1.

<https://github.com/yankovalenko94/JS_task_answers> - კურსის რეპოზიტორია

რეფერენსი - <https://www.sendpotion.com/>

Список необходимых плагинов:

* All Autocomplete
* Auto Complete Tag
* Code Runner
* Import Cost
* JavaScript (ES6) code snippets
* jshint
* Live Server
* Multiple clipboards for VSCode
* Reactjs code snippets
* Sass
* Theme - Oceanic Next
* vscode-icons

წიგნები

* [Современный учебник JavaScript](https://learn.javascript.ru/)
* Книга "JavaScript. Полное руководство", Флэнаган Дэвид  
  7е издание 2021го года.
* [Серия книг «You Don’t Know JS»](https://github.com/azat-io/you-dont-know-js-ru)
* [Выразительный Javascript](https://karmazzin.gitbook.io/eloquentjavascript_ru/)
* «JavaScript. Шаблоны» Стоян Стефанов
* [телеграм-канале](https://t.me/petrychenko_ivan) по хэштэгу #полезности и другим. Там тоже можно подчерпнуть информацию для новичков.

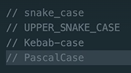
<https://jshint.com/docs/options/>

 ბრაუზერს ეუბნება რომ ვმუშაობთ ახალი თანამედროვე სტანდარტის და სტილის სკრიპტთან.

9.

Use strict - <https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Strict_mode>



 ფოლდერების და ცვლადის სახელების ჩაწერის სტილი, და მთავარი camelCase

infinity

console.log(4/0);

[Running] node "c:\Users\Home\Desktop\UDEMY\tempCodeRunnerFile.js"

Infinity

-infinity

console.log(-4/0);

[Running] node "c:\Users\Home\Desktop\UDEMY\tempCodeRunnerFile.js"

-Infinity

NaN

console.log("ika" \* 9);

[Running] node "c:\Users\Home\Desktop\UDEMY\tempCodeRunnerFile.js"

NaN

ობიექტის სტრუქტურა და სინტაქსი

const obj = {

  name: "ika",

  age: 37,

  isMarried: true

};

ობიექტში რო მივწვდეთ მნიშვნელობას

const obj = {

  name: "ika",

  age: 37,

  isMarried: true

};

console.log(obj.name);

[Running] node "c:\Users\Home\Desktop\UDEMY\tempCodeRunnerFile.js"

ika

მეორე მეთოდიც არსებობს

const obj = {

  name: "ika",

  age: 37,

  isMarried: true

};

console.log(obj["name"]);

[Running] node "c:\Users\Home\Desktop\UDEMY\tempCodeRunnerFile.js"

ika

ობიექტებში ქი(key) სტრინგია, შეგვიძლია ბრჭყალებითაც ჩავწეროთ.

ობიექტში ელემენტის ჩამატება

const obj = {

  name: "ika",

  age: 37,

  isMarried: true

};

obj.surname = "beridze";

console.log(obj);

[Running] node "c:\Users\Home\Desktop\UDEMY\tempCodeRunnerFile.js"

{ name: 'ika', age: 37, isMarried: true, surname: 'beridze' }

მეორე გზაც არსებობს ჩამატების

const obj = {

  name: "ika",

  age: 37,

  isMarried: true

};

obj['surname'] = "beridze";

console.log(obj);

console.log(obj.surname);

[Running] node "c:\Users\Home\Desktop\UDEMY\tempCodeRunnerFile.js"

{ name: 'ika', age: 37, isMarried: true, surname: 'beridze' }

beridze

მასივის სტრუქტურა და სინტაქსი

let arr = ["ika", 2, true, "name", {}, []];

მასივის ელემენტებზე წვდომა

let arr = ["ika", 2, true, "name", {}, []];

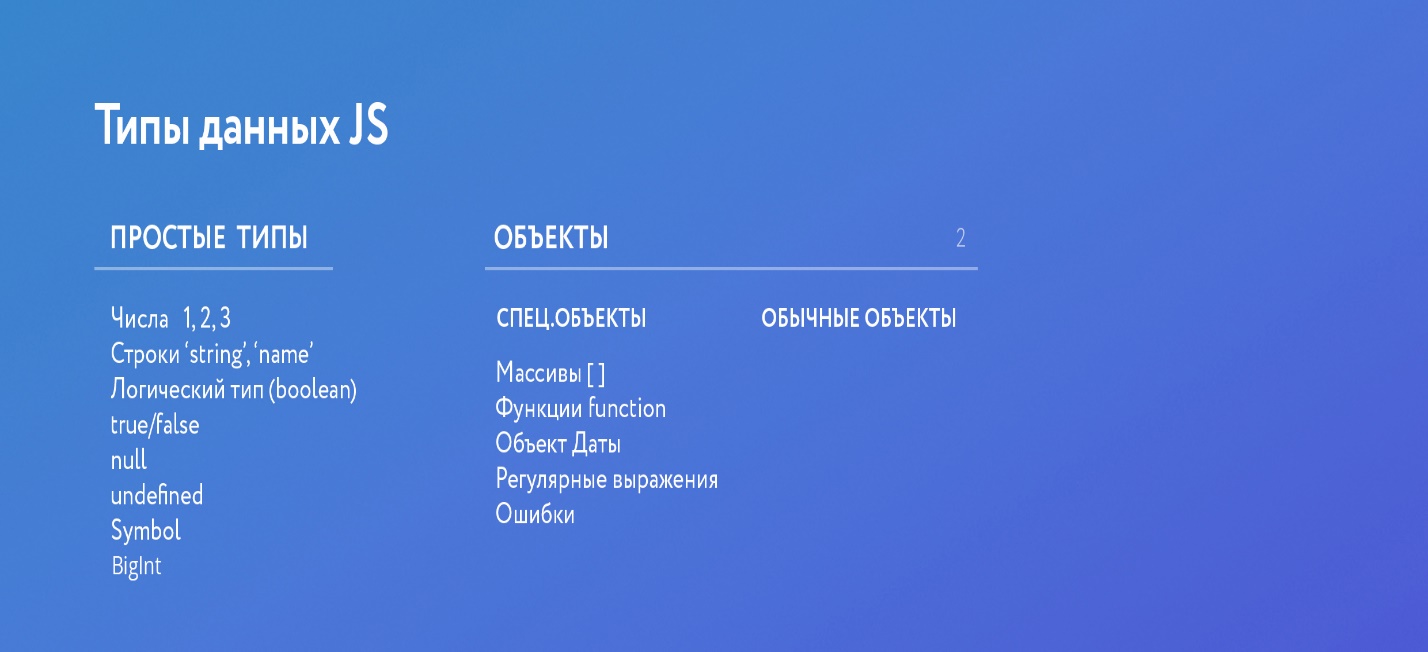
console.log(arr[0]);

[Running] node "c:\Users\Home\Desktop\UDEMY\tempCodeRunnerFile.js"

ika

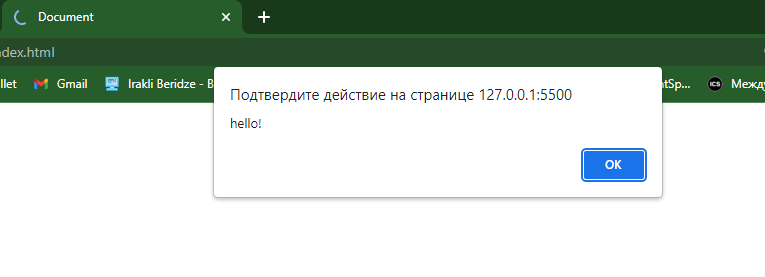
სტატია Symbol-ზე - <https://learn.javascript.ru/symbol>

სტატია BigInt-ზე - https://medium.com/@hydrock/bigint-%D0%BD%D0%BE%D0%B2%D1%8B%D0%B9-%D1%82%D0%B8%D0%BF-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-%D0%B2-js-dd5c29446570

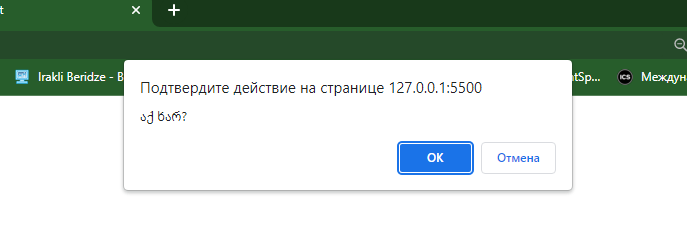


ბრაუზერში მოდალური ფანჯრის გამოტანა

alert('hello!');



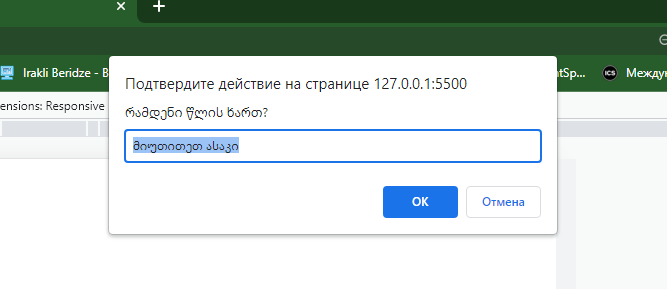
let question = confirm ("აქ ხარ?");



Ok - ზე დაჭერა დააბრუნებს true-ს, cancel-ზე დაჭერა - false-ს.

let answer = prompt('რამდენი წლის ხართ?', 'მიუთითეთ ასაკი');

console.log(answer);



კონსოლში დააბრუნებს იმ მნიშვნელობას რასაც მივუთითებთ. თუ ცარიელს დავტოვებთ ფლეისჰოლდერის მნიშვნელობას დააბრუნებს. მომხმარებლის მიერ შეყვენილი ინფუთები ყოველთვის არის სტრინგი.

ცვლადის მნიშვნელობის ტიპის გაგება

let answer = prompt('რამდენი წლის ხართ?', 'მიუთითეთ ასაკი');

console.log(typeof(answer));

//string

თუ გვინდა რომ ცვლადი ინტეჯერი გახდეს მაშინ + უნდა დავუწეროთ წინ

let answer = +prompt('რამდენი წლის ხართ?', 'მიუთითეთ ასაკი');

console.log(typeof(answer));

// number

მომხმარებლების მიერ მოცემული პასუხების მასივში ჩაწერა

let answers = [];

answers[0] = prompt('how old are you?', '');

answers[1] = prompt('whats your name?', '');

answers[2] = prompt('where are you from?', '');

document.write(answers);



მასივის არის ობიექტი

let answers = [];

console.log(typeof(answers));

[Running] node

object

null-ზეც typeof-ს გამოაქვს ობიექტი, მაგრამ ეს ოფიციალურად აღიარებული შეცდომაა

console.log(typeof(null));

[Running] node

object

confirm, prompt და alert არ მუშაობენ კონსოლში. ისინი ბრაუზერში ჩაშენებული ფუნქციებია.

სტატია

document.write - <https://developer.mozilla.org/ru/docs/Web/API/Document/write>

typeof - <https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Operators/typeof>

ინტერპოლაცია

კონკატენაციისგან განსხვავებით უფრო მოსახერხებელი სტილით სტრინგების ჩაწერის მეთოდი. მუშაობს მხოლოდ ( ` ` )ბექთიქებზე.

let category = 'toys';

console.log(`https://someurl.com/${category}/5`);

[Running] node

https://someurl.com/toys/5

სტატია

$ { } template literals - <https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Template_literals>

კონკატენაცია

console.log(4 + '5');

console.log(4 + +'5');

console.log('con' + 'cat' + 'ana' + 'tion');

[Running] node

45

9

concatanation

ინკრიმენტი და დეკრიმენტი

ერთის გაზრდა და ერთით შემცირება მოცემული რიცხვითი მნიშვნელობის

let incr = 10,

    decr = 10;

incr++;

decr--;

console.log(incr);

console.log(decr);

[Running] node

11

9

პოსტფიქსული ვარიანტი - ჯერ ცვლადის მნიშვნელობა გამოაქვს კონსოლში და მერე აკეთებს მათემატიკურ მოქმედებას

let incr = 10,

    decr = 10;

console.log(incr++);

console.log(decr--);

[Running] node

10

10

პრეფიქსულ ვარიანტში ჯერ მათემატიკურ მოქმედებას აკეთებს და მიღებული შედეგი გამოაქვს კონსოლში

let incr = 10,

    decr = 10;

console.log(--incr);

console.log(++decr);

[Running] node

9

11

% - ნაშთის ოპერატორი, მოდულო

= მნიშვნელობის მინიჭება

== ტოლობის ოპერატორი, ოღონდ ცვლადის ტიპებს არ ამოწმებს

=== მკაცრი ტოლობის ოპერატორი, ცვლადის ტიპებსაც ამოწმებს

!= არ უდრის, ოღონდ ცვლადის ტიპებს არ ამოწმებს

!== არ უდრის, ცვლადის ტიპებსაც ამოწმებს

&& - 2 ამპერსანტი, ლოგიკური ოპერატორი „და“

| | - ლოგიკური ოპერატორი „ან“

! - ლოგიკური „უარყოფა“

სტატია

პრიორიტეტები - <https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Operators/Operator_Precedence#table>

ბითვაიზ ოპერატორები - <https://learn.javascript.ru/bitwise-operators>

Git

git init - რეპოზიტორიის შექმნა

git config --local user.name ika

git config --local user.email [iberidze38@gmail.com](mailto:iberidze38@gmail.com)

git status - სტატუსის შემოქმება

git add -A - ყველა ფაილის დამატება რეპოზიტორიაში

git commit -a -m"first comit" - დაკომიტება

git log - კომიტების ისტორიის ნახვა

git remote add origin <https://github.com/irakli85/udemy.git>

git push -u origin master - მხოლოდ ერთხელ, შემდეგში git push

git branch branchName - ბრენჩის შექმნა

git checkout branchName - ბრენჩზე გადართვა

git checkout -b branchName - ბრენჩის შექმნა და თან მასზე გადართვა ერთდროულად

git merge – 2 ბრენჩის გაერთიანება

git clone https://github.com/irakli85/fylo-landing-page-with-two-column-layout-master.git fyloProject - ტერმინალიც კომანდი, რეპოზიტორიის ლინკი, ფოლდერის სახელი რომელშიც გვინდა კლონირება

git pull - გითჰაბიდან ლოკალურად ჩამოტვირთვა, პროექტის განახლება

რესურსები

<https://githowto.com/ru> - გიტის სასწავლო კურსი

<https://git-scm.com/book/ru/v2/%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5-%D0%9E-%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B5-%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8F-%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B9> - გიტის დოკუმენტაცია

<https://learngitbranching.js.org/?locale=ru_RU> - ინტერაქტიული გიტის სასწავლო პლატფორმა

<https://www.codecademy.com/article/command-line-commands> - ტერმინალის ქომანდები

ტერმინალი ქომანდები

PS C:\Users\Home\Desktop\UDEMY> cd .. - ფოლდერიდან გამოსვლა

PS C:\Users\Home\Desktop> cd UDEMY - ფოლდერში შესვლა

ფაილებთან მუშაობის შემდეგ ჩასატარებელი ბრძანებების კომბინაცია:

git add -A - ყველა ფაილის ან ცვლილებისდამატება რეპოზიტორიაში

git commit -a -m"some comit" - დაკომიტება

git push - გითჰაბზე ატვირთვა