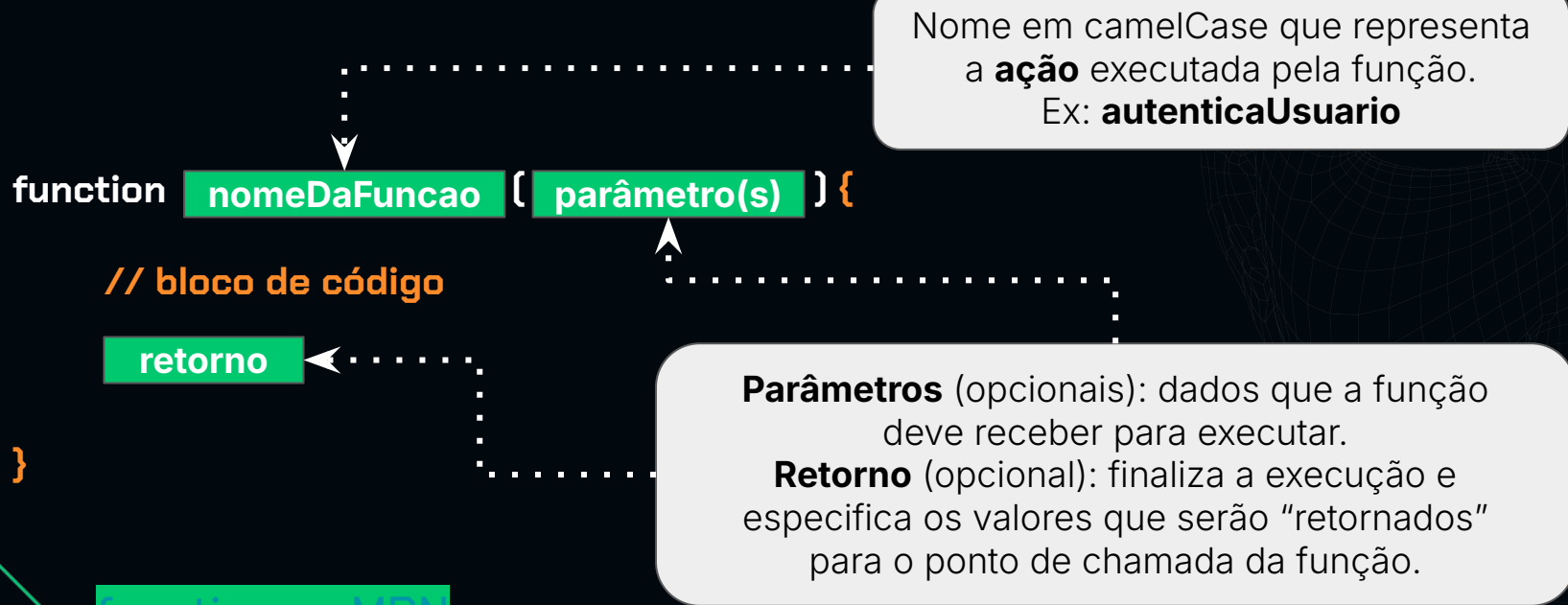


O que são FUNÇÕES?

Funções são **blocos de código reutilizáveis** que executam uma tarefa específica.



[function no MDN](#)

Sintaxe da Função

```
function darBoasVindas (nome) {  
  return `Boas vindas, ${nome}`;  
}
```

Função é
definida

```
darBoasVindas('Solange');  
console.log(darBoasVindas('Juliana'));
```

Função é
executada/chamada
recebendo os **argumentos**
necessários.

Valor é **retornado** para o ponto onde a função foi chamada e **a execução é encerrada**.
Ex: string concatenada com o valor do parâmetro.

Sintaxe da Função

```
function darBoasVindas() {  
  console.log('olá, boas vindas!');  
}
```

```
darBoasVindas();
```

Função não recebe
parâmetros

Função é
executada/chamada sem
receber **argumentos**.

Função não tem **retorno**, apenas
executa o que está no bloco {}.

Sintaxe da Função

```
function divide(dividendo, divisor) {  
    return dividendo / divisor;  
};
```

Parâmetros têm
identificadores

```
const resultado1 = divide(12, 2);  
const resultado2 = divide(2, 12);
```

Argumentos devem ser
passados para a função
na ordem correta

```
console.log(resultado1); //6  
console.log(resultado2); //0.166..
```

Ordem dos argumentos
influencia o resultado

Valores podem ser
retornados para variáveis

Sintaxe da Função

```
let resultado = 0;

function divide(dividendo, divisor = 2) {
  resultado = dividendo / divisor;
  const texto = `resultado é ${resultado}`;
};

divide(12);
console.log(resultado); //6
console.log(texto); //erro
```

Funções acessam
variáveis no **escopo
global**

É possível iniciar
parâmetros com **valores
pré-definidos**

Atenção com a
ordem dos argumentos

Outros escopos não
acessam variáveis dentro
do **escopo da função**

EXPRESSÃO DE FUNÇÃO

Expressões de função fazem parte de expressões ou instruções maiores, e podem ser **anônimas**.

`const nomeVariavel = function (parâmetro(s)) {`

`// bloco de código`

`retorno`

`}`

Expr. de função podem ser atribuídas a variáveis

Podem receber nomes em casos específicos.
Normalmente são anônimas.

Sintaxe da EXPRESSÃO DE FUNÇÃO

```
const imprimeOlaMundo = function() {  
  console.log(`olá, mundo!`);  
};
```

Expr. de função podem
ser atribuídas a variáveis
e ser **anônimas**

Sintaxe da EXPRESSÃO DE FUNÇÃO

```
const imprimeOlaMundo = function() {  
  console.log(`olá, mundo!`);  
};  
  
const boasVindas = function(nome) { return `olá, ${nome}` };
```

São opções mais
compactas do que
declarações de função

Sintaxe da EXPRESSÃO DE FUNÇÃO

```
const imprimeOlaMundo = function() {  
  console.log(`olá, mundo!`);  
};  
  
const boasVindas = function(nome) { return `olá, ${nome}` };  
  
const fatorial = function fatorial(num) {  
  if (num === 0 || num === 1) return 1;  
  return num * fatorial(num - 1);  
};
```

Blocos **if** também podem
ser encurtados
(apenas uma linha, sem {})

Podem receber nomes
em casos específicos
como **recursão**

Sintaxe da EXPRESSÃO DE FUNÇÃO

```
const imprimeOlaMundo = function() {  
  console.log(`olá, mundo!`);  
};  
  
const boasVindas = function(nome) { return `olá, ${nome}` };  
  
const fatorial = function fatorial(num) {  
  if (num === 0 || num === 1) return 1;  
  return num * fatorial(num - 1);  
};
```

```
imprimeOlaMundo() //olá, mundo!  
console.log(boasVindas('Ricardo')); //olá, Ricardo  
console.log(fatorial(5)); //120
```

ARROW FUNCTION (função seta)

Arrow functions têm sintaxe mais compacta e são sempre **anônimas**.
Têm uso mais restrito do que expressões e declarações de função.

```
.....  
↓  
const nomeVariavel = [ parâmetro(s) ] => {  
    // bloco de código  
    retorno  
}
```

Arrow function podem ser atribuídas a variáveis e são **anônimas**.
Não usam a palavra-chave **function** na declaração.

Usam a sintaxe de "seta"
var = parâmetro ⇒ {}

[arrow function no MDN](#)

Sintaxe da ARROW FUNCTION

```
const soma = (num1, num2) => {  
  console.log(num1 + num2);  
};
```

Arrow function podem ser atribuídas a variáveis e são **anônimas**.

Sintaxe da ARROW FUNCTION

```
const soma = (num1, num2) => {  
  console.log(num1 + num2);  
};
```

```
const boasVindas = nome => `olá, ${nome}`; <.....
```

Funções com **apenas uma linha de retorno** podem dispensar a instrução **return** e as chaves **{}**

Funções com **apenas um parâmetro** podem dispensar os parênteses.
Funções sem parâmetro devem declarar **()**

Sintaxe da ARROW FUNCTION

```
const soma = (num1, num2) => {  
  console.log(num1 + num2);  
};  
  
const boasVindas = nome => `olá, ${nome}`;  
  
const fatorial = (num) => {  
  if (num === 0 || num === 1) return 1;  
  return num * fatorial(num - 1);  
};
```

Caso tenham mais de uma linha de instruções, é necessária a instrução **return** e cercar o bloco com chaves {}

Sintaxe da ARROW FUNCTION


```
const soma = (num1, num2) => {  
  console.log(num1 + num2);  
};  
  
const boasVindas = nome => `olá, ${nome}`;  
  
const fatorial = (num) => {  
  if (num === 0 || num === 1) return 1;  
  return num * fatorial(num - 1);  
};
```

```
soma(2, 2); //olá, mundo!  
console.log(boasVindas('Aline')); //olá, Aline  
console.log(fatorial(10)); //3628800
```

FUNÇÕES CALLBACK

Funções callback são passadas como argumento de outra função, de onde podem receber valores. São executadas a partir da função externa.

```
setTimeout(function() {  
  console.log('olá, mundo');  
}, 2000);
```



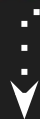
[setTimeout no MDN](#)

Uma **função anônima** é passada como **um dos argumentos** da função `setTimeout`

FUNÇÕES CALLBACK

Funções callback são passadas como argumento de outra função, de onde podem receber valores. São executadas a partir da função externa.

```
setTimeout(function() {  
  console.log('olá, mundo');  
}, 2000);
```



```
setTimeout(() => console.log('olá, mundo'), 2000);
```

O uso de **arrow functions** é um padrão comum em callbacks

FUNÇÕES CALLBACK

```
setTimeout(exibeFrase, 5000);
```

```
function exibefrase() {  
  console.log('olá, mundo');  
}
```

É possível desacoplar as funções, definindo callbacks separadas da função externa, chamada de **função de ordem superior**

[High Order Functions na Alura](#)

Compartilhe um resumo de seus novos
conhecimentos em suas redes sociais.

[#aprendizadoalura](#)

alura



Escola Programação