Web Development

AAPP009-4-2 & Version #VD1

Introduction to CSS

(*Cascading Style Sheet*)

Topic & Structure of The Lesson

- The rationale for CSS
- The syntax of CSS
- Where CSS styles can be located
- The different types of CSS selectors
- What the CSS cascade is and how it works
- The CSS box model
- CSS text styling

Learning Outcomes

- **At the end of this topic, You should be able to**
  - Understand the syntax of CSS and where the CSS styles can be located
  - Identify the different types of CSS selectors.
  - Understand what the CSS cascade is and how it works.

Key Terms You Must Be Able To Use

- If you have mastered this topic, you should be able to use the following terms correctly in your assignments and exam:

- ✓ absolute units
- ✓ attribute selector
- ✓ author-created style
- ✓ sheets
- ✓ browser style sheets
- ✓ cascade
- ✓ class selector
- ✓ collapsing margins
- ✓ combinators
- ✓ contextual selector
- ✓ CSS
- ✓ CSS3 modules
- ✓ declaration
- ✓ declaration block
- ✓ descendant selector
- ✓ element box
- ✓ element selectors
- ✓ em units
- ✓ embedded style sheets
- ✓ external style sheets
- ✓ generic font
- ✓ grouped selector
- ✓ id selector
- ✓ inheritance
- ✓ inline styles
- ✓ internal styles
- ✓ location
- ✓ percentages
- ✓ presentation
- ✓ property:value pair
- ✓ pseudo-class selector
- ✓ pseudo-element selector
- ✓ relative units
- ✓ rem
- ✓ responsive design
- ✓ selector
- ✓ specificity
- ✓ style rules
- ✓ TRouBLe
- ✓ universal element selector
- ✓ user style sheets
- ✓ vendor prefixes
- ✓ web font stack
- ✓ x-height

What is CSS?

- CSS is a W3C standard for describing the appearance of HTML elements.
- Another common way to describe CSS’s function is to say that CSS is used to define the **presentation** of HTML documents.
- With CSS, we can assign font properties, colors, sizes, borders, background images, and even position elements on the page.
- CSS can be added directly to any HTML element (via the style attribute), within the <head> element, or, most commonly, in a separate text file that contains only CSS.

Benefits of CSS

- **Improved control over formatting.**
  - The degree of formatting control in CSS is significantly better than that provided in HTML. CSS gives web authors fine-grained control over the appearance of their web content.

- **Improved site maintainability.**
  - Websites become significantly more maintainable because all formatting can be centralized into one CSS file, or a small handful of them. This allows you to make site-wide visual modifications by changing a single file.

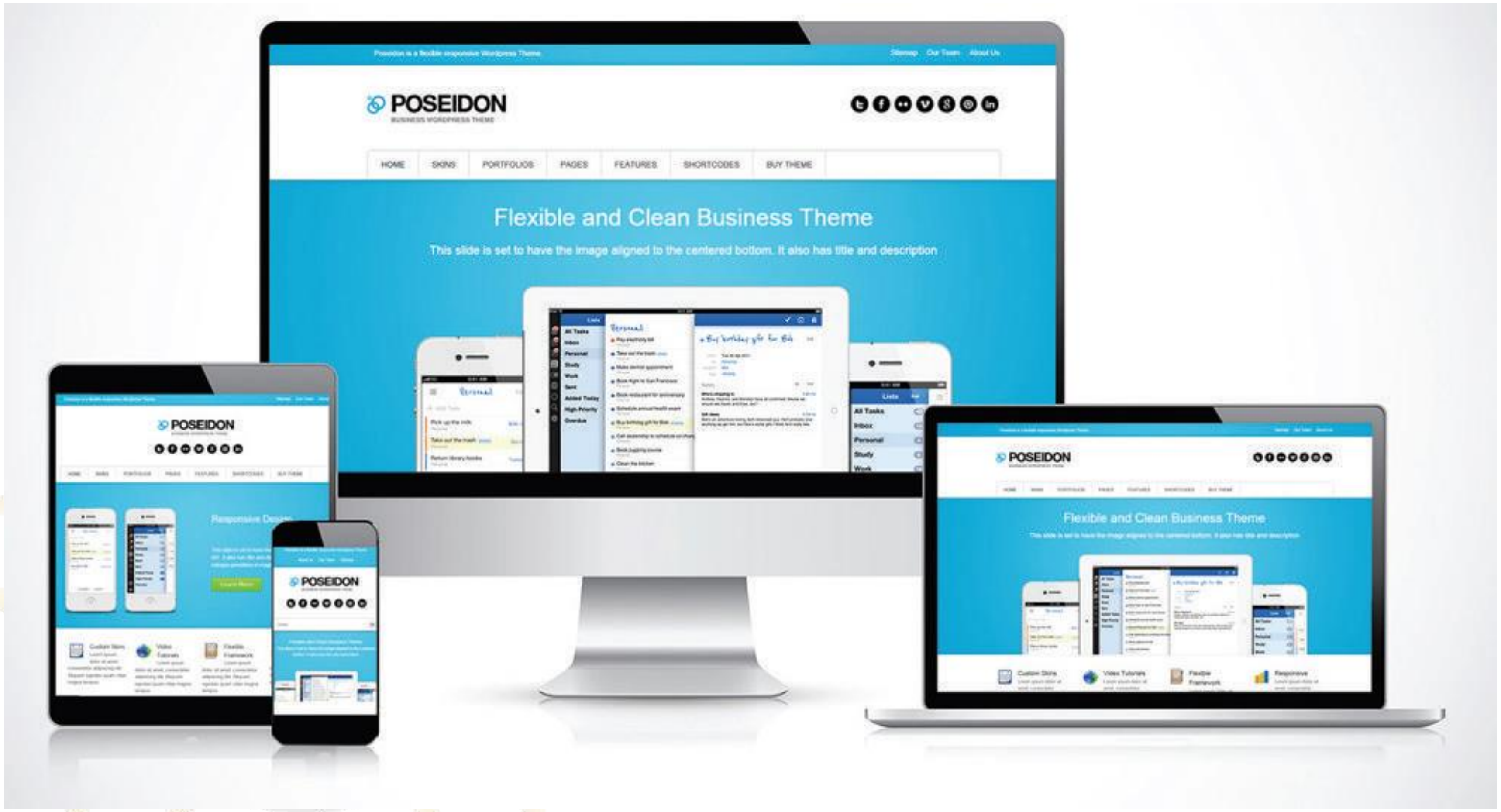
Benefits of CSS

- **Improved accessibility.**
  - CSS-driven sites are more accessible. By keeping presentation out of the HTML, screen readers and other accessibility tools work better, thereby providing a significantly enriched experience for those reliant on accessibility tools.
- **Improved page download speed.**
  - A site built using a centralized set of CSS files for all presentation will also be quicker to download because each individual HTML file will contain less style information and markup, and thus be smaller.

Benefits of CSS

- **Improved output flexibility.**
  - CSS can be used to adopt a page for different output media. This approach to CSS page design is often referred to as **responsive design**. Page 9 illustrates a site that responds to different browser and window sizes.

Benefits of CSS


POSEIDON
Flexible and Clean Business Theme
This slide is set to have the image aligned to the centered bottom. It also has title and description

CSS Versions

- In the early 1990s, a variety of different style sheet standards were proposed, including JavaScript style sheets, which was proposed by Netscape in 1996.
- By the end of 1996 the CSS Level 1 Recommendation was published.
- A year later, the CSS Level 2 Recommendation (also more succinctly labeled simply as CSS2) was published.
- Even though work began over a decade ago, an updated version of the Level 2 Recommendation, CSS2.1, did not become an official W3C Recommendation until June 2011.

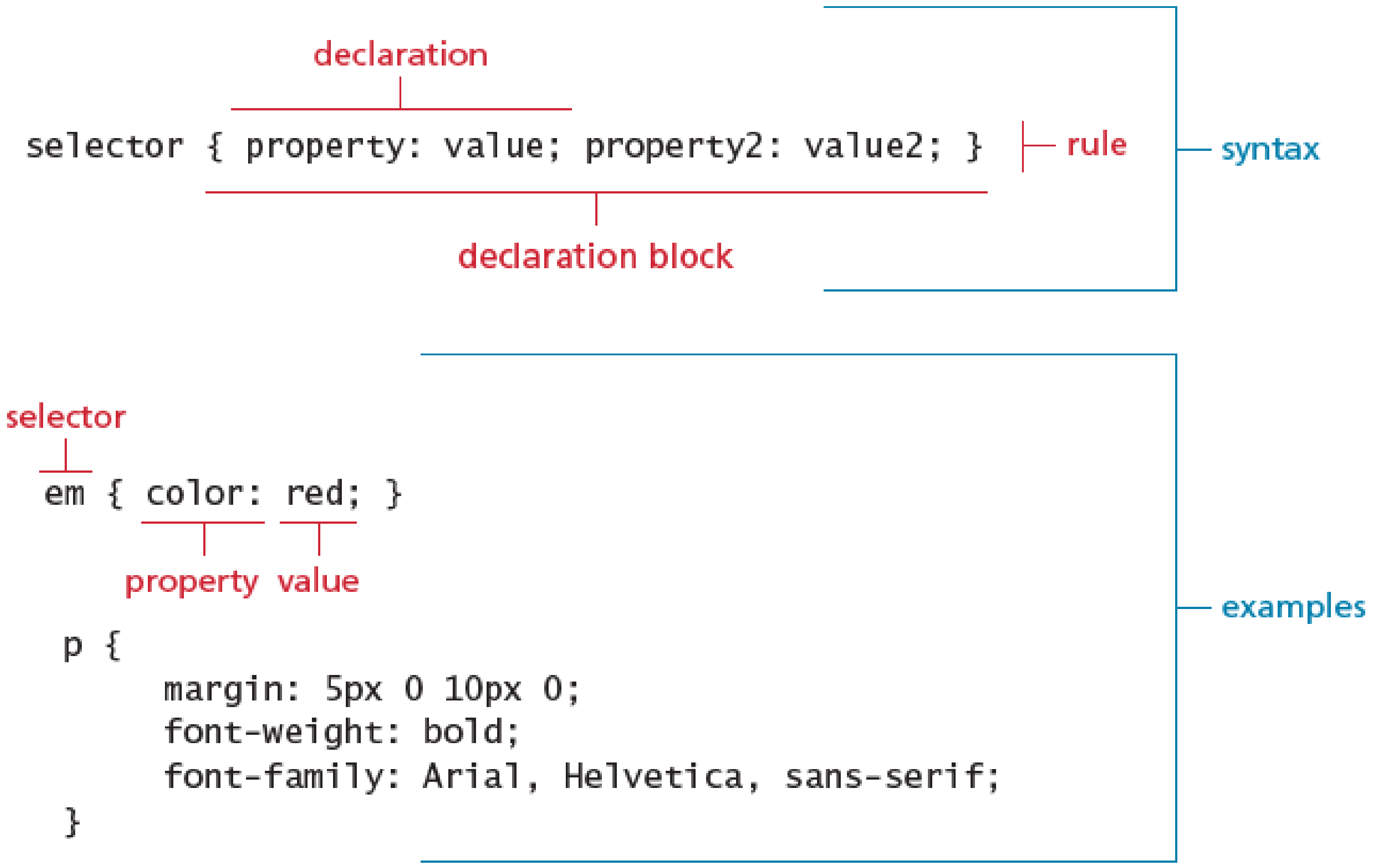
CSS Versions

- And to complicate matters even more, all through the last decade (and to the present day as well), during the same time the CSS2.1 standard was being worked on, a different group at the W3C was working on a CSS3 draft.
- To make CSS3 more manageable for both browser manufacturers and web designers, the W3C has subdivided it into a variety of different **CSS3 modules**.
- So far the following CSS3 modules have made it to official W3C Recommendations: ***CSS Selectors, CSS Namespaces, CSS Media Queries, and CSS Color***.

CSS Syntax

- A CSS document consists of one or more **style rules**.
- A rule consists of a **selector** that identifies the HTML element or elements that will be affected, followed by a series of **property:value pairs** (each pair is also called a **declaration**).
- The series of declarations is also called the **declaration block**.
- A declaration block can be together on a single line, or spread across multiple lines.
- The browser ignores white space (i.e., spaces, tabs, and returns) between your CSS rules so you can format the CSS however you want.

Figure 1: CSS syntax



```
selector { property: value; property2: value2; }
```

```
em { color: red; }
```

```
p {
      margin: 5px 0 10px 0;
      font-weight: bold;
      font-family: Arial, Helvetica, sans-serif;
}
```

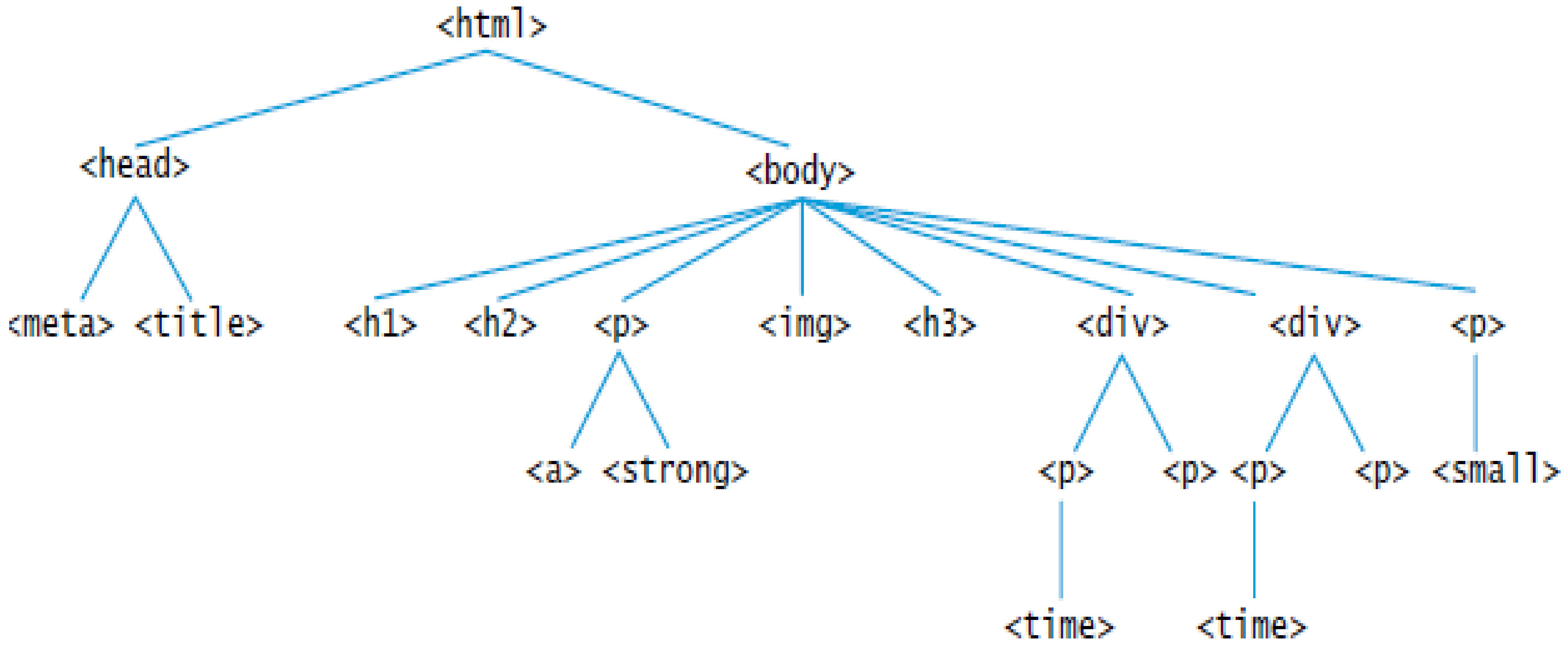
CSS Syntax

- Notice that each declaration is terminated with a semicolon.
- The semicolon for the last declaration in a block is in fact optional.
- However, it is sensible practice to also terminate the last declaration with a semicolon as well; that way, if you add rules to the end later, you will reduce the chance of introducing a rather subtle and hard-to-discover bug.

CSS Selectors

- Every CSS rule begins with a **selector**. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule.
- Another way of thinking of selectors is that they are a pattern that is used by the browser to select the HTML elements that will receive the style.

HTML DOCUMENT OUTLINE /TREE



CSS Selectors

- When defining CSS rules, you will need to first use a selector to tell the browser which elements will be affected by the property values.
- CSS selectors allow you to select individual or multiple HTML elements.
- Type of selectors:
    1. Element Selectors
    2. Class Selectors
    3. Id Selectors
    4. Attribute Selectors
    5. Pseudo-Element and Pseudo-Class Selectors
    6. Contextual Selectors

CSS – Element Selectors

- **Element selectors** select all instances of a given HTML element.
- The example CSS rules in Figure 1 in page 13 illustrate two element selectors.
- You can select all elements by using the **universal element selector**, which is the * (asterisk) character.
- You can select a group of elements by separating the different element names with commas.
- This is a sensible way to reduce the size and complexity of your CSS files, by combining multiple identical rules into a single rule.

CSS – Element Selectors

- An example **grouped selector** is shown as below, along with its equivalent as three separate rules.

```
/* commas allow you to group selectors */
p, div, aside {
   margin: 0;
   padding: 0;
}
/* the above single grouped selector is equivalent to the
   following: */
p {
   margin: 0;
   padding: 0;
}
div {
   margin: 0;
   padding: 0;
}
aside {
   margin: 0;
   padding: 0;
}
```

CSS – Class Selectors

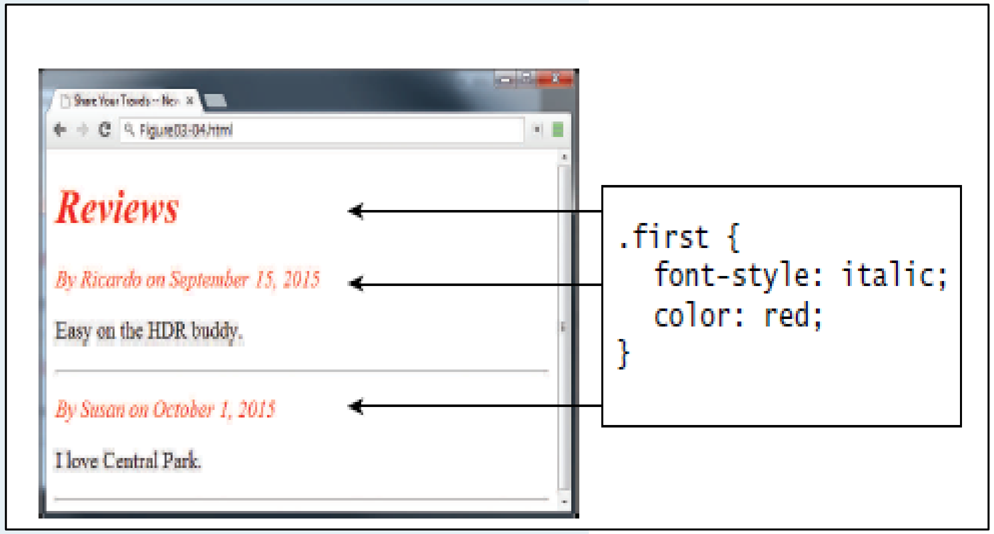
- A **class selector** allows you to simultaneously target different HTML elements regardless of their position in the document tree.
- If a series of HTML elements have been labeled with the same class attribute value, then you can target them for styling by using a class selector, which takes the form: period (.) followed by the class name.
- Figure 2 illustrates an example of styling using a class selector. The result in the browser is shown in Figure 3.

CSS – Class Selectors

- A **class selector** allows you to simultaneously target different HTML elements regardless of their position in the document tree.
- If a series of HTML elements have been labeled with the same class attribute value, then you can target them for styling by using a class selector, which takes the form: period (.) followed by the class name.
- Figure 2 illustrates an example of styling using a class selector. The result in the browser is shown in Figure 2.

Figure 2: Class selector example

```
<head>
   <title>Share Your Travels </title>
  <style>
         .first {
          font-style: italic;
          color: red;
         }
   </style>
</head>
<body>
   <h1 class="first">Reviews</h1>
   <div>
      <p class="first">By Ricardo on <time>September 15, 2015</time></p>
      <p>Easy on the HDR buddy.</p>
   </div>
   <hr/>

   <div>
      <p class="first">By Susan on <time>October 1, 2015</time></p>
      <p>I love Central Park.</p>
   </div>
   <hr/>
</body>
```



Reviews

By Ricardo on September 15, 2015

Easy on the HDR buddy.

By Susan on October 1, 2015

I love Central Park.

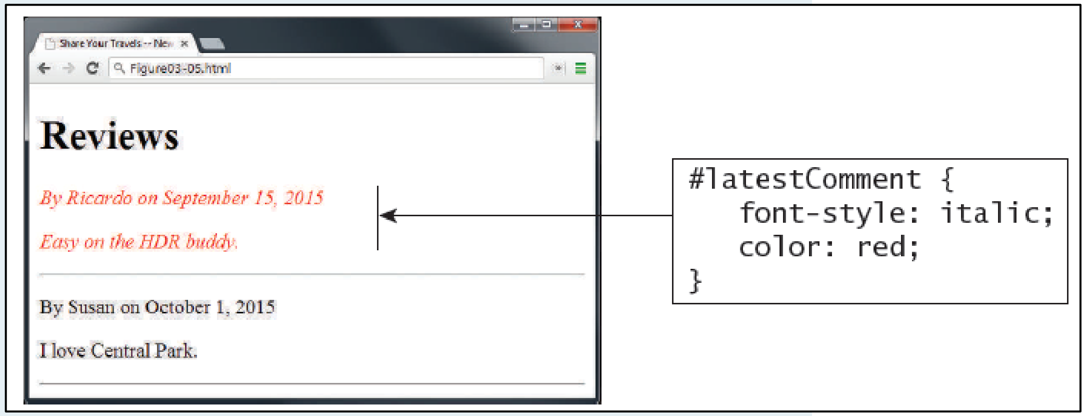
```
.first {
   font-style: italic;
   color: red;
}
```

CSS – ID Selector

- An **id selector** allows you to target a specific element by its id attribute regardless of its type or position.
- If an HTML element has been labeled with an id attribute, then you can target it for styling by using an id selector, which takes the form: pound/hash (#) followed by the id name.
- Figure 3 illustrates an example of styling using an id selector. The result in the browser is shown in Figure 3.

Figure 3: Id selector example

```
<head lang="en">
   <meta charset="utf-8">
   <title>Share Your Travels -- New York - Central Park</title>
  <style>
         #latestComment {
          font-style: italic;
          color: red;
         }
</style>
</head>
<body>
   <h1>Reviews</h1>
   <div id="latestComment">
      <p>By Ricardo on <time>September 15, 2015</time></p>
      <p>Easy on the HDR buddy.</p>
   </div>
   <hr/>

   <div>
      <p>By Susan on <time>October 1, 2015</time></p>
      <p>I love Central Park.</p>
   </div>
   <hr/>

</body>
```



Reviews

By Ricardo on September 15, 2015

Easy on the HDR buddy.

By Susan on October 1, 2015

I love Central Park.

```
#latestComment {
   font-style: italic;
   color: red;
}
```

CSS – Attribute Selectors

- An **attribute selector** provides a way to select HTML elements either by the presence of an element attribute or by the value of an attribute.
- This can be a very powerful technique, but because of uneven support by some of the browsers, not all web authors have used them.
- Attribute selectors can be a very helpful technique in the styling of hyperlinks and images.
- For instance, perhaps we want to make it more obvious to the user when a pop-up tooltip is available for a link or image.

CSS – Attribute Selectors

- We can do this by using the following attribute selector:

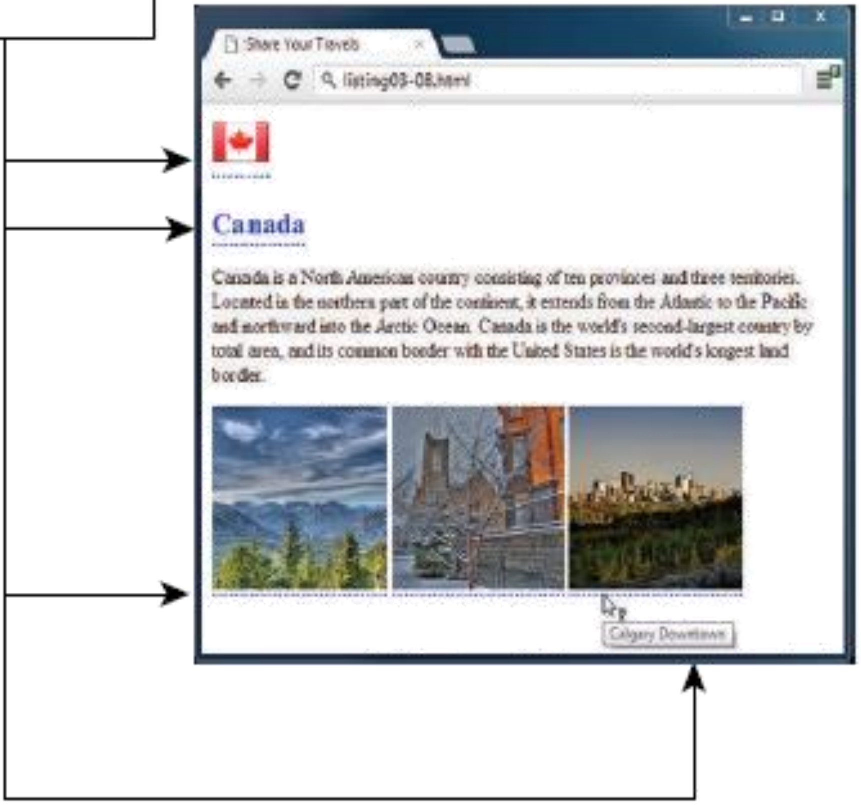
```
[title] { … }
```

- This will match any element in the document that has a title attribute.

Figure 4: Attribute selector example

```
<head lang="en">
   <meta charset="utf-8">
   <title>Share Your Travels</title>
     <style>
          [title] {
          cursor: help;
          padding-bottom: 3px;
          border-bottom: 2px dotted blue;
          text-decoration: none;
        }
   </style>
</head>
<body>
   <div>
      <img src="images/flags/CA.png" title="Canada Flag" />
      <h2><a href="countries.php?id=CA" title="see posts from Canada">
          Canada</a>
      </h2>
      <p>Canada is a North American country consisting of ... </p>
      <div>
          <img src="images/square/6114907897.jpg"
               title="At top of Sulphur Mountain" />
          <img src="images/square/6592317633.jpg"
               title="Grace Presbyterian Church" />
          <img src="images/square/6592914823.jpg"
               title="Calgary Downtown" />
     </div>
   </div>
</body>
```

```
[title] {
   cursor: help;
   padding-bottom: 3px;
   border-bottom: 2px dotted blue;
   text-decoration: none;
}
```



Canada

Canada is a North American country consisting of ten provinces and three territories. Located in the northern part of the continent, it extends from the Atlantic to the Pacific and northward into the Arctic Ocean. Canada is the world's second-largest country by total area, and its common border with the United States is the world's longest land border.

CSS – Attribute Selectors

Selector	Matches	Example
[]	A specific attribute.	`[title]` Matches any element with a title attribute
[=]	A specific attribute with a specific value.	`a[title="posts from this country"]` Matches any <a> element whose title attribute is exactly "posts from this country"
[~=]	A specific attribute whose value matches at least one of the words in a space-delimited list of words.	`[title~="Countries"]` Matches any `title` attribute that contains the word "Countries"
[^=]	A specific attribute whose value begins with a specified value.	`a[href^="mailto"]` Matches any <a> element whose `href` attribute begins with "mailto"
[*=]	A specific attribute whose value contains a substring.	`img[src*="flag"]` Matches any <img> element whose `src` attribute contains somewhere within it the text "flag"
[$=]	A specific attribute whose value ends with a specified value.	`a[href$=".pdf"]` Matches any <a> element whose `href` attribute ends with the text ".pdf"

CSS – Pseudo-Element and Pseudo-Class Selectors

- A **pseudo-element selector** is a way to select something that does not exist explicitly as an element in the HTML document tree but which is still a recognizable selectable object.
- For instance, you can select the first line or first letter of any HTML element using a pseudo-element selector.
- A **pseudo-class selector** does apply to an HTML element, but targets either a particular state or, in CSS3, a variety of family relationships.

Common Pseudo-Class and Pseudo-Element Selectors

Selector	Type	Description
`a:link`	pseudo-class	Selects links that have not been visited
`a:visited`	pseudo-class	Selects links that have been visited
`:focus`	pseudo-class	Selects elements (such as text boxes or list boxes) that have the input focus.
`:hover`	pseudo-class	Selects elements that the mouse pointer is currently above.
`:active`	pseudo-class	Selects an element that is being activated by the user. A typical example is a link that is being clicked.
`:checked`	pseudo-class	Selects a form element that is currently checked. A typical example might be a radio button or a check box.
`:first-child`	pseudo-class	Selects an element that is the first child of its parent. A common use is to provide different styling to the first element in a list.
`:first-letter`	pseudo-element	Selects the first letter of an element. Useful for adding drop-caps to a paragraph.
`:first-line`	pseudo-element	Selects the first line of an element.

CSS – Pseudo-Element and Pseudo-Class Selectors

- The most common use of this type of selectors is for targeting link states.
- By default, the browser displays link text blue and visited text links purple.
- Figure 5 illustrates the use of pseudo-class selectors to style not only the visited and unvisited link colors, but also the hover color, which is the color of the link when the mouse is over the link.
- Do be aware that this state does not occur on touch screen devices.

Figure 5: Styling a link using pseudo-class selectors

```
<head>
   <title>Share Your Travels</title>
  <style>
       a:link {
       text-decoration: underline;
       color: blue;
     }
       a:visited {
       text-decoration: underline;
       color: purple;
     }
       a:hover {
       text-decoration: none;
       font-weight: bold;
     }
       a:active {
       background-color: yellow;
     }
  </style>
</head>
<body>
    <p>Links are an important part of any web page. To learn more about
       links visit the <a href="#">W3C</a> website.</p>
   <nav>
     <ul>
       <li><a href="#">Canada</a></li>
       <li><a href="#">Germany</a></li>
       <li><a href="#">United States</a></li>
     </ul>
   </nav>
</body>
```

CSS – Pseudo-Element and Pseudo-Class Selectors

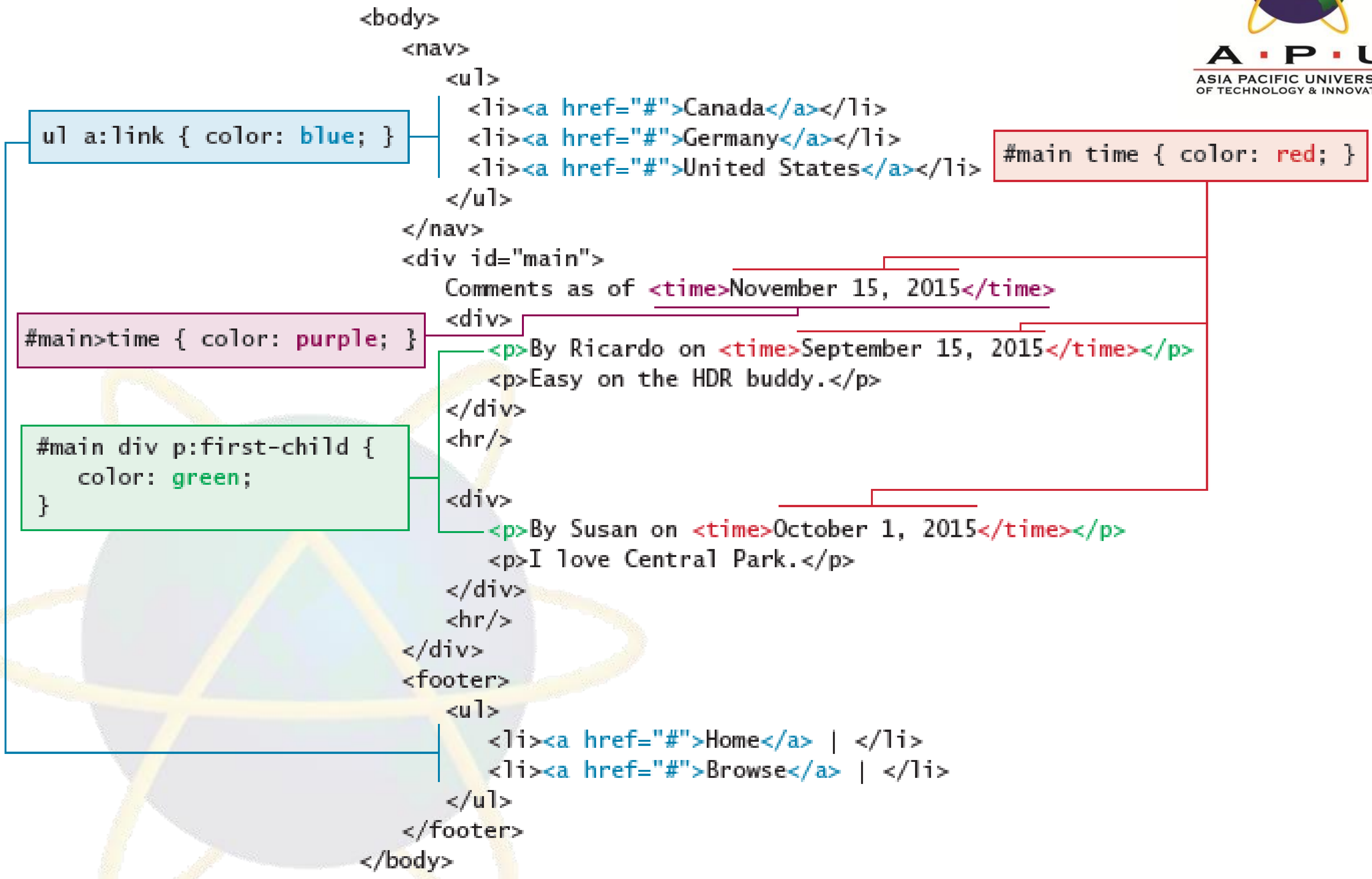
- Note the syntax of pseudo-class selectors: the colon (:) followed by the pseudo-class selector name.
- Do be aware that a space is *not* allowed after the colon.
- Believe it or not, the order of these pseudo-class elements is important.
- The :link and :visited pseudo-classes should appear before the others.
- Some developers use a mnemonic to help them remember the order.
- My favorite is “Lord Vader, Former Handle Anakin” for Link, Visited, Focus, Hover, Active.

CSS – Contextual Selectors

- A **contextual selector** (in CSS3 also called **combinators**) allows you to select elements based on their *ancestors*, *descendants*, or *siblings*.
- That is, it selects elements based on their context or their relation to other elements in the document tree.
- While some of these contextual selectors are used relatively infrequently, almost all web authors find themselves using descendant selectors.
- A **descendant selector** matches all elements that are contained within another element.
- The character used to indicate descendant selection is the space character.

CSS – Contextual Selectors

Selector	Matches	Example
Descendant	A specified element that is contained somewhere within another specified element.	`div p` Selects a `<p>` element that is contained somewhere within a `<div>` element. That is, the `<p>` can be any descendant, not just a child.
Child	A specified element that is a direct child of the specified element.	`div>h2` Selects an `<h2>` element that is a child of a `<div>` element.
Adjacent sibling	A specified element that is the next sibling (i.e., comes directly after) of the specified element.	`h3+p` Selects the first `<p>` after any `<h3>`.
General sibling	A specified element that shares the same parent as the specified element.	`h3~p` Selects all the `<p>` elements that share the same parent as the `<h3>`.



```
ul a:link { color: blue; }
```

```
#main time { color: red; }
```

```
#main>time { color: purple; }
```

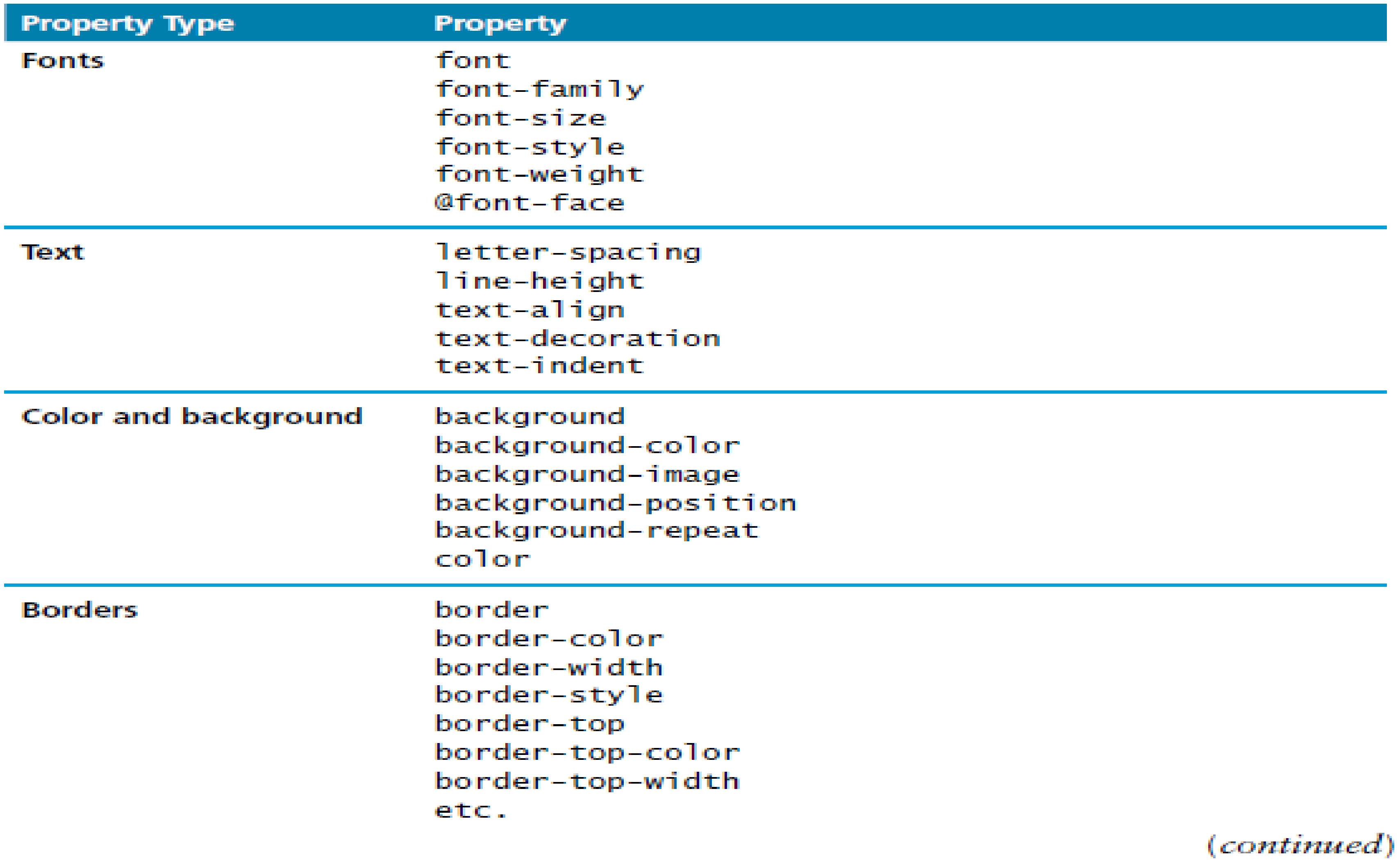
```
#main div p:first-child {
   color: green;
}
```

```
<body>
   <nav>
      <ul>
         <li><a href="#">Canada</a></li>
         <li><a href="#">Germany</a></li>
         <li><a href="#">United States</a></li>
      </ul>
   </nav>
   <div id="main">
      Comments as of <time>November 15, 2015</time>
      <div>
         <p>By Ricardo on <time>September 15, 2015</time></p>
         <p>Easy on the HDR buddy.</p>
      </div>
      <hr/>

      <div>
         <p>By Susan on <time>October 1, 2015</time></p>
         <p>I love Central Park.</p>
      </div>
      <hr/>
   </div>
   <footer>
      <ul>
         <li><a href="#">Home</a> | </li>
         <li><a href="#">Browse</a> | </li>
      </ul>
   </footer>
</body>
```

CSS Properties

- Each individual CSS declaration must contain a property. These property names are predefined by the CSS standard.
- The CSS2.1 recommendation defines over a hundred different property names, so some type of reference guide, whether in a book, online, or within your web development software, can be helpful.
- This chapter will only be able to cover most of the common CSS properties.

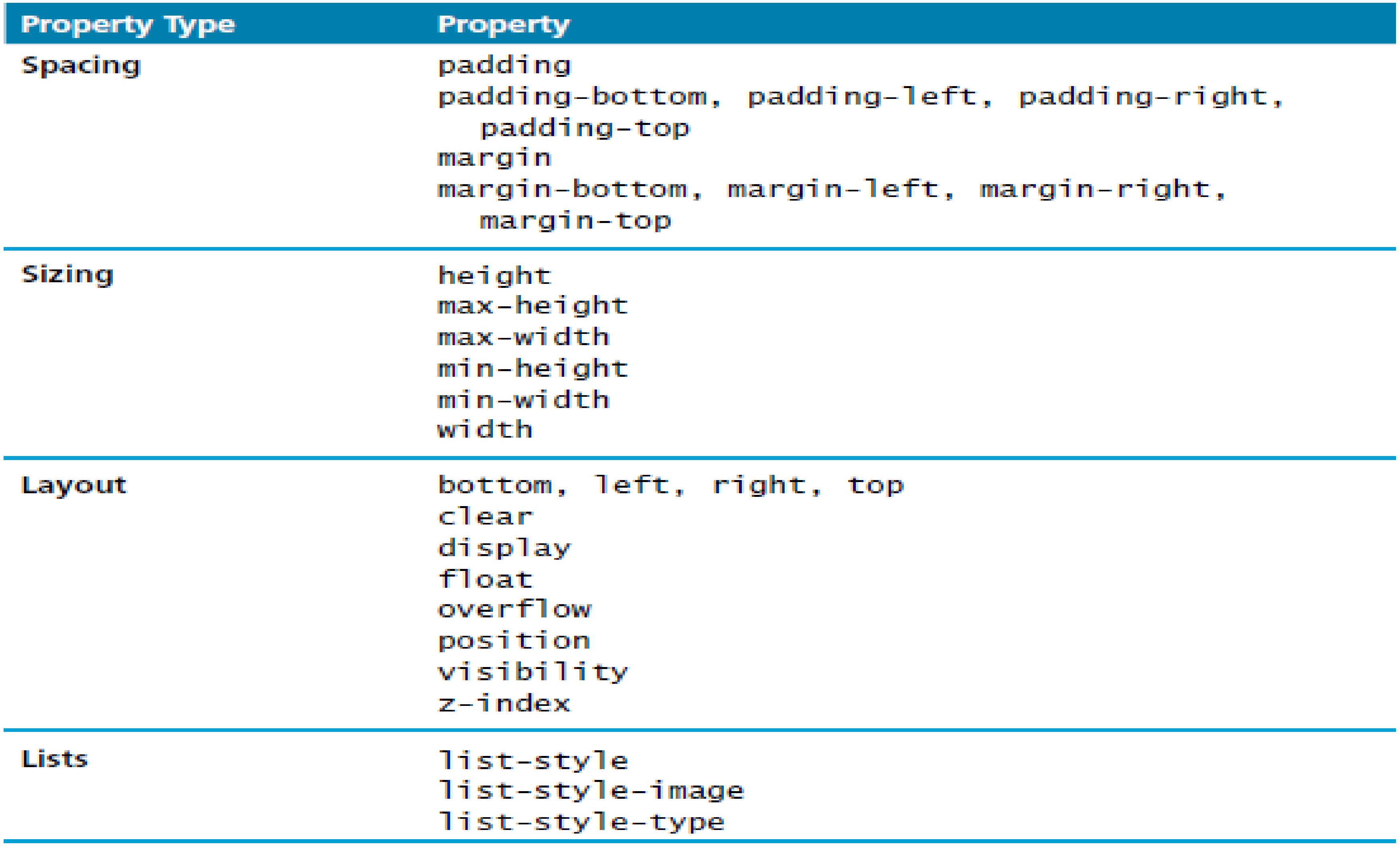
CSS Properties



Property Type	Property
Fonts	font font-family font-size font-style font-weight @font-face
Text	letter-spacing line-height text-align text-decoration text-indent
Color and background	background background-color background-image background-position background-repeat color
Borders	border border-color border-width border-style border-top border-top-color border-top-width etc.

(continued)

CSS Properties



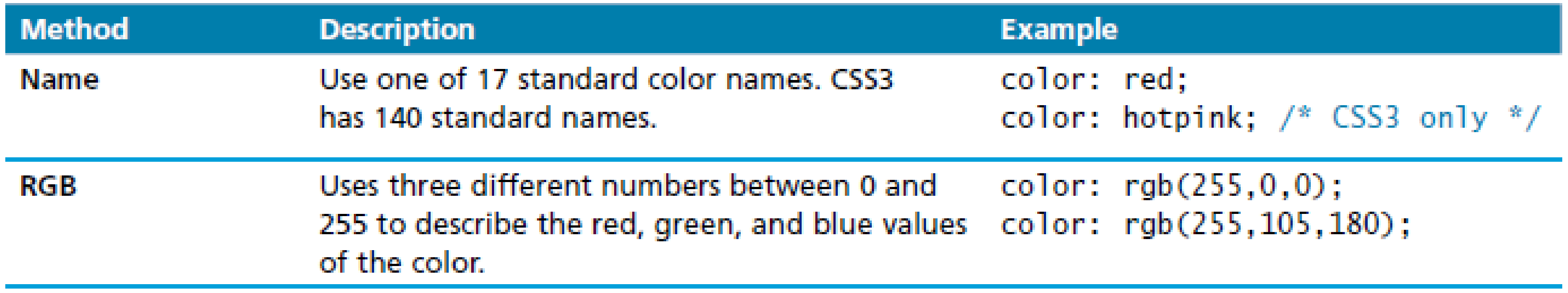
Property Type	Property
Spacing	padding padding-bottom, padding-left, padding-right, padding-top margin margin-bottom, margin-left, margin-right, margin-top
Sizing	height max-height max-width min-height min-width width
Layout	bottom, left, right, top clear display float overflow position visibility z-index
Lists	list-style list-style-image list-style-type

CSS Values

- Each CSS declaration also contains a value for a property.
- The unit of any given value is dependent upon the property.
- Some property values are from a predefined list of keywords.
- Others are values such as length measurements, percentages, numbers without units, color values, and URLs.

CSS Color Values

- Colors would seem at first glance to be the most clear of these units.
- But as we will see in more detail in future, color can be a complicated thing to describe.
- CSS supports a variety of different ways of describing color.



Method	Description	Example

Name	Use one of 17 standard color names. CSS3 has 140 standard names.	`color: red;`
		`color: hotpink; /* CSS3 only */`

RGB	Uses three different numbers between 0 and 255 to describe the red, green, and blue values of the color.	`color: rgb(255,0,0);`
		`color: rgb(255,105,180);`

CSS Color Values

Hexadecimal	Uses a six-digit hexadecimal number to describe the red, green, and blue value of the color; each of the three RGB values is between 0 and FF (which is 255 in decimal). Notice that the hexadecimal number is preceded by a hash or pound symbol (#).	`color: #FF0000;` `color: #FF69B4;`
RGBa	This defines a partially transparent background color. The "a" stands for "alpha", which is a term used to identify a transparency that is a value between 0.0 (fully transparent) and 1.0 (fully opaque).	`color: rgb(255,0,0, 0.5);`
HSL	Allows you to specify a color using Hue Saturation and Light values. This is available only in CSS3. HSLA is also available as well.	`color: hsl(0,100%,100%);` `color: hsl(330,59%,100%);`

CSS Measurement Values

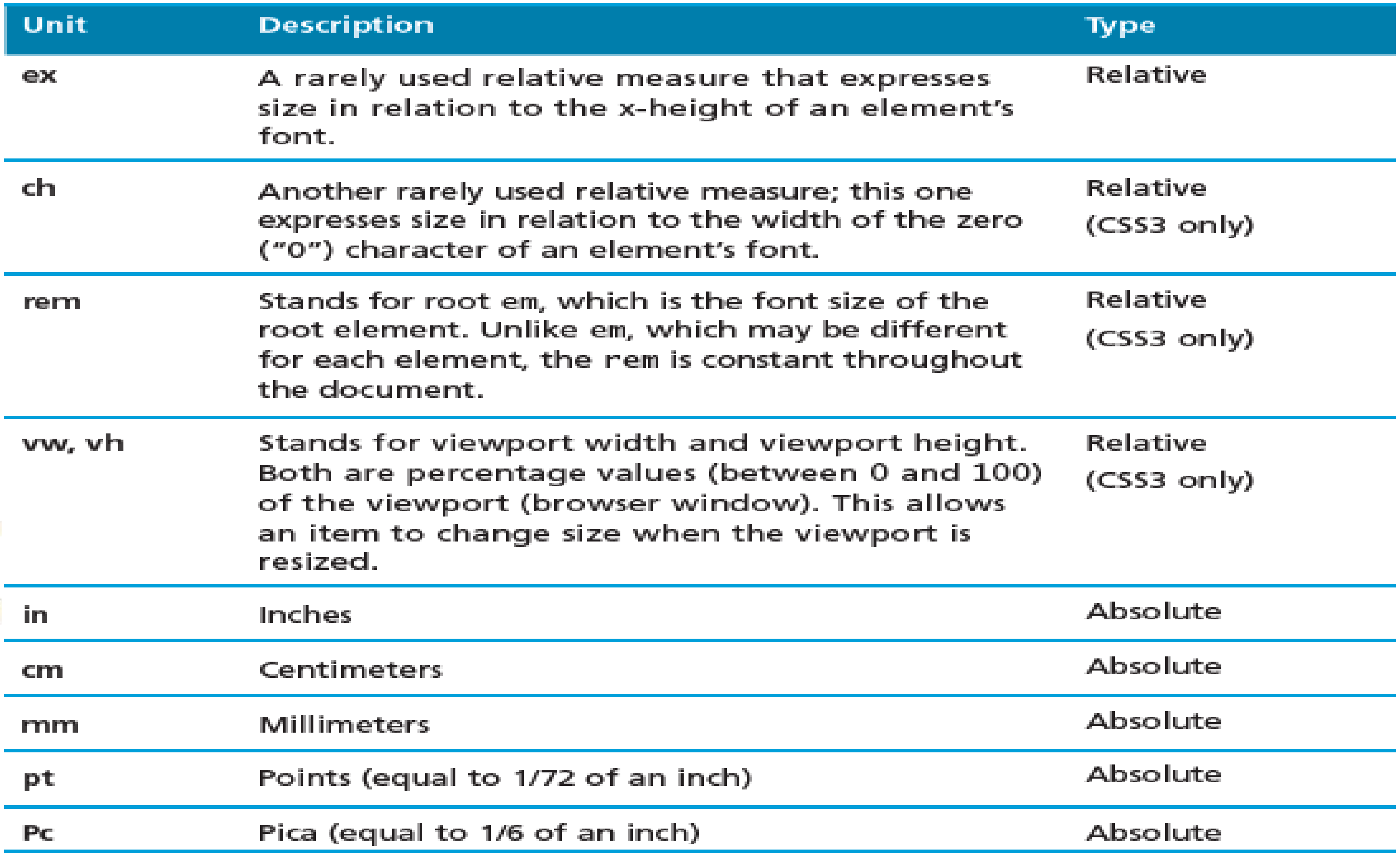
- Just as there are multiple ways of specifying color in CSS, so too there are multiple ways of specifying a unit of measurement.
- When working with print design, we generally make use of straightforward absolute units such as inches or centimeters and picas or points.
- However, because different devices have differing physical sizes as well as different pixel resolutions and because the user is able to change the browser size or its zoom mode, these absolute units don’t always make sense with web element measures.

CSS Measurement Values

Unit	Description	Type
px	Pixel. In CSS2 this is a relative measure, while in CSS3 it is absolute (1/96 of an inch).	Relative (CSS2) Absolute (CSS3)
em	Equal to the computed value of the font-size property of the element on which it is used. When used for font sizes, the em unit is in relation to the font size of the parent.	Relative
%	A measure that is always relative to another value. The precise meaning of % varies depending upon the property in which it is being used.	Relative

(continued)

CSS Measurement Values



Unit	Description	Type
ex	A rarely used relative measure that expresses size in relation to the x-height of an element's font.	Relative
ch	Another rarely used relative measure; this one expresses size in relation to the width of the zero ("0") character of an element's font.	Relative (CSS3 only)
rem	Stands for root em, which is the font size of the root element. Unlike em, which may be different for each element, the rem is constant throughout the document.	Relative (CSS3 only)
vw, vh	Stands for viewport width and viewport height. Both are percentage values (between 0 and 100) of the viewport (browser window). This allows an item to change size when the viewport is resized.	Relative (CSS3 only)
in	Inches	Absolute
cm	Centimeters	Absolute
mm	Millimeters	Absolute
pt	Points (equal to 1/72 of an inch)	Absolute
Pc	Pica (equal to 1/6 of an inch)	Absolute

CSS Measurement Values

- Some of these are **relative units**, in that they are based on the value of something else, such as the size of a parent element.
- Others are **absolute units**, in that they have a real-world size.
- Unless you are defining a style sheet for printing, it is recommended you avoid using absolute units.
- Pixels are perhaps the one popular exception (though, as we shall see later, there are also good reasons for avoiding the pixel unit).
- In general, most of the CSS that you will see uses either px, em, or % as a measure unit.

Location of Styles

- CSS style rules can be located in three different locations.
- These three are not mutually exclusive, in that you could place your style rules in all three.
- In practice, however, web authors tend to place all of their style definitions in one (or more) external style sheet files.

Inline Styles

- **Inline styles** are style rules placed within an HTML element via the style attribute.
- An inline style only affects the element it is defined within and overrides any other style definitions for properties used in the inline style.
- Notice that a selector is not necessary with inline styles and that semicolons are only required for separating multiple rules.
- Using inline styles is generally discouraged since they increase bandwidth and decrease maintainability (because presentation and content are intermixed and because it can be difficult to make consistent inline style changes across multiple files).
- Inline styles can, however, be handy for quickly testing out a style change.

Inline Styles

```
<h1>Share Your Travels</h1>
<h2>style="font-size: 24pt"Description</h2>
...
<h2>style="font-size: 24pt; font-weight: bold;">Reviews</h2>
```

Embedded / Internal Style Sheet

- **Embedded style sheets** (also called **internal styles**) are style rules placed within the <style> element (inside the <head> element of an HTML document.
- While better than inline styles, using embedded styles is also by and large discouraged. Since each HTML document has its own <style> element, it is more difficult to consistently style multiple documents when using embedded styles.
- Just as with inline styles, embedded styles can, however, be helpful when quickly testing out a style that is used in multiple places within a single HTML document.
- We sometimes use embedded styles in the book or in lab materials for that reason.

Embedded / Internal Style Sheet

```
<head lang="en">
   <meta charset="utf-8">
   <title>Share Your Travels -- New York - Central Park</title>
   <style>
      h1 { font-size: 24pt; }
      h2 {
       font-size: 18pt;
       font-weight: bold;
      }
   </style>
</head>
<body>
   <h1>Share Your Travels</h1>
   <h2>New York - Central Park</h2>

   ...
```

External Style Sheet

- **External style sheets** are style rules placed within a external text file with the **.css** extension.
- This is by far the most common place to locate style rules because it provides the best maintainability.
- When you make a change to an external style sheet, all HTML documents that reference that style sheet will automatically use the updated version.
- The browser is able to cache the external style sheet, which can improve the performance of the site as well.
- To reference an external style sheet, you must use a <link> element (within the <head> element).
- You can link to several style sheets at a time; each linked style sheet will require its own <link> element.

External Style Sheet

```
<head lang="en">
   <meta charset="utf-8">
   <title>Share Your Travels -- New York - Central Park</title>
   <link rel="stylesheet" href="styles.css" />
</head>
```

The Cascade: How Styles Interact

- In an earlier Pro Tip in this chapter, it was mentioned that in fact there are three different types of style sheets: author-created, user-defined, and the default browser style sheet.
- As well, it is possible within an author-created stylesheet to define multiple rules for the same HTML element.
- For these reasons, CSS has a system to help the browser determine how to display elements when different style rules conflict.

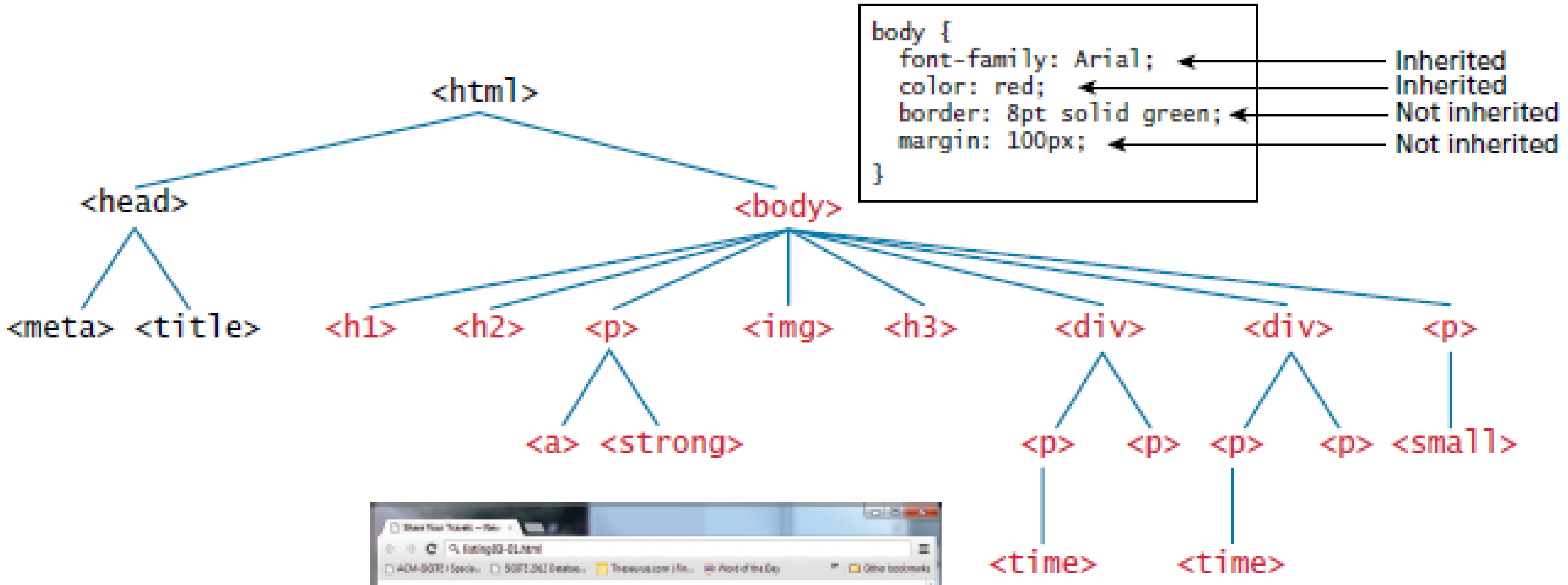
The Cascade: How Styles Interact

- The “Cascade” in CSS refers to how conflicting rules are handled.
- The visual metaphor behind the term **cascade** is that of a mountain stream progressing downstream over rocks (and not that of a popular dishwashing detergent).
- The downward movement of water down a cascade is meant to be analogous to how a given style rule will continue to take precedence with child elements (i.e., elements “below” in a document outline in page 16)
- CSS uses the following cascade principles to help it deal with conflicts: ***inheritance, specificity, and location.***

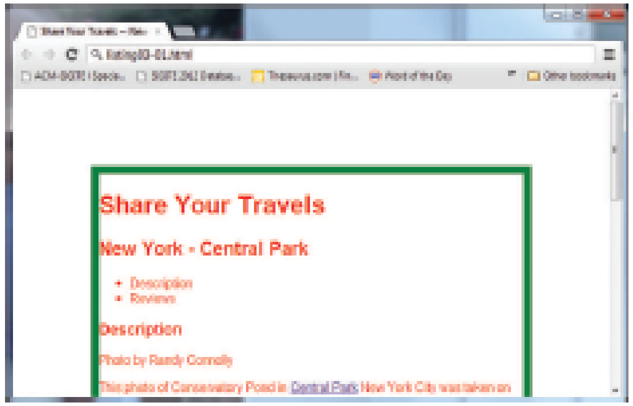
Inheritance

- **Inheritance** is the first of these cascading principles.
- Many (but not all) CSS properties affect not only themselves but their descendants as well.
- Font, color, list, and text properties are inheritable; layout, sizing, border, background, and spacing properties are not.

Example 1



```
body {
  font-family: Arial;
  color: red;
  border: 8pt solid green;
  margin: 100px;
}
```



Share Your Travels

New York - Central Park

- Description
- Reviews

Description

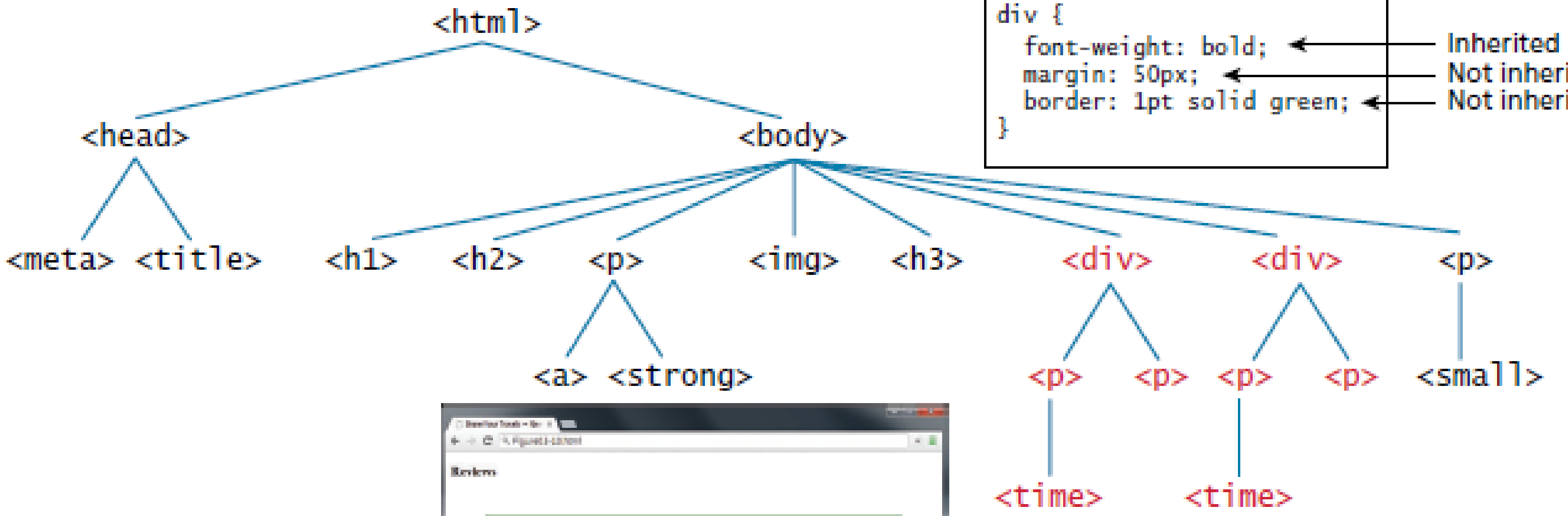
Photo by Randy Connolly

This photo of Conservatory Pond in Central Park New York City was taken on

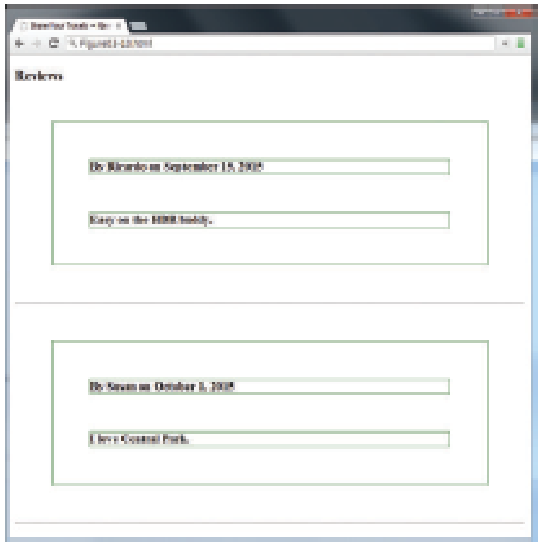
Inheritance

- In the first example, only some of the property rules are inherited for the <body> element.
- That is, only the body element (thankfully!) will have a thick green border and the 100-px margin; however, all the text in the other elements in the document will be in the Arial font and colored red.

Example 2



```
div {
   font-weight: bold;          ← Inherited
   margin: 50px;               ← Not inherited
   border: 1pt solid green;    ← Not inherited
}
```



Reviews

By Ricardo on September 15, 2015

Easy on the HDR buddy.

By Susan on October 1, 2015

I love Central Park.

Inheritance

- In the second example, you can assume there is no longer the body styling but instead we have a single style rule that styles *all* the <div> elements.
- The <p> and <time> elements within the <div> inherit the bold font-weight property but not the margin or border styles.
- However, it is possible to tell elements to inherit properties that are normally not inheritable, as shown in example 3.
- In comparison to example 2, notice how the <p> elements nested within the <div> elements now inherit the border and margins of their parent.

Example 3: Using the *inherit* value

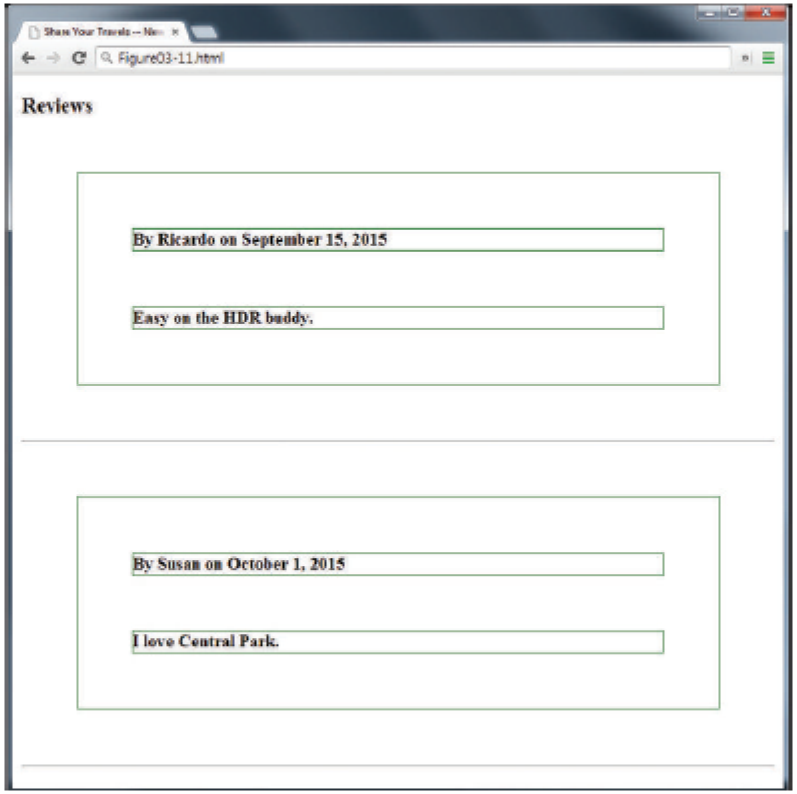

Share Your Travels -- New

Figure03-11.html

Reviews

By Ricardo on September 15, 2015

Easy on the HDR buddy.

By Susan on October 1, 2015

I love Central Park.

```
div {
  font-weight: bold;
  margin: 50px;
  border: 1pt solid green;
}
p {
  border: inherit;
  margin: inherit;
}
```

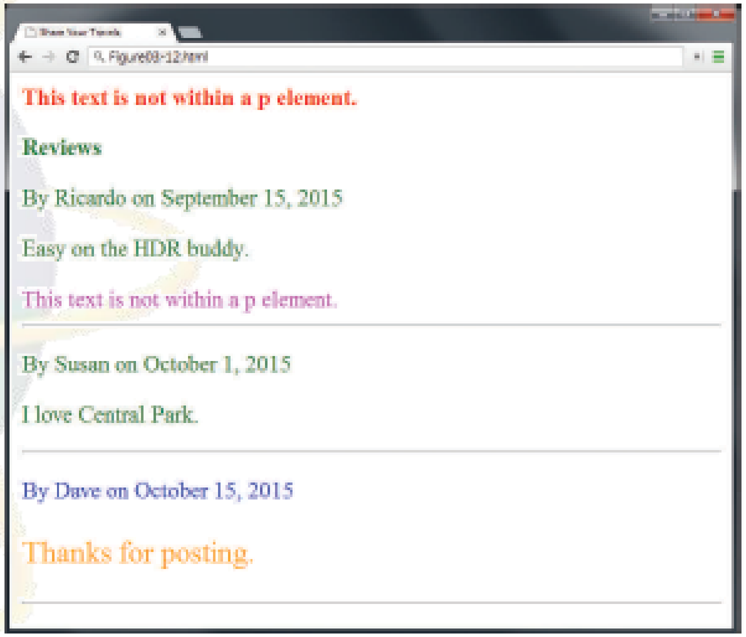
```
<h3>Reviews</h3>
<div>
   <p>By Ricardo on <time>September 15, 2015</time></p>
   <p>Easy on the HDR buddy.</p>
</div>
<hr/>

<div>
   <p>By Susan on <time>October 1, 2015</time></p>
   <p>I love Central Park.</p>
</div>
<hr/>
```

Specificity

- **Specificity** is how the browser determines which style rule takes precedence when more than one style rule could be applied to the same element.
- In CSS, the more specific the selector, the more it takes precedence (i.e., overrides the previous definition).
- Another way to define specificity is by telling you how it works.
- The way that specificity works in the browser is that the browser assigns a weight to each style rule; when several rules apply, the one with the greatest weight takes precedence.

```
body {
  font-weight: bold;
  color: red;
}

div {
  font-weight: normal;
  color: magenta;
}

p {
  color: green;
}

.last {
  color: blue;
}

#verylast {
  color: orange;
  font-size: 16pt;
}
```

```
<body>
   This text is not within a p element.
   <p>Reviews</p>
   <div>
      <p>By Ricardo on <time>September 15, 2015</time></p>
      <p>Easy on the HDR buddy.</p>
      This text is not within a p element.
   </div>
   <hr/>

   <div>
      <p>By Susan on <time>October 1, 2015</time></p>
      <p>I love Central Park.</p>
   </div>
   <hr/>

   <div>
      <p class="last">By Dave on <time>October 15, 2015</time></p>
      <p class="last" id="verylast">Thanks for posting.</p>
   </div>
   <hr/>
</body>
```



This text is not within a p element.

Reviews

By Ricardo on September 15, 2015

Easy on the HDR buddy.

This text is not within a p element.

By Susan on October 1, 2015

I love Central Park.

By Dave on October 15, 2015

Thanks for posting.

Example 4: *Specificity*

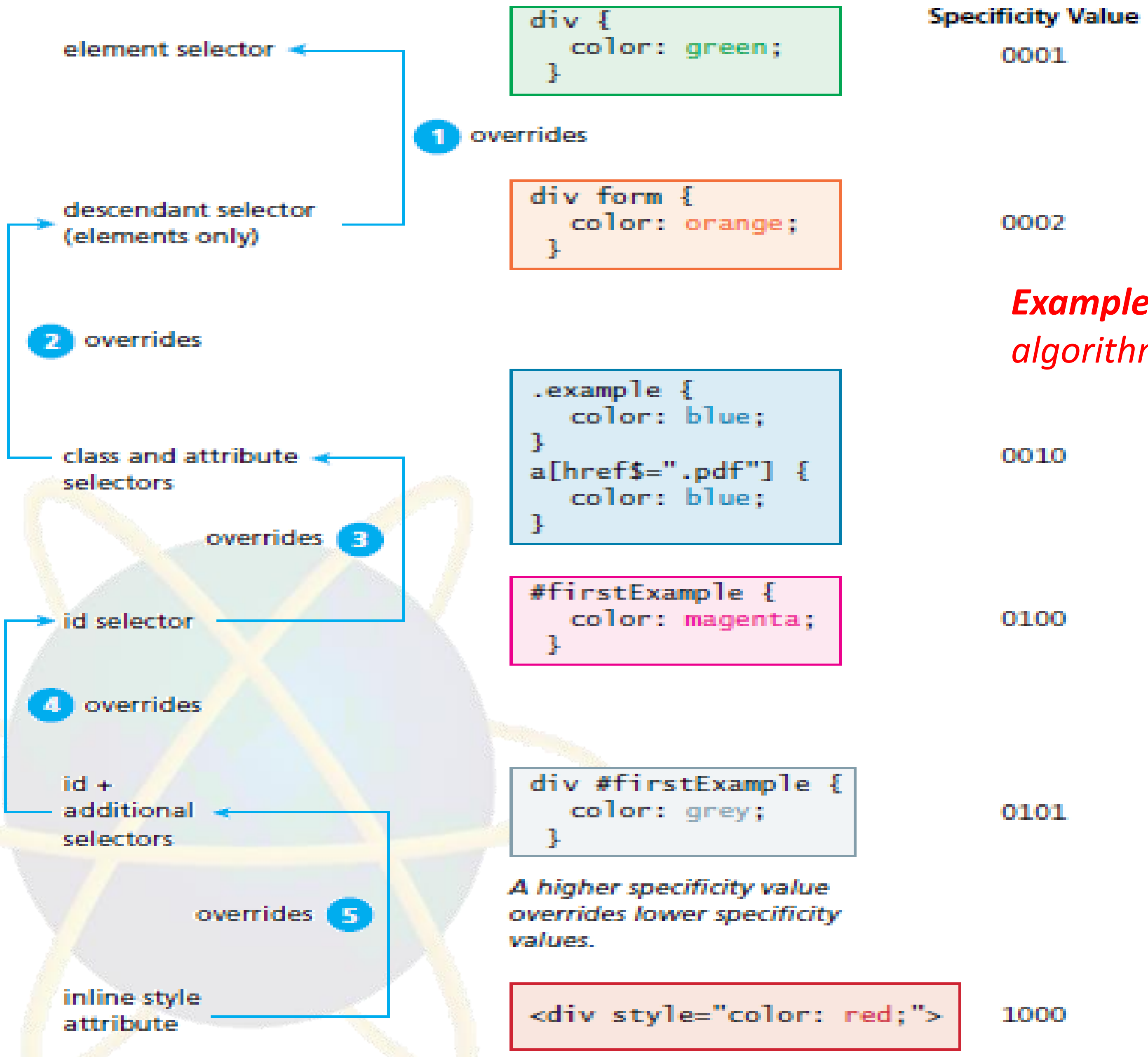
Specificity

- In the example 4, the color and font-weight properties defined in the <body> element are inheritable and thus potentially applicable to all the child elements contained within it.
- However, because the <div> and <p> elements also have the same properties set, they *override* the value defined for the <body> element because their selectors (<div> and <p>) are more specific.
- As a consequence, their font-weight is normal and their text is colored either green or magenta.

Specificity

- As you can see in Example 4, class selectors take precedence over element selectors, and id selectors take precedence over class selectors.
- The precise algorithm the browser is supposed to use to determine specificity is quite complex.
- A simplified version is shown in Example 5 next page.

Example 5: *Specificity algorithm*



```
div {
   color: green;
 }
```

```
div form {
   color: orange;
 }
```

```
.example {
   color: blue;
}
a[href$=".pdf"] {
   color: blue;
}
```

```
#firstExample {
   color: magenta;
 }
```

```
div #firstExample {
   color: grey;
 }
```

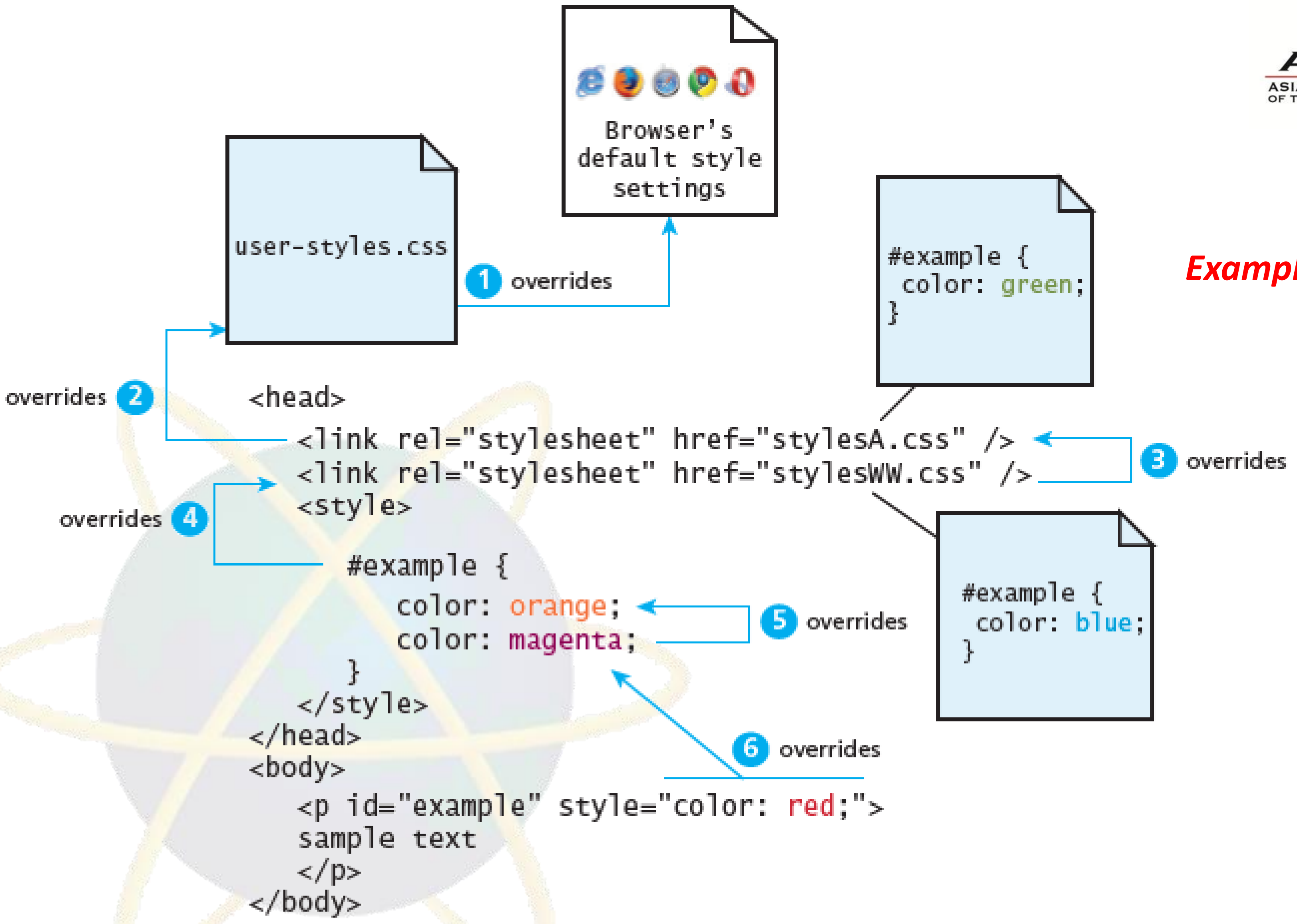
A higher specificity value overrides lower specificity values.

```
<div style="color: red;">
```

Location

- Finally, when inheritance and specificity cannot determine style precedence, the principle of **location** will be used.
- The principle of location is that when rules have the same specificity, then the latest are given more weight.
- For instance, an inline style will override one defined in an external author style sheet or an embedded style sheet.
- Similarly, an embedded style will override an equally specific rule defined in an external author style sheet if it appears after the external sheet's <link> element.
- Styles defined in external author style sheet X will override styles in external author style sheet Y if X's <link> element is after Y's in the HTML document.

Example 6: Location



```
#example {
 color: green;
}
```

```
#example {
 color: blue;
}
```

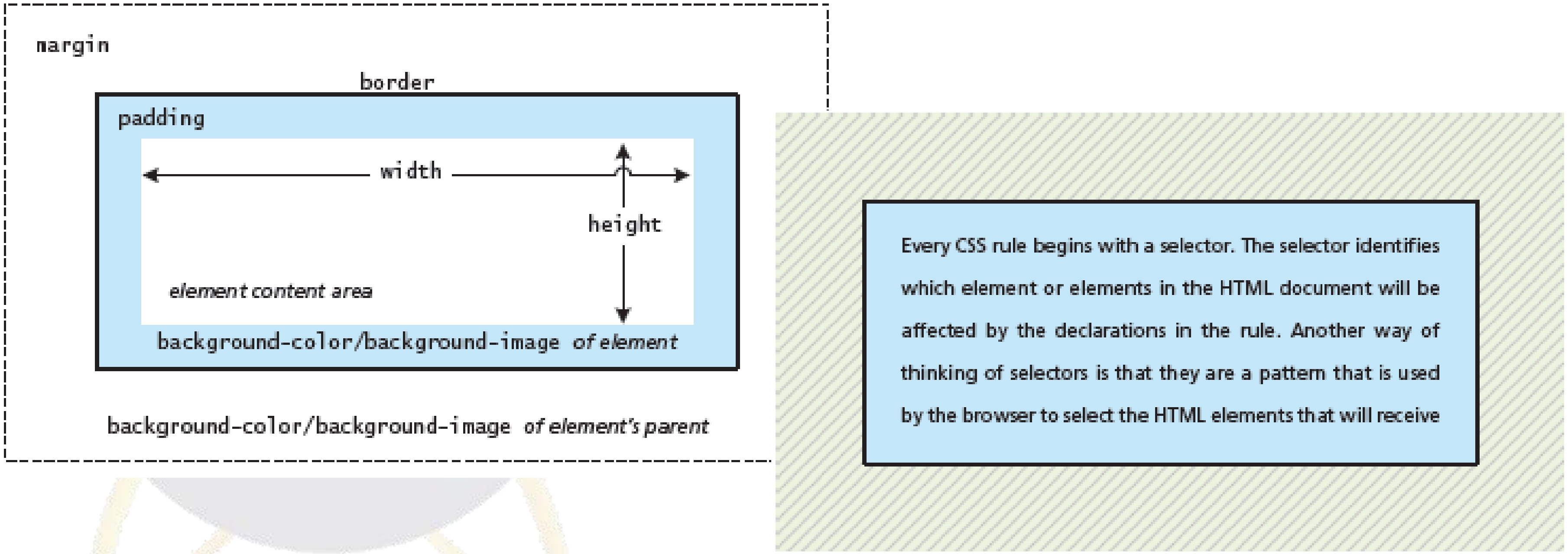
```
<head>
   <link rel="stylesheet" href="stylesA.css" />
   <link rel="stylesheet" href="stylesWW.css" />
   <style>
      #example {
         color: orange;
         color: magenta;
      }
   </style>
</head>
<body>
   <p id="example" style="color: red;">
   sample text
   </p>
</body>
```

Location

- When the same style property is defined multiple times within a single declaration block, the last one will take precedence.
- Example 6 illustrates how location affects precedence.
- Can you guess what will be the color of the sample text in example 6?
- The answer to the question is: The color of the sample text in Example 6 will be red.
- What would be the color of the sample text if there wasn’t an inline style definition?
- It would be magenta.

The Box Model

- In CSS, all HTML elements exist within an **element box.**
- In order to become proficient with CSS, you must become familiar with the element box.


margin
border
padding
width
height
element content area
background-color/background-image of element
background-color/background-image of element's parent

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern that is used by the browser to select the HTML elements that will receive

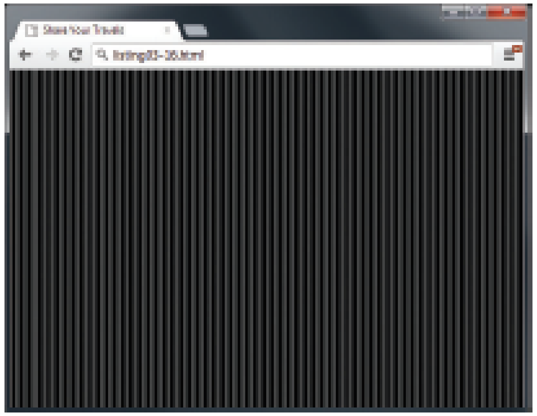
Background

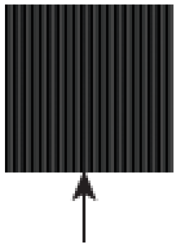
- The background color or image of an element fills an element out to its border (if it has one, that is).
- In contemporary web design, it has become extremely common to use CSS to display purely presentational images (such as background gradients and patterns, decorative images, etc.) rather than using the <img> element.

Common Background Properties

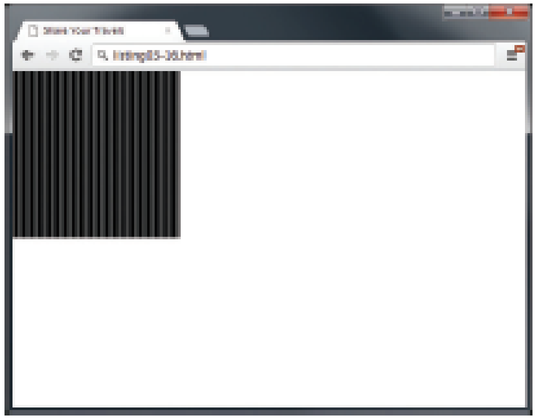
Property	Description
`background`	A combined shorthand property that allows you to set multiple background values in one property. While you can omit properties with the shorthand, do remember that any omitted properties will be set to their default value.
`background-attachment`	Specifies whether the background image scrolls with the document (default) or remains fixed. Possible values are: `fixed`, `scroll`.
`background-color`	Sets the background color of the element. You can use any of the techniques shown in Table 3.2 for specifying the color.
`background-image`	Specifies the background image (which is generally a jpeg, gif, or png file) for the element. Note that the URL is relative to the CSS file and not the HTML. CSS3 introduced the ability to specify multiple background images.
`background-position`	Specifies where on the element the background image will be placed. Some possible values include: `bottom`, `center`, `left`, and `right`. You can also supply a pixel or percentage numeric position value as well. When supplying a numeric value, you must supply a horizontal/vertical pair; this value indicates its distance from the top left corner of the element, as shown in Figure 3.16.
`background-repeat`	Determines whether the background image will be repeated. This is a common technique for creating a tiled background (it is in fact the default behavior), as shown in Figure 3.17. Possible values are: `repeat`, `repeat-x`, `repeat-y`, and `no-repeat`.
`background-size`	New to CSS3, this property lets you modify the size of the background image.

Background repeat

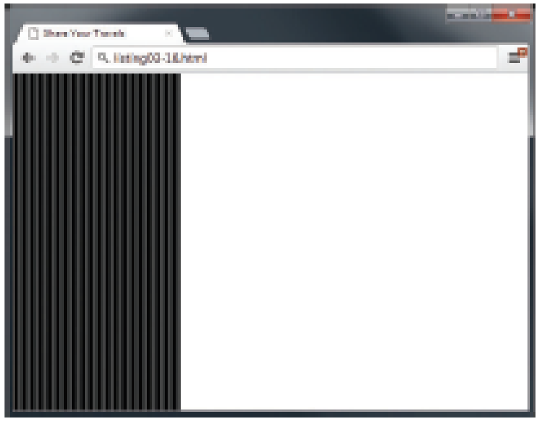




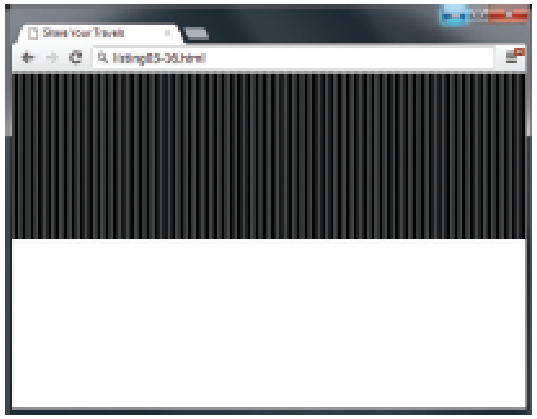
```
background-image: url(../images/backgrounds/body-background-tile.gif);
background-repeat: repeat;
```



```
background-repeat: no-repeat;
```

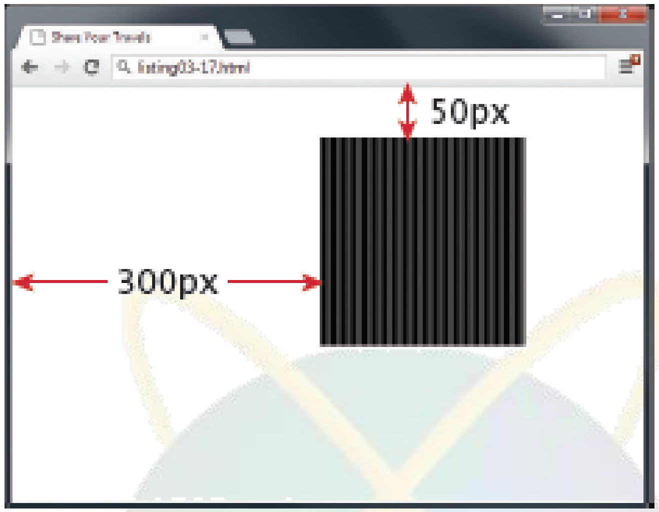


```
background-repeat: repeat-y;
```



```
background-repeat: repeat-x;
```

Background position

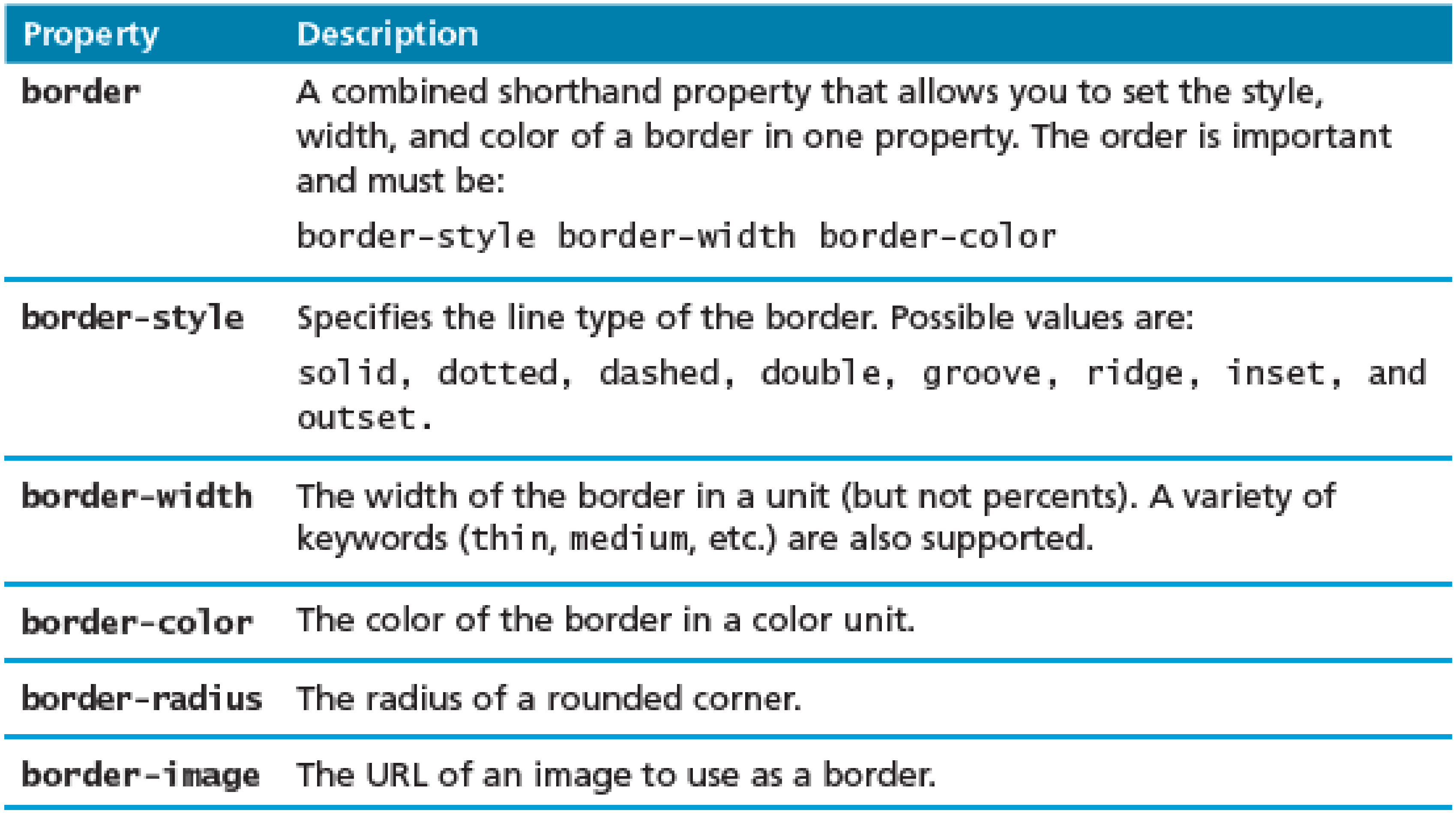


```
body {
        background: white url(../images/backgrounds/body-background-tile.gif) no-repeat;
        background-position: 300px 50px;
}
```

Borders

- Borders provide a way to visually separate elements.
- You can put borders around all four sides of an element, or just one, two, or three of the sides.
- Border widths are perhaps the one exception to the general advice against using the pixel measure.
- Using em units or percentages for border widths can result in unpredictable widths as the different browsers use different algorithms (some round up, some round down) as the zoom level increases or decreases.
- For this reason, border widths are almost always set to pixel units.

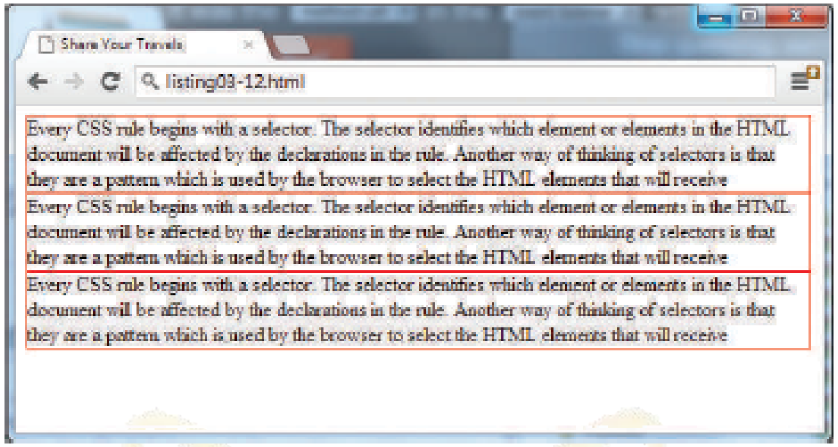
Border Properties



Property	Description
border	A combined shorthand property that allows you to set the style, width, and color of a border in one property. The order is important and must be: `border-style border-width border-color`
border-style	Specifies the line type of the border. Possible values are: `solid`, `dotted`, `dashed`, `double`, `groove`, `ridge`, `inset`, and `outset`.
border-width	The width of the border in a unit (but not percents). A variety of keywords (`thin`, `medium`, etc.) are also supported.
border-color	The color of the border in a color unit.
border-radius	The radius of a rounded corner.
border-image	The URL of an image to use as a border.

Margins and Padding

- Margins and padding are essential properties for adding white space to a web page, which can help differentiate one element from another.
- Margins add spacing around an element’s content, while padding adds spacing within elements.
- Borders divide the margin area from the padding area.
- What this means is that when the **vertical** margins of two elements touch, only the largest margin value of the elements will be displayed, while the smaller margin value will be collapsed to zero.
- Horizontal margins, on the other hand, **never** collapse.
- To complicate matters even further, there are a large number of special cases in which adjoining vertical margins do **not** collapse


Share Your Travels

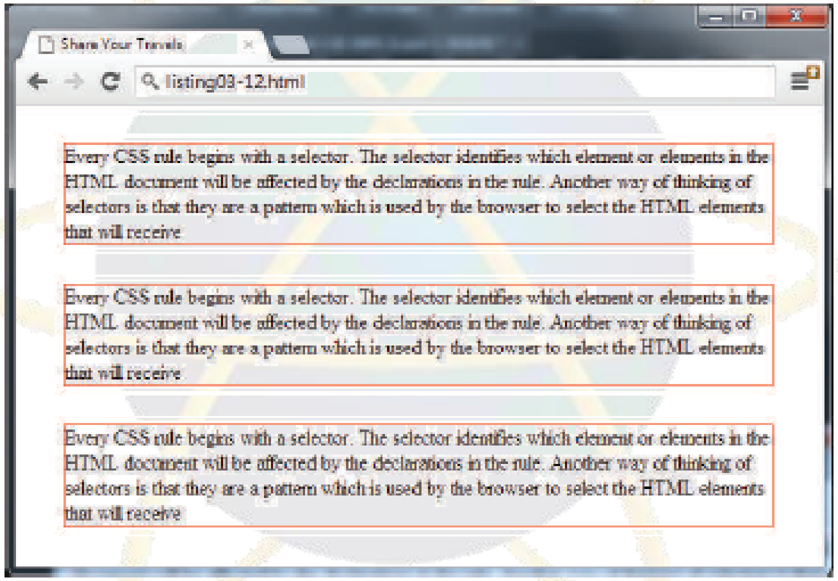
listing03-12.html

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the browser to select the HTML elements that will receive

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the browser to select the HTML elements that will receive

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the browser to select the HTML elements that will receive

```
p {
   border: solid 1pt red;
   margin: 0;
   padding: 0;
}
```


Share Your Travels

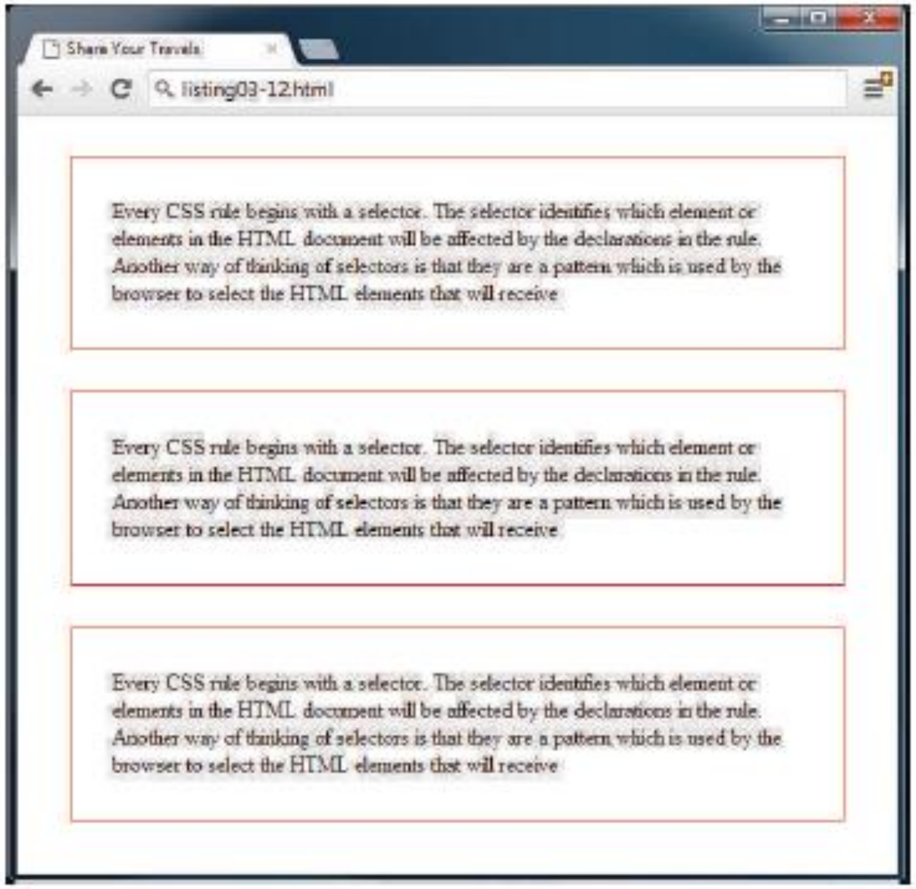
listing03-12.html

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the browser to select the HTML elements that will receive

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the browser to select the HTML elements that will receive

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the browser to select the HTML elements that will receive

```
p {
   border: solid 1pt red;
   margin: 30px;
   padding: 0;
}
```


Share Your Travels

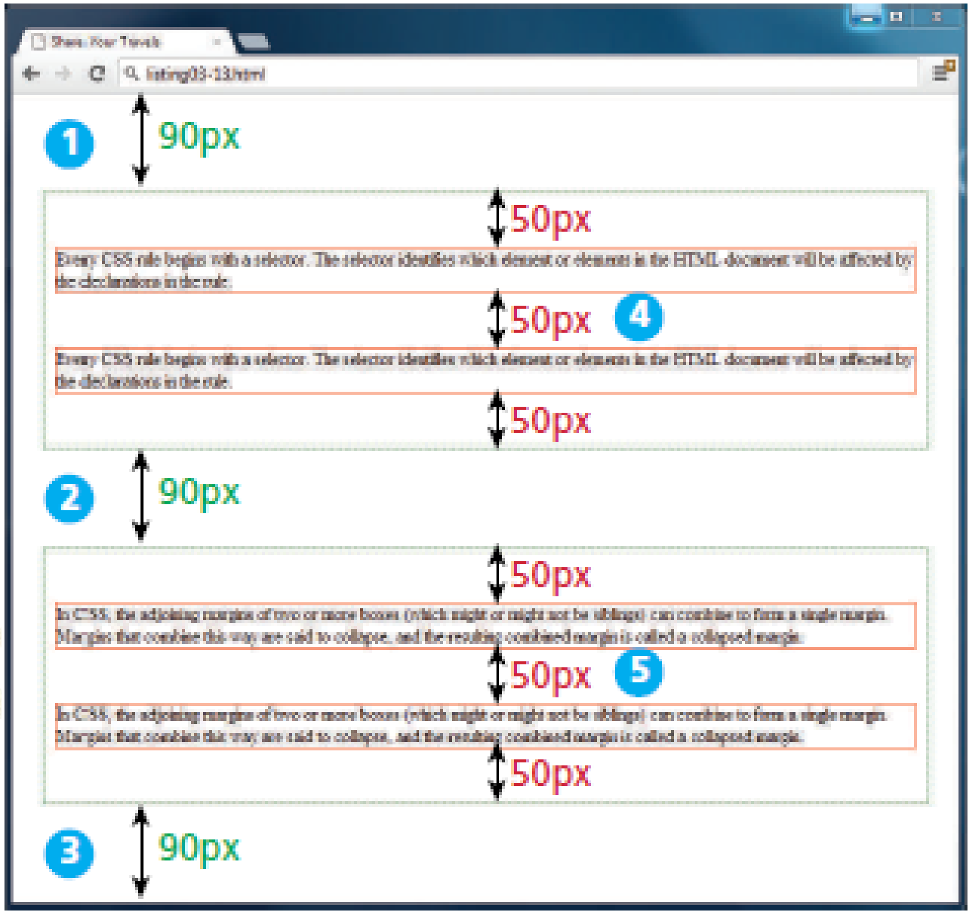
listing03-12.html

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the browser to select the HTML elements that will receive

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the browser to select the HTML elements that will receive

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the browser to select the HTML elements that will receive

```
p {
   border: solid 1pt red;
   margin: 30px;
   padding: 30px;
}
```



Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule.

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule.

In CSS, the adjoining margins of two or more boxes (which might or might not be siblings) can combine to form a single margin. Margins that combine this way are said to collapse, and the resulting combined margin is called a collapsed margin.

In CSS, the adjoining margins of two or more boxes (which might or might not be siblings) can combine to form a single margin. Margins that combine this way are said to collapse, and the resulting combined margin is called a collapsed margin.

```
<div>
  <p>Every CSS rule ...</p>
  <p>Every CSS rule ...</p>
</div>
<div>
  <p>In CSS, the adjoining ... </p>
  <p>In CSS, the adjoining ... </p>
</div>
```

```
div {
   border: dotted 1pt green;
   padding: 0;
  margin: 90px 20px;
}
```

```
p {
   border: solid 1pt red;
   padding: 0;
   margin: 50px 20px;
}
```

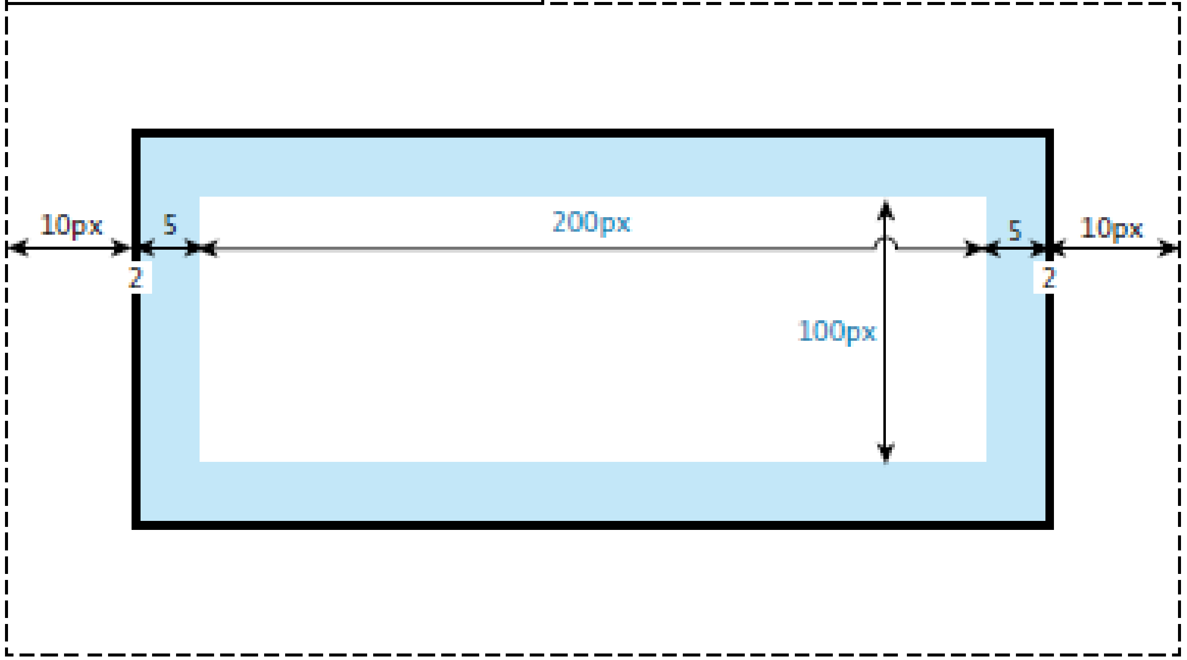
Box Dimensions

- Box dimensions (i.e., the width and height properties) also frequently trouble new CSS authors.
- One reason is that only block-level elements and nontext inline elements such as images have a width and height that you can specify.
- By default (in CSS this is the auto value), the width of and height of elements is the actual size of the content.
- For text, this is determined by the font size and font face; for images, the width and height of the actual image in pixels.
- Since the width and the height only refer to the size of the content area, the total size of an element is equal to the size of its content area plus the sum of its padding, borders, and margins.

Calculating an element's true size

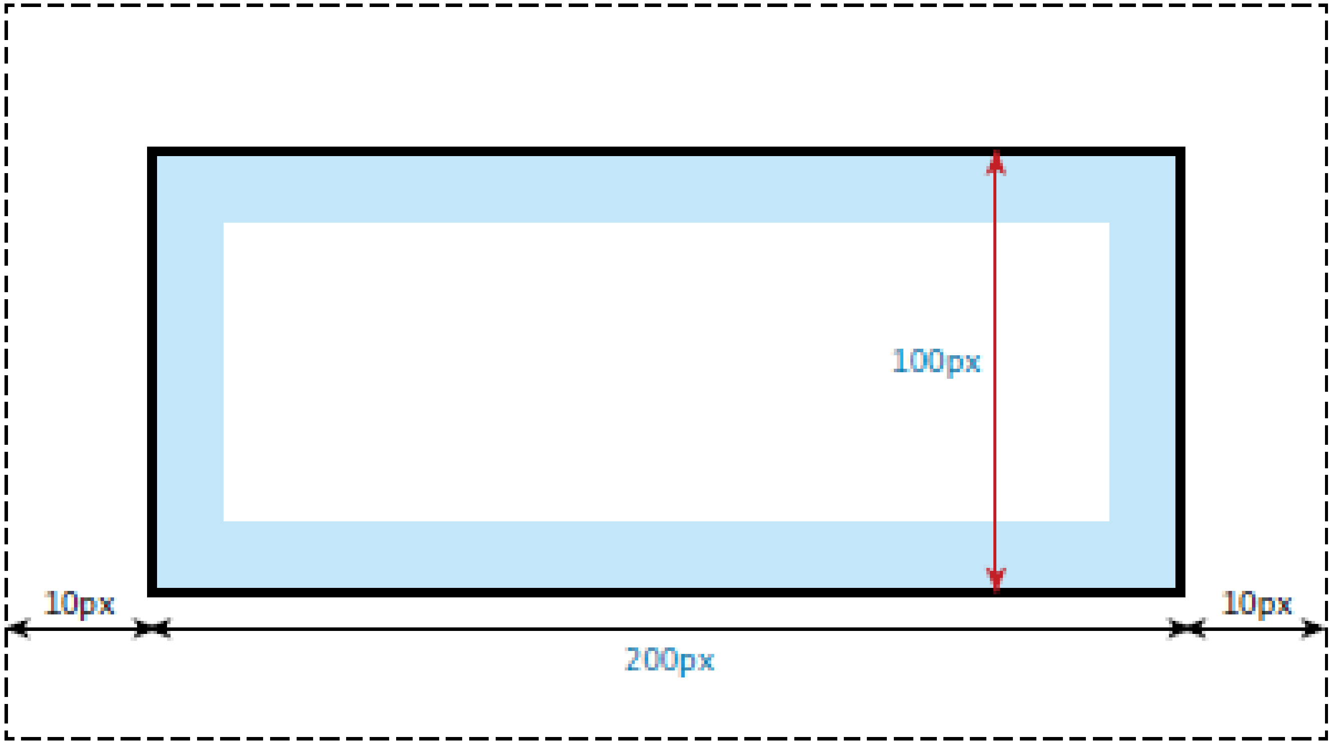
```
div {
  box-sizing: content-box;
  width: 200px;
  height: 100px;
  padding: 5px;
  margin: 10px;
  border: solid 2pt black;
}
```

True element width = 10 + 2 + 5 + 200 + 5 + 2 + 10 = 234 px
True element height = 10 + 2 + 5 + 100 + 5 + 2 + 10 = 134 px

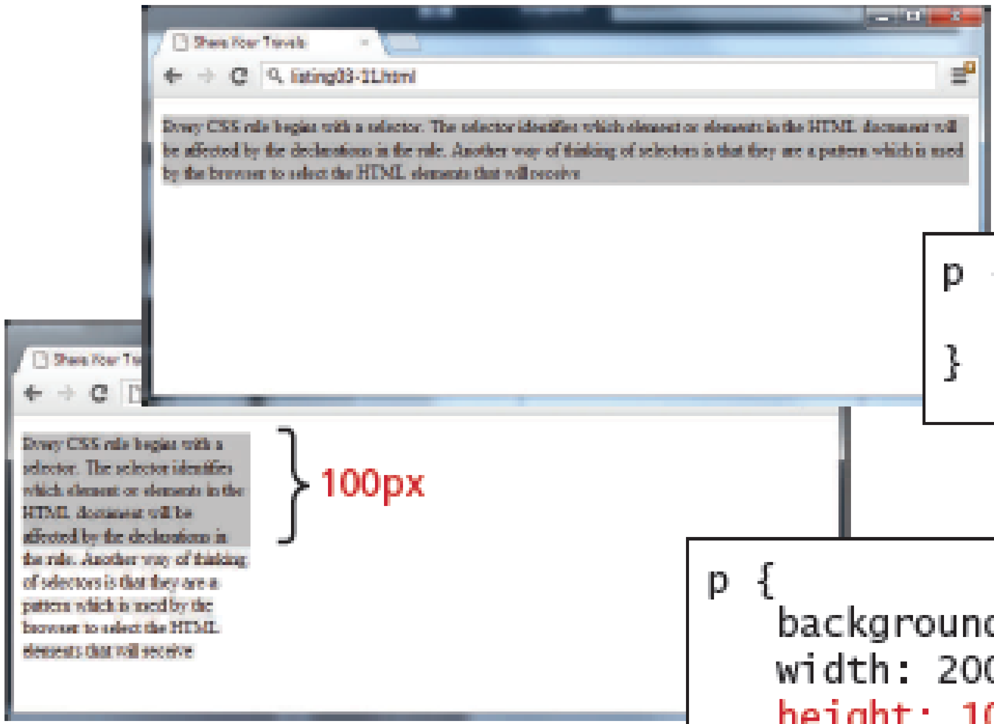


```
div {
  ...
  box-sizing: border-box;
}
```

True element width = 10 + 200 + 10 = 220 px
True element height = 10 + 100 + 10 = 120 px



Limitations of height property



Share Your Travels

listing03-11.html

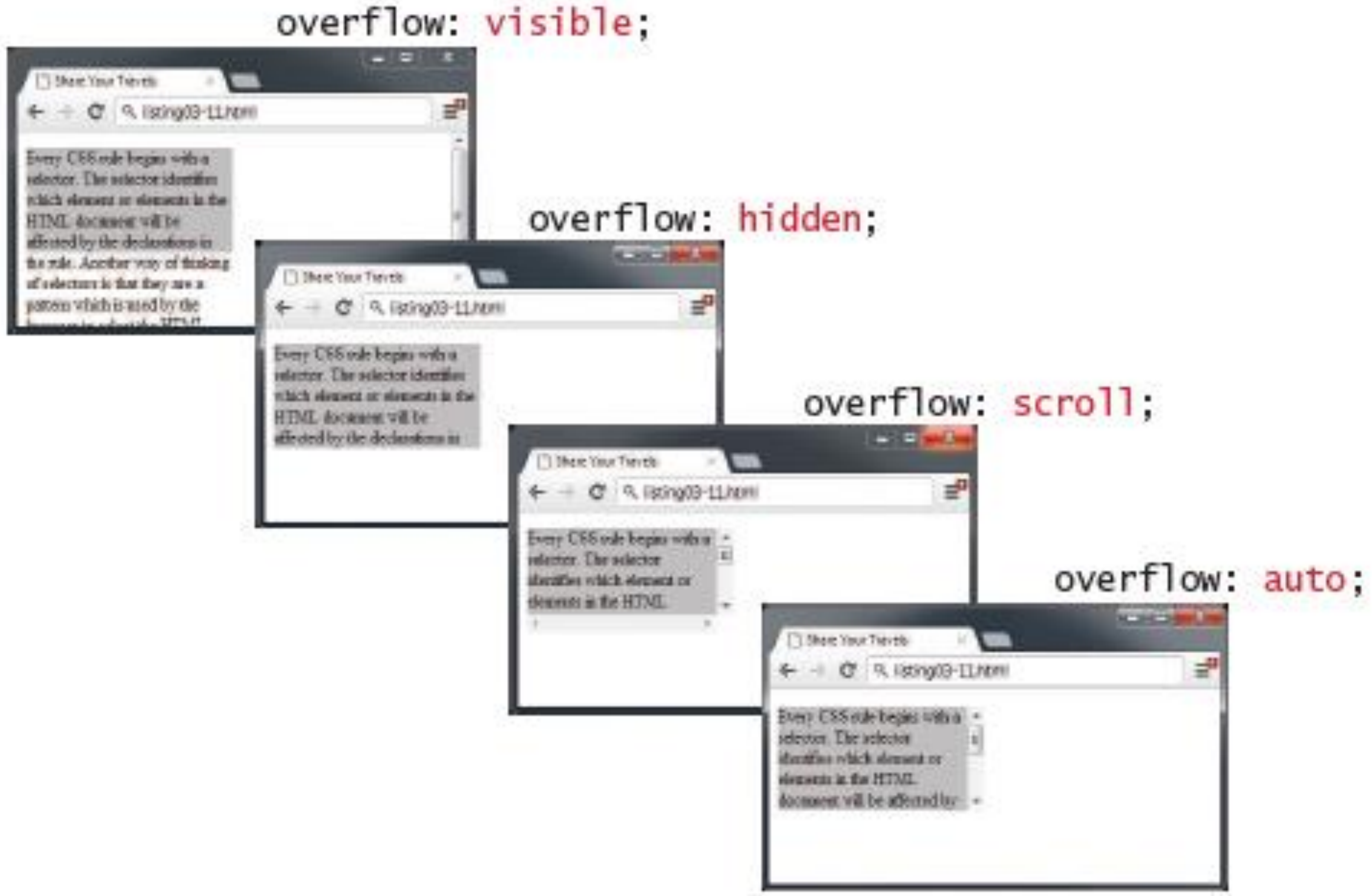
Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the browser to select the HTML elements that will receive

```
p {
    background-color: silver;
}
```

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the browser to select the HTML elements that will receive

```
p {
    background-color: silver;
    width: 200px;
    height: 100px;
}
```

Overflow property



overflow: visible;

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern which is used by the

overflow: hidden;

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in

overflow: scroll;

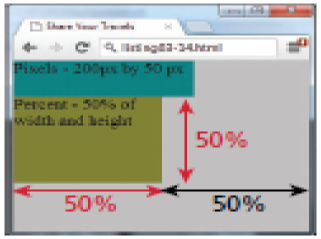
Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML

overflow: auto;

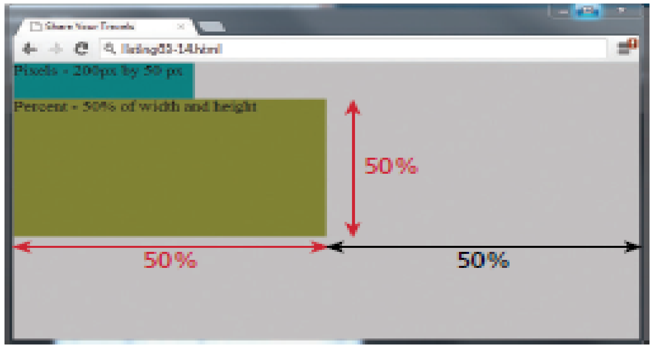
Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by

Box sizing via percents

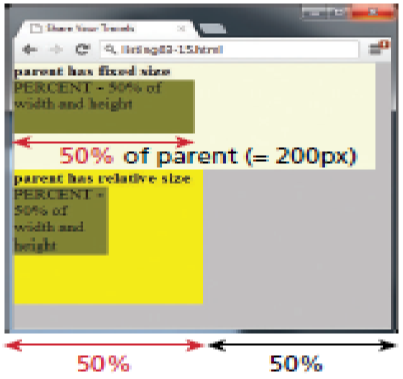
```
<style>
  html,body {
     margin:0;
     width:100%;
     height:100%;
     background: silver;
  }
  .pixels {
     width:200px;
     height:50px;
     background: teal;
  }
  .percent {
     width:50%;
     height:50%;
     background: olive;
  }
```



```
<body>
   <div class="pixels">
     Pixels - 200px by 50 px
   </div>
   <div class="percent">
     Percent - 50% of width and height
   </div>
</body>
```



```
  .parentFixed {
     width:400px;
     height:150px;
     background: beige;
  }
  .parentRelative {
     width:50%;
     height:50%;
     background: yellow;
  }
</style>
```



```
<body>
<div class="parentFixed">
   <strong>parent has fixed size</strong>
   <div class="percent">
      PERCENT - 50% of width and height
   </div>
</div>
<div class="parentRelative">
   <strong>parent has relative size</strong>
   <div class="percent">
      PERCENT - 50% of width and height
   </div>
</div>
</body>
```

CSS Text Styling

- CSS provides two types of properties that affect text.
- The first we call font properties because they affect the font and its appearance.
- The second type of CSS text properties are referred to here as paragraph properties since they affect the text in a similar way no matter which font is being used.

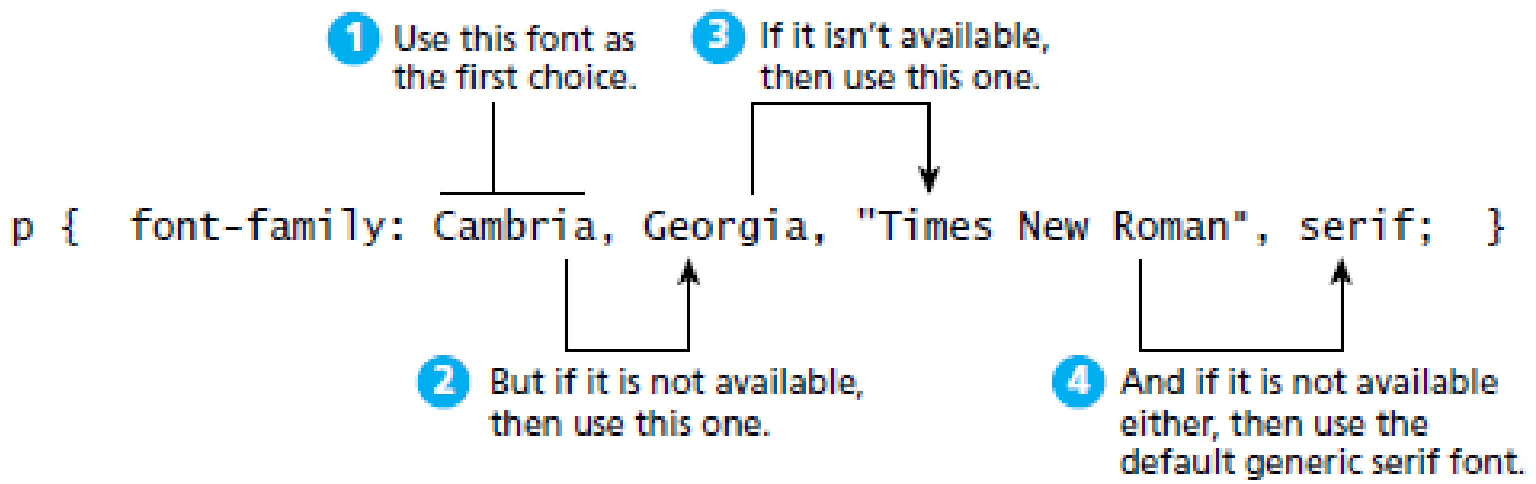
CSS Font Family

- The first of these problems involves specifying the font family.
- A word processor on a desktop machine can make use of any font that is installed on the computer; browsers are no different.
- However, just because a given font is available on the web developer's computer, it does not mean that that same font will be available for all users who view the site.
- For this reason, it is conventional to supply a so-called **web font stack**, that is, a series of alternate fonts to use in case the original font choice is not on the user's computer.

Font Properties

Property	Description
`font`	A combined shorthand property that allows you to set the family, style, size, variant, and weight in one property. While you do not have to specify each property, you must include at a minimum the font size and font family. In addtion, the order is important and must be: `style weight variant size font-family`
`font-family`	Specifies the typeface/font (or generic font family) to use. More than one can be specified.
`font-size`	The size of the font in one of the measurement units.
`font-style`	Specifies whether `italic`, `oblique` (i.e., skewed by the browser rather than a true italic), or `normal`.
`font-variant`	Specifies either `small-caps` text or none (i.e., regular text).
`font-weight`	Specifies either `normal`, `bold`, `bolder`, `lighter`, or a value between 100 and 900 in multiples of 100, where larger number represents weightier (i.e., bolder) text.

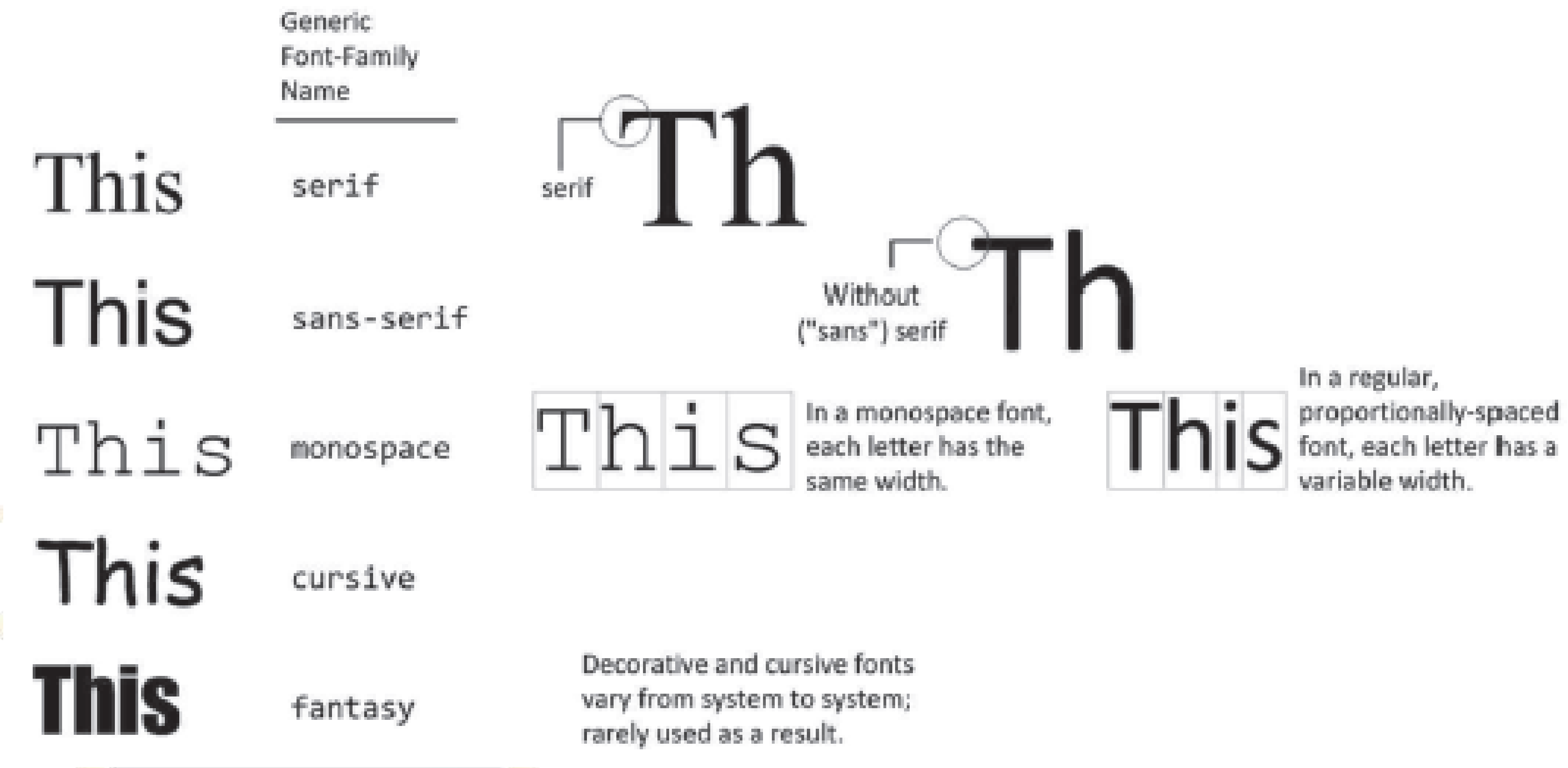
Specifying the font family



```
p {  font-family: Cambria, Georgia, "Times New Roman", serif;  }
```

1 Use this font as the first choice.
3 If it isn't available, then use this one.
2 But if it is not available, then use this one.
4 And if it is not available either, then use the default generic serif font.

The different font families



	Generic Font-Family Name
This	serif
This	sans-serif
This	monospace
This	cursive
This	fantasy

In a monospace font, each letter has the same width.

In a regular, proportionally-spaced font, each letter has a variable width.

Decorative and cursive fonts vary from system to system; rarely used as a result.

CSS Font Family

- One common approach is to make your font stack contain, in this order, the following: *ideal*, *alternative*, *common*, and then *generic*.
- Take for instance, the following font stack:

font-family { "Hoefler Text", Cambria, "Times New Roman", serif; }

- You might love the appearance of Hoefler Text, which is installed on most Macs, so it is your *ideal* choice for your site; however, it is not installed on very many PCs or Android devices.

CSS Font Family

- Cambria is on most PC and Mac computers and is your *alternative* choice.
- Times New Roman is installed on almost all PCs and Macs so it is a safe *common* choice; but because you would prefer Cambria to be used instead of Times New Roman, you placed Cambria first.
- Finally, Android or Blackberry users might not have any of these fonts, so you finished the font stack with the *generic* serif since all your other choices are all serif fonts.

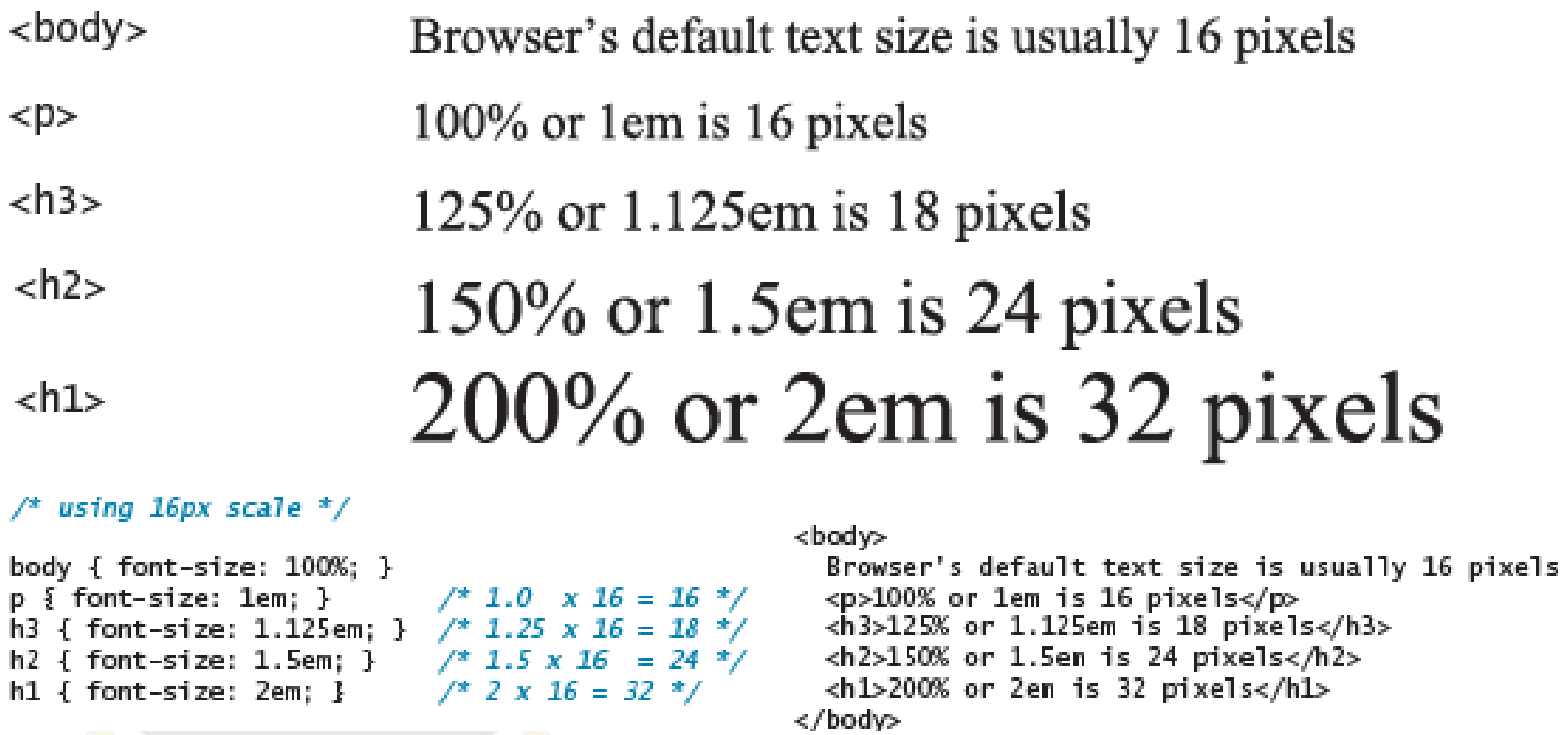
CSS Font Sizes

- Another potential problem with web fonts is font sizes. In a print-based program such as a word processor, specifying a font size is unproblematic.
- Making some text 12 pt will mean that the font's bounding box (which in turn is roughly the size of its characters) will be 1/6 of an inch tall when printed, while making it 72 pt will make it roughly one inch tall when printed.
- However, absolute units such as points and inches do not translate very well to pixel-based devices.
- Somewhat surprisingly, pixels are also a problematic unit.
- Newer mobile devices in recent years have been increasing pixel densities so that a given CSS pixel does not correlate to a single device pixel.

CSS Font Sizes

- So while sizing with pixels provides precise control, if we wish to create web layouts that work well on different devices, we should learn to use relative units such as **em units** or **percentages** for our font sizes (and indeed for other sizes in CSS as well).
- One of the principles of the web is that the user should be able to change the size of the text if he or she so wishes to do so; using percentages or em units ensures that this user action will “work,” and not break the page layout.
- When used to specify a font size, both em units and percentages are relative to the parent’s font size.

Using percents and em units for font sizes

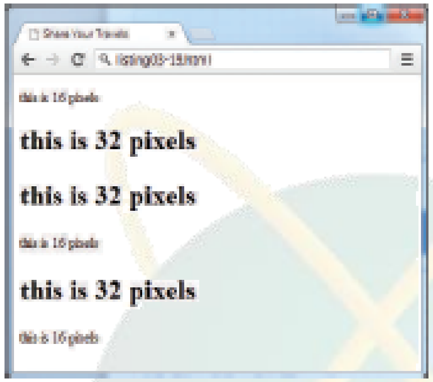


`<body>`	Browser's default text size is usually 16 pixels
`<p>`	100% or 1em is 16 pixels
`<h3>`	125% or 1.125em is 18 pixels
`<h2>`	150% or 1.5em is 24 pixels
`<h1>`	200% or 2em is 32 pixels

```
/* using 16px scale */

body { font-size: 100%; }
p { font-size: 1em; }          /* 1.0  x 16 = 16 */
h3 { font-size: 1.125em; }     /* 1.25 x 16 = 18 */
h2 { font-size: 1.5em; }       /* 1.5 x 16  = 24 */
h1 { font-size: 2em; }         /* 2 x 16 = 32 */
```

```
<body>
  Browser's default text size is usually 16 pixels
  <p>100% or 1em is 16 pixels</p>
  <h3>125% or 1.125em is 18 pixels</h3>
  <h2>150% or 1.5em is 24 pixels</h2>
  <h1>200% or 2em is 32 pixels</h1>
</body>
```

```
<body>
  <p>this is 16 pixels</p>
  <h1>this is 32 pixels</h1>
  <article>
     <h1>this is 32 pixels</h1>
     <p>this is 16 pixels</p>
     <div>
        <h1>this is 32 pixels</h1>
        <p>this is 16 pixels</p>
     </div>
  </article>
</body>
```



this is 16 pixels

this is 32 pixels

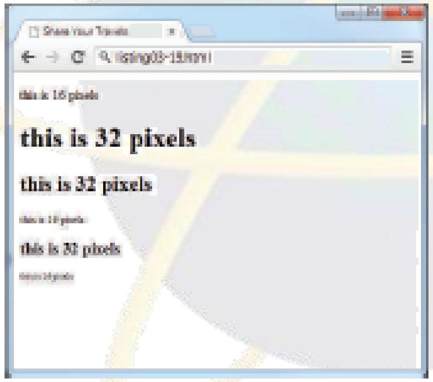
this is 32 pixels

this is 16 pixels

this is 32 pixels

this is 16 pixels

```
/* using 16px scale */

body { font-size: 100%; }
p    { font-size: 1em; }       /* 1 x 16 = 16px */
h1   { font-size: 2em; }       /* 2 x 16 = 32px */
```



this is 16 pixels

this is 32 pixels

this is 32 pixels

this is 16 pixels

this is 32 pixels

this is 16 pixels

```
/* using 16px scale */

body { font-size: 100%; }
p    { font-size: 1em; }
h1   { font-size: 2em; }

article { font-size: 75% }   /* h1 = 2 * 16 * 0.75 = 24px
                                 p = 1 * 16 * 0.75 = 12px */

div  { font-size: 75% }      /* h1 = 2 * 16 * 0.75 * 0.75 = 18px
                                 p = 1 * 16 * 0.75 * 0.75 = 9px */
```

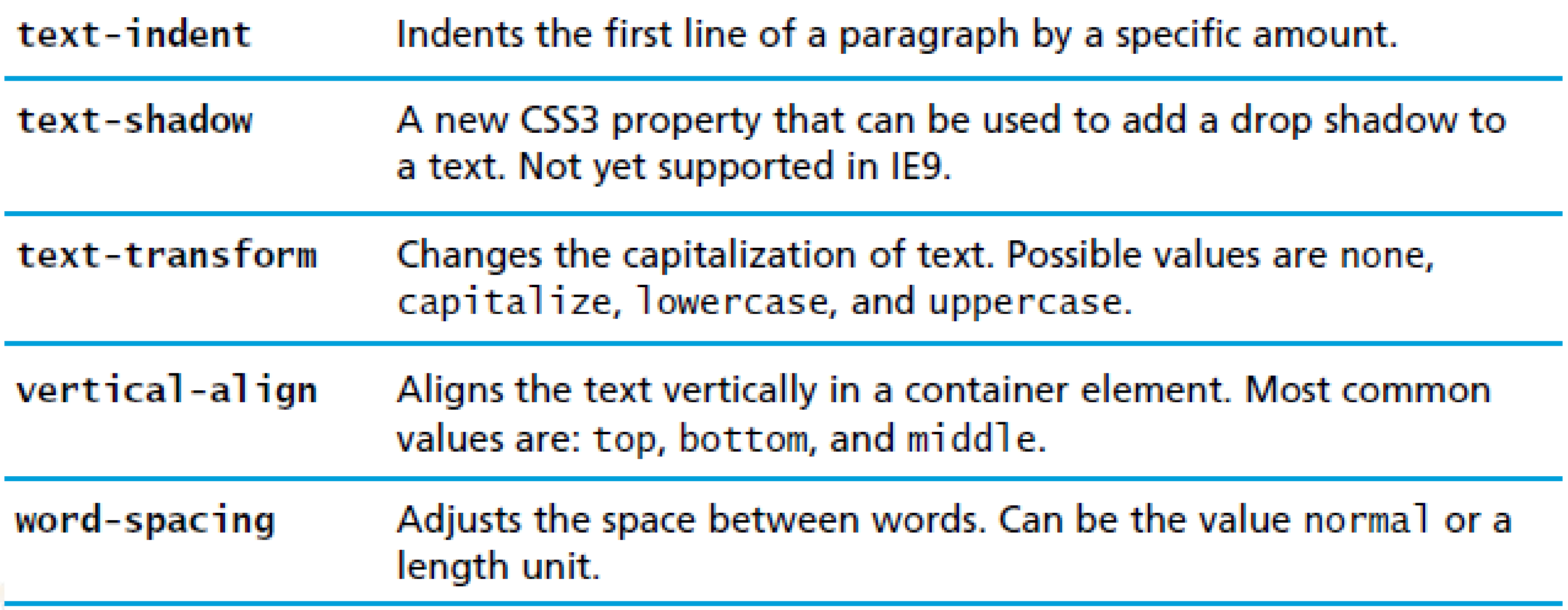
CSS Paragraph Properties

- Just as there are properties that affect the font in CSS, there are also a range of CSS properties that affect text independently of the font.
- Many of the most common text properties are shown in the table in next page, and like the earlier font properties, many of these will be familiar to anyone who has used a word processor.

Text Properties

Property	Description
`letter-spacing`	Adjusts the space between letters. Can be the value `normal` or a length unit.
`line-height`	Specifies the space between baselines (equivalent to leading in a desktop publishing program). The default value is `normal`, but can be set to any length unit. Can also be set via the shorthand `font` property.
`list-style-image`	Specifies the URL of an image to use as the marker for unordered lists.
`list-style-type`	Selects the marker type to use for ordered and unordered lists. Often set to none to remove markers when the list is a navigational menu or a input form.
`text-align`	Aligns the text horizontally in a container element in a similar way as a word processor. Possible values are `left`, `right`, `center`, and justify.
`text-decoration`	Specifies whether the text will have lines below, through, or over it. Possible values are: none, `underline`, `overline`, `line-through`, and `blink`. Hyperlinks by default have this property set to `underline`.
`text-direction`	Specifies the direction of the text, left-to-right (`ltr`) or right-to-left (`rtl`).

Text Properties



`text-indent`	Indents the first line of a paragraph by a specific amount.
`text-shadow`	A new CSS3 property that can be used to add a drop shadow to a text. Not yet supported in IE9.
`text-transform`	Changes the capitalization of text. Possible values are none, `capitalize`, `lowercase`, and `uppercase`.
`vertical-align`	Aligns the text vertically in a container element. Most common values are: `top`, `bottom`, and `middle`.
`word-spacing`	Adjusts the space between words. Can be the value `normal` or a length unit.

Quick Review Question

1. What are the main benefits of using CSS?
2. Compare the approach the W3C has used with CSS3 in comparison to CSS2.1.
3. What are the different parts of a CSS style rule?
4. What is the difference between a relative and an absolute measure unit in CSS?
5. Why are relative units preferred over absolute units in CSS?
6. What is an element selector and a grouped element selector? Provide an example of each.
7. What are class selectors? What are id selectors? Briefly discuss why you would use one over the other.

Quick Review Question

8. What are contextual selectors? Identify the four different contextual selectors.
9. What are pseudo-class selectors? What are they commonly used for?
10. What does cascade in CSS refer to?
11. What are the three cascade principles used by browsers when style rules conflict? Briefly describe each.
12. Illustrate the CSS box model. Be sure to label each of the components of the box.
13. What is a web font stack? Why are they necessary?

Summary of Main Teaching Points

- Cascading Style Sheets are a vital component of any modern website.
- This chapter provided a detailed overview of most of the major features of CSS.

Question and Answer Session

Q & A

What we will cover next

- JavaScript - Client Side Scripting