Unit3 Introduction to Javascript programming

At the end of this unit, the learner should be able to:

- write a correct javascript basic program
- manipulate javascripts varaibles and operators

I. Programming

Programming is about giving the computer a series of instructions in a format it understands. Programs can be a few lines long or millions of lines long, with complex solutions built from simple building blocks. Part of the art of programming is breaking larger programs into smaller pieces that perform specific jobs. The smaller pieces are then combined to fulfill the purpose of the main program. There are many programming languages. Some of them include: Java, C, PHP, Python, JavaScript and so on. In this course the JavaScript programming language will be our concerned.

II. JavaScript

JavaScript is an incredibly popular programming language, mostly seen in web browsers but gaining popularity in other contexts. On web pages it adds interactivity, from simple animation effects to form validation. JavaScript enables you to manipulate objects on the page, such as links, images, and styles. You can also use JavaScript to control the browser itself by changing the size of the browser window, moving the browser window around the screen, and activating or deactivating elements of the interface.

a. The <script> Tag

The <script> tag is used to include a JavaScript script in a web page. The contents of the <script> tag are expected to be JavaScript source code. There are a couple of other ways to use JavaScript in your pages, too, but the <script> tag is a good place to start. For the best results across all browsers, you should include the type attribute in the script tag, which specifies the type of content that the tag contains. For JavaScript, use text/javascript.

b. The Structure of a JavaScript program

When you include any JavaScript code in an HTML document (apart from using
the <script> tag), you should also follow a few other conventions:

- HTML standards prior to HTML5 required that the <script> tag be placed between the <head> and </head> tags at the start of your document, not inside the <body> tag. However, for performance reasons, it’s almost always a better idea to put your <script> tags at the bottom of the page.
- Unlike HTML, which uses the <!-- comment tag -->, comments inside JavaScript code use the // symbol at the start of a line. Any line of JavaScript code that starts with these characters will be treated as a comment and ignored.

Taking these two points into consideration, here’s how the <script> tag is normally used:

```
<html>
<head>
<title>Test script</title>
</head>
<body>
Your Web content goes here
<script>
// Your JavaScript code goes here
</script>
</body>
```

</html>

You can place your <script> tag in either the head or the body of your document. But because JavaScript forces the browser to load it as a single thread, it's best to place it at the bottom of your pages, right before the closing </body> tag. This ensures that your pages' load as quickly as possible.

The following program enable us to display a single javascript windows with the message hello world!

```
</html>
<head>
<title>Test script</title>
</head>
<body bgcolor="orange">
<h1> hello and welcome to my first javascript page</h1>
<script type="text/javascript">
alert(`hellow world!`);
</script>
</body>
</html>
```

c. **The src Attribute**

Besides the language attribute, the <script> tag can also include an src attribute, which allows a JavaScript script stored in a separate file to be included as part of the current web page. This feature enables you to share JavaScript code across an entire website. When used this way, the <script> tag takes the following form:

<script src="http://www.example.com/script.js">

The src attribute will accept any valid URL, on the same server or on another. Naming your JavaScript files with the extension .js is required.

The following code is used when integrating an external javascript file in a normal html page using the src attribute:

```
<script type="text/javascript">
alert(`hellow world!`);
</script>
```

The above javascript code is located in the file **script.js.**

```
<html>
<head>
<title>Test script</title>
</head>
<body bgcolor="orange">
<h1> hello and welcome to my first javascript page</h1>
<script src="/script.js">
</body>
</html>
```

The html code above is located in the file hello.html and stored in the same directory with the script.js file.

III. **Variables**

Like all programming languages, JavaScript supports the use of variables. A *variable* is a user-defined container that can hold a number, text, or an object. A *variable* is a named value in your program. Whenever you use the name in the program, it's replaced with the value. You could create a variable called score and give it the value 100. Then, if you tell the computer to "display the score," it will display 100. Now, variables can change, hence the name, so later in the program, maybe in response to some action a player takes, you can update the score. If you add 50 to score and tell the computer to "display the score," it will now display 150.

So how can you use JavaScript to make this magic happen?

1. **Declaring variables and assigning values**

Letting the computer know about information you'd like to store requires two steps:

- You need to *set a name* you can use to refer to your data in the program, like *score* or *playerName* or *taxRate*.
- You need to *link the name with the value* you want to store: something like *set score equal to 100* or *make 'George' the playerName* or *let the tax rate be 12%*.

a. **Declaring variables**

Registering a name to represent a value is called *variable declaration*. You declare a variable by using the var keyword. The following listing shows the code statement needed to declare a variable called score.

var score;

The var keyword tells the computer to take the next word in the statement and turn it into a variable. Figure 6.1 annotates the code statement

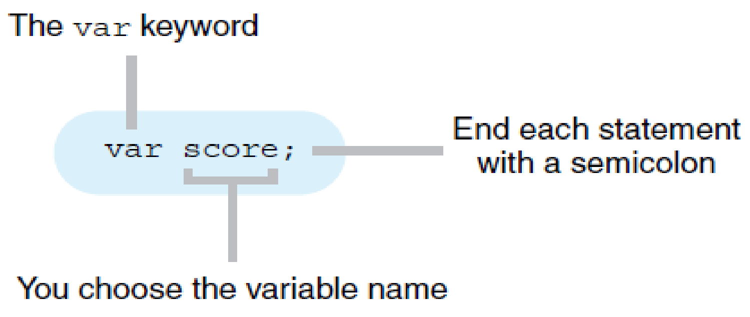

The var keyword
var score;
End each statement with a semicolon
You choose the variable name

Figure6.1 Declaring a variable

b. **Assigning values to variables**

Your program now knows about the variable score. But how do you assign it a value? You use the humble equals symbol, =. Figure 6.2 illustrates the equals symbol at work.

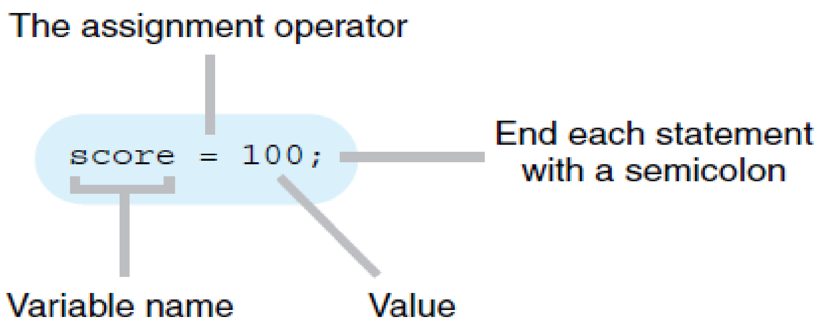


Figure 6.2: Assigning a value to a variable

The Figure 6.3 present both the declaration and assignment.

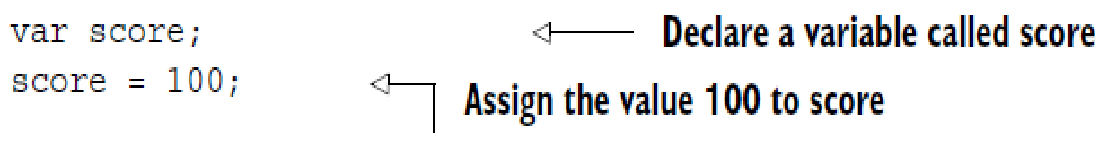

var score;
score = 100;
Declare a variable called score
Assign the value 100 to score

Figure 6.3: Declaration and assignment of a variable.

You assign the variable score the value 100. In general, you assign the value on the right of the equals sign to the variable on the left of the equals sign (figure 6.4). When you use the equals sign to assign a value, JavaScript gives it a special name, the *assignment operator*.

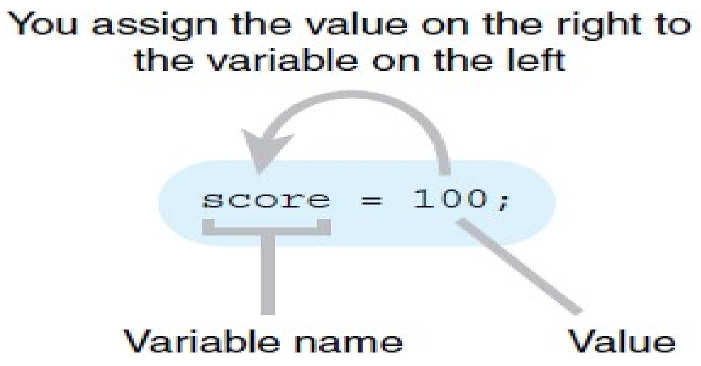
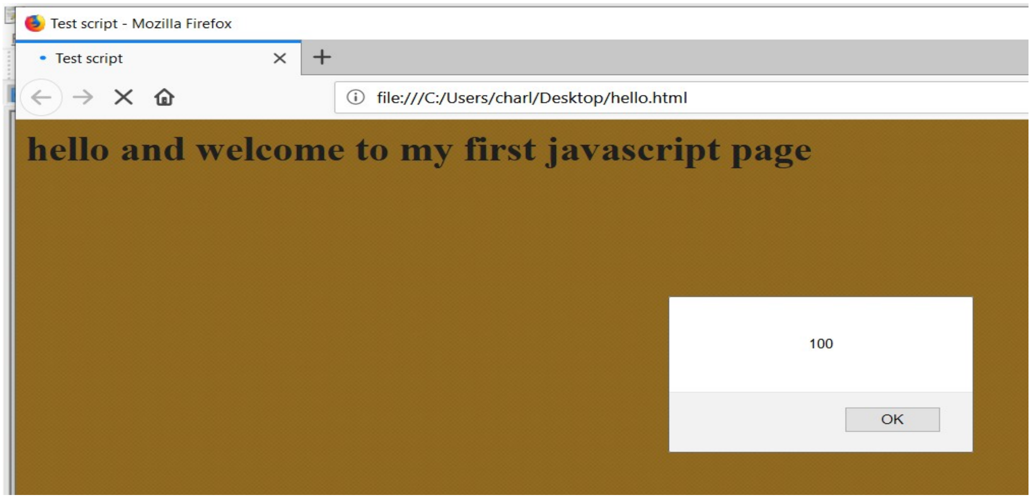

You assign the value on the right to the variable on the left
score = 100;
Variable name
Value

Figure 6.4: The equals sign is called the assignment operator.

You have declared a variable and assigned it a value. It's time to display it on the screen. The code is as follows:

```
<html>
<head>
<title>Test script</title>
</head>
<body bgcolor="orange">
<h1> hello and welcome to my first javascript page</h1>
<script type="text/javascript">
var score;
score=100;
alert(score);
</script>
</body>
</html>
```

Output:


Test script - Mozilla Firefox
Test script
file:///C:/Users/charl/Desktop/hello.html
OK

hello and welcome to my first javascript page

100

Figure6.5: A web page displaying 100 in a window.

There is a possibility to display a message on the screen by joining pieces of text using the addition symbol, +. Joining pieces of text is called *string concatenation* and + is the *string concatenation operator*. The following code illustrates it:

```
<html>
<head>
<title>Test script</title>
</head>
<body bgcolor="orange">
<h1> hello and welcome to my first javascript page</h1>
<script type="text/javascript">
var playerName="Tom";
var town="Bamenda";
alert(playerName +" is located in " +town);
</script>
</body>
</html>
```

You can also personalised the message by asking a user to insert data and display them. The prompt() function enables to collect data typed by the user through the keyboard. The code is below:

```
<html>
<head>
<title>Test script</title>
</head>
<body bgcolor="orange">
<h1> hello and welcome to my first javascript page</h1>
<script type="text/javascript">
var playerName= prompt("enter your name");
var town= prompt("enter your town");
alert(playerName +" is located in " +town);
</script>
</body>
</html>
```

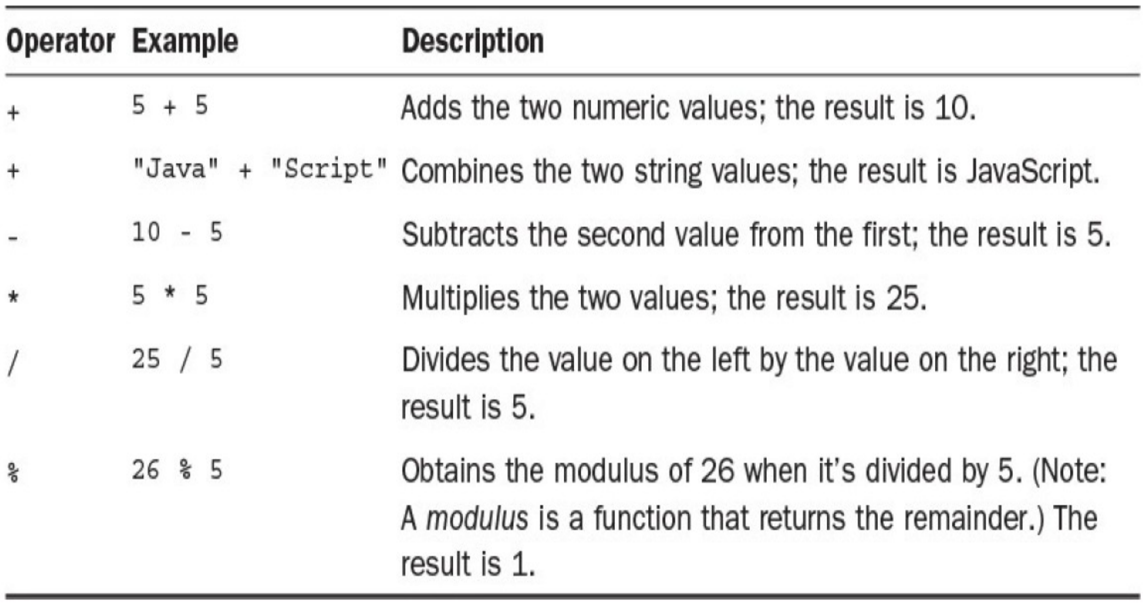
IV. Operators and Expressions

An *expression* is a snippet of code that can be evaluated to return a result. You may recognize the term expression from math, and indeed, many expressions are mathematical expressions. Here's a simple mathematical expression: 10 * 50

In this case, JavaScript will multiply 10 by 50. That's an expression. There are also string expressions. For example, you can use the + operator to join strings together, like this:

"The bird a nest," + " the spider a web," + " man friendship."

Expressions are built using operators. You've already seen a couple: * for multiplication and + for combining strings or adding numbers. Table 6.1 lists more operators provided by JavaScript, along with examples.



Operator	Example	Description
+	`5 + 5`	Adds the two numeric values; the result is 10.
+	`"Java" + "Script"`	Combines the two string values; the result is JavaScript.
-	`10 - 5`	Subtracts the second value from the first; the result is 5.
*	`5 * 5`	Multiplies the two values; the result is 25.
/	`25 / 5`	Divides the value on the left by the value on the right; the result is 5.
%	`26 % 5`	Obtains the modulus of 26 when it's divided by 5. (Note: A *modulus* is a function that returns the remainder.) The result is 1.

Table 6.1 JavaScript Operators and Expressions

The code below add two variable and display the result:

```
<html>
<head>
<title>Test script</title>
</head>
<body bgcolor="orange">
<h1> hello and welcome to my first javascript page</h1>
<script type="text/javascript">
var firstNumber=10;
var secondNumber=50;
var result=firstNumber+secondNumber;
alert(result);
</script>
</body>
</html>
```

You can also collect the two numbers from the user but, you must use the casting function parseInt() so that the number can be converted from text to numerical values.

```
<html>
<head>
<title>Test script</title>
</head>
<body bgcolor="orange">
<h1> hello and welcome to my first javascript page</h1>
<script type="text/javascript">
var firstNumber=prompt("enter your first number");
var secondNumber=prompt("enter your second number");
var result=parseInt(firstNumber)+parseInt(secondNumber);
alert("the result is:"+result);
</script>
</body>
```

</html>
You can also update the current value of a variable as follows:

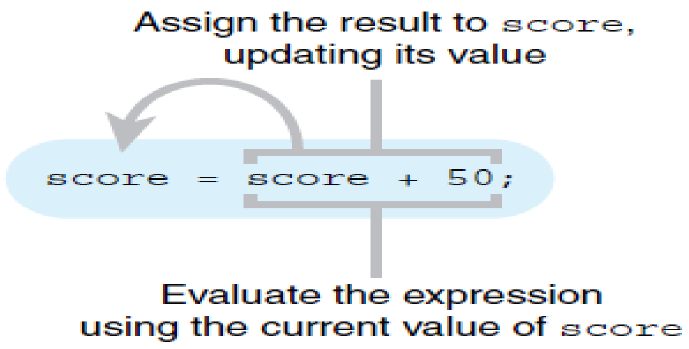

Assign the result to score, updating its value
score = score + 50;
Evaluate the expression using the current value of score

Figure6.6: Updating a variable with the result of a calculation involving itself

Assignment

1. What HTML tag is used to embed JavaScript scripts in a page?
2. Where can we insert the javascript code on a web page?
3. Write an algorithm able to perform the perimeter as well as the area of a rectangle and then and translate it into javascript language. The values of the length and the width are being collected from the keyboard.
4. How to insert comments in javascript and what rules should be followed to have good variable names?
5. What is the difference between a program and an algorithm?
6. What is a data type?
7. List the four primitive data types.