Федеральное государственное автономное образовательное учреждение высшего профессионального образования

«Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

Отчет

Практическое занятие №4

Построение и исследование программной системы

на основе шаблонов проектирования

Выполнил: Студент гр. K4112с

Маркова Юлия

Санкт-Петербург

2020 г.

**Цель:**

* изучить шаблоны проектирования GoF и закрепить основы разработки систем на их основе;
* освоить IDE MS Visual Studio в части разработки диаграммы классов.

**Ход работы:**

С помощью упражнений и контрольных заданий руководства: Шаблоны проектирования GoF были реализованы следующие паттерны проектирования:

1. Adapter

Данный паттерн позволяет работать совместно с классами, имеющими несовместимые интерфейсы.

Kost поддерживает интерфейс IGame

AdapterGame поддерживает интерфейс IGame

Monet НЕ поддерживает интерфейс IGame

Через AdapterGame класс Monet может работать с IGame.

Пример приведен на основе задания 1 из практической работы.

Паттерн используется в том случае, когда необходимо адаптировать решение задачи в более общий вид и поддержать единообразную работу разных сущностей.

1. Abstract Factory

Данный шаблон реализует интерфейс, создающий семейство взаимосвязанных объектов с определенными свойствами и общим поведением. При использовании шаблона легко и быстро можно вносить изменения в существующий функционал – добавлять или расширять свойства объектов. Удобно использовать для создания потока объектов со схожими свойствами или поведением, «как на фабрике».

1. Factory Method

Паттерн проектирования, позволяющий делегировать создание экземпляров класса абстрактном интерфейсу. По своей сути является структурным звеном абстрактной фабрики. Используется при реализации шаблона Абстрактная фабрика.

1. Singleton

Шаблон проектирования, позволяющий отслеживать, что в системе должен быть только один объект определенного класса. Может запрещать создание новых экземпляров данного класса. Например, удобно использовать в системе, где необходимо отслеживать работу с файлом или какими-либо данными. Одновременно лишь один пользователь имеет доступ к чтению/записи информации в файл.

1. Strategy

Паттерн, позволяющий описать возможные варианты (свойства) алгоритма и реализующий каждый из них в отдельном классе. В последствии в зависимости от выбора пользователя используется лишь одна Стратегия с помощью класса контекста, в который передается выбор пользователя.

1. Template Method

Шаблон проектирования, разбивающий алгоритм на последовательность шагов (действий). При выполнении программы вызывается шаблонный метод, включающий в себя все декомпозированные действия, но в разной реализации.

Удобно использовать при наличии заданных шаблонных условий для реализации того или иного действия.

1. Facade

Паттерн, способный скрыть большое количество сложных операций и их порядок с помощью упрощенного интерфейса. В реализации получаем пошаговый вызов метода и поочередное выполнение шагов для различных задач, способных к выполнению системой.

Удобно использовать при наличии фиксированных возможных действиях системы, составляя алгоритм, необходимый для получения того или иного результата.

1. Chain of Responsibility

Шаблон, предоставляющий возможность связывать классы, обрабатывающие запросы пользователя таким образом, что каждый из них может либо выполнить запрос сам, либо передать следующему связанному обработчику.

1. Command

Паттерн проектирования, представляющий действие, объект которого содержит в себе как само действие, так и его параметры. Система хранит весь список команд (действий) и позволяет запоминать их последовательность, используя массив данных при выполнении.

Удобно использовать в системах, для которых важным свойством является возможность отмены операции, ее повторение или предварительный итог.

1. Decorator

Шаблон, позволяющий использовать базовую функциональность (параметры, свойства) объектов или добавлять дополнительные, в зависимости от потребностей пользователя.

**Вывод:**

В ходе выполнения практического задания были рассмотрены примеры непосредственного применения порождающих, поведенческих и структурных паттернов проектирования в конкретных задачах.

Паттерны служат для правильного структурирования и упрощения классовых элементов системы.

В работе были реализованы контрольные задания по каждому из приведенных выше паттернов проектирования. По результатам работы можно сделать вывод, что правильное использование шаблонов поможет переиспользовать уже написанный код и избавиться от написания лишнего.