.join - pretvara niz u string

let vezba = [ime1, prezime1, sifra1, email1];

let stringNiza = vezba.join(', ');

Object.values objekat pretvara u niz

<p id="demo"></p>

const person = {

  name: "John",

  age: 30,

  city: "New York"

};

document.getElementById("demo").innerHTML = Object.values(person);

.toString( , ) = konvertuje numericku vrednost u string.

let number = 1425;

let numberString = number.toString();

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

    console.log(numberString[i]);

}

.toFixed(2) - u ovom slucaju zaokruzuje decimale koje se pojavljuju na 2 decimale.

  let average = (num1 + num2 + num3) / 3;

    if(average >= 10){

        div.css("color", "green");}

else { div.css("color", "red");}

    div2.html(`<h1> ${average.toFixed(2)} </h1>`);

.split - pravi niz od inputa

const itemsInput = document.getElementById("itemsInput");

function onGenerate() {

  const list = document.createElement("ul");

  const itemsArray = itemsInput.value.split(","); // ['aa', 'bbb', 'ccc'];

Konvertovanje json u objekat I objekat u json string:

// Parse the json text file into JS object

let jsObject = JSON.parse(json);

// We can use it like object now

console.log(jsObject.trainer);

// We can create objects into json strings as well

let newJson = JSON.stringify(jsObject);

.includes

const validateEmail3 = (email) => email.includes("@");

const myEmail = "perapera@yahoo.com";

.reduce - za nizove, svaki element uzima. Ne menja izvorni niz.

const prices = [35, 17, 15];

prices.reduce((result, element) => {

    return (result += element);

  }, 0);

.sort - sortira od najmanjeg do najveceg, **menja** niz!

const prices = [35, 17, 15];

prices.sort();

console.log(prices);

- ako hocu da ne menjam prvi niz a da napravim drugi koji je sortiran.

const prices = [35, 17, 15];

const copyOfPrices = [...prices];

copyOfPrices.sort();

console.log(copyOfPrices);

- sortiranje od najveceg do najmanjeg

const prices = [35, 17, 15];

const copyOfPrices = [...prices];

copyOfPrices.sort((a, b) => b - a);

console.log(copyOfPrices);

.map()- se koristi za transformisanje svakog elementa u nizu i vraća novi niz sa transformisanim elementima.

const pets = ["Bobby", "Cicko", "Rex", "Kitty", "Bambi"];

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

const prices = [35, 17, 15];

const uppercasedPets = pets.map((element) => element.toUpperCase());

console.log("uppercasedPets", uppercasedPets);

const multipliedBy3 = grades.map((element) => element \* 3);

console.log("multipliedBy3", multipliedBy3);

const betterGrades = grades.map((element) => {

  if (element === 5) {

    return element;

  }

  return element + 1;

});

console.log('betterGrades', betterGrades);

.filter - se koristi za filtriranje elemenata niza na osnovu zadatog kriterijuma i vraća novi niz koji sadrži samo elemente koji zadovoljavaju taj kriterijum.

const pets = ["Bobby", "Cicko", "Rex", "Kitty", "Bambi"];

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

const prices = [35, 17, 15];

const petsWithB = pets.filter((element) => element.startsWith("B"));

console.log(petsWithB); // ["Bobby", "Bambi"]

console.log(pets); // ["Bobby", "Cicko", "Rex", "Kitty", "Bambi"]

const evenNumbers = grades.filter((element) => (element % 2 === 0));

console.log('evenNumbers', evenNumbers);

const olderThan18 = employees.filter((element)=> (element.age > 18));

console.log('olderThan18', olderThan18);

const females = employees.filter((element) => element.gender === 'female');

console.log('females', females);

Glavna razlika između **filter** i **map** je u tome što **filter** odabira elemente na osnovu nekog uslova, dok **map** transformiše svaki element u nizu prema funkciji koja je prosleđena. Obe metode ne menjaju originalni niz, već vraćaju novi niz sa rezultatima.

- Kada pise da vratimo npr full name ili ime nekoga ko je iz nekog grada, itd, onda map jer to vraca samo te podatke a ne ceo objekat.

- setTimeout()

const displayTeacherName = () => {

  console.log("Teacher: Aida P.");

};

const displayAcademyName = () => {

  console.log("Quinshift academy 2024");

};

setTimeout(displayTeacherName, 3000); // on hold

displayAcademyName(); // ne blokira ovu funkciju

- setInterval()

let counter = 1;

let intervalId;

const runTimer = () => {

  console.log(counter);

  counter++;

  if(counter === 6){

    clearInterval(intervalId);

  }

};

intervalId = setInterval(runTimer, 2000);// vraca neki Id 346534tk=46464

**For**

const names = ["Alice", "Bob", "Charlie", "David", "Eve"];

function fillList() {

  let list = document.getElementById("list");

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

    let item = document.createElement("li");

    item.textContent = names[i];

    list.appendChild(item);

  }

}

Validation:

function Student (firstName, lastName, age, email){

    this.firstName = firstName || "...";

    this.lastName = lastName || "...";

    this.age = Number(age) || 0;

    this.email = email || "no email";

}

let database = [];

const btnAdd = document.getElementById("btn");

const firstNameInput = document.getElementById("fname");

btnAdd.onclick = function(){

    const firstName = firstNameInput.value;

let isFormValide = true;

     if (!firstName){

        console.warn("First name not entered");

        isFormValide = false;

     }

     if(isFormValide){

        const newStudent = new Student (firstName, lastName, age, email);

        database.push(newStudent);

        console.log('database', database);

     }

}

Object - Osnovni nacin da se kreira, moze I kroz konstruktor funkcije.

const student = {

  firstName: "Niko",

  lastName: "Nikic",

  years: 18,

  getFullName: function () {

    return `${this.firstName} ${this.lastName}`;

  },

};

Copy of an object:

const original = new Student("Niko", "Nikic", 18);

const copy = { ...original };

Delete - for arrays

const numbers = [12, -2, 10];

delete numbers[1];// [12, undefined, 10]

Dodavanje na kraju - .push

Dodavanje na pocetku  .unshift

Skidanje sa pocetka .shift

Skidanje sa kraja .pop

While -

let i = 0;

/

while (i <= 5) {

  console.log(i);

  i++;

}

Do … While -

let i = 0;

do {

  console.log("Vrednost i:", i);

  i++;

} while (i < 5);

For … Of -

function Student(firstName, lastName) {

  firstName ? firstName : "---";

  this.firstName = firstName || "---";

  this.lastName = lastName || "---";

  this.getFullName = function () {

    console.log(`${this.firstName} ${this.lastName}`);

  };

}

const students = [new Student("Aida", "P"), new Student("Miroslav", "P")];

for (let element of students) {

  element.getFullName();

}

forEach -

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

numbers.forEach(function(number, index) {

  console.log("Element na indeksu " + index + " je " + number);

});

Object.create() - kreiranje novog objekta, stvaranje novog ali se ne menja bazni objekat.

const obj = {

  name: "Aida",

};

// const copy = { ...obj };

const copy = Object.create(obj);

copy.name = "Almir";

Object.assign() - spaja dva objekta u jedan, ali tada I dva bazna objekta dobijaju atribute drugog objekta. Da bi se to sprecila, da bi oni ostali isti, a samo da ih spojimo u treci objekat, dodaju se zagrade {}.

const studentBasicInfo = {

  firstName: "John",

  lastName: "Doe",

  age: 23,

};

const studentAdditionalInfo = {

  address: "Other Adress",

  street: "Other Street",

  streetNumber: 12,

  mobile: "111 111 111",

};

const student = Object.assign({}, studentBasicInfo, studentAdditionalInfo);

Object.keys() - vraca niz naziva atributa.

Object.values() - vraca niz vrednosti tih atributa.

Object.entries() - vraca kombinaciju. Niz koji sadrzi naziv I vrednost.

console.log(Object.keys(student));

console.log(Object.values(student));

console.log(Object.entries(student));