

Arrays

Objetivo da Aula

Entender o que é, para que serve e como manipular um array em JavaScript.

Apresentação

Nesta aula entraremos mais profundamente sobre o conceito de array, aprendendo o que é, para que serve e como criar um array em **JavaScript**. O array, muito utilizado na programação, nada mais é do que uma estrutura utilizada para armazenar e organizar valores em uma única variável. Esses valores, chamados de elementos, podem ser acessados rapidamente por suas posições numéricas, denominadas como **índices**. Vamos entender melhor sobre essa estrutura? Vem comigo!

1. Array

Antes de definirmos array, é preciso que fique claro a diferença entre variáveis simples e compostas. Você já aprendeu o que é uma variável, lembra? Uma variável simples permite apenas o armazenamento de um único valor. Já a variável composta permite o armazenamento de vários elementos, sejam eles de um mesmo tipo de dados ou não, isso significa que podemos ter em um array um elemento inteiro (int), outro elemento *string* e outro elemento do tipo objeto. Comentamos acima que um array nada mais é do que uma estrutura utilizada para armazenar e organizar dados (elementos) em uma única variável. Logo, podemos concluir que um array é uma variável composta. Concorda?



Destaque

Os arrays em **JavaScript** são dinâmicos. O que isso significa? Significa que podem crescer ou diminuir conforme a necessidade sem que seja necessário declarar um tamanho fixo para o array ao criá-lo ou ao realocá-lo, caso seu tamanho mude (FLANAGAN, 2013).

Para acessar os elementos que compõem o array utilizaremos a sua posição numérica, chamada de índice, que começa por zero. Lembrando que podemos trocar, acrescentar e remover elementos conforme a necessidade. E como podemos criar um array em *JavaScript*? “A maneira mais fácil de criar um array é com um array literal, que é simplesmente uma lista de elementos de array separados com vírgulas dentro de colchetes” (FLANAGAN, 2013). Veja:

- **var meuArray = [];**

//um array sem elementos.

- **var meuArray = [4, 5, 7, 10, 20];**

//um array com 5 elementos numéricos.

- **var meuArray = [2.5, true, “Ana”];**

//3 elementos de vários tipos.

Além disso, um array literal não necessariamente precisa ter valores constantes. Veja:

- **var x = 1024;**
- **var meuArray = [x, x+1, x+2 x+3].**

A outra maneira de criar array é utilizando o construtor **new Array()**. Essa construtora pode ser chamada de 3 formas diferentes (FLANAGAN, 2013):

- **var meuArray = new Array();**

//sem argumentos, ou seja, um array vazio. Equivalente ao array literal (var meuArray = []).

- **var meuArray = new Array(10);**

//um array com comprimento especificado. Utilizado quando se sabe exatamente quantos elementos vão fazer parte do array, ou seja, este array terá 10 posições.

- **var meuArray = new Array(1, 2, 3, 4, 5, “teste”).**

//neste caso os argumentos do construtor se tornam elementos do novo array.

Vale ressaltar que, apesar de termos diversas formas diferentes de criar um array, o tipo literal é o mais utilizado e recomendado. Por quê? Pelo simples fato de ser mais prático, de ter maior legibilidade e também por ser mais veloz em tempo de execução.

1.1. Como Acessar os Elementos de um Array?

Para acessar os elementos de um array basta fazer a busca pelo seu índice. Veja:

```
JS 84 unsaved changes X
1 var cursos = ["Análise", "Redes", "Sistema de Informação"];
2 console.log(cursos); //todo o array "Análise", "Redes", "Sistema de Informação"
3 console.log(cursos[0]); //array na posição 0 - "Análise"
4 console.log(cursos[1]); //array na posição 1 - "Redes"
5 console.log(cursos[2]); //array na posição 2 - "Sistema de Informação"
```

Na *linha 2*, estamos exibindo na tela todos os elementos do array `cursos`. Nas *linhas 3-5*, estamos exibindo na tela o elemento específico de uma determinada posição. Fácil, não é mesmo? Também podemos percorrer o array utilizando o *for... of*, estrutura de repetição vista na unidade 2.

```
JS 96 unsaved changes X
1 var cursos = ["Análise", "Redes", "Sistema de Informação"];
2 for (var elemento of cursos)
3   console.log(elemento);
```

Console

```
"Análise"
"Redes"
"Sistema de Informação"
```

Como dito anteriormente, o array é uma estrutura não-tipada e dinâmica, ou seja, um elemento pode ser de qualquer tipo e um array pode crescer ou diminuir segundo a necessidade. Sendo assim, vamos conhecer alguns métodos que facilitam bastante o dia a dia dos desenvolvedores.

1.2. Como Adicionar Elementos em um Array?

- **unshift()** – adiciona um ou mais elementos no início do array. Veja:

```
JS 32 unsaved changes X
1 var cursos = ["Análise", "Redes", "Sistema de Informação"];
2 cursos.unshift("administração");
3 console.log(cursos);
```

```
Console
// [object Array] (4)
["administração","Análise","Redes","Sistema de Informação"]
```

- **push()** – adiciona um ou mais elementos no final do array. Veja:

```
JS 29 unsaved changes X
1 var cursos = ["Análise", "Redes", "Sistema de Informação"];
2 cursos.push("administração");
3 console.log(cursos);
```

```
Console
// [object Array] (4)
["Análise","Redes","Sistema de Informação","administração"]
```


- **splice()** – altera o conteúdo do array, adicionando novos elementos enquanto remove elementos antigos. Veja:

```
JS 84 unsaved changes X
1 let mes = ['Jan', 'Mar', 'Abril', 'Jun'];
2 mes.splice(1, 0, 'Fev'); // Inserção de FEV no índice 1
3 console.log(mes);
4 //Resultado: ['Jan', 'Fev', 'Mar', 'Abril', 'Jun']
5
6 mes.splice(4, 1, 'Maio'); // substitui o elemento do índice 4 por MAIO
7 console.log(mes);
8 //Resultado: ['Jan', 'Fev', 'Mar', 'Abril', 'Maio']
```

Na primeira parte do exemplo, estamos usando o `splice(1, 0, 'Fev')` para inserir o mês de "Fev" no índice 1, o segundo argumento foi 0 porque neste caso não queremos remover nenhum elemento do array. Na segunda parte do exemplo, estamos usando o `splice(4, 1, 'Maio')` para remover o elemento do índice 4 (Jun) e substituir por "Maio", para isso, tivemos que inserir o número 1 no segundo argumento para indicar que apenas 1 elemento será removido.

1.3. Como Remover Elementos de um Array?

- **shift()** – remove um ou mais elementos no início do array. Veja:



```

1 var cursos = ["Administração", "Análise", "Redes", "Sistema de Informação"];
2 cursos.shift();
3 console.log(cursos);

```

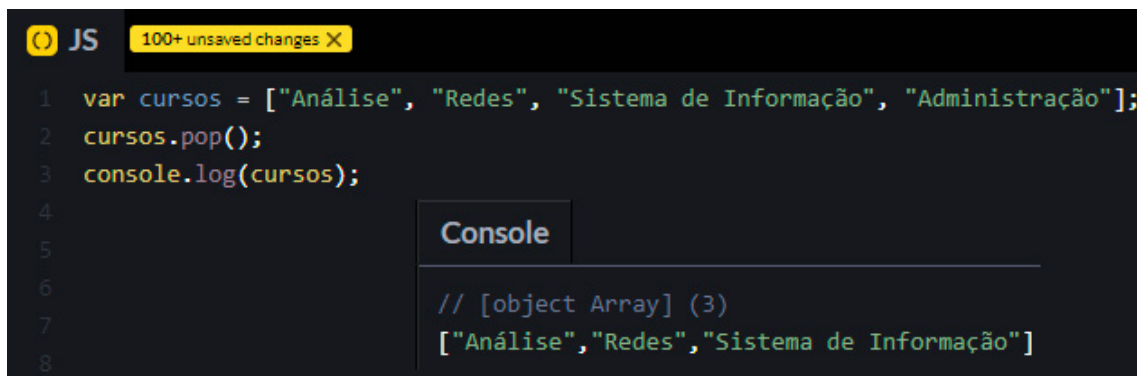
Console

```

// [object Array] (3)
["Análise","Redes","Sistema de Informação"]

```

- **pop()** – remove um ou mais elementos no final do array. Veja:



```

1 var cursos = ["Análise", "Redes", "Sistema de Informação", "Administração"];
2 cursos.pop();
3 console.log(cursos);

```

Console

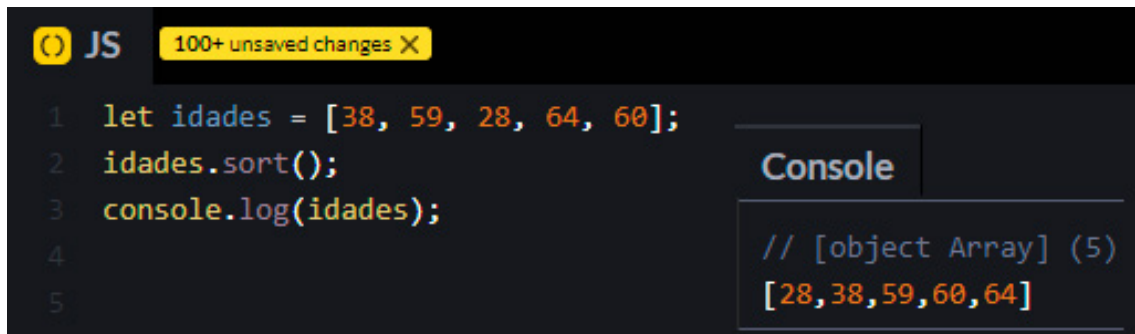
```

// [object Array] (3)
["Análise","Redes","Sistema de Informação"]

```

1.4. Como Ordenar um Array Numérico?

Para ordenar um array numérico basta utilizarmos o método `sort()`. Veja:



```

1 let idades = [38, 59, 28, 64, 60];
2 idades.sort();
3 console.log(idades);
4
5

```

Console

```

// [object Array] (5)
[28, 38, 59, 60, 64]

```

Considerações Finais

Nesta aula você teve a chance de entender melhor o que é e para que serve um array. Podemos dizer que o conceito de array é um dos mais importantes na programação, pois traz muita praticidade no desenvolvimento de aplicações mais robustas e complexas. Lembrando que um array em *JavaScript* não possui tamanho nem tipo fixos, com isso, podemos acrescentar, remover e trocar os elementos de posição livremente. Aprendemos como criar um array, conhecemos alguns métodos usados para incluir, alterar, excluir e ordenar elementos de um array. Legal, né? Não esqueça de acessar o material complementar, ele é muito importante para arrematar os conhecimentos adquiridos aqui. Na próxima aula vamos falar sobre o arrays multidimensionais. Te espero!

Materiais Complementares



JavaScript Array – aprenda o que é, como criar e usar:
<https://blog.betrybe.com/javascript/javascript-array/>



Variáveis Compostas – Curso JavaScript:
<https://youtu.be/XdkW62tkAgU>

Referências

FLANAGAN, David. *JavaScript: O guia definitivo*. Porto Alegre, RS: Bookman, 2013.