BACK-END

AULA 2

**Parte 1:**

Término e envio de pesquisa; Socialização da pesquisa.

**Parte 2:**

**JavaScript**

* Linguagem de alto nível e dinâmica.
* Muito usada em desenvolvimento web.
* Trabalha a interatividade.
* Funciona tanto no front-end (navegador; cliente-side), quanto no back-end (servidor; Node.js).
* Baseada em classes, programação funcional e assíncrona (permite a execução de mais de um processo de uma vez)

**Console.log**

É o “print” do JavaScript.

Ex Python: print()

**Ex JavaScript: console.log()**

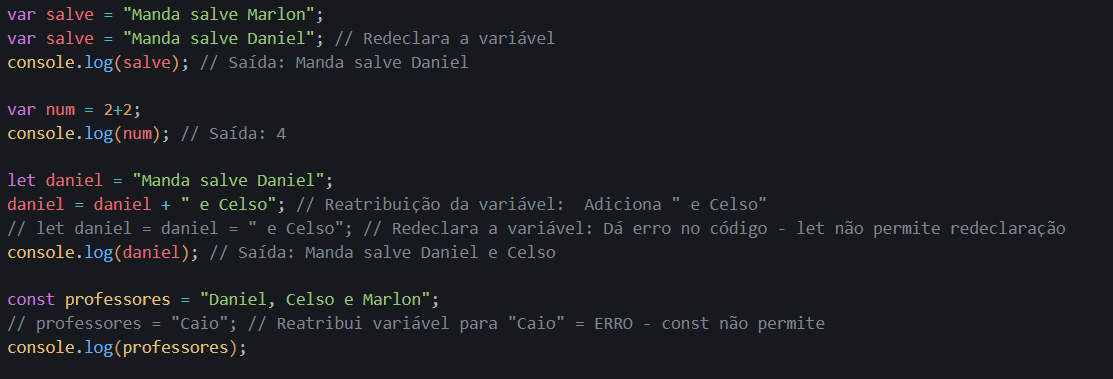
Ex Java: system.out.printIn()

**Prática no VSCode e Chrome**

**Variáveis**

No JavaScript, existem três formas para declarar variáveis:

* var
  + Pode ser redeclarada e reatribuída
  + Tem escopo de função
  + Forma antiga de declaração, não é mais recomendado
  + Ex: var frase = “Salve Daniel”;  
     console.log(frase) // Escreve: Salve Daniel
* let
  + Pode ser reatribuído, mas não pode ser redeclarado no mesmo escopo
  + Tem escopo de bloco
  + Forma nova de declaração, é mais recomendado
  + Ex: let idade = 25;  
     idade = 30;  
     console.log(idade) // Mostra: 30
* const
  + Não pode ser reatribuída
  + ...

Exemplos: 

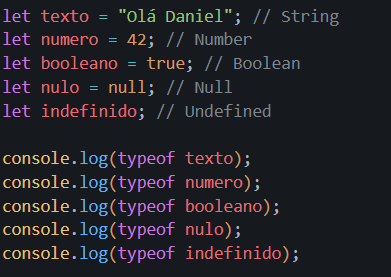
**camelCase**

Em JavaScript, existe um padrão de utilizar camelCase para nomear variáveis e funções. A primeira palavra deve sempre começar com letra minúscula, a primeira letra de cada NOVA palavra deve começar com letra maiúscula.

Ex: valorDoPedido; nomeCompleto.

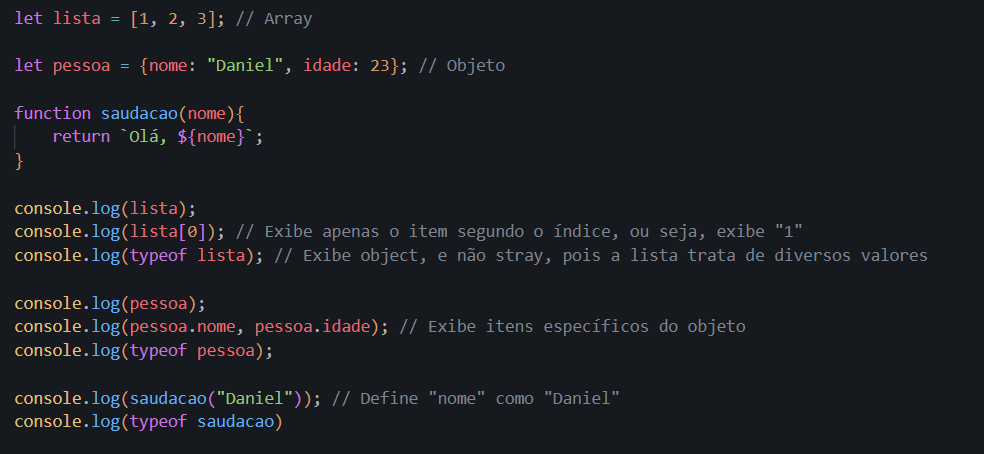
**Tipos de Dados**

* Dados primitivos:
  + String: “Texto”
  + Number: numero
  + Boolean: True or False
  + Nulo: null – valor nulo (Mostrado como object, pois pode vir a ser um no futuro, já que não possui valor)
  + Indefinido: - Não define valor
  + Exemplos:



* Dados Estruturais:
  + Array: Lista de valores;
  + Object: Coleção chave-valor, utilizado em entidades complexas;
  + Function: Conjunto de instruções;

Exemplos:



(Caso você use uma estrutura como: `Olá, ${nome}`; são necessárias crases: `)

**Operadores Aritméticos**

Mesmos operadores da matemática:

“ + ”; “ – “; “ \* ”; “ / “.

% - Modulação

\*\* - Exponenciação

**Operadores relacionais**

Mesmos do Python:

“>” – Maior que (1 > 2 // False)

“<” – Menor que (1 < 2 // True)

“>=” Maior ou Igual (1 >= 1 // True)

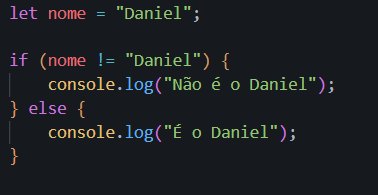
“<=” Menor ou Igual (5 <= 10 // True)

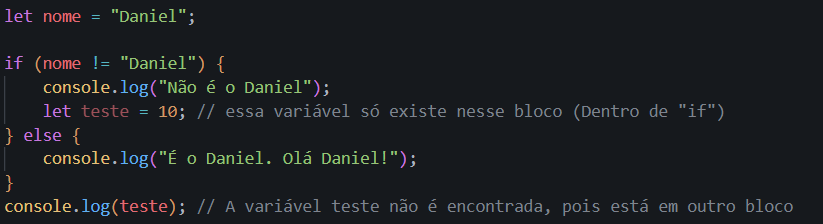
**Operadores Lógicos**

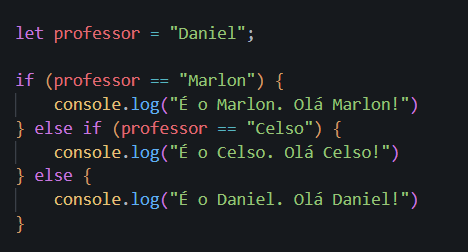
* && - E lógico (true && false // False)
* || - OU lógico (true || false // True)
* ! – Negação (!true // False)

**Estruturas condicionais**

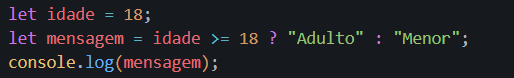
* **if** e **else**:



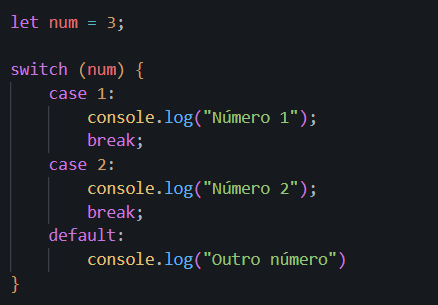
* Exemplo com “let” sendo testado: 
* else if:



* Estrutura de tomada de decisão:

// Saída: Adulto

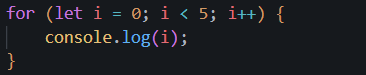
* Switch Case:

Alterna entre casos.

**Laços de repetição**

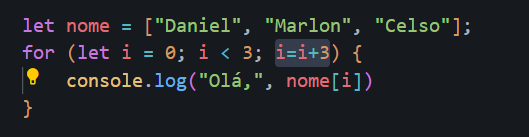
Utiliza-se para repetir um processo várias vezes:

* for:

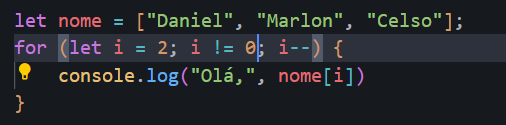


“i” começa em 0, enquanto “i” for menor que 5, adicionar 1 ao valor de “i”. Mostrar o valor de “i”. Resultado:





i=i+3 pode definir quanto será adicionado a i, nesse caso o código só seria executado uma vez e escreveria “Olá, Daniel”.



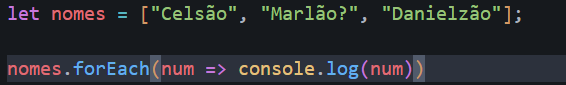
i-- (faz o oposto do i++)

resultado:

“Olá, Celso”

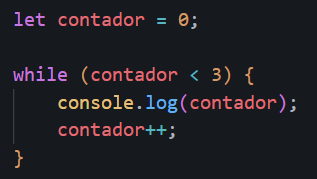
“Olá, Marlon”

forEach:



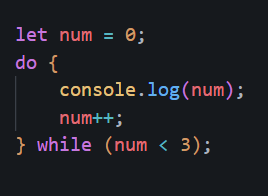
O forEach() é um método de arrays em JavaScript que executa uma função para cada um dos elementos da lista, em ordem.

while:



Utilizado quando não se sabe a quantidade de vezes que tem que repetir

do while:



Executa uma vez o código antes de testar o while, diferente do while que testa primeiro e depois executa.

**Lista (Arrays):**