**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Національний Технічний Університет України**

**«Київський Політехнічний Інститут»**

*Факультет інформатики та обчислювальної техніки*

*Кафедра обчислювальної техніки*

**\**

**Лабораторна робота №4**

*з дисципліни «Інженерія програмного забезпечення»*

*на тему: «Структурні шаблони проектування. Шаблони Flyweight, Adapter, Bridge, Facade»*

**Виконав:**

студент 2-го курсу ФІОТ

групи ІО-44

*Барабаш Т.А.*

**Номер залікової книжки:** 4403

**Варіант:** 3

**Перевірив:**

Старший викладач

к.т.н, с.н.с.

*Антонюк А.І.*

**Київ – 2015**

**ЛАБОРАТОРНА РОБОТА №4**

*Структурні шаблони проектування. Шаблони Flyweight, Adapter, Bridge, Facade.*

**Мета:** Вивчення структурних шаблонів. Отримання базових навичок з застосування шаблонів Flyweight, Adapter, Bridge, Facade.

**I. Завдання**

1. Закріпити призначення шаблонів проектування ПЗ, їх класифікацію. Знати назву і коротку характеристику кожного з шаблонів, що відносяться до певного класу.
2. Повторити структурні шаблони проектування ПЗ. Знати загальну характеристику структурних шаблонів та призначення кожного з них.
3. Детально вивчити структурні шаблони проектування Flyweight, Adapter, Bridge, Facade. Для кожного з них:

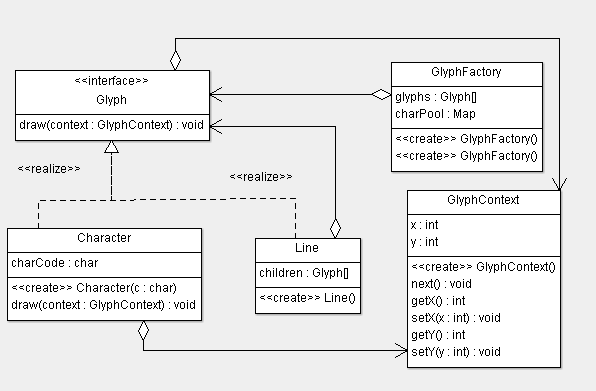
* вивчити Шаблон, його призначення, альтернативні назви, мотивацію, випадки, коли його застосування є доцільним, та результати такого застосування;
* знати особливості реалізації Шаблону, споріднені шаблони, відомі випадки його застосування в програмних додатках;
* вільно володіти структурою Шаблону, призначенням його класів та відносинами між ними;
* вміти розпізнавати Шаблон в UML діаграмі класів та будувати сирцеві коди Java-класів, що реалізують шаблон.

1. В підготованому проекті (ЛР1) створити програмний пакет com.lab111.labwork4. В пакеті розробити інтерфейси і класи, що реалізують завдання (згідно варіанту) з застосуванням одного чи декількох шаблонів (п.3). У класах, що розробляються, повністю реалізувати методи, пов'язані з функціюванням Шаблону. Методи, що реалізують бізнес-логіку, закрити заглушками з виводом на консоль інформації про викликаний метод та його аргументи.
2. За допомогою автоматизованих засобів виконати повне документування розроблених класів (також методів і полів), при цьому документація має в достатній мірі висвітлювати роль певного класу в загальній структурі Шаблону та особливості конкретної реалізації.
3. **Визначення варіанту завдання:**

**Номер залікової книжки:** 4403;

**Варіант:** 4403 *mod* 11 = **3** – *Визначити специфікації класів та реалізацію методів для об'єктів- гліфів латинського алфавіту та об'єктів-строк. Об'єкти-гліфи мають атрибути координат та використовуються в якості складових при побудові композитних об'єктів-строк. Забезпечити ефективне використання пам'яті при роботі з великою кількістю об'єктів-гліфів. Реалізувати метод виводу рядка.*

**ІI. Діаграма класів**



**IIІ. Код програми**

/\*\*

\*

\*/

**package** com.lab111.labwork4;

/\*\*

\* A glyph interface. Sets the mandatory draw-method that has to be implemented

\* by objects of the classes which use this interface.

\*

\* **@author** Taras Barabash

\* **@version** 1.0

\*

\*/

**public** **interface** Glyph {

/\*\*

\* Method draws a glyph in a given context.

\*

\* **@param** context

\*/

**public** **void** draw(GlyphContext context);

}

/\*\*

\*

\*/

**package** com.lab111.labwork4;

/\*\*

\* The class represents the context or the external state of the divisive glyph.

\* At the time it has to show the coordinates of divisive glyphs in a line.

\*

\* **@author** Taras Barabash

\* **@version** 1.0

\*

\*/

**public** **class** GlyphContext {

/\*\*

\* Coordinates

\*/

**private** **int** x;

**private** **int** y;

/\*\*

\* Constructor by default, sets the coordinates at the start-point of (0;0).

\*/

**public** GlyphContext() {

**this**.x = 0;

**this**.y = 0;

}

/\*\*

\* Increments the number of symbols in line.

\*/

**public** **void** next() {

**if** (x == 5) {

x = 0;

y++;

}

x++;

}

/\*\*

\* Getter for x-coordinate

\*

\* **@return** x coordinate

\*/

**public** **int** getX() {

**return** **this**.x;

}

/\*\*

\* Setter for x-coordinate

\*

\* **@param** x

\* coordinate

\*/

**public** **void** setX(**int** x) {

**this**.x = x;

}

/\*\*

\* Getter for y-coordinate.

\*

\* **@return** y coordinate

\*/

**public** **int** getY() {

**return** **this**.y;

}

/\*\*

\* Setter for y-coordinate

\*

\* **@param** y

\* coordinate

\*/

**public** **void** setY(**int** y) {

**this**.y = y;

}

}

/\*\*

\*

\*/

**package** com.lab111.labwork4;

**import** java.util.\*;

/\*\*

\* The factory class. It's responsible for creation of different glyph-objects.

\* Though it can also create lines, it's not a mandatory. The opportunity to do

\* that is left cause of further ideas of refactoring.

\*

\* **@author** Taras Barabash

\* **@version** 1.0

\*/

**public** **class** GlyphFactory {

**private** Glyph[] glyphs;

**private** Map charPool;

/\*\*

\* A basic constructor without parameters, initializes the HashMap.

\*/

**public** GlyphFactory() {

charPool = **new** HashMap<Integer, Character>();

}

/\*\*

\* Constructor with glyph-parameters.

\*

\* **@param** glyphs

\*/

**public** GlyphFactory(Glyph... glyphs) {

**this**.glyphs = **new** Glyph[glyphs.length];

**for** (**int** i = 0; i < glyphs.length; i++)

**this**.glyphs[i] = glyphs[i];

}

**public** Character getCharacter(**char** c) {

Character thisChar = (Character) charPool.get(c);

**if** (thisChar == **null**) {

thisChar = **new** Character(c);

charPool.put(c, thisChar);

}

**return** thisChar;

}

/\*\*

\* Method returns line. Non-mandatory, created cause of thoughts of further

\* refactoring.

\*

\* **@param** glyphs

\* **@return**

\*/

**public** Line getLine(Glyph... glyphs) {

**return** **new** Line(glyphs);

}

}

/\*\*

\*

\*/

**package** com.lab111.labwork4;

/\*\*

\* The class of the line (or string, semantically). Represents an non-divisive

\* glyph. It's a composite class so here implementation of specific "add",

\* "remove" or "get"-methods can be found. As children classes it collects other

\* glyphs like lines or, mostly, Characters.

\*

\* **@author** Taras Barabash

\*

\*/

**public** **class** Line **implements** Glyph {

**private** Glyph[] children;

/\*\*

\* Constructor, sets the "children" of this composite onject.

\*

\* **@param** glyphs

\*/

**public** Line(Glyph... glyphs) {

children = **new** Glyph[glyphs.length];

**for** (**int** i = 0; i < glyphs.length; i++)

children[i] = glyphs[i];

}

/\*\*

\* Method allows to add new glyph

\*

\* **@param** glyph

\*/

**public** **void** add(Glyph glyph) {

Glyph[] buf = **new** Glyph[children.length + 1];

**for** (**int** i = 0; i < buf.length - 1; i++) {

buf[i] = children[i];

}

buf[children.length] = glyph;

children = **new** Glyph[buf.length];

children = buf;

}

/\*\*

\* Method allows to remove a child from the array.

\*

\* **@param** ind

\* index of a child to remove

\*/

**public** **void** remove(**int** ind) {

**if** (children == **null**) {

**throw** **new** IndexOutOfBoundsException();

}

Glyph[] buffer = **new** Glyph[children.length - 1];

**if** (ind == 0) {

**for** (**int** i = 1; i < children.length; i++) {

buffer[i - 1] = children[i];

}

children = **new** Glyph[buffer.length];

children = buffer;

} **else** **if** (ind == children.length - 1) {

**for** (**int** i = 0; i < children.length - 1; i++) {

buffer[i] = children[i];

}

children = **new** Glyph[buffer.length];

children = buffer;

} **else** {

**for** (**int** i = 0; i < ind; i++) {

buffer[i] = children[i];

}

**for** (**int** i = ind; i < children.length - 1; i++) {

buffer[i] = children[i + 1];

}

children = **new** Glyph[buffer.length];

children = buffer;

}

}

/\*\*

\* Method allows to get a child on the given index.

\*

\* **@param** ind

\* index of child to be returned.

\* **@return** Glyph

\*/

**public** Glyph getChild(**int** ind) {

**if** (ind < children.length)

**return** children[ind];

**return** **null**;

}

/\*\*

\* An implementation of method that's required by the interface. In this

\* implementation the draw-methods of children-glyphs are called.

\*/

**public** **void** draw(GlyphContext context) {

**for** (Glyph g : children) {

g.draw(context);

context.next();

}

}

}

/\*\*

\*

\*/

**package** com.lab111.labwork4;

/\*\*

\* The class represents a divisive implementation of Glyph. Thus it includes

\* only an internal state. An external state is passed to the draw-method via

\* the context. A Character is never created by simple instancing. Can only be

\* created with the help of primitive fabric.

\*

\* **@author** Taras Barabash

\* **@version** 1.0

\*/

**public** **class** Character **implements** Glyph {

**private** **char** charCode;

/\*\*

\* An ordinary constructor which adds an internal state.

\*

\* **@param** c

\* char, an internal state.

\*/

**public** Character(**char** c) {

charCode = c;

}

/\*\*

\* The method draws this glyph in a specific context that is transfered with

\* arguments.

\*/

**public** **void** draw(GlyphContext context) {

System.***out***.println(charCode + "(" + context.getX() + ";"

+ context.getY() + ")");

};

}

/\*\*

\*

\*/

**package** com.lab111.labwork4;

/\*\*

\* **@author** Taras Barabash

\*

\*/

**public** **class** Lab4 {

**public** **static** **void** main(String[] args) {

GlyphFactory factory = **new** GlyphFactory();

Glyph[] glyphs = { factory.getCharacter('i'),

factory.getCharacter('t'), factory.getCharacter('w'),

factory.getCharacter('o'), factory.getCharacter('r'),

factory.getCharacter('k'), factory.getCharacter('s'),

factory.getCharacter('s'), factory.getCharacter('k'),

factory.getCharacter('r'), factory.getCharacter('o'),

factory.getCharacter('v'), factory.getCharacter('t'),

factory.getCharacter('i') };

Line line = factory.getLine(glyphs);

GlyphContext context = **new** GlyphContext();

line.draw(context);

}

}

**ІV. Висновок**

Ознайомились з видами шаблонів проектування програмного забезпечення. Розроблена відповідна тестова програма. Результати успішної роботи тестової програми наведені нижче підтверджують правильність обраних рішень.

i(0;0)

t(1;0)

w(2;0)

o(3;0)

r(4;0)

k(5;0)

s(1;1)

**A**

[**add(Glyph)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Line.html#add-com.lab111.labwork4.Glyph-) - Method in class com.lab111.labwork4.[Line](file:///C:\Workspace\template\doc\com\lab111\labwork4\Line.html)

Method allows to add new glyph

**C**

[**Character**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Character.html) - Class in [com.lab111.labwork4](file:///C:\Workspace\template\doc\com\lab111\labwork4\package-summary.html)

The class represents a divisive implementation of Glyph.

[**Character(char)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Character.html#Character-char-) - Constructor for class com.lab111.labwork4.[Character](file:///C:\Workspace\template\doc\com\lab111\labwork4\Character.html)

An ordinary constructor which adds an internal state.

[com.lab111.labwork4](file:///C:\Workspace\template\doc\com\lab111\labwork4\package-summary.html) - package com.lab111.labwork4

**D**

[**draw(GlyphContext)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Character.html#draw-com.lab111.labwork4.GlyphContext-) - Method in class com.lab111.labwork4.[Character](file:///C:\Workspace\template\doc\com\lab111\labwork4\Character.html)

The method draws this glyph in a specific context that is transfered with arguments.

[**draw(GlyphContext)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Glyph.html#draw-com.lab111.labwork4.GlyphContext-) - Method in interface com.lab111.labwork4.[Glyph](file:///C:\Workspace\template\doc\com\lab111\labwork4\Glyph.html)

Method draws a glyph in a given context.

[**draw(GlyphContext)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Line.html#draw-com.lab111.labwork4.GlyphContext-) - Method in class com.lab111.labwork4.[Line](file:///C:\Workspace\template\doc\com\lab111\labwork4\Line.html)

An implementation of method that's required by the interface.

**G**

[**getCharacter(char)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphFactory.html#getCharacter-char-) - Method in class com.lab111.labwork4.[GlyphFactory](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphFactory.html)

[**getChild(int)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Line.html#getChild-int-) - Method in class com.lab111.labwork4.[Line](file:///C:\Workspace\template\doc\com\lab111\labwork4\Line.html)

Method allows to get a child on the given index.

[**getLine(Glyph...)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphFactory.html#getLine-com.lab111.labwork4.Glyph...-) - Method in class com.lab111.labwork4.[GlyphFactory](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphFactory.html)

Method returns line.

[**getX()**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html#getX--) - Method in class com.lab111.labwork4.[GlyphContext](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html)

Getter for x-coordinate

[**getY()**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html#getY--) - Method in class com.lab111.labwork4.[GlyphContext](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html)

Getter for y-coordinate.

[**Glyph**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Glyph.html) - Interface in [com.lab111.labwork4](file:///C:\Workspace\template\doc\com\lab111\labwork4\package-summary.html)

A glyph interface.

[**GlyphContext**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html) - Class in [com.lab111.labwork4](file:///C:\Workspace\template\doc\com\lab111\labwork4\package-summary.html)

The class represents the context or the external state of the divisive glyph.

[**GlyphContext()**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html#GlyphContext--) - Constructor for class com.lab111.labwork4.[GlyphContext](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html)

Constructor by default, sets the coordinates at the start-point of (0;0).

[**GlyphFactory**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphFactory.html) - Class in [com.lab111.labwork4](file:///C:\Workspace\template\doc\com\lab111\labwork4\package-summary.html)

The factory class.

[**GlyphFactory()**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphFactory.html#GlyphFactory--) - Constructor for class com.lab111.labwork4.[GlyphFactory](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphFactory.html)

A basic constructor without parameters, initializes the HashMap.

[**GlyphFactory(Glyph...)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphFactory.html#GlyphFactory-com.lab111.labwork4.Glyph...-) - Constructor for class com.lab111.labwork4.[GlyphFactory](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphFactory.html)

Constructor with glyph-parameters.

**L**

[**Lab4**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Lab4.html) - Class in [com.lab111.labwork4](file:///C:\Workspace\template\doc\com\lab111\labwork4\package-summary.html)

[**Lab4()**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Lab4.html#Lab4--) - Constructor for class com.lab111.labwork4.[Lab4](file:///C:\Workspace\template\doc\com\lab111\labwork4\Lab4.html)

[**Line**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Line.html) - Class in [com.lab111.labwork4](file:///C:\Workspace\template\doc\com\lab111\labwork4\package-summary.html)

The class of the line (or string, semantically).

[**Line(Glyph...)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Line.html#Line-com.lab111.labwork4.Glyph...-) - Constructor for class com.lab111.labwork4.[Line](file:///C:\Workspace\template\doc\com\lab111\labwork4\Line.html)

Constructor, sets the "children" of this composite onject.

**M**

[**main(String[])**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Lab4.html#main-java.lang.String:A-) - Static method in class com.lab111.labwork4.[Lab4](file:///C:\Workspace\template\doc\com\lab111\labwork4\Lab4.html)

**N**

[**next()**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html#next--) - Method in class com.lab111.labwork4.[GlyphContext](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html)

Increments the number of symbols in line.

**R**

[**remove(int)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\Line.html#remove-int-) - Method in class com.lab111.labwork4.[Line](file:///C:\Workspace\template\doc\com\lab111\labwork4\Line.html)

Method allows to remove a child from the array.

**S**

[**setX(int)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html#setX-int-) - Method in class com.lab111.labwork4.[GlyphContext](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html)

Setter for x-coordinate

[**setY(int)**](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html#setY-int-) - Method in class com.lab111.labwork4.[GlyphContext](file:///C:\Workspace\template\doc\com\lab111\labwork4\GlyphContext.html)

Setter for y-coordinate