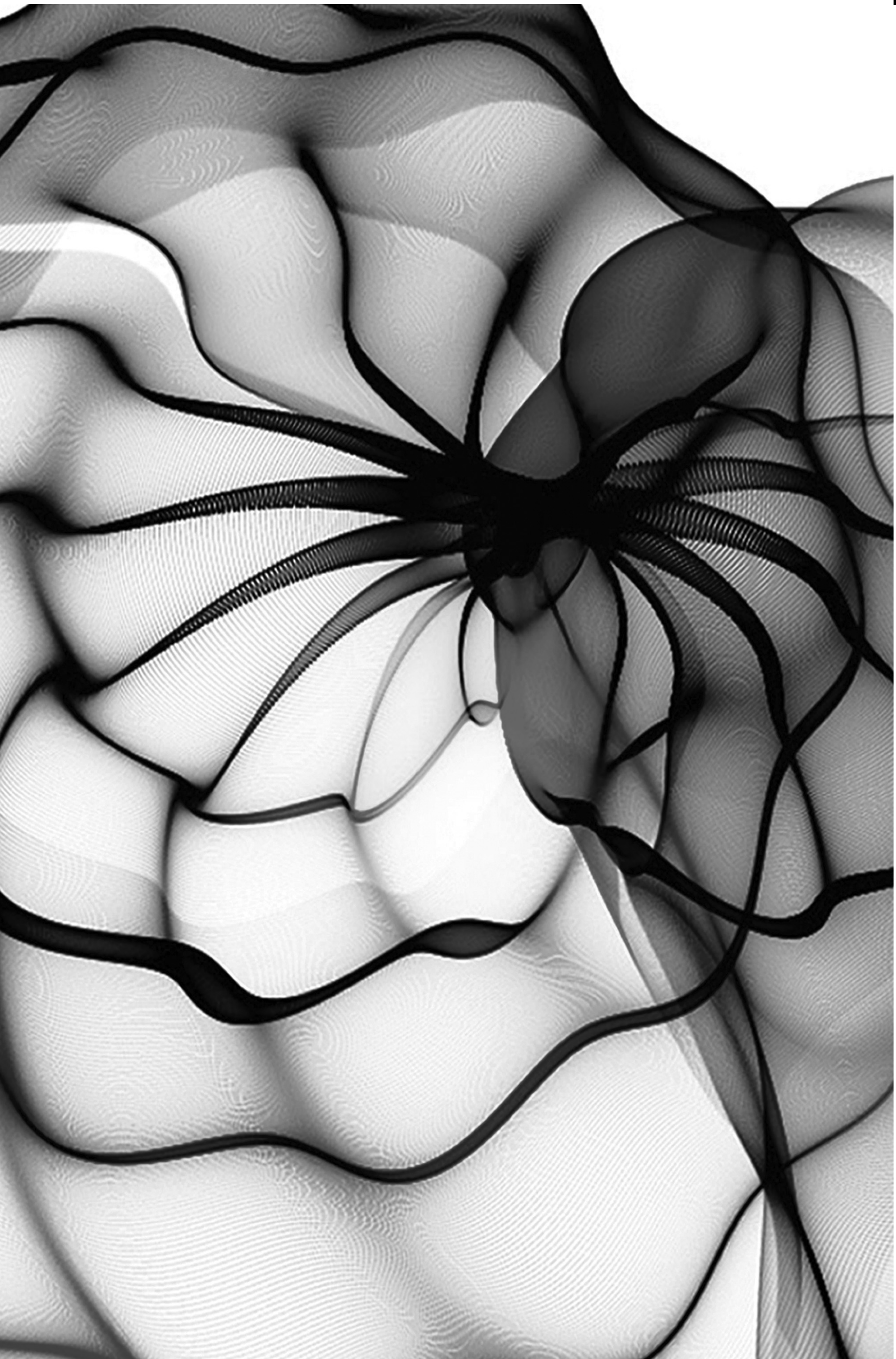
Introduction to HTML5: Part 1

2



He had a wonderful talent for packing thought close, and rendering it portable.
—Thomas Babington Macaulay

High thoughts must have high language.
—Aristophanes

Objectives

In this chapter you'll:

- Understand important components of HTML5 documents.
- Use HTML5 to create web pages.
- Add images to web pages.
- Create and use hyperlinks to help users navigate web pages.
- Mark up lists of information.
- Create tables with rows and columns of data.
- Create and use forms to get user input.

Lines 15–18 demonstrate a `"description"` `meta` element. The `content` attribute of such a `meta` element provides a three- to four-line description of a site, written in sentence form. Search engines also use this description to catalog your site and sometimes display this information as part of the search results.

Software Engineering Observation 2.2

`meta` elements are not visible to users. They must be placed inside the `head` section of your HTML5 document; otherwise they will not be read by search engines.

2.14 Web Resources

`www.deitel.com/html5`

Visit our online HTML5 Resource Center to find categorized links to mostly free HTML5 introductions, tutorials, demos, videos, documentation, books, blogs, forums, sample chapters and more.

Summary

Section 2.1 Introduction

- HTML5 is a markup language that specifies the structure and content of documents that are displayed in web browsers.

Section 2.2 Editing HTML5

- Computers called web servers store HTML5 documents.
- Clients (for example, web browsers running on your local computer or smartphone) request specific resources (p. 38) such as HTML5 documents from web servers.

Section 2.3 First HTML5 Example

- The document type declaration (`DOCTYPE`; p. 39) is *required* in HTML5 documents so that browsers render the page in standards mode (p. 39).
- HTML5 comments (p. 39) always start with `<!--` (p. 39) and end with `-->` (p. 39). The browser ignores all text inside a comment.
- The `html` element (p. 40) encloses the head section (represented by the `head` element; p. 40) and the body section (represented by the `body` element; p. 40).
- The head section contains information about the HTML5 document, such as its title (p. 40). It also can contain special document-formatting instructions called style sheets (p. 40) and client-side programs called scripts (p. 40) for creating dynamic web pages.
- The body section contains the page's content, which the browser displays when the user visits the web page.
- HTML5 documents delimit an element with start and end tags. A start tag (p. 40) consists of the element name in angle brackets (for example, `<html>`). An end tag (p. 40) consists of the element name preceded by a forward slash (`/`) in angle brackets (for example, `</html>`).
- The `title` element names a web page. The title usually appears in the colored bar (called the title bar; p. 40) at the top of the browser window and also appears as the text identifying a page when users add your page to their list of **Favorites** or **Bookmarks**.
- The paragraph element (p. 40), denoted with `<p>` and `</p>`, helps define the structure of a document. All the text placed between the `<p>` and `</p>` tags forms one paragraph.

Section 2.4 W3C HTML5 Validation Service

- You must use proper HTML5 syntax to ensure that browsers process your documents properly.
- The World Wide Web Consortium (W3C) provides a validation service (`validator.w3.org`; p. 41) for checking a document's syntax.

Section 2.5 Headings

- HTML5 provides six heading elements (`h1` through `h6`; p. 41) for specifying the relative importance of information. Heading element `h1` is considered the most significant and is rendered in a larger font than the other five. Each successive heading element (`h2`, `h3`, etc.) is rendered in a progressively smaller font.

Section 2.6 Linking

- Hyperlinks (p. 42) reference (or link to) other resources, such as HTML5 documents and images.
- The `strong` element (p. 43) typically causes the browser to render text in a bold font.
- Links are created using the `a` (anchor) element (p. 43). The `href` ("hypertext reference") attribute (p. 43) specifies the location of a linked resource, such as a web page, a file or an e-mail address.
- Anchors can link to an e-mail address using a `mailto:` URL (p. 44). When someone clicks this type of anchored link, most browsers launch the default e-mail program to initiate an e-mail message addressed to the linked address.

Section 2.7 Images

- The `img` element's (p. 46) `src` attribute (p. 46) specifies an image's location.
- Every `img` element in an HTML5 document must have an `alt` attribute (p. 47). If a browser cannot render an image, the browser displays the `alt` attribute's value.
- The `alt` attribute helps you create accessible web pages (p. 47) for users with disabilities, especially those with vision impairments who use text-only browsers.
- Void HTML5 elements (such as `img`; p. 47) contain only attributes, do not mark up text and do not have a closing tag.

Section 2.8 Special Characters and Horizontal Rules

- HTML5 provides character entity references in the form `&`*code*`;` (p. 49) for representing characters.
- Most browsers render a horizontal rule (p. 51), indicated by the `<hr>` tag (a void element), as a horizontal line with a blank line above and below it.
- Special characters can also be expressed as numeric character references (p. 51)—decimal or hexadecimal (hex; p. 51) values.
- Most browsers render the `del` element (p. 51) as strike-through text. With this format users can indicate document revisions.

Section 2.9 Lists

- The unordered-list element `ul` (p. 51) creates a list in which each item begins with a bullet symbol (called a disc). Each entry in an unordered list is an `li` (list item) element (p. 51). Most web browsers render these elements on a new line with a bullet symbol indented from the beginning of the line.
- Lists may be nested to represent hierarchical data relationships.
- The ordered-list element `ol` (p. 52) creates a list in which each item begins with a number.

Section 2.10 Tables

- Tables are frequently used to organize data into rows and columns. Tables are defined with the `table` element (p. 54).
- The `caption` element (p. 54) specifies a table's title. The text inside the `<caption>` tag is rendered above the table by most browsers. It's good practice to include a general description of a table's information in the table element's `summary` attribute—one of the many HTML5 features that make web pages more accessible to users with disabilities. Speech devices use this attribute to make the table more accessible to users with visual impairments.
- A table has three distinct sections: head, body and foot (p. 56). The head section (or header cell) is defined with a `thead` element (p. 56), which contains header information such as column names.
- Each `tr` element (p. 56) defines an individual table row (p. 56). The columns in the head section are defined with `th` elements (p. 56).
- The table body, defined in a `tbody` element (p. 56), contains the table's primary data.
- The foot section is defined with a `tfoot` element (p. 56). The text placed in the footer commonly includes calculation results and footnotes.
- You can create larger data cells using the attributes `rowspan` (p. 57) and `colspan` (p. 57). The values assigned to these attributes specify the number of rows or columns occupied by a cell.
- The `br` element (p. 57) causes most browsers to render a line break (p. 57). Any markup or text following a `br` element is rendered on the next line.

Section 2.11 Forms

- HTML5 provides forms (p. 58) for collecting information from a user.
- Forms can contain visual and nonvisual components. Visual components include clickable buttons and other graphical user-interface components with which users interact. Nonvisual components, called hidden inputs (p. 61), store any data that you specify, such as e-mail addresses and HTML5 document file names that act as links.
- A form is defined by a `form` element (p. 60).
- Nonvisual components, called `hidden` inputs (p. 61), store any data that you specify.
- Attribute `method` (p. 60) specifies how the form's data is sent to the web server.
- The `action` attribute (p. 60) in the `form` element specifies the URL of the script on the web server that will be invoked to process the form's data.
- The `text` input (p. 61) inserts a text field into the form. Users can type data into text fields.
- The `input` element's `size` attribute (p. 61) specifies the number of characters visible in the text field. Optional attribute `maxlength` (p. 61) limits the number of characters input into the text field.
- The `submit` input (p. 61) is a button that, when pressed, sends the user to the location specified in the `form`'s attribute. The `reset input` element sets the text displayed on the button (the default value is **Reset** if you omit the `value` attribute).
- The `textarea` element (p. 61) inserts a multiline text area into a form. The number of rows is specified with the `rows` attribute (p. 61) and the number of columns (i.e., characters per line) with the `cols` attribute (p. 61).
- The `password` input (p. 64) inserts a password box with the specified `size` (maximum number of characters allowed).
- A password box allows users to enter sensitive information, such as credit card numbers and passwords, by "masking" the information input with asterisks (*). Asterisks are usually the masking character used for password boxes. The actual value input is sent to the web server, not the characters that mask the input.

- `checkboxes` (p. 64) enable users to select from a set of options. When a user selects a `checkbox`, a check mark appears in the `checkbox`. Otherwise, the `checkbox` remains empty. `checkboxes` can be used individually or in groups. `checkboxes` that are part of the same group have the same `name`.
- `radio` buttons (p. 64) are similar to `checkboxes`, except that only one `radio` button in a group can be selected at any time. The `radio` buttons in a group all have the same `name` attribute and are distinguished by their different `value` attributes.
- The `select` element (p. 65) provides a drop-down list from which the user can select an item. The `name` attribute identifies the drop-down list. The `option` element adds items to the drop-down list.

Section 2.12 Internal Linking

- Internal linking (p. 67) is a mechanism that enables the user to jump between locations in the same document.
- To link to a tag with its attribute inside the same web page, the `href` attribute of an anchor element includes the `id` attribute value preceded by a pound sign (as in `#features`).

Section 2.13 `meta` *Elements*

- Search engines catalog sites by following links from page to page (often known as spidering or crawling) and saving identification and classification information for each page.
- One way that search engines catalog pages is by reading the content in each page's `meta` elements (p. 67), which specify information about a document.
- Two important attributes of the `meta` element are `name` (p. 67), which identifies the type of `meta` element, and `content` (p. 67), which provides information search engines use to catalog pages.
- The `content` attribute of a `keywords` `meta` element provides search engines with a list of words that describe the page. These words are compared with words in search requests.
- The `content` attribute of a `description` `meta` element provides a three- to four-line description of a site, written in sentence form. Search engines also use this description to catalog your site and sometimes display this information as part of the search results.

Self-Review Exercises

2.1 State whether each of the following is *true* or *false*. If *false*, explain why.
  a) An ordered list cannot be nested inside an unordered list.
  b) Element `br` represents a line break.
  c) Hyperlinks are denoted by `link` elements.
  d) The width of all data cells in a table must be the same.
  e) You're limited to a maximum of five internal links per page.

2.2 Fill in the blanks in each of the following:
  a) The __________ element inserts a horizontal rule.
  b) A superscript is marked up using the __________ element, and a subscript is marked up using the __________ element.
  c) The least significant heading element is __________ and the most significant heading element is __________.
  d) Element __________ marks up an unordered list.
  e) Element __________ marks up a paragraph.
  f) The __________ attribute in an `input` element inserts a button that, when clicked, resets the contents of the form.
  g) The __________ element marks up a table row.
  h) __________ are usually used as masking characters in a password box.

Answers to Self-Review Exercises

2.1 a) False. An ordered list can be nested inside an unordered list and vice versa. b) True. c) False. Hyperlinks are denoted by `a` elements. d) False. You can specify the width of any column, either in pixels or as a percentage of the table width. e) False. You can have an unlimited number of internal links.

2.2 a) `hr`. b) `sup`, `sub`. c) `h6`, `h1`. d) `ul`. e) `p`. f) `type = "reset"`. g) `tr`. h) Asterisks.

Exercises

2.3 Use HTML5 to create a document that contains the following text:

```
Internet and World Wide Web How to Program: Fifth Edition
Welcome to the world of Internet programming. We have provided
coverage for many Internet-related topics.
```

Use `h1` for the title (the first line of text), `p` for text (the second and third lines of text). Insert a horizontal rule between the `h1` element and the `p` element. Open your new document in a web browser to view the marked-up document.

2.4 An image named `deitel.png` is 200 pixels wide and 150 pixels high. Write an HTML5 statement using the `width` and `height` attributes of the `img` element to perform each of the following transformations:

a) Increase the size of the image by 100 percent.
b) Increase the size of the image by 50 percent.
c) Change the width-to-height ratio to 2:1, keeping the `width` attained in part (a).

2.5 Create a link to each of the following:

a) The file `index.html`, located in the `files` directory.
b) The file `index.html`, located in the `text` subdirectory of the `files` directory.
c) The file `index.html`, located in the `other` directory in your parent directory. [*Hint:* `..` signifies parent directory.]
d) The President's e-mail address (`president@whitehouse.gov`).
e) The file named `README` in the `pub` directory of `ftp.cdrom.com`. [*Hint:* Use `ftp://`.]

2.6 Create an HTML5 document containing an ordered list of three items—ice cream, soft serve and frozen yogurt. Each ordered list should contain a nested, unordered list of your favorite flavors. Provide three flavors in each unordered list.

2.7 Create an HTML5 document that uses an *image* as an *e-mail link*. Use attribute `alt` to provide a description of the image and link.

2.8 Create an HTML5 document that contains links to your five favorite daily deals websites (possibly Groupon, Living Social, etc.). Your page should contain the heading "My Favorite Daily Deals Web Sites." Click on each of these links to test your page.

2.9 Create an HTML5 document that contains an unordered list with links to all the examples presented in this chapter. [*Hint*: Place all the chapter examples in an `examples` directory, then link to the files in that directory.]

2.10 Identify each of the following HTML5 items as either an *element* or an *attribute*:

a) `html`
b) `width`
c) `href`
d) `br`
e) `h3`

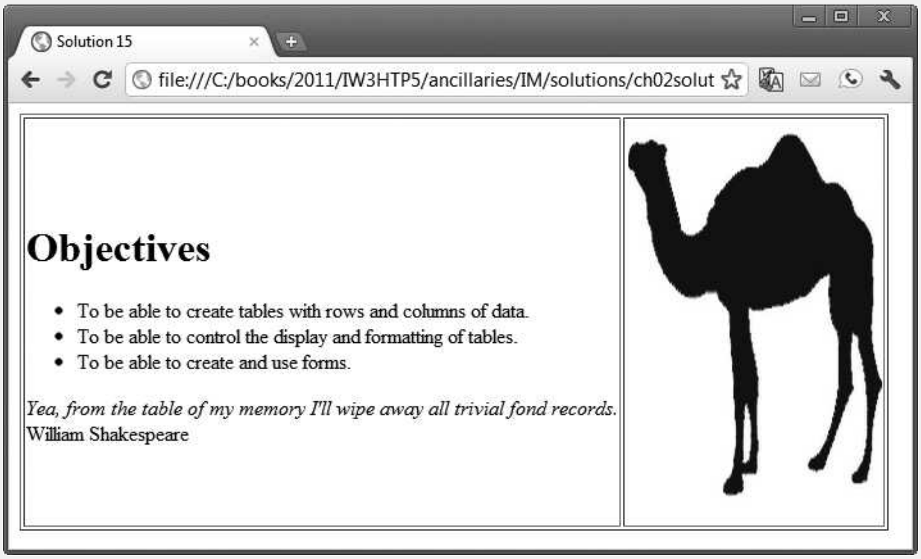
f) `a`
g) `src`

2.11 State which of the following statements are *true* and which are *false*. If *false*, explain why.
a) A valid HTML5 document cannot contain uppercase letters in element names.
b) HTML5 documents can have the file extension `.htm`.
c) `&less;` is the character entity reference for the less-than (<) character.
d) In a valid HTML5 document, `<li>` can be nested inside either `<ol>` or `<ul>` tags.

2.12 Fill in the blanks in each of the following:
a) HTML5 comments begin with `<!--` and end with ________.
b) In HTML5, attribute values can be enclosed in ________.
c) ________ is the character entity reference for an ampersand.
d) Element ________ can be used to make text bold.

2.13 Categorize each of the following as an element or an attribute:
a) `width`
b) `td`
c) `th`
d) `name`
e) `select`
f) `type`

2.14 Create the HTML5 markup that produces the table shown in Fig. 2.18. Use `<em>` and `<strong>` tags as necessary. The image (`camel.png`) is included in the Chapter 2 examples directory.


Solution 15
file:///C:/books/2011/IW3HTP5/ancillaries/IM/solutions/ch02solut

Objectives

- To be able to create tables with rows and columns of data.
- To be able to control the display and formatting of tables.
- To be able to create and use forms.

Yea, from the table of my memory I'll wipe away all trivial fond records.
William Shakespeare

Fig. 2.18 | HTML5 table for Exercise 2.14.

2.15 Write an HTML5 document that produces the table shown in Fig. 2.19.

2.16 A local university has asked you to create an HTML5 document that allows prospective college students to provide feedback about their campus visit. Your HTML5 document should contain a form with text fields for a name and e-mail. Provide checkboxes that allow prospective students to

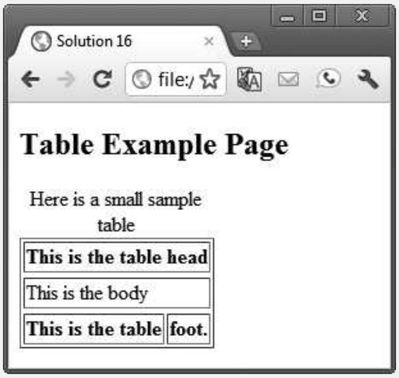


Table Example Page

Here is a small sample table

This is the table head	
This is the body	
This is the table	foot.

Fig. 2.19 | HTML5 table for Exercise 2.15.

indicate what they liked most about the campus. The checkboxes should include: campus, students, location, atmosphere, dorm rooms and sports. Also, provide radio buttons that ask the prospective students how they became interested in the college. Options should include: friends, television, Internet and other. In addition, provide a text area for additional comments, a submit button and a reset button. Use post to sent the information in the form to `http://www.deitel.com`.

2.17 Create an HTML5 document titled "How to Get Good Grades." Use `<meta>` tags to include a series of keywords that describe your document.

3

Introduction to HTML5: Part 2

Form ever follows function.
—Louis Sullivan

I listen and give input only if somebody asks.
—Barbara Bush

Objectives

In this chapter you'll:

- Build a form using the new HTML5 `input` types.
- Specify an `input` element in a form as the one that should receive the focus by default.
- Use self-validating `input` elements.
- Specify temporary `placeholder` text in various `input` elements
- Use `autocomplete` `input` elements that help users re-enter text that they've previously entered in a form.
- Use a `datalist` to specify a list of values that can be entered in an `input` element and to autocomplete entries as the user types.
- Use HTML5's new page-structure elements to delineate parts of a page, including headers, sections, figures, articles, footers and more.

3.4.9 `footer` Element

The **`footer` element** (lines 165–176) describes a *footer*—content that usually appears at the bottom of the content or `section` element. In this example, we use the `footer` to describe the copyright notice and contact information. You can use CSS3 to style the `footer` and position it on the page.

3.4.10 Text-Level Semantics: `mark` Element and `wbr` Element

The **`mark` element** (lines 72–74) highlights the text that's enclosed in the element. The **`wbr` element** (line 168) indicates the appropriate place to break a word when the text wraps to multiple lines. You might use `wbr` to prevent a word from breaking in an awkward place.

Summary

Section 3.2 New HTML5 Form `input` Types

- HTML5 introduces several new form `input` types and attributes. These are not yet universally supported by all browsers.
- The Opera browser offers robust support of the new `input` types.
- We provide sample outputs from a variety of browsers so that you can see how the `input` types behave differently in each.

Section 3.2.1 `input` Type `color`

- The `color input` type (p. 80) enables the user to enter a color.
- Most browsers render the `color input` type as a text field in which the user can enter a hexadecimal code.
- In the future, when the user clicks a color input, browsers will likely display a dialog from which the the user can select a color.
- The `autofocus` attribute (p. 80)—which can be used in only one `input` element on a form—places the cursor in the text field after the browser loads and renders the page. You do not need to include `autofocus` in your forms.
- The new HTML 5 `input` types self validate on the client side, eliminating the need to add JavaScript code to validate user input and reducing the amount of invalid data submitted.
- When a user enters data into a form then submits the form, the browser immediately checks that the data is correct.
- If you want to bypass validation, you can add the `formnovalidate` attribute (p. 82) to `input` type `submit`.
- Using JavaScript, we can customize the validation process.

Section 3.2.2 `input` Type `date`

- The `date input` type (p. 82) enables the user to enter a date in the format `yyyy-mm-dd`.
- Firefox and Internet Explorer all display a text field in which a user can enter a date such as `2012-01-27`.
- Chrome and Safari display a spinner control (p. 82)—a text field with an up-down arrow (⬍) on the right side—allowing the user to select a date by clicking the up or down arrows.
- Opera displays a calendar.

Section 3.2.3 **`input`** *Type* **`datetime`**

- The `datetime` `input` type (p. 82) enables the user to enter a date (year, month, day), time (hour, minute, second, fraction of a second) and the time zone set to UTC (Coordinated Universal Time or Universal Time, Coordinated).

Section 3.2.4 **`input`** *Type* **`datetime-local`**

- The `datetime-local` `input` type (p. 82) enables the user to enter the date and time in a *single* control.
- The date is entered as year, month, day, hour, minute, second and fraction of a second.

Section 3.2.5 **`input`** *Type* **`email`**

- The `email` `input` type (p. 83) enables the user to enter an e-mail address or list of e-mail addresses separated by commas.
- If the user enters an invalid e-mail address (i.e., the text entered is not in the proper format) and clicks the **Submit** button, a callout asking the user to enter an e-mail address is rendered pointing to the `input` element.
- HTML5 does not validate whether an e-mail address entered by the user actually exists—rather it just validates that the information is in the *proper format*.
- The `placeholder` attribute (p. 83) allows you to place temporary text in a text field. Generally, `placeholder` text is light gray and provides an example of the text and text format the user should enter. When the focus is placed in the text field (i.e., the cursor is in the text field), the `placeholder` text disappears—it's not "submitted" when the user clicks the **Submit** button (unless the user types the same text).
- Add descriptive text to the right of each `input` element in case the user's browser does not support `placeholder` text.
- The `required` attribute (p. 84) forces the user to enter a value before submitting the form.
- You can add `required` to any of the `input` types. If the user fails to fill enter a required item, a callout pointing to the empty element appears, asking the user to enter the information.

Section 3.2.6 **`input`** *Type* **`month`**

- The `month` `input` type (p. 84) enables the user to enter a year and month in the format `yyyy-mm`, such as `2012-01`.
- If the user enters a month in an improper format and clicks the **Submit** button, a callout stating that an invalid value was entered appears.

Section 3.2.7 **`input`** *Type* **`number`**

- The `number` `input` type (p. 84) enables the user to enter a numerical value.
- The `min` attribute sets the minimum valid number, in this case `"0"`.
- The `max` attribute sets the maximum valid number, which we set to `"7"`.
- The `step` attribute determines the increment in which the numbers increase. For example, if we set the `step` to `"2"`, the number in the spinner control will increase or decrease by two each time the up or down arrow, respectively, in the spinner control is clicked.
- The `value` attribute sets the initial value displayed in the form.
- The spinner control includes only the valid numbers. If the user attempts to enter an invalid value by typing in the text field, a callout pointing to the `number` `input` element will instruct the user to enter a valid value.

Section 3.2.8 **`input`** *Type* **`range`**

- The `range` `input` type (p. 85) appears as a *slider* control in Chrome, Safari and Opera.
- You can set the minimum and maximum and specify a value.
- The slider appears at the value in the range when the HTML5 document is rendered.
- The `range` `input` type is inherently self-validating when it's rendered by the browser as a slider control, because the user is unable to move the slider outside the bounds of the minimum or maximum value.

Section 3.2.9 **`input`** *Type* **`search`**

- The `search` `input` type (p. 85) provides a search field for entering a query and is functionally equivalent to an input of type `text`.
- When the user begins to type in the search field, Chrome and Safari display an **`X`** that can be clicked to clear the field.

Section 3.2.10 **`input`** *Type* **`tel`**

- The `tel` `input` type (p. 86) enables the user to enter a telephone number.
- At the time of this writing, the `tel` `input` type is rendered as a text field in all of the browsers.
- The length and format of telephone numbers varies greatly based on location, making validation quite complex. HTML5 does not self validate the `tel` `input` type. To ensure that the user enters a phone number in a proper format, you can use the `pattern` attribute.
- When the user enters a phone number in the wrong format, a callout requesting the proper format appears, pointing to the `tel` `input` element.

Section 3.2.11 **`input`** *Type* **`time`**

- The `time` `input` type (p. 86) enables the user to enter an hour, minute, second and fraction of a second.

Section 3.2.12 **`input`** *Type* **`url`**

- The `url` `input` type (p. 87) enables the user to enter a URL. The element is rendered as a text field. If the user enters an improperly formatted URL, it will not validate. HTML5 does not ensure that the URL entered actually exists.

Section 3.2.13 **`input`** *Type* **`week`**

- The `week` `input` type (p. 87) enables the user to select a year and week number in the format `yyyy-Wnn`.
- Opera renders week control with a down arrow that, when clicked, brings up a calendar control.

Section 3.3.1 **`input`** *Element* **`autocomplete`** *Attribute*

- The `autocomplete` attribute (p. 87) can be used on `input` types to automatically fill in the user's information based on previous input.
- You can enable `autocomplete` for an entire form or just for specific elements.

Section 3.3.2 **`datalist`** *Element*

- The `datalist` element (p. 90) provides input options for a `text` `input` element. The browser can use these options to display `autocomplete` options to the user.

Section 3.4 Page-Structure Elements

- HTML5 introduces several new page structure elements.

Section 3.4.1 **header** *Element*

- The header element (p. 96) creates a header for the page that contains text, graphics or both.
- The header element may be used multiple times on a page and often includes HTML headings.
- The time element (p. 96) enables you to identify a date, a time or both.

Section 3.4.2 **nav** *Element*

- The nav element (p. 96) groups navigation links.

Section 3.4.3 **figure** *Element and* **figcaption** *Element*

- The figure element (p. 96) describes an image in the document so that it could be moved to the side of the page or to another page.
- The figcaption element (p. 96) provides a caption for the image in the figure element.

Section 3.4.4 **article** *Element*

- The article element (p. 96) describes content that's separate from the main content of the page and might be used or distributed elsewhere, such as a news article, forum post or blog entry.
- article elements can be nested.

Section 3.4.5 **summary** *Element and* **details** *Element*

- The summary element (p. 96) displays a right-pointing arrow next to a summary or caption when the document is rendered in a browser. When clicked, the arrow points downward and reveals the content in the details element (p. 96).

Section 3.4.6 **section** *Element*

- The section element (p. 96) describes a section of a document, usually with a heading for each section.
- section elements can be nested.

Section 3.4.7 **aside** *Element*

- The aside element (p. 96) describes content that's related to the surrounding content (such as an article) but that's somewhat separate from the flow of the text.
- nav elements can be nested in an aside element.

Section 3.4.8 **meter** *Element*

- The meter element (p. 97) renders a visual representation of a measure within a range.
- Useful meter attributes are min, max and value.

Section 3.4.9 **footer** *Element*

- The footer element (p. 98) describes a *footer*—content that usually appears at the bottom of the content or section element.
- You can use CSS3 to style the footer and position it on the page.

Section 3.4.10 Text-Level Semantics: **mark** *Element and* **wbr** *Element*

- The mark element (p. 98) enables you to highlight text.
- The wbr element (p. 98) indicates the appropriate place to break a word when the text wraps to multiple lines. You might use wbr to prevent a word from breaking in an awkward place.

Self-Review Exercises

3.1 Fill in the blanks in each of the following:

a) The `color input` type enables the user to enter a color. At the time of this writing, most browsers render the `color input` type as a text field in which the user can enter a __________.
b) The __________ attribute allows you to place temporary text in a text field.
c) If you want to bypass validation, you can add the `formnovalidate` attribute to `input` type __________.
d) The __________ attribute forces the user to enter a value before submitting the form.
e) The __________ control is typically displayed for the `number input` type and includes only the valid numbers.
f) The __________ `input` type enables the user to enter an hour, minute, second and fraction of second.
g) The __________ element provides input options for a `text input` element.
h) The __________ element describes content that's separate from the main content of the page and could potentially be used or distributed elsewhere, such as a news article, forum post or blog entry.
i) The __________ element describes the text that usually appears at the bottom of the content or the bottom of a `section` element.
j) The __________ element indicates the appropriate place to break a word when the text wraps to multiple lines.

3.2 State whether each of the following is *true* or *false*. If *false*, explain why.

a) Any particular HTML5 form `input` types must render identically in every HTML5-compliant browser.
b) When the focus is placed in the text field (i.e., the cursor is in the text field), the `placeholder` text is submitted to the server.
c) You do not need to include `autofocus` in your forms.
d) The new HTML 5 `input` types are self validating on the client side, eliminating the need to add complicated scripts to your forms to validate user input and reducing the amount of invalid data submitted.
e) The `range input` type is inherently self-validating when it's rendered by the browser as a slider control, because the user is unable to move the slider outside the bounds of the minimum or maximum value.
f) HTML5 self validates the `tel input` type.
g) If the user enters an improperly formatted URL in a `url input` type, it will not validate. HTML5 does not validate that the URL entered actually exists.
h) The `nav` element displays a drop-down menu of hyperlinks.
i) The `header` element may be used only one time on a page.
j) `nav` elements can be nested in an `aside` element.
k) You might use the `brk` to prevent awkward word breaks.

Answers to Self-Review Exercises

3.1 a) hexadecimal code. b) `placeholder`. c) `submit`. d) `required`. e) spinner. f) `time`. g) `datalist`. h) `article`. i) `footer`. j) `wbr`.

3.2 a) False. The rendering of `input` types can vary among browsers. b) False. When the focus is placed in the text field, the `placeholder` text disappears. It's not "submitted" when the user clicks the **Submit** button (unless the user types the same text). c) True. d) True. e) True. f) False. The length and format of telephone numbers varies greatly based on location, making validation quite complex, so HTML5 does not self validate the `tel input` type. To ensure that the user enters a phone number in a proper format, we use the `pattern` attribute. g) True. h) False. The `nav` element groups navigation links. i) False. The `header` element may be used multiple times on a page and often includes HTML headings (`<h1>` through `<h6>`) j) True. k) False. You might use the `wbr` to prevent awkward word breaks.

Exercises

3.3 Fill in the blanks in each of the following:

a) The ________ attribute—used in a single `input` element on a form—automatically highlights the `input` element and, if appropriate, places the cursor in the text field after the browser loads and renders the page.
b) The new HTML 5 `input` types are ________ on the client side.
c) The ________ `input` type enables the user to enter a numerical value.
d) The ________ `input` type is inherently self-validating when it's rendered by the browser as a slider control, because the user is unable to move the slider outside the bounds of the minimum or maximum value.
e) The ________ attribute can be used on `input` types to automatically fill in the user's information based on previous input.
f) The ________ element provides a caption for the image in the `figure` element.
g) The `summary` element displays a right-pointing arrow next to a summary or caption when the document is rendered in a browser. When clicked, the arrow points downward and reveals the content in the ________ element.
h) The `mark` element enables you to ________.

3.4 State whether each of the following is *true* or *false*. If *false*, explain why.

a) Browsers that render the `color input` type as a text field require the user to enter a color name.
b) When a user enters data into a form then submits the form (typically, by clicking the **Submit** button), the browser immediately checks that the data is correct.
c) HTML5 can validate whether an e-mail address entered by the user actually exists.
d) You can add `required` to any of the `input` types.
e) You can enable `autocomplete` only for specific `input` elements.
f) The `time` element enables you to indentify a date, a time or both.
g) The `caption` element provides a caption for the image in a `figure` element.
h) The `details` element displays a right-pointing arrow next to a summary or caption when the document is rendered in a browser. When clicked, the arrow points downward and reveals the content in the `summary` element.
i) The `footer` element describes content that usually appears at the bottom of the content or `section` element.
j) The `highlight` element enables you to highlight text.

3.5 Write an HTML5 element (or elements) to accomplish each of the following tasks:

a) Students were asked to rate the food in the cafeteria on a scale of 1 to 10. Use a `meter` element with text to its left and right to indicate that the average rating was 7 out of 10.
b) Create a `details` element that displays the `summary` text `"Survey Results"` for Part (a). When the user clicks the arrow next to the `summary` text, an explanatory paragraph about the survey should be displayed.
c) Create a `text input` element for a first name. The element should automatically receive the focus when the form is rendered in a browser.
d) Modify Part (c) to eliminate the `label` element and use `placeholder` text in the `input` element.

e) Use a `datalist` to provide an `autocomplete` list for five states.
f) Create a `range` input element that allows the user to select a number from 1 to 100.
g) Specify that `autocomplete` should not be allowed for a form. Show only the form's opening tag.
h) Use a `mark` element to highlight the second sentence in the following paragraph.

```
<p>Students were asked to rate the food in the cafeteria
   on a scale of 1 to 10. The average result was 7.</p>
```

3.6 *(Website Registration Form with Optional Survey)* Create a website registration form to obtain a user's first name, last name and e-mail address. In addition, include an optional survey question that asks the user's year in college (e.g., Freshman). Place the optional survey question in a `details` element that the user can expand to see the question.

3.7 *(Creating an Autocomplete Form)* Create a simple search form using a `search input` element in which the user can enter a search query. Using the Firefox web browser, test the form by entering `January` and submitting the form. Then enter a `J` in the input element to see previous entries that started with `J`—`January` should be displayed below the input element. Enter `June` and submit the form again. Now enter a `J` in the `input` element to see previous entries that started with `J`—`January` and `June` should be displayed below the `input` element. Try this with your own search queries as well.

3.8 *(Creating an Autocomplete Form with a* ***`datalist`****)* Create an `autocomplete input` element with an associated `datalist` that contains the days of the week.

3.9 *(Laying Out Book Pages in HTML5: Creating the Sections)* Mark up the paragraph text from Section 3.2.1 of this chapter as a web page using page-structure elements. The text is provided in the `exerciseTextAndImages` folder with this chapter's examples. Do not include the figures in this exercise.

3.10 *(Laying Out Book Pages in HTML5: Adding Figures)* Modify your solution to Exercise 3.9 to add the section's graphics as figures. The images are provided in the `exerciseTextAndImages` folder with this chapter's examples.

3.11 *(Laying Out Book Pages in HTML5: Adding a* ***`details`*** *Element)* Modify your solution to Exercise 3.10 to add the table in Fig. 3.5. Use the figure caption as the `summary` and format the table as an HTML table element inside the `details` element.