Programação Front End – AULA 16

Matheus Moresco
Engenharia de Software – 3 Período
2025/01

Introdução

- Compreender como representar e manipular coleções de dados com arrays e objetos
- Praticar métodos como push, map, filter, forEach, etc.
- Entender como combinar arrays e objetos para representar dados mais complexos

Arrays e JavaScript

- Um **array** é uma **lista ordenada de valores**, onde cada valor tem uma **posição (índice)**. Ele pode armazenar qualquer tipo de dado: números, strings, objetos, funções e até outros arrays.

```
const frutas = ["maçã", "banana", "laranja"];
```

- Os arrays em JavaScript começam do índice 0.
- Você pode acessar elementos assim:

```
console.log(frutas[0]); // "maçã"
console.log(frutas[2]); // "laranja"
```

Adicionar e remover elementos

- Para adicionar ou remover elementos de um array podemos usar os seguintes métodos:

  - **Push**: Adiciona no final
  - **Pop**: remove o ultimo elemento
  - **Shift**: Adiciona no começo
  - **Unshift**: remove o primeiro elemento

```
const nomes = ["Ana", "João"];

nomes.push("Carlos");      // adiciona no final
nomes.unshift("Maria");    // adiciona no início

nomes.pop();               // remove o último
nomes.shift();             // remove o primeiro
```

Acessar e buscar

- Para acessar os elementos do array podemos usar o acesso direto pela posição, mas tambem podemos buscar elementos usando o método **indexOf**().

```
nomes.length              // tamanho do array
nomes.indexOf("João")     // retorna o índice (ou -1 se não encontrar)
nomes[0]                  // retorna o elemento da posição 0
```

Percorrer arrays

Temos duas maneira de percorrer um array usando JavaScript:

- Usando um **for** Tradicional:

```
for (let i = 0; i < nomes.length; i++) {
  console.log(nomes[i]);
}
```

- Usando um **forEach :**

```
nomes.forEach((nome, i) => {
  console.log(`Nome ${i}: ${nome}`);
});
```

Métodos dos arrays

- **.map()** - Cria um novo array com base no original.

```
const numeros = [1, 2, 3];
const dobro = numeros.map(n => n * 2);  // [2, 4, 6]
```

- **.filter()** - Cria um novo array com elementos que passam por um teste.

```
const maiores = numeros.filter(n => n > 1);  // [2, 3]
```

- **.find()** - Retorna o primeiro item que satisfaz uma condição.

```
const encontrado = numeros.find(n => n > 1); // 2
```

Métodos dos arrays

- **.includes()** - Verifica se um valor está presente.

```
numeros.includes(2); // true
```

- **.reduce()** - Reduz o array a um único valor (muito usado para somas).

```
const soma = numeros.reduce((acum, valor) => acum + valor, 0); // 6
```

Objetos em JavaScript

- Um **objeto** é uma coleção de **pares chave-valor**.
- Ele serve para **representar uma entidade** com características (propriedades) e comportamentos (métodos).

```
const pessoa = {
  nome: "João",
  idade: 25,
  profissao: "Desenvolvedor"
};
```

- Nesse exemplo:
  - "**nome**", "**idade**" e "**profissao**" são chaves (ou propriedades)
  - "**João**", **25**, "**Desenvolvedor**" são os valores associados

Acessando e alterando Objetos

Quando estamos trabalhando com objetos em JavaScript temos 2 maneiras principais de acessar os valores presentes dentro deste objetos:

- Notação de ponto

```
console.log(pessoa.nome); // João
pessoa.idade = 26;
```

- Notação de colchetes

```
console.log(pessoa["profissao"]); // Desenvolvedor
pessoa["profissao"] = "Engenheiro";
```

Objetos dentro de arrays

- Muito usado para representar coleções de dados estruturados, como listas de usuários, produtos, etc.
- Você pode combinar isso com métodos de array como .map(), .filter() etc.

```
const usuarios = [
  { nome: "Ana", idade: 20 },
  { nome: "Carlos", idade: 30 }
];
```

Exemplo prático

- Permitir **criar**, **listar**, **editar** e **remover** usuários de um array, manipulando o DOM dinamicamente.

Exemplo prático

- Usar o javaScript para cadastrar novos cursos na página de cursos

Exercício

1. Na página dos sites favoritos, permita que o usuário adicione comentários aos seus sites.
    1. No final da página crie
        1. Crie um input do tipo textarea para o comentário
        2. Crie um input de texto para o username
        3. Crie um botão de “Comentar”.
    2. Ao clicar no botão dispare uma ação de comentar
    3. Crie um elemento que lista os comentários no final da página.

*Não precisa implementar persistência de dados