

SWE 363: Web Engineering & Development

Module 3

Cascading Style Sheets



Objectives

- ☐ Learn the basics of CSS
- ☐ Learn the different CSS types
- ☐ Use style sheets to separate presentation from content

- ❑ What is CSS?
- ❑ CSS Syntax
- ❑ Types of CSS
 - Inline stylesheet
 - Embedded stylesheet
 - External (linked) stylesheet
- ❑ Selectors
- ❑ The Cascade
- ❑ CSS background properties
- ❑ CSS Box Model
- ❑ Text and Box Shadow
- ❑ Rounded Corners
- ❑ Color & Gradients
- ❑ CSS3 Multiple Columns
- ❑ CSS Pseudo-classes
- ❑ Positioning Elements
- ❑ Floating Elements

References

- ❑ *“Internet and World Wide Web How to Program”, International Edition, 5/E,* Pearson Education Inc. 2012. Chapters 4 & 5.
- ❑ *“Fundamentals of Web Development”* Book by Randy Connolly and Ricardo Hoar, Pearson Education Inc. 2015. Chapters 3 & 5
- ❑ W3C
 - W3 Schools CSS Tutorial: <http://www.w3schools.com/css/default.asp>
 - Cascading Style Sheets: <http://www.w3.org/Style/CSS/>

What is CSS?

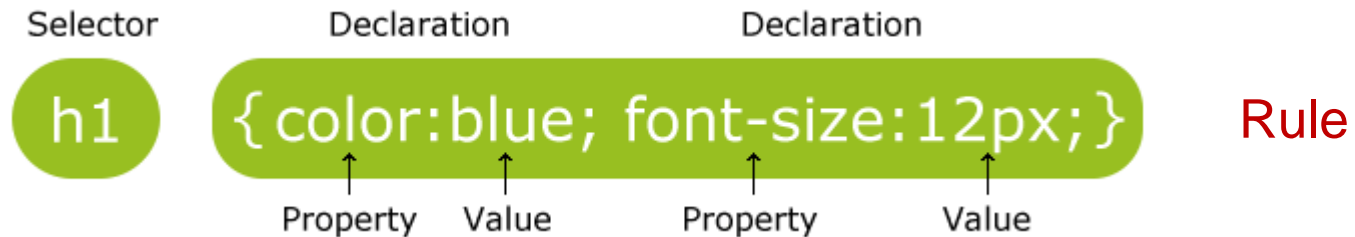
- ❑ HTML was intended to describe the **content** of a document
- ❑ HTML acquired **more and more tags** to control appearance
 - Content and appearance became more **intertwined**
- ❑ >> HTML should not describe the formatting or presentation of documents.
- ❑ **Solution:** specify the **presentation** of elements **separately** from the **structure** of the document
 - While HTML describes the **content** of the page, CSS (**Cascading Style Sheets**) describes **how** information is to be displayed, not **what** is being displayed
- ❑ CSS is a **W3C standard** for describing the appearance of HTML elements
 - With CSS, we can assign font **properties, colors, sizes, borders, background images**, and even **position elements** on the page

What is CSS?

- ❑ A **stylesheet** is a **set of rules** defining **how** an html element will be “presented” in the browser.
 - These rules are targeted to specific elements in the html document
- ❑ How 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.
- ❑ Some benefits of using CSS:
 - Improved site maintainability, changeability and reusability
 - Improved control over formatting. Rich design and layout.
 - Improved output flexibility (**responsive design**)
 - It's easier to generate documents with **consistent look**
 - All pages in the site look the same

CSS Syntax

CSS Rules: Selectors & Declarations



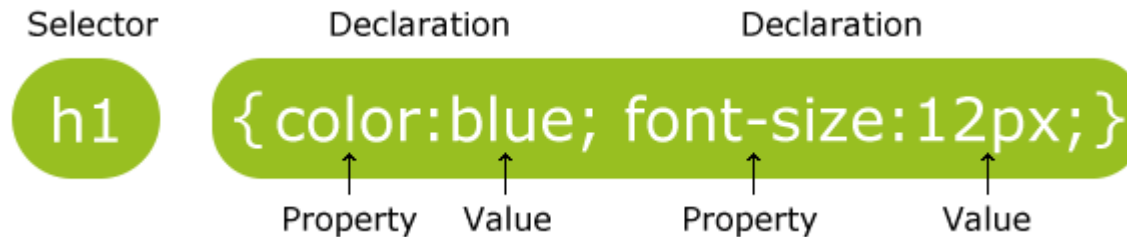
- ❑ A CSS Style Sheet is basically a collection of [rules](#), each rule contains 2 parts:
 - [Selector \(s\)](#)
 - [Declaration \(s\)](#), which describe how these elements should be displayed.
- ❑ Each **rule starts with** a **selector**
 - The *selector* identifies *which element or elements* in the HTML document will be affected by the declarations in the rule
 - selectors are *html or xml tags* to be affected

Allows one style to be applied simultaneously to many tags

```
selector, ..., selector {  
    property: value;  
    ...  
    property: value  
}
```

CSS Syntax

CSS Rules: Properties & Values



- ❑ Declarations consist of 2 parts: a **property** and a **value**.
 - **property** and **value** describe the **appearance of that tag**;
 - Properties and values are separated with a colon (:)
 - a **semicolon** must be used between different declarations.

- ❑ 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**.
 - E.g. for color property: Use one of **17** standard color names. CSS3 has **140** standard names.

- ❑ If the value is multiple words, put **“quotes”** around the value

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 box-shadow color opacity |
| Borders | border* border-color border-width border-style border-top, border-left, ...* border-image* border-radius |

CSS Properties

| Property Type | Property |
|---------------|--|
| Spacing | padding padding-bottom, padding-left, ... margin margin-bottom, margin-left, ... |
| 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 |
| Effects | animation* filter perspective transform* transition* |

For further info.:

<http://www.htmlhelp.com/reference/css/properties.html>

CSS Syntax

Relative and absolute Units

- ❑ The units are measured can be either
 - **absolute** such as pixels, points and so on
 - **relative** such as percentages (%) and em units.
- ❑ Specifying CSS units is **mandatory** for **non-zero values**,
 - there is no default unit.
 - Missing or ignoring a unit would be treated as an error.
- ❑ **Relative Length Units**
 - Relative length units specify a length relative to another length property.
 - The **em** and **ex** units depend on the font size that's applied to the element.

| Unit | Description |
|------|----------------------------------|
| em | the current font-size |
| ex | the x-height of the current font |

```
P {  
  font-size: 16px;  
  line-height: 2.5em;  
}
```

CSS Syntax

Relative and absolute Units

❑ Absolute Length Units

- are fixed in relation to each other. They are highly dependent on the output medium, so are mainly useful when the output environment is known.

```
h1 { margin: 0.5in; }    /* inches */
```

```
h2 { line-height: 3cm; } /* centimeters */
```

```
h3 { word-spacing: 4mm; } /* millimeters */
```

```
h4 { font-size: 12pt; }  /* points */
```

```
h5 { font-size: 1pc; }   /* picas */
```

```
h6 { font-size: 12px; }  /* picas */
```

- 1in is equal to 2.54cm
- In CSS, one point is defined as 1/72 inch (0.353mm)
- 1pc is equal to 12pt.
- 1px is equal to 0.75pt.

| 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 |
| 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 |

CSS Syntax

Examples

- ❑ `h1 {color: green; font-family: Verdana}`
- ❑ `h1,h2,h3 {font-family: Arial, sans-serif;}`
 - use 1st available font
- ❑ `p, table, li { /* apply to all these tags */
font-family: "Courier New"; /* quote values containing spaces */
margin-left: 15pt; /* specify indentation */
}`
- ❑ `p, li, th, td {font-size: 80%;} /* 80% of size in containing element */`
- ❑ `body{background-color:#FAEBD7} /* colors can be specified in hex */`

Location of Styles

Inline Styles

- ❑ Inline styles are style rules placed within an HTML element via the style attribute,
- ❑ Used for styling a particular element

```
<p style="font-family: sans-serif; color: red;">
```

This is a paragraph

```
</p>
```

HTML

- ❑ Higher precedence than embedded or linked styles
- ❑ Use to override an external or embedded style specification.
- ❑ Using inline styles is generally discouraged and should be avoided when possible ?
 - increase bandwidth and decrease maintainability

Location of Styles

Embedded Styles

❑ *Embedded style* sheets (also called *internal styles*)

- Styles are defined within the `<style> </style>` element , which is placed inside the `<head>` element of an HTML document

```
<head>
  <style>
    p { font-family: sans-serif; color: red; }
    h2 { background-color: yellow; }
  </style>
</head>
```

HTML

❑ Applicable to an *entire document*

❑ Using embedded styles is *discouraged*?

- Since each HTML document has its own `<style>` element, it is more difficult to consistently style multiple documents when using embedded styles.

External Style Sheet

- ❑ *External style* sheets are **style rules placed within a external text file** with the .css extension.
 - can be applied by referencing the file inside the **<head>** tag

```
p { font-family: verdana, sans-serif; font-size: 12pt; color: red}  
h1 {font-family: serif; font-size: 14pt; color: green}  
h2 {font-family: serif; font-size: 11pt; color: blue}
```

CSS

Save this text
file as *test.css*

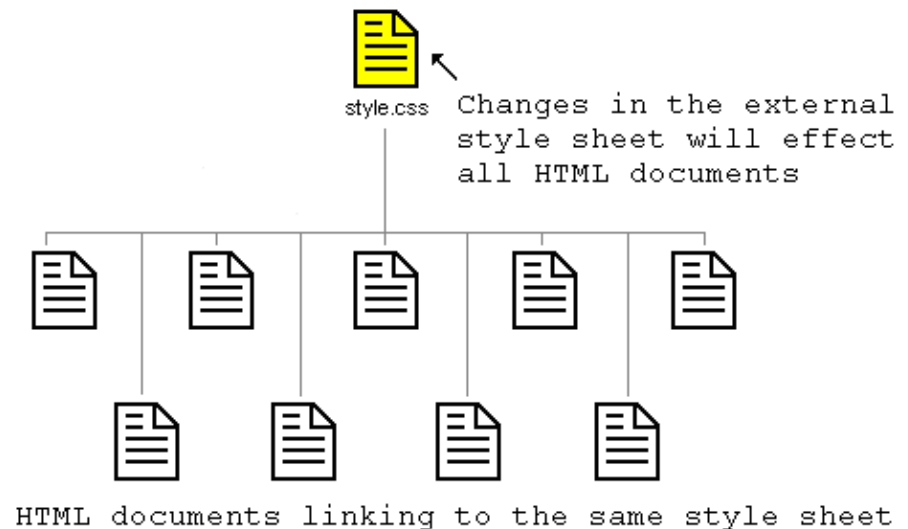
```
<head>  
  <link rel="stylesheet" href="test.css" />  
</head>
```

HTML

To apply the stylesheet
“*test.css*” to an HTML
document, call it in from
the **header**:

External Style Sheets..

- ❑ This single stylesheet can be used to define the look of **multiple pages**.

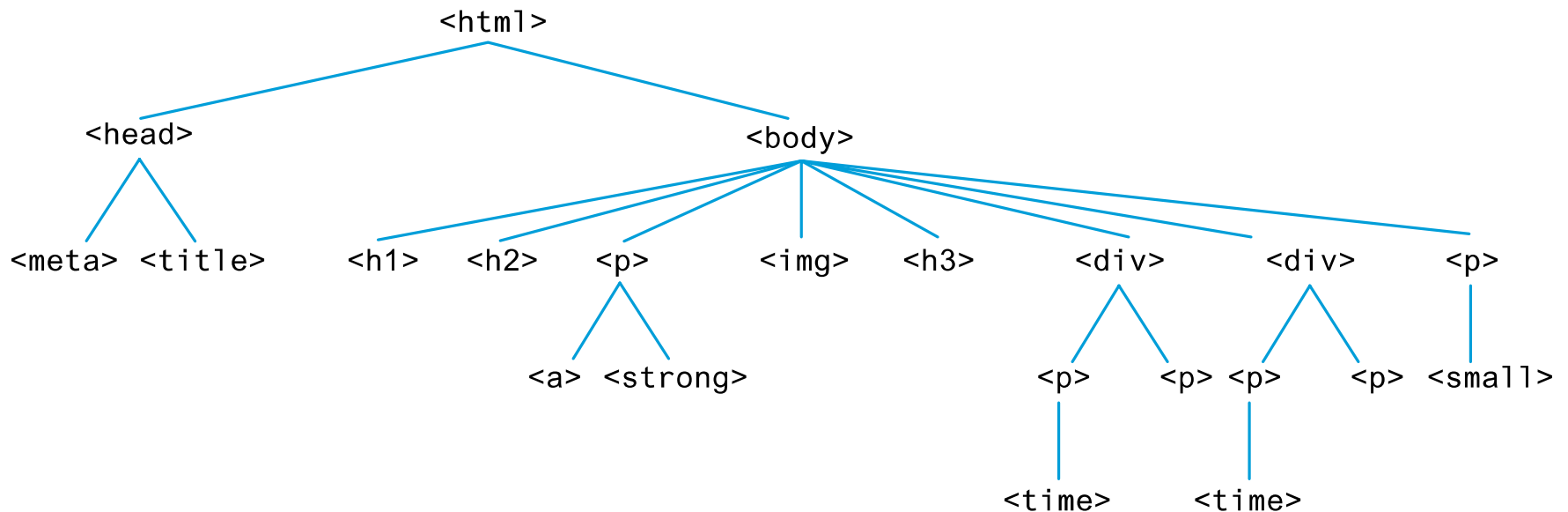


❑ Advantages

- **A uniform “look and feel”** by applying the same style to multiple documents
- **Maintainability** – making changes to styles will be easier and faster
- **Better performance** – since it can be cached once for all pages in the site, it reduces download time and bandwidth usage on the server
- **Reusability** reduces the development time and effort

Selectors

❑ Remember the DOM tree



Selectors

Element Selectors

- ❑ **Element selectors** select all instances of a given HTML element.

```
selector
|
h1 { color: red; }
      |      |
      property value

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

- ❑ A **class selector** allows you to target **different HTML elements** (at the same time) regardless of their position in the document tree using the same class attribute value.

With a **class selector** you can define different styles for the same type of HTML element

- First define the class
- Then use the class in your HTML code

Selectors

Class Selector

```
<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>
```



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

An **id selector** allows you to target a specific element by its id attribute regardless of its type or position

Selectors

id Selector

```
<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>
```



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

Example

```
body {  
    background-color: darkslategrey;  
    color: azure;  
    font-size: 1.1em;  
}  
h1{  
    color: coral;  
}  
#intro {  
    font-size: 1.3em;  
}  
.colorful {  
    color: orange;  
}
```

External Styles

Allow you to define styles for the whole website.

This has a style applied via a class.

This is H2.

```
<html>  
  <head>  
    <title>My Example</title>  
    <link rel="stylesheet" href="sty.css">  
  </head>  
  <body>  
    <h1>External Styles</h1>  
    <p id="intro">Allow you to define styles for the whole website.</p>  
    <p class="colorful">This has a style applied via a class.</p>  
    <h2 class="colorful"> This is h2.</h2>  
  </body>  
</html>
```

Selectors

attribute Selector

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

Result

- [Internal link](#)
- [Example link](http://example.com)
- [Insensitive internal link](#)
- [Example org link](http://example.org)

```
a {  
  color: blue;  
}  
  
/* Internal links, beginning with "#" */  
a[href^="#"] {  
  background-color: gold;  
}  
  
/* Links with "example" anywhere in the URL */  
a[href*="example"] {  
  background-color: silver;  
}  
  
/* Links with "insensitive" anywhere in the URL,  
   regardless of capitalization */  
a[href*="insensitive" i] {  
  color: cyan;  
}  
  
/* Links that end in ".org" */  
a[href$=".org"] {  
  color: red;  
}
```

```
<ul>  
  <li><a href="#internal">Internal link</a></li>  
  <li><a href="http://example.com">Example link</a></li>  
  <li><a href="#InSensitive">Insensitive internal link</a></li>  
  <li><a href="http://example.org">Example org link</a></li>  
</ul>
```


Selectors

attribute Selector

| 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 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" |

The Cascade: How Styles Interact

- ❑ The “**Cascade**” in CSS refers to how conflicting rules are handled when multiple CSS rules are applied to the same elements.
 - For example, if there are two rules defining the color of your **h1** elements, the rule that comes last in the cascade order will “trump” the other.
- ❑ An inline style (inside an HTML element) has the **highest priority**, which means that it will override every style declared inside the <head> tag, in an external style sheet, and in the browser (default value).
- ❑ CSS uses the following **cascade principles to help it deal with conflicts**:
 - **inheritance**,
 - **specificity**, and
 - **location**

The Cascade: Inheritance

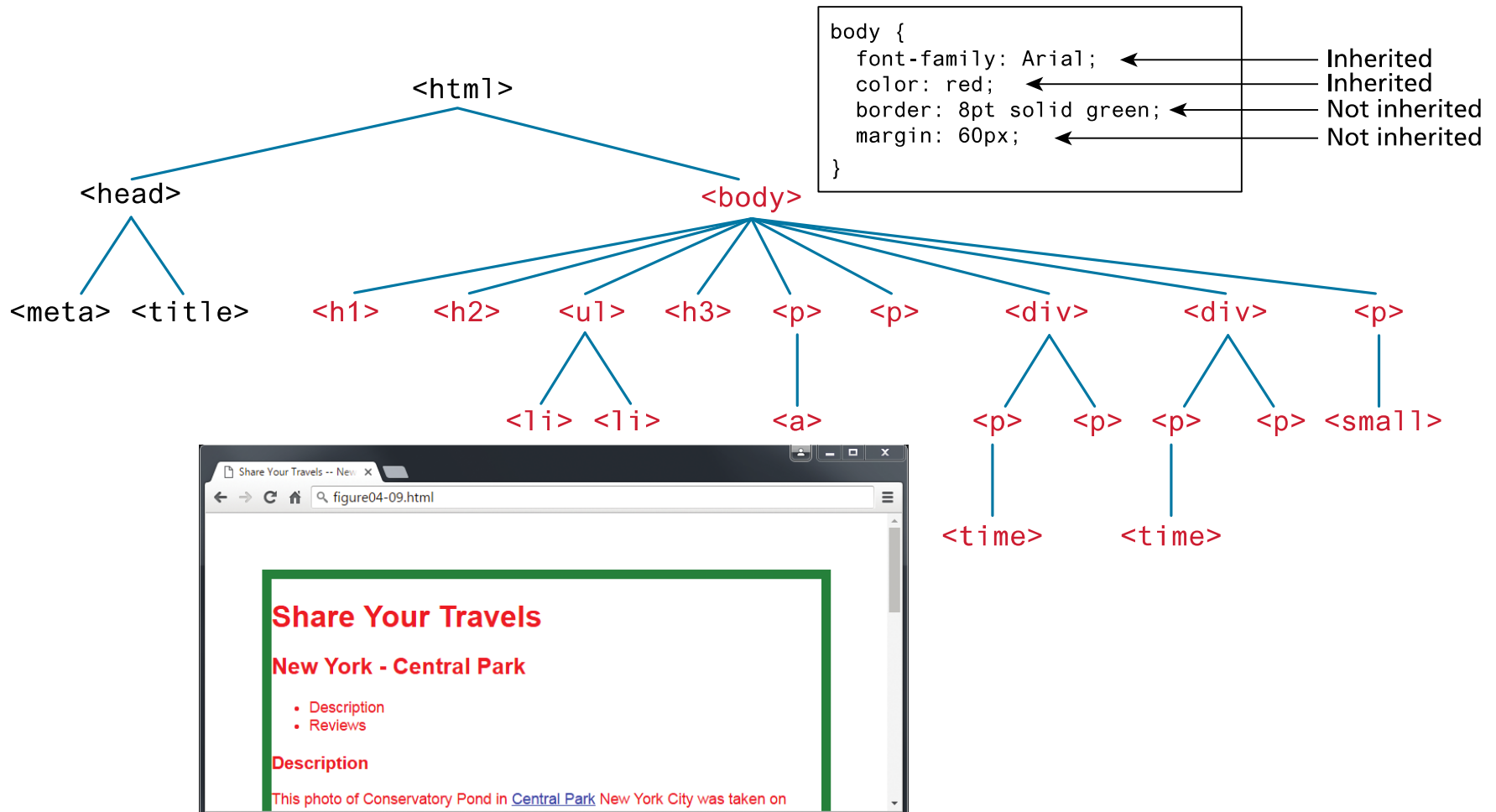
- ❑ Multiple values of one property can be set or **inherited** on the same element
- ❑ **Inheritance**: which style prevails when several are present?
 - Inline (local) **overrides** internal (global)
 - Internal (global) **overrides** external (linked).
- ❑ Tags embedded in other tags inherit style attributes, e.g.
 - **<p>** inherits from **<body>** because it can only appear in the **<body>** section
 - **** inherits from **** because **** appears inside **** tag

```
<body style="color : red">  
  <p>This paragraph will appear with red text.</p>  
  <p style="color : green">This paragraph will appear with green text because it  
    explicitly overrides the red text inherited from the body tag.</p>  
</body>
```

The Cascade: Inheritance

- ❑ Many (but not all) CSS properties affect not only themselves but their descendants as well.
 - Are inheritable:
 - Font,
 - color,
 - list, and
 - text properties
 - Not inheritable:
 - layout,
 - sizing,
 - border,
 - background, and
 - spacing properties

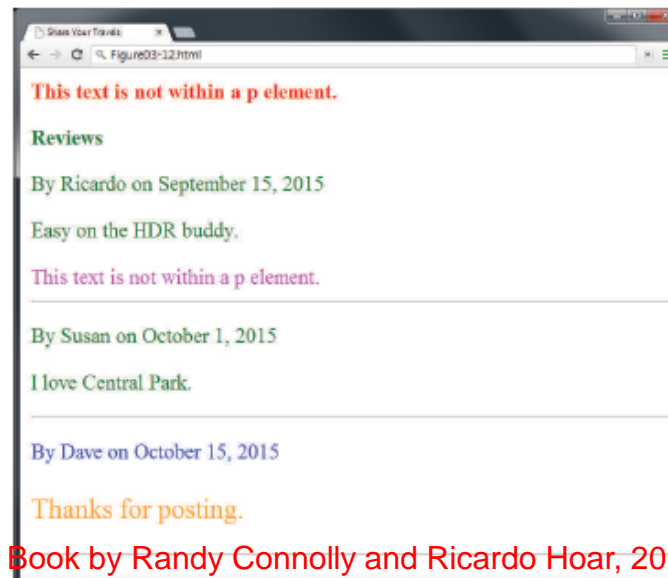
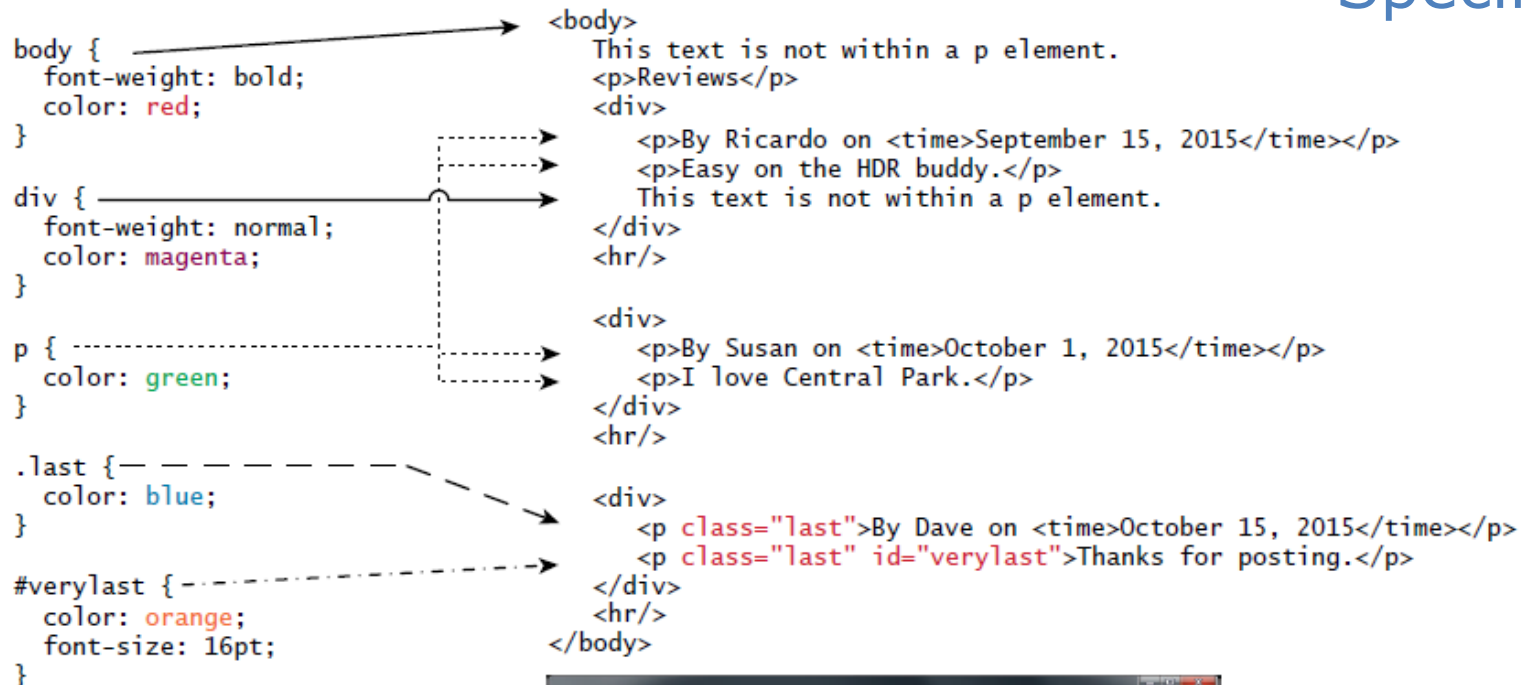
The Cascade: Inheritance



The Cascade: 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).
- ❑ Properties defined for **child and descendant elements** have a **higher specificity** than properties defined for **parent and ancestor elements**.
- ❑ Conflicts are resolved in favor of properties with a **higher specificity**, so the **child's styles take precedence**.

The Cascade: Specificity

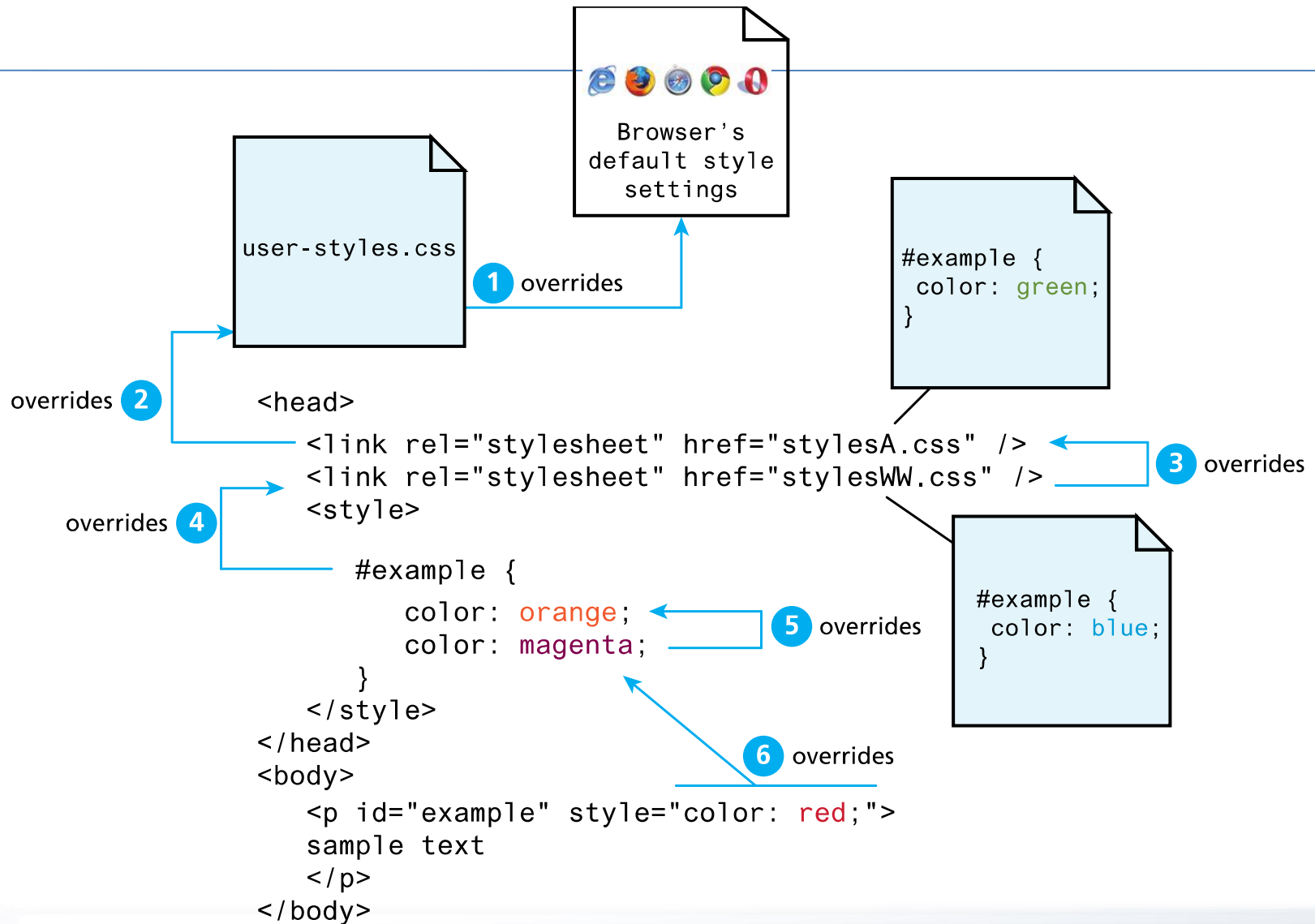


In the figure, class selectors take precedence over element selectors, and id selectors take precedence over class selectors.

The Cascade: Location

- ❑ 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.
 - ❑ The way styles will be used when there is more than one style specified for an HTML element:
 - For instance, an inline style will override one defined in an external author style sheet or an embedded style sheet.
-
- 
1. **Browser** default
 2. **External Style** Sheet (Linked) (in an external .css file)
 3. **Internal** Style Sheet (Global, or embedded) (inside the <head> tag)
 4. **Inline** Style (Local) (inside HTML element)

The Cascade: Location



Reference External Files

<link>

- ❑ <link> can be used to reference external files other than a CSS

- Link syntax

```
<link rel="relation_type" href="url" type="link_type" />
```

- **rel**: must be "stylesheet" to tell HTML the link is for a stylesheet
- **href**: attribute provides the location of the external file containing the style sheet
- **type**: attribute that specifies the MIME type of the related document needed to download the file, usually "text/css"

```
<head>  
  <title>Cascading Style Sheets</title>  
  <link href="styles.css" rel="stylesheet" type="text/css" />  
</head>
```

Reference External Files

@import

- ❑ **@import** can be used in the *<style> tag*, or used in a *.css file* by itself as a CSS command
- ❑ Essentially allows for **multiple inheritance** of style sheets attributes
 - e.g., a subsite style sheet may override a general site style sheet
 - An HTML page may override the subsite's style sheet

```
<STYLE TYPE="text/css" >  
  <!--  
    @import url(http://www.htmlhelp.com/style.css);  
    @import url(/stylesheets/punk.css);  
  -->  
</STYLE>
```

- In case of conflicting styles, the later will be applied

Reference External Files

@import: Example



```
h1 {color:brown; font-family:sans-serif}  
... other styles ...
```

inherit



```
@import url(Site.css)  
h1 {color:green; font-family:Monotype}  
... other styles ...
```

inherit



```
<style>  
  @import url(Subsite.css)  
  h1 {color:red; font-family:cursive}  
</style>
```

(Inherits styles
and overrides
some styles)

(Inherits Subsite.css
styles and overrides
some styles)

CSS Background

Backgrounds

- ❑ CSS background properties are used to define the background effects for elements.
 - CSS can control the **backgrounds of block-level elements**

```
<style type = "text/css">  
  body { background-image: url(logo.png);  
         background-position: bottom right;  
         background-repeat: no-repeat;  
         background-attachment: fixed;  
         background-color: lightgrey; }
```

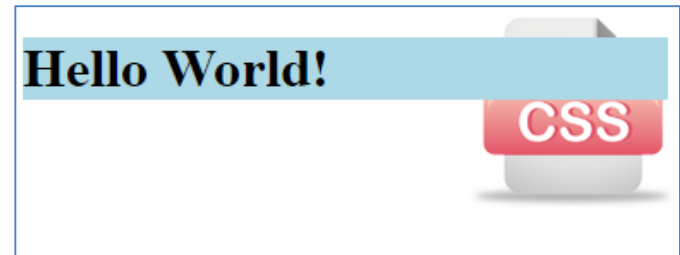
| Property | Description |
|-----------------------|---|
| background-color | color to fill background |
| background-image | image to place in background |
| background-position | placement of bg image within element |
| background-repeat | whether/how bg image should be repeated |
| background-attachment | whether bg image scrolls with page |
| background | shorthand to <u>set all background properties</u> |

Backgrounds.. Examples

```
<style>
  h1 {
    background-color: lightblue;
  }
  body {
    background-image: url("css_file.png");
    background-repeat: repeat-y;
  }
</style>
```



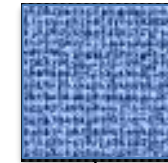
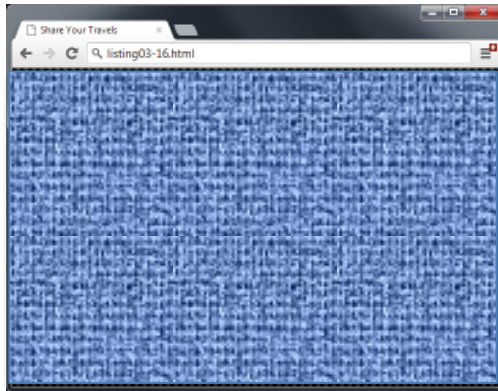
```
<style>
  h1 {
    background-color: lightblue;
  }
  body {
    background-image: url("css_file.png");
    background-repeat: no-repeat;
    background-position: right top;
  }
</style>
```



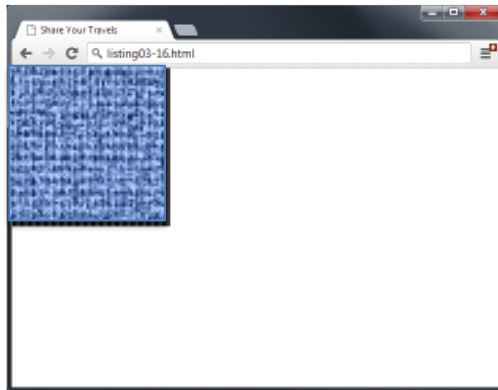
```
background-position: 370px 20px;
```

Backgrounds..

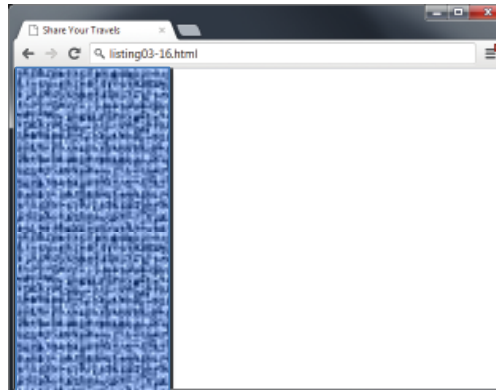
Examples



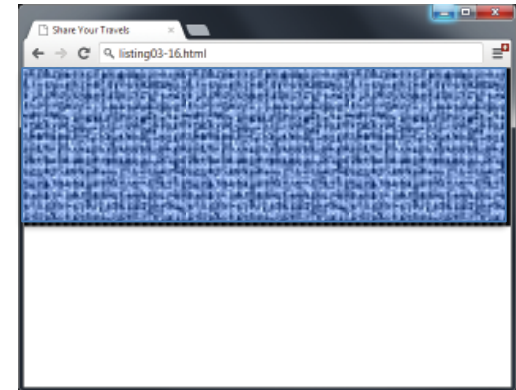
`background-image: url(../images/backgrounds/body-background-tile.gif);`
`background-repeat: repeat;`



`background-repeat: no-repeat;`



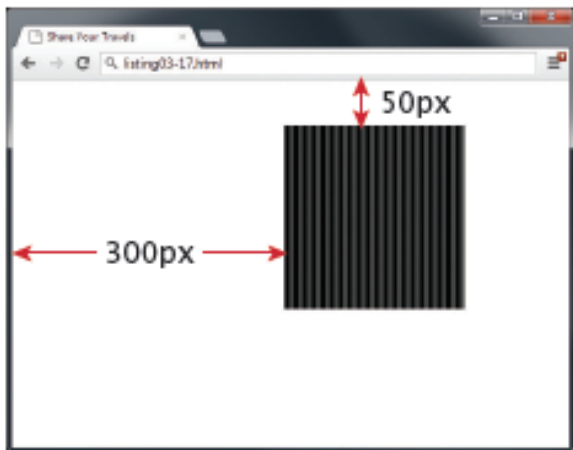
`background-repeat: repeat-y;`



`background-repeat: repeat-x;`

Backgrounds..

Examples



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

Backgrounds..

- ❑ The CSS **background** shorthand property lets you adjust all of the available background style options at once, including color image, origin and size, repeat method, and other features.

```
body {background: #ffffff url("img_tree.png")  
      no-repeat right top; }
```

- ❑ The initial values assigned by the **background** shorthand property :
 - **background-image**: none
 - **background-position**: 0% 0%
 - **background-size**: auto auto
 - **background-repeat**: repeat
 - **background-origin**: padding-box
 - **background-clip**: border-box
 - **background-attachment**: scroll
 - **background-color**: transparent

Backgrounds..

background-attachment

- ❑ If a **background-image** is specified, the **background-attachment** CSS property determines whether that image's position is fixed within the viewport, or scrolls along with its containing block.
 - **background-attachment: scroll; (default)**
 - The background scrolls along with the element.
 - **background-attachment: Fixed;**
 - The background is fixed with regard to the **viewport** . Even if an element has a scrolling mechanism, the background doesn't move with the element.
 - Example: https://www.w3schools.com/css/tryit.asp?filename=trycss_background-image_attachment
 - **background-attachment: Local;**
 - The background scrolls along with the element's contents. If the element has a scrolling mechanism, the background scrolls with the element's contents
 - Example: <https://jsfiddle.net/ltorvalds024/zf2v7yqs/>

Element Dimensions

- ❑ Specifying the **width** and **height** of an Element
 - can be set by using properties **height** and **width**
 - Their values can be **relative** or **absolute**

- ❑ **text-align** Property
 - Text in an element can be centered using **text-align: center;**
 - other values for the text-align property are **left** and **right**

- ❑ **overflow** Property
 - Problem with setting both **vertical** and **horizontal** dimensions of an element
 - Content might sometimes exceed the set boundaries, in which case the element must be made large enough for all the content to fit
 - >> set the **overflow property: scroll**, to add scroll bars if the text overflows the boundaries set for it

Element Dimensions

Example

```
<p style="width: 20%; margin-left: 10em">  
    Now is the winter of our discontent....
```

```
</p>
```

```
<p style="width: 80%; text-align: center">  
    Now is the winter of our discontent....
```

```
</p>
```

```
<p style="width: 20%; height: 150px; overflow: scroll">  
    Now is the winter of our discontent....
```

```
</p>
```

```
<head>  
    <style type="text/css">  
        p {  
            background-color: lightskyblue;  
            margin-left: 5em;  
            text-indent : 50px;  
        }  
    </style>  
</head>
```

The `text-indent` property specifies the indentation of the first line in a text-block.

Now is the winter of our discontent....
Made glorious summer by this sun of York; And
all the clouds that lour'd upon our house In the
deep bosom of the ocean buried. Now are our
brows bound with victorious wreaths; Our
bruised arms hung up for monuments; Our stern
alarums changed to merry meetings, Our
dreadful marches to delightful measures. Grim-
visaged war hath smooth'd his wrinkled front;

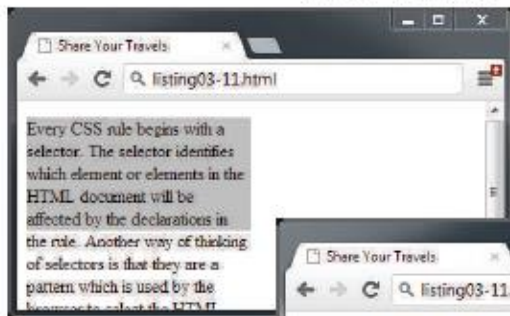
Now is the winter of our discontent.... Made glorious summer by this sun of York; And all the clouds that lour'd upon our house In the deep bosom of the ocean buried. Now are our brows bound with victorious wreaths; Our bruised arms hung up for monuments; Our stern alarums changed to merry meetings, Our dreadful marches to delightful measures. Grim-visaged war hath smooth'd his wrinkled front;

Now is the winter of our discontent...
Made glorious summer by this sun of York;
And all the clouds that lour'd upon our house
In the deep bosom of the ocean buried. Now
are our brows bound with victorious wreaths;
Our bruised arms hung up for monuments;
Our stern alarums changed to merry meetings,

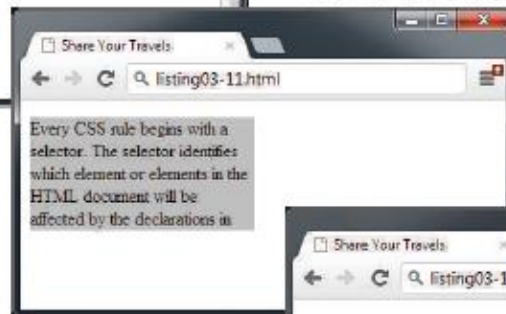
Element Dimensions

Overflow Property

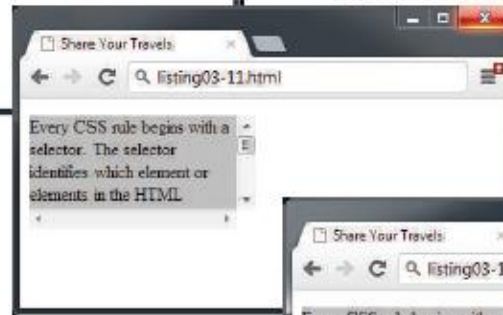
`overflow: visible;`



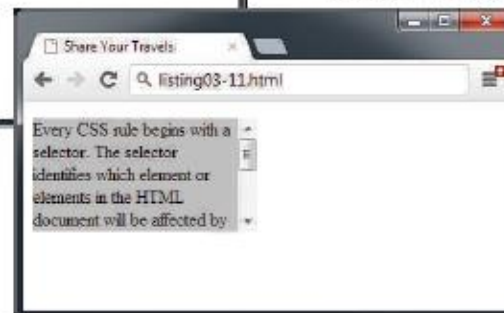
`overflow: hidden;`



`overflow: scroll;`



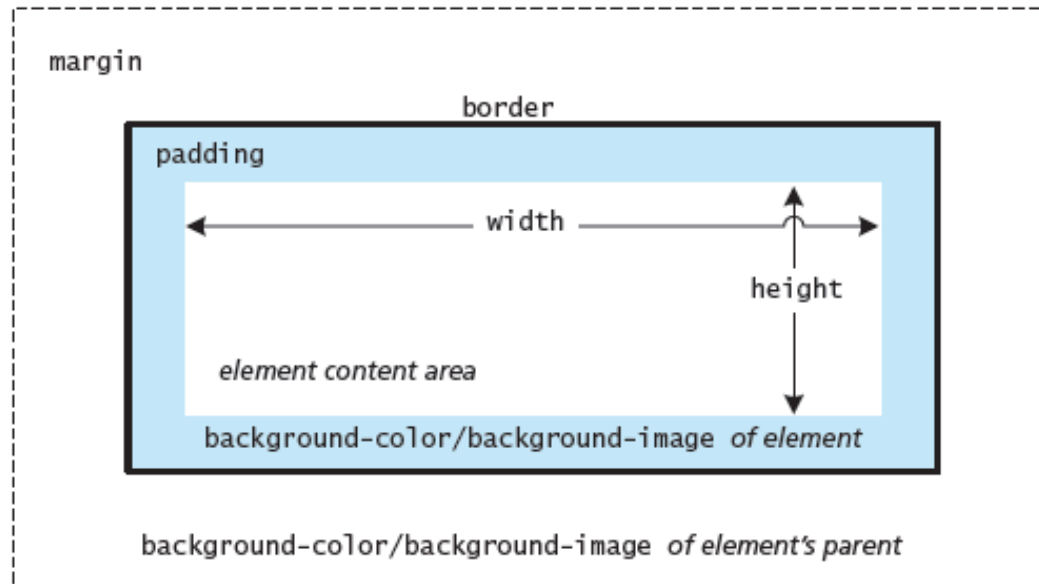
`overflow: auto;`



CSS Box Model

CSS Box Model

- ❑ Every block element in CSS is effectively **inside a box**, and can have **margins**, **padding** and **borders** applied to it.

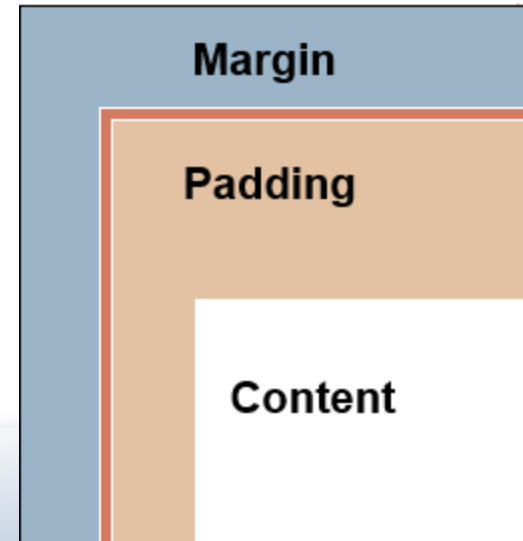


- ❑ **Box widths** can be specified in **absolute** values (e.g. px) or in **relative** values, usually:
 - em - width values **relative** to the size of the font in ems
 - percentage - width values **relative** the containing box's content region
- ❑ The **root** (or top-most) element's containing box is effectively the browser window.

CSS Box Model

Margins & Padding

- ❑ **Margins** and **Padding** may seem similar at first glance.
 - But each has its own effect on content, particularly on any **backgrounds** assigned to block and div elements.
- ❑ **Margins**
 - define the space **around elements** outside the border
 - Margin properties can have negative values in order to **deliberately overlap content**
 - Margin properties **will affect** the position of background elements (graphics and/or colors) in relation to the edges of the containing block element
 - Margin properties can be defined independently on top, right, bottom and left, or all-at-once using CSS shorthand

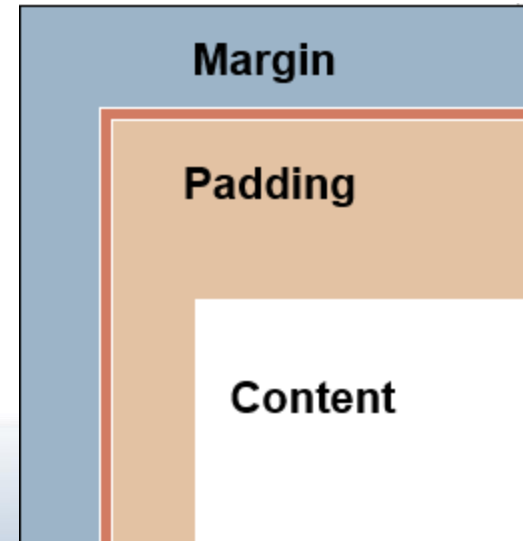


CSS Box Model

Margins & Padding

□ Padding

- Padding defines the space **around elements inside the border**; i.e between the border and the content itself
- Padding properties **cannot have negative values**
- Padding properties **do not affect** the position of background elements (graphics and/or colors) in the containing block element; only the position of content.
- Padding properties can be defined independently on top, right, bottom and left, or all-at-once using CSS shorthand

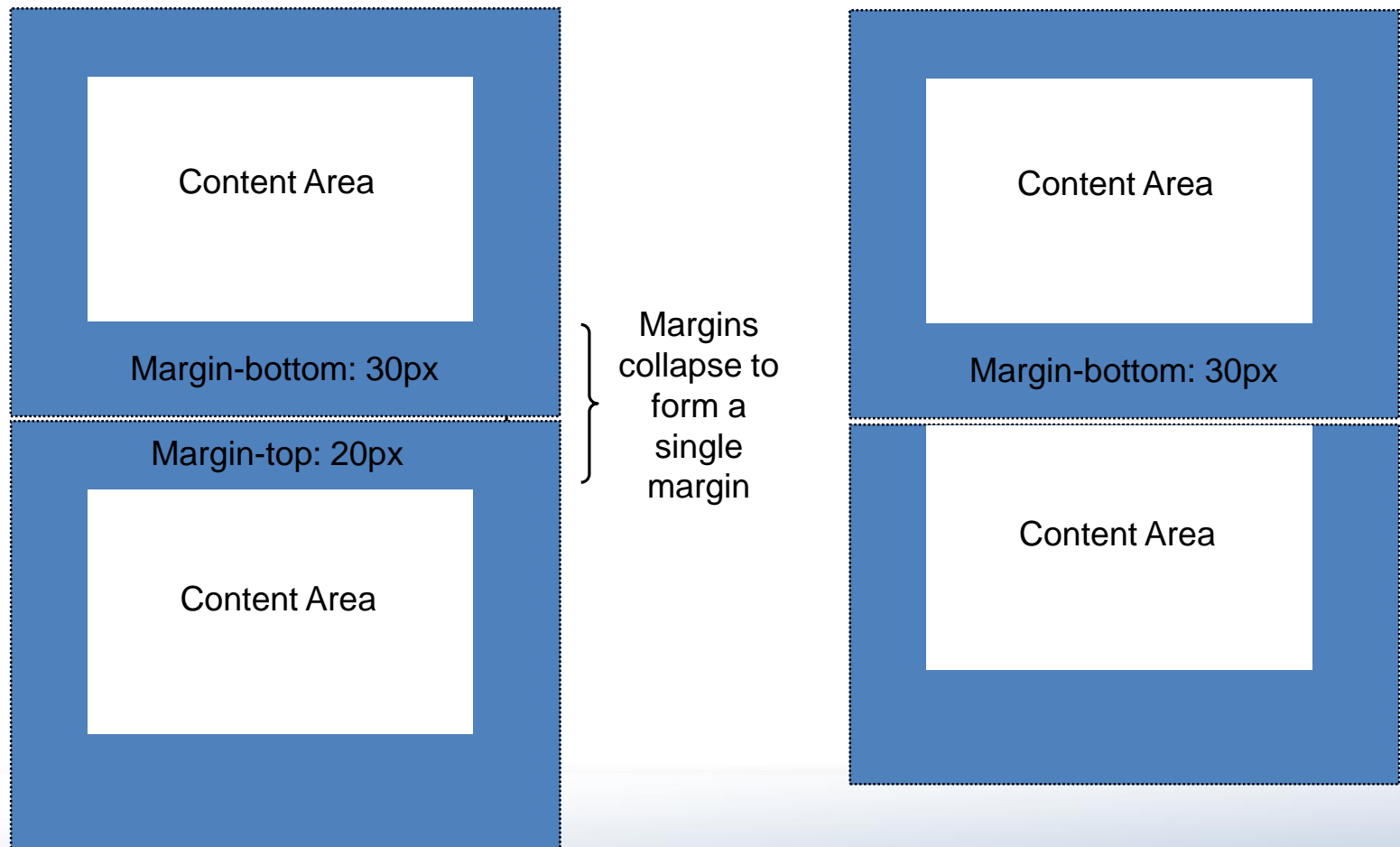


Margin Collapse: Stacked Elements

- ❑ When two or more **vertical margins** meet, they will collapse to form a single margin
- ❑ The **height of this combined margin** will be equal the height of the larger of the two margins
- ❑ Margin collapse applies when:
 - Two or more block elements are stacked one above the other,
 - or when one block element is contained within another block element

CSS Box Model

Margin Collapse: Stacked Elements



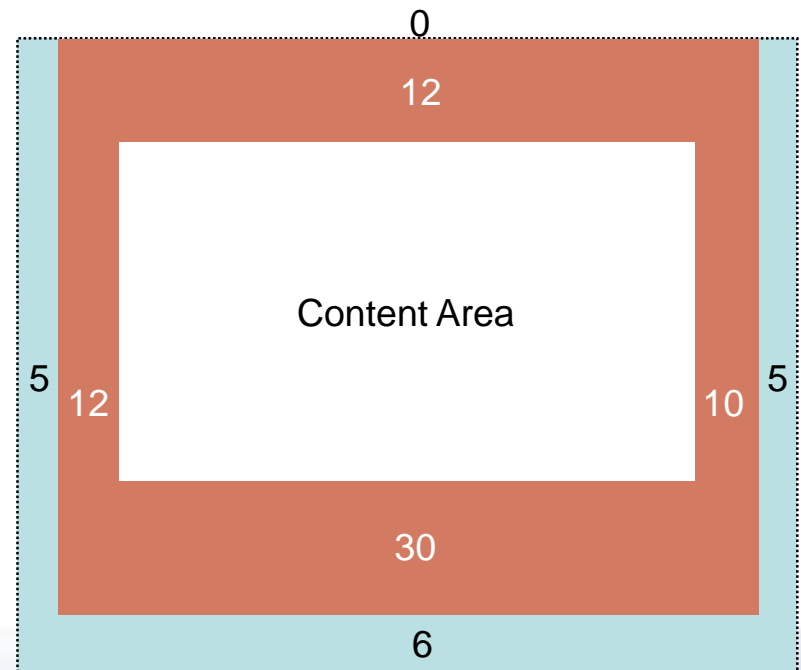
CSS Box Model

CSS Shorthand: Margin & Padding

- ❑ For margin and padding (and others), CSS provides a number of **shorthand properties** that can save on writing lines and lines of code.

- Instead of writing this:

```
#container {  
    margin-top: 0;  
    margin-right: 5px;  
    margin-bottom: 6px;  
    margin-left: 5px;  
    padding-top: 20px;  
    padding-right: 10px;  
    padding-bottom: 30px;  
    padding-left: 12px;  
}
```

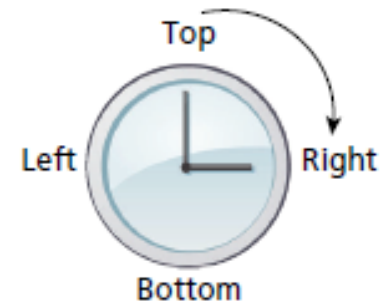
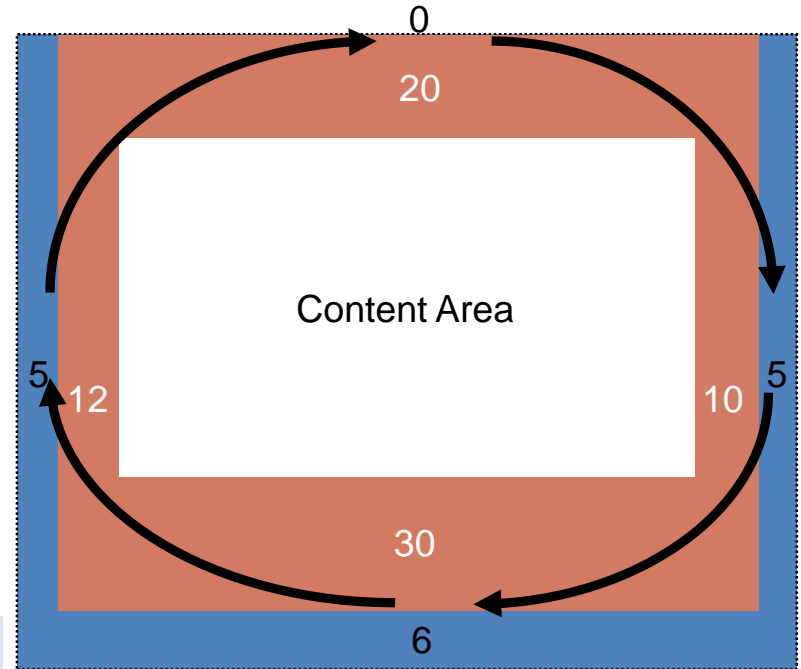


CSS Box Model

CSS Shorthand: Margin & Padding

- ❑ ...Its much easier to write this:
- ❑ The sequence order is **always clockwise**, starting from the top

```
#container {  
    padding: 20px 10px 30px 12px;  
    margin: 0px 5px 6px 5px;  
}
```



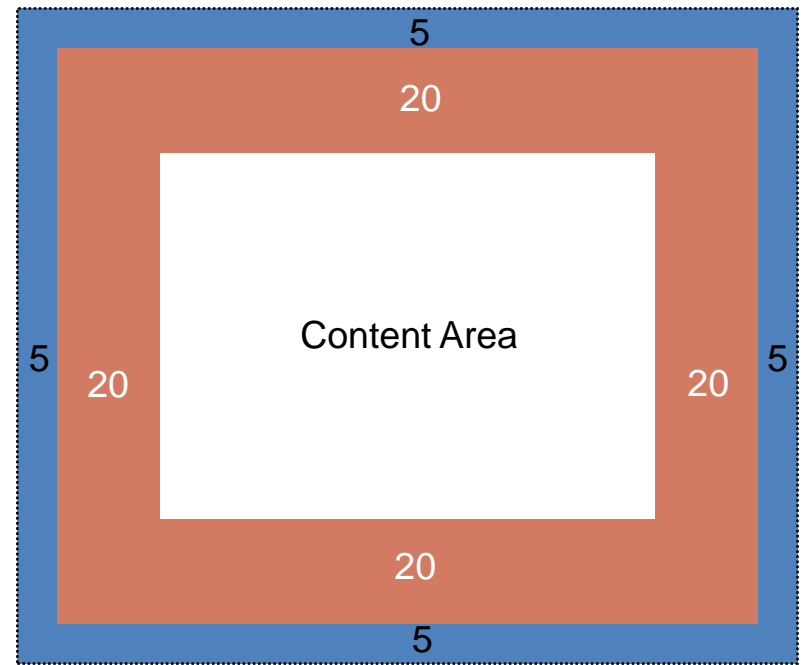
CSS Box Model

CSS Shorthand: Margin & Padding

- ❑ You can also apply just one value, example:

```
#container {  
    padding: 20px;  
    margin: 5px;  
}
```

Which will apply the value specified equally on all 4 sides



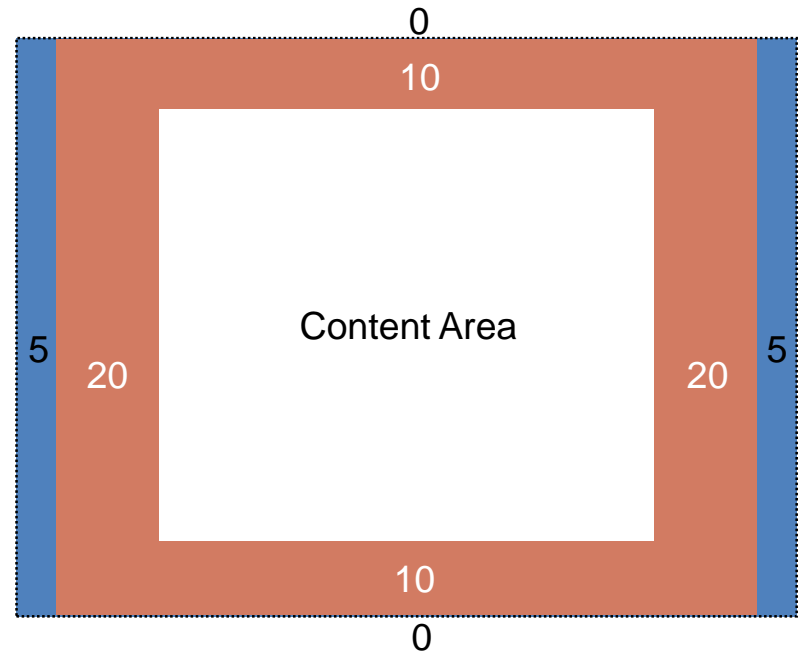
CSS Box Model

CSS Shorthand: Margin & Padding

❑ And you can apply two values, example:

```
#container {  
    padding: 10px 20px;  
    margin: 0px 5px;  
}
```

- The **first** value is applied to the *top* and *bottom*
- The **second** value is applied to the *left* and *right*



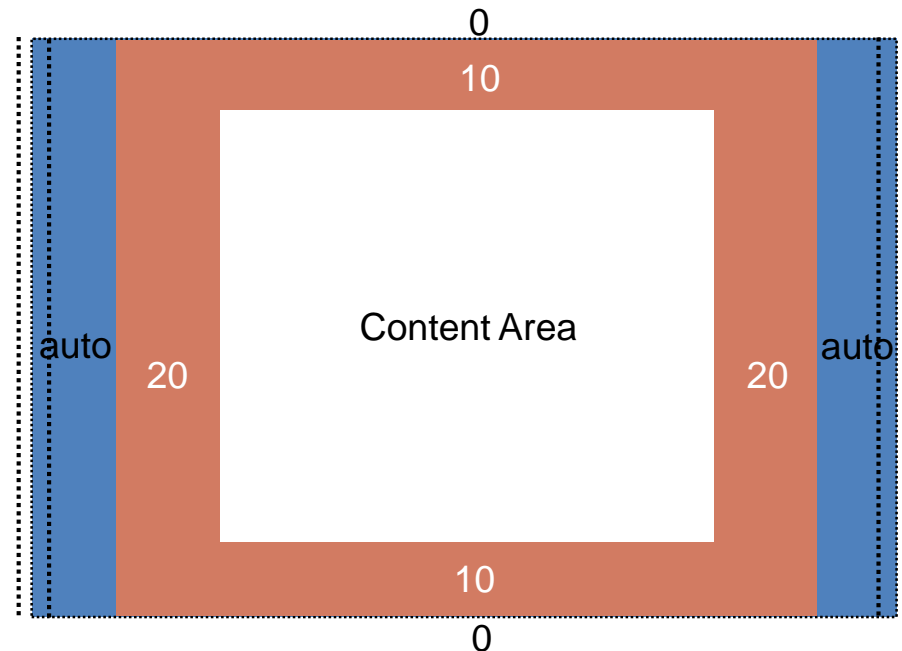
CSS Box Model

CSS Shorthand: Margin & Padding

- ❑ A useful value to remember is 'auto':

```
#container {  
    padding: 10px 20px;  
    margin: 0px auto;  
}
```

- Usually applied to the **left & right** areas of the margin property, **auto** is useful for centering a block container element in the browser window
- The element will then take up the specified width, and the remaining space will be split equally between the left and right margins



CSS Box Model

Borders

- ❑ Borders can be applied to any block element
 - Borders always come outside the padding area, but inside the margin area.
 - Border properties *cannot* have negative values



- ❑ If a border is not specified, **the default is no-border**
 - no border appears and no space between any margin and padding is allocated for a border
- ❑ Border **properties** can be defined **independently** on top, right, bottom and left, or **all-at-once** using CSS shorthand

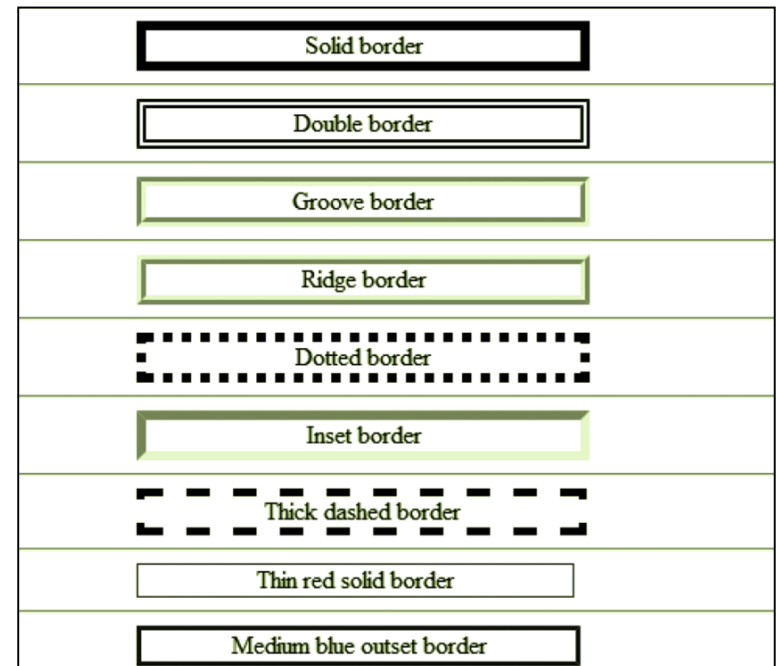
CSS Box Model

Borders

❑ The core border properties are:

- **Width:** absolute (px, in, cm, or 'thin', 'medium', 'thick'), or relative (ems)
- **Style:** dotted, dashed, solid, double, groove, ridge, inset, outset, hidden, etc
- **Color:** 'blue', 'red', #FF9900, etc

❑ You can also create the effect of a border **by using a graphic image** in a CSS background property, instead of the border property



CSS Box Model

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: <code>border-style border-width border-color</code> |
| 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. |

❑ Examples:

- https://www.w3schools.com/css/css_border.asp
- https://www.w3schools.com/cssref/css3_pr_border-image.asp
- https://www.w3schools.com/css/tryit.asp?filename=trycss3_border-radius

Text and Box Shadows

Text Shadows

- ❑ The *text-shadow* property makes it easy to add a text shadow effect to any text

```
<style type = "text/css">
  h1
  {
    text-shadow: -4px 4px 6px dimgrey; /* add shadow */
    font-size: 400%; /* increasing the font size */
  }
</style>
```

- **Horizontal offset** of the shadow
 - (-) moves the text-shadow to the left; (+) moves it to the right.
- **Vertical offset** of the shadow
 - (-) moves the shadow up, (+) moves it down.
- **Blur radius** — with **0px** would result in a shadow with a sharp edge (no blur).
 - The greater the value, the greater the blurring of the edges.
- **Color** of the text-shadow.

- Internet Explorer 9 and earlier do not support the text-shadow property.

Text Shadows

```
<style>
  h1 {
    text-shadow: 2px 2px;
  }
  h2 {
    text-shadow: 2px 2px red;
  }
  h3 {
    text-shadow: 2px 2px 5px red;
  }
  h4 {
    color: white;
    text-shadow: 2px 2px 4px #000000;
  }
  h5 {
    text-shadow: 0 0 3px #FF0000;
  }
  h6 {
    text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF;
  }
  p {
    color: white;
    text-shadow: 1px 1px 2px black, 0 0 25px blue, 0 0 5px darkblue;
  }
</style>
```

It will produce the following result –

Tutorialspoint.com

Tutorialspoint.com

Tutorialspoint.com

Tutorialspoint.com

Tutorialspoint.com

Tutorialspoint.com

Tutorialspoint.com

This property accepts a **comma-separated** list of shadows to be applied to the text.

Box Shadows

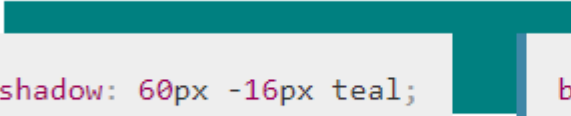
- ❑ The box-shadow property add **one or more** shadow effects around an element's frame
- ❑ You can shadow **any block-level element** in CSS3
 - **Horizontal offset** of the shadow
 - **Vertical offset** of the shadow
 - **Blur radius**
 - **Color**

Box Shadow

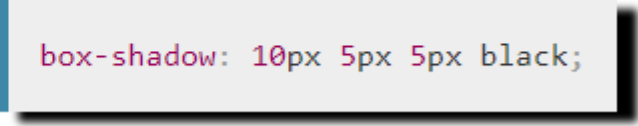
```
/* offset-x | offset-y | color */  
box-shadow: 60px -16px teal;
```

```
/* offset-x | offset-y | blur-radius | color */  
box-shadow: 10px 5px 5px black;
```

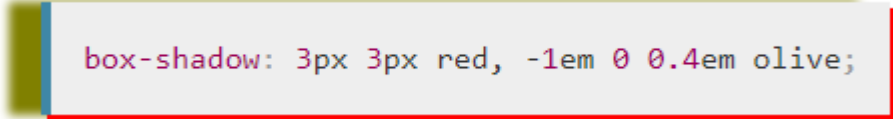
```
/* Any number of shadows, separated by commas */  
box-shadow: 3px 3px red, -1em 0 0.4em olive;
```



```
box-shadow: 60px -16px teal;
```



```
box-shadow: 10px 5px 5px black;
```



```
box-shadow: 3px 3px red, -1em 0 0.4em olive;
```

Rounded Corners

Rounded Corners

The *border-radius* property is used to add rounded corners to an element.

```
border-radius: 25px 10px 30px 20px;
```

- ❑ If you specify only **one value** for the border-radius property,
 - this radius will be applied to **all 4 corners**.
- ❑ You can specify **each corner separately** if you wish. Here are the rules:
 - **Four values**: top-left, top-right, bottom-right, bottom-left
 - **Three values**: top-left, top-right and bottom-left, bottom-right
 - **Two values**: top-left and bottom-right, top-right and bottom-left
 - **One value**: all four corners are rounded equally

Rounded Corners..

```
<style>
  #rcorners1 {
    border-radius: 15px 50px 30px 5px;
    background: #a44170;
    padding: 20px;
    width: 100px;
    height: 100px;
  }
  #rcorners2 {
    border-radius: 15px 50px 30px;
    background: #a44170;
    padding: 20px;
    width: 100px;
    height: 100px;
  }
  #rcorners3 {
    border-radius: 15px 50px;
    background: #a44170;
    padding: 20px;
    width: 100px;
    height: 100px;
  }
</style>
```

```
<p id="rcorners1"></p>
<p id="rcorners2"></p>
<p id="rcorners3"></p>
```



Rounded Corners..

- ❑ The possible values for *Rounded Corners*

| Property | Description |
|--|--|
| border-radius | A shorthand property for setting all the four border-*-*-radius properties |
| border-top-left-radius | Defines the shape of the border of the top-left corner |
| border-top-right-radius | Defines the shape of the border of the top-right corner |
| border-bottom-right-radius | Defines the shape of the border of the bottom-right corner |
| border-bottom-left-radius | Defines the shape of the border of the bottom-left corner |

Color & Gradients

- ❑ CSS3 allows you to **express color in several ways** in addition to standard color names or hexadecimal values
- ❑ CSS3 has Supported additional color properties as follows
 - RGBA colors
 - HSL colors
 - HSLA colors
 - Opacity
- ❑ **RGB** or **RGBA** (*Red Green Blue Alpha*) gives you **greater control over the exact colors** in your web pages.
 - The value for each color— **Red**, **Green** and **Blue** —can range from 0 to 255.
 - The alpha value—which represents opacity—can be any value in the range 0.0 (fully **transparent**) through 1.0 (fully **opaque**).
 - `background: rgba(255, 0, 0, 0.5);`
- ❑ There are over **140 HTML color names**, whereas there are **16,777,216 different RGB colors** (256 x 256 x 256) and varying opacities of each.

Color...

- ❑ CSS3 also allows you to express color using **HSL** (hue, saturation, lightness) or **HSLA** (hue, saturation, lightness, alpha) values.
- ❑ The **hue** is a *color or shade expressed as a value* from 0 to 359 representing the degrees on a color wheel (a wheel is 360 degrees).
- ❑ The colors on the wheel progress in the **order of the colors of the rainbow**—red, orange, yellow, green, blue, indigo and violet.
 - The value for **red**, which is at the beginning of the wheel, is **0**.
 - **Green** hues have values around **120** and **blue** hues have values around **240**.
 - A hue value of 359, which is just left of 0 on the wheel, would result in a red hue.



- ❑ **Saturation**— the intensity of the hue—is expressed as a percentage, where 100% is **fully saturated** (the full color) and 0% is **gray**.
- ❑ **Lightness** —the intensity of light or luminance of the hue—is also expressed as a percentage.
 - A lightness of 50% is the **actual hue**.
 - If you decrease the amount of light to 0%, the color appears **completely dark** (black).
 - If you increase the amount of light to 100%, the color appears **completely light** (white).
- ❑ For example, if you wanted to use an hsla value to get the color red
 - `background: hsla (0, 100%, 50%, 0.5);` \equiv `background: rgba(255, 0, 0, 0.5);`

Gradients

❑ Gradients displays the combination of two or more colors

❑ Types of gradients

- Linear Gradients
- Radial Gradients

➤ Internet Explorer 9 and earlier versions do not support gradients.

❑ *Linear gradients* are a type of image that gradually transitions from one color to the next **horizontally**, **vertically** or **diagonally**.

▪ Example

```
div {  
  height: 100px;  
  background: blue; /* For browsers that do not support gradients */  
  background: linear-gradient(to top, blue , yellow);  
}
```

Bottom to Top



❑ Using Angles

- If you want **more control over the direction** of the gradient, you can define an **angle**, instead of the predefined directions (*to bottom, to top, to right, to left, to bottom right, etc.*)
- Syntax
`background: linear-gradient(angle, color-stop1, color-stop2);`

- ❑ The angle is specified as an angle between a **horizontal line** and the **gradient line**.

```
Div {  
  height: 100px;  
  background: blue; /* For browsers that do not support gradients */  
  background: linear-gradient(120deg, blue, yellow);  
}
```



❑ Using Multiple Color Stops

- with multiple color, the default order (from top to bottom)

```
background: linear-gradient(red 10%, green 55%, blue 80%);
```

❑ Using Transparency

- To add transparency, we use the `rgba()` function to define the color stops.
- The last parameter in the `rgba()` function can be a value from **0 to 1**, and it defines the transparency of the color: **0** indicates full **transparency**, **1** indicates **full color** (no transparency).
- EXAMPLE

```
background: linear-gradient(to right, rgba(0,0,255,0), rgba(0,0,255,1));
```



❑ Repeating a linear-gradient

- The `repeating-linear-gradient()` function is used to repeat linear gradients:

- ❑ **Radial gradients** are similar to linear gradients, but the color changes gradually from an inner point (the start) to an outer circle (the end)

- To create a radial gradient you must also define **at least two color stops**.

- Syntax

`background: radial-gradient (shape size at position, start-color, ..., last-color);`

➤ By default, shape is *ellipse*, size is *farthest-corner*, and position is *center*.

- Examples

`background: radial-gradient(circle, red, yellow, green);`

- ❑ **Use of Different Size Keywords**

- The size parameter defines the size of the gradient. It can take four values:

➤ *closest-side*, *farthest-side*, *closest-corner* or *farthest-corner*

Multiple Background Images

- ❑ CSS3 allows you to add multiple background images for an element, through the *background-image property*.
 - The different background images are **separated by commas**,
 - The images are **stacked on top of each other** (first image is closest to the viewer)
- ❑ A sample syntax of multi background images

```
#example {  
    background-image: img1.gif, img2.gif ;  
    background-position: right bottom, left top;  
    background-repeat: no-repeat, repeat;  
}
```

- ❑ A background shorthand property.

```
#example {  
    background: img1.gif right bottom no-repeat,  
               img2.gif left top repeat;  
}
```

Multiple Background Images..

❑ Example



background-image:

```
url(https://mdn.mozillademos.org/files/11305/firefox.png),  
url(https://mdn.mozillademos.org/files/11307/bubbles.png),  
linear-gradient(to right, rgba(30, 75, 115, 1), rgba(255, 255, 255, 0));
```

Multiple Background Images..

❑ CSS3 Background Properties

| Property | Description |
|--|---|
| <u>background</u> | A shorthand property for setting all the background properties in one declaration |
| <u>background-clip</u> | Specifies the painting area of the background |
| <u>background-image</u> | Specifies one or more background images for an element |
| <u>background-origin</u> | Specifies where the background image(s) is/are positioned |
| <u>background-size</u> | Specifies the size of the background image(s) |

Image Borders

- ❑ The CSS3 *border-image* property allows drawing an image on the borders of block elements.
 - This makes drawing complex looking widgets much simpler than it has been



- ❑ The border-image is used instead of the border styles given by the *border-style* properties.

```
#bitmap {  
    border: 30px solid transparent;  
    padding: 20px;  
    border-image: ImageName.png 30;  
}
```

```
border-image: url(imageName.png) 30 round/stretch;
```

Image Borders

❑ CSS3 Border Properties

| Property | Description |
|--|---|
| <u>border-image</u> | A shorthand property for setting all the border-image-* properties |
| <u>border-image-source</u> | Specifies the path to the image to be used as a border |
| <u>border-image-slice</u> | Specifies how to slice the border image |
| <u>border-image-width</u> | Specifies the widths of the border image |
| <u>border-image-outset</u> | Specifies the amount by which the border image area extends beyond the border box |
| <u>border-image-repeat</u> | Specifies whether the border image should be repeated, rounded or stretched |

CSS3 Multiple Columns ..

❑ Specify the Gap Between Columns

- Example: Specify a 40 pixels gap between the columns:

```
div
{
  column-gap:40px;
}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure

dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Nam liber tempor cum soluta nobis eleifend option congue

nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius.

CSS3 Multiple Columns ..

❑ Column Rules

- ❑ The column-rule property sets the width, style, and color of the rule between columns.

- Example: Specify the width, style and color of the rule between columns:

```
div
```

```
{
```

```
  column-rule:3px outset #ff00ff;
```

```
}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure

dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Nam liber tempor cum soluta nobis eleifend option congue

nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius.

CSS Pseudo-classes

CSS Pseudo-classes

❑ A pseudo-class is used to **define a special state of an element**.

❑ For example, it can be used to:

- Style an element when a user mouse over it
- Style visited and unvisited links differently
- Style an element when it gets focus

❑ The syntax of pseudo-classes:

```
selector:pseudo-class {  
    property:value;  
}
```

❑ **:hover** pseudoclass

- used to apply styles to an element when the mouse cursor is over it

❑ **display** property

- allows a programmer to decide if an element is displayed as a block element, inline element, or is not rendered at all (none)

CSS Pseudo-classes

Display Property

- ❑ Configures how an element is displayed
 - **display: none ;**
 - The element will not be displayed.
 - **display: block ;**
 - Displays an element as a block element (like <p>). It starts on a new line, and takes up the whole width.
 - The element is rendered as a block element --even if it is actually an inline element, such as a hyperlink.
 - **display: inline;**
 - Displays an element as an inline element (like). Any height and width properties will have no effect
 - The element will be rendered as an inline element – even if it is actually a block element – such as a .

CSS Pseudo-classes

Display Property

```
<!DOCTYPE html>
<html>
<head>
<style>
p {color: red;}

p.ex1 {display: none;}
p.ex2 {display: inline;}
p.ex3 {display: block;}
p.ex4 {display: inline-block;}
</style>
</head>
<body>
<h1>The display Property</h1>

<h2>display: none:</h2>
<div>
SWE 363 course. <p class="ex1">HELLO WORLD!</p> will be offered next semester.
</div>

<h2>display: inline:</h2>
<div>
SWE 363 course. <p class="ex2">HELLO WORLD!</p> will be offered next semester.
</div>

<h2>display: block:</h2>
<div>
SWE 363 course. <p class="ex3">HELLO WORLD!</p> will be offered next semester.
</div>

<h2>display: inline-block:</h2>
<div>
SWE 363 course. <p class="ex4">HELLO WORLD!</p> will be offered next semester.
</div>

</body>
</html>
```

The display Property

display: none:

SWE 363 course. will be offered next semester.

display: inline:

SWE 363 course. HELLO WORLD! will be offered next semester.

display: block:

SWE 363 course.

HELLO WORLD!

will be offered next semester.

display: inline-block:

SWE 363 course. HELLO WORLD! will be offered next semester.

CSS Pseudo-classes

Example 1

- ❑ An example of using the *:hover pseudo-class* on a `<div>` element:

```
<head>
  <style>
    div {
      background-color: green;
      color: white;
      padding: 25px;
      text-align: center;
    }
    div:hover {
      background-color: blue;
    }
  </style>
</head>
<body>
  <div>Mouse Over Me</div>

</body>
```



CSS Pseudo-classes

Example2

- ❑ **:hover** over a <div> element to show a <p> element (like a tooltip):

```
<head>
  <style>
    p {
      display: none;
      background-color: yellow;
      padding: 20px;
    }

    div:hover p {
      display: block;
    }
  </style>
</head>

<body>
  <div>Hover over me to show the p element
    <p>Tada! Here I am!</p>
  </div>
</body>
```

Hover over me to show the p element

Tada! Here I am!

CSS Pseudo-classes

Example3

- ❑ Show and hide a "dropdown" menu on mouse hover:

```
<head>
  <style>
    ul {background-color: yellow;}
    ul li ul {
      display: none;
    }
    ul li:hover ul {display: block;}
    ul li ul li:hover {background: red;}
  </style>
</head>

<body>
<div>
  <ul>
    <li>
      <a href="#">Dropdown Link</a>
      <ul>
        <li><a href="#">Link 1</a></li>
        <li><a href="#">Link 2</a></li>
        <li><a href="#">Link 3</a></li>
      </ul>
    </li>
  </ul>
</div>
</body>
```

• Dropdown Link

• Dropdown Link

◦ Link 1

◦ Link 2

◦ Link 3

• Dropdown Link

◦ Link 1

◦ Link 2

◦ Link 3

https://www.w3schools.com/cssref/sel_hover.asp

Free CSS drop down menu:

<http://cssmenu maker.com/css-drop-down-menu>

More CSS Pseudo Classes

| Selector | Example | Example description |
|--------------------------------|-----------------|---|
| :active | a:active | Selects the active link |
| :checked | input:checked | Selects every checked <input> element |
| :disabled | input:disabled | Selects every disabled <input> element |
| :empty | p:empty | Selects every <p> element that has no children |
| :enabled | input:enabled | Selects every enabled <input> element |
| :first-child | p:first-child | Selects every <p> elements that is the first child of its parent |
| :first-of-type | p:first-of-type | Selects every <p> element that is the first <p> element of its parent |
| :focus | input:focus | Selects the <input> element that has focus |
| :hover | a:hover | Selects links on mouse over |
| | | |

Positioning Elements

https://www.w3schools.com/css/css_positioning.asp

Positioning Elements

- ❑ The **position property** specifies the type of positioning method used for an element.
- ❑ There are **four** different position values:
 - static
 - relative
 - fixed
 - absolute
 - sticky
- ❑ Elements are then positioned using the **top**, **bottom**, **left**, and **right** properties.
 - However, these properties will not work unless the position property is set first.
 - They also work differently depending on the position value.

<https://developer.mozilla.org/en-US/docs/Web/CSS/position>

https://www.w3schools.com/cSS/css_positioning.asp

position: *static*;

- ❑ HTML elements are positioned static by default.
- ❑ Static positioned elements are not affected by the top, bottom, left, and right properties.
- ❑ An element with *position: static*; is not positioned in any special way; it is always positioned according to the normal flow of the page:

```
<head>
  <style>
    div.static {
      position: static;
      border: 3px solid #73AD21;
    }
  </style>
</head>
<body>
  <h2>position: static;</h2>
  <p>is not positioned in any special way:</p>
  <div class="static">
    This div element has position: static;
  </div>
</body>
```

position: static;

is not positioned in any special way:

This div element has position: static;

position: *relative*;

- ❑ An element with `position: relative;` is positioned **relative to its normal position**.
- ❑ Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position.

```
<head>
  <style>
    div.relative {
      position: relative;
      left: 30px;
      border: 3px solid #73AD21;
    }
  </style>
</head>
<body>
  <h2>position: relative;</h2>
  <p>is positioned relative to its normal position:</p>
  <div class="relative">
    This div element has position: relative;
  </div>
</body>
```

position: relative;

is positioned relative to its normal position:

This div element has position: relative;

position: *fixed*;

- ❑ An element with position: **fixed**; is positioned relative to the viewport,
 - it **always stays in the same place even if the page is scrolled**.
 - The top, right, bottom, and left properties are used to position the element.

```
<head>
  <style>
    div.fixed {
      position: fixed;
      bottom: 0;
      right: 0;
      width: 300px;
      border: 3px solid #73AD21;
    }
  </style>
</head>
<body>
  <h2>position: fixed;</h2>
  <p>is positioned relative to the viewport.</p>
  <div class="fixed">
    This div element has position: fixed;
  </div>
</body>
```

position: fixed;

is positioned relative to the viewport,

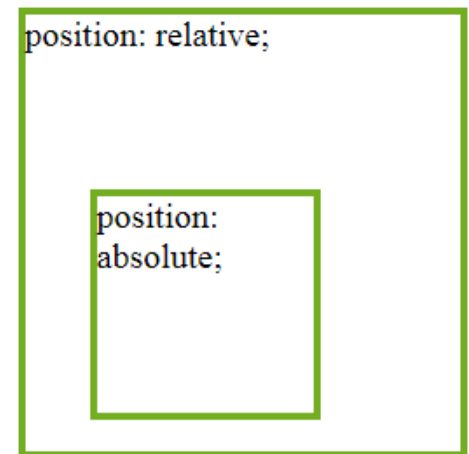
This div element has position: fixed;

position: *absolute*;

- ❑ An element with *position: absolute*; is positioned relative to the nearest positioned ancestor
 - instead of positioned relative to the viewport, like fixed.
- ❑ if an absolute positioned element **has no positioned ancestors**, it uses the document body, and moves along with page scrolling.

```
<head>
  <style>
    div.relative {
      position: relative;
      left : 30px; width: 200px; height: 200px;
      border: 3px solid #73AD21; }
    div.absolute {
      position: absolute;
      top: 80px; left: 30px; width: 100px;
      height: 100px; border: 3px solid #73AD21; }
  </style>
</head>
<body>
  <p> relative to the nearest positioned ancestor</p>
  <div class="relative">position: relative;
    <div class="absolute"> position: absolute;</div>
  </div>
</body>
```

relative to the nearest positioned ancestor



position: *sticky*;

- ❑ An element with *position: sticky*; is positioned based on the user's scroll position.
- ❑ In this example, the sticky element sticks to the top of the page (top: 0), when you reach its scroll position.

```
<head>
  <style>
    div.sticky {
      position: -webkit-sticky;
      position: sticky;
      top: 0;
      padding: 5px;
      background-color: #cae8ca;
      border: 2px solid #4CAF50;
    }
  </style>
</head>
<body>
  <div class="sticky">I am sticky!</div>
```

```
<p>Try to <b>scroll</b> inside this frame to
understand how sticky positioning works.</p>
```

I am sticky!

Try to **scroll** inside this frame to understand how sticky positioning works.

Note: IE/Edge 15 and earlier versions do not support sticky position.

In this example, the sticky element sticks to the top of the page (top: 0), when you reach its scroll position.

Overlapping Elements

- ❑ When elements are positioned, **they can overlap other elements**.
- ❑ The *z-index property* specifies the stack order of an element (which element should be placed in front of, or behind, the others)
- ❑ An element can have a **positive** or **negative** stack order:
- ❑ Example : the image has a z-index of -1, so it is placed behind the text.

```
<head>
  <style>
    img {
      position: absolute;
      left: 0px;
      top: 0px;
      z-index: -1;
    }
  </style>
</head>
<body>
  <h1>This is a heading</h1>
  
  <p> The image is placed behind the text.</p>
</body>
```



<https://developer.mozilla.org/en-US/docs/Web/CSS/z-index>

https://www.w3schools.com/cssref/tryit.asp?filename=trycss_zindex

Floating Elements

Source of this part:

“Fundamentals of Web Development” Book by Randy Connolly and Ricardo Hoar, 2015


Floating Elements

- ❑ **Floating** allows you to **move an element out of its position** in the normal flow via the CSS *float* property
 - An element can be floated to the **left** or to the **right** .
 - It is moved all the way to the far left or far right of its containing block and the rest of **the content is “reflowed” around the floated element**

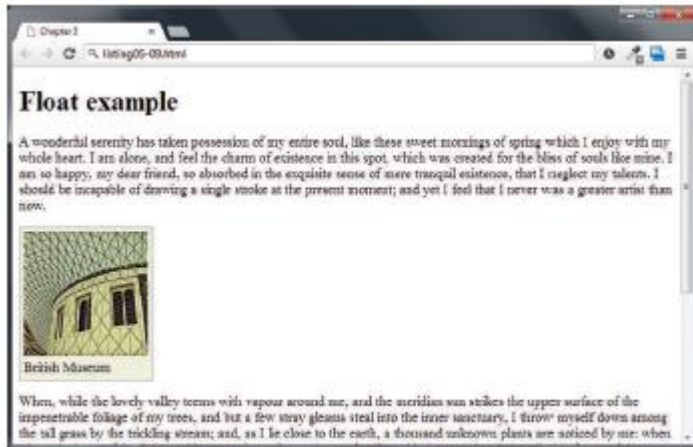
```
<head>
  <style>
    img {
      float: right;
    }
  </style>
</head>

<body>
  <p>
    
</body>
```

This is some text. This is some text. This is some
text. This is some text. This is some text. This is
some text. This is some text. This is some text. This
is some text. This is some text. This is some text.
This is some text. This is some text. This is some
text. This is some text. This is some text. This is
some text. This is some text. This is some text. This
is some text. This is some text. This is some text.



Floating Elements Example



```
<h1>Float example</h1>
<p>A wonderful serenity has taken ...</p>
<figure>
  
  <figcaption>British Museum</figcaption>
</figure>
<p>When, while the lovely valley ...</p>
```

```
figure {
  border: 1pt solid #A8A8A8;
  background-color: #EDEDDE;
  margin: 0;
  padding: 5px;
  width: 150px;
}
```

Notice that a floated block-level element must have a width specified.



```
figure {
  ...
  width: 150px;
  float: left;
}
```



```
figure {
  ...
  width: 150px;
  float: right;
  margin: 10px;
}
```



```

<article>
  <h1>Float example</h1>
  <p>A wonderful serenity has taken possession of ... </p>

  <figure>
    
    <figcaption>British Museum</figcaption>
  </figure>

  <p>When, while the lovely valley teems with ...</p>

  <p>O my friend -- but it is too much for my ...</p>
</article>

```

Floating Elements

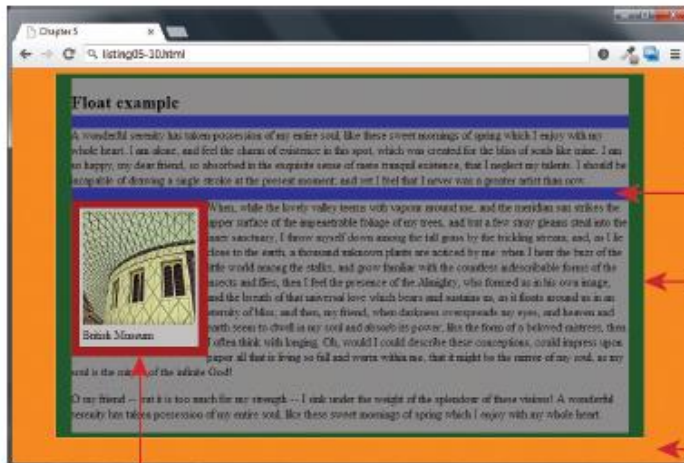
Floating within a Container



```

article {
  background-color: #898989;
  margin: 5px 50px;
  padding: 5px 20px;
}
p { margin: 16px 0; }
figure {
  border: 1pt solid #262626;
  background-color: #c1c1c1;
  padding: 5px;
  width: 150px;
  float: left;
  margin: 10px;
}

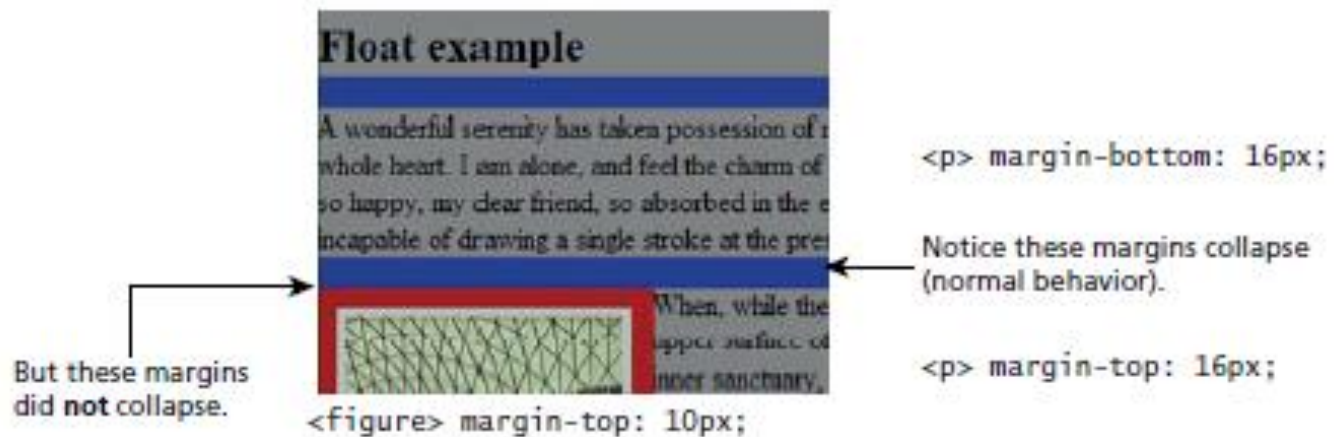
```



Floating Elements

Floating within a Container

- ❑ Margins do not collapse on floated block-level elements.

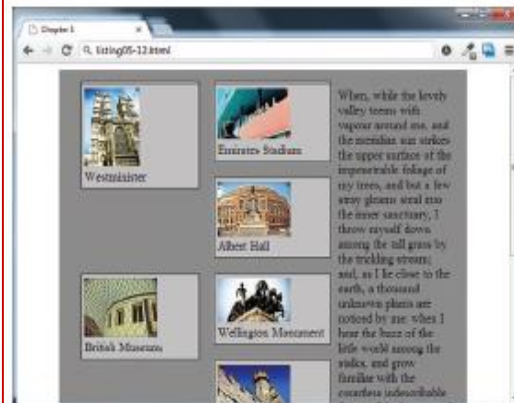


Floating Elements

Floating Multiple Items Side by Side

- ❑ One of the more common usages of floats is to place multiple items side by side on the same line.
 - each floated item in the container will be nestled up beside the previously floated item.

As the window resizes, the content in the containing block (the `<article>` element), will try to fill the space that is available to the right of the floated elements.



```
<article>
  <figure>
    
    <figcaption>Westminster</figcaption>
  </figure>
  <figure>
    
    <figcaption>Emirates Stadium</figcaption>
  </figure>
  <figure>
    
    <figcaption>Albert Hall</figcaption>
  </figure>
  <figure>
    
    <figcaption>British Museum</figcaption>
  </figure>
  <figure>
    
    <figcaption>Wellington Monument</figcaption>
  </figure>
  <figure>
    
    <figcaption>Lewes Castle</figcaption>
  </figure>
  <p>When, while the lovely valley teems ..
</p>
</article>
```

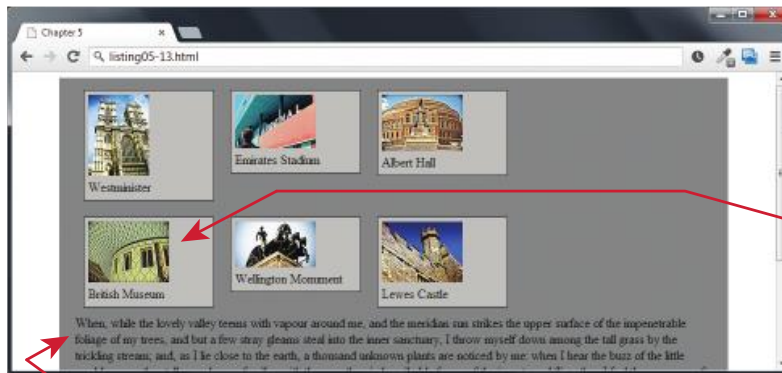
```
figure {
  ...
  width: 150px;
  float: left;
}
```

Floating Elements

Floating Multiple Items Side by Side

- ❑ you can stop elements from flowing around a floated element by using the **clear** property

```
.first { clear: left; }
```



```
<article>
  <figure>
    
    <figcaption>Westminister</figcaption>
  </figure>
  <figure>
    
    <figcaption>Emirates Stadium</figcaption>
  </figure>
  <figure>
    
    <figcaption>Albert Hall</figcaption>
  </figure>
  <figure class="first">
    
    <figcaption>British Museum</figcaption>
  </figure>
  <figure>
    
    <figcaption>Wellington Monument</figcaption>
  </figure>
  <figure>
    
    <figcaption>Lewes Castle</figcaption>
  </figure>
  <p class="first">When, while the lovely ...
</article>
```

Floating Elements

Clear Property

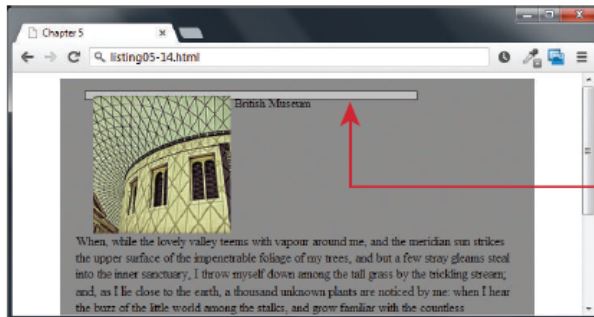
| Value | Description |
|-------|--|
| left | The left-hand edge of the element cannot be adjacent to another element |
| right | The right-hand edge of the element cannot be adjacent to another element |
| both | Both the left-hand and right-hand edges of the element cannot be adjacent to another element |
| none | The element can be adjacent to other elements |

Floating Elements

Containing Floats

- ❑ Another problem that can occur with floats is when an element is floated within a containing block that contains only floated content.
 - In such a case, the containing block essentially disappears

```
<article>
  <figure>
    
    <figcaption>British Museum</figcaption>
  </figure>
  <p class="first">When, while the lovely valley ...
</article>
```



Notice that the `<figure>` element's content area has shrunk down to zero (it now just has padding space and borders).

```
figure img {
  width: 170px;
  float: left;
  margin: 0 5px;
}
figure figcaption {
  width: 100px;
  float: left;
}
figure {
  border: 1pt solid #262626;
  background-color: #c1c1c1;
  padding: 5px;
  width: 400px;
  margin: 10px;
}
.first { clear: left; }
```


Floating Elements

Containing Floats

- ❑ One solution would be to float the container as well, but depending on the layout this might not be possible.
 - A better solution would be to use the *overflow property*



Setting the overflow property to auto solves the problem.

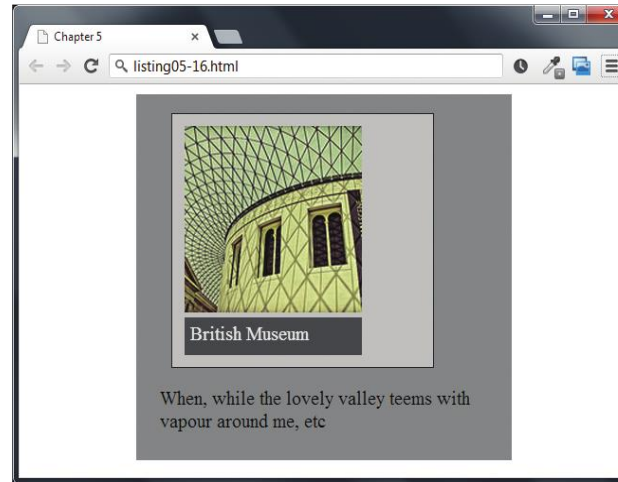
```
figure img {
    width: 170px;
    float: left;
    margin: 0 5px;
}
figure figcaption {
    width: 100px;
    float: left;
}
figure {
    border: 1pt solid #262626;
    background-color: #c1c1c1;
    padding: 5px;
    width: 400px;
    margin: 10px;
    overflow: auto;
}
```

Floating Elements

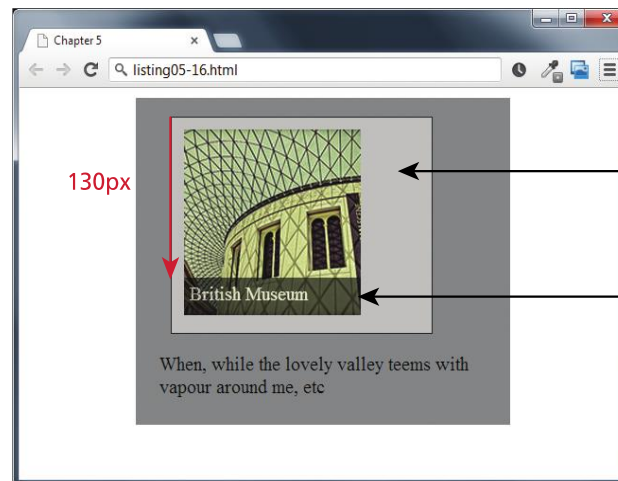
Overlaying and Hiding Element

- ❑ One of the more common design tasks with CSS is to **place two elements on top of each other**, or to **selectively hide and display elements**

- In such a case, relative positioning is used to create the **positioning context** for a subsequent absolute positioning move



```
figure {  
  border: 1pt solid #262626;  
  background-color: #c1c1c1;  
  padding: 10px;  
  width: 200px;  
  margin: 10px;  
}  
  
figcaption {  
  background-color: black;  
  color: white;  
  opacity: 0.6;  
  width: 140px;  
  height: 20px;  
  padding: 5px;  
}
```



```
figure {  
  ...  
  position: relative;  
}  
  
figcaption {  
  ...  
  position: absolute;  
  top: 130px;  
  left: 10px;  
}
```

This creates the positioning context.

This does the actual move.

CSS Editors and Validators

❑ CSS Editors

- Topstyle Pro from Bradbury Software
<http://www.bradsoft.com/topstyle/index.asp>

❑ CSS Validation:

- This tool can help you make sure that your code is correct and will work on CSS3-compliant browsers.

<http://jigsaw.w3.org/css-validator/>

