

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 №6**  
із дисципліни «*Технології розробки програмного забезпечення*»  
Тема: «*Патерни проектування*»

**Виконав:**

Студент групи IA-34  
Ястремський Богдан

**Перевірив:**

асистент кафедри ІСТ  
Мягкий Михайло Юрійович

**Тема:** Патерни проектування.

**Мета:** Вивчити структуру шаблонів «Abstract Factory», «Factory Method», «Memento», «Observer», «Decorator» 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 «Абстрактна фабрика» використовується для створення сімейств об'єктів без вказівки їх конкретних класів. Для цього виноситься загальний інтерфейс фабрики (AbstractFactory) і створюються його реалізації для різних сімейств продуктів. Цей шаблон передусім структурує знання про схожі об'єкти (що називаються сімействами, як класи для доступу до БД) і створює можливість взаємозаміни різних сімейств.

Шаблон «Фабричний метод» визначає інтерфейс для створення об'єктів певного базового типу. Це зручно, коли хочеться додати можливість створення об'єктів не базового типу, а деякого дочірнього. Фабричний метод у такому разі є зачіпкою для впровадження власного конструктора об'єктів.

Шаблон «Memento» використовується для збереження і відновлення стану об'єктів без порушення інкапсуляції. Об'єкт «Memento» служить виключно для збереження змін над початковим об'єктом (Originator).

Шаблон «Observer» визначає залежність «один-до-багатьох» таким чином, що коли один об'єкт змінює власний стан, усі інші об'єкти отримують про це сповіщення і мають можливість змінити власний стан також.

Шаблон «Декоратор» призначений для динамічного додавання функціональних можливостей об'єкту під час роботи програми. Декоратор деяким чином «обертає» (за рахунок агрегації) початковий об'єкт зі збереженням його функцій, проте дозволяє додати додаткові дії. Такий шаблон надає гнучкіший спосіб зміни поведінки об'єкту чим просте спадкоємство, оскільки початкова функціональність зберігається в повному об'ємі. Більше того, таку поведінку можна застосовувати до окремих об'єктів, а не до усієї системи в цілому.

### Хід роботи:

- Повна діаграма класів (без шаблону).

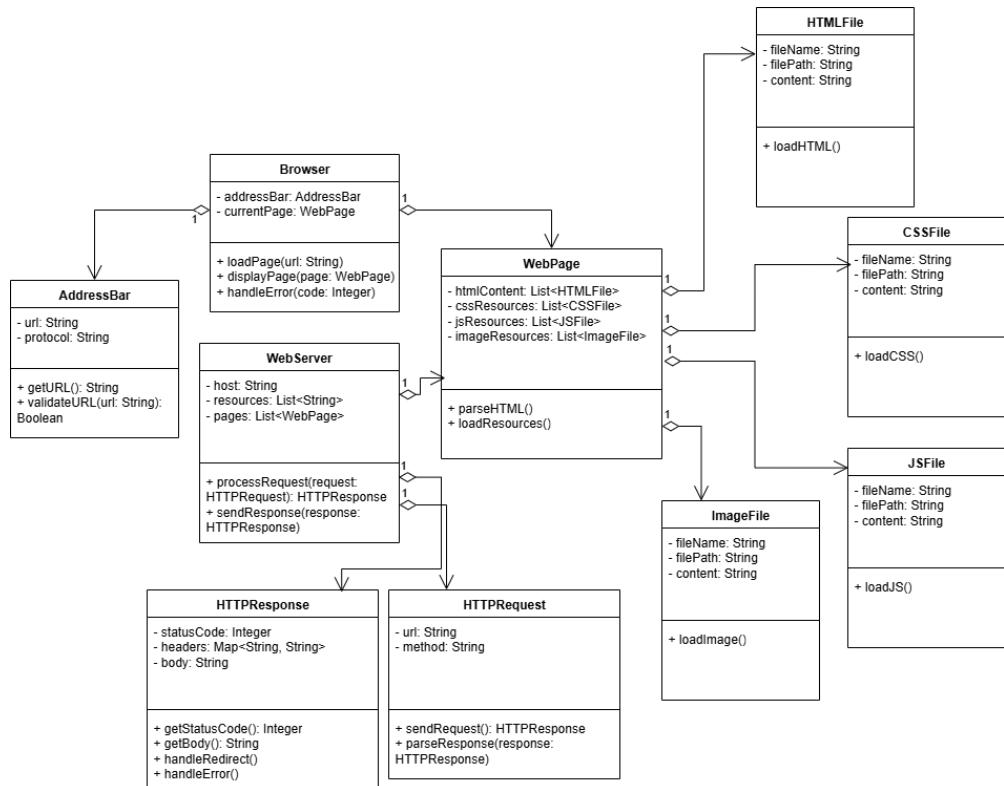


Рис. 6.1 – Діаграма класів системи

- Діаграма класів з використанням шаблону «Factory method»:

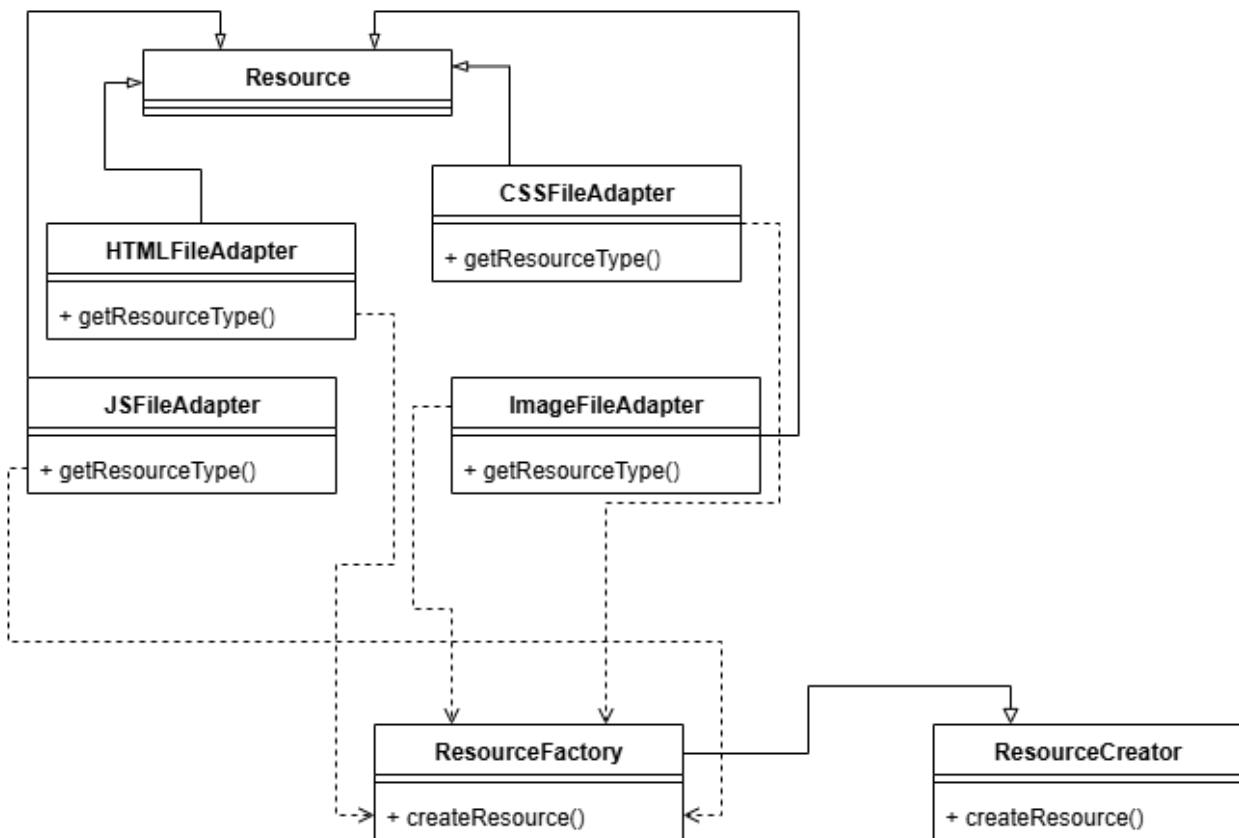


Рис. 6.2 – Діаграма класів системи з використанням шаблону «Factory Method»

Method»

Я визначаю абстрактний клас `ResourceCreator`, який містить в собі метод для створення ресурсів. Далі я визначаю творця-фабрику `ResourceFactory`, який вже буде створювати конкретні продукти-ресурси (які я назвав адаптерами), базуючись на розширеннях.

Далі я для кожного типу файлу створив власний продукт-ресурс, який містить в собі метод для повернення типу ресурсу для подальшої роботи з ним.

На вершині в нас інтерфейс ресурс, який визначає основні методи для роботи з файлами, які потім знадобляться для завантажування ресурсів.

У якості використання патерну, я оновив метод `WebPage.parseHTML()`, який тепер базується тільки на регулярних виразах по перевірці тегів, які віднаходять ті, що треба для обробки.

Таким чином, я додаю один із цих ресурсів до загальних ресурсів, не перевіряючи їх нічим, окрім регулярних виразів, сприошуєчи власне метод, який містив купу умов, а зараз потребує в рази менше коду.

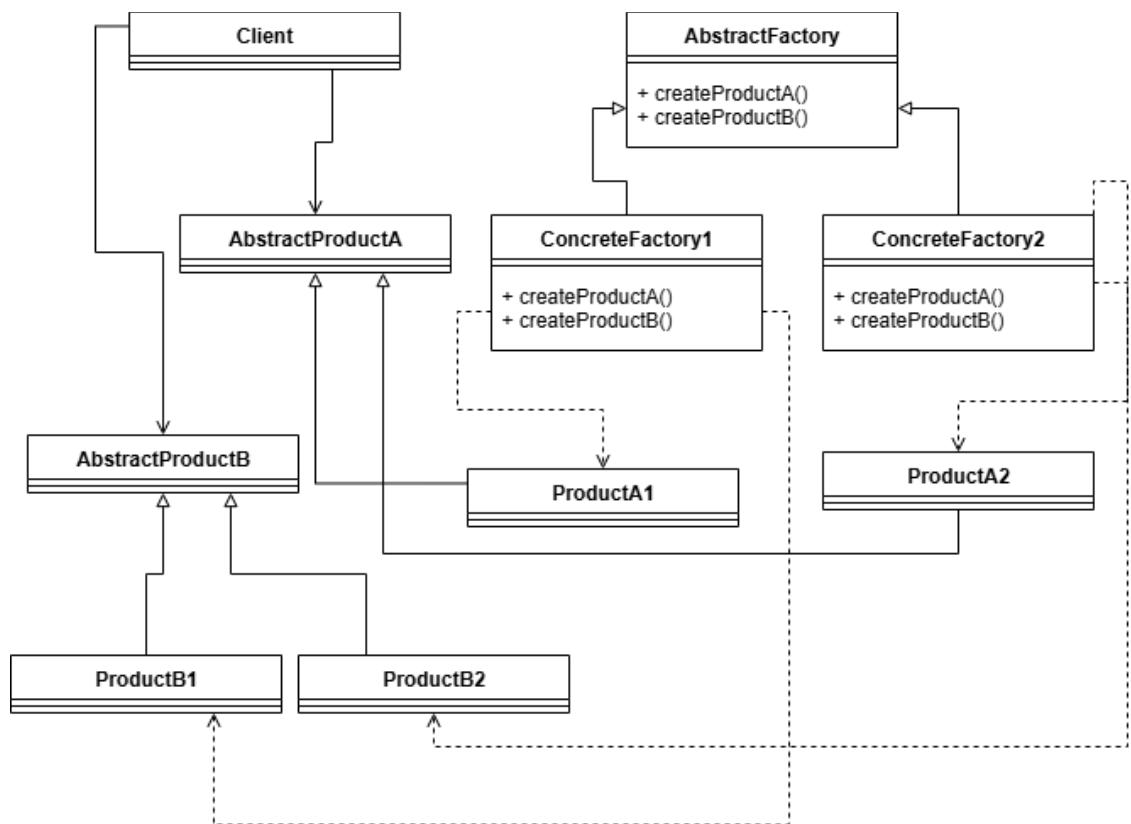
**Висновок:** на даному лабораторному занятті я продовжив знайомством з поняттям патернів проектування, які є їх види, та їхні плюси і мінуси. В якості використованого патерну я обрав «Factory Method», та імплементував його як створювач ресурсів для завантаження на сторінку. Я позбавив клас від прив'язки до конкретних класів, використовуючи регулярні вирази як парсер для кожного необхідного мені типу ресурсу. Паралельно з цим, я спростила підтримку коду, виділивши всі виробництва ресурсів в одне місце. Не дивлячись на те, що додати новий тип даних, типу відео, аудіо і т.д. дуже легко в цьому патерні – створивши відповідний ConcreteProduct – ієархія отримується доволі складна.

## Відповіді на контрольні питання:

1. Яке призначення шаблону «Абстрактна фабрика»?

Шаблон «Абстрактна фабрика» використовується для створення сімейств об'єктів без вказівки їх конкретних класів. Цей шаблон передусім структурує знання про схожі об'єкти (що називаються сімействами, як класи для доступу до БД) і створює можливість взаємозаміни різних сімейств.

2. Нарисуйте структуру шаблону «Абстрактна фабрика».



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

Основними класами, які входять до шаблону проєктування «Абстрактна фабрика» є:

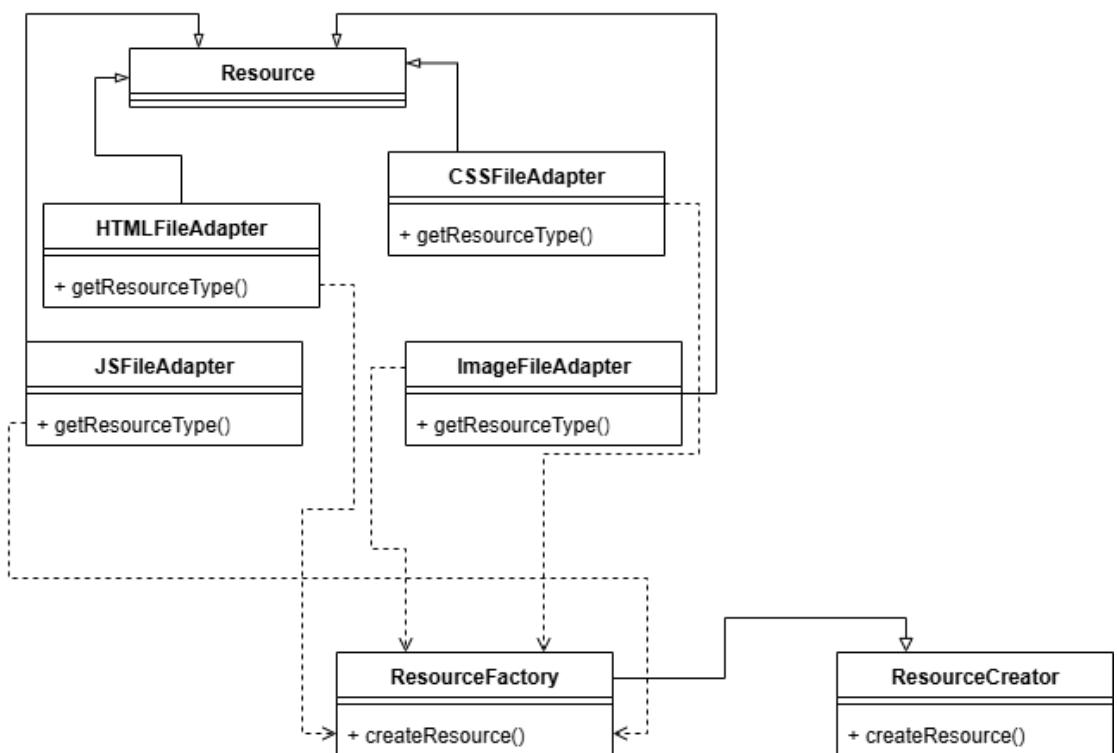
- Абстрактна фабрика (Abstract Factory) - Визначає інтерфейс для створення всіх абстрактних продуктів у певному сімействі.
- Конкретна фабрика (Concrete Factory) - Реалізує інтерфейс Абстрактної фабрики для створення конкретних продуктів певного сімейства.

- Абстрактний продукт (Abstract Product) - Визначає інтерфейс для конкретного типу об'єктів у сімействі
- Конкретний продукт (Concrete Product) - Конкретна реалізація Абстрактного продукту
- Клієнт (Client) - Код, який використовує фабрику та продукти.

4. Яке призначення шаблону «Фабричний метод»?

Шаблон «Фабричний метод» визначає інтерфейс для створення об'єктів певного базового типу. Це зручно, коли хочеться додати можливість створення об'єктів не базового типу, а деякого дочірнього.

5. Нарисуйте структуру шаблону «Фабричний метод».



6. Які класи входять в шаблон «Фабричний метод», та яка між ними взаємодія?

До шаблону проектування «Фабричний метод» входять такі класи:

- Продукт (Product) - Інтерфейс або абстрактний клас, який оголошує тип об'єктів, що виробляються фабричним методом. Усі конкретні продукти повинні реалізовувати цей інтерфейс.

- Конкретний продукт (Concrete Product) - конкретна реалізація інтерфейсу Продукту.
- Творець (Creator) - Абстрактний клас або інтерфейс, який оголошує фабричний метод — метод, що повертає об'єкт типу Продукт.
- Конкретний творець (Concrete Creator) - Підклас класу Творця, який перевизначає (реалізує) фабричний метод для створення та повернення конкретного екземпляра Конкретного продукту.

7. Чим відрізняється шаблон «Абстрактна фабрика» від «Фабричний метод»?

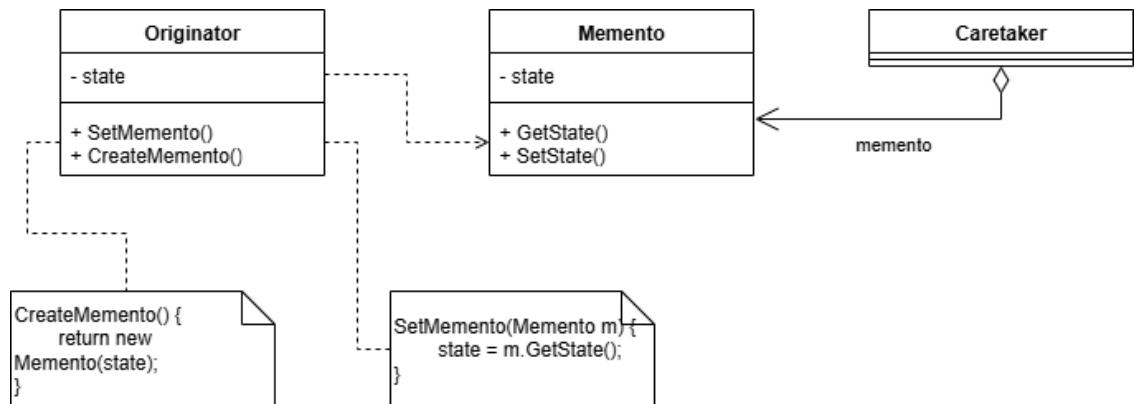
Перший створює цілі сімейства взаємопов'язаних продуктів, а другий - один конкретний продукт.

Перший має кілька методів створення (по одному на кожен тип продукту), другий - один фабричний метод.

8. Яке призначення шаблону «Знімок»?

Шаблон використовується для збереження і відновлення стану об'єктів без порушення інкапсуляції.

9. Нарисуйте структуру шаблону «Знімок».



10. Які класи входять в шаблон «Знімок», та яка між ними взаємодія?

До цього шаблону входять три основні класи:

- Творець (Originator) - Це об'єкт, стан якого потрібно зберігати та відновлювати. Він містить важливі дані (стан) і відповідає як

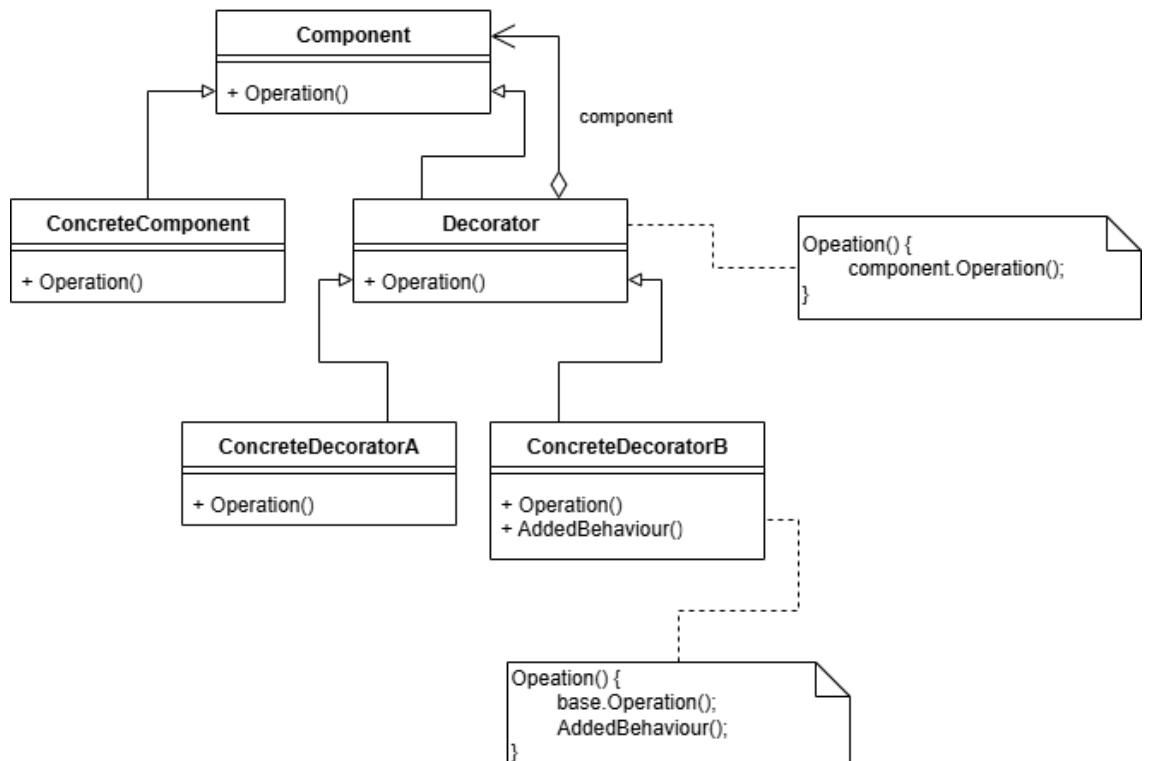
за створення знімка зі свого поточного стану, так і за відновлення свого стану з переданого йому знімка.

- Знімок (Memento) - Об'єкт-сховище, який зберігає внутрішній стан об'єкта Творця на певний момент часу. Це об'єкт із дуже обмеженим доступом.
- Опікун (Caretaker) - Відповідає за зберігання та управління об'єктами Знімка. Вирішує, коли потрібно зберегти стан (запитати знімок у Творця) і коли його відновити (передати знімок Творцю).

11. Яке призначення шаблону «Декоратор»?

Шаблон призначений для динамічного додавання функціональних можливостей об'єкту під час роботи програми. Декоратор деяким чином «обертає» (за рахунок агрегації) початковий об'єкт зі збереженням його функцій, проте дозволяє додати додаткові дії.

12. Нарисуйте структуру шаблону «Декоратор».



13. Які класи входять в шаблон «Декоратор», та яка між ними взаємодія?

До цього шаблону входять такі основні класи та інтерфейси:

- Компонент (Component) - Інтерфейс або абстрактний клас, який визначає загальну поведінку як для вихідних об'єктів, так і для декораторів.
- Конкретний компонент (Concrete Component) - Вихідний клас об'єкта, до якого додається нова поведінка.
- Декоратор (Decorator) - Абстрактний клас, який також реалізує інтерфейс Компонента. Він зберігає посилання на інший об'єкт Компонента.
- Конкретний декоратор (Concrete Decorator) - Конкретна реалізація Декоратора, яка додає специфічну нову функціональність.

14. Які є обмеження використання шаблону «декоратор»?

- Велика кількість крихітних класів.
- Важко конфігурувати об'єкти, які загорнуто в декілька обгорток одночасно.