

Sao chép và So sánh Object trong JavaScript

Trong JavaScript, việc sao chép và so sánh Object không đơn giản như các kiểu dữ liệu nguyên thủy. Bài viết này sẽ giúp bạn hiểu rõ cách thực hiện sao chép nông (shallow copy), sao chép sâu (deep copy) và so sánh Object một cách chính xác. 🔍

1. Sao chép Object

Sao chép nông (Shallow Copy)

Sao chép nông chỉ sao chép các thuộc tính cấp một. Nếu Object có thuộc tính là một Object khác, nó vẫn giữ tham chiếu đến Object gốc.

Cách 1: Dùng `Object.assign()`

```
const obj = { name: "Khang" };
const copy = Object.assign({}, obj);
copy.name = "ThayKhangJS";
console.log(obj.name); // "Khang"
```

Cách 2: Dùng Spread Operator (`...`)

```
const obj = { name: "Khang" };
const copy = { ...obj };
copy.name = "ThayKhangJS";
console.log(obj.name); // "Khang"
```


 **Lưu ý:** Nếu Object chứa thuộc tính là một Object khác, nó vẫn giữ tham chiếu đến Object gốc.

Sao chép sâu (Deep Copy)

Sao chép sâu giúp tạo một bản sao độc lập, không giữ liên kết với Object gốc.


Cách 1: Dùng `JSON.parse()` và `JSON.stringify()`

```
const obj = { name: "Khang", info: { age: 25 } };
const copy = JSON.parse(JSON.stringify(obj));
copy.info.age = 30;
console.log(obj.info.age); // 25
```

 **Hạn chế:** Không sao chép được Function, `undefined`, `Symbol`.

Cách 2: Dùng `structuredClone()` (ES2021+)

```
const obj = { name: "Khang", info: { age: 25 } };
const copy = structuredClone(obj);
copy.info.age = 30;
console.log(obj.info.age); // 25
```

 **Ưu điểm:** Sao chép đầy đủ Object mà không mất dữ liệu.

2. So sánh Object



Trong JavaScript, khi so sánh hai Object, trình thông dịch sẽ kiểm tra **tham chiếu bộ nhớ** thay vì giá trị bên trong.

 **Toán tử `===` không hoạt động như mong đợi**

```
const obj1 = { name: "Khang" };
const obj2 = { name: "Khang" };

console.log(obj1 === obj2); //  false (do khác địa chỉ ô nhớ)
```


 **Sử dụng `Object.is()` để kiểm tra giá trị đặc biệt**


```
console.log(Object.is(NaN, NaN)); //  true
console.log(Object.is(+0, -0)); //  false
```

 `Object.is()` hữu ích hơn `===` khi so sánh các giá trị đặc biệt như `NaN` hoặc `+0` / `-0`.

 **So sánh Object bằng `JSON.stringify()` (không nên lạm dụng)**

```
const obj1 = { name: "Khang" };
const obj2 = { name: "Khang" };

console.log(JSON.stringify(obj1) === JSON.stringify(obj2)); //  true
```

 **Nhược điểm:** Không so sánh được các Object chứa Function, `undefined`, `Symbol`.

Tóm tắt nhanh

 **Sao chép nông (Shallow Copy):** `Object.assign()`, Spread Operator (`...`)

 **Sao chép sâu (Deep Copy):** `JSON.parse(JSON.stringify())`, `structuredClone()`

 **So sánh Object:** Dùng `===`, `Object.is()`, `JSON.stringify()` tùy trường hợp.

 **Hiểu rõ cách sao chép và so sánh Object sẽ giúp bạn tránh lỗi trong lập trình!** 