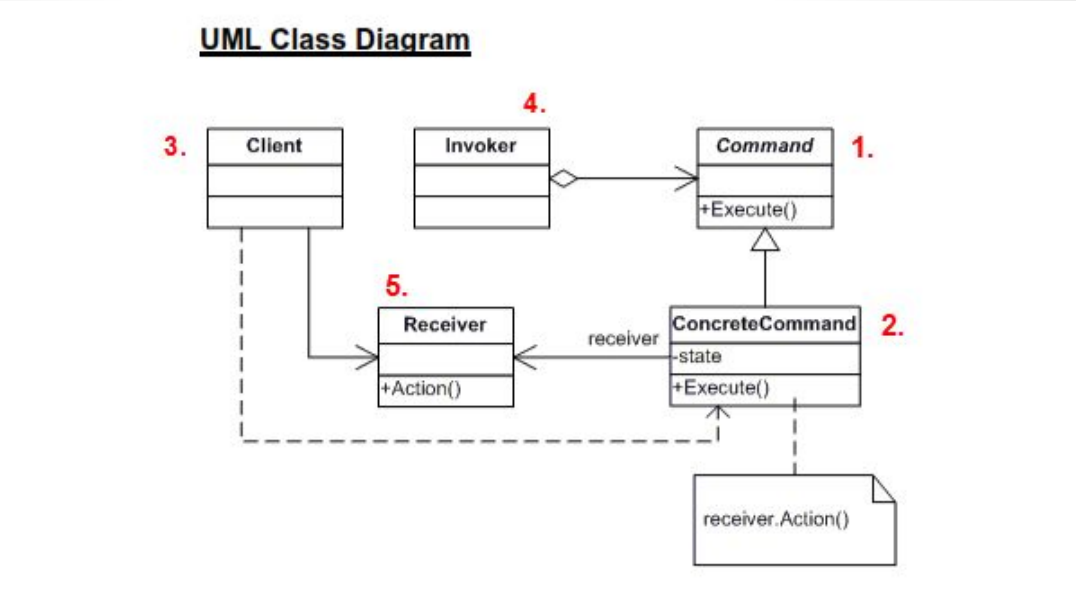
Шаблоны проектирования

Singleton, Factory, Strategy, Composite, Observer

LoftSchool
от мыслителя к создателю

Что такое шаблоны проектирования?

Шаблон проектирования или паттерн - это решение проблемы проектирования программного обеспечения которое часто или регулярно возникает при разработке



Для чего это нужно?

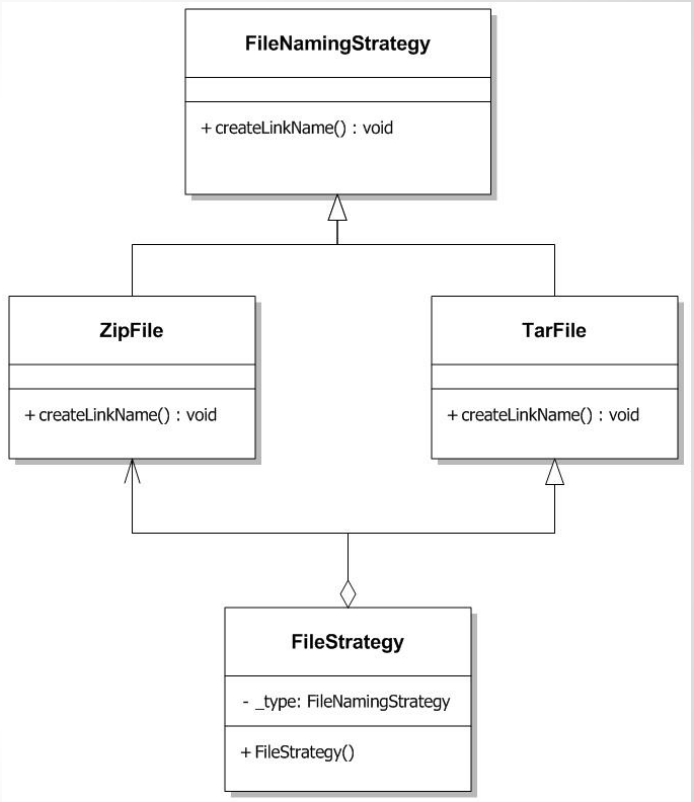
При разработке ПО постоянно что-то меняется - начальник говорит что надо добавить новую функциональность, программисты находят системные ошибки. Многие разработчики сталкивались с этим много раз и в ходе работы они выработали способы предвосхитить эти проблемы или свести их к минимуму их влияние на код.

Еще раз, я не понял?!

Шаблоны проектирования это хорошо протестированные решения проблем, часто встречающиеся при разработке ПО. Чаще всего эти проблемы связаны с гибкостью программных систем. Шаблоны проектирования - это не решения для каких то типов ПО. Это общие решения часто встречающихся проблем при разработке любых типов ПО.

Strategy method

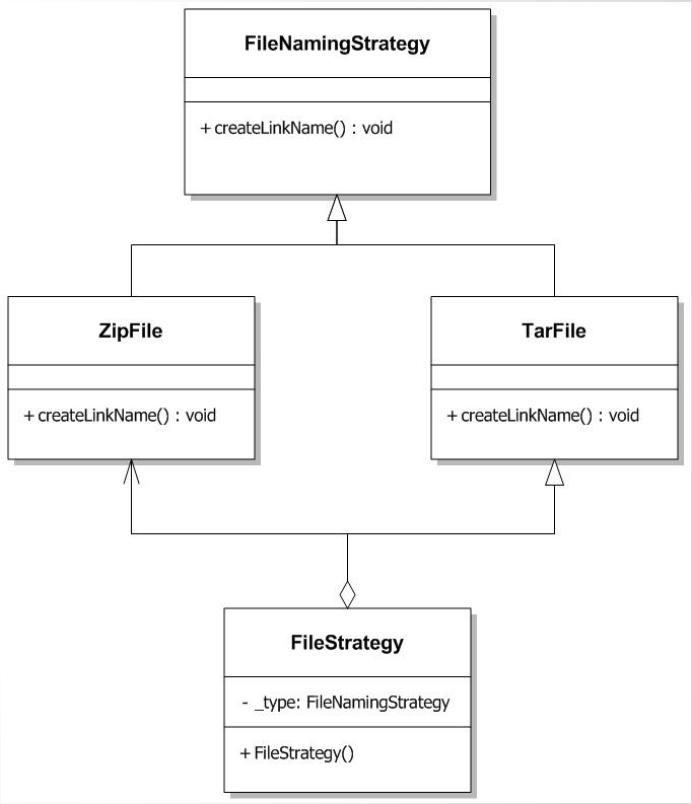
Strategy - поведенческий шаблон проектирования, предназначенный для определения семейства алгоритмов, инкапсуляции каждого из них и обеспечения их взаимозаменяемости. Это позволяет выбирать алгоритм путем определения соответствующего класса. Шаблон Strategy позволяет менять выбранный алгоритм независимо от объектов-клиентов, которые его используют.



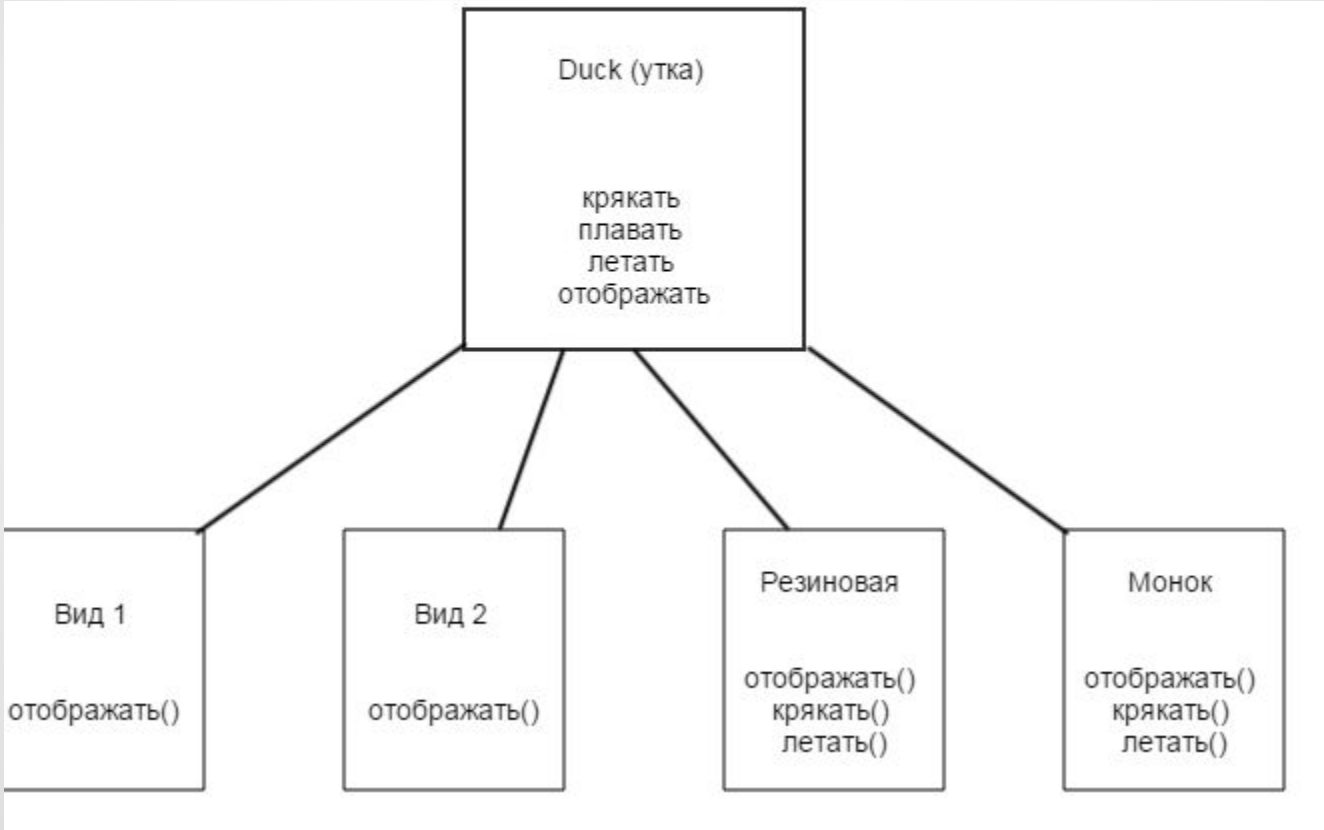
Strategy method

Какие проблемы решает данный шаблон:

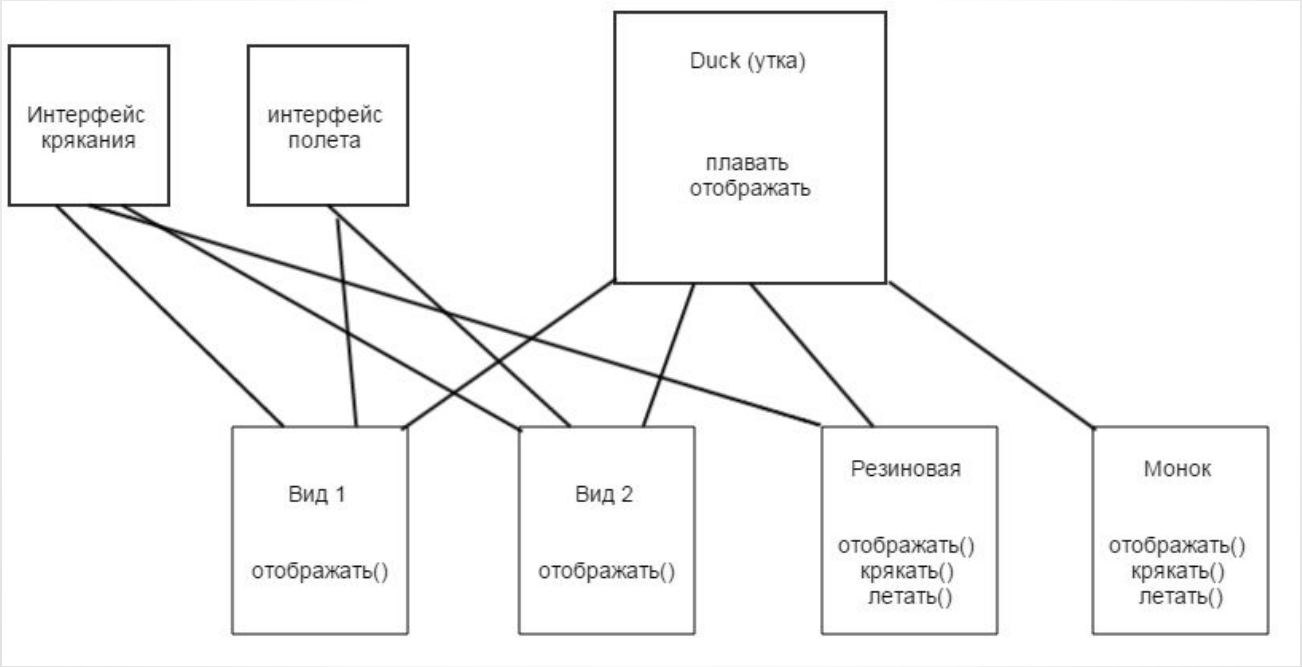
- Обеспечивает различные варианты алгоритмов поведения на основании типа клиента



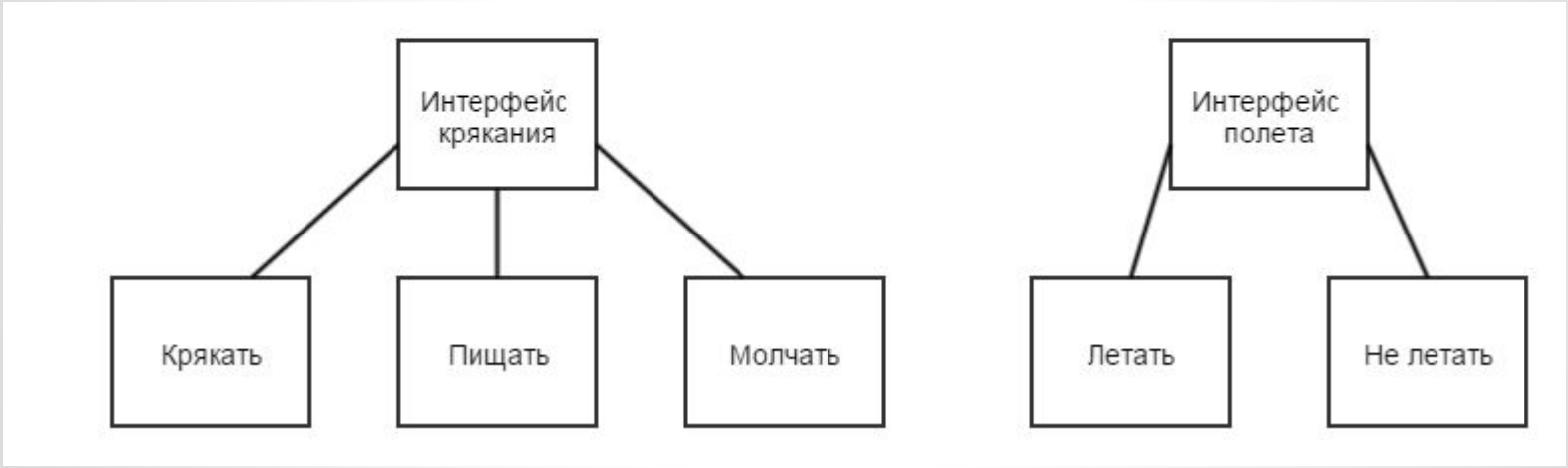
Strategy method поговорим об утках



Strategy method поговорим об утках (вариант 2)



Strategy method поговорим об утках

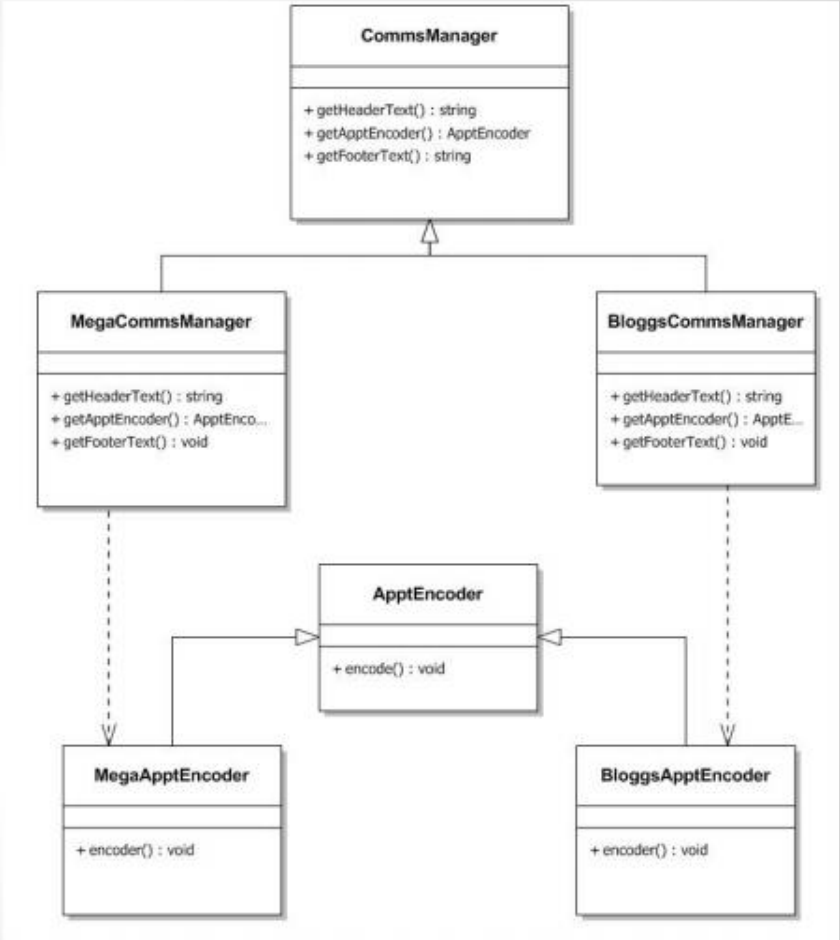


Демонстрация

(пример реализации)

Factory method

Фабричный метод - порождающий шаблон проектирования, предоставляющий подклассам интерфейс для создания экземпляров некоторого класса. В момент создания наследники могут определить, какой класс создавать. Иными словами, Фабрика делегирует создание объектов наследникам родительского класса. Это позволяет использовать в коде программы не специфические классы, а манипулировать абстрактными объектами на более высоком уровне.



Factory method

Какие проблемы решает данный шаблон:

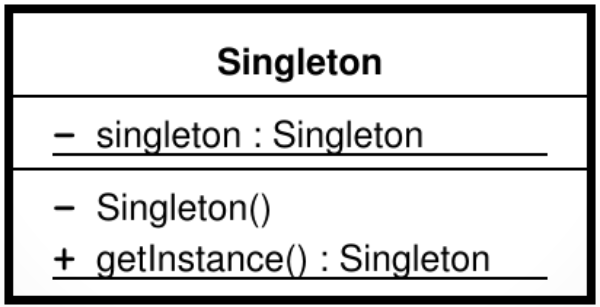
- Позволяет достаточно просто добавлять новые типы для обработки объектов
- Позволяет сделать код создания объектов более универсальным, не привязываясь к конкретным классам, а оперируя лишь общим интерфейсом

Демонстрация

(пример реализации)

Singleton

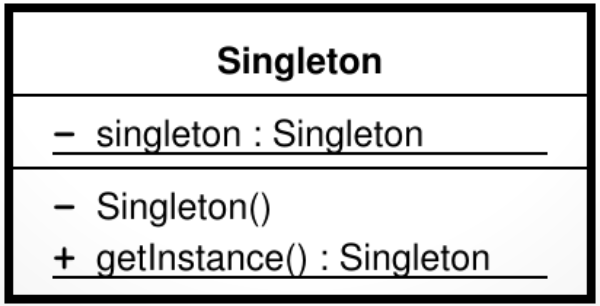
- Одиночка (англ. Singleton) — порождающий шаблон проектирования, гарантирующий, что в однопоточном приложении будет единственный экземпляр класса с глобальной точкой доступа.



Singleton

Какие проблемы решает данный шаблон:

- Создается объект доступный их любого места системы, но сохраняется не в глобальной области видимости, где он может быть случайно изменен
- Уменьшить количество создаваемых объектов в системе, что повышает ее производительность



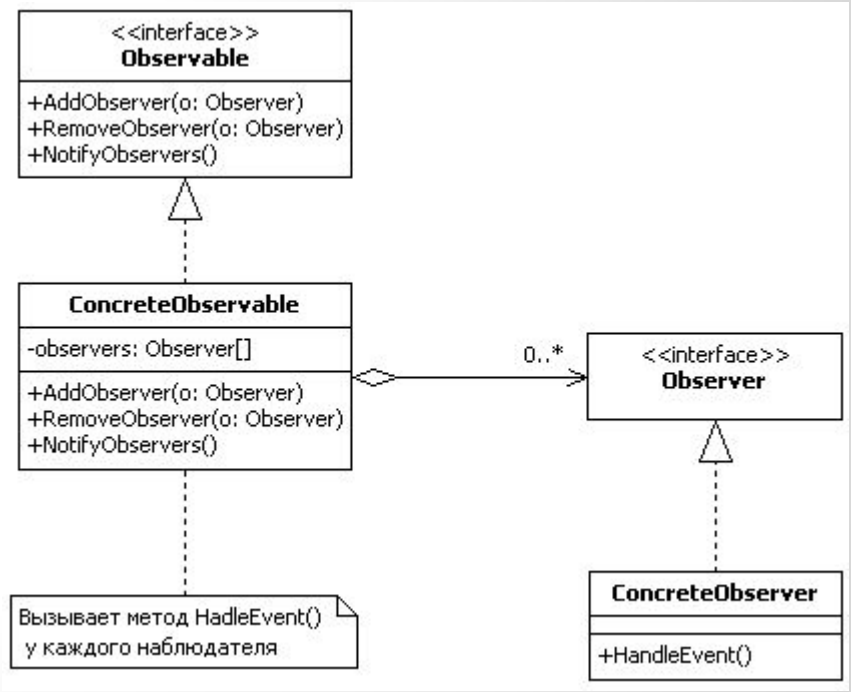
Демонстрация

(пример реализации)

Observer (наблюдатель)

Какие проблемы решает данный шаблон:

Создает механизм у класса, который позволяет получать экземпляру объекта этого класса оповещения от других объектов об изменении их состояния, тем самым наблюдая за ними

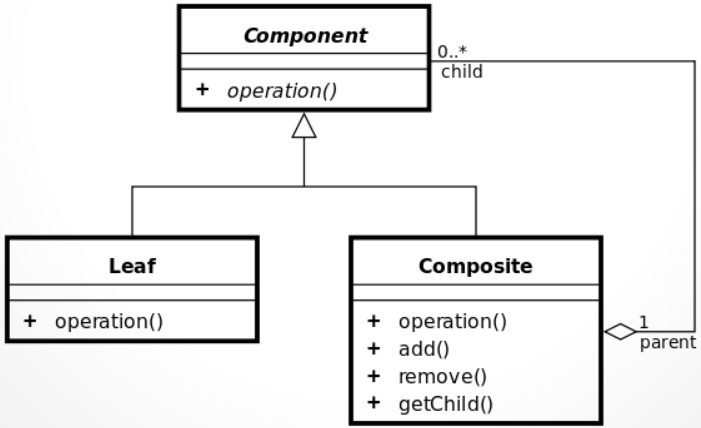


Демонстрация

(пример реализации)

Компоновщик (composite)

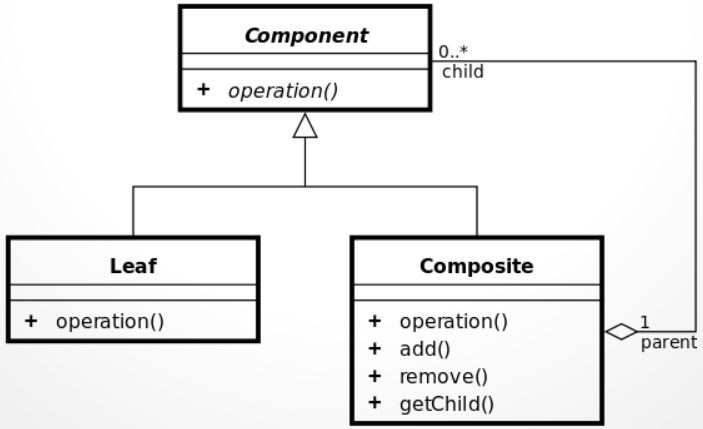
Структурный шаблон проектирования, объединяющий объекты в древовидную структуру для представления иерархии от частного к целому. Компоновщик позволяет клиентам обращаться к отдельным объектам и к группам объектов одинаково.



Компоновщик (composite)

Какие проблемы решает данный шаблон:

- Паттерн определяет иерархию классов, которые одновременно могут состоять из примитивных и сложных объектов, упрощает архитектуру клиента, делает процесс добавления новых видов объекта более простым.

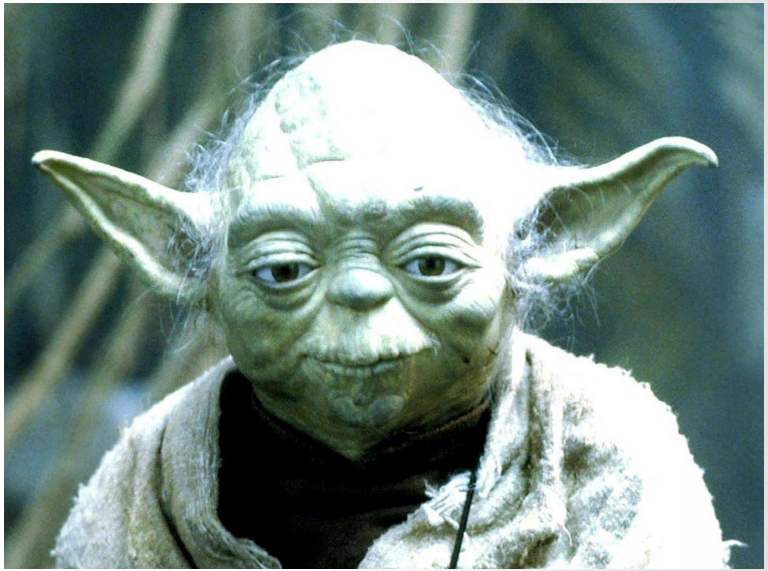


Демонстрация

(пример реализации)

Что нужно сделать после вебинара?

1. Пересмотреть запись вебинара
2. Прочитать методичку
3. Продолжить чтение книг из списка “рекомендованной литературы”
4. Задавать свои вопросы в общем чате
5. Ожидать следующий вебинар



LoftSchool
от мыслителя к создателю