Operadores Lógicos

Ya sabemos lo que es un if, ya sabemos lo que es un else, ya conocimos los operadores de comparación:

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

Y ahora nos queda conocer qué son los operadores lógicos, de esta manera podremos comenzar a armar programas que resuelvan problemáticas de mayor complejidad.

Operadores Lógicos, ¿qué son?

Son operadores que nos permiten hacer comparaciones más complejas. Pensemos por ejemplo en algún operador conocido, el mayor igual:

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

Si la edad es mayor o igual a 18, entonces mostramos ese cartel por consola. Supongamos ahora que necesitamos que se cumplan sí o sí dos condiciones como por ejemplo

- Condición #1: Que la edad sea mayor que 18
- Condición #2: Que viva en Mendoza

Para este caso, podemos hacer dos cosas.

1. Anidar un if else
2. Utilizar el operador lógico AND

Anidar un if else

```
5   const edad = 18;
6   const provincia = "Mendoza";
7
8   if (edad >= 18) {
9     if (provincia === "Mendoza") {
10      console.log("El usuario cumple con los requisitos");
11    } else {
12      console.log("El usuario no es de Mendoza");
13    }
14  } else {
15    console.log("El usuario no es mayor de 18 años");
16  }
```

Esta lógica, como vimos en presentaciones anteriores, nos permite ir paso a paso y hacer validaciones específicas. Podemos fácilmente darnos cuenta cuál es la condición que falla: si es la edad o la provincia.

Si bien esta forma de plantear un programa con sus validaciones es correcto, hay programas en donde no vamos a necesitar tanto detalle. Imaginemos la situación en donde lo único que nos importa es que el usuario sea mayor que 18 y viva en Mendoza, si no cumple con alguno de estos requisitos (sin importar cuál), quiero arrojar un mensaje de error. Es en este caso que entran los operadores lógicos

Utilizar el operador lógico AND

Siguiendo con el ejemplo anterior, en nuestro programa vamos a tener dos condiciones:

- Condición #1: Que la edad sea mayor que 18
- Condición #2: Que viva en Mendoza

y para evaluar dos condiciones en simultáneo, utilizamos el operador lógico **&&**

```
8  if (edad >= 18 && provincia == "Mendoza") {
9    console.log("El usuario cumple con los requisitos");
10 } else {
11   console.log("El usuario NO cumple con los requisitos");
12 }
```

El operador AND se escribe utilizando doble ampersand &&. Lo que hace es permite evaluar más de una condición en simultáneo y lo que dice es:

> Todas las condiciones se deben cumplir para dar **true**

Si alguna de las condiciones **no** se cumple, el resultado nunca será true por lo cuál nunca entraría al código de la línea 9, en donde el usuario cumple con los requisitos.

Para entender más a fondo este operador, chusmeá la presentación de operadores lógicos, en la sección &&.

Operador lógico OR

En el anterior, todas las condiciones se tenían que cumplir sí o sí, en este **con que al menos una condición se cumpla** la evaluación va a dar verdadera.

Pensemos por ejemplo en que queremos formar un equipo que sea de personas que sepan programar o que quieran aprender a programar. Si lo pensamos, tenemos dos condiciones

1. Condición #1: Que sepa programar
2. Condición #2: Que quiera aprender a programar

La diferencia con el programa anterior, es que acá solo se tiene que cumplir **una o la otra,** no hace falta que se cumplan ambas. Esa es la función de este operador:

Solo una de las condiciones se debe cumplir para dar **true**

Utilizar el operador lógico OR

Siguiendo el ejemplo anterior, para evaluar estas condiciones vamos a usar el operado OR, que se escribe así: `||` (sí, esas dos barras)

Retomando el programa anterior, podemos decir que tenemos dos condiciones:

1. Condición #1: Que sepa programar
2. Condición #2: Que quiera aprender a programar

Si se cumple por lo menos una, la evaluación dará true

```
const sabeProgramar = false;
const quiereAprenderAProgramar = true;

if(sabeProgramar === true || quiereAprenderAProgramar === true) {
  console.log("Bienvenido/a al equipo!");
} else {
  console.log("No pudiste entrar al equipo :( ");
}
```

Para entender más a fondo este operador, chusmeá la presentación de operadores lógicos, en la sección ||.

Operador Lógico NOT

Este operador es un operador de negación. Si lo aplicamos sobre algo que es verdadero, va a dar falso, si lo aplicamos sobre algo que es falso, va a dar verdadero. En otras palabras, lo invierte.

Para utilizarlo, se usa el signo de exclamación `!`

Si tenemos

```
console.log(!true);
```

esto dará false, porque estoy negando el true.

```
console.log(!false);
```

esto dará true, porque estoy negando el false.

Utilizando el operador Lógico NOT

Pensemos en un ejemplo en donde tenemos una clave y vamos a impedir el paso a aquellos usuarios cuya clave no coincide con la nuestra

```
32  if(password != "xyz123322r#$&") {
33      console.log("Contraseña Incorrecta");
34  } else {
35      console.log("Contraseña correcta");
36  }
```

Si la contraseña es distinta a la que estamos evaluando, entonces no lo dejamos pasar y si es correcta sí lo dejamos pasar.