HAZZA INSTITUTE OF TECHNOLOGY

iqracity

MERN – ES6 + React

Module 07: JavaScript Basics

Outline

Module 07

- JavaScript Basics
- Understanding concepts of Connecting JavaScript
- Understanding concepts of Operators
- Understanding concepts of Control Flow
- Understanding concepts of While Loops
- Understanding concepts of For Loops

Outline

Frontend Web Development (Interactivity) - JavaScript

- JavaScript basics
- JavaScript ES6
- JavaScript in DOM and BOM
- JavaScript Web APIs

JavaScript basics

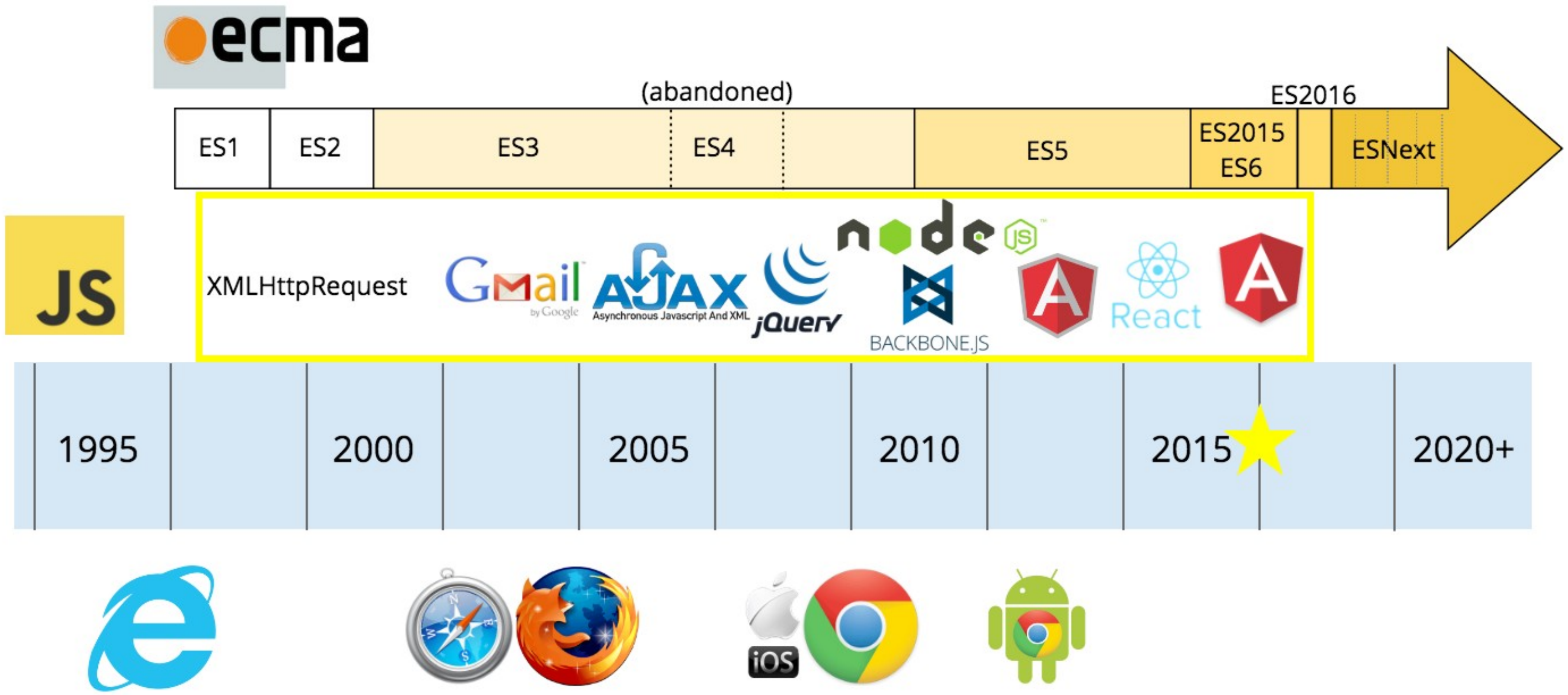
JavaScript - interactivity

What Can JavaScript Do?

- JavaScript Can Change HTML Content
- JavaScript Can Change HTML Attribute Values
- JavaScript Can Change HTML Styles (CSS)
- JavaScript Can Hide/Show HTML Elements (DOM/BOM)
- Send/Receive data

History of JavaScript

JavaScript Basics



JavaScript Where to Put code

JavaScript basics

- The <script> Tag
  - In HTML, JavaScript code is inserted between <script> and </script> tags. Example.
- JavaScript in <head> or <body>
  - 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.
  - Placing scripts at the bottom of the <body> element improves the display speed, because script interpretation slows down the display. Example.
- External JavaScript

External file: myScript.js

```
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
```

Example

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

JavaScript Output

JavaScript basics

- Writing into an HTML element, using **innerHTML**. Example.
  - Changing the innerHTML property of an HTML element is a common way to display data in HTML.
- Writing into the HTML output using **document.write()**. Example. Note.
  - For **testing** purposes, it is convenient to use document.write()
- Writing into an alert box, using **window.alert()**. Example.
  - You can use an alert box to display data:
- Writing into the browser console, using **console.log()**. Example.
  - For **debugging** purposes, you can call the console.log() method in the browser to display data.

JavaScript Programs

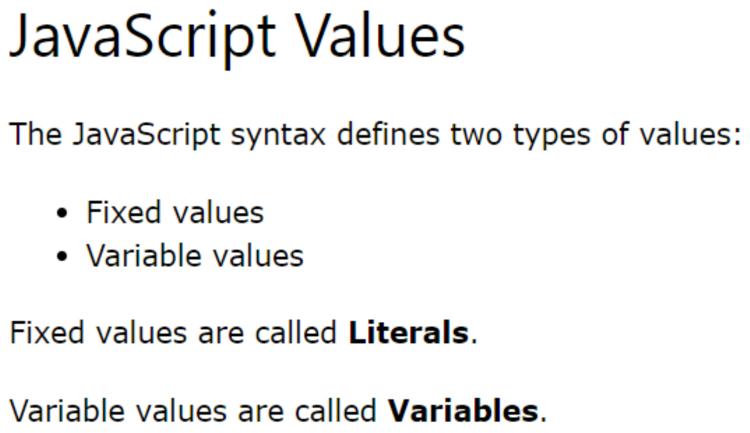
JavaScript basics

- 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.
- Details

JavaScript Syntax

JavaScript basics

- JavaScript syntax is the set of rules, **how** JavaScript programs are **constructed**:



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 Syntax

JavaScript basics

- JavaScript syntax is the set of rules, **how** JavaScript programs are **constructed**:

JavaScript Literals

The two most important syntax rules for fixed values are:

1. **Numbers** are written with or without decimals:

```
10.50

1001
```

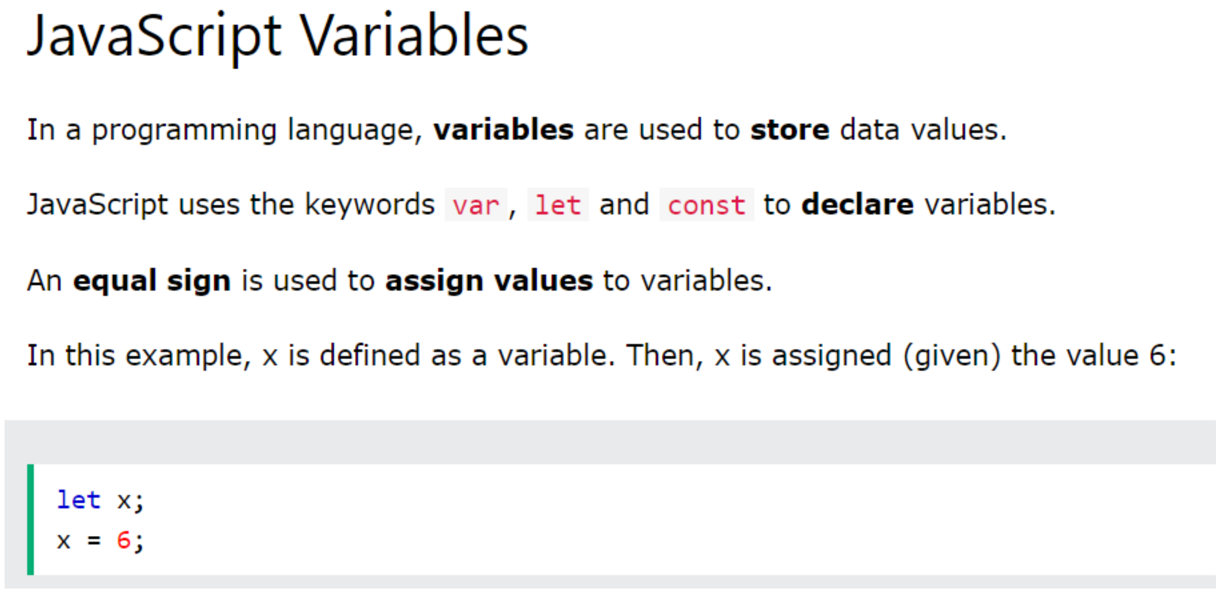
2. **Strings** are text, written within double or single quotes:

```
"John Doe"

'John Doe'
```

JavaScript Syntax

JavaScript basics

- JavaScript syntax is the set of rules, **how** JavaScript programs are **constructed**:



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.

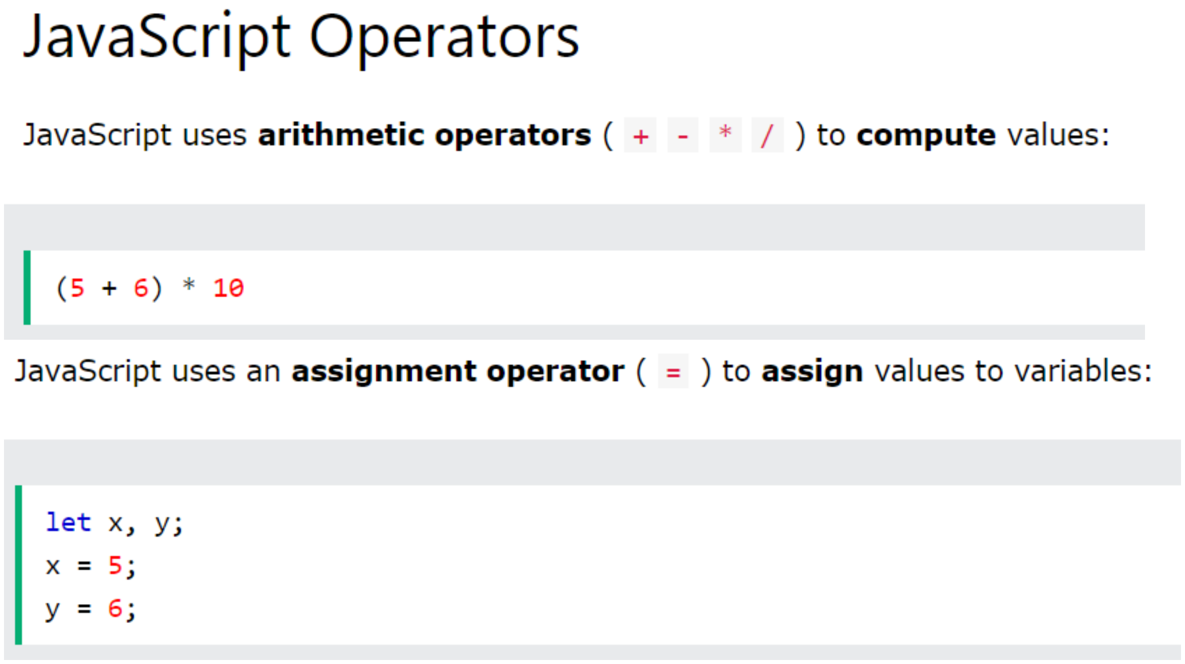
In this example, x is defined as a variable. Then, x is assigned (given) the value 6:

```
let x;
x = 6;
```

JavaScript Syntax

JavaScript basics

- JavaScript syntax is the set of rules, **how** JavaScript programs are **constructed**:



JavaScript Operators

JavaScript uses **arithmetic operators** (`+` `-` `*` `/`) to **compute** values:

```
(5 + 6) * 10
```

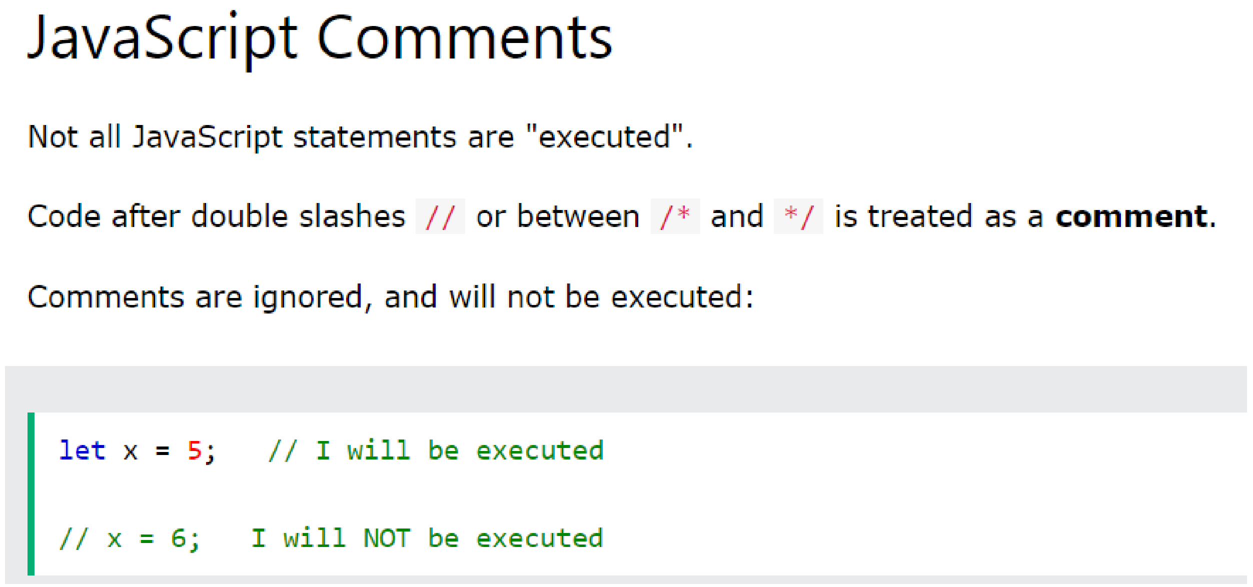
JavaScript uses an **assignment operator** (`=`) to **assign** values to variables:

```
let x, y;
x = 5;
y = 6;
```

JavaScript Syntax

JavaScript basics

- JavaScript syntax is the set of rules, **how** JavaScript programs are **constructed**:



JavaScript Comments

Not all JavaScript statements are "executed".

Code after double slashes `//` or between `/*` and `*/` is treated as a **comment**.

Comments are ignored, and will not be executed:

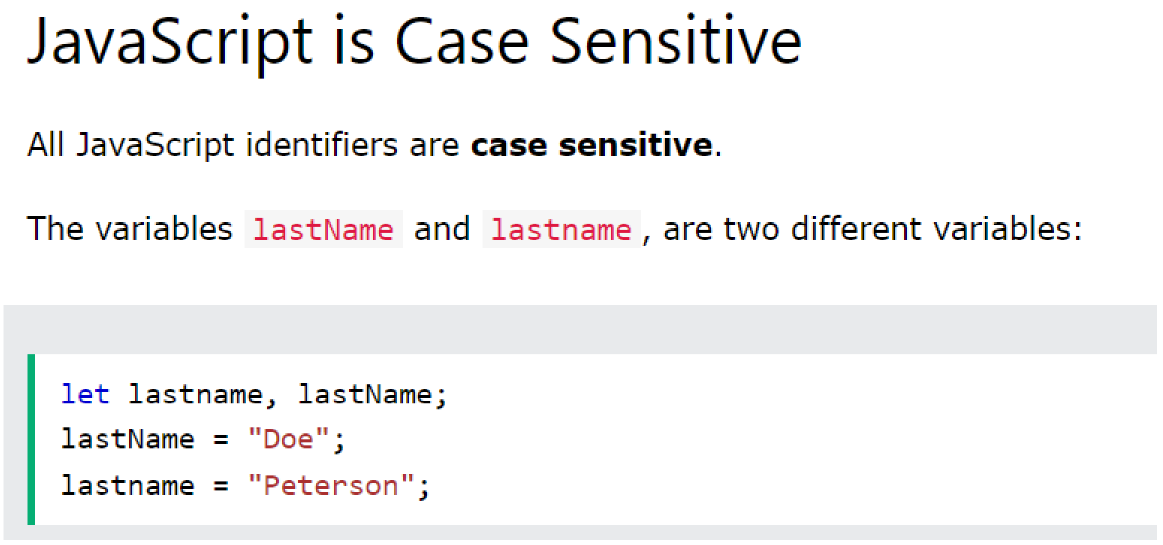
```
let x = 5;   // I will be executed

// x = 6;   I will NOT be executed
```

JavaScript Syntax

JavaScript basics

HAZZA INSTITUTE OF TECHNOLOGY

- JavaScript syntax is the set of rules, **how** JavaScript programs are **constructed**:



JavaScript is Case Sensitive

All JavaScript identifiers are **case sensitive**.

The variables `lastName` and `lastname`, are two different variables:

```
let lastname, lastName;
lastName = "Doe";
lastname = "Peterson";
```

JavaScript Functions

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

HAZZA INSTITUTE OF TECHNOLOGY

JavaScript Function Syntax

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

Function names can contain letters, digits, underscores, and dollar signs (same rules as variables).

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
}
```

JavaScript Functions

HAZZA INSTITUTE OF TECHNOLOGY

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

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":

Example

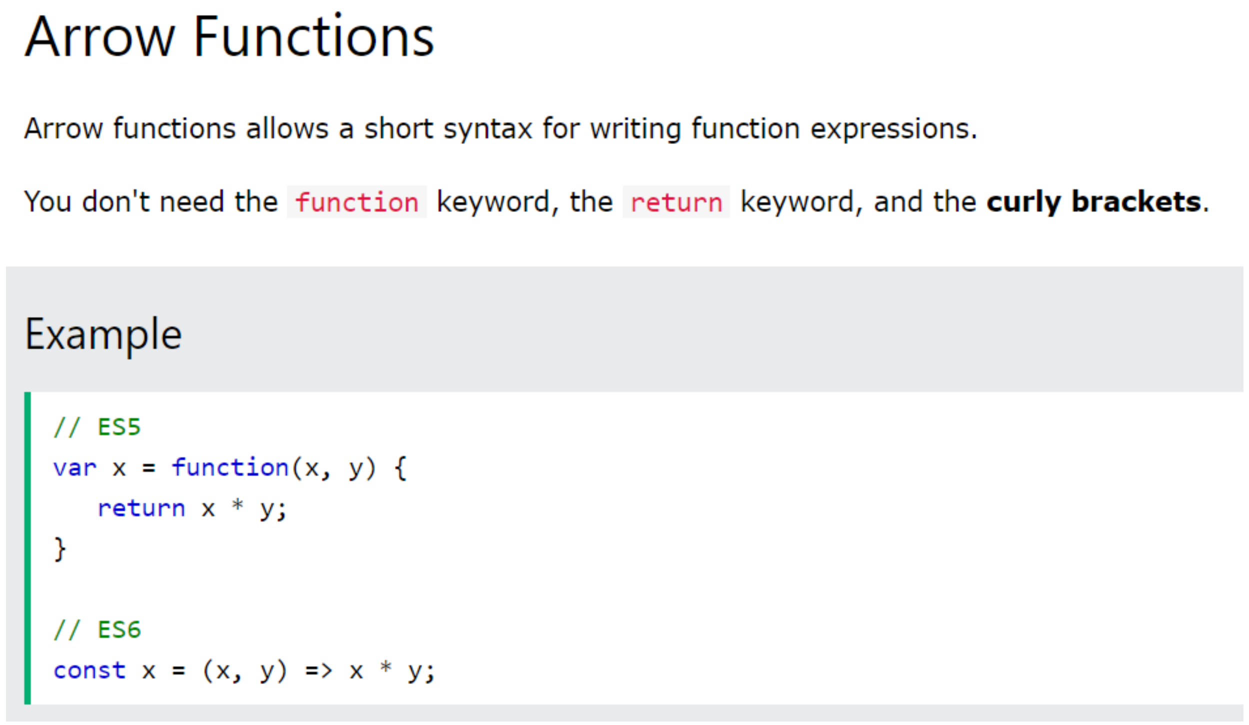
Calculate the product of two numbers, and return the result:

```
let x = myFunction(4, 3);   // Function is called, return value will end up in x

function myFunction(a, b) {
  return a * b;             // Function returns the product of a and b
}
```

JavaScript Functions

HAZZA INSTITUTE OF TECHNOLOGY

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



Arrow Functions

Arrow functions allows a short syntax for writing function expressions.

You don't need the `function` keyword, the `return` keyword, and the **curly brackets**.

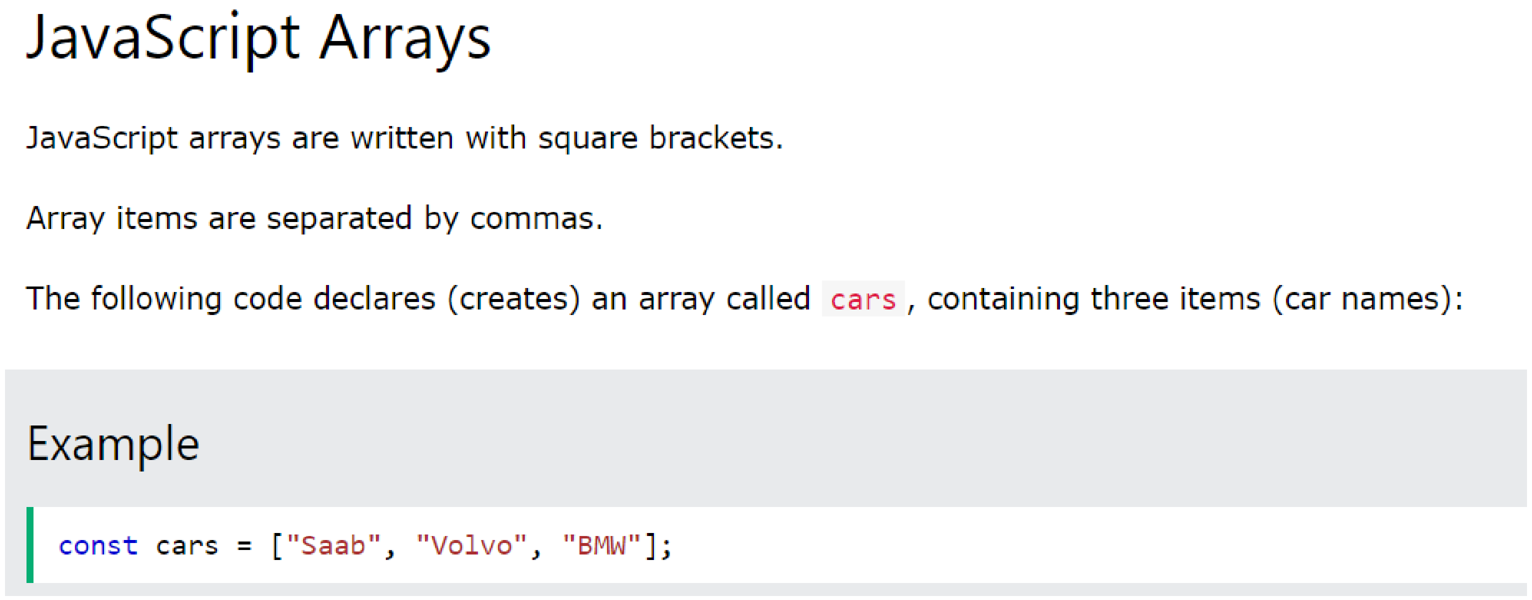
Example

```
// ES5
var x = function(x, y) {
   return x * y;
}

// ES6
const x = (x, y) => x * y;
```

JavaScript Arrow Function (w3schools.com)

JavaScript Arrays

JavaScript basics



JavaScript Arrays

JavaScript arrays are written with square brackets.

Array items are separated by commas.

The following code declares (creates) an array called `cars`, containing three items (car names):

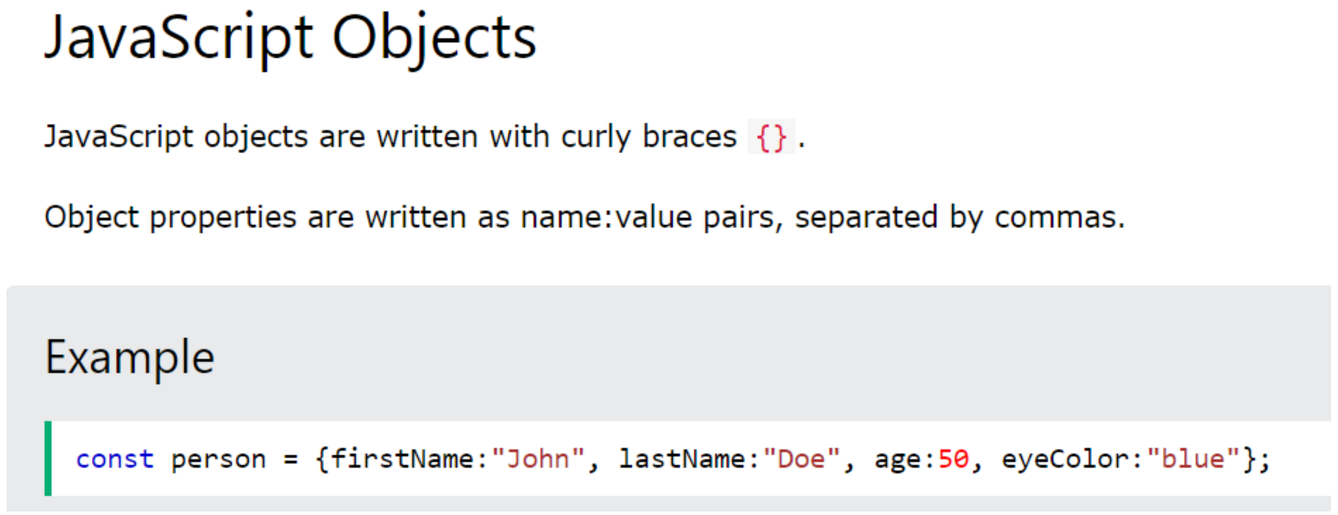
Example

```
const cars = ["Saab", "Volvo", "BMW"];
```

Array indexes are zero-based, which means the first item is [0], second is [1], and so on.

JavaScript Objects

JavaScript basics



JavaScript Objects

JavaScript objects are written with curly braces `{}`.

Object properties are written as name:value pairs, separated by commas.

Example

```
const person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
```

The object (person) in the example above has 4 properties: firstName, lastName, age, and eyeColor.

JavaScript Loops

JavaScript basics

HAZZA INSTITUTE OF TECHNOLOGY

Different Kinds of Loops

JavaScript supports different kinds of loops:

- `for` - loops through a block of code a number of times
- `for/in` - loops through the properties of an object
- `for/of` - loops through the values of an iterable object
- `while` - loops through a block of code while a specified condition is true
- `do/while` - also loops through a block of code while a specified condition is true

JavaScript Loops

HAZZA INSTITUTE OF TECHNOLOGY

JavaScript basics

The For Loop

The `for` loop has the following syntax:

```
for (statement 1; statement 2; statement 3) {
  // code block to be executed
}
```

Statement 1 is executed (one time) before the execution of the code block.

Statement 2 defines the condition for executing the code block.

Statement 3 is executed (every time) after the code block has been executed.

Example

```
for (let i = 0; i < 5; i++) {
  text += "The number is " + i + "<br>";
}
```

JavaScript Loops

JavaScript basics

The For In Loop

The JavaScript `for in` statement loops through the properties of an Object:

Syntax

```
for (key in object) {
  // code block to be executed
}
```

Example

```
const person = {fname:"John", lname:"Doe", age:25};

let text = "";
for (let x in person) {
  text += person[x];
}
```

JavaScript Loops

JavaScript basics

The For Of Loop

The JavaScript `for of` statement loops through the values of an iterable object.

It lets you loop over iterable data structures such as Arrays, Strings, Maps, NodeLists, and more:

Syntax

```
for (variable of iterable) {
  // code block to be executed
}
```

Example

```
let language = "JavaScript";

let text = "";
for (let x of language) {
text += x;
}
```

JavaScript Loops

JavaScript basics

The While Loop

The `while` loop loops through a block of code as long as a specified condition is true.

Syntax

```
while (condition) {
  // code block to be executed
}
```

Example

```
while (i < 10) {
  text += "The number is " + i;
  i++;
}
```

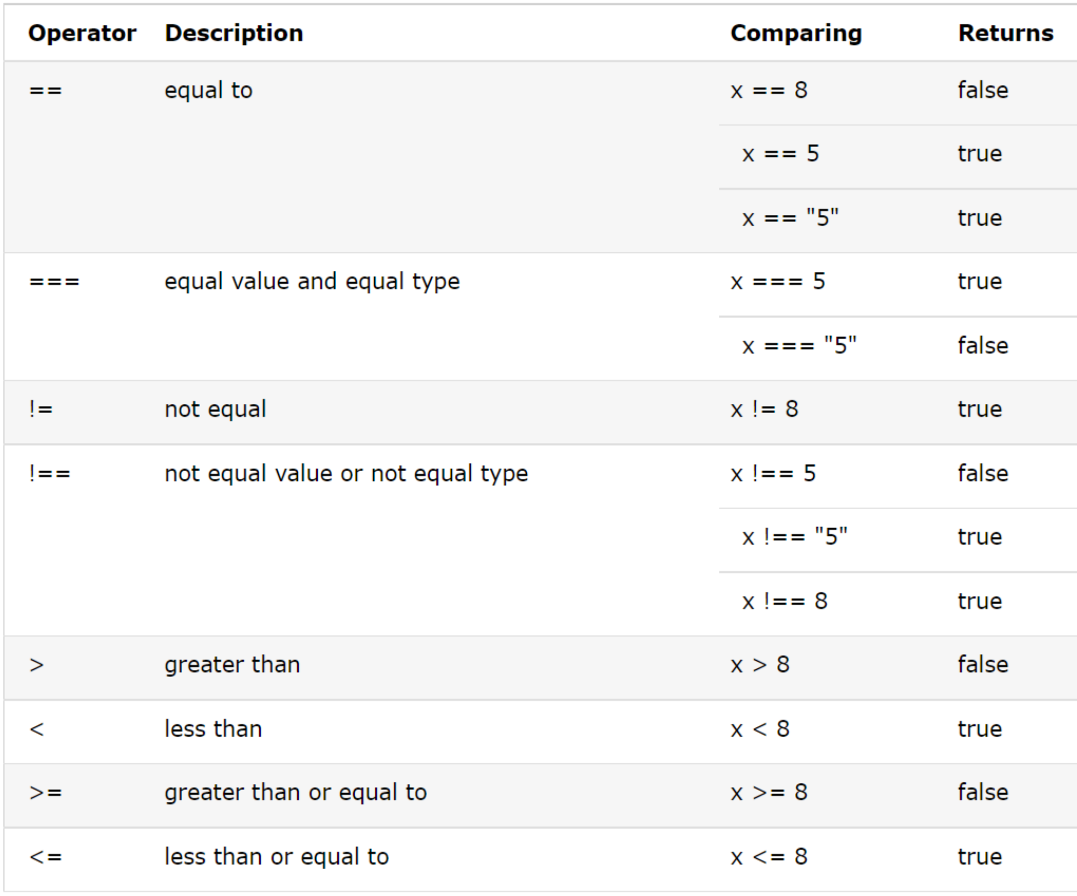
JavaScript Where to Put code

JavaScript basics

JavaScript Comparison and Logical Operators

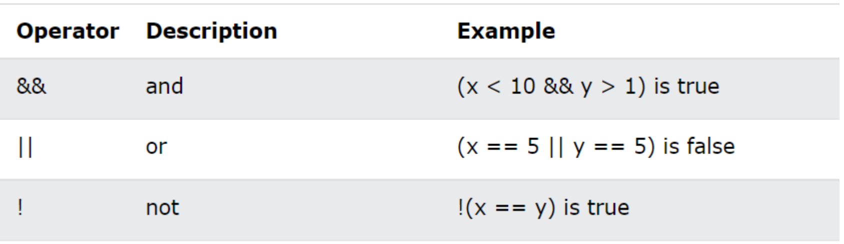
JavaScript basics

- Given that x = 5



Operator	Description	Comparing	Returns
==	equal to	x == 8	false
		x == 5	true
		x == "5"	true
===	equal value and equal type	x === 5	true
		x === "5"	false
!=	not equal	x != 8	true
!==	not equal value or not equal type	x !== 5	false
		x !== "5"	true
		x !== 8	true
>	greater than	x > 8	false
<	less than	x < 8	true
>=	greater than or equal to	x >= 8	false
<=	less than or equal to	x <= 8	true

- Given that x = 6 and y = 3



Operator	Description	Example
&&	and	(x < 10 && y > 1) is true
||	or	(x == 5 || y == 5) is false
!	not	!(x == y) is true

JavaScript Flow Control

JavaScript basics

HAZZA INSTITUTE OF TECHNOLOGY

In JavaScript we have the following conditional statements:

- Use `if` to specify a block of code to be executed, if a specified condition is true
- Use `else` to specify a block of code to be executed, if the same condition is false
- Use `else if` to specify a new condition to test, if the first condition is false
- Use `switch` to specify many alternative blocks of code to be executed

The if Statement

Use the `if` statement to specify a block of JavaScript code to be executed if a condition is true.

Syntax

```
if (condition) {
  //  block of code to be executed if the condition is true
}
```

JavaScript Flow Control

JavaScript basics

HAZZA INSTITUTE OF TECHNOLOGY

The else Statement

Use the `else` statement to specify a block of code to be executed if the condition is false.

```
if (condition) {
  //  block of code to be executed if the condition is true
} else {
  //  block of code to be executed if the condition is false
}
```

Example

```
if (hour < 18) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}
```

JavaScript Flow Control

JavaScript basics

The else if Statement

Use the `else if` statement to specify a new condition if the first condition is false.

Syntax

```
if (time < 10) {
  greeting = "Good morning";
} else if (time < 20) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}
```

JavaScript Flow Control

JavaScript basics

HAZZA INSTITUTE OF TECHNOLOGY

The JavaScript Switch Statement

Use the `switch` statement to select one of many code blocks to be executed.

Syntax

```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

JavaScript Flow Control

JavaScript basics

The JavaScript Switch Statement

Use the `switch` statement to select one of many code blocks to be executed.

Syntax

```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

JavaScript Flow Control

JavaScript basics

Conditional (Ternary) Operator

JavaScript also contains a conditional operator that assigns a value to a variable based on some condition.

Syntax

```
variablename = (condition) ? value1:value2
```

Example

```javascript
let voteable = (age < 18) ? "Too young":"Old enough";
```

Summary

Module 07

- JavaScript Basics
- Understanding concepts of Connecting JavaScript
- Understanding concepts of Operators
- Understanding concepts of Control Flow
- Understanding concepts of While Loops
- Understanding concepts of For Loops