



Aula 3: Loops

>

Funções



Objetivos



1. Ensinar como utilizar algumas das mais famosas declarações

>

do Javascript





5



If/else

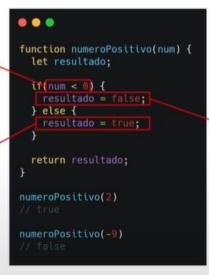


Condição

<

Declaração 2 (dentro de else):

Ocorre caso a condição seja falsa



Declaração 1 (dentro de if): Ocorre caso a condição seja verdadeira >



If/else



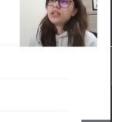
```
function numeroPositivo(num) {
 let resultado;
  if(num < 0) {
   resultado = false;
  } else {
  return resultado;
numeroPositivo(2)
```

```
function numeroPositivo(num) {
 let resultado;
 const ehNegativo = num < 0;</pre>
 resultado = false;
} else {
```

```
. . .
function numeroPositivo(num) {
  const ehNegativo = num < 0;</pre>
```

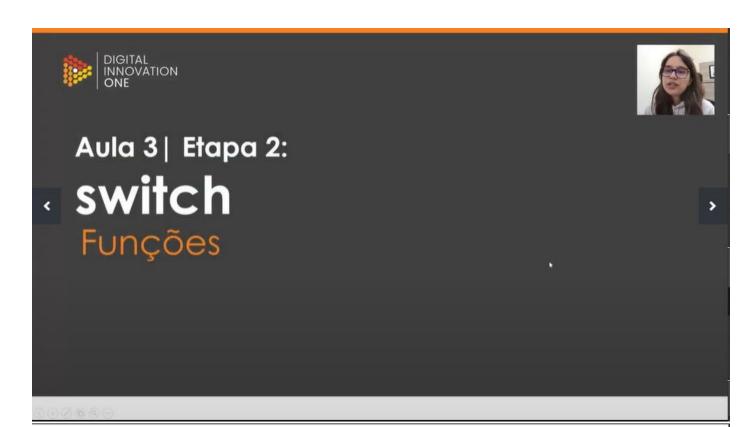


If/else



```
> numeroPositivo(2)
                                                 "Esse número é positivo!"
                                                 > numeroPositivo(-2)
function numeroPositivo(num) {
                                                 "Esse número é negativo!"
  const ehNegativo = num < 0;</pre>
  const maiorQueDez = num > 10;
                                                 > numeroPositivo(40)
                                                   "Esse número é positivo e maior que 10!"
  if(ehNegativo) {
    return "Esse número é negativo!";
  } else if (!ehNegativo && maiorQueDez) {
    return "Esse número é positivo e maior que 10!"
  return "Esse número é positivo!";
```

Javascript não tem elseif, as palavras sempre estão espaçadas!





Switch/case

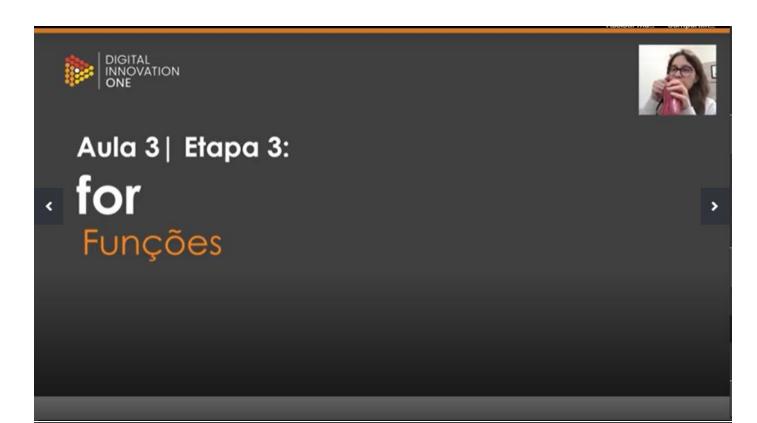


```
function getAnimal(id) {
   switch(id) {
    case 1:
       return "cão";
   case 2:
       return "gato";
   case 3:
       return "pássaro";
   default:
       return "peixe";
   }
}

getAnimal(1) // cão
   getAnimal(4) // peixe
   getAnimal("1") // peixe
```

- Equivale a uma comparação de tipo e valor (===)
- Sempre precisa de um valor "default"
- Ideal para quando se precisa comparar muitos valores

<





for



Loop dentro de elementos iteráveis (arrays, strings).

```
function multiplicaPorDois(arr) {
  let multiplicados = [];

  for(let i = 0; i < arr.length; i++) {
    multiplicados.push(arr[i] * 2);
  }

  return multiplicados;
}

const meusNumeros = [2, 33, 456, 356, 40];

multiplicaPorDois(meusNumeros);
// [4, 66, 912, 712, 80]</pre>
```

<

3



For...in



Loop entre propriedades enumeráveis de um objeto.

```
function forInExemplo(obj) {
  for(prop in obj) {
    console.log(prop);
  }
}

const meuObjeto = {
  nome: "João",
    idade: "20",
    cidade: "Salvador"
}

forInExemplo(meuObjeto);
// nome
// idade
// cidade
```

```
function forInExemplo(obj) {
   for(prop in obj) {
     console.log(obj[prop]);
   }
}

const meuObjeto = {
   nome: "João",
   idade: "20",
   cidade: "Salvador"
}

forInExemplo(meuObjeto);

// João
// 20
// Salvador
```



For...of



Loop entre estruturas iteráveis (arrays, strings).

```
function logLetras(palavra) {
  for(letra of palavra) {
    console.log(letra);
  }
}

const palavra = "abacaxi";

logLetras(palavra)
// a
// b
// a
// c
// a
// c
// a
// x
// i
```

```
function logNumeros(nums) {
  for(num of nums) {
    console.log(num);
  }
}

const nums = [30, 20, 233, 2];

logLetras(nums)
// 30
// 20
// 233
// 2
```

~





Aula 3 | Etapa 4:

While e do...while

Funções

000000



while



function exemploWhile() {
 let num = 0

 while(num <= 5){
 console.log(num);
 num++;
 }
}

exemploWhile()
// 0
// 1
// 2
// 3
// 4</pre>

Executa instruções até que a condição se torne falsa.

>



Do...while



```
<
```

```
function exemploDoWhile() {
  let num = 0;

  do {
    console.log(num);
    num++;
  } while(num <= 5)
}

exemploDoWhile()
// 0
// 1
// 2
// 3
// 4
// 5</pre>
```

```
function exemploDoWhile() {
  let num = 6;

  do {
    console.log(num);
    num++;
  } while(num <= 5)
}

exemploDoWhile()
// 6</pre>
```

Executa instruções até que a condição se torne falsa.

Porém a primeira execução sempre ocorre.