UNIVERSIDAD TECNOLÓGICA DE AGUASCALIENTES

Aplicaciones web

2do PARCIAL

Lenguajes de programación del lado del servidor (JavaScript)

JOSUE ABDI GONZALEZ GARCIA

CARLOS FERNANDO OVALLE GARCÍA

AGUASCALIENTES, AGS. 30 OCTUBRE, 2025

Índice de Contenidos – JavaScript

Fundamentos de JavaScript

JavaScript Intermedio

JavaScript Avanzado

Introducción a JavaScript:

JavaScript es el lenguaje de programación de la web.

Puede calcular, manipular y validar datos.

Puede actualizar y cambiar tanto HTML como CSS.

¿Por qué estudiar JavaScript?

JavaScript es uno de los **3 idiomas** todos los desarrolladores web debe aprende:

1. **HTML** definir el contenido de las páginas web
2. **CSS** especificar el diseño de las páginas web
3. **JavaScript** programar el comportamiento de las páginas web

JavaScript puede cambiar el contenido HTML

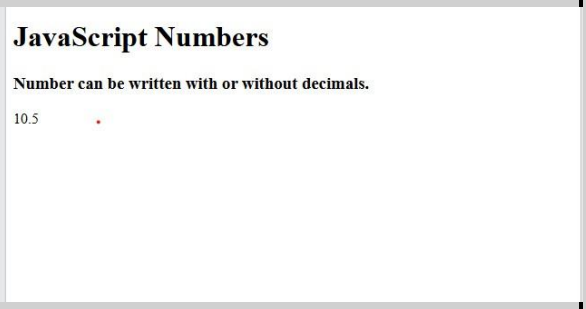
Uno de los muchos métodos HTML de JavaScript es getElementById().

El siguiente ejemplo "encuentra" un elemento HTML (con id="demo"), y cambia el contenido del elemento (innerHTML) a "Hola JavaScript"

Fundamentos de JavaScript

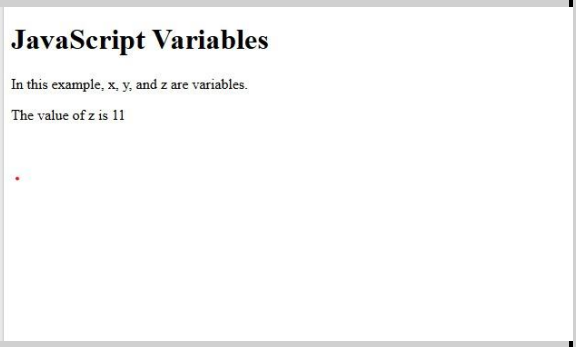
1. **Sintaxis de JavaScript:**
   Conjunto de reglas que determinan cómo se debe escribir correctamente el código JavaScript para que el navegador lo interprete.

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Numbers</h1>
<h3>Number can be written with or without decimals.</h3>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = 10.50;
</script>

</body>
</html>
```



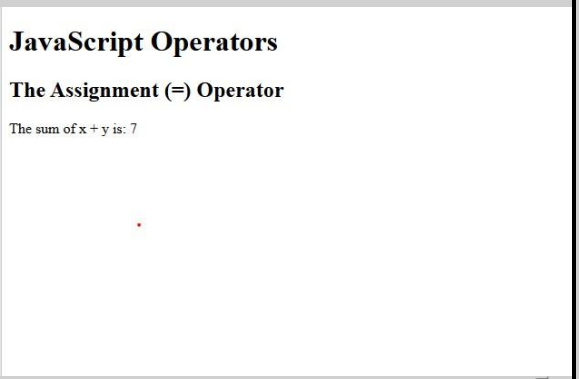
2. **Variables en JavaScript:**
   Espacios de memoria donde se almacenan datos que pueden cambiar durante la ejecución del programa. Se declaran con var, let o const.

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Variables</h1>

<p>In this example, x, y, and z are variables.</p>

<p id="demo"></p>

<script>
let x = 5;
let y = 6;
let z = x + y;

document.getElementById("demo").innerHTML = "The value of z is " + z;
</script>

</body>
</html>
```



3. **Operadores en JavaScript:**
   Símbolos que permiten realizar operaciones sobre valores y variables (por ejemplo: +, -, *, /, ==, ===, &&, ||).

```
<!DOCTYPE html>
<html>
<body>

<h1>JavaScript Operators</h1>
<h2>The Assignment (=) Operator</h2>

<p id="demo"></p>

<script>
// Assign the value 5 to x
let x = 5;
// Assign the value 2 to y
let y = 2;
// Assign the value x + y to z
let z = x + y;
// Display z
document.getElementById("demo").innerHTML = "The sum of x + y is: " + z;
</script>

</body>
</html>
```



4. **Condicionales If en JavaScript:**
   Estructura que permite ejecutar diferentes bloques de código dependiendo de si una condición es verdadera o falsa.

```
if (condition1) {
  // code to execute if condition1 is true
} else if (condition2) {
  // code to execute if the condition1 is false and condition2 is true
} else {
  // code to execute if the condition1 is false and condition2 is false
}
```

5. **Bucles en JavaScript:**
   Estructuras que repiten un bloque de código mientras se cumpla una condición (for, while, do...while).

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Loops</h1>
<h2>The for Loop</h2>

<p id="demo"></p>

<script>
const cars = ["BMW", "Volvo", "Saab", "Ford", "Fiat", "Audi"];

let text = "";
for (let i = 0; i < cars.length; i++) {
  text += cars[i] + "<br>";
}

document.getElementById("demo").innerHTML = text;
</script>

</body>
</html>
```

JavaScript Loops

The for Loop

BMW
Volvo
Saab
Ford
Fiat
Audi

6. **Cadenas de Texto (Strings) en JavaScript:**
   Conjunto de caracteres encerrados entre comillas simples, dobles o invertidas.
   Sirven para manejar texto en el programa.

```
<!DOCTYPE html>
<html>
<body>

<h1>JavaScript Strings</h1>
<p>Strings are written inside quotes. You can use single or double quotes:
</p>

<p id="demo"></p>

<script>
let carName1 = "Volvo XC60"; // Double quotes
let carName2 = 'Volvo XC60'; // Single quotes

document.getElementById("demo").innerHTML =
carName1 + " " + carName2;
</script>

</body>
</html>
```

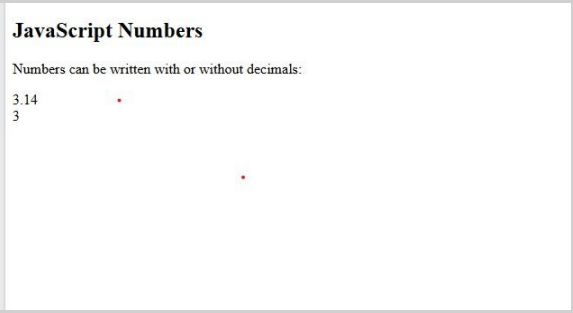
JavaScript Strings

Strings are written inside quotes. You can use single or double quotes:

Volvo XC60 Volvo XC60

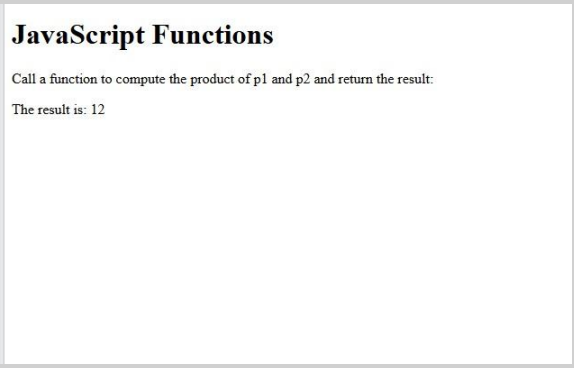
7. **Números en JavaScript:**
   Tipo de dato usado para representar valores numéricos, ya sean enteros o decimales.

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Numbers</h2>

<p>Numbers can be written with or without decimals:</p>

<p id="demo"></p>

<script>
let x = 3.14;
let y = 3;
document.getElementById("demo").innerHTML = x + "<br>" + y;
</script>

</body>
</html>
```



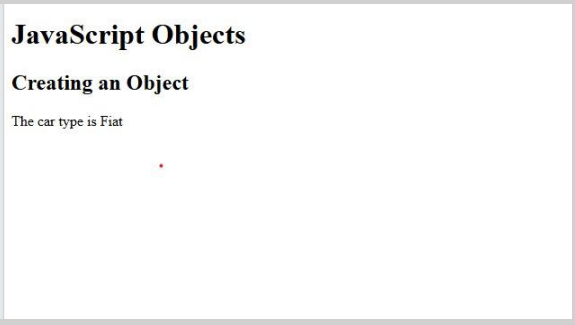
8. **Funciones en JavaScript:**
   Bloques de código reutilizables que realizan una tarea específica y pueden recibir y devolver valores.

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Functions</h1>

<p>Call a function to compute the product of p1 and p2 and return the
result:</p>

<p id="demo"></p>

<script>
// Function to compute the product of p1 and p2
function myFunction(p1, p2) {
  return p1 * p2;
}

let result = myFunction(4, 3);
document.getElementById("demo").innerHTML = "The result is: " + result;
</script>

</body>
</html>
```



9. **Objetos en JavaScript:**
   Colecciones de propiedades y métodos que representan entidades con características y comportamientos.

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Objects</h1>
<h2>Creating an Object</h2>

<p id="demo"></p>

<script>
// Create an Object:
const car = {type:"Fiat", model:"500", color:"white"};

// Display Data from the Object:
document.getElementById("demo").innerHTML =
"The car type is " + car.type;
</script>

</body>
</html>
```

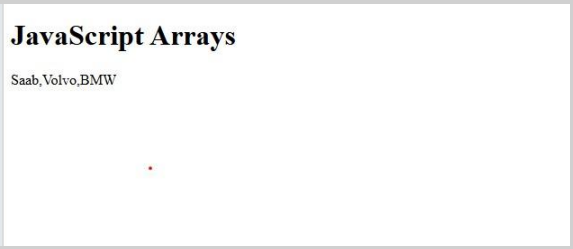


10. **Fechas en JavaScript:**
    Se gestionan con el objeto Date, que permite crear, modificar y formatear fechas y horas.

```html
<!DOCTYPE html>
<html>
<body>

<h1>JavaScript Dates</h1>
<h2>Using new Date()</h2>
<p>new Date() without arguments, creates a date object with the current
date and time:</p>

<p id="demo"></p>

<script>
const d = new Date();
document.getElementById("demo").innerHTML = d;
</script>

</body>
</html>
```

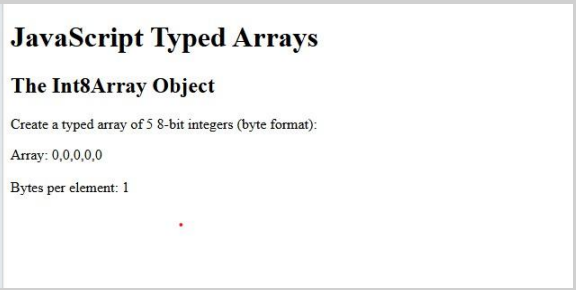
11. **Arrays en JavaScript:**
    Estructuras que almacenan múltiples valores en una sola variable, accesibles mediante índices.

```html
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Arrays</h1>

<p id="demo"></p>

<script>
const cars = ["Saab", "Volvo", "BMW"];
document.getElementById("demo").innerHTML = cars;
</script>

</body>
</html>
```



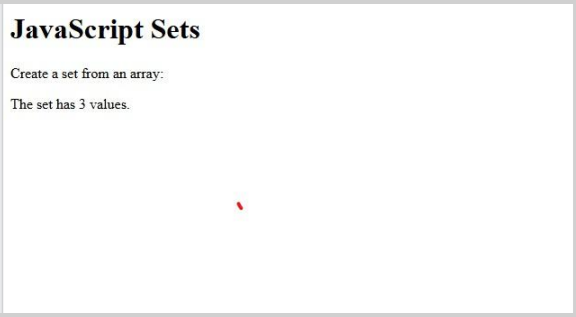
12. **Arrays Tipados (Typed Arrays):**
    Arreglos especiales que manejan datos binarios en memoria, útiles para operaciones de alto rendimiento.

```html
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Typed Arrays</h1>
<h2>The Int8Array Object</h2>
<p>Create a typed array of 5 8-bit integers (byte format):</p>

<p id="demo"></p>

<script>
const myArr = new Uint8Array(5);
document.getElementById("demo").innerHTML = "Array: " + myArr + "<br>
<br>Bytes per element: " + myArr.BYTES_PER_ELEMENT;
</script>

</body>
</html>
```



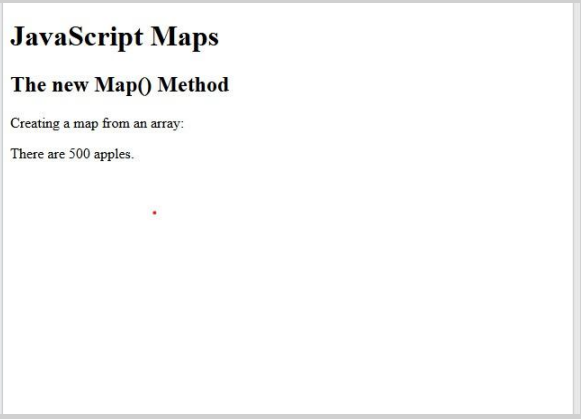
13. **Conjuntos (Sets) en JavaScript:**
    Colecciones de valores únicos, sin elementos duplicados, que permiten operaciones eficientes de búsqueda.

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Sets</h1>
<p>Create a set from an array:</p>

<p id="demo"></p>

<script>
// Create a Set
const letters = new Set(["a","b","c"]);

// Display set.size
document.getElementById("demo").innerHTML = "The set has " + letters.size
+ " values.";
</script>

</body>
</html>
```



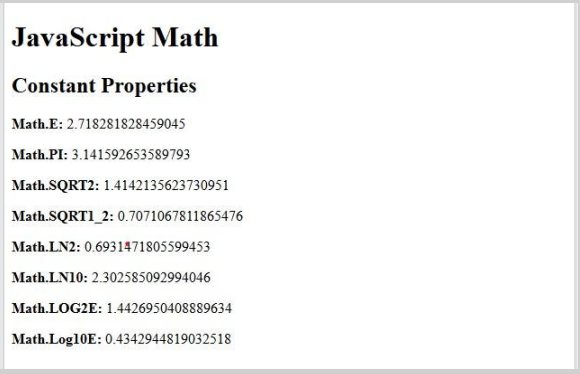
14. **Mapas (Maps) en JavaScript:**
    Estructuras que almacenan pares clave-valor, donde las claves pueden ser de cualquier tipo de dato.

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Maps</h1>
<h2>The new Map() Method</h2>

<p>Creating a map from an array:</p>

<p id="demo"></p>

<script>
// Create a Map
const fruits = new Map([
  ["apples", 500],
  ["bananas", 300],
  ["oranges", 200]
]);

let numb = fruits.get("apples");
document.getElementById("demo").innerHTML = "There are " + numb + "
apples.";
</script>

</body>
</html>
```



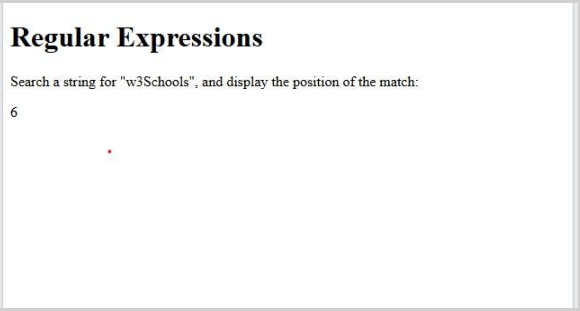
15. **Matemáticas (Math) en JavaScript:**
    Objeto incorporado que proporciona funciones y constantes matemáticas (como Math.sqrt(), Math.random() o Math.PI).

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Math</h1>
<h2>Constant Properties</h2>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
"<p><b>Math.E:</b> " + Math.E + "</p>" +
"<p><b>Math.PI:</b> " + Math.PI + "</p>" +
"<p><b>Math.SQRT2:</b> " + Math.SQRT2 + "</p>" +
"<p><b>Math.SQRT1_2:</b> " + Math.SQRT1_2 + "</p>" +
"<p><b>Math.LN2:</b> " + Math.LN2 + "</p>" +
"<p><b>Math.LN10:</b> " + Math.LN10 + "</p>" +
"<p><b>Math.LOG2E:</b> " + Math.LOG2E + "</p>" +
"<p><b>Math.Log10E:</b> " + Math.LOG10E + "</p>";
</script>

</body>
</html>
```



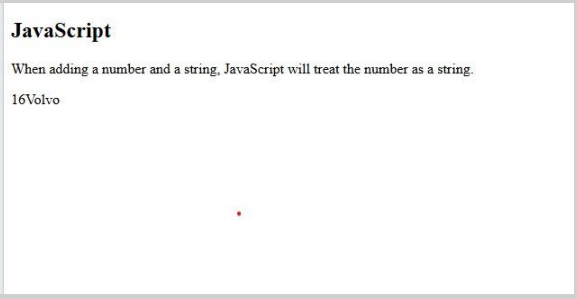
16. **Expresiones Regulares (RegExp):**
    Patrones que se usan para buscar, validar o reemplazar texto dentro de cadenas.

```
<!DOCTYPE html>
<html>
<body>
<h1>Regular Expressions</h1>

<p>Search a string for "w3Schools", and display the position of the match:
</p>

<p id="demo"></p>

<script>
let text = "Visit W3Schools!";
let n = text.search(/w3Schools/i);
document.getElementById("demo").innerHTML = n;
</script>

</body>
</html>
```



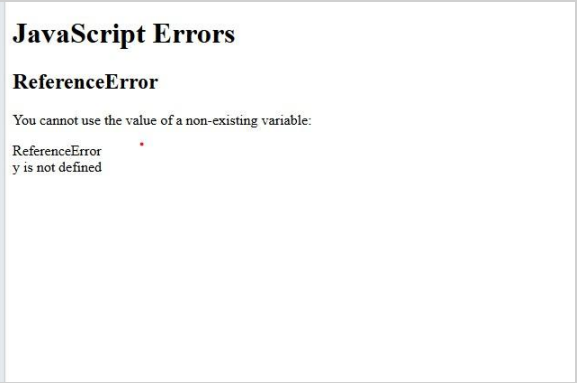
17. **Tipos de Datos en JavaScript:**
    Clasificaciones de valores, como string, number, boolean, object, undefined, null, y symbol.

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript</h2>

<p>When adding a number and a string, JavaScript will treat the number as
a string.</p>

<p id="demo"></p>

<script>
let x = 16 + "Volvo";
document.getElementById("demo").innerHTML = x;
</script>

</body>
</html>
```



18. **Manejo de Errores en JavaScript:**
    Uso de estructuras como try, catch, throw y finally para controlar y gestionar errores durante la ejecución.

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Errors</h1>
<h2>ReferenceError</h2>

<p>You cannot use the value of a non-existing variable:</p>

<p id="demo"></p>

<script>
let x = 5;
try {
  x = y + 1;
}
catch(err) {
  let text = err.name + "<br>" + err.message;
  document.getElementById("demo").innerHTML = text;
}
</script>

</body>
</html>
```



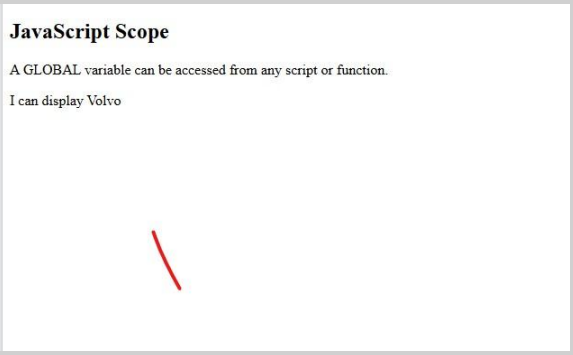
JavaScript Intermedio

19. **Eventos en JavaScript:**
    Acciones que ocurren en la página (clics, teclas presionadas, carga de la página) y que pueden activar funciones.

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript HTML Events</h1>
<h2>The onclick Attribute</h2>

<button onclick="document.getElementById('demo').innerHTML=Date()">The
time is?</button>

<p id="demo"></p>

</body>
</html>
```

20. **Programación con JavaScript8u:**
    Proceso de crear lógica, funciones y estructuras que permiten desarrollar aplicaciones web dinámicas.

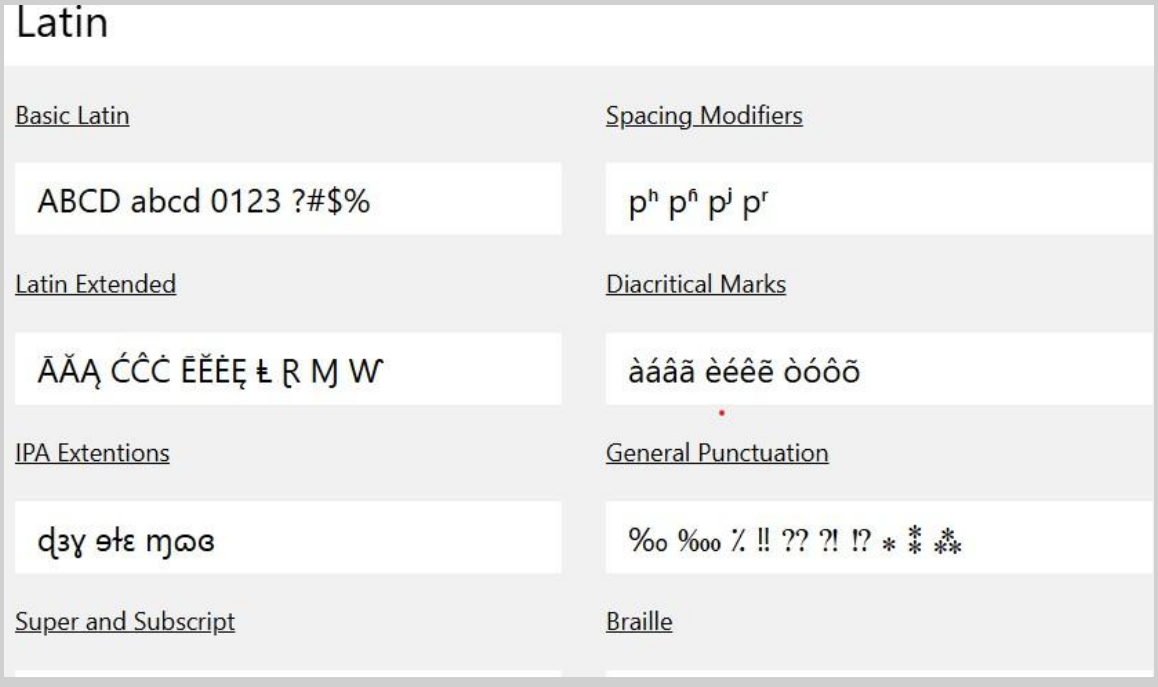
```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Scope</h2>

<p>A GLOBAL variable can be accessed from any script or function.</p>

<p id="demo"></p>

<script>
let carName = "Volvo";
myFunction();

function myFunction() {
  document.getElementById("demo").innerHTML = "I can display " + carName;
}
</script>

</body>
</html>
```



21. **Referencias de JavaScript:**
    Enlaces a documentación o guías oficiales sobre las funciones, objetos y métodos disponibles en el lenguaje

Statement	Description
{ }	Creates an block of statements
async function	Creates an AsyncFunction object
async function*	Creates an AsyncGeneratorFunction object
await using	Declares local variables that are asynchronously disposed
break	Exits a switch or a loop
class	Declares a class
const	Declares a variable with a constant value
continue	Breaks one iteration (in the loop) if a specified condition occurs, and continues with the next iteration in the loop
debugger	Stops the execution of JavaScript, and calls (if available) the debugging function
do...while	Executes a block of statements and repeats the block while a condition is true
for	Loops through a block of code a number of times

22. **Caracteres UTF-8 en JavaScript:**
    Sistema de codificación de caracteres que permite usar símbolos, letras y emojis de casi todos los idiomas.



23. **Versiones de JavaScript:**
    Actualizaciones del lenguaje definidas por ECMAScript (ES5, ES6, ES7, etc.), que agregan nuevas características y mejoras.

JavaScript, invented by Brendan Eich in 1995, became an **ECMA** standard in 1997.

ECMAScript is the official name of the **JavaScript standard**.

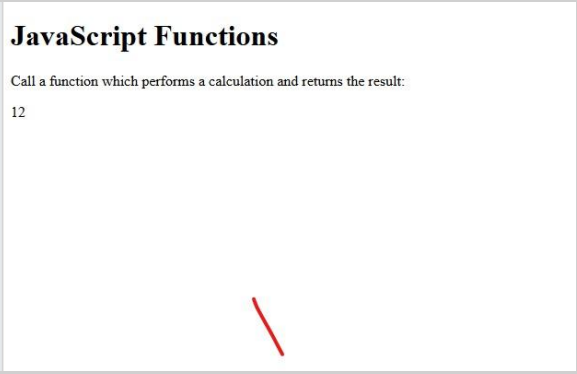
From 1997 ECMA versions was abbreviated by numbers. (ES1, ES2, ES3, ES5, ES6).

2016 - 2025, versions are named by year (ECMAScript 2016, 2017, 2018 ... 2025).

JavaScript Avanzado

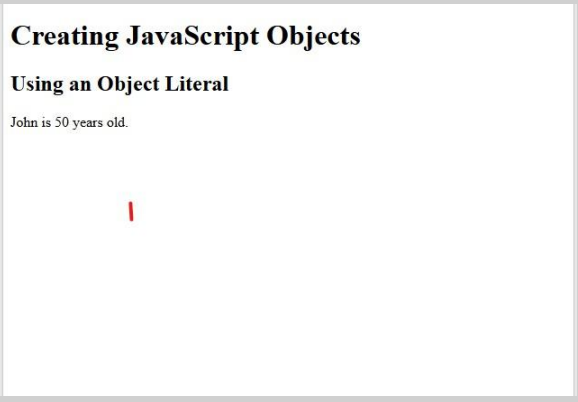
24. **Funciones Avanzadas:**
    Conceptos como funciones flecha, funciones anónimas, callbacks, closures y funciones de orden superior.

```
<!DOCTYPE html>
<html>
<body>

<h1>JavaScript Functions</h1>

<p>Call a function which performs a calculation and returns the result:
</p>

<p id="demo"></p>

<script>
let x = myFunction(4, 3);
document.getElementById("demo").innerHTML = x;

function myFunction(a, b) {
  return a * b;
}
</script>

</body>
</html>
```



25. **Objetos Avanzados:**
    Temas como la herencia prototípica, propiedades dinámicas y el uso de Object.create(), Object.keys(), etc.

```
<!DOCTYPE html>
<html>
<body>
<h1>Creating JavaScript Objects</h1>
<h2>Using an Object Literal</h2>

<p id="demo"></p>

<script>
// Create an Object:
const person = {
  firstName: "John",
  lastName: "Doe",
  age: 50,
  eyeColor: "blue"
};

// Display Data from the Object:
document.getElementById("demo").innerHTML =
person.firstName + " is " + person.age + " years old.";
</script>

</body>
</html>
```



26. **Clases en JavaScript:**
    Sintaxis moderna para crear objetos y definir su comportamiento mediante constructores y métodos.
27. **Iteraciones Avanzadas:**

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Classes</h1>
<p>Creating two car objects from a car class:</p>
```

JavaScript Classes

Creating two car objects from a car class:

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Loops</h1>
<h2>The for Loop</h2>

<p id="demo"></p>

<script>
let text = "";

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

document.getElementById("demo").innerHTML = text;
</script>

</body>
</html>
```

JavaScript Loops

The for Loop

The number is 0
The number is 1
The number is 2
The number is 3
The number is 4

Uso de estructuras como for...of, for...in, map(), filter(), reduce() para recorrer y manipular datos.

28. **Promesas (Promises):**
    Objetos que representan la finalización o el fracaso de una operación asíncrona, mejorando el manejo del tiempo en código.

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Variables</h1>

<p>In this example, x, y, and z are variables.</p>

<p id="demo"></p>

<script>
let x = 5;
let y = 6;
let z = x + y;

document.getElementById("demo").innerHTML = "The value of z is " + z;
</script>

</body>
</html>
```

JavaScript Variables

In this example, x, y, and z are variables.

The value of z is 11

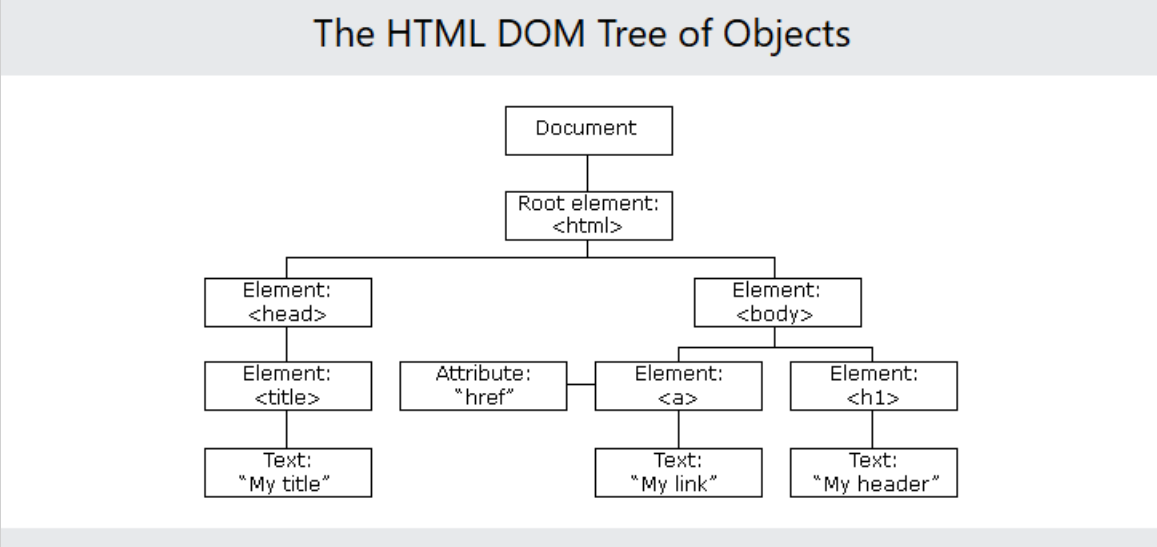
29. **Módulos en JavaScript:**
    Permiten dividir el código en archivos independientes (import y export), facilitando la organización y reutilización.

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Modules</h1>

<p id="demo"></p>

<script type="module">
import { name, age } from "./person.js";

let text = "My name is " + name + ", I am " + age + ".";

document.getElementById("demo").innerHTML = text;
</script>

</body>
</html>
```

JavaScript Modules

My name is Jesse, I am 40.

30. **DOM HTML con JavaScript:**
    Interfaz que permite acceder y manipular los elementos de una página web (texto, imágenes, botones, etc.).



31. **Objeto Window:**
    Representa la ventana del navegador y proporciona métodos y propiedades para interactuar con ella (alert(), setTimeout(), etc.).

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Window</h2>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
"Browser inner window width: " + window.innerWidth + "px<br>" +
"Browser inner window height: " + window.innerHeight + "px";
</script>

</body>
</html>
```

JavaScript Window

Browser inner window width: 634px
Browser inner window height: 480px

32. **Web API:**
    Conjunto de interfaces que permiten interactuar con funcionalidades del navegador como geolocalización, almacenamiento o multimedia.

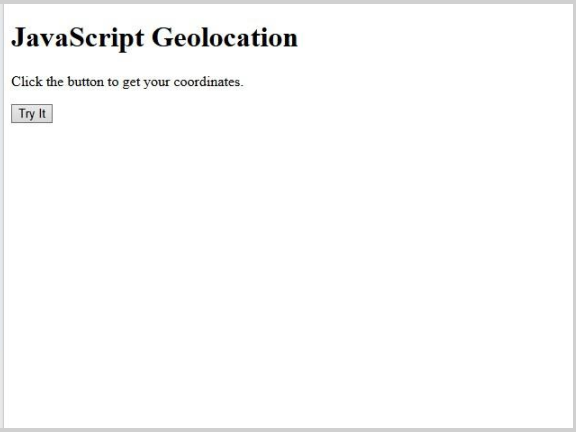
```
<!DOCTYPE html>
<html>
<body>

<div id="demo">
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
    this.responseText;
  }
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>

</body>
</html>
```

The XMLHttpRequest Object

Change Content

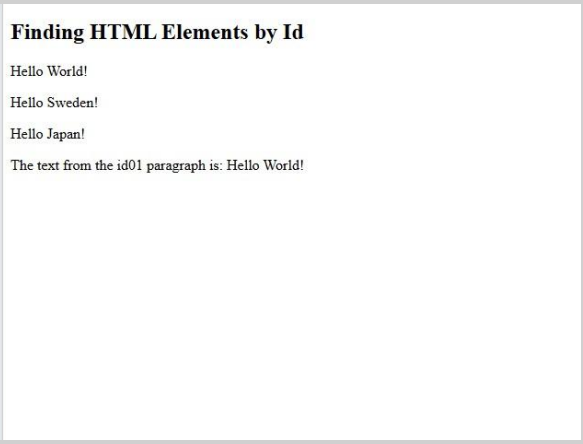
33. **AJAX:**
    Técnica para intercambiar datos con el servidor sin recargar la página, permitiendo contenido dinámico.

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Geolocation</h1>

<p>Click the button to get your coordinates.</p>

<button onclick="getLocation()">Try It</button>

<p id="demo"></p>

<script>
const x = document.getElementById("demo");

function getLocation() {
  try {
    navigator.geolocation.getCurrentPosition(showPosition);
  } catch(err) {
    x.innerHTML = err;
  }
}

function showPosition(position) {
  x.innerHTML = "Latitude: " + position.coords.latitude +
  "<br>Longitude: " + position.coords.longitude;
}
```



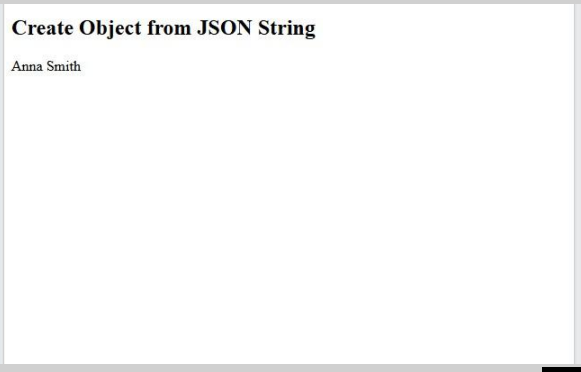
34. **JSON:**
    Formato de intercambio de datos ligero, basado en texto, usado para comunicar información entre cliente y servidor.

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js">
</script>
</head>
<body>

<h2>Finding HTML Elements by Id</h2>
<p id="id01">Hello World!</p>
<p id="id02">Hello Sweden!</p>
<p id="id03">Hello Japan!</p>

<p id="demo"></p>

<script>
$(document).ready(function() {
  var myElements = $("#id01");
  $("#demo").text("The text from the id01 paragraph is: " +
myElements[0].innerHTML);
});
</script>

</body>
</html>
```



35. **jQuery:**
    Biblioteca de JavaScript que simplifica la manipulación del DOM, manejo de eventos y peticiones AJAX.

```
<!DOCTYPE html>
<html>
<body>

<h2>Create Object from JSON String</h2>

<p id="demo"></p>

<script>
let text = '{"employees":[' +
'{"firstName":"John","lastName":"Doe" },' +
'{"firstName":"Anna","lastName":"Smith" },' +
'{"firstName":"Peter","lastName":"Jones" }]}';

const obj = JSON.parse(text);

document.getElementById("demo").innerHTML =
obj.employees[1].firstName + " " + obj.employees[1].lastName;
</script>

</body>
</html>
```



36. **Gráficos con JavaScript:**
    Uso de bibliotecas o APIs (como Canvas o Chart.js) para dibujar y visualizar gráficos o animaciones en la web.

Conclusión:

JavaScript es un lenguaje esencial en el desarrollo web moderno, ya que permite dar vida e interactividad a las páginas y aplicaciones en línea. Su versatilidad lo convierte en una herramienta poderosa tanto para principiantes como para desarrolladores experimentados. A través de sus fundamentos —como las variables, funciones, objetos y estructuras de control— se sientan las bases para comprender cómo se comportan los programas.

En los niveles intermedio y avanzado, JavaScript ofrece recursos más sofisticados como clases, módulos, promesas y manejo del DOM, que facilitan la creación de aplicaciones dinámicas, seguras y escalables. Además, su integración con APIs, AJAX y bibliotecas como jQuery amplía sus capacidades y potencia el desarrollo de experiencias web más fluidas.

En conclusión, dominar JavaScript es fundamental para cualquier persona interesada en la programación web, pues proporciona las herramientas necesarias para transformar ideas en soluciones digitales funcionales e innovadoras.

Bibliografía:

w3schools.
https://www.w3schools.com/js/js_graphics.asp