Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №4

З предмету: «Інженерія програмного забезпечення»

номер залікової книжки 2230

Виконав: студент групи ІО-22

Щербина М.В.

Перевірив: Абу Усбах А.Н.

Київ 2013р.

Варіант 8

8. Визначити специфікації класів, які подають графічні об'єкти у редакторі векторної графіки (прямокутник) через різні інтерфейси API1 та API2. Забезпечити прозору для користувача можливість заміни реалізації графічних об'єктів. Реалізувати метод малювання елементу.

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;

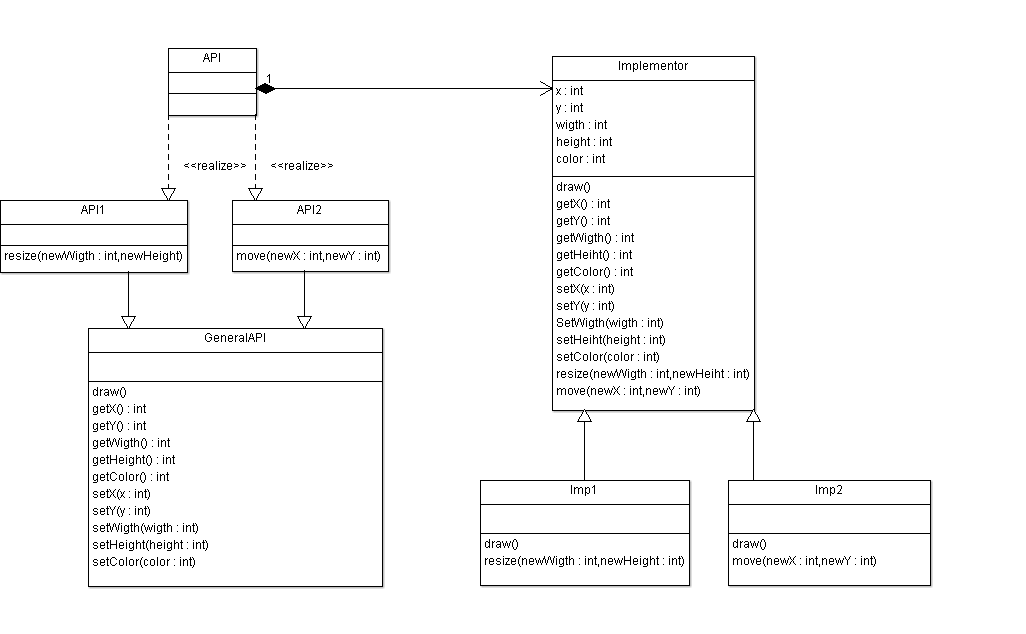
● знати особливості реалізації Шаблону, споріднені шаблони, відомі випадки його застосування в програмних додатках;

● 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.

4. 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). В розроблюваних класах повністю реалізувати методи, пов'язані з функціюванням Шаблону.

5. 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.



**package** com.lab111.labwork4;

**public** **interface** GeneralAPI {

**public** **void** draw();

**public** **int** getX();

**public** **int** getY();

**public** **int** getWigth();

**public** **int** getHeight();

**public** **int** getColor();

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

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

**public** **void** setWigth(**int** wigth);

**public** **void** setHeight(**int** height);

**public** **void** setColor(**int** color);

}

**package** com.lab111.labwork4;

/\*\*

\* Интрефейс 1

\*

\*/

**public** **interface** API1 **extends** GeneralAPI {

**public** **void** resize(**int** newWigth, **int** newHeight);

}

**package** com.lab111.labwork4;

/\*\*

\* Интрефейс 2

\*

\*/

**public** **interface** API2 **extends** GeneralAPI {

**public** **void** move(**int** newX, **int** newY);

}

**package** com.lab111.labwork4;

/\*\*

\* Общая реализация

\*

\*/

**public** **class** Implementor {

**protected** **int** x;

**protected** **int** y;

**protected** **int** wigth;

**protected** **int** height;

**protected** **int** color;

**public** Implementor(**int** x, **int** y, **int** wigth, **int** height, **int** color) {

**super**();

**this**.x = x;

**this**.y = y;

**this**.wigth = wigth;

**this**.height = height;

**this**.color = color;

}

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

**return** x;

}

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

**return** y;

}

**public** **int** getWigth() {

**return** wigth;

}

**public** **int** getHeiht() {

**return** height;

}

**public** **int** getColor() {

**return** color;

}

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

**this**.x = x;

}

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

**this**.y = y;

}

**public** **void** SetWigth(**int** wigth) {

**this**.wigth = wigth;

}

**public** **void** setHeiht(**int** height) {

**this**.height = height;

}

**public** **void** setColor(**int** color) {

**this**.color = color;

}

**public** **void** resize(**int** newWigth, **int** newHeiht) {

System.*out*.println("Implementor.resize()");

}

**public** **void** move(**int** newX, **int** newY) {

System.*out*.println("Implementor.move()");

}

**public** **void** draw() {

System.*out*.println("Implementor.draw()");

}

}

**package** com.lab111.labwork4;

/\*\*

\* Реализация 1

\*

\*/

**public** **class** Imp1 **extends** Implementor {

**public** Imp1(**int** x, **int** y, **int** wigth, **int** height, **int** color) {

**super**(x, y, wigth, height, color);

}

**public** **void** draw() {

System.*out*.println("Imp1.draw()");

}

**public** **void** resize(**int** newWigth, **int** newHeight) {

System.*out*.println("Imp1.resize()");

}

}

**package** com.lab111.labwork4;

/\*\*

\* Реализация 2

\*

\*/

**public** **class** Imp2 **extends** Implementor {

**public** Imp2(**int** x, **int** y, **int** wigth, **int** height, **int** color) {

**super**(x, y, wigth, height, color);

}

**public** **void** draw() {

System.*out*.println("Imp2.draw()");

}

**public** **void** move(**int** newX, **int** newY) {

System.*out*.println("Imp2.move()");

}

}

**package** com.lab111;

**import** com.lab111.labwork4.\*;

**public** **final** **class** TestMain {

/\*\*

\* Constructor.

\*

\*/

**private** TestMain() {

**super**();

}

/\*\*

\* Invokes at application startup.

\*

\* **@param** args

\* Parameters from command line

\*/

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

System.*out*.println("Интерфейс 1");

API1 rectangle = **new** API(10, 10, 5, 5, 100, 2);

rectangle.draw();

rectangle.resize(20, 7);

rectangle.draw();

System.*out*.println();

System.*out*.println("Интерфейс 2");

API2 rectangle2 = **new** API(10, 10, 5, 5, 100, 1);

rectangle2.draw();

rectangle2.move(1, 7);

rectangle2.draw();

}

}