# Theory: Arrays

19 minutes 0 / 5 problems solved

Skip this topic Start practicing

900 users solved this topic. Latest completion was about 5 hours ago.

Imagine a situation: you're writing a shopping list before going to a supermarket. To save time, you're listing the items exactly in the order you're going to shop around: first, you'll get food, then some cleaning supplies, next you'll buy some stationery... If you were writing the shopping list not on paper but with the help of JavaScript, `Arrays` would fit that purpose best.

`Arrays` are an ordered collection that allows us to keep the values in a strictly defined order. We can access the elements by index: if you need item 15, you get item 15!

Now let's take a look at a few examples and see what arrays are all about.

## §1. Setting an array

An `Array` that already has some values can be set this way:

```
let items = ["cheese", "shampoo", "pen"];
```

To set an empty `Array`, use one of the two ways shown below:

```
let emptyArrayOne = new Array();
let emptyArrayTwo = [];
```

## §2. Reading elements by index

The main difference between `Objects` and `Arrays` lays in getting the values: you get object values through the string key, and array values through the ordinal index (the element's number). The indexes are numbered starting from zero.

Let's print out every single element in the example with items using the indexes:

```
console.log(items[0]);
console.log(items[1]);
console.log(items[2]);
```

In the output, we get the following:

```
cheese
shampoo
pen
```

The zeroth element here is `cheese`, the 1st is `shampoo`, and the 2nd is `pen`. The third element doesn't exist, therefore we can't call it.

To display the whole `Array`, we can call it by its name, without any indexes:

```
console.log(items);
```

## §3. Writing elements by index

To write a new element or rewrite an old one, use the following syntax:

```
items[1] = "soap";
items[3] = "book";
```

Also, say we set the fifth element skipping the fourth and therefore leaving it empty:

Current topic:

Arrays

Topic depends on:

× Variables

Topic is required for:

Slicing

ForEach method

For loops

Table of contents:

```
items[5] = "postcard";
console.log(items);
```

The output of this code will be the following:

```
Array(6) [ "cheese", "soap", "pen", "book", undefined, "postcard" ]
```

`Array's` length is kept in its property `length`:

```
console.log(items.length);
```

## §4. Arrays in queues

Usually, `Arrays` are used with two kinds of data structures: `queue` and `stack`.

To understand what a `queue` is, you can imagine an ordinary queue in a food store: new people join the queue at its end, and those in the beginning leave first. That's why we need two methods to implement a `queue`: one that adds elements to the end of the `Array` and one that deletes the elements at the beginning of the `queue`. These two methods are `push()` for adding one or more elements to the end of an array and return a new array length and `shift()` for deleting the zeroth element of array and returns its value:

```
items.push("flowers");
items.shift();
console.log(items);
```

Note that also the deleted element shifts the `queue`, so the first element becomes zeroth, the second turns first and so on.

```
Array(6) [ "soap", "pen", "book", undefined, "postcard", "flowers" ]
```

## §5. Arrays in stacks

To get the idea behind `stacks`, imagine a stack (a pile) of paper sheets. We stack these sheets in a certain order, one by one. We add each new elements to the top, and when we need to take a specific sheet out of the stack, we also have to start from the top, from the very last element.

We need two methods for implementing a `stack`: adding an element to the end and deleting the last one. We already know how to add an element to the end: that's our good old `push()` method. The method `pop()` is responsible for deleting the last element and returns its value:

```
items.pop();
console.log(items);
```

The output will be as the following:

```
Array(5) [ "soap", "pen", "book", undefined, "postcard" ]
```

As expected, the last element (flowers) was deleted.

## §6. Conclusion

In the current topic we considered a new data structure, `Arrays`. We learned about the operations of reading, writing, adding a new element to the end of `Array` and deleting the first and the last element from it. In the examples, we also got familiar with `queues` and a `stacks`.

Now it's time to put the new knowledge into practice.

Report a typo

90 users liked this theory. 1 didn't like it. **What about you?**

😍 🙂 😐 🙁 😡

Start practicing

Comments (0) Hints (0) Useful links (1) Show discussion