

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота № 6
з дисципліни «Технології розроблення
програмного забезпечення»
Тема: «Патерни проектування»

Виконав:
студент групи IA-32
Задорожний Костянтин
Леонідович

Перевірив:
Мягкий Михайло
Юрійович

Київ - 2025

Тема: Патерни проектування.

Мета: Вивчити структуру шаблонів «Abstract Factory», «Factory Method», «Memento», «Observer», «Decorator» та навчитися застосовувати їх в реалізації програмної системи.

Завдання

- Ознайомитись з короткими теоретичними відомостями.
- Реалізувати частину функціоналу робочої програми у вигляді класів та їхньої взаємодії для досягнення конкретних функціональних можливостей.
- Реалізувати один з розглянутих шаблонів за обраною темою.
- Реалізувати не менше 3-х класів відповідно до обраної теми.
- Підготувати звіт щодо виконання лабораторної роботи. Поданий звіт повинен містити: діаграму класів, яка представляє використання шаблону в реалізації системи, навести фрагменти коду по реалізації цього шаблону.

Предметна область:

22. FTP-server (state, builder, memento, template method, visitor, client-server)
FTP-сервер повинен вміти коректно обробляти і відправляти відповіді під протоколу FTP, з можливістю створення користувачів (з паролями) і доступних їм папок, розподілу прав за стандартною схемою (rwe), ведення статистики з'єднань, обмеження максимальної кількості підключень і максимальної швидкості поширення глобально і окремо для кожного облікового запису.

Хід роботи

Для виконання цієї лабораторної роботи я використав патерн: «Знімок» (Memento).

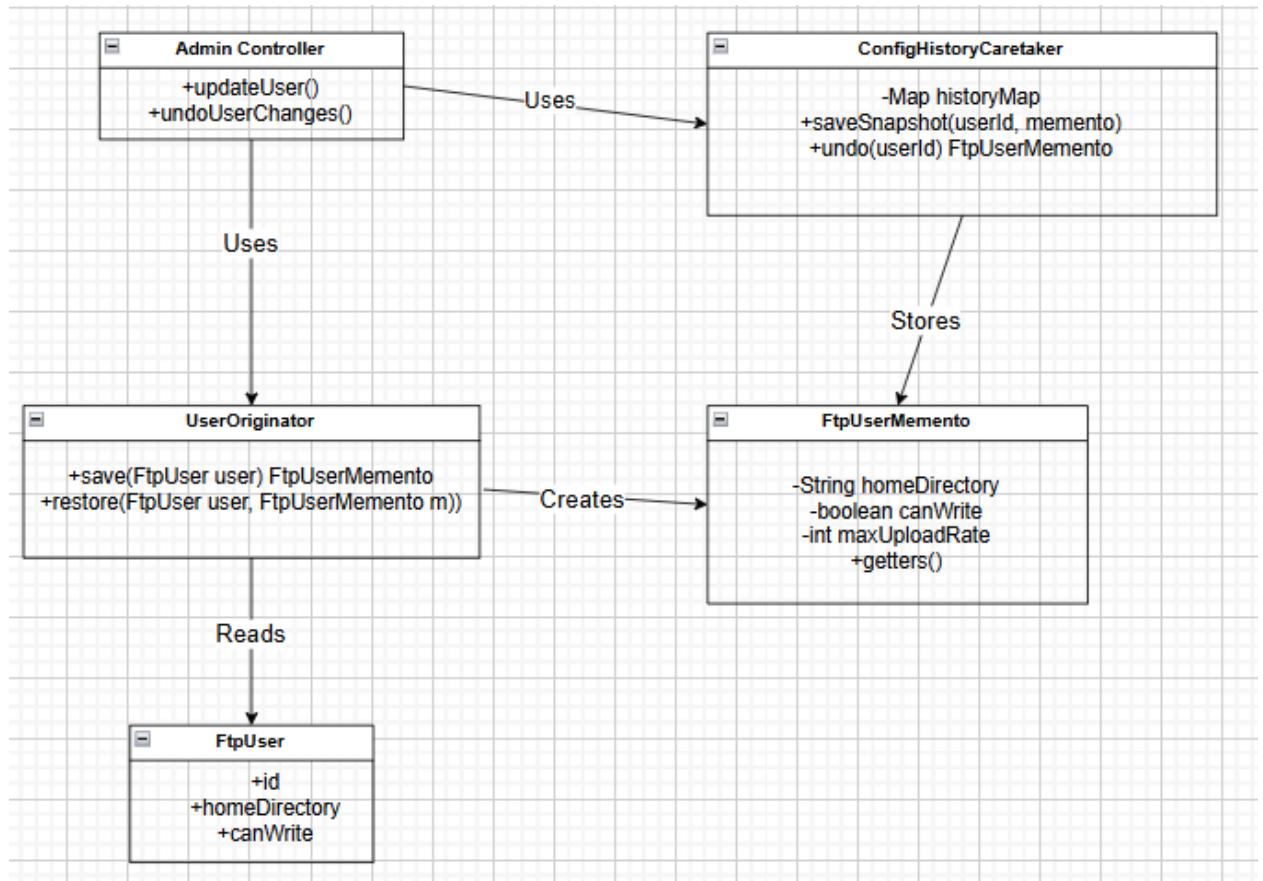
Цей поведінковий патерн проектування дозволить зберігати та відновлювати минулий стан об'єкта, не розкриваючи подroбниць його реалізації.

Структура патерну:

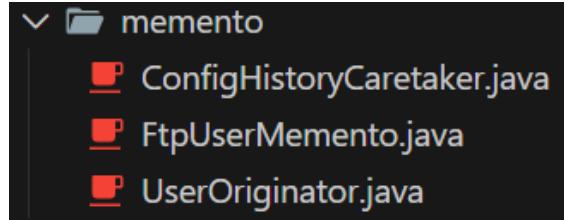
1. Originator (Творець): Об'єкт, стан якого ми хочемо зберігати. Він створює знімок (CreateMemento) і може відновитися з нього (SetMemento). У проекту це буде клас-обгортка над FtpUser.
2. Memento (Знімок): Простий об'єкт, що зберігає стан творця. Він не повинен дозволяти іншим об'єктам змінювати цей стан (імутабельний).
3. Caretaker (Опікун): Відповідає за зберігання знімків (історію). Він не повинен змінювати вміст знімка, лише зберігати його та віддавати творцю, коли потрібно.

Додам можливість редагувати користувачів. Перед збереженням змін Memento буде робити "Знімок". Якщо адмін натисне кнопку "Undo" - відновить стан з історії.

Діаграма структури патерну Memento для FTP server:



Реалізація патерну у коді:



ConfigHistoryCaretaker:

```
1 package com.example.ftp.service.memento;
2
3 import org.springframework.stereotype.Service;
4 import java.util.HashMap;
5 import java.util.Map;
6 import java.util.Stack;
7
8 @Service
9 public class ConfigHistoryCaretaker {
10
11     private final Map<Long, Stack<FtpUserMemento>> historyMap = new HashMap<>();
12
13     public void saveSnapshot(Long userId, FtpUserMemento memento) {
14         historyMap.putIfAbsent(userId, new Stack<>());
15         historyMap.get(userId).push(memento);
16     }
17
18     public FtpUserMemento undo(Long userId) {
19         if (!historyMap.containsKey(userId) || historyMap.get(userId).isEmpty()) {
20             return null;
21         }
22         return historyMap.get(userId).pop();
23     }
24
25     public boolean hasHistory(Long userId) {
26         return historyMap.containsKey(userId) && !historyMap.get(userId).isEmpty();
27     }
28 }
```

FtpUserMemento:

```
1 package com.example.ftp.service.memento;
2
3 public class FtpUserMemento {
4     private final String homeDirectory;
5     private final boolean canWrite;
6     private final boolean canRead;
7     private final int maxUploadRate;
8     private final int maxDownloadRate;
9
10    public FtpUserMemento(String homeDirectory, boolean canWrite, boolean canRead, int maxUploadRate,
11                           int maxDownloadRate) {
12        this.homeDirectory = homeDirectory;
13        this.canWrite = canWrite;
14        this.canRead = canRead;
15        this.maxUploadRate = maxUploadRate;
16        this.maxDownloadRate = maxDownloadRate;
17    }
18
19    public String getHomeDirectory() {
20        return homeDirectory;
21    }
22
23    public boolean isCanWrite() {
24        return canWrite;
25    }
26
27    public boolean isCanRead() {
28        return canRead;
29    }
30 }
```

UserOriginator:

```
1 package com.example.ftp.service.memento;
2
3 import com.example.ftp.model.FtpUser;
4 import org.springframework.stereotype.Component;
5
6 @Component
7 public class UserOriginator {
8
9     public FtpUserMemento save(FtpUser user) {
10         return new FtpUserMemento(
11             user.getHomeDirectory(),
12             user.isCanWrite(),
13             user.isCanRead(),
14             user.getMaxUploadRate(),
15             user.getMaxDownloadRate());
16     }
17
18     // Метод відновлення (Restore State)
19     public void restore(FtpUser user, FtpUserMemento memento) {
20         user.setHomeDirectory(memento.getHomeDirectory());
21         user.setCanWrite(memento.isCanWrite());
22         user.setCanRead(memento.isCanRead());
23         user.setMaxUploadRate(memento.getMaxUploadRate());
24         user.setMaxDownloadRate(memento.getMaxDownloadRate());
25     }
26 }
```

Результати:

Edit User: admin

Username:

Home Directory:

Can Write:

Max Upload Rate (0 = unlimited):

Update & Save StateBack to List

Edit User: admin

Username:

admin

Home Directory:

C:/ftp/admin

Can Write:

Max Upload Rate (0 = unlimited):

10000

[Update & Save State](#)

[Undo Changes](#)

[Back to List](#)

Username	Home Directory	Can Write	Upload Limit	Actions
admin	C:/ftp/admin	true	10000	Edit

Завдяки патерну Memento я реалізував редактування інформації о користувачах FTP серверу, а саме можливість зміни директорії та дозволену максимальну швидкість передачи даних на сервер.

Висновки:

Я вивчив структуру шаблонів «Abstract Factory», «Factory Method», «Memento», «Observer», «Decorator» та навчився застосовувати їх в реалізації програмної системи.