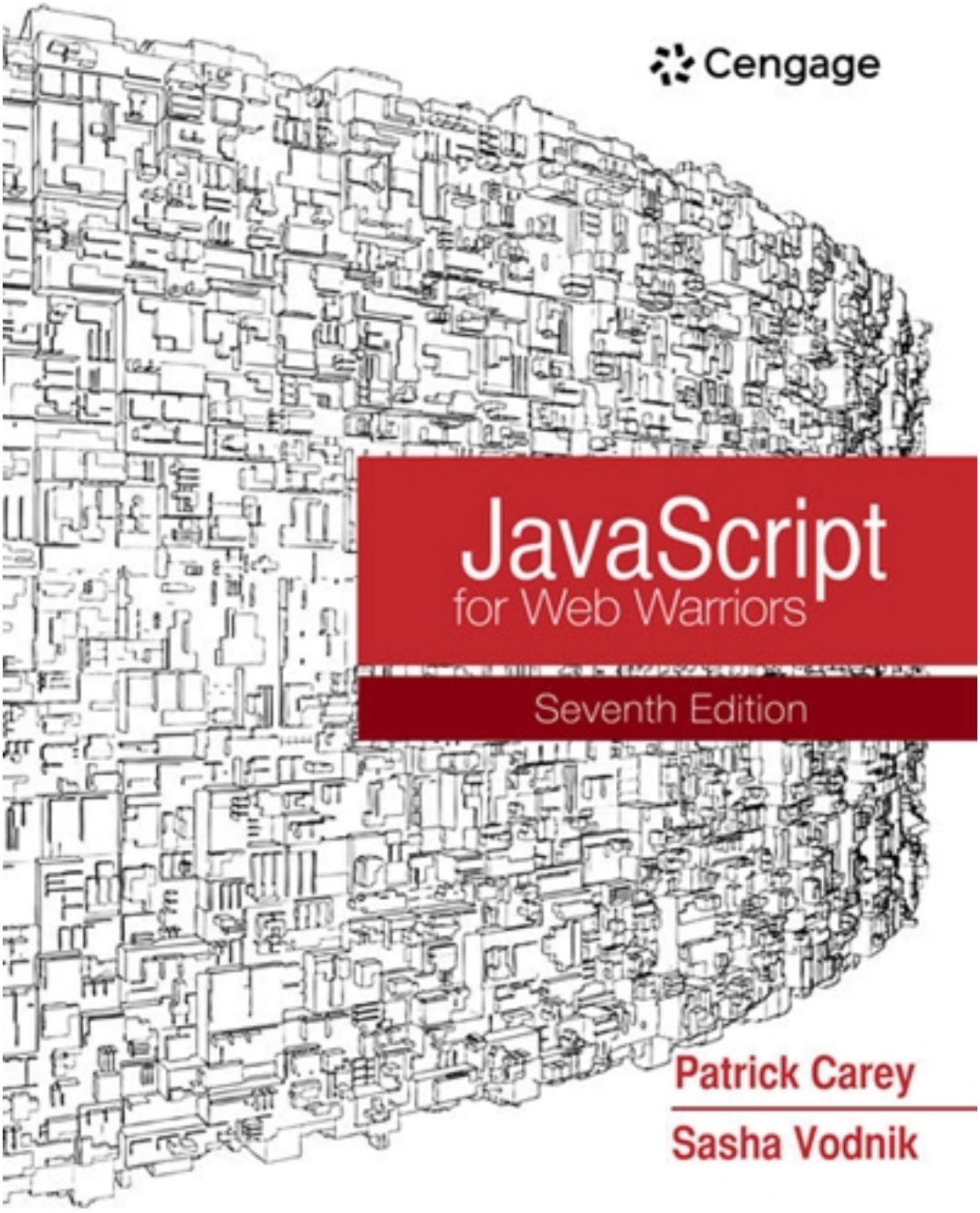
Solutions for JavaScript for Web Warriors 7th Edition by Carey

CLICK HERE TO ACCESS COMPLETE Solutions




# Solutions

# Solution and Answer Guide

Carey, JavaScript Web Warriors 7e, 9780357638002, Chapter 1: Introduction to JavaScript

## Table of Contents



## Quick Check Answers

### Quick Check 1

1. How does a scripting language like JavaScript differ from a programming language like C#?

   **JavaScript does not require a compiler, but instead needs a JavaScript interpreter which reads and runs the code as it is loaded.**

   **Feedback**: Scripting languages like JavaScript differ from programming languages like Java and C#, which need to be compiled (written for machine code) before they can be run.

2. What are the three core foundations upon which JavaScript is built?

   **ECMAScript, the Document Object Model (DOM), the Browser Object Model (BOM)**

**Feedback**: ECMAScript is a scripting language that is used by language other than JavaScript. One of things that makes JavaScript unique is that combines ECMAScript with the Document Object Model (DOM) that specifies how to interact with the contents of the web page and the Browser Object Model (BOM) that specifies how to interact with the browser.

3. In client/server architecture, what is a client? What is a server?

**The server is usually some sort of database from which a client requests information. A server fulfills a request for information by managing the request or serving the requested information to the client—hence the term, client/server. One of the primary roles of the client, or front end, in a two-tier system is the presentation of an interface to the user. The user interface gathers information from the user, submits it to a server, or back end, then receives, formats, and presents the results.**

**Feedback**: In understanding how JavaScript interacts with the user's computer and the computer hosting the website, it's important to distinguish between the client (the user's computer) and the server in which the website resides.

## Quick Check 2

1. What HTML element is used to embed JavaScript code within an HTML file?

**The `script` element**

**Feedback**: The `<script>` tag encloses all JavaScript statements that are embedded within the HTML file. It is also used to reference external JavaScript files by including the src attribute within the tag. However, it cannot do both at the same time.

2. What JavaScript command do you use to write the HTML content <h1>Plant Types</h1> to the web page document?

```
document.write("<h1>Plant Types</h1>")
```

**Feedback**: The `document.write()` method is used to write content into the web page at the location in the HTML file where the script element has been placed.

3. Provide the code to write the text, "Major Page Heading" as JavaScript block comment.

```
/*
   Major Page Heading
*/
```

**Feedback**: Block comments must start with the `/*` characters and end with the `*/` characters. All text between those two characters is treated as comment text.

4. Provide the code to write the text, “Major Page Heading” as a JavaScript line comment.

```
// Major Page Heading
```

**Feedback**: Line comment are single line comments that begin with the `//` characters. They can only extend through a single line. If one needs a multi-line comment, one should use a block comment.

Quick Check 3

1. What are the three JavaScript keywords for declaring a variable?

**`let`, `var`, and `const`**

**Feedback**: The `let` keyword represents the standard introduced in ES6 (in 2016.) The `var` keyword is the old standard and is still typically used in JavaScript code. The `const` keyword was also introduced in ES6 and should only be used for constants (unchanging values.) The main difference between `var` and `let` is in the scope of the variable (a topic introduced in Chapter 2.)

2. What is the difference between declaring and initializing a variable?

**Using the `let`, `var`, or `const` keywords to create a variable is called declaring the variable. When you declare a variable, you can also assign a specific value to, or initialize, the variable by adding an equal sign ( = ) after the variable name, followed by the value you’re assigning to the variable.**

**Feedback**: Variables can be declared with or without an initial value. JavaScript, being a loosely typed language (as discussed in Chapter 2) can be introduced with a formal declaration, but that is considered poor programming practice.

3. What is returned by expression "100" + 10?

**10010**

**Feedback**: When the `+` operator is used with text strings, it combines the text strings even if those strings contain numeric values.

4. What is an event handler for?

**When an event occurs, your script executes any code that responds to that specific event on that specific element. This code is known as the event handler.**

**Feedback**: Event handlers are required for JavaScript code to respond to actions or events occurring within the web page or web browser after the script has been completely loaded. These events can be such things as a user clicking a form button, text being typed into a input control, or a form being submitted to the server for processing.

Quick Check 4

1. Why should you place scripts at the end of an HTML document's body section?

   **The elements in an HTML document are rendered in the order in which they occur in the document, and each script is processed when the HTML element that contains it is parsed by a browser. When processing a script in the head section or in the middle of HTML content, browsers do not continue rendering the web page until the script is loaded and executed. If a script is very large or complex, this could cause the page to be displayed with only some of its content and formatting until the script finishes loading. If you instead place your script elements just before the end of the body section, you allow browsers to render all the simple HTML content immediately on the user's screen, and then load and process any JavaScript that works with that content. This ensures that users can see and interact with the entire web page as quickly as possible.**

   **Feedback**: By default, when the browser encounters a script it immediately executes the contents of the script before continuing to load the rest of the web page. If the script needs to work with elements of the web page as referenced in the Document Object Model, all of the web page should be loaded and, thus, it is best to put the script at the end of the HTML file. However, this can be mitigated for external JavaScript files by placing the `defer` attribute in the `<script>` tag.

2. How do you incorporate the contents of a JavaScript source file into an HTML document?

   **To access JavaScript code that is saved in an external file, you use the `src` attribute of the script element. You assign to the `src` attribute the URL of a JavaScript source file.**

   **Feedback**: When the browser encounters a `<script>` tag that loads an external JavaScript file it will immediately load the contents of that file (pausing the loading of the rest of the web page) unless the defer or async attribute is included with the tag.

## Review Questions Answers

1. A programming language like Java requires a:

   a. Interpreter

   b. Document Object Model

   c. compiler

   d. Browser Object Model

   **Answer:** C

   **Feedback**: Programming languages which are not interpreted need a compiler to transform the program code into machine code

2. HTML is an example of a:

   a. programming language
   b. machine language
   c. scripting language
   d. markup language

   **Answer:** D

   **Feedback**: A markup language is a language that defines the content, structure, and appearance of a document.

3. The syntax specifications for JavaScript are defined in:

   a. HTML
   b. the Document Object Model
   c. the Browser Object Model
   d. ECMAScript

   **Answer:** D

   **Feedback**: JavaScript is a scripting language based on the standards of ECMAScript which is constantly developed and adapted to meet the needs of modern browsers and devices.

4. JavaScript is built upon:

   a. ECMAScript
   b. the Document Object Model
   c. the Browser Object Model
   d. ECMAScript, the Document Object Model, and the Browser Object Model

   **Answer:** D

   **Feedback**: JavaScript is built on three foundations: the scripting language ECMAScript, the Document Object Model (DOM) that describes how to access the contents and actions within a web page, and the Browser Object Model (BOM) that describes how to access the features and behaviors of the browser

5. The specifications for the Document Object Model are determined by:

   a. each browser alone
   b. each device alone
   c. the World Wide Web Consortium (W3C)
   d. the European Computer Manufacturers Association (ECMA)

   **Answer:** C

**Feedback**: The specifications of the DOM are maintained by the World Wide Web Consortium (W3C) which also is responsible for the development of standards for HTML and CSS.

6. Which of the following is not a language used by web developers?

   a. JavaScript

   b. HTML

   c. CSS

   d. machine code

   **Answer:** D

   **Feedback**: Machine code is code that is understand by computers or computer devices. Programming languages which are compiled translate their commands into machine code.

7. A system consisting of a client and a server is known as a _____.

   a. mainframe topology

   b. double-system architecture

   c. two-tier system

   d. wide area network

   **Answer:** C

   **Feedback**: In traditional client/server architecture, the server is usually some sort of database from which a client requests information. A system consisting of a client and a server is known as a two-tier system.

8. What is usually the primary role of a client?

   a. locating records that match a request

   b. heavy processing, such as calculations

   c. data storage

   d. the presentation of an interface to the user

   **Answer:** D

   **Feedback**: One of the primary roles of the client, or front end, in a two-tier system is the presentation of an interface to the user. The user interface gathers information from the user, submits it to a server, or back end, then receives, formats, and presents the results returned from the server. In traditional client/server architecture, the server is usually some sort of database from which a client requests information. A system consisting of a client and a server is known as a two-tier system.

9. Which of the following functions does the processing tier not handle in a three-tier client/server system?

   a. processing and calculations

b. reading and writing of information to the data storage tier

c. the return of any information to the client tier

d. data storage

**Answer:** D

**Feedback**: A three-tier client/server system—also known as a multitier client/server system or n-tier client/server system—consists of three distinct pieces: the client tier, the processing tier, and the data storage tier.

10. Which of the following uses the correct case?

a. `Document.write()`

b. `document.write()`

c. `document.Write()`

d. `Document.Write()`

**Answer:** B

**Feedback**: JavaScript objects and methods are written in all lowercase letters.

11. Which of the following is *not* a valid identifier?

a. $InterestRate

b. 2QInterest Rate

c. interestRate

d. _interestRate

**Answer**: B

**Feedback**: You can use numbers in an identifier but not as the first character.

12. When you assign a specific value to a variable on its creation, you _____ it.

a. declare

b. call

c. assign

d. initialize

**Answer**: D

**Feedback**: Before you can use a variable, you declare it which creates the variable for storing data and objects; optionally, the variable can be initialized which assigns it an initial value.

13. Code that tells a browser what to do in response to a specific event on a specific element is called a(n) _____.

a. method

b. event handler

c. response

d. procedure

**Answer**: B

**Feedback**: When an event occurs, your script executes any code that responds to that specific event on that specific element. This code is known as the event handler.

14. Which method displays a dialog box with an OK button?

a. `document.write()`

b. `document.writeln()`

c. `window.alert()`

d. `window.popup()`

**Answer**: C

**Feedback**: The `window.alert()` method displays a dialog box with an OK button.

15. Which of the following is not a JavaScript keyword used to declare a variable?

a. `variable`

b. `var`

c. `let`

d. `const`

**Answer**: A

**Feedback**: Variables are declared using the `var`, `let`, or `const` keywords. If the script references page elements which have not been loaded by the browser an error will result and the script will fail to run.

16. What potential problems can occur if you load a script prior to the page being entirely loaded by the browser?

**If the JavaScript program attempts to reference a page element prior to that element being loaded by the browser, an error will result.**

17. How can you make the browser not parse and load an external script file until after the page has loaded?

**Add the `defer` attribute to the `<script>` tag.**

18. When should you use an external JavaScript file instead of embedding your JavaScript code within the HTML file?

**There are several situations where this is preferable: a) When you need to share code among several different web pages, b) When you working in a team with different team**

**members responsible for different aspects of the website development (such as content, design and layout, and programming) and you want to keep those files separate, and c) In order to dedicate documents to the tasks for which they are design with HTML files solely focused on page content and structure, CSS files focused only on page layout and design, and JavaScript files dedicated only to programming-related tasks.**

19. Provide the JavaScript code to write the text "Copyright 2023" as a line comment. Provide the code to write the same text as a block comment.

```
// Copyright 2023
/*
   Copyright 2023
*/
```

20. What is a library?

**In addition to storing scripts for multiple pages in the same website, sometimes JavaScript source files store especially useful generic scripts used on many different websites. These files, known as libraries, are often developed by a single programmer or a team of programmers and distributed online. Many libraries are developed to solve a problem on one website and turn out to be useful for other sites as well. Programmers often make libraries available for free reuse.**

## Hands-On Project Solutions

### Hands-On Project 1-1

See project01-01.html in the js01/project01 folder posted on the Instructor Companion Site.

### Hands-On Project 1-2

See project01-02.html and project01-02.js in the js01/project02 folder posted on the Instructor Companion Site.

### Hands-On Project 1-3

See project01-03.html in the js01/project03 folder posted on the Instructor Companion Site.

### Hands-On Project 1-4

See project01-04.html and project01-04.js in the js01/project04 folder posted on the Instructor Companion Site.

### Hands-On Project 1-5 (Debugging Challenge)

See project01-05.html in the js01/project05 folder posted on the Instructor Companion Site.

## Case Project Solutions

### Individual Case Project

The Individual Case Project for each chapter in this book will build on a website that you create on a subject of your choice. To begin, choose a topic for your website. This can be a topic related to your major, or a personal interest or activity. Plan a website containing at least four pages with a common

layout and navigation system. Note that you'll add pages to your site in later chapters, so ensure that your navigation system can support additional content. Ensure that all of your web pages pass validation.

**Grading rubric:**

- Website containing at least four HTML documents, all incorporating a common navigation system
- All pages must pass validation.

## Team Case Project

Working in a team of 4–8 people, discuss and agree on a topic for your website. This may be a topic related to your major, another area of study, your college or university, or a shared interest. Work together to plan a website containing, at a minimum, a number of pages equal to the number of group members and to create a common layout and navigation system. Note that you'll add pages to your site in later chapters, so ensure that your navigation system can support additional content. Decide as a group who will create which page and create the pages individually. When you've finished creating the individual pages, ensure they pass validation and then work together to assemble the resulting website, identifying and fixing any issues as a group.

**Grading rubric:**

- Website containing at least as many HTML documents as there are group members, all incorporating a common navigation system
- All pages must pass validation.

# Instructor Manual

Carey/Vodnik, *JavaScript for Web Warriors* 7e, 2022, 9780357638002; Chapter 1: Introduction to JavaScript

## Table of Contents

## Purpose and Perspective of the Chapter

The purpose of this chapter is to provide a rapid introduction to JavaScript for readers already familiar with HTML and CSS. It begins with a refresher/crash course on the context for JavaScript: website-related programming languages, JavaScript's background, concepts such as the DOM and BOM, and client/server architectures and scripting for web development. After a brief mention of tooling choices (IDEs versus code editors), the chapter then covers an array of basic concepts, introducing some JavaScript syntax through many code examples. The reader learns about, for instance, the basics of using `<script>` elements to embed or link JavaScript; object terminology; the `document.write()`, `window.alert()`, and `document.getElementById()` methods; using comments; simple operations with variables, including expressions; and the purpose and basic syntax of event handlers.

## Cengage Supplements

The following product-level supplements provide additional information that may help you in preparing your course. They are available in the Instructor Resource Center.

- Test Bank: Cengage Testing, powered by Cognero® (contains assessment questions and problems)
- PowerPoint (provides text-based lectures and presentations)
- Master List of Learning Objectives (provides a comprehensive list of learning objectives for all modules)
- JavaScript Data and Solutions Files
- MindTap Educator Guide (contains a detailed outline of the MindTap)
- Guide to Teaching Online (includes pedagogical considerations and resources for teaching online)

## List of Student Downloads

Students should download the following items from the Student Companion Center to complete the activities and assignments related to this chapter:

- JavaScript Data Files
- JavaScript Snippets ReadMe (provides helpful information related to the embedded coding Snippets)

## Chapter Objectives

The following objectives are addressed in this chapter:

1.1 Explain the history of JavaScript and scripting languages and how each has been developed for its current use

1.2 Write content into a web page using JavaScript

1.3 Add JavaScript code to a web page

1.4 Create and apply JavaScript variables

1.5 Work with event handlers within a web page

1.6 Connect to an external JavaScript File

## Complete List of Chapter Activities and Assessments

For additional guidance refer to the Teaching Online Guide.

| Chapter Objective | PPT slide | Activity/Assessment | Duration |
|---|---|---|---|
| 1.3 | 30 | Activity 1.1: Knowledge Check 1 | 5 minutes |
| 1.2, 1.4 | 30 | Activity 1.1: Knowledge Check 2 | 5 minutes |
| 1.3, 1.4, 1.5, 1.6 | 46 | Activity 1.2: Think, Pair, and Share | 15-20 minutes |
| 1.1 | 47 | Activity 1.3 Discussion Question 1 | 15-20 minutes |
| 1.3, 1.4, 1.5 | 47 | Activity 1.3 Discussion Question 2 | 15-20 minutes |
| 1.1 – 1.6 | 48 | Self-Assessment | 30 minutes |
| 1.3 | | Web Page Evaluation | 30-35 minutes |
| 1.2, 1.3 | | Personal Profile Page | 10-15 minutes |
| 1.1 | | Quick Quiz 1 | 5 minutes |
| 1.3 | | Quick Quiz 2 | 5 minutes |
| 1.4 | | Quick Quiz 3 | 5 minutes |
| 1.4 | | Quick Quiz 4 | 5 minutes |
| 1.5 | | Quick Quiz 5 | 5 minutes |
| 1.3 | | Quick Quiz 6 | 5 minutes |
| 1.6 | | Quick Quiz 7 | 5 minutes |
| 1.3 | | Quick Quiz 8 | 5 minutes |
| 1.1 – 1.6 | | Chapter 1 Review Questions | 30-35 minutes |
| 1.1 – 1.6 | | Chapter 1 Hands-On Projects | 20-40 minutes / Hands-On Project |

**[return to top]**

## Key Terms

**Application Programming Interface (API):** a set of procedures that access an application such as a web page or a web browser.

**argument:** specific information that must be provided to a method, which is placed between the parentheses after the method name.

**assignment operator:** the equal sign. The equal sign in the declaration statement is called an assignment operator because it assigns the value on the right side of the expression to the variable on the left side of the expression.

**back end:** see *server*.

**block comment:** a comment that hides multiple lines of code. You create a block comment by adding `/*` to the first line that you want included in the block, and you close a comment block by typing `*/` after the last character in the block.

**Browser Object Model (BOM):** the Application Programming Interface that describes how to access the features and behaviors of the browser itself.

**camel case:** a method of capitalization that uses a lowercase letter for the first letter of the first word in a variable name, with subsequent words starting with an initial cap.

**Cascading Style Sheets (CSS):** a markup language used to define how a web page will appear on a specified device.

**character data (CDATA):** a section of a document that is not interpreted as markup.

**client:** the component of a traditional client/server architecture that presents an interface to the user, submits information to the server, and receives, formats, and presents the results returned from the server; also known as the *front end*.

**client-side scripting:** a form of web programming used for situations in which the scripting language runs on a local browser (on the client tier) instead of on a web server (on the processing tier).

**code editor:** software that simply manages the writing of HTML, CSS, and JavaScript code within a graphical interface.

**comments:** lines of code that are not processed by browsers, which you use to add notes about your code.

**compiled:** in reference to instructions written in a programming language, transformed into machine code that can be understood by the computer or computer device.

**compiler:** a program that transforms instructions written in a programming language into machine code.

**declare:** to create a variable for storing data and objects.

**Document Object Model (DOM):** the Application Programming Interface that describes how to access the contents of the web page and user actions within that page.

**ECMA-262:** the specification for the scripting language developed by the European Computer Manufacturers Association that works across a wide range of devices and browsers; also known as *ECMAScript*.

**ECMAScript:** see *ECMA-262*.

**European Computer Manufacturers Association (ECMA):** the organization that develops the standards for the commonly accepted browser scripting language ECMAScript, which forms the core of the JavaScript language.

**event:** a specific circumstance (such as an action performed by a user or an action performed by the browser) that is monitored by JavaScript and that your script can respond to in some way.

**event handler:** code that responds to a specific event on a specific element, and that is executed by the script when that event occurs.

**expression:** a literal value or variable, or a combination of literal values, variables, operators, and other expressions, that can be evaluated by a JavaScript interpreter to produce a result.

**front end:** see *client*.

**Hypertext Markup Language (HTML):** a markup language used to define the content and structure of a web page.

**Hypertext Transfer Protocol (HTTP):** the main system used on the web for exchanging data.

**identifier:** the name you assign to a variable.

**initialize:** to assign a variable that has already been declared an initial value.

**Integrated Development Environment (IDE):** software used to manage all of the facets of website development, including the writing and testing of JavaScript code.

**interpreter:** a program that reads code written in a scripting language line-by-line and scans it for errors as it runs.

**JavaScript:** a programming language that adds complex interactive features to a website.

**JavaScript source file:** a text file saved with a .js extension that contains JavaScript statements.

**JScript:** a variant of JavaScript that was supported by Microsoft's Internet Explorer in the late 1990s.

**keywords:** special words that are part of the JavaScript language syntax and thus cannot be used for identifiers; also known as *reserved words*.

**library:** a JavaScript source file that stores especially useful generic scripts used on many different websites, often developed by a single programmer or team of programmers and distributed online.

**line comment:** a code comment that occupies only a single line or part of a line. To create a line comment, you add two slashes (`//`) before the text you want to use as a comment.

**literal:** a value such as a text string or a number.

**literal string:** text that is contained within double or single quotation marks; also known as a *text string*.

**machine code:** code that can be directly understood by a computer or computer device.

**markup language:** a language that defines the content, structure, and appearance of a document.

**method:** a procedure associated with an object.

**middle tier:** see *processing tier*.

**multitier client/server system:** see *three-tier client/server system*.

***n*-tier client/server system:** see *three-tier client/server system*.

**object:** programming code and data that can be treated as an individual unit or component.

**operand:** any of the variables or literals contained in an expression.

**operator:** a symbol (e.g., the addition or multiplication symbol) used in an expression to manipulate operands.

**parsed character data (PCDATA):** a section of a document that is interpreted as markup.

**passing arguments:** providing one or more arguments for a method.

**procedure:** a group of statements within a computer program representing a logical unit that performs a specific task.

**processing tier:** the component of a three-tier client/server system that handles the interaction between the web browser client and the data storage tier; also known as the *middle tier*.

**programming language:** a set of instructions directing the actions of the computer or computer device.

**property:** a piece of data, such as a color or a name, which is associated with an object.

**reserved words:** special words that are part of the JavaScript language syntax and thus cannot be used for identifiers; also known as *keywords*.

**script:** a computer program written in a scripting language.

**scripting language:** a member of a subcategory of programming languages that do not require compiling but instead are run directly from a program or script. Scripting languages need to be interpreted.

**server:** a device or application from which a client requests information in a traditional client/server architecture. A server fulfills a request for information by managing the request or serving the requested information to the client.

**server-side scripting:** programming using a scripting language that is executed from a web server.

**statement:** one of the individual lines of code that make up a JavaScript program in a document.

**static:** term used to describe a web page that cannot change after it is rendered by the browser.

**text string:** text that is contained within double or single quotation marks; also known as a *literal string*.

**three-tier client/server system:** a client/server system that consists of three distinct pieces: the client tier, the processing tier, and the data storage tier. The client tier, or user interface tier, is still the web browser. However, the database portion of the two-tier client/server system is split into a processing tier and the data storage tier. It is also known as a *multitier client/server system* or an n-*tier client/server system*.

**two-tier system:** a system consisting of a client and a server.

**validating parser:** a program that checks whether a web page is well formed and whether the document conforms to a specific language definition known as a DTD.

**validation:** the process of verifying that your document is well formed and checking that the elements in your document are correctly written according to the element definitions in a specific DTD.

**variables:** the values a program stores in computer memory; technically speaking, specific locations in the computer's memory where data are stored.

**web:** a technology developed in the mid-1990s to share data across a network of linked documents; also known as the *World Wide Web*.

**web application:** a program that is executed on a server but is accessed through a web page loaded in a client browser.

**well formed:** term used to describe an HTML document that conforms to the rules and requirements of the language.

**World Wide Web:** see *web*.

**World Wide Web Consortium (W3C):** the organization that manages the specifications for the DOM and the development of HTML and CSS.

[return to top]

## What's New in This Chapter

The following elements are improvements in this chapter from the previous edition:

**New and Expanded Topics**

- Added coverage of ECMAScript through ES11
- Moved overview of Document Object Model (DOM) and Browser Object Model (BOM) into Chapter 1
- Updated coverage of IDEs and code editors to reflect current software development.
- Added coverage of the let and const keywords for defining variables and constants.
- Added coverage of `defer` and `async` for loading external scripts.

**New and Revised Projects**

- Updated and simplified the chapter project code.
- Updated and simplified the end-of-chapter projects.

[return to top]

## Chapter Outline

**Exploring the JavaScript Language (1.1)**

I. Introduction (1.1)

   a. Define **JavaScript** ass a programming language that adds complex interactive features to a website.
   b. Briefly discuss JavaScript's place in the history of the **World Wide Web** (**web**).

II. Introducing Scripting Languages (1.1)

   a. Overview how, in general, **programming languages** are **compiled** into **machine code** by a **compiler** for the computer's use.
   b. Differentiate **scripting languages** from compiled languages, describing how **scripts** are read line-by-line by an **interpreter**.
   c. Differentiate **markup languages** such as **HTML (Hypertext Markup Language)** and **CSS (Cascading Style Sheets)** from other types of languages.

d. Use Figure 1-1 to summarize how HTML, CSS, and JavaScript code work together to create web pages and apps.

III. JavaScript and ECMAScript (1.1)
   a. Discuss the history of JavaScript and its variants such as **JScript**.
   b. Explain the **European Computer Manufacturers Association**'s **(ECMA**'s**)** development of **ECMAScript** (**ECMA-262**) using Figure 1-2.

IV. The DOM and BOM (1.1)
   a. List and describe the three foundations upon which the full implementation of JavaScript is built: ECMAScript, the **Document Object Model (DOM)**, and the **Browser Object Model (BOM)**.
   b. Using the DOM and BOM as examples, discuss **Application Programming Interfaces (APIs)**.
   c. Using Figure 1-3, compare the specifications for the DOM as managed by the **World Wide Web Consortium (W3C)** with the various implementations of the BOM.

V. Understanding Client/Server Architecture (1.1)
   a. Use Figure 1-4 to describe the **two-tier system** consisting of a **client** (**front end**) and a **server** (**back end**) upon which the web is built.
   b. Discuss how data is exchanged on the web using **Hypertext Transfer Protocol (HTTP)**.
   c. Use Figure 1-5 to explain how a **three-tier client/server system** (a.k.a. **multitier client/server system** or ***n*-tier client/server system**) evolves the two-tier system by adding a **processing tier** (**middle tier**), which may reside on the same hardware as the data storage tier and which handles the interaction between the client and data tier.

VI. JavaScript and Client-Side Scripting (1.1)
   a. Discuss how the **client-side scripting** JavaScript provides adds interactivity to **static** web pages written in HTML.
   b. Summarize the security concerns and consequent restrictions surrounding JavaScript programs and their access to related system components.

VII. Understanding Server-Side Scripting (1.1)
   a. Use Figure 1-6 to explain the purpose of **server-side scripting** and the cooperation between server-side and JavaScript programs during web page use.

VIII. Should You Use Client-Side or Server-Side Scripting? (1.1)
   a. Describe the general-purpose division of labor between client- and server-side scripts: allow the client to handle the user interface processing and light processing, but have the web server perform intensive calculations and data storage.
   b. Discuss the reasons for performing as much processing as possible on the client: more powerful **web applications** made possible by distributed

processing among multiple clients; faster applications; and lower infrastructure and power costs.

**Writing a JavaScript Program (1.2, 1.3)**

I. IDEs and Code Editors (1.2, 1.3)
   a. Demonstrate the key features of one or more popular **Integrated Development Environments (IDEs)**.
   b. Demonstrate the key features of one or more popular **code editors**.
   c. Introduce the sample web page used for this chapter's exercises by presenting Figure 1-7.

II. The `script` Element (1.3)
   a. Using Figure 1-8, demonstrate how the `script` element is used to add JavaScript to a web page.

III. JavaScript Statements (1.3)
   a. Describe JavaScript **statements**, pointing out the use of the ending semicolon by convention.

IV. Understanding JavaScript Objects (1.3)
   a. Introduce the basic terminology of object-based programming: **objects** that combine pieces of data, or **properties**, with **procedures** to perform tasks, or **methods**.
   b. Present the basic syntax for JavaScript methods with and without **arguments**, discussing the purpose of **passing arguments**.

V. Using the `write()` Method (1.2)
   a. Explain the function, best practice use, and potential pitfalls of the `Document` object's `write()` method.
   b. Demonstrate the use of the `write()` method using Figures 1-9 and 1-10, noting that it requires a **text string** (**literal string**) as an argument.

VI. Case Sensitivity in JavaScript (1.2, 1.3)
   a. Note that JavaScript object names must be all lowercase.
   b. Present examples of errors that can occur with JavaScript due to inconsistent capitalization.

VII. Adding Comments to a JavaScript Program (1.3)
   a. Explain the typical uses of and best practices for code **comments**.
   b. Demonstrate how to add **line comments** and **block comments** to JavaScript code using Figure 1-11.

**Writing Basic JavaScript Code (1.4)**

I. Using Variables (1.4)

a. Define **variables** as specific computer memory locations that store program data, which often changes.

II. Assigning Variable Names (1.4)
   a. State the rules and conventions for variable **identifiers** (names), using Figure 1-12 to present the **reserved words** (**keywords**) that cannot be used.
   b. Present sample identifiers and point out that it is best practice to use **camel case**.

III. Declaring and Initializing Variables (1.4)
   a. Distinguish between **declaring** and **initializing** variables.
   b. Using Figure 1-13, demonstrate the use of the `let`, `var`, and `const` keywords with an **assignment operator** to declare or declare and initialize variables, noting the differences among these keywords.

**Building Expressions with Variables (1.4)**

I. Building an Expression (1.4)

   a. Describe JavaScript **expressions**, including the **operands** (variables and/or **literals**) and **operators** involved.
   b. Present several examples of expressions using the assignment (=) and addition (+) operators, pointing out the different results when using the addition operator with strings, numbers, and string-number combinations.

II. Modifying Variables (1.4)
   a. Demonstrate how variable values can be changed repeatedly using expressions.

**Understanding Events (1.5)**

I. Introduction (1.5)

   a. Describe the types of **events** monitored by JavaScript.
   b. Compare the device availability of and triggers for several JavaScript events using Figure 1-14.

II. Working with Elements and Events (1.5)
   a. Present the syntax for a JavaScript event handler added as an attribute of the element.
   b. Use Figure 1-15 to overview the types of event attributes that can be added to various HTML elements.
   c. Introduce the purpose and results of the `window.alert()` method using Figure 1-16.
   d. Demonstrate how to use event attributes with the `window.alert()` method to display messages to the user.

III. Referencing Web Page Elements (1.5)

a. Describe how to obtain and manipulate HTML element values using the `document.getElementById()` method.
b. Demonstrate the use of this method within an event handler using Figures 1-17 and 1-18.

**Structuring JavaScript Code (1.2, 1.3)**

I. Including a `script` Element for Each Code Section (1.3)

a. Present an HTML file with two script sections and its output using Figure 1-19.
b. Discuss the connections among the script sections in a single HTML file.

II. Placing the `script` Element (1.2, 1.3)

a. Explain the best practices for placing script sections within HTML files along with the reasons behind them.

**Creating a JavaScript Source File (1.6)**

I. Introduction (1.6)

a. Explain how a **JavaScript source file** is created and saved.

II. Referencing an External File (1.6)

a. Present the HTML syntax for attaching a web page to a JavaScript source file.
b. Point out that separate `script` elements must be used for embedding JavaScript and referencing an external JavaScript file.

III. Using the `async` and `defer` Keywords (1.6)

a. Use Figure 1-20 to explain how adding the `async` or `defer` attribute to a `script` element in an HTML file alters the sequencing of tasks as the web page loads.
b. Discuss the benefits of placing JavaScript code in an external file rather than embedding it within an HTML file.

IV. Connecting to a JavaScript File (1.6)

a. Present the sample JavaScript file in Figure 1-21.
b. Demonstrate how to connect an HTML file with this JavaScript file using Figure 1-22.

V. Working with Libraries (1.6)

a. Discuss the origin and utility of popular JavaScript **libraries**.
b. Present the syntax for incorporating a third-party library using a `script` element.

**Validating Web Pages (1.2, 1.3)**

I. Describe the qualities of **well-formed** HTML documents.

II. Demonstrate the use of a **validating parser** to perform the **validation** process, explaining the results.
III. Explain the distinction between **character data** (**CDATA**) and **parsed character data** (**PCDATA**) and why this is important when working with XHTML instead of HTML.
IV. Demonstrate the use of a CDATA section to prevent the parsing of embedded JavaScript in an XHTML document.

[return to top]

## Discussion Questions

You can assign these questions several ways: in a discussion forum in your LMS; as whole-class discussions in person; or as a partner or group activity in class.

1. How does the Internet affect your daily life? Try to imagine and describe how it would be different if the Internet ceased to be available to you. (1.1)
2. What JavaScript characteristics do you consider most and least helpful to you as a programmer based on what you have learned so far? Explain why. (1.3, 1.4, 1.5)

[return to top]

## Suggested Usage for Lab Activities

1. Coding Snippets, ungraded (embedded in the MindTap Reader):
    a. `script` Elements
    b. The JavaScript Source file

[return to top]

## Additional Activities and Assignments

1. **Web Page Evaluation**: Open a web page you visit often. View the underlying code for the web page and copy it into a text document. Examine the code to determine whether the web page represents a well-formed document. Write a summary report that includes the URL you visited, your observations about the code, and the method you used to evaluate it. Submit your report along with the web page source code. (1.3)
2. **Personal Profile Page:** Using a text editor or IDE, create a well-formed web page document that displays your name and some other basic information about you that you would like to share with the class. (1.2, 1.3)

**Quick Quiz 1**

1. In the web's two-tier client/server system, the client is an application called a(n) _____. (1.1)
   Answer: web browser

2. True or False: HTML and CSS are common web page scripting languages that are interpreted as they are run. (1.1)
   Answer: False

3. The architecture of the web evolves into a _____ client architecture once you start adding databases and other types of applications to a web server. (1.1)
   Answer: three-tier *or* multitier

4. Writing programs that prepare and process data and handle communication between the client and data storage tiers for a web application using a language such as PHP is known as _____. (1.1)
   Answer: server-side scripting

5. True or False: JavaScript is a client-side scripting language that allows web page authors to develop interactive web pages and sites. (1.1)
   Answer: True

6. The three foundations upon which the full implementation of JavaScript is built are ECMAScript, the _____, and the Browser Object Model. (1.1)
   Answer: Document Object Model *or* DOM

7. A set of procedures that access an application such as a web page or a web browser, such as the Browser Object Model, is known as a(n) _____. (1.1)
   Answer: Application Programming Interface *or* API

**Quick Quiz 2**

1. A JavaScript program is composed of _____ that are contained within the `script` element and that, by convention, end in semicolons. (1.3)
   Answer: statements

2. JavaScript is object based, which means that JavaScript programmers use and create objects with associated data, called properties, and associated procedures, called _____. (1.3)
   Answer: methods

3. Often, you must pass one or more pieces of information, called _____, to a procedure associated with an object so that it will be executed as desired. (1.3)
   Answer: arguments

4. Good JavaScript programmers typically add helpful notes to their source code documents by inserting _____, lines of code that are not processed by browsers. (1.3)
   Answer: comments

**Quick Quiz 3**

1. The values a program stores in computer memory are commonly called _____. (1.4)
   Answer: variables
2. The name you assign to a variable is called a(n) _____. (1.4)
   Answer: identifier
3. Special words that are part of the JavaScript language syntax are called ____. (1.4)
   Answer: reserved words *or* keywords
4. True or False: Identifiers must begin with an ASCII letter. (1.4)
   Answer: False
5. True or False: The plus sign (+) is called the assignment operator. (1.4)
   Answer: False
6. True or False: JavaScript is case sensitive. (1.4)
   Answer: True

**Quick Quiz 4**

1. A combination of literal values, variables, and operators that can be evaluated by a JavaScript interpreter to produce a result is called a(n) _____. (1.4)
   Answer: expression
2. In the statement `totalDue = totalMerch + salesTax + 8`, `totalMerch` and `8` are examples of _____. (1.4)
   Answer: operands

**Quick Quiz 5**

1. JavaScript code that is executed in response to a specific event for a specific web page element is a(n) _____. (1.5)
   Answer: event handler
2. The Document Object Model allows you to reference any element on a web page using the value assigned to its HTML _____ attribute. (1.5)
   Answer: `id`

**Quick Quiz 6**

1. True or False: Statements in one script section of an HTML file are accessible to all subsequent script sections. (1.3)
   Answer: True
2. True or False: Because multiple scripts placed within an HTML file can access one another, they can be positioned anywhere in the file without a risk for errors during page loading. (1.3)
   Answer: False

**Quick Quiz 7**

1. True or False: To access JavaScript code that is saved in an external file, a programmer uses the `js` attribute of the `<script>` element. (1.6)
   Answer: False
2. To tell the browser to parse the HTML and JavaScript code for a page together, add the _____ attribute to the opening `<script>` tag referencing an external `.js` file. (1.6)
   Answer: `async`

**Quick Quiz 8**

1. A(n) _____ is a program that checks whether a web page is well formed and whether the document conforms to a specific language definition known as a DTD. (1.3)
   Answer: validating parser
2. True or False: A section of a document that is not interpreted as markup is referred to as character data, or CDATA. (1.3)
   Answer: True

**[return to top]**

# Additional Resources

## Cengage Video Resources

- No MindTap Videos for Chapter 1.

## Internet Resources

- CERN: A short history of the Web: **https://home.cern/science/computing/birth-web/short-history-web**
- MDN Web Docs: What is JavaScript?: **https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript**

- ECMA International: ECMA-262 ECMAScript 2020 language specification: **https://www.ecma-international.org/publications-and-standards/standards/ecma-262/**
- MDN Web Docs: Document Object Model (DOM): **https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model**
- W3 Schools: JavaScript and HTML DOM Reference: **https://www.w3schools.com/jsref/default.asp**
- MDN Web Docs: How the Web works: **https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/How_the_Web_works**
- W3 Schools: PHP Tutorial: **https://www.w3schools.com/php/default.asp**
- W3 Schools: ASP and ASP.NET Tutorials: **https://www.w3schools.com/asp/default.asp**
- Ruby: **https://www.ruby-lang.org/en/**
- W3 Schools: HTML <script> Tag: **https://www.w3schools.com/tags/tag_script.asp**
- W3 Schools: JavaScript Objects: **https://www.w3schools.com/js/js_objects.asp**
- W3 Schools: HTML DOM write() Method: **https://www.w3schools.com/jsref/met_doc_write.asp**
- MDN Web Docs: Storing the information you need—variables: **https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/Variables**
- MDN Web Docs: Overview of events and handlers: **https://developer.mozilla.org/en-US/docs/Web/Guide/Events/Overview_of_Events_and_Handlers**
- W3 Schools: JavaScript Events: **https://www.w3schools.com/js/js_events.asp**
- Node.js: **https://nodejs.org/en/**
- jQuery: **https://jquery.com**
- Modernizr: **https://modernizr.com**

**[return to top]**

# Appendix

## Generic Rubrics

Providing students with rubrics helps them understand expectations and components of assignments. Rubrics help students become more aware of their learning process and progress, and they improve students' work through timely and detailed feedback. Customize these rubrics as you wish. The writing rubric indicates 40 points and the discussion rubric indicates 30 points.

## Standard Discussion Rubric

| Criteria | Meets Requirements | Needs Improvement | Incomplete |
|---|---|---|---|
| Participation | Submits or participates in discussion by the posted deadlines. Follows all assignment. instructions for initial post and responses.<br>5 points | Does not participate or submit discussion by the posted deadlines. Does not follow instructions for initial post and responses.<br>3 points | Does not participate in discussion.<br>0 points |
| Contribution Quality | Comments stay on task. Comments add value to discussion topic. Comments motivate other students to respond.<br>20 points | Comments may not stay on task. Comments may not add value to discussion topic. Comments may not motivate other students to respond.<br>10 points | Does not participate in discussion.<br>0 points |
| Etiquette | Maintains appropriate language. Offers criticism in a constructive manner. Provides both positive and negative feedback.<br>5 points | Does not always maintain appropriate language. Offers criticism in an offensive manner. Provides only negative feedback.<br>3 points | Does not participate in discussion.<br>0 points |