---

13 Clean Code Principles :

Writing *clean code* is more than just making your code work — it's about writing code that is readable, maintainable, and understandable for other developers. Many top software companies like Google, Microsoft, and Amazon follow certain clean code principles to ensure high-quality code.

Here are **13 essential clean code principles** that professionals follow:

The 13 Clean Code Principles:

1. **Meaningful Names**: Use descriptive variable and function names that reflect their purpose.
2. **Small Functions**: Functions should be short and do one thing only.
3. **Single Responsibility Principle (SRP)**: Each class/function should have only one reason to change.
4. **Don't Repeat Yourself (DRY)**: Avoid code duplication.
5. **Write Code for Humans**: Code should be easy to read and understand.
6. **Avoid Comments (When Possible)**: Let your code be self-explanatory. Use comments only when necessary.
7. **Consistent Formatting**: Use a clear and uniform style for indentation, spaces, and braces.
8. **Avoid Magic Numbers/Strings**: Use named constants instead of hardcoded values.
9. **Handle Errors Gracefully**: Use proper error handling to avoid unexpected crashes.
10. **Write Tests**: Create unit tests to ensure the reliability of your code.
11. **Minimize Dependencies**: Avoid unnecessary reliance on external modules.

12. **Prefer Composition Over Inheritance**: Use composition for better flexibility.

13. **Refactor Often**: Continuously improve and clean up your code without changing its behavior.

Messy Code Example (Not Clean):

```
def c(p, t):
    if t == 1:
        print("Hi " + p)
    elif t == 2:
        print("Bye " + p)
    else:
        print("Wrong input")
```

This code has:

- Unclear function and variable names.
- Magic numbers.
- No error handling.
- Poor readability.

Cleaned Up Version (Applying Clean Code Principles):

```
def greet_user(name: str, greeting_type: str):
    GREETINGS = {
        "hello": f"Hi {name}",
        "goodbye": f"Bye {name}"
    }

    message = GREETINGS.get(greeting_type.lower(), "Invalid greeting type")
    print(message)
```

Improvements:

- Clear function and variable names.
- Removed magic numbers by using a dictionary.
- Better error message for invalid input.
- Code is easier to read and extend.

Conclusion

Clean code is not just about style — it's about writing code that makes sense to *others* and to *your future self*. Applying these 13 principles helps teams collaborate better and build reliable software. Start small, refactor regularly, and always write code like the next person reading it is someone who has to fix your bugs!
