<LAB365> SENAI

SEJAM BEM-VINDOS!

Jornada DEV AM. CE. RR

Formação gratuita
em Desenvolvimento
Front-End SENAI

Começamos em breve!

A aula vai iniciar em **5 minutos**.

Relaxe e aproveite a música enquanto aguardamos todos chegarem!

Lógica com JavaScript #02

SENAI

Variáveis

Agenda

- Variáveis e funções
  - Revisão
  - Regras
- Operadores de comparação
- Estrutura de condição

O que são Variáveis?

- Usamos variáveis em nosso código sempre que queremos **armazenar** um valor para ser reutilizado posteriormente;

- Uma variável funciona como uma “etiqueta” que indica o endereço do valor atribuído à ela na memória do nosso computador.

Espaço de armazenamento

- Podemos imaginar a memória do computador como um enorme “armário” cheio de “gavetas”;
- Guardamos nosso valor “Olá mundo!” em uma gaveta de nome “primeiraVariavel”.

```
let primeiraVariavel = "Olá mundo!";
```

Funções

Funções

Funções são blocos de código reutilizáveis que executam uma tarefa específica. Elas permitem:

- Organizar o código em blocos lógicos
- Evitar repetição de código (DRY - Don't Repeat Yourself)
- Facilitar a manutenção e testes
- Criar abstrações para operações complexas

Sintaxe

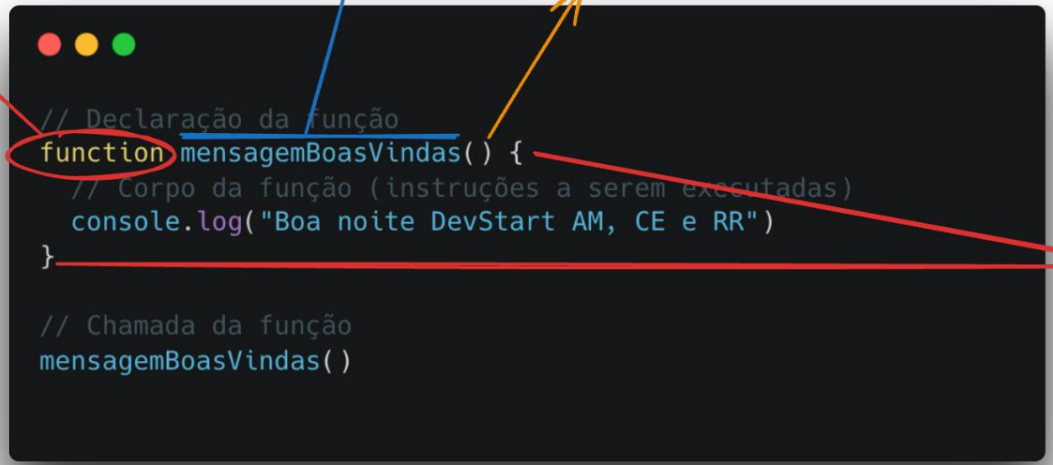
```
// Declaração da função
function mensagemBoasVindas() {
  // Corpo da função (instruções a serem executadas)
  console.log("Boa noite DevStart AM, CE e RR")
}

// Chamada da função
mensagemBoasVindas()
```

Sintaxe

Palavra reservada para a declaração da função.

Nome da sua função

Parênteses - característica principal de uma função



Delimitam o bloco de código

Função com parâmetros

```
// Função nomeada (declaração)
function somar(a, b) {
    return a + b;
}

// Chamada
let resultado = somar(5, 3); // 8
```

Exercício função com parâmetros

Regras de Variáveis e Funções em JavaScript

Regras de Variáveis e Funções em JavaScript

- JavaScript possui **convenções** e **regras** importantes para o uso de variáveis e funções.
- Entender essas regras evita erros comuns no código.

Não é possível redefinir variáveis

Errado:

```
let nome = "Aurora"
let nome = "Amora" // Erro: já foi declarada
```

Correto:

```
let nome = "Aurora"
nome = "Amora"
```

Case-sensitive (maiúsculas e minúsculas)

Variáveis e funções diferenciam letras maiúsculas e minúsculas.

```
let nome = "João"
let Nome = "Maria"

console.log(nome) // João
console.log(Nome) // Maria
```

São **variáveis diferentes**!

Funções também diferenciam maiúsculas

```
function test() {
  console.log("função minúscula")
}

function Test() {
  console.log("função com maiúscula")
}

test() // função minúscula
Test() // função com maiúscula
```

São **funções diferentes**!

Não use variáveis antes de declarar

Errado:

```
console.log(nome) // Erro: não declarada
let nome = "Aurora"
```

Correto:

```
let nome = "Aurora"
console.log(nome)
```

Resumindo as regras

- Javascript diferencia maiúsculas de minúsculas (case-sensitive).
- Variáveis e funções são **sensíveis a maiúsculas/minúsculas**.
- Não use variáveis antes de declará-las.

Revisão

OPERADORES JAVASCRIPT

OPERADORES JAVASCRIPT

- **Operadores matemáticos:** são os operadores utilizados para realizar operações matemáticas com dados do tipo number.

Operador	Operação	Exemplo
+	Soma	`1 + 1 // 2`
-	Subtração	`10 - 4 // 6`
*	Multiplicação	`5 * 1000 // 5000`
**	Exponenciação	`2 ** 8 // 256`
/	Divisão	`1024 / 8 // 128`
%	Resto	`18 % 2 // 0` `9 % 4 // 1`

OPERADORES JAVASCRIPT

- **Operadores de comparação:** são operadores que comparam os valores e sempre retorna um boolean.

Operador	Operação	Exemplo
`==`	Igual a	`"1" == 1  // true`
`===`	Estritamente igual a	`"1" === 1  // false`
`>, <`	Maior que, menor que	`20 > 20  // false`
`>=, <=`	Maior ou igual, menor ou igual	`20 <= 20  // true`
`!=`	Diferente de	`1 != 2  // true`
`!==`	Estritamente diferente de	`1 !== "1"  // true`

Vamos praticar

Operadores de comparação

- Operadores de comparação
- Comparação de igualdade

Variáveis e Funções

- retribuição de valor
- Criar sua primeira função
- Criar uma função que recebe parâmetros
- Criar um função de conversão de moedas
- Conversor de metros para quilômetros

Intervalo!

Finalizamos o nosso primeiro período de hoje.

Nos vemos em 20min.

Início: 19:50
Retorno: 20:10

Estrutura condição

Estruturas Condicionais

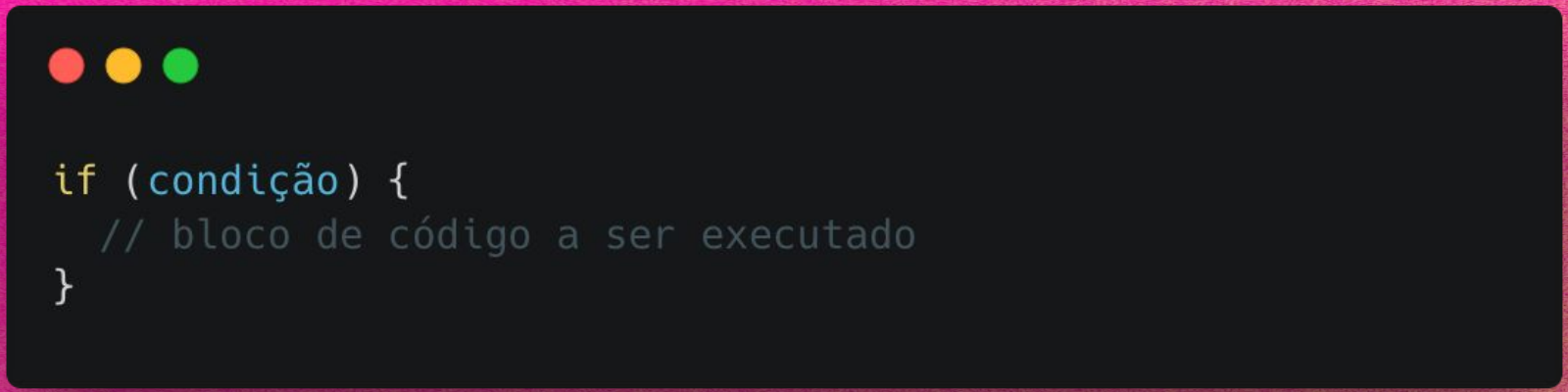
As estruturas condicionais permitem que seu código execute diferentes blocos de instruções com base em condições específicas. **É como tomar decisões no dia a dia**: se chove, pegamos o guarda-chuva; se não, deixamos em casa.

Por que usar ?

- Controlar o fluxo de execução do programa.
- Executar código diferente com base em condições.
- Criar lógica de negócios complexa.

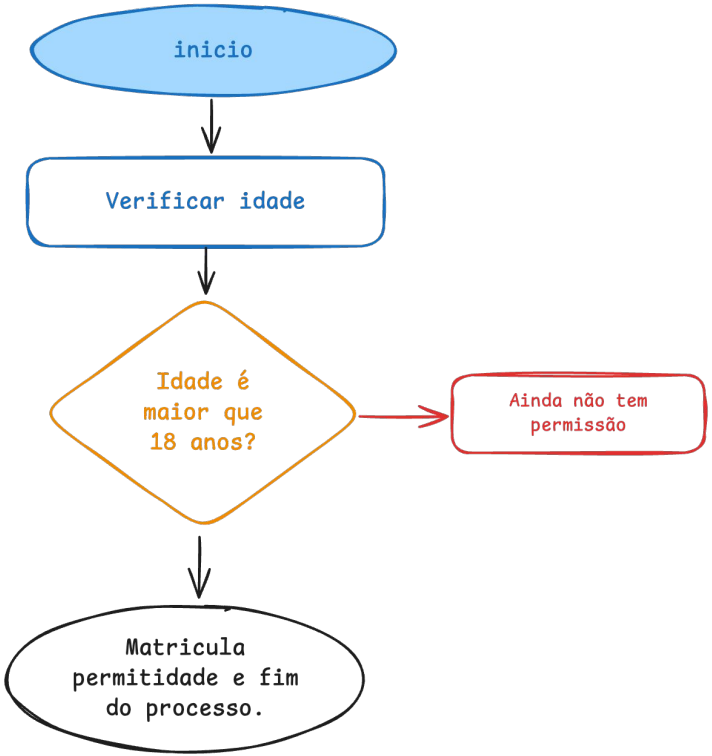
Comando de decisão SE (if)

O **if** executa um bloco de código se a condição for verdadeira.



```
if (condição) {
  // bloco de código a ser executado
}
```

Exemplo prático




```
let idade = 19;

if(idade > 18) {
  return console.log("✅ Matricula aceita!")
}

console.log("❌ Ainda não tem permissão.")
```

Qual será a sua saída?

```
let idade = 18;

if(idade > 18) {
  console.log("Você é maior de idade.");
}
```

Vamos praticar

Comparação

- Idade mínima para dirigir
- Verificação de login
- Temperatura do clima
- Saldo em conta
- Validação de entrada de texto (.trim())

Condicional no DevStart

- Recurso de ramificação
- Votação brasileira
- Positivo?
- Lista de compras

Dúvidas?

A prática é a maior homenagem que podemos fazer ao conhecimento.


<LAB365>
SENAI