ARRAYS JAVASCRIPT

Samuel Amaro

9 de abril de 2021

Neste artigo final do módulo, nós vamos dar uma olhada em arrays - um elegante meio de armazenar uma lista de itens em uma mesmo variável. Aqui nós vemos o porquê isto é útil, depois exploraremos como criar uma array, recuperar, adicionar e remover itens armazenados em uma array, e mais.

Objetivo: Entender o que é array e como manipular ela em JavaScript.

O que é um array?

Arrays são geralmente descritas como "lista de objetos"; elas são basicamente objetos que contem múltiplos valores armazenados em uma lista. Um objeto array pode ser armazenado em variáveis e ser tratado de forma muito similar a qualquer outro tipo de valor, a diferença está em podermos acessar cada valor dentro da lista individualmente, e fazer super úteis e eficientes coisas com a lista, como laço através da lista e fazer a mesma coisa para cada valor. Talvez nós pegamos uma série de produtos e seus preços armazenados em uma array, e nós queremos fazer um laço através de todos eles e mostrar em um recibo, enquanto somamos todos os preços e mostramos o preço total ao final.

Se nós não tivessemos arrays, teríamos que armazenar cada item em uma variável separada, então chamar o código para mostrar e adicionar separadamente cada item. Isto seria muito mais longo de escrever, menos eficiente e mais suscetível a erros. Se nós temos 10 itens para adicionar na fatura, isto é ruim o bastante, mas e se fosse 100 itens ou 1000? Nós vamos retornar a este exemplo mais tarde neste artigo.

Como no artigo anterior, vamos aprender o básico de arrays introduzindo com alguns exemplos dentro de um console JavaScript.

Criando um array

Arrays são contruídas de colchetes, os quais contém uma lista de itens separada por vírgulas.

1. Vamos supor que queremos armazenar uma lista de compras em uma array — nós temos algo como o seguinte. Digite as linhas abaixo no seu console:

```
1
2  var shopping = ['bread', 'milk', 'cheese', 'hummus', '
     noodles'];
3  shopping;
```

Listing 1: Criando uma array

2. Neste caso, cada item na array é uma string, mas tenha em mente que você pode armazenar qualquer item em uma array — string, número, objeto, outra variável, até outra array. Você pode também misturar e combinar tipos de itens — eles não têm que ser todos números, strings, etc. Tente isto:

```
1
2      var sequence = [1, 1, 2, 3, 5, 8, 13];
3      var random = ['tree', 795, [0, 1, 2]];
```

Listing 2: array com armazena qualquer tipo

3. Tente criar um par de arrays você mesmo, antes de seguir em frente.

Acessando e modificando itens de um array

Você pode acessar itens individuais em uma array usando a notação de colchetes, da mesma forma que você acessa as letras em uma string.

1. Digite o seguinte no seu console

```
1
2    shopping[0];
3    //return "bread"
```

Listing 3: acessando primeiro item do array

2. Você também pode modificar um item em uma array simplesmente dando um novo valor ao item. Tente isto:

```
1
2  shopping[0] = 'tahini';
3  shopping;
```

```
4  //shopping vai retornar agora [ "tahini", "milk", "cheese",
     "hummus", "noodles"]
```

Listing 4: modificando item em um array

Nota: Nós dissemos isto antes, mas como lembrete — computadores começam a contar do 0!

3. Note que uma array dentro de uma array é chamada de array multidimensional. Você pode acessar um item dentro de uma array que está localizada dentro de outra array, colocando dois conjuntos de colchetes juntos. Por exemplo, para acessar um dos itens dentro de uma array, que é o terceiro item dentro da array `random` (veja a seção anterior), nós podemos fazer algo tipo isto:

```
1
2  random[2][2];
```

Listing 5: acessando um array multidimensional

4. Antes de continuar, faça algumas modificações nos exemplos, crie seus próprios arrays e veja o que funciona e o que não funciona. Divirta-se!

Encontrando o comprimento de uma array

Você pode encontrar o comprimento de uma array (quantos itens existem nela) exatamente do mesmo jeito que você encontra o comprimento (em caracteres) de uma string — usando a propriedade `length`. Tente o seguinte:

```
1
2    sequence.length;
3    //deve retornar 7
```

Listing 6: conprimento de um array

Isto tem outras funcionalidades, mas é mais comum usar em um laço para seguir todos os itens em uma array. Então, por exemplo:

```
1
2    var sequence = [1, 1, 2, 3, 5, 8, 13];
3    for (var i = 0; i < sequence.length; i++) {
4            console.log(sequence[i]);
5    }
```

Listing 7: percorrendo(iterando) sobre os itens de um array

Você irá aprender sobre laços propriamente em um artigo futuro, mas, brevemente, este código está dizendo:

1. Comece o laço no item número 0 na array.

2. Pare o laço no item de número igual ao comprimento da array. Isto funcionará para uma array de qualquer tamanho, mas neste caso vai parar o laço no item 7 (isto é bom, como o último item — que nós queremos que o laço cubra — é 6.

3. Para cada item, imprima no console do navegador com `console.log()`.

Alguns métodos úteis em array

Nesta seção vamos ver alguns métodos relacionados a array úteis que nos permitem dividir strings em itens de array e vice-versa, e adicionar novos itens em arrays.

Convertendo entre strings e arrays

Frequentemente você vai se deparar com alguns dados contidos em uma grande e longa string, e você pode querer separar os itens em uma forma mais útil e então manipular eles, como mostrar em uma tabela. Para fazer isto, nós podemos usar o método `split()`. Nesta forma mais simples, ela pega um parâmetro solitário, o caracter que você deseja separar da string e retorna o restante antes e depois do item separado na array.

Nota: Ok, isto é tecnicamente um método de string, não um método de array, mas nós podemos colocar em arrays já que cai bem.

1. Vamos brincar com isto para ver como funciona. Primeiro, crie uma string no seu console:

```
1
2       var myData = 'Manchester,London,Liverpool,Birmingham,
    Leeds,Carlisle';
```

Listing 8: criando string

2. Agora vamos dividir isto em cada vírgula:

```
1
2  var myArray = myData.split(',');
3  myArray;
```

Listing 9: dividindo string

3. Finalmente, tentamos encontrar o comprimento da sua nova array, e recuperar alguns itens dela:

```
myArray.length;
myArray[0]; // the first item in the array
myArray[1]; // the second item in the array
myArray[myArray.length-1]; // the last item in the array
```

4. Você também pode ir no sentido oposto usando o método `join()`. Tente o seguinte:

```
    var myNewString = myArray.join(',');
    myNewString;
```

5. Outro jeito de converter uma array em uma string é usar o método `toString()`. `toString()` é indiscutivelmente mais simples `join()` pois não necessita um parâmetro, mas mais limitado. Com `join()` você pode especificar diferentes separadores (tente o passo 4 com um caracter diferente da vírgula).

```
    var dogNames = ['Rocket','Flash','Bella','Slugger'];
    dogNames.toString(); //Rocket,Flash,Bella,Slugger
```

Adicionando e removendo itens de arrays

Nós ainda não falamos sobre adicionar e remover itens de uma array — vamos dar uma olhada agora. Nós vamos usar a array `myArray` que acabamos de criar na última seção. Se você não viu a última seção, primeiro crie a array no seu console:

```
    var myArray = ['Manchester', 'London', 'Liverpool', 'Birmingham
    ', 'Leeds', 'Carlisle'];
```

Listing 10: criando um array

Antes de tudo, para adicionar ou remover um item no final de uma array, nós podemos usar `push()` e `pop()` respectivamente.

1. note que você precisa para incluir um ou mais itens ao final da sua array. Tente isto:

```
    myArray.push('Cardiff');
    myArray;
    myArray.push('Bradford', 'Brighton');
    myArray;
```

Listing 11: incluir itens no final do array

2. O novo comprimento da array é retornado quando a chamada do método completa. Se você quer armazenar o novo comprimento da array em uma variável, você poderia fazer algo como isto:

```
1
2     var newLength = myArray.push('Bristol');
3     myArray;
4     newLength;
```

3. Removendo o último item da array é tão simples como um `pop()` nele. Tente isto:

```
1
2  myArray.pop();
```

Listing 12: removendo ultimo item do array e tão simples

4. O item que foi removido é retornado quando a chamada do método completa. Para salvar o item em uma nova variável, você poderia fazer isto:

```
1
2     var removedItem = myArray.pop();
3     myArray;
4     removedItem;
```

Listing 13: removendo iten de um array

`unshift()` e `shift()` funciona exatamente do mesmo modo que `push()` e `pop()`, respectivamente, exceto que eles funcionam no começo da array, não no final.

1. Primeiro `unshift()` — tente os seguintes comandos:

```
1
2     myArray.unshift('Edinburgh');
3     myArray;
```

2. Agora `shift()`;Tente estes!

```
1
2     var removedItem = myArray.shift();
3     myArray;
4     removedItem;
```