# Fundamentals of Computer Engineering

## Module VI - Unit II
## JavaScript

Teachers: Moisés Martínez (1ºA  English)

Year: 2022 - 2023

Universidad Francisco de Vitoria
UFV Madrid

*Grado en Ingeniería Informática*
*Escuela Politécnica Superior*

Universidad
Francisco de Vitoria
UFV Madrid

# JavaScript

*Grado en Ingeniería Informática*
*Escuela Politécnica Superior*

## JavaScript

JavaScript is a cross-platform, object-oriented scripting language used to make webpages interactive (e.g., having complex animations, clickable buttons, popup menus, etc.).

JavaScript contains a standard library of objects, such as Array, Date, and Math, and a core set of language elements such as operators, control structures, and statements.

- Client-side JavaScript extends the core language by supplying objects to control a browser and its Document Object Model (DOM).
- Server-side JavaScript extends the core language by supplying objects relevant to running JavaScript on a server.

## JavaScript

- JavaScript is a lightweight scripting language made for client-side execution on the browser. Since it is not designed as a general-purpose language and is specially engineered for web applications, the set of libraries is also geared primarily towards web applications.
- JavaScript is an interpreted language instead of a compiled one. In that sense, it is closer to languages like Ruby and Python.
- The engine browser interprets JS's source code, line by line and runs it. In contrast, a compiled language needs to be compiled into a byte-code code executable.
- JavaScript is heavily used for easing the interactions between users and web applications.
- JavaScript is platform-independent; it can run on any computer irrespective of the operating systems used.
- This language is mainly used by web programmers to give quick responses to user actions without the need to send information about what the user has done to the server and wait for a response from it.

## JavaScript

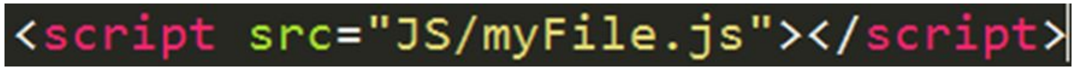
- JavaScript is still working even if the internet connection is interrupted.
- Allows asynchronous communications with the backend using Ajax.  Ajax is the acronym for Asynchronous Javascript And XML. It is an architectural design that elegantly resolves the updating of HTML content based on having an open communication channel with the server once the web page has been loaded.
- There are numerous frameworks such as JQuery (JavaScript libraries), Angular, Node and even JavaScript compilers such as CoffeeScript.

## JavaScript

JavaScript can be used in a browser in two different ways:

- The Web Console tool built into mostly browser is useful tool for experimenting with JavaScript; you can use it in two modes: single-line input mode, and multi-line input mode.
- The Web page by embedding Javascript through a script or code snippet.

  - JavaScript can appear embedded in a <script> element within an HTML document
  - JavaScript can appear loaded by a file.



```
<script src="JS/myFile.js"></script>
```

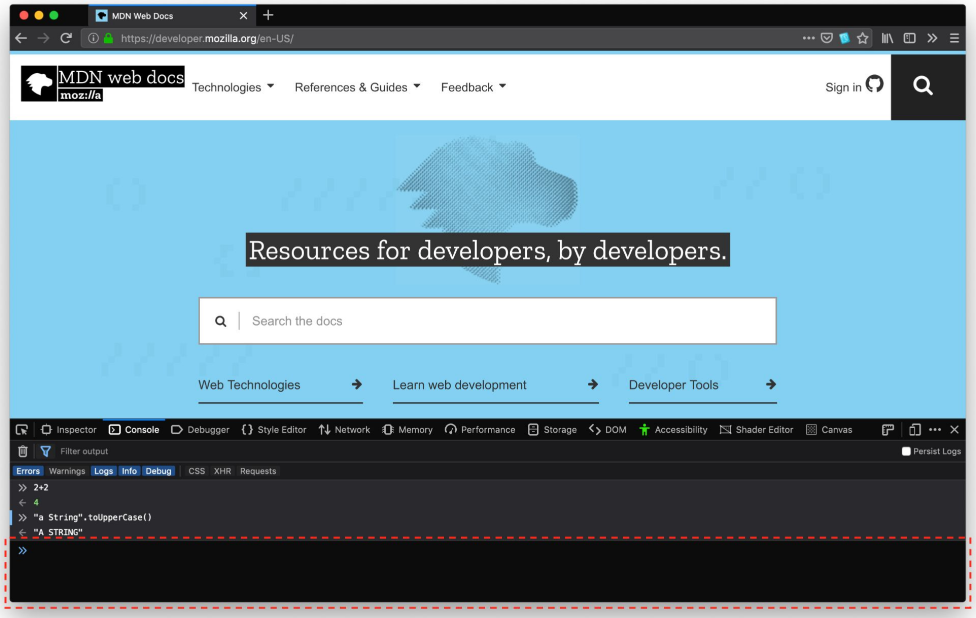
Universidad Francisco de Vitoria UFV Madrid

Universidad
Francisco de Vitoria
UFV Madrid

## JavaScript console

# JavaScript Console

To open the Web Console press (Ctrl+Shift+I) on Windows and Linux or (Cmd -Option-K on Mac). If you are using Chrome can you open it in the config menu and select “More Tools” and “Developers tools".



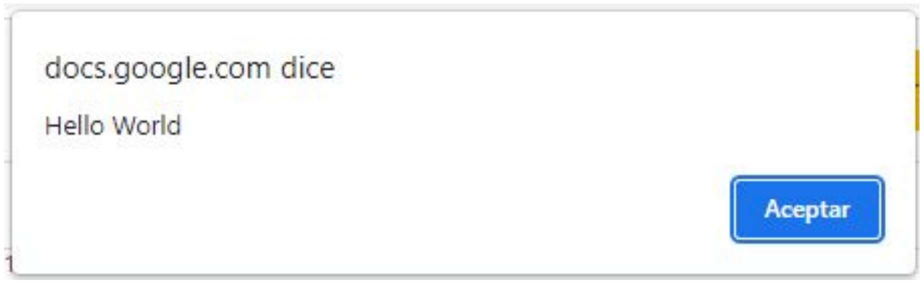
```
>> 2+2
<- 4
>> "a String".toUpperCase()
<- "A STRING"
>>
```

## JavaScript Console

To get started with writing JavaScript, open can open the Web Console in multi-line mode, and write our first "Hello world" JavaScript code:

```
(function(){
  "use strict";
  function printHello(yourName) {
    alert(`Hello ${yourName}`);
  }

  printHello('World');
})();
```



docs.google.com dice

Hello World

JS's strict mode is a way to opt in to a restricted variant of JavaScript, thereby implicitly opting-out of "sloppy mode".

Universidad
Francisco de Vitoria
UFV Madrid

## JavaScript language

## JavaScript Language

JavaScript borrows most of its syntax from Java, C, and C++, but it has also been influenced by Awk, Perl, and Python.

JavaScript is case-sensitive and uses the Unicode character set. For example, the word name could be used as a variable name.

```
const name = "anything";
```

But, the variable name is not the same as Name because JavaScript is case sensitive.

## JavaScript Language

### Comments

The syntax of comments is the same as in C or C++ and in many other languages:

```
// a one line comment

/* this is a longer,
 * multi-line
 * comment
 */
```

Universidad Francisco de Vitoria UFV Madrid

## Variables

JavaScript has three kinds of variable declarations.

- var which declares a variable, optionally initializing it to a value.
- let which declares a block-scoped, local variable, optionally initializing it to a value.
- const declares a block-scoped, read-only named constant.

The names of variables, called **identifiers**, conform to certain rules.

- A JS identifier usually starts with a letter, underscore (_), or dollar sign ($). Subsequent characters can also be digits (0 – 9).
- Some examples of legal names are Number_hits, temp99, $credit, and _name.

## JavaScript Language

### Variables

In a statement like var x = 42, the var x part is called a declaration, and the = 42 part is called an initializer. The declaration allows the variable to be accessed later in code without throwing a ReferenceError, while the initializer assigns a value to the variable.

```
var x;
console.log(x); // This will show logs "undefined"
```

The initializer is optional if we use var or let declarations. If a variable is declared without an initializer, it is assigned the value undefined.

## JavaScript Language

### Variables

**The scope of a variable is its lifetime in the program**. This means that the scope of a variable is the block of code in the entire program where the variable is declared, used, and can be modified. A variable may belong to one of the following scopes in JS:

- Global scope: The default scope for all code running in script mode.
- Module scope: The scope for code running in module mode.
- Function scope: The scope created with a function.
- Block scope: The scope created with a pair of curly braces (a block).

```
if (Math.random() > 0.8) {
  const a = 5;
}
console.log(a);  // It will show ReferenceError: a is not defined
```

## Constants

Constants are read-only identifies. It syntax is the same as any variable identifier: it must start with a letter, underscore, or dollar sign ($), and can contain alphabetic, numeric, or underscore characters.

```
const PI = 3.14;
```

- Constants cannot change value through assignment or be re-declared while the script is running.
- You cannot declare a constant with the same name as a function or variable in the same scope.

## Data Types

The standard JS defines eight data types:

- Boolean: true and false.
- Number: An integer or floating point number. For example: 95 or 3.14159.
- BigInt: An integer with arbitrary precision. For example: 9374838325474O992n.
- String: A sequence of characters that represent a text value. For example: "Howdy".
- Symbol: A data type whose instances are unique and immutable.
- null: A special keyword denoting a null value. (Because JavaScript is case-sensitive, null is not the same as Null, NULL, or any other variant.)
- undefined: A top-level property whose value is not defined.
- Object.

JavaScript Language

## Data Types

JavaScript is a **dynamically typed language**. This means you don't have to specify the data type of a variable when you declare it. It also means that data types are automatically converted as-needed during script execution.

```
var value = 98;

value = 'I am not a number anymore';
```

JavaScript is dynamically typed, this assignment does not cause an error message.

## Conditional statements

A conditional statement is a set of commands that executes if a specified condition is true. JavaScript supports two conditional statements: if...else and switch.

A if statement allows a program to execute a statement if a logical condition is true.

```
if (condition) {
  statement1;
} else {
  statement2;
}
```

**There are two optional statements (else and else if) clause to execute a statement if the condition is false.**

JavaScript Language

## Conditional statements

A conditional statement is a set of commands that executes if a specified condition is true. JavaScript supports two conditional statements: if...else and switch.

A switch statement allows a program to evaluate an expression and attempt to match the expression's value to a case label. If a match is found, the program executes the associated statement.

```
switch (expression) {
  case label1:
    statements1;
    break;
  // …
  default:
    statementsDefault;
}
```

## JavaScript Language

### For statements

A for loop repeats until a specified condition evaluates to false. The JavaScript for loop is similar to the C for loop.

```
for ([initialExpression]; [conditionExpression]; [incrementExpression])
  statement
```

JavaScript Language

## For statements

If we want to count the number of selected options in a scrolling list:

### HTML

```
<form name="selectForm">
        <label for="musicTypes">Choose  your music style</label>
        <select id="musicTypes" name="musicTypes" multiple>
                <option selected>R&B</option>
                <option>Pop</option>
                <option>Rock</option>
                <option>Electronic</option>
                <option>Reggaeton</option>
                <option>Dance</option>
        </select>
        <button id="btn" type="button">How many are selected?</button>
</form>
```

### JS

```
function howManyStyles(selectObject) {
  let numberSelected = 0;
  for (let i = 0; i < selectObject.options.length; i++) {
    if (selectObject.options[i].selected) {
      numberSelected++;
    }
  }
  return numberSelected;
}

const btn = document.getElementById('btn');

btn.addEventListener('click', () => {
  const musicTypes = document.selectForm.musicTypes;
  console.log(`You have selected ${howManyStyles(musicTypes)} option(s).`);
});
```

Universidad Francisco de Vitoria UFV Madrid

## While statements

The do...while statement repeats until a specified condition evaluates to false.

```
do
  statement
while (condition);
```

- The do…while statement is always executed once before the condition is checked.
- If condition is true, the statement executes again. At the end of every execution, the condition is checked. When the condition is false, execution stops, and control passes to the statement following do...while.

## While statements

A while statement executes its statements as long as a specified condition evaluates to true.

```
while (condition)
  statement
```

- If the condition becomes false, statement within the loop stops executing and control passes to the statement following the loop.
- The condition test occurs before statement in the loop is executed. If the condition returns true, statement is executed and the condition is tested again. If the condition returns false, execution stops, and control is passed to the statement following while.

## Functions

Functions are one of the fundamental building blocks in JavaScript. A function in JavaScript is similar to a procedure set of statements that performs a task or calculates a value.

```
function squareOfNumber(number) {
  return number * number;
}
```

- The name of the function.
- A list of parameters to the function, enclosed in parentheses and separated by commas.
- The JavaScript statements that define the function, enclosed in curly brackets { }.
- The return statement (optional) that return a value when function has finished.

Universidad
Francisco de Vitoria
UFV Madrid

## Writing JavaScript

*Grado en Ingeniería Informática*
*Escuela Politécnica Superior*

## Writing JavaScript

### Writing JS in my code

```
<!DOCTYPE html>
<html lang='en'>
<html>
	<head>
		<title>test FCE</title>
	</head>
	<body>
		<script>
			document.write('Hola Mundo');
		</script>
	</body>
</html>
```

## Writing JavaScript

### Writing JS in my code

```
<!DOCTYPE html>
<html lang='en'>
<html>
	<head>
		<title>test FCE</title>
	</head>
	<body>
		<script>
			// My first comment in JS
			document.write("<h3>Heading level 3 in JS</h3>");
		</script>
	</body>
</html>
```

Universidad Francisco de Vitoria UFV Madrid

## Writing JavaScript

### Writing JS in my code

#### HTML

```
<!DOCTYPE html>
<html lang='en'>
<head>
        <meta charset="utf-8">
        <title> JavaScript uning a js file</title>
        <script src="file.js"></script>
</head>
<body>
        <h1> Web with alert message </h1>
</body>
</html>
```

#### JS

```
document.write("<h3>Heading level 3 in JS</h3>");
```

Universidad
Francisco de Vitoria
UFV Madrid

## Can use JS to manipulate DOM?

## Do you remember the DOM?

**How can I make JavaScript modify the DOM?**

We can find element by using id, name or CSS class

```
document.getElementById(id)
```

**How can I change the content?**

```
element.setAttribute(attribute, value)
```

**Can I add or delete elements dynamically?**

```
document.createElement(element)
```

Javascript allows to manipulate the page in another way based on the interaction with the user (see more)

Universidad
Francisco de Vitoria
UFV Madrid

## How to send a email using JS?

## How to send a email using JS?

You can use 3rd Party API to send an email (Not secure). You can use tools like https://dev.mailjet.com/

- Register for Mailjet to get an API key and Secret.
- Create a function to call the API to send an email.

This mode is not secure, because you must put your API Key and Secret in the client code. But It can be a nice option to learn.

## How to send a email using JS?

```
function sendMail(name, email, subject, message) {
  const myHeaders = new Headers();
  myHeaders.append("Content-Type", "application/json");
  myHeaders.set('Authorization', 'Basic ' + base64.encode('<API Key>'+":" +'<Secret Key>'));

  const data = JSON.stringify({
    "Messages": [{
      "From": {"Email": "<YOUR EMAIL>", "Name": "<YOUR NAME>"},
      "To": [{"Email": email, "Name": name}],
      "Subject": subject,
      "TextPart": message
    }]
  });

  const requestOptions = {
    method: 'POST',
    headers: myHeaders,
    body: data,
  };

  fetch("https://api.mailjet.com/v3.1/send", requestOptions)
    .then(response => response.text())
    .then(result => console.log(result))
    .catch(error => console.log('error', error));
}
```

Universidad
Francisco de Vitoria
UFV Madrid

## BOM (Browser Object Model)

## BOM

Versions 3.0 of the Internet Explorer and Netscape Navigator browsers introduced the concept of the Browser Object Model, or BOM, which allows you to access and modify the properties of the browser's own windows.

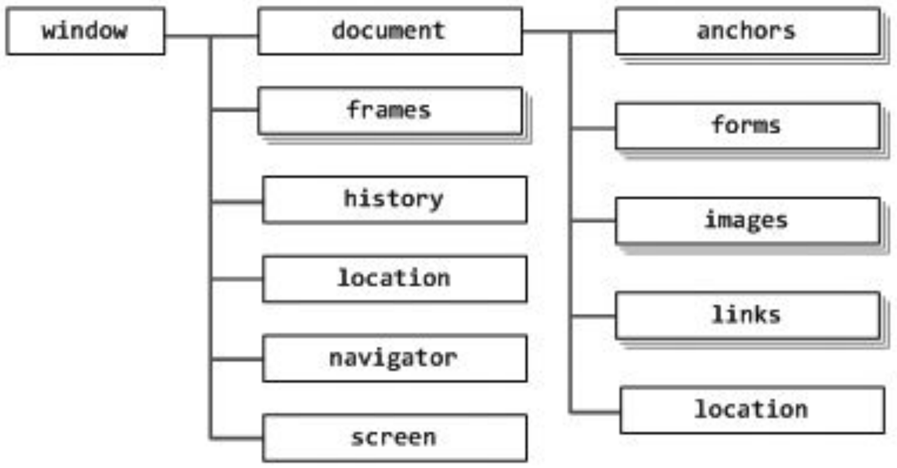
- BOM makes possible to resize and move the browser window, modify the text displayed in the status bar, and perform many other manipulations unrelated to the content of the HTML page.
- The biggest drawback of BOM is that, contrary to what happens with DOM, no entity is in charge of standardizing it or defining minimum interoperability between browsers.

**The BOM provides you with objects that expose the web browser's functionality.**

## BOM

BOM is composed of several interrelated objects. BOM allow us:

- Create, move, resize, and close browser windows.
- Obtain information about the browser itself.
- Properties of the current page and user screen.
- Cookie management.
- ActiveX objects in Internet Explorer.



Universidad Francisco de Vitoria UFV Madrid

## References

### References

- https://developer.mozilla.org/es/docs/Web/JavaScript/Guide
- https://www.tutorialspoint.com/javascript/
- http://www.w3schools.com/js/default.asp