III. Duyệt và Xử Lý Object

Duyệt Object với `for...in`

`for...in` là vòng lặp dùng để duyệt qua tất cả các thuộc tính enumerable (liệt kê được) của một Object.

```
const user = {
  name: "Alice",
  age: 25,
  email: "alice@example.com"
};

for (let key in user) {
  console.log(`${key}: ${user[key]}`);
}
```

Lưu ý: `for...in` cũng lặp qua các thuộc tính kế thừa từ prototype. Nếu chỉ muốn lấy các thuộc tính riêng của Object, bạn nên dùng `hasOwnProperty()`:

```
for (let key in user) {
  if (user.hasOwnProperty(key)) {
    console.log(`${key}: ${user[key]}`);
  }
}
```

Các phương thức làm việc với Object

`Object.keys()`

Trả về một mảng chứa tất cả các key của Object.

```
const user = { name: "Alice", age: 25, email: "alice@example.com" };
const keys = Object.keys(user);
console.log(keys); // ["name", "age", "email"]
```

`Object.values()`

Trả về một mảng chứa tất cả các giá trị của Object.

```
const values = Object.values(user);
console.log(values); // ["Alice", 25, "alice@example.com"]
```

`Object.entries()`

Trả về một mảng của các cặp `[key, value]` trong Object.

```
const entries = Object.entries(user);
console.log(entries);
// [["name", "Alice"], ["age", 25], ["email", "alice@example.com"]]
```

`Object.assign()`

Dùng để sao chép thuộc tính từ một hoặc nhiều Object khác vào Object đích.

```
const target = { a: 1 };
const source = { b: 2, c: 3 };
const result = Object.assign(target, source);
console.log(result); // { a: 1, b: 2, c: 3 }
```

Lưu ý: `Object.assign()` thực hiện shallow copy, nghĩa là chỉ sao chép các giá trị tham chiếu, không phải sao chép sâu.

```
const obj1 = { nested: { key: "value" } };
const obj2 = Object.assign({}, obj1);
obj2.nested.key = "new value";
console.log(obj1.nested.key); // "new value" (vì cùng tham chiếu)
```