*Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования*

***«Московский государственный технический университет имени Н.Э. Баумана»   
(МГТУ им. Н.Э. Баумана)***

***Кафедра ИУ3, 3 курс, 6 семестр.***

**Отчёт**

**по лабораторной работе №2**

# “[Создание фигуры диаграммы на JavaFX](http://www.agentlab.ru/confluence/pages/viewpage.action?pageId=54001678)

# [с использованием Xtend2](http://www.agentlab.ru/confluence/pages/viewpage.action?pageId=54001678)”

**по курсу**

**“Разработка программного обеспечения”**

Выполнил:

Шалаев И.В.

Группа:

ИУ3-61

Проверил:

Иванов А.М.

**Москва, 2014**

# Цель работы

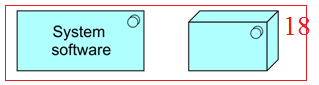
Цель работы — формирование у студентов ряда навыков:

* Использование передовых технологий разработки GUI толстых клиентов
* Использование DSL языков на примере Xtend2
* Основы работы в системе контроля версий Git

# Задание

Создать Eclipse-проект с векторной фигуркой  в соответствии со своим вариантом задания и требованиями, добавить его в репозитарий системы контроля версий.

Вариант 18:



# Настройка инструментов

Предварительные требования:

* Eclipse с установленным e(fx)clipse с предыдущей лабораторной работы

Последовательность настройки:

1. Создать аккаунт на Github и выслать логин на [amivanoff@gmail.com](mailto:amivanoff@gmail.com) для предоставления прав на запись

2. Настроить подключение Eclipse к Github

* Использование руководство [Настройка Eclipse IDE для доступа к репозитарию GitHub по протоколу HTTP](http://www.agentlab.ru/confluence/pages/viewpage.action?pageId=41058472)
  + Указать другой адрес репозитария, скопировав его со страницы <https://github.com/agentlab/jfxeditor>

3. Установить в Eclipse поддержку языка Xtend2

* См. <https://www.eclipse.org/xtend/download.html>
  + там выбрать "Latest Release"

Листинг программы:

BaseFigure.xtend:

**package** ru.agentlab.jfxed.figures

**import** javafx.scene.Node

**import** javafx.scene.layout.Pane

**import** javafx.scene.input.MouseEvent

**class** BaseFigure {

**protected** Pane root

**var** **double** sceneStartX = 0

**var** **double** sceneStartY = 0

**var** **double** figureStartTranslateX = 0

**var** **double** figureStartTranslateY = 0

**var** Node node = **null**;

**def** **public** Pane getRoot() {

**return** root

}

**def** **void** initHandlers() {

root.onMousePressed = [ MouseEvent event |

*println*("setOnMousePressed")

sceneStartX = event.getSceneX()

sceneStartY = event.getSceneY()

node = event.source **as** Node

figureStartTranslateX = node.translateX

figureStartTranslateY = node.translateY

]

root.onMouseDragged = [ MouseEvent event |

*println*("setOnMouseDragged");

*println*(event.sceneX + " " + event.sceneY);

root.translateX = figureStartTranslateX + event.sceneX - sceneStartX

root.translateY = figureStartTranslateY + event.sceneY - sceneStartY

]

}

}

BootFigures.java:

**package** ru.agentlab.jfxed.figures;

**import** ru.agentlab.jfxed.figures.clazz.LineFig;

**import** ru.agentlab.jfxed.figures.clazz.Rectangle;

**import** ru.agentlab.jfxed.figures.clazz.Coub;

**import** javafx.application.Application;

**import** javafx.scene.Scene;

**import** javafx.scene.layout.AnchorPane;

**import** javafx.stage.Stage;

**public** **class** BootFigures **extends** Application {

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

*launch*(args);

}

@Override

**public** **void** start(Stage primaryStage) **throws** Exception {

primaryStage.setTitle("System Software");

AnchorPane root = **new** AnchorPane();

**final** Rectangle rect = **new** Rectangle();

**final** Coub coub = **new** Coub();

root.getChildren().add(rect.getRoot());

root.getChildren().add(coub.getRoot());

Scene scene = **new** Scene(root, 400, 150);

primaryStage.setScene(scene);

primaryStage.show();

}

}

Coub.xtend:

**package** ru.agentlab.jfxed.figures.clazz

**import** javafx.scene.layout.VBox

**import** javafx.scene.paint.Color;

**import** javafx.scene.shape.Polygon

**import** javafx.scene.shape.Polyline

**import** javafx.scene.Group;

**import** ru.agentlab.jfxed.figures.BaseFigure

**import** javafx.scene.shape.Path

**import** javafx.scene.shape.ArcTo

**import** javafx.scene.shape.LineTo

**import** javafx.scene.shape.MoveTo

**class** Coub **extends** BaseFigure {

Path path

ArcTo arcto

ArcTo arcto1

ArcTo arcto2

LineTo line1

LineTo line2

MoveTo move1

MoveTo move2

**new** (){

root = **new** VBox

path = **new** Path()

arcto = **new** ArcTo()

arcto1 = **new** ArcTo()

arcto2 = **new** ArcTo()

line1 = **new** LineTo()

line2 = **new** LineTo()

move1 = **new** MoveTo()

move2 = **new** MoveTo()

**var** pligon = **new** Polygon()

pligon.getPoints().addAll(#[20.0, 0.0, 240.0, 00.0, 240.0, 100.0, 220.0, 120.0, 00.0, 120.0, 0.0, 20.0])

pligon.setFill(Color.*CYAN*)

pligon.setStroke(Color.*BLACK*)

pligon.setStrokeWidth(1)

**var** line = **new** Polyline(#[0.0, 20.0, 220.0, 20.0, 240.0, 0.0, 220.0, 20.0, 220.0, 120.0])

move1.setX(190)

move1.setY(35)

arcto.setX(200)

arcto.setY(45)

arcto.setRadiusX(7)

arcto.setRadiusY(7)

arcto1.setX(190)

arcto1.setY(35)

arcto1.setRadiusX(7)

arcto1.setRadiusY(7)

move2.setX(200)

move2.setY(45)

line1.setX(205)

line1.setY(40)

arcto2.setX(195)

arcto2.setY(30)

arcto2.setRadiusX(7)

arcto2.setRadiusY(7)

line2.setX(190)

line2.setY(35)

path.getElements().add(move1)

path.getElements().add(arcto)

path.getElements().add(arcto1)

path.getElements().add(move2)

path.getElements().add(line1)

path.getElements().add(arcto2)

path.getElements().add(line2)

**var** group = **new** Group(pligon,line,path)

root.children += group

root.translateX = 260

root.translateY = 5

initHandlers()

}

}

Line.xtend:

**package** ru.agentlab.jfxed.figures.clazz

**import** javafx.scene.layout.VBox

**import** ru.agentlab.jfxed.figures.BaseFigure

**import** javafx.scene.shape.Path

**import** javafx.scene.shape.ArcTo

**import** javafx.scene.shape.LineTo

**import** javafx.scene.shape.MoveTo

**class** LineFig **extends** BaseFigure {

Path path

ArcTo arcto

ArcTo arcto1

ArcTo arcto2

LineTo line1

LineTo line2

MoveTo move1

MoveTo move2

**new** (){

root = **new** VBox

path = **new** Path()

arcto = **new** ArcTo()

arcto1 = **new** ArcTo()

arcto2 = **new** ArcTo()

line1 = **new** LineTo()

line2 = **new** LineTo()

move1 = **new** MoveTo()

move2 = **new** MoveTo()

move1.setX(0)

move1.setY(15)

arcto.setX(10)

arcto.setY(25)

arcto.setRadiusX(7)

arcto.setRadiusY(7)

arcto1.setX(0)

arcto1.setY(15)

arcto1.setRadiusX(7)

arcto1.setRadiusY(7)

move2.setX(10)

move2.setY(25)

line1.setX(15)

line1.setY(20)

arcto2.setX(5)

arcto2.setY(10)

arcto2.setRadiusX(7)

arcto2.setRadiusY(7)

line2.setX(0)

line2.setY(15)

path.getElements().add(move1)

path.getElements().add(arcto)

path.getElements().add(arcto1)

path.getElements().add(move2)

path.getElements().add(line1)

path.getElements().add(arcto2)

path.getElements().add(line2)

root.translateX = 125

root.translateY = -10

root.children += path

}

}

Rectangle.xtend:

**package** ru.agentlab.jfxed.figures.clazz

**import** javafx.scene.layout.VBox

**import** javafx.scene.text.Text

**import** ru.agentlab.jfxed.figures.BaseFigure

**class** Rectangle **extends** BaseFigure {

LineFig line1 = **new** LineFig()

**new** (){

root = **new** VBox

**val** s = Rectangle.getResource("Style.css").toExternalForm()

root.stylesheets += s

root.styleClass += "ClassFigure-Rect";

root.styleClass += "ClassFigure-Text";

root.translateX = 5

root.translateY = 5

root.getChildren().add(line1.getRoot());

root.children += **new** Text => [

text = " System\nsoftware\n"

]

initHandlers()

}

}

Style.css:

.ClassFigure-Rect{

-fx-background-color: CYAN;

-fx-border-color: black;

-fx-stroke-type: inside;

-fx-arc-height: 5;

-fx-arc-width: 5;

-fx-padding: 15 80;

}

.ClassFigure-Text{

-fx-font: bold 14pt *"Arial"*;

-fx-stroke-type: inside;

-fx-stroke-width: 0;

}

Результат компиляции:

