Clean Code

Writing Better Software

emerson@paduan.pro.br

"Any fool can write code that a computer can understand. Good programmers write code that humans can understand."

Martin Fowler

emerson@paduan.pro.br

Origem

As técnicas do Clean Code apareceram pela primeira vez no livro “Clean Code: A Handbook of Agile Software Craftsmanship“, lançado em 2008, e escrito por Robert C. Martin, conhecido na comunidade como Uncle Bob.

O que é Clean Code?

Um código fácil de ser lido e entendido por outros. Também é um código fácil de se dar manutenção.
Estabelece um conjunto de boas práticas e orientações sobre como desenvolver códigos.

Importância

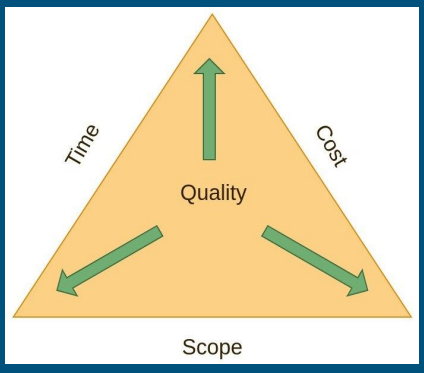
A proporção de leitura para escrita do código é de 10:1

Robert C. Martin

emerson@paduan.pro.br

Qualidade

O Código Limpo se enquadra nos critérios de Qualidade do Triângulo da Qualidade, uma variante do que é mais comumente conhecido como Triângulo de Gerenciamento de Projetos.



emerson@paduan.pro.br

O problema da complexidade

O que é complexidade?

Complexidade é qualquer coisa relacionada à estrutura de um sistema de software que torna difícil entender e modificar o sistema.

Causas da complexidade?

1. Dependencia
2. Obscuridade

Quais os sintomas?

1. Problema do desencadeamento da alteração
2. Problema da carga cognitiva
3. Problema do gerenciamento do desconhecido

emerson@paduan.pro.br

O problema da complexidade

Causas da complexidade?

1. Dependencia
2. Obscuridade

Como combater

Metodologias Ágeis:

Um dos valores do manifesto ágil diz:

"Indivíduos e interações sobre processos e ferramentas".

emerson@paduan.pro.br

Code smells

emerson@paduan.pro.br

Code Smells

Code Smells são alguns sintomas que podemos identificar e que nos remetem a uma má aplicação do Clean Code de uma forma geral.

emerson@paduan.pro.br

Code Smells

Rigidez

Seu software é difícil de mudar. Qualquer mudança, por mínima que seja, causa uma cascata de outras mudanças.

Fragilidade

Uma simples mudança quebra seu software em diversos locais. É o famosos "cobre o pé, descobre a cabeça".

Code Smells

Imobilidade

Você não consegue reutilizar partes do seu código em outros projetos por que isto requer um esforço gigantes. Em resumo, tudo está muito acoplado.

Code Smells

Complexidade desnecessária

Você usa padrões e arquiteturas que tornam seu código mais burocrático do que efetivo. É o famoso "e se", onde pensamos em tudo que o software pode ter um dia e já "deixamos tudo pronto".

"E se eu quiser voar com meu carro um dia?", bem, se um dia você quiser voar, aí você constrói as asas, mas se não vai precisar voar agora, foca em construir apenas o carro.

Code Smells

Repetição desnecessária

Você precisa repetir o mesmo código em diversos lugares.

Opacidade

Seu código é difícil de entender.

General Rules

Regra do escoteiro

Pessoas escoteiras têm uma regra em relação ao acampamento: devem deixá-lo mais limpo do que o encontraram antes.

A Regra do Escoteiro, por outro lado, sugere uma abordagem alternativa, que é simplesmente tentar garantir que, a cada edição, você deixe o código melhor do que o encontrou.

Configuração separada do código fonte

Os dados de configuração, como strings de conexão com o banco de dados, devem ser adicionados em um arquivo separado do código fonte.
Isso permite, por exemplo, alterar a configuração do banco com facilidade sem a necessidade de modificar o código da aplicação.

Tratar corretamente os erros

É importante realizar o tratamento de erros tanto para garantir o bom funcionamento do software, quanto para exibir mensagens claras sobre o problema encontrado.

A negligência com os erros da aplicação podem causar:

- a interrupção do sistema,

- a mensagem exibida ao usuário da aplicação não adequada,

- dificuldade em localizar erros