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 *№5*

із дисципліни ***«Технології розробки програмного забезпечення»***

Тема: ***«Патерни проектування»***

Виконав:

Студент групи ІА-34

Ястремський Богдан

Перевірив:

асистент кафедри ІСТ

Мягкий Михайло Юрійович

Київ — 2025

Тема: Патерни проектування.

Мета: Вивчити структуру шаблонів «Adapter», «Builder», «Command», «Chain of responsibility», «Prototype» та навчитися застосовувати їх в реалізації програмної системи.

Застосунок (№6): Web-browser (proxy, chain of responsibility, factory method, template method, visitor, p2p).

Веб-браузер повинен мати можливість зробити наступне: мати адресний рядок для введення адреси сайту, переміщатися і відображати структур html документа, переглядати підключений javascript та css файли, перегляд всіх підключених ресурсів (зображень), коректна обробка відповідей з сервера (коди відповідей HTTP) – переходи при перенаправленнях, відображення сторінок 404 і 502/503.

Короткі теоретичні відомості:

Шаблон "Adapter" (Адаптер) використовується для адаптації інтерфейсу одного об'єкту до іншого. Наприклад, існує декілька бібліотек для роботи з принтерами, проте кожна має різний інтерфейс (хоча однакові можливості і призначення). Має сенс розробити уніфікований інтерфейс (сканування, асинхронне сканування, двостороннє сканування, потокове сканування і тому подібне), і реалізувати відповідні адаптери для приведення бібліотек до уніфікованого інтерфейсу.

Шаблон «Builder» (Будівельник) використовується для відділення процесу створення об'єкту від його представлення. Це доречно у випадках, коли об'єкт має складний процес створення (наприклад, Webсторінка як елемент повної відповіді web- сервера) або коли об'єкт повинен мати декілька різних форм створення (наприклад, при конвертації тексту з формату у формат).

Шаблон "command" (команда) перетворить звичайний виклик методу в клас. Таким чином дії в системі стають повноправними об'єктами. Об'єкт команда сама по собі не виконує ніяких фактичних дій окрім перенаправлення

запиту одержувачеві (тобто команди все ж виконуються одержувачем), однак ці об'єкти можуть зберігати дані для підтримки додаткових функцій відміни, логування і інше.

Шаблон «Chain of responsibility» (Ланцюжок відповідальності) частково можна спостерігати в житті, коли підписання відповідного документу проходить від його складання у одного із співробітників компанії через менеджера і начальника до головного начальника, який ставить свій підпис.

Шаблон «Prototype» (Прототип) використовується для створення об'єктів за «шаблоном» (чи «кресленням», «ескізом») шляхом копіювання шаблонного об'єкту, який називається прототипом. Для цього визначається метод «клонувати» в об'єктах цього класу.

Хід роботи:

1. Повна діаграма класів (без шаблону).

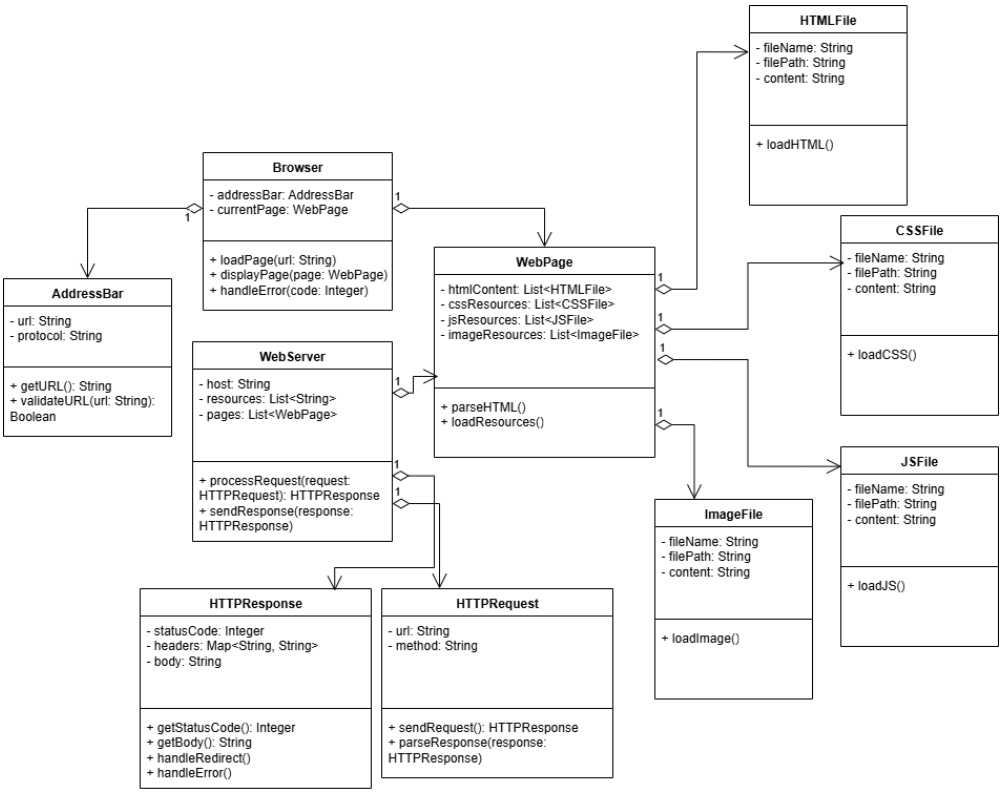


Рис. 5.1 – Діаграма класів системи

2. Діаграма класів з використанням шаблону «Chain of Responsibility»:

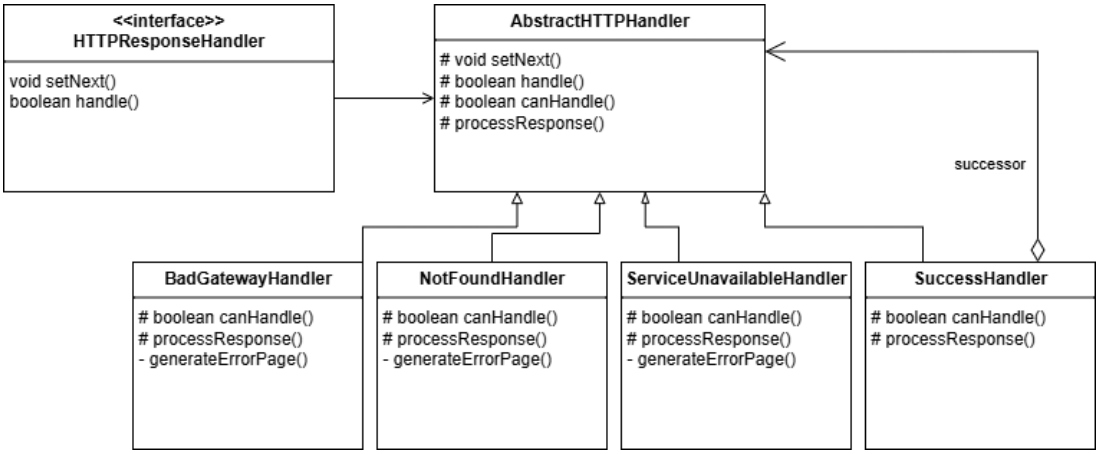


Рис. 5.2 – Діаграма класів системи з використанням шаблону «Chain of Responsibility»

Я визначаю інтерфейс HTTPResponseHandler, який визначає в собі методи для обробки відповіді і перевірки можливості обробки. Потім я

визначаю абстрактний клас AbstractHTTPHandler, який реалізує ці методи.

Потім я створюю наслідників від цього класу, для кожного типу помилки:

- **SuccessHandler.java** - обробник 200 OK
- **NotFoundHandler.java** - обробник 404
- **BadGatewayHandler.java** - обробник 502
- **ServiceUnavailableHandler.java** - обробник 503

Цей патерн допоміг полегшити функцію для локального сервера, і тепер замість хард-коду html-сторінок про помилку, я їх визначаю в класах, а ланцюг відповідальності вже видає конкретну сторінку..

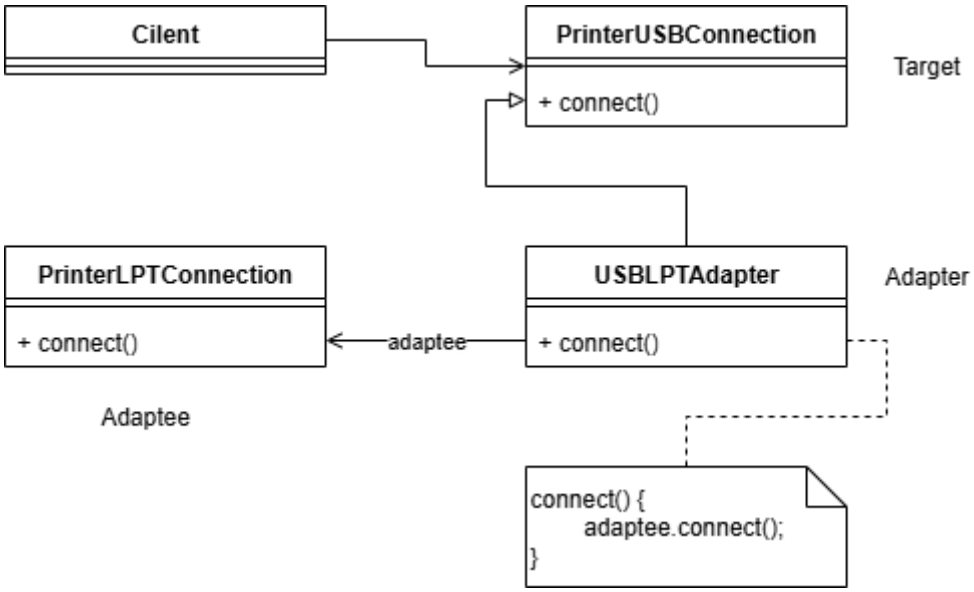
Висновок: на даному лабораторному занятті я продовжив знайомством з поняттям патернів проектування, які є їх види, та їхні плюси і мінуси. В якості використовуваного патерну я обрав «Chain of responsibility», та імплементував його як обробник помилок в локальному сервері (де тестуються помилки 404, 502, 503). Цей патерн ідеальний тим, що легко можна додати нову помилку для обробки, і для цього не доведеться хард-кодити сторінки помилок прямо в обробнику запитів – можна одразу за ланцюжком визначити проблему і повернути сторінку з помилкою. Також мені не треба визначати, хто буде її обробляти – це все робить ланцюжок, який легко модифікувати.

Відповіді на контрольні питання:

1. Яке призначення шаблону «Адаптер»?

   Шаблон "Adapter" (Адаптер) використовується для адаптації інтерфейсу одного об'єкту до іншого.

2. Нарисуйте структуру шаблону «Адаптер».



3. Які класи входять в шаблон «Адаптер», та яка між ними взаємодія?

   Клієнт (Client): Использует интерфейс, к которому привык.

   Адаптер (Adapter): Посередник, який приймає виклики від клієнта і перетворює їх у зрозумілі для Adaptee.

   Adaptee: Об'єкт з несумісним з клієнтом інтерфецсом.

   Target Interface: Очікуваний клієнтом інтерфейс (реалізується адаптером).

4. Яка різниця між реалізацією «Адаптера» на рівні об’єктів та на рівні класів?

   Адаптер на рівні об'єктів:

   1. Створюється окремий об'єкт, який містить адаптовані методи.
   2. Цей адаптер «обгортає» вже існуючий об'єкт і викликає його методи через свої власні.
   3. Це гнучкий підхід, оскільки адаптер працює з конкретними екземплярами класів.

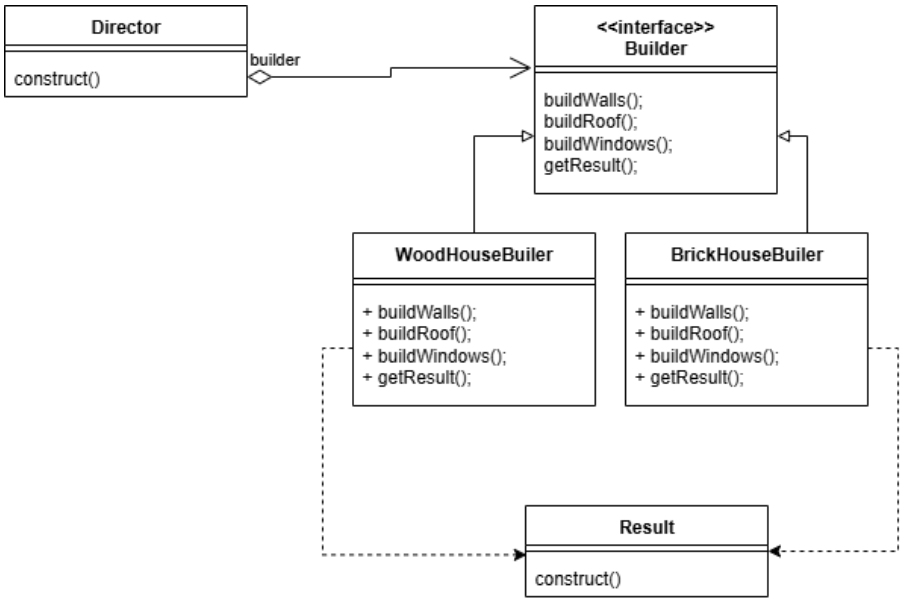
Адаптер на рівні класів:

1. Створюється новий клас, що наслідує або імплементує інтерфейс, що вимагається.
2. Адаптує один клас до іншого через зміну структури на рівні класу.
3. Цей підхід змінює поведінку для всіх екземплярів класу.

5. Яке призначення шаблону «Будівельник»?

   Шаблон «Builder» (Будівельник) використовується для відділення процесу створення об'єкту від його представлення.

6. Нарисуйте структуру шаблону «Будівельник».



7. Які класи входять в шаблон «Будівельник», та яка між ними взаємодія?

   Builder: Абстрактний клас/інтерфейс, який оголошує методи для наслідників.

   Director: Знає як використоувати будівельника, і оголошує метод побудови, який містить всі методи будівника.

   Product: Кінцевий об'єкт, що створюється.

8. У яких випадках варто застосовувати шаблон «Будівельник»?

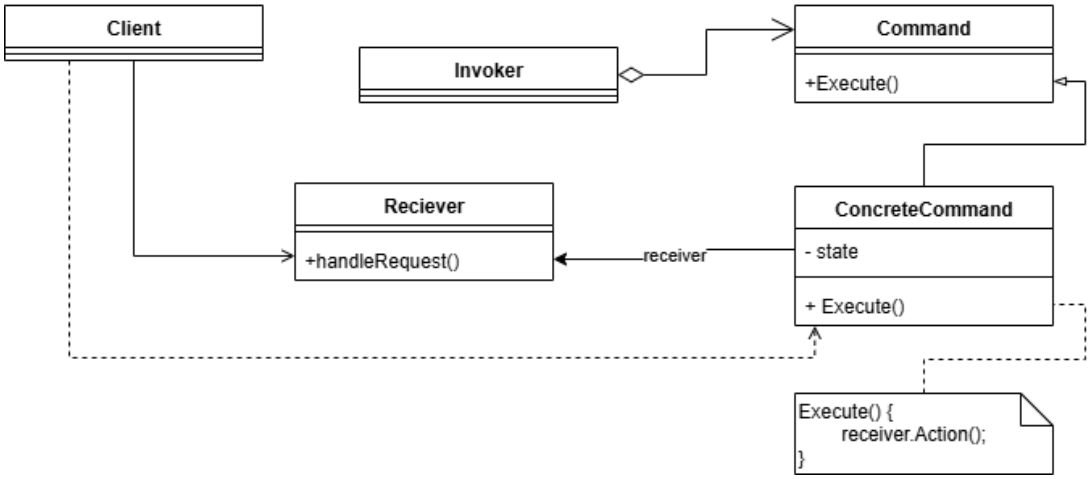
   Це доречно у випадках, коли об'єкт має складний процес створення (наприклад, Webсторінка як елемент повної відповіді web- сервера)

або коли об'єкт повинен мати декілька різних форм створення (наприклад, при конвертації тексту з формату у формат).

9. Яке призначення шаблону «Команда»?

   Шаблон "command" (команда) перетворить звичайний виклик методу в клас. Таким чином дії в системі стають повноправними об'єктами.

10. Нарисуйте структуру шаблону «Команда».



11. Які класи входять в шаблон «Команда», та яка між ними взаємодія?

   Ініціатор (Invoker): Клієнт, який має посилання на об'єкт Command та викликає його метод execute().

   Команда (Command): Абстрактний інтерфейс, що оголошує метод execute().

   Конкретна команда (Concrete Command): Клас, який реалізує інтерфейс Command. У ньому є посилання на об'єкт Receiver і метод execute() викликає метод Receiver.

   Приймач (Receiver): Об'єкт, що виконує реальну дію, яка була інкапсульована в об'єкт Command.

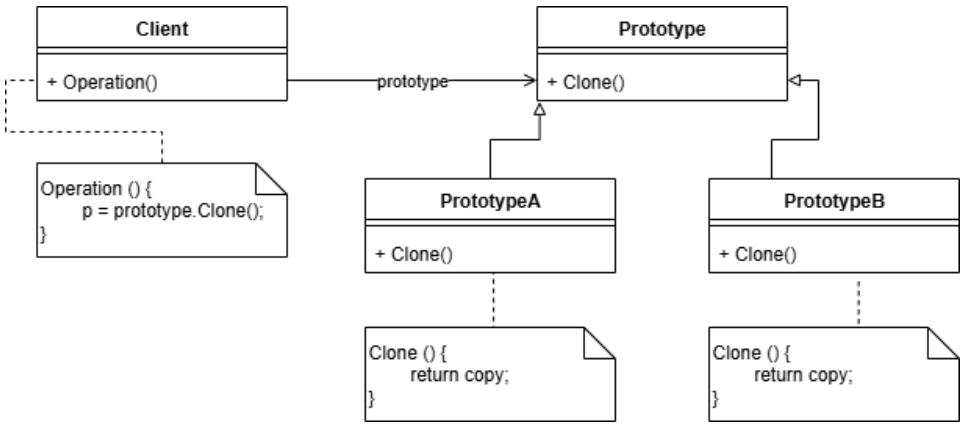
12. Розкажіть як працює шаблон «Команда».

   Шаблон «Команда» дозволяє інкапсулювати запит як об'єкт, що включає всі необхідні параметри для виконання дії, і передати його до виконавця (отримувача). Це дозволяє розділити ініціатора запиту (відправника) і виконавця (отримувача), що забезпечує більшу гнучкість у реалізації команд.

13. Яке призначення шаблону «Прототип»?

    Шаблон «Prototype» (Прототип) використовується для створення об'єктів за «шаблоном» (чи «кресленням», «ескізом») шляхом копіювання шаблонного об'єкту, який називається прототипом.

14. Нарисуйте структуру шаблону «Прототип».



15. Які класи входять в шаблон «Прототип», та яка між ними взаємодія?

    Клієнт (Client): Створює та використовує прототипи. Він знає, який об'єкт потрібно клонувати, але йому не важливо, чи він є екземпляром класу, успадкованого від «Прототипу», чи ні, оскільки він може викликати метод clone() у будь-якого об'єкта.

    Прототип (Prototype): Абстрактний клас або інтерфейс, який повідомляє метод clone(). Усі класи, які мають бути клонованими, реалізують цей метод.

    Конкретний прототип (ConcretePrototype): Клас, який реалізує метод clone() та повертає копію свого власного об'єкта.

16. Які можна привести приклади використання шаблону «Ланцюжок відповідальності»?

    - Обробка запитів у веб-сервісах
    - Обробка подій в графічних інтерфейсах
    - Система обробки помилок (як в даній лабораторній)
    - Обробка запитів в бізнес-процесах