КИЇВСЬКИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

**Лабораторна робота №9**

З інженерії програмного забезпечення

**Тема**: «ПОРОДЖУВАЛЬНІ ШАБЛОНИ. ШАБЛОНИ ABSTRACT FACTORY, BUILDER»

залікова книжка № 3223

Виконав:

студент ІІ курсу ФІОТ

група ІО-32

Попенко Руслан

Перевірив:

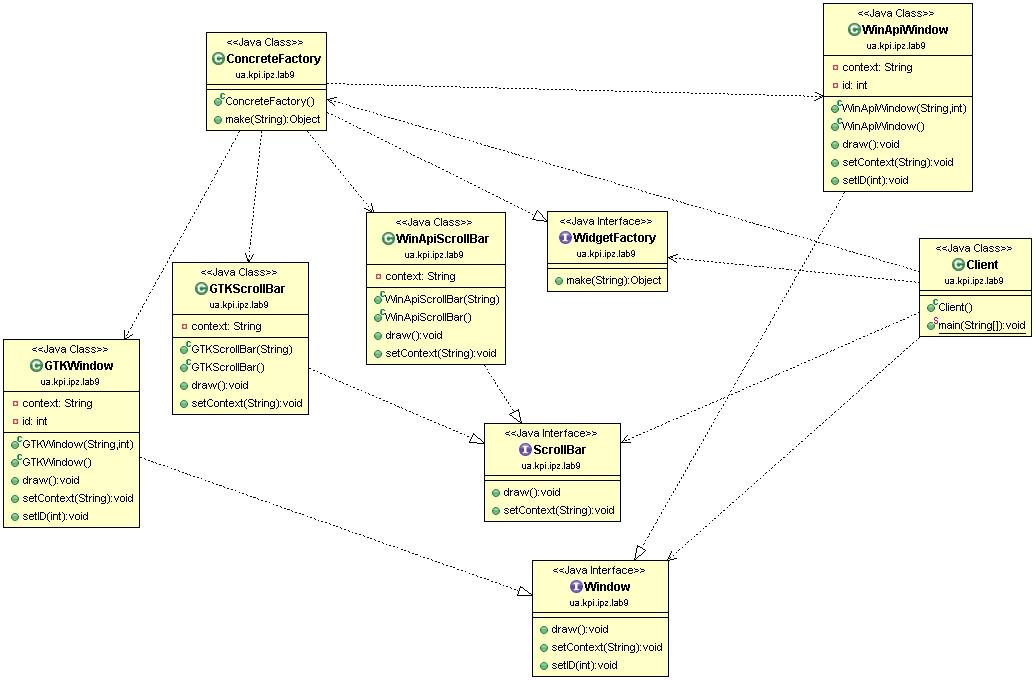
Антонюк А. І.

Київ-2014

**Мій варіант 0**

*Визначити специфікації класів для подання сімейства віджетів графічного інтерфейсу користувача з реалізацією на різних АРІ (WinAPI, GTK). Забезпечити можливість прозорого для клієнта розширення реалізацією для інших API (Qt, OSX).*

**Діаграма класів**

****

**Роздруківка тексту**

**package** ua.kpi.ipz.lab9;

/\*\*

\* A class that needs to create new objects of some kind

\* without information about the internal structure (hence, without knowing

\* the way to create) of those objects. Thus, it uses abstract factory to create new

\* objects.

\* **@author** Ruslan Popenko

\* **@version** 9.0

\*

\*/

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

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

WidgetFactory **factory**=**new** ConcreteFactory();

ScrollBar **bar1**=(ScrollBar)factory.make("gtk scrollbar");

ScrollBar **bar2**=(ScrollBar)factory.make("winapi scrollbar");

Window **win1**=(Window)factory.make("gtk window");

Window **win2**=(Window)factory.make("winapi window");

bar1.setContext("bar1");

bar2.setContext("bar2");

win1.setContext("win");

win1.setID(1);

win2.setContext(("win"));

win2.setID(2);

bar1.draw();

bar2.draw();

win1.draw();

win2.draw();

}

}

**package** ua.kpi.ipz.lab9;

/\*\*

\* A class represents a factory of different Products with a function of

\* dispatchering. Client can get s apecific Product by knowing the key, that is

\* set to the specific prototype (or simply an order to create)

\*

\* **@author** Ruslan Popenko

\* **@version** 9.0

\*/

**public** **class** **ConcreteFactory** **implements** WidgetFactory {

/\*\*

\* An ordinary constructor by default

\*/

**public** **ConcreteFactory**() {

}

/\*\*

\* A method that returns a specific product (according to the key).

\*

\* **@param** key

\* a string, key that is associated with specific product.

\*/

**public** **Object** **make**(**String** key) {

**if** (key.equalsIgnoreCase("GTK scrollbar"))

**return** **new** GTKScrollBar();

**if** (key.equalsIgnoreCase("GTK window"))

**return** **new** GTKWindow();

**if** (key.equalsIgnoreCase("WinApi scrollbar"))

**return** **new** WinApiScrollBar();

**if** (key.equalsIgnoreCase("WinApi window"))

**return** **new** WinApiWindow();

**else**

**return** 0;

}

}

**package** ua.kpi.ipz.lab9;

/\*\*

\* The class represents a specific implementation of scrollbar for GTK graph.system.

\* **@author** Ruslan Popenko

\* **@version** 9.0

\*/

**public** **class** **GTKScrollBar** **implements** ScrollBar{

**private** **String** context;

/\*\*

\* Constructor that sets a context (spec.information)

\* **@param** context specific information, String.

\*/

**public** **GTKScrollBar**(**String** context){

setContext(context);

}

/\*\*

\* A constructor by default.

\*/

**public** **GTKScrollBar**(){

context=**new** String("");

}

/\*\*

\* Method prints some basic information about a product. Nothing important, that's

\* just a cap, so it gives opportunity to identificate a concrete object.

\*/

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

**System**.***out***.println("A GTK Scrollbar "+context);

}

/\*\*

\* The method sets a context (some specific information) of a scrollbar.

\*/

**public** **void** **setContext**(**String** cont){

context=cont;

}

}

**package** ua.kpi.ipz.lab9;

/\*\*

\* The class represents a specific implementation of a window for GTK graph.system.

\* **@author** Ruslan Popenko

\* **@version** 9.0

\*/

**public** **class** **GTKWindow** **implements** Window{

**private** **String** context;

**private** **int** id;

/\*\*

\* Constructor that sets a context (spec.information) and an id of window

\* **@param** context specific information, String.

\* **@param** id identifier.

\*/

**public** **GTKWindow**(**String** context, **int** id){

**this**.context=context;

**this**.id=id;

}

/\*\*

\* A constructor by default.

\*/

**public** **GTKWindow**(){

context=**new** String("");

id=0;

}

/\*\*

\* Method prints some basic information about a product. Nothing important, that's

\* just a cap, so it gives opportunity to identify a concrete object.

\*/

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

**System**.***out***.println("GTK Window "+context+" "+id);

}

/\*\*

\* The method sets a context (some specific information) of a window.

\*/

**public** **void** **setContext**(**String** cont){

context=cont;

}

/\*\*

\* The method sets an identifier of a window.

\*/

**public** **void** **setID**(**int** id){

**this**.id=id;

}

}

**package** ua.kpi.ipz.lab9;

/\*\*

\* A general interface for Scrollbar-items.

\*

\* **@author** Ruslan Popenko

\* **@version** 9.0

\*

\*/

**public** **interface** ScrollBar{

/\*\*

\* Method allows to print information about a product (scrollbar of some kind)

\*/

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

/\*\*

\* Method sets some specific information (quite abstract), "caps" for a product

\* **@param** cont context

\*/

**public** **void** **setContext**(**String** cont);

}

**package** ua.kpi.ipz.lab9;

/\*\*

\* A class represents a factory of environment-elements with a function of

\* dispatchering. Objects of this class aggregate prototypes with specific keys.

\* Client can get access to a prototype via the key, that is set to the specific

\* prototype.

\* The factory also returns clones of the requested prototype.

\*

\* **@author** Ruslan Popenko

\* **@version** 9.0

\*

\*/

**public** **interface** WidgetFactory {

**public** **Object** **make**(**String** key);

}

**package** ua.kpi.ipz.lab9;

/\*\*

\* The class represents a specific implementation of scrollbar for WinApi graph.system.

\* **@author** Ruslan Popenko

\* **@version** 9.0

\*/

**public** **class** **WinApiScrollBar** **implements** ScrollBar{

**private** **String** context;

/\*\*

\* Constructor that sets a context (spec.information)

\* **@param** context specific information, String.

\*/

**public** **WinApiScrollBar**(**String** context){

setContext(context);

}

/\*\*

\* A constructor by default.

\*/

**public** **WinApiScrollBar**(){

context=**new** String("");

}

/\*\*

\* Method prints some basic information about a product. Nothing important, that's

\* just a cap, so it gives opportunity to identificate a concrete object.

\*/

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

**System**.***out***.println("A WinAPI Scrollbar "+context);

}

/\*\*

\* The method sets a context (some specific information) of a scrollbar.

\*/

**public** **void** **setContext**(**String** cont){

context=cont;

}

}

**package** ua.kpi.ipz.lab9;

/\*\*

\* The class represents a specific implementation of window for WinApi graph.system.

\* **@author** Ruslan Popenko

\* **@version** 9.0

\*/

**public** **class** **WinApiWindow** **implements** Window{

**private** **String** context;

**private** **int** id;

/\*\*

\* Constructor that sets a context (spec.information) and an id of window

\* **@param** context specific information, String.

\* **@param** id identifier.

\*/

**public** **WinApiWindow**(**String** context, **int** id){

**this**.context=context;

**this**.id=id;

}

/\*\*

\* A constructor by default.

\*/

**public** **WinApiWindow**(){

context=**new** String("");

id=0;

}

/\*\*

\* Method prints some basic information about a product. Nothing important, that's

\* just a cap, so it gives opportunity to identify a concrete object.

\*/

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

**System**.***out***.println("WinAPI window "+context+" "+id);

}

/\*\*

\* The method sets a context (some specific information) of a window.

\*/

**public** **void** **setContext**(**String** cont){

context=cont;

}

/\*\*

\* The method sets an identifier of a window.

\*/

**public** **void** **setID**(**int** id){

**this**.id=id;

}

}

**package** ua.kpi.ipz.lab9;

/\*\*

\* A general interface for window-items.

\*

\* **@author** Ruslan Popenko

\* **@version** 9.0

\*

\*/

**public** **interface** Window {

/\*\*

\* Method allows to print information about a product (window of some kind)

\*/

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

/\*\*

\* Method sets some specific information (quite abstract), "caps" for a product

\* **@param** cont context

\*/

**public** **void** **setContext**(**String** cont);

/\*\*

\* Method sets an id-number of a window.

\* **@param** id identification number, int.

\*/

**public** **void** **setID**(**int** id);

}

Результат роботи програми

A GTK Scrollbar bar1

A WinAPI Scrollbar bar2

GTK Window win 1

WinAPI window win 2