1. Variable and Function names

1.1 Use intention-revealing names

Bad Code

JAVASCRIPT

```javascript
const a = "Rahul";
const b = "Attuluri";
console.log(a);
console.log(b);
```

Clean Code

JAVASCRIPT

```javascript
const firstName = "Rahul";
const lastName = "Attuluri";
console.log(firstName);
console.log(lastName);
```

1.2 Make your variable names easy to pronounce

Bad Code

JAVASCRIPT

```javascript
let fName, lName;
let cntr;
let full = false;
if (cart.size > 100) {
 full = true;
}
```

Clean Code

JAVASCRIPT

```javascript
let firstName, lastName;
let counter;
const maxCartSize = 100;
const isFull = cart.size > maxCartSize;
```

2. Better Functions

2.1 Less arguments are better

Bad Code

JAVASCRIPT

```javascript
function Circle(x, y, radius) {
  this.x = x;
  this.y = y;
  this.radius = radius;
```

```
    this.radius = radius;
}
```

Clean Code

```
function Circle(center, radius) {
  this.x = center.x;
  this.y = center.y;
  this.radius = radius;
}
```

2.2 Use Arrow Functions when they make code cleaner

Bad Code

```
const count = 0;
function incrementCount(num) {
 return num + 1;
}
```

Clean Code

```
const count = 0;
const incrementCount = (num) => num + 1;
```

2.3 Use Async await for asynchronous code

Bad Code

```
myPromise
  .then(() => {
    return func1();
  })
  .then(() => {
    return func2();
  });
```

Clean Code

```
const doAllTasks = async () => {
  await myPromise;
  await func1();
  await func2();
};
doAllTasks();
```

3. Comments

3.1 Noise comments are bad

Bad Code

JAVASCRIPT

```javascript
/** The day of the month.
    The week of a year.
*/
let dayOfMonth, weekOfYear;
```

3.2 If code is readable you don't need comments

Bad Code

JAVASCRIPT

```javascript
// Check to see if the employee is eligible for full benefits
if (employee.workingHours > 100 && employeeAge > 65){
    isEligibleForFullBenefits();
}
```

Clean Code

JAVASCRIPT

```javascript
if (employee.workingHours > 100 && employeeAge > 65){
    isEligibleForFullBenefits();
}
```

4. Other code guidelines

1. There are a lot more, that you can do to identify and avoid bad code.

2. Below is a list of some code smells and anti-patterns to avoid.

4.1 Remove Dead code

Dead code is just as bad as duplicate code. There's no reason to keep it in your codebase. If it's not being called, get rid of it!

Bad Code

JAVASCRIPT

```javascript
const x = 30;
const y = 25;
function myFunction() {
  if (x > 18) {
    console.log(x);
  }
}
myFunction();
```

Clean Code

JAVASCRIPT

```javascript
const x = 30;
function myFunction() {
  if (x > 18) {
    console.log(x);
  }
}
myFunction();
```

MARKED AS COMPLETE

Submit Feedback