Java Script

NEXT

GENIUS space

План

JavaScript -що це таке

Підключення скрипта

Структура коду

Змінні

Визначення властивостей та методів

const и let

Типи данних

Вступ до JavaScript

JavaScript було створено для того, щоб “оживити веб-сторінки”.

Програми на цій мові називаються скриптами. Їх можна писати прямо на сторінці в коді HTML і вони автоматично виконуються при завантаженні сторінки.

JS використовується для роботи зі стилями і елементами сторінки, з введеними данними, сервером, базою данних і локальним сховищем

01

JavaScript - це найпопулярніша мова програмування у світі.

02

JavaScript - це мова програмування HTML-сторінок та Web.

03

JavaScript - це мова програмування, яку досить легко вивчити.

JavaScript - це високорівнева мова програмування, яка використовується для створення динамічних веб-сторінок та веб-додатків. Вона є однією з трьох основних технологій веб-розробки, разом з HTML та CSS.

Програми цією мовою називаються скриптами.
Скрипти пишуться та виконуються як простий текст.

При створенні JS називався: “LiveScript”. Назву було змінено з метою популяризаціїї цієї мови (тому, що тоді на піку популярності була Java)

На данний момент ці дві мови мають дуже мало спільного

У Front-end розробці JavaScript використовується для:

- нескладних обчислень
- перевірки та маніпуляції даними
- зберігання інформації у браузері користувача
- динамічної зміни HTML-документа
- реакціїна дії користувача
- створення інтерактивних елементів: галерей, графіків тощо.
- взаємодія з сервером та базою даних

На сьогоднішній день використовуючи JavaScript можна створювати:

- веб-застосунки використовуючи фреймворки React, Vue, Angular та інші
- бекенд-програми на Node.js
- мобільні програми використовуючи React Native або Ionic
- десктоп-додатки за допомогою Electron
- мікроконтролери з Johnny-Five та Espruino
- тощо

Щоб додати скрипт на веб-сторінку, у HTML-файлі використовується тег script, в атрибуті src якого вказуємо посилання на зовнішній JavaScript-файл.
Щоб підключити JavaScript із зовнішнього файлу:
Створіть файл із розширенням .js
Потім вкажіть шлях до файлу скрипта в атрибуті src тега script.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<title>Lesson 1</title>
</head>
<body>
<script src="js/script.js"></script>
</body>
</html>
```

При підключенні кількох JavaScript-файлів до сторінки інтерпретатор обробляє їх у тому порядку, в якому вони вказані в HTML-файлі.

Браузер завантажує та відображає HTML поступово. Якщо браузер бачить тег script, то за стандартом він повинен спочатку виконати його, і лише потім обробити решту коду HTML-файлу. Тому інтерпретатор буде послідовно обробляти файли script-1.js та script-2.js.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<title>Lesson 1</title>
</head>
<body>
<script src="js/script-1.js"></script>
<script src="js/script-2.js"></script>
</body>
</html>
```

Структура коду

Інструкції – це синтаксичні конструкції та команди, які виконують дії.

Можна писати стільки інструкцій, скільки завгодно. Інструкції можна розділяти крапкою з комою.

Коментарі можна писати як на одному рядку: починаючи його з // так і на декількох рядках, розділяючи їх за допомогою /* … */.

Зазвичай ми використовуємо коментарі для опису того, як і чому наш код працює.

Коментувати варто - загальну архітектуру, опис “високого рівня”, важливі рішення, особливо, якщо вони не є очевидним.
Варто уникати коментарів які описують, як код працює і що він робить. Це виправдано лише коли немає можливості написати простий і зрозумілий кожному код

Змінні

Змінна це “іменована частинка сховища”, в якій зберігаються дані.

Оголосити змінну можна використовую let , const або var

Використовуйте за замовчуванням const, більшість змінних будуть оголошені саме так.
Використовуйте let, якщо потрібно присвоїти змінній інше значення під час виконання скрипту.
Оголошення за допомогою var подібне до let. У більшості випадків ми можемо замінити let на var або навпаки і очікувати, що це буде працювати.
Але var – це зовсім інший звір, який походить з дуже давніх часів. Зазвичай він не використовується в сучасних скриптах, але все ще може переховуватися у старих.

Як називати змінні

Одним з важливих критеріїв хорошого коду є його читання. Наприклад, ми три місяці тому написали код, а сьогодні нам потрібно було щось змінити. Змінити властивості товару або змінити виробника.

Тому дуже важливо, щоб функції, змінні та методи легко читалися та розумілися.

Як правило, при створенні імен для строкової змінної використовують іменник.

Вам і тому, хто читатиме ваш код, має бути зрозуміло, що знаходиться в змінній за її назвою.

Використовуйте лише англійську мову при утворенні назви змінної

Числові змінні

Майже у кожному проекті прийнято такі правила.

- кількість (count)
- код чогось (code)
- розмір (size, length)
- номер (number)

Змінні з кількох слів пишуться разом.
Наведений нижче спосіб називається "верблюжою нотацією" або, по-англійськи, "camelCase".
var borderLeftWidth;

Також часто застосовується альтернативний стандарт, коли кілька слів пишуться через знак нижнього підкреслення.
var border_left_width;

Бульове значення
Логічною назвою для булевої змінної – питання з відповіддю так чи ні.

це (is)
містить (has/contain)
може (can)
повинен (shoud)
можливість (able)

Властивість

У нас з вами є властивості: зріст, вага, колір очей, тобто якісь описові характеристики. Так само і у даних є характеристики, наприклад у рядка є властивість її довжини. Синтаксис звернення до якості дуже простий.

сутність.ім'я_властивості

Метод

Метод це виклик дії, наприклад сісти чи плавати, тобто якась активна операція. Так само і дані мають свої методи, наприклад можна додати або видалити елементи з колекції, перевести рядок в різний регістр і т. д. Синтаксис виклику методу дуже схожий на звернення до властивості, але в кінці додається пара дужок.

сутность.имя_метода()

Відмінність const і let полягає в тому, що const забороняє повторне надання змінної будь-якого значення.
Буде розумно використовувати let і const так:
Використовуйте за замовчуванням const, більшість змінних будуть оголошені саме так.
Використовуйте let, якщо потрібно присвоїти змінній інше значення під час виконання скрипту.

Застаріле ключове слово "var"

Оголошення за допомогою var подібне до let. У більшості випадків ми можемо замінити let на var або навпаки і очікувати, що це буде працювати:

Але var – походить з дуже давніх часів. Зазвичай він не використовується в сучасних скриптах, але все ще може переховуватися у старих.

Типи даних

Значення в JavaScript завжди має певний тип даних. Наприклад, рядок або число.

У JavaScript є вісім основних типів даних:

- Число (number)
- BigInt
- Рядок (string)
- Булевий або логічний тип (boolean)
- Null
- Undefined
- Об’єкт (object)
- Symbol

Оператор typeof повертає тип аргументу. Це корисно, коли ми хочемо обробляти значення різних типів по-різному або просто хочемо зробити швидку перевірку. Виклик typeof x повертає рядок із назвою типу:

String

Рядки або просто текст. Рядок починається і закінчується одиночною ', або подвійними лапками". Відкриваюча і закриваюча лапки повинні бути однакові.

Boolean

Логічний тип даних, прапори стану. Має лише два значення: true та false. Наприклад на запитання чи включено світло в кімнаті можна відповісти так (true) чи ні (false).

undefined

Ще одне спеціальне значення. За умовчанням, коли змінна оголошується, але не ініціалізується, її значення не визначено, їй присвоюється невизначений.

null

Особливе значення, яке насправді означає ніщо. Використовується в тих ситуаціях, коли необхідно явно вказати відсутність значення.

Number

Цілі числа та числа з плаваючою комою. Після оголошення змінної можна ініціалізувати її числовим значенням.

BigInt

У JavaScript, тип “number” не може містити числа більші за (253-1) (це 9007199254740991), або менші за -(253-1) для від’ємних чисел. Це технічне обмеження, спричинене їхньою внутрішньою реалізацією.

Для більшості потреб цього достатньо, але бувають випадки, коли нам потрібні дійсно великі числа, наприклад, для криптографії або мікроксекундних часових міток (timestamps).

Значення з типом BigInt створюється через додавання n у кінець цілого числа:

Object
Тип object є особливим типом.

Усі інші типи називаються “примітивами”, тому що їхні значення можуть містити тільки один елемент (це може бути рядок, число, або будь-що інше). В об’єктах же зберігаються колекції даних і більш складні структури.

Symbol Символи
“Символ” являє собою унікальний ідентифікатор.

Створити символ можна за допомогою Symbol() Символи гарантовано будуть унікальними. Навіть якщо ми створюємо багато символів з однаковим описом, вони мають різні значення. Опис – це просто мітка, яка ні на що не впливає.:

Додаткові матеріали:

https://uk.javascript.info/structure
https://uk.javascript.info/variables
https://uk.javascript.info/types

https://www.freecodecamp.org/news/how-you-can-improve-your-workflow-using-the-javascript-console-bdd7823a9472

https://www.freecodecamp.org/news/javascript-naming-conventions-dos-and-don-ts-99c0e2fdd78a

https://www.w3schools.com/js/js_reserved.asp

Домашня робота

//***1***
// Робота зі змінними
// Оголосіть дві змінні: name та city.
// Присвойте значення "Іван" змінній name.
// Скопіюйте значення зі змінної name в city.
// Виведіть значення змінної city, використовуючи функцію console.log (яка повинна показати “Іван”).

//***2***
//Який буде результат виконання скрипта?
// let name = "Olga";
// console.log(`привіт ${1}`); //
// console.log(`привіт ${"name"}`); //
// console.log(`привіт ${name}`); // ?

//***3***

//Видобути число зі змінних

// let a = "5";

// let b = "13cvb";

// let c = "12.9sxdcfgv";

// вивести в консоль тип

//***4***

//Зробіть, щоб 0.1 + 0.2 = 0.3

//***5**

//Поверніть найбільше число с набору 20, 10, 50, 40

//***6**
//Поверніть випадкове число в діапазоні від 2 до 4

//***7**
//дізнатись довжину message
// const message = "Welcome to Bahamas!";

//***8**
//вивести в консоль message великими літерами

//***9**
// створити пустий об*єкт
// додати туди ім*я, вік і місто
// вивести результат в консоль
// видалити місто
// додати буль з ключем: like flowers
// вивести результат в консоль

//***10**
// За допомогою циклу “for...in” вивести в консоль ключі і значення об*єкта