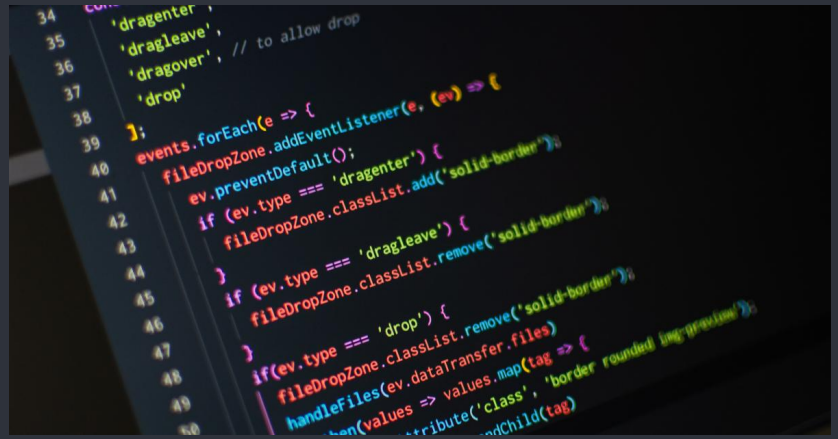
index.html

home.php

Clean Code{

[Código Limpo]

}



Jessica Estuqui

Clean Code

[Definição e Importância]

- Prática de escrever código fonte de forma clara, concisa e compreensível;
- Conjunto de diretrizes e boas práticas visando facilitar a leitura, manutenção e colaboração do desenvolvimento de software;
- Adotar o clean code torna a colaboração e a manutenção mais eficientes;
- Torna-se menos propenso a erros;
- Maior estabilidade do software;
- Melhor experiência para o usuário.

Clean Code

[Princípios Fundamentais]

- SOLID - Single Responsibility Principle (Princípio da Responsabilidade Única): uma classe deve ter uma única responsabilidade;

- SOLID OPEN/CLOSE Principle (Princípio Aberto/Fechado): classes e módulos abertos para extensão e fechados para modificação;

- SOLID - Liskov Substitution Principle (Princípio da Substituição de Liskov): estabelece que objetos de uma subclasse devem poder ser substituídos por objetos da classe;

Clean Code

[Princípios Fundamentais]

- SOLID - Interface Segregation Principle (Princípio da Segregação de Interfaces): sugere que uma classe não deve ser forçada a implementar interfaces que não utiliza;

- SOLID - Dependency Inversion Principle (Princípio da Inversão de Dependência): módulos de alto nível não devem depender de módulos de baixo nível.

- Aplicando esses princípios → Código modular, extensível e fácil manutenção → Contribui para a qualidade do projeto.

Clean Code

[Nomenclaturas e Convenções]

- Nomenclatura adequada → Código legível e compreensível;
- Nome das variáveis, funções e classes → Nomes que reflitam seu propósito e função;
- Código auto explicativo ao invés de comentários excessivos;
- Aplicando essas práticas → Colaboração eficaz e construção de sistemas coesos;

Clean Code

[Funções e Métodos]

- Funções curtas e coesas → facilita a depuração e leitura;
- Evitar alterações em variáveis globais e retornos múltiplos;
- Alinhamento e indentação;
- Ao adotar essa práticas → produção de códigos modulares, testáveis e de fácil manutenção;

Clean Code

[Comentários e Documentação]

- Quando usar comentários → Explicar partes do código que não são óbvias;
- Priorizar código auto explicativo através de nomenclatura clara e estrutura lógica;
- Documentar API’s e Interfaces é essencial;
- Revisão e atualização da documentação;
- Aplicando essas práticas → Garante que o conhecimento sobre o código seja preservado, facilitando a colaboração e manutenção.

Livro: Clean Code

Autor: Robert Cecil Martin {


```
1
2
3
4
5
6
7
8
9
10
11
12
13
14  }
```