

Unidad2:



Nombre: Carolina Del Carmen Robaina Jimenéz

1-Evolución de Javascript

JavaScript apareció por primera vez dentro de Netscape Navigator 2 en 1995, basándose en el lenguaje CEnvi.

Como tuvo mucho éxito, Microsoft incorporó su propia implementación de JavaScript llamada JScript en Internet Explorer.

Como había problemas con las implementaciones, ECMA procedió a estandarizar la versión JavaScript 1.1, pero no fue hasta JavaScript 1.3 que se hizo la primera implementación completa del estándar ECMAScript.

En 2004 surge la propuesta de ECMAScript 4, que quiere añadirle a JavaScript el tipado y las clases de Java. Debido a la revolución que suponían estos cambios, surgió la contrapropuesta de ECMAScript 3.1, que evita introducir nuevos conceptos y añade en su lugar nuevas características.

Al final, se eligió ECMAScript 3.1, que pasó a ECMAScript 5. ECMAScript 4 se relegó al proyecto Harmony, que luego pasó a ser ECMAScript 6 incluyendo algunas mejoras propuestas inicialmente por ECMAScript 4.

2-Definir qué es GNU/GPL

Es la licencia pública general, usada para la mayoría de los programas de GNU y para más de la mitad de los paquetes de software libre.

Para que un software sea libre, hay que publicarlo bajo una licencia de software libre como esta.

3-Explicación de todos los operadores de JavaScript

Operadores de asignación

Nombre	Operador abreviado	Significado
Asignación	$x = y$	$x = y$
Asignación de adición	$x += y$	$x = x + y$
Asignación de resta	$x -= y$	$x = x - y$
Asignación de multiplicación	$x *= y$	$x = x * y$
Asignación de división	$x /= y$	$x = x / y$
Asignación de residuo	$x \% = y$	$x = x \% y$
Asignación de exponenciación	$x ** = y$	$x = x ** y$
Asignación de desplazamiento a la izquierda	$x << = y$	$x = x << y$
Asignación de desplazamiento a la derecha	$x >> = y$	$x = x >> y$
Asignación de desplazamiento a la derecha sin signo	$x >>> = y$	$x = x >>> y$
Asignación AND bit a bit	$x \& = y$	$x = x \& y$
Asignación XOR bit a bit	$x \wedge = y$	$x = x \wedge y$
Asignación OR bit a bit	$x = y$	$x = x y$
Asignación AND lógico	$x \&\& = y$	$x \&\& (x = y)$
Asignación OR lógico	$x = y$	$x (x = y)$
Asignación de anulación lógica	$x ?? = y$	$x ?? (x = y)$

Operadores de comparación

Operador	Descripción	Ejemplos que devuelven true
Igual (==)	Devuelve true si los operandos son iguales.	3 == var1 "3" == var1 3 == '3'
No es igual (!=)	Devuelve true si los operandos no son iguales.	var1 != 4 var2 != "3"
Estrictamente igual (===)	Devuelve true si los operandos son iguales y del mismo tipo. Consulta también Object.is y similitud en JS.	3 === var1
Desigualdad estricta (!==)	Devuelve true si los operandos son del mismo tipo pero no iguales, o son de diferente tipo.	var1 !== "3" 3 !== '3'
Mayor que (>)	Devuelve true si el operando izquierdo es mayor que el operando derecho.	var2 > var1 "12" > 2
Mayor o igual que (>=)	Devuelve true si el operando izquierdo es mayor o igual que el operando derecho.	var2 >= var1 var1 >= 3
Menor que (<)	Devuelve true si el operando izquierdo es menor que el operando derecho.	var1 < var2 "2" < 12
Menor o igual que (<=)	Devuelve true si el operando izquierdo es menor o igual que el operando derecho.	var1 <= var2 var2 <= 5

Operadores aritméticos

Operador	Descripción	Ejemplo
Residuo (%)	Operador binario. Devuelve el resto entero de dividir los dos operandos.	12 % 5 devuelve 2.
Incremento (++)	Operador unario. Agrega uno a su operando. Si se usa como operador prefijo (++x), devuelve el valor de su operando después de agregar uno; si se usa como operador sufijo (x++), devuelve el valor de su operando antes de agregar uno.	Si x es 3, ++x establece x en 4 y devuelve 4, mientras que x++ devuelve 3 y , solo entonces, establece x en 4.
Decremento (--)	Operador unario. Resta uno de su operando. El valor de retorno es análogo al del operador de incremento.	Si x es 3, entonces --x establece x en 2 y devuelve 2, mientras que x-- devuelve 3 y, solo entonces, establece x en 2.
Negación unaria (-)	Operador unario. Devuelve la negación de su operando.	Si x es 3, entonces -x devuelve -3.
Positivo unario (+)	Operador unario. Intenta convertir el operando en un número, si aún no lo es.	+"3" devuelve 3. +true devuelve 1.
Operador de exponenciación (**)	Calcula la base a la potencia de exponente, es decir, base^exponente	2 ** 3 returns 8. 10 ** -1 returns 0.1.

Operadores bit a bit

Operador	Uso	Descripción
AND a nivel de bits	$a \& b$	Devuelve un uno en cada posición del bit para los que los bits correspondientes de ambos operandos son unos.
OR a nivel de bits	$a b$	Devuelve un cero en cada posición de bit para el cual los bits correspondientes de ambos operandos son ceros.
XOR a nivel de bits	$a \wedge b$	Devuelve un cero en cada posición de bit para la que los bits correspondientes son iguales. [Devuelve uno en cada posición de bit para la que los bits correspondientes son diferentes].
NOT a nivel de bits	$\sim a$	Invierte los bits de su operando.
Desplazamiento a la izquierda	$a \ll b$	Desplaza a en representación binaria b bits hacia la izquierda, desplazándose en ceros desde la derecha.
Desplazamiento a la derecha de propagación de signo	$a \gg b$	Desplaza a en representación binaria b bits a la derecha, descartando los bits desplazados.
Desplazamiento a la derecha de relleno cero	$a \ggg b$	Desplaza a en representación binaria b bits hacia la derecha, descartando los bits desplazados y desplazándose en ceros desde la izquierda.

Operadores lógicos

Operador	Uso	Descripción
AND Lógico (&&)	expr1 && expr2	Devuelve expr1 si se puede convertir a false; de lo contrario, devuelve expr2. Por lo tanto, cuando se usa con valores booleanos, && devuelve true si ambos operandos son true; de lo contrario, devuelve false.
OR lógico ()	expr1 expr2	Devuelve expr1 si se puede convertir a true; de lo contrario, devuelve expr2. Por lo tanto, cuando se usa con valores booleanos, devuelve true si alguno de los operandos es true; si ambos son falsos, devuelve false.
NOT lógico (!)	!expr	Devuelve false si su único operando se puede convertir a true; de lo contrario, devuelve true.

Operadores de cadena

El operador de concatenación (+) concatena dos valores de cadena, devolviendo otra cadena que es la unión de los dos operandos de cadena.

El operador de asignación abreviada += también se puede utilizar para concatenar cadenas.

Operador condicional ternario

El operador condicional es el único operador de JavaScript que toma tres operandos. El operador puede tener uno de dos valores según una condición. La sintaxis es:

`condition ? val1 : val2`

Operador coma

El operador coma (,) simplemente evalúa ambos operandos y devuelve el valor del último operando. Este operador se utiliza principalmente dentro de un bucle for, para permitir que se actualicen múltiples variables cada vez a través del bucle.

Operadores unarios

Una operación unaria es una operación con un solo operando.

El operador **delete** elimina la propiedad de un objeto.

El operador **typeof** devuelve una cadena que indica el tipo de operando no evaluado.

El operador **void** especifica una expresión que se evaluará sin devolver un valor.

Operadores relacionales

Un operador relacional compara sus operandos y devuelve un valor Boolean basado en si la comparación es verdadera.

El operador **in** devuelve true si la propiedad especificada está en el objeto especificado.

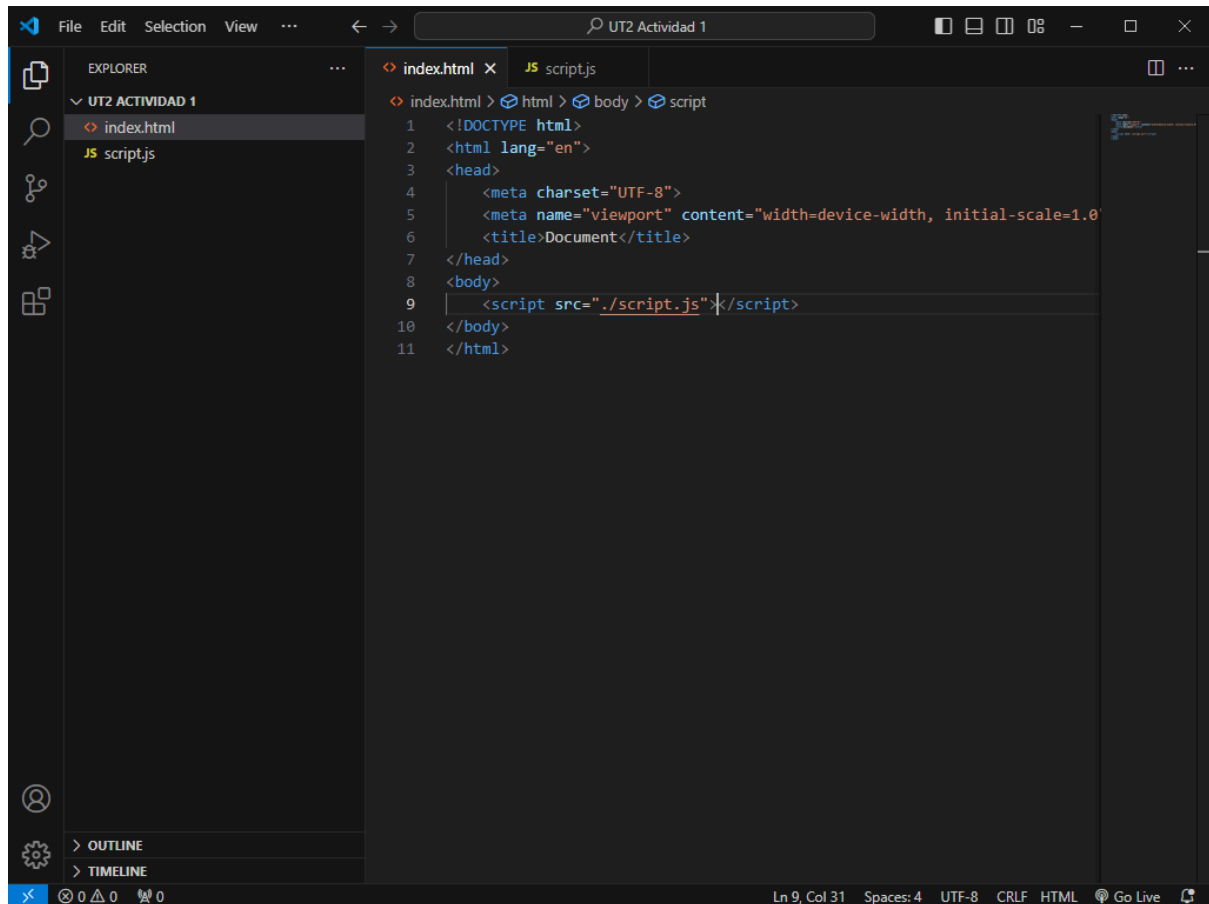
El operador **instanceof** devuelve true si el objeto especificado es del tipo de objeto especificado.

4. Ejercicio: "Hola Mundo" en HTML y JavaScript externo

Objetivo: Crear una página web simple que muestre el mensaje "Hola Mundo" en el navegador, utilizando un archivo HTML y un archivo JavaScript externo.

Instrucciones:

1. Crea un archivo HTML llamado index.html que cargue un archivo JavaScript externo.
Escribiendo ! en un archivo html nos rellena una plantilla básica de HTML

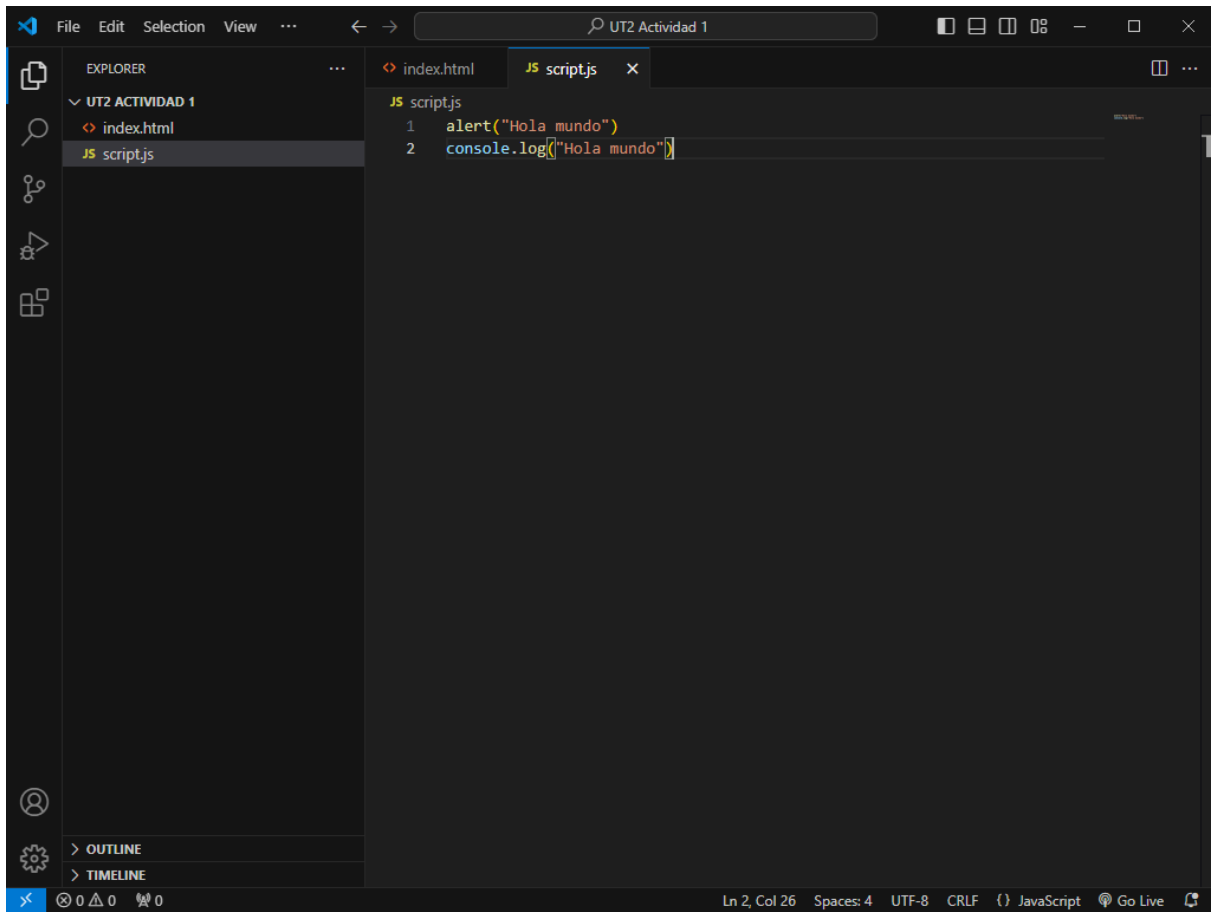


The screenshot shows the Visual Studio Code editor interface. The Explorer panel on the left shows a project named 'UT2 ACTIVIDAD 1' containing two files: 'index.html' and 'script.js'. The main editor area displays the content of 'index.html', which is a basic HTML template. The code is as follows:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9   <script src="../script.js"></script>
10 </body>
11 </html>
```

The status bar at the bottom indicates the current position is Line 9, Column 31, with 4 spaces, using UTF-8 encoding and CRLF line endings. The file is identified as HTML, and the 'Go Live' button is visible on the right.

2. Crea un archivo JavaScript llamado script.js donde se mostrará el mensaje "Hola Mundo" utilizando la consola del navegador y también en la página web.



Abrir en el navegador

