**Паттерны написания кода**

Способствует тому, чтобы все разработчики говорили на одном языке. Это значительно упрощает поддержку и развитие кодовой базы проектов, за счёт снижения порога проектных знаний.

Признаки хорошей кодовой базы:

1. Масштабируемость (легко вносить изменения)
2. Невысокий порог знаний вхождения в проект
3. Код должен быть простым

SOLID

1. Single responsibility (принцип единой ответственности)

**Один класс (одна конкретная сущность) должен решать одну задачу. У каждого класса одна зона ответственности, всё декомпозируется на модули.**

Решает проблему сильно связного кода, когда сущность решает много задач (GodObject)

1. Open / closed principle (принцип открытости / закрытости)

**Программные сущности должны быть открыты для расширения, но закрыты для модификаций (изменений).**

1. Liskov’s substitution principle (**принцип подстановки Барбары Лисков**)

**Функции / сущности, которые используют родительский тип, должны точно также работать и с дочерними классами.**

**Наследуемый класс должен дополнять поведение базового класса, а не замещать его.**

1. Interface segregation principle (принцип разделения интерфейса)

**Программные сущности не должны зависеть от методов, которые они не используют.**

Разбиваем большие интерфейсы на множество маленьких, которые решают специфичные задачи.

1. Dependency inversion principle (принцип инверсии зависимостей)

**Модули высокого уровня не должны зависеть от модулей более низкого уровня. Все они должны зависеть от абстракций, а абстракции в это время не должны зависеть от деталей; детали должны зависеть от абстракций.**

