Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №5

З предмету: «Інженерія програмного забезпечення»

номер залікової книжки 2230

Виконав: студент групи ІО-22

Щербина М.В.

Київ 2013р.

Варіант 7

7.Визначити специфікації класів для подання елементів графічного інтерфейсу користувача — GUI (вікна, кнопки, текстові області). Реалізувати механізм реакції на події в будь-якому з елементів.

**Завдання**

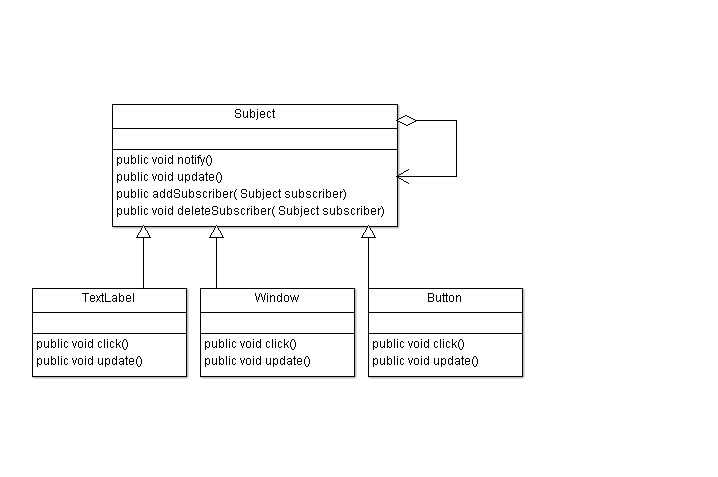
1. Вивчити шаблони поведінки для проектування ПЗ. Знати загальну характеристику шаблонів поведінки та призначення кожного з них.

2. Детально вивчити шаблони поведінки для проектування ПЗ - Iterator, Mediator та Observer. 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.

3. В підготованому проекті (ЛР1) створити програмний пакет com.lab111.labwork5. В пакеті розробити інтерфейси і класи, що реалізують завдання (згідно варіанту) з застосуванням одного чи декількох шаблонів (п.2). В розроблюваних класах 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.

4. 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;

**import** com.lab111.labwork5.\*;

**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) {

Button button = **new** Button();

Window window = **new** Window();

TextLabel text = **new** TextLabel();

button.addSubscriber(window);

text.addSubscriber(window);

button.click();

text.click();

}

}

**package** com.lab111.labwork5;

**public** **class** Window **extends** Subject {

**public** **void** click() {

System.*out*.println("Window.Click()");

notifySubcriber();

}

**public** **void** update(Subject updatedObject) {

System.*out*.println("Window принял уведомление об изменении "

+ updatedObject.toString());

}

**public** String toString() {

**return** "window";

}

}

**package** com.lab111.labwork5;

**public** **class** TextLabel **extends** Subject {

**public** **void** click() {

System.*out*.println("Text.Click()");

notifySubcriber();

}

**public** **void** update(Subject updatedObject) {

System.*out*.println("Text принял уведомление об изменении "

+ updatedObject.toString());

}

**public** String toString() {

**return** "text";

}

}

**package** com.lab111.labwork5;

**import** java.util.ArrayList;

**import** java.util.List;

/\*\*

\*

\*

\*/

**public** **class** Subject {

**public** List<Subject> subscribers;//

**public** Subject() {

**super**();

subscribers = **new** ArrayList<Subject>();

}

/\*\*

\* отправляет информацию о своем обновлении

\*/

**public** **void** notifySubcriber() {

**for** (Subject subscriber : subscribers) {

subscriber.update(**this**);

}

}

/\*\*

\* принимает обновленый обьект

\*

\* **@param** updatedObject

\*/

**public** **void** update(Subject updatedObject) {

}

/\*\*

\* добавляет нового подписчика

\*

\* **@param** subscriber

\*/

**public** **void** addSubscriber(Subject subscriber) {

subscribers.add(subscriber);

}

/\*\*

\* удаляет подписчика

\*

\* **@param** subscriber

\*/

**public** **void** deleteSubscriber(Subject subscriber) {

subscribers.remove(subscriber);

}

}

**package** com.lab111.labwork5;

**public** **class** Button **extends** Subject {

**public** **void** click() {

System.*out*.println("Button.Click()");

notifySubcriber();

}

**public** **void** update(Subject updatedObject) {

System.*out*.println("Button принял уведомление об изменении "

+ updatedObject.toString());

}

**public** String toString() {

**return** "button";

}

}