Javascript Basics – Module 1

So far on the course we have looked at HTML and CSS, and also an introduction to programming and coding.

In this module we'll start to focus on JavaScript and how it can be used to interact with web pages to add functionality and effects.

The good news is that JavaScript's syntax is very similar to the code examples that we have already covered.

As usual, if you have any questions then fire away!

Confidential

Review of Previous Module

In the previous module we started to look at coding and programming.

We discussed the different types of programming languages and the difference between programs and scripts.

We also looked a quite a few examples of code and commonly used programming structures such as conditional structures and looping structures.

Topics for this module

In this module we'll start to look at JavaScript, including the following topics

- Introduction to JavaScript
- Internal and external scripts
- Statements and comments
- Variables and data
- Operators
- Conditional structures
- Looping structures

Introduction to Javascript

JavaScript is an interpreted (scripting) language that was specifically created for use with web pages.

It was first created as LiveScript but its name was later changed to JavaScript.

It uses a syntax that is derived fro the C programming language, as is the case with many other high-level languages, and so it is relatively easy to become familiar with.

Introduction to Javascript

First of all it should be noted that JavaScript can be used on the client-side in web browsers and also as a server side language.

Its use as a client side language is by far the most common way in which JavaScript is used, and that is what we will be concentrating on in this course.

In the bootcamps we'll look at the use of server-side JavaScript.

Internal & External Scripts

Like CSS, JavaScript code can be contained in either internal scripts or external scripts.

Internal scripts are defined with opening and closing <script> and </script> tags within the HTML document and external scripts are saved in separate files and then imported into the HTML document for use.

This is a very similar concept to the use of external style sheets, but the syntax for including the external files is slightly different.

Internal & External Scripts

First of all, let’s look at an example of an internal script

```
<html>
  <head>
    <title>Example JavaScript</title>
  </head>
  <body>
    <h1>JavaScript Example</h1>

    <script type="text/javascript">
      window.alert('Hi there folks');
    </script>

  </body>
</html>
```

Internal & External Scripts

In the previous example <script> tags have been used to mark the start at end of a block of JavaScript code. A type attribute has been used in the opening tag with a value of “text/javascript”. This is required in HTML4 but not in HTML5

```
    <script type="text/javascript">
      window.alert('Hi there folks');
    </script>
```

The browser will treat everything between the tags as JavaScript code and execute it at that stage in the process of building the page.

Confidential

Exercise 1

Create a new HTML document named 'internal-javascript-1.html'

Use the following code for the body section.

```
<body>
  <h1>Internal JavaScript</h1>
  <p>Here is paragraph 1...</p>
  <script type="text/javascript">
    window.alert('Hello World');
  </script>
  <p>Here is paragraph 2...</p>
</body>
```

Then open the script in a browser, or access it via a local URL, and see what happens.

Confidential

Exercise 1 – Results

In the exercise you should have seen the following when the page loaded:

- First of all the heading and the first paragraph appeared
- Then the JavaScript code was executed and a pop-up was displayed
- Only when the pop-up was dismissed did the final paragraph appear

The browser executed the JavaScript at the location that it had been include in the HTML document. In this case it displayed a modal window. JavaScript code waits for a modal window to be dismissed before continuing.

Once the JavaScript had finished the browser resumed with the rendering of the page.

Internal & External Scripts

External scripts are incorporated into the HTML document with a slightly modified version of the <script> tags.

In this case no code is included between the tags but instead an attribute is used to reference to the external script.

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

As with internal scripts, and code will be executed at the point in the HTML document that the <script> tag occurs.

For the moment we'll continue to use an internal script for the exercises.

Exercise 2

Create a new HTML document named 'internal-javascript-2.html'

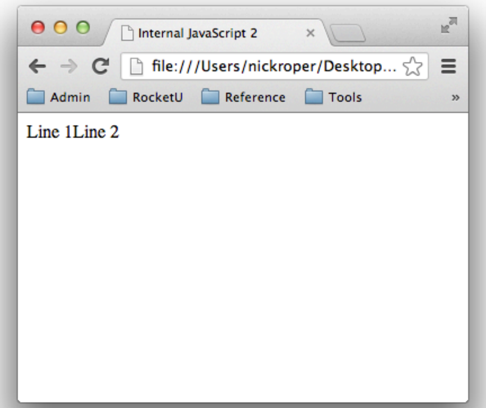
Use the following code for the body section.

```
<body>
  <script type="text/javascript">
    document.write('Line 1');
    document.write('Line 2');
  </script>
</body>
```

Then open the script in a browser, or access it via a local URL, and see what happens.

Exercise 2 – Results

The JavaScript code in exercise 2 displays the following in the browser:



Line 1Line 2

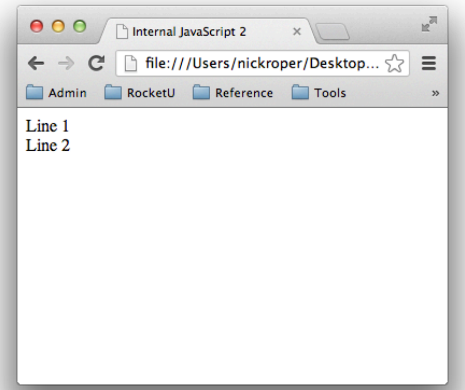
How could we get the text on separate lines?

Exercise 2 – Results

The document.write() statement tells the browser to add some text to the HTML code at that point. It then renders it as though it had been in the original document.

Adding a <br> tag after the first line of text will make the browser add a new line.

```
document.write('Line 1<br>');
document.write('Line 2');
```



Line 1
Line 2

Statements and Comments

JavaScripts code is made up of one or more statements.

By convention, each statement is entered on a separate line.

Each statement should be terminated with a ‘;’ character.

Statements can be indented to improve the readability of the code.

Comments can (and should) be added to help document the code

The next slide shows an example...

Statements and Comments

```
<script>

 // Assign your name to a variable
 var myName = ‘Nick’;

 // Set a counter limit
 var limit = 5;

 // Execute a loop to display output
 for (var count=1; count <= limit; count++) {

   document.write('Hi ' + myName + ' the count is: ' + count + '<br>');

 }

</script>
```

Comments

Terminating ‘;’ characters

Variables and Data Types

Variables are containers for information.

First, the variable should be declared using the following syntax:

```
var myName;
```

This creates the variable but at the moment it has no value.

Values are stored in variable by assigning them to the variable:

```
myName = ‘Nick’;
```

A variable can be created and assigned with a value in a single statement:

```
var myName = ‘Nick’;
```

Variables and Data Types

It is also possible to declare and assign more than one variable in a single statement:

```
var title=‘The Hobbit’, author=‘Tolkien’, genre=‘Fantasy’;
```

The statement can also be spilt over separate lines:

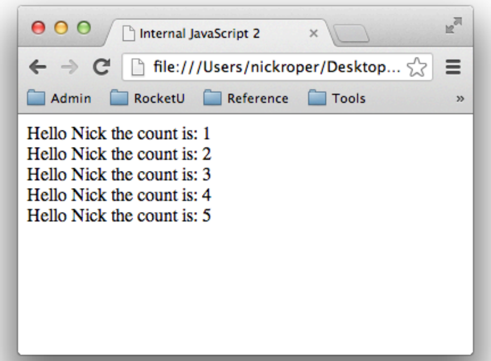
```
var title=‘The Hobbit’,
author=‘Talkies’,
genre=‘Fantasy’;
```

Exercise 3

Create a new HTML document named 'internal-javascript-3.html'

Copy the code from the previous slide to create an internal script in the body.

Load the page and view the results, you should see something like this:



```
Hello Nick the count is: 1
Hello Nick the count is: 2
Hello Nick the count is: 3
Hello Nick the count is: 4
Hello Nick the count is: 5
```

Operators

Operators are the symbols that are used in expressions in order to do things like add or multiply numbers together.

```
var total = 3 + 5 - 2;
```

The + operator actually has two uses in JavaScript – both to add numeric values together and to join, or **concatenate**, text vales to each other.

```
var greeting = 'Hello ' + firstname;
```

The most commonly used operators are listed on the next slide

Operators

Operator	Use	Example	
+	Addition	a + 2	Adds 2 to the value of a
-	Subtraction	a - 3	Subtracts 3 from the value of a
*	Multiplication	a * 3	Multiplies the value of a by 3
/	Division	a / 3	Divides a by three
%	Modulus	a % 3	Divides a by three and gives the remainder
++	Increment	a++	Adds 1 to the value of a
--	Decrement	a--	Subtracts 1 from the value of a

Conditional Expressions

Conditional expressions are used to evaluate whether some thing is true or false and then to carry out some statements based on the result.

A typical example of using a conditional expression is as part of an if structure:

Conditional expression

```
var hourofday = 10;

if (hourofday < 12) {
  document.write('Good morning');
} else {
  document.write('Good afternoon');
}
```

Confidential

Exercise 3

The following statements will assign a value between 0 and 6 that represents the day of the week. A value of 0 is returned for Sunday, 1 for Monday, etc.

```
var d = new date();
var daynum = d.getDay();
```

Edit the code from exercise 4 so that a message is displayed after the loop is executed. The message should either be:

Hooray, today is the weekend!
or
Boo – today is a workday

The message should depend on the current day.

Confidential

Loops – for loop

We've already seen a loop in action in an earlier exercise. Let's look at the code again:

```
var limit = 5;

for (var count=1; count <= limit; count++) {

  document.write('Hi ' + myName + ' the count is: ' + count + '<br>');

}
```

This is an example of a for loop – one of the most common types of loop in programming. In fact it is pretty much identical to the example for loop that we looked at in the previous module.

Can you remember how it works?

Confidential

Loops – while loop

Here’s another type of loop – this time a while loop:

```
var limit = 5, i = 1;

while(i < limit) {
  document.write(‘The value of i is ‘ + i);
  i++;
}
```

A while loop evaluates a conditional expression and uses the result (true or false) to decide whether to execute the statements inside the curly braces. As soon as the expression returns false the loop terminates.

If the expression returns false immediately then the statements will never be executed.

Can you think of any possible issues with a while loop?

Confidential

Module Summary

In this module we started to use JavaScript, the main scripting language for the web.

We looked at the basic syntax of statements and expressions and also variables and data.

We saw how to use JavaScript to add content to a web page and the use of conditional expressions for making decisions and controlling the operation of loops.

Confidential

Labs

Your instructor will upload the labs for this module to Basecamp after the session ends.

Confidential

Questions

Do you have any questions before we move on to the next module

?

Don't worry, if you think of something later then just ask!