SOLID

Владимир Поляков
vladimir.p.polyakov@gmail.com

https://github.com/drxaos-edu/spring-demo

SOLID

- Single responsibility principle
- Open/closed principle
- Liskov substitution principle
- Interface segregation principle
- Dependency inversion principle

Robert C. Martin

Принцип единственной ответственности

Каждый объект должен иметь одну ответственность и эта ответственность должна быть полностью инкапсулирована в класс.

Принцип открытости/закрытости

Программные сущности (классы, модули, функции и т. п.) должны быть открыты для расширения, но закрыты для изменения.

Принцип подстановки Барбары Лисков

Функции, которые используют базовый тип, должны иметь возможность использовать подтипы базового типа, не зная об этом.

(поведение наследуемых классов не должно противоречить поведению, заданному базовым классом)

Принцип разделения интерфейса

Клиенты не должны зависеть от методов, которые они не используют.

(слишком «толстые» интерфейсы необходимо разделять на более маленькие и специфические, чтобы клиенты знали только о методах, которые необходимы им в работе)

Принцип инверсии зависимостей

Модули верхних уровней не должны зависеть от модулей нижних уровней. Оба типа модулей должны зависеть от абстракций.

Абстракции не должны зависеть от деталей. Детали должны зависеть от абстракций.

Dependency Injection

(внедрение зависимостей)

Одна из реализаций IoC

Guice, Spring, OSGi, …

Вопросы

vladimir.p.polyakov@gmail.com
https://github.com/drxaos-edu