Паттерны проектирования

Содержание

- Введение
- Принципы классификации
- Паттерны проектирования классов
  - Структурные
  - Поведенческие
  - Порождающие
- Архитектурные паттерны
  - Структурные
  - Управления
- Паттерны интеграции приложений
- Антипаттерны
- Заключение
- Литература

Введение

- Паттерн (шаблон) проектирования - это
  - формализованное описание часто встречающейся задачи проектирования
  - удачное решение данной задачи
  - рекомендации по применению этого решения в различных ситуациях.
- Однозначно идентифицируется именем.

Принципы классификации

- Применение паттернов возможно на различных уровнях
- В зависимости от «размера» затрагиваемых элементов, паттерны можно разбить на:
  - Паттерны уровня классов
  - Паттерны уровня архитектуры
  - Паттерны уровня интеграции

Паттерны проектирования классов

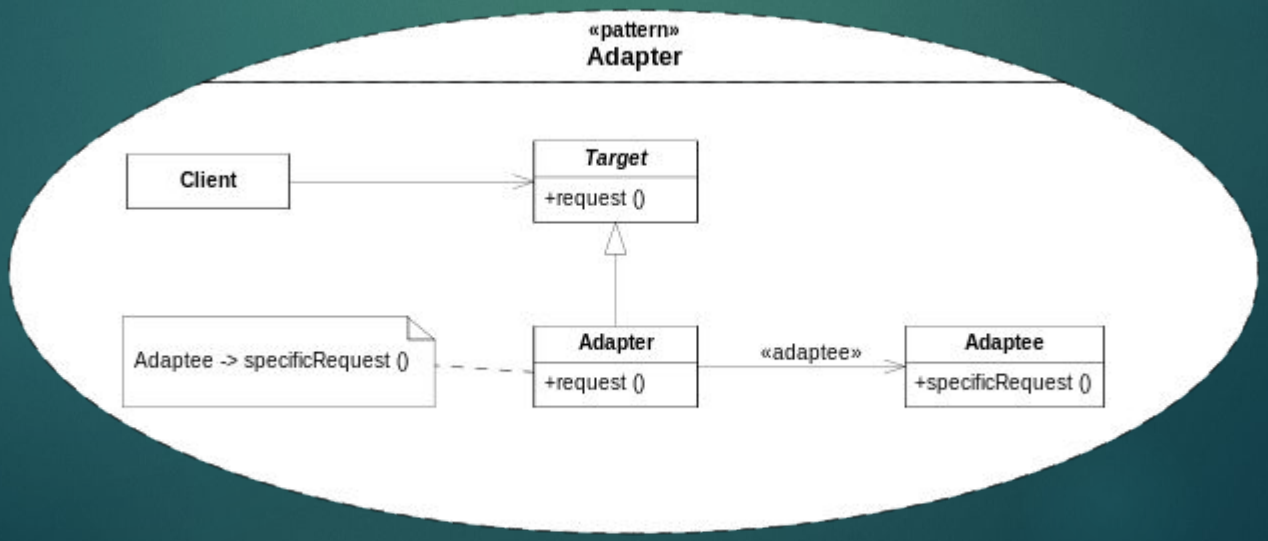
- Управляют «микро» элементами систем
- Имеют альтернативные решения
- Разбиваются на три группы:
  - Структурные
  - Распределения обязанностей (поведенческие)
  - Порождающие (создающие)

Структурные паттерны

- Отвечают за организацию классов и объектов для построение более крупных структур
- Примеры паттернов
  - Adapter (Адаптер)
  - Decorator (Декоратор)
  - Facade (Фасад)
  - Proxy (Прокси)

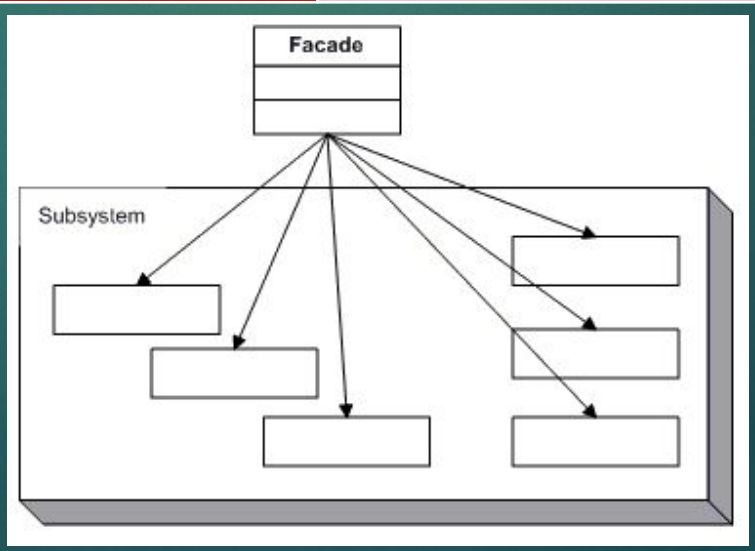
Adapter

Задача	Обеспечить взаимодействие несовместимых интерфейсов
Способ решения	Конвертация исходного интерфейса к целевому посредством промежуточного объекта - адаптера
Следствие	Адаптер позволяет взаимодействовать объектам независимо от различий в их интерфейсе



Facade

Задача	Обеспечить унифицированный интерфейс с набором разрозненных подсистем
Способ решения	Выделение одной точки взаимодействия – фасад, за которой сокрыть все подробности работы
Следствие	Фасад позволяет вносить изменения не теряя совместимости

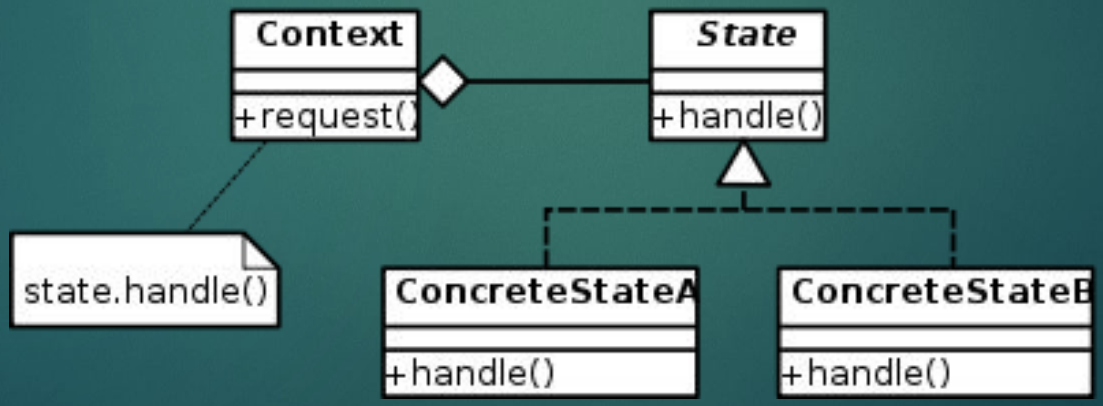


Поведенческие паттерны

- Определяют взаимодействие между объектами, увеличивая таким образом его гибкость
- Примеры паттернов
  - Command (Команда)
  - State (Состояние)
  - Mediator (Посредник)

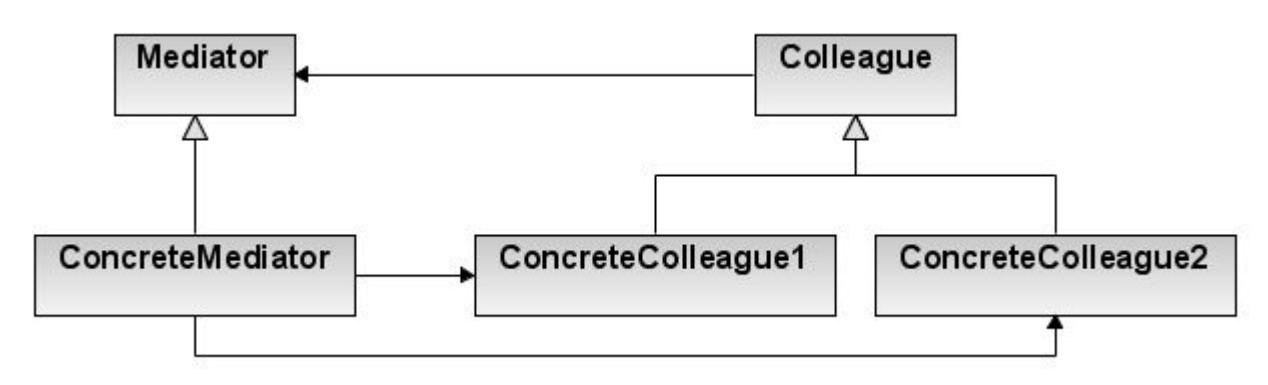
State

Задача	Инкапсулировать различное поведение, основанное на состоянии объекта
Способ решения	Выделение интерфейса «состояние», который реализуют все реальные состояния объекта
Следствие	Работа с объектом, меняющим поведение во время работы, через единый интерфейс



Mediator

Задача	Обеспечить взаимодействие объектов без необходимости явно ссылаться друг на друга
Способ решения	Выделение объекта, инкапсулирующий способ взаимодействия множества объектов
Следствие	Устранена связанность между «Коллегами», управление централизовано



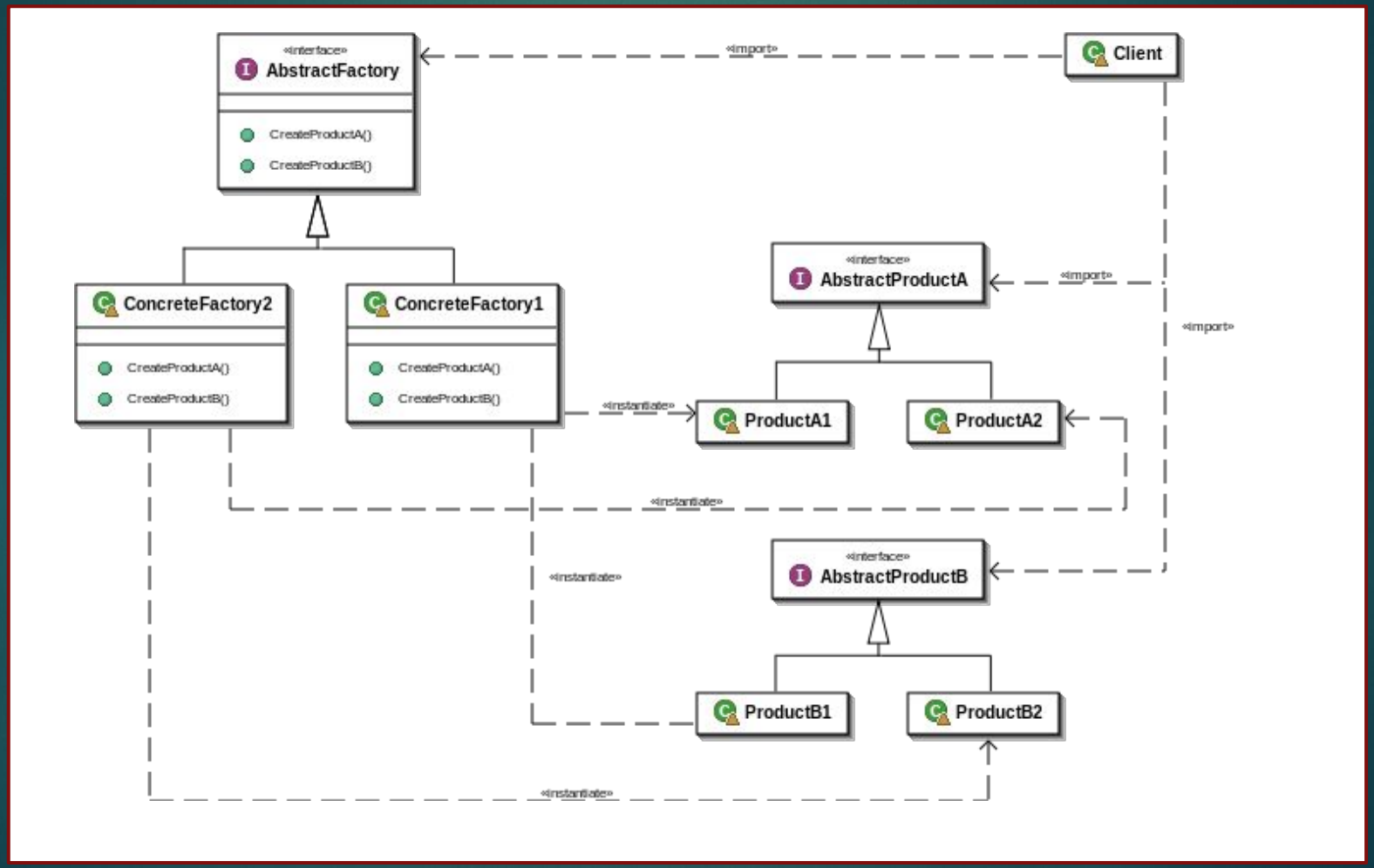
Порождающие паттерны

- Шаблоны, которые абстрагируют процесс создания (инстанцирования) объектов
- Примеры паттернов
  - Abstract Factory (Абстрактная фабрика)
  - Singleton (Синглтон)
  - Prototype (Прототип)

Abstract factory

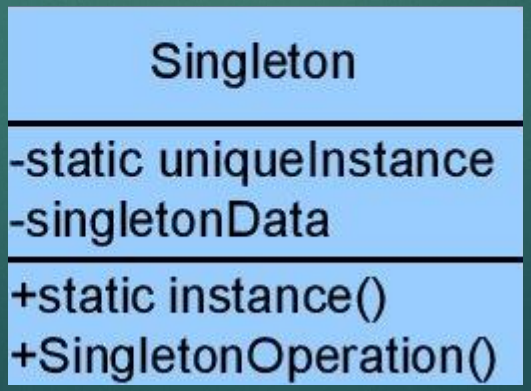
Задача	Создать семейство взаимосвязанных или взаимозависимых обьектов (не специфицируя их конкретных классов).
Способ решения	Создать абстрактный класс, в котором объявлен интерфейс для создания конкретных классов.
Следствие	Изолированы конкретные классы, но их количество фиксировано

Abstract Factory



Singleton

Задача	Необходим лишь один экземпляр специального класса с единственной точкой доступа.
Способ решения	Создать класс и определить статический метод класса, возвращающий этот единственный объект.
Следствие	Получение контролируемого доступа к единому экземпляру.



Архитектурные паттерны

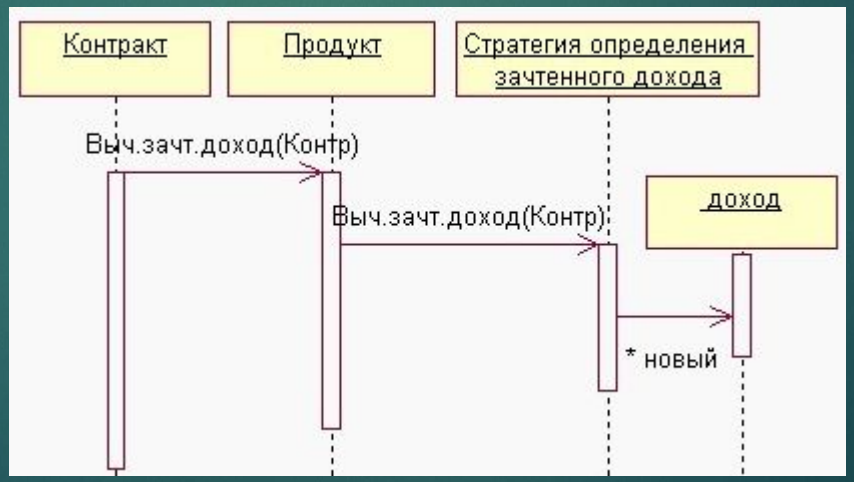
- Организуют подсистемы приложения в целом (на «макро» уровне)
- Делятся на две группы
  - Структурные
  - Управленческие

Структурные паттерны

- Организуют компоненты в подсистемы приложения
- Примеры паттернов
  - Клиент-сервер
  - Репозиторий
  - Объектная модель
  - Слои

Объектная модель

- Система представляется в виде взаимосвязанных бизнес-объектов.
- Каждый из объектов наделяется только функциями, отвечающими его природе



Репозиторий

- Предоставляет общее хранилище для группы подсистем
- Выступает в роли пассивного элемента, которым управляют сами подсистемы
- Все подсистемы согласованы с моделью данных репозитория
- Пример: хранилище платформы DocsVision

Паттерны управления

- Организация взаимодействия между архитектурными элементами системы
- Делятся на три группы
  - Активного управления
  - Управления на событиях
  - Работы с БД
- Примеры паттернов
  - Request-Response
  - Active Record
  - Row Data Gateway

Паттерны интеграции

- Организуют взаимодействие между приложениями
- Выделяются три группы
  - Структурные (вновь!) – отвечают за варианты объединения компонентов в единую метасистему
  - По методу интеграции – описывают способы объединения компонентов
  - Организация обмена информацией – описывает способы сообщения между компонентами

Антипаттерны

- Шаблоны ошибок, которые совершаются при решении различных задач.
- Частью практик хорошего программирования является именно избежание анти-паттернов.

Примеры антипаттернов

- Программирования в режиме КОПИ-ПАСТ
- Спагетти-код и Лазанья-код
- Золотой молоток
- Магические числа
- Hard code и Soft code
- Велосипеды, особенно одноколесные
- God Object
- Подавление ошибок

Заключение

- Паттерн проектирования – инструмент в руках опытного разработчика, который позволяет решать типичные задачи подходящим образом.
- Паттерны не являются абсолютной истиной при программировании
- Каждый разработчик проходит три стадии:
  - Не знание шаблонов.
  - Шаблонное сумасшествие.
  - Шаблонный дзен.

Литература

- Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. Приемы обьектно - ориентированного проектирования Паттерны Проектирования.
- Эрик Фримен, Элизабет Фримен, Кэтти Сьерра, Берт Бейтс. Паттерны проектирования
- Мартин Фаулер, Дейвид Райс, Мэттью Фоммел, Эдвард Хайет, Роберт Ми, Рэнди Стаффорд. Шаблоны корпоративных приложений
- Шпора по паттернам http://habrahabr.ru/post/210288/
- «Энциклопедия» паттернов http://citforum.ru/SE/project/pattern/
- Антипаттерны http://habrahabr.ru/post/59005/