Міністерство освіти і науки, молоді та спорту України

Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота №9**

По дисципліні «Інженерія програмного забезпечення»

Виконав: Перевірив:

Студент групи ІО-21 Викладач

Коноз А.О. Абу Усбах О. Н.

Дата здачі\_\_\_\_\_\_\_\_\_\_\_\_\_

Захищено з балом\_\_\_\_\_

Київ 2013

**Завдання**

1. Повторити шаблони, що породжують. Знати загальну характеристику шаблонів, що породжують та призначення кожного з них.

2. Детально вивчити шаблони, що породжують - Abstract Factory та Builder. 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.

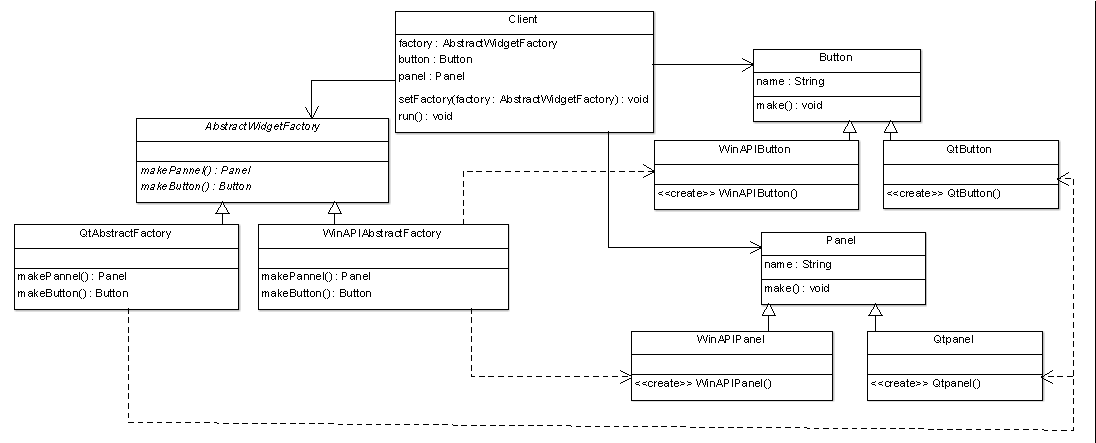
3. В підготованому проекті (ЛР1) створити програмний пакет com.lab111.labwork9. В пакеті розробити інтерфейси і класи, що реалізують завдання (згідно варіанту) з застосуванням одного чи декількох шаблонів (п.2). В розроблюваних класах повністю реалізувати методи, пов'язані з функціюванням Шаблону. Методи, що реалізують бізнес-логіку закрити заглушками з виводом на консоль інформації про викликаний метод та його аргументи. Приклад реалізації бізнес-методу:

void draw(int x, int y){

System.out.println(“Метод draw з параметрами x=”+x+” y=”+y);

}

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.labwork9;

/\*\*

\* **@author** User

\* Main class

\*/

**public** **class** Main {

/\*\*

\* **@param** args

\*/

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

AbstractWidgetFactory factory1 = **new** WinAPIAbstractFactory();

AbstractWidgetFactory factory2 = **new** QtAbstractFactory();

Client client = **new** Client();

client.setFactory(factory1);

client.run();

client.setFactory(factory2);

client.run();

}

}

**package** com.lab111.labwork9;

/\*\*

\* **@author** User

\* Class implement function of factory for Qt widget

\*/

**public** **class** QtAbstractFactory **extends** AbstractWidgetFactory{

@Override

**public** Panel makePannel() {

**return** **new** Qtpanel();

}

@Override

**public** Button makeButton() {

**return** **new** QtButton();

}

}

**package** com.lab111.labwork9;

/\*\*

\* **@author** User

\*Class for panel

\*/

**public** **class** Panel {

**public** String name;

**public** **void** make(){

System.*out*.println(name + " panel");

}

}

**package** com.lab111.labwork9;

/\*\*

\* **@author** User

\* Class implement function of client

\*/

**public** **class** Client {

/\*\*

\* fields of client

\*/

**private** AbstractWidgetFactory factory;

**private** Button button;

**private** Panel panel;

/\*\*

\* Set factory

\* **@param** factory

\*/

**public** **void** setFactory(AbstractWidgetFactory factory){

**this**.factory = factory;

}

/\*\*

\* Run application

\*/

**public** **void** run(){

button = factory.makeButton();

panel = factory.makePannel();

button.make();

panel.make();

}

}

**package** com.lab111.labwork9;

**public** **class** Qtpanel **extends** Panel{

**public** Qtpanel(){

**super**.name = "Qt";

}

}

**package** com.lab111.labwork9;

**public** **class** QtButton **extends** Button {

**public** QtButton(){

**super**.name = "Qt";

}

}

**package** com.lab111.labwork9;

/\*\*

\* **@author** User

\* Interface of factory

\*/

**public** **abstract** **class** AbstractWidgetFactory {

**public** **abstract** Panel makePannel();

**public** **abstract** Button makeButton();

}

**package** com.lab111.labwork9;

/\*\*

\* **@author** User

\* Class for button

\*/

**public** **class** Button {

**public** String name;

**public** **void** make(){

System.*out*.println(name + " button");

}

}

**package** com.lab111.labwork9;

**public** **class** WinAPIPanel **extends** Panel{

**public** WinAPIPanel (){

**super**.name = "WinAPI";

}

}

**package** com.lab111.labwork9;

**public** **class** WinAPIButton **extends** Button{

**public** WinAPIButton(){

**super**.name = "WinAPI";

}

}

**package** com.lab111.labwork9;

**public** **class** WinAPIAbstractFactory **extends** AbstractWidgetFactory {

@Override

**public** Panel makePannel() {

**return** **new** WinAPIPanel();

}

@Override

**public** Button makeButton() {

**return** **new** WinAPIButton();

}

}