МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра программной инженерии

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«_27_» _12_ 2017 г.

ШАБЛОН ПРОЕКТИРОВАНИЯ «КОМПОНОВЩИК»

Методические указания по выполнению лабораторной работы по дисциплине "Проектирование и архитектура программных систем" для студентов направления подготовки 09.03.04 "Программная инженерия"

Курск 2017

УДК 004.652

Составители: В.Г. Белов, Т.М. Белова

Рецензент
Кандидат технических наук, доцент кафедры программной инженерии ЮЗГУ И.Н. Ефремова

Шаблон проектирования «Компоновщик»: методические указания по выполнению лабораторной работы по дисциплине "Проектирование и архитектура программных систем" для студентов направления подготовки 09.03.04 "Программная инженерия" / Юго-Зап. гос. ун-т; сост.: В.Г. Белов, Т.М. Белова, – Курск, 2017. – 24 с.: ил. 30.

Изложена последовательность действий по реализации и использованию шаблона проектирования «Компоновщик» в интегрированной среде разработки Eclipse.

Материал предназначен для студентов направления подготовки бакалавров 09.03.04 «Программная инженерия», а также будет полезен студентам всех направлений подготовки, изучающим технологии проектирования программно-информационных систем.

Текст печатается в авторской редакции.

Подписано в печать 24.12.17. Формат 60x84 1/16.
Усл. печ. л. 1,4. Уч.-изд. л. 1,3. Тираж 100 экз. Заказ 4364. Бесплатно.
Юго-Западный государственный университет
305040, Курск, ул.50 лет Октября, 94.

Содержание

1 Цель лабораторной работы

Целью лабораторной работы является приобретение знаний умений и навыков для использования возможностей шаблона проектирования «Компоновщик».

Данный шаблон предназначен, прежде всего, для использования в таких случаях, когда нужно обраться к отдельным объектам и к группам объектов одинаково. Задача состоит в том, что необходимо реализовать алгоритмы добавления, удаления и просмотра элементов в древовидной структуре.

2 Порядок выполнения лабораторной работы

1. В интегрированной среде разработке Eclipse создайть новый Java-проект Coposite для разработки шаблона проектирования «Компоновщик» (рисунки 1–2).

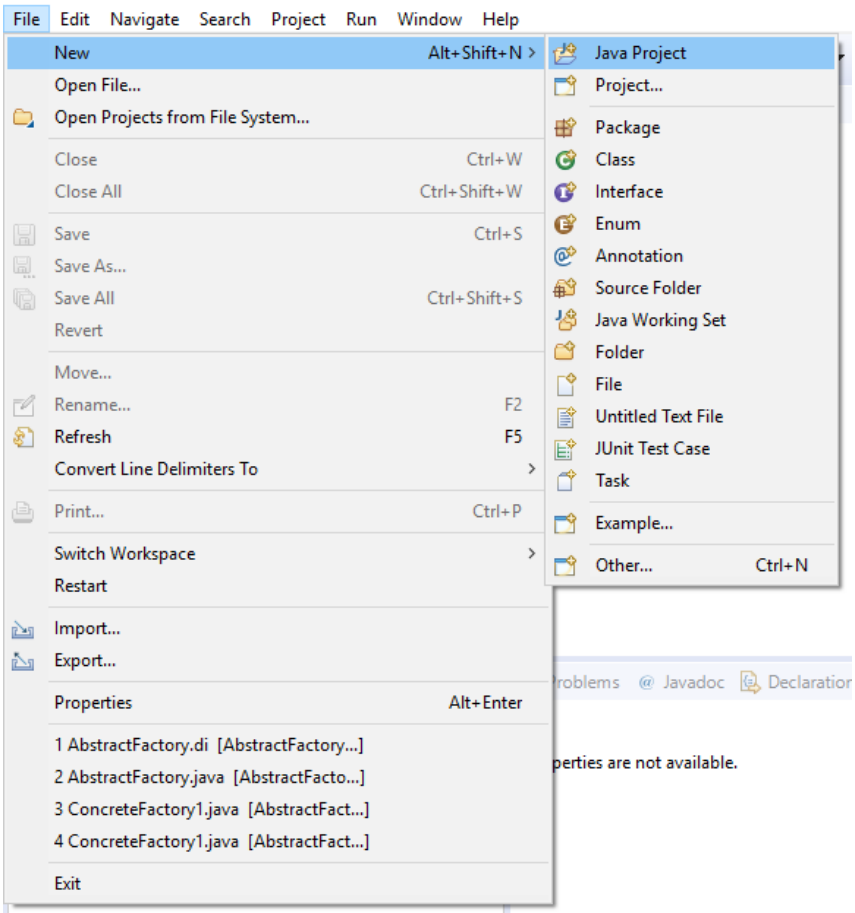

Рисунок 1

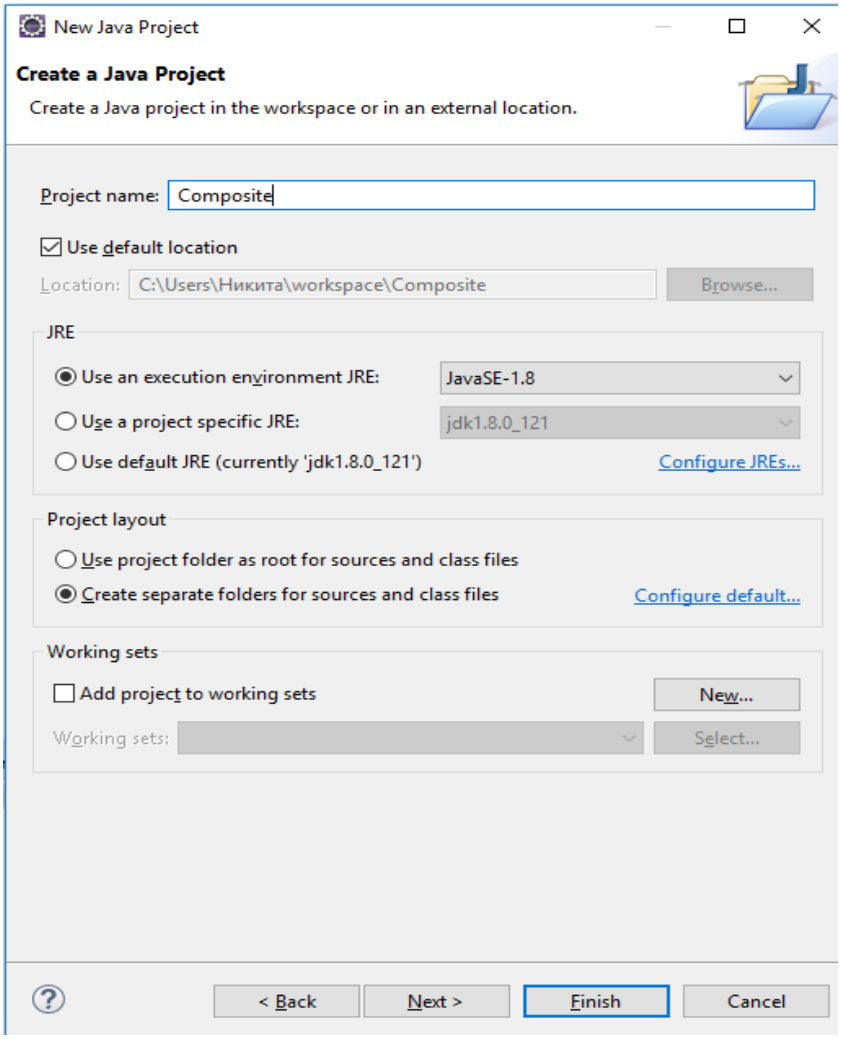

Рисунок 2

2. В проекте создайте папку Model для проектирования диаграммы классов (рисунки 3–4). В созданной папке добавьте необходимые файлы для работы с инструментом проектирования UML-диаграмм Papyrus (рисунки 5–10).

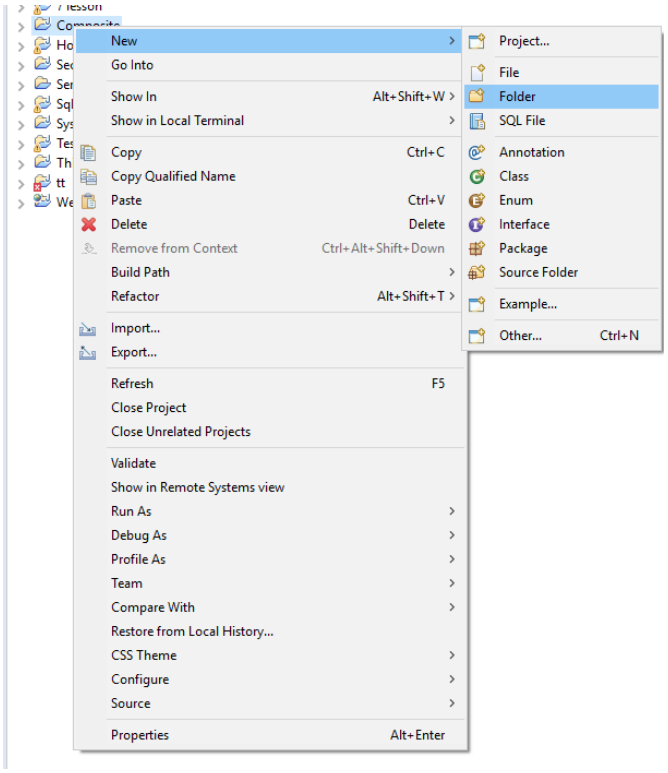
New
Go Into
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
Close Project
Close Unrelated Projects
Validate
Show in Remote Systems view
Run As
Debug As
Profile As
Team
Compare With
Restore from Local History...
CSS Theme
Configure
Source
Properties Alt+Enter
Project...
File
Folder
SQL File
Annotation
Class
Enum
Interface
Package
Source Folder
Example...
Other... Ctrl+N

Рисунок 3

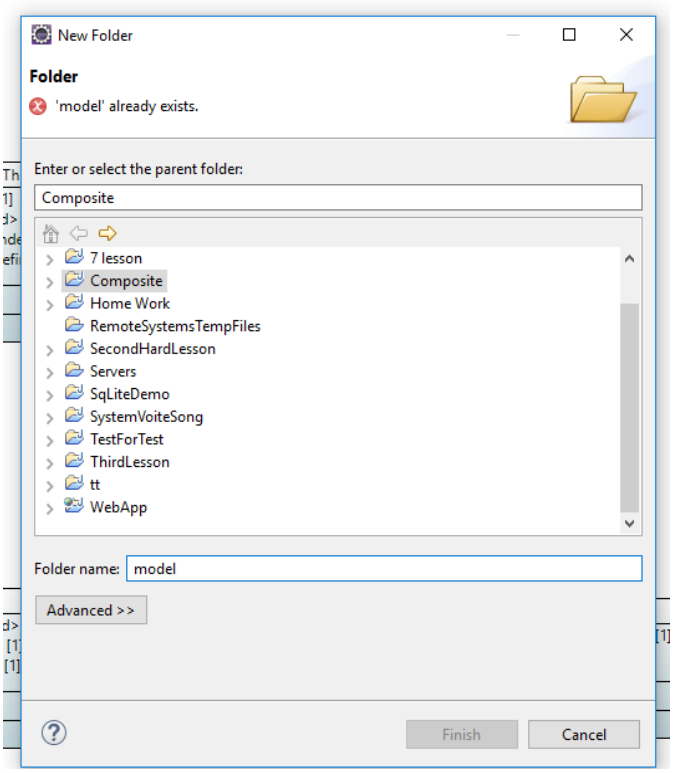
New Folder
Folder
'model' already exists.
Enter or select the parent folder:
Composite
7 lesson
Composite
Home Work
RemoteSystemsTempFiles
SecondHardLesson
Servers
SqLiteDemo
SystemVoiteSong
TestForTest
ThirdLesson
tt
WebApp
Folder name: model
Advanced >>
Finish
Cancel

Рисунок 4

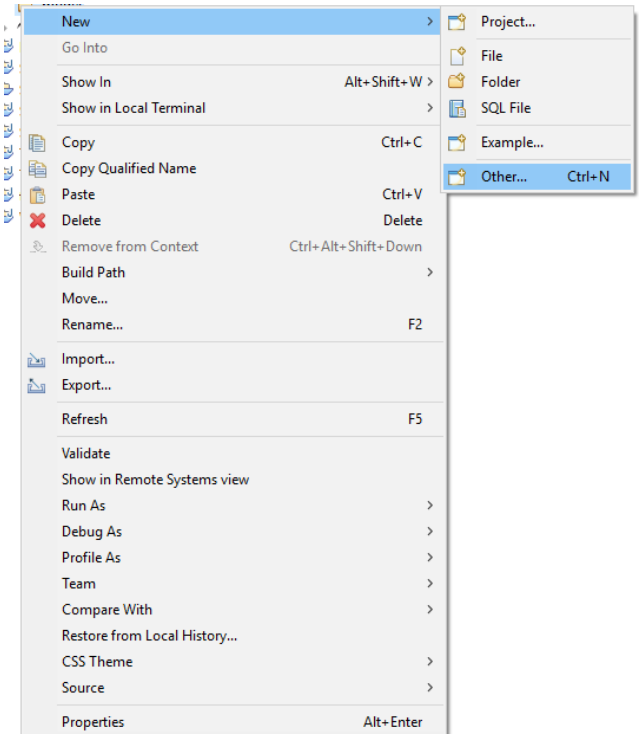

Рисунок 5

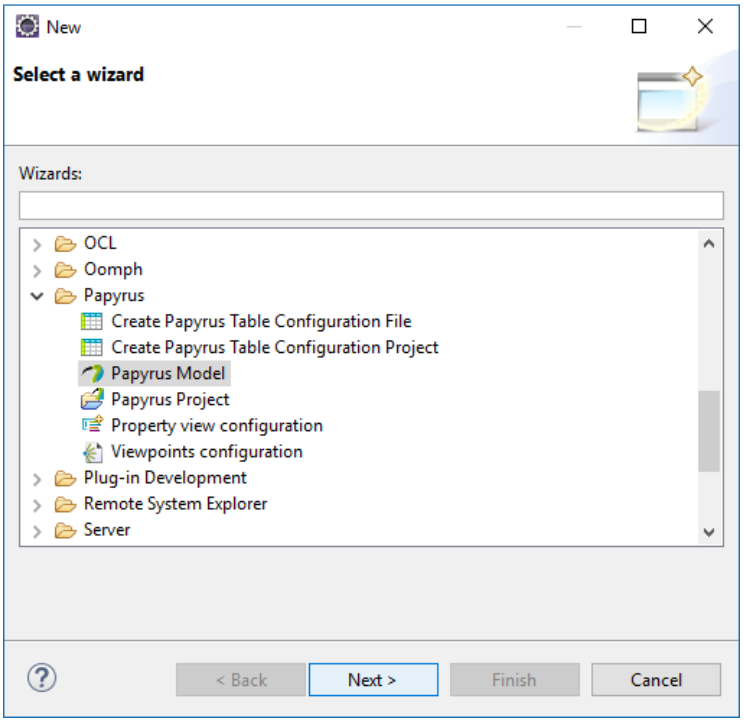

Рисунок 6

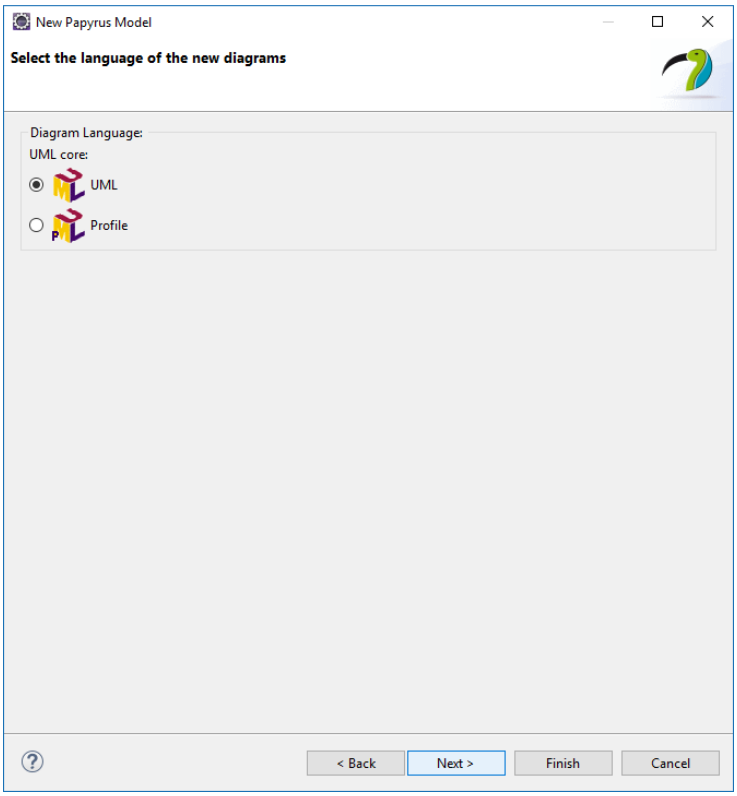

Рисунок 7

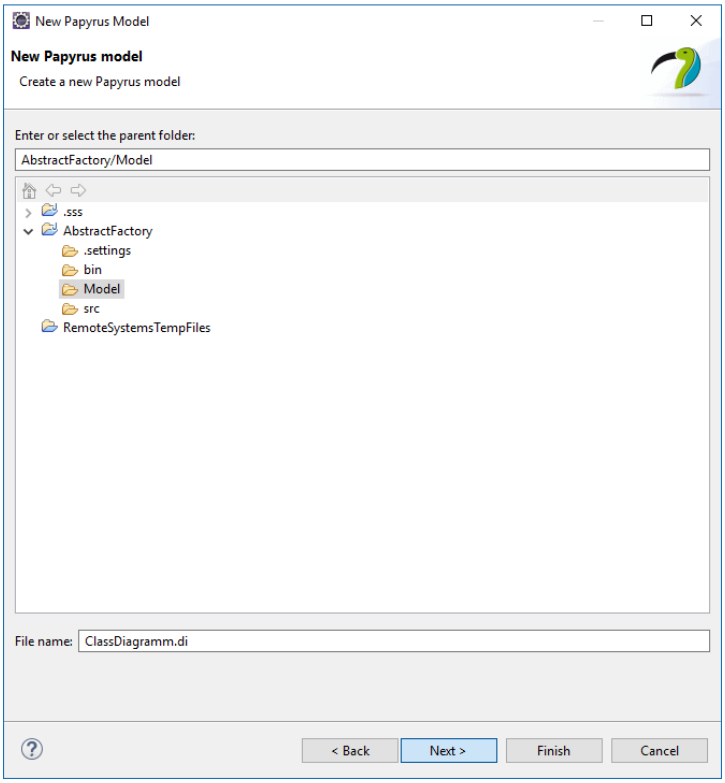
New Papyrus Model
New Papyrus model
Create a new Papyrus model
Enter or select the parent folder:
AbstractFactory/Model
.sss
AbstractFactory
.settings
bin
Model
src
RemoteSystemsTempFiles
File name:
ClassDiagramm.di
< Back
Next >
Finish
Cancel

Рисунок 8

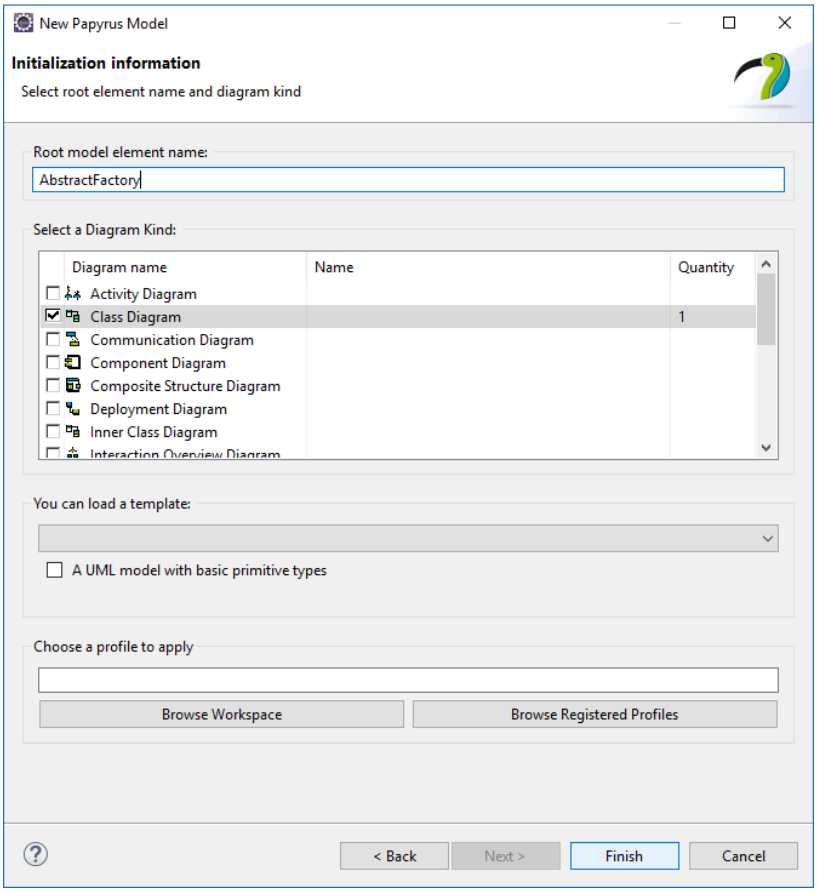

Рисунок 9

Рисунок 10

3. Для того чтобы использовать типы данных языка Java в диаграмме классов, необходимо подключить к проекту диаграмм Coposite библиотеку JavaPrimitiveTypes (рисунок 11–13).

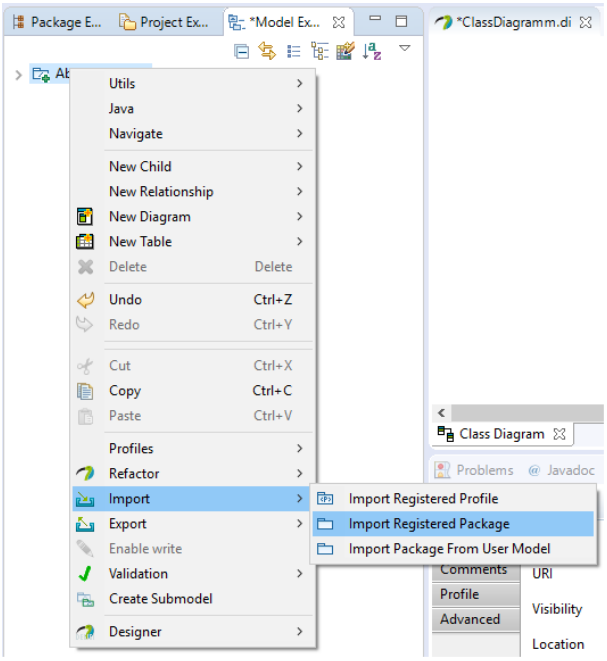

Рисунок 11

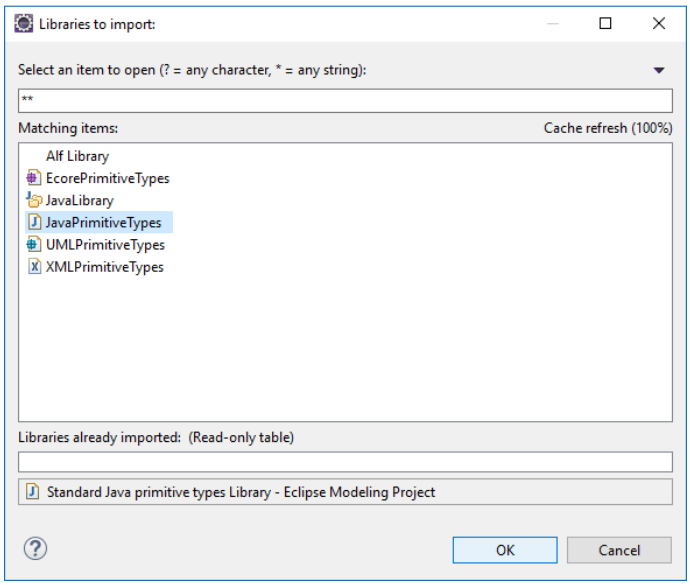

Рисунок 12

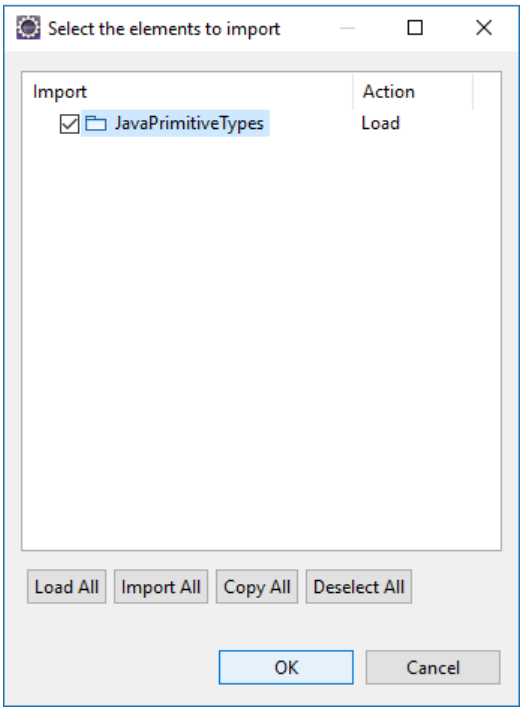

Рисунок 13

4. Разработать в проектировщике UML-диаграмм Papyrus диаграмму классов шаблона проектирования «Компоновщик», как показано на рисунке 14.

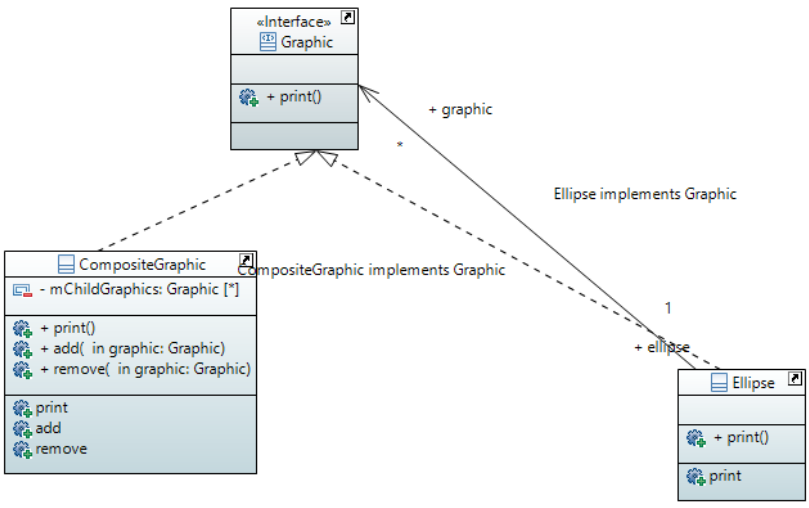

Рисунок 14

5. Для того чтобы сгенерировать Java-код из полученной диаграммы классов, необходимо подключить к проекту диаграмм Coposite профиль Java (рисунок 15–16).

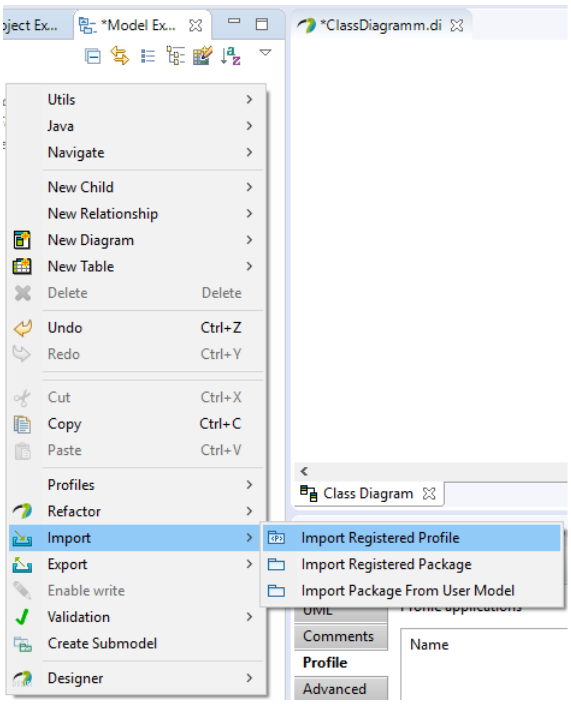

Рисунок 15

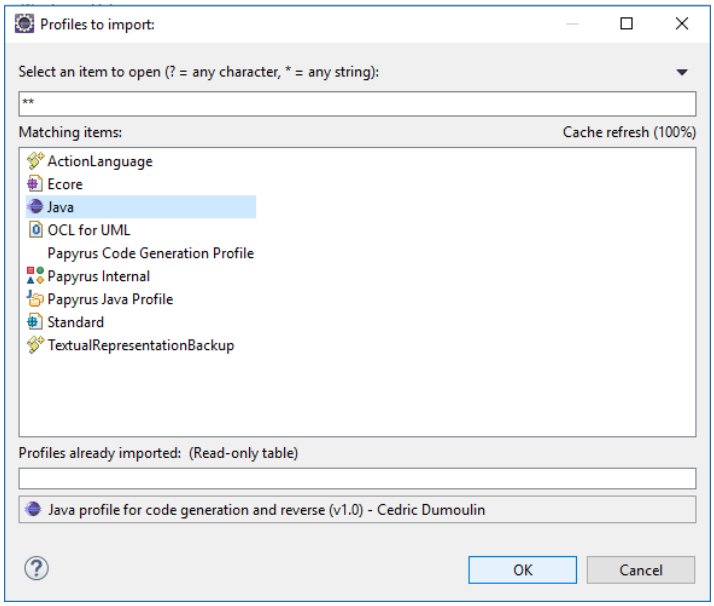

Рисунок 16

6. В свойствах пакета OSCoposite из диаграммы классов добавить профиль приложения Java (рисунки 17–19) и стереотип пакета JavaPackage_ (рисунки 20–21).

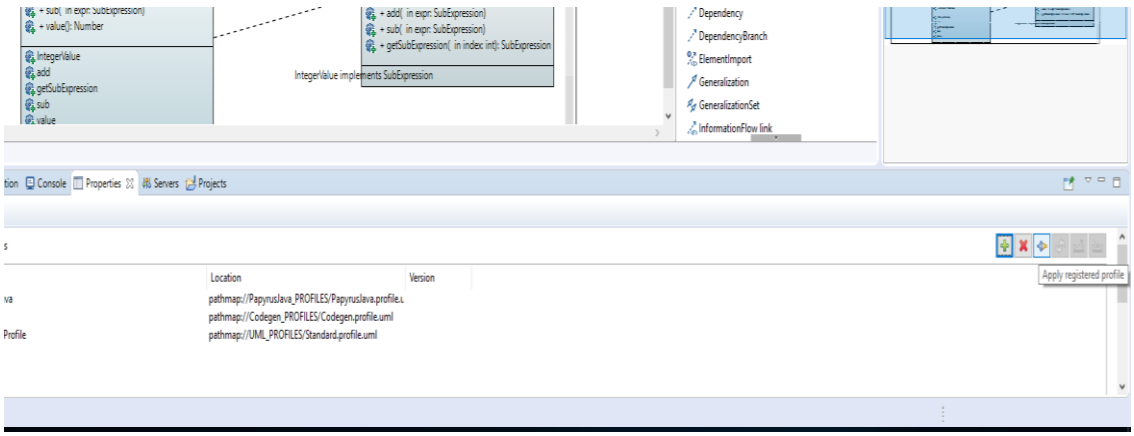

Рисунок 17

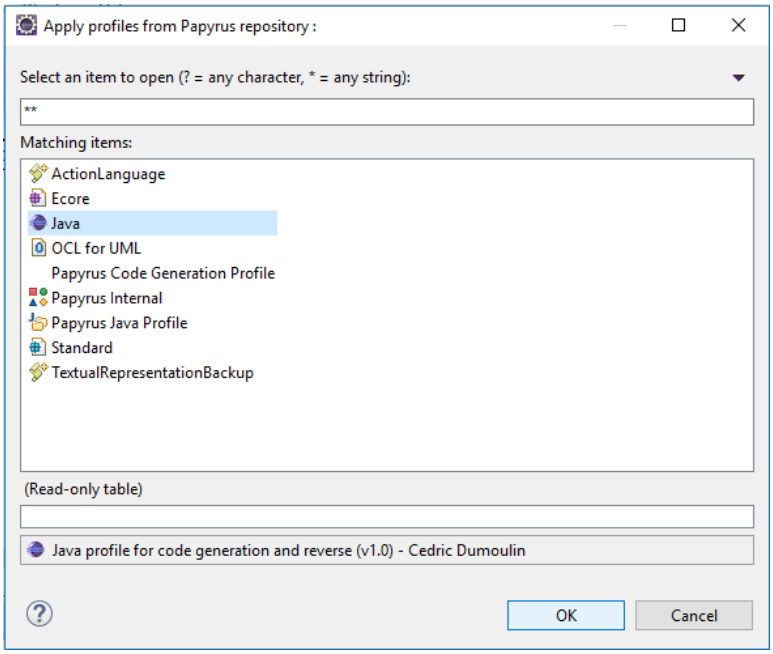

Рисунок 18

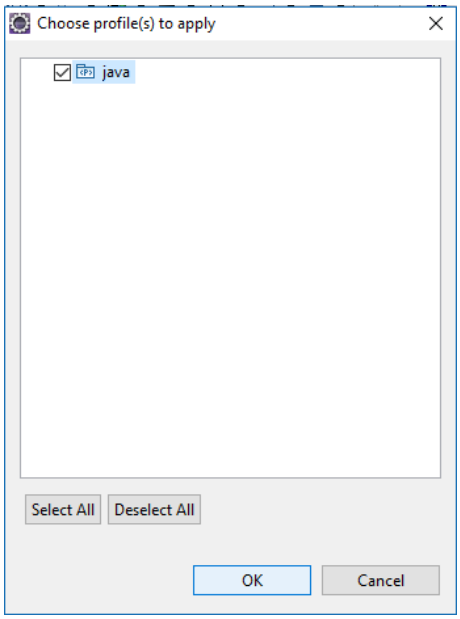

Рисунок 19

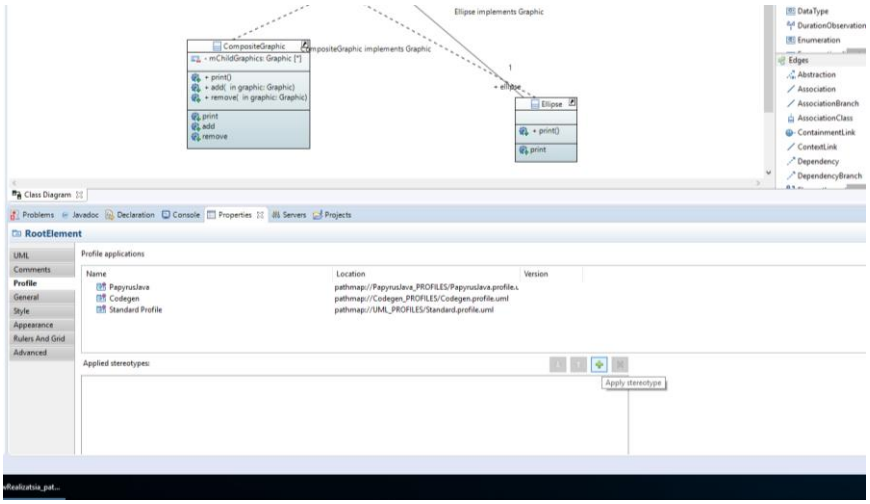

Рисунок 20

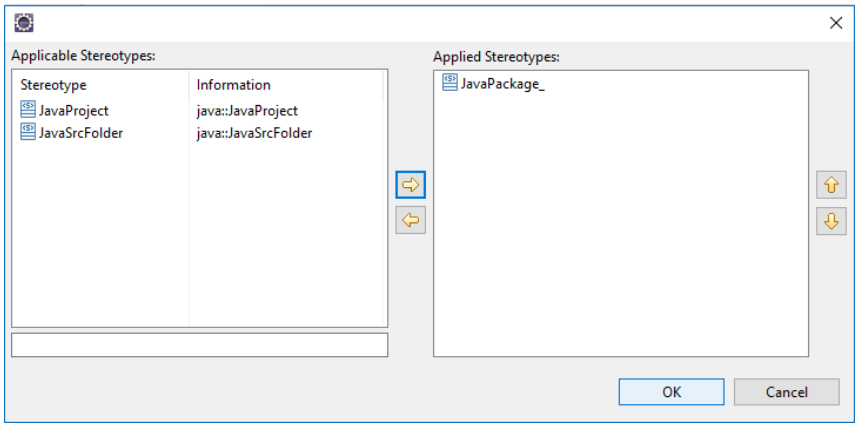

Рисунок 21

7. Сгенерировать Java-код из диаграммы классов. Для этого необходимо нажать правой кнопкой мыши на диаграмму классов и выбрать Generate Java Code из выпадающего меню (рисунок 22). В результате сгенерированные файлы с классами и интерфейсами паттерна будут находиться в папке src в пакете OSCoposite (рисунок 23).

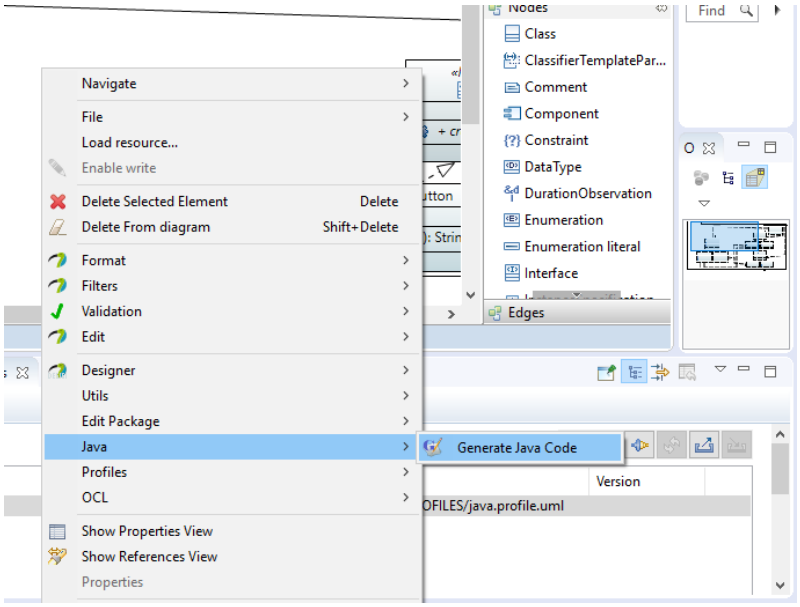

Рисунок 22

Рисунок 23

8. Необходима корректировка сгенерированного кода абстрактного интерфейса Graphic (рисунок 24) и классов-наследников CompositeGraphic, Ellipse (рисунки 25, 26). Ellipse играет роль листьев в древовидной системе, а CompositeGraphic группирует все листья в единую систему.

```
1
2  public interface Graphic {
3      public void print();
4  }
```

Рисунок 24

```
public class Ellipse implements Graphic {
    public void print() {
        System.out.println("Ellipse");
    }
}
```

Рисунок 25

```
import java.util.List;
import java.util.ArrayList;

public class CompositeGraphic implements Graphic {
    private List<Graphic> mChildGraphics = new ArrayList<Graphic>();

    //Prints the graphic.
    public void print() {
        for (Graphic graphic : mChildGraphics) {
            graphic.print();
        }
    }

    //Adds the graphic to the composition.
    public void add(Graphic graphic) {
        mChildGraphics.add(graphic);
    }

    //Removes the graphic from the composition.
    public void remove(Graphic graphic) {
        mChildGraphics.remove(graphic);
    }
}
```

Рисунок 26

9. Создайте в проекте класс ComposieExaple для тестирования шаблона проектирования «Компоновщик» (рисунки 27-29).

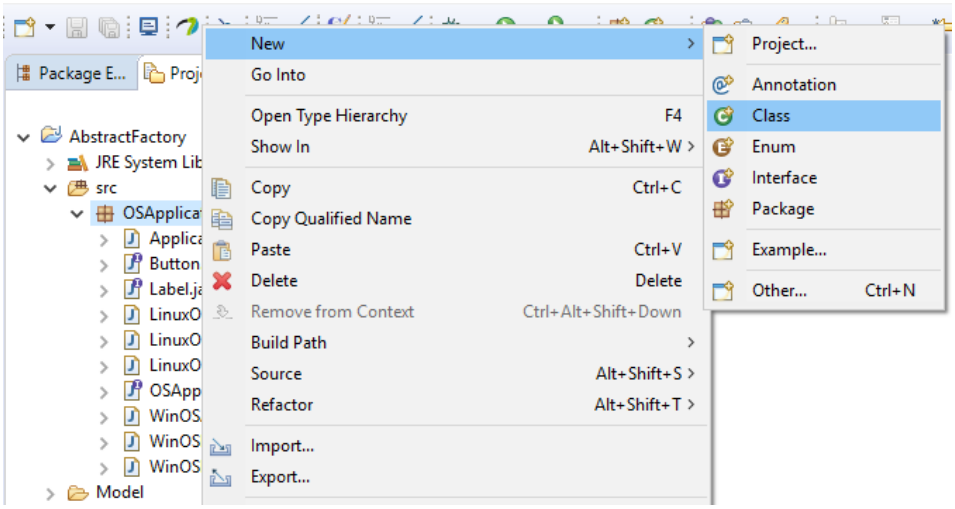

Рисунок 27

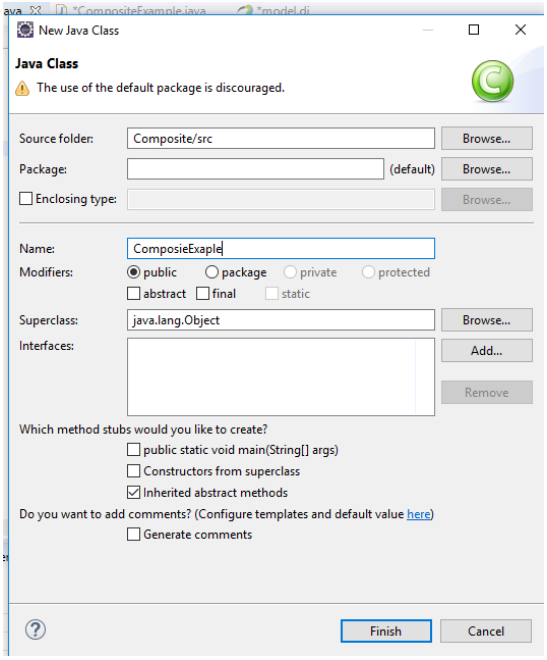

Рисунок 28

```
public class CompositeExample {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
            Ellipse ellipse1 = new Ellipse();
            Ellipse ellipse2 = new Ellipse();
            Ellipse ellipse3 = new Ellipse();
            Ellipse ellipse4 = new Ellipse();

            //Initialize three composite graphics
            CompositeGraphic graphic = new CompositeGraphic();
            CompositeGraphic graphic1 = new CompositeGraphic();
            CompositeGraphic graphic2 = new CompositeGraphic();

            //Composes the graphics
            graphic1.add(ellipse1);
            graphic1.add(ellipse2);
            graphic1.add(ellipse3);

            graphic2.add(ellipse4);

            graphic.add(graphic1);
            graphic.add(graphic2);

            //Prints the complete graphic (four times the string "Ellipse").
            graphic.print();
    }

}
```

Рисунок 29

10. Для компиляции класса ComposieExaple нажмите кнопку Run  , чтобы запустить тестирование программы. В окне Console можно увидеть результаты тестирования (рисунок 30).

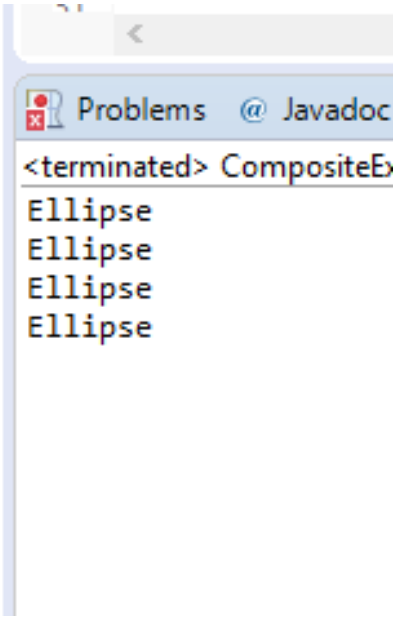
Problems
@ Javadoc
<terminated> CompositeEx
Ellipse
Ellipse
Ellipse
Ellipse

Рисунок 30

3 Содержание отчета по лабораторной работе

В сводный отчет по лабораторным работам в качестве одного из разделов или подразделов включаются скриншоты, показывающие проектирование и реализацию паттерна проектирования «Компоновщик».

4 Индивидуальные задания

1. Есть три типа героя – король, воин и маг. У каждого своя атака, тип оружия и наносимый урон. Каждый игрок может собирать свою армию, на основе 3 типов героев. Реализовать программный продукт, позволяющий формировать армию. При разработке использовать поведенческий шаблон проектирования «Компоновщик».
2. Существуют различные легковые машины, которые используют разные источники энергии: электричество, бензин, газ. Есть гибридные автомобили. Каждый автосалон продает различные автомобили. Реализовать программный продукт, позволяющий каждому автосалону составить сформировать список автомобилей с разными источниками энергии. При разработке использовать поведенческий шаблон проектирования «Компоновщик».
3. Есть различные продукты, каждый продукт имеет марку, название и цену. Продукты продаются в магазинах. Реализовать программный продукт, позволяющий добавлять в магазин различные продукты. При разработке использовать поведенческий шаблон проектирования «Компоновщик».
4. Для того, что бы пойти в школу, ученики собирают портфель. В каждом портфеле должны быть дневник, пенал,

тетради и учебники. Реализовать программный продукт, позволяющий сформировать портфель ученика. При разработке использовать поведенческий шаблон проектирования «Компоновщик».

5. В офисе работают люди. У каждого человека есть свои трудовые обязанности. Реализовать программный продукт, который формирует работников в офисе. При разработке использовать поведенческий шаблон проектирования «Компоновщик».
6. В каждой квартире есть мебель. Каждая мебель имеет имя и занимает определенную площадь. Реализовать программный продукт, который позволяет разместить мебель в комнате. При разработке использовать поведенческий шаблон проектирования «Компоновщик».
7. Рабочий стол в компьютере состоит из иконок. Каждая иконка имеет название и размер, который она занимает на жестком диске. Реализовать программный продукт, который позволит сформировать иконки на рабочем столе. При разработке использовать поведенческий шаблон проектирования «Компоновщик».

5 Вопросы к защите лабораторной работы

1. Что такое шаблон проектирования?
2. Какую структуру представляет шаблон «Компоновщик»?
3. Возможно ли предоставить единообразный доступ к листовым и к составным элементам древовидной структуры?
4. В каком случае нам необходимо наследовать интерфейс доступа?
5. Как возможно обеспечить доступ к составным элементам древовидной структуры?
6. Как возможно обеспечить доступ к листовым элементам древовидной структуры?
7. Какое количество листовых элементов может быть в древовидной структуре?