**Data Types:**

* **String > Number:**
  + parseInt(n) // número inteiro ex: 6
  + parseFloat(n) // número real ex: 6.5
  + Number(n) // se vira para fazer a conversão
  + typeof // identifica o tipo da variável
* **Number > String:**
  + String(n) ou n.toString()

**Tipos de Dados em JavaScript:**

* **Number**
  + Infinity
  + NaN
* **String**
* **Boolean**
* **Null**
* **Undefined**
* **Object**
  + Array
* **Function**

**Var:**

* **Evite usar var**, pois ele tem um comportamento de escopo menos previsível (escopo de função) e pode causar bugs.
* Ele permite **redeclaração** e **hoisting**, o que pode confundir.  
  Use let ou const no lugar.

**Let:**

* Use let quando precisar de uma variável que pode mudar de valor.
* Tem **escopo de bloco** (existe apenas dentro de {}).

**Const:**

* Use const para variáveis que **não devem ser reatribuídas**.
* Tem **escopo de bloco**.
* Ideal para valores constantes, arrays ou objetos que não serão reatribuídos.

**Formatações de String:**

* s.length // Tamanho da string
* s.toUpperCase() // Converte para maiúsculas
* s.toLowerCase() // Converte para minúsculas

**Formatações de Números:**

var n1 = 1543.5

n1.toFixed(2) // 1543.50 (duas casas decimais)

n1.toFixed(2).replace('.', ',') // 1543,50

n1.toLocaleString('pt-BR', {style: 'currency', currency: 'BRL'}) // "R$ 1.545,50"

**Operadores JS:**

* **Aritméticos:**

5 + 2 // 7

5 - 2 // 3

5 \* 2 // 10

5 / 2 // 2.5 (divisão real)

5 % 2 // 1 (divisão inteira)

5 \*\* 2 // 25 (potência, 5^2)

* **Exemplo de operação encadeada:**

var a = 5 + 3 // 8

var b = a % 5 // 3

var c = 5 \* b \*\* 2 // 45

var d = 10 - a / 2 // 6

var e = 6 \* 2 / d // 2

var f = b % e + 4 / e // 3

**Auto-atribuições:**

var n = 3

n += 4 // 7

n /= 2 // 3.5

n \*= 4 // 14

n \*\*= 2 // 196

n %= 5 // 1

**Operadores Relacionais:**

* 5 > 2 // true
* 7 < 4 // false
* 8 >= 8 // true
* 9 <= 7 // false
* 5 == 5 // true
* 4 != 4 // false

**Operadores de Identidade:**

* 5 == '5' // true (não testa o tipo)
* 5 === '5' // false (testa se é idêntico, incluindo o tipo)
* 5 === 5 // true

**Operadores Lógicos:**

* **! (negação)**
  + !true // false
* **&& (conjunção)**
  + true && true // true
  + true && false // false
* **|| (disjunção)**
  + true || false // true
  + false || false // false

**Operador Ternário:**

* condição ? valor\_true : valor\_false
  + Exemplo: media >= 7.0 ? "Aprovado" : "Reprovado"

**DOM (Document Object Model):**

* **Árvore DOM**: Representação hierárquica dos elementos HTML.

**Selecionando elementos no DOM:**

* **Por Marca:** getElementsByTagName()
* **Por ID:** getElementById()
* **Por Nome:** getElementsByName()
* **Por Classe:** getElementsByClassName()
* **Por Seletor:** querySelector() ou querySelectorAll()

**Estruturas Condicionais:**

* **Simples:**

if (condição) {

// código

}

* **Composta:**

if (condição) {

// código

} else {

// código

}

* **Aninhadas:**

if (condição1) {

// código

} else if (condição2) {

// código

} else {

// código

}

**Repetições:**

* **While:**

while (condição) {

// bloco de código

}

* **Do...While:**

do {

// bloco de código

} while (condição)

* **For:**

for (início; teste; incremento) {

// bloco de código

}

**Array (Variáveis Compostas):**

let frutas = ['maçã', 'banana', 'laranja']

console.log(frutas[0]) // maçã

num[3] = 6 // Adicionar um valor no índice 3

num.push(10) // Adicionar valor ao final do array

num.sort() // Ordenar o array

num.indexOf(7) // Procurar o valor 7 no array