**Software Engineering**

**Laboratory work №7**

**"BEHAVIORS. MEMENTO, STATE, COMMAND, INTERPRETER templates »**

Completed:

student of IО-83

Malashkin Vyacheslav

Checked:

Antonyuk A.I.

Kiev 2019

**Laboratory work №7**

**Objective:** To study behavior patterns. Learn basic skills in applying Memento, State, Command, and Interpreter templates.

1. ***Option task***

8318 mod 8 = 6

6. Define the specifications of the classes that respond to menu items and toolbar buttons. Provide for dynamic change of reaction, as well as formation of macro reactions (sequence of predefined reactions).

1. ***Program code***

**package** com.lab111.labwork7;

/\*\*

\* **@author** vch\_m

\* **@version** 7.1

\* In this labwork I used pattern Сommand

\*/

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

/\*\*

\* Constructor.

\*

\*/

**private** Main() {

**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("Dynamic replacement of the push response");

menu.setComand(comand2);

menu.Action();

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

panel.Action();

}

}

**package** com.lab111.labwork7;

/\*\*

\* **@author** vch\_m

\* **@version** 7.1

\*/

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

**public** Panel(Comand c) {

**super**(c);

}

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

System.*out*.println("panel action");

**super**.Action();

}

}

**package** com.lab111.labwork7;

/\*\*

\* **@author** vch\_m

\* **@version** 7.1

\*/

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

**public** Menu(Comand c) {

**super**(c);

}

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

System.*out*.println("menu action");

**super**.Action();

}

}

**package** com.lab111.labwork7;

**import** java.util.ArrayList;

/\*\*

\* **@author** vch\_m

\* **@version** 7.1

\* Macro command

\*/

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

**public** MacroComand(Comand c) {

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

comand.add(c);

}

**private** ArrayList<Comand> comand;

/\*

\* macro command execution

\*/

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

System.*out*.println("macro command");

**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;

/\*\*

\* **@author** vch\_m

\* **@version** 7.1

\* Message initiator

\*/

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

Invocer(Comand c) {

comand = c;

}

**public** Comand comand;

/\*\*

\* perform an action

\*/

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

comand.Execute();

}

/\*\*

\* change team

\*

\* **@param** newComand

\* - new command

\*/

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

comand = newComand;

}

}

**package** com.lab111.labwork7;

/\*\*

\* **@author** vch\_m

\* **@version** 7.1

\* command 2

\*/

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

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

System.*out*.println("command 2");

}

}

**package** com.lab111.labwork7;

/\*\*

\* **@author** vch\_m

\* **@version** 7.1

\* command 1

\*/

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

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

System.*out*.println("command 1");

}

}

**package** com.lab111.labwork7;

/\*\*

\* **@author** vch\_m

\* **@version** 7.1

\* Interface describing the command

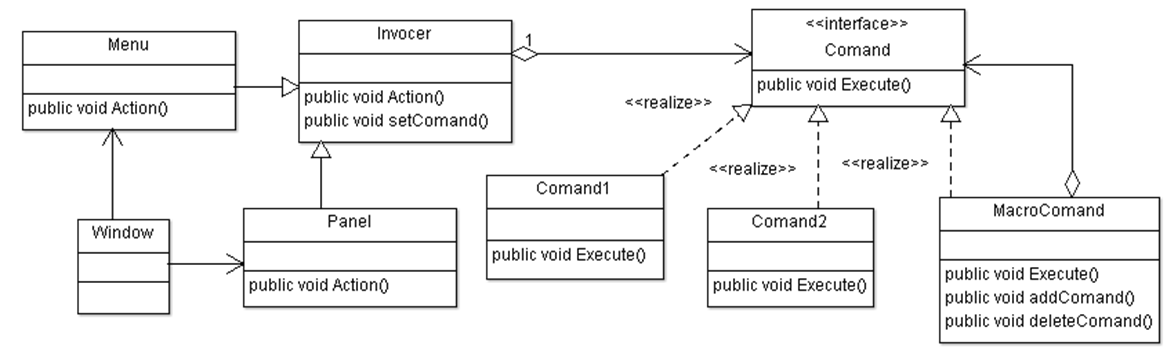
\*/

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

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

}

1. ***Class diagram***

****

1. ***Test results***

menu action

Team 1

Dynamic replacement of the push response

menu action

Team 2

panel action

Macro command

Team 2

Team 1

**Conclusion:** In the course of laboratory work, I studied the required patterns and implemented one of them on the example of a laboratory task.