**>\_**

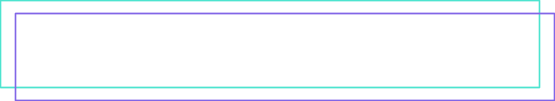
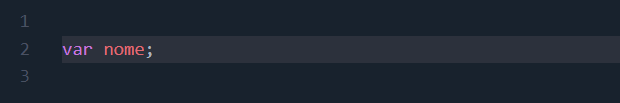


O que são variáveis?

As **variáveis** são recursos das linguagens de programação que nos **permitem armazenar** informações. Podemos pensar em variáveis como se fossem gavetas: cada uma delas funciona como uma variável diferente, onde é possível guardar **coisas** dentro, e utilizar, quando necessário.

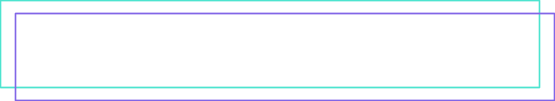
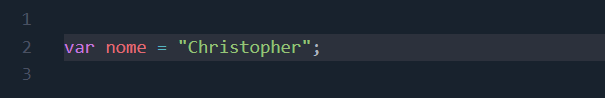
**>\_**

Declaração e tipos de variáveis



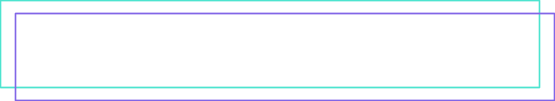
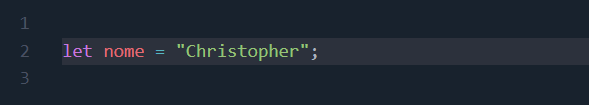
Podemos observar uma **declaração** de variável no exemplo abaixo:

Para iniciar a declaração, **utilizamos** a palavra reservada **var**, em seguida atribuímos o identiﬁcador **nome**, que usaremos quando quisermos referenciar a variável.

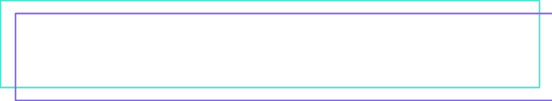
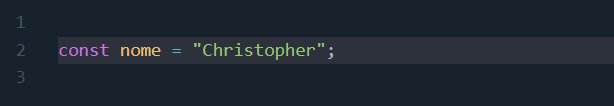


Também é possível **atribuir valores**, como no exemplo abaixo:

Nesta segunda declaração, **atribuímos** um valor **à variável**, diferente da primeira declaração. Com isso, a variável **nome** possui **uma informação** – um texto – que poderíamos acessar a qualquer momento.



No Javascript também há **outra forma** de declarar variáveis, **utilizando** a palavra reservada **let** ao invés de **var**. Em geral, a utilização de **let possui** mais **benefícios** em comparação ao **var**, por isso, devemos utilizar, **por padrão**, **let** para declarar variáveis.



Também é possível declarar **variáveis** com valores **imutáveis** – que não mudam – utilizando **constantes**. Esse recurso é muito utilizado, e considerado uma boa prática, pois dessa forma podemos assegurar o valor da variável, **impedindo** qualquer **alteração**.

Com a utilização de constantes, **caso** o valor seja **alterado**, o compilador **irá gerar** um

**erro**, impedindo a execução do algoritmo. Com isso tornamos nosso código mais seguro.

# >\_

## O que são tipos de dados?

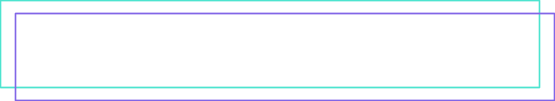
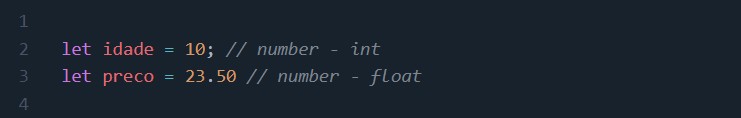
Na programação trabalhamos com **diferentes tipos de informações**. Devido a isso, existem diferentes **tipos de dados** para lidar com cada uma delas. Usar o tipo de dado correto **para** lidar com **certa informação** é essencial para um bom funcionamento do código.

# >\_

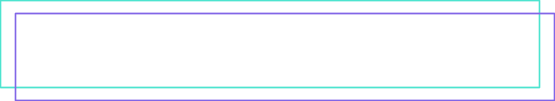
## Diferentes tipos de dados

Existem três tipos de dados **mais comuns** na programação, que são o **Number**, **String** e

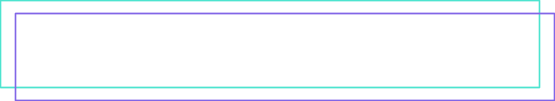
**Boolean**. Abaixo veremos uma descrição sobre cada um deles:



* **Number**: Tipo de dado numérico, muito utilizado para cálculos. Em algumas linguagens existem subdivisões dentro desse tipo, como **Int** indicando números inteiros, e **Float** indicando números com ponto ﬂutuante (reais).



* **String**: Tipo de dado **textual**, que pode armazenar **qualquer** tipo de **caractere**, inclusive números. É importante ressaltar que por ser do tipo String, mesmo que possua números, **não** é recomendado **realizar cálculos**, pois o resultado pode acabar sendo uma concatenação;

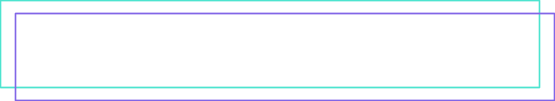
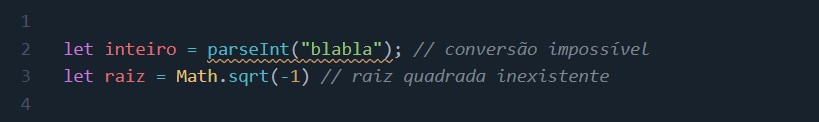


* **Boolean**: Tipo de dado lógico, sendo **possível** apenas dois valores, **true (verdadeiro)** ou **false (falso)**. É utilizado para armazenar valores lógicos oriundos de comparações de quaisquer tipos de dados.

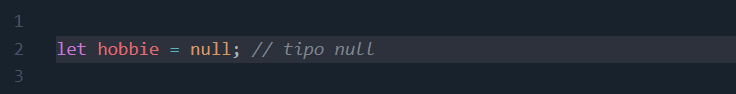
>\_

Tipos de dados especiais

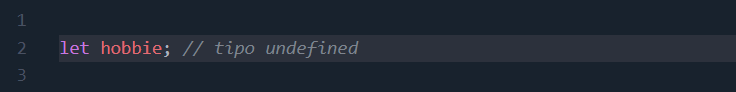
Também existem outros **tipos** de dados, chamados **especiais**. São eles:



* **NaN**: **Not a Number (Não é um número) -** Esse tipo de dado **indica** uma operação em que o valor resultante **não** pode ser passado como **um número**;



* **Null**: **Indica** um tipo **vazio** – um elemento **sem** qualquer **valor** –, porém uma variável com **Null** no Javascript é considerada com valor, no caso um valor nulo;



* **Undeﬁned**: Diferente do tipo **Null**, **Undeﬁned** signiﬁca uma variável sem valor deﬁnido – com **ausência de valor** – e normalmente ocorre ao declarar uma variável sem **atribuição**.

# >\_

O que são operadores?

Os **operadores** são recursos que **nos permitem** manipular variáveis, **realizar**

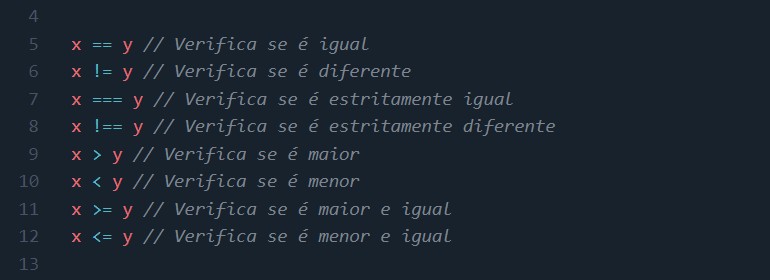
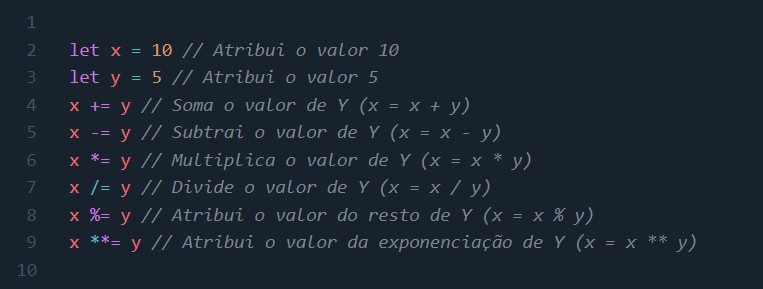
comparações e **operações** com seus valores.

# >\_

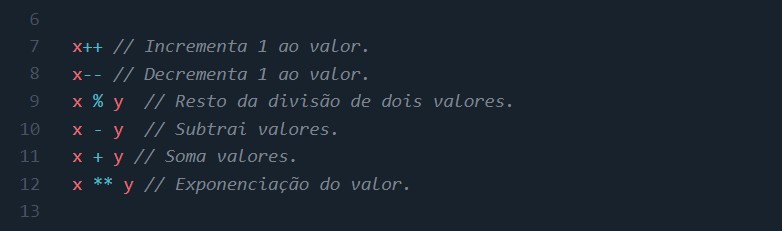
Tipos de Operadores

Existem diversos tipos de operadores, os mais comuns são os de **Atribuição**, **Comparação**, **Aritméticos**, **Lógicos** e **Concatenação**.

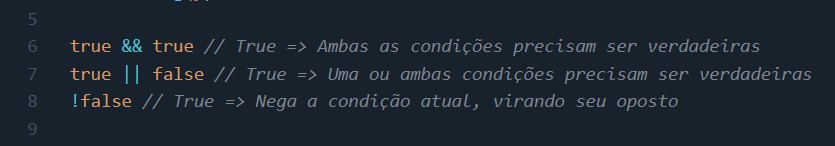
* **Atribuição**: **Atribui** valores às variáveis;



* **Comparação**: Realizam **comparações** que retornam **true** ou **false;**

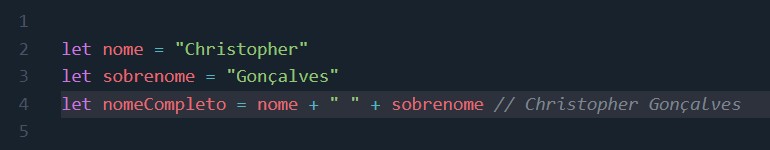


* **Aritméticos**: Realizam operações como **Soma**, **Subtração** etc;



* **Lógicos**: Veriﬁcam comparações e retornam **true** ou **false**. São utilizados os sinais **&&** (Ambos valores verdadeiros), **||** (Ambos ou um valor verdadeiro) e **!** (Nega a condição);

Tem ordem de verificação independente de qual ordem estão, primeiro é verificando **!**, depois **&&** e por último **||**.



* **Concatenação: Concatena** textos – juntam suas partes – **em** apenas **uma**.

window.prompt(``) //abre uma janela pedindo para preencher  
window.confirm(``)//abre uma janela pedindo para confirmar com CONFIRM e CANCELAR  
window.alert(``)//abre uma janela com um alerta ou informação para o usuário  
document.write(``) //escreve na tela  
document.writeln(``) //escreve e pula a linha

\*\*Transformar em números  
Number.parseInt(variavel); //transformar string ou numero float em Inteiro  
Number.parseFloat(variavel); // transformar string ou inteiro em Float  
Number(variavel); // transforma string em número, o JS decide se vai ser Inteiro ou Float  
variável.toFixed(2); // coloca 2 casa decimais em um Number.  
variável.toFixed(2).replace(‘.’ , ’,’)// replace troca algum dado, no caso de .(ponto) para (virgula) para transformar casas decimais para o padrão BR.  
variável.toLocaleString(‘pt-BR’, {style: ‘currency’, currency: ‘BRL’}) //adiciona o formato da moeda brasileira ou de outros países como USD dólar, EUR euro.

\*\* Transformar em string  
String(variavel) || variável.toString() // converte número ou outros dados em String.

\*\*padronizar forma das letras  
toUpperCase() || variável.toUpperCase() //transforma toda String em letras maiúsculas.  
toLowerCase() || variável.toLowerCase() // transforma toda String em letras minúsculas.  
s.length // conta quantos caracteres a string tem.