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

З дисципліни: Технології розроблення програмного забезпечення

Тема: “Паттерн Abstract Factory”

Виконав:

студент групи ІА-14

Рисаков Богдан

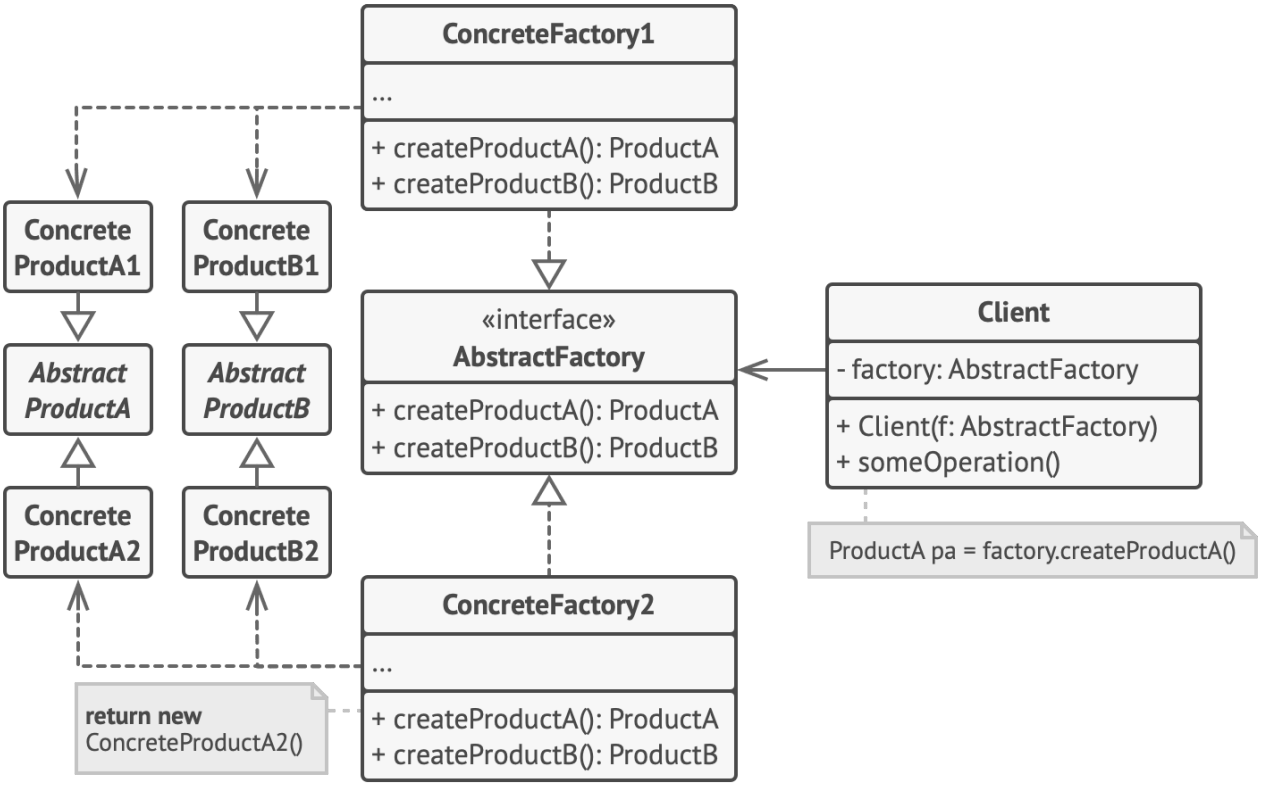
Перевірив:

Мягкий Михайло Юрійович

Київ, 2023

Мета: Реалізувати паттерн Abstract Factory у проекті на тему System Activity Monitor

Абстрактна фабрика — це породжувальний патерн проектування, що дає змогу створювати сімейства пов'язаних об'єктів, не прив'язуючись до конкретних класів створюваних об'єктів.



Приклад структури реалізації та використання патерну. Reference

Переваги:

- Гарантує поєднання створюваних продуктів.
- Звільняє клієнтський код від прив'язки до конкретних класів продукту.
- Виділяє код виробництва продуктів в одне місце, спрощуючи підтримку коду.
- Спрощує додавання нових продуктів до програми.
- Реалізує *принцип відкритості/закритості*.

Недоліки:

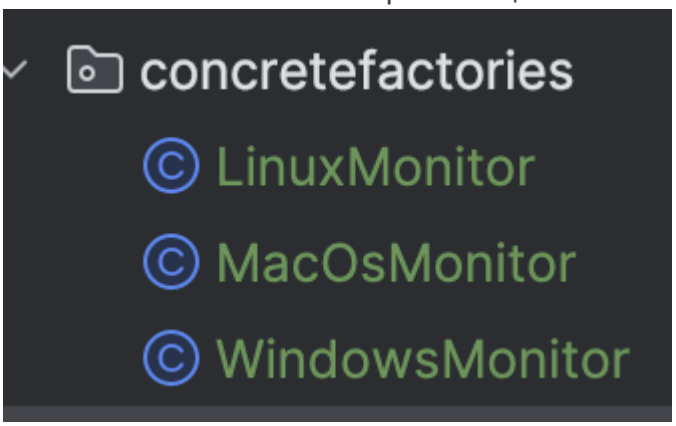
- Ускладнює код програми внаслідок введення великої кількості додаткових класів.
- Вимагає наявності всіх типів продукту в кожній варіації.

Реалізація:
Проблема: На різних платформах не можна використовувати однакові засоби для моніторингу.

1) Зробимо класс Абстрактної фабрики

```
public abstract class AbstractMonitor {

   public String osType;

   public AbstractMonitor(String osType) {
       this.osType = osType;
   }

   abstract public Monitoring setCpuLoadMonitor();

   abstract public Monitoring setKeyLoggerMonitor();

   abstract public Monitoring setMemoryMonitor();

   abstract public Monitoring setMouseTrackerMonitor();

   abstract public Monitoring setWindowsMonitor();
}
```

таким чином ми можемо зробити це:



Під кожну ОС буде використана конкретна фабрика яка має усі сервиси для моніторингу за працею системи

Я буду реалізовувати тільки для windows.

```java
@Getter
@Setter
public class WindowsMonitor extends AbstractMonitor {

    private String osType = "Windows";

    public WindowsMonitor(String osType) {
        super(osType);
    }

    @Override
    public Monitoring setCpuLoadMonitor() {
        return new CpuLoadMonitoringService();
    }

    @Override
    public Monitoring setKeyLoggerMonitor() {
        return new KeyLoggerMonitoringService();
    }

    @Override
    public Monitoring setMemoryMonitor() {
        return new MemoryMonitoringService();
    }

    @Override
    public Monitoring setMouseTrackerMonitor() {
        return new MouseTrackerMonitoringService();
    }

    @Override
    public Monitoring setWindowsMonitor() {
        return new WindowsMonitoringService();
    }
}
```

Таким чином можна буде моніторити різні платформи не змінюючи глобальну поведінку програми - для додавання нової платформи треба буде лише реалізувати 5 сервісів для моніторингу для нової ОС

Висновок: Я реалізував паттерн Абстрактна фабрика