

# JAVASCRIPT PARA INICIANTEs

---

Funções

## Funções

---

Bloco de código que pode ser executado e reutilizado. Valores podem ser passados por uma função e a mesma retorna outro valor.

```
function areaQuadrado(lado) {  
  return lado * lado;  
}
```

```
areaQuadrado(4) // 16  
areaQuadrado(5) // 25  
areaQuadrado(2) // 4
```

*Chamada de function declaration*

## Funções

```
function pi() {  
  return 3.14;  
}  
  
var total = 5 * pi(); // 15.7
```

Parênteses `()` executam uma  
função

## Parâmetros e Argumentos

---

Ao **criar** uma função, você pode definir **parâmetros**.

Ao **executar** uma função, você pode passar **argumentos**.

```
// peso e altura são os parâmetros
function imc(peso, altura) {
  const imc = peso / (altura ** 2);
  return imc;
}

imc(80, 1.80) // 80 e 1.80 são os argumentos
imc(60, 1.70) // 60 e 1.70 são os argumentos
```

*Separar por vírgula cada  
parâmetro. Você pode definir  
mais de um parâmetro ou nenhum  
também*

## Parênteses executa a função

---

```
function corFavorita(cor) {  
  if(cor === 'azul') {  
    return 'Você gosta do céu';  
  } else if(cor === 'verde') {  
    return 'Você gosta de mato';  
  } else {  
    return 'Você não gosta de nada';  
  }  
}  
  
corFavorita(); // retorna 'Você não gosta de nada'
```

*Se apenas definirmos a função com o function e não executarmos a mesma, nada que estiver dentro dela irá acontecer*

## Argumentos podem ser funções

Chamadas de Callback, geralmente são funções que ocorrem após algum evento.

```
addEventListener('click', function() {  
  console.log('Clicou');  
});  
  
// A função possui dois argumentos  
// Primeiro é a string 'click'  
// Segundo é uma função anônima
```

*Funções anônimas são aquelas em que o nome da função não é definido, escritas como*

```
function() {} ou () => {}
```

## Pode ou não retornar um valor

---

Quando não definimos o return, ela irá retornar `undefined`. O código interno da função é executado normalmente, independente de existir valor de return ou não.

```
function imc(peso, altura) {  
  const imc = peso / (altura ** 2);  
  console.log(imc);  
}  
  
imc(80, 1.80); // retorna o imc  
console.log(imc(80, 1.80)); // retorna o imc e undefined
```

## Valores retornados

---

Uma função pode retornar qualquer tipo de dado e até outras funções.

```
function terceiraIdade(idade) {  
  if(typeof idade !== 'number') {  
    return 'Informe a sua idade!';  
  } else if(idade >= 60) {  
    return true;  
  } else {  
    return false;  
  }  
}
```

*Cuidado, retornar diferentes tipos de dados na mesma função não é uma boa ideia.*



## Escopo

---

Variáveis e funções definidas dentro de um bloco `{ }`, não são visíveis fora dele.

```
function precisoVisitar(paisesVisitados) {  
  var totalPaises = 193;  
  return `Ainda faltam ${totalPaises - paisesVisitados} paises pa  
}  
console.log(totalPaises); // erro, totalPaises não definido
```

## Escopo Léxico

Funções conseguem acessar variáveis que foram criadas no contexto `pai`

```
var profissao = 'Designer';

function dados() {
  var nome = 'André';
  var idade = 28;
  function outrosDados() {
    var endereco = 'Rio de Janeiro';
    var idade = 29;
    return `${nome}, ${idade}, ${endereco}, ${profissao}`;
  }
  return outrosDados();
}

dados(); // Retorna 'André, 29, Rio de Janeiro, Designer'
outrosDados(); // retorna um erro
```

## Hoisting

---

Antes de executar uma função, o JS 'move' todas as funções declaradas para a memória

```
imc(80, 1.80); // imc aparece no console
```

```
function imc(peso, altura) {  
  const imc = peso / (altura ** 2);  
  console.log(imc);  
}
```

## Exercício

---

```
// Crie uma função para verificar se um valor é Truthy

// Crie uma função matemática que retorne o perímetro de um quadrado
// lembrando: perímetro é a soma dos quatro lados do quadrado

// Crie uma função que retorne o seu nome completo
// ela deve possuir os parâmetros: nome e sobrenome

// Crie uma função que verifica se um número é par

// Crie uma função que retorne o tipo de
// dado do argumento passado nela (typeof)

// addEventListener é uma função nativa do JavaScript
// o primeiro parâmetro é o evento que ocorre e o segundo o Callback
// utilize essa função para mostrar no console o seu nome completo
// quando o evento de clique ocorrer
```

```
function precisoVisitar(paisesVisitados) {  
  var totalPaises = 193;  
  return `Ainda faltam ${totalPaises - paisesVisitados} países pa  
}  
function jaVisitei(paisesVisitados) {  
  return `Já visitei ${paisesVisitados} do total de ${totalPaises  
}  
precisoVisitar(20);  
jaVisitei(20);
```