const “название” = - переменная

const “название” = () - функция

const “название” = {} объект

Функция записанная в свойствах объекта – метод

Функция() – вызов функции

let arr = []; let arr = new Array(); - два вида записи массива

console.log(arr[0]) – пример вывода в консоль элемента массива по его индексу

…название - копия

\*\*

Глаголы-префиксы для функций:

get - получить какое-то значение

set - установить какое-то значение\*

create - создать какую-то сущность

update - обновить какую-то сущность

delete - удалить какую-то сущность

show - показать что-то

hide - скрыть что-то

search - найти что-то

calc - вычислить что-то

check - проверить чтотто

const get Pi = () => 3.14

const setWeather = (weather) => { /\* … \*/ }

const createUser = (user) => { /\* … \*/ }

const update User = (id) => { /\* . \*/ }

const delete User = (id) => { /\* \*/ }

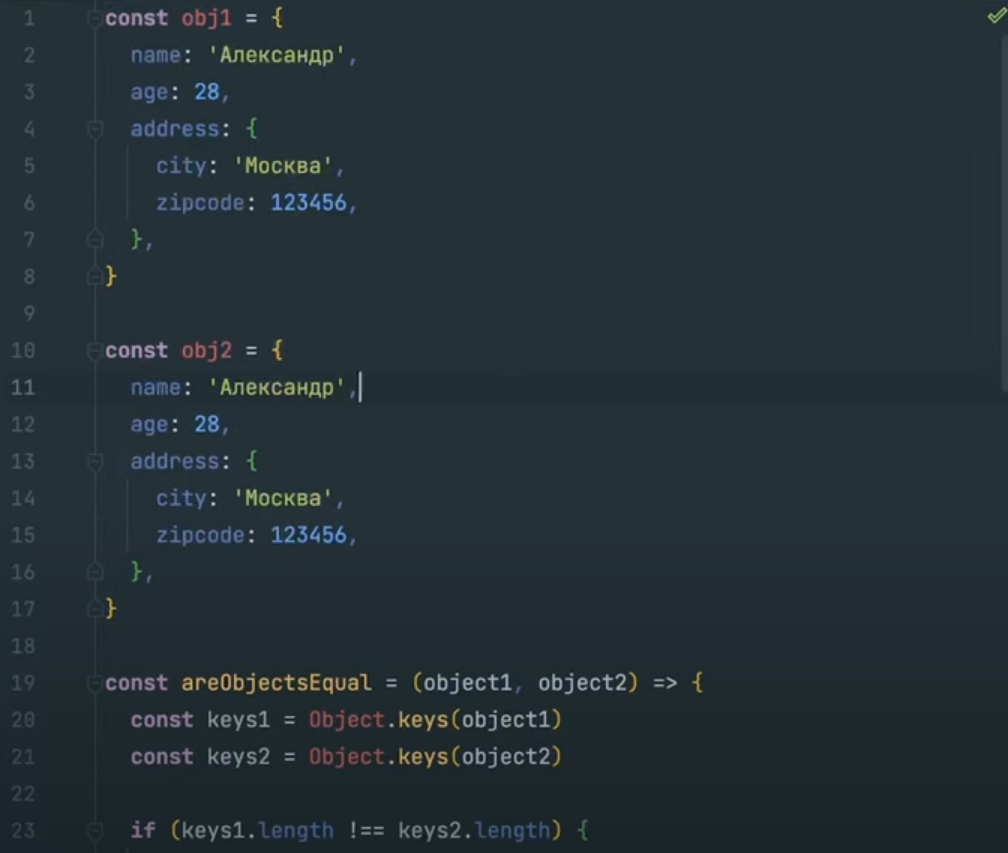
const showMenu = () => document.querySelector( selectors: '.menu').class1

const hideMenu = () => document.querySelector( selectors: '.menu').classI

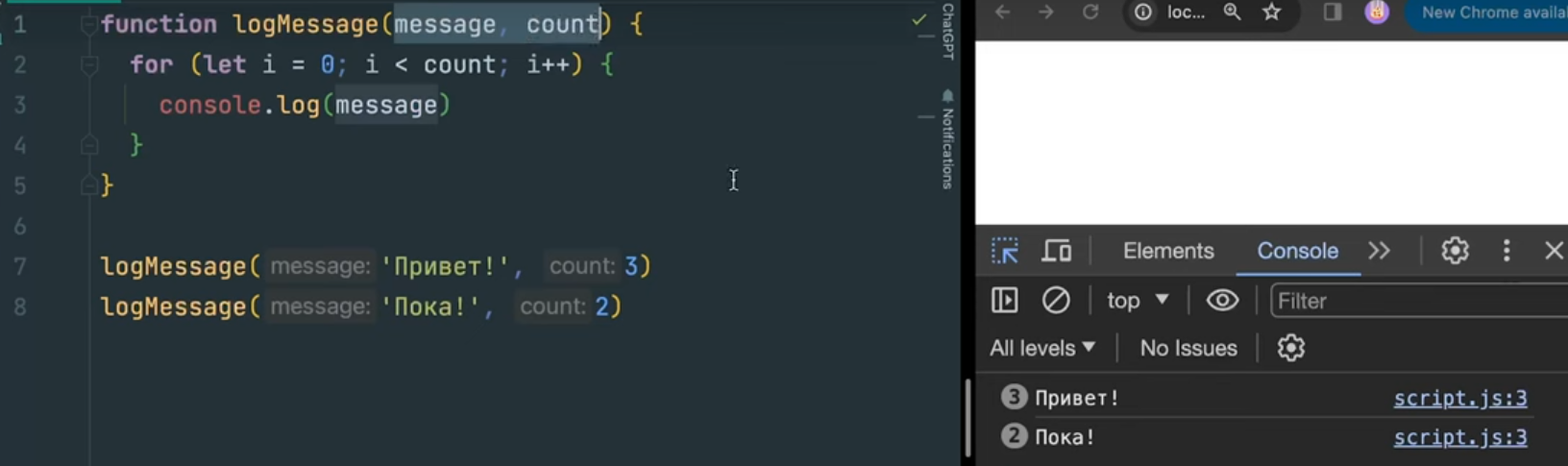
const search Products = (query) => products.filter(({ name }-) => name.

const calcSum = (a, b) => a + b

const check Age = (age) => age > 18|



# ФУНКЦИЯ (№9)



logMessage – название

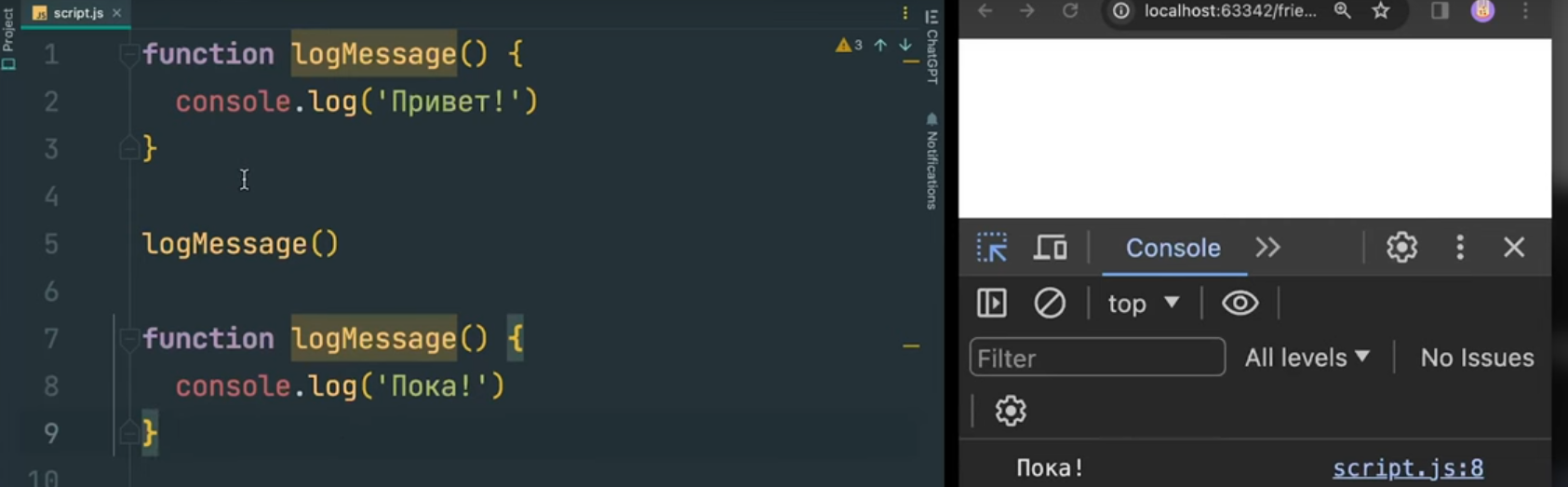
(message, count) – параметры

При вызове – аргументы

Функция записанная в свойствах объекта – метод

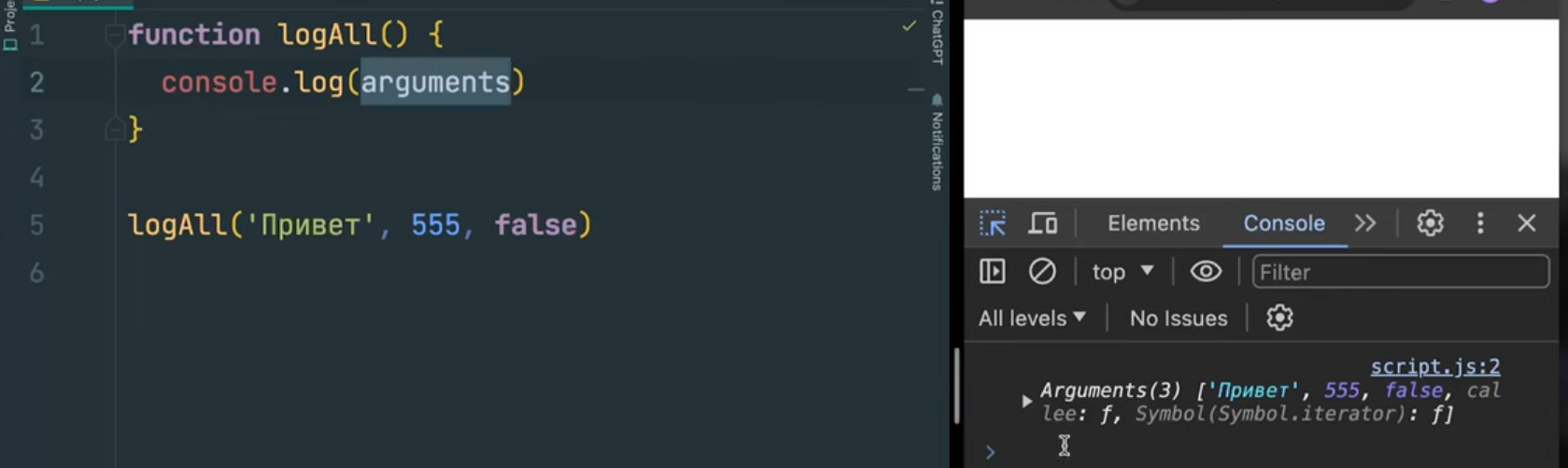
Функция() – вызов функции

## Function Declaration



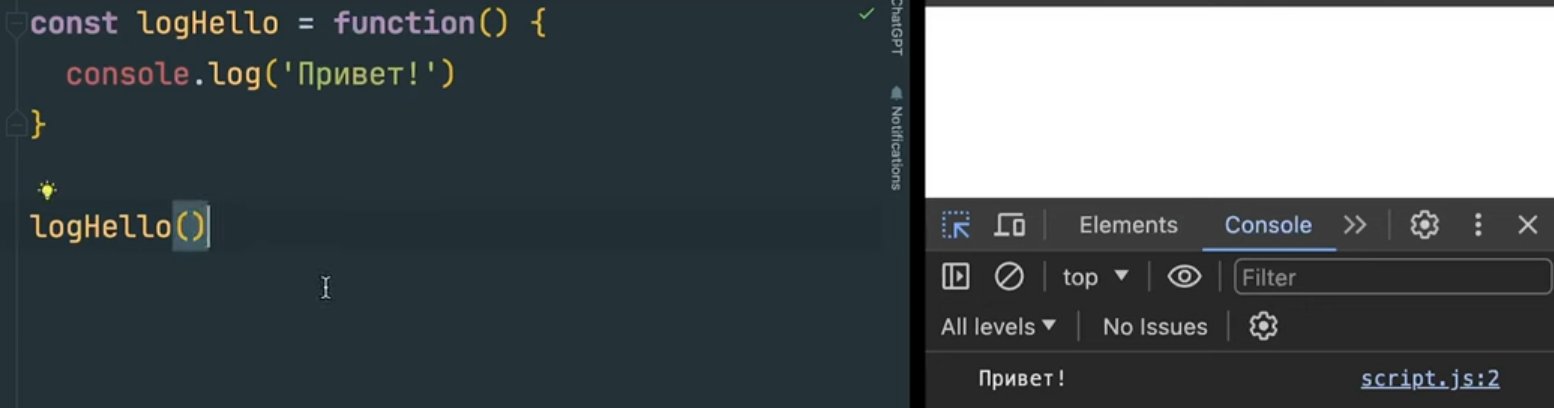
1. Все функции “поднимаются” в самое начало

Можно вызывать функцию до её объявления, но вызвано будет самое последнее её переименование



1. Можно получить доступ к не явной переменной “arguments”
2. Своеобразная работа с this

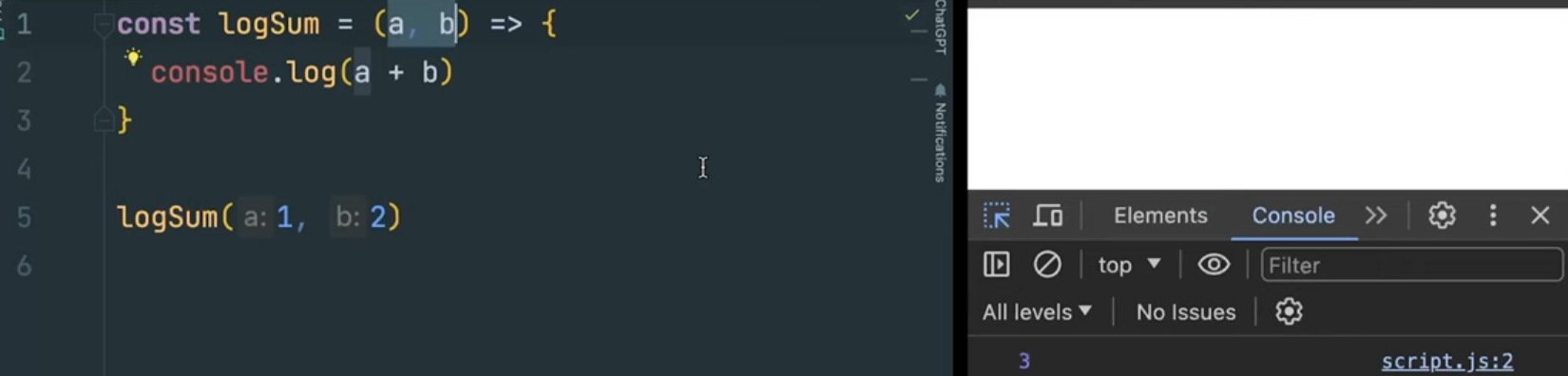
## Function Expression



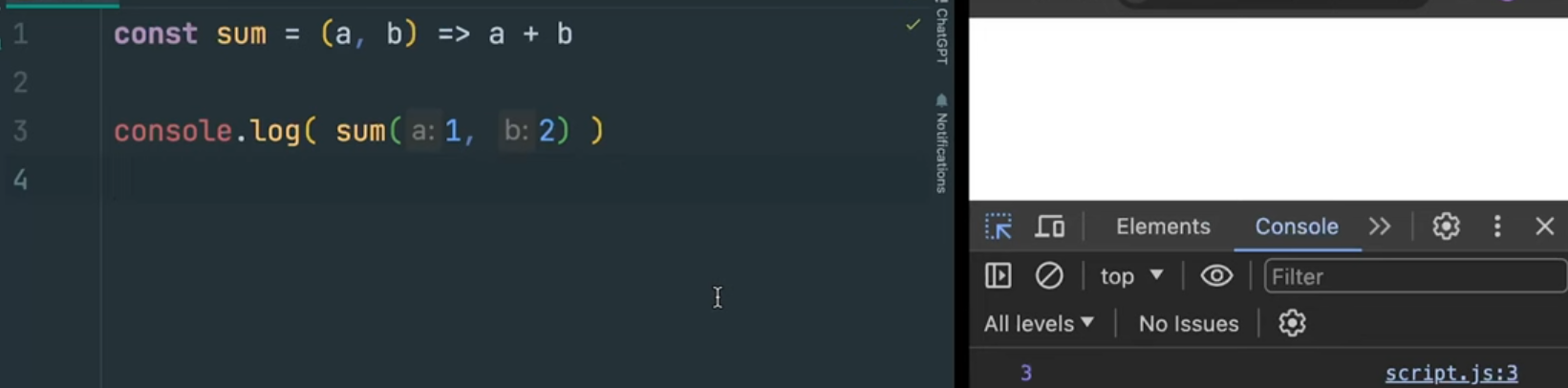
Запись функции как функциональное выражение и присваиваем в переменную const/let/var

1. Нельзя переопределить (c let можно)
2. Нельзя вызвать раньше объявления

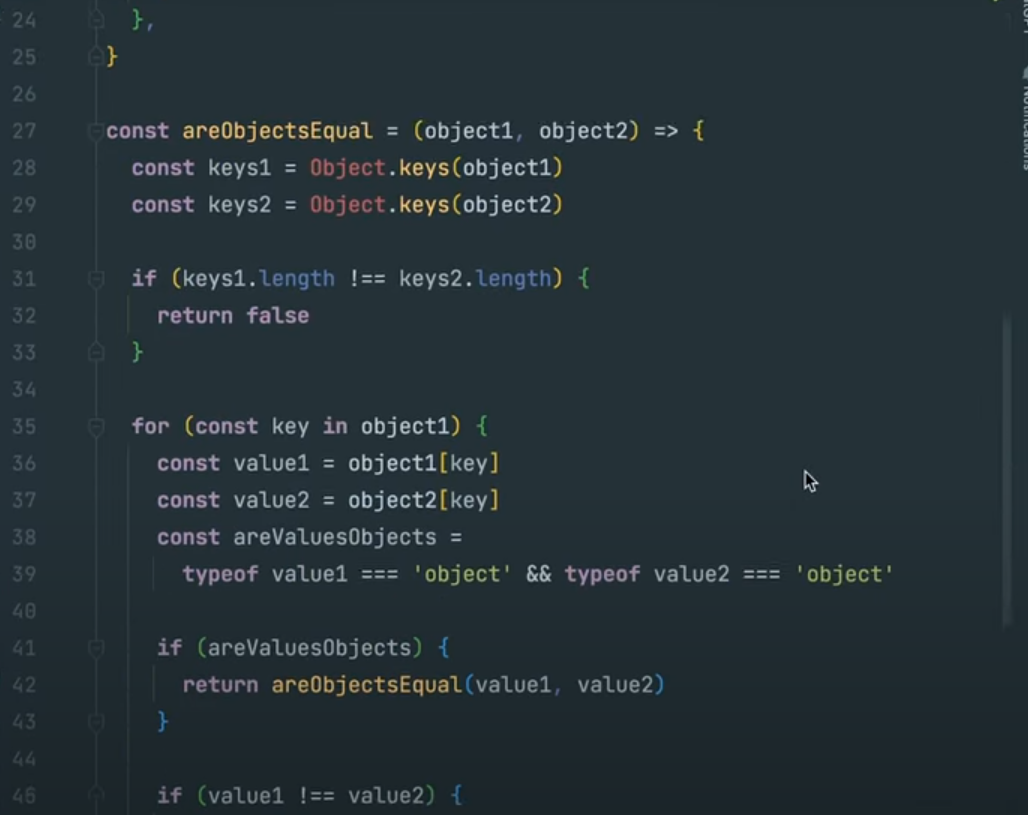
## Стрелочные функции (Arrow Function)

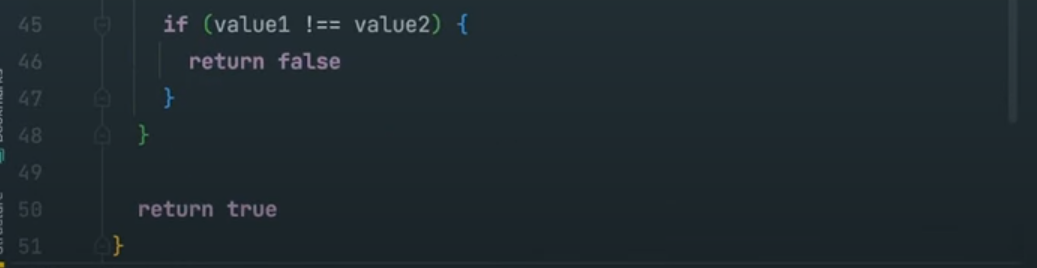


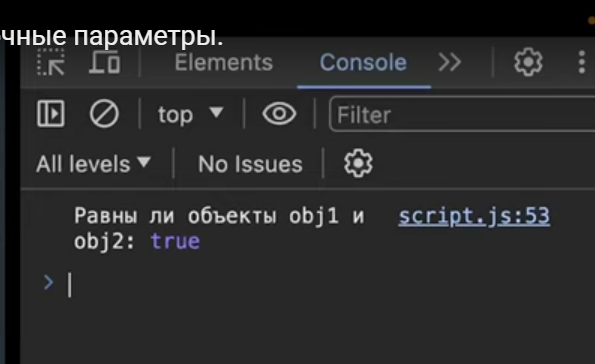
1. Нельзя вызвать раньше объявления
2. Нет доступа к не явной переменной “arguments”
3. Нет своего контекста и “this” покажет на родительскую область видимости
4. Не явный возврат из функции при простой записи “в одну строку” *(не понял до конца)*



# СРАВНЕНИЕ ДВУХ ОБЪЕКТОВ (СЛЕГКА УГЛУБЛЕННОЕ)

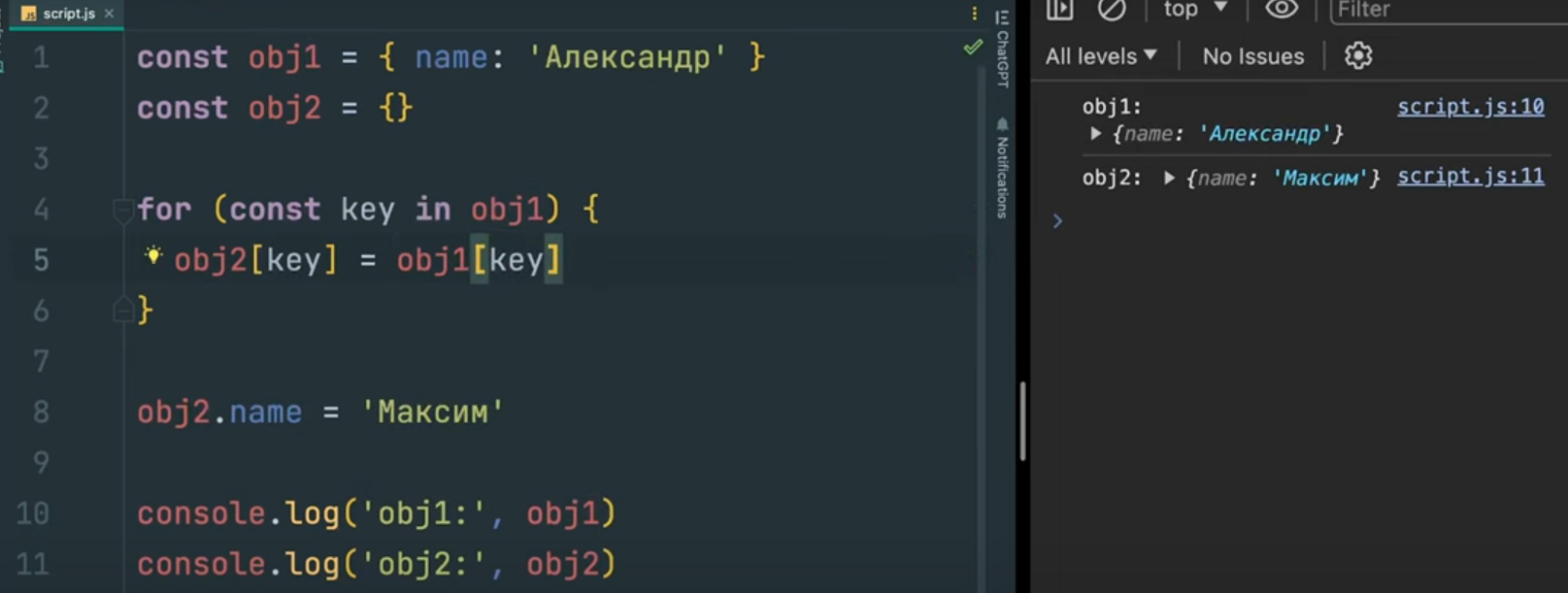






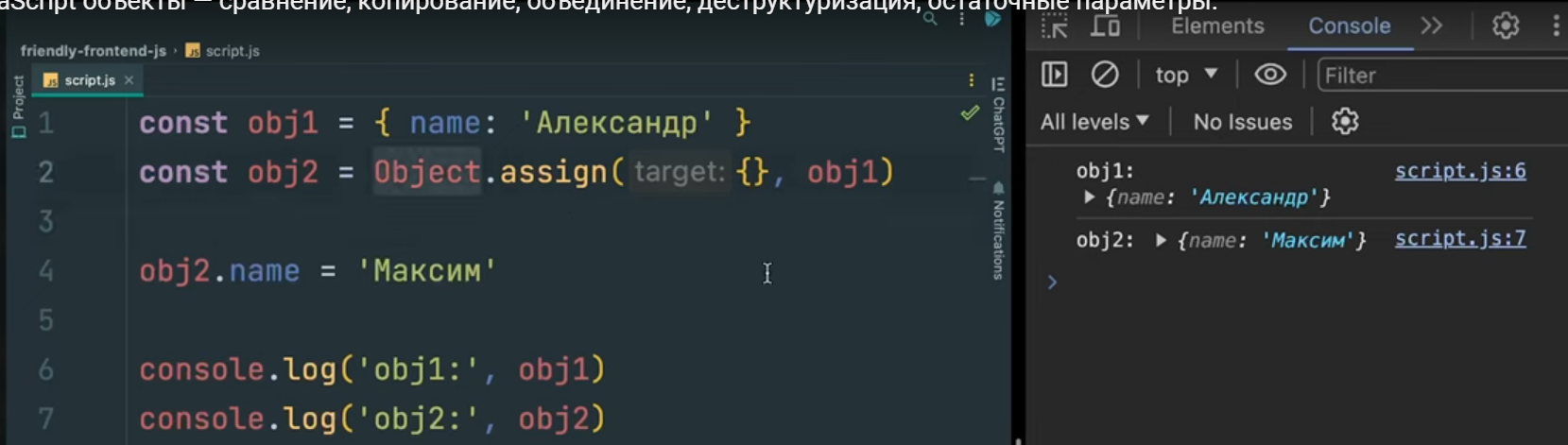
# КОПИРОВАНИЕ ОБЪЕКТА

## Долгое

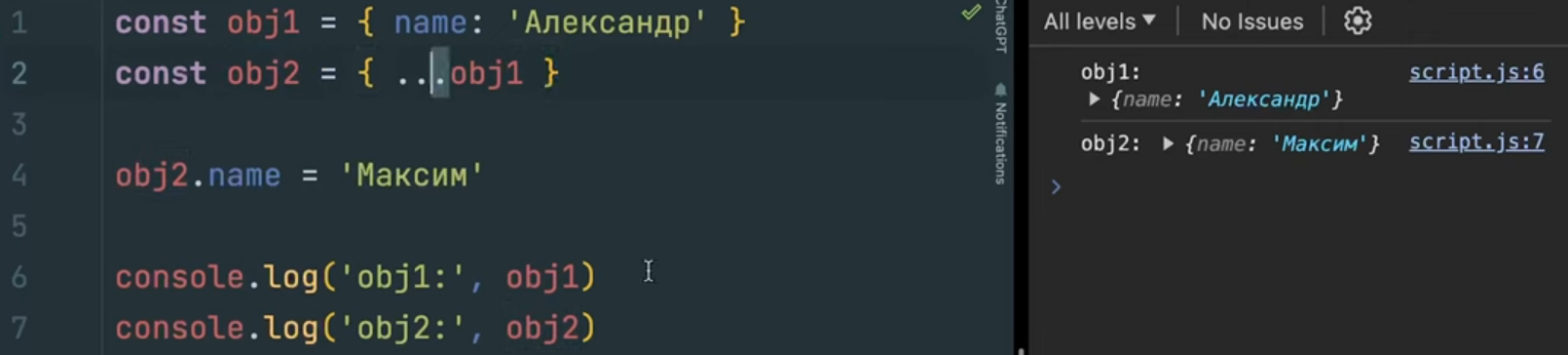


## Быстро

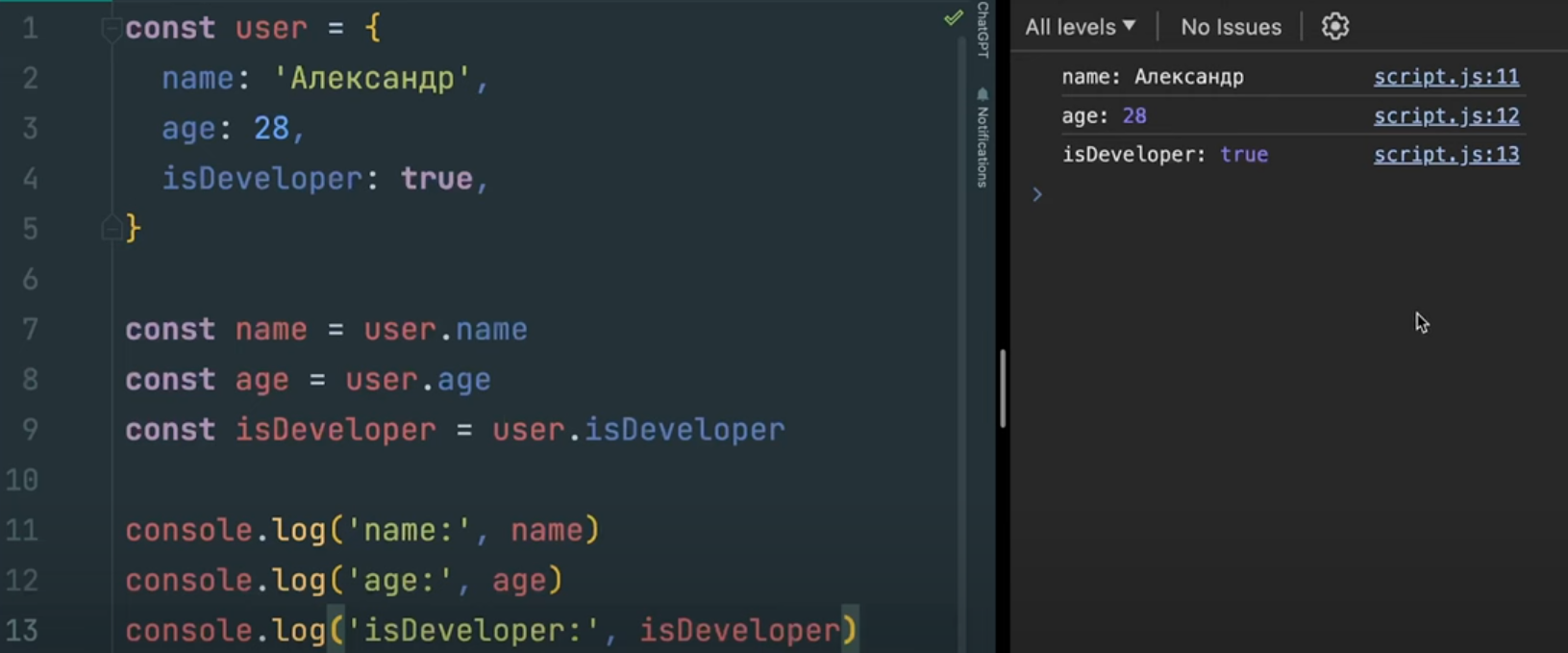
### (Object.assign)

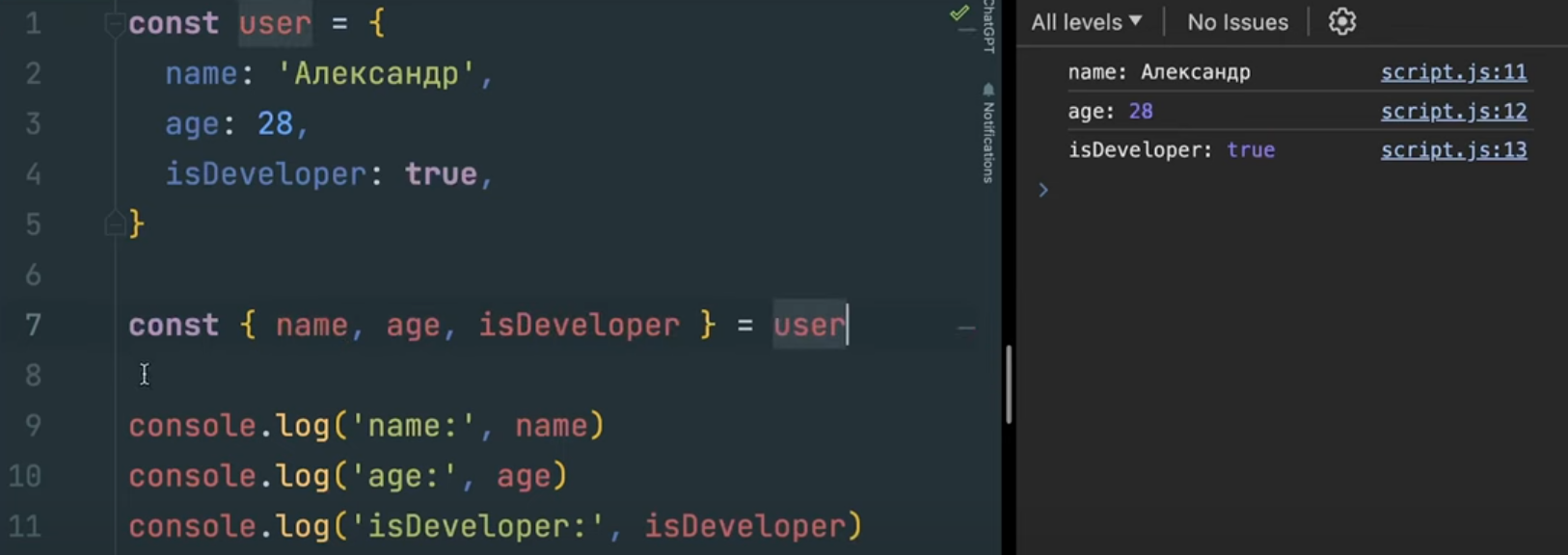


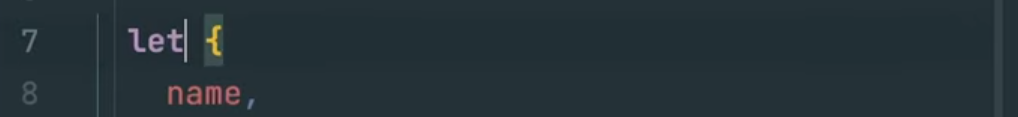
### Оператор расширения (Spread Operator)



# ДЕСТРУКТУРИЗАЦИЯ ОБЪЕКТА (№11 19:06)







Если будут одноименные свойства у объекта user, они попадут в константные переменные. Такой метод можно использовать для массивов и объектов. Можно объявлять и через “let”.

# PROMISE

Promise - специальный объект-надстройка для работы с асинхронным кодом

Promise имеет 3 состояния:

* pending - ожидание, исходное состояние
* fulfilled - выполнено успешно, получен результат
* rejected - выполнено с ошибкой

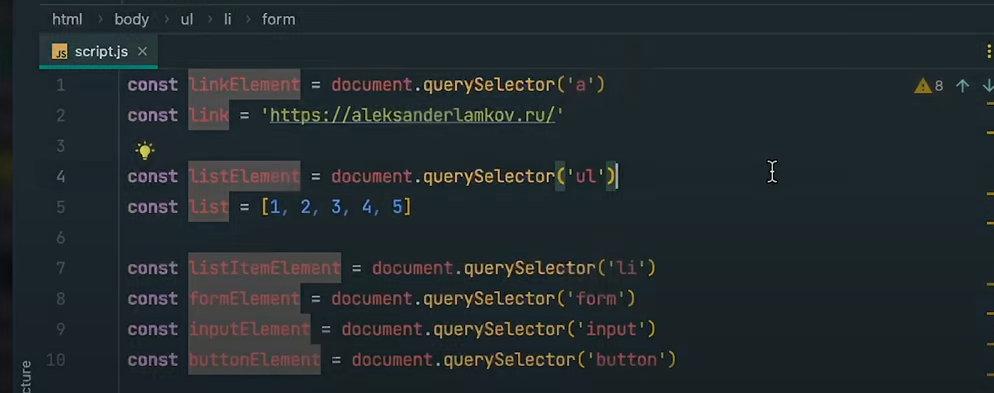
Основные методы Promise:

* then() - обрабатывает fulfilled состояние
* catch() - обрабатывает rejected состояние

# DOM

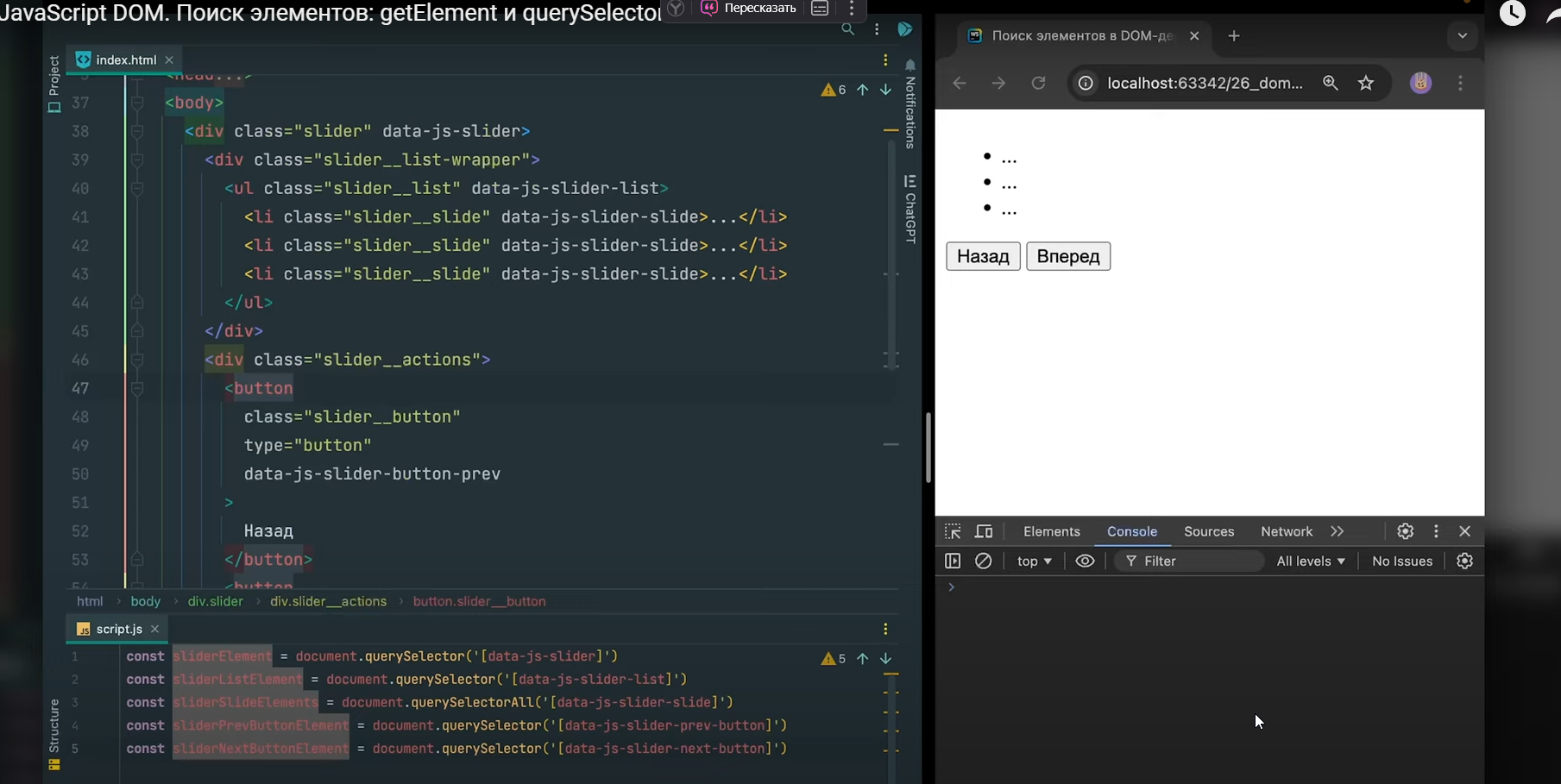


## Как лучше называть переменные, содержащие DOM-элементы (№25 11:16)





## КАК лучше находить элементы в DOM-дереве (№25 12:45)



# События клавиатуры

<https://www.toptal.com/developers/keycode> - Сервис для проверки JavaScript-данных о нажатой клавише