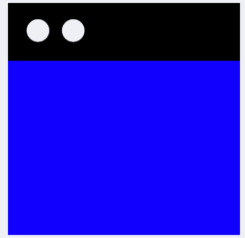
Code Academy



Gytis Juozenas

Masyvų metodai ir callback funkcijos

2022

JavaScript programavimo kalba

Šiandien išmoksite

01 Kokie yra paprastieji masyvų metodai

03 Kas yra callback funkcijos ir kur jos naudojamos

02 Kokie yra kompleksiniai masyvų metodai

Paprasti Masyvų Metodai: join ir toString

Sintaksė:

array.join(separator);

- metodas join sujungia masyve esančius elementus į stringą pagal paduotą atskyrimo būdą, pvz. Per tašką, šauktuką, dvitaškį ar kitokį simbolį.

-masyvą į stringą gali sujungti ir metodas array.toString(), tačiau negalima nurodyti kaip sujungti, ir rezultatas visada bus vertės atskirtos kableliais.

Pvz.:

let array = ['Du', 'gaideliai', 'baltus', 'zirnius', 'kule'];

let joined = array.join('-') // Du-gaideliai-baltus-zirnius-kule
joined = array.join(': ') // Du: gaideliai: baltus: zirnius: kule

joined = array.join('') // Dugaideliaibaltuszirniuskule joined

= array.join(' ') // Du gaideliai baltus zirnius kule

let stringified = array.toString() // Du,gaideliai,baltus,zirnius,kule

Paprasti Masyvų Metodai: splice

Sintaksė:

array.splice(index, howmany, item1, ..., itemX)

- Nurodoma indekso pozicija - required
- Kiek elementų pašalinti - optional
- Elementai, kuriuos pridėti vietoj pašalintų - optional

Grąžinama pašalintų elementų masyvas (array).

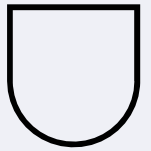
Pvz.: (pašaliname 1 elementą viduryje):

```
let arr = ["I", "knowledge", "Javascript"];
let whatsLeft = arr.splice(1, 1, "like", "learning");
console.log(arr); // [ "I", "like", "learning", "Javascript" ]
console.log(whatsLeft); // [ "knowledge" ]
```

Practice Time

1. Pašalinkite pirmus tris elementus iš masyvo aprašyto apačioje ir pakeiskite juos šiais: "Lets", "dance“.
2. Sujunkite atnaujinto masyvo elementus į vieną string.

const arr = ["I", "study", "JavaScript", "right", "now"];



Paprasti Masyvų Metodai: slice

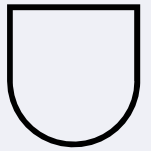
Sintaksė: arr.slice([start], [end])

Grąžinamas **naujas** masyvas (array), nukopijuojantis visus elementus nuo pradžios [start] iki pabaigos [end] (neįskaitant) pabaigos [end]);

Pvz.:

```
let arr = ["t", "e", "s", "t"];
console.log(arr.slice(1, 3)); // e, s (copy from 1 to 3)

console.log(arr.slice(-2)); // s, t (copy from end of an array 2 elements)
```



Paprasti Masyvų Metodai: concat

Metodas arr.concat sukuria naują masyvą, kuriame yra kitų masyvų vertės ir papildomi elementai.

Sintaksė:

arr.concat(arg1, arg2...)

Rezultatas yra naujas masyvas, kuriame yra elementai iš arg1, arg2 ir kt.

Practice Time

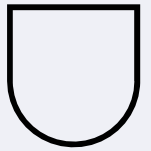
1.Naudodamiesi concat metodu sujunkite du masyvus ir dar pridėkite prie jų galo string, number ir object:

let first = ['slice', 'splice', 'concat'];

let second = ['push', 'pop', 'shift', 'unshift']

'length', 7, {subject: 'methods'}

2. Išloginkit naują masyvą.



Paprasti Masyvų Metodai: indexOf / lastIndexOf

Sintaksė:

Kaip ir string metodas indexOf/lastIndexOf skliausteliuose paduodama reikšmė, kurios indekso mes ieškome.

Pvz. Turime masyvą:

```
let arr = [1, 0, false];

console.log( arr.indexOf(0) ); // 1
console.log( arr.indexOf(false) ); // 2
console.log( arr.indexOf(null) ); // -1
```

Callback funkcijos

*1. button.addEventListener('click', **changeBackground**)*

2. button.addEventListener('click', () =>
{ document.body.style.backgroundColor =
'yellow';
})

- callback funkcijų naudojimas užtikrina, kad tam tikras kodas pasileis tik tuomet, kuomet bus įvykdyta kita užduotis, pvz. Pirmiausia turi būti paspaustas mygtukas, kad pasileistų kita funkcija.

-pavyzdyje naudojama **changeBackground** funkcija atlieka callback vaidmenį, todėl, kad addEventListener() taip pat yra funkcija, kuri tikisi dviejų parametrų - įvykio pavadinimo, ir funkcionalumo, kurį turi atlitki **tik įvykiui pasileidus**.

- Galima sukurti ir savo funkcijas, kurios į parametrus priims kitą funkciją;
- Javascript turi daug metodų (funkcijų), kurie reikalauja callback funkcijų.

[Daugiau apie callback funkcijas]

Practice Time

1. Turime akordų masyvą:
   let accords = ["D", "G", "C", "C7", "F"];
2. Parašykite kodą, kuris prie akordo (raidės) pridės 7 -> G7. Tačiau jeigu prie akordo 7 jau yra - turime ignoruoti.

// core funkcija kuri eis per array ir grąžins rezultatą
// callback funkcija kuri bus pritaikoma kiekvienam masyvo elementui

Test array → Expected Result

["G", "F", "C"] → ["G7", "F7", "C7"]

["Dm", "G7", "E", "A"] → ["Dm7", "G7", "E7", "A7"]

["F", "E", "A7", "Ab7", "Gm7", "C7"] → ["F7", "E7", "A7", "Ab7", "Gm7", "C7"]

Kompleksiniai masyvų metodai

1. forEach()

Šis metodas gali padėti pasiekti masyvo elementus.

```
const arr = [1, 2, 3, 4, 5, 6];
```

1. būdas, funkcionalumą apsirašome metodo viduje

```
arr.forEach(value => {
    console.log(value); // output: 1 2 3 4 5 6
});
```

2. būdas, pasirašome funkciją, kurią perduosime metodui vykdyti.

```
const consoleItems = (item) => console.log(item)

arr.forEach(consoleItems);
```

- **forEach**() metodas suveikia kaip **coreFunction**, o funkcija viduje kaip **callbackFunction**

Practice Time

1. Turime skaičių masyvą:
let numbers = [5, 1, 7, 2, -9, 8, 2, 7, 9, 4, -5, 2, -6, 8, -4, 6];

2. Parašykite funkciją, kuri suks forEach ciklą per masyvą ir sukurs po naują elementą kiekvienai vertei su jos indexu ir atspausdins HTML‘e

Rezultatas:

Index Nr: 0, value: 5

Index Nr: 1, value: 1

...

3. Kas turi laiko, pasibandykite dviem būdais, su callback funkcija ir be.

Kompleksiniai masyvų metodai

2. map()

Šis metodas sukuria naują masyvą, iškviečiant pateiktą funkciją kiekvienam masyvo (array) elementui.

```
const arr = [1, 2, 3, 4, 5, 6];

// Pridėkime po 1 prie kiekvieno elemento esančio arr masyve

const oneAdded = arr.map(num => num + 1);
// Viską saugome naujame kintamajame
console.log(oneAdded); // output [2, 3, 4, 5, 6, 7]
```

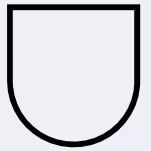
- Originalus masyvas išlieka nepakitęs

```
console.log(arr); // output: [1, 2, 3, 4, 5, 6]
```

Practice Time

Turime skaičių masyvą:
let numbers = [5, 1, 7, 2, -9, 8, 2, 7, 9, 4, -5, 2, -6, 8, -4, 6];
Žemiau yra aprašytos dvi užduotys su aukščiau pavaizduotu masyvu:

1. Parašykite funkciją arrDoubled, kuri sukuria ir grąžina naują masyvą, kurio elementai padauginti iš 2;
2. Parašykite funkciją arrMultiplied, kuri sukuria ir grąžina naują masyvą, kurio elementai padauginti iš skaičiaus kuris nurodytas iškviečiant funkciją. Tai reiškia jums reikės paduoti du parametrus į funkciją;
3. Parašykite funkciją getBudgets kuri grąžina sumą visų biudžetų; CodePen
4. Naudodamiesi map() metodu praeikite pro visą objektą ir sukurkite naują masyvą kuris grąžina tik vardus.



Kompleksiniai masyvų metodai

3. includes()

Šis metodas patikrina masyvą ir jeigu kaip argumentas nurodytas elementas masyve yra, grąžina true.

```
const arr = [1, 2, 3, 4, 5, 6];

arr.includes(2); // output: true
arr.includes(7); // output: false
```

Kompleksiniai masyvų metodai

4. filter()

Šis metodas sukuria naują masyvą su elementais, atitikusiais sąlygą funkcijos viduje.

```
const arr = [1, 2, 3, 4, 5, 6];

// item(s) greater than 3
const filtered = arr.filter(num => num > 3);
console.log(filtered); // output: [4, 5, 6]

console.log(arr); // output: [1, 2, 3, 4, 5, 6]
```

Practice Time

Pasiimkite startinius kodus iš čia this codepen

1. Parašykite funkciją isPersonInArray kuri iškviečiama paims du parametrus names masyvą ir ranka įrašytą vardą (string). Jeigu įrašytas vardas egzistuoja masyve tuomet turi kodas tikrinti ar tai vyriškos lyties ar moteriškos lyties vardas. Jeigu vyriškos output turėtų būti toks: Welcome Mr. Name, jeigu moteriškos: Welcome Miss. Name, jei vardas masyve nerastas: Unfotunately Name is not in our list. (includes())
2. Parašykite funkciją arrCountTwos kuri paima masyvą kaip parametrą ir grąžina skaičių kiek tame masyve yra dvejetų. (filter())

Kompleksiniai masyvų metodai

5. some()

Šis metodas patikrina, ar **bent vienas** iš masyvo (array) elementų atitiko sąlygą. Jei atitiko, grąžinama true, jeigu neatitiko false.

```
const arr = [1, 2, 3, 4, 5, 6];

// at least one element is greater than 4?  const
largeNum = arr.some(num => num > 4);
console.log(largeNum); // output: true
```

Kompleksiniai masyvų metodai

6. every()

Šis metodas patikrina, ar **visi** masyvo (array) elementai atitiko sąlygą. Jei atitiko, grąžinama true, jeigu neatitiko false.

```
const arr = [1, 2, 3, 4, 5, 6, 11];

// all elements are less than 10
const lessTen = arr.every(num => num < 10);
console.log(lessTen); // output: false
```

Practice Time

Pasiimkite startinius kodus iš čia this codepen

1. Patikrinkite ar masyve monies yra bent viena neigiama reikšmė; (some())
2. Parašykite funkciją belowHundred kuri pasiims masyvą kaip parametrą ir patikrins ar bent vienas skaičius yra mažiau nei 100. Jei taip reikės išfiltruoti masyvą ir grąžinti tik tas reikšmes, kurios atitinka kriterijų, jeigu visi atitinka tuomet išmesti žodžius „All numbers are above 100“; (some() & filter())
3. Pasirašykite funkciją symbolified kuri paima masyvą kaip parametrą ir tikrina ar visi vardai yra ilgesni nei 3 raidės. Jei taip tuomet tikrina ar bent viename varde yra raidė a. Jei taip tuomet išfiltruoja tuos vardus ir pirmas a raides pakeičia į ženklą @ ir galiausiai grąžina masyvą su vardais kur a raidės yra pakeistos @ ženklais.
   output > ["S@ulė", "P@ulius", "S@ndra"]