Guide on writing clean and readable code

Clean code is a term used to describe computer code that is easy to read, understand, and maintain. Here are some tips on writing clean and readable code:

1. Follow a consistent coding style and formatting guidelines.
2. Use meaningful and descriptive variable, function, and class names.
3. Break down complex tasks into smaller, more manageable functions or methods.
4. Write comments to explain the purpose and logic of your code.
5. Keep the code modular and reusable by avoiding duplication.
6. Ensure a clear flow of execution by organizing your code in a logical order.
7. Write unit tests to verify the correctness of your code.
8. Use version control to track changes and collaborate with others.
9. Continuously refactor and improve your code to eliminate redundancy and improve readability.

Effectiveness, efficiency and Simplicity

Remember, writing clean code is an ongoing process that requires attention to detail and continuous improvement.

1. Format and Syntax: Keep Functions and Methods Short, Consistent Formatting and Indentation, Use Meaningful Whitespace.
2. Re-usability: Minimize code duplication
3. Comments: Code should be self-explanatory whenever possible
   Clear flow of exectuion
4. Naming Variable: Meaningful Variable and Function Names, avoid global variables
   DRY (Don't Repeat Yourself) Principle

Functions

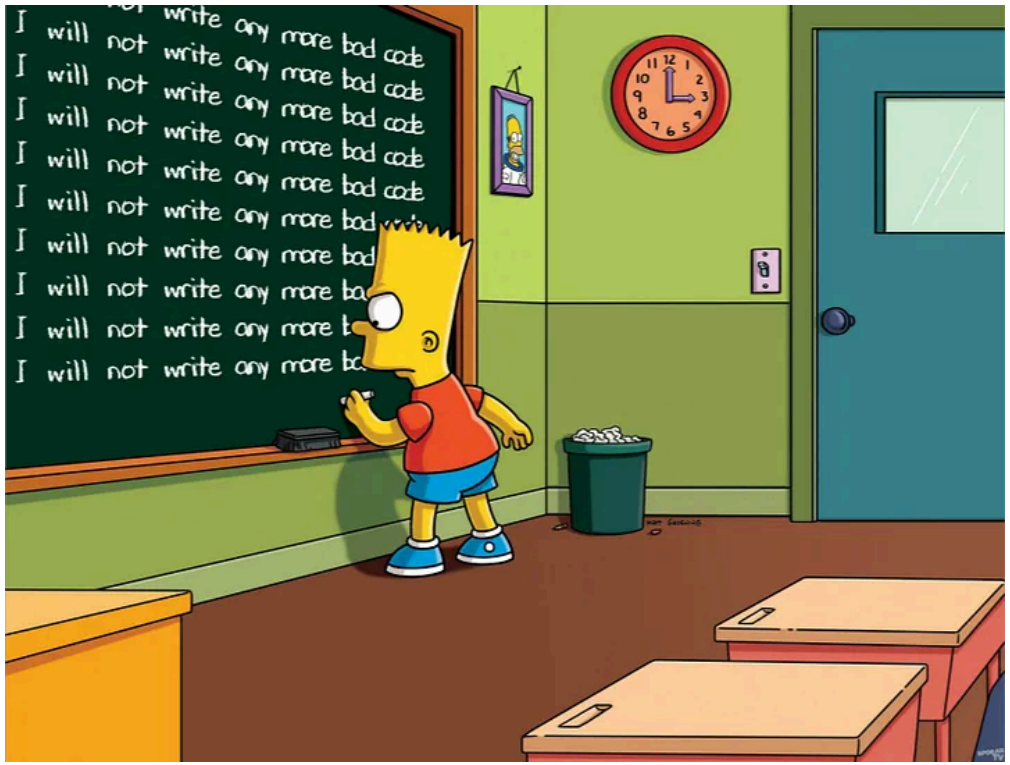
```
function(...) {
  // high level of abstraction

  // intermediate level of abstraction

  // low level of abstraction

}
```

Simplicity: guidelines we can follow

- Can you easily understand what the program does at each line?
- Do functions and variables have names that clearly represent their responsibilities?
- Is the code indented correctly and spaced with the same format all along the codebase?
- Is there any documentation available for the code? Are comments used to explain complex parts of the program?
- How quick can you identify in which part of the codebase are certain features of the program? Can you delete/add new features without the need of modifying many other parts of the code?
- Does the code follow a modular approach, with different features separated in components?
- Is code reused when possible?

Comments

- Use comments to highlight assumptions or restrictions, to record any complex or non-obvious code logic, and explain the goals and usefulness of complicated algorithms.
- good idea to include comments in your code because it makes it easier to comprehend and maintain
- Effectively placed comments


I will not write any more bad code
I will not write any more bad code
I will not write any more bad code
I will not write any more bad code
I will not write any more bad code
I will not write any more bad
I will not write any more bad
I will not write any more b
I will not write any more b