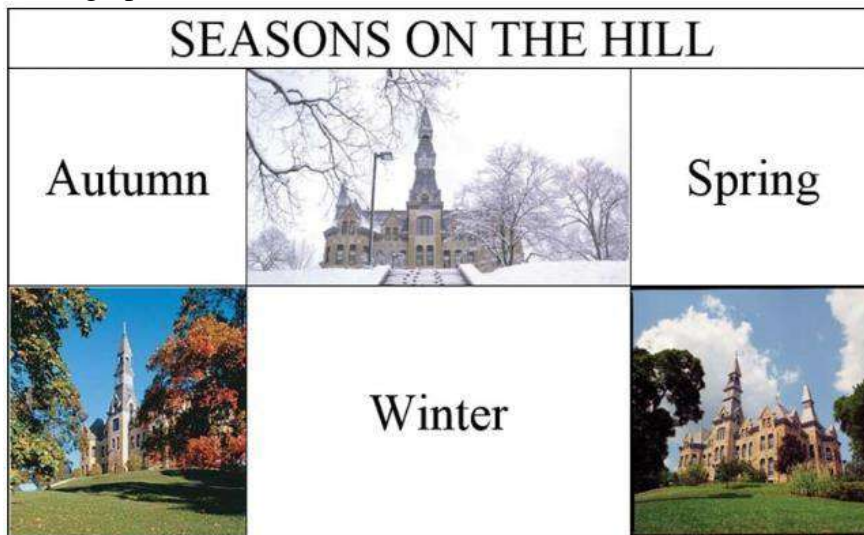


Module-4: Tables and CSS, Links and Images

At its core, a table is a group of cells organized in a two-dimensional structure with rows and columns. Normally, we think of a table's cells as holding data, but tables can also be used purely for presentation purposes. To use a table for presentation purposes, you position content at particular locations on the web page using a row-column layout scheme.

Data tables very often hold numbers, but they can hold text and other types of content as well.

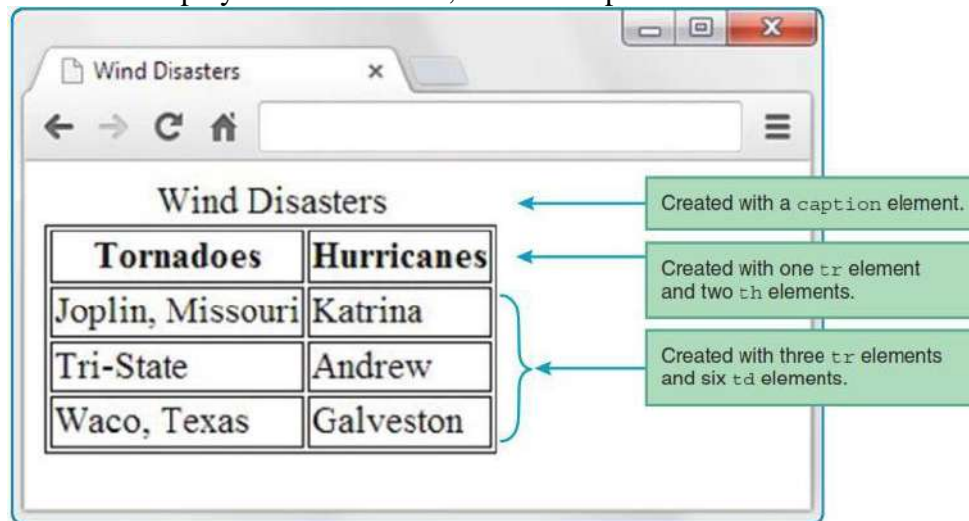
Unlike data tables, layout tables are not limited to holding data—they are allowed to hold any type of content. Their purpose is to position that content with a row-column layout scheme. Consider this graphic:



We begin this chapter by describing data tables and the HTML table elements used to implement them such as table, tr, th, td, and so on.

Table Elements

Let's start by looking at a simple data table—the wind disasters table in [figure 5.1](#). It's a data table in that it displays data, the names of famous tornadoes and hurricanes, in a row-column format. To create a data table, start with a table container element, fill the table element with a tr element for each of its rows, and fill each tr element with th elements for header cells and td elements for data cells. If you'd like to display a title for a table, embed a caption element within the table container.



```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="author" content="John Dean">
<title>Wind Disasters</title>
<style>
  table, th, td {border: thin solid;}
</style>
</head>

<body>
<table>
  <caption>Wind Disasters</caption>
  <tr><th>Tornadoes</th><th>Hurricanes</th></tr>
  <tr><td>Joplin, Missouri</td><td>Katrina</td></tr>
  <tr><td>Tri-State</td><td>Andrew</td></tr>
  <tr><td>Waco, Texas</td><td>Galveston</td></tr>
</table>
</body>
</html>

```

Explicitly apply a border around the entire table and around each cell.

Indent all the code that appears within the table start and end tags.

If you include a caption element within a table container, the caption element must be the first element within the table. As you'd expect, a table's caption displays above the table's grid by default. If you want the caption's text displayed at the bottom, you can use the following CSS type selector rule:

```
caption {caption-side: bottom;}
```

By default, browsers use boldface font for table header cells. Use `th` for a cell that is a description for other cells' content; use `td` for all other cells in the table. The table element is a block element. As you learned earlier, unless all the code in a block element can fit on one line.

Formatting a Data Table: Borders, Alignment, and Padding

To specify whether or not you want borders for a table, you should use CSS's border-style property. To specify the border's width, you should use CSS's border-width property. For example, here's the CSS type selector rule used in the Wind Disasters web page:

```
table, th, td {border: thin solid;}
```

Why are there no border-style and border-width properties? That's a trick question. You might recall from Chapter 3 that border is a shorthand property that handles a set of border-related properties. In this example, the border property's first value is thin, which goes with the border-width property, and the border property's second value is solid, which goes with the border-style property. With table, th, and td all listed in the rule, the resulting web page displays a thin solid border around the entire table, around each header cell, and around each data cell.

Now onto the next two formatting features—cell alignment and cell padding. If you add no CSS to a table, then you'll end up using the browser's default CSS values. Table header cells (th) have a default alignment of center and a default weight of bold. Table data cells (td) have a default alignment of left. Both th and td cells have a default padding of none. To adjust the

horizontal alignment of text in table cells, use CSS's text-align property with a value of left, right, or center. To adjust the padding around the text in table cells, use CSS's padding property with a pixel value (e.g., 5px).

Look at the Wind Disasters web page in [FIGURE 5.3](#). Note how the header text ("Tornadoes" and "Hurricanes") is left aligned. Note how there's padding around every cell's text. Also note the border widths—the outer border is thicker than the cell borders, and there's an even thicker border below the header cells.

Have you got the CSS figured out? To add left alignment and padding to every cell, use this type selector rule:

```
th, td {
    text-align: left;
    padding: 10px;
}
```

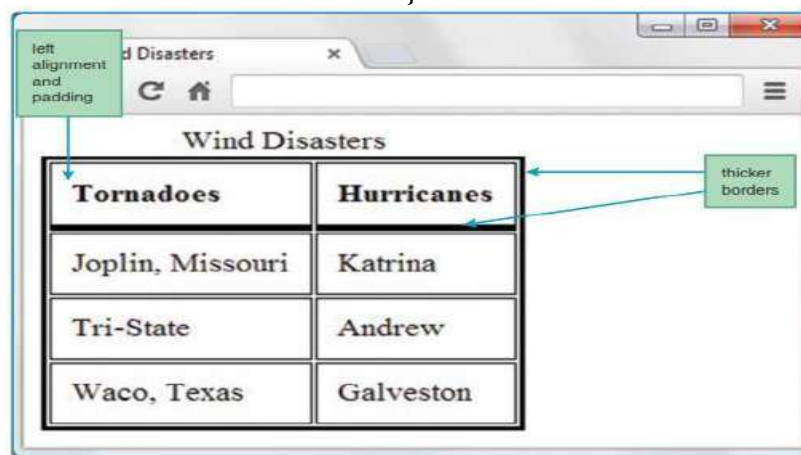


Figure 5.3 Wind Disasters web page with improved formatting

Figure 5.2's style container contains the following CSS rule, which creates thin border lines around the entire table and also around each cell:

```
table, th, td {border: thin solid;}
```

To assign a medium width to the table's outer border and to assign a thick width to the header cells' bottom borders, add the following type selector rules below the rule:

```
table {border-width: medium;}
th {border-bottom-width: thick;}
```

Note that all three rules deal with borders. The first and second rules both provide values for the table element's border-width property. The second rule's border-width property is explicit (border-width: medium), whereas the first rule's border-width property is built into the border shorthand property (border: thin). So will the table's border width be medium or thin? If two CSS rules refer to the same property, the rule that appears later overrides the prior rule's property value. Therefore, because the border-width: medium rule appears later, it wins and the table's border width will be medium.

In Figure 5.3, note how there's a gap between each of the borders. More specifically, there's a gap between the table's exterior border and the individual cells' borders, and there's a

gap between the borders for adjacent cells. If you'd like to eliminate those gaps and merge the borders, use the `border-collapse` CSS property with a value of `collapse`, like this:

```
table { border-collapse: collapse; }
```

CSS Structural Pseudo-Class Selectors

When you have a collection of elements, sometimes you want to display one or more of those elements differently from the rest. You could do that by including `class` attributes in each element that you want to display differently. You would then use the `class` attribute's value as a class selector in a CSS rule. But if the number of elements that you want to display with a special format is large, then quite a few `class="value"` code insertions would be required.

When there is regularity in the locations of certain elements within a collection of elements, you can avoid the `class="value"` code insertions described and, instead, implement that functionality with a structural pseudo-class CSS rule. Pseudo-classes conditionally select elements from a group of elements specified by a standard selector. For example, the following code uses a standard `tr` type selector to select all the `tr` elements in a web page, and the `:first-of-type` pseudo-class checks each of those elements to see if it is a first `tr` element within a particular table.

```
tr:first-of-type { background-color: palegreen; }
```

A pseudo-class is called a “pseudo-class” because using a pseudo-class is similar to using a class attribute, but the two entities are not identical. A pseudo-class is like a class selector in that it matches particular instances of elements (that's what happens with elements that use `class` attributes). But they are different from class selectors in that they don't rely on the `class` attribute.

we formally say that the pseudo-class checks for a first `tr` element from among a group of sibling `tr` elements. Sibling elements are elements that have the same parent element. An element is considered to be a parent of another element if it contains the other element with just one level of nesting. This mimics the notion of a human parent. A human is a parent of its children, but is not a parent of its grandchildren. the most popular structural pseudo-classes they are `:first-of-type`, `:last-of-type`, and `:nth-of-type()`.

`:last-of-type`.

As you might guess, the `:last-of-type` pseudo-class checks each of the elements selected by a standard selector to see if the element is a last element from among a group of sibling elements. So the following example selects `li` elements that are at the bottom of unordered lists:

```
ul > li:last-of-type { background-color: palegreen; }
```

Pseudo-Class	Description
<code>:first-of-type</code>	Selects first element in sibling group of a particular type.
<code>:last-of-type</code>	Selects last element in sibling group of a particular type.
<code>:nth-of-type()</code>	Uses parentheses value to select an element or group of elements.

FIGURE 5.4 Popular structural pseudo-class selectors

Note the `ul > li` child selector notation, which means that an `li` element is selected only if it is a child of a `ul` element. Note that there are no spaces on either side of the pseudo-class's colon. That's a style rule.

:nth-of-type(). Unlike the other pseudo-classes so far, the :nth-of-type() pseudo-class has parentheses. Inside the parentheses, you provide a value that indicates which element or group of elements you want to select. For example, in the following CSS rule, we put 3 in the parentheses to select the third data cell within a row of sibling data cells.

```
td:nth-of-type(3) {text-align: right;}
```

That rule matches every third td element within each row of td elements and causes those td elements to be right-aligned. Effectively, that causes tables to display their third columns with right-aligned data. For example, in the following CSS rule, we put even in the parentheses to select every even-numbered row:

```
tr:nth-of-type(even) {background-color: lightblue;}
```

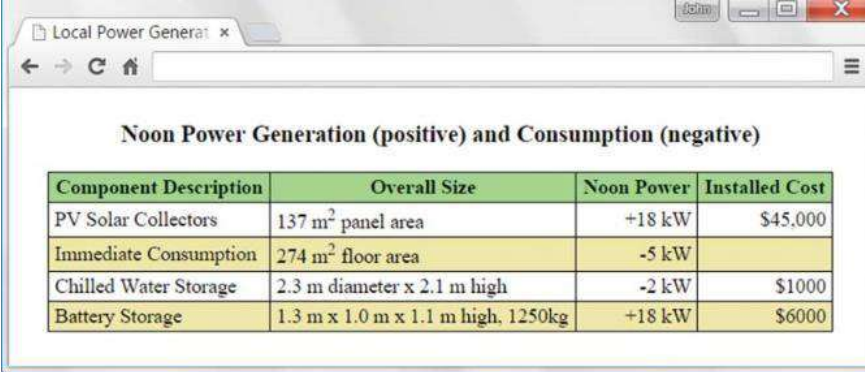
As an alternative to putting a number, even, or odd in the parentheses, you can use an expression of the form $a + b$, where a and b are constant integers and n is a variable named n . By using such an expression, you can specify interleaved groups of elements. This is better explained.

```
tr:nth-of-type(5n+2) {background-color: red;}
```

That rule selects every fifth tr element starting with the second row. In other words, it selects rows 2, 7, 12, 17, and so on

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="author" content="John Dean">
<title>Local Power Generation and Consumption</title>
<style>
  h3 {text-align: center;}
  table {border-collapse: collapse; margin: 0 auto;}
  th, td {border: thin solid; padding: 2px 5px;}
  td:nth-of-type(3), td:nth-of-type(4) {text-align: right;}
  tr:first-of-type {background-color: palegreen;}
  tr:nth-of-type(2n+3) {background-color: palegoldenrod;}
</style>
</head>
<body>
<table>
  <caption>
    <h3>Moon Power Generation (positive) and Consumption (negative)</h3>
  </caption>
  <tr>
    <th>Component Description</th> <th>Overall Size</th>
    <th>Moon Power</th> <th>Installed Cost</th>
  </tr>
  <tr>
    <td>FV Solar Collectors</td> <td>137 m<sup>2</sup> panel area</td>
    <td>18 kW</td> <td>$45,000</td>
  </tr>
  <tr>
    <td>Immediate Consumption</td> <td>274 m<sup>2</sup> floor area</td>
    <td>-5 kW</td> <td></td>
  </tr>
  <tr>
    <td>Chilled Water Storage</td> <td>2.3 m diameter x 2.1 m high</td>
    <td>-2 kW</td> <td>$1000</td>
  </tr>
  <tr>
    <td>Battery Storage</td> <td>1.3 m x 1.0 m x 1.1 m high, 1250kg</td>
    <td>18 kW</td> <td>$6000</td>
  </tr>
</table>
</body>
</html>
```

FIGURE 5.5 Source code for Power Table web page



Component Description	Overall Size	Noon Power	Installed Cost
PV Solar Collectors	137 m ² panel area	+18 kW	\$45,000
Immediate Consumption	274 m ² floor area	-5 kW	
Chilled Water Storage	2.3 m diameter x 2.1 m high	-2 kW	\$1000
Battery Storage	1.3 m x 1.0 m x 1.1 m high, 1250kg	+18 kW	\$6000

Figure 5.6 power table web page

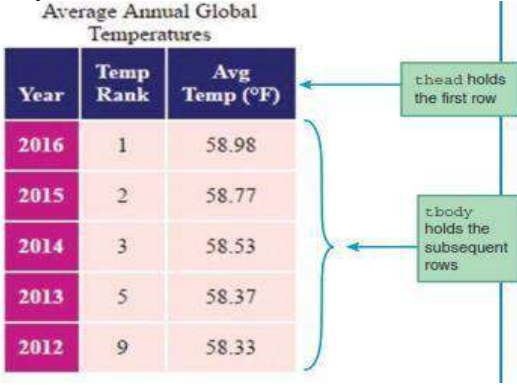
thead and tbody Elements

Normally, you'll put table header cells at the top of a table's columns, but sometimes you'll also want to put them at the left of each row. For example, in the Global Temperatures web page in Figure 5.7, note the year values in header cells at the left. If you have header cells at the left, very often you'll want to differentiate those header cells from the ones at the top. The preferred way to differentiate is to put the top cells' row (or rows) in a `thead` element and put the subsequent rows in a `tbody` element.

Take a look at the Global Temperatures `thead` and `tbody` code in Figure 5.8. The `thead` element contains a `tr` element and three `th` elements within the `tr` element. The `tbody` element contains several `tr` elements, with each `tr` element holding a `th` element and two `td` elements. Here are simplified versions of the descendant selector rules used to color `thead`'s header cells differently from `tbody`'s header cells:

```
thead th {background-color: midnightblue;}
tbody th {background-color: mediumvioletred;}
```

Besides using `thead` and `tbody`, there are other ways to distinguish the top header cells from the left-side header cells. For example, you could use class attributes with one value for the top header cells and a different value for the left-side header cells. However, that would lead to cluttered code—a class attribute for every the cell.



Year	Temp Rank	Avg Temp (°F)
2016	1	58.98
2015	2	58.77
2014	3	58.53
2013	5	58.37
2012	9	58.33

Figure 5.7 Global temperatures web page

Let's now examine a few noteworthy CSS rules from the Global Temperatures web page that are unrelated to `thead` and `tbody`. Here's the first such rule:

```
body {display: flex; justify-content: center;}
```

In the CSS rule, the `display: flex;` property-value pair creates a flexbox layout (also called a flexible box layout).

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="author" content="John Dean">
<title>Global Temperatures</title>
<style>
  body {display: flex; justify-content: center;}
  table, th, td {border: none;}
  th, td {padding: 10px;}
  thead th {
    background-color: midnightblue;
    color: white;
    vertical-align: bottom;
  }
  tbody th {
    background-color: mediumvioletred;
    color: white;
  }
  td {
    background-color: mistyrose;
    text-align: center;
  }
</style>
</head>
<body>
<table>
  <caption>Average Annual Global Temperatures</caption>
  <thead>
    <tr>
      <th>Year</th>
      <th>Temp<br>Rank</th>
      <th>Avg<br>Temp (&deg;F)</th>
    </tr>
  </thead>
  <tbody>
    <tr><th>2016</th><td>1</td><td>58.98</td></tr>
    <tr><th>2015</th><td>2</td><td>58.77</td></tr>
    <tr><th>2014</th><td>3</td><td>58.53</td></tr>
    <tr><th>2013</th><td>5</td><td>58.37</td></tr>
    <tr><th>2012</th><td>9</td><td>58.33</td></tr>
  </tbody>
</table>
</body>
</html>

```

To center a block element (like table), apply this CSS code to the element's parent container.

To position text vertically within its container, use the vertical-align property.

If a row's content is too long to fit on one line, then put indented cell elements on separate lines.

The code shows two values for the margin property—0 and auto.

```
table {margin: 0 auto;}
```

Using a bottom value for the vertical-align property causes a cell's text to be aligned at the bottom.

```
thead th {vertical-align: bottom;}
```

The border: none property-value pair means that the browser will not draw border lines.

```
table, th, td {border: none;}
```

This is a merged cell that uses rowspan="2".

This is a merged cell that uses colspan="2".

Eras	Events	
Mesozoic 251 to 65.5 mya	Evolutionary split between reptiles and dinosaurs	
	South America breaks away from Africa	
Cenozoic 65.5 mya to today	Modern mammals appear	
	Tool-making humanoids appear	
	First Rolling Stones reunion tour	

Cell Spanning

If you want to create a merged cell that spans more than one column, you'll need to add a `colspan` attribute to a `th` or `td` element. Figure 5.10 shows the code for the My Favorite Eras web page. In particular, examine the code for the table's first row, and note `colspan="2"`, which creates a merged cell that spans two columns.

```
<tr><th>Eras</th><th colspan="2">Events</th></tr>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="author" content="John Dean">
<title>Earth Eras</title>
<style>
  table {border: thin solid;}
  th, td {border: thin solid; padding: 10px;}
  thead th {background-color: lawngreen;}
  tbody th {background-color: lightcyan;}
</style>
</head>

<body>
<table>
  <caption>My Favorite Eras</caption>
  <thead>
    <tr><th>Eras</th><th colspan="2">Events</th></tr>
  </thead>
  <tbody>
    <tr>
      <th rowspan="2">Mesozoic<br>251 to 65.5 mya</th>
      <td>Evolutionary split between reptiles and dinosaurs</td>
      <td>235 mya</td>
    </tr>
    <tr>
      <td>South America breaks away from Africa</td>
      <td>105 mya</td>
    </tr>
    <tr>
      <th rowspan="3">Cenozoic<br>65.5 mya to today</th>
      <td>Modern mammals appear</td>
      <td>40 mya</td>
    </tr>
    <tr><td>Tool-making humanoids appear</td><td>2 mya</td></tr>
    <tr>
      <td>First Rolling Stones reunion tour</td>
      <td>11,000 years ago</td>
    </tr>
  </tbody>
</table>
</body>
</html>
```

The image includes two green callout boxes with arrows pointing to specific code elements: "colspan attribute" points to `colspan="2"` in the first row of the table, and "rowspan attribute" points to `rowspan="2"` in the Mesozoic header cell.

Figure 5.10 Source code for My Favorite eras web page

If you want to create a merged cell that spans more than one row, add a `rowspan` attribute to the cell's `th` or `td` element. Thus, as shown in Figure 5.9, the Mesozoic header cell spans two rows.

Web Accessibility

Web accessibility means that disabled users can use the Web effectively. Most web accessibility efforts go toward helping users with visual disabilities, but web accessibility also attempts to address the needs of users with hearing, cognitive, and motor skills disabilities.

To promote the social good (through equal opportunities for disabled people) and to promote their own businesses, many companies have policies that require web developers and software purchasers to follow web accessibility standards.

Typically, visually impaired users have screen readers to read web pages. A screen reader is software that figures out what the user's screen is displaying and sends a text description of it to a speech synthesizer. The speech synthesizer then reads the text aloud.

Screen readers rely on the fact that most data tables have header cells in the first row or the first column. When screen readers see such "simple" data tables, they assume that each header in the first row describes the data cells that are below it. Likewise, if there are headers in the first column, screen readers assume that each of those headers describes the data cells that are at the right of the header.

If you have a data table in which one or more header cells are not in the first row or column (i.e., it's not a simple table), then you should consider adding code to make the table more web accessible. The details element provides a description of the table's content so that a screen reader can read the description and get a better understanding of the nature of the table's organization. The Grading Weights table in [figure 5.11](#) has header cells in its second row, so the table is a good candidate for a web accessibility makeover. In the figure, note the right-facing triangle under the table's title. If the user clicks the triangle, the browser will display "help" details that describe the table's content. The triangle and the text that describes the table both come from a details element embedded in the table's caption element.

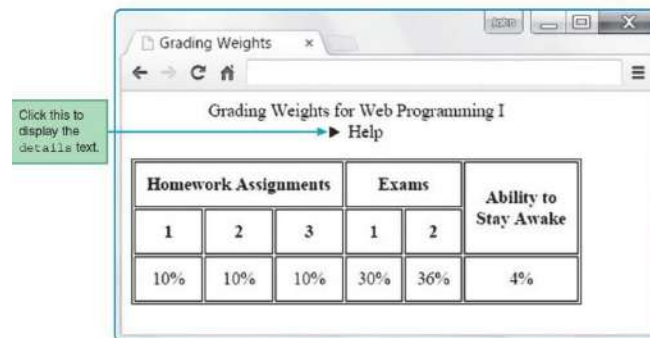


Figure 5.11 Grading Weights web page

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="author" content="John Dean">
<title>Grading Weights</title>
<style>
  table, th, td {border: thin solid;}
  th, td {
    text-align: center;
    padding: 10px;
  }
  caption {margin-bottom: 15px;}
</style>
</head>
<body>
<table>
  <caption>
    Grading Weights for Web Programming I
    <details>
      <summary>Help</summary>
      The first 3 columns show weights for the 3 homework assignments.
      The next 2 columns show weights for the 2 exams.
      The last column shows the weight for staying awake during class.
    </details>
  </caption>
```

Figure 5.12a Source code for Grading Weights web page

That tells the screen reader that the table is for presentation/layout purposes, not for storing data,
`<table role="presentation">`

```
<tr>
  <th colspan="3" id="hw">Homework Assignments</th>
  <th colspan="2" id="exams">Exams</th>
  <th rowspan="2" id="awake">Ability to<br>Stay Awake</th>
</tr>
<tr>
  <th id="hw1" headers="hw">1</th>
  <th id="hw2" headers="hw">2</th>
  <th id="hw3" headers="hw">3</th>
  <th id="exam1" headers="exams">1</th>
  <th id="exam2" headers="exams">2</th>
</tr>
<tr>
  <td headers="hw hw1">10%</td>
  <td headers="hw hw2">10%</td>
  <td headers="hw hw3">10%</td>
  <td headers="exams exam1">30%</td>
  <td headers="exams exam2">36%</td>
  <td headers="awake">4%</td>
</tr>
</table>
</body>
</html>
```

FIGURE 5.12B Source code for Grading Weights web page

CSS display Property with Table Values

There are two main ways to implement layout tables with CSS. If you want the layout boundaries to grow and shrink the way they do for an HTML table element, then use the CSS `display` property with table values. On the other hand, if you want the layout boundaries to be fixed (no growing or shrinking), then use CSS position properties.

The `display` property's table Values

`display`: inline property-value pair to display an address element in the flow of its surrounding sentence. The `display` property can be used for much more than just inlining block element content. It can also be used to emulate the various parts of a table. The table value enables an element, like a `div` element, to behave like a table. Here's how you can do that:

```
<style>
  .table { display: table; }
  ...
</style>
<body>
  <div class="table">
    ...
  </div>
</body>
```

Table Values for the <code>display</code> Property	Description
<code>table</code>	Used to mimic a <code>table</code> element.
<code>table-caption</code>	Used to mimic a <code>caption</code> element.
<code>table-row</code>	Used to mimic a <code>tr</code> element.
<code>table-cell</code>	Used to mimic a <code>td</code> element or a <code>th</code> element.
<code>table-header-group</code>	Used to mimic a <code>thead</code> element.
<code>table-row-group</code>	Used to mimic a <code>tbody</code> element.

FIGURE 5.13 Table values for the CSS `display` property

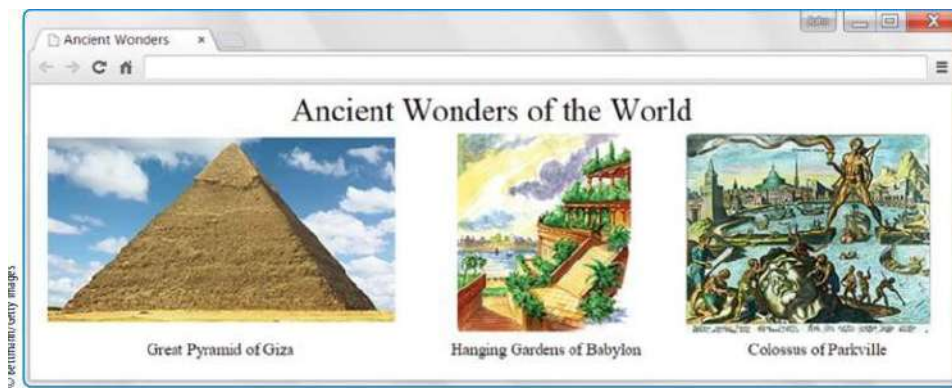


FIGURE 5.14 Ancient Wonders web page

you're supposed to use the table element only for data tables and not for table layout.

Example Web page

The pictures and the labels under the pictures are displayed using a two-row, three-column layout scheme. You could implement that layout scheme with either a table element or with CSS, so which is more appropriate? Because the web page does not contain data, you should use CSS.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="author" content="John Dean">
<title>Ancient Wonders</title>
<style>
  .table {
    display: table;
    border-spacing: 20px;
  }
  .caption {
    display: table-caption;
    font-size: xx-large;
    text-align: center;
  }
  .row {display: table-row;}
  .row > * {
    display: table-cell;
    text-align: center;
  }
</style>
</head>
<body>
<div class="table">
  <div class="caption">Ancient Wonders of the World</div>
  <div class="row">
    <span></span>
    <span></span>
    <span></span>
  </div>
  <div class="row">
    <span>Great Pyramid of Giza</span>
    <span>Hanging Gardens of Babylon</span>
    <span>Colossus of Parkville</span>
  </div>
</div>
</body>
</html>
```

For table behavior.

For caption behavior.

For tr behavior.

This causes all children of .row elements to behave like table cells (i.e., td or th elements).

This causes this div element to behave like a table.

This causes this div element to behave like a caption.

This causes this div element to behave like a row.

FIGURE 5.15 Source code for Ancient Wonders web page

In figure 5.15's Ancient Wonders style container, note the CSS rules that use the .table, .caption, and .row class selectors. In the body container, note how those selector names are used to implement a table using div elements—<div class="table">, <div class="caption">, and <div class="row">. To implement table cells with the display property, you need to use the table-cell value.

```
img {display: table-cell;}
```

The img element is a void element, not a container, so it's inappropriate to try to use CSS to turn it into a table cell. The solution is to surround the img elements with span containers and use span as the target for a table-cell rule.

```
span {display: table-cell;},
```

If we were to use the CSS rule, then every span element would be implemented as a table cell. That's OK for the current version of the Ancient Wonders web page, but as a web developer, you should think about making your web pages maintainable. That means you should accommodate the possibility that you or someone else adds to your web page sometime in the future. If a span element is added that's not part of a table, the CSS rule would attempt to make it behave like a table cell.

We use a child selector rule that matches every element that is a child of a row element. This is justified because it's reasonable to assume that within a row element, every child element is a data cell.

```
.row > * {display: table-cell;}
```

You might recall that the > symbol is known as a combinator because it combines two selectors into one. The selector at the left, .row, matches all the elements in the Ancient Wonders web page that have . The universal selector at the right, *, matches any element. When the two selectors are combined with >, the resulting child selector matches every element that is a child of a row element.

the border-spacing property

By default, tables created with the display property are displayed with no gaps between their cells. For the Ancient Wonders web page, that default behavior would have led to pictures that were touching. To avoid that ugliness, you can use the border-spacing property.

```
.table {  
  display: table;  
  border-spacing: 20px;  
}
```

a Element

To implement a link, you'll need to use the a element. Here's an example a element that implements a link to Park University's website:

```
<a href="http://www.park.edu">Park University</a>
```

The text that appears between an a element's start tag and end tag forms the link label that the user sees and clicks on. So in this code, the link label is "Park University." By default, browsers display link labels with underlines. So this code renders like this

[Park University](#)

The blue color indicates that the linked-to page has not been visited. We'll discuss visited and unvisited links later on. When the user clicks on a link, the browser loads the resource specified by the href attribute. For this example, the "resource" is another web page, so when the user clicks on the "Park bute. For this example, the "resource" is another web page, so when the user clicks on the "Park University" link, the browser loads that web page—the one at <http://www.park.edu>.

Besides enabling a user to load a resource (which usually means jumping to another web page), the a element can be used as a mechanism that enables a user to download a file of any type—image file, video file, PDF file, Microsoft Word file, and so on.

```
<a download href="http://www.park.edu/catalogs/catalog2018-2019.pdf">
    Park University 2018-2019 catalog</a>
```

Different types of href Values

Types of href Attribute Values	Where the Link Jumps To
absolute URL	Find the resource on a different web server than the current web page.
relative URL	Find the resource on the same web server as the current web page. Specify the location of the resource by providing a path from the current web page's directory to the destination web page.
jump within current web page	Find the resource within the current web page. Specify the location of the resource by providing an id value for an element in the web page.

FIGURE 6.1 href attribute values

As you know, the a element's href attribute value specifies the resource that is to be loaded. In addition to specifying the resource, the href attribute value indicates where the link jumps to in order to find the resource. Indicating where the link jumps to is not a trivial task.

For an example that uses an absolute URL, suppose you want to add a link on a Facebook page that directs the user to an Instagram page.

```
<a href=" https://www.instagram.com/hannahDavis.html">
    Hannah's Instagram</a>
```

https is another popular protocol that you can use with the href attribute. It stands for hypertext transfer protocol secure. So the https protocol provides more security for communications than does http. To jump to a web page that resides on the same web server as the current web page, for the link's href attribute, use a relative URL. The relative URL specifies a path from the current web page's directory to the destination web page. To jump to a designated location within the current web page, for the link's href attribute, use a value starting with # such that that value matches an id attribute's value for an element in the web page.

Relative URLs

A relative URL value allows you to jump to a web page that resides on the same web server as the current web page. It does so by specifying a path from the current directory to the destination web page. The current directory is the directory where the current web page resides. The destination web page is the page that the user jumps to after clicking on the link. In forming a path for a relative URL value, you'll need to understand how files and directories are organized in a directory tree

structure. Note the example directory tree in figure 6.2. It shows the container relationships between all the files and directories that are within the oleung directory.

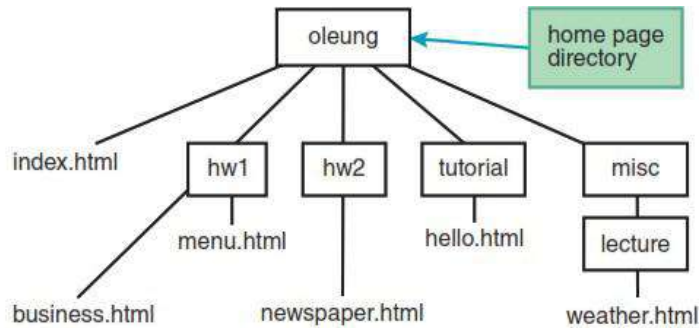


FIGURE 6.2 Example directory tree

The oleung directory is the home page directory for student Olivia Leung. A home page is the default first page a user sees when the user visits a website.

A subdirectory (also called a subfolder or a child folder) is a directory that is contained within another directory.

In forming a path for a relative URL value, you'll need to navigate between a starting directory and a target file. In doing so, you'll need to follow these rules:

- ❖ Use /'s to separate directories and files.
- ❖ Use ".." to go from a directory to the directory's parent directory.

In the second bullet, what does parent directory mean? In Figure 6.2, oleung is the parent directory of hw1 because oleung and hw1 are connected by a single line with oleung on the top. If you're in a directory and you want to go to that directory's parent directory, you need to use .. For example, suppose you want to provide a link on the newspaper page that takes a user to the home page, index.html. Because the newspaper page is in the hw2 directory, the hw2 directory is considered to be the current directory. Here's the relevant code:

```
<a href="../../index.html">Olivia's Home Page</a>
```

Relative path examples

Using Figure 6.2's directory tree, can you try to come up with the code for an `a` element that resides in the index.html file and takes the user to the business page?

```
<a href="hw1/business.html">Business Page</a>
```

Note that href's value starts with hw1. In coming up with the answer on your own, did you start with .. instead? It's a common error among beginning web programmers to feel the need to use .. to go up from the current web page to that web page's directory. Try to avoid that misconception. There's no need to go up. If you're in a web page, then you're already in that web page's directory. The index.html page is the website's home page, and as such, it's the first page that you look at when you visit a website. To help with a user's viewing experience, home pages should normally contain links to other pages on the website.

```
<a href="misc/lecture/weather.html">Weather Page</a>
```

Because the link resides on the home page, the path originates from the home page directory, oleung.

Path-absolute URLs

les, the relative URL's path started at the current web page's directory. As an alternative, you can have the relative URL's path start at the web server's root directory. A web server's root directory

is the directory on the web server that contains all the directories and files that are considered to be part of the web server. URL value that starts with a / is referred to as a path-absolute URL, and it's a special type of relative URL (it's considered a relative URL because the path is relative to the current web server's root directory). The term path-absolute URL comes from the WHATWG.

Navigation Within a Web Page

when you want a link that takes the user to some specified point within the current web page. That can be particularly useful for long web pages, so the user can quickly jump to designated destinations within the web page. For example, note the blue links near the top of the Clock Tower web page in figure 6.6. If the user clicks the Clock Tower Photograph link, then the web page scrolls within the browser window so that the link's target (the clock tower photograph itself) gets positioned at the top of the browser window. If a web page has internal links, then the web page will normally be sufficiently long so as to justify the internal links. Note the Back to the Top link at the bottom of the page. With a long web page, it's common to have such a link so the user can quickly get back to the top after scrolling to the bottom.

Syntax for Internal Link

To jump to a designated location within the current web page, you need to use a value starting with # such that that value matches an id attribute's value for an element in the web page. Here's the Clock Tower web page's code that links to the pledge drive section:

```
<a href="#pledge-drive">Pledge Drive</a>
```

And here's the element that that link jumps to:

```
<h3 id="pledge-drive">Pledge Drive</h3>
```



FIGURE 6.6 Clock Tower web page

Note the spelling for pledge-drive. Standard coding conventions suggest using hyphens to separate multiple words in an id value. For a given web page, you can have only one element with a particular id value. So because pledge-drive appears in this h3 statement, pledge-drive cannot be used as an id value anywhere else within the Clock Tower web page

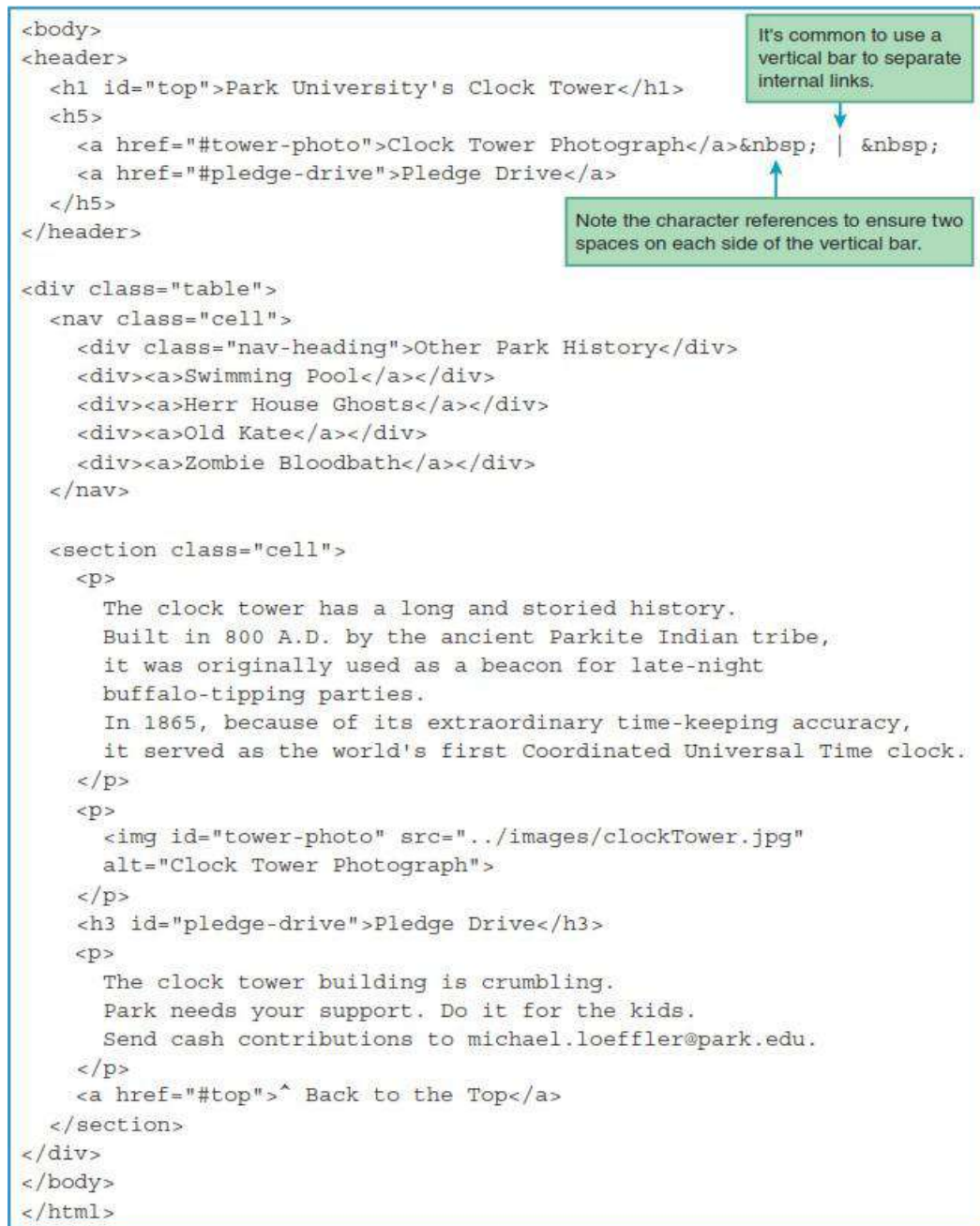


FIGURE 6.7A Source code body container for Clock Tower web page

Walking through the Clock tower Web page's Source Code

Let's start by examining the body container code that separates the tower-photo and pledge-drive links:

 |

Using a vertical bar (|) to separate internal links is a very common technique. The vertical bar has no impact on the web page's functionality; it's just for appearance purposes. By inserting a character reference next to a space character at the left of the vertical bar and also at the right of the vertical bar, a navigation bar is a group of links that enable users to navigate (link to) the various pages on a website.


```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="author" content="John Dean">
<title>Park University Clock Tower</title>
<style>
  header {text-align: center;}
  .table {display: table;}
  .cell {
    display: table-cell;
    padding: 0px 10px 10px;
  }
  nav.cell {width: 80px;}
  /* Avoid yellow background at left extending above the
     page's main content. */
  .cell > :nth-child(1) {margin-top: 0;}
  nav {
    color: darkred;
    background-color: lemonchiffon;
  }
  .nav-heading {background-color: gold;}

  /* Adjust link appearances. */
  a {
    text-decoration: none;
    font-family: Tahoma, Geneva, sans-serif;
  }
  a:hover {text-decoration: underline;}
  nav a {font-size: .8em;}
  nav > * {margin: 1em 0;}
</style>
</head>

```

The .table and .cell rules form a table that holds the navigation bar at the left and the main content at the right.

Fixed width for the navigation area at the left.

:nth-child() selector

FIGURE 6.7B Source code head container for Clock Tower web page

Clock tower Web page's CSS rules

note that there is no .row selector. For a browser to render a table, the cells need to be inside rows, so how can there be no .row selector that implements the rows? Here's the deal: if you have table-cell elements that are not surrounded by a table-row element (or even a table element), then the browser engine generates an anonymous (hidden) table-row element around those table-cell elements. Here's the relevant code from Figure 6.7A:

```

<div class="table">
  <nav class="cell">
    ...
  <section class="cell">
    ...
</div>

```

Anonymous table-row element start tag goes here.

Anonymous table-row element end tag goes here.

Now let's turn to the web page's CSS rules. Here's the CSS rule for the navigation area's cell:

```
nav.cell {width: 80px;}
```

It means that the browser engine locates all nav elements and then matches only those nav elements that use cell as a class attribute value. Now let's talk about its padding property:

```

.cell {
  display: table-cell;
  padding: 0px 10px 10px; }

```

It also requires setting no margin above each of the top elements in the navigation bar cell and the main content area cell. Here's the relevant CSS rule from the Clock Tower web page's code:

```
.cell > :nth-child(1) {margin-top: 0;}
```

As you know, this rule is known as a child selector, where child selectors use the > symbol to match elements that are child elements of other elements. The :nth-child() selector thing is a pseudo-class. Pseudo-class begins with a colon and it conditionally selects elements from a group of elements specified by the selector at the immediate left of the colon.

CSS for Links

Clock Tower Photograph, is blue, indicating that the link has not been clicked. On the other hand, the right link, labeled Clock Tower Photograph, is purple, indicating that the link has been clicked in the past. By default, the major browsers use blue text for unclicked links and purple text for clicked links. More formally, those links are referred to as unvisited links and visited links, respectively. Be aware that end users have the ability to override the link colors specified by the browser by adjusting their browser's settings. But also be aware that as a developer, you have even more power than the user in this regard. You can use CSS to override the link colors specified by the browser, and those CSS rules override the user's link color browser settings as well.

For unvisited links, use this syntax:

```
a:link {color: color-value;}
```

The a is the element type for a link element. The :link thing is a pseudo-class. It qualifies the a element type by searching only for links that have not been visited. As you'd expect, the a:link selector is known as a pseudo-class selector.

For visited links, use a:visited, like this:

```
a:visited {color: color-value;}
```

By default, browsers display links with underlines. If that leads to visual clutter or confusion with regular text that's underlined, and you'd like to have no link underlining, then use text-decoration: none, like this:

```
a {text-decoration: none;}
```

If link underlines are disabled using this CSS rule, but you want to display underlines when the mouse hovers over a link, use the a:hover pseudo-class selector with text-decoration: underline, like this:

```
a:hover {text-decoration: underline;}
```

In case you were wondering, the :hover selector matches any element that is being hovered over, not just links.

```
nav a {font-size: .8em;}
```

```
nav > * {margin: 1em 0;}
```

The first rule specifies a smallish font size of .8em for all link elements in the nav container.

The second rule adds a 1em margin to the top and bottom of all elements that are child elements of the nav container.

```
nav a {cursor: not-allowed;}
```

With that rule in place, hovering your mouse over the placeholder links causes your mouse pointer to change to a blocked icon (⛔), indicating that the link is inactive.

Bitmap Image Formats: GIF, JPEG, PNG

There are two basic categories of image files—bitmap image files and vector graphics files. With bitmap image files, an image is comprised of a group of pixels. For example, an icon, which is simply a small image file, typically has 16 rows with 16 pixels in each row. Within a bitmap image file, every pixel gets mapped to a particular color value, and each color value is a sequence of bits (where a bit is a 0 or a 1⁶). For a browser to display a bitmap image, it displays each pixel's mapped color. The three most common formats for bitmap image files (also called raster image files) on the Web are GIF, JPEG, and PNG.

Bitmap Image Formats	Description
GIF	Good for limited-color images such as line drawings, icons, and cartoon-like illustrations.
JPEG	Good for high-quality photographs.
PNG	Flexible; good for limited-color images and also high-quality photographs.

FIGURE 6.9 Common formats for web page bitmap image files

GIF Image File Format



GIF files use a filename extension of .gif. GIF stands for Graphics Interchange Format. In creating a GIF file from an original picture, the original picture's colors are mapped to an 8-bit palette of colors. That means each pixel uses 8 bits, and those 8 bits determine the pixel's color. And the entire set of colors forms the image's color palette. Each image has its own color palette with its own set of colors. So for the peanut butter image seen here you'd need to capture colors such as red, brown, and blue. Each of those colors would have an 8-bit sequence associated with them, such as 01011010 for red, 11011001 for brown, and 00010110 for blue. With 8 bits for each color value, there are 256 different ways to arrange the 0's and 1's. You can prove this to yourself by writing all the different permutations of eight bits, or you can just remember that the number of permutations is equal to 2 raised to the power of the number of bits, where $2^8 = 256$. That means each GIF image file can handle a maximum of 256 distinct colors. If a picture has more than 256 distinct colors, then when creating the GIF file from the picture, some of the colors won't be stored accurately. Instead they'll be stored with similar colors that are part of the GIF file's color palette, and that leads to the GIF file's image being a degraded version of the original photograph. Color degradation is nonexistent or imperceptible for limited-color images such as line drawings, icons, or cartoon-like illustrations, and that's why GIF files are good for those types of things and not good for color photographs. By using only 8 bits of storage for each pixel, GIF files are able to achieve relatively small file sizes. A fun feature of the GIF file format is that it supports simple animation. In rendering an animated GIF, the browser displays a sequence of image frames in quick succession.

JPEG Image File Format

JPEG stands for Joint Photographic Experts Group, and JPEG is pronounced "jay-peg." JPEG files use a filename extension of .jpeg or .jpg. In creating a JPEG file from an original picture, the

original picture's colors are mapped to a 24-bit palette of colors. With 24 bits, there are approximately 16 million permutations of 0's and 1's in each color value ($2^{24} = 16,777,216$). That means approximately 16 million unique colors can be represented, and that's more colors than the human eye can discern. Note this picture of the New York Catskills' famed fall foliage:



a GIF file would have to approximate many of the colors in order to conform to the constraints of its relatively small color palette. But with the JPEG format, all the photograph's original colors can be displayed accurately. JPEG images can vary widely in terms of color degradation and file size. Nonetheless, for photographs, optimized JPEG files almost always produce a better balance between file size and quality than the other file formats.

PNG Image File Format

PNG stands for Portable Network Graphics, and PNG is pronounced “ping.” PNG files use a filename extension of .png. The PNG format was invented in 1996 as an open-source alternative to the GIF format because the GIF format was copyright protected with a patent owned by Unisys. So each time someone made a new GIF file, they were supposed to pay a license fee to Unisys. Oftentimes, GIF file creators didn't bother to pay the license fee, which was illegal. To avoid such illicit activity, the web community eventually invented their own open-source format for image files, and the PNG format was born. After the PNG format's inception in 1996, the GIF copyright expired, so it's now legal to create and use GIF files without fear of retribution. The PNG format provides more flexibility in terms of clarity versus file size. In creating a PNG file from an original picture, you can choose to map each pixel to only a few bits all the way up to 64 bits per pixel. The PNG format provides more flexibility in terms of transparency. You can create images with different levels of transparency for different parts of an image. GIF images can have only two levels of transparency—completely opaque or completely transparent. PNG images can have 256 levels of transparency.

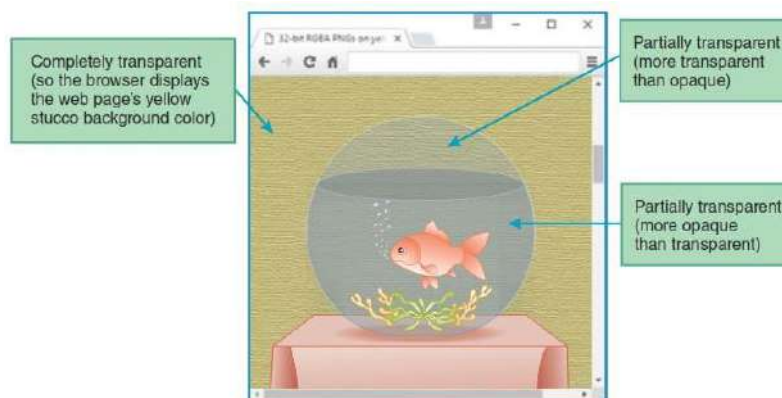


FIGURE 6.10 Aquarium PNG image with multiple levels of transparency

Note the web page in [figure 6.10](#). In particular, note the yellow stucco background behind the aquarium. The background comes from the web page and not from the image file. You can't see the image's original area outside of the aquarium (and the image's rectangular edges) because it's covered over by the web page's background. The browser is able to "cover over" that area because the PNG file is configured to be transparent there. The PNG format allows that sort of complete transparency, but it also allows partial transparency. In the figure, notice the partially transparent glass near the top of the aquarium and the partially transparent water within the aquarium.

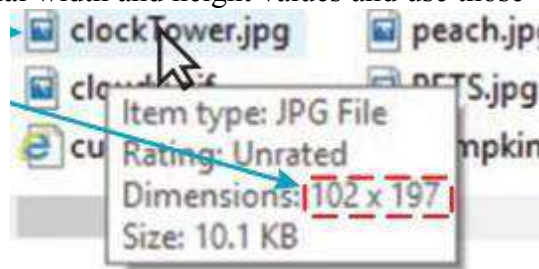
img Element

Let's jump right into an img element example:

```

```

The src attribute specifies the image's filename and the location of the image's file. In this img element, the filename is winkingSmiley.gif. The location is indicated by the path that comes before the filename. The path that comes before winkingSmiley.gif is ../images/. Do you remember what the .. is for? The .. says to move up to the parent directory of the current directory. After .. comes /images, which says to go down to the images directory. That's where the winkingSmiley.gif file is supposed to reside. If the image file and current web page are in the same directory, then there's no need for a path. The alt attribute's value should normally be a description of the picture. The alt stands for "alternative" because it provides an alternative for displaying the image in case the image is unviewable. Formally, we refer to the alt attribute as a fallback mechanism because it provides content for the browser to fall back on if the image can't be displayed. That content is referred to as fallback content. The image might be unviewable. There could be a problem with the file, such as it might not exist, it might be corrupted, or it might be in a different location than what's specified in the src attribute's path. Other reasons for not being able to view a file have nothing to do with the file; they have to do with the user. To make web page downloads faster, users might disable the ability to download images on their browsers. An img element's width and height attributes specify the image's size in pixels. In implementing an img element, you should find the image's actual width and height values and use those values for the img element's width



and height attributes.

Responsive Images

Users view web pages on different types of platforms—desktops, laptops, tablets, and smartphones. If you want your code to be "responsive" (i.e., you want it to dynamically generate different layouts and images for different platforms), you'll need to learn responsive web design. Responsive web design (RWD) is the practice of writing code that dynamically generates web pages that conform to different screen sizes and viewing orientations (portrait or landscape). We'll describe how to implement responsive images using the resolution switching technique and the art direction technique.



FIGURE 6.15 How a responsive web page's layout can display differently on different platforms

resolution Switching

Resolution switching is when you provide a list of images for different versions of the same picture where the images are identical in terms of aspect ratio, where aspect ratio is the ratio of an image's width to height. Figure 6.16 shows the `img` element for a web page that employs resolution switching to display one of three different pictures on a web page. Note the `srcset` attribute, which provides a comma-separated list of image filenames with an image width next to each filename.

```

```

The `srcset` attribute holds the image filenames and their widths.

The `sizes` attribute holds viewport width conditions and associated image-slot widths.

FIGURE 6.16 Code fragment for resolution-switching `img` element

In Figure 6.16, note the `sizes` attribute. Its value helps the browser choose the most appropriate file from among the `srcset` attribute's list of image files. The `sizes` attribute provides a comma-separated list of values. Each value has two parts—a condition that checks the width of the browser window's viewport and the width of the slot in which the image displays. The viewport is the area below the address bar where web page content displays. As you can imagine, mobile devices have the smallest viewports, whereas laptops and desktop monitors have the largest viewports.

From Figure 6.16, here's the `sizes` attribute's first value:

max-width: 340px) 90vw

The max width: 340px condition means that if the device's viewport is less than or equal to 340 pixels, then the web page uses 90vw for its image slot width. The vw unit stands for viewport window, and it's used for specifying an image slot width as a percentage of the viewport's width. The 90vw value means that the image slot width spans 90% of the viewport's width. In the figure, the `sizes` attribute's third value has just one part—the width of the slot in which the image displays. There's no viewport width condition because the last value in the list serves as the default. If the viewport width does not match any of the prior conditions, then the default image slot width applies.

Art Direction

For both resolution switching and art direction, you provide a list of images for different versions of the same picture. With resolution switching, the different versions are different sizes, but they

have the same aspect ratio. On the other hand, with art direction, the different versions can have different aspect ratios as a result of cropping the original picture in different ways.

To implement responsive images with the art direction technique, you wrap the images in a picture container. Specifically, the picture container holds a group of source elements, where each source element value has two parts—a condition that checks the width of the browser window's viewport and an image filename. The source element's media attribute provides the condition that checks the viewport's width, and the source element's srcset attribute provides the image filename. In the figure, the first source element uses the condition `max-width: 799px` to check for a viewport width of less than or equal to 799 pixels. The second source element uses the condition `max-width: 800px` to check for a viewport width of greater than or equal to 800 pixels.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="author" content="John Dean">
<title>Dominican Republic</title>
</head>
<body>
<h1>Joe Hartman School in the Dominican Republic</h1>
<picture>
  <source media="(max-width: 799px)" srcset="../images/drKids-450w.jpg">
  <source media="(min-width: 800px)" srcset="../images/drKids-800w.jpg">
  
</picture>
<p>
  During this past spring break, Jordan participated in a service trip to
  the Dominican Republic. One of the group's primary activities was
  helping to build an annex to a grade school. In the above picture, you
  can see Jordan with some of the kids. She's the one wearing sunglasses.
</p>
</body>
</html>
```

The media attribute holds a viewport width condition.

The srcset attribute holds an image filename.

FIGURE 6.17 Source code for Dominican Republic Kids web page



FIGURE 6.18 Dominican Republic Kids web page on a desktop monitor



In processing a picture element, the browser checks for the first source element whose condition is true and loads that source element's associated image file. The `img` element's `src` attribute provides an image file that the browser loads if none of the source element conditions are true. Regardless of the conditions, the `img` element's `alt` attribute does its usual thing—it provides fallback content in case the selected image cannot be displayed.

Positioning Images

The `img` element is an inline element (more formally, a phrasing element), so it gets displayed within the normal flow of its surrounding text. That works well for small images (icons), but not so well for medium and large images. See the smiley face image in [FIGURE 7.1](#). It's a small image, an icon, and its default inline positioning works well. But the tree photographs? They're taller and their default inline positioning causes the browser to generate quite a bit of dead space above the text lines in which they're embedded. In general, you should avoid such dead space. To avoid wasted space around a medium or large image, you might want to display it on a line by itself by surrounding it with a block element.



FIGURE 7.1 Trees web page with default picture positioning

To float an image, you apply a CSS rule to the `img` element, where the CSS rule uses the `float` property and a value of `left` or `right`. You can see the complete source code for the Trees web page in [FIGURE 7.3](#), but, for your convenience, here are the CSS rules in charge of floating the two tree photographs:

```
.left {float: left; margin: 8px;}
.right {float: right; margin: 8px;}
```

The `margin` property in this example is not required for floating, but without it, the image's adjacent text will display uncomfortably close to the image. Here are the `img` elements that use these CSS rules:

```


```

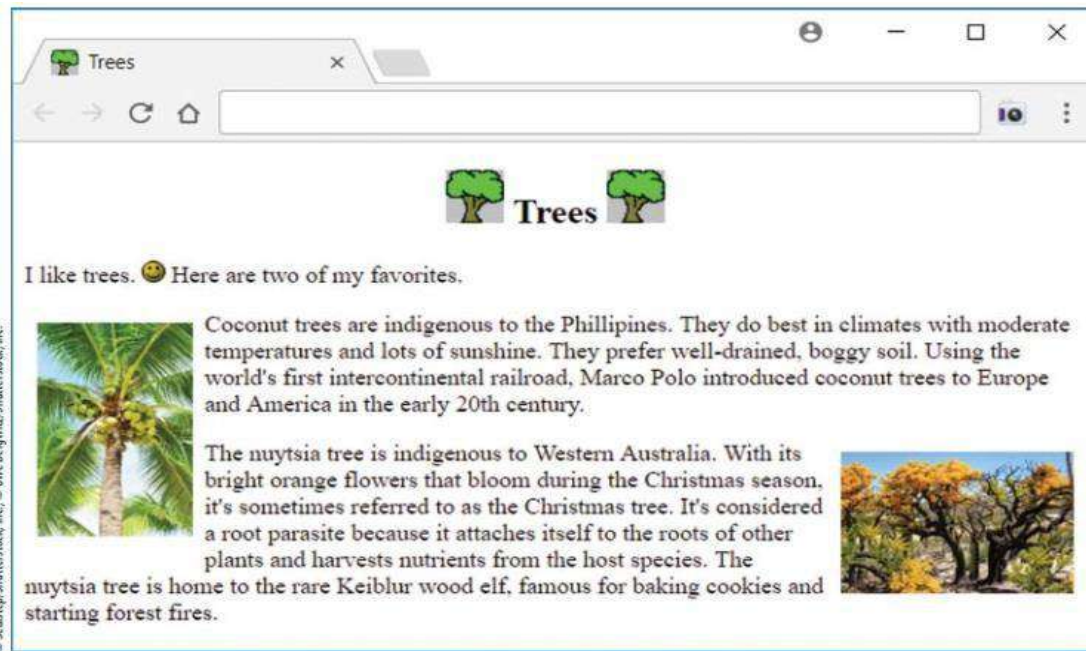



FIGURE 7.2 Trees web page using the `float` property for picture positioning

the `text-align` property is in charge of aligning not just text, but all inline content within the element to which the `text-align` property is being applied. The `img` element is an inline element. So applying `text-align: center` to the `h2` heading works great for centering the “Trees” text and the surrounding tree icons.

Shortcut Icon

To mark your web page with a shortcut icon in the browser’s tab area, in your web page’s head container, include a link element with `rel="icon"`.

```
<link rel="icon" href="../images/tree.png">
```

The `rel` attribute specifies the relationship between the current web page and the linked-to entity. So for a shortcut icon, the `rel` attribute specifies `icon`. Do you recall what else the link element has been used for? It’s used for linking to an external CSS file. In that case, the `rel` attribute specifies `stylesheet` (`rel="stylesheet"`).

For shortcut icons, the W3C recommends using a GIF or PNG file with dimensions of 16×16 pixels or 32×32 pixels. If you use an image file with dimensions different from 16×16 or 32×32 , the browser will adjust its size, so it fits in the small square area reserved for the shortcut icon in the browser window’s tab. But as you know, degradation occurs when an image file’s size gets adjusted, particularly if the original file’s size does not scale perfectly with the new file’s size. shortcut icons are sometimes called website icons. Shortcut icons are also called bookmark icons because when you display your bookmarks/favorites list, the icons appear next to the names in the list.

iframe Element

Instead of loading an entire web page within an `iframe`’s browsing context, it’s more common to load a stored image into a browsing context. For example, in the Art Exhibit web page in [FIGURE 7.4](#), an `iframe` element is used to display the large picture in the center. The Art Exhibit web page’s website stores image files for that picture and three other pictures that all have the same dimensions. In addition, the website stores image files for the four smaller pictures at the left of Figure 7.4. Those pictures are smaller versions of the larger pictures. When the user clicks one of

the smaller images, the browser grabs the larger version of the clicked image and copies it to the browsing context area. A thumbnail is a smallish image that serves as a representative for a larger version of that same image. Thumbnails are often used to help with the organization of a group of images

```
<iframe class="cell" name="full-size" width="480" height="320"
src="../images/houseRenderings/kitchen.jpg"></iframe>
```



FIGURE 7.4 Art Exhibit web page

In order to load a new image into the browsing context area, we need to be able to refer to the iframe element, and the name attribute's value allows us to do that. As you can see in the previous code, the iframe element's name attribute has a value of full-size.

```
<body>
<h1>Modular House Exhibit, Park University's Graphic Design Program</h1>
<h3>
  Click a thumbnail at the left in order to move it to the viewing frame
  at the right.
</h3>
<div class="table">
  <nav class="cell">
    <a href="../images/houseRenderings/kitchen.jpg" target="full-size">
      </a>
    <a href="../images/houseRenderings/living.jpg" target="full-size">
      </a>
    <a href="../images/houseRenderings/stairway.jpg" target="full-size">
      </a>
    </nav>
    <iframe class="cell" name="full-size" width="480" height="320"
      src="../images/houseRenderings/kitchen.jpg"></iframe>
  </div>
</body>
</html>
```

FIGURE 7.5A Source code body container for Art Exhibit web page

1. Give the code to create a webpage with the output as shown below.

Wind Disasters

Tornadoes	Hurricanes
Joplin, Missouri	Katrina
Tri-State	Andrew
Waco, Texas	Galveston

2. Explain different href attribute values and their jumping effects
3. Mention and briefly describe the Common formats for web page bitmap image files.
4. What is the need of Responsive web design (RWD)? Explain resolution switching with an example.
5. Illustrate the situation in which webpage designer thinks of CSS rule uses the float property and a value of left or right. Justify how the property and values are useful in such situations
6. What is browsing context? Name the element used to create it.
7. Discuss table elements along with spanning row and columns.
8. Apply the following table elements to display the following table: table elements : table, td, tr, th, tbody, tfoot, thread.

SLNO	USN	Name	Dept
1			
2			
Total No. of rows	2	-	-

9. Write HTML code for following table.

Time Day	9.00 am to 1.15 pm	2.00 pm to 5.00 pm
Mon to Fri	Sub FI Theory class ABC/EFG/XYZ	ML/WTB Lab AD block, 1 st *floor
Sat	Sub FI Extra curricular activity	

10. Explain basic table structure. Create an HTML document for fig.

ONE	TWO
THREE	FOUR

11. what are bitmaps image formats, explain with any one format.
12. Briefly explain the following terms i) Positioning Images ii) Shortcut Icon iii) iframe Element
13. Briefly explain the term CSS for links.