Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота № 9

з дисципліни: «Інженерія програмного забезпечення»

Тема: «Шаблони, що породжують. Шаблони Abstract Factory та Builder»

Виконав:

студент групи ІО-33

Заколенко Роман

Перевірив:

Антонюк А.І.

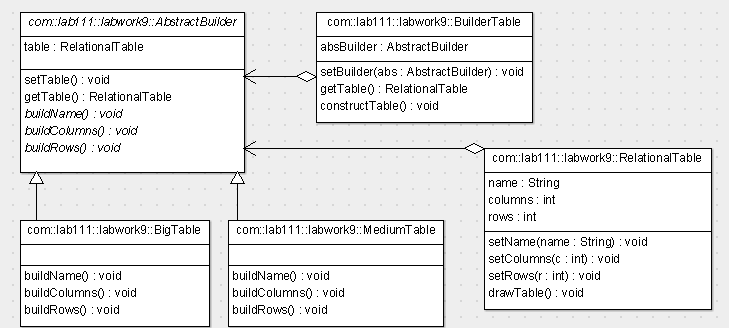
2014

**Мета роботи:** Вивчення породжувальних шаблонів. Отримання базових навичок з застосування шаблонів Abstract Factory, Builder.

**Варіант завдання - 8 (3308 mod 11 = 8) :**

Визначити специфікації класів для подання реляційної таблиці та будівника проекції таблиць.

**UML-діаграма**



**Лістинг**

**package** com.lab111.labwork9;

/\*\*

\* This class contains data of relation table

\*

\* **@author** Roman Zakolenko

\* **@version** 9.8

\*/

**public** **class** RelationalTable {

/\*\*

\* This field contains name of table

\*/

**private** String name;

/\*\*

\* This field contains amount of table columns

\*/

**private** **int** columns;

/\*\*

\* This field contains amount of table rows

\*/

**private** **int** rows;

/\*\*

\* **@param** name

\* assigned with the field name

\*/

**public** **void** setName(String name) {

**this**.name = name;

}

/\*\*

\* **@param** c

\* assigned with the field columns

\*/

**public** **void** setColumns(**int** c) {

**this**.columns = c;

}

/\*\*

\* **@param** r

\* assigned with the field rows

\*/

**public** **void** setRows(**int** r) {

**this**.rows = r;

}

/\*\*

\* This method draws a table

\*/

**public** **void** drawTable() {

System.*out*.println(**this**.name + " : columns = " + **this**.columns

+ ", rows = " + **this**.rows);

}

}

}

**package** com.lab111.labwork9;

/\*\*

\* This class extends AbstractBuilder

\*

\* **@author** Roman Zakolenko

\* **@version** 9.8

\*/

**public** **class** BigTable **extends** AbstractBuilder {

/\* (non-Javadoc)

\* @see com.lab111.labwork9.AbstractBuilder#buildName()

\*/

@Override

**public** **void** buildName() {

**this**.table.setName("Big Table");

}

/\* (non-Javadoc)

\* @see com.lab111.labwork9.AbstractBuilder#buildColumns()

\*/

@Override

**public** **void** buildColumns() {

**this**.table.setColumns(100);

}

/\* (non-Javadoc)

\* @see com.lab111.labwork9.AbstractBuilder#buildRows()

\*/

@Override

**public** **void** buildRows() {

**this**.table.setRows(100);

}

}

**package** com.lab111.labwork9;

/\*\*

\* This class extends AbstractBuilder

\*

\* **@author** Roman Zakolenko

\* **@version** 9.8

\*/

**public** **class** MediumTable **extends** AbstractBuilder {

/\* (non-Javadoc)

\* @see com.lab111.labwork9.AbstractBuilder#buildName()

\*/

@Override

**public** **void** buildName() {

**this**.table.setName("Medium Table");

}

/\* (non-Javadoc)

\* @see com.lab111.labwork9.AbstractBuilder#buildColumns()

\*/

@Override

**public** **void** buildColumns() {

**this**.table.setColumns(10);

}

/\* (non-Javadoc)

\* @see com.lab111.labwork9.AbstractBuilder#buildRows()

\*/

@Override

**public** **void** buildRows() {

**this**.table.setRows(10);

}

}

**package** com.lab111.labwork9;

/\*\*

\* This abstract class AbstractBuilder

\*

\* **@author** Roman Zakolenko

\* **@version** 9.8

\*/

**public** **abstract** **class** AbstractBuilder {

/\*\*

\* This fields contains link of object RelationTable

\*/

**public** RelationalTable table;

/\*\*

\* This method sets the value of the field table

\*/

**public** **void** setTable() {

**this**.table = **new** RelationalTable();

}

/\*\*

\* **@return** value of field table

\*/

**public** RelationalTable getTable() {

**return** **this**.table;

}

/\*\*

\* Abstract method buildName()

\*/

**public** **abstract** **void** buildName();

/\*\*

\* Abstract method buildColumns()

\*/

**public** **abstract** **void** buildColumns();

/\*\*

\* Abstract method buildRows()

\*/

**public** **abstract** **void** buildRows();

}

**package** com.lab111.labwork9;

/\*\*

\* The class is created through which a particular object

\*

\* **@author** Roman Zakolenko

\* **@version** 9.8

\*/

**public** **class** BuilderTable {

/\*\*

\* This field contains link of object AbstractBuilder

\*/

**private** AbstractBuilder absBuilder;

/\*\*

\* **@param** abs

\* assigned with the field absBuilder

\*/

**public** **void** setBuilder(AbstractBuilder abs) {

**this**.absBuilder = abs;

}

/\*\*

\* **@return** link of object RelationTable

\*/

**public** RelationalTable getTable() {

**return** **this**.absBuilder.getTable();

}

/\*\*

\* This method constructs the table

\*/

**public** **void** constructTable() {

absBuilder.setTable();

absBuilder.buildName();

absBuilder.buildColumns();

absBuilder.buildRows();

}

}

**package** com.lab111.labwork9;

/\*\*

\* **@author** Roman Zakolenko

\* **@version** 9.8

\*/

**public** **class** Lab9 {

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

BuilderTable builder = **new** BuilderTable();

builder.setBuilder(**new** BigTable());

builder.constructTable();

RelationalTable bigTable = builder.getTable();

bigTable.drawTable();

builder.setBuilder(**new** MediumTable());

builder.constructTable();

RelationalTable mediumTable = builder.getTable();

mediumTable.drawTable();

}

}

**Результати роботи програми**

Big Table : columns = 100, rows = 100

Medium Table : columns = 10, rows = 10

**Висновки:**

В результаті виконання лабораторної роботи я вивчив два породжувальних шаблони, а саме Abstract Factory та Builder, а також практично реалізував шаблон Builder.