Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №7

З предмету: «Інженерія програмного забезпечення»

номер залікової книжки 2230

Виконав: студент групи ІО-22

Щербина М.В.

Київ 2013р.

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

2. Детально вивчити шаблони поведінки для проектування ПЗ -Memento, State, Command та Interpreter

. Для кожного з них:

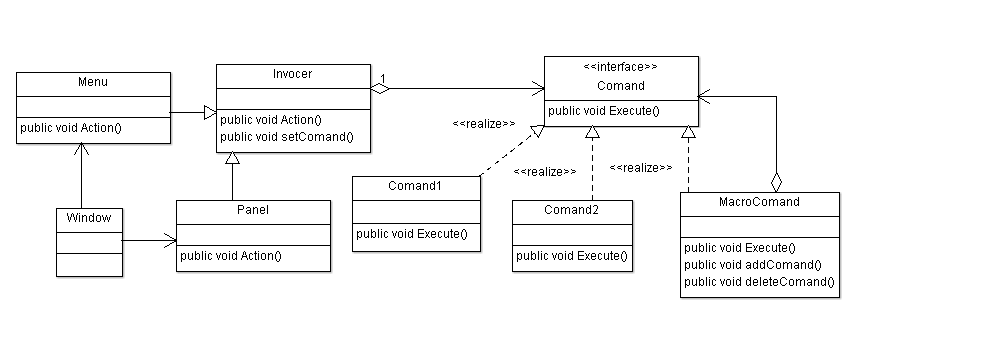
* вивчити Шаблон, його призначення, альтернативні назви, мотивацію, випадки коли його застосування є доцільним та результати такого застосування;
* знати особливості реалізації Шаблону, споріднені шаблони, відомі випадки його застосування в програмних додатках;
* вільно володіти структурою Шаблону, призначенням 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.labwork6. 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.

Варіант 6

Визначити специфікації класів, що подають реакції на натискання пунктів меню та кнопок інструментальної панелі. Забезпечити можливість динамічної зміни реакції, а також формування макро-реакцій (послідовність з наперед заданих реакцій)



**package** com.lab111;

**import** com.lab111.labwork7.\*;

**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) {

Comand1 comand1 = **new** Comand1();

Comand2 comand2 = **new** Comand2();

MacroComand macro = **new** MacroComand(comand2);

macro.addComand(comand1);

Menu menu = **new** Menu(comand1);

menu.Action();

System.*out*.println("Динамическая замена реакции на нажатие");

menu.setComand(comand2);

menu.Action();

Panel panel = **new** Panel(macro);

panel.Action();

}

}

**package** com.lab111.labwork7;

**public** **class** Panel **extends** Invocer {

**public** Panel(Comand c) {

**super**(c);

}

**public** **void** Action() {

System.*out*.println("действие панели");

**super**.Action();

}

}

**package** com.lab111.labwork7;

**public** **class** Menu **extends** Invocer {

**public** Menu(Comand c) {

**super**(c);

}

**public** **void** Action() {

System.*out*.println("действие меню");

**super**.Action();

}

}

**package** com.lab111.labwork7;

**import** java.util.ArrayList;

/\*\*

\* Макро-команда

\*

\*/

**public** **class** MacroComand **implements** Comand {

**public** MacroComand(Comand c) {

comand = **new** ArrayList<Comand>();

comand.add(c);

}

**private** ArrayList<Comand> comand;

/\*

\* выполнение макро-команды

\*/

**public** **void** Execute() {

System.*out*.println("Макро-команда");

**for** (Comand element : comand)

element.Execute();

}

**public** **void** addComand(Comand newComand) {

comand.add(newComand);

}

**public** **void** deleteComand(Comand deleteComand) {

comand.add(deleteComand);

}

}

**package** com.lab111.labwork7;

/\*\*

\* Инициатор посылки сообщения

\*

\*/

**public** **class** Invocer {

Invocer(Comand c) {

comand = c;

}

**public** Comand comand;

/\*\*

\* выполнить действие

\*/

**public** **void** Action() {

comand.Execute();

}

/\*\*

\* изменить команду

\*

\* **@param** newComand

\* - новая команда

\*/

**public** **void** setComand(Comand newComand) {

comand = newComand;

}

}

**package** com.lab111.labwork7;

/\*\*

\* Команда 2

\*

\*/

**public** **class** Comand2 **implements** Comand {

**public** **void** Execute() {

System.*out*.println("Команда 2");

}

}

**package** com.lab111.labwork7;

/\*\*

\* Команда 1

\*

\*/

**public** **class** Comand1 **implements** Comand {

**public** **void** Execute() {

System.*out*.println("Команда 1");

}

}

**package** com.lab111.labwork7;

/\*\*

\* Интерфейс описывающий команду

\*

\*/

**public** **interface** Comand {

**public** **void** Execute();

}