Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Лабораторна робота № 8

З дисципліни «Технології розроблення програмного забезпечення»

Тема: «Шаблони «COMPOSITE»,
«FLYWEIGHT», «INTERPRETER»,
«VISITOR»»

Виконала: Лапа Руслана Ігорівна

студент групи ІА-11

Дата здачі

Захищено з балом

Перевірив:

ст. вик. кафедри ІСТ

Колеснік В. М.

Київ 2023

Тема: Шаблони «COMPOSITE», «FLYWEIGHT», «INTERPRETER», «VISITOR».

Мета: реалізувати один з розглянутих шаблонів.

Хід роботи:

..8 Powershell terminal (strategy, command, factory method, template method, interpreter, client-server)

Термінал для powershell повинен нагадувати типовий термінал з можливістю налаштування кольорів синтаксичних конструкцій, розміру вікна, фону вікна, а також виконання команд powershell і виконуваних файлів, а також працювати в декількох вікнах терміналу (у вкладках або одночасно шляхом розділення вікна).

1. Реалізувати не менше 3-х класів відповідно до обраної теми

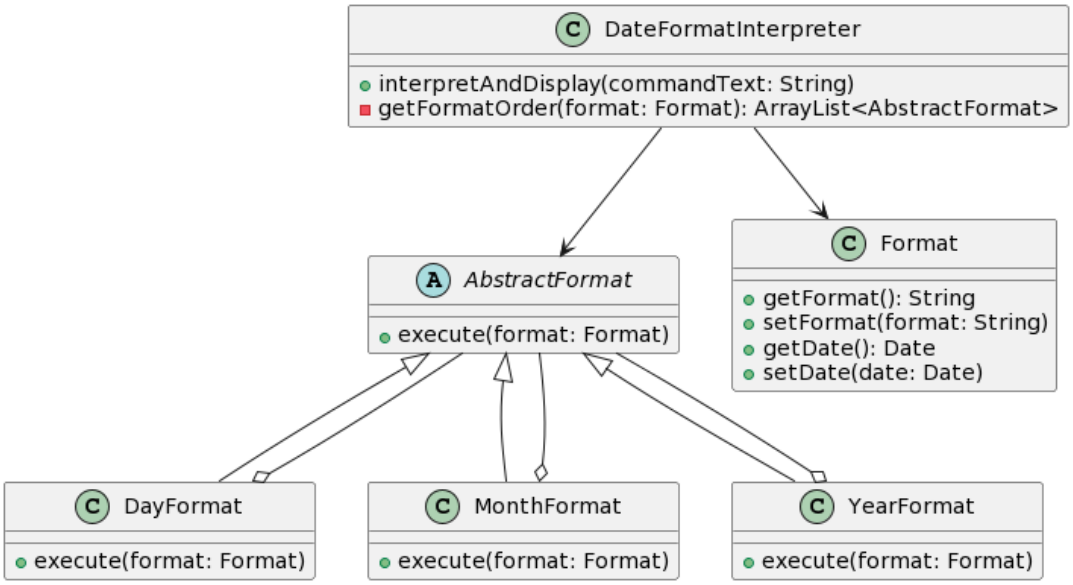

Рис. 1 – Діаграма класів для нижче описаного коду

2. Реалізація шаблону «Interpreter»

```java
package com.example.terminal.Model.Execution;

3 inheritors
public abstract class AbstractFormat {
    3 implementations
    public abstract void execute(Format format);
}
```

```java
package com.example.terminal.Model.Execution;

import java.util.Date;


public class DayFormat extends AbstractFormat {

    @Override
    public void execute(Format format) {
        String format1 = format.getFormat();
        Date date = format.getDate();
        int day = date.getDate();
        String tempFormat = format1.replaceAll( regex: "DD", Integer.toString(day));
        format.setFormat(tempFormat);
    }
}
```

```java
package com.example.terminal.Model.Execution;

import java.util.Date;


public class MonthFormat extends AbstractFormat {

    @Override
    public void execute(Format format) {
        String format1 = format.getFormat();
        Date date = format.getDate();
        int month = date.getMonth()+1;
        String tempFormat = format1.replaceAll( regex: "MM", Integer.toString(month));
        format.setFormat(tempFormat);
    }
}
```

```java
package com.example.terminal.Model.Execution;

import java.util.Date;


public class YearFormat extends AbstractFormat {

    @Override
    public void execute(Format format) {
        String format1 = format.getFormat();
        Date date = format.getDate();
        int year = date.getYear() + 1900;
        String tempFormat = format1.replaceAll( regex: "YYYY", Integer.toString(year));
        format.setFormat(tempFormat);
    }
}
```

```java
package com.example.terminal.Model.Execution;

import java.util.Date;

public class Format {

    public String format;
    2 usages
    public Date date;

    5 usages
    public String getFormat() { return format; }
    4 usages
    public void setFormat(String format) { this.format = format; }
    3 usages
    public Date getDate() { return date; }
    1 usage
    public void setDate(Date date) { this.date = date; }
}
```

```java
package com.example.terminal.Model.Execution;

import ...


public class DateFormatInterpreter {

    2 usages
    private final TextFlow textFlow;

    1 usage
    public DateFormatInterpreter(TextFlow textFlow) {
        this.textFlow = textFlow;
    }

    1 usage
    public void interpretAndDisplay(String commandText) {
        Format format = new Format();
        format.setFormat(commandText);
        format.setDate(new Date());

        ArrayList<AbstractFormat> formatOrderList = getFormatOrder(format);

        for (AbstractFormat abstractFormat : formatOrderList) {
            abstractFormat.execute(format);
        }

        textFlow.getChildren().addAll(new Text( s: format.getFormat() + "\n"));
    }
```

```java
    1 usage
    private ArrayList<AbstractFormat> getFormatOrder(Format format) {
        ArrayList<AbstractFormat> formatOrderList = new ArrayList<>();
        String[] strArray = format.getFormat().split( regex: "-");
        for (String string : strArray) {
            if (string.equalsIgnoreCase( anotherString: "MM")) {
                formatOrderList.add(new MonthFormat());
            } else if (string.equalsIgnoreCase( anotherString: "DD")) {
                formatOrderList.add(new DayFormat());
            } else {
                formatOrderList.add(new YearFormat());
            }
        }
        return formatOrderList;
    }
}
```

Висновок: у цій лабораторній роботі я ознайомилась з різними шаблонами проєктування і реалізувала шаблон “Interpreter ” .