Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО».

Факультет программной инженерии и компьютерной техники

Архитектура программных систем
Лабораторная работа №2

Выполнил
Путинцев Данил Денисович
Группа P3307
Проверил(а)
Преподаватель: Перл Иван Андреевич

Санкт-Петербург 2025 год

Текст задания

Из списка шаблонов проектирования GoF и GRASP выбрать 3-4 шаблона и для каждого из них придумать 2-3 сценария, для решения которых могу применены выбранные шаблоны.

Сделать предположение о возможных ограничениях, к которым можем привести использование шаблона в каждом описанном случае. Обязательно выбрать шаблоны из обоих списков.

Шаблоны проектирования GoF

Singleton

Сценарий №1. Использование для хранения Zookeeper-параметров

Singleton позволяет не повторять затратные инициализации, поэтому его можно использовать для подключения к Zookeeper и хранения в памяти Zookeeper-параметров. Также это обеспечит одну точку доступа к параметрам и весь код сможет ссылаться на них.

Возможные ограничения:

- Зависим от глобального состояния, поэтому нельзя написать хорошие модульные тесты, так как они становятся недетерминированными
- Возможны проблемы с многопоточностью при создании экземпляра. Необходимо использовать механизмы для обеспечения потокобезопасности

Сценарий №2. Использование глобальных объектов для хранения состояний

Так как паттерн Singleton гарантирует создания единственного экземпляра класса, то можно использовать этот экземпляр для хранения состояний глобальных объектов.

Возможные ограничения:

- Экземпляр класса доступен из любой части программы и изменить его состояние может кто угодно, что делает сложным анализ багов
- Код становится более связным, создаются излишние зависимости, что усложняет поддержку и удобство написания кода

Memento

Сценарий №1. Сохранение и восстановление состояния в текстовом редакторе

Шаблон Mementro позволит нам сохранять и восстанавливать прошлые состояния объектов, не раскрывая подробностей их реализации. Допустим, при любом изменении компьютер сохраняет состояние, которое было до изменения. А при нажатии Ctrl+Z он восстанавливает это состояние.

Возможные ограничения:

- Если снимков будет очень много, то это будет занимать много памяти
- Если при любом изменении сохранять состояние, то это негативно скажется на производительности.

Сценарий №2. Система сохранения/загрузки прогресса в компьютерной игре

При прохождении миссии компьютер сохраняет наш прогресс, а также предлагает нам или перезаписать старый снимок либо создать новый.

Возможные ограничения:
- Требуется продумать, что конкретно будем сохранять, так как сохранять состояние каждого объекта будет очень трудозатратно. Всё это усложняет реализацию сохранения и загрузки, так как необходимо решить, что нужно обязательно сохранять, а от чего можно и отказаться
- Если вышло обновление игры, то может добавиться или убраться некоторые параметры, что нарушит целостность снимков

Шаблоны проектирования GRASP

Information Expert

Сценарий №1. Расчет стоимости заказа в интернет-магазине

Приведу пример из моей курсовой работы по Информационным системам. У меня есть класс ProductInfo, который хранит цены на товары определенных продуктов, а также есть класс Order, который хранит информацию о стоимости заказа. Для того чтобы правильно использовать этот паттерн, необходимо прописать логику стоимости отдельного товара в классе ProductInfo, в том числе в этот класс можно прописать логику начисления НДС (хм…, надо сделать парсер с сайта ФНС, а то я помню времена, когда НДС был 18%, а сейчас уже хотят сделать 22%), если товары относятся к разным категориям налогообложения. Помимо этого, в класс Account, можно прописать скидки пенсионерам и другим категориям. А потом подтягивать данные значения в класс Order и рассчитывать финальную версию стоимости заказа, и таким образом не превращать свою кодовую базу в спагетти-код.

Возможные ограничения:
- Сложно реализовать транзакции, так как необходимо обращаться ко множеству классов

Сценарий №2. Валидация пользовательских данных.

Лучше множество валидаторов для каждого класса, чем один большой супер-валидатор для всей формы

Возможные ограничения:
- Создаем 10+ валидаторов вместо одного, возможны проблемы с производительностью
- При изменении спецификации необходимо залазить в 10+ классов для проверки, что увеличивает человеческий фактор