07

Dominando o reduce

Você já deve ter ouvido falar em somatórios. O somatório de uma lista de números é a soma de todos os números daquela lista, como por exemplo:

```
let numeros = [1, 2, 3, 4]; // Somatório = 1 + 2 + 3 + 4 = 10
```

Um exemplo de função que nos retorne o somatório de um array de números poderia ser assim:

```
function somatorio(array) {

    var resultado = 0;
    for(let i = 0; i < array.length; i++){
        resultado = array[i] + resultado;
    }

    return resultado;
}
```

A mesma coisa usando `forEach` :

```
function somatorio(array) {
    let resultado = 0;
    array.forEach(item => resultado+= item);
    return resultado;
}
```

Existe um outro conceito matemático conhecido como **produtório**, que é análogo ao somatório, só que ao invés de somarmos os números , nós os multiplicamos. Por exemplo:

```
var numeros = [1, 2, 3, 4]; // Produtório = 1 * 2 * 3 * 4 = 24
```

Juntando este seu novo conhecimento matemático com o conhecimento de JavaScript adquirido neste capítulo, qual das funções abaixo nos retorna o *produtório* de um array de números corretamente, **usando a função `reduce` ?**

A

```
let numeros = [1, 2, 3, 4];
let resultado = numeros.reduce((total, num) {
    return num * num;
}, 1);
```

B

```
let numeros = [1, 2, 3, 4];
let resultado = numeros.reduce((total, num) {
```

```
    return total * (total * num);
}, 1);
```

```
let numeros = [1, 2, 3, 4];
let resultado = numeros.reduce(function(total, num) {
    return total * num;
}, 1);
```

Isso aí! Essa é a opção correta!

D

```
let numeros = [1, 2, 3, 4];
let resultado = numeros.reduce(function(total, num) {
    return total * num;
}, 0);
```

A função `reduce` recebe **dois** parâmetros: **uma função** e **um valor inicial**. Na função interna ao `reduce`, o primeiro parâmetro é o valor da última iteração, que neste caso é o `total`. O segundo parâmetro é o valor da iteração atual, neste caso o novo número que queremos multiplicar.

O `reduce` executa sua função interna a cada iteração, então no nosso caso ele multiplica o valor anterior (`total`) pelo valor da iteração atual (`num`). Como o *produtório* é a multiplicação de uma sequência de números, no nosso caso o que está acontecendo é o seguinte:

Supondo o array:

```
var numeros = [1, 2, 3, 4];
```

O `total` se inicia com o valor **1*, *definido pelo segundo parâmetro da função `reduce`.**

É feita a primeira iteração, pegando o primeiro valor do array (`1`) :

```
return total * num; // Leia-se: return 1 * 1 e coloque este valor em total.
```

Na segunda iteração, com o segundo valor do array (`2`):

```
return total * num; // Leia-se return 1 * 2 e coloque este valor em total, que agora vale 2;
```

Na terceira iteração, com o segundo valor do array (`3`):

```
return total * num; // Leia-se return 2 * 3 e coloque este valor em total, que agora vale 6;
```

Na segunda iteração, com o segundo valor do array (`4`):

```
return total * num; // Leia-se return 6 * 4 e coloque este valor em total, que agora vale 24;
```

E no final ele devolve para nós o valor `24`, que é o valor esperado do *produtório*!

PRÓXIMA ATIVIDADE