



iscte

INSTITUTO
UNIVERSITÁRIO
DE LISBOA



emprego
digital

Módulo 4: Introdução à programação em javascript

Aula 07

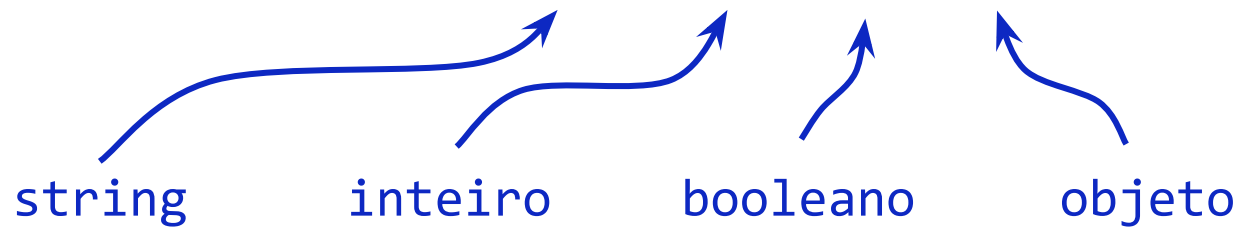
Callbacks e Exercícios



Lista de Objetos

- Nos exemplos anteriores foi utilizada uma lista de strings, mas na realidade nada obriga a lista a ser desse tipo, ou sequer a ter os elementos todos do mesmo tipo. Por exemplo:

```
let lista = ["texto", 1, true, {}];
```



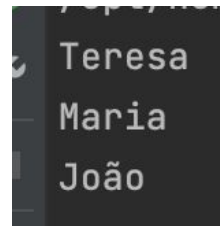
Lista de Objetos

```
let alunos = [  
  {  
    nome: "Teresa",  
    idade: 28  
  }, {  
    nome: "Maria",  
    idade: 23  
  }, {  
    nome: "João",  
    idade: 25  
  },  
];
```

- Um dos tipos mais comuns de estruturas são as listas de objetos.
- Com estruturas destas podemos ter uma quantidade indeterminada de informações agrupadas (nome e idade, por cada aluno)

Exercício 1

- Como poderíamos imprimir o nome dos alunos da lista que está do lado direito?



```
Teresa  
Maria  
João
```

Bónus: e se quisermos imprimir a lista pela ordem inversa?

```
let alunos = [  
  {  
    nome: "Teresa",  
    idade: 28  
  }, {  
    nome: "Maria",  
    idade: 23  
  }, {  
    nome: "João",  
    idade: 25  
  },  
];
```

Funções como argumento

- Funções de callback operam como variáveis normais **enviadas no argumento** de outra função, e são invocadas por essa função:

```
function imprimir(funcao) {  
    console.log(funcao());  
}
```

```
function teste1() {  
    return "Bom dia!";  
}  
imprimir(teste1);
```

```
function teste2() {  
    return "Hello World!";  
}  
imprimir(teste2);
```

Callbacks - Outro exemplo

```
function imprimirNvezes(n, funcao) {  
  for(let i = 0; i < n; i++) {  
    console.log(funcao(i));  
  }  
}
```

```
function teste1(num) {  
  return 2 ** num;  
}  
imprimirNvezes(10, teste1);
```

```
function teste2(num) {  
  return 3 ** num;  
}  
imprimirNvezes(10, teste2);
```

Outra forma de iterar nas listas


- O JavaScript disponibiliza-nos outras formas de interagir com os elementos das listas.
- O método **forEach**, por exemplo, vai chamar **uma função para cada elemento** de uma lista, e vai colocar o valor do item da lista no **primeiro parâmetro** dessa função.
- Essa função que o **forEach** executa é chamada de “*callback*”.

```
let alunos = [  
  {nome: "Teresa", idade: 28},  
  {nome: "Maria", idade: 23},  
  {nome: "João", idade: 25}  
];  
  
function imprimirNome(aluno) {  
  console.log(aluno.nome);  
}  
  
alunos.forEach(imprimirNome);
```

Funções anónimas

- Nem sempre é necessário declarar uma função isoladamente para o código conseguir chamar.
- Uma função anónima é uma função que **não tem nome**, e pode definida diretamente no *callback* do **forEach**, por exemplo.

```
let alunos = [  
  {nome: "Teresa", idade: 28},  
  {nome: "Maria", idade: 23},  
  {nome: "João", idade: 25}  
];  
  
alunos.forEach(function (aluno) {  
  console.log(aluno.nome);  
});
```



Exercício 3

Escreve um algoritmo que imprime o nome e a idade de cada aluno da lista à direita, utilizando o método `.forEach()`

```
Teresa 28  
Maria 23  
João 25
```

Bónus: como podemos imprimir apenas os alunos que têm mais de 24 anos?

```
let alunos = [  
  {  
    nome: "Teresa",  
    idade: 28  
  }, {  
    nome: "Maria",  
    idade: 23  
  }, {  
    nome: "João",  
    idade: 25  
  },  
];
```

.map()

- O método **.map()** itera numa lista e devolve uma lista nova, com o resultado devolvido pelo callback:

```
let alunos = [  
  {nome: "Teresa", idade: 28},  
  {nome: "Maria", idade: 23},  
  {nome: "João", idade: 25}  
];  
  
let nomes_alunos = alunos.map(function (aluno) {  
  return aluno.nome;  
});  
  
console.log(nomes_alunos);  
>> ["Teresa", "Maria", "João"]
```

.filter()

- O método **.filter()** devolve uma lista nova, com os elementos da lista originais que passaram numa função de validação:

```
let alunos = [
  {nome: "Teresa", idade: 28},
  {nome: "Maria", idade: 23},
  {nome: "João", idade: 25}
];

let alunos_24 = alunos.filter(function (aluno) {
  return aluno.idade > 24;
});

console.log(alunos_24);
>> [{nome: 'Teresa', idade: 28},
     {nome: 'João', idade: 25}]
```

Exercício 2

- Escreve um algoritmo que transforma a lista à direita numa lista de nomes completos dos alunos utilizando a função map:

```
['Teresa Andrade', 'Maria Ventura', 'João Pereira']
```

Bônus: Agora já sabendo como usar o .filter, imprime a lista acima mas apenas para quem tem mais de 24 anos

```
let alunos = [  
  {  
    nome: "Teresa",  
    apelido: "Andrade",  
    idade: 28  
  }, {  
    nome: "Maria",  
    apelido: "Ventura",  
    idade: 23  
  }, {  
    nome: "João",  
    apelido: "Pereira",  
    idade: 25  
  },  
];
```

O futuro profissional começa aqui

iscte

INSTITUTO
UNIVERSITÁRIO
DE LISBOA



emprego
digital



UPskill