*Московский Государственный Технический Университет им. Н.Э. Баумана*

*Факультет «Информатика и системы управления»*

*Кафедра ИУ-3*

***Отчёт по лабораторной работе №5***

*по курсу*

***«Разработка ПО»***

*Выполнил: Соловьев Д.В.*

*Проверил: Иванов А.М.*

*Москва, 2014*

## Задание

Создайте расширение для Eclipse используя фигуру полученную в процессе лабораторной работы №2.

1. Чтобы познакомиться с технологией выполните указания обучающего материала

* [Extending the Eclipse IDE - Plug-in development](http://www.vogella.com/tutorials/EclipsePlugIn/article.html)

2. Откройте Eclipse и обновите локальную копию репозитария

* В перспективе Git, надо сделать Pull

3. Переключитесь на стандартную целевую платформу Eclipse

* Настройки Eclipse > Plug-in Development > Target Platform, установить **Running Platform**

4. Импортируйте в Workspace проекты:

* **ru.agentlab.jfxed**
* **ru.agentlab.jfxed.figures**
* **ru.agentlab.jfxed.figures.class**

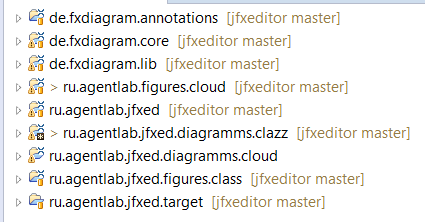
4. Запустите редактор

* Конфигурация запуска **Eclipse Jfxed**

## Цель задания

Цель задания: познакомит студентов с технологией расширения Eclipse своими плагинами.

## Реализация



*Рис.1 – Структура папок*

*Файл ClassFigure.xtend (для рисования фигуры)*

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

*import javafx.scene.text.Text*

*import ru.agentlab.jfxed.IFigure*

*import javafx.scene.shape.ArcTo*

*import javafx.scene.shape.Path*

*import javafx.scene.shape.MoveTo*

*import de.fxdiagram.core.XNode*

*import javafx.scene.shape.LineTo*

*import javafx.scene.Group*

*import javafx.scene.layout.HBox*

*import javafx.scene.layout.VBox*

*class ClassFigure extends XNode implements IFigure {*

*private Path path*

*val nameTextBox = new Text*

*var String name*

*new(String name) {*

*super(name)*

*this.name = name*

*}*

*new() {*

*super(" Send \n Shipping list")*

*name = " Send \n Shipping list"*

*path = new Path => [*

*styleClass += "ClassFigure"*

*elements += new MoveTo(-30, 0)*

*// Create cloud.*

*elements += new ArcTo(1, 1, 0, 0, -15, true, true)*

*elements += new ArcTo(1, 1, 0, 50, -45, true, true)*

*elements += new ArcTo(1, 1, 0, 110, -50, true, true)*

*elements += new ArcTo(1, 1, 0, 150, -40, true, true)*

*elements += new ArcTo(1, 1, 0, 180, -20, true, true)*

*elements += new ArcTo(1, 1, 0, 210, 5, true, true)*

*elements += new ArcTo(1, 1, 0, 160, 25, true, true)*

*elements += new ArcTo(1, 1, 0, 110, 45, true, true)*

*elements += new ArcTo(1, 1, 0, 60, 50, true, true)*

*elements += new ArcTo(1, 1, 0, 20, 30, true, true)*

*elements += new ArcTo(1, 1, 0, -30, 0, true, true)*

*// Create top arrow.*

*elements += new MoveTo(49, -107)*

*elements += new LineTo(65, -75)*

*elements += new MoveTo(65, -90)*

*elements += new LineTo(65, -75)*

*elements += new MoveTo(55, -84)*

*elements += new LineTo(65, -75)*

*// Create bottom line.*

*elements += new MoveTo(134, 63)*

*elements += new LineTo(145, 85)*

*]*

*node = new VBox => [*

*children += new Group => [*

*children += path*

*children += nameTextBox => [*

*text = name*

*styleClass += "ClassFigure-Text"*

*]*

*]*

*]*

*}*

*def setName(String name) {*

*nameTextBox.text = name*

*}*

*def getName() {*

*nameTextBox.text*

*}*

*override getRoot() {*

*return this*

*}*

*}*

*Файл ClazzDiagram.xtend*

*package ru.agentlab.jfxed.diagramms.cloud*

*import com.hp.hpl.jena.ontology.OntModel*

*import com.hp.hpl.jena.query.QueryExecutionFactory*

*import com.hp.hpl.jena.query.QueryFactory*

*import com.hp.hpl.jena.query.QuerySolution*

*import com.hp.hpl.jena.rdf.model.Resource*

*import de.fxdiagram.core.XDiagram*

*import ru.agentlab.jfxed.IDiagram*

*import ru.agentlab.jfxed.figures.clouds.ClassFigure*

*public class ClazzDiagram implements IDiagram {*

*static String SOURCE = "http://www.agentlab.ru/jfxed/onto/cloud"*

*static String NS = SOURCE + "#"*

*override createJfx(OntModel jenaModel, XDiagram jfxDiagram) {*

*val queryString ='''*

*PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>*

*select ?uri*

*where {*

*?uri rdf:type <«SOURCE»#ConceptClass>*

*}*

*'''*

*val query = QueryFactory.create(queryString)*

*// Execute the query and obtain results*

*val qe = QueryExecutionFactory.create(query, jenaModel)*

*val results = qe.execSelect()*

*for ( ; results.hasNext() ; )*

*{*

*val QuerySolution soln = results.nextSolution()*

*val Resource x = soln.getResource("uri") // Get a result variable by name.*

*val target = new ClassFigure() => [*

*layoutX = 180*

*layoutY = 180*

*name = x.localName*

*]*

*jfxDiagram => [*

*nodes += target*

*]*

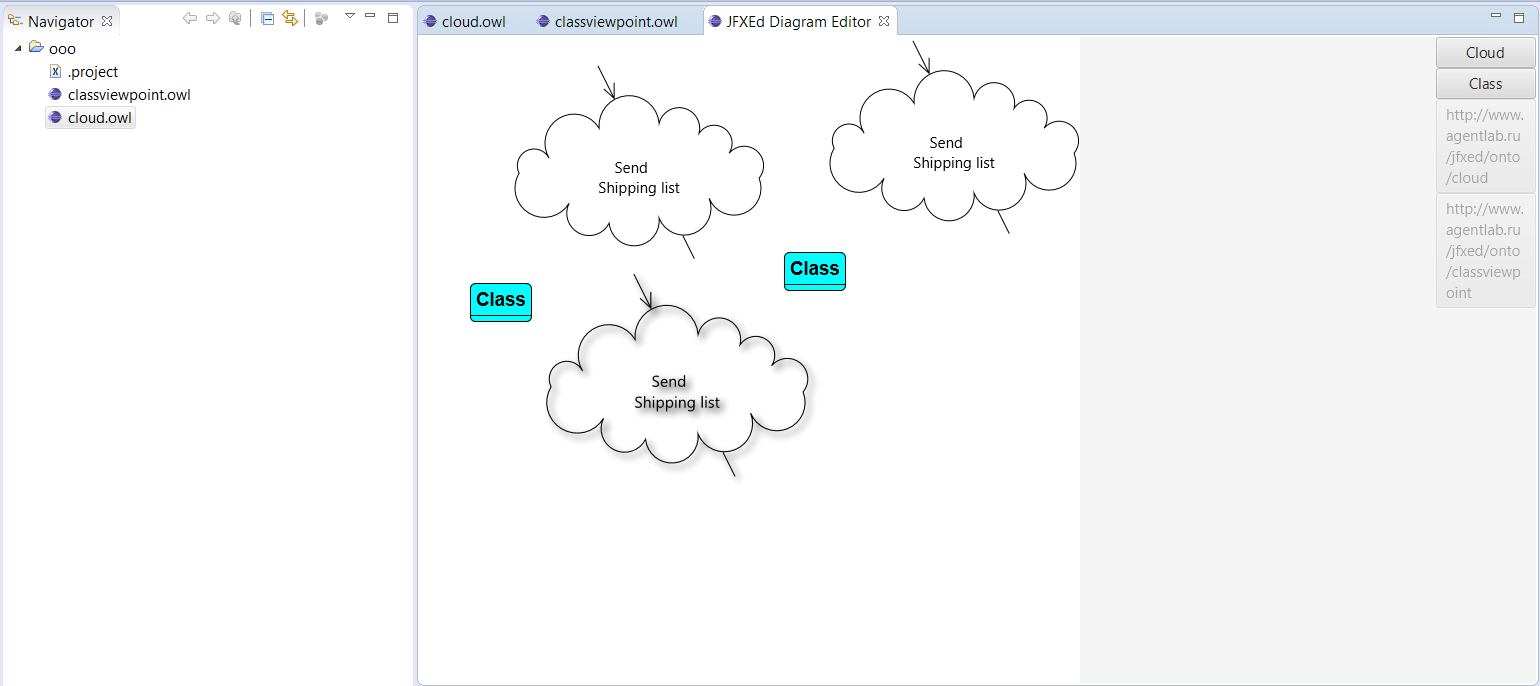
*}*

*qe.close()*

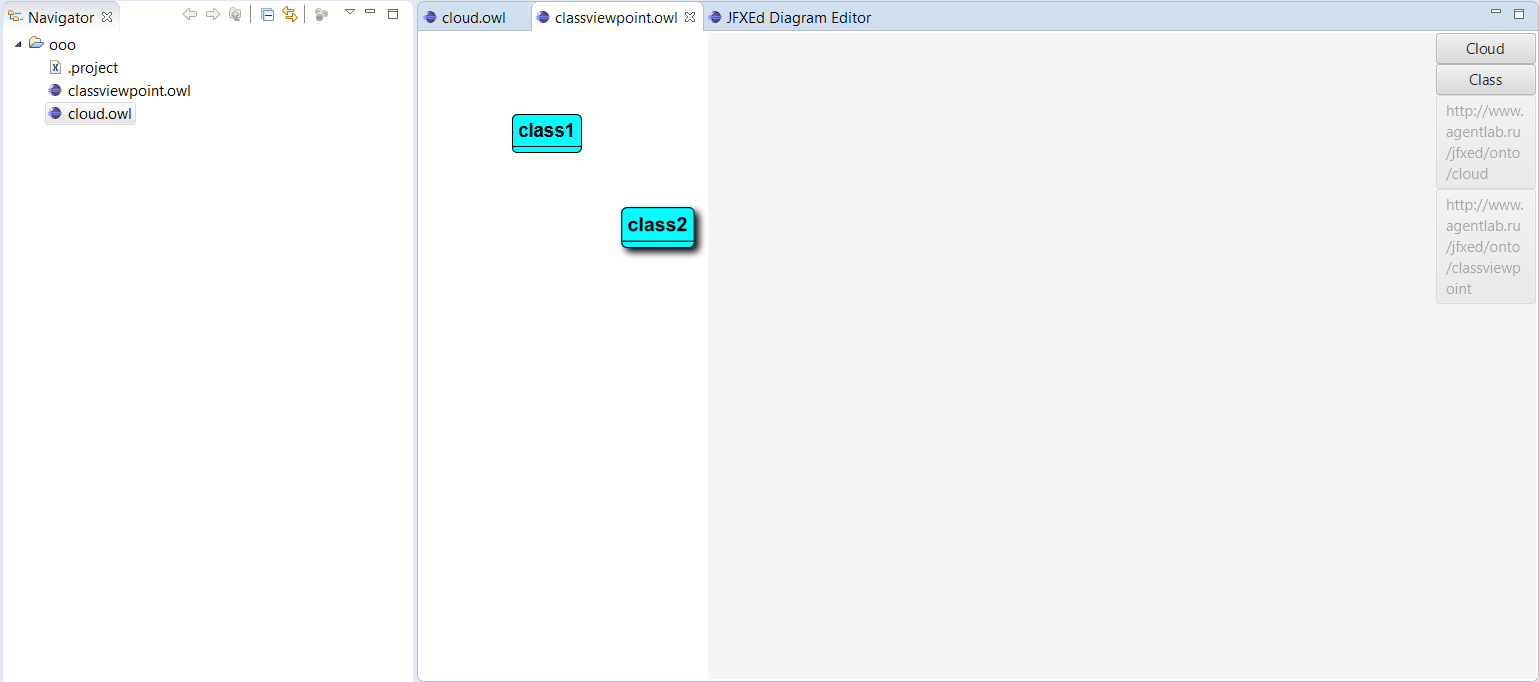
*}*

*}*

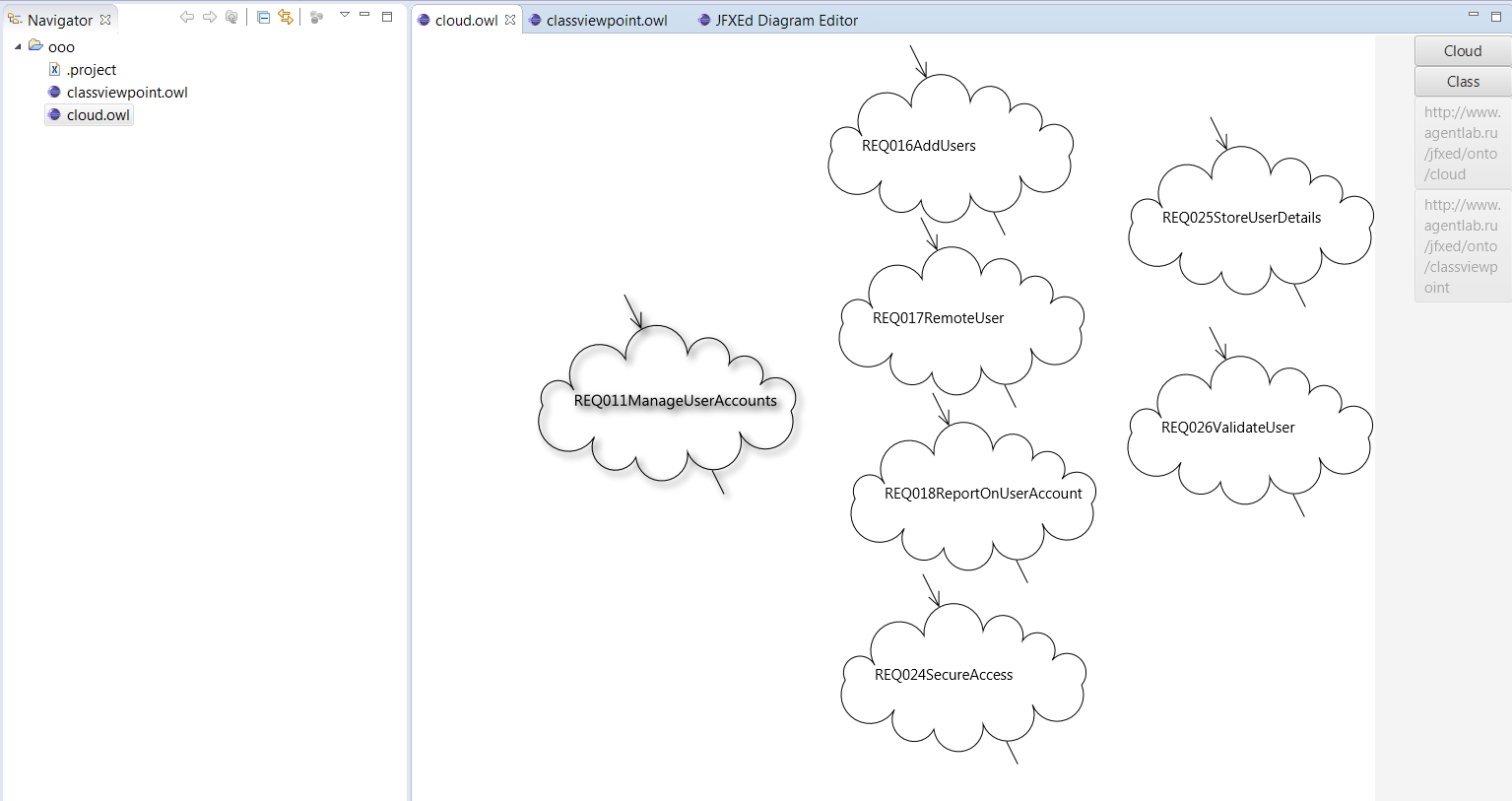
## Тестирование



*Рис.2 – Рисование отдельных фигурок*

**

*Рис.3 – Рисование примера из двух фигурок (Class1 и Class2)*

**

*Рис.4 – Рисование индивидуального примера с помощью реализованной фигуры, соответствующего диаграмме*