JavaScript Array Interview Questions in 2023

To brush up on the basics of JavaScript, we have compiled the most important JavaScript array interview questions and answers below. Go through each of the questions and get ready for the latest JavaScript.

Question 1: How do you create an empty array in JavaScript?

Answer: You can create an empty array in JavaScript using either of the following methods:

- Using array literal syntax: let arr = [];
- Using the Array() constructor: let arr = new Array();

Question 2: How do you access the first and last elements of an array?

Answer: You can access the first element of an array using index 0 and the last element using the index array.length - 1. For example:

```
let arr = [1, 2, 3, 4];
let firstElement = arr[0]; // 1
let lastElement = arr[arr.length - 1]; // 4
```

Question 3: How do you add an element to the end of an array?

Answer: You can add an element to the end of an array using the push() method. For example:

```
let arr = [1, 2, 3];
arr.push(4);
console.log(arr); // [1, 2, 3, 4]
```

Question 4: How do you remove the last element from an array?

Answer: You can remove the last element from an array using the pop() method. For example:

```
let arr = [1, 2, 3];
arr.push(4);
console.log(arr); // [1, 2, 3, 4]
```

Question 5: How do you loop through an array in JavaScript?

Answer: You can loop through an array using either a for loop or a forEach() method. For example:

- Using a for loop:

```
let arr = [1, 2, 3];

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

console.log(arr[i]);

}
```

- Using the forEach() method:

```
let arr = [1, 2, 3];

arr.forEach(function(element) {

console.log(element);

});
```

Question 6: How do you check if an element exists in an array?

Answer: You can check if an element exists in an array using the indexOf() method. If the element is not found, indexOf() returns -1. For example:

```
let arr = [1, 2, 3];

if (arr.indexOf(2) !== -1) {

console.log('Element found');

} else {

console.log('Element not found');

}
```

Question 7: How do you remove an element from an array at a specific index?

Answer: You can remove an element from an array at a specific index using the splice() method. For example, to remove the element at index 2:

```
let arr = [1, 2, 3];

arr.splice(2, 1);

console.log(arr); // [1, 2]
```

Question 8: How do you concatenate two arrays in JavaScript?

Answer: You can concatenate two arrays using the concat() method. For example:

let arr1 = [1, 2];

let arr2 = [3, 4];

let newArr = arr1.concat(arr2);

console.log(newArr); // [1, 2, 3, 4]

After these basic JavaScript array interview questions, below we have provided some array interview questions in JavaScript for experienced candidates.

Flatten Array JavaScript Interview Questions With Answers

In this section of JS interview questions now is time for some of the top flatten array JavaScript interview questions. Check JavaScript interview questions on the flattened array below.

Question 1: Write a function to flatten a nested array in JavaScript.

Answer:

function flattenArray(arr) {

return arr.reduce(function(flat, toFlatten) {

return flat.concat(Array.isArray(toFlatten) ? flattenArray(toFlatten) : toFlatten);

}, []);

}

Question 2: What does the reduce() method do in the flattenArray() function above?

Answer: The reduce() method in JavaScript takes an array and applies a function to each element, accumulating the result into a single value. In the flattenArray() function above, the reduce() method is used to concatenate the current element (either a flattened sub-array or a non-array value) to the flattened array so far.

Question 3: Can you give an example of a nested array that the flattenArray() function would be able to flatten?

Answer 3: Sure! Here's an example of a nested array that the flattenArray() function would be able to flatten:

var nestedArray = [1, [2, [3, 4], 5], 6];

flattenArray(nestedArray); // returns [1, 2, 3, 4, 5, 6]

Question 4: Can you explain how the flat() method can be used to flatten an array in JavaScript?

Answer: The flat() method is a built-in method in JavaScript that can be used to flatten an array. It takes a depth parameter, which specifies how many levels of nested arrays should be flattened. If no depth parameter is provided, it defaults to 1. Here's an example usage:

var nestedArray = [1, [2, [3, 4], 5], 6];

nestedArray.flat(2); // returns [1, 2, 3, 4, 5, 6]

Question 5: What are some potential issues to watch out for when flattening arrays in JavaScript?

Answer: One potential issue to watch out for is the risk of creating a very large flattened array, which could lead to performance issues or memory errors. Another issue to be aware of is the possibility of circular references in nested arrays, which could cause infinite recursion if not handled properly. Finally, different flattening methods (e.g. using reduce() vs. using flat()) may have different performance characteristics, so it's important to choose the method that's most appropriate for your use case.

JavaScript Array Manipulation Interview Questions With Answers

JavaScript Array Manipulation refers to the process of changing the content of an array in JavaScript. Among the JavaScript array coding interview questions, this topic is given extra importance.

Question 1: What is the difference between .map() and .forEach()?

Answer:

.map() and .forEach() are both array methods that allow you to loop through an array, but they differ in what they return.

- .map() returns a new array with the same length as the original array, where each element is the result of applying a callback function to the original element.
- .forEach() does not return anything, but it simply executes a callback function on each element of the array.

Example:

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

*const doubledNumbers = numbers.map(num => num * 2);*

console.log(doubledNumbers); // [2, 4, 6, 8, 10]

*numbers.forEach(num => console.log(num * 2)); // 2, 4, 6, 8, 10*

Question 2: How do you remove an element from an array in JavaScript?

Answer: You can remove an element from an array using the .splice() method. This method modifies the original array by removing or replacing existing elements and/or adding new elements.

Example:

const fruits = ['apple', 'banana', 'orange', 'mango'];

// remove 'orange'

fruits.splice(2, 1);

console.log(fruits); // ['apple', 'banana', 'mango']

In this example, we use the .splice() method to remove the third element (index 2) in the fruits array.

Question 3: What is the difference between .filter() and .find()?

Answer: Both .filter() and .find() are array methods that allow you to search for elements in an array that meet certain criteria.

- .filter() returns a new array with all elements that pass a certain test provided by a callback function.
- .find() returns the value of the first element in the array that passes a certain test provided by a callback function.

Example:

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

const evenNumbers = numbers.filter(num => num % 2 === 0);

console.log(evenNumbers); // [2, 4]

const firstEvenNumber = numbers.find(num => num % 2 === 0);

console.log(firstEvenNumber); // 2

In this example, we use the .filter() method to create a new array with only the even numbers in the numbers array. We use the .find() method to return the first even number in the numbers array.

Question 4: How do you sort an array in JavaScript?

Answer: You can sort an array using the .sort() method. This method modifies the original array by sorting its elements in place.

Example:

const fruits = ['banana', 'apple', 'orange', 'mango'];

fruits.sort();

console.log(fruits); // ['apple', 'banana', 'mango', 'orange']

In this example, we use the .sort() method to sort the fruits array in alphabetical order.

Question 5: How do you flatten a nested array in JavaScript?

You can flatten a nested array (i.e. an array that contains other arrays as elements) using the .flat() method. This method returns a new array with all sub-array elements concatenated into it recursively up to the specified depth.

Example:

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

const flattenedNumbers = numbers.flat(2);

console.log(flattenedNumbers); // [1, 2, 3, 4,

How to get first 3 elements of array in JavaScript?

To get the first three elements of an array in JavaScript, you can use the slice() method with a starting index of 0 and an ending index of 3. For example, to get the first three numbers from an array of numbers, you can write: let firstThreeNumbers = numbers.slice(0, 3);

What is Array[-1] in JavaScript?

Array[-1] in JavaScript will return the last element of the array, since negative index values count backwards from the end of the array. So, for example, if you have an array of numbers, you can access the last element using array[-1].