***Operadores***

Aritméticos:  
5 + 2 7  
5 – 2 3  
5 \* 2 10  
5 / 2 2.5   
5 % 2 1 // Resto da Divisão por Inteiro  
5 \*\* 2 25 // Potencia   
  
Precedência   
()   
\*\*  
\* / %  
 + -

### Atribuição: ++ (antes ou depois) -- += -= \*= /= %=

Simplificação  
var n = 3 Variáveis   
n = n + 4 // Simplificando fica n+=  
  
Relacionais:

**booleanas**são variáveis capazes de conter apenas 1 de 2 valores: verdadeiro ou falso.  
5 > 2 true // maior que  
7 < 4 false //menor que  
8 >=8 true // maior ou igual  
9<=7 false // menor ou igual  
5 == 5 true // igual  
4 != 4 false // diferente  
!== //desigual restrito  
  
Identidade  
5 == 5 true  
5 == ‘5’ true //entre “” é uma string  
5 === “5” false //false porque === significa IDENTICO, e 5 é num e “5” é uma String  
5 === 5 true

Lógicos  
*! // negação*  
!true – false  
!false – true  
  
*&& // conjunção*  
true&&true -true  
true&&false -false  
false&&true -false  
false&&false -false  
  
*|| // disjunção*  
true||true -true  
true||false -true  
false||true -true  
false||false -false  
  
Precedência  
() \*\* /..  
> < >= ..  
!  
&&  
||  
  
Exemplos  
idade >= 15 && idade <= 17 // a idade está entre 15 e 17?  
estado == ‘RJ’ || estado == ‘SP’ //o estado é RJ ou SP?  
salário > 1500 && sexo != ‘m’ //o salario é acima de 1500 e não é um homen?

Ternário  
**?:**  
(Média >= 7) ? (“aprovado”) : (reprovado)  
  
var média = 5.5  
média > 7? ’aprovado’ : ’reprovado  
*reprovado  
  
var x = 8  
var res = x % 2 == 0 ? 5: 9  
  
res é 5*

n = n + 4 // Simplificando fica n+=

Imagine que você precise comparar dois valores dentro de uma condição para que ela seja cumprida. Mas como devemos utilizar cada operador para isso? Descubra aqui nesse artigo!

Sempre que queremos avaliar uma condição no código, precisamos utilizar operadores de comparação para isso. Um exemplo, é quando queremos imprimir um número apenas se ele for positivo. Nesse caso, utilizamos o famoso “**if**” e colocamos a condição dentro do parêntesis, como mostro abaixo:

javascript

if(numero > 0){

console.log(numero);

}

Quando analisamos a condição (numero > 0), podemos perceber que o símbolo "**>**" coloca como condição **o valor da esquerda ser maior que o valor da direita**. É como se dentro do parêntesis estivesse escrito: “numero maior que 0”. Sendo assim, para cada tipo de comparação, temos um operador diferente. Veja a tabela abaixo:

| **Operador** | **Operação** | **Exemplo** |
| --- | --- | --- |
| > | Maior que | (a > b) |
| < | Menor que | (a < b) |
| >= | Maior ou igual a | (a >= b) |
| <= | Menor ou igual a | (a <= b) |
| == | Igual a | (a == b) |
| != | Diferente de | (a !== b) |
| === | Idêntico a | (a === b) |
| !== | Não idêntico a | (a !== b) |
| && | E/and | (a && b) |
| ll | Ou/or | (a ll b) |

**Diferença entre “==” e “===”**

É possível perceber na tabela acima, que existe tanto o símbolo de “igual a” (==), quanto o de “idêntico a” (===). Mas qual eu devo utilizar para cada situação? Se usarmos o operador “==” em uma condição, saberemos que ela só será verdadeira se **o valor da esquerda for o mesmo valor da direita**, por exemplo:

javascript

if (true == “true”) *//retorna true*

if (10 == “10”) *//retorna true*

if (true == true) *//retorna true*

if (10 == 10) *//retorna true*

Mas suponha que seja preciso fazer essa validação tendo certeza de que o número da direita é realmente um número, e não apenas possui o mesmo caractere. O que fazemos?

É nessa hora que entra o símbolo de “idêntico a” (===). Ele não só **compara os valores** dos dois lados da equação, como também verifica **se eles são do mesmo tipo**. Por exemplo:

javascript

if (true === “true”) *//retorna false*

if (10 === “10”) *//retorna false*

if (true === true) *//retorna true*

if (10 === 10) *//retorna true*

E assim como os operadores “==” e “===”, usamos a mesma lógica para os seus inversos “!=” e “!==”. Veja a seguir:

if (true != “true”) *//retorna false*

if (4 != “4”) *//retorna false*

if (true !== “true”) *//retorna true*

if (4 !== “4”) *//retorna true*

if (true !== true) *//retorna false*

if (4 !== 4) *//retorna false*