
<th>Hexadecimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aqua</td>
<td>(0, 255, 255)</td>
<td>#FFFFF</td>
<td>Navy</td>
<td>(0, 0, 128)</td>
<td>#00080</td>
</tr>
<tr>
<td>Black</td>
<td>(0, 0, 0)</td>
<td>#000000</td>
<td>Olive</td>
<td>(128, 128, 0)</td>
<td>#808000</td>
</tr>
<tr>
<td>Blue</td>
<td>(0, 0, 255)</td>
<td>#000FF</td>
<td>Purple</td>
<td>(128, 0, 128)</td>
<td>#FF0000</td>
</tr>
<tr>
<td>Fuchsia</td>
<td>(255, 0, 255)</td>
<td>#FF00FF</td>
<td>Red</td>
<td>(255, 0, 0)</td>
<td>#C0C0C0</td>
</tr>
<tr>
<td>Gray</td>
<td>(128, 128, 128)</td>
<td>#808080</td>
<td>Silver</td>
<td>(192, 192, 192)</td>
<td>#008080</td>
</tr>
<tr>
<td>Green</td>
<td>(0, 128, 0)</td>
<td>#008000</td>
<td>Teal</td>
<td>(0, 128, 128)</td>
<td>#00FF00</td>
</tr>
<tr>
<td>Lime</td>
<td>(0, 255, 0)</td>
<td>#00FF00</td>
<td>White</td>
<td>(255, 255, 255)</td>
<td>#FFFFF0</td>
</tr>
<tr>
<td>Maroon</td>
<td>(128, 0, 0)</td>
<td>#800000</td>
<td>Yellow</td>
<td>(255, 255, 0)</td>
<td>#FFFF00</td>
</tr>
</tbody>
</table>
Setting Foreground and Background Color

To set the background color of an element, use the property

```
background-color: color;
```

where `color` is a color name or a color value.

To set the foreground or text color of an element, use the following property:

```
color: color;
```
Enhancements to Color in CSS3

CSS3 also supports the Hue Saturation Lightness (HSL) model that describes colors based on hue, saturation, and lightness

\[ \text{hsl}(\text{hue}, \text{saturation}, \text{lightness}) \]
Enhancements to Color in CSS3

Figure 3-14  HSL color saturation model

hsl(38, 90%, 60%)

- red = 0°
- orange at about 38°
- blue = 240°
- green = 120°
- 90% saturation; 60% lightness
- hue expressed as degrees on the color wheel
- saturation varies from 0% to 100%
- lightness varies from 0% to 100%

New Perspectives on HTML, CSS, and Dynamic HTML 5th Edition
CSS3 also allows page designers to augment RGB and HSL color values by specifying a color’s opacity. Opacity defines how much of the colors below the surface of the current object show through to affect its appearance.

\[
\text{rgba}(\text{red}, \text{green}, \text{blue}, \text{opacity})
\]

\[
\text{hsla}(\text{hue}, \text{saturation}, \text{lightness}, \text{opacity})
\]
Enhancements to Color in CSS3

Figure 3-15  Setting a semi-transparent color

```html
h2 {
    background-color: rgb(0, 165, 0);
    color: rgba(255, 255, 255, 0.8);
}
```

Older browsers will display h2 text in white.

White color with 80% opacity.

Figure 3-16  Heading text in semi-transparent white

```
{Hours

- Farm Shop: 9 am - 5 pm Mon - Fri; 9 am - 3 pm Sat
- The Corn Maze: 11 am - 9 pm Sat; 11 am - 5 pm Sun
- The Haunted Maze: 5 pm - 9 pm Fri & Sat
- Petting Barn: 9 am - 4 pm Mon - Fri; 11 am - 3 pm Sat & Sun
}
```

Green background bleeds through the semi-transparent white text.
Selectors and Text Styles

Back for another year, every night the Sunny Acres Corn Maze becomes the Sunny Acres Haunted Maze. This season’s maze begins with an encounter with an evil organist (played by the Council Bluffs Choir’s own Glenn Frye). Explore the maze and find the zombie grave yard, the pumpkin tower, and a haunted house made of corn—not to mention the spiders, ghosts, witches, and other assorted villains that inhabit the maze.

The Haunted Maze is open the last three weekends of October with a special event on Halloween. Come early in the day to explore the larger Corn maze of bats, then participate in the Halloween Festival! Special stations will be set up in the Haunted Maze for trick-or-treaters. A smaller maze is also available for younger children.

HOURS
- Open 5 pm - 9 pm on October 14, 15, 21, 22, 28, and 29
- Halloween Festival October 31 from 3 pm - 9 pm

ADMISSION
- $5.00
- $6.50
- Under 60 cents

New Perspectives on HTML, CSS, and Dynamic HTML 5th Edition
Contextual Selectors

Web pages are structured documents in which elements are nested within other elements, forming a hierarchy of elements.

To create styles that take advantage of this tree structure, CSS allows you to create contextual selectors whose values represent the locations of elements within the hierarchy:

- Parent elements
- Child elements
- Sibling elements
- Descendant elements
Contextual Selectors

**Figure 3-20** Contextual selectors

<table>
<thead>
<tr>
<th>Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>Matches any element in the hierarchy</td>
</tr>
<tr>
<td>e</td>
<td>Matches any element, e, in the hierarchy</td>
</tr>
<tr>
<td>e1, e2, e3, ...</td>
<td>Matches the group of elements e1, e2, e3, ...</td>
</tr>
<tr>
<td>e f</td>
<td>Matches any element, f, that is a descendant of an element, e</td>
</tr>
<tr>
<td>e&gt;f</td>
<td>Matches any element, f, that is a direct child of an element, e</td>
</tr>
<tr>
<td>e+f</td>
<td>Matches any element, f, that is immediately preceded by a sibling element, e</td>
</tr>
<tr>
<td>e-f</td>
<td>Matches any element, f, that is a sibling to an element, e</td>
</tr>
</tbody>
</table>

**Figure 3-22** Applying a contextual selector

```html
section h1 {
  background-color: rgb(125, 186, 240);
  color: white;
}
```

style rule applied only to h1 headings nested within section elements

New Perspectives on HTML, CSS, and Dynamic HTML 5th Edition
Attribute Selectors

Selectors also can be defined based on attributes and attribute values associated with elements.

Two attributes, *id* and *class*, are often key in targeting styles to a specific element or group of elements.
## Attribute Selectors

<table>
<thead>
<tr>
<th>Selector</th>
<th>Description</th>
<th>Example</th>
<th>Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>#id</td>
<td>The element with the id value, id</td>
<td>#intro</td>
<td>The element with the id intro</td>
</tr>
<tr>
<td>.class</td>
<td>All elements with the class value, class</td>
<td>.main</td>
<td>All elements belonging to the main class</td>
</tr>
<tr>
<td>elem.class</td>
<td>All elem elements with the class value class</td>
<td>p.main</td>
<td>All paragraphs belonging to the main class</td>
</tr>
<tr>
<td>elem[att]</td>
<td>All elem elements containing the att attribute</td>
<td>a[href]</td>
<td>All hypertext elements containing the href attribute</td>
</tr>
<tr>
<td>elem[att=&quot;text&quot;]</td>
<td>All elem elements whose att attribute equals text</td>
<td>a[href=&quot;gloss.htm&quot;]</td>
<td>All hypertext elements whose href attribute equals gloss.htm</td>
</tr>
<tr>
<td>elem[att-=&quot;text&quot;]</td>
<td>All elem elements whose att attribute contains the word text</td>
<td>a[rel-=&quot;glossary&quot;]</td>
<td>All hypertext elements whose rel attribute contains the word glossary</td>
</tr>
<tr>
<td>elem[att /=&quot;text&quot;]</td>
<td>All elem elements whose att attribute value is a hyphen-separated list of words beginning with text</td>
<td>p[id]=&quot;first&quot;</td>
<td>All paragraphs whose id attribute starts with the word first in a hyphen-separated list of words</td>
</tr>
<tr>
<td>elem[att=&quot;text&quot;]</td>
<td>All elem elements whose att attribute begins with text (CSS3)</td>
<td>a[rel=&quot;prev&quot;]</td>
<td>All hypertext elements whose rel attribute begins with prev</td>
</tr>
<tr>
<td>elem[att$=&quot;text&quot;]</td>
<td>All elem elements whose att attribute ends with text (CSS3)</td>
<td>a[href$=&quot;org&quot;]</td>
<td>All hypertext elements whose href attribute ends with org</td>
</tr>
<tr>
<td>elem[att^=&quot;text&quot;]</td>
<td>All elem elements whose att attribute contains the value text (CSS3)</td>
<td>a[href^=&quot;faq&quot;]</td>
<td>All hypertext elements whose href attribute contains the text string faq</td>
</tr>
</tbody>
</table>
Styling Web Page Text

The default font used by most browsers is Times New Roman, but you can specify a different font for any page element using the property

```css
font-family: fonts;
```

fonts is a comma-separated list of specific or generic font names.
Styling Web Page Text

**Figure 3-26** Web safe fonts

<table>
<thead>
<tr>
<th>Font</th>
<th>Font Family</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arial</td>
<td>font-family: Arial, Helvetica, sans-serif;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Arial Black</td>
<td>font-family: 'Arial Black', gadget, sans-serif;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Century Gothic</td>
<td>font-family: 'Century Gothic', sans-serif;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Comic Sans MS</td>
<td>font-family: 'Comic Sans MS', cursive;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Courier New</td>
<td>font-family: 'Courier New', Courier, monospace;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Georgia</td>
<td>font-family: Georgia, serif;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Impact</td>
<td>font-family: Impact, Charcoal, sans-serif;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Lucida Console</td>
<td>font-family: 'Lucida Console', Monaco, monospace;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Palatino Linotype</td>
<td>font-family: 'Palatino Linotype', 'Book Antiqua', Palatino, serif;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Tahoma</td>
<td>font-family: Tahoma, Geneva, sans-serif;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Times New Roman</td>
<td>font-family: 'Times New Roman', Times, serif;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Trebuchet MS</td>
<td>font-family: 'Trebuchet MS', Helvetica, sans-serif;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
<tr>
<td>Verdana</td>
<td>font-family: Verdana, Geneva, sans-serif;</td>
<td>abcdefghijklmnopqrstuvwxyz1234567890</td>
</tr>
</tbody>
</table>
Setting Font Face and Sizes

To define a font face, use the style property

\[
\text{font-family: fonts;}
\]

where *fonts* is a comma-separated list of fonts that the browser can use with the element. List specific fonts first and complete the list with a generic font.

To set a font size, use the style property

\[
\text{font-size: size;}
\]

where *size* is a CSS unit of length in either relative or absolute units.
Setting Font Face and Sizes

To set kerning (the space between letters), use the following style property:

```
letter-spacing: size;
```

To set tracking (the space between words), use the following style property:

```
word-spacing: size;
```
Setting the Line Height

Figure 3-31  Setting the line height

line-height set to 0.75em

font-family: sans-serif; font-size: 2em; letter-spacing: 0.3em; word-spacing: 0.8em; line-height: 0.75em;
Setting Font and Text Appearance

To specify the font style, use

```
font-style: type;
```

where `type` is `normal`, `italic`, or `oblique`.

To specify the font weight, use

```
font-weight: type;
```

where `type` is `normal`, `bold`, `bolder`, `light`, `lighter`, or a font weight value.
Setting Font and Text Appearance

To specify a text decoration, use

text-decoration: type;

where type is none, underline, overline, or line-through.

To transform text, use

text-transform: type;

where type is capitalize, uppercase, lowercase, or none.

To display a font variant of text, use

font-variant: type;

where type is normal or small-caps.
## Aligning Text Vertically

### Figure 3-35

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>Aligns the element with the bottom of lowercase letters in surrounding text (the default)</td>
</tr>
<tr>
<td>bottom</td>
<td>Aligns the bottom of the element with the bottom of the lowest element in surrounding content</td>
</tr>
<tr>
<td>middle</td>
<td>Aligns the middle of the element with the middle of the surrounding content</td>
</tr>
<tr>
<td>sub</td>
<td>Subscripts the element</td>
</tr>
<tr>
<td>super</td>
<td>Superscripts the element</td>
</tr>
<tr>
<td>text-bottom</td>
<td>Aligns the bottom of the element with the bottom of the font of the surrounding content</td>
</tr>
<tr>
<td>text-top</td>
<td>Aligns the top of the element with the top of the font of the surrounding content</td>
</tr>
<tr>
<td>top</td>
<td>Aligns the top of the element with the top of the tallest object in the surrounding content</td>
</tr>
</tbody>
</table>
Combining All Text Formatting in a Single Style

You can combine most of the text and font styles into a single property using the shortcut font property

```
font: font-style font-variant font-weight font-size/line-height font-family;
```
Combining All Text Formatting in a Single Style

**Figure 3-36** Designing the footer address

```javascript
/* Footer styles */
footer address {
  background-color: rgb(55, 102, 55);
  color: white;
  color: rgba(255, 255, 255, 0.8);
  font: normal small-caps 0.8em/4em 'Times New Roman', Times, serif;
  text-align: center;
}
```

- set the background color to dark and the text color to white or semi-transparent white
- shortcut font property
- center the text horizontally

**Figure 3-37** Reformatted address text

**Directions**

- From Council Bluffs, proceed east on I-80
- Take Exit 38 North to the Drake Frontage Road
- Turn right on Highway G
- Proceed east for 2.5 miles
- Sunny Acres is on your left; watch for the green sign

**Sunny Acres**  TAMMY & BRENT NIELSEN  1977 HIGHWAY G  COUNCIL BLUFFS, IA  51503
Working with Web Fonts

Figure 3-39  Inserting a Web Font

```css
@font-face {
  font-family: 'MobileRegular';
  src: url('mobile-webfont.eot');
  src: local('e'),
    url('mobile-webfont.woff') format('woff'),
    url('mobile-webfont.ttf') format('truetype'),
    url('mobile-webfont.svg#mobile-webfont') format('svg');
}

/* Body styles */
body {
  background-color: white;
  font-family: MobileRegular, Verdana, Geneva, sans-serif;
  line-height: 1.4em;
}

/* Heading styles */
section h1 {
  background-color: rgb(125, 186, 240);
  color: white;
  font-size: 1.7em;
  letter-spacing: 0.4em;
  line-height: 1.8em;
  text-indent: 1em;
}

h2 {
  background-color: rgb(0, 165, 0);
  color: white;
  color: rgba(255, 255, 255, 0.8);
  letter-spacing: 0.4em;
  line-height: 1.8em;
  text-indent: 1em;
}
```

instructs browsers to use the Web font first, if available, as the default for all body text

set the line height to accommodate the new font
# Understanding the CSS @rules

## Figure 3-41  CSS @rules

<table>
<thead>
<tr>
<th>@rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>@charset &quot;encoding&quot;;</code></td>
<td>Defines the character encoding used in an external style sheet where <code>encoding</code> is the name of the character set</td>
</tr>
<tr>
<td><code>@import url(url) media</code></td>
<td>Imports an external style sheet file located at <code>url</code>. The optional <code>media</code> attribute provides a comma-separated list of media devices to be used with the style sheet</td>
</tr>
<tr>
<td><code>@media media {</code></td>
<td></td>
</tr>
<tr>
<td><code>styles</code></td>
<td></td>
</tr>
<tr>
<td><code>}</code></td>
<td></td>
</tr>
<tr>
<td><code>@page location {</code></td>
<td></td>
</tr>
<tr>
<td><code>margins</code></td>
<td>Defines the page margins for printed output where <code>location</code> is either left, right, or first for left page, right page, or first page, and <code>margins</code> set the margin widths</td>
</tr>
<tr>
<td><code>}</code></td>
<td></td>
</tr>
<tr>
<td><code>@font-face {</code></td>
<td></td>
</tr>
<tr>
<td><code>font_description</code></td>
<td>Defines the properties of a custom Web font where <code>font_description</code> indicates the source and features of the font</td>
</tr>
<tr>
<td><code>}</code></td>
<td></td>
</tr>
<tr>
<td><code>@namespace prefix uri</code></td>
<td>Defines an XML namespace where <code>prefix</code> is the namespace prefix and <code>uri</code> is the location of the namespace</td>
</tr>
</tbody>
</table>
Lists and Pseudo-Items

The first-of-type structural pseudo-class matches the element in a collection.

The text-transform property transforms the text in uppercase, lowercase, or sentence case.

The first letter pseudo-element matches the first letter of the element.

An initial cap effect occurs when the first letter of the first line appears larger than the surrounding text.

The first-line pseudo-element matches the first line of the element.

The hover dynamic pseudo-class matches when a user is hovering the mouse pointer over the element.

The list-style-type property specifies the marker that is displayed with the list.

The list-style-image property specifies an image to be displayed as the marker.
Designing Styles for Lists

To change the marker displayed in ordered or unordered lists, you apply the style

```
list-style-type: type;
```

<table>
<thead>
<tr>
<th>list-style-type</th>
<th>Marker(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>disc</td>
<td>●</td>
</tr>
<tr>
<td>circle</td>
<td>○</td>
</tr>
<tr>
<td>square</td>
<td>■</td>
</tr>
<tr>
<td>decimal</td>
<td>1, 2, 3, 4, ...</td>
</tr>
<tr>
<td>decimal-leading-zero</td>
<td>01, 02, 03, 04, ...</td>
</tr>
<tr>
<td>lower-roman</td>
<td>i, ii, iii, iv, ...</td>
</tr>
<tr>
<td>upper-roman</td>
<td>I, II, III, IV, ...</td>
</tr>
<tr>
<td>lower-alpha</td>
<td>a, b, c, d, ...</td>
</tr>
<tr>
<td>upper-alpha</td>
<td>A, B, C, D, ...</td>
</tr>
<tr>
<td>lower-greek</td>
<td>α, β, γ, δ, ...</td>
</tr>
<tr>
<td>upper-greek</td>
<td>A, B, Γ, Δ, ...</td>
</tr>
<tr>
<td>none</td>
<td>no marker displayed</td>
</tr>
</tbody>
</table>
Designing a List

To define the appearance of the list marker, use the style

```css
list-style-type: type;
```

where `type` is `disc`, `circle`, `square`, `decimal`, `decimal-leading-zero`, `lower-roman`, `upper-roman`, `lower-alpha`, `upper-alpha`, `lower-greek`, `upper-greek`, or `none`.

To insert a graphic image as a list marker, use the style

```css
list-style-image: url(url);
```

where `url` is the URL of the graphic image file.
Designing a List

To set the position of list markers, use the style:

```css
list-style-position: position;
```

where `position` is inside or outside.

To define all of the list style properties in a single style, use the following style:

```css
list-style: type url(url) position;
```

To set the indentation of a list, apply the style:

```css
padding-left: size;
```

where `size` is the length that the list should be indented.
Using Pseudo-Classes and Pseudo-Elements

A pseudo-class is a classification of an element based on its current status, position, or use in the document:

```
selector:pseudo-class { styles; }
```

<table>
<thead>
<tr>
<th>Pseudo-Class</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>link</td>
<td>The link has not yet been visited by the user.</td>
<td>a:link {color: red;}</td>
</tr>
<tr>
<td>visited</td>
<td>The link has been visited by the user.</td>
<td>a:visited {color: green;}</td>
</tr>
<tr>
<td>active</td>
<td>The element is in the process of being activated or clicked by the user.</td>
<td>a:active {color: yellow;}</td>
</tr>
<tr>
<td>hover</td>
<td>The mouse pointer is hovering over the element.</td>
<td>a:hover {color: blue;}</td>
</tr>
<tr>
<td>focus</td>
<td>The element has received the focus of the keyboard or mouse pointer.</td>
<td>input:focus {background-color: yellow;}</td>
</tr>
</tbody>
</table>
# Using Pseudo-Classes and Pseudo-Elements

## Figure 3-54 Structural pseudo-classes

<table>
<thead>
<tr>
<th>Pseudo-Class</th>
<th>Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>root</td>
<td>The top element in the document hierarchy (the html element)</td>
</tr>
<tr>
<td>empty</td>
<td>An element with no children</td>
</tr>
<tr>
<td>only-child</td>
<td>An element with no siblings</td>
</tr>
<tr>
<td>first-child</td>
<td>The first child of the parent element</td>
</tr>
<tr>
<td>last-child</td>
<td>The last child of the parent element</td>
</tr>
<tr>
<td>first-of-type</td>
<td>The first element of the parent that matches the specified type</td>
</tr>
<tr>
<td>last-of-type</td>
<td>The last element of the parent that matches the specified type</td>
</tr>
<tr>
<td>nth-of-type(n)</td>
<td>The n&lt;sup&gt;th&lt;/sup&gt; element of the parent of the specified type</td>
</tr>
<tr>
<td>nth-last-of-type(n)</td>
<td>The n&lt;sup&gt;th&lt;/sup&gt; from the last element of the parent of the specified type</td>
</tr>
<tr>
<td>only-of-type</td>
<td>An element that has no siblings of the same type</td>
</tr>
<tr>
<td>lang(code)</td>
<td>The element that has the specified language indicated by code</td>
</tr>
<tr>
<td>not(s)</td>
<td>An element not matching the specified selector, s</td>
</tr>
</tbody>
</table>

New Perspectives on HTML, CSS, and Dynamic HTML 5th Edition
# Pseudo-Elements

## Figure 3-56  Pseudo-elements

<table>
<thead>
<tr>
<th>Pseudo-Element</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>first-letter</td>
<td>The first letter of the element text</td>
<td>p:first-letter {font-size:200%}</td>
</tr>
<tr>
<td>first-line</td>
<td>The first line of the element text</td>
<td>p:first-line {text-transform: uppercase}</td>
</tr>
<tr>
<td>before</td>
<td>Content inserted directly before the element</td>
<td>p:before {content:“Special!“}</td>
</tr>
<tr>
<td>after</td>
<td>Content inserted directly after the element</td>
<td>p:after {content:“eof“}</td>
</tr>
</tbody>
</table>