educación IT
CODE YOUR FUTURE

JavaScript desde cero

Módulo 1

Operadores

Operadores: introducción

En programación, los ***operadores*** son **símbolos o palabras clave que se utilizan para realizar manipulaciones sobre valores y variables**. Estas operaciones pueden incluir: aritmética, comparación, asignación, lógica, entre otras.

Los operadores son fundamentales para **construir expresiones y sentencias**. Al combinar variables, constantes y valores literales con operadores, es posible: realizar cálculos, evaluar condiciones y realizar diversas acciones en un programa.

Asignación, incremento y decremento

- El operador de ***asignación*** se utiliza para **guardar un valor en una variable**. Ya lo hemos utilizado previamente. Veamos el ejemplo contiguo:
- El operador de ***incremento*** se indica mediante el **prefijo** **++**. Incrementa la variable en una unidad, tal como se muestra en la segunda imagen.
- El operador de ***decremento*** se indica mediante el **prefijo** **--**. Decrementa la variable en una unidad.

```
let iva = 1.21
```

```
let numero = 12
++numero
console.log('Valor de la variable numero:', numero)
// valor de numero: 13
```

Operadores matemáticos

JavaScript permite realizar operaciones matemáticas:

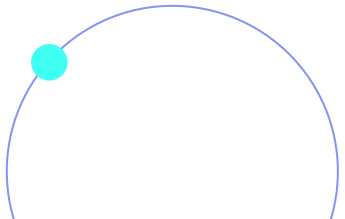
- **suma** (**+**),
- **resta** (**-**),
- **multiplicación** (*****) y
- **división** (**/**).

Veamos algunos ejemplos en las diapositivas a continuación.

La definición de **valores numéricos con decimales no utiliza separador de miles**, y usa **punto** (**.**) **como separador decimal**, por ejemplo:

```
let haberes = 255340.87
let bonoTrimestral = 25400
let totalHaberesMes = haberes + bonoTrimestral

console.log('Haberes a depositar:', totaHaberesMes)
```



Operadores JavaScript orientados a **operaciones aritméticas**, u operadores para realizar operaciones aritméticas básicas:

Tipo de dato	Descripción
+	**Sumar:** en JS, el operador **+** permite sumar dos valores numéricos. También se utiliza para concatenar valores de variables que pueden ser del tipo string, o combinadas como ser un `string` y un `number`.
-	**Restar:** El operador **-** permite realizar una resta entre dos valores numéricos.
*****	**Multiplicar:** El operador ***** permite realizar una multiplicación entre dos valores numéricos.
/	**Dividir:** El operador **/** permite realizar una división entre dos valores numéricos.
%	**Módulo, o Resto:** el operador **%** permite realizar una división entre dos valores numéricos, pero retornará el módulo, o resto de la división.

Los operadores matemáticos se pueden combinar con el operador de asignación para lograr un resultado más rápido y conciso:

En la misma línea de código, se realiza una operación aritmética sobre el valor almacenado en la variable, y se actualiza.

Este tipo de operadores matemáticos combinados, es equivalente al uso convencional de los operadores matemáticos y de asignación que vemos en el segundo ejemplo de código.

Cualquiera de las dos opciones es totalmente válida y funcional en JavaScript.

```
let numero = 9

numero += 4    //operador de igualdad acumulativa
numero -= 1    //operador de asignación de resta
numero *= 1    //operador de asignación de multiplicación
numero /= 3    //operador de asignación divisora
numero %= 4    //operador de asignación de módulo
```

```
let numero = 9

numero = numero + 4
numero = numero - 1
numero = numero * 1
numero = numero / 3
numero = numero % 4
```

Operador +

El operador + se utiliza también para concatenar valores del tipo `String`. Puede ser útil para concatenar dos tipos de datos `String` en un único valor, o para concatenar un tipo de dato `String` y un tipo de dato `Number`, indistintamente del orden en el que se defina.

Para esto último, el resultado final siempre será un tipo de dato `String`, por sobre algún dato numérico.

```
let numero = 9
let leyenda = 'El valor de la variable numero, es: '

let mensaje = leyenda + numero
// El valor de la variable numero, es: 9
```

Operadores de comparación

También llamados “operadores relacionales”, se emplean para comparar y establecer la relación entre ellos. Devuelven un valor booleano (*true*, verdadero, o *false*, falso) basado en la condición.

Veamos la tabla de la siguiente pantalla.

Tipo de dato	Descripción
`===`	Operador de comparación igual: permite comparar dos valores específicos, y retornar un valor booleano **`TRUE`**, si **`valorA`** es igual a **`valorB`**, o **`FALSE`**, si no lo es.
`>`	Operador de comparación mayor que: permite comparar dos valores específicos, y retornar un valor booleano **`TRUE`**, si **`valorA`** es mayor que **`valorB`**, o **`FALSE`**, si no lo es.
`>=`	Operador de comparación mayor o igual que:: permite comparar dos valores específicos, y retornar un valor booleano **`TRUE`**, si **`valorA`** es mayor, o igual, que **`valorB`**, o **`FALSE`**, si no lo es.
`<`	Operador de comparación menor que: permite comparar dos valores específicos, y retornar un valor booleano **`TRUE`**, si **`valorA`** es menor que **`valorB`**, o **`FALSE`**, si no lo es.
`<=`	Operador de comparación mayor que: permite comparar dos valores específicos, y retornar un valor booleano **`TRUE`**, si **`valorA`** es menor, o igual, que **`valorB`**, o **`FALSE`**, si no lo es.
`!==`	Operador de comparación distinto de: permite comparar dos valores específicos, y retornar un valor booleano **`TRUE`**, si **`valorA`** es distinto de **`valorB`**, o **`FALSE`**, si son iguales.

Ejemplo

En el siguiente ejemplo, se pueden verificar los distintos tipos de operadores de comparación, cómo aplicarlos en la realización comparativa de valores, y qué resultado nos retornará cada uno, en cada caso:

```
let nombre = "EducaciónIT"
let empresa = "EDUCACIONIT"

let nroA = 2005
let nroB = 1996

nombre === empresa       //retornará false

nombre !== empresa       //retornará true

nroA > nroB              //retornará true

nroA >= nroB             //retornará true

nroA < nroB              //retornará false

nroA <= nroB             //retornará false
```

Operadores lógicos

Permiten **combinar o invertir valores que se utilizan en comparaciones múltiples**.

Tipo de dato	Descripción
&& **(AND)**	**Operador lógico AND:** se representa con el símbolo **&&** (doble *ampersand*), y permite establecer una **comparación combinada**. Por ejemplo, comparar si `valorA` es igual a `valorB` **&&** si `valorC` es igual a `valorD`. Si ambas condiciones se cumplen, retornará `TRUE` como resultado y, si una de las dos condiciones no se cumple, retornará `FALSE` como resultado.
|| **(OR)**	**Operador lógico OR:** se representa con el símbolo **||** (doble *pipe*), y permite establecer una **comparación combinada / alternativa**. Por ejemplo, comparar si `valorA` es igual a `valorB` **||** si `valorC` es igual a `valorD`. Si al menos una de estas dos comparaciones se cumple, retornará `TRUE` como resultado y, si ninguna se cumple, retornará `FALSE`.
! **(NOT)**	**Operador lógico NOT:** se representa con el carácter **!** (signo de exclamación), y **se combina** generalmente con alguno de los operadores de comparación.

Ejemplo de aplicación

Veamos un ejemplo de aplicación de operadores lógicos. Analicemos **qué valor retorna cada uno de ellos, y porqué lo retorna**.

En este escenario, encontramos la utilización de operadores de comparación combinados con los operadores lógicos.

> El escenario en sí tiene mucho de lo que alguna vez aprendimos en las **matemáticas básicas** que vimos en la escuela. Para refrescar la memoria, compartimos este link que amplía la referencia de la ***tabla de la verdad***, herramienta matemática aplicada en este contexto.

```
let nombre = "EducaciónIT"
let empresa = "EDUCACIONIT"

let nroA = 2005
let nroB = 1996

nombre !== empresa && nroA > nroB    //retornará true

nombre === empresa && nroA > nroB    //retornará false

nombre === empresa || nroA < nroB    //retornará false

empresa !== nombre || nroA > nroB    //retornará true
```

Revisión

- Repasar qué es un operador.
- Trabajar con variables y operadores en sus **diferentes tipos**.
- Implementar operadores y mostrar los datos en la consola JS.
- Aplicar todas las propiedades en el ***Proyecto integrador***.
- Realizar las preguntas necesarias al/la docente antes de continuar.

¡Sigamos trabajando!