S T Q Q S S D

__/__/__

Resumo - 05

The SOLID principles of object-oriented design

Introduction:

- SOLID é um conjunto de princípios para melhorar a estrutura e a manutenção de código orientado a objetos
- Criado por Robert. C. Martin (Uncle bob) para promover boas práticas de design de software
- Facilita a extensibilidade, reutilização e evita código rígido e acoplado.

1. Single responsability principle (SRP):

- Uma classe deve ter apenas UMA razão para mudar
- Cada classe deve ter apenas UMA única responsabilidade no sistema
- Evita que a classe fique bagunçada com múltiplas funções e vire um "monstro"

2. Open/closed principle (OCP):

- O código deve ficar aberto para extensão, mas fechado para modificação
- Novas funcionalidades devem ser implementadas sem alterar o código existente

__/__/__ ②

S T Q Q S

↝ Uso de abstrações e heranças

3. Liskov substitution principle (LSP):

↝ Subtipos devem ser substituíveis por seus tipos base sem quebrar o código

↝ Herança deve garantir que a subclasse possa ser usada no lugar da superclasse sem alterar o comportamento esperado

4. Interface Segregation ~~principle~~ princípal (ISP):

↝ Interfaces devem ser específicas para cada necessidade

↝ Uma classe não deve ser forçada a implementar métodos que não usa.

5. Dependency Inversion principle (DIP):

↝ Módulos de alto nível não devem depender de módulos de baixo nível, ambos devem depender de abstrações.

↝ Abstrações não devem depender de detalhes, detalhes devem depender de abstrações

↝ Uso de ~~inversão~~ injeção de dependência para reduzir acoplamento e facilitar testes

spiral

O S S D

__/__/__

③

Conclusion:

→ Aplicar SOLID melhora a qualidade do código, facilitando a manutenção, extensibilidade e reutilização

→ Reduz acoplamento e torna o sistema mais modular e testável

→ Essencial para projetos escaláveis e de longo prazo