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

Технології розроблення програмного забезпечення
Лабораторна робота №8
ШАБЛОНИ «COMPOSITE», «FLYWEIGHT»,
«INTERPRETER», «VISITOR»

Виконала
студентка групи ІА–24:
Кармазіна А. В.

Перевірив:
Мягкий М. Ю.

Київ 2025

Зміст

Завдання 3

Тема: 26 Download manager 3

Теоретичні відомості 3

Хід роботи 5

Реалізація шаблону Composite 5

Висновок 7

Завдання

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

Тема: 26 Download manager (iterator, command, observer, template method, composite, p2p)

Інструмент для скачування файлів з інтернету по протоколах http або https з можливістю продовження завантаження в зупиненому місці, розподілу швидкостей активним завантаженням, ведення статистики завантажень, інтеграції в основні браузери (firefox, opera, internet explorer, chrome) (https://github.com/Karmazinanastya/TRPZ_Karmazina_labs/tree/main/FileDownloader)

Теоретичні відомості

1. Composite (Компонування)

Шаблон Composite дозволяє обробляти групи об'єктів і одиничні об'єкти однаково. Він використовується для організації об'єктів в ієрархічну структуру, де окремі елементи та їх групи оброCompositebляються однаково. Це дозволяє створити дерево об'єктів, де кожен об'єкт може бути або одиничним елементом, або групою інших елементів.

Основні компоненти:

Component: загальний інтерфейс або абстрактний клас для всіх елементів дерева.

Leaf: одиничний елемент, який не має дочірніх елементів.

Composite: складений елемент, що має дочірні елементи.

2. Flyweight (Летючий вагон)

Шаблон Flyweight оптимізує використання пам'яті, дозволяючи зберігати спільні дані для великої кількості об'єктів, зменшуючи таким чином кількість об'єктів, що створюються. Він розділяє стан об'єкта на внутрішній (незмінний)

та зовнішній (змінний). Внутрішній стан є спільним для всіх об'єктів, а зовнішній — може змінюватися для кожного об'єкта.

Основні компоненти:

Flyweight: інтерфейс для спільних частин стану об'єкта.

ConcreteFlyweight: конкретна реалізація, яка зберігає спільний стан.

FlyweightFactory: відповідальна за управління створенням та доступом до Flyweight об'єктів.

3. Interpreter (Інтерпретатор)

Шаблон Interpreter дозволяє визначити граматику певної мови та створити інтерпретатор для її обробки. Він використовується для реалізації мови виразів, наприклад, для аналізу і виконання арифметичних або логічних виразів.

Основні компоненти:

AbstractExpression: інтерфейс або абстрактний клас для всіх елементів граматики.

TerminalExpression: конкретний вираз, що не містить інших виразів.

NonTerminalExpression: вираз, що складається з інших виразів.

Context: контекст, що містить дані, необхідні для інтерпретації.

4. Visitor (Візитер)

Шаблон Visitor дозволяє додавати нову поведінку до існуючих об'єктів без змін в їх класах. Це здійснюється за допомогою візитерів, які обходять структуру об'єктів і виконують операції над ними. Це корисно для додавання нових операцій без модифікації самих об'єктів.

Основні компоненти:

Visitor: інтерфейс для конкретних операцій.

ConcreteVisitor: конкретна реалізація операцій для кожного типу елементів.

Element: елементи, які приймають візитера і дозволяють йому виконувати операції.

ConcreteElement: конкретний елемент, який реалізує прийом візитера.

Хід роботи

Реалізація шаблону **Composite**

Шаблон Composite дозволяє організувати об'єкти в ієрархічну структуру, де одиничні елементи (листя) та групи об'єктів (композити) обробляються однаково. Це дозволяє обробляти дерева об'єктів, де кожен об'єкт може бути або одиничним елементом, або складеним елементом, що містить інші об'єкти.

Цей шаблон реалізовано через інтерфейс IDownloadComponent, який визначає методи ExecuteAsync та UndoAsync для виконання та скасування команд, а також класи DownloadLeaf і DownloadComposite, що реалізують цей інтерфейс.

```
5 references
public interface IDownloadComponent
{
    4 references
    Task ExecuteAsync();
    4 references
    Task UndoAsync();
}
```

Рис. 1 - Інтерфейс IDownloadComponent

Цей інтерфейс визначає два методи:

ExecuteAsync: метод для виконання операції.

UndoAsync: метод для скасування операції.

Цей інтерфейс використовується як базовий для всіх елементів дерева об'єктів.

```
3 references
public class DownloadLeaf : IDownloadComponent
{
    private readonly DownloadListBoxItem _item;

    2 references
    public DownloadLeaf(DownloadListBoxItem item)
    {
        _item = item;
    }

    2 references
    public async Task ExecuteAsync()
    {
        await _item.DownloadAsync();
    }

    2 references
    public Task UndoAsync()
    {
        _item.TokenSource?.Cancel();
        return Task.CompletedTask;
    }
}
```

Рис. 2 - Клас DownloadLeaf

DownloadLeaf представляє собою листя в ієрархії — одиничний елемент, який не має дочірніх елементів. Цей клас реалізує методи ExecuteAsync та UndoAsync для конкретного завдання — завантаження файлу або скасування цього завдання.

```
public class DownloadComposite : IDownloadComponent
{
    private readonly List<IDownloadComponent> _children = new();

    1 reference
    public void Add(IDownloadComponent component)
    {
        _children.Add(component);
    }

    1 reference
    public void Remove(IDownloadComponent component)
    {
        _children.Remove(component);
    }

    3 references
    public async Task ExecuteAsync()
    {
        foreach (var child in _children)
        {
            await child.ExecuteAsync();
        }
    }

    3 references
    public async Task UndoAsync()
    {
        foreach (var child in _children)
        {
            await child.UndoAsync();
        }
    }
}
```

Рис. 3 - Клас DownloadComposite

DownloadComposite є складеним елементом, який містить кілька об'єктів IDownloadComponent (як листя, так і інші композиції). Цей клас дозволяє додавати і видаляти компоненти, а також виконувати або скасовувати операції для всіх дочірніх елементів.

DownloadLeaf: Кожен об'єкт DownloadLeaf представляє завантаження одного файлу. Це одиничний елемент, який виконує операцію завантаження через DownloadAsync та може бути скасований через UndoAsync.

DownloadComposite: Якщо потрібно виконати завантаження кількох файлів одночасно або організувати операції в групи, створюється об'єкт DownloadComposite, який може містити кілька DownloadLeaf або навіть інші DownloadComposite об'єкти. Виконання або скасування операцій на композиції викликає ці методи для кожного дочірнього елемента.

CommandInvoker: Команди, які виконуються для кожного елемента, можуть бути оброблені за допомогою CommandInvoker. Це дозволяє виконувати

операції над кожним елементом або групою елементів у вигляді команд і забезпечує зручний механізм скасування дій.

Висновок

Шаблон Composite ефективно вирішує проблему управління складними ієрархічними структурами об'єктів, дозволяючи однаково обробляти як окремі елементи, так і групи елементів. Цей шаблон реалізований через інтерфейс IDownloadComponent, що дозволяє працювати з завантаженнями файлів як з окремими одиничними елементами (листями), так і з їх групами (композитами). Завдяки цьому, отримуємо гнучкість у виконанні та скасуванні операцій на різних рівнях ієрархії без необхідності змінювати код для окремих випадків.