Front End Web Development

8/31/2022

can code communities

JavaScript Introduction

JavaScript is one of the 3 languages all web developers must learn:

1. HTML to define the content of web pages

2. CSS to specify the layout of web pages

3. JavaScript to program the behavior of web pages

JavaScript Introduction

- JavaScript Can Change HTML Content
- JavaScript Can Change HTML Attribute Values
  - Ex. the src att of an <img> tag
- JavaScript Can Change HTML Styles (CSS)
- JavaScript Can Hide HTML Elements
  - By changing the display style
- JavaScript Can Show HTML Elements
  - By changing the display style

JavaScript Values

The JavaScript syntax defines two types of values:

- Fixed values
- Variable values

Fixed values are called Literals.

Variable values are called Variables.

JavaScript Literals

The two most important syntax rules for fixed values are:

1. Numbers are written with or without decimals

2. Strings are text, written within double or single quotes:

JavaScript Variables

In a programming language, variables are used to store data values.

JavaScript uses the keywords var, let and const to declare variables.

An equal sign is used to assign values to variables.

JavaScript is Case Sensitive

All JavaScript identifiers are case sensitive.

The variables lastName and lastname, are two different variables

Where to put JavaScript

- JavaScript code is inserted between <script> and </script> tags.

Where to put JavaScript

You can place any number of scripts in an HTML document.

Scripts can be placed in the <body>, or in the <head> section of an HTML page, or in both.

JavaScript in <head>

In this example, a JavaScript function is placed in the <head> section of an HTML page.

```
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Document</title>
8      <link rel="stylesheet" href="style.css">
9      <script></script>
10 </head>
```

JavaScript in <body>

In this example, a JavaScript function is placed in the <body> section of an HTML page.

```
11  <body>
12
13  <h2>My Web Page</h2>
14  <p>My first paragraph.</p>
15
16  <script>
17  </script>
18
19  </body>
```

External JavaScript

Scripts can also be placed in external files:

- External scripts are practical when the same code is used in many different web pages.
- JavaScript files have the file extension .js.

To use an external script, put the name of the script file in the src (source) attribute of a <script> tag:

```
8     <link rel="stylesheet" href="style.css">
9     <script src="script.js"></script>
```

External JavaScript

Placing scripts in external files has some advantages:

- It separates HTML and code
- It makes HTML and JavaScript easier to read and maintain
- External JavaScript files can speed up page loads

JavaScript Display

JavaScript can "display" data in different ways:

- Writing into an HTML element, using innerHTML.
- Writing into the HTML output using document.write().
- Writing into the browser console, using console.log().

Using document.write

For testing purposes, it is convenient to use document.write():

- document. write can be used in between script tags in your HTML to text out different functions.
- You can use document. write to display numbers as well as do mathematical operations

```
16  <script>
17  document.write(25);
18  </script>
```

Exercise

- Add multiple document.write scripts to your HTML
  - One should display a single number
  - One should add up two numbers
  - One should add up 3 or more numbers
  - One should multiply two numbers
  - One should divide two numbers

Using window.alert()

You can use an window.alert() to show an alert box to display data:

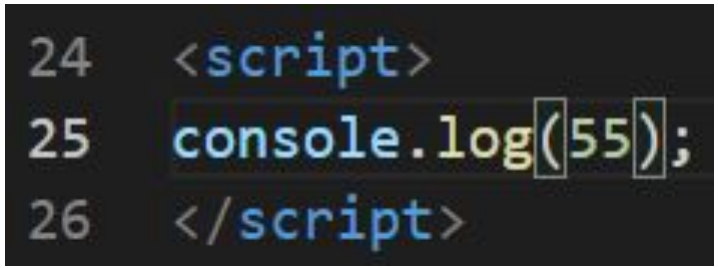
```
20  <script>
21  window.alert(5 + 6);
22  </script>
```

Exercise

- Add a window.alert() script to your page
  - Have one display a single number
  - Have one display two numbers added together
  - Have one display two numbers subtracted

Using console.log()

For debugging purposes, you can call the console.log() method in the browser to display data.



Exercise

- Use multiple document.write() scripts throughout your page with different mathematical functions
- Use multiple window.alert() scripts to display a number in an alert on your page
- Use console.log() to display a number in your console

JavaScript Variables

JavaScript variables are containers for storing data values.

In this example, x, y, and z, are variables, declared with the var keyword:

```
var x = 5;

var y = 6;

var z = x + y;
```

JavaScript Comments

JavaScript comments can be used to explain JavaScript code, and to make it more readable.

JavaScript comments can also be used to prevent execution, when testing alternative code.

Single Line Comments

Single line comments start with //.

Any text between // and the end of the line will be ignored by JavaScript (will not be executed).

Multi-line Comments

Multi-line comments start with /* and end with */.

Any text between /* and */ will be ignored by JavaScript.

Exercise

- Add 3 paragraphs to your page with ids
- Use 4 document.write() scripts throughout your page with different mathematical functions
- Use 2 window.alert() scripts to display a number in an alert on your page
- Use 4 console.log() scripts to display a number in your console

JavaScript Variables

JavaScript variables are containers for storing data values.

In this example, x, y, and z, are variables, declared with the var keyword:

```
var x = 5;

var y = 6;

var z = x + y;
```

Semicolons ;

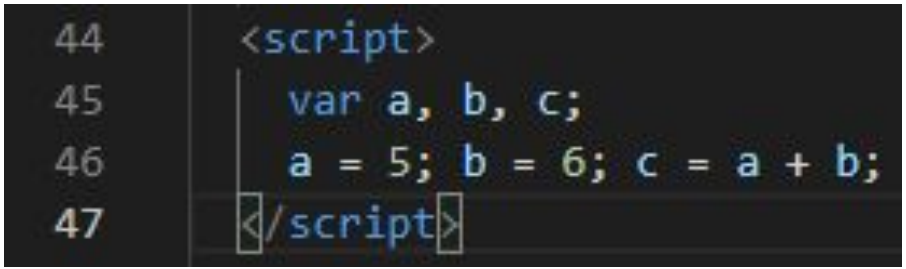
Semicolons separate JavaScript statements.

Add a semicolon at the end of each executable statement:

```
37    <script>
38      var a, b, c;     // Declare 3 variables
39      a = 5;           // Assign the value 5 to a
40      b = 6;           // Assign the value 6 to b
41      c = a + b;       // Assign the sum of a and b to c
42    </script>
```

Semicolons ;

When separated by semicolons, multiple statements on one line are allowed:



Exercises

- Using variables create 5 document.write() outputs
- Using variables create 2 window.alert() outputs
- Using variables create 5 console.log() outputs

Case Sensitivity

All JavaScript identifiers are case sensitive.

The variables lastName and lastname, are two different variables:

```
49  <script>
50      var lastname, lastName;
51      lastName = "Doe";
52      lastname = "Peterson";
53  </script>
```

JavaScript Statements

JavaScript statements are composed of:

Values, Operators, Expressions, Keywords, and Comments.

Most JavaScript programs contain many JavaScript statements.

The statements are executed, one by one, in the same order as they are written.

JavaScript programs (and JavaScript statements) are often called JavaScript code.

JavaScript Programs

A computer program is a list of "instructions" to be "executed" by a computer.

In a programming language, these programming instructions are called statements.

A JavaScript program is a list of programming statements.

In HTML, JavaScript programs are executed by the web browser.

JavaScript Functions

A JavaScript function is a block of code designed to perform a particular task.

A JavaScript function is executed when "something" invokes it (calls it).

Why do we use Functions?

You can reuse code: Define the code once, and use it many times.

You can use the same code many times with different arguments, to produce different results.

JavaScript Function Syntax

A JavaScript function is defined with the function keyword, followed by a name, followed by parentheses ()

The parentheses may include parameter names separated by commas:

(parameter1, parameter2, ...)

The code to be executed, by the function, is placed inside curly brackets: {}

```
function name(parameter1, parameter2, parameter3) {
  // code to be executed
}
```

Function Invocation

The code inside the function will execute when "something" invokes (calls) the function:

- When an event occurs (when a user clicks a button)
- When it is invoked (called) from JavaScript code
- Automatically (self invoked)

Function Return

When JavaScript reaches a return statement, the function will stop executing.

If the function was invoked from a statement, JavaScript will "return" to execute the code after the invoking statement.

Functions often compute a return value. The return value is "returned" back to the "caller":

```
56  <p id="demo"></p>
57
58  <script>
59    function myFunction(p1, p2) {
60      return p1 * p2;
61    }
62    document.getElementById("demo").innerHTML = myFunction(4, 3);
63  </script>
```

Exercise

- Add 10 paragraphs to your code each with a unique ID
- Write out 10 different Functions to replace your each of the paragraphs in your HTML