Chapter 2

A JavaScript subset for jQuery users

Objectives

Applied

1. Given the specification for a JavaScript application that requires only the skills presented in this chapter, code, test, and debug the application.

Knowledge

1. Describe two ways to include JavaScript in the head of an HTML document.
2. Describe the use of JavaScript comments, including "commenting out" portions of JavaScript code.
3. List the primary rules for creating a JavaScript identifier.
4. Describe the three primitive data types used in JavaScript: numeric, string, and Boolean.
5. Describe the rules for evaluating an arithmetic expression, including order of precedence and the use of parentheses.

Objectives (continued)

6. Describe the use of variable declarations and assignment statements.
7. Describe the use of the \n escape sequence in a string literal.
8. Describe the syntax for referring to a method or property of an object.
9. Describe the way Number, String, and Date objects are created.
10. Given the name of one of the methods or properties that are presented in this chapter for window, document, Textbox, Number, Date, and String objects, describe the use of the method or property.
11. Describe the rules for evaluating a conditional expression, including the use of the isNan function, the order of precedence for logical operators, and the use of parentheses.
12. Describe the flow of control for an if, while, do-while, or for statement.
13. Describe how an index is used to access the data that's stored in an array.

Objectives (continued)

14. Describe the length property of an array.
15. Describe the creation and use of both function expressions and function declarations.
16. Distinguish between local and global variables and describe how the use of strict mode affects the use of variables.
17. Describe the creation and use of event handlers, including an event handler for the load event of the window object.

Two attributes of the script element

```
src
type
```

A script element in the head section that loads an external JavaScript file

```
<script src="calculate_mpg.js"></script>
```

A script element that embeds JavaScript in the head section

```
<head>
    ...
    <script>
        alert("The Calculate MPG application");
        var miles = prompt("Enter miles driven");
        miles = parseFloat(miles);
        var gallons =
            prompt("Enter gallons of gas used");
        gallons = parseFloat(gallons);
        var mpg = miles/gallons;
        mpg = parseInt(mpg);
        alert("Miles per gallon = " + mpg);
    </script>
</head>
```

Terms

- external JavaScript file
- embedded JavaScript

A block of JavaScript code with the comments highlighted

```
/* this application validates a user's entries for
joining our email list */
var $ = function (id) {          // the standard $ function
    return document.getElementById(id);
}
// this function gets and validates the first user entry
var joinList = function () {
    var emailAddress1 = $("email_address1").value;
    var emailAddress2 = $("email_address2").value;

    if (emailAddress1 == "") {
        alert("Email Address is required.");
    } else {
        $("email_form").submit();
    }
}
```

How to code comments

- For a single-line comment, code two slashes followed by the comment.
- For a block comment, enclose the comment in /* and */ symbols.

The basic syntax rules for JavaScript

- JavaScript is case-sensitive.
- Each JavaScript statement ends with a semicolon.
- JavaScript ignores extra whitespace within statements.

How to split a statement over two or more lines

- Split a statement after:
  - an arithmetic or relational operator such as +, -, *, /, =, ==, >, or <
  - an opening brace ({), bracket ([ ), or parenthesis
  - a closing brace (})
- Do not split a statement after:
  - an identifier, a value, or the *return* keyword
  - a closing bracket ( ]) or closing parenthesis

A JavaScript function with the identifiers highlighted

```
var processEntries = function() {
    var investment = parseFloat( $("investment").value );
    var rate = parseFloat( $("annual_rate").value );
    var years = parseInt( $("years").value );
    $("future_value").value =
        calculateFV(investment,rate,years);
}
```

Rules for creating identifiers

- Identifiers can only contain letters, numbers, the underscore, and the dollar sign.
- Identifiers can’t start with a number.
- Identifiers are case-sensitive.
- Identifiers can’t be the same as reserved words.

Valid identifiers in JavaScript

```
subtotal
index_1
$
taxRate
calculate_click
$log
```

Camel casing versus underscore notation

```
taxRate             tax_rate
calculateClick      calculate_click
emailAddress        email_address
```

Naming recommendations for identifiers

- Use meaningful names for identifiers. That way, your identifiers aren’t likely to be reserved words.
- Be consistent: Either use camel casing (taxRate) or underscores (tax_rate) to identify the words within the variables in your scripts.

Terms

- statement
- whitespace
- comment
- comment out
- identifier
- camel casing

Examples of number values

```
15       // an integer
-21      // a negative integer
21.5     // a decimal value
-124.82  // a negative decimal value
-3.7e-9  // floating-point notation for -0.0000000037
```

Examples of string values

```
"JavaScript"    // a string with double quotes
'String Data'   // a string with single quotes
""              // an empty string
```

The two Boolean values

```
true    // equivalent to true, yes, or on
false   // equivalent to false, no, or off
```

Terms

- number data type
- integer
- decimal value
- Infinity
- -Infinity
- floating-point number
- string data type
- string
- empty string
- Boolean data type
- Boolean value

Common arithmetic operators

Operator	Example	Result
`+`	`5 + 7`	`12`
`-`	`5 - 12`	`-7`
`*`	`6 * 7`	`42`
`/`	`13 / 4`	`3.25`
`%`	`13 % 4`	`1`
`++`	`counter++`	adds 1 to counter
`--`	`counter--`	subtracts 1 from counter

The order of precedence for arithmetic expressions

Order	Operators	Direction
1	`++`	Left to right
2	`--`	Left to right
3	`*` `/` `%`	Left to right
4	`+` `-`	Left to right

Examples of precedence and the use of parentheses

```
3 + 4 * 5           // Result is 23
(3 + 4) * 5         // Result is 35

13 % 4 + 9          // Result is 10
13 % (4 + 9)        // Result is 0

100 + 100 * 2       // Result is 300
100 + (100 * 2)     // Result is still 300
```

Terms

- numeric expression
- arithmetic operator
- modulus operator
- order of precedence

The most useful assignment operators

```
=
+=
```

How to declare numeric variables

```
var subtotal;                              // one variable
var investment, interestRate, years;       // three variables
```

How to declare variables and assign values to them

```
var subtotal = 74.00;              // subtotal = 74.00
var salesTax = subtotal * .1;      // salesTax = 7.4
```

How to code compound assignment statements

```
var subtotal = 74.95;              // subtotal = 74.95
subtotal += 20.00;                 // subtotal = 94.95
```

Three ways to increment a counter variable

```
var counter = 1;                    // counter = 1
counter = counter + 1;              // counter now = 2
counter += 1;                       // counter now = 3
counter++;                          // counter now = 4
```

A floating-point result that isn't precise

```
var subtotal = 74.95;              // subtotal = 74.95

var salesTax = subtotal * .1;
// salesTax = 7.495000000000001
```

Terms

- variable
- declare a variable
- assign a value to a variable
- assignment statement
- assignment operator
- numeric literal

The concatenation operator for strings

Operator	Example	Result
+	`"Grace " + "Hopper"`	`"Grace Hopper"`
	`"Months: " + 120`	`"Months: 120"`

Escape sequences that can be used in strings

Operator	Description
`\n`	Starts a new line in a string.
`\"`	Puts a double quotation mark in a string.
`\'`	Puts a single quotation mark in a string.

How to declare string variables without assigning values to them

```
var zipCode;                          // one variable
var lastName, state, zipCode;         // three variables
```

How to declare string variables and assign values to them

```
var firstName = "Grace", lastName = "Hopper";
var fullName = lastName + ", " + firstName;
// fullName is "Hopper, Grace"
```

How to code compound assignment statements

With string data

```
var firstName = "Grace", lastName = "Hopper";
var fullName = lastName;
fullName += ", ";
fullName += firstName;
// fullName is "Hopper, Grace"
```

With mixed data

```
var months = 120;
message = "Months: ";
message += months;
// message is "Months: 120"
```

How escape sequences can be used in a string

```
var message =
    "A valid variable name\ncannot start with a number.";
var message =
    "This isn\'t the right way to do this.";
```

How to declare Boolean variables and assign values to them

```
var isValid = false;
```

Terms

- concatenate
- string literal
- escape sequence

Common methods of the window object

```
alert(string)
prompt(string,default)
```

The syntax for calling a method of an object

```
objectName.methodName(parameters)
```

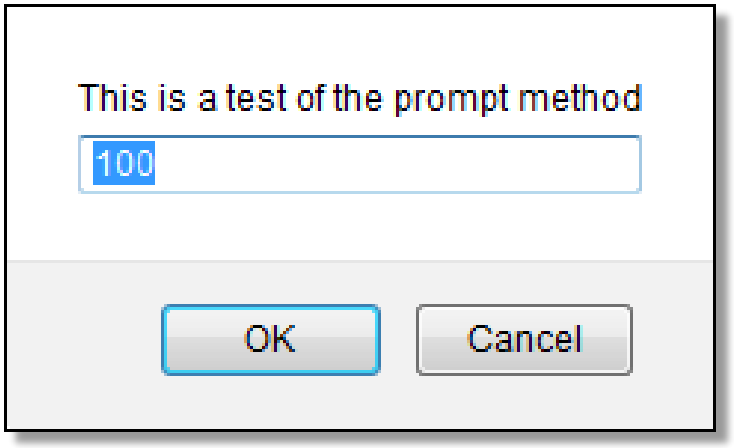
A statement that calls the alert method of the window object

```
window.alert("This is a test of the alert method");
```

A statement that calls the prompt method with the object name omitted

```
var userEntry =
    prompt("This is a test of the prompt method", 100);
```

The prompt dialog box that's displayed



One property of the window object

```
location
```

The syntax for accessing a property of an object

```
objectName.propertyName
```

A statement that displays the location property of the window object

```
alert(window.location);
```

Terms

- object
- method
- property
- call a method
- parameter
- window object
- global object
- dot operator

One method of the document object for getting HTML objects

```
getElementById(id)
```

Examples of the getElementById method

The HTML for two text boxes

```
<label for="miles">Miles Driven:</label>
<input type="text" id="miles" name="miles"><br>
<label for"gallons">Gallons of Gas:</label>
<input type="text" id="gallons" name="gallons"><br>
```

The JavaScript that gets the objects for the text boxes

```
var miles = document.getElementById("miles");
var gallons = document.getElementById("gallons");
```

Two methods of the window object for working with numbers

```
parseInt(string)
parseFloat(string)
```

The parsing that's done by the parse methods

- Only the first number in the string is returned.
- Leading and trailing spaces are removed.
- If the first character cannot be converted to a number, NaN is returned.

Examples of parse methods

```
var miles = prompt("Enter Miles Driven", 1234.567);
miles = parseInt(miles);              // miles = 1234

var gallons = prompt("Enter Gallons of Gas", 24.675);
gallons = parseFloat(gallons);        // gallons = 24.675

var message = "Hello out there!”;
message = parseFloat(message);        // message = NaN
```

Members of the Textbox object

One method

`focus()`

Two properties

`value`

`disabled`

One method of the Number object

`toFixed(`*`digits`*`)`

How to get a string value from a text box

An HTML tag that defines a text box

```
<input type="text" id="first_name">
```

JavaScript without chaining

```
var firstName = document.getElementById("first_name");
firstName = firstName.value;
```

JavaScript with chaining

```
var firstName =
    document.getElementById("first_name").value;
```

How to get a number value from a text box

An HTML tag that defines a text box

```
<input type="text" id="sales_amount">
```

Without chaining

```
var salesAmount =
    document.getElementById("sales_amount");
salesAmount = salesAmount.value;
salesAmount = parseFloat(salesAmount);
```

With chaining

```
var salesAmount = parseFloat(
    document.getElementById("sales_amount").value);
```

Other examples of chaining

```
var salesAmount =
    parseFloat(document.getElementById(
        "sales_amount").value).toFixed(2);

document.getElementById("first_name").value = "";

document.getElementById("first_name").focus();
```

How to create a JavaScript object

The syntax for creating an object

```
new ObjectType();
```

A statement that creates a Date object

```
var today = new Date();
```

A few of the methods of a Date object

```
toDateString()
getFullYear()
getDate()
getMonth()
Examples that use
```

Examples that use a Date object

```
var today = new Date();
alert ( today.toDateString() );
alert ( today.getFullYear() );
alert ( today.getDate() );
alert ( today.getMonth() );
```

Properties and methods of a String object

One property of a String object

`length`

A few of the methods of a String object

`indexOf(search,position)`

`substr(start,length)`

`substring(start,stop)`

`toLowerCase()`

`toUpperCase()`

Examples that use a String object

```
var name = "Grace Hopper";
var nameUpper = name.toUpperCase();
var nameLength = name.length;        // nameLength = 12
var index = name.indexOf(" ");       // index = 5
var firstName = name.substr(0, index);
```