DIGITAL
INNOVATION
ONE

Princípios SOLID na prática com Java

Alex Garcia
IT Architect – Santander Tecnologia e Inovação

DIGITAL INNOVATION ONE

Mais sobre mim

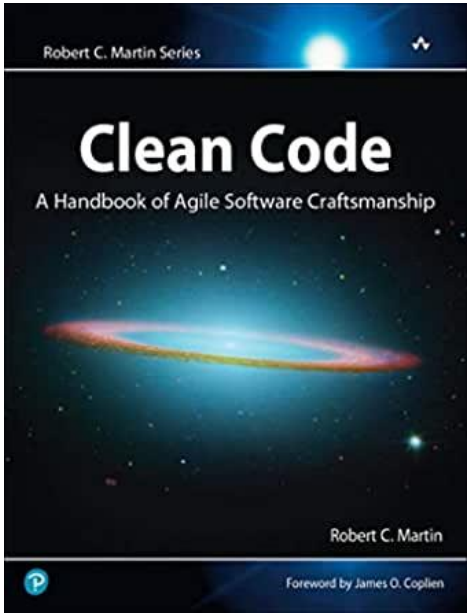
- Atuo há mais de 12 anos com TI
- Gosto de Sistemas Distribuídos e IoT
- Leitura, filmes e esportes (tudo balela)
- Qualquer coisa me adiciona lá:

  https://www.linkedin.com/in/alexfgarcia

Ícone: https://www.flaticon.com/authors/good-ware

DIGITAL INNOVATION ONE

Por que estamos aqui?

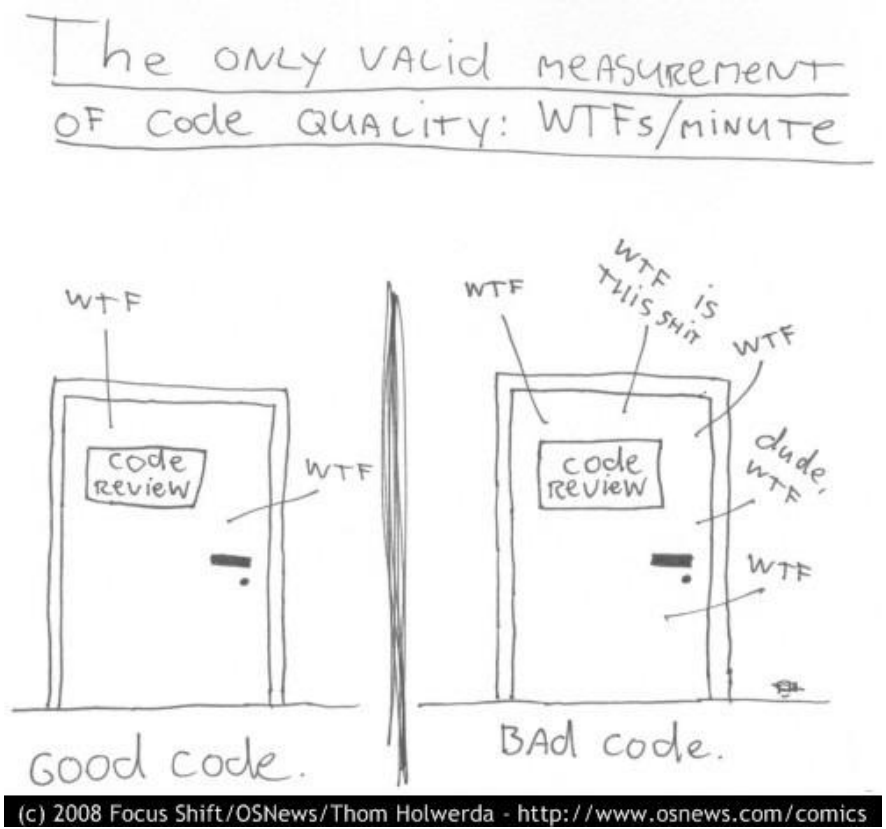


"It is not enough for code to work"

Robert C. Martin. Clean Code. 2008
a.k.a. Uncle Bob

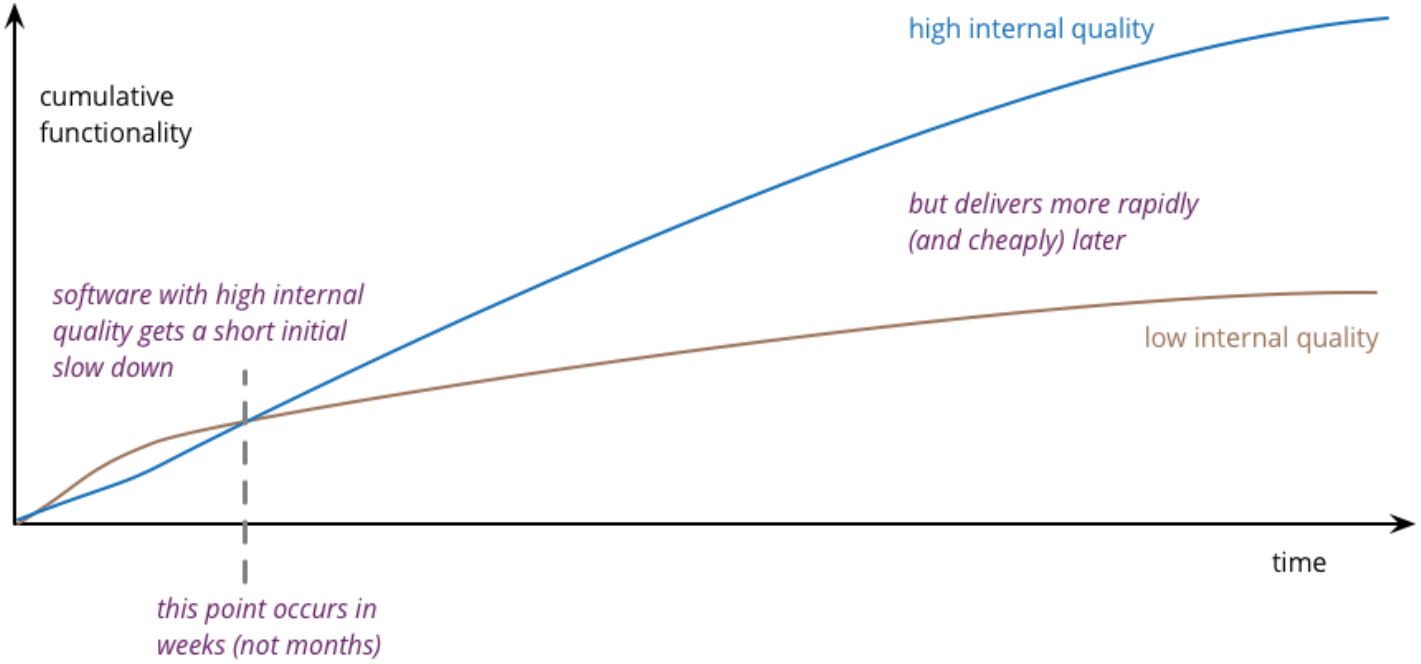
DIGITAL INNOVATION ONE

Código ruim?



DIGITAL INNOVATION ONE

Custo de Mudanças



Martin Fowler: Is High Quality Software Worth the Cost?
https://martinfowler.com/articles/is-quality-worth-cost.html

DIGITAL INNOVATION ONE

Princípios SOLID

Acrônimo para 5 princípios de desenho de software que auxiliam a manter os débitos técnicos sob controle

Subconjunto de princípios propostos por Uncle Bob em "Design Principles and Design Patterns" (2000)

DIGITAL INNOVATION ONE

Percurso

Etapa 1	**S**ingle Responsibility Principle
Etapa 2	**O**pen Closed Principle
Etapa 3	**L**iskov Substitution Principle
Etapa 4	**I**nterface Segregation Principle
Etapa 5	**D**ependency Inversion Principle

DIGITAL INNOVATION ONE

Requisitos

- ✔ Conceitos de Programação Orientada a Objetos
- ✔ Java 11 (Comumente a linguagem não importa tanto 😬)
- ✔ IntelliJ CE (ou IDE de sua preferência) com Maven 3 e JDK 11
- ✔ Git (código-fonte: https://github.com/alexfabgarcia/yasp)

Etapa 1: Single Responsibility Principle

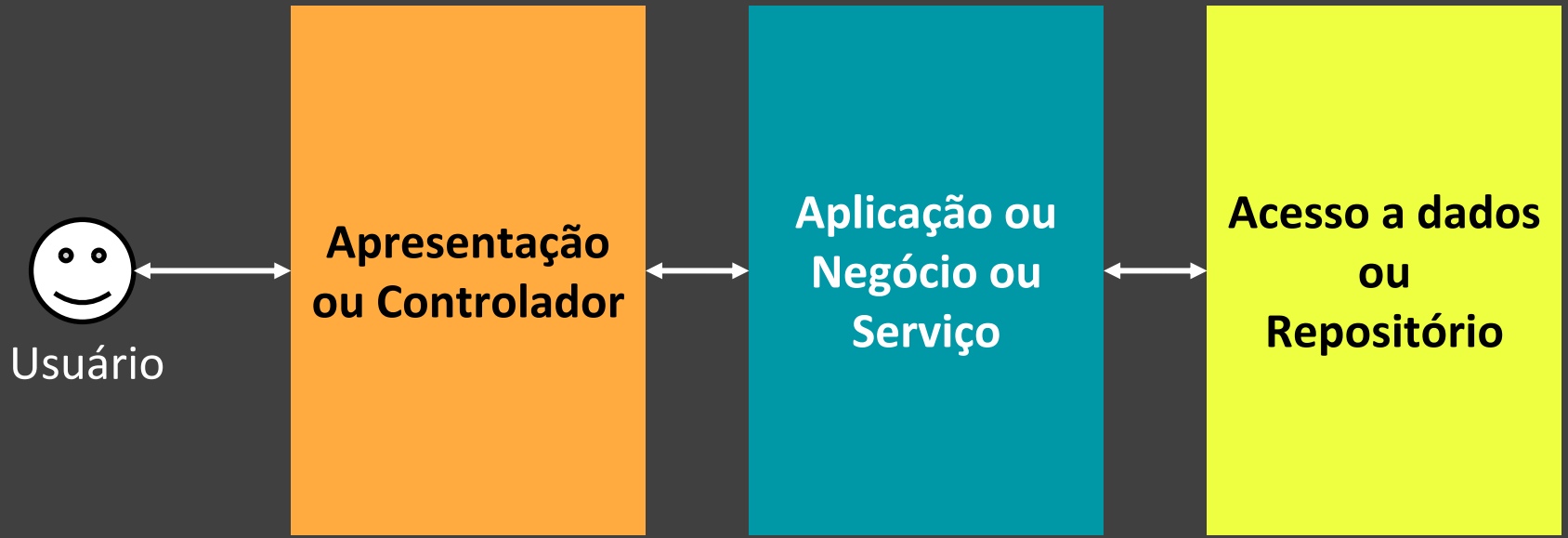
Princípios SOLID na prática com Java

Single Responsibility

- Cada função, classe ou módulo deve ter um e apenas um motivo para ser modificado(a)
- Por quê?
  - Legibilidade e manutenibilidade
  - Menor acoplamento, maior coesão
  - Facilita a escrita de testes

DIGITAL INNOVATION ONE

Demo



Arquitetura convencional em três camadas

DIGITAL INNOVATION ONE

Single Responsibility

- Identificar razões para mudança
  - If / Switch (principalmente aninhados)
  - Método monstro
  - Classe Deus
- Extrair responsabilidades aplicando o SRP

Etapa 2: Open Closed Principle

Princípios SOLID na prática com Java

DIGITAL INNOVATION ONE

Open Closed

- Funções, classes e módulos devem estar fechados para modificação e abertos para extensão
- Por quê?
  - Minimiza o risco de regressão de bugs
  - Favorece o desacoplamento em conjunto com SRP

Etapa 3: Liskov Substitution Principle

Princípios SOLID na prática com Java

DIGITAL INNOVATION ONE

Liskov Substitution

- Seja S um subtipo de T, então os objetos do tipo T em um programa podem ser substituídos por objetos do tipo S sem quebrar o funcionamento deste programa.

Etapa 4: Interface Segregation Principle

Princípios SOLID na prática com Java

Interface Segregation

- Clientes não devem ser forçados a depender de métodos que eles não usam
- Seja o mais específico possível, evitando interfaces com muitos métodos

Etapa 5: Dependency Inversion Principle

Princípios SOLID na prática com Java

DIGITAL INNOVATION ONE

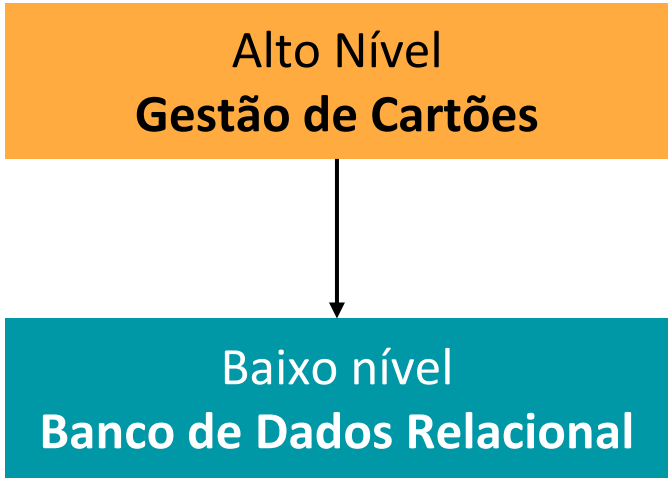
Dependency Inversion

1. **Módulos de alto nível** não devem depender de **módulos de baixo nível**. Ambos devem depender de abstrações.
2. **Abstrações** não devem depender de detalhes. Detalhes devem depender da abstrações.

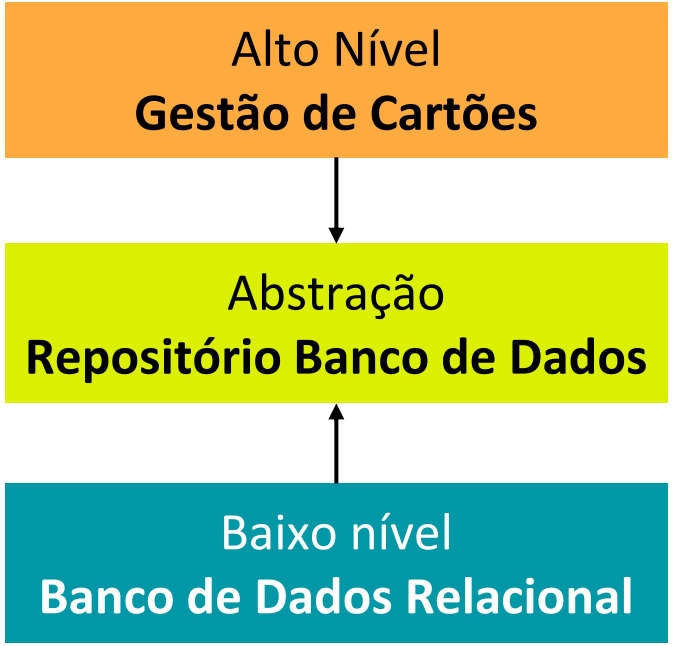
DIGITAL INNOVATION ONE

Dependency Inversion

Depender do Detalhe



Depender da abstração



DIGITAL INNOVATION ONE

Dependency Inversion

1. Dependency Inversion Principle (DIP)
2. Inversion of Control (IoC)
3. Dependency Injection (DI)

DIGITAL INNOVATION ONE

Para saber mais

SOLID e Design de Software na prática
https://www.youtube.com/watch?v=4oVByCJJkRI

The SOLID principles in Pictures
https://medium.com/backticks-tildes/the-s-o-l-i-d-principles-in-pictures-b34ce2f1e898

DevDojo - Maratona Java Virado no Jiraya:
https://www.youtube.com/watch?v=VKjFuX91G5Q&list=PL62G310vn6nFIsOCC0H-C2infYgwm8SWW

Alguns Livros:

- Martin, Robert C. Clean code. Pearson Education, 2009.
- Guerra, Eduardo. Design Patterns com Java. Editora Casa do Código, 2014.
- Weissmann, Henrique Lobo. Vire o jogo com Spring Framework. Editora Casa do Código, 2014.

DIGITAL INNOVATION ONE

Obrigado galerinha! Dúvidas?

> Comunidade online DIO (discord)