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

ЛАБОРАТОРНА РОБОТА №5

з дисципліни «Технології розроблення програмного забезпечення»

Тема: «Патерни проектування. FTP-server»

Виконав:
студент групи ІА-34
Сухоручкін Гліб

Перевірив:
Асистент кафедри ІСТ
Мягкий М.Ю.

Київ – 2025

Зміст:

Вступ

Метою цієї роботи є вивчити структуру шаблонів «Adapter», «Builder», «Command», «Chain of responsibility», «Prototype» та навчитися застосовувати їх в реалізації програмної системи.

Проєкторваною системою цієї лабороторної роботи є FTP-server (використовуючи state, builder, memento, template method, visitor, client-server).

FTP-сервер повинен вміти коректно обробляти і відправляти відповіді по 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.

Теоретичні відомості

Шаблон «Builder»

Призначення патерну: Шаблон «Builder» (Будівельник) використовується для відділення процесу створення об'єкту від його представлення [6]. Це доречно у випадках, коли об'єкт має складний процес створення (наприклад, Web- сторінка як елемент повної відповіді web- сервера) або коли об'єкт повинен мати декілька різних форм створення (наприклад, при конвертації тексту з формату у формат).

Проблема: Візьмемо процес побудови відповіді на запит web-сервера. Побудова складається з наступних частин: додавання стандартних заголовків (дата/час, ім'я сервера, інш.), код статусу (після пошуку відповідної сторінки на сервері), заголовки відповіді (тип вмісту, інш.), утримуване, інше.

Рішення: Кожен з цих етапів може бути абстрагований в окремий метод будівельника. Це дасть наступні вигоди:

- Гнучкіший контроль над процесом створення сторінки;

- Незалежність від внутрішніх змін – наприклад, зміна назви сервера не сильно порушить процес побудови відповіді;

Переваги та недоліки:

+ Дозволяє використовувати один і той самий код для створення різноманітних продуктів.

- Клієнт буде прив’язаний до конкретних класів будівельників, тому що в інтерфейсі будівельника може не бути методу отримання результату

Хід роботи

Діаграма класів

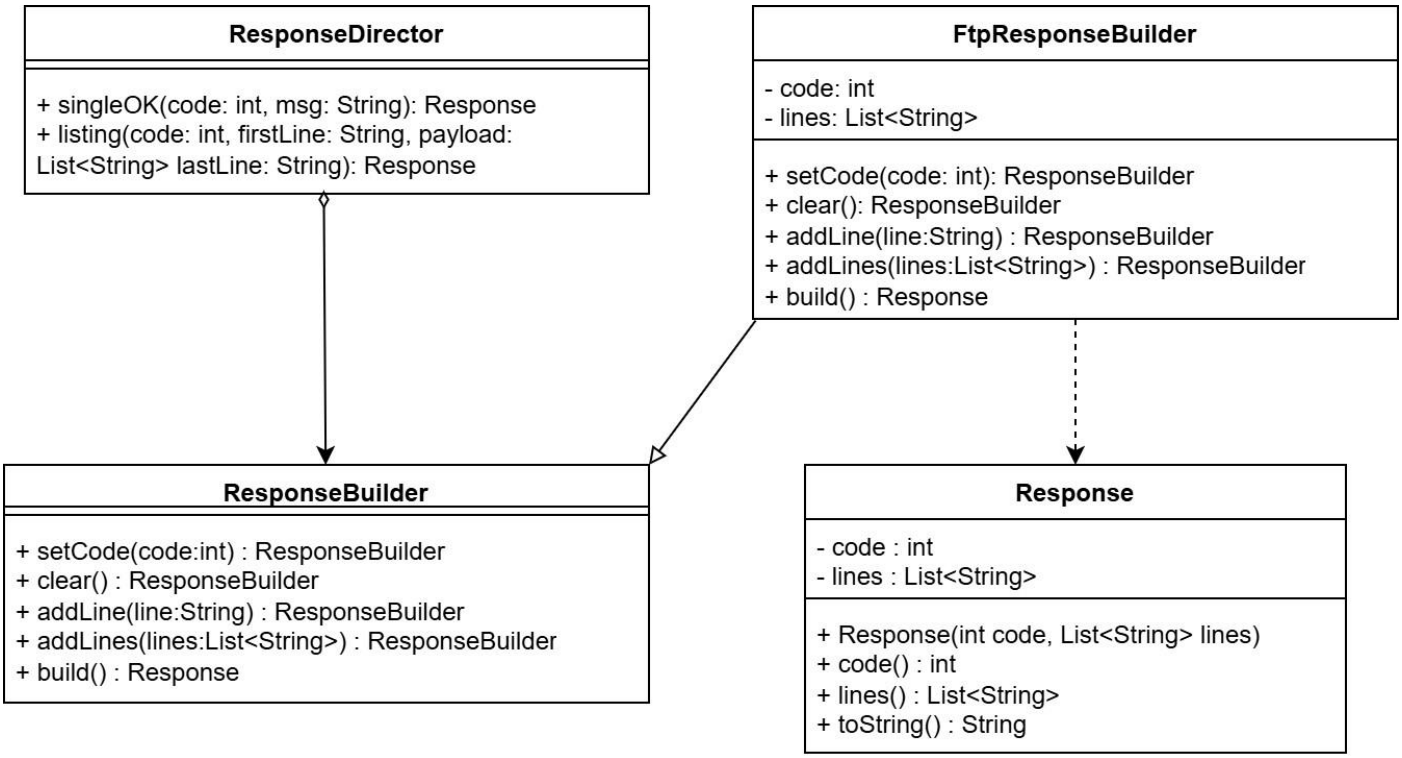


Рис 1. Патерн «Стан» у FTP-сервері.

Пояснення діаграми классів

Response – «продукт» із кодом відповіді та списком рядків; сам відповідає за форматування FTP-відповідей у toString().

ResponseBuilder – інтерфейс будівельника з fluent-методами для коду та рядків.

FtpResponseBuilder – конкретний будівельник, який накопичує стан (код + рядки) і створює Response у build().

ResponseDirector – утилітний «директор», що інкапсулює готові сценарії побудови: singleOK(…) для однотипних відповідей і listing(…) для багаторядкових (XYZ-… … XYZ …).

Вихідний код

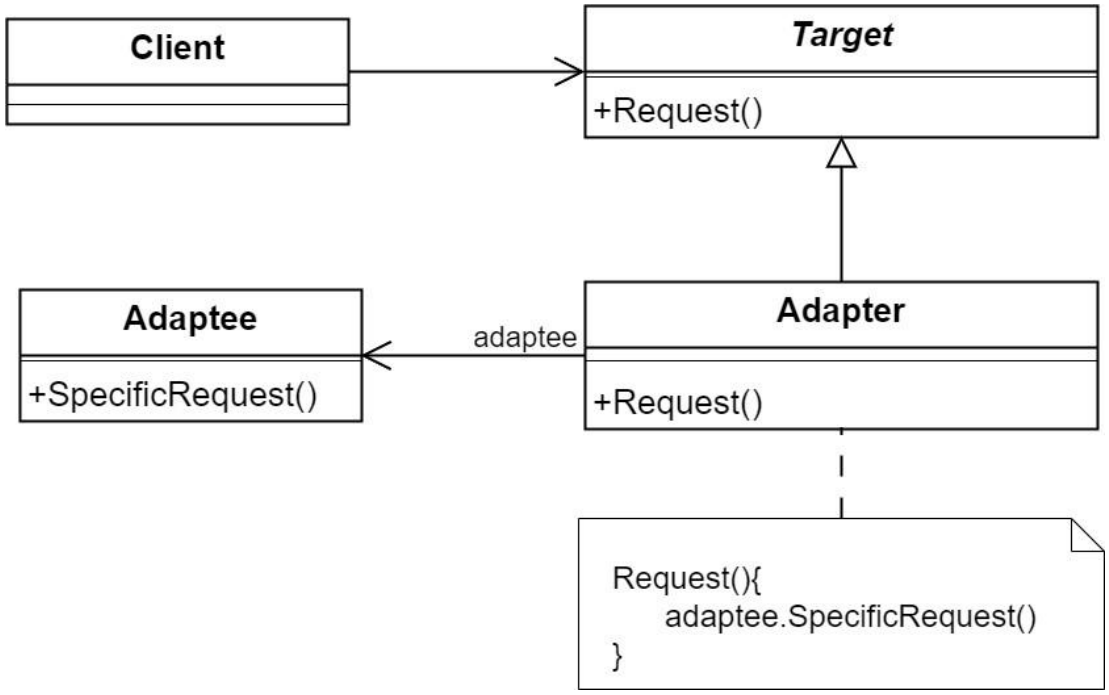
https://github.com/GlebTiKsTRPZ/lab5

Відповіді на питання до лабораторної роботи

1. Яке призначення шаблону «Адаптер»?

Дозволяє несумісним класам співпрацювати, перетворюючи інтерфейс наявного класу (Adaptee) до очікуваного клієнтом інтерфейсу (Target).

2. Нарисуйте структуру шаблону «Адаптер».



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

Зазвичай: Client, Target, Adapter, Adaptee. Client викликає методи Target; Adapter реалізує Target і делегує виклики до Adaptee.specificRequest() (за потреби – з перетворенням параметрів/результатку).

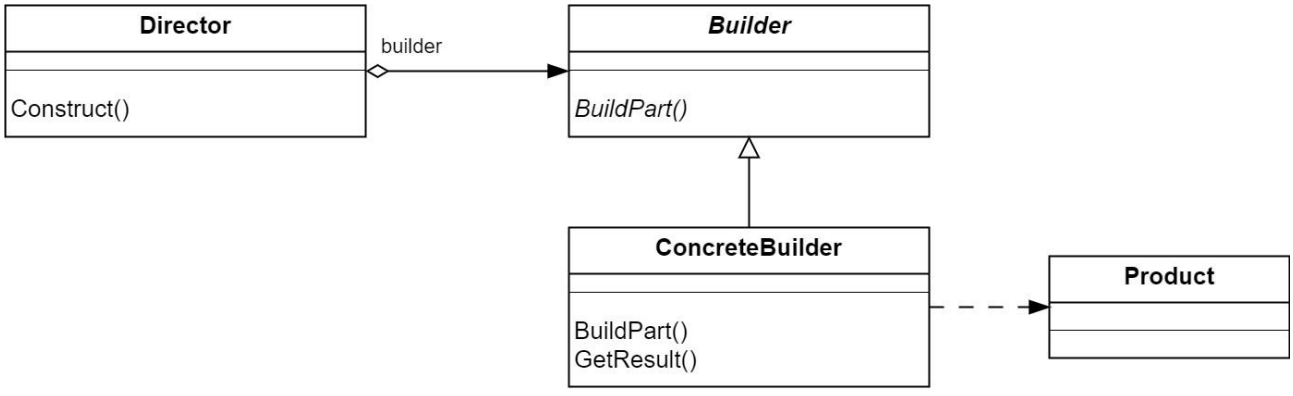
4. Яка різниця між реалізацією «Адаптера» на рівні об’єктів та на рівні класів?

Об’єктний адаптер використовує композицію (гнучкий, може адаптувати підкласи Adaptee). Класовий адаптер базується на множинному наслідуванні Target і Adaptee (жорсткіший зв’язок, але без додаткових полів).

5. Яке призначення шаблону «Будівельник»?

Відокремлює поетапне конструювання складного об’єкта від його представлення, даючи змогу тими самими кроками отримувати різні варіанти продукту та контролювати порядок збирання.

6. Нарисуйте структуру шаблону «Будівельник».



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

Director керує послідовністю кроків Builder; абстрактний Builder задає інтерфейс побудови; ConcreteBuilder реалізує кроки та накопичує стан; Product — результат, що повертається через getResult().

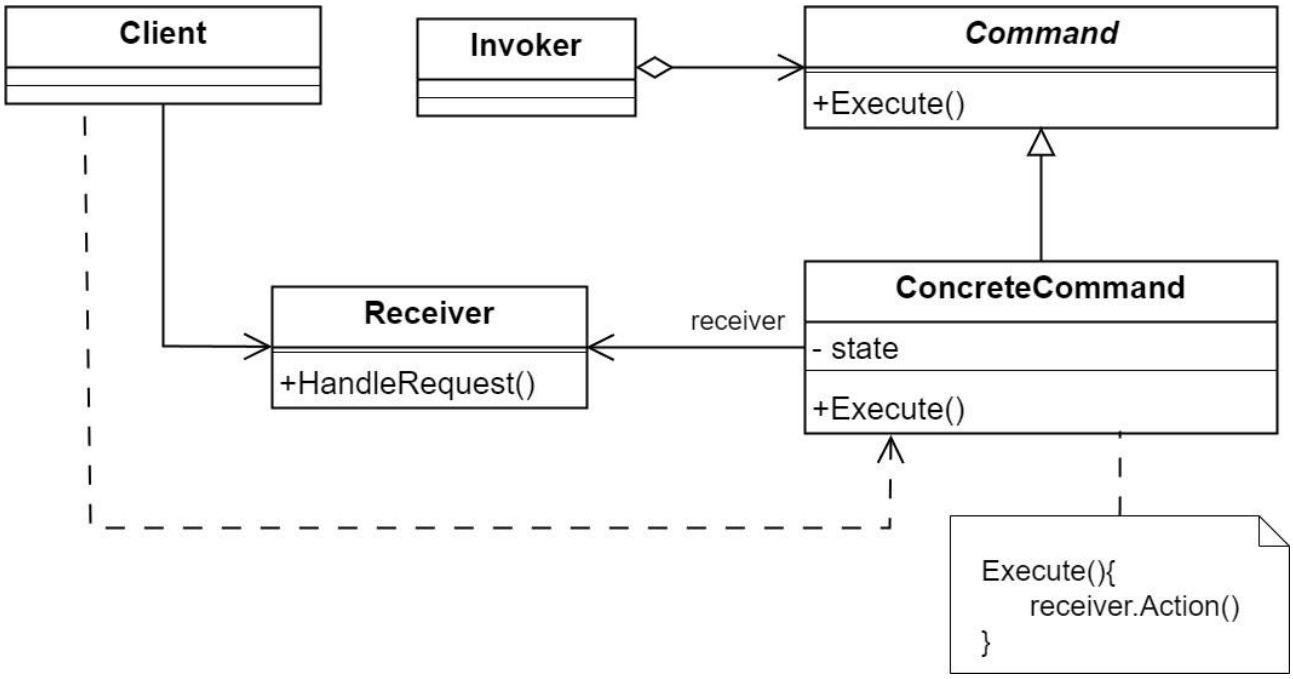
8. У яких випадках варто застосовувати шаблон «Будівельник»?

Коли об’єкт складний і створюється багатоетапно; потрібні різні подання/конфігурації; є багато необов’язкових параметрів; важливий контроль порядку побудови та інваріантів.

9. Яке призначення шаблону «Команда»?

Інкапсулювати запит у вигляді об’єкта для роз’єднання відправника й отримувача, підтримки undo/redo, черг, макрокоманд і логування.

10. Нарисуйте структуру шаблону «Команда».



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

Client створює ConcreteCommand і встановлює Receiver; Invoker зберігає/викликає execute(); ConcreteCommand делегує дію методам Receiver

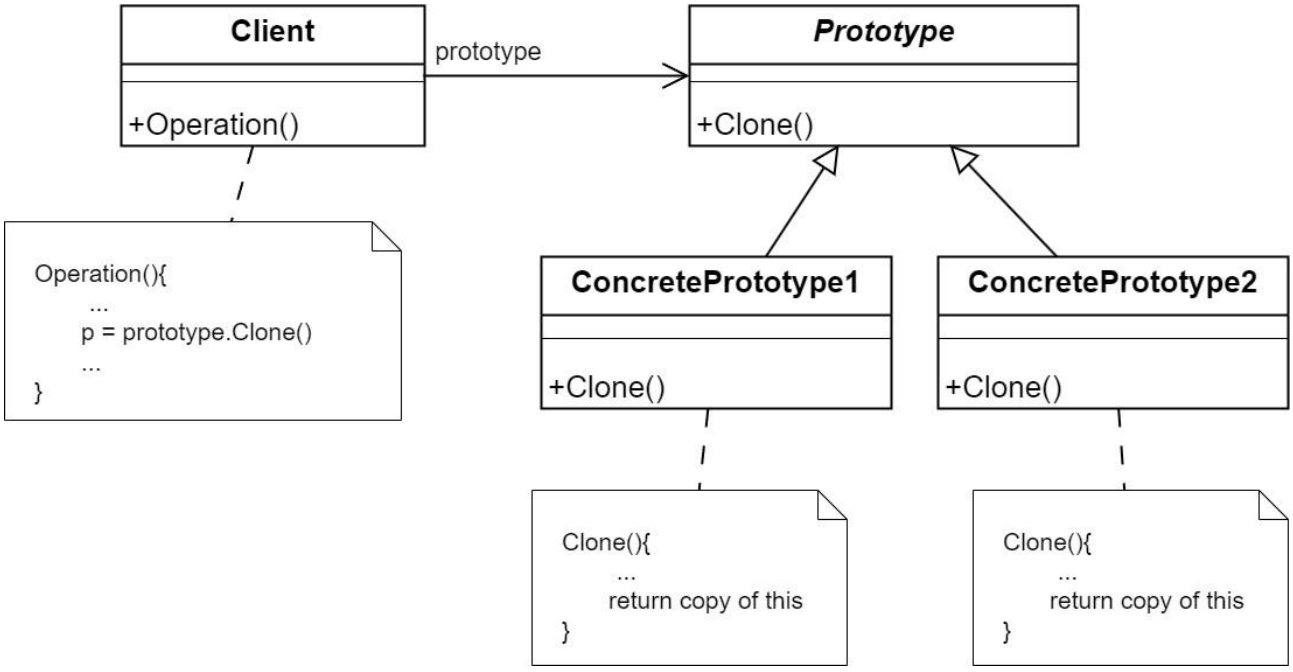
12. Розкажіть як працює шаблон «Команда».

Операції системи представляються об’єктами з єдиним інтерфейсом; Invoker викликає їх у потрібний момент; команди можна компонувати, ставити в чергу та відміняти, зберігаючи історію або стан.

13. Яке призначення шаблону «Прототип»?

Створювати нові екземпляри шляхом клонування наявних прототипів, уникаючи дорогих конструкторів та розростання ієрархій, із можливістю швидкого варіювання конфігурацій.

14. Нарисуйте структуру шаблону «Прототип».



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

Prototype визначає clone(); ConcretePrototype(и) реалізують поверхневе/глибоке копіювання; Client (часто через реєстр прототипів) обирає прототип і викликає clone(), а далі за потреби налаштовує екземпляр.

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

Обробка подій у складних UI (баблінг по ієрархії), middleware у веб-фреймворках, багатоетапна валідація запитів, ескалація звернень у підтримці, ланцюжок логерів/фільтрів.

Висновок

У цій лабораторній роботі впроваджено патерн «Будівельник» для формування FTP-відповідей: створено Response, ResponseBuilder/FtpResponseBuilder та ResponseDirector, що відокремлюють процес побудови від представлення. Це уніфікувало одно- й багаторядкові відповіді (зокрема PWD і LIST) та спростило підтримку й подальше розширення логіки сервера.

Такий підхід робить протокол взаємодії з клієнтами та інтеграціями стабільним і передбачуваним: усі відповіді мають єдиний формат, тому зміни текстів, структур або розширення набору команд менше впливають на зовнішні скрипти й партнерські системи, що знижує ризики збоїв у продакшені та витрати на підтримку.