Веб програмування

Основи JavaScript

Анонс

JavaScript — це мова, що відкриває нам неймовірні можливості для створення динамічних та інтерактивних вебсторінок. В цьому уроці ми зануримося у світ JavaScript та вивчимо її основи.

JavaScript дозволяє нам створювати відгуки на дії користувачів, змінювати вміст сторінок «на льоту», анімувати об’єкти, і багато іншого.

Якщо HTML відповідає за структуру сторінки, а CSS — за її стиль, то JavaScript — це мова, що додає життя та взаємодію до цього «тріо».

Сторінка 1
Вступ до JavaScript

JavaScript — це мова програмування, яка грає важливу роль у веброзробці та використовується в різних галузях для створення динамічних та інтерактивних додатків.

JavaScript був розроблений Бренданом Айком в 1995 році в компанії Netscape Communications. Початково він називався LiveScript, але пізніше його перейменували на JavaScript, для асоціювання з мовою програмування Java.
JavaScript використовується для різноманітних завдань, від покращення користувацького інтерфейсу до розробки серверних додатків:

1.Веброзробка
JavaScript додає інтерактивність, валідацію форм, анімацію та інші динамічні ефекти до вебсторінок. Наприклад, можна створити анімовані меню, слайдери та взаємодію з сервером без перезавантаження сторінки.

2. Вебзастосунки:
JavaScript використовується для створення вебзастосунків, таких як чати, соціальні мережі, інтерактивні карти тощо. Наприклад, додаток **Google Maps** використовує JavaScript для відображення і взаємодії з картами.

3. Графіка та анімація
JavaScript використовується для створення 2D та 3D-анімацій, ігор та візуалізацій. Бібліотеки, як-от **Three.js**, дозволяють створювати складні графічні ефекти просто за допомогою коду.

4. Взаємодія з API
JavaScript дозволяє взаємодіяти з різними API, такими як Facebook, Google Maps API тощо. Це дозволяє вбудовувати зміст та функціональність з інших додатків на своїй вебсторінці.

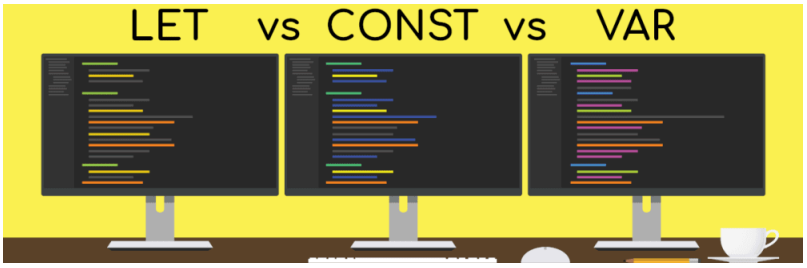
Завдання 1

Проведіть пошук щодо практичного використання JavaScript в популярних розробках.

Де ваш пошук був більш вдалим: через пошуковий запит чи з допомогою бота на кшталт ChatGPT?

Наведіть 3 найцікавіших приклади.

Змінні в JavaScript



JavaScript — це мова програмування, яка використовується для створення динамічних інтерактивних вебсторінок.

Оголошення змінних
Змінні в JavaScript використовуються для зберігання та керування даними.

Оголошення змінних за допомогою let:
let age = 25;
age = 30;

Оголошення змінних за допомогою var:
var name = "Котик";

Оголошення констант за допомогою const:
const country = "Україна";

Якщо вас цікавить відмінність оголошення змінних з допомогою **var та let, перегляньте матеріал.**

Приклад
Використання змінних та констант:

```
console.log("Привіт, мене звуть " + name + " і мені " + age + " років.");
console.log("Я з " + country);
age = 35;
name = "Alice";
console.log("Тепер мені " + age + " років.");
console.log("А мене тепер звуть " + name + ".");
```

Середовища для програмування на JavaScript

JDoodle (платформа SpiderMonkey)

Repl.it (HTML + CSS + JS)

Завдання 2

Дослідимо роботу з JavaScript у двох онлайн-середовищах
JDoodle (платформа SpiderMonkey)

Repl.it (HTML + CSS + JS)

1. Перейдіть за покликанням і розгляньте основні компоненти IDE:
    - де писати код;
    - як і де вводити дані.
2. Протестуйте код прикладу, який надається за замовчуванням.
3. Змініть приклад на виведення Hello World.
4. Спробуйте виконати додавання 2 чисел.

Напишіть короткий висновок щодо зручності роботи з кожним середовищем і які пункти завдання вам вдалося виконати.

Особливості введення-виведення в різних фреймворках

Фреймворк (англ. framework) — це набір готових інструментів, бібліотек, шаблонів та правил, які допомагають розробникам створювати програми або вебзастосунки швидше і з меншими зусиллями.

Фреймворки надають структуру та загальний набір функціональності для розробки, що допомагає уникнути дублювання коду та забезпечує більш організований підхід до роботи. JavaScript використовується в різних **фреймворках** для створення різних видів програм. Список популярних фреймворків JavaScript:

- React;
- Angular;
- Vue.js;
- Node.js.

Кожен фреймворк може мати свої **особливості** введення та виведення даних.

Наприклад, у фреймворку **Angular** для введення можна використовувати `ngModel`:

```
<input [(ngModel)]="name" placeholder="Введіть ваше ім'я">
<p>Привіт, {{ name }}!</p>
```

Для фреймворку **Svelte:**

```
<script>
  let greeting = "Світ Svelte!";
</script>
<main>
  <h1>{greeting}</h1>
</main>
```

SpiderMonkey, двигун JavaScript в браузері Firefox, також має свої можливості введення-виведення.
Наприклад:

```
print("Привіт Spudernonkey");
```

Часом навіть складно впізнати, що це все та ж мова Java Script, але насправді принципи вебпрограмування залишаються однаковими.

Завдання 3

Для того, щоб використання різноманітних фреймворків не викликало непереборних труднощів, потрібно знати, як запустити код JS, написаний у них. Для цього знадобиться середовище запуску коду різних фреймворків. Проведемо у цьому середовищі тестування традиційно першої програми **Hello World**
Щоб поєднати пари «фреймворк — код», знайдіть **онлайн-середовище,** де цей код можна протестувати для цього фреймворку.

Express.js	const express = require('express');
	const app = express();
	const port = 3000;
	app.get('/', (req, res) => {
	  res.send('Hello, World!');
	});
	app.listen(port, () => {
	  console.log(`Server listening at http://localhost:${port}`);
	});

React	import React from 'react'; import ReactDOM from 'react-dom'; ReactDOM.render(Hello, World! from React , document.getElementById('root'));
SpiderMonkey	print ("Hello World")
Node.js	console.log('Hello World');

Vue.js	<template>
	
	   Hello, World
	
	</template>
	<script>
	export default {
	  name: 'App',
	};
	</script>

Арифметичні дії в JavaScript

SpiderMonkey — це рушій JavaScript, який є одним з перших і зараз використовується в середовищі Mozilla Firefox для виконання JavaScript-коду.
Ми розглядатимемо теми цього уроку на платформі SpiderMonkey, бо це найпростіший спосіб введення-виведення і для нього є зручне онлайн-середовище.
JDoodle (платформа SpiderMonkey)

Основні арифметичні дії

Виконувати арифметичні дії в JavaScript дуже просто. Ось кілька прикладів:

```
let sum = 5 + 3;        // Додавання
let difference = 10 - 4;   // Віднімання
let product = 7 * 6;    // Множення
let quotient = 20 / 5;  // Ділення
```

Введення та виведення з використанням readline()

Для отримання введення від користувача можна використовувати функцію **readline()**:

```
let a = readline(); // Отримати введення від користувача
print("Ви ввели: " + a); // Вивести введене значення
```

Перетворення рядка на число

Щоб використовувати введене значення як число, можна використовувати функцію **parseInt()**:

```
let a = parseInt(readline()); //Отримати та перетворити значення на число
print("Ви ввели число: " + a); // Вивести введене число
```

Приклад з арифметичними діями та введенням-виведенням

```
let a = parseInt(readline()); // Отримати та перетворити значення на число
let b = parseInt(readline()); // Отримати та перетворити значення на число
let sum = a + b; // Знайти суму двох чисел
print("Сума: " + sum); // Вивести результат
```

Завдання 4

Тестування середовища програмування та виконання простих програм добре починати зі звичайного калькулятора

1. Введіть 2 числа.
2. Виконайте по черзі 4 арифметичні дії +, -, *, /.
3. Змініть код так, щоб результатом ділення було ціле число.
4. Перевірте, що буде, якщо поділити на 0.
5. Перевірте множення на великих числах.

Працюємо в середовищі **JDoodle** **(платформа SpiderMonkey).**

Напишіть, що вас здивувало в цій арифметиці або що довелося уточнювати додатково.

Умови в JavaScript

JavaScript надає різні способи використання умов для розгалуження програми за допомогою умовних операторів **if, else if** та **else.**

Оператор if

Оператор **if** дозволяє виконувати певний блок коду, якщо вказана умова вірна.
Приклад:


```
let age = 18;

if (age >= 18) {
    console.log("Ви досягли повнолітнього віку.");
}
```

Оператор else if

Якщо є кілька можливих умов, ви можете використовувати оператор **else if:**

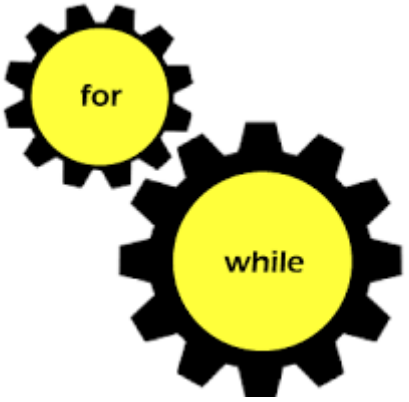
```
let time = 14;
if (time < 12) {
    console.log("Доброго ранку!");
} else if (time < 18) {
    console.log("Доброго дня!");
} else {
    console.log("Доброго вечора!");
}
```

Цикли for та while в JavaScript

У програмуванні нерідко виникає потреба виконувати певні дії багаторазово. Це може стосуватися обробки списку елементів, виконання однакових операцій з різними даними та інших завдань. Для цього в JavaScript існують цикли for та while, які допомагають автоматизувати цей процес.

Цикл for

Цикл **for** дозволяє виконувати певний блок коду певну кількість разів. Він складається з трьох частин: ініціалізації, умови виконання та кроку.
Синтаксис циклу for виглядає так:




```
for (початок; умова; крок) {
    // блок коду, який виконується на кожній ітерації
}
```

Приклад:

```
for (let i = 0; i < 5; i++) {
    print("Ітерація номер " + i);
}
```

Цикл while

Цикл **while** дозволяє виконувати блок коду, поки задана умова є істинною.

```
while (умова) {
    // блок коду, який виконується поки умова істинна
}
```

Приклад:

```
let count = 0;
while (count < 3) {
    print("Поточний лічильник: " + count);
    count++;
}
```

Тест

Перейдіть на виконання тесту і перевірте свої знання **(Тест)**

Підсумуємо

Ми засвоїли три ключові алгоритмічні структури в мові програмування JavaScript, які будуть важливими для нашого подальшого розуміння та роботи з цією мовою:

1. Лінійні алгоритми
Попрактикувалися виконувати послідовні дії та оператори, які допомагають створювати прості послідовні програми.

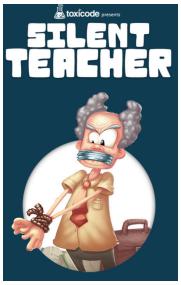
2. Умовні конструкції
Розібралися з операторами умови, які дозволяють виконувати різні дії залежно від умов, що виконуються в програмі.

3. Цикли
Розглянули використання різних типів циклів для повторення виконання коду, що допомагає нам оптимізувати та автоматизувати завдання.

JavaScript використовує ці алгоритмічні інструменти для створення динамічних та інтерактивних вебзастосунків

Розуміння цих концепцій буде важливим кроком для подальшого розвитку навичок в програмуванні на JavaScript.

Завдання



Завітайте на ресурс it.toxycode.
Перевірте, наскільки ви зрозуміли основні алгоритмічні структури та не зважайте, що перші завдання занадто легкі. Все поступово!

До речі, напишіть, скільки часу у вас пішло на це завдання, наскільки ви зрозуміли тему

Рефлексія

Моє розумне навчання

На початку ми поставили такі питання: ____

1. **Чи отримали ви відповіді на ключові питання під час роботи над темою?**

(виберіть варіант)

- ☐ Так
- ☐ Ні
- ☐ Частково так

Щодо ключових питань:

- ☐ я поки не можу сформулювати свою думку
- ☐ я дещо дізнався / дізналася
- ☐ я можу розповісти про це іншим
- ☐ я можу переконати інших, що це важливо

2. **Що допомогло в роботі? Виберіть всі варіанти, що підходять.**

- ☐ Матеріалів уроку було достатньо для виконання завдання і тестів
- ☐ Допоміг додатковий пошук в інтернеті
- ☐ Допомогло залучення штучного інтелекту
- ☐ Допомогло обговорення з іншими
- ☐ Допомогло повторення попередніх тем

3. **Яку назву ви вважаєте більш влучною для цієї теми?_____**
4. **Які з основних питань теми ви можете визначити як важливі? перелік або відповідь учня**
5. **Які з основних питань теми для вас були новими? перелік або відповідь учня**
6. **Яке з питань ви б вибрали для своєї доповіді на міжнародній конференції? Напишіть назву англійською мовою._____**
7. **Яке запитання ви б поставили з цієї теми?_____**
8. **Про що б ви розповіли друзям з цієї теми, щоб зацікавити їх?_____**
9. **Якби ви знімали фільм на цю тему, як би він називався?_____**
10. **Як вивчення даної теми може вплинути на ваше майбутнє?**
    - **Я розумію, де це можна використати у житті**
    - **Вивчене допомогло мені у виборі професії**
    - **Мені наразі складно це описати**
    - **Тема мене мало «зачепила»**