Operadores de comparación

Ya sabemos la estructura del código if/else, pero todavía nos falta un fundamento crucial para poder comenzar a utilizar los condicionales: Los operadores lógicos y de comparación.

Operadores de comparación

Nos sirven y usamos para comparar elementos entre sí:

¿Es dos mayor a 4?

¿La edad del usuario es mayor o igual que 18?

¿La contraseña ingresada es correcta?

¿El número ingresado es igual al número ganador?

¿Su DNI es distinto al ingresado?

Si prestamos atención, en todas estas preguntas hay dos comparaciones y son estas comparaciones las que vamos a empezar a usar en los condicionales que creemos.

Los operadores lógicos en JavaScript se escriben de la siguiente manera:

Operador	Escritura
Mayor	>
Menor	<
Mayor Igual	>=
Menor Igual	<=
Igual	==
Igual estricto	===
Distinto	!

Utilizando operadores de comparación

Supongamos que el usuario ingresa su edad mediante un prompt:

Ejemplo #1 – Mayor igual

let edad = prompt('ingrese su edad');

Si queremos saber si esta persona es mayor de edad, podemos preguntar si edad es mayor o igual a 18:

```
const edad = 18;
if (edad >= 18) {
  console.log("Tu edad e MAYOR a 18");
}
```

Ejemplo #2 – Igual Estricto

Imaginemos el mismo ejemplo anterior, pero esta vez edad va a ser un String y no un Number. Si hacemos la comparación utilizando == vamos a ver que sí se ejecuta la función dentro de las llaves, porque lo que hace el operador == es comparar el valor:

```
const edad = "18";
if (edad == 18) {
  console.log("Tu edad es igual a 18");
}
```

En este caso, “18” (string) es igual a 18 (number) porque el valor 18 es igual para ambos.

Muchas veces puede pasar que no solo queramos comparar valor, sino también querramos comparar el tipo de dato (si es un number, un string, un boolean, etc). Es en esta ocasión en donde usamos el triple igual o comparador estricto ===

```
const edad = "18";
if (edad === 18) {
  console.log("Tu edad es igual a 18");
}
```

En este segundo ejemplo, el código entre las llaves no se ejecutará porque si bien el valor 18 es el mismo, el tipo de dato no:

- Edad es un string
- 18 es un number

Ejemplo #3 - Menor

Supongamos que queremos listar ciertos productos cuyo precio sea menor a $500, en este caso podríamos utilizar un menor:

```
const precio = 320;
if (precio < 500) {
  console.log("El precio es menor que 500");
}
```

Ejemplo #4 – Booleano

Siempre que pensamos en el condicional if, lo que pensamos es si se cumple una condición, entonces ejecuto el código entre llaves y sino ejecuto lo que está dentro del else. Otra manera de ver esto mismo, es pensar “si la condición es verdadera” entonces ejecuto el código entre llaves. Un ejemplo de esto podría ser:

```
if (esMujer === true) {
  console.log("La persona es mujer");
} else {
  console.log("La persona es de un género distinto a mujer");
}
```

Sin embargo si bien este código funciona, podemos hacer una simplificación:

Hacer

```
if (esMujer === true)
```

o hacer

```
if (esMujer)
```

es exactamente lo mismo. Cuando evaluamos un variable booleana, podemos evitar ese "=== true" porque recordemos que lo que evalúa el if **siempre va a ser** si condición es true.

Ejemplo #5 – Distinto

Supongamos que queremos listar a todas las personas que viven en el AMBA (Área Metropolitana de Buenos Aires), podríamos hacer algo como:

```
const area = "AMBA";
if (area === "AMBA") {
  console.log("El usuario es del AMBA");
}
```

Sin embargo, podríamos también invertir la lógica y decir "Si el usuario NO es del amba, decimos que NO es del AMBA).

```
if (area != "AMBA") {
  console.log("El usuario NO es del AMBA");
}
```

Si bien ahora parece algo similar, con el tiempo utilizar el operador de negación tendrá muchas utilidades.