



91%

ATIVIDADES
5 DE 8DISCORD
ALURAFÓRUM DO
CURSOVOLTAR
PARA
DASHBOARDCONHEÇA O
VITRINE.DEVMODOS
NOTURNO

Guia do Mochileiro JavaScript

Método .concat()

```
[ 🏀 , 🏀 , 🏀 ].concat( [ 🏈 , 🏈 ] ) => [ 🏀 , 🏀 , 🏀 , 🏈 , 🏈 ]
```

O método `concat` junta dois (ou mais) arrays em um novo array, sem alterar os já existentes.

Método .pop()

```
[ 🏀 , 🏀 , 🏀 , 🏀 , 🏈 ].pop( ) => [ 🏀 , 🏀 , 🏀 , 🏀 ]
```

O método `pop` remove o último elemento de um array.

Método .push()



59.3k xp



91%



ATIVIDADES
5 DE 8



DISCORD
ALURA



FÓRUM DO
CURSO



VOLTAR
PARA
DASHBOARD



CONHEÇA O
VITRINE.DEV

```
[🏀, 🏀, 🏀].push(🏈) => [🏀, 🏀, 🏀, 🏈]
```

O método `push` adiciona um novo elemento no final do array, aumentando seu tamanho.

Método `.includes()`

```
[🏈, 🏈, ⚽, 🏀].includes(⚽) => true
```

O método `includes` verifica se um elemento está contido em um array e retorna `true` se o elemento estiver contido ou `false` caso contrário.

Método `.fill()`

```
[🏈, 🏈, ⚽, 🏀].fill(⚽, 1) => [🏈, ⚽, ⚽, 🏀] [🏈, 🏈, ⚽, 🏀].fill(⚽) => [⚽, ⚽, ⚽, ⚽]
```

O método `fill` preenche os elementos especificados em um array com um determinado valor.



59.3k xp



91%



ATIVIDADES
5 DE 8



DISCORD
ALURA



FÓRUM DO
CURSO



VOLTAR
PARA
DASHBOARD



CONHEÇA O
VITRINE.DEV

Método .indexOf()

```
[ 🏈 , 🏉 , ⚽ , 🏀 , ⚾ ].indexOf( ⚽ ) => 2
```

O método `indexOf` retorna o primeiro índice encontrado de um valor especificado. Se o valor não for encontrado o método retorna -1.

Método .reverse()

```
[ 🏈 , 🏉 , ⚽ , 🏀 ].reverse( ) => [ 🏀 , ⚽ , 🏉 , 🏈 ]
```

O método `reverse` inverte a ordem dos elementos de um array e substitui o array original.

Método .slice()

```
[ 🏈 , 🏉 , ⚽ , 🏀 , ⚾ ].slice( 1, 3 ) => [ 🏉 , ⚽ ]
```

O método `slice` retorna elementos de um array, selecionados de determinada posição de início até determinada posição final. O elemento na posição final não é incluso.



59.3k xp



91%



ATIVIDADES
5 DE 8



DISCORD
ALURA



FÓRUM DO
CURSO










VOLTAR
PARA
DASHBOARD

































CONHEÇA O
VITRINE.DEV

Método .some()

```
[  ,  ,  ,  ,  ].some( (  ) => { return bola ===  } ) => true
```

O método `some` verifica se algum elemento do array passa em um teste. Esse teste é feito através de uma função callback. O método executa a função de callback para cada elemento uma vez e retorna `true` se o teste for `true` para um dos elementos, e `false` se o teste for `false` para todos os elementos. Além disso, o método não executa a função callback para arrays vazios e não altera o array.

Método .join()

```
[  ,  ,  ,  ,  ].join() =>  ,  ,  ,  ,  [  ,  ,  ,  ,  ].join(' ') =>      [  ,  ,  ,  ,  ].join('-') =>  -  -  -  - 
```

O método `join` puxa elementos de um array e lista no formato de string, o resultado da operação puxou as propriedades do array e as listou de acordo com o que foi determinado.



59.3k xp



91%



ATIVIDADES
5 DE 8



DISCORD
ALURA



FÓRUM DO
CURSO



VOLTAR
PARA
DASHBOARD



CONHEÇA O
VITRINE.DEV

Método .shift()

```
[🏈, 🏐, 🏐, 🏐, 🏐].shift( ) ⇒ [🏐, 🏐, 🏐, 🏐]
```

O método `shift` é parecido com o método `.pop()` mas ao invés de remover o último elemento do array, ele é usado para remover o primeiro elemento do array.

Método .unshift()

```
[🏐, 🏐, 🏐].unshift(🏐) ⇒ [🏐, 🏐, 🏐, 🏐]
```

O método `unshift` é parecido com o que método `.push()` realiza, mas ao invés de adicionar no final do array, ele é utilizado para adicionar um elemento no início de um array.

Método .splice()

```
[🏈, 🏐, 🏈, 🏐, 🏐].splice( 1, 2, 8 ) ⇒ [🏈, 8, 🏐,  
🏐 ] [🏈, 🏐, 🏈, 🏐, 🏐].splice( 2, 3) ⇒ [🏈, 🏐 ]
```



59.3k xp



91%



ATIVIDADES
5 DE 8



DISCORD
ALURA



FÓRUM DO
CURSO



VOLTAR
PARA
DASHBOARD



CONHEÇA O
VITRINE.DEV

Com o método `splice` conseguimos escolher um índice inicial e final para substituírmos valores no lugar deles. E também podemos remover itens, no segundo exemplo, foram removidos três elementos a partir da posição dois.

Método `.length()`

```
[ [ 🏏 , 🏈 , ⚽ , 🏀 , 🏐 ] .length(3) ⇒ [ 🏏 , 🏈 , ⚽ ] [ 🏏 , 🏈 , ⚽ , 🏀 , 🏐 ] .length() ⇒ 5
```

O método `length` define ou retorna o número de elementos em um array.

Método `.sort()`

```
[ 🏏 , 🏈 , ⚽ , 🏀 , 🏐 ] .sort() ⇒ ⚽ , 🏏 , 🏀 , 🏈 , 🏐
```

O método `sort` ordena os elementos do próprio array e retorna o array. A ordenação padrão é de acordo com a pontuação de código unicode.



59.3k xp



91%



ATIVIDADES
5 DE 8



DISCORD
ALURA



FÓRUM DO
CURSO



VOLTAR
PARA
DASHBOARD



CONHEÇA O
VITRINE.DEV

Método .toString()

```
[ 🏈 , 🏀 , ⚽ , 🏆 , 🏐 ].toString ⇒ 🏈,🏀,⚽,🏆,🏐
```

O método `toString` retorna uma string com todos os valores do array separados por vírgulas.

Método .findIndex()

```
[ 🏈 , 🏀 , ⚽ , 🏆 , 🏐 ].findIndex(emoji => emoji === "⚽") ⇒  
2
```

O método `findIndex` retorna o índice no array do primeiro elemento que satisfizer a função de teste provida. Caso contrário, retorna -1, indicando que nenhum elemento passou no teste.

Método .find()

```
[ 🏈 , 🏀 , ⚽ , 🏆 , 🏐 ].find( bola => bola.formato ===  
"redonda" ) => 🏈
```



59.3k xp



91%



ATIVIDADES
5 DE 8



DISCORD
ALURA



FÓRUM DO
CURSO



VOLTAR
PARA
DASHBOARD



CONHEÇA O
VITRINE.DEV

O método `find` é utilizado para procurar um elemento dentro do array que atenda a condição atribuída a ele, que retornará o primeiro elemento encontrado. Ele percorre todo o array buscando o elemento que atenda a condição e retorna o primeiro que foi encontrado, caso não encontre retorna undefined. Este método não executa caso o array esteja vazio e não altera o array original

Método `.at()`

```
[ [ 🏏 , 🏈 , ⚽ , 🏀 , 🏐 ] .at(-1) ⇒ 🏐 / [ 🏏 , 🏈 , ⚽ , 🏀 ,  
🏐 ] .at(1) ⇒ 🏈
```

O `at` acessar os elementos de um array usando um índice inteiro positivo e negativo, sendo que o índice negativo enumera os itens de trás para frente, portando o 🏐 está posicionado no índice -1.

Método `.isArray()`

```
var bolas = [ 🏏 , 🏈 , ⚽ , 🏀 , 🏐 ] Array.isArray(bolas) ⇒  
true
```




59.3k xp



91%



ATIVIDADES
5 DE 8



DISCORD
ALURA



FÓRUM DO
CURSO



VOLTAR
PARA
DASHBOARD



CONHEÇA O
VITRINE.DEV

O método `Array.isArray` verifica se a variável é um array e retorna `true` ou `false`.

Método `.every()`

```
[ 🏈 , 🏀 , ⚽ , 🏆 , 🏐 ].every( ( bola ) => { return bola ===  
🏐 } ) => false
```

O método `every` verifica se cada elemento do array passa em um teste. Esse teste é feito através de uma função callback. O método executa a função de callback para cada elemento uma vez e retorna `true` se o teste for `true` para todos os elementos, e `false` se o teste for `false` para pelo menos um elemento. Além disso, o método não executa a função callback para arrays vazios e não altera o array.

Método `.filter()`

```
[ 🏈 , 🏀 , ⚽ , 🏆 , 🏐 ].filter( ( bola ) => { return bola !==  
🏀 } ) => [ 🏈 , ⚽ , 🏆 , 🏐 ]
```

O método `filter` usa uma função callback de teste e executa ela para cada elemento do array. Ele retorna um novo array com os



59.3k xp



91%



ATIVIDADES
5 DE 8



DISCORD
ALURA



FÓRUM DO
CURSO



VOLTAR
PARA
DASHBOARD



CONHEÇA O
VITRINE.DEV

elementos que passarem no teste.

Método .map()

```
[🏀, 🏀, 🏀, 🏀].map( console.log(( bola ) => { return 🏈 }) )  
=> [ 🏈, 🏈, 🏈, 🏈 ]
```

O método `map` usa uma função callback de teste e executa ela para cada elemento do array, retornando um novo array modificado. Não executa a função de callback para arrays vazios.

Método .forEach()

```
[🏀, 🏀, 🏀, 🏀].forEach( ( bola ) => { console.log( bola +  
🏈) } ) => 🏀🏈 🏀🏈 🏀🏈 🏀🏈
```

O método `forEach` usa uma função callback e executa ela para cada elemento do array. Não executa a função de callback para arrays vazios e não retorna um novo array, diferente do método `map`.

Método .copyWithin()



59.3k xp



91%



ATIVIDADES
5 DE 8



DISCORD
ALURA



FÓRUM DO
CURSO



VOLTAR
PARA
DASHBOARD



CONHEÇA O
VITRINE.DEV

```
[ 🏈 , 🏀 , ⚽ , 🏆 ].copyWithin(2,0) ⇒ [ 🏈 , 🏀 , 🏈 , 🏀 ]
```

O método `copyWithin` copia os elementos do array para outra posição no array, ele não adiciona itens apenas substitui os valores existentes.

Método `.lastIndexOf()`

```
[ 🏈 , 🏀 , ⚽ , 🏆 , 🏐 ].lastIndexOf(⚽) ⇒ 2
```

O método `lastIndexOf` retorna o valor do último índice especificado, se o valor não for encontrado ele retorna -1.

Método `.valueOf()`

```
[ ⚽ , 🏈 , 🏆 , 🏐 ].valueOf() ⇒ [ ⚽ , 🏈 , 🏆 , 🏐 ]
```

O método `valueOf` é usado para retornar o array. É um método padrão do objeto Array. Este método retorna todos os itens na mesma matriz. Ele não altera o conteúdo original da matriz e não contém nenhum valor de parâmetro.



59.3k xp



91%



ATIVIDADES
5 DE 8



DISCORD
ALURA



FÓRUM DO
CURSO








VOLTAR
PARA
DASHBOARD








CONHEÇA O
VITRINE.DEV

Método .reduce()

```
[, , , ].reduce((valor, elemento) => valor + elemento, 0)) => 
```

O método `reduce` executa uma função redutora para o elemento array, ele retorna um único valor que é o resultado acumulado da função. Ele não executa a função para elementos de array vazios e não altera a matriz original.

Método .keys()

```
[, , , , ].keys() => ["0", "1", "2", "3"]
```

O método `keys` retorna um novo array composto pelas chaves (posições) do array o qual ele foi aplicado.