Member-only story

10 Javascript Exercises with Arrays

Andrei Borisov · Follow

6 min read · Apr 30, 2020

Listen Share More

Basically, this article could be useful for junior and middle Javascript software engineers. I prepared ten exercises with arrays, which I hope will help you to improve your coding skills.

Each task provided has an explanation, expected result, and solution. I do not affirm my solutions are the best ways to solve each of the exercises, but I believe they could help you to find some new ideas if you stuck.

Also, I would like to mention that I won't handle all the error cases, like passing undefined, null, or wrong data types. I provide a basic solution, not writing a library for production.

You can start in your repository or clone mine. In that repository, you can find a full list of exercises and solutions. Also, you can easily check your solutions with pre-created tests. Link: https://github.com/andrewborisov/javascript-practice.

1. **Fill.** Write a function that creates a new array with given values

```
/**
  * Task description: Write a method that creates a new array with given
values
  * Expected Result: (3, 'a') => ['a', 'a', 'a']
  * Task Complexity: 1 of 5
  * @param {number} arraySize - size of array
  * @param {?} value - value to fill
  * @returns {Array}
*/
const fill = (arraySize, value) => {
 throw new Error('Put your solution here');
}

const data = 3;
const valueToFill = 'a';

console.log(fill(data, valueToFill)) // ['a', 'a', 'a']
```

"Fill" function solution.

2. **Reverse.** Write a function that reverts the input array. Please, do not use `array.reverse();` to make this task more enjoyable.

```
/**
  * Task description: Write a method that reverts input array
  * Expected Result: [1, 2, 3] => [3, 2, 1]
  * Task Complexity: 1 of 5
  * @param {Array} array - Array of any elements
  * @returns {Array}
*/
const reverse = (array) => {
 throw new Error('Put your solution here');
}

const data = [1, 2, 3];

console.log(reverse(data)); // [3, 2, 1]
```

“Reverse” function solution.

3. **Compact.** Write a method that clears array from all unnecessary elements, like false, undefined, empty strings, zero, null

```
/**
  * Task description: Write a method that clears array from all
unnecessary elements, like false, undefined, empty strings, zero, null
  * Expected Result: [0, 1, false, 2, undefined, '', 3, null] => [1, 2,
3]
  * Task Complexity: 1 of 5
  * @param {Array} array - An array of any elements
  * @returns {Array}
*/
const compact = (array) => {
 throw new Error('Put your solution here');
}

const data = [0, 1, false, 2, undefined, '', 3, null];

console.log(compact(data)) // [1, 2, 3]
```

“Compact” function solution.

4. **From Pairs.** Write a method that returns an object composed of key-value pairs.

```
/**
  * Task description: Write a method that returns an object composed of
key-value pairs.
  * Expected Result: [['a', 1], ['b', 2]] => { a: 1, b: 2 }
  * Task Complexity: 2 of 5
  * @param {Array} array - a deep array of pairs
  * @returns {Array}
*/
const fromPairs = (array) => {
 throw new Error('Put your solution here');
}
```

```
const data = [['a', 1], ['b', 2]];

console.log(fromPairs(data)) // { 'a': 1, 'b': 2 }
```

“From Pairs” function solution.

5. **Without.** The method should return an array without listed values. Let’s try to find a solution when input data only primitive data types.

```
/**
  * Task description: Write a method that returns an array without
listed values
  * Expected Result: [1, 2, 3, 1, 2] without 1, 2 => [3]
  * Task Complexity: 2 of 5
  * @param {Array} array - Array of primitive data types
  * @param {?} args list of values to remove
  * @returns {Array}
*/
const without = (array, ...args) => {
 throw new Error('Put your solution here');
}

const data = [1, 2, 3, 1, 2];

console.log(without(data, 1, 2)); // [3]
```

“Without” function solution.

6. **Unique.** Write a method that returns a duplicate-free array

```
/**
  * Task description: Write a method that returns a duplicate-free array
  * Expected Result: Duplicate-free array [1, 2, 3, 1, 2] => [1, 2, 3]
  * Task Complexity: 2 of 5
  * @param {Array<string | number>} array - Array of primitive data
types
  * @returns {Array}
*/
const unique = (array) => {
 throw new Error('Put your solution here');
}

const data = [1, 2, 1, 2, 3];
```

```
console.log(unique(data)); // [1, 2, 3]
```

“Unique” function solution.

7. IsEqual. Write a function that compares two arrays and returns true if they are identical.

```
/**
  * Task description: Write a method that makes a shallow compare of two
arrays and returns true if they are identical.
```

Open in app ↗

Search Medium

L

```
*/
const isEqual = (firstArray, secondArray) => {
  throw new Error('Put your solution here');
}

const arr1 = [1, 2, 3, 4];
const arr2 = [1, 2, 3, 4];
const arr3 = [1, 2, 3, 5];
const arr4 = [1, 2, 3, 4, 5];

console.log(isEqual(arr1, arr2)); // true
console.log(isEqual(arr1, arr3)); // false
console.log(isEqual(arr1, arr4)); // false
```

“IsEqual” function solution.

8. **Flatten.** Write a function that turns a deep array into a plain array. Please, do not use `array.flat();` to make this task more enjoyable.

```
/**
  * Task description: Write a method that turns a deep array into a
plain array
  * Expected Result: [1, 2, [3, 4, [5]]] => [1, 2, 3, 4, 5]
  * Task complexity: 3 of 5
  * @param {Array} array - A deep array
  * @returns {Array}
*/
const flatten = (array) => {
 throw new Error('Put your solution here');
}

const data = [1, 2, [3, 4, [5]]];

console.log(flatten(data)); // [1, 2, 3, 4, 5]
```

“Flatten” function solution.

9. **Chunk.** Write a method that splits an array into parts of determined size

```
/**
  * Task description: Write a method that splits an array into parts of
determined size
  * Expected Result: ([1, 2, 3, 4, 5], 2) => [[1, 2], [3, 4], [5]]
  * Task complexity: 3 of 5
  * @param {Array} array - An array of any elements
  * @param {number} size - size of chunks
  * @returns {Array}
*/
const chunk = (array, size) => {
 throw new Error('Put your solution here');
}

const data = [1, 2, 3, 4, 5, 6, 7];

console.log(chunk(data, 2)) // [[1, 2], [3, 4], [5, 6], [7]]
console.log(chunk(data, 3)) // [[1, 2, 3], [4, 5, 6], [7]]
```

“Chunk” function solution.

10. **Intersection.** Create an array of unique values that are included in all given arrays. Let’s start with primitive data types.

```
/**
  * Task description: Write a method that creates an array of unique
values that are included in all given arrays
  * Expected Result: ([1, 2], [2, 3]) => [2]
  * Task complexity: 4 of 5
  * @param {?} arrays - Arrays of primitive data types
  * @returns {Array}
*/
const intersection = (...arrays) => {
 throw new Error('Put your solution here');
}

const arr1 = [1, 2];
const arr2 = [2, 3];

const arr3 = ['a', 'b'];
const arr4 = ['b', 'c'];
const arr5 = ['b', 'e', 'c'];

const arr6 = ['b', 'b', 'e'];
const arr7 = ['b', 'c', 'e'];
const arr8 = ['b', 'e', 'c'];

console.log(intersection(arr1, arr2)) // [2]
console.log(intersection(arr3, arr4, arr5)) // ['b']
console.log(intersection(arr6, arr7, arr8)) // ['b', 'e']
```

“Intersection” function solution.

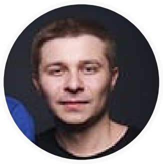
As I think, the coding practice gives you a significant impact on your skills, so for me, it is crucial to learn something new every day.

I hope the tasks mentioned above could help you to learn something new or just have some fun.

What’s next?

If you liked this one you can continue with the next collection of tasks for objects: https://medium.com/@andrey.igorevich.borisov/10-javascript-exercises-with-objects-8942cc502754

JavaScript Arrays Coding Javascript Arrays Junior Developer



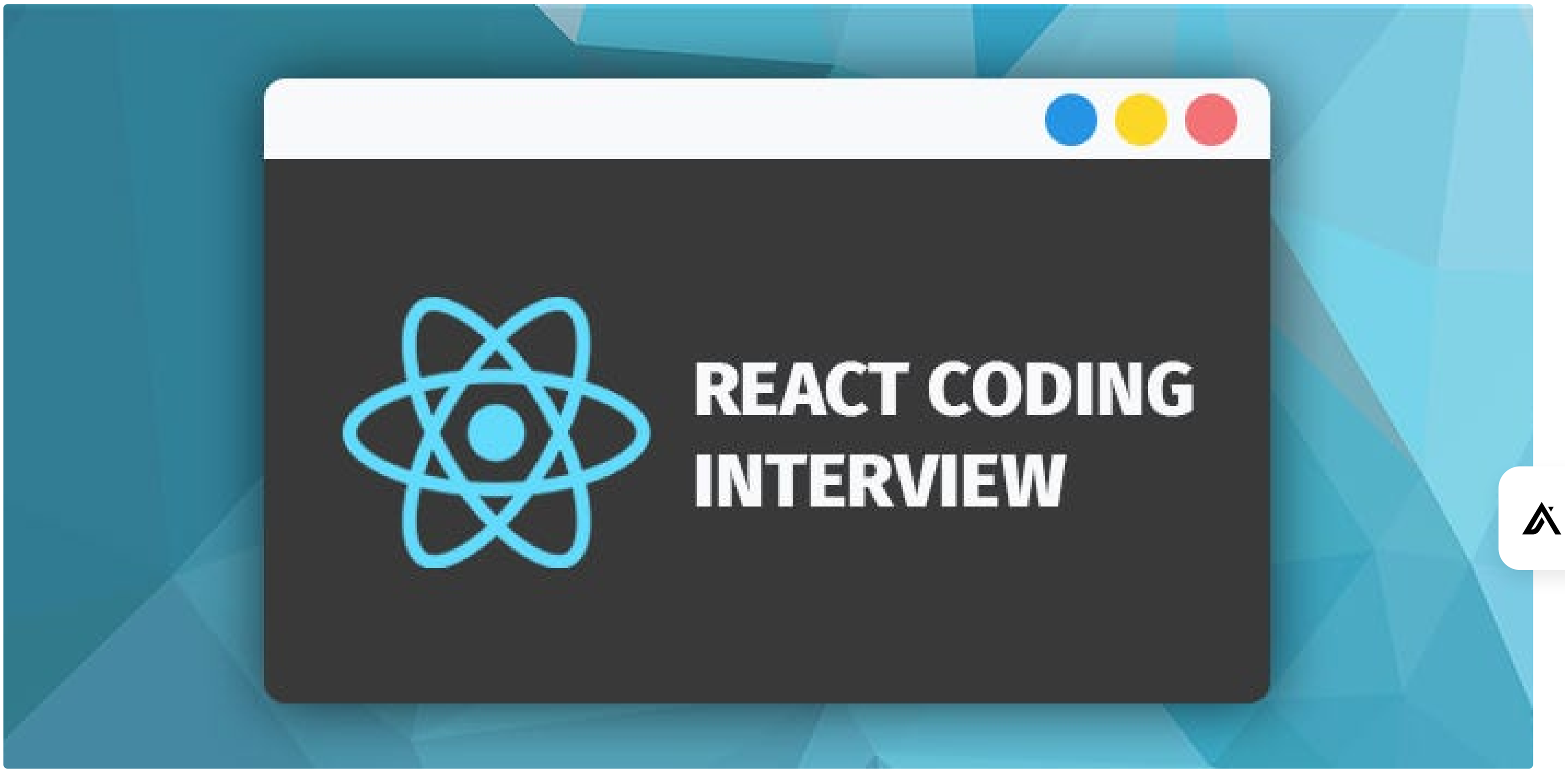
Follow

Written by Andrei Borisov

151 Followers

Senior Software engineer at EPAM

More from Andrei Borisov



Andrei Borisov in Geek Culture

React coding interview task

In this article, I would like to share a task for a junior or middle front-end developer on ReactJS.

✦ · 7 min read · Aug 2, 2021

204 3

Andrei Borisov

10 Javascript Exercises with Objects

Continuing the idea of ten exercises for arrays, I made a collection of tasks for objects. Like the previous one, this collection is...

✦ · 6 min read · May 14, 2020

44

Andrei Borisov

Создание монорепозиториев на NX

Создание и настройка монорепозитория при использовании библиотеки Nx. Создание React приложения на Nx. Разработка кастомных схем через Nx.

· 6 min read · Apr 23, 2020

4

Andrei Borisov

Что нужно знать front-end разработчику, чтобы переехать в США?

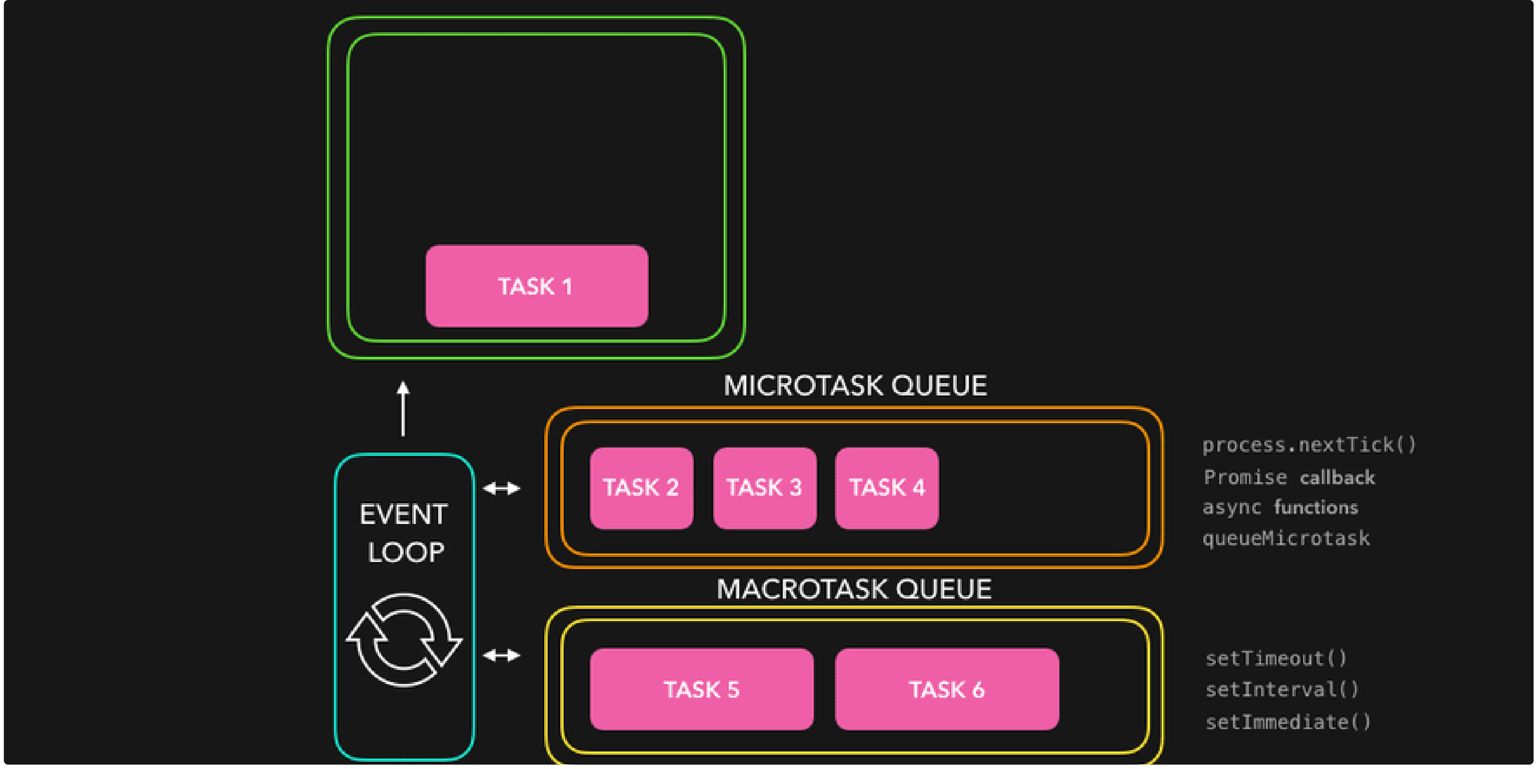
В это статье я расскажу о навыках, которые нужны javascript разработчику для переезда в СШ

· 4 min read · Jul 13, 2020

53

See all from Andrei Borisov

Recommended from Medium



Jeswanth Reddy in Version 1

Difference Between Promise and Async/Await

If you’re reading this, you probably understand how the promise and async/await are different in th execution context.

2 min read · May 12

195 2

Avinash Kumar

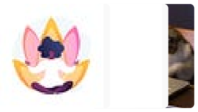
Slice and Splice in JavaScript?

slice()-: Slice is used to get a new array by selecting a sub-array from a given array and does not change the original array.

2 min read · Aug 10

3

Lists



Stories to Help You Grow as a Software Developer

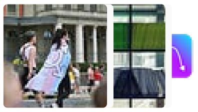
19 stories · 295 saves

General Coding Knowledge

20 stories · 235 saves

The New Chatbots: ChatGPT, Bard, and Beyond

13 stories · 91 saves



Generative AI Recommended Reading

52 stories · 170 saves

Abidullah

Mastering JavaScript Shorthand: Array Spread and Object Destructuring

Discover the power of JavaScript shorthand with array spread and object destructuring. Simplify code, boost productivity, and write elegant...

2 min read · Aug 9

8

Rabi Siddique in Level Up Coding

💻 15 JavaScript Techniques to Enhance Your Code

As a popular programming language for web development, JavaScript offers a lot of features and functionalities. Whether you’re a seasoned…

7 min read · Mar 14

559 8

Emma Delaney

Top 10 tricky JavaScript questions that I used to ask in interviews

Some tricky interview scheduling questions you will face. These questions seem easy, but there is something fishy about them. So today I'm...

4 min read · May 24

307 15

Cihan in Interesting Coding

Advanced One-Liner Codes in JavaScript in Detail

Discover the power of concise and efficient code with comprehensive guide to advanced one-liner codes in JavaScript.

✦ · 10 min read · Mar 17

147 1

See more recommendations