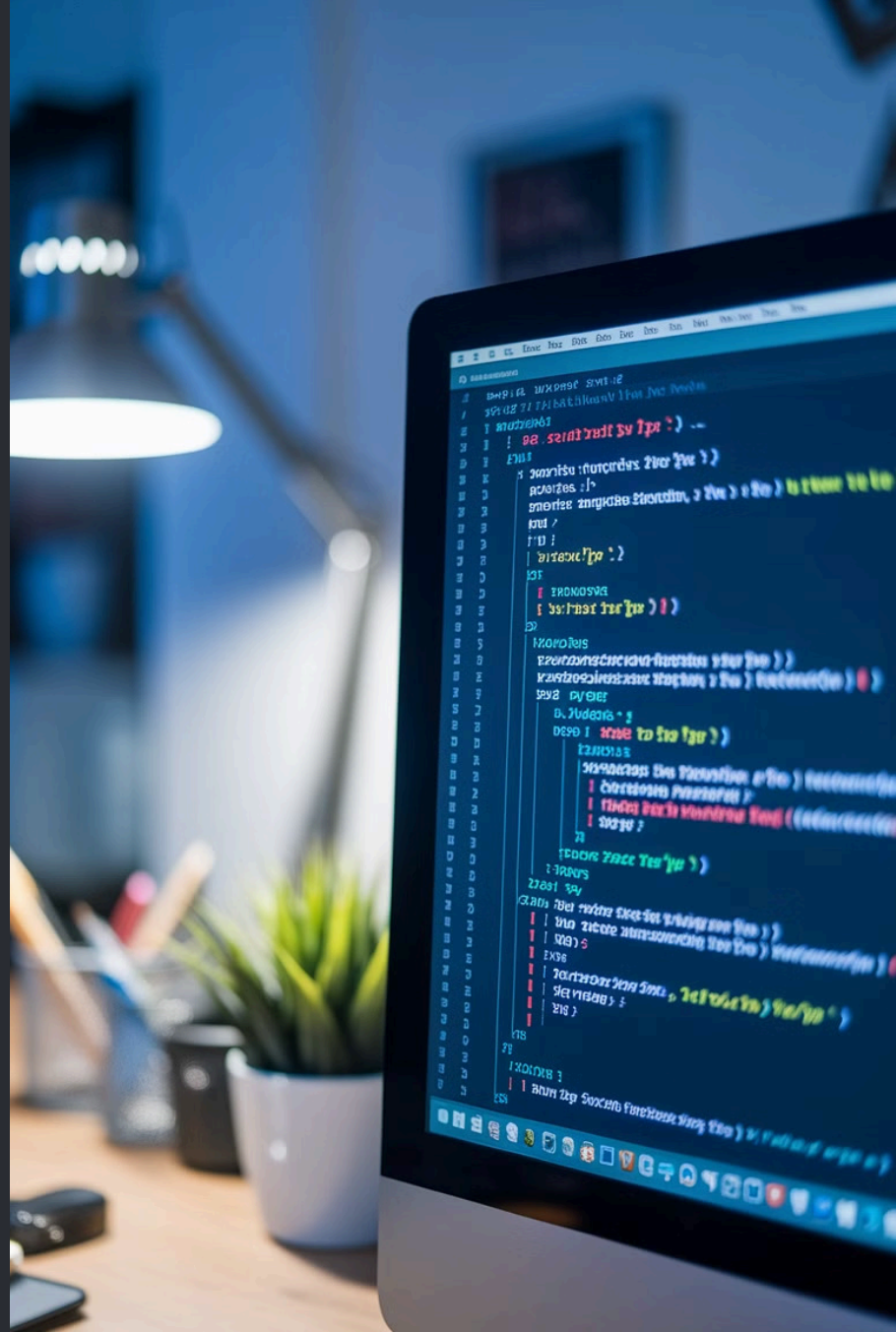
Основи JavaScript: Курс для початківців

Ласкаво просимо на курс вивчення JavaScript! Ми розглянемо основи мови програмування, яка змінила веб-розробку.



Зміст курсу

1 **Вступ до JavaScript**

Історія та значення мови у сучасній розробці.

2 **Синтаксис та основи**

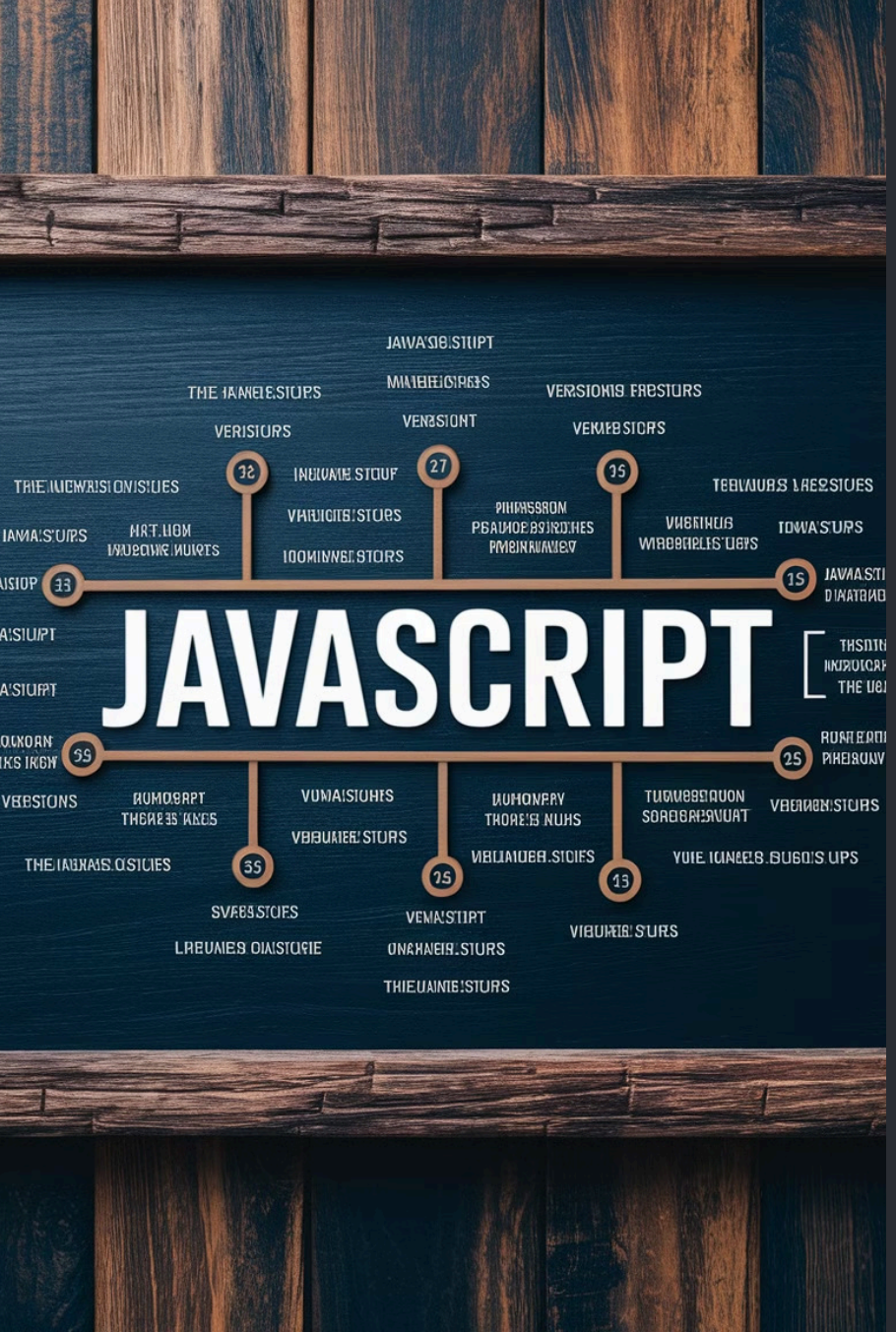
Правила написання коду та базові принципи.

3 **Змінні, типи даних та функції**

Основні будівельні блоки програм JavaScript.

4 **Структури управління**

Керування потоком виконання програми.



Вступ до JavaScript

1

Створення (1995)

Розроблена Бренданом Айком як мова сценаріїв для браузера Netscape.

2

Стандартизація

Стала однією з найпопулярніших мов програмування у світі.

3

Сучасне використання

Використовується для веб-розробки, серверних додатків та мобільних платформ.

Синтаксис JavaScript

Чутливість до регістру

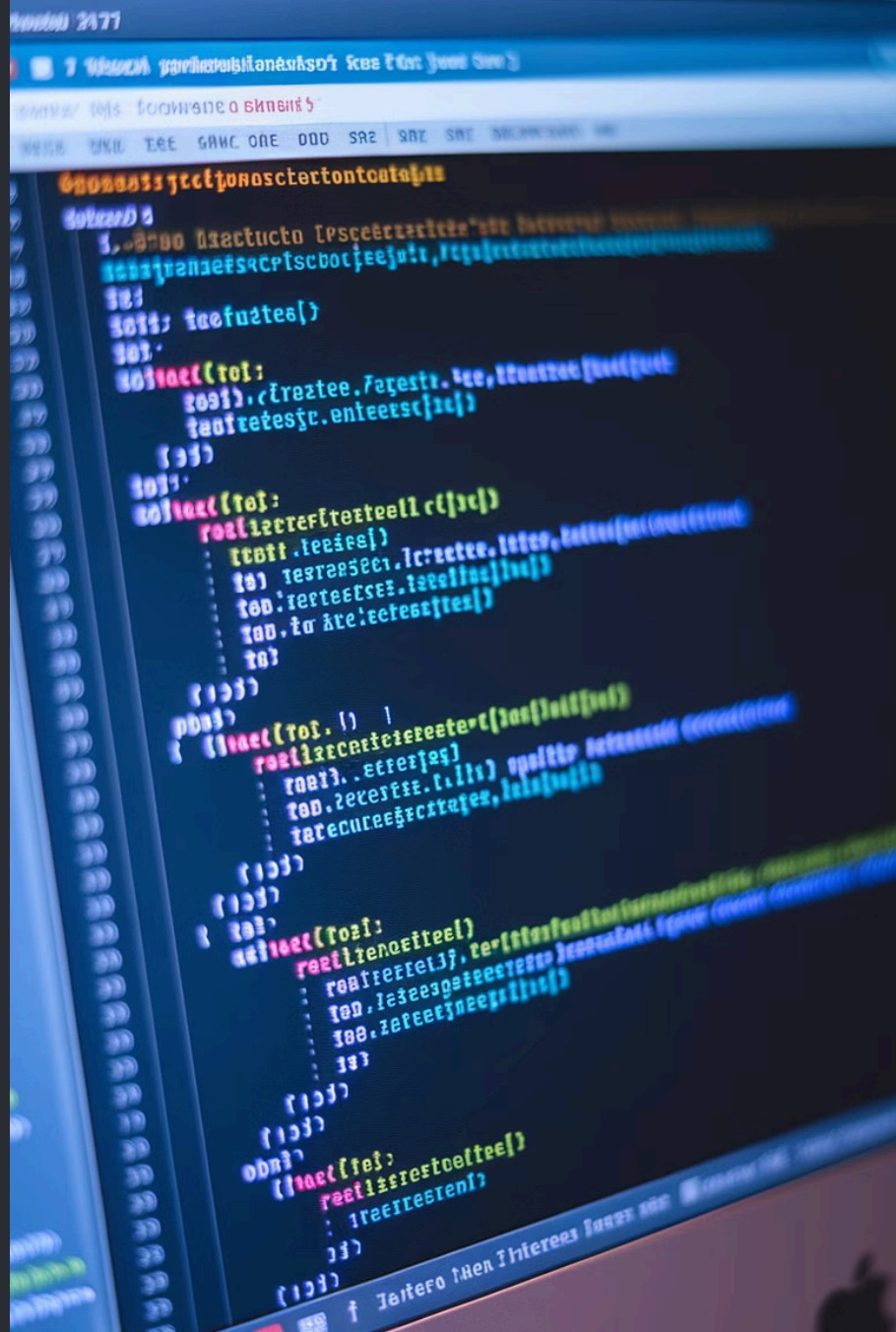
myVariable та myvariable вважаються різними змінними.

Крапка з комою

Оператори зазвичай завершуються крапкою з комою (;).

Розміщення коду

Код може бути в тегах <script> або у зовнішніх файлах .js.



Змінні в JavaScript

var

Традиційний спосіб оголошення змінних з функціональною областю видимості.

```
var myVariable = 10;
```

let

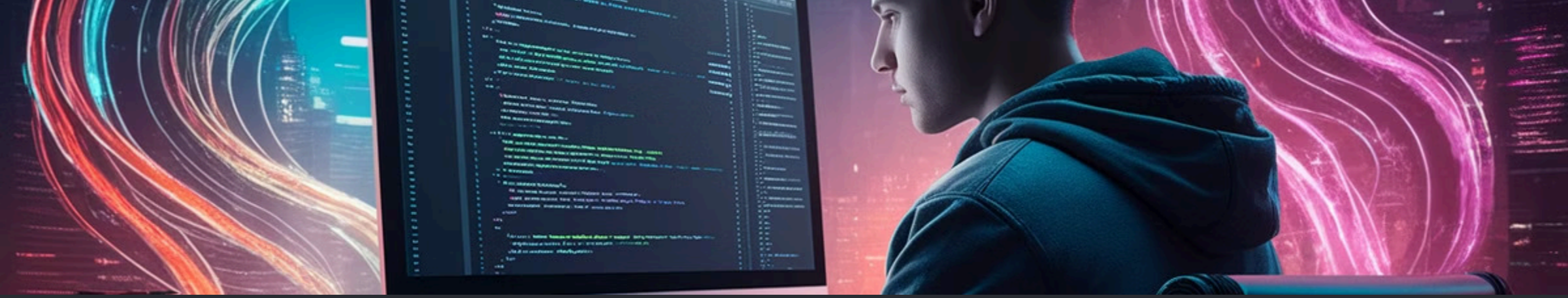
Сучасний спосіб оголошення змінних з блоковою областю видимості.

```
let name = "Аліса";
```

const

Для оголошення констант, значення яких не можна змінити.

```
const PI = 3.14;
```



Типи даних

Number

Числові значення: цілі числа та числа з плаваючою точкою (10, 3.14).

String

Текстові дані в лапках ("Привіт, світ!").

Boolean

Логічні значення true або false.

Array

Колекції значень [1, 2, 3].

Оператори в JavaScript

Порівняння

==, ===, !=, !==, >, <, >=, <= (рівність, строга рівність, нерівність)

Арифметичні

+, -, *, /, % (додавання, віднімання, множення, ділення, залишок)

Логічні

&&, ||, ! (і, або, не)

Структури управління: Умовні оператори

1	**if** Виконує код, якщо умова істинна.
2	**else if** Перевіряє додаткову умову, якщо попередня хибна.
3	**else** Виконується, якщо жодна умова не істинна.

Структури управління: Цикли

for

Виконує блок коду певну кількість разів. Ідеальний для ітерацій з відомою кількістю повторень.

for...of

Спеціально для перебору елементів масивів, рядків та інших ітерабельних об'єктів у JavaScript.

while

Перевіряє умову перед кожною ітерацією і виконує блок коду, поки умова залишається істинною.

do...while

Гарантовано виконує блок коду хоча б один раз, потім перевіряє умову для подальших ітерацій.

Цикли дозволяють автоматизувати повторювані завдання. Правильний вибір типу циклу залежить від контексту задачі та від того, чи відома кількість ітерацій заздалегідь.

```
// Приклад циклу for
for (let i = 0; i < 5; i++) {
  console.log(i); // Виведе числа від 0 до 4
}

// Приклад циклу while
let j = 0;
while (j < 5) {
  console.log(j); // Виведе числа від 0 до 4
  j++;
}
```

Приклад функції

Розглянемо структуру та використання функцій в JavaScript на прикладі:

```
// Оголошення функції
function greet(name) {
  // Тіло функції
  return "Привіт, " + name + "!";
}

// Виклик функції
const userName = "Аліса";
console.log(greet(userName)); // Вивід у консоль: Привіт, Аліса!
```

Оголошення функції

Створюємо функцію з назвою greet, яка приймає параметр name.

Тіло функції

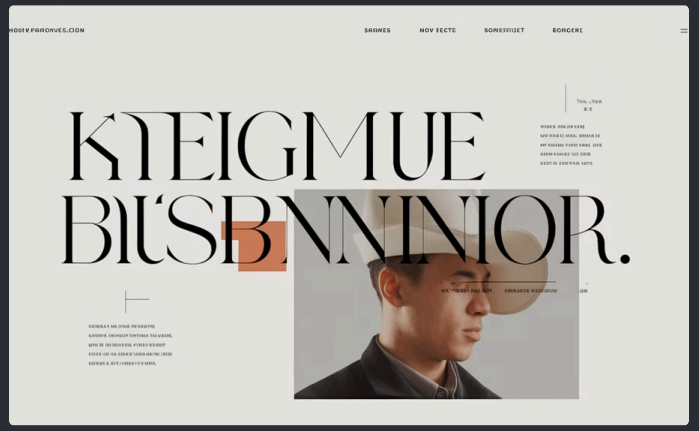
Конкатенація рядків для формування привітання, яке повертається як результат.

Виклик функції

Створюємо змінну для імені користувача і передаємо її як аргумент при виклику функції.

Функції дозволяють створювати багаторазово використовуваний код, який може приймати дані, обробляти їх та повертати результат.

Практичне застосування

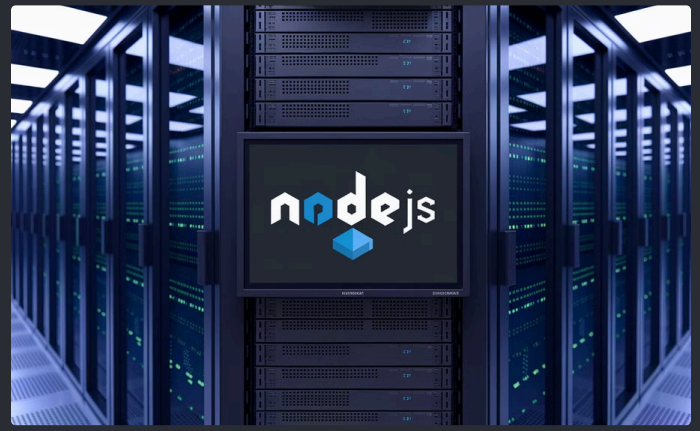


Веб-сайти

Створення інтерактивних елементів та динамічного контенту.

Мобільні додатки

Розробка кросплатформних додатків з React Native.



Серверна розробка

Створення API та серверної логіки з Node.js.