

Aulas Senac

Desenvolvimento Web para Adolescentes

Pablo Garcia

Funções

Funções

- O que é uma função ?
- Por que usar funções?
- Quais benefícios?
- O que acontece se não se preocupar com organização e reaproveitamento?



O que é uma função

javascript

```
function saudacao(nome) {  
    return `Olá, ${nome}!`;  
}  
  
console.log(saudacao("Ana")); // Saída: Olá, Ana!
```

Uma função é um bloco de código reutilizável que executa uma tarefa específica. Ela pode receber entradas, chamadas de parâmetros ou argumentos, realizar operações com esses dados e, em seguida, retornar um resultado ou apenas executar uma ação. A função permite agrupar um conjunto de instruções sob um nome, o que facilita seu uso repetido.

O que é uma função

Prática

1. Função de Boas-Vindas

Objetivo: Criar uma função básica que exiba uma mensagem.

2. Função de Saudação Personalizada

Objetivo: Trabalhar com parâmetros e argumentos.

3. Soma de Dois Números

Objetivo: Realizar operações com números dentro de funções.

Por que usar funções?

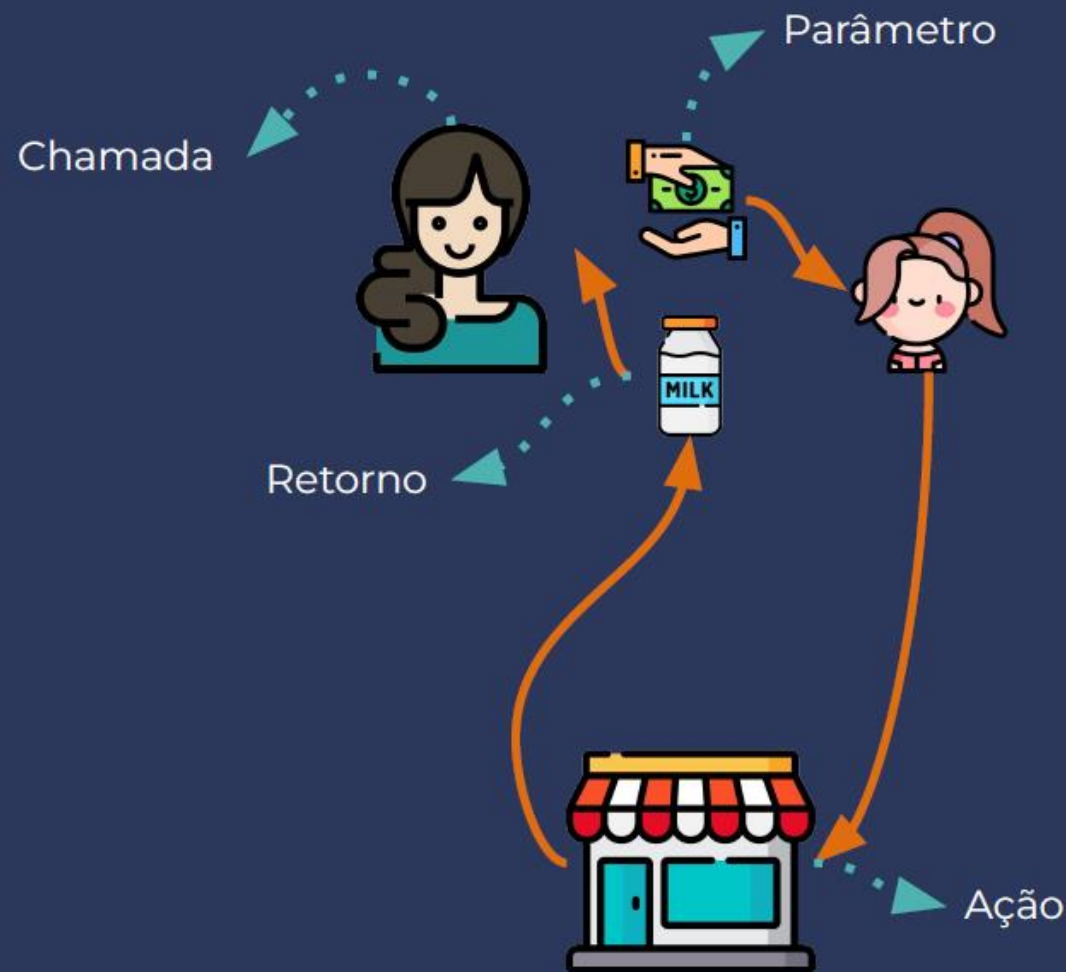


Funções são usadas para organizar o código, permitindo dividir um problema em partes menores e mais fáceis de gerenciar. Elas permitem a reutilização de código, evitam duplicação e facilitam a manutenção, além de tornar o código mais legível.

Quais os benefícios?

- 1. Reutilização de código:** Uma vez definida, a função pode ser chamada em diferentes partes do programa sem a necessidade de reescrever o mesmo código várias vezes.
- 2. Organização:** Funções ajudam a quebrar um código complexo em pedaços menores e mais gerenciáveis, o que torna o desenvolvimento e a depuração mais fáceis.
- 3. Manutenção:** Se um erro for encontrado ou uma melhoria precisar ser feita, você só precisa alterar a função uma vez, e isso afetará todos os lugares onde ela é chamada.
- 4. Legibilidade:** Funções bem nomeadas tornam o código mais fácil de entender, pois encapsulam detalhes de implementação complexos em um único ponto.

Funções



Toda função para ser executada precisa ser chamada. A chamada pode ser feita por alguém ou por um evento automático.

Parâmetros são informações que são passadas para as funções, são os inputs que as funções precisam para processar.

A ação é o passo a passo, é o algoritmo, é a sequência de ações que precisa ser executada para realizar o que foi mandado.

O retorno é a entrega do resultado para quem chamou a função.

Prática

4. Média de Três Números

Objetivo: Manipular números com uma média aritmética.

5. Conversão de Temperatura (Celsius para Fahrenheit)

Objetivo: Aplicar fórmulas dentro de uma função.

6. Verificação de Número Par

Objetivo: Trabalhar com operadores de comparação e estrutura condicional.

Funções

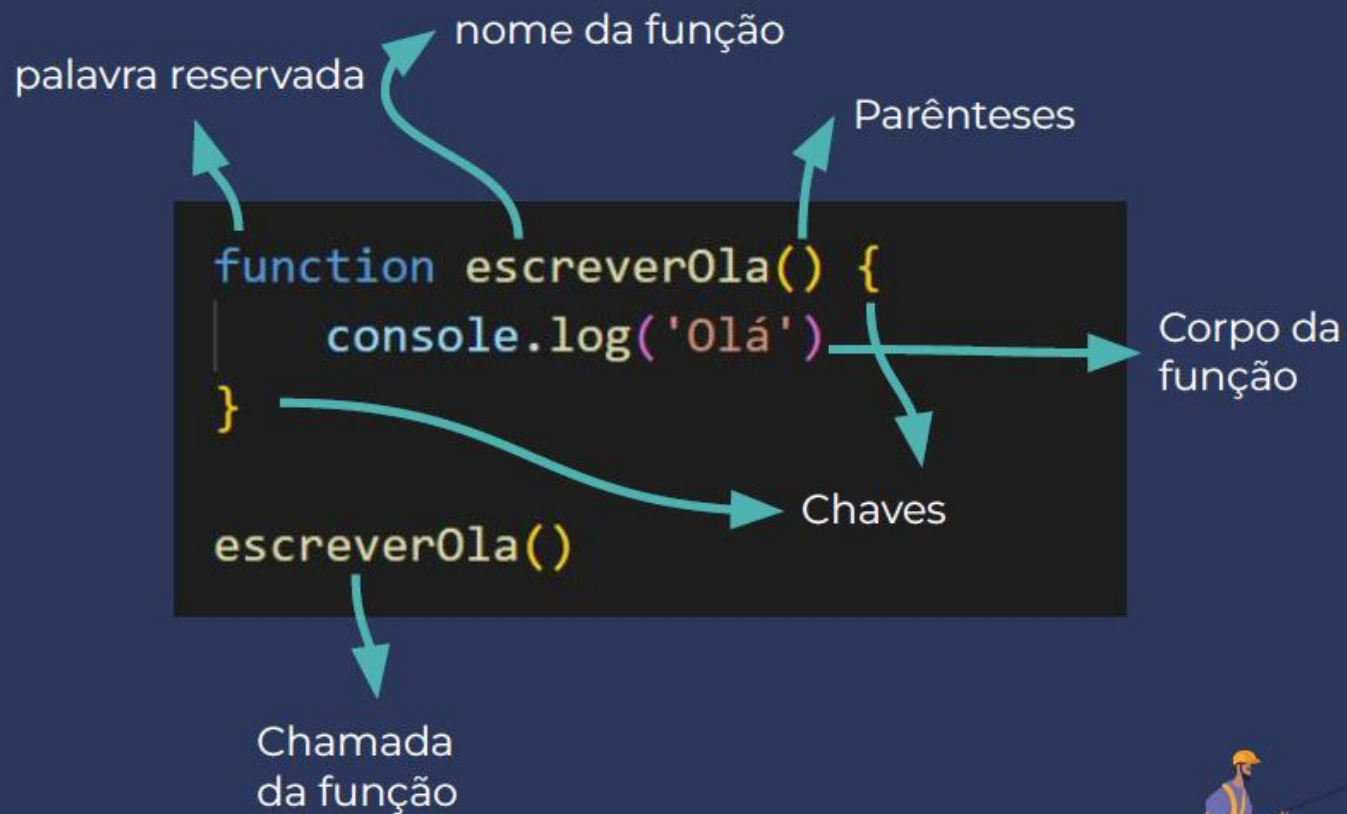
Tecnicamente

- São **ações** executadas assim que são **chamadas** ou em decorrência de algum **evento**
- Uma **função** pode receber **parâmetros** e retornar um **resultado**



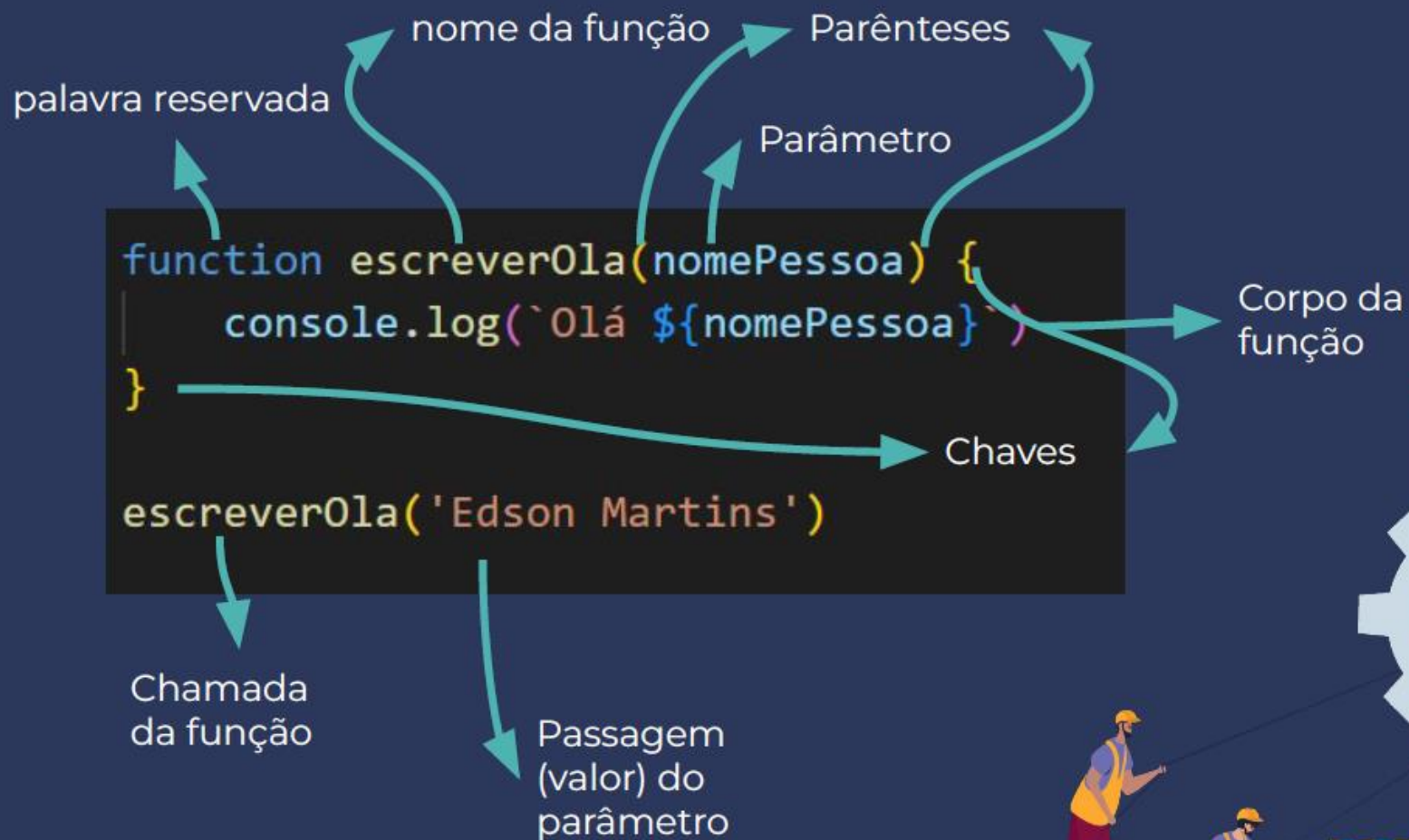
Funções

Sem parâmetros e sem retorno



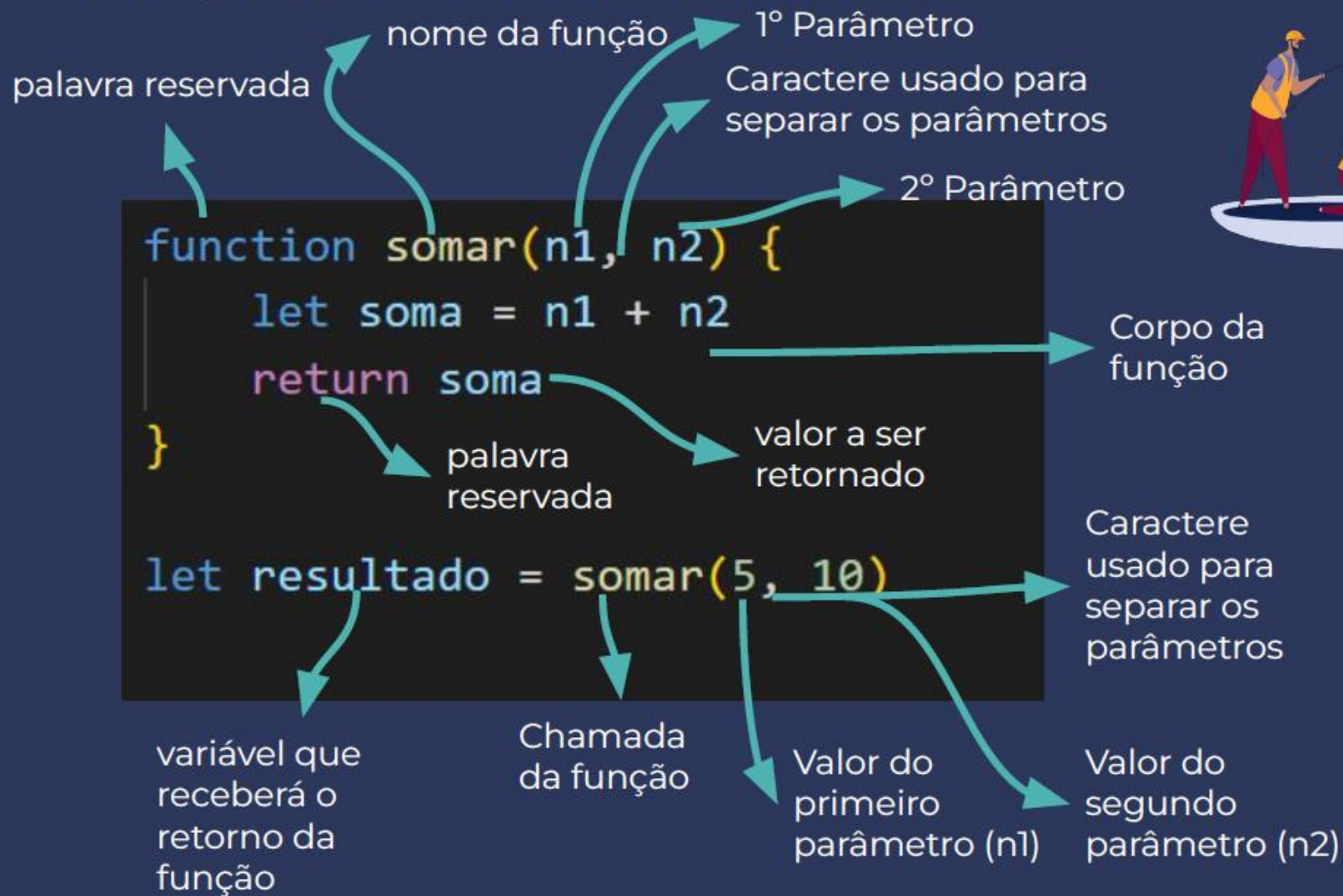
Funções

Com parâmetros e sem retorno



Funções

Com parâmetros e com retorno



Prática

7. Fatorial de um Número

Objetivo: Trabalhar com loops e operações matemáticas.

8. Conta Quantas Vogais Existem em uma String

Objetivo: Manipular strings e utilizar loops para contar caracteres específicos.

9. Verifica Palíndromo


Objetivo: Trabalhar com strings, manipulação de textos e lógica condicional.

10. Função que Retorna Números Primos em um Intervalo

Objetivo: Trabalhar com lógica avançada e loops.

JavaScript

Declaração de variáveis

A large yellow square containing the letters "JS" in a bold, black, sans-serif font, representing the JavaScript logo.

JS

JavaScript

Escopo de uma variável

Quando nos referimos a **escopo de variável** estamos nos referindo a qual **local** de nosso código uma determinada variável pode ser **acessada**.

No JavaScript existem 2 tipos de escopos:

- Local
- Global

```
1                                     Escopo global
2  var myGlobalVariable = 'My global variable.';
3
4                                     Escopo local
5  function showName() {
6      var myLocalVariable = 'My local variable.';
7
8      //Showing my global variable.
9      console.log(myGlobalVariable);
10
11     //Showing my local variable.
12     console.log(myLocalVariable);
13 }
14
15
16 //Showing my local variable.
17 console.log(myLocalVariable);
18
```


JavaScript

Escopo Global

Uma variável global é definida quando declaramos uma variável **fora de qualquer função**, assim ela torna acessível a qualquer parte da nossa aplicação ou site, podendo ser lida e alterada.

```
1                                     Escopo global
2  var myName = 'Yure';
3
4  function showName() {
5
6      var myLastName = 'Pereira';
7      console.log('My name is ' + myName);
8
9      //Modificando a variável global.
10     myName += ' ' + myLastName;
11
12 }
13
14 showName();
15 console.log('My name is ' + myName);
16
```


JavaScript

Escopo Local

Uma variável se torna **local** quando ela é declarada dentro de uma função, de tal maneira a qual ela **somente estará acessível dentro dessa função**.

```
1  var myName = 'Yure';
2                                     Escopo local
3
4  function showName() {
5
6      var myLastName = 'Pereira';
7      console.log('My name is ' + myName + ' ' + myLastName);
8
9  }
10
11 console.log('My name is ' + myName + ' ' + myLastName);
12 showName();
```

JavaScript

Escopo Local

Caso existam sub-funções dentro de uma função pai, todas as sub-funções também terão acesso a todas as variáveis declaradas dentro da função pai.

```
1
2  var myName = 'Yure';
3
4  function showName() {
5
6      var myLastName = 'Pereira';
7
8      function formatName() {
9          myLastName = myLastName.toUpperCase();
10      };
11
12      formatName();
13
14      console.log('My name is ' + myName + ' ' + myLastName);
15
16  }
17
18  showName();
19
```

Escopo local

Sub função

Obrigado

Desenvolvimento Web para Adolescentes
Pablo Garcia
Funções