
A Arte de Código Limpo

Evitando a morte por código

— Por Leonardo Colman Lopes —

Agenda

- Quem sou eu?
- O que é Código Limpo
- A companhia morta pelo código
- Nomes
- Funções
- Classes
- Comentários



Leonardo Colman Lopes

Artista de Software

Evangelista de Código Limpo



Kerooker



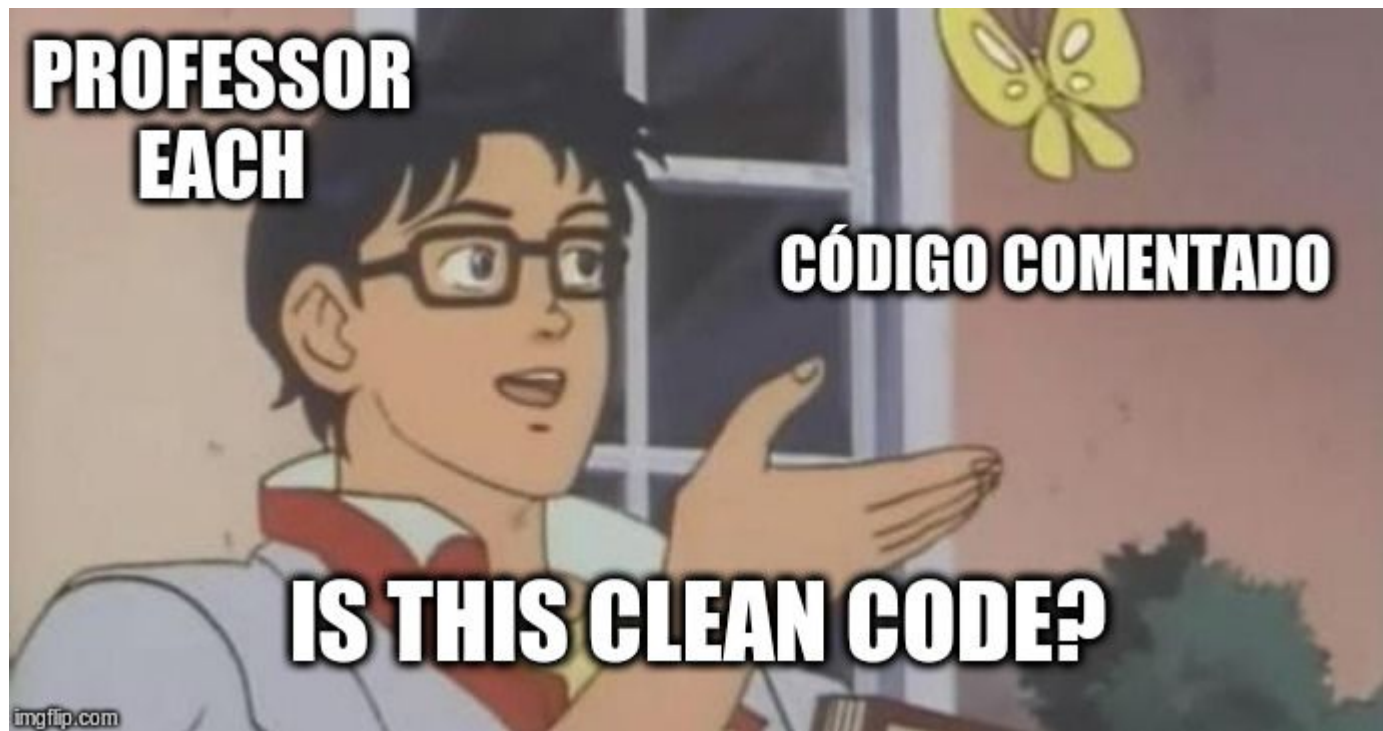
Kerooker



leonardo.colman98@gmail.com

- DASlano e Representante Discente
- Desenvolvedor Open source em qualidade de Software
- Fanático por Kotlin
- Fã de Test Driven Development (fale comigo sobre!)
- Co-autor do app Jopiter

O que é código limpo



Código limpo...

... Deve ser lido como uma prosa bem escrita

... Foi escrito por alguém que se importa

... É quando cada método que você lê **faz exatamente o que você esperava**

Qualquer um pode escrever código que um computador entende, mas **poucos conseguem escrever código que humanos podem entender**



Grady Booch
Criador do UML

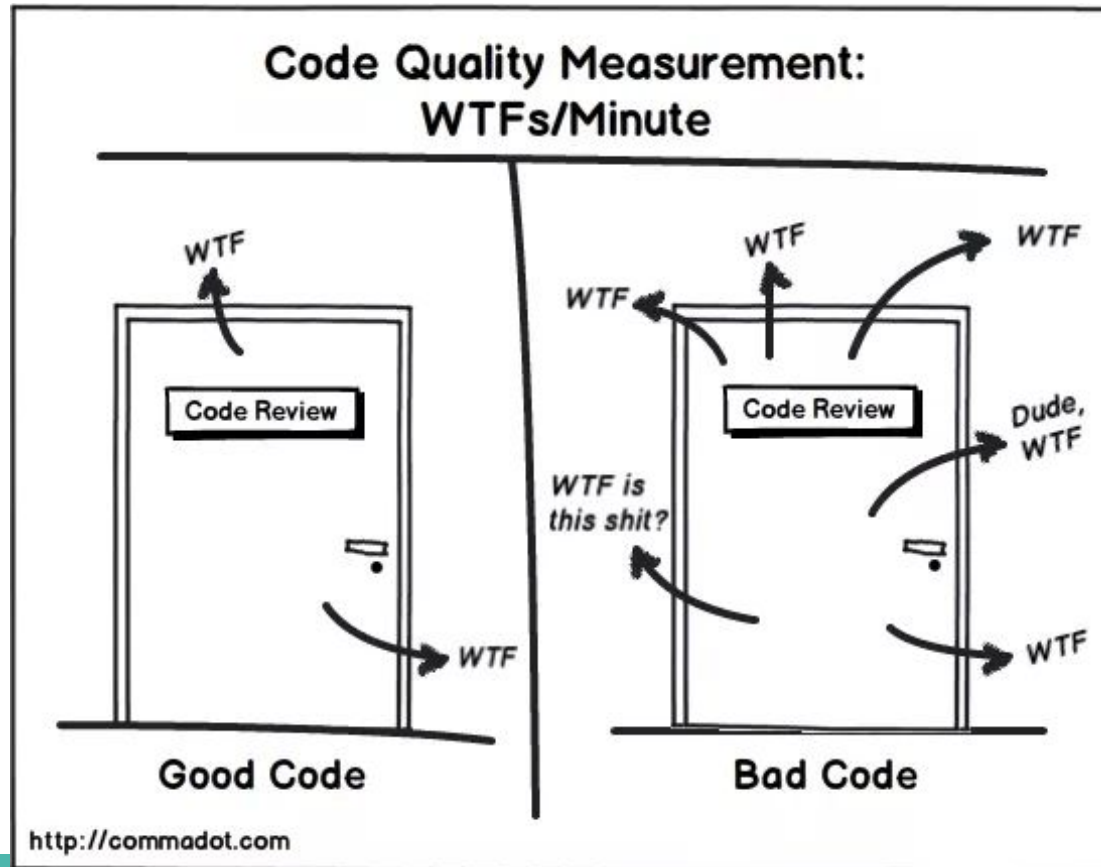
Michael Feathers
Trabalho eficaz com código legado

Ward Cunningham
Inventor da Wiki, eXtreme Programming

Martin Fowler
Autor de *Refatoração*

Sem surpresas!

O código deve fazer exatamente o que esperamos

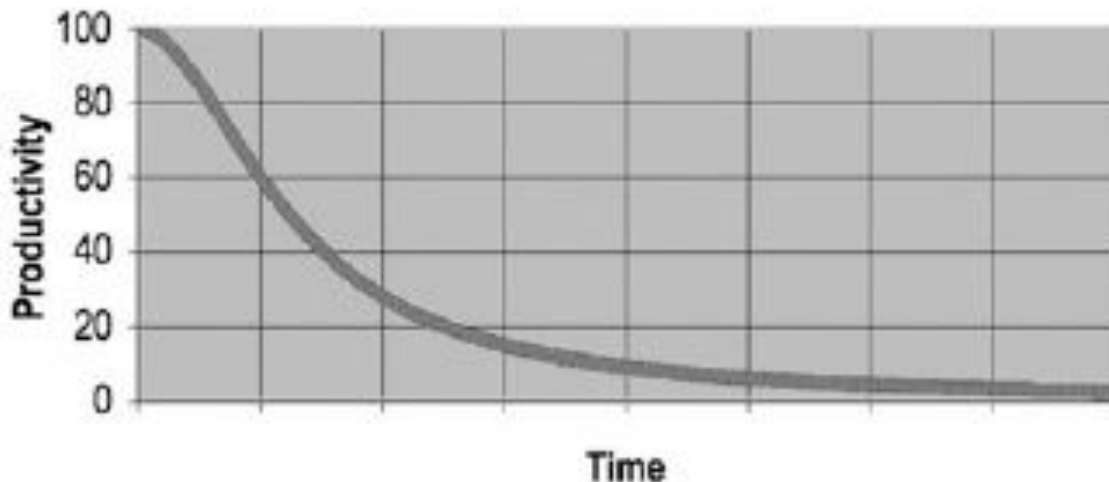


Por que código limpo?

Custo real de software = manutenção

80% do total

Código bagunçado → Produtividade **ZERO**



Por que código limpo?

Nós **LEAMOS** 10x mais do que **ESCREVEMOS**!

→ Deixe o código **mais fácil de ler**, mesmo que seja difícil escrever



Regra do Escoteiro

“Sempre deixe o código que você está editando melhor do que você encontrou”

Robert Martin (Uncle Bob)



O dia que você parar de melhorar e refatorar o código
É o dia que ele vira **LEGADO**

A Companhia morta
pelo código

Sword Inc - A companhia morta por código



O código era tão bagunçado, mas
tão bagunçado, que não era possível
dar manutenção!



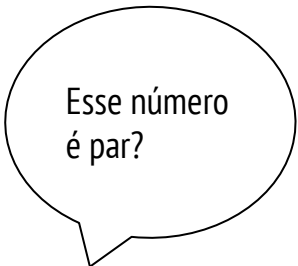
Nomes

Funções devem ser VERBOS/AÇÕES


product() → searchProduct() ; requestProduct()
client() → registerClient() ; authenticateClient()

Booleanos devem responder sim/não

goldClient() → isGoldClient()
hostsValid() → areHostsValid()
even() → isEven()



Esse número
é par?



Os hosts
configurados
são válidos?

Classes são SUBSTANTIVOS

Client, Order, CustomerService, UserDatabase

Evite nomes sem significado

ClientInfo, ClientData → Client

Data → Do que? → OrderData → Order

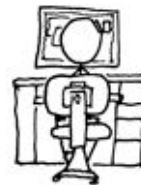
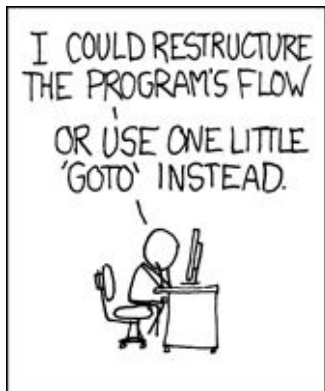
Remova indicadores de interface

ICustomerService → CustomerService

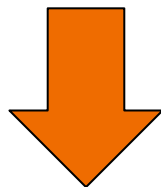
UserDatabaseImpl → UserDatabase

Como você entende o que uma função faz?

- Leio aquele `/*A função XXX faz YYY*/` pré-histórico
- Caço todos que a chamam e entendo o contexto
- Leio a implementação e descriptografo o significado



Isso NUNCA deveria ser necessário!



NOMES devem comunicar o objetivo, **sempre**

Achou um nome melhor para algo?




RENOMEIE! REFATORE!

Leva 3s com uma IDE...


Nomes devem ser PRONUNCIÁVEIS

`getInvcdTl()` → `getInvoiceableCreditTimeLimit()`

`var genymdhms` → `var generationTimestamp`



generation date, year,
month, day, hour,
minute, and second



gen-yah-muddahims

Evite abreviações

A não ser muito conhecidas, como HTML ou HTTP

Nomes devem ser CONSISTENTES

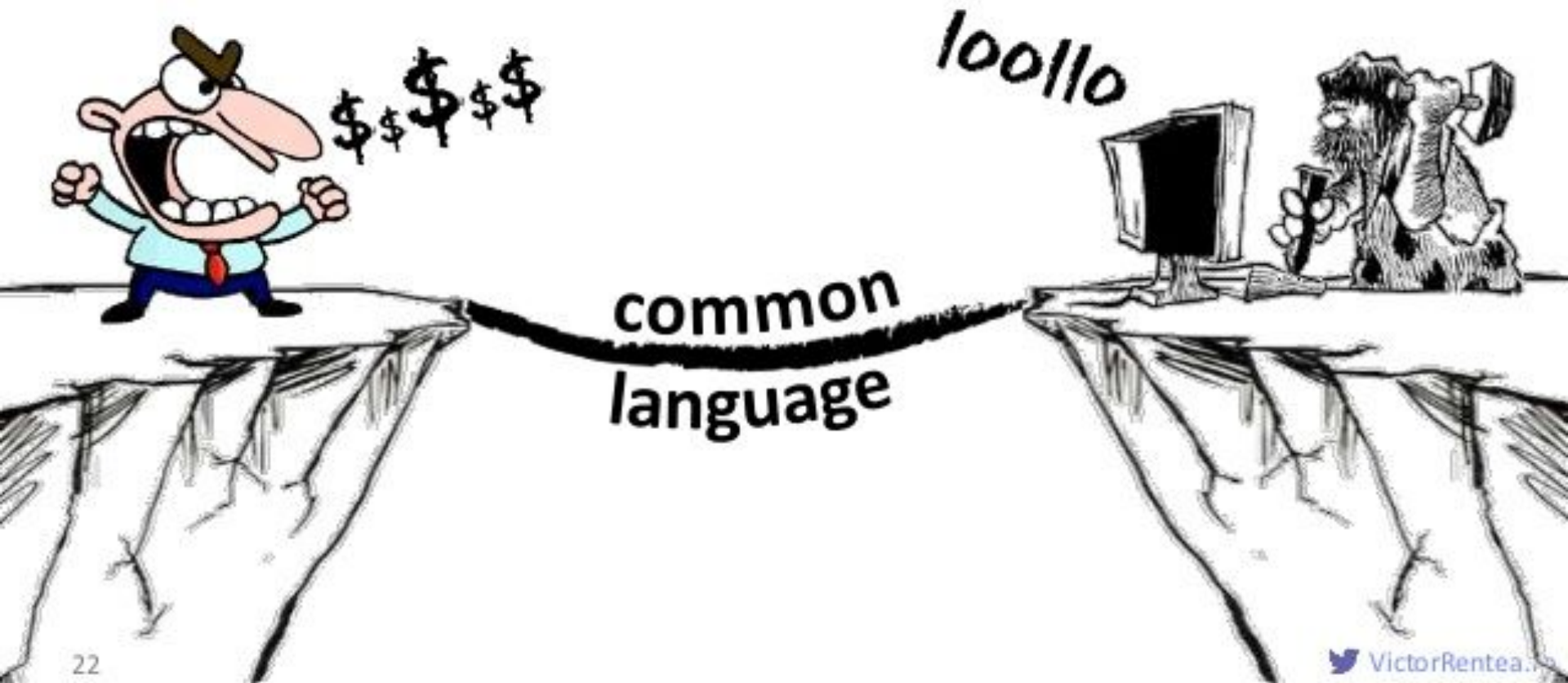
.find...() .fetch...() .get...() ?

Use a convenção da equipe/projeto
Não fique mudando

Deixe-os únicos
Sinônimos podem confundir

Buyer, Client, User ou Customer?

Business-IT Gap



Funções

Uma função deve fazer **uma única coisa**.
Ela deve fazer isso bem, e deve fazer
apenas isso.

Uncle Bob

Uma função deve ser

PEQUENA

Mas... Quão pequenas?

5 linhas.



Por que tão pequenas?

Para fazer uma, e apenas uma coisa

Afinal... Não dá pra fazer muito mais do que isso em 5 linhas

Para que ela tenha um BOM NOME

Se ela só faz uma coisa, você CONSEGUE encontrar um bom nome

Uma função é como uma paisagem, um terreno, um bairro

Quem cresce lá, quem viveu por lá, quem construiu aquele lugar conhece tudo

Já o resto do time...

A nova integrante da equipe...

0 estagiário...

[illegible]

Change Request #323

O que você faz?

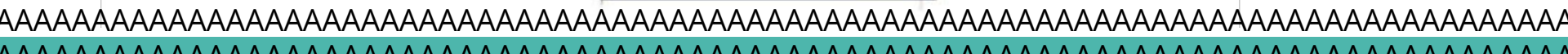
```
if(cr323) {
    doWhatFeatureDoes()
}
```

```
..., boolean cr323)
```



Apagar tudo e
começar de
novo?

Se eu apagar isso continua funcionando igual
Melhor deixar...



Por que temos medo de funções pequenas?

Elas performam mal?

Não

Métodos pequenos executam muito mais rápido
“Just in time compiler optimization”

Se realmente for uma preocupação...

“Meça, não chute”

Kirk Pepperdine

“Otimização prematura é a raiz de todo o mal”

Donald Knuth

Por que temos medo de funções pequenas?

Ao invés de um terreno familiar



Agora preciso lidar com milhões de funções pequenas
Não lembro nem os nomes!

Mas seu time irá te agradecer
Você irá se agradecer
Daqui uns 6 meses

Sem parâmetros BOOLEANOS

```
removeOrders(customer, false, true)
```



Preguiça?
Velocidade?
Legado.

No máximo 3 parâmetros

```
removeOrders(customer, order, owner, store, price,  
valueToCashback, productIds, ...)
```



Provavelmente
uma classe se
esconde aqui

Sem parâmetros NULLABLE

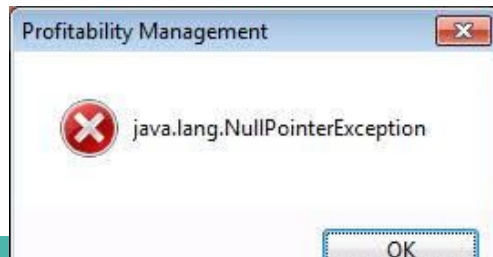
```
if (customer != null) { ... } else { ... }
```



Claramente
fazendo duas
coisas

Sem retornos NULLABLE

Use Optional ou jogue uma exceção



Use apenas UNCHECKED exceptions

```
try {} catch {}  
throws XXX
```



Poluição de código pra todo lado!

**Amamos RuntimeException
(porque são invisíveis)**

Funções muito grandes

São onde as classes se escondem!

- Usam muitas vezes a mesma variável
- Parecem fazer a mesma coisa várias vezes
- Não estão muito conectadas com o resto



Classes

Procure manter valores IMUTÁVEIS em Data Classes

- Se criados válidos, continuam válidos
- Thread-Safe
- HashMap/Tree safe

Data Class = Classe que expõe TODO o seu estado

```
class Client {  
    private final String name;  
    private final String surname;  
  
    public Client(String name, String surname) {  
        this.name = name;  
        this.surname = surname;  
    }  
    // Setters + Getters  
}
```

Exponha COMPORTAMENTO, não detalhes

Detalhes VÃO Mudar. Comportamento NÃO

```
car.engineStarted = true
```

```
car.setEngineStarted(true)
```

```
car.startEngine()
```

Esconda informações

Mostre a menor quantidade possível de informação

```
var remainingKm = car.getGasLeft() + car.getKmsPerLiter()
```

```
var remainingKm = car.calculateRemainingKm()
```



KEEP
IT
SIMPLE
STUPID

A hand-drawn illustration of the phrase 'KEEP IT SIMPLE STUPID' on a light gray background with a black border. The letters are stylized and colored: 'K' is orange, 'E' is black, 'E' is black, 'P' is black, 'I' is blue, 'T' is black, 'S' is green, 'I' is black, 'M' is black, 'P' is black, 'L' is black, 'E' is black, 'S' is yellow, 'T' is black, 'U' is black, 'P' is black, 'I' is black, and 'D' is black.

- Não dê milhões de voltas
- Não crie abstrações desnecessárias
- Não aplique todos os padrões só porque “São bons padrões”
- Não tente otimizar sem saber que precisa

Deixe as coisas simples!

Comentários

Comentários = falhamos

É uma prova escrita de nossa incompetência

PARE E PENSE

Por que estou escrevendo um comentário?

- Deixar claro o que uma função faz?
- Explicar o que é uma constante?
- Fazer uma piada com o nome de uma variável?

Comentários REDUNDANTES

```
/**  
 * Gets the Player Name  
 * @returns The Name of the Player  
 */  
fun getPlayerName(): String {
```

Comentários MENTIROÇOS

```
/**  
 * Gets the Player Name  
 * @returns The Name of the Player  
 */  
fun getPlayerUsername(): String {
```

Função Refatorada

Código Comentado

**Esqueci de descomentar?
Era importante pra funcionar?**

Código Distante do Código

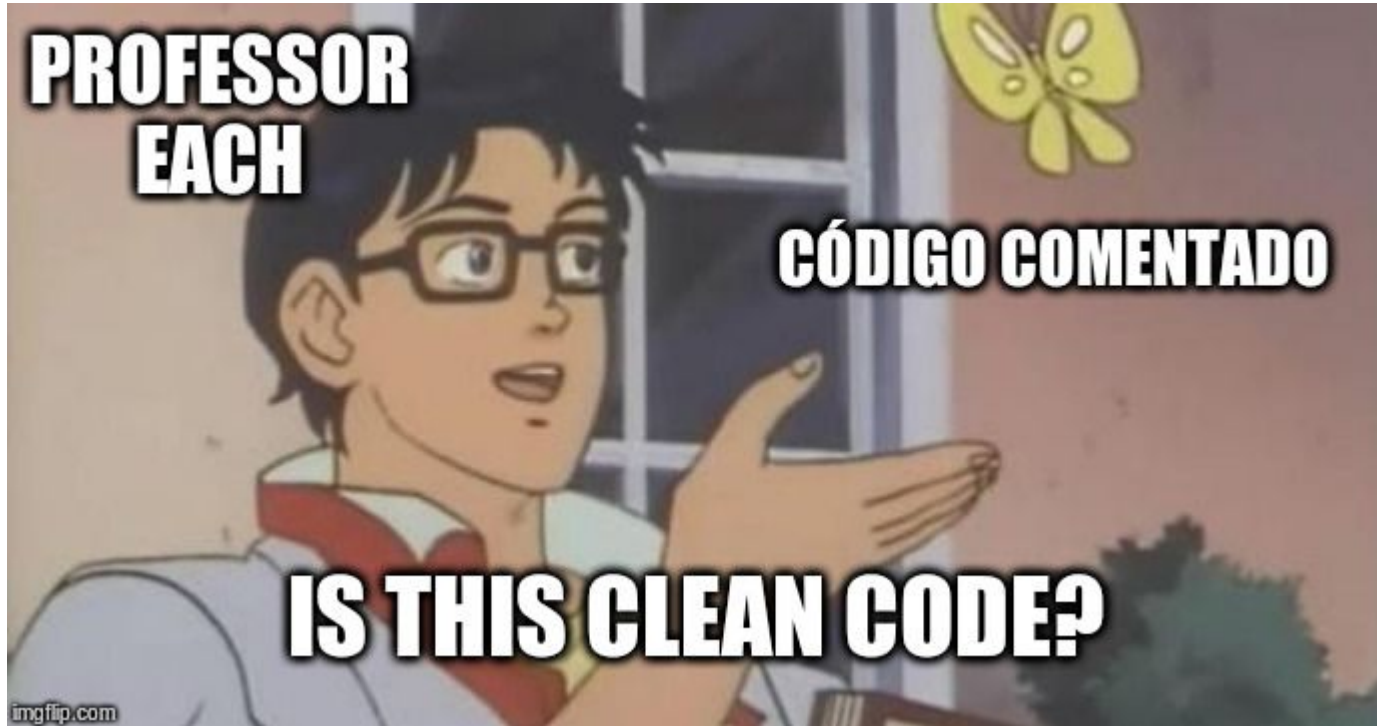
Comentário se refere a qual parte?

Bons Comentários

Explicam o que o código é incapaz de fazer

- Deixa claro o porquê de uma chamada aparentemente esquisita
- Coloca o link do algoritmo implementado
- Menciona consequências possíveis
 - `// Esse método não é Thread-Safe`

O que é código limpo



A Arte de Código Limpo

Evitando a morte por código

— Por Leonardo Colman Lopes —

Referências

- Livro Clean Code (praticamente tudo)
- Eu mesmo e minha experiência
- Inspirado na apresentação “The Art of Clean Code”, por Victor Rentea
 - <https://www.slideshare.net/VictorRentea/the-art-of-clean-code>