Aula 13

find(), reduce(), some(), e every().

index.html × | find.js

index.html > html > body > script

```html
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Aula 9: Arrays - Busca, Agregação e Verificação </title>
    </head>
<body>
    <h1>Aula 9: Métodos de Arrays - Busca, Agregação e Verificação </h1>


    <script src="find.js"></script>
    <script src="reduce.js"></script>
    <script src="some.js"></script>
    <script src="every.js"></script>
</body>
</html>
```

index.html | find.js

find.js > ...

```js
// AULA 9 - TÓPICO 1: MÉTODO find() (Exemplo Super Simplificado)
// O método find() é usado para BUSCAR o PRIMEIRO elemento em um array
// que satisfaça uma condição específica.
// Se encontrar, retorna o elemento. Se não, retorna 'undefined'.
console.log("--- JS: aula9-1-find.js (Super Simplificado) ---");
console.log("1. Método find()");

// Nosso array de exemplo: uma lista de frutas.
const cestaDeFrutas = ["Maçã", "Banana", "Laranja", "Uva", "Manga"];
console.log("\nCesta de Frutas:", cestaDeFrutas);

// Queremos encontrar a fruta "Laranja" na cesta.
// A função passada para o .find() é chamada de "callback".
// Ela é executada para cada fruta na cesta, uma por uma.
// - 'frutaAtual' representa a fruta que o .find() está inspecionando no momento.
const frutaEncontrada = cestaDeFrutas.find(frutaAtual => {
    // A condição para encontrar: a frutaAtual deve ser igual a "Laranja".
    // Se a condição for verdadeira (true), o .find() para e retorna essa 'frutaAtual'.
    console.log(`  Verificando: ${frutaAtual}`); // Para vermos o processo
    return frutaAtual === "Laranja";
});

// Verificando o resultado:
if (frutaEncontrada) {
    console.log("Fruta encontrada na cesta:", frutaEncontrada); // Deve ser "Laranja"
} else {
    console.log("A fruta 'Laranja' não foi encontrada.");
}

// Exemplo de busca por uma fruta que não existe:
const frutaNaoExistente = cestaDeFrutas.find(frutaAtual => {
    return frutaAtual === "Pêra";
});
console.log("\nBuscando por 'Pêra':", frutaNaoExistente); // Deve ser 'undefined'
```

index.html | reduce.js | find.js

reduce.js > ...

```
// AULA 9 - TÓPICO 2: MÉTODO reduce() (Exemplo Super Simplificado)
// O método reduce() é poderoso! Ele "reduz" todos os elementos de um array
// a um ÚNICO valor final (como uma soma, um produto, um objeto, etc.).
// Ele faz isso aplicando uma função (o "redutor") que você define.
console.log("\n--- JS: aula9-2-reduce.js (Super Simplificado) ---");
console.log("2. Método reduce()");

// Nosso array de exemplo: uma lista de números para somar.
const listaDeGastos = [10, 25, 5, 30, 15]; // Valores em Reais, por exemplo
console.log("\nLista de Gastos (R$):", listaDeGastos);

// Queremos somar todos os gastos para obter o total.
// A função passada para o .reduce() é o nosso "redutor".
// Ela recebe dois argumentos principais:
// 1. 'acumulador': O valor acumulado das operações anteriores. Pense nele como o "subtotal".
// 2. 'gastoAtual': O elemento atual do array que está sendo processado.

// O '0' no final é o VALOR INICIAL do 'acumulador'.
// Na primeira vez, 'acumulador' será 0, e 'gastoAtual' será o primeiro item (10).
const totalDosGastos = listaDeGastos.reduce((acumulador, gastoAtual) => {
    console.log(`  Subtotal (acumulador): R$${acumulador}, Gasto Atual: R$${gastoAtual}`);
    // A função deve RETORNAR o novo valor do acumulador para a próxima rodada.
    // Aqui, estamos somando o gasto atual ao subtotal.
    return acumulador + gastoAtual;
}, 0); // O '0' é crucial aqui para iniciar a soma corretamente.

console.log("Total dos gastos calculado com reduce: R$", totalDosGastos); // Deve ser 85
```

index.html | some.js | reduce.js | find.js

some.js > ...

```js
// AULA 9 - TÓPICO 3: MÉTODO some() (Exemplo Super Simplificado)
// O método some() verifica se PELO MENOS UM elemento do array
// satisfaz uma determinada condição.
// Retorna 'true' se encontrar algum, e 'false' caso contrário.
// Ele para de verificar assim que encontra o primeiro elemento que satisfaz.
console.log("\n--- JS: aula9-3-some.js (Super Simplificado) ---");
console.log("3. Método some()");

// Nosso array de exemplo: idades de um grupo de pessoas.
const idadesDoGrupo = [15, 22, 17, 30, 12];
console.log("\nIdades do Grupo:", idadesDoGrupo);

// Queremos saber se há ALGUMA pessoa maior de idade (>= 18) no grupo.
// A função callback é executada para cada idade.
// - 'idadeAtual' é a idade sendo verificada.
const existeAlguemMaiorDeIdade = idadesDoGrupo.some(idadeAtual => {
    console.log(`  Verificando idade (some): ${idadeAtual}`); // Para vermos o processo
    // Se a condição (idadeAtual >= 18) for verdadeira para QUALQUER idade,
    // o .some() para e retorna 'true'.
    return idadeAtual >= 18;
});

if (existeAlguemMaiorDeIdade) {
    console.log("Sim, existe pelo menos uma pessoa maior de idade no grupo."); // Deve ser true
} else {
    console.log("Não, ninguém é maior de idade no grupo.");
}

// Exemplo onde a condição não é satisfeita por ninguém:
const todasMenoresQueDez = idadesDoGrupo.some(idadeAtual => idadeAtual < 10);
console.log("\nAlguma idade é menor que 10?", todasMenoresQueDez); // Deve ser false
```

index.html | every.js | some.js | reduce.js | find.js

every.js > ...

```js
// AULA 9 - TÓPICO 4: MÉTODO every() (Exemplo Super Simplificado)
// O método every() verifica se TODOS os elementos do array
// satisfazem uma determinada condição.
// Retorna 'true' apenas se TODOS satisfizerem, e 'false' caso contrário.
// Ele para de verificar assim que encontra o primeiro elemento que NÃO satisfaz.
console.log("\n--- JS: aula9-4-every.js (Super Simplificado) ---");
console.log("4. Método every()");

// Nosso array de exemplo: notas de alunos em uma prova.
const notasDaTurma = [7, 8, 9, 10, 6];
console.log("\nNotas da Turma:", notasDaTurma);

// Queremos saber se TODOS os alunos foram aprovados (nota >= 7).
// A função callback é executada para cada nota.
// - 'notaAtual' é a nota sendo verificada.
const todosForamAprovados = notasDaTurma.every(notaAtual => {
    console.log(`  Verificando nota (every): ${notaAtual}`); // Para vermos o processo
    // Se a condição (notaAtual >= 7) for falsa para QUALQUER nota,
    // o .every() para e retorna 'false'.
    return notaAtual >= 7;
});

if (todosForamAprovados) {
    console.log("Parabéns! Todos os alunos foram aprovados.");
} else {
    console.log("Atenção! Pelo menos um aluno não atingiu a nota para aprovação."); // Deve
}

// Exemplo onde todos satisfazem:
const numerosPositivos = [1, 10, 100, 5];
const todosSaoPositivos = numerosPositivos.every(num => num > 0);
console.log("\nTodos os números são positivos?", todosSaoPositivos); // Deve ser true

console.log("\nFim dos exemplos super simplificados da Aula 9!");
```

Atividade

Exercício 1: Procurando uma Cor 🎨

- **Arquivo:** `aula09/js/exercicios/exercicio1-procurando-cor.js`
- **Enunciado:**
    1. Crie um array chamado `listaDeCores` com nomes de cores (strings). Adicione pelo menos 5 cores, incluindo "Vermelho".
    2. Utilize o método `find()` para encontrar a cor "Vermelho" na `listaDeCores`.
    3. Exiba no console: "A cor 'Vermelho' foi encontrada!" se ela existir, ou "A cor 'Vermelho' não está na lista." caso contrário.
    4. Teste também buscando por uma cor que não está na lista.

Exercício 2: Somando as Compras 🛒

- **Arquivo:** `aula09/js/exercicios/exercicio2-somando-compras.js`
- **Enunciado:**
    1. Crie um array chamado `valoresDasCompras` contendo vários números que representam os preços de itens comprados.
    2. Utilize o método `reduce()` para calcular o **valor total** da compra (a soma de todos os preços).
    3. Exiba no console a mensagem: "O valor total das compras é R$ [Soma Total]."

Exercício 3: Algum Número Ímpar? 🔢

- **Arquivo:** `aula09/js/exercicios/exercicio3-numero-impar.js`
- **Enunciado:**

    1. Crie um array chamado `sequenciaNumerica` com diversos números inteiros.
    2. Utilize o método `some()` para verificar se **pelo menos um** número na `sequenciaNumerica` é ímpar.
    3. Exiba no console: "Sim, existe pelo menos um número ímpar na sequência." se a condição for verdadeira, ou "Não, todos os números na sequência são pares." se for falsa.

Exercício 4: Todas as Palavras com 'A'? 🅰️

- **Arquivo:** `aula09/js/exercicios/exercicio4-palavras-com-a.js`
- **Enunciado:**
  1. Crie um array chamado `listaDePalavras` com várias palavras (strings).
  2. Utilize o método `every()` para verificar se **todas** as palavras na `listaDePalavras` contêm a letra "a" (maiúscula ou minúscula).
     - Dica: Dentro do callback do `every()`, você pode converter a palavra para minúsculas usando `.toLowerCase()` antes de verificar com `.includes('a')`.
  3. Exiba no console: "Incrível! Todas as palavras contêm a letra 'a'." se a condição for verdadeira, ou "Ops! Nem todas as palavras contêm a letra 'a'." se for falsa.

Adapte o exercício 1 para receber 5 cores de um formulário HTML (DOM)