

Funções

Bloco de código que pode ser executado e reutilizado.

Valores podem ser passados por uma função e a mesma retorna outro valor.

Function

```
1  function areaQuadrado(lado) {  
2      return lado * lado;  
3  }  
4  areaQuadrado(4) // 16  
5  areaQuadrado(5) // 25  
6  areaQuadrado(2) // 4
```

Função

```
1  function pi() {  
2      return 3.14;  
3  }  
4  
5  var total = 5 * pi(); // 15.7
```

Parâmetros e Argumentos

Parâmetros

```
1  // peso e altura são os parâmetros  
2  function imc(peso, altura) {  
3      const imc = peso / (altura ** 2);  
4      return imc;  
5  }  
6  
7  imc(80, 1.80) // 80 e 1.80 são os argumentos  
8  imc(60, 1.70) // 60 e 1.70 são os argumentos
```

Os parâmetros são separados por vírgulas

Fazer a chamada usando parênteses executa a função:

Executar função

```
1 function corFavorita(cor) {  
2   if(cor === 'azul') {  
3     return 'Você gosta do céu';  
4   } else if(cor === 'verde') {  
5     return 'Você gosta de mato';  
6   } else {  
7     return 'Você não gosta de nada';  
8   }  
9 }  
10 corFavorita(); // retorna 'Você não gosta de nada'
```

Argumentos podem ser funções

callbacks

```
1 addEventListener('click', function() {  
2   console.log('Clicou');  
3 });
```

A função possui dois argumentos:

- Primeiro é a string 'click'
- Segundo é uma função anônima

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

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

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

Pode ou não retornar um valor

```
1 function imc(peso, altura) {  
2   const imc = peso / (altura ** 2);  
3 }  
4
```

```
5   imc(80, 1.80); // retorna o imc
6   console.log(imc(80, 1.80)); // retorna undefined
```

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.

Valores retornados

```
1  function terceiraIdade(idade) {
2      if(typeof idade !== 'number') {
3          return 'Informe a sua idade!';
4      } else if(idade >= 60) {
5          return true;
6      } else {
7          return false;
8      }
9  }
```

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



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

Escopo

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

Escopo

```
1  function exerciciosResolvidos(resolvidos) {
2      var totalExercicios = 10;
3      return `Ainda faltam ${totalExercicios - resolvidos} exercicios para
4  resolver`;
5  }
6
   console.log(totalExercicios); // erro, totalExercicios não definido
```

Escopo Léxico

Funções conseguem acessar variáveis que foram criadas no contexto pai

Escopo

```
1  var profissao = 'Desenvolvedor';
2
3  function dados() {
4      var nome = 'Joãozinho';
5      var idade = 28;
6      function outrosDados() {
7          var endereco = 'Rio de Janeiro';
8          var idade = 29;
9          return `${nome}, ${idade}, ${endereco}, ${profissao}`;
10     }
11     return outrosDados();
12 }
13
14 dados(); // Retorna 'Joãozinho, 29, Rio de Janeiro, Desenvolvedor'
15 outrosDados(); // retorna um erro
```

Hoisting

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

Hoisting

```
1  imc(80, 1.80); // imc aparece no console
2  function imc(peso, altura) {
3      const imc = peso / (altura ** 2);
4      console.log(imc);
5  }
```

For Loop

O for loop possui 3 partes, **início**, **condição** e **incremento**

For

```
1  for (var numero = 0; numero < 10; numero++) {
2      console.log(numero);
3  }
4  // Retorna de 0 a 9 no console
```

While Loop

While

```
1  var i = 0;
2  while (i < 10) {
3      console.log(i);
4      i++;
5  }
6  // Retorna de 0 a 9 no console
```

Exercício

- Crie uma função para verificar se um valor é `True`
- 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 'scroll' ocorrer.

- Crie uma função e utilize as estruturas de controle e repetição que aprendemos para imprimir no console a letra da música *Um elefante incomoda*, utilize a função para verificar se o número é par neste desenvolvimento.

Um Elefante Incomoda - Galinha Pintadinha DVD 2

