Шаблоны проектирования

Паттерн проектирования

Описание идеи решения часто встречающейся архитектурной задачи при разработке ПО.

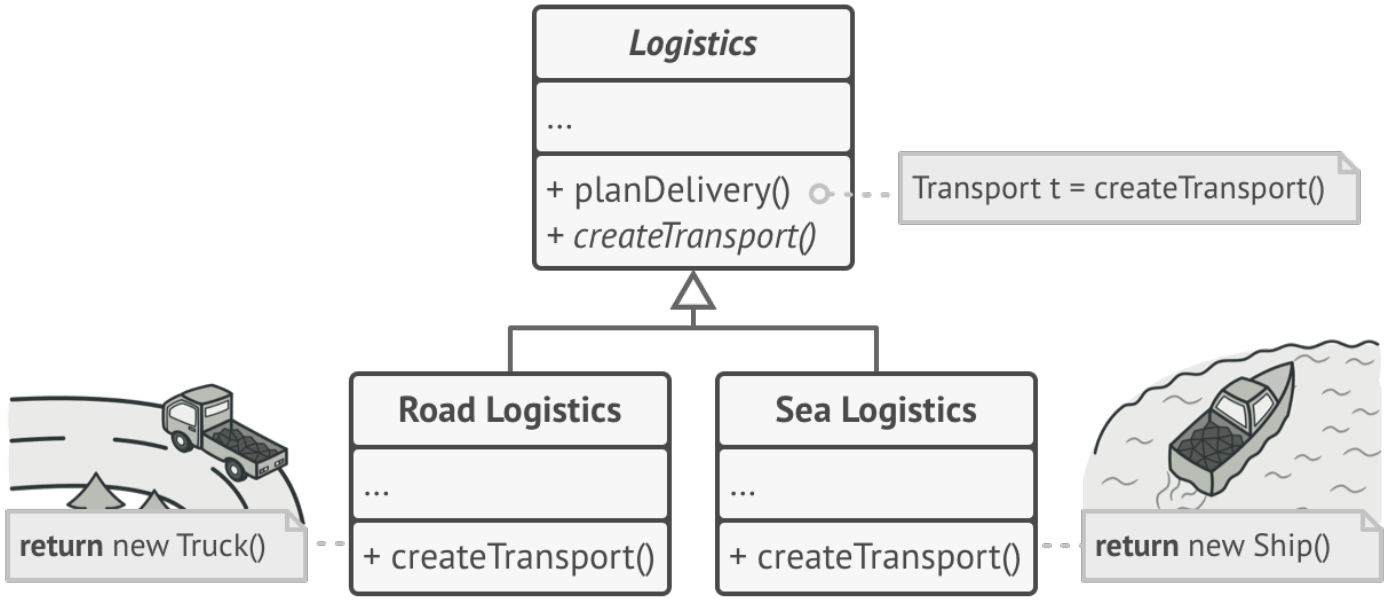
Это не готовый кусок кода или библиотека, а общий подход к реализации решения.

Паттерны бывают порождающие, структурные и поведенческие.

Порождающие шаблоны проектирования

Factory

- Предоставляем дочерним классам интерфейс для создания экземпляры некоторого класса. Конкретный класс может быть определён в момент создания.



Singleton

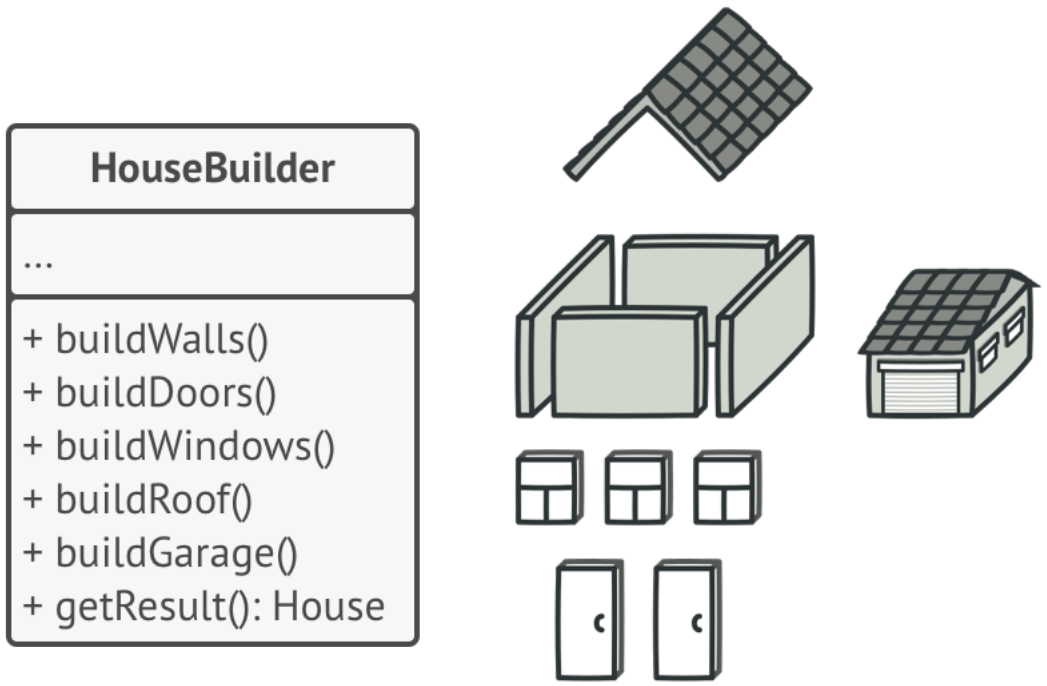
- У класса есть только один экземпляр.
- Пример: подключение к базе данных, конфигурация приложения.

Singleton

- У класса есть только один экземпляр.
- Пример: подключение к базе данных, конфигурация приложения.
- Вопрос: почему не глобальные переменные?

Builder

- Позволяет создавать сложные объекты пошагово. Конструируем класс вне этого класса.

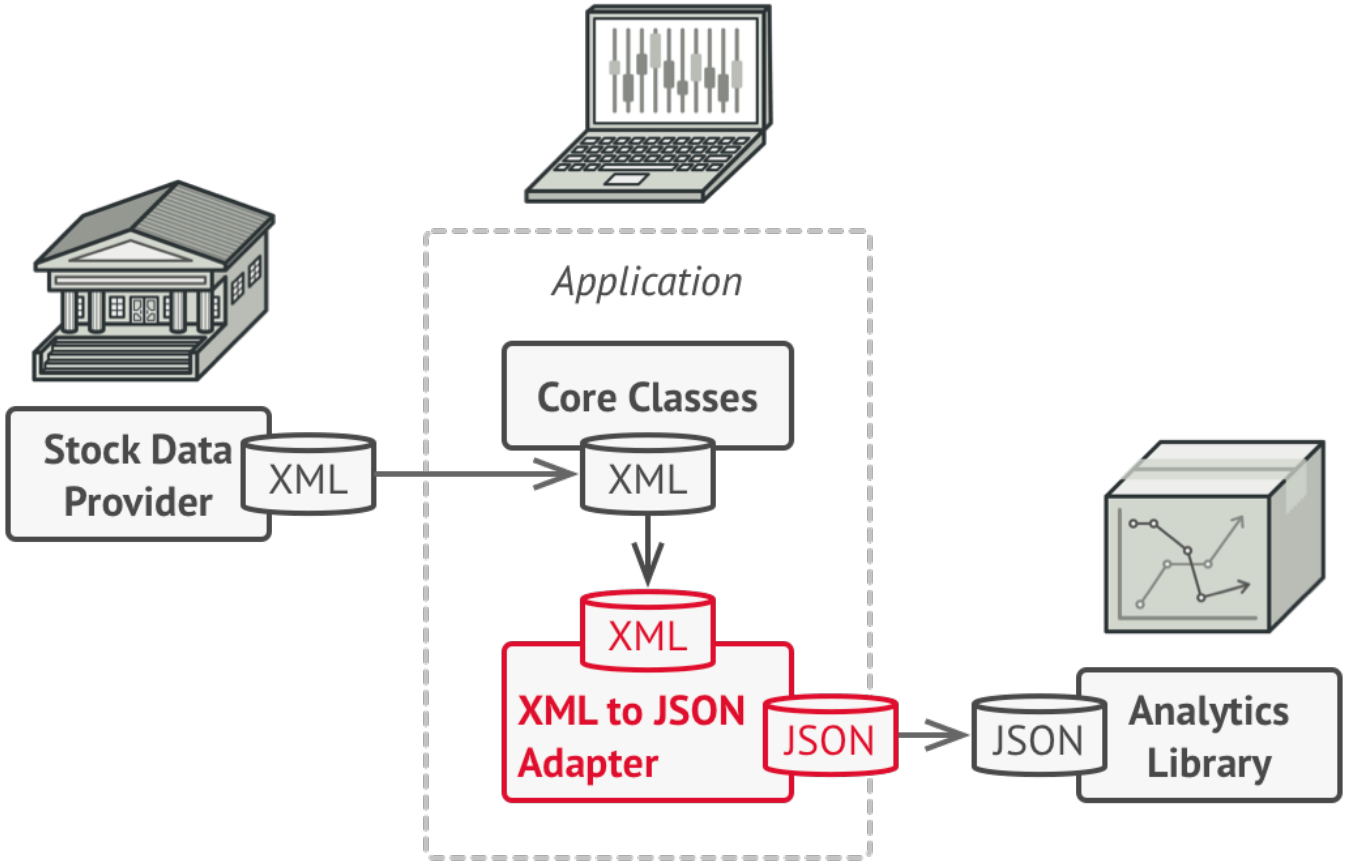


А ещё есть

- Abstract factory
- Prototype

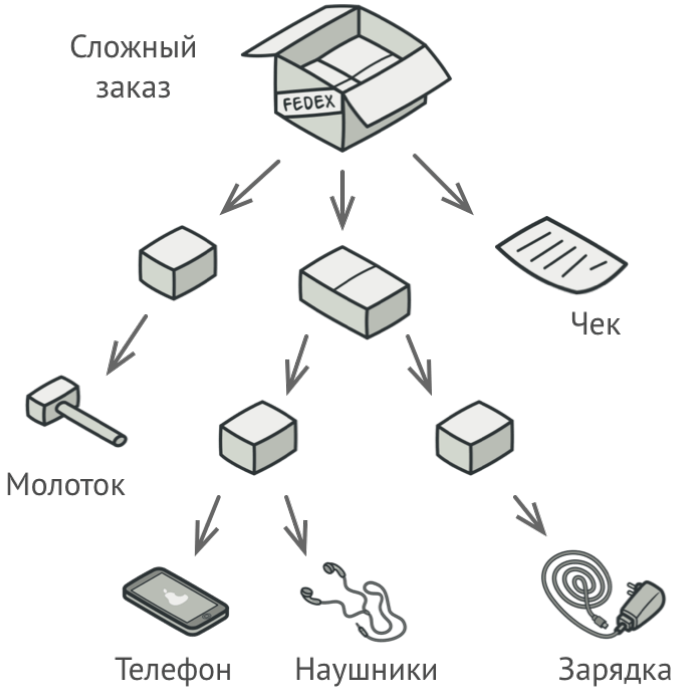
Структурные шаблоны проектирования

Adapter


Application
Core Classes
XML
Stock Data Provider
XML
XML
XML to JSON Adapter
JSON
JSON
Analytics Library

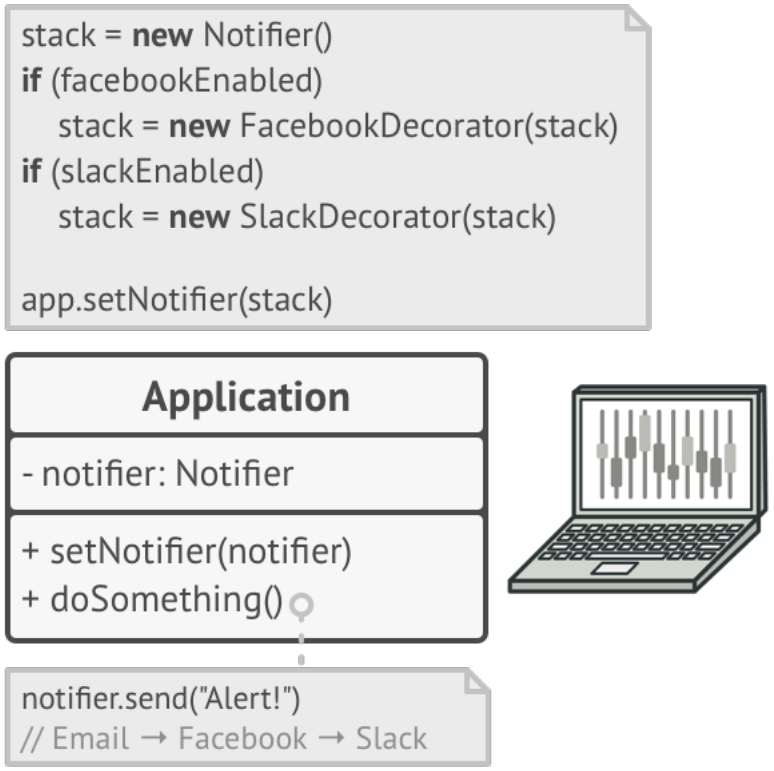
Composite

- Позволяет единообразно работать с древовидной структурой объектов.



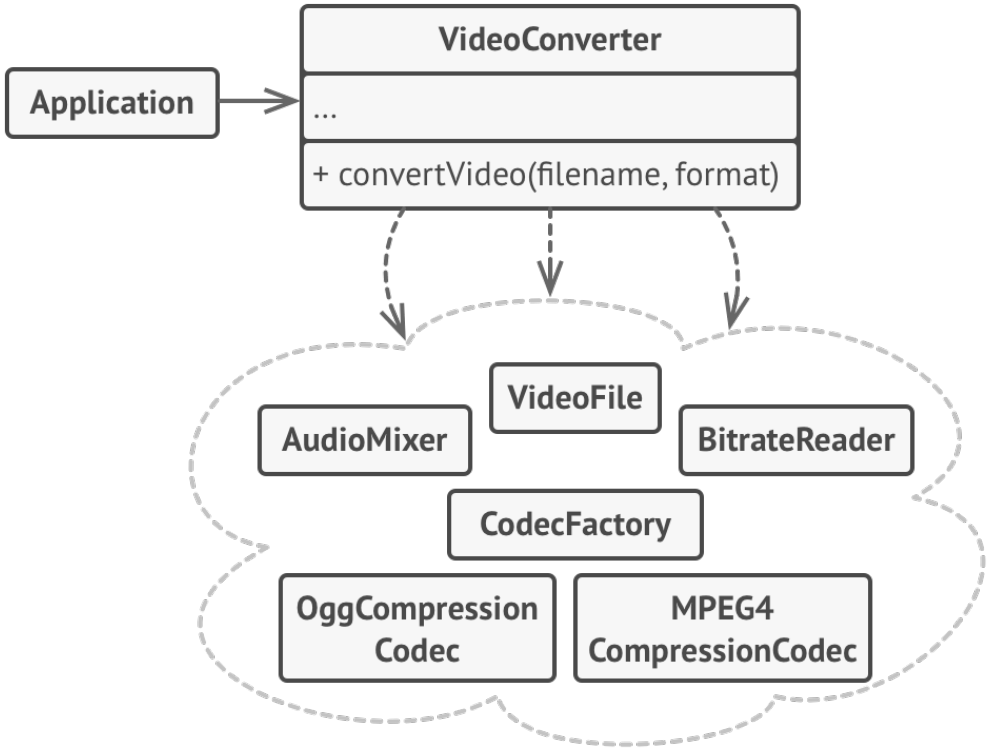
Decorator

- Позволяет добавлять объектам новую функциональность, не изменяя их.



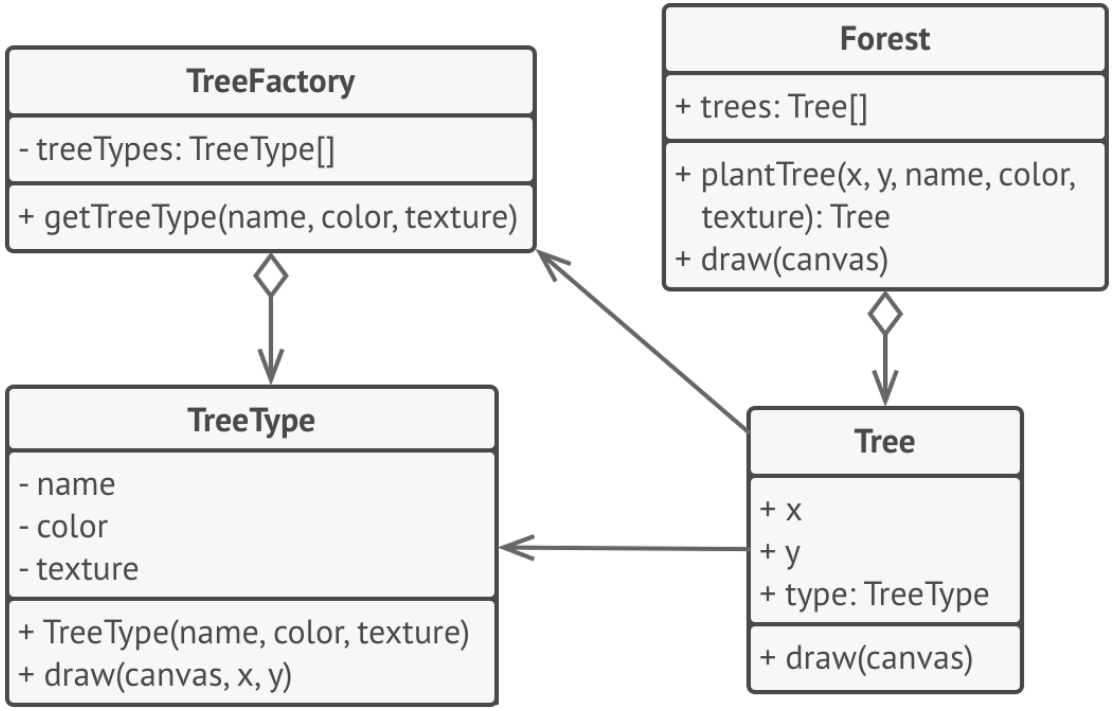
Facade

- Позволяет предоставлять простой интерфейс к сложной системе.



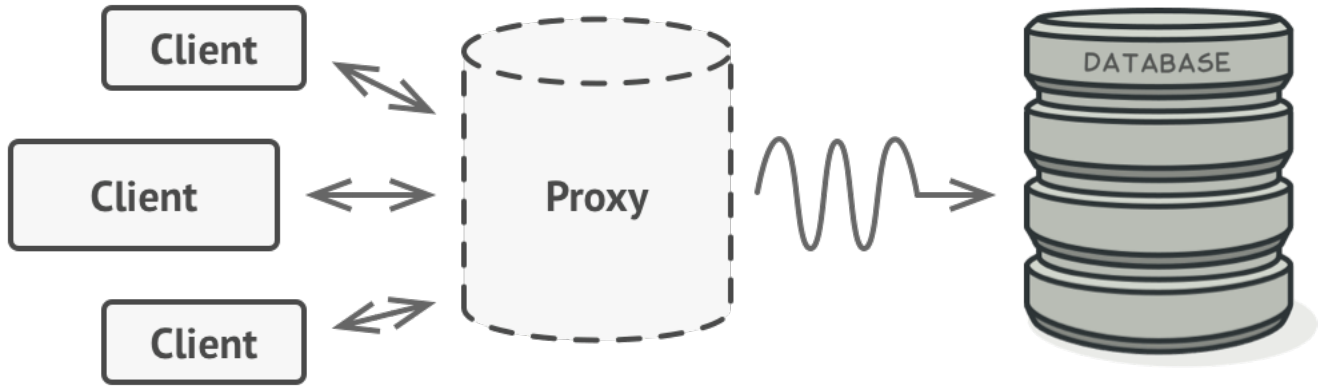
Flyweight

- Позволяет экономить оперативную память, не создавая существующие объекты заново.



Proxy

- Реализует прослойку между клиентом и настоящим объектом (используется для кэширования, контроля доступа, логгирования).



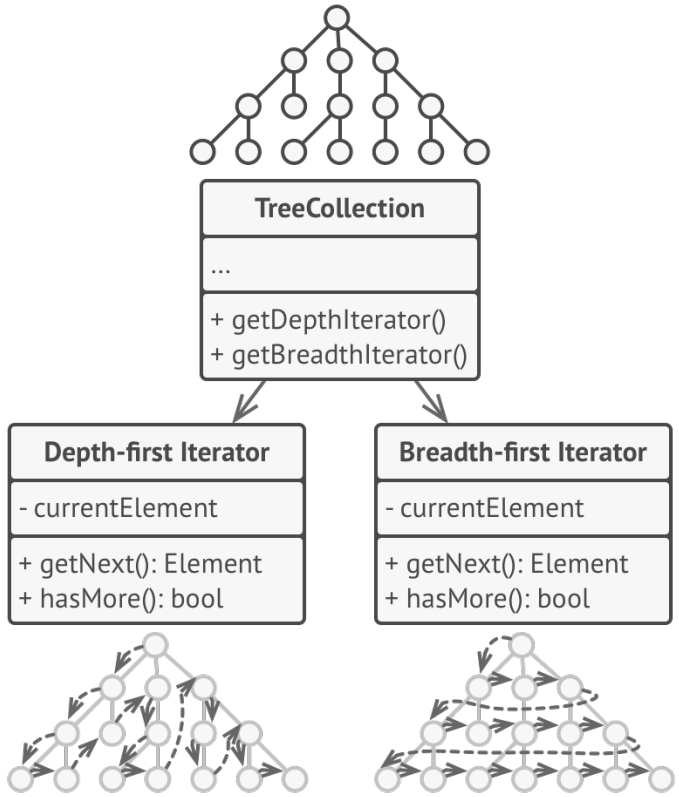
А ещё есть

- Bridge

Поведенческие шаблоны проектирования

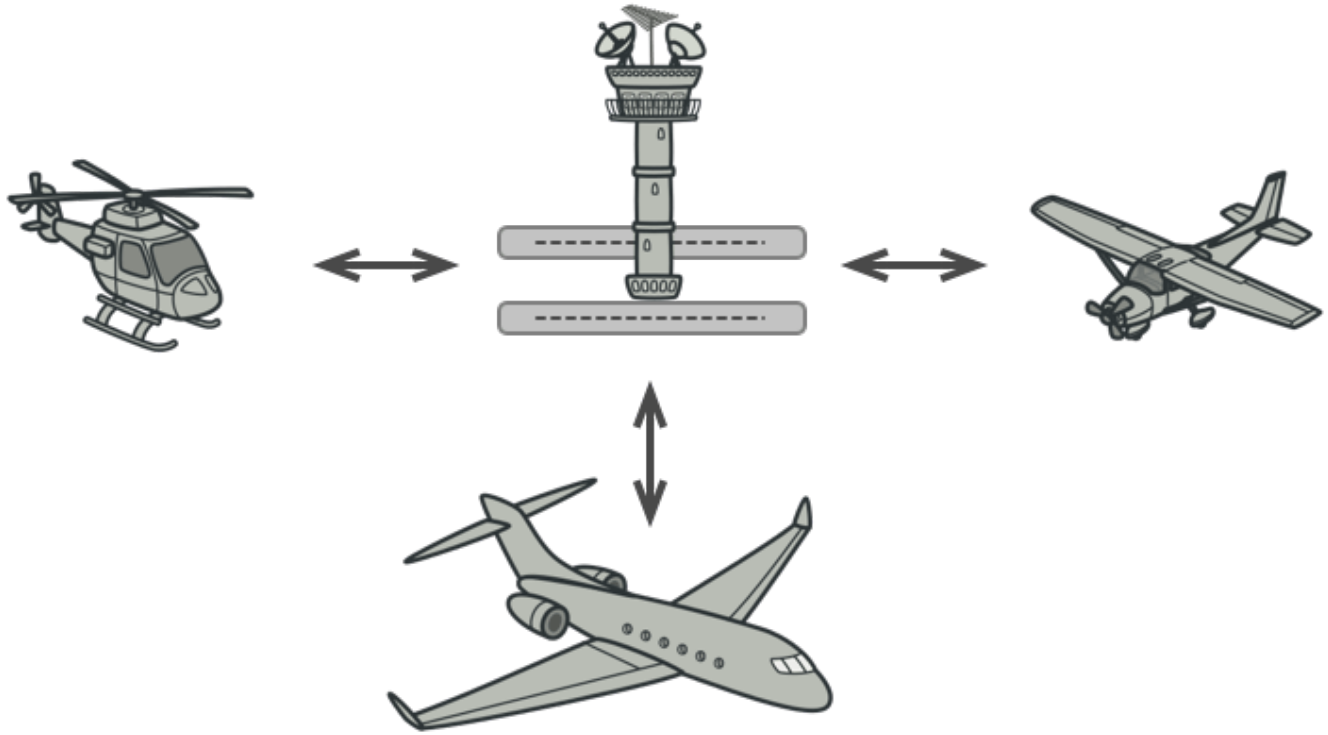
Iterator

- Позволяет единообразно осуществлять обход коллекции.



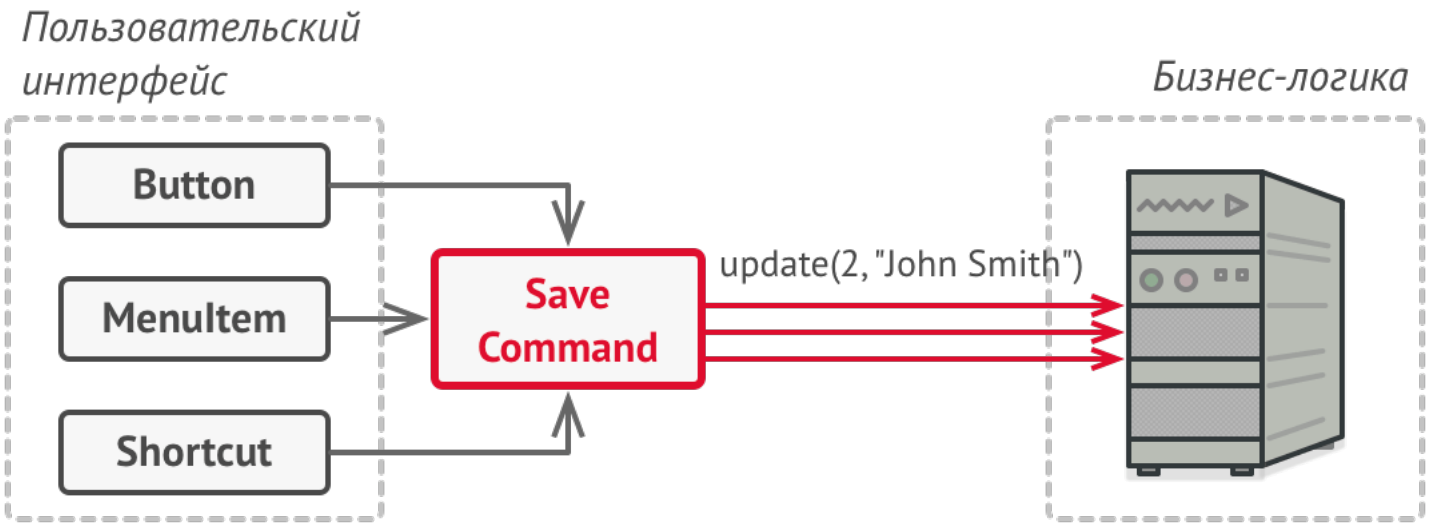
Mediator

- Позволяет создавать промежуточные слои для общения множества объектов. Уменьшает связность между ними.



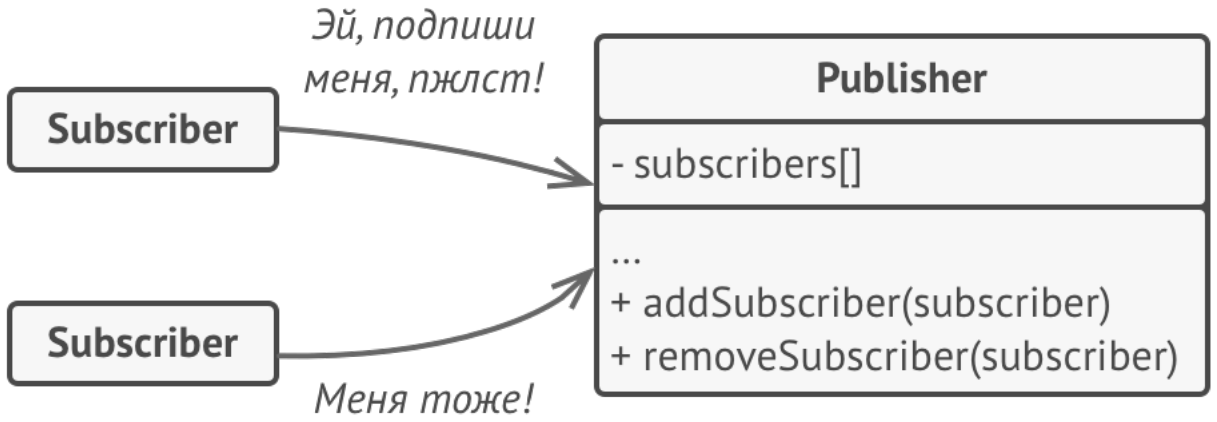
Command

- Позволяет представлять действие в виде объекта.



Observer

- Позволяет одним объектам «подписываться» на события, происходящие с другими объектами.



А ещё есть

- Chain of Responsibility
- Memento
- State
- Strategy
- Template Method
- Visitor

Почитать

- https://refactoring.guru/ru/design-patterns/catalog [картинки и некоторые примеры оттуда] (хорошие объяснения и примеры кода на Python)
- https://www.amazon.com/Design-Patterns-Elements-Reusable-Object-Oriented/dp/0201633612
- https://www.tutorialspoint.com/python_design_patterns/index.htm