JavaScript, Sixth Edition

Chapter 4 Solutions

Short Quiz 1

* 1. Explain the difference between syntax errors, run-time errors, and logic errors. Provide an example of each.

Syntax errors occur when the interpreter fails to recognize code. For example, if a programmer attempts to use a method that does not exist or omits a method’s closing parenthesis, the scripting engine generates a syntax error.

A run-time error occurs when the JavaScript interpreter encounters a problem while a program is executing. Run-time errors differ from syntax errors in that they do not necessarily represent JavaScript language errors. Instead, run-time errors occur when the interpreter encounters code that it cannot execute. For example, consider the statement createRecommendation();, which calls a custom JavaScript function. This statement does not generate a syntax error, because it is legal (and usually necessary) to create and then call custom functions in a JavaScript program. However, if your program includes the call statement but does not include code that creates the function in the first place, your program generates a run-time error. The error occurs when the interpreter attempts to call the function and is unable to find it.

A logic error is a flaw in a program’s design that prevents the program from running as you anticipate. One example of a logic error is the creation of an infinite loop, in which a loop statement never ends because its conditional expression is never updated or is never false.

* 1. Where can you find error messages in a browser?

In the browser’s console.

* 1. Suppose your browser console lists a single error, which you find and fix. Why is it important to save your work and reload the page in the browser?

Generally when a browser encounters a syntax error, the browser is unable to process any JavaScript code that follows the error. Because there are a number of possible reasons for—and ways of fixing—any given error, the processor can no longer reliably parse the code that comes after the error. Therefore, saving and reloading a page after fixing an error may uncover additional errors.

Short Quiz 2

1. What are the two different statements you can add to your code to provide you with additional information while you’re debugging?

window.alert()

console.log()

1. What statement would you use to log the text “itemTotal: ” plus the value of the itemTotal variable to the console?

console.log("itemTotal: " + itemTotal);

1. When is commenting out code useful in debugging?

Commenting out code is useful when you have narrowed down an error to a specific part of the code but don’t know exactly which line or lines contain the error.

Short Quiz 3

1. What is a breakpoint? How do you set a breakpoint?

A breakpoint is a designation added to a specific statement in a program that causes program execution to pause when it reaches that statement. To set a breakpoint, you click the line number for the line where program execution should stop.

1. Explain the differences between stepping in (or into), stepping over, and stepping out.

Stepping in or stepping into,executes an individual line of code and then pauses until you instruct the debugger to continue. This feature gives you an opportunity to evaluate program flow and structure as code is being executed.

As you use the Step Into button to move through code, the debuggers stop at each line within every function of the JavaScript program. However, when stepping through a program to trace a logical error, it is convenient to be able to skip functions that you know are functioning correctly. The second option, known as stepping over,allows you to skip function calls. The program still executes each function that you step over, but it appears in each debugger as if a single statement executes.

The final option, stepping out, executes all remaining code in the current function. If the current function was called from another function, all remaining code in the current function executes and the debugger stops at the next statement in the calling function.

1. What is the call stack? How do you use it in debugging?

The call stack is the ordered list maintained by a JavaScript processor containing all the procedures, such as functions, methods, or event handlers, that have been called but have not yet finished processing. Each time a program calls a procedure, the procedure is added to the top of the call stack and then removed after it finishes executing. The ability to view a list showing the contents of the call stack is very useful when tracing logic errors in large programs with multiple functions. Viewing the call stack, along with using tracing commands, makes it easier to locate the specific function causing the problem.

Short Quiz 4

1. When is it necessary to include exception handling in your code? Give an example.

You use exception handling to test any type of input or functionality that is external to a program. The main reason for using exception handling in JavaScript is to evaluate user input.

1. What statement in the code to handle an exception do you use to specify an error message?

throw

1. How do you reference a previously generated error message in a catch() statement?

Whatever argument name you specify when creating the catch statement is the variable

name you use within *statements* to refer to the text of the thrown exception.

Short Quiz 5

1. How can code editors designed for web development help you in identifying errors in your HTML?

These editors automatically highlight syntax errors in HTML, CSS, and JavaScript code as you type. This means that rather than needing to scrutinize each line of code, you can count on the editor to visually draw your attention to any errors it identifies—often even before you test your code.

1. What code would you enter on the command line to declare a variable named [BEG CODE]cost[END CODE] with a value of 75, and then log to the console the result of multiplying the cost variable by 1.2?

var cost = 75;cost\*1.2;[END CODE in BT]

1. Explain how coding in strict mode can help you write better code.

For debugging purposes, it can be useful to develop and test all your code using strict mode. This not only helps you notice issues in your code that may be causing problems, but it also helps you strengthen your coding techniques to use current best practices, rather than relying on some parts of the language that are still technically part of the specification but whose use is not generally recommended

# Review Questions

* + 1. What type of error occurs when the interpreter fails to recognize code?
       1. Debugging
       2. Syntax
       3. Run-time
       4. Logic
    2. \_\_\_\_\_\_ errors are problems in the design of a program that prevent it from running as you anticipate.
       1. Application
       2. Syntax
       3. Logic
       4. Run-time
    3. When a JavaScript interpreter encounters a problem while a program is executing, that problem is called a(n) \_\_\_ error.
       1. application
       2. syntax
       3. logic
       4. run-time
    4. Which of the following statements causes a syntax error?
       1. var firstName = "";
       2. document.write(Available points: " + availPoints);
       3. readyState = true;
       4. "use strict";
    5. Which of the following statements writes the value of the selection variable to the console?
       1. console.log("selection");
       2. document.console("selection");
       3. console.alert(selection);
       4. console.log(selection);
    6. Which of the following for statements is logically incorrect?
       1. for (var count = 10; count <= 0; count++) {

document.write(count);

}

* + - 1. for (var count = 0; count <= 10; count++) {

document.write (count);

}

* + - 1. for (var count = 10; count >= 0; count--) {

document.write (count);

}

* + - 1. for (var count = 5; count >= 0; count--) {

document.write (count);

}

* + 1. Which of the following modes temporarily suspends, or pauses, program execution so that you can monitor values and trace program execution?
       1. Suspend
       2. Step
       3. Break
       4. Continue
    2. Which command executes all the statements in the next function in browser debugging tools?
       1. Step out
       2. Step over
       3. Step
       4. Step in/into
    3. After you throw an error, you use a(n) \_\_\_ statement to handle the error.
       1. try
       2. throw
       3. catch
       4. finally
    4. In **\_\_\_\_\_\_\_\_\_\_**, some features are removed from the JavaScript language, while other features require more stringent syntax.
       1. exception handling
       2. strict mode
       3. debugging tools
       4. debugger mode
    5. Which of the following pieces of information is passed as an argument from a throw statement to a catch statement?
       1. Error number
       2. Error message
       3. Line number
       4. URL
    6. What statement can you add to your code to effectively serve the same role as a breakpoint?
       1. break;
       2. breakpoint;
       3. debug;
       4. debugger;
    7. The watch list in browser debugging tools lets you monitor the value of a(n) \_\_\_\_\_\_ during program execution.
       1. function
       2. exception handler
       3. expression
       4. statement
    8. The \_\_\_\_\_\_\_\_\_\_\_\_\_ is the ordered list maintained by a JavaScript processor containing all the procedures, such as functions, methods, or event handlers, that have been called but have not yet finished processing.
       1. variables list
       2. watch list
       3. strict mode
       4. call stack
    9. Which of the following exception handling code executes regardless of whether its associated try block throws an exception?
       1. throw "Please enter your last name.";
       2. catch(lNameError) {

return false;

}

* + - 1. catch(lNameError) {

window.alert(lNameError)

return false;

}

* + - 1. finally {

lNameValid = true;

}

* + 1. What is the advantage of tracing errors using the window.alert() method? What is the advantage of using the console.log() method instead?

The window.alert() method lets you monitor values as they change during program execution.

The console.log() method lets you trace a bug in your program by analyzing a list of values rather than by trying to interpret the values displayed in alert dialog boxes on a case-by-case basis.

* + 1. Explain how to debug code by commenting it out.

Another method of locating bugs in a JavaScript program is to identify lines that you think may be causing problems and transform them into comments. To do so, you simply add // to the start of a single line, or /\* to the start of a block and \*/ to the end of the block, just as you would to create any other comment. This process is known as commenting out code.

* + 1. Explain two different ways that a text editor specialized for web development can help you in preventing errors and debugging code.

A code editor designed for web development can highlight both errors in your HTML code and syntax errors in your JavaScript while you type, saving you the trouble of locating and fixing them later.

* + 1. When and why should you use exception handling with your JavaScript programs?

You use exception handling to test any type of input or functionality that is external to a program. The main reason for using exception handling in JavaScript is to evaluate user input.

* + 1. Explain what strict mode is, how to implement it, and how it’s useful in reducing coding errors.

In strict mode, some features are removed from the JavaScript language, while other features require more stringent syntax.

To request that JavaScript processors parse your code in strict mode, you add the following statement to your code:

"use strict";

While coding in strict mode may in fact generate more errors, it can be a helpful debugging tool. Many of the features that strict mode prohibits or requires to be used in a certain way are well known for causing hard-to-find bugs in code that implements them using nonstrict syntax.

# Case Projects

## Individual Case Project

Add exception handling to the code for one of the forms on your personal web site. If your site does not include a form, add one first. Your code should display one or more relevant error messages in an appropriate location. After you finalize your code, write a summary of the debugging methods from this chapter that you used in this project, describing how you used each one in your code.

Grading rubric: Students should submit a web page that incorporates exception handling using try, throw, and catch statements (and, optionally, finally statements). They should also submit a written description of the debugging methods they used when finalizing their code, which may include window.alert(), console.log(), comments, breakpoints, tracing, the call stack, a JavaScript-optimized code editor, HTML validation, analyzing logic, the command line, debugger, or strict mode.

## Team Case Project

Divide your team into two subgroups and assign each group a page from your team project website that includes JavaScript code. Within each subgroup, introduce at least three bugs into the code for the page you’ve been assigned. Exchange documents with the other subgroup, and then work as a team to debug the code provided by the other team. As you debug, record which debugging methods you use, including whether each was helpful in resolving a given issue. When the document works as expected, create a report. For each bug, describe the behavior you expected as well as the erroneous behavior that the bug caused, describe the methods you used to debug it, including whether each method was helpful or not. Also specify the line number or numbers of the code that contained the error, and show the incorrect as well as the corrected code in your report.

Grading rubric: Each team should submit two written summaries of debugging processes—one for each of two subgroups. The report should describe at least three separate bugs, and for each bug, should describe the following:

* The behavior the group expected as well as the erroneous behavior that the bug caused
* The methods the group used to debug it, including whether each method was helpful or not
* The line number or numbers of the code that contained the error
* The incorrect code and the corrected code