Web Technologies Manual (Frontend)

Dan Austin K. Higwit
dan@qwento.com

November 25, 2018

Contents

Abstract

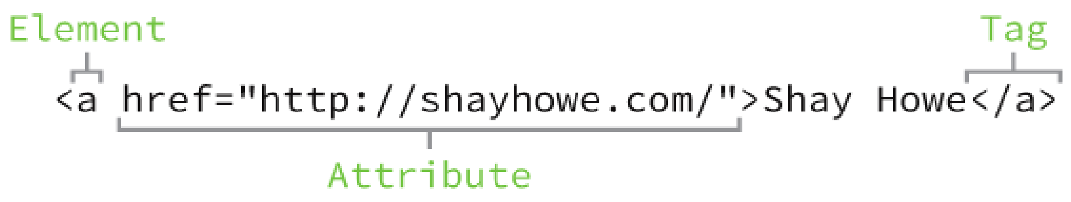
This is a compilation of notes and code snippets for html, css, sass and other related web technologies. html css sass Manual.

1 Basic Structure

HTML, HyperText Markup Language, gives content structure and meaning by defining that content as, for example, headings, paragraphs, or images. CSS, or Cascading Style Sheets, is a presentation language created to style the appearance of content—using, for example, fonts or colors.

1.1 HTML elements, tags, atttributes

The parts of an html consists of the following parts:



1.2 HTML document template

All HTML documents have a required structure that includes the following declaration and elements: <!DOCTYPE html>, <html>, <head>, and <body>.

Subsequent paragraphs, however, are indented.

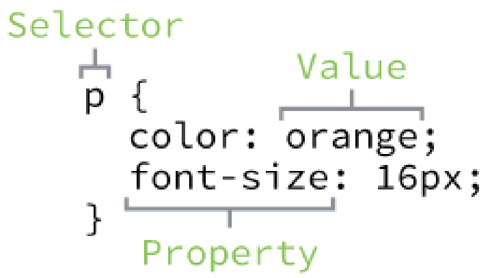
```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Hello World</title>
  </head>
  <body>
    <h1>Hello World</h1>
    <p>This is a web page.</p>
  </body>
</html>
```

Self-Closing Elements Some elements simply receive their content or behavior from attributes within a single tag. The <meta> element is one of these elements. The content of the previous <meta> element is assigned with the use of the charset attribute and value. Other common selfclosing elements include

```html
<br>, <img>, <meta>, <wbr>, <embed>, <input>, <param>
<hr>, <link>, <source>
```

2 CSS Terms

In addition to HTML terms, there are a few common CSS terms you will want to familiarize yourself with. These terms include selectors, properties, and values. As with the HTML terminology, the more you work with CSS, the more these terms will become second nature.



2.1 Selectors

A selector designates exactly which element or elements within our HTML to target and apply styles (such as color, size, and position) to. Selectors generally target an attribute value, such as an id or class value, or target the type of element, such as <h1> or <p>.

```css
    p {...}
```

2.2 Properties

So far we've selected an element with a selector and determined what style we'd like to apply with a property. Now we can determine the behavior of that property with a value. Values can be identified as the text between the colon, :, and semicolon, ;. Here we are selecting all <p> elements and setting the value of the color property to be orange and the value of the font-size property to be 16 pixels.

```css
p {
  color:..;
  font-size:..;
}
```

2.3 Values

Values can be identified as the text between the colon, :, and semicolon, ;. Here we are selecting all <p> elements and setting the value of the color property to be orange and the value of the font-size property to be 16 pixels.

```css
p {
  color: orange;
  font-size: 16px;
}
```

2.4 Selectors

Values can be identified as the text between the colon, :, and semicolon, ;. Here we are selecting all <p> elements and setting the value of the color property to be orange and the value of the font-size property to be 16 pixels.

Type Selectors target elements by their element type. For example, should we wish to target all division elements, <div>, we would use a type selector of div.

```
div {...}
<div>...</div>
```

Class Selectors allow us to select an element based on the element's class attribute value.

```
.awesome {...}
<div class="awesome">...</div>
<p class="awesome">...</p>
```

ID Selectors are even more precise than class selectors, as they target only one unique element at a time.

```
#shayhowe {...}
<div id="shayhowe">...</div>
```

2.5 Referencing CSS

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

3 HTML in depth

3.1 Semantic Elements

Semantic code describes the value of content on a page, regardless of the style or appearance of that content. There are several benefits to using semantic elements, including enabling computers, screen readers, search engines, and other devices to adequately read and understand the content on a web page.

Identifying Divisions Spans Divisions, or <div>s, and <span>s are HTML elements that act as containers solely for styling purposes. As generic containers, they do not come with any overarching meaning or semantic value. Paragraphs are semantic in that content wrapped within a <p> element is known and understood as a paragraph. <div>s and <span>s do not hold any such meaning and are simply containers.

Block vs. Inline Elements Most elements are either block- or inline-level elements. What's the difference? Block-level elements begin on a new line, stacking one on top of the other, and occupy any available width. Block-level elements may be nested inside one another and may wrap inline-level elements. We'll most commonly see block-level elements used for larger pieces of content, such as paragraphs. Inline-level elements do not begin on a new line. They fall into the normal flow of a document, lining up one after the other, and only maintain the width of their content. Inline-level elements may be nested inside one another; however, they cannot wrap block-level elements. We'll usually see inline-level elements with smaller pieces of content, such as a few words.

Divs and Spans divs act as block elements spans act inline

```
<div>...</div> //are default block elements
<span>...</span> //are default inline elements
```

Headings Headings are block-level elements, and they come in six different rankings, <h1> through <h6>.

```
<h1>Heading Level 1</h1>
<h2>Heading Level 2</h2>
<h3>Heading Level 3</h3>
<h4>Heading Level 4</h4>
<h5>Heading Level 5</h5>
<h6>Heading Level 6</h6>
```

Paragraphs Paragraphs are defined using the <p> block-level element.

```
<p>Steve Jobs was a co-founder and longtime chief executive officer at Apple. On June 12, 2005, Steve gave the
    commencement address at Stanford University.</p>

<p>In his address Steve urged graduates to follow their dreams and, despite any setbacks, to never give up
   ndash;advice which he sincerely took to heart.</p>
```

Bold, Strong Here are the two HTML options for creating bold text in action:

```
<!- Strong importance ->
<p><strong>Caution:</strong> Falling rocks.</p>

<!- Stylistically offset ->
<p>This recipe calls for <b>bacon</b> .</p>
```

Italics, Emphasis Here are the two HTML options for creating bold text in action:

```
<!- Stressed emphasis ->
<p>I <em>love</em> Chicago!</p>

<!- Alternative voice or tone ->
<p>The name <i>Shay</i> means a gift.</p>
```

3.2 HTML5 Structure

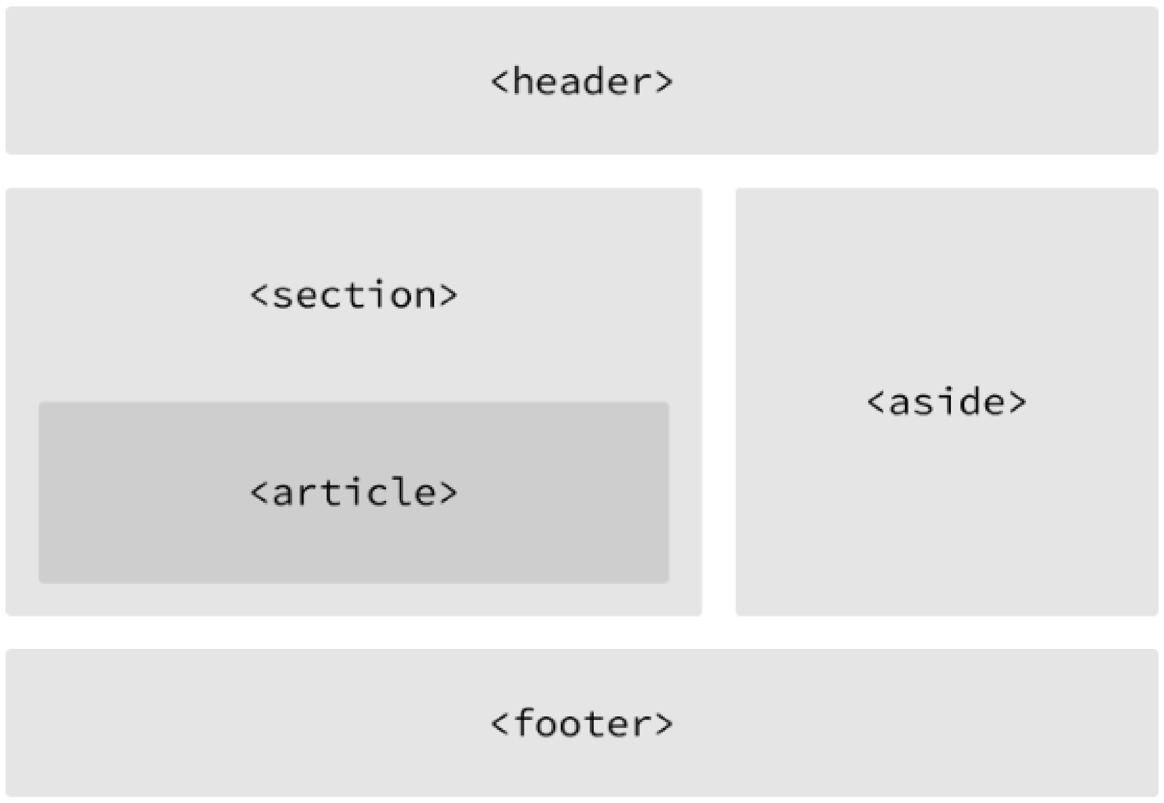
For the longest time the structure of a web page was built using divisions. The problem was that divisions provide no semantic value, and it was fairly difficult to determine the intention of these divisions. Fortunately HTML5 introduced new structurally based elements, including the <header>, <nav>, <article>, <section>, <aside>, and <footer> elements.

<header> element, like it sounds, is used to identify the top of a page, article, section, or other segment of a page. In general, the <header> element may include a heading, introductory text, and even navigation.

```
<header>.</header>
```

<nav> element identifies a section of major navigational links on a page. The <nav> element should be reserved for primary navigation sections only, such as global navigation, a table of contents, previous/next links, or other noteworthy groups of navigational links.

```
<nav>.</nav>
```



The <article> element is used to identify a section of independent, self-contained content that may be independently distributed or reused. We'll often use the <article> element to mark up blog posts, newspaper articles, user-submitted content, and the like.

```
<article>.</article>
```

The <section> element is used to identify a thematic grouping of content, which generally, but not always, includes a heading. The grouping of content within the <section> element may be generic in nature, but it's useful to identify all of the content as related.

```
<section>.</section>
```

The <aside> element holds content, such as sidebars, inserts, or brief explanations, that is tangentially related to the content surrounding it. When used within an <article> element, for example, the <aside> element may identify content related to the author of the article.

```
<aside>.</aside>
```

The <footer> The <footer> element identifies the closing or end of a page, article, section, or other segment of a page. Generally the <footer> element is found at the bottom of its parent. Content within the <footer> element should be relative information and should not diverge from the document or section it is included within.

```
        <footer>.</footer>
```

Creating Hyperlinks Along with text, one of the core components of the Internet is the hyperlink, which provides the ability to link from one web page or resource to another. Hyperlinks are established using the anchor, <a>, inline-level element. In order to create a link from one page (or resource) to another, the href attribute, known as a hyperlink reference, is required. The href attribute value identifies the destination of the link.

```
<a href="http://shayhowe.com">Shay</a>
```

Wrapping Block-level elements with anchors By nature the anchor element, <a>, is an inline element, and, according to web standards, inline-level elements may not wrap block-level elements. With the introduction of HTML5, however, anchor elements specifically have permission to wrap either block-, inline-, or any other level elements. This is a break from the standard convention, but it's permissible in order to enable entire blocks of content on a page to become links.

3.3 Relative and Absolute Paths

The two most common types of links are links to other pages of the same website and links to other websites. These links are identified by their href attribute values, also known as their paths.

Should the page being linked to reside within a different directory, or folder, the href attribute value needs to reflect this as well. Say the about.html page resides within the pages directory; the relative path would then be pages/about.html.

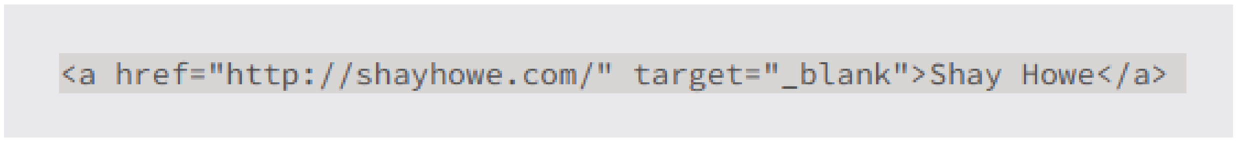
Linking to other websites outside of the current one requires an absolute path, where the href attribute value must include the full domain. A link to Google would need the href attribute value of http://google.com, starting with http and including the domain, .com in this case.

```
<!- Relative Path ->
<a href="about.html">About</a>

<!- Absolute Path ->
<a href="http://www.google.com/">Google</a>
```

Linking to an Email Address To create an email link, the href attribute value needs to start with mailto: followed by the email address to which the email should be sent. To create an email link to shay@awesome.com, for example, the href attribute value would be mailto:shay@awesome.com.

```
<a href="mailto:shay@awesome.com?subject=Reaching%20Out&body=How%20are%20you">Email Me</a>
```

Opening Links in a New Window To trigger the action of opening a link in a new window. The target attribute determines exactly where the link will be displayed,



```
<a href="http://shayhowe.com/" target="_blank">Shay Howe</a>
```

Linking to Parts of the Same Page Periodically we'll see hyperlinks that link to part of the same page the link appears on. A common example of these same-page links are "Back to top" links that return a user to the top of a page.

```
<body id="top">
 ...
  <a href="#top">Back to top</a>
 ...
</body>
```

4 CSS in depth

4.1 The Cascade

CSS is read from top to bottom, styles declared at a later section usually override pre-existing styles of the same importance

```
        \\background ends up as green
        p {
          background: orange;
          font-size: 24px;
        }
        p {
          background: green;
        }
```

4.2 Calculating Specificity

The type selector has the lowest specificity weight and holds a point value of 0-0-1. The class selector has a medium specificity weight and holds a point value of 0-1-0. Lastly, the ID selector has a high specificity weight and holds a point value of 1-0-0.

id > class > type This is the heirarchy of priority, with ID being the most specific

4.3 Combining Selectors

When selectors are combined they should be read from right to left. The selector farthest to the right, directly before the opening curly bracket, is known as the key selector. The key selector identifies exactly which element the styles will be applied to. Any selector to the left of the key selector will serve as a prequalifier.

```
//HTML
<div class="hotdog">
  <p>.</p>
  <p>.</p>
  <p class="mustard">.</p>
</div>

//CSS
.hotdog p {
  background: brown;
}
.hotdog p.mustard {
  background: yellow;
}
```

Because the new class selector, mustard, falls all the way to the right of the combined selector, it is the key selector, and all of the individual selectors coming before it are now prequalifiers.

Specificity Within Combined Selectors Looking at our combined selectors from before, the first selector, .hotdog p, had both a class selector and a type selector. Knowing that the point value of a class selector is 0-1-0 and the point value of a type selector is 0-0-1, the total combined point value would be 0-1-1, found by adding up each kind of selector. The second selector, .hotdog p.mustard, had two class selectors and one type selector. Combined, the selector has a specificity point value of 0-2-1. The 0 in the first column is for zero ID selectors, the 2 in the second column is for two class selectors, and the 1 in the last column is for one type selector.Comparing the two selectors, the second selector, with its two classes, has a noticeably higher specificity weight and point value. As such it will take precedence within the cascade. If we were to flip the order of these selectors within our style sheet, placing the higher-weighted selector above the lower-weighted selector as shown here, the appearance of their styles would not be affected due to each selector's specificity weight.

4.4 Layering Styles with Multiple Classes

One way to keep the specificity weights of our selectors low is to be as modular as possible, sharing similar styles from element to element. And one way to be as modular as possible is to layer on different styles by using multiple classes.

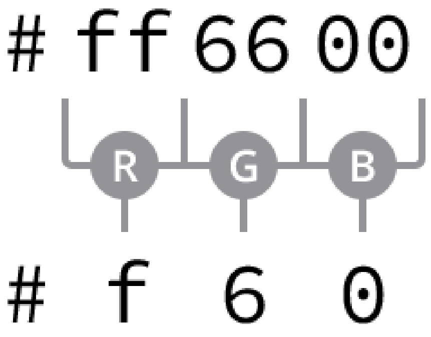
```
//html
<a class="btn btn-danger">.</a>
<a class="btn btn-success">.</a>

//css
.btn {
  font-size: 16px;
}
.btn-danger {
  background: red;
}
.btn-success {
  background: green;
}
```

4.5 Common CSS property values

Colors All color values within CSS are defined on an sRGB (or standard red, green, and blue) color space. Colors within this space are formed by mixing red, green, and blue color channels together.

```
.testColors{
  //range is from (0,255) for rgb
  //alpha is from (0,1)
  //complete black rgba(r,g,b,a)
  background: rgba(0,0,0,0)
}
```

Hex Colors 0 equals black and f equals white. Six-character hexadecimal values may be written as three-character hexadecimal values when the red, green, and blue color channels each contain a repeating character



```
.task {
  background: #800000;
}
.count {
  background: #ff0;
}
```

4.6 HSL HSLa Colors

HSL color values are stated using the hsl() function, which stands for hue, saturation, and lightness. Within the parentheses, the function accepts three comma-separated values, much like rgb(). The second and third values, the saturation and lightness, are percentage values from 0 to 100%. The saturation value identifies how saturated with color the hue is, with 0 being grayscale and 100% being fully saturated. The lightness identifies how dark or light the hue value is, with 0 being completely black and 100% being completely white. Returning to our shade of orange, as an HSL color value it would be written as hsl(24, 100%, 50%).

```
.task {
  background: hsl(0, 100%, 25%);
}
.count {
  background: hsl(60, 100%, 50%);
}

.task {
  background: hsla(0, 100%, 25%, .25);
}
.count {
  background: hsla(60, 100%, 50%, 1);
}
```

4.7 Lengths

Pixels The pixel is equal to 1/96th of an inch; thus there are 96 pixels in an inch. The exact measurement of a pixel, however, may vary slightly between high-density and low-density viewing devices.

```
p {
  font-size: 14px;
}
```

Percentages Percentages, represented by the % unit notation, are one of the most popular relative values. Percentage lengths are defined in relation to the length of another object. For example, to set the width of an element to 50%, we have to know the width of its parent element, the element it is nested within, and then identify 50% of the parent element's width.

```
        .col {
          width: 50%;
        }
```

Em A single em unit is equivalent to an element's font size. So, for example, if an element has a font size of 14 pixels and a width set to 5em, the width would equal 70 pixels (14 pixels multiplied by 5).

```
        .banner {
          font-size: 14px;
          width: 5em;
        }
```

5 Box Model

Exactly how elements are displayed—as block-level elements, inline elements, or something else—is determined by the display property. Every element has a default display property value; however, as with all other property values, that value may be overwritten. There are quite a few values for the display property, but the most common are block, inline, inline-block, and none.

```
      p {
        display: block;
      }

      p {
        display: inline;
      }
```

inline-block, Using this value will allow an element to behave as a block-level element, accepting all box model properties

```
      p {
        display: inline-block;
      }
```

5.1 What is a box?

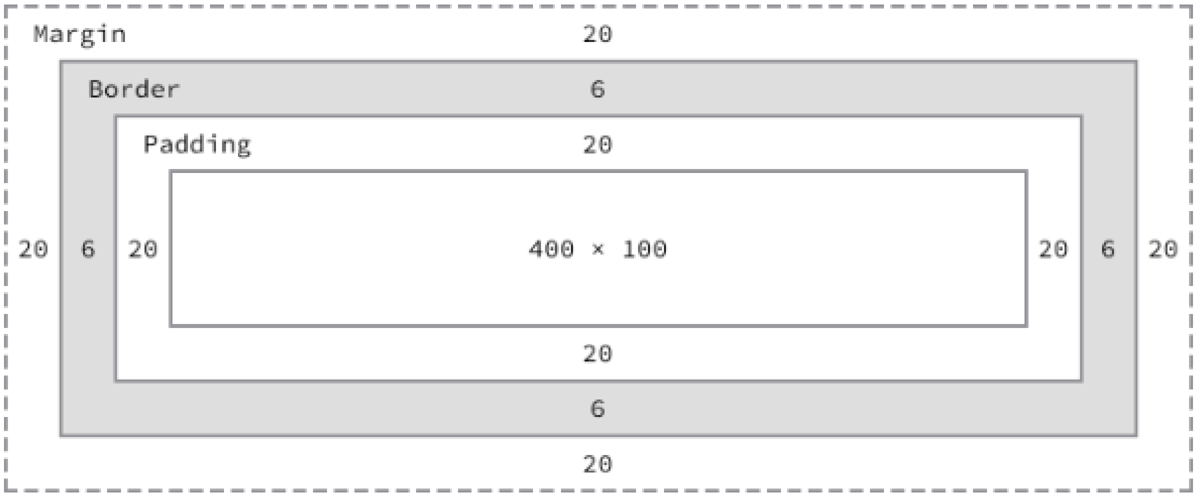
According to the box model concept, every element on a page is a rectangular box and may have width, height, padding, borders, and margins.

Properties Each part of the box model corresponds to a CSS property: width, height, padding, border, and margin.

```
        div {
          border: 6px solid #949599;
          height: 100px;
          margin: 20px;
          padding: 20px;
          width: 400px;
        }
```

Total Width/Height can be calculted through the following formula

```
        margin-right + border-right + padding-right + width + padding-left + border-left + margin-left

        margin-top + border-top + padding-top + height + padding-bottom + border-bottom + margin-bottom
```



5.2 Margin Padding Declarations

To set one value for the top and bottom and another value for the left and right sides of an element, specify two values: top and bottom first, then left and right. Here we are placing margins of 10 pixels on the top and bottom of a <div> and margins of 20 pixels on the left and right:

```
\\if 2 values top-bottom, left-right
div {
  margin: 10px 20px;
}
```

4-values specify those values in the order of top, right, bottom, and left, moving clockwise

```
div {
  margin: 10px 20px 0 15px;
}
```

5.3 Borders

Here is the code for a 6-pixel-wide, solid, gray border that wraps around all four sides of a <div>:

```
div {
  border: 6px solid #949599;
}
```

Individual Border Sides As with the margin and padding properties, borders can be placed on one side of an element at a time if we'd like. Doing so requires new properties: border-top, border-right, border-bottom, and border-left.

```
div {
  border-bottom: 6px solid #949599;
}

div {
  border-bottom-width: 12px;
}

div {
  border-bottom-width: 12px;
}
```

5.4 Box Sizing

Content Box The content-box value is the default value, leaving the box model as an additive design. If we don't use the box-sizing property, this will be the default value for all elements.

```
        div {
        -webkit-box-sizing: content-box;
           -moz-box-sizing: content-box;
                box-sizing: content-box;
        }
```

Padding Box When using the padding-box value, if an element has a width of 400 pixels and a padding of 20 pixels around every side, the actual width will remain 400 pixels.

```
        div {
          box-sizing: padding-box;
        }
```

Border Box Lastly, the border-box value alters the box model so that any border or padding property values are included within the width and height of an element.

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

6 Positioning

There are a few different types of positioning within CSS, and each has its own application.

6.1 Floats

the float property allows us to take an element, remove it from the normal flow of a page, and position it to the left or right of its parent element. All other elements on the page will then flow around the floated element. An <img> element floated to the side of a few paragraphs of text, for example, will allow the paragraphs to wrap around the image as necessary.

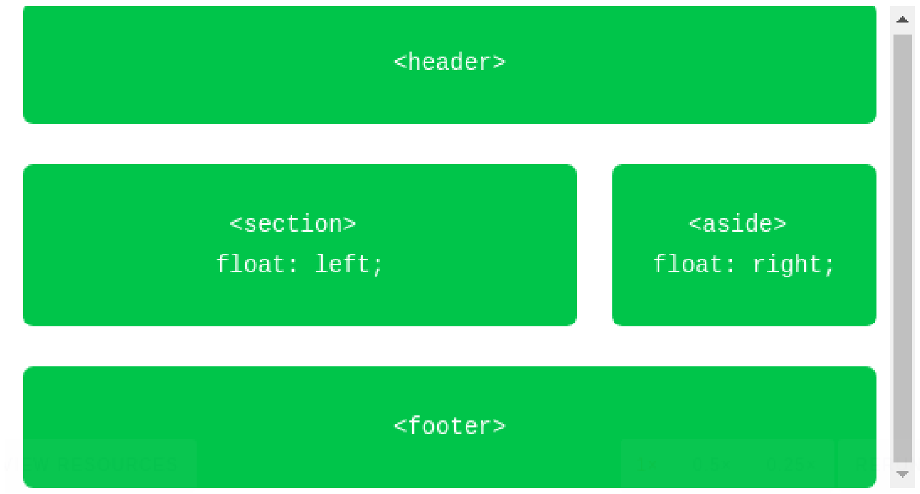
```
img {
  float: left;
}
```

```
section {
  float: left;
  margin: 0 1.5%;
  width: 63%;
}
aside {
  float: right;
  margin: 0 1.5%;
  width: 30%;
}
```

Header, Footer Act as block elements by setting section to float left, and aside to float right, with enough total width to fit into the page, they appear side by side

6.2 Float Clearfix

```
\\HTML
<header>.</header>
<div class="group">
  <section>.</section>
  <aside>.</aside>
</div>
<footer>.</footer>

\\CSS
```



```
.group:before,
.group:after {
  content: "";
  display: table;
}
.group:after {
  clear: both;
}
.group {
  clear: both;
  *zoom: 1;
}
section {
  float: left;
  margin: 0 1.5%;
  width: 63%;
}
aside {
  float: right;
  margin: 0 1.5%;
  width: 30%;
}
```

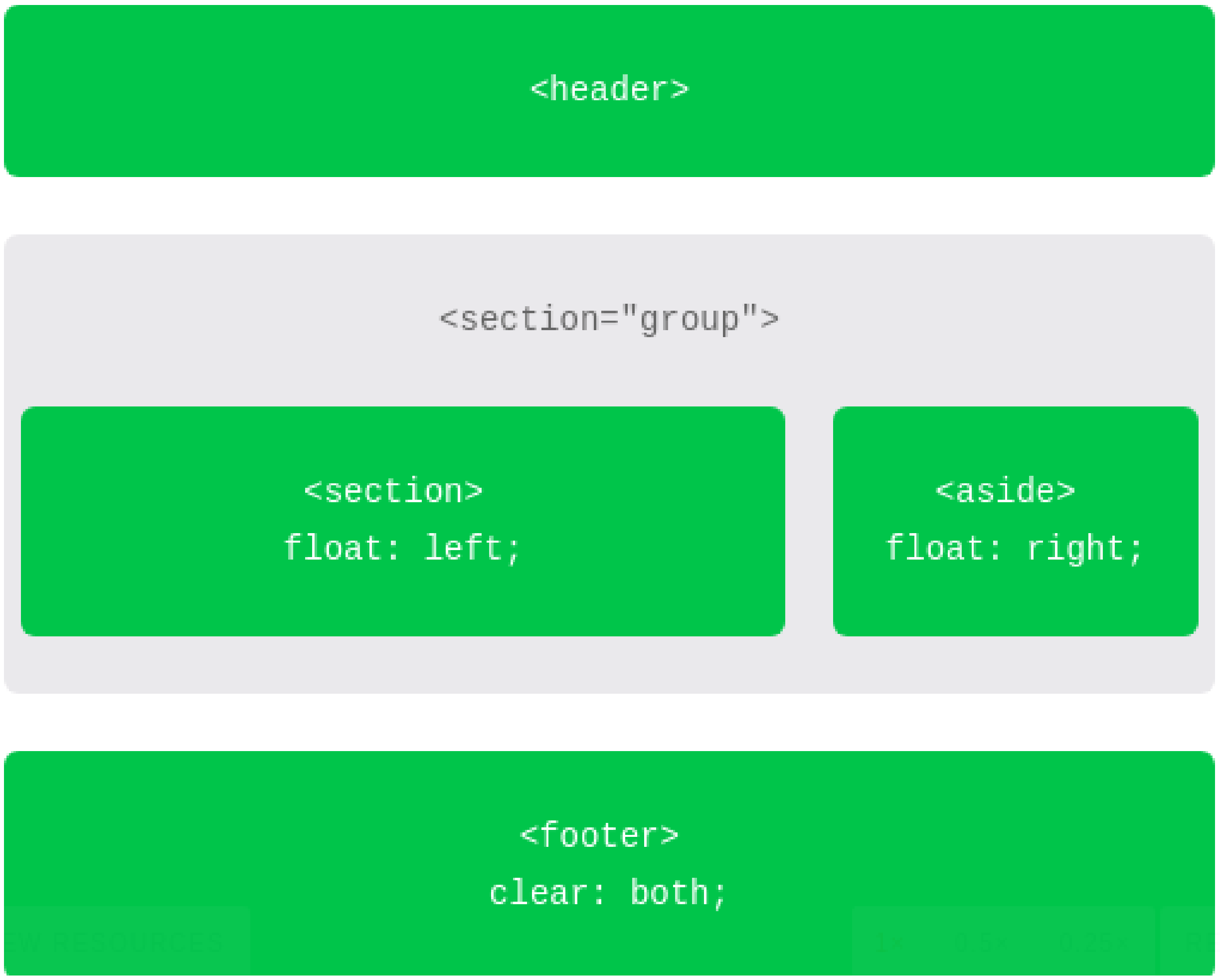
the CSS is clearing any floated elements within the element with the class of group and returning the flow of the document back to normal. More specifically, the :before and :after pseudo-elements, as mentioned in the Lesson 4 exercise, are dynamically generated elements above and below the element with the class of group. Those elements do not include any content and are displayed as table-level elements, much like block-level elements. The dynamically generated element after the element with the class of group is clearing the floats within the element with the class of group, much like the clear from before. And lastly, the element with the class of group itself also clears any floats that may appear above it, in case a left or right float may exist. It also includes a little trickery to get older browsers to play nicely. It is more code than the clear: both; declaration alone, but it can prove to be quite useful. Looking at our two-column page layout from before, we could wrap the <section> and <aside> elements with a parent element. That parent element then needs to contain the floats within itself. The code would look like this:

6.3 Inline-block

Inline-block elements are displayed on the same line as one another, they include a single space between them. When the size of each single space is added to the width and horizontal margin values of all the elements in the row, the total width becomes too great, pushing the last <section> element to a new row. In order to display all of the <section> elements on the same row, the white space between each <section> element must be removed.

Easiest way to remove the white space is to put the closing tags and opening tags on the same line

```
//1st way
<header>.</header>
<section>
...
</section><section>
...
```

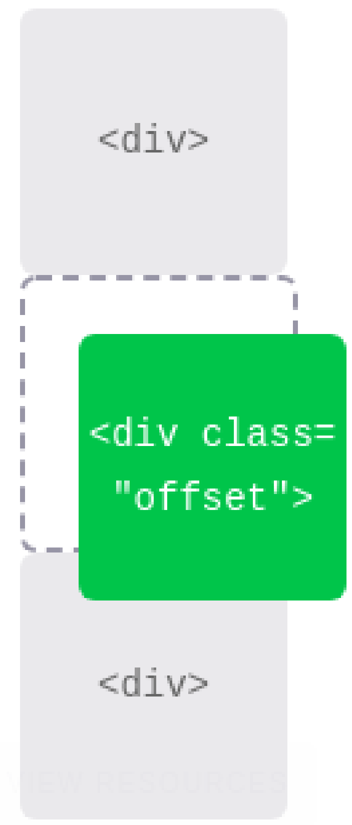


```
</section><section>
...
</section>
<footer>.</footer>

//2nd way
<header>.</header>
<section>
 ...
</section><!-
-><section>
 ...
</section><!-
-><section>
 ...
 </section>
 <footer>.</footer>
```

6.4 Relative Positioning

The relative value for the position property allows elements to appear within the normal flow a page, leaving space for an element as intended while not allowing other elements to flow around it; however, it also allows an element's display position to be modified with the box offset properties. For example, consider the following HTML and CSS:

```
\\html
<div>.</div>
<div class="offset">.</div>
<div>.</div>

\\css
div {
```



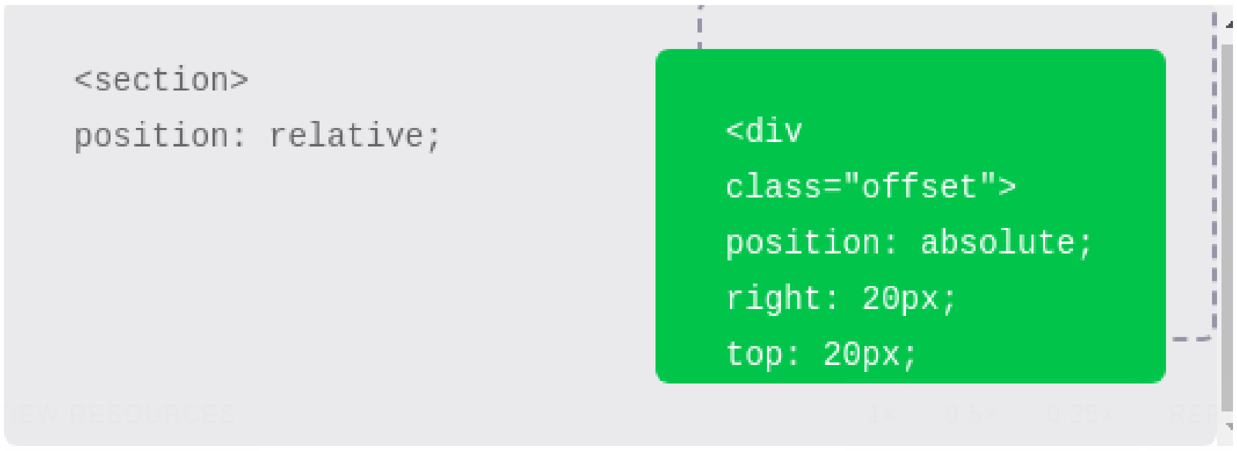
```
  height: 100px;
  width: 100px;
}
.offset {
  left: 20px;
  position: relative;
  top: 20px;
}
```

6.5 Absolute Positioning

The absolute value for the position property is different from the relative value in that an element with a position value of absolute will not appear within the normal flow of a document, and the original space and position of the absolutely positioned element will not be preserved. Additionally, absolutely positioned elements are moved in relation to their closest relatively positioned parent element. Should a relatively positioned parent element not exist, the absolutely positioned element will be positioned in relation to the <body> element. That's quite a bit of information; let's take a look at how this works inside some

```
//HTML
<section>
  <div class="offset">.</div>
</section>


//css
section {
  position: relative;
}
div {
  position: absolute;
  right: 20px;
  top: 20px;
}
```

With relatively positioned elements, the box offset properties identify in which direction an element would be moved in relation to itself. With absolutely positioned elements, the box offset properties identify in which direction an element will be moved in relation



```
<section>
position: relative;
```

```
<div
class="offset">
position: absolute;
right: 20px;
top: 20px;
```

to its closest relatively positioned parent element.

7 Working With Typography

7.1 Typeface vs Font

A typeface is what we see. It is the artistic impression of how text looks, feels, and reads. A font is a file that contains a typeface. Using a font on a computer allows the computer to access the typeface.

7.2 Font Properties

The first declared font, starting from the left, is the primary font choice. Should the first font be unavailable, alternative fonts are declared after it in order of preference from left to right. All properties have inherit.

```
body {
  font-style: italic; //normal, italic, oblique;
  font-size: 14px; //in pixels;
  color:#555; //abbreviated hex;
  font-weight: bold; //normal, bold, bolder, lighter;
  //100, 200, 300, 400, 500, 600, 700, 800, and 900
  font-variant: small-caps; //normal, small-caps;
  font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;
}
```

In this case, Helvetica Neue is the preferred font to display. If this font is unavailable or not installed on a given device, the next font in the list—Helvetica—will be used, and so on.

7.3 Line Height

The best practice for legibility is to set the line-height to around one and a half times our font-size property value. This could be quickly accomplished by setting the line-height to 150%, or just 1.5.

```
body {
  line-height:22px;
}

//can also be used to center text
body {
  line-height:22px;
  height:22px;
  text-align: center; //left,right,cetner,justify
  text-decoration: underline; // none, underline, overline, line-through
  text-indent: 20px;
  text-shadow: 3px 6px 2px rgba(0, 0, 0, .3); //hoffset, voffset, blur radius, color
  text-transform: uppercase;
  letter-spacing: -.5em;
  word-spacing: .25em;
}

//hover example
```

```
a:hover {
  color: #a9b2b9;
}
```

7.4 Use Web Safe Fonts

By default there are a few fonts that are pre-installed on every computer, tablet, smart-phone, or other web-browsing-capable device.

```
        ArialCourier New, CourierGaramondGeorgiaLucida Sans, Lucida Grande, LucidaPalatino LinotypeTahomaTimes
             New Roman, TimesTrebuchetVerdana
```

7.5 Embedding Web Fonts

We also have the ability to upload fonts to a server and include them on a website via the CSS @font-face at-rule. This capability has done wonders for online typography. Now, more than ever, typography is coming to life online.

```
@font-face {
  font-family: "Lobster";
  src: local("Lobster"), url("lobster.woff") format("woff");
}
body {
  font-family: "Lobster", "Comic Sans", cursive;
}
```

7.6 Including Citations Quotes

Writing online sometimes involves citing different sources or quotations. All of the different citation and quotation cases can be covered semantically in HTML using the <cite>, <q>, and <blockquote> elements.

```
<p>The book <cite><a href="http://www.amazon.com/Steve-Jobs-Walter-Isaacson/dp/1451648537">Steve Jobs</a></
    cite> is truly inspirational.</p>

<p>Steve Jobs once said, <q>One home run is much better than two doubles.</q></p>

<p><a href="http://www.businessweek.com/magazine/content/06_06/b3970001.htm">Steve Jobs</a> once said, <q cite
    ="http://www.businessweek.com/magazine/content/06_06/b3970001.htm">One home run is much better than two
    doubles.</q></p>

<blockquote>
  <p> #8220;In most people #8217;s vocabularies, design is a veneer. It #8217;s interior decorating. It #8217;
      s the fabric of the curtains, of the sofa. But to me, nothing could be further from the meaning of
      design. Design is the fundamental soul of a human-made creation that ends up expressing itself in
      successive outer layers of the product. #8221;</p>
</blockquote>



<blockquote cite="http://money.cnn.com/magazines/fortune/fortune_archive/2000/01/24/272277/index.htm">
  <p> #8220;In most people #8217;s vocabularies, design is a veneer. It #8217;s interior decorating. It #8217;
      s the fabric of the curtains, of the sofa. But to me, nothing could be further from the meaning of
      design. Design is the fundamental soul of a human-made creation that ends up expressing itself in
      successive outer layers of the product. #8221;</p>
  <p><cite> #8212; Steve Jobs in <a href="http://money.cnn.com/ magazines/fortune/fortune_archive
      /2000/01/24/272277/index.htm"> Fortune Magazine</a></cite></p>
</blockquote>
```

8 Backgrounds Gradients

Within CSS, element backgrounds can be a solid color, an image, a gradient, or a combination of these. As we decide how to implement these backgrounds, we should keep in mind that every background contributes to the overall appearance of our website.

Background Color When using an RGBa or HSLa value as a transparent background color, it's a good idea to provide a fallback color, too, because not all browsers recognize RGBa or HSLa values.

```
div {
  background-color: #b2b2b2; //fallback color
  background-color: rgba(0, 0, 0, .3);
}
```

Background Image The url() function value will be the background image's path, and the familiar rules for creating hyperlink paths apply here. Keep an eye out for different directories, and be sure to show exactly where the image resides. The path will be placed inside parentheses and quoted.

```
div {
  background-image: url("alert.png");
  background-repeat: no-repeat; //by default bg image repeats
  //repeat also accepts repeat, repeat-x, repeat y
}
```

Position By default, background images are positioned at the left top corner of an element. However, by using the background-position property, we can control exactly where the background image is placed relative to that corner. The background-position property requires two values: a horizontal offset (the first value) and a vertical offset (the second value).

```
          div {
            background-image: url("alert.png");
            background-position: 20px 10px;
            //keywords are also useful left, top; right, top; left,
            background-repeat: no-repeat;
          }
```

Shorthand for backgroudn and image values The background-color, background-image, background-position, and background-repeat properties may be rolled up into a shorthand value for the background property alone.

```
div {
  background: #b2b2b2 url("alert.png") 20px 10px no-repeat;
}
```

8.1 Gradient Backgrounds

Within CSS, gradient backgrounds are treated as background images. We can create a gradient using the background or background-image properties, just like a regular background image. The property value for a gradient background varies depending on what type of gradient we'd like, linear or radial.

Linear Gradients are identified by using the linear-gradient() function within the background or background-image property. The linear-gradient() function must include two color values, the first of which will be the beginning color value and the second of which will be the ending color value. The browser will then handle the transition between the two colors.

```
div {
  background: #466368;
  //this is for different browsers
  background: -webkit-linear-gradient(#648880, #293f50);
  background:    -moz-linear-gradient(#648880, #293f50);
  background:         linear-gradient(#648880, #293f50);
}

//usable directions are left, right bottom, top
//example witch changing the direction to right bottom
div {
  background: #466368;
  background: linear-gradient(to right bottom, #648880, #293f50);
}
```

Radial Gradients Radial background gradients work just like linear gradients and share many of the same values. For radial gradients, instead of using the linear-gradient() function within the background or background-image property, we'll use the radial-gradient() function.

Radial gradients work from the inside to the outside of an element. Thus, the first color identified within the radial-gradient() function will sit in the absolute center of the element, and the second color will sit on the outside of an element.

```
          div {
            background: #466368;
            background: radial-gradient(#648880, #293f50);
          }
```

Gradient Color Stops At a minimum, gradient backgrounds will transition from one color to another; however, we may add multiple colors to a gradient and have the browser transition between all of them.

```
div {
  background: #648880;
  background: linear-gradient(to right, #f6f1d3, #648880, #293f50);
}
```

8.2 Using Multiple Background Images

we can now use more than one background image on an element by comma-separating multiple background values within a background or background-image property. The background image value that comes first will be the foremost background image, and the background image that's listed last will be the rearmost background image. Any value between the first and the last will reside within the middle ground accordingly.

```
div {
  background:  url("foreground.png") 0 0 no-repeat, url("middle-ground.png") 0 0 no-repeat, url("
      background.png") 0 0 no-repeat;
}
```

8.3 New Background Properties

The background-size property allows us to change the size of a background image, while the background-clip and background-origin properties allow us to control where a background image is cropped and where a background image is contained within the element (inside the border or inside the padding, for example).

Background Size When using length values, we can specify a width and a height value by using two space-separated values. The first value will set the width of the background image, while the second value will set the height of the background image. It's important to note that percentage values are in relation to the element's size, not the background image's original size.

```
div {
  background: url("shay.jpg") 0 0 no-repeat;
  //this sets it auto width and 75% height
  background-size: auto 75%;
  border: 2px dashed #9799a7;
  height: 240px;
  width: 200px;
}
```

Cover Contain The cover keyword value specifies that the background image will be resized to completely cover an element's width and height. The contain keyword value, on the other hand, specifies that the background image will be resized to reside entirely contained within an element's width and height.

```
div {
  background: url("shay.jpg") 0 0 no-repeat;
  //this sets it auto width and 75% height
  background-size: cover; //contain cover expands h and w to parent element, contain dosent;
  border: 2px dashed #9799a7;
  height: 240px;
  width: 200px;
}
```

CSS3 Background Clip Background Origin The background-clip property specifies the surface area a background image will cover, and the background-origin property specifies where the background-position should originate. The introduction of these two new properties corresponds with the introduction of three new keyword values: border-box, padding-box, and content-box. Each of these three values may be used for the background-clip and background-origin properties.

```css
div {
  background: url("shay.jpg") 0 0 no-repeat;
  background-clip: padding-box;
  background-origin: border-box;
}
```

Fig 7.04

The `border-box` value extends the background into the border of an element

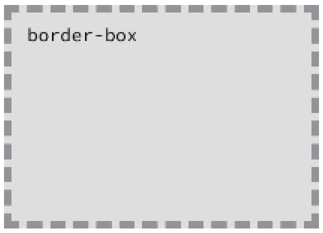


Fig 7.05

The `padding-box` value extends the background into the padding of an element, but the background is contained within any border

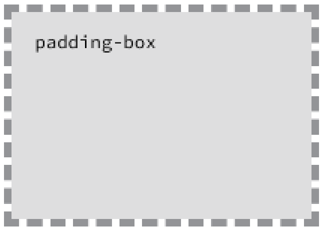
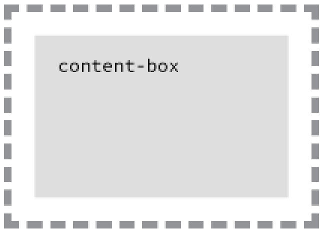


Fig 7.06

The `content-box` value contains the background within the border and padding of an element



9 Lists

When we want to use a list on a website, HTML provides three different types to choose from: unordered, ordered, and description lists. Choosing which type of list to use—or whether to use a list at all—comes down to the content and the most semantically appropriate option for displaying that content.

9.1 Unordered Lists

An unordered list is simply a list of related items whose order does not matter. Creating an unordered list in HTML is accomplished using the unordered list block-level element, <ul>. Each item within an unordered list is individually marked up using the list item element, <li>.

```html
<ul>
  <li>Orange</li>
  <li>Green</li>
  <li>Blue</li>
</ul>
```

9.2 Ordered Lists

The ordered list element, <ol>, works very much like the unordered list element; individual list items are created in the same manner. The main difference between an ordered list and an unordered list is that with an ordered list, the order in which items are presented is important.

```
<ol>
  <li>Head north on N Halsted St</li>
  <li>Turn right on W Diversey Pkwy</li>
  <li>Turn left on N Orchard St</li>
</ol>
```

Start/Reverse Attribute defines the number from which an ordered list should start. By default, ordered lists start at 1

```
<ol start="30">
  <li>Head north on N Halsted St</li>
  <li>Turn right on W Diversey Pkwy</li>
  <li>Turn left on N Orchard St</li>
</ol>

//reversed list displays number in reverse
<ol reversed>
  <li>Head north on N Halsted St</li> //3
  <li>Turn right on W Diversey Pkwy</li> //2
  <li>Turn left on N Orchard St</li> //1
</ol>
```

Value attribute may be used on an individual <li> within an ordered list, this changes the number to whatever value you assign

```
<ol>
  <li>Head north on N Halsted St</li>
  <li value="9">Turn right on W Diversey Pkwy</li>
  <li>Turn left on N Orchard St</li>
</ol>
```

9.3 Description Lists

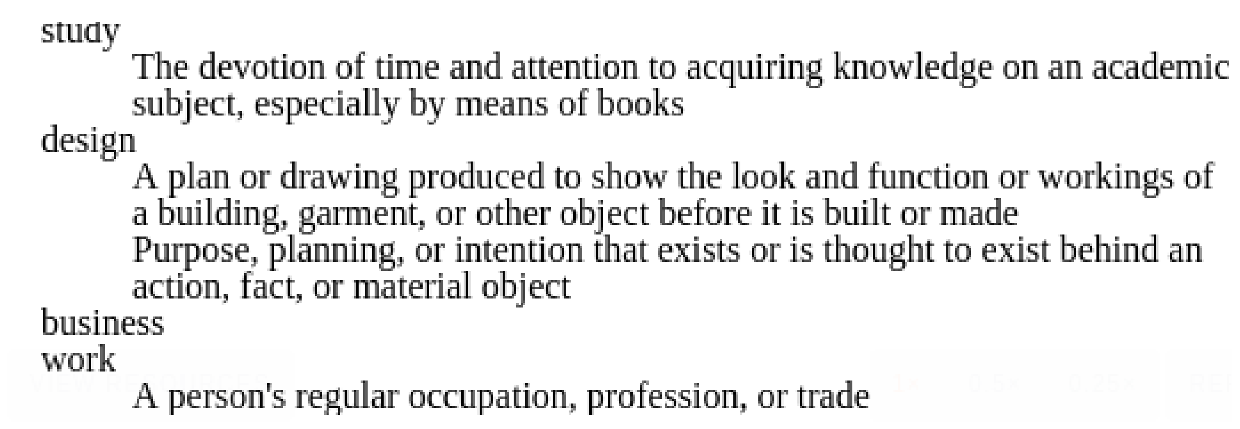
Creating a description list in HTML is accomplished using the description list block-level element, <dl>. Instead of using a <li> element to mark up list items, the description list requires two block-level elements: the description term element, <dt>, and the description element, <dd>.

```
<dl>
  <dt>study</dt>
    <dd>The devotion of time and attention to acquiring knowledge on an academic subject, especially by means
        of books</dd>
    <dt>design</dt>
    <dd>A plan or drawing produced to show the look and function or workings of a building, garment, or other
        object before it is built or made</dd>
    <dd>Purpose, planning, or intention that exists or is thought to exist behind an action, fact, or material
         object</dd>
  <dt>business</dt>
  <dt>work</dt>
    <dd>A persons regular occupation, profession, or trade</dd>
</dl>
```

9.4 Nesting Lists

One feature that makes lists extremely powerful is their ability to be nested. Every list may be placed within another list; they can be nested continually. But the potential to nest lists indefinitely doesn't provide free rein to do so. Lists should still be reserved specifically for where they hold the most semantic value.

```
<ol>
  <li>Walk the dog</li>
  <li>Fold laundry</li>
  <li>
    Go to the grocery and buy:
    <ul>
      <li>Milk</li>
      <li>Bread</li>
```

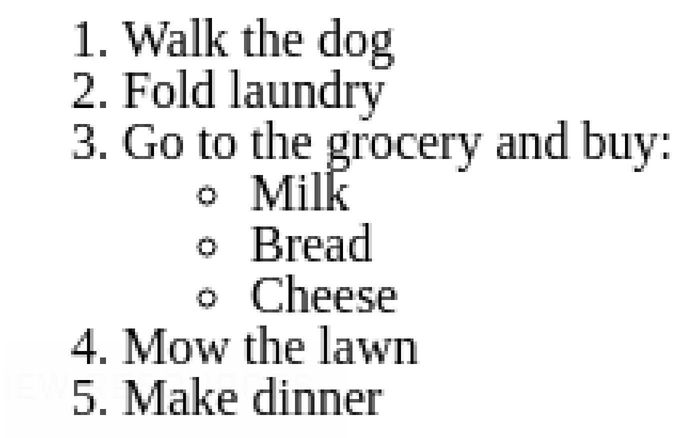


study
: The devotion of time and attention to acquiring knowledge on an academic subject, especially by means of books

design
: A plan or drawing produced to show the look and function or workings of a building, garment, or other object before it is built or made
: Purpose, planning, or intention that exists or is thought to exist behind an action, fact, or material object

business
work
: A person's regular occupation, profession, or trade

```
      <li>Cheese</li>
    </ul>
  </li>
  <li>Mow the lawn</li>
  <li>Make dinner</li>
</ol>
```

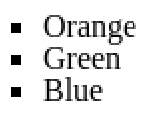


1. Walk the dog
2. Fold laundry
3. Go to the grocery and buy:
   - Milk
   - Bread
   - Cheese
4. Mow the lawn
5. Make dinner

9.5 List Item Styling

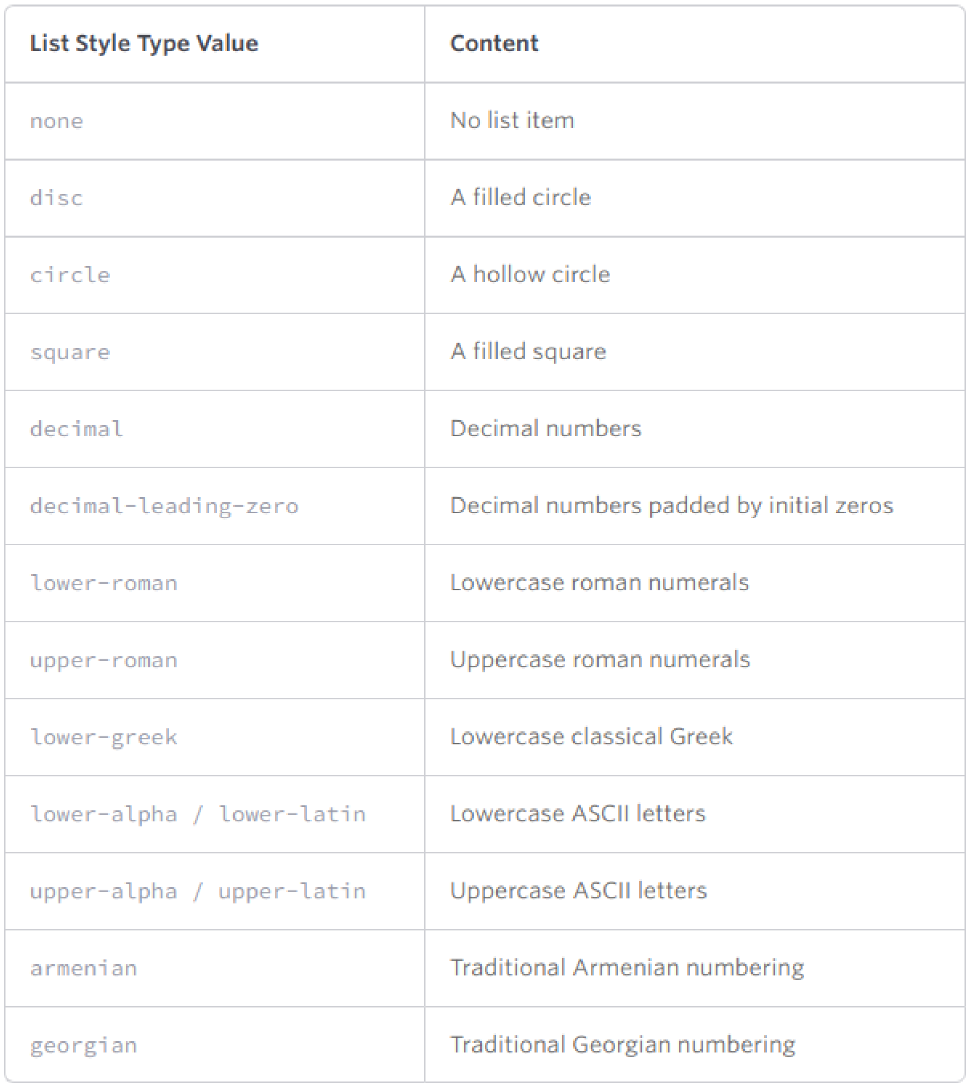
The list-style-type property is used to set the content of a list item marker. The available values range from squares and decimal numbers all the way to Armenian numbering, and the style may be placed on either the <ul>, <ol>, or <li> elements within CSS. Any list-style-type property value can be added to either unordered or ordered lists. With this in mind, it is possible to use a numeric list item marker on an unordered list and a nonnumeric marker on an ordered list.

```
//html
<ul>
  <li>Orange</li>
  <li>Green</li>
  <li>Blue</li>
</ul>

//css
ul {
  list-style-type: square;
}
```



▪ Orange
▪ Green
▪ Blue

9.6 List Type Values

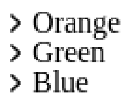


List Style Type Value	Content
`none`	No list item
`disc`	A filled circle
`circle`	A hollow circle
`square`	A filled square
`decimal`	Decimal numbers
`decimal-leading-zero`	Decimal numbers padded by initial zeros
`lower-roman`	Lowercase roman numerals
`upper-roman`	Uppercase roman numerals
`lower-greek`	Lowercase classical Greek
`lower-alpha / lower-latin`	Lowercase ASCII letters
`upper-alpha / upper-latin`	Uppercase ASCII letters
`armenian`	Traditional Armenian numbering
`georgian`	Traditional Georgian numbering

9.7 Image Marker

The process includes removing any default list-style-type property value and adding a background image and padding to the <li> element.

```
      //html
<ul>
  <li>Orange</li>
  <li>Green</li>
  <li>Blue</li>
</ul>
      //css
li {
  background: url("arrow.png") 0 50% no-repeat;
  list-style-type: none;
  padding-left: 12px;
}
```

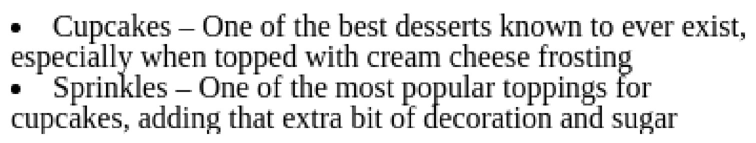


› Orange
› Green
› Blue

9.8 Shorthand List Style Property

The order of these shorthand values should be list-style-type followed by list-style-position.

```
      ul {
        list-style: circle inside;
      }
      ol {
        list-style: lower-roman;
      }
```



• Cupcakes – One of the best desserts known to ever exist, especially when topped with cream cheese frosting
• Sprinkles – One of the most popular toppings for cupcakes, adding that extra bit of decoration and sugar

9.9 Horizontal List

The quickest way to display a list on a single line is to give the <li> elements a display property value of inline or inline-block. Doing so places all the <li> elements within a single line, with a single space between each list item.

```
//html
<ul>
  <li>Orange</li>
  <li>Green</li>
  <li>Blue</li>
</ul>
//css
li {
  display: inline-block;
  margin: 0 10px;
}
```

9.10 Floating List

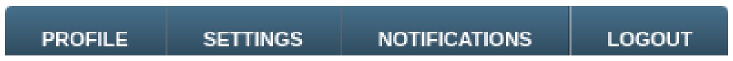
Changing the display property value to inline or inline-block is quick; however, it removes the list item marker. If the list item marker is needed, floating each <li> element is a better option than changing the display property.

```
//html
<ul>
  <li>Orange</li>
  <li>Green</li>
  <li>Blue</li>
</ul>

//css
li {
  float: left;
  margin: 0 20px;
}
```

9.11 Navigational List Example

We'll often develop, and find, navigation menus using unordered lists. These lists are commonly laid out as horizontal lists, using either of the two techniques previously mentioned. Here is an example of a horizontal navigation menu marked up using an unordered list with <li> elements displayed as inline-block elements.

```
//html
<nav class="navigation">
  <ul>
    <li><a href="#">Profile</a></li><!-
    -><li><a href="#">Settings</a></li><!-
    -><li><a href="#">Notifications</a></li><!-
    -><li><a href="#">Logout</a></li>
  </ul>
</nav>

//css
.navigation ul {
  font: bold 11px "Helvetica Neue", Helvetica, Arial, sans-serif;
  margin: 0;
  padding: 0;
  text-transform: uppercase;
}
.navigation li {
  display: inline-block;
}
.navigation a {
  background: #395870;
  background: linear-gradient(#49708f, #293f50);
  border-right: 1px solid rgba(0, 0, 0, .3);
  color: #fff;
  padding: 12px 20px;
  text-decoration: none;
}
.navigation a:hover {
  background: #314b60;
  box-shadow: inset 0 0 10px 1px rgba(0, 0, 0, .3);
}
.navigation li:first-child a {
  border-radius: 4px 0 0 4px;
}
.navigation li:last-child a {
  border-right: 0;
  border-radius: 0 4px 4px 0;
}
```



9.12 Nav Element

We're going to want to change the display value of our <li> elements to inline-block to get all of them to align in a horizontal row. When we do that, though, we'll also need to account for the blank space left between each <li> element. We'll begin by setting all of the <li> elements within any element with the class attribute value of nav to be displayed inline-block, to include some horizontal margins, and to be vertically aligned to the top of the element. Additionally, we'll use the :last-child pseudo-class selector to identify the last <li> element and reset its right margin to 0. Doing so ensures that any horizontal space between the <li> element and the edge of its parent element is removed. Within our main.css file, below our existing navigation styles, let's add the following CSS:

```
//header
<nav class="nav primary-nav">
  <ul>
    <li><a href="index.html">Home</a></li><!-
    -><li><a href="speakers.html">Speakers</a></li><!-
    -><li><a href="schedule.html">Schedule</a></li><!-
    -><li><a href="venue.html">Venue</a></li><!-
    -><li><a href="register.html">Register</a></li>
  </ul>
</nav>

//css
.nav li {
  display: inline-block;
  margin: 0 10px;
  vertical-align: top;
}
.nav li:last-child {
  margin-right: 0;
}
```

10 Adding Media

HTML provides ways to embed rich media in the form of images, audio tracks, and videos, as well as to embed content from another web page in the form of an inline frame.

10.1 Images

To add images to a page, we use the <img> inline element. The <img> element is a self-containing, or empty, element, which means that it doesn't wrap any other content and it exists as a single tag. For the <img> element to work, a src attribute and value must be included to specify the source of the image. The src attribute value is a URL, typically relative to the server where a website is hosted.

```
<img src="dog.jpg" alt="A black, brown, and white dog wearing a kerchief">
```

Sizing One option is to use the width and height attributes directly within the <img> tag in HTML.Additionally, images may be sized using the width and height properties in CSS. When both the HTML attributes and CSS properties are used, the CSS attributes will take precedence over the HTML attributes.

```
img {
  height: 200px;
  width: 200px;
  display: block; //by default images are inline
}
```

Float left or right To do this we use the float property with a value of either left or right.

```
img {
  background: #eaeaed;
  border: 1px solid #9799a7;
  float: right;
  margin: 8px 0 0 20px;
  padding: 4px;
}
```

10.2 Audio

HTML5 provides a quick and easy way to add audio files to a website by way of the <audio> element. As with the <img> element, the <audio> element accepts a source URL specified by the src attribute. Unlike the <img> element, though, the <audio> element requires both opening and closing tags, which we'll discuss soon.

```
<audio src="jazz.ogg"></audio>
```

Audio Attributes Several other attributes may accompany the src attribute on the <audio> element; the most popular include autoplay, controls, loop, and preload. The autoplay, controls, and loop attributes are all Boolean attributes. As Boolean attributes, they don't require a stated value. Instead, when each is present on the <audio> element its value will be set to true, and the <audio> element will behave accordingly.

```
<audio src="jazz.ogg" autoplay></audio>
<audio src="jazz.ogg" controls></audio>
```

Audio Fallbacks Because it was introduced in HTML5, some browsers may not support the <audio> element. In this case, we can provide a link to download the audio file after any <source> elements within the <audio> element.

```
  <audio controls>
    <source src="jazz.ogg" type="audio/ogg">
    <source src="jazz.mp3" type="audio/mpeg">
    <source src="jazz.wav" type="audio/wav">
    Please <a href="jazz.mp3" download>download</a> the audio file.
  </audio>
```

10.3 Video

Adding video in HTML5 is very similar to adding audio. We use the <video> element in place of the <audio> element. All of the same attributes (src, autoplay, controls, loop, and preload) and fallbacks apply here, too With the <audio> element, if the controls Boolean attribute isn't specified the audio clip isn't displayed. With videos, if the controls Boolean attribute is not specified the video will display.

```
<video src="earth.ogv" controls></video>
```

Poster One additional attribute available for the <video> element is the poster attribute. The poster attribute allows us to specify an image, in the form of a URL, to be shown before a video is played. The example below uses a screen capture from the video as the poster for the Earth video.

```
<video src="earth.ogv" controls poster="earth-video-screenshot.jpg"></video>
```

Video Fallbacks As with the <audio> element, video fallbacks are also necessary. The same markup format, with multiple <source> elements for each file type and a plain text fallback, also applies within the <video> element

```
<video controls>
  <source src="earth.ogv" type="video/ogg">
  <source src="earth.mp4" type="video/mp4">
  Please <a href="earth.mp4" download>download</a> the video.
</video>
```

Embedded Videos One additional fallback option that could be used in place of a plain text fallback is to use a YouTube or Vimeo embedded video. These video hosting websites allow us to upload our videos, provide a standard video player, and enable us to embed our videos onto a page using an inline frame.

10.4 Iframes

Another way to add content to a page is to embed another HTML page within the current page. This is done using an inline frame, or <iframe> element. The <iframe> element accepts the URL of another HTML page within the src attribute value; this causes the content from the embedded HTML page to be displayed on the current page.

```
<iframe src="https://www.google.com/maps/embed?..."></iframe>
```

10.5 Figure

The <figure> block-level element is used to identify and wrap self-contained content, often in the form of media. It may surround images, audio clips, videos, blocks of code, diagrams, illustrations, or other self-contained media.

```
<figure>
  <img src="dog.jpg" alt="A black, brown, and white dog wearing a kerchief">
</figure>

<figure>
  <img src="dog.jpg">
  <figcaption>A beautiful black, brown, and white hound dog wearing kerchief.</figcaption>
</figure
```

11 Building Forms

Forms are an essential part of the Internet, as they provide a way for websites to capture information from users and to process requests, and they offer controls for nearly every imaginable use of an application. Through controls or fields, forms can request a small amount of information—often a search query or a username and password—or a large amount of information—perhaps shipping and billing information or an entire job application.

A beautiful black, brown, and white hound dog wearing kerchief.

11.1 Initializing a Form

To add a form to a page, we'll use the <form> element. The <form> element identifies where on the page control elements will appear. Additionally, the <form> element will wrap all of the elements included within the form, much like a <div> element.

```html
<form action="/login" method="post">
 ...
</form>
```

A handful of different attributes can be applied to the <form> element, the most common of which are action and method. The action attribute contains the URL to which information included within the form will be sent for processing by the server. The method attribute is the HTTP method browsers should use to submit the form data. Both of these <form> attributes pertain to submitting and processing data.

11.2 Text Fields Textareas

When it comes to gathering text input from users, there are a few different elements available for obtaining data within forms. Specifically, text fields and textareas are used for collecting text- or string-based data. This data may include passages of text content, passwords, telephone numbers, and other information.

Text Fields One of the primary elements used to obtain text from users is the <input> element. The <input> element uses the type attribute to define what type of information is to be captured within the control. The most popular type attribute value is text, which denotes a single line of text input.Along with setting a type attribute, it is best practice to give an <input> element a name attribute as well. The name attribute value is used as the name of the control and is submitted along with the input data to the server.

```html
<input type="text" name="username">
```

Different types These values were added to provide clearer semantic meaning for inputs as well as to provide better controls for users. Should a browser not understand one of these HTML5 type attribute values, it will automatically fall back to the text attribute value. Below is a list of the new HTML5 input types.

- color
- date
- datetime
- email
- month
- number
- range
- search
- tel
- time
- url
- week

```html
<input type="date" name="birthday">
<input type="time" name="game-time">
<input type="email" name="email-address">
<input type="url" name="website">
<input type="number" name="cost">
<input type="tel" name="phone-number">
```

Fig 10.01

iOS7 controls for an <input> element with a type attribute value of date

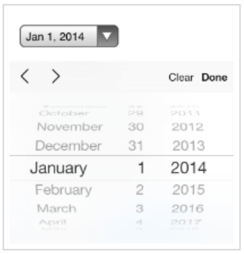


Fig 10.04

iOS7 controls for an <input> element with a type attribute value of url

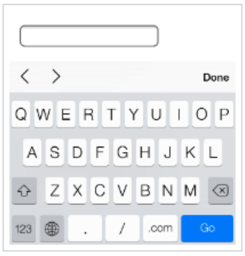


Fig 10.02

iOS7 controls for an <input> element with a type attribute value of time

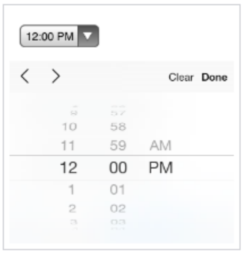


Fig 10.05

iOS7 controls for an <input> element with a type attribute value of number

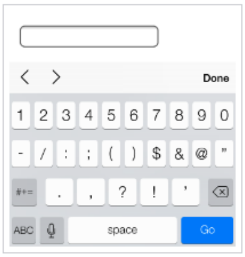


Fig 10.03

iOS7 controls for an <input> element with a type attribute value of email

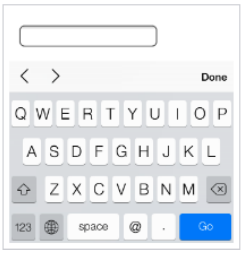
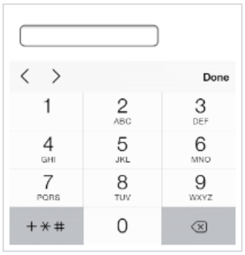


Fig 10.06

iOS7 controls for an <input> element with a type attribute value of tel



11.3 Text Area

Another element that's used to capture text-based data is the <textarea> element. The <textarea> element differs from the <input> element in that it can accept larger passages of text spanning multiple lines. The <textarea> element also has start and end tags that can wrap plain text.

```html
<textarea name="comment">Add your comment here</textarea>
```

Multiple Choice Apart from text-based input controls, HTML also allows users to select data using multiple choice and drop-down lists. There are a few different options and elements for these form controls, each of which has distinctive benefits.

Radio Buttons Radio buttons are an easy way to allow users to make a quick choice from a small list of options. Radio buttons permit users to select one option only, as opposed to multiple options.

```html
<input type="radio" name="day" value="Friday" checked> Friday
<input type="radio" name="day" value="Saturday"> Saturday
<input type="radio" name="day" value="Sunday"> Sunday
```

Check Boxes Check boxes are very similar to radio buttons. They use the same attributes and patterns, with the exception of checkbox as their type attribute value. The difference between the two is that check boxes allow users to select multiple values and tie them all to one control name, while radio buttons limit users to one value.

```html
<input type="checkbox" name="day" value="Friday" checked> Friday
<input type="checkbox" name="day" value="Saturday"> Saturday
<input type="checkbox" name="day" value="Sunday"> Sunday
```

Drop Down lists To create a drop-down list we'll use the <select> and <option> elements. The <select> element wraps all of the menu options, and each menu option is marked up using the <option> element. The name attribute resides on the <select> element, and the value attribute resides on the <option> elements that are nested within the <select> element. The value attribute on each <option> element then corresponds to the name attribute on the <select> element.

```html
<select name="day">
  <option value="Friday" selected>Friday</option>
  <option value="Saturday">Saturday</option>
  <option value="Sunday">Sunday</option>
</select>
```

Multiple Selections The Boolean attribute multiple, when added to the <select> element for a standard drop-down list, allows a user to choose more than one option from the list at a time. Additionally, using the selected Boolean attribute on more than one <option> element within the menu will preselect multiple options.

```html
<select name="day" multiple>
  <option value="Friday" selected>Friday</option>
  <option value="Saturday">Saturday</option>
  <option value="Sunday">Sunday</option>
</select>
```

11.4 Form Buttons

After a user inputs the requested information, buttons allow the user to put that infor- mation into action. Most commonly, a submit input or submit button is used to process the data.

Submit Input Users click the submit button to process data after filling out a form. The submit button is created using the <input> element with a type attribute value of submit. The value attribute is used to specify the text that appears within the button.

```html
<input type="submit" name="submit" value="Send">
```

Submit Button The <button> element performs the same way as the <input> element with the type attribute value of submit; however, it includes opening and closing tags, which may wrap other elements. By default, the <button> element acts as if it has a type attribute value of submit, so the type attribute and value may be omitted from the <button> element if you wish.

```html
<button name="submit">
  <strong>Send Us</strong> a Message
</button>
```

11.5 Other Inputs

Besides the applications we've just discussed, the <input> element has a few other use cases. These include passing hidden data and attaching files during form processing.

```html
<button name="submit">
  <strong>Send Us</strong> a Message
</button>
```

11.6 Hidden Input

Hidden inputs provide a way to pass data to the server without displaying it to users. Hidden inputs are typically used for tracking codes, keys, or other information that is not pertinent to the user but is helpful when processing the form. This information is not displayed on the page; however, it can be found by viewing the source code of a page. It should therefore not be used for sensitive or secure information.

```html
<input type="hidden" name="tracking-code" value="abc-123">
```

11.7 File Input

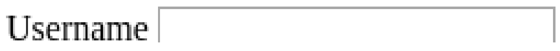
To allow users to add a file to a form, much like attaching a file to an email, use the file value for the type attribute.

```html
<input type="file" name="file">
```

11.8 Form Elements

By using labels, fieldsets, and legends, we can better organize forms and guide users to properly complete them.

```html
<label for="username">Username</label>
<input type="text" name="username" id="username">
```



<label> If desired, the <label> element may wrap form controls, such as radio buttons or check boxes. Doing so allows omission of the for and id attributes.

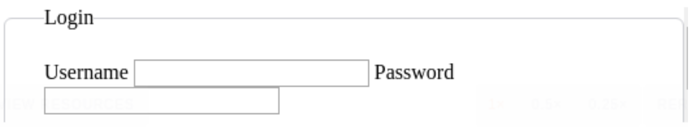
```html
<label>
  <input type="radio" name="day" value="Friday" checked> Friday
</label>
<label>
  <input type="radio" name="day" value="Saturday"> Saturday
</label>
<label>
  <input type="radio" name="day" value="Sunday"> Sunday
</label>
```

<fieldset> group form controls and labels into organized sections. Much like a <section> or other structural element, the <fieldset> is a block-level element that wraps related elements, specifically within a <form> element, for better organization. Fieldsets, by default, also include a border outline, which can be modified using CSS.

```html
<fieldset>
  <label>
    Username
    <input type="text" name="username">
  </label>
  <label>
    Password
    <input type="text" name="password">
  </label>
</fieldset>
```

Legend provides a caption, or heading, for the <fieldset> element. The <legend> element wraps text describing the form controls that fall within the fieldset. The markup should include the <legend> element directly after the opening <fieldset> tag. On the page, the legend will appear within the top left part of the fieldset border.

```html
<fieldset>
  <legend>Login</legend>
  <label>
    Username
    <input type="text" name="username">
  </label>
  <label>
    Password
    <input type="text" name="password">
  </label>
</fieldset>
```



Disabled The disabled Boolean attribute turns off an element or control so that it is not available for interaction or input. Elements that are disabled will not send any value to the server for form processing.

```
<label>
  Username
  <input type="text" name="username" disabled>
</label>
```

Placeholder The placeholder HTML5 attribute provides a hint or tip within the form control of an <input> or <textarea> element that disappears once the control is clicked in or gains focus. This is used to give users further information on how the form input should be filled in, for example, the email address format to use.

```
//value pre-populates the data for the field, while placeholder is a hint
<label>
  Email Address
  <input type="email" name="email-address" placeholder="name@domain.com">
</label>
```

Required The required HTML5 Boolean attribute enforces that an element or form control must contain a value upon being submitted to the server. Should an element or form control not have a value, an error message will be displayed requesting that the user complete the required field. Currently, error message styles are controlled by the browser and cannot be styled with CSS. Invalid elements and form controls, on the other hand, can be styled using the :optional and :required CSS pseudo-classes.Validation also occurs specific to a control's type. For example, an <input> element with a type attribute value of email will require not only that a value exist within the control, but also that it is a valid email address.

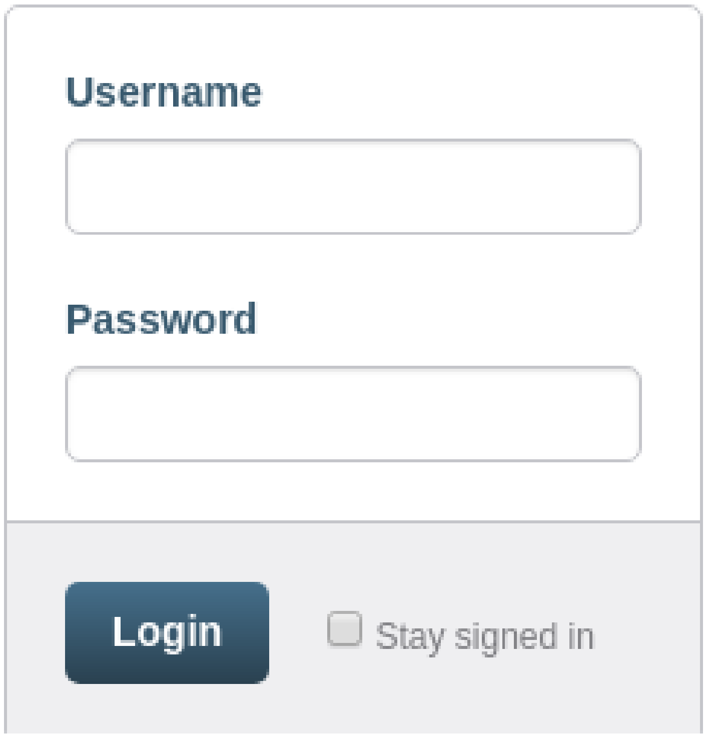
```
<label>
  Email Address
  <input type="email" name="email-address" required>
</label>
```

Additional Attributes Other form and form control attributes include, but are not limited to, the following. Please feel free to research these attributes as necessary.

- accept
- autocomplete
- autofocus
- formaction
- formenctype
- formmethod
- formnovalidate
- formtarget
- max
- maxlength
- min
- pattern
- readonly
- selectionDirection
- step

Login Example The following is an example of a complete login form that includes several different elements and attributes to illustrate what we've covered so far. These elements are then styled using CSS.

```
//html
<form>
  <fieldset class="account-info">
    <label>
      Username
      <input type="text" name="username">
    </label>
    <label>
      Password
      <input type="password" name="password">
    </label>
  </fieldset>
  <fieldset class="account-action">
    <input class="btn" type="submit" name="submit" value="Login">
    <label>
      <input type="checkbox" name="remember"> Stay signed in
    </label>
  </fieldset>
</form>

//css
*,
*:before,
*:after {
   box-sizing: border-box;
}
form {
  border: 1px solid #c6c7cc;
  border-radius: 5px;
  font: 14px/1.4 "Helvetica Neue", Helvetica, Arial, sans-serif;
  overflow: hidden;
  width: 240px;
}
fieldset {
  border: 0;
  margin: 0;
  padding: 0;
}
input {
  border-radius: 5px;
  font: 14px/1.4 "Helvetica Neue", Helvetica, Arial, sans-serif;
  margin: 0;
}
.account-info {
  padding: 20px 20px 0 20px;
}
.account-info label {
  color: #395870;
  display: block;
  font-weight: bold;
  margin-bottom: 20px;
}
.account-info input {
  background: #fff;
  border: 1px solid #c6c7cc;
   box-shadow: inset 0 1px 1px rgba(0, 0, 0, .1);
  color: #636466;
  padding: 6px;
  margin-top: 6px;
  width: 100%;
}
.account-action {
  background: #f0f0f2;
  border-top: 1px solid #c6c7cc;
  padding: 20px;
}
.account-action .btn {
  background: linear-gradient(#49708f, #293f50);
  border: 0;
  color: #fff;
  cursor: pointer;
  font-weight: bold;
  float: left;
  padding: 8px 16px;
}
.account-action label {
  color: #7c7c80;
  font-size: 12px;
  float: left;
  margin: 10px 0 0 20px;
}
```



12 Tables

HTML tables were created to provide a straightforward way to mark up structured tabular data and to display that data in a form that is easy for users to read and digest.

12.1 Creating a Table

Tables are made up of data that is contained within columns and rows, and HTML supplies several different elements for defining and structuring these items. At a minimum a table must consist of <table>, <tr> (table row), and <td> (table data) elements. For greater structure and additional semantic value, tables may include the <th> (table header)

Table We use the <table> element to initialize a table on a page. Using the <table> element signifies that the information within this element will be tabular data displayed in the necessary columns and rows.

```
<table>
 ...
</table>
```

Table Row Once a table has been defined in HTML, table rows may be added using the <tr> element. A table can have numerous table rows, or <tr> elements. Depending on the amount of information there is to display, the number of table rows may be substantial.

```
<table>
  <tr>
   ...
  </tr>
  <tr>
   ...
  </tr>
</table>
```

Table Data Once a table is defined and rows within that table have been set up, data cells may be added to the table via the table data, or <td>, element. Listing multiple <td> elements one after the other will create columns within a table row.

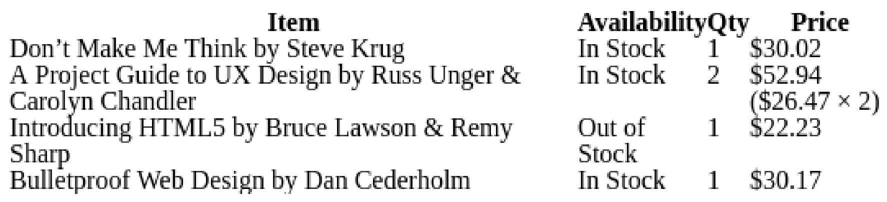
```
<table>
  <tr>
```

```
    <td>Don #8217;t Make Me Think by Steve Krug</td>
    <td>In Stock</td>
    <td>1</td>
    <td>$30.02</td>
  </tr>
  <tr>
    <td>A Project Guide to UX Design by Russ Unger  #38; Carolyn Chandler</td>
    <td>In Stock</td>
    <td>2</td>
    <td>$52.94 ($26.47  #215; 2)</td>
  </tr>
  <tr>
    <td>Introducing HTML5 by Bruce Lawson  #38; Remy Sharp</td>
    <td>Out of Stock</td>
    <td>1</td>
    <td>$22.23</td>
  </tr>
  <tr>
    <td>Bulletproof Web Design by Dan Cederholm</td>
    <td>In Stock</td>
    <td>1</td>
    <td>$30.17</td>
  </tr>
</table>
```

Don't Make Me Think by Steve Krug	In Stock	1$30.02
A Project Guide to UX Design by Russ Unger & Carolyn Chandler	In Stock	2$52.94 ($26.47 × 2)
Introducing HTML5 by Bruce Lawson & Remy Sharp	Out of Stock	1$22.23
Bulletproof Web Design by Dan Cederholm	In Stock	1$30.17

Table Header To designate a heading for a column or row of cells, the table header element, <th>, should be used.

```
<table>
  <tr>
    <th scope="col">Item</th>
    <th scope="col">Availability</th>
    <th scope="col">Qty</th>
    <th scope="col">Price</th>
  </tr>
  <tr>
    <td>Don #8217;t Make Me Think by Steve Krug</td>
    <td>In Stock</td>
    <td>1</td>
    <td>$30.02</td>
  </tr>
  <tr>
    <td>A Project Guide to UX Design by Russ Unger  #38; Carolyn Chandler</td>
    <td>In Stock</td>
    <td>2</td>
    <td>$52.94 ($26.47  #215; 2)</td>
  </tr>
  <tr>
    <td>Introducing HTML5 by Bruce Lawson  #38; Remy Sharp</td>
    <td>Out of Stock</td>
    <td>1</td>
    <td>$22.23</td>
  </tr>
  <tr>
    <td>Bulletproof Web Design by Dan Cederholm</td>
    <td>In Stock</td>
    <td>1</td>
    <td>$30.17</td>
  </tr>
</table>
```

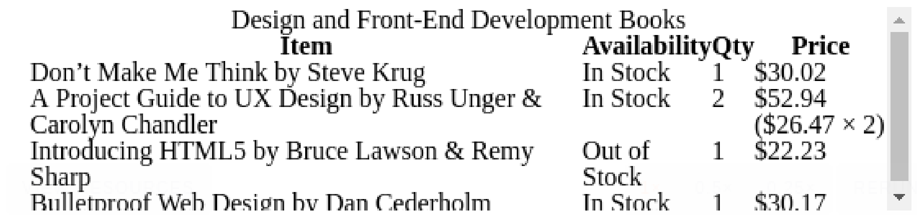


Item	Availability	Qty	Price
Don’t Make Me Think by Steve Krug	In Stock	1	$30.02
A Project Guide to UX Design by Russ Unger & Carolyn Chandler	In Stock	2	$52.94 ($26.47 × 2)
Introducing HTML5 by Bruce Lawson & Remy Sharp	Out of Stock	1	$22.23
Bulletproof Web Design by Dan Cederholm	In Stock	1	$30.17

12.2 Table Structure

Additional elements include these elements include <caption>, <thead>, <tbody>, and <tfoot>

Table Caption The <caption> element provides a caption or title for a table. A caption will help users identify what the table pertains to and what data they can expect to find within it. The <caption> element must come immediately after the opening <table> tag, and it is positioned at the top of a table by default.

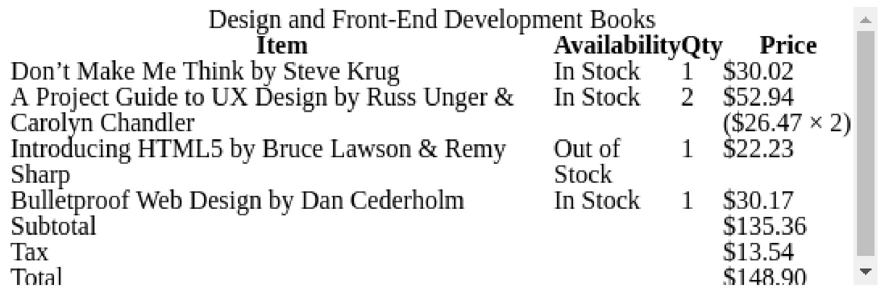


Design and Front-End Development Books

Item	Availability	Qty	Price
Don’t Make Me Think by Steve Krug	In Stock	1	$30.02
A Project Guide to UX Design by Russ Unger & Carolyn Chandler	In Stock	2	$52.94 ($26.47 × 2)
Introducing HTML5 by Bruce Lawson & Remy Sharp	Out of Stock	1	$22.23
Bulletproof Web Design by Dan Cederholm	In Stock	1	$30.17

```
<table>
  <caption>Design and Front-End Development Books</caption>
 ...
</table>
```

Table Head, Body Foot The table head element, <thead>, wraps the heading row or rows of a table to denote the head. The table head should be placed at the top of a table, after any <caption> element and before any <tbody> element. After the table head may come either the <tbody> or <tfoot> elements. Originally the <tfoot> element had to come immediately after the <thead> element, but HTML5 has provided leeway here. These elements may now occur in any order so long as they are never parent elements of one another. The <tbody> element should contain the primary data within a table, while the <tfoot> element contains data that outlines the contents of a table.

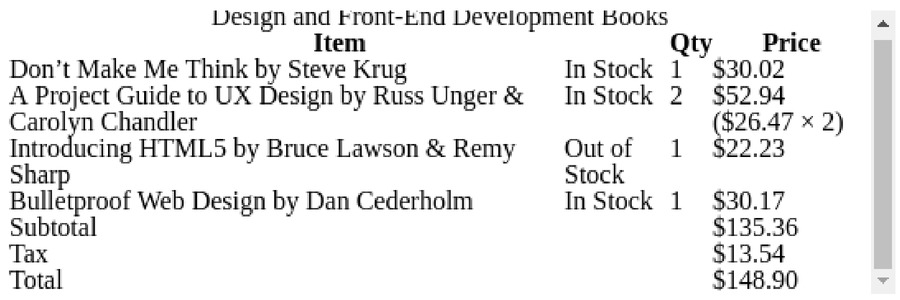


Design and Front-End Development Books

Item	Availability	Qty	Price
Don’t Make Me Think by Steve Krug	In Stock	1	$30.02
A Project Guide to UX Design by Russ Unger & Carolyn Chandler	In Stock	2	$52.94 ($26.47 × 2)
Introducing HTML5 by Bruce Lawson & Remy Sharp	Out of Stock	1	$22.23
Bulletproof Web Design by Dan Cederholm	In Stock	1	$30.17
Subtotal			$135.36
Tax			$13.54
Total			$148.90

```
<table>
  <caption>Design and Front-End Development Books</caption>
  <thead>
    <tr>
      <th scope="col">Item</th>
      <th scope="col">Availability</th>
      <th scope="col">Qty</th>
      <th scope="col">Price</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Don #8217;t Make Me Think by Steve Krug</td>
      <td>In Stock</td>
      <td>1</td>
      <td>$30.02</td>
    </tr>
   ...
  </tbody>
  <tfoot>
    <tr>
      <td>Subtotal</td>
      <td></td>
      <td></td>
      <td>$135.36</td>
    </tr>
    <tr>
      <td>Tax</td>
      <td></td>
      <td></td>
      <td>$13.54</td>
    </tr>
    <tr>
      <td>Total</td>
      <td></td>
      <td></td>
      <td>$148.90</td>
    </tr>
  </tfoot>
</table>
```

Combining Cells In these cases we can use the colspan and rowspan attributes. These two attributes work on either the <td> or <th> elements. The colspan attribute is used to span a single cell across multiple columns within a table, while the rowspan attribute is used to span a single cell across multiple rows. Each attribute accepts an integer value that indicates the number of cells to span across, with 1 being the default value.

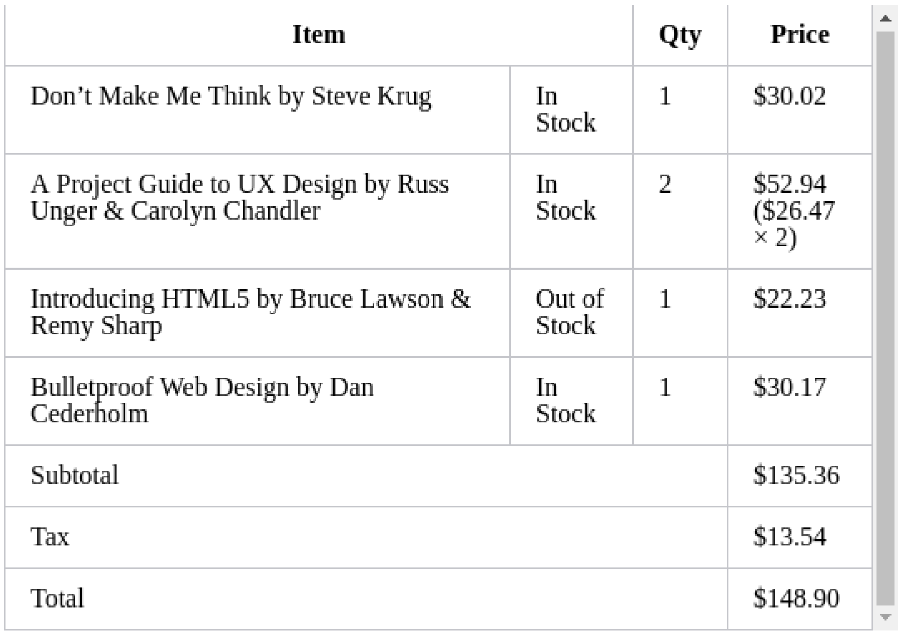


Design and Front-End Development Books

Item		Qty	Price
Don’t Make Me Think by Steve Krug	In Stock	1	$30.02
A Project Guide to UX Design by Russ Unger & Carolyn Chandler	In Stock	2	$52.94 ($26.47 × 2)
Introducing HTML5 by Bruce Lawson & Remy Sharp	Out of Stock	1	$22.23
Bulletproof Web Design by Dan Cederholm	In Stock	1	$30.17
Subtotal			$135.36
Tax			$13.54
Total			$148.90

```
<table>
  <caption>Design and Front-End Development Books</caption>
  <thead>
    <tr>
      <th scope="col" colspan="2">Item</th>
      <th scope="col">Qty</th>
      <th scope="col">Price</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Don #8217;t Make Me Think by Steve Krug</td>
      <td>In Stock</td>
      <td>1</td>
      <td>$30.02</td>
    </tr>
   ...
  </tbody>
  <tfoot>
    <tr>
      <td colspan="3">Subtotal</td>
      <td>$135.36</td>
    </tr>
    <tr>
      <td colspan="3">Tax</td>
      <td>$13.54</td>
    </tr>
    <tr>
      <td colspan="3">Total</td>
      <td>$148.90</td>
    </tr>
  </tfoot>
</table>
```

Borders Effective use of borders can help make tables more comprehensible. Borders around a table or individual cells can make a large impact when a user is trying to interpret data and quickly scan for information. When styling table borders with CSS there are two properties that will quickly come in handy: border-collapse and border-spacing. The border-collapse property determines a table's border model. There are three values for the border-collapse property: collapse, separate, and inherit. By default, the border-collapse property value is separate, meaning that all of the different borders will stack up next to one another, as described above. The collapse value, on the other hand, condenses the borders into one, choosing the table cell as the primary border.

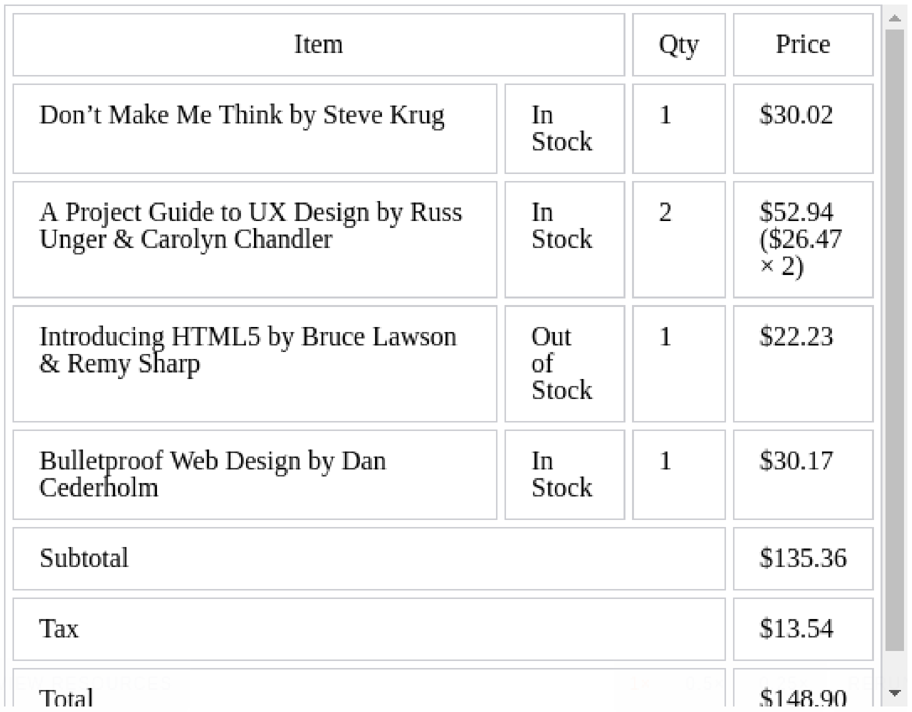


Item		Qty	Price
Don't Make Me Think by Steve Krug	In Stock	1	$30.02
A Project Guide to UX Design by Russ Unger & Carolyn Chandler	In Stock	2	$52.94 ($26.47 × 2)
Introducing HTML5 by Bruce Lawson & Remy Sharp	Out of Stock	1	$22.23
Bulletproof Web Design by Dan Cederholm	In Stock	1	$30.17
Subtotal			$135.36
Tax			$13.54
Total			$148.90

```
table {
  border-collapse: collapse;
}
th,
td {
  border: 1px solid #cecfd5;
  padding: 10px 15px;
}
```

Border Spacing a table with a 1-pixel border around the entire table and a 1-pixel border around each cell will have a 2-pixel border all around every cell because the borders stack up next to one another. Adding in a border-spacing value of 4 pixels separates the borders by 4 pixels. The border-spacing property works only when the border-collapse property value is separate, its default value. If the border-collapse property hasn't been previously used, we can use the border-spacing property without worry. Additionally, the border-spacing property may accept two length values: the first value for horizontal spacing and the second value for vertical spacing. The declaration border-spacing: 5px 10px;,

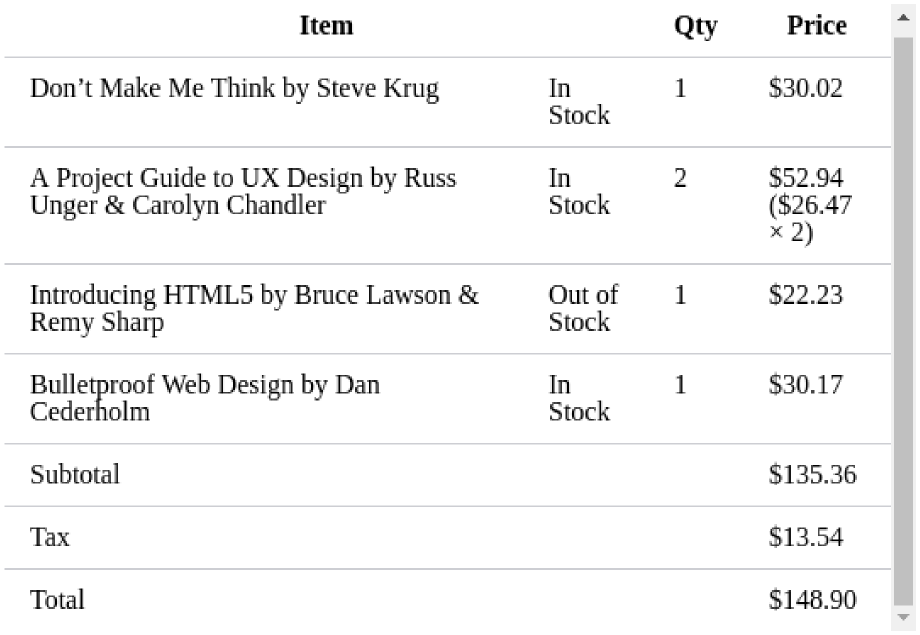
```
 table {
  border-collapse: separate;
  border-spacing: 4px;
}
table,
th,
td {
  border: 1px solid #cecfd5;
}
th,
td {
```



Item		Qty	Price
Don’t Make Me Think by Steve Krug	In Stock	1	$30.02
A Project Guide to UX Design by Russ Unger & Carolyn Chandler	In Stock	2	$52.94 ($26.47 × 2)
Introducing HTML5 by Bruce Lawson & Remy Sharp	Out of Stock	1	$22.23
Bulletproof Web Design by Dan Cederholm	In Stock	1	$30.17
Subtotal			$135.36
Tax			$13.54
Total			$148.90

```
  padding: 10px 15px;
}
```

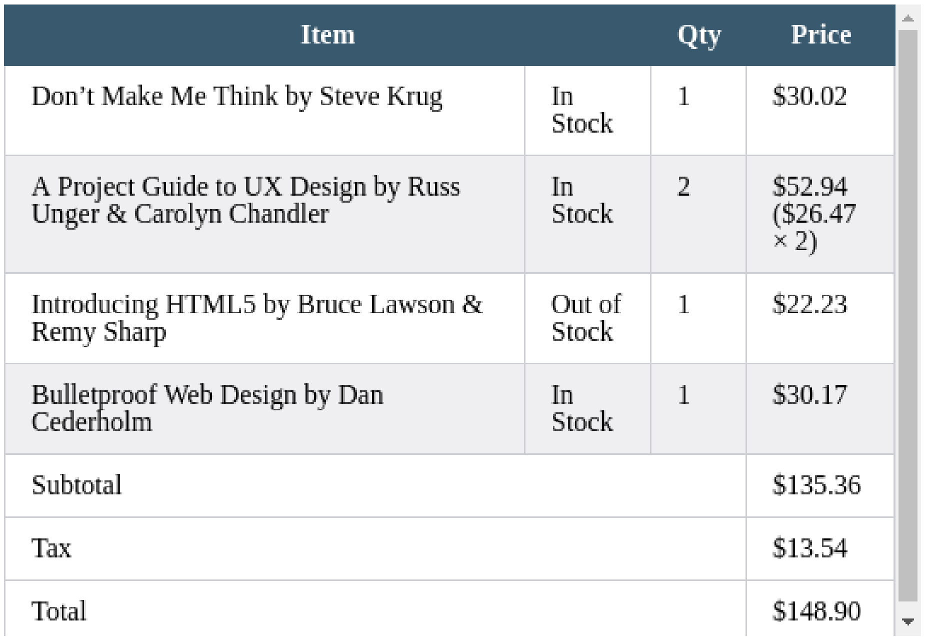
Adding Borders to Rows We’ll begin by making sure the table’s border-collapse property value is set to collapse, and then we’ll add a bottom border to each table cell, regardless of whether it’s a <th> or <td> element. If we wish, we can remove the bottom border from the cells within the last row of the table by using the :last-child pseudo-class selector to select the last <tr> element within the table and target the <td> elements within that row. Additionally, if a table is using the structural elements, we’ll want to make sure to prequalify the last row of the table as being within the <tfoot> element.



Item		Qty	Price
Don’t Make Me Think by Steve Krug	In Stock	1	$30.02
A Project Guide to UX Design by Russ Unger & Carolyn Chandler	In Stock	2	$52.94 ($26.47 × 2)
Introducing HTML5 by Bruce Lawson & Remy Sharp	Out of Stock	1	$22.23
Bulletproof Web Design by Dan Cederholm	In Stock	1	$30.17
Subtotal			$135.36
Tax			$13.54
Total			$148.90

```
table {
  border-collapse: collapse;
}
th,
td {
  border-bottom: 1px solid #cecfd5;
  padding: 10px 15px;
}
tfoot tr:last-child td {
  border-bottom: 0;
}
```

Table Striping In the effort to make tables more legible, one common design practice is to “stripe” table rows with alternating background colors. This makes the rows clearer and provides a visual cue for scanning information. One way to do this is to place a class on every other <tr> element and set a background color to that class. Another, easier way is to use the :nth-child pseudo-class selector with an even or odd argument to select every other <tr> element.

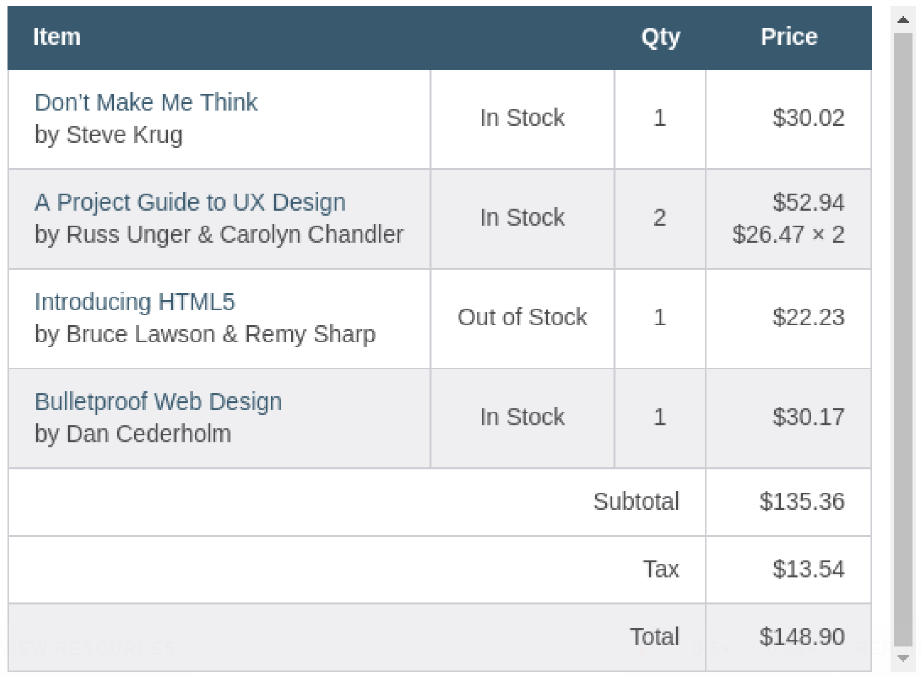


Item		Qty	Price
Don’t Make Me Think by Steve Krug	In Stock	1	$30.02
A Project Guide to UX Design by Russ Unger & Carolyn Chandler	In Stock	2	$52.94 ($26.47 × 2)
Introducing HTML5 by Bruce Lawson & Remy Sharp	Out of Stock	1	$22.23
Bulletproof Web Design by Dan Cederholm	In Stock	1	$30.17
Subtotal			$135.36
Tax			$13.54
Total			$148.90

```
table {
  border-collapse: separate;
  border-spacing: 0;
}
th,
td {
  padding: 10px 15px;
}
thead {
  background: #395870;
  color: #fff;
}
tbody tr:nth-child(even) {
  background: #f0f0f2;
}
td {
  border-bottom: 1px solid #cecfd5;
  border-right: 1px solid #cecfd5;
}
td:first-child {
  border-left: 1px solid #cecfd5;
}
```

Aligning Text To align text vertically, however, the vertical-align property is used. The vertical-align property works only with inline and table-cell elements—it won’t work for block, inline-block, or any other element levels. The vertical-align property accepts a handful of different values; the most popular values are top, middle, and bottom. These values vertically position text in relation to the table cell, for table-cell elements, or to the closest parent element, for inline-level elements. By revising the HTML and CSS to include the text-align and vertical-align properties, we can clean up the layout of our table of books. Note that the data within the table becomes much clearer and more digestible.

```
//html
<table>
```

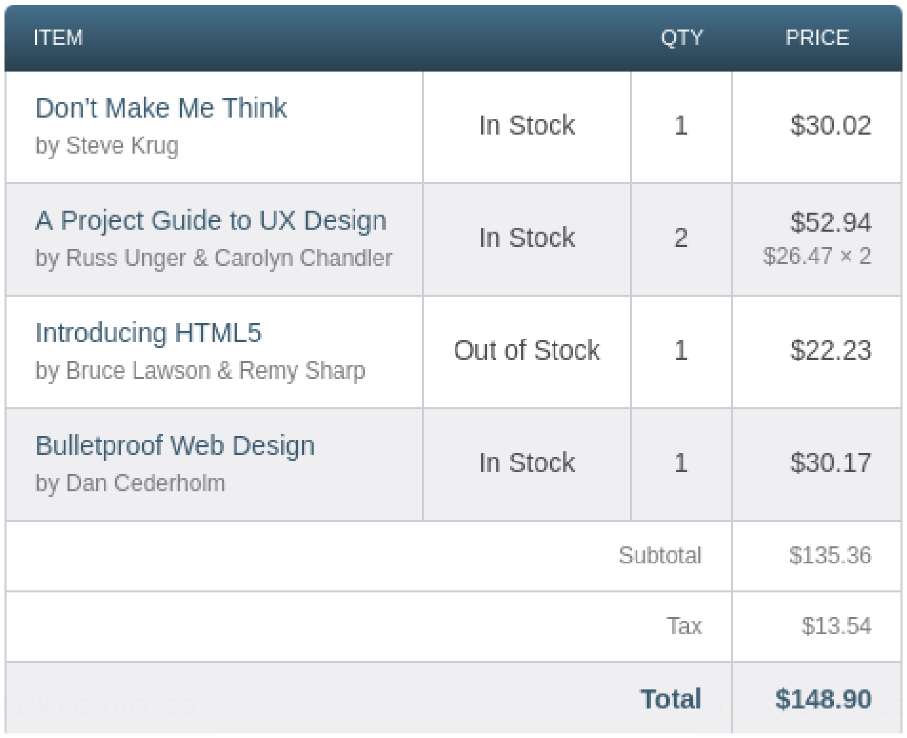


Item		Qty	Price
Don't Make Me Think by Steve Krug	In Stock	1	$30.02
A Project Guide to UX Design by Russ Unger & Carolyn Chandler	In Stock	2	$52.94 $26.47 × 2
Introducing HTML5 by Bruce Lawson & Remy Sharp	Out of Stock	1	$22.23
Bulletproof Web Design by Dan Cederholm	In Stock	1	$30.17
		Subtotal	$135.36
		Tax	$13.54
		Total	$148.90

```
<thead>
  <tr>
    <th scope="col" colspan="2">Item</th>
    <th scope="col">Qty</th>
    <th scope="col">Price</th>
  </tr>
</thead>
<tbody>
  <tr>
    <td>
      <strong class="book-title">Don #8217;t Make Me Think</strong> by Steve Krug
    </td>
    <td class="item-stock">In Stock</td>
    <td class="item-qty">1</td>
    <td class="item-price">$30.02</td>
  </tr>
  <tr>
    <td>
      <strong class="book-title">A Project Guide to UX Design</strong> by Russ Unger  #38; Carolyn Chandler
    </td>
    <td class="item-stock">In Stock</td>
    <td class="item-qty">2</td>
    <td class="item-price">$52.94 <span class="item-multiple">$26.47  #215; 2</span></td>
  </tr>
  <tr>
    <td>
      <strong class="book-title">Introducing HTML5</strong> by Bruce Lawson  #38; Remy Sharp
    </td>
    <td class="item-stock">Out of Stock</td>
    <td class="item-qty">1</td>
    <td class="item-price">$22.23</td>
  </tr>
  <tr>
    <td>
      <strong class="book-title">Bulletproof Web Design</strong> by Dan Cederholm
    </td>
    <td class="item-stock">In Stock</td>
    <td class="item-qty">1</td>
    <td class="item-price">$30.17</td>
  </tr>
</tbody>
<tfoot>
  <tr>
    <td colspan="3">Subtotal</td>
    <td>$135.36</td>
    </tr>
    <tr>
      <td colspan="3">Tax</td>
      <td>$13.54</td>
    </tr>
    <tr>
      <td colspan="3">Total</td>
      <td>$148.90</td>
    </tr>
  </tfoot>
</table>

//css
table {
  border-collapse: separate;
  border-spacing: 0;
  color: #4a4a4d;
  font: 14px/1.4 "Helvetica Neue", Helvetica, Arial, sans-serif;
}
th,
td {
  padding: 10px 15px;
  vertical-align: middle;
}
thead {
  background: #395870;
  color: #fff;
}
th:first-child {
  text-align: left;
}
tbody tr:nth-child(even) {
  background: #f0f0f2;
}
td {
  border-bottom: 1px solid #cecfd5;
  border-right: 1px solid #cecfd5;
}
td:first-child {
  border-left: 1px solid #cecfd5;
}
.book-title {
  color: #395870;
  display: block;
}
.item-stock,
.item-qty {
  text-align: center;
}
.item-price {
  text-align: right;
}
.item-multiple {
  display: block;
}
tfoot {
  text-align: right;
}
tfoot tr:last-child {
  background: #f0f0f2;
}
```

Rounded Corners

```
table {
  border-collapse: separate;
  border-spacing: 0;
  color: #4a4a4d;
  font: 14px/1.4 "Helvetica Neue", Helvetica, Arial, sans-serif;
}
th,
td {
  padding: 10px 15px;
  vertical-align: middle;
}
thead {
  background: #395870;
  background: linear-gradient(#49708f, #293f50);
  color: #fff;
  font-size: 11px;
  text-transform: uppercase;
}
th:first-child {
  border-top-left-radius: 5px;
  text-align: left;
}
th:last-child {
  border-top-right-radius: 5px;
}
tbody tr:nth-child(even) {
  background: #f0f0f2;
}
td {
  border-bottom: 1px solid #cecfd5;
  border-right: 1px solid #cecfd5;
}
td:first-child {
  border-left: 1px solid #cecfd5;
}
.book-title {
  color: #395870;
  display: block;
}
.text-offset {
  color: #7c7c80;
  font-size: 12px;
}
.item-stock,
.item-qty {
  text-align: center;
}
.item-price {
  text-align: right;
}
.item-multiple {
  display: block;
}
tfoot {
  text-align: right;
}
tfoot tr:last-child {
  background: #f0f0f2;
  color: #395870;
  font-weight: bold;
}
tfoot tr:last-child td:first-child {
  border-bottom-left-radius: 5px;
}
tfoot tr:last-child td:last-child {
  border-bottom-right-radius: 5px;
}
```



ITEM		QTY	PRICE
Don't Make Me Think by Steve Krug	In Stock	1	$30.02
A Project Guide to UX Design by Russ Unger & Carolyn Chandler	In Stock	2	$52.94 $26.47 × 2
Introducing HTML5 by Bruce Lawson & Remy Sharp	Out of Stock	1	$22.23
Bulletproof Web Design by Dan Cederholm	In Stock	1	$30.17
		Subtotal	$135.36
		Tax	$13.54
		Total	**$148.90**