1. Con số không có ý nghĩa rõ ràng

Ví dụ:

*// Bad practice*

for (let i = 0; i < 60; i++) {

*// do something*

}

*// Good practice*

const MINUTES\_OF\_THE\_HOUR = 60;

for (let i = 0; i < MINUTES\_OF\_THE\_HOUR; i++) {

*// do something*

}

1. Tráng dùng nhiều vòng lặp lồng nhau
2. Không phụ thuộc vào command

Ví dụ:

Bad practice:

Const int a = 60 // số phút của 1 giờ

Good practice:

Const int MINUTES\_OF\_THE\_HOUR = 60

1. Tránh viết function quá dài, mỗi function chỉ nên thực hiện 1 chức năng

Ví dụ:

*// Bad practice*

const addSub = (a, b) => {

*// add*

const addition = a + b;

*// sub*

const sub = a - b;

*// returning as a string*

return `${addition}${sub}`;

};

*// Good practice*

*// add*

const add = (a, b) => {

return a + b;

};

*// sub*

const sub = (a, b) => {

return a - b;

};

1. Tránh lặp code
2. Đặt tên biến theo chuẩn camel case

Tên bắt đầu bằng chữ cái viết thường và tất cả chữ cái đầu tiên của những từ tiếp theo sẽ được viết hoa (áp dụng cho biến, hàm, các định danh khác ).

1. Đặt tên có ý nghĩa
2. Ưu tiên tên mô tả chi tết hơn ngắn gọn
3. Sử dụng một động từ nhất quán cho mỗi khái niệm
4. Viết hoa các giá trị const

11. tên có ý nghĩa

Sử dụng tên tiết lộ mục đích

Ví dụ:

Tránh tên với thông tin sai lệch

Tên phải phân biệt được ý nghĩa

Sử dụng tên có thể tìm kiếm

Tránh những ký tự mã hoá

12. hàm

Hàm nên chia thành những chức năng nhỏ, để có thể tái sử dụng

Mỗi hàm nên thực hiện 1 chức năng

Hàm không có tác dụng phụ

Không lặp lại

Link:

<https://hackernoon.com/how-to-write-clean-code-d557d998bb08>

<https://techmaster.vn/posts/36509/13-quy-uoc-can-thiet-de-viet-clean-code>