



# Criação e manipulação de arrays.

## Introdução

Em JavaScript, o array é uma única variável que é usada para armazenar diferentes elementos. É frequentemente usado quando queremos armazenar uma lista de elementos e acessá-los por uma única variável. Ao contrário da maioria das linguagens em que o array é uma referência a várias variáveis, no JavaScript array é uma única variável que armazena vários elementos.

## Objetivos da aula

- Definir os conceitos de Arrays.
- Reconhecer Arrays e seus principais métodos na linguagem Javascript
- Demonstrar a implementação de Arrays na linguagem Javascript.

## Resumo

O objeto Array permite armazenar vários valores em uma única variável. Ele armazena uma coleção sequencial de tamanho fixo de elementos do mesmo tipo. Uma matriz é usada para armazenar uma coleção de dados, mas geralmente é mais útil pensar em uma matriz como uma coleção de variáveis do mesmo tipo.

## Sintaxe:

Use a seguinte sintaxe para criar um objeto Array:



```
1 var frutas = new Array ("maçã", "laranja", "manga");
```

Fonte: Autor

O parâmetro Array é uma lista de strings ou inteiros. Ao especificar um único parâmetro numérico com o construtor Array, você especifica o comprimento inicial da matriz. O comprimento máximo permitido para uma matriz é 4.294.967.295.

Você pode criar uma matriz simplesmente atribuindo os valores:

```
3 var frutas = ["maçã", "laranja", "manga"];
```

Fonte: Autor

Você usará números ordinais para acessar e definir valores dentro de uma matriz da seguinte maneira:

```
1  frutas [0] é o primeiro elemento
2  frutas [1] é o segundo elemento
3  frutas [2] é o terceiro elemento
```

Fonte: Autor

## Propriedades da matriz

Aqui está uma lista das propriedades do objeto Array junto com sua descrição:



Propriedade	Descrição
constructor (construtor)	Retorna uma referência à função de matriz que criou o objeto.
index	A propriedade representa o índice baseado em zero da correspondência na string.
input	Esta propriedade está presente apenas em matrizes criadas por correspondências de expressões regulares.
length	Reflete o número de elementos em uma matriz.
prototype	A propriedade prototype permite adicionar propriedades e métodos a um objeto.

Fonte: Autor

### length: propriedade de comprimento de matriz

A propriedade JavaScript array length retorna um inteiro sem sinal de 32 bits que especifica o número de elementos em um array.

#### Sintaxe:

1      array.length

Fonte: Autor

Retorna o comprimento do array.

```
1  var arr = new Array( 10, 20, 30 );  
2  document.write("arr.length is : " + arr.length);  
3  
4  Retorno:  
5  arr.length é: 3
```



Fonte: Autor

## Propriedade do construtor de Array

A propriedade do construtor de array JavaScript retorna uma referência à função de array que criou o protótipo da instância.

### Sintaxe:

1 `array.constructor`

Valor de retorno: retorna à função que criou a instância deste objeto.

```
3  var arr = new Array (10, 20, 30);  
4  document.write ("arr.constructor é:" + arr.constructor);
```

Fonte: Autor

Retorno:

2 `arr.constructor é: function Array () {[código nativo]}`

Fonte: Autor

## Operações básicas em matrizes



### Adicionar um elemento ao final de uma matriz

Para adicionar um elemento ao final de uma matriz, você usa o método `push ()`:

```
2 let mares = ['Mar Negro', 'Mar do Caribe', 'Mar do Norte', 'Mar Báltico'];
3 mares.push ('Mar Vermelho');
4
5 console.log (mares);
6
7 Saída:
8 ['Mar Negro', 'Mar do Caribe', 'Mar do Norte', 'Mar Báltico', 'Mar Vermelho']
9
```

### Adicionando um elemento ao início de uma matriz

Para adicionar um elemento ao início de uma matriz, você usa o método `unshift ()`:

```
2 let mares = ['Mar Negro', 'Mar do Caribe', 'Mar do Norte', 'Mar Báltico'];
3 mares.unshift ('Mar Vermelho');
4
5 console.log (mares);
6
7 Saída:
8 ['Mar Vermelho', 'Mar Negro', 'Mar do Caribe', 'Mar do Norte', 'Mar Báltico']
```

Fonte: Autor

### Removendo um elemento do final de uma matriz

Para remover um elemento do final de uma matriz, você usa o método `pop ()`:

```
2 let mares = ['Mar Negro', 'Mar do Caribe', 'Mar do Norte', 'Mar Báltico'];
3 const lastElement = mares.pop ();
4 console.log (lastElement);
5
6 Saída:
7 Mar Báltico
```



Fonte: Autor

## Removendo um elemento do início de uma matriz

Para remover um elemento do início de uma matriz, você usa o método `shift()`:

```
2 let mares = ['Mar Negro', 'Mar do Caribe', 'Mar do Norte', 'Mar Báltico'];
3 const firstElement = mares.shift();
4
5 console.log(firstElement);
6
7 Saída: Mar Negro
```

Fonte: Autor

## Dica quente para você não esquecer

- Trabalhando com Vetores. Saiba mais sobre a aplicação de vetores no link:  
<https://siteantigo.portaleducacao.com.br/conteudo/artigos/informatica/trabalhando-com-vetores/31845> (Acesso em 15/11/2022)
- Como ordenar um array de objetos em JavaScript  
<https://www.youtube.com/watch?v=yc0TYlnZlp4> (Acesso em 15/11/2022)

## Conteúdo Bônus



Para aprofundar seus conhecimentos sobre arrays em JavaScript, recomendamos a pesquisa sobre o livro “JavaScript: The Definitive Guide” (JavaScript: O Guia Definitivo) escrito por David Flanagan.

Este livro aborda de forma abrangente a linguagem JavaScript, incluindo o uso de arrays e outros recursos importantes. Ele explora os conceitos, sintaxe e principais métodos relacionados a arrays, além de fornecer exemplos práticos e detalhados.

Para encontrar esse conteúdo, você pode pesquisar por seu título e autor em plataformas gratuitas de busca de livros, como o Google Books ou a biblioteca virtual do seu instituto de ensino.

Lembrando que, ao realizar a pesquisa, é importante procurar por fontes confiáveis que ofereçam acesso gratuito ao conteúdo. Isso garantirá que você encontre o livro completo ou trechos relevantes para o seu estudo.

A leitura desse livro proporcionará um aprofundamento significativo em arrays e em JavaScript como um todo, complementando o conteúdo abordado na aula. Aproveite essa oportunidade para expandir seus conhecimentos e aprimorar suas habilidades na linguagem JavaScript.

## Referência Bibliográfica

FLANAGAN, David. **JavaScript: O Guia Definitivo**. 6ª Ed. Porto Alegre: Bookman, 2013.

FREEMAN, Eric. **Use a cabeça!**: programação JavaScript. 1ª Ed. São Paulo: Alta Books, 2016.

## ATIVIDADE PRÁTICA

## Título da Prática: Usando arrays



**Objetivos:** Compreender a implementação de arrays

**Materiais, Métodos e Ferramentas:** Para realizar esta prática vamos utilizar o Visual Studio Code.


### Prática

Essa prática consiste em analisar um array de números e aplicar uma série de verificações utilizando operadores lógicos e aritméticos. O objetivo é determinar certas propriedades do array e identificar se as condições especificadas são atendidas pelos elementos.

Dado um array de números inteiros, o programa deve realizar duas verificações distintas. Na primeira verificação, é necessário identificar se pelo menos um número no array não é par ou é menor que zero. Para isso, deve-se percorrer o array e aplicar as seguintes condições a cada elemento: se o número não for par (ou seja, seu resto da divisão por 2 é diferente de zero) ou se for menor que zero, então a condição não é atendida. Caso ao menos uma dessas condições seja verdadeira para algum número do array, a resposta será 'Sim'. Caso contrário, será 'Não'.

Na segunda verificação, o programa deve determinar se todos os números no array são divisíveis por 3 e 5 simultaneamente. Novamente, é necessário percorrer o array e aplicar as seguintes condições a cada elemento: se o número não for divisível por 3 (ou seja, seu resto da divisão por 3 não é zero) ou se não for divisível por 5, então a condição não é atendida. Se todas as condições forem verdadeiras para todos os elementos do array, a resposta será 'Sim'. Caso contrário, será 'Não'.



Ao final das verificações, o programa exibirá os resultados no console. Essas verificações fornecem uma análise das propriedades do array  permitindo identificar se há números que não são pares ou menores que zero, assim como determinar se todos os números são divisíveis por 3 e 5 simultaneamente.

Boa sorte!

## Resolução

```
// Array de números
```

```
const numeros = [5, 10, 15, 20, 25];
```

```
// Verificar se pelo menos um número do array não é par ou menor que 0
```

```
let algumNaoParOuMenorQueZero = false;
```

```
for (let i = 0; i < numeros.length; i++) {
```

```
  if (numeros[i] % 2 !== 0 || numeros[i] < 0) {
```

```
    algumNaoParOuMenorQueZero = true;
```

```
    break;
```

```
  }
```

```
}
```

```
// Verificar se todos os números do array são divisíveis por 3 e 5
```

```
let todosDivisiveisPor3E5 = true;
```

```
for (let i = 0; i < numeros.length; i++) {
```



```
    if (numeros[i] % 3 !== 0 || numeros[i] % 5 !== 0) {
```

```
        todosDivisiveisPor3E5 = false;
```

```
        break;
```

```
    }
```

```
}
```

```
console.log("Pelo menos um número não é par ou menor que zero?",  
    algumNaoParOuMenorQueZero);
```

```
console.log("Todos os números são divisíveis por 3 e 5?",  
    todosDivisiveisPor3E5);
```

**Ir para exercício**