**1. Let và Const**

* let: Cho phép khai báo biến có phạm vi trong block ({}).
* const: Dùng để khai báo hằng số, không thể gán lại giá trị mới.

javascript

Sao chépChỉnh sửa

let x = 10;

x = 20; // Được phép

const y = 30;

y = 40; // Lỗi: Assignment to constant variable

**2. Arrow Function**

Cách viết gọn hơn của function thông thường.

javascript

Sao chépChỉnh sửa

const add = (a, b) => a + b;

console.log(add(5, 3)); // 8

Lưu ý:

* Không có this riêng, nó kế thừa this từ phạm vi bên ngoài.
* Không cần return nếu chỉ có một biểu thức.

**3. Template Literals**

Dùng dấu ` để viết chuỗi nhiều dòng và chèn biến dễ dàng với ${}.

javascript

Sao chépChỉnh sửa

const name = "Alice";

const greeting = `Hello, ${name}!`;

console.log(greeting); // Hello, Alice!

**4. Destructuring Assignment**

Trích xuất giá trị từ mảng hoặc object dễ dàng.

**Với mảng:**

javascript

Sao chépChỉnh sửa

const numbers = [1, 2, 3];

const [a, b, c] = numbers;

console.log(a, b, c); // 1 2 3

**Với object:**

javascript

Sao chépChỉnh sửa

const person = { name: "Bob", age: 25 };

const { name, age } = person;

console.log(name, age); // Bob 25

**5. Default Parameters**

Định nghĩa giá trị mặc định cho tham số trong hàm.

javascript

Sao chépChỉnh sửa

const greet = (name = "Guest") => `Hello, ${name}`;

console.log(greet()); // Hello, Guest

console.log(greet("Alice")); // Hello, Alice

**6. Spread & Rest Operators (...)**

* Spread: Trải các phần tử của mảng hoặc object.

javascript

Sao chépChỉnh sửa

const arr1 = [1, 2, 3];

const arr2 = [...arr1, 4, 5]; // [1, 2, 3, 4, 5]

* Rest: Gom các tham số vào một mảng.

javascript

Sao chépChỉnh sửa

const sum = (...numbers) => numbers.reduce((a, b) => a + b, 0);

console.log(sum(1, 2, 3, 4)); // 10

**7. Object Property Shorthand**

Viết gọn khi tên thuộc tính trùng với tên biến.

javascript

Sao chépChỉnh sửa

const name = "Alice";

const age = 30;

const person = { name, age };

console.log(person); // { name: "Alice", age: 30 }

**8. Enhanced Object Literals**

Có thể định nghĩa phương thức ngắn gọn trong object.

javascript

Sao chépChỉnh sửa

const user = {

name: "John",

greet() {

return `Hello, ${this.name}`;

}

};

console.log(user.greet()); // Hello, John

**9. Promises & Async/Await**

**Promises:**

javascript

Sao chépChỉnh sửa

const fetchData = () => {

return new Promise((resolve, reject) => {

setTimeout(() => resolve("Data fetched"), 1000);

});

};

fetchData().then(console.log); // Data fetched

**Async/Await:**

javascript

Sao chépChỉnh sửa

const fetchDataAsync = async () => {

const data = await fetchData();

console.log(data);

};

fetchDataAsync(); // Data fetched

**10. Modules (Import/Export)**

Cách tổ chức code tốt hơn với module.

**math.js**

javascript

Sao chépChỉnh sửa

export const add = (a, b) => a + b;

export const subtract = (a, b) => a - b;

**main.js**

javascript

Sao chépChỉnh sửa

import { add, subtract } from "./math.js";

console.log(add(5, 3)); // 8

**11. Optional Chaining (?.)**

Tránh lỗi khi truy cập thuộc tính không tồn tại.

javascript

Sao chépChỉnh sửa

const user = { profile: { name: "Alice" } };

console.log(user.profile?.name); // Alice

console.log(user.contact?.phone); // undefined (không lỗi)

**12. Nullish Coalescing (??)**

Chỉ lấy giá trị mặc định nếu biến là null hoặc undefined.

javascript

Sao chépChỉnh sửa

const value = null;

const result = value ?? "Default Value";

console.log(result); // "Default Value"