HOME ES6 ES.NEXT JS BOM JS DOM SNIPPETS

Home / JavaScript Tutorial / JavaScript Multidimensional Array

JavaScript Multidimensional Array

ADVERTISEMENT

Summary: in this tutorial, you will learn how to work with JavaScript multidimensional array.

Introduction to JavaScript multidimensional array

JavaScript does not provide the multidimensional array natively. However, you can create a multidimensional array by defining an array of elements, where each element is also another array.

For this reason, we can say that a JavaScript multidimensional array is an array of arrays. The easiest way to define a multidimensional array is to use the array literal notation.

To declare an empty multidimensional array, you use the same syntax as declaring one-dimensional array:

```
let activities = [];
```

The following example defines a two-dimensional array named activities:

```
let activities = [
    ['Work', 9],
    ['Eat', 1],
    ['Commute', 2],
    ['Play Game', 1],
```

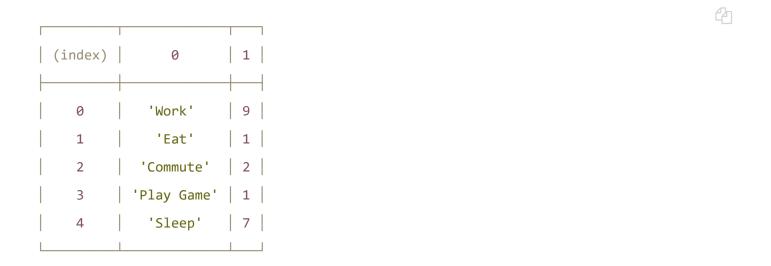
```
['Sleep', 7]
```

In the activities array, the first dimension represents the activity and the second one shows the number of hours spent per day for each.

To show the activities array in the console, you use the console.table() method as follows:

```
console.tables(activities);
```

The following illustrates the output:



Note that the (index) column is for the illustration that indicates the indices of the inner array.

To access an element of the multidimensional array, you first use square brackets to access an element of the outer array that returns an inner array; and then use another square bracket to access the element of the inner array.

The following example returns the second element of the first inner array in the activities array above:

```
console.log(activities[0][1]); // 9
```



Adding elements to the JavaScript multidimensional array

You can use the Array methods such as <code>push()</code> and <code>splice()</code> to manipulate elements of a multidimensional array.

For example, to add a new element at the end of the multidimensional array, you use the <code>push()</code> method as follows:

```
activities.push(['Study',2]);

console.table(activities);

(index) | 0 | 1 |
| 0 | 'Work' | 9 |
| 1 | 'Eat' | 1 |
| 2 | 'Commute' | 2 |
| 3 | 'Play Game' | 1 |
| 4 | 'Sleep' | 7 |
| 5 | 'Study' | 2 |
```

To insert an element in the middle of the array, you use the splice() method. The following inserts an element in the second position of the activities array:

```
activities.splice(1, 0, ['Programming', 2]);
console.table(activities);
```

Here is the output:



(index)	0	1 1
0	'Work'	9
1	'Programming'	2
2	'Eat'	1
3	'Commute'	2
4	'Play Game'	1
5	'Sleep'	7
6	'Study'	2
		1 1

This example calculates the percentage of the hours spent on each activity and appends the percentage to the inner array.

```
activities.forEach(activity => {
    let percentage = ((activity[1] / 24) * 100).toFixed();
    activity[2] = percentage + '%';
});
console.table(activities);
```

The following shows the output in the console:

(index)	0	1 1	2
0	' 'Work'	9	'38%'
1	'Programming'	2	'8%'
2	'Eat'	1	'4%'
3	'Commute'	2	'8%'
4	'Play Game'	1	'4%'
5	'Sleep'	7	'29%'



```
6 | 'Study' | 2 | '8%'
```

Removing elements from the JavaScript multidimensional array

To remove an element from an array, you use the pop() or splice() method.

For example, the following statement removes the last element of the activities array.

```
activities.pop();
console.table(activities);
```

Output:

(index)	0	1 1	2
0	'Work'	9	'38%'
1	'Programming'	2	'8%'
2	'Eat'	1	'4%'
3	'Commute'	2	'8%'
4	'Play Game'	1	'4%'
5	'Sleep'	7	'29%'

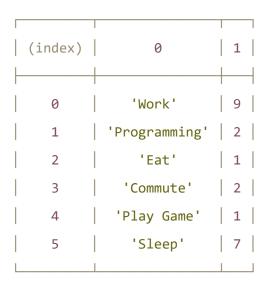
Similarly, you can remove the elements from the inner array of the multidimensional array by using the pop() method. The following example removes the percentage element from the inner arrays of the activities array.

```
activities.forEach((activity) => {
    activity.pop(2);
```



```
});
console.table(activities);
```

Output:



Iterating over elements of the JavaScript multidimensional array

To iterate a multidimensional array, you use a nested for loop as in the following example.

```
// Loop the outer array
for (let i = 0; i < activities.length; i++) {
    // get the size of the inner array
    var innerArrayLength = activities[i].length;
    // Loop the inner array
    for (let j = 0; j < innerArrayLength; j++) {
        console.log('[' + i + ',' + j + '] = ' + activities[i][j]);
    }
}</pre>
```

The first loop iterates over the elements of the outer array and the nested loop iterates over elements of the inner array.

The following shows the output of the script in the console:

```
[0,0] = Work
[0,1] = 9
[1,0] = Eat
[1,1] = 1
[2,0] = Commute
[2,1] = 2
[3,0] = Play Game
[3,1] = 1
[4,0] = Sleep
[4,1] = 7
[5,0] = Study
[5,1] = 2
```

Or you can use the forEach() method twice:

```
activities.forEach((activity) => {
    activity.forEach((data) => {
        console.log(data);
    });
});
```

Output:

```
Work

9
Eat

1
Commute

2
Play Game

1
```



27

42n

What is JavaScript

Install a JavaScript Code Editor

Meet the Console Tab of Devtools

JavaScript Hello World

JAVASCRIPT FUNDAMENTALS

Syntax

Variables

Data Types

Number

Boolean

JAVASCRIPT OPERATORS

Unary Operators

Assignment Operators

Logical Operators

Comparison Operators

CONTROL FLOW

if else

switch case

while

do-while

for

break

continue

Sleep

7

Study

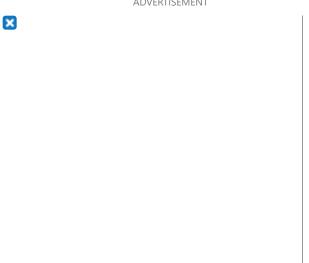
2

In this tutorial, you have learned how to use an array of arrays to create a JavaScript multidimensional array.

Was this tutorial helpful?



ADVERTISEMENT







ADVERTISEMENT

Search this website

GETTING STARTED

JAVASCRIPT STRINGS

JavaScript Strings

String Type

Concatenating Strings: concat()

Splitting a String into Substrings: split()

Locating a Substring Forward: indexOf()

Locating a Substring Backward:

lastIndexOf()

Extracting a Substring from a String:

substring()

Removing Whitespaces from Both Ends:

trim()

Extracting a part of a String: slice()

JAVASCRIPT ARRAY

JavaScript Array

Stack

Queue

Manipulate Elements: splice()

Sort Elements: sort()

Locate Elements: indexOf()

Check If Every Element Passes a Test:

every()

Check If At Least One Element Passes a

Test: some()

Filter Array Elements: filter()

Reduce an Array Into a Value: reduce()

Transform Array Elements: map()

Loop Through Array: forEach()

Multidimensional Array

JAVASCRIPT FUNCTIONS

Functions

Functions are First-Class Citizens

Anonymous Functions

IIFE

Callback Functions

JAVASCRIPT OBJECTS

JavaScript Objects

Object Properties

Prototype

Object Creation Patterns

Prototypal Inheritance

this Keyword

for...in Loop

ADVANCED FUNCTIONS

Passing By Value

Returning Multiple Values

Function Type

The call() Method

The apply() method

The bind() Method

Recursive Functions

Closures

REGULAR EXPRESSIONS

Basic Regular Expressions

search()

match()

replace()

Character Classes

Anchors

JAVASCRIPT RUNTIME

Execution Context

Call Stack

Event Loop

Hoisting

Variable Scopes

ADVERTISEMENT

ABOUT JAVASCRIPT TUTORIAL

The JavaScript Tutorial website helps you learn JavaScript programming from scratch quickly and effectively.

RECENT TUTORIALS

Npm Publish

Npm Uninstall

Npm View

Npm List

Npm Semantic Versioning

SITE LINKS

About Us

Contact Us

Privacy Policy

Copyright © 2020 by JavaScript Tutorial Website. All Right Reserved.