Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №4

Технології розроблення програмного забезпечення

«ШАБЛОНИ «SINGLETON», «ITERATOR», «PROXY», «STATE», «STRATEGY»»

Виконала:

студентка групи ІА-23

Єрмак Д. Р

Перевірив:

Мягкий М. Ю.

Київ 2024

Тема лабораторних робіт:

IRC client (singleton, builder, abstract factory, template method, composite, client-server)

Клієнт для IRC-чатів з можливістю вказівки порту і адреси з'єднання, підтримка базових команд (підключення до чату, створення чату, установка імені, реєстрація, допомога і т.д.), отримання метаданих про канал.

Завдання:

1. Ознайомитися з короткими теоретичними відомостями.
2. Реалізувати частину функціоналу робочої програми у вигляді класів та їхньої взаємодії для досягнення конкретних функціональних можливостей.
3. Застосування одного з розглянутих шаблонів при реалізації програми.

Зміст

Хід роботи:

Крок 1. Теоретичні відомості

Шаблон проєктування — це формалізований опис типового рішення, що часто використовується при проєктуванні інформаційних систем, має загальновживану назву та рекомендації для застосування в різних ситуаціях. Його основні переваги включають полегшення створення структури системи, виділення значимих елементів та зв’язків, що робить модель більш зрозумілою та простою для вивчення. Шаблони підвищують стійкість системи до змін, спрощують її розширення та вдосконалення. Вони представляють собою ескізи архітектурних рішень, які зручно застосовувати у відповідних обставинах.

Розглянемо шаблони, надані для лабораторної роботи

SINGLETON

Шаблон Singleton гарантує створення лише одного екземпляра класу з глобальним доступом до нього, що є зручним для зберігання загального стану, як-от конфігурації чи доступ до бази даних. Програма завжди звертається до

єдиного доступного екземпляра, забезпечуючи його унікальність і захист від випадкових змін.

ITERATOR

Шаблон Iterator полегшує роботу з колекціями, дозволяючи послідовно отримувати доступ до їхніх елементів, не розкриваючи внутрішньої структури. Це особливо зручно для перебору великих масивів або списків, адже ітератор забезпечує єдиний інтерфейс доступу незалежно від способу реалізації колекції.

PROXY

Шаблон Proxy створює об’єкт-замісник, який контролює доступ до основного об’єкта, що є корисним для обмеження або розширення функціональності за допомогою додаткової логіки, наприклад, безпеки чи кешування. Замісник перехоплює виклики до основного об'єкта, виконує додаткові дії і потім передає запит далі.

STATE

Шаблон State дозволяє керувати поведінкою об’єкта залежно від його поточного стану. Це актуально, коли об’єкт може перебувати в різних станах, кожен з яких змінює його поведінку.

STRATEGY

Шаблон Strategy забезпечує можливість вибору різних алгоритмів для виконання певного завдання в процесі роботи програми. Це дозволяє об'єкту змінювати свою поведінку залежно від зовнішніх умов або потреб користувача без зміни основного коду. Кожен алгоритм оформлений як окремий клас, і об'єкт використовує потрібну стратегію відповідно до ситуації, роблячи код гнучким і масштабованим.

Крок 2. Реалізація шаблону проєктування для майбутньої системи

Для реалізації програми IRCClient оберемо шаблон проєктування Singleton, оскільки він гарантує створення лише одного екземпляра класу IRCClient для

всієї програми. У випадку з IRC-чатом важливо, щоб лише один об’єкт керував з’єднанням із сервером, що дозволяє уникнути дублювання підключень та забезпечити централізоване керування сесією чату.

```
package org.example.Singleton;

import org.example.IRCConnection;

public class IRCClient {  12 usages  Yermak Diana
    private static IRCClient specimen;  3 usages
    private final IRCConnection connection;  2 usages

    private IRCClient() { connection = new IRCConnection(); }

    public static synchronized IRCClient getSpecimen() {  3 usages  Yermak Diana
        if (specimen == null) {
            specimen = new IRCClient();
        }
        return specimen;
    }

    public IRCConnection getConnection() { return connection; }
}
```

Рис. 1 – Код класу IRCClient

Клас IRCClient реалізований за патерном Singleton, що є важливим для реалізації IRCClient, оскільки він забезпечує:

1. Єдине з'єднання з сервером: Завдяки єдиному екземпляру класу IRCClient програма зберігає лише одне з’єднання із сервером через об’єкт IRCConnection. Це економить ресурси, оскільки уникає створення зайвих підключень.
2. Контроль доступу: Метод getSpecimen() забезпечує централізований доступ до єдиного екземпляра IRCClient, що спрощує керування з’єднанням з IRC-сервером та його використання в різних частинах програми.
3. Синхронізація та потокобезпечність: Завдяки ключовому слову synchronized метод getSpecimen() запобігає можливості одночасного створення декількох екземплярів в багатопотоковому середовищі, що дуже важливо для стабільної роботи чату.

У нашому випадку Singleton забезпечує контрольований доступ до єдиного з'єднання з сервером і допомагає уникнути проблем, пов'язаних з дублюванням підключень, що значно підвищує ефективність та надійність майбутньої програми IRCClient.

Крок 3. Зображення структури шаблону

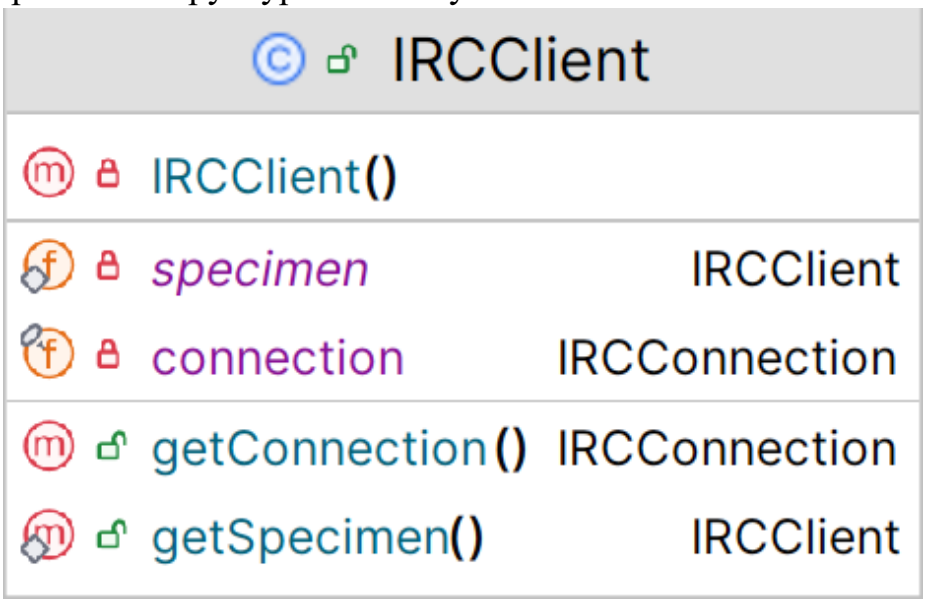

Рис. 2 – Структура шаблону Singleton

Шаблон Singleton, який реалізовано в класі IRCClient, вважається «анти-шаблоном» або ж поганою практикою проєктування. Це зумовлено тим, що Singleton фактично діє як глобальна змінна, яка має стан. Такий підхід ускладнює відстеження та підтримку стану глобальних об'єктів, а також ускладнює тестування, оскільки по всьому коду існує жорстка прив'язка до єдиного екземпляра IRCClient. Будь-які зміни в реалізації можуть потребувати суттєвого оновлення багатьох частин коду. Альтернативні рішення для контролю доступу часто є більш гнучкими та дозволяють уникнути прив’язки до єдиного екземпляра. Але перевагами використання даного шаблону є те, що це гарантує наявність єдиного екземпляру класу, отже єдине з’єднання з сервером та потокобезпечність.

Висновок: У ході виконання даної лабораторної роботи було розглянуто структуру та призначення шаблонів проєктування «SINGLETON», «ITERATOR», «PROXY», «STATE», «STRATEGY». Кожен з розглянутих шаблонів має свої переваги та недоліки. Далі було обрано шаблон, який підійде

для реалізації майбутньої системи. Останнім етапом була реалізація шаблону та дослідження його структури, переваг у використанні для заданої системи.