SINTAXIS BÁSICA DE JAVASCRIPT

ETIQUETA SCRIPT

Cuando va dentro del html, el código va entre las etiquetas **<script>** y **</script>•**

Puede ir en cualquier parte la página, pero es recomendable que lo haga en el encabezado, o al final del código del body, para facilitar la lectura.

```
<html>
    <head>
        <title>Mi primer script</title>
        <script type="text/javascript”>
        // Esta función cargará el texto Hola Javascript
                alert('Hola Javascript!');
        </script>
    </head>
    <body>
    </body>
</html>
```

Javascript es una secuencia de **declaraciones** (*sentencias)* que son ejecutadas por el navegador.

```
document.getElementById("demo").innerHTML="Hello Dolly";
```

Cada sentencia corresponde con un “comando” para el navegador:

- ✓ Es como decirle: “Haz esto”

```
document.getElementById("demo").innerHTML="Hello Dolly";
```

En el documento busca el elemento y escribe dentro

Consideraciones en la escritura:

- ✓ Distingue entre MayúsculaS y minúsculas
- ✓ Ignora los espacios en blanco

```
var foo =         'hola mundo';
```

- ✓ Los paréntesis indican prioridad

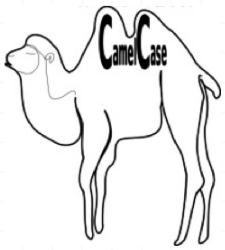
```
2 * 3 + 5;    // es igual a 11, la multiplicación ocurre primero
2 * (3 + 5);  // es igual a 16, por los paréntesis, la suma ocurre primero
```

- ✓ La tabulación ayuda a leer mejor el código pero los espacios en blanco y las nuevas líneas no tiene valor

```
var foo = function() {
  console.log('hola');
};
```

Las sentencias se separan con ;

- ✓ No es obligatorio, pero es una buena costumbre.
- ✓ Se ponen, normalmente, al final.




```
innerHTML="Hello Dolly";
```

El orden de ejecución de las sentencias es el de aparición en el código.

```
1 document.getElementById("demo").innerHTML="Hello Dolly";
2 document.getElementById("myDIV").innerHTML="How are you?";
```

En los nombres de variables y funciones se usa el sistema **camelCase** *(precioFinal)*

Las sentencias se pueden agrupar en **bloques**:

- ✓ Comienzan con { y finalizan con }
- ✓ Los bloques sirven para definir grupos de sentencias que deben ejecutarse conjuntamente
- ✓ Las funciones son un ejemplo de sentencias agrupadas

```
function myFunction()
{
document.getElementById("demo").innerHTML="Hello Dolly";
document.getElementById("myDIV").innerHTML="How are you?";
}
```

Hay dos tipos de **comentarios**:

- ✓ Comentarios en una línea → Comienzan con // (sin cierre)

```
// Write to a heading:
document.getElementById("myH1").innerHTML="Welcome to my Homepage";
// Write to a paragraph:
document.getElementById("myP").innerHTML="This is my first paragraph.";
```

- ✓ Comentarios de más de una línea comienzan con /* y cierran con */

```
/*
The code below will write
to a heading and to a paragraph,
and will represent the start of
my homepage:
*/
document.getElementById("myH1").innerHTML="Welcome to my Homepage";
document.getElementById("myP").innerHTML="This is my first paragraph.";
```

Ejercicio

Accede a este fichero JSFidle o Codepen y comenta solo el código que pasa un registro a la consola para que deje de funcionar.

Solución JSFidle – Codepen

DEPURACIÓN DE CÓDIGO

Errores codificando JS

Los errores más habituales suelen ser de dos tipos:

a) **Errores de sintaxis**:

- Elementos no cerrados
- Codificación incorrecta
- Escritura con errores
- Mayúsculas por minúsculas o viceversa

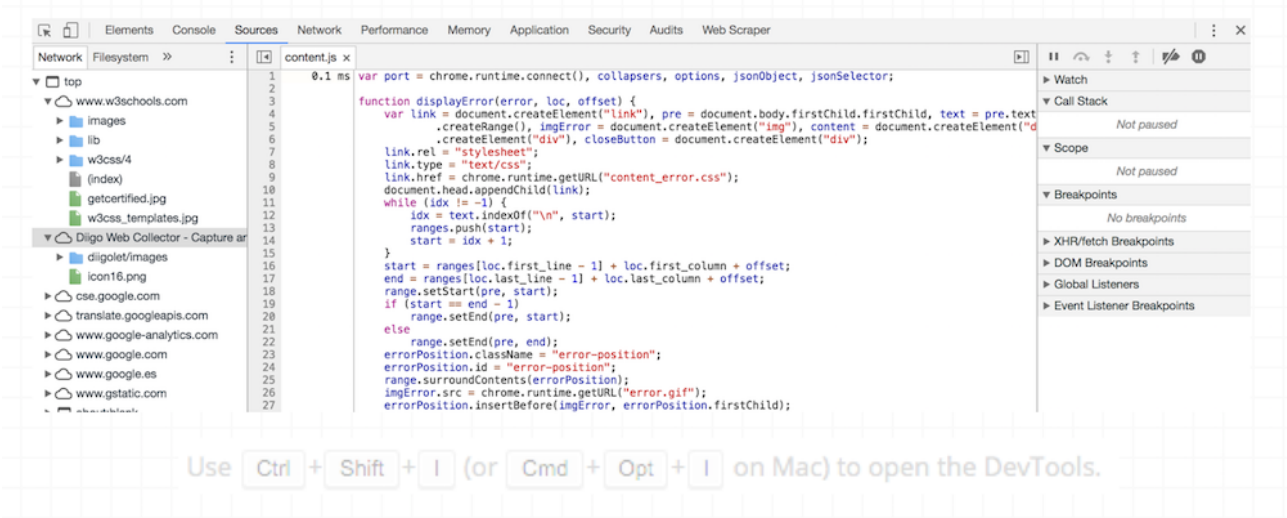
b) **Diferencias entre navegadores**:

- Métodos que no funcionan en algún navegador

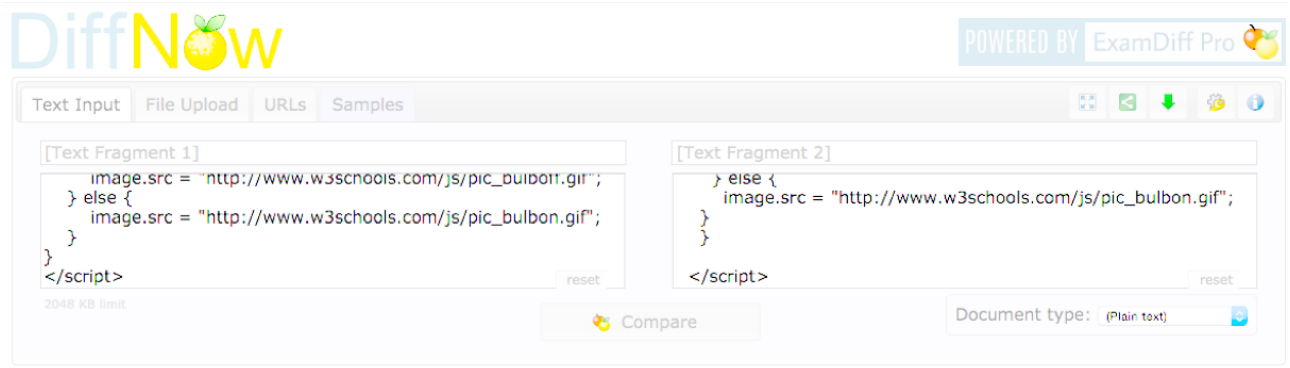
¿Cómo evaluar los errores?

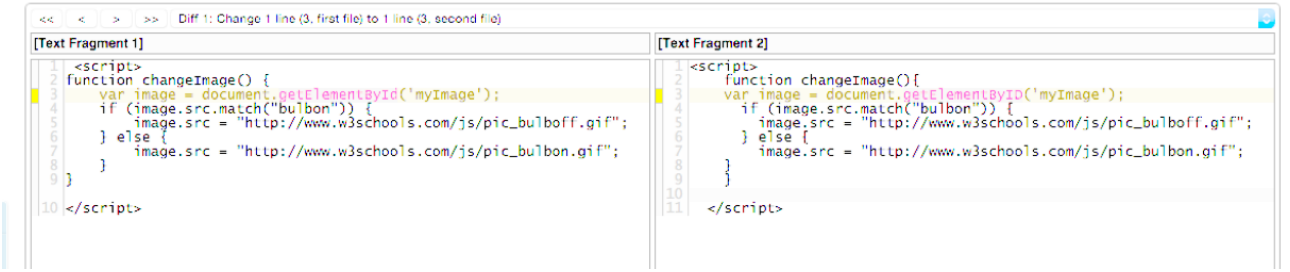
Se pueden evaluar mediante aplicaciones como:

- **Herramienta de desarrolladores** de Chrome para la depuración de código JS como Chrome DevTools



- **Comparadores de código** como Diffnow, que permiten revisar dos versiones de código distintas y localizar las diferencias entre ambas.





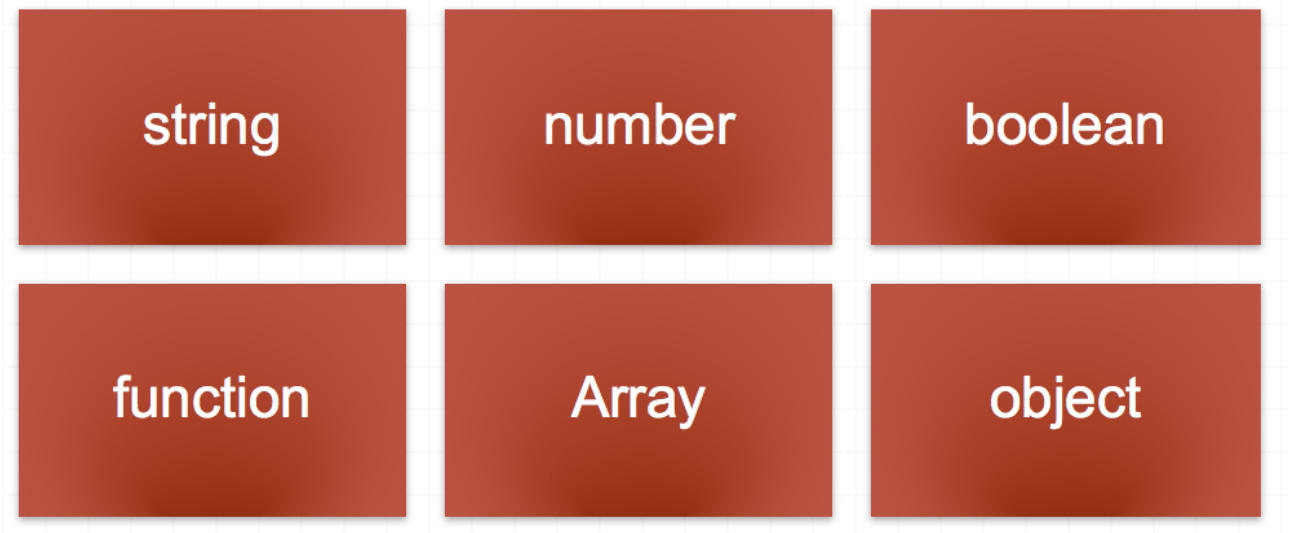
- **Analizadores de código** como jshint, cuya función es evaluar el código para conocer el número de funciones, variables, argumentos, etc. que contiene.



TIPOS DE DATOS

Javascript puede almacenar los siguientes **tipos de datos**:

- string (cadenas)
- number (números)
- boolean (booleanos)
- function (funciones)
- array (arreglo)
- object (objetos)



Cada lenguaje puede almacenar diferentes tipos de datos. Es fundamental comprender cuáles podemos usar y qué características tienen.

String

```
// string
var companyName = “Google”;
```

Number

```
// number
var pi = 3.14;
var year = 2013;
```

Boolean

```
// boolean
var flag = true;
var FALSE = false;
```

Para estructuras condicionales

Function

```
// function
var sayHello = function () {
alert('hello world!');
}
```

Array

```
// array
var numberArray = [1, 2, 3];
var animals = new Array["cat", "dog", "mouse", "lion”];
```

Object

```
// object / json
var person = {
name: 'Barack Hussein Obama II',
age: '51',
title: '44th President of the United States’ }
```

Arrays

- Los **arrays** se conocen también, en otros lenguajes, como vector, matriz, formación….
- En javascript los arrays están **indexados con base cero**, es decir, que el primer item se considera como posición[0], el segundo [1] y así con el resto.
- Se escriben entre corchetes, y los valores se separa con comas

Objetos

- Se escriben entre llaves, y se establece un par nombre:valor, separados por comas (utiliza el sistema de almacenamiento y la sintaxis de JSON que veremos luego más tarde)

Arrays multidimensionales

Un array multidimensional es un array de arrays.

```
var my2Darray = [
                ["bob", "alex", "joe", "wendy"],
                ["teresa", "jodi", "sara", "peter"],
                ["jack", 44, 90],
                [true, false]
];
```

	0	1	2	3
0	bob	Alex	Joe	wendy
1	teresa	Jodi	Sara	Peter
2	Jack	44	90	
3	True	false		

- my2Darray[0][1] //This would return "alex"
- my2Darray[2][2] //This would return 90
- my2Darray[3][0] //This would return true
- my2Darray[2][3] //This is undefined, since there is no fourth elemen

Vamos a ver un ejemplo de array de dos dimensiones. Funciona como una matriz de base 0. En dos dimensiones, la primera es la fila y la segunda la columna. Aunque los arrays pueden tener n dimensiones, normalmente no pasan de 3 porque si no son muy complejos.

Grupo Ciberimaginario | Manuel Gertrudix - Alejandro Carbonell | 2025/2026 | Esta obra está bajo una Licencia Creative Commons Atribución 4.0 Internacional. Los contenidos citados se ajustan a lo regulado en el art. 32 del TRLPI de España

(cc) BY