Университет ИТМО

Факультет ФПИ и КТ

Лабораторная работа №2

По дисциплине

“Архитектура программных систем”

Выполнил:

Кочнев Р. Д.

Группа: Р33081

Преподаватель:

Перл И.А.

Санкт-Петербург, 2023

## 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.

### 1 Описание паттернов

#### Singleton (GoF)

*В приложении существует только один экземпляр класса*

Примеры использования:

* Конфигурация приложения – за конфигурацию отвечает 1 класс, в котором задается вся конфигурация.
* Подключение к базе данных – все это делается через 1 класс, в котором создается соединение.
* Логгер – На все приложение создается логгер, который позволяет выводить информацию о работе приложения.

Минусы:

* Тестирование компонентов, которые зависят от одиночки, может быть сложным, поскольку они могут зависеть от глобального состояния, которое может изменяться в различных тестовых сценариях.
* Использование синглтона может стать проблемой в случае необходимости масштабировать приложение горизонтально. Если требуется балансировка нагрузки или распределенные вычисления, могут возникнуть проблемы.
* Использование синглтона может сделать объект доступным глобально в приложении, что может сделать код менее чистым и усложнить отслеживание зависимостей

#### 1.2 Factory Method (GoF)

*Создание различных объектов с разными характеристиками.*

Примеры использования:

* Вывод данных в различные типы фалов – на вход принимает тип выходного файла и создает файл нужного типа
* Создание геометрических фигур – на вход принимает тип фигуры, на выход генерирует фигуру нужного типа
* Создание персонажей NPC в играх – на вход принимает тип NPC, на выход генерирует объект персонажа

Минусы:

* Введение фабричных методов может усложнить структуру кода. Это может сделать систему менее понятной и увеличить сложность ее поддержки.
* Фабричный метод предполагает создание подклассов, что может влиять на производительность, особенно при наличии большого количества различных объектов.
* Для разработчиков, может потребоваться дополнительное время для понимания, какие продукты как создаются.

#### 1.3 Decorator (GoF)

*Добавление функциональности уже готовому объекту*

Примеры использования:

* Добавление различных подтипов фигур – создание квадратов разных цветов, с закруглёнными углами и т.п.
* Оформление текстового документа - добавления дополнительных стилей, шрифтов или цветов к тексту в документе
* Добавление опций к кофейному напитку – добавление молока, сахара или взбитых сливок к напитку

Минусы:

* Последовательное наложение декораторов может привести к увеличению сложности кода и усложнению его понимания, особенно если есть много вложенных декораторов.
* Удаление декоратора может оказаться неудобным, особенно если они используются в разных частях системы, и удаление одного может повлиять на другие компоненты

#### 1.4 Polymorphism (GRASP)

*Способность функции обрабатывать данные разных типов*

Примеры использования:

* Работа с различными типами фигур - единый интерфейс, позволяющий рисовать различные фигуры
* Различные математические операции – единый интерфейс, позволяющий делать различные вычисления
* Воспроизведение различных типов аудио файлов - абстракция для воспроизведения мультимедийных файлов различных форматов

Минусы:

* При изменении интерфейса, используемого для полиморфных объектов, может потребоваться внимательное обновление всех производных классов.
* В полиморфизм может влиять на производительность из-за динамического разрешения типов во время выполнения.
* Если интерфейс или абстрактный класс, используемый для полиморфизма, изменяется, это может создать проблемы совместимости в больших проектах.