**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

|  |  |
| --- | --- |
| Подп. и дата |  |
| Инв. № дубл. |  |
| Взам. инв. № |  |
| Подп. и дата |  |
| Инв. № подл |  |

СОГЛАСОВАНО  
Профессор департамента программной инженерии, кандидат технических наук  
  
\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.М. Гринкруг  
«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

УТВЕРЖДАЮ  
Академический руководитель образовательной программы «Программная инженерия»  
  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  
«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

**ПРОГРАММНЫЙ КОМПОНЕНТ OPENGLVIEWER БИБЛИОТЕКИ JAVABEANS-КОМПОНЕНТ ДЛЯ 3D-ГРАФИКИ**

**Пояснительная записка**

**ЛИСТ УТВЕРЖДЕНИЯ**

**RU.17701729.04.01-01 81 01-1-ЛУ**

**Исполнитель**

Студент группы БПИ163

\_\_\_\_\_\_\_\_\_\_\_\_ / Д.Е. Крайнов /

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

**Москва 2019**

**УТВЕРЖДЕН**

**RU.17701729.04.01-01 81 01-1-ЛУ**

**ПРОГРАММНЫЙ КОМПОНЕНТ OPENGLVIEWER БИБЛИОТЕКИ JAVABEANS-КОМПОНЕНТ ДЛЯ 3D-ГРАФИКИ**

**Пояснительная записка**

**RU.17701729.04.01-01 81 01-1**

**Листов 33**

|  |  |
| --- | --- |
| Подп. и дата |  |
| Инв. № дубл. |  |
| Взам. инв. № |  |
| Подп. и дата |  |
| Инв. № подл |  |

**Москва 2019**

СОДЕРЖАНИЕ

[1. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 4](#_Toc6483337)

[1.1. Наименование программы 4](#_Toc6483338)

[1.2. Документ, на основании которого ведется разработка 4](#_Toc6483339)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 5](#_Toc6483340)

[2.1. Функциональное назначение 5](#_Toc6483341)

[2.2. Эксплуатационное назначение 5](#_Toc6483342)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 6](#_Toc6483343)

[3.1. Постановка задачи на разработку программы 6](#_Toc6483344)

[3.2. Описание алгоритма функционирования программы и применяемых методик 6](#_Toc6483345)

[3.2.1. Описание пользовательской последовательности действий при работе с программой 10](#_Toc6483346)

[3.3. Описание и обоснование выбора метода организации входных и выходных данных 10](#_Toc6483347)

[3.3.1. Описание и обоснование выбора метода организации входных данных 10](#_Toc6483348)

[3.3.2. Описание и обоснование выбора метода организации выходных данных 11](#_Toc6483349)

[3.4. Описание и обоснование выбора технических и программных средств 11](#_Toc6483350)

[4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 12](#_Toc6483351)

[4.1. Предполагаемая потребность 12](#_Toc6483352)

[4.2. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами 12](#_Toc6483353)

[5. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ 13](#_Toc6483354)

[Приложение 1 14](#_Toc6483355)

[Приложение 2 17](#_Toc6483356)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 33](#_Toc6483357)

# ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

* 1. Наименование программы  
     Наименование программы: «Программный компонент OpenGLViewer библиотеки JavaBeans-компонент для 3D-графики» («The OpenGLViewer Software Component for JavaBeans 3D Graphics Components Library»). Краткое название: “OpenGLViewer”.
  2. Документ, на основании которого ведется разработка  
     Приказ декана факультета компьютерных наук Национального исследовательского университета «Высшая школа экономики» № 2.3-02/1012-0 2 от 10.12.2018.

# НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

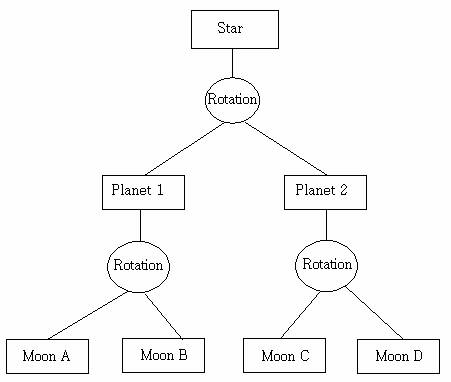
* 1. Функциональное назначение  
     Функциональным назначением компонента является предоставление возможности графического рендеринга моделей, представленных в виде графа сцены [1], с помощью технологии OpenGL [2] и графического процессора пользовательского компьютера.
  2. Эксплуатационное назначение  
     Программный компонент предлагается к эксплуатированию программистами в области компьютерной графики и разработчиками 3D-моделей в качестве инструмента для тестирования созданных графических моделей.

# ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

* 1. Постановка задачи на разработку программы  
     Компонент должен предоставить интерфейс для взаимодействия с ним другим JavaBeans-компонентам библиотеки, в составе которой он должен работать. В частности, компонент OpenGLViewer должен:
* получать объект 3D-сцены (граф сцены) для отображения;
* поддерживать собственный вид на переданную ему сцену;
* обеспечивать графическое отображение (рендеринг) этого вида на экране, используя возможности библиотеки OpenGL с помощью программного интерфейса Java OpenGL (JOGL) [3];
* предоставлять возможность параллельного наблюдения сцены с помощью нескольких экземпляров компонента;
* предоставлять возможность манипулирования сценой с помощью методов получения и модификации графа сцены и её настроек (геттеры и сеттеры).

## Описание алгоритма функционирования программы и применяемых методик

Отрисовка и представление сцен в компоненте OpenGLViewer происходит посредством хранения и обхода структуры данных под названием «граф сцены».



*Рисунок 1. Пример графа сцены*

Граф сцены – древовидная структура данных, вершинами которой являются графические объекты или трансформации над ними, и между ними имеются логические связи. К примеру, на рис. 1 изображен граф сцены простейшей планетной системы. Ко всем потомкам объекта звезды (“Star”) применяется трансформация поворота (“Rotation”), а к потомкам объектов планет – дополнительно еще и поворот вокруг этих самых планет.

Стандарта разработки графов сцены нет, компонент OpenGLViewer использует собственную упрощенную реализацию графа. Абстрактный класс **GLObject** является типом узла графа, одно из его полей – ссылка на список потомков данного объекта, тоже являющихся экземплярами класса GLObject. Экземпляры разных дочерних типов данного класса содержат информацию о состоянии объекта – его размере, цвете, текстуре и т.п.

Обход графа производится рекурсивно в глубину, к каждому объекту применяются постепенно накапливающиеся трансформации. Непосредственная отрисовка производится абстрактным методом draw() класса GLObject с помощью команд OpenGL, OpenGL Utility Library и OpenGL Utility Toolkit. Каждый компонент, представляющий графический примитив (простейший отображаемый на экране геометрический объект, из которого складываются модели), должен сделать реализацию данного метода. Для простоты отображения в классе GLObject определен метод drawTree(), производящий процесс отрисовки для самого объекта и для всех его потомков.

Подключение необходимых возможностей технологии OpenGL (свет, текстуры и т.д.), обработка события изменения размеров окна, хранение ссылки на граф сцены, а также постоянный вызов главного метода отрисовки раз в единицу времени (в данном случае это кадр), управление состоянием сцены (включение и остановка отрисовки, рендеринг отладочного текста и осей координат) – ответственность компонента **OpenGLViewer.**

Данный компонент также поддерживает собственный вид на сцену: среди прочих его полей присутствует угол поворота камеры и ее позиция в пространстве, которые можно менять через предоставляемый классом программный интерфейс.

В свою очередь, компонент **GLViewerCanvas** является обёрткой над OpenGLViewer и обрабатывает события, связанные с движением мыши, нажатием различных клавиш и т.д.

Примерная последовательность инициализации компонентов:

* Из различных примитивов-подклассов класса **GLObject** формируется граф сцены;
* Создаётся **OpenGLViewer**. Объект графа либо добавляется в OpenGLViewer при создании объекта последнего, либо устанавливается как свойство через сеттер впоследствии;
* Создаётся **GLViewerCanvas**. Экземпляр компонента **OpenGLViewer** либо добавляется в GLViewerCanvas при создании объекта последнего, либо устаналивается как свойство через сеттер впоследствии;
* GLViewerCanvas добавляется в любой компонент, способный отображать другие компоненты (например, javax.swing.JFrame).

Архитекура компонентов набора следует спецификации JavaBeans, которая в свою очередь является фактической реализацией парадигмы компонентно-ориентированного программирования в среде Java.

Компонентно-ориентированное программирование – парадигма программирования, основная идея которой – представление и разработка программ как набора компонентов - независимых модулей, направленных на выполнение одной определенной задачи. Каждый компонент компилируется и подключается к программному проекту отдельно от остальных.

Выделяется несколько основных преимуществ данного подхода [4].

Во-первых, код, написанный с использованием компонентов, проще разрабатывать. В силу ограниченности действий, выполняемых каждым компонентом, становится проще понимать, как добиться той функциональности, какая требуется в финальной программе. По этой же причине легко создавать новые вариации программ с дополнительными функциями путём замены или добавления компонентов.

Во-вторых, изолированность и заточенность компонентов под конкретное действие позволяет проще отслеживать ошибки, возникающие в результате работы.

Наконец, компоненты проще тестировать, так как сразу понятно, что должен и не должен делать компонент.

Спецификация JavaBeans – по сути, реализация данной компонентной модели. Bean – любой класс, написанный по нескольким правилам:

* в классе присутствует публичный конструктор без параметров;
* все поля – приватные, работа с ними осуществляется через публичные методы доступа (геттеры) и модификации (сеттеры);
* класс имплементирует интерфейс Serializable (должна присутствовать возможность преобразования экземпляра класса в поток байтов для сохранения в файловую систему, базу данных и т.п.);
* геттеры не принимают параметров, возвращают объект определенного типа, названы в стиле get\*(), где \* - название поля;
* сеттеры принимают параметр, ничего не возвращают, названы в стиле set\*(), где \* - название поля.

Подобная стандартизация классов нашла своё применение, в первую очередь, в приложениях, предназначенных для визуального построения программ. Повторное использование компонентов и возможность их конфигурации на лету, а также возможность использования событийной модели (оповещения при изменении свойств класса) – основные преимущества спецификации JavaBeans. В частности, виджеты библиотеки Swing для создания графических интерфейсов в Java являются «бинами».

Набор компонентов OpenGLViewer разрабатывается в соответствии с данной спецификацией, что позволит использовать компоненты набора – двух- и трехмерные объекты и сцены, состоящие из данных объектов, а также экран просмотра и вспомогательные компоненты – в визуальных редакторах интерфейсов, фактически – без необходимости написания какого-либо кода.

### Описание пользовательской последовательности действий при работе с программой

Ниже приводится краткая последовательность действий, выполняемых программой в процессе работы в автономном режиме:

* Пользователь запускает .jar-файл, содержащий программу;
* Программа создаёт окно, содержащее экземпляр компонента OpenGLViewer и вспомогательное меню для работы с компонентом, а также инициализирует пустой граф сцены;
* Пользователь может либо загрузить готовый граф из файла через меню, либо начать добавлять новые объекты в граф;
  + Во время добавления через меню каждого нового объекта пользователю будет предложено указать его начальные координаты, размер, цвет и другие параметры в зависимости от типа объекта;
* Пользователь может также настроить отображение осей координат (включено/выключено), отображение отладочного текста (включено/выключено) и состояние отображения сцены (рендеринг включен/выключен);
* Во время просмотра пользователь может пользоваться возможностью обзора сцены с помощью клавиш-стрелок, Shift и Ctrl для перемещения по всем осям координат и мышки для изменения угла обзора или приближения/отдаления сцены;
* После окончания работы со сценой пользователь может сохранить текущую сцену в файл (происходит сериализация графа сцены в формат json).

## Описание и обоснование выбора метода организации входных и выходных данных

### Описание и обоснование выбора метода организации входных данных

В качестве входных данных для компонента OpenGLViewer используется объект графа сцены (класс GLObject или производные от него). В случае работы набора компонентов как отдельного приложения (автономный режим) граф может либо создаваться пользователем в окне OpenGLTestFrame с помощью меню добавления объектов, либо как сериализованный в json объект GLObject (опция меню окна).

В случае подключения компонентов в сторонние библиотеки или приложения граф сцены задаётся в компоненте OpenGLViewer через геттеры и сеттеры.

### Описание и обоснование выбора метода организации выходных данных

В качестве выходных данных компонент OpenGLViewer выводит отрисованное изображение сцены в окно компонента.

Если пользователь работает с набором компонентов в режиме отдельного приложения, есть возможность сохранить созданную пользователем сцену в файл в формате json (сериализовать граф сцены). Данное действие можно выполнить с помощью меню компонента OpenGLTestFrame.

## Описание и обоснование выбора технических и программных средств

Языком разработки компонентов является Java, для работы с OpenGL используется библиотека Java OpenGL (JOGL), для сериализации компонентов – библиотека gson [6]. Версия OpenGL 2.1 выбрана в целях сохранения максимальной совместимости с как можно более широким кругом технических средств.

Были обозначены следующие минимальные требования к аппаратному обеспечению, необходимые для работы программы:

* + Минимальная тактовая частота процессора – 1 Ггц;
  + Минимальный объем ОЗУ – 512 Мб;
  + Минимальное свободное место на жёстком диске для хранения программы – 10 Мб;
  + Монитор с минимальным разрешением 800х600;
  + Видеокарта с минимальным размером видеопамяти 512 Мб;
  + Клавиатура и мышь.

Минимальные требования к программному обеспечению, необходимые для работы программы:

* ОС Windows XP или новее;
* Java SE Runtime Environment 8 или новее;
* Драйвер видеокарты с поддержкой OpenGL 2.1 или новее.

# ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

* 1. Предполагаемая потребностьНабор компонентов потенциально упростит просмотр, отладку и нахождение ошибок при построении 3D-моделей и таким образом может использоваться программистами в области компьютерной графики и 3D-моделистами.
  2. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогамиДанное решение является полностью бесплатным.

# ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. Understanding and Implementing Scene Graphs [Electronic resource]. // GameDev.net [Official website]. URL: <http://archive.gamedev.net/archive/reference/programming/features/scenegraph/index.html> (accessed: 14.04.2019)

2. OpenGL 2.1 Reference Pages [Electronic resource]. // The Khronos Group Inc [Official website]. URL: <https://www.khronos.org/registry/OpenGL-Refpages/gl2.1/> (accessed: 14.04.2019)

3. JOGL – Java Binding for the OpenGL API [Electronic resource]. // JogAmp.org - Java graphics, audio, media and processing libraries exposing OpenGL, OpenCL, OpenAL and OpenMAX [Official website]. URL: <http://jogamp.org/jogl/www/> (accessed: 14.04.2019)

4. Advantages of component-based development [Electronic resource]. // ResearchGate [Official website]. URL: <https://www.researchgate.net/figure/Advantages-of-component-based-development_tbl1_220588371> (accessed: 14.04.2019)

5. JavaBeans Spec [Electronic resource]. // Oracle [Official website]. URL: <https://www.oracle.com/technetwork/articles/javaee/spec-136004.html> (accessed: 14.04.2019)

6. A Java serialization/deserialization library to convert Java objects into JSON and back [Electronic resource]. // Github [Official website]. URL: <https://github.com/google/gson> (accessed: 14.04.2019)

7. gson/RuntimeTypeAdapterFactory.java [Electronic resource]. // Github [Official website]. URL: <https://github.com/google/gson/blob/master/extras/src/main/java/com/google/gson/typeadapters/RuntimeTypeAdapterFactory.java> (accessed: 14.04.2019)

# Приложение 1

**Описание и функциональное назначение классов**

**Пакет** ru.dansstuff.simpleopengl

|  |  |
| --- | --- |
| **Имя класса** | **Назначение** |
| Main | Статический класс, содержащий единственный метод main(), запускающий программу в автономном режиме. |

**Пакет** ru.dansstuff.simpleopengl.math

|  |  |
| --- | --- |
| **Имя класса** | **Назначение** |
| Vec3 | Класс, представляющий трехмерный числовой вектор. |
| Vec4 | Класс, представляющий четырехмерный числовой вектор. |

**Пакет** ru.dansstuff.simpleopengl.misc.gson

|  |  |
| --- | --- |
| **Имя класса** | **Назначение** |
| RuntimeTypeAdapterFactory | Класс-дополнение библиотеки gson для сериализации объектов [7] |

**Пакет** ru.dansstuff.simpleopengl.misc.helpers

|  |  |
| --- | --- |
| **Имя класса** | **Назначение** |
| ObjectCreationFrameFactory | Класс для создания оконных объектов конфигурации новых примитивов |
| SceneFileHelper | Статический класс для сериализации графов сцены |

**Пакет** ru.dansstuff.simpleopengl.objects

|  |  |
| --- | --- |
| **Имя класса** | **Назначение** |
| Box | Класс, представляющий примитив «Параллелепипед» |
| Cylinder | Класс, представляющий примитив «Цилиндр» |
| DirectionalLight | Класс, представляющий источник света |
| EmptyObject | Класс, представляющий пустой примитив для привязки сцены |
| GLObject | Базовый класс примитивов и вершин графа, от которого наследуются все классы-примитивы |
| Line | Класс, представляющий примитив «Линия» |
| OpenGLColor | Класс, представляющий цвет в палитре RGB |
| Sphere | Класс, представляющий примитив «Сфера» |
| Triangle | Класс, представляющий примитив «Треугольник» |

**Пакет** ru.dansstuff.simpleopengl.objects.windows

|  |  |
| --- | --- |
| **Имя класса** | **Назначение** |
| BoxFrame | Класс, представляющий окно создания примитива «Параллелепипед» |
| CylinderFrame | Класс, представляющий окно создания примитива «Цилиндр» |
| DirectionalLightFrame | Класс, представляющий окно создания источника света |
| EmptyObjectFrame | Класс, представляющий окно создания пустого примитива |
| LineFrame | Класс, представляющий окно создания примитива «Линия» |
| SphereFrame | Класс, представляющий окно создания примитива «Сфера» |
| TriangleFrame | Класс, представляющий окно создания примитива «Треугольник» |
| TypeBaseFrame | Абстрактный класс, от которого наследуются все классы пакета |

**Пакет** ru.dansstuff.simpleopengl.operations

|  |  |
| --- | --- |
| **Имя класса** | **Назначение** |
| OpenGLOperation | Интерфейс, представляющий трансформацию сцены или объекта |
| Rotation | Класс, представляющий трансформацию поворота |
| Translation | Класс, представляющий трансформацию сдвига |

**Пакет** ru.dansstuff.simpleopengl.viewer

|  |  |
| --- | --- |
| **Имя класса** | **Назначение** |
| GLViewerCanvas | Класс-обёртка над OpenGLViewer для привязки обработчиков событий |
| OpenGLViewer | Главный компонент рендеринга сцены |

**Пакет** ru.dansstuff.simpleopengl.viewer.listeners

|  |  |
| --- | --- |
| **Имя класса** | **Назначение** |
| OpenGLViewerPopupMenu | Контекстное меню, отображаемое при нажатии на окно OpenGLViewer правой клавишей мыши |
| OpenGLViewerKeyListener | Обработчик события нажатия на клавишу клавиатуры |
| OpenGLViewerMouseListener | Обработчик событий, связанных с движением или нажатием на различные клавиши мыши |

**Пакет** ru.dansstuff.simpleopengl.window

|  |  |
| --- | --- |
| **Имя класса** | **Назначение** |
| OpenGLTestFrame | Окно для работы набора компонентов в режиме отдельного приложения |

# Приложение 2

**Описание и функциональное назначение полей/свойств и методов классов**

**Примечание:** в списках методов не описываются методы доступа (геттеры) и модификации (сеттеры). Поля, снабжённые геттерами, описываются вместе с модификатором @G, поля, снабжённые сеттерами, описываются вместе с модификатором @S.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.**Main** | | | | | | | |
| **Поля** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | | Описание | |
| width | private static | | int | | | Длина окна OpenGLTestFrame. | |
| height | private static | | int | | | Ширина окна OpenGLTestFrame. | |
| **Методы** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | Параметры | | Описание |
| main | | public static | | void | String[] args | | Запускает программу. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.math.**Vec3** | | | | | | | |
| **Поля** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | | Описание | |
| @G @S x | private | | float | | | X-компонента вектора. | |
| @G @S y | private | | float | | | Y-компонента вектора. | |
| @G @S z | private | | float | | | Z-компонента вектора. | |
| **Методы** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | Параметры | | Описание |
| Vec3 | | public | | Vec3 | float x, float y, float z | | Конструктор класса. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.math.**Vec4** | | | | | | | |
| **Поля** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | | Описание | |
| @G @S x | private | | float | | | X-компонента вектора. | |
| @G @S y | private | | float | | | Y-компонента вектора. | |
| @G @S z | private | | float | | | Z-компонента вектора. | |
| @G @S w | private | | float | | | W-компонента вектора. | |
| **Методы** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | Параметры | | Описание |
| Vec4 | | public | | Vec4 | float x, float y, float z, float w | | Конструктор класса. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.misc.helpers.**ObjectCreationFrameFactory** | | | | | | | |
| **Поля** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | | Описание | |
| frameMap | private | | Map<Class, Class> | | | Отображение для получения объектов окон создания примитивов. | |
| **Методы** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | Параметры | | Описание |
| getFrame | | public static | | JFrame | Class clazz, OpenGLTestFrame parent | | Возвращает экземпляр окна создания примитива в зависимости от переданного класса примитива. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.misc.helpers.**SceneFileHelper** | | | | | | | |
| **Поля** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | | Описание | |
| rFactory | private static | | RuntimeTypeAdapterFactory<GLObject> | | | Экземпляр RuntimeTypeAdapterFactory для сериализации и десериализации полиморфных типов. | |
| gson | private static | | Gson | | | Сериализатор объектов. | |
| **Методы** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | Параметры | | Описание |
| readScene | | public static | | GLObject | File file | | Десериализует файл в объект графа сцены. |
| writeScene | | public static | | void | GLObject root, File file | | Сериализует объект графа сцены в указанный файл. |
| getSceneJson | | public static | | string | GLObject root | | Получает строковую репрезентацию сериализованного графа сцены |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.misc.objects.windows.**TypeBaseFrame**  Примечание: все производные классы данного класса, содержащиеся в пакете ru.dansstuff.simpleopengl.misc.objects.windows, не описываются в данном Приложении, поскольку в них присутствует только конфигурация расположения графических виджетов, связанных с полями тех примитивов, которые должны быть созданы. | | | | | | | |
| **Поля** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | | Описание | |
| @G @S parent | protected | | OpenGLTestFrame | | | Экземпляр окна OpenGLTestFrame для добавления объектов. | |
| **Методы** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | Параметры | | Описание |
| createObject | | protected abstract | | void |  | | Добавляет объект с созданной в окне конфигурацией в граф сцены. |
| getNum | | protected | | int | JTextField field | | Получает численный пользовательский ввод из графических виджетов, наследующихся от компонента JTextField. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.objects.**GLObject** | | | | | | | |
| **Поля** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | Описание | |
| @G @S id | | protected | | String | | Идентификатор объекта. | |
| @G @S children | | protected | | List<GLObject> | | Список потомков объекта. | |
| @G type | | protected | | String | | Тип объекта. | |
| @G @S textureFile | | protected | | String | | Путь к текстуре объекта в файловой системе. | |
| @G @S texture | | protected | | com.jogamp.opengl.util.texture.Texture | | Текстура объекта. | |
| @G frameClass | | protected | | Class | | Класс окна создания объекта. | |
| **Методы** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Параметры | | Описание |
| draw | public abstract | | void | | com.jogamp.opengl.GL2 gl | | Отрисовывает объект. |
| update | public abstract | | void | |  | | Обновляет состояние объекта. |
| resolveTexturesForTree | public | | void | | Map<String, Texture> textureMap | | Загружает текстуры из файлов и присваивает их всем объектам графа сцены, которые являются потомками данного объекта. |
| getTextureFromFile | private | | com.jogamp.opengl.util.texture.Texture | | File file | | Получает текстуру из файла |
| resolveTexture | public | | void | | Map<String, Texture> textureMap | | Назначает уже существующую текстуру данному объекту, либо получает ее из файла |
| addChild | public | | void | | GLObject child | | Добавляет потомка данного объекта в граф сцены |
| drawTree | public | | void | | com.jogamp.opengl.GL2 gl | | Отрисовывает граф сцены, начиная с данного объекта |
| getObjectsCount | public | | int | |  | | Возвращает количество объектов в графе сцены, начиная с данного объекта |
| getTreeAsList | public | | List<GLObject> | |  | | Возвращает репрезентацию графа сцены в виде одномерного списка |
| clear | public | | void | |  | | Очищает граф сцены |
| getObjectTypes | public static | | Set<Class> | |  | | Возвращает все зарегистрированные типы примитивов |

|  |  |  |  |
| --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.objects.**Box extends GLObject** | | | |
| **Поля** | | | |
| Имя | Модификатор доступа | Тип | Описание |
| @G @S center | private | Vec3 | Центр объекта. |
| @G @S length | private | float | Длина объекта. |
| @G @S color | private | OpenGLColor | Цвет объекта. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.objects.**Cylinder extends GLObject** | | | | |
| **Поля** | | | | |
| Имя | Модификатор доступа | | Тип | Описание |
| @G @S center | private | Vec3 | | Центр объекта. |
| @G @S radius | private | float | | Радиус объекта. |
| @G @S height | private | float | | Высота объекта. |
| @G @S color | private | OpenGLColor | | Цвет объекта. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.objects.**DirectionalLight extends GLObject** | | | |
| **Поля** | | | |
| Имя | Модификатор доступа | Тип | Описание |
| @G @S color | private | Vec4 | Цвет объекта (с альфа-каналом). |
| @G @S pos | private | Vec4 | Позиция объекта. |
| @G index | private | int | Цвет объекта. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.objects.**EmptyObject extends GLObject** | | | |
| **Поля** | | | |
| Имя | Модификатор доступа | Тип | Описание |
| @G @S center | private | Vec3 | Центр объекта. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.objects.**Line extends GLObject** | | | |
| **Поля** | | | |
| Имя | Модификатор доступа | Тип | Описание |
| @G @S p1 | private | Vec3 | Первая точка объекта. |
| @G @S p2 | private | Vec3 | Вторая точка объекта. |
| @G @S color | private | OpenGLColor | Цвет объекта. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.objects.**Sphere extends GLObject** | | | |
| **Поля** | | | |
| Имя | Модификатор доступа | Тип | Описание |
| @G @S center | private | Vec3 | Центр объекта. |
| @G @S radius | private | float | Радиус объекта. |
| @G @S color | private | OpenGLColor | Цвет объекта. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.objects.**Triangle extends GLObject** | | | |
| **Поля** | | | |
| Имя | Модификатор доступа | Тип | Описание |
| @G @S p1 | private | Vec3 | Первая точка объекта. |
| @G @S p2 | private | Vec3 | Вторая точка объекта. |
| @G @S p3 | private | Vec3 | Третья точка объекта. |
| @G @S color | private | OpenGLColor | Цвет объекта. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.objects.**OpenGLColor** | | | |
| **Поля** | | | |
| Имя | Модификатор доступа | Тип | Описание |
| @G @S r | private | float | r-компонента цвета. |
| @G @S g | private | float | g-компонента цвета. |
| @G @S b | private | float | b-компонента цвета. |
| COLORS | public static | OpenGLColor[] | Массив различных цветов. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Интерфейс** ru.dansstuff.simpleopengl.operations.**OpenGLOperation** | | | | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Параметры | Описание |
| doOperation | - | void | GLAutoDrawable drawable | Производит операцию трансформации над объектом. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.operations.**Rotation** | | | |
| **Поля** | | | |
| Имя | Модификатор доступа | Тип | Описание |
| @G @S x | private | float | x-компонента поворота. |
| @G @S y | private | float | y-компонента поворота. |
| @G @S z | private | float | z-компонента поворота. |
| @G @S angle | private | float | Угол поворота. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.operations.**Translation** | | | |
| **Поля** | | | |
| Имя | Модификатор доступа | Тип | Описание |
| @G @S x | private | float | x-компонента сдвига. |
| @G @S y | private | float | y-компонента сдвига. |
| @G @S z | private | float | z-компонента сдвига. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.viewer.listeners.**OpenGLViewerKeyListener extends KeyAdapter** | | | | | | | |
| **Поля** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | Описание | |
| viewer | | private | | OpenGLViewer | | Экземпляр OpenGLViewer, к которому нужно применять указанные пользователем действия | |
| **Методы** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Параметры | | Описание |
| keyPressed | public | | void | | KeyEvent e | | Обрабатывает событие нажатия клавиши клавиатуры |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.viewer.listeners.**OpenGLViewerMouseListener extends MouseAdapter** | | | | | | | |
| **Поля** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | Описание | |
| viewer | | private | | OpenGLViewer | | Экземпляр OpenGLViewer, к которому нужно применять указанные пользователем действия | |
| **Методы** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Параметры | | Описание |
| mousePressed | public | | void | | MouseEvent e | | Обрабатывает событие нажатия клавиши мыши |
| mouseReleased | public | | void | | MouseEvent e | | Обрабатывает событие отпускания клавиши мыши |
| mouseDragged | public | | void | | MouseEvent e | | Обрабатывает событие движения мыши при нажатой левой клавише мыши |
| mouseWheelMoved | public | | void | | MouseWheelEvent e | | Обрабатывает событие движения колесика мыши |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.viewer.listeners.**OpenGLViewerPopupMenu** | | | | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Параметры | Описание |
| OpenGLViewerPopupMenu | public | OpenGLViewerPopupMenu | OpenGLViewer viewer | Конструктор класса. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.viewer.**GLViewerCanvas extends GLCanvas** | | | | | | | |
| **Поля** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | Описание | |
| @G @S curPos | | private | | java.awt.Point | | Позиция мыши при перетаскивании | |
| @G @S viewer | | private | | OpenGLViewer | | Экземпляр компонента OpenGLViewer | |
| **Методы** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Параметры | | Описание |
| setListeners | private | | void | |  | | Создает экземпляры обработчиков событий для данного класса |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.viewer.**OpenGLViewer** | | | | | | | |
| **Поля** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | Описание | |
| @G @S gl | | private | | com.jogamp.opengl.GL2 | | Экземпляр класса GL для использования команд OpenGL | |
| @G @S glu | | private | | com.jogamp.opengl.glu.GLU | | Экземпляр класса GL для использования команд OpenGL Utility Library | |
| textRenderer | | private | | com.jogamp.opengl.util.awt.TextRenderer | | Отрисовщик текста | |
| @G @S curMousePos | | private | | java.awt.Point | | Последнее положение мыши на экране после перетаскивания | |
| @G @S rotn | | private | | Vec3 | | Текущий угол поворота камеры | |
| @G @S center | | private | | Vec3 | | Координаты центра сцены | |
| @G @S drawAxis | | private | | boolean | | Настройка отображения осей координат | |
| @G @S drawDebugText | | private | | boolean | | Настройка отображения отладочного текста | |
| @G @S enabled | | private | | boolean | | Настройка состояния рендерера (включен/выключен) | |
| @G @S needTextureResolution | | private | | boolean | | Настройка необходимости заново загрузить текстуры объектов | |
| axis | | private | | List<GLObject> | | Объекты осей координат для отрисовки | |
| pendingOperations | | private | | Queue<OpenGLOperation> | | Очередь операций по трансформации сцены | |
| @G @S root | | private | | GLObject | | Ссылка на объект графа сцены | |
| **Методы** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Параметры | | Описание |
| init | public | | void | | com.jogamp.opengl.GLAutoDrawable glAutoDrawable | | Инициализирует отрисовщики OpenGL и GLU, устанавливает их возможности |
| dispose | public | | void | | com.jogamp.opengl.GLAutoDrawable glAutoDrawable | | Вызывается при окончании работы с объектом |
| display | public | | void | | com.jogamp.opengl.GLAutoDrawable glAutoDrawable | | Вызывается раз в кадр для отрисовки сцены |
| reshape | public | | void | | com.jogamp.opengl.GLAutoDrawable glAutoDrawable, int x, int y, int width, int height | | Вызывается во время изменения размеров окна |
| getAxis | public | | List<GLObject> | |  | | Возвращает список объектов осей координат для отрисовки |
| drawDebugText | public | | void | | com.jogamp.opengl.GLAutoDrawable glAutoDrawable | | Отрисовывает отладочный текст |
| rotLeft | public | | void | | float deg | | Поворачивает камеру вокруг сцены на deg градусов влево |
| rotRight | public | | void | | float deg | | Поворачивает камеру вокруг сцены на deg градусов вправо |
| rotUp | public | | void | | float deg | | Поворачивает камеру вокруг сцены на deg градусов вверх |
| rotDown | public | | void | | float deg | | Поворачивает камеру вокруг сцены на deg градусов вниз |
| scale | public | | void | | int direction | | Приближает/отдаляет камеру к сцене |
| moveForward | public | | void | | float dist | | Двигает камеру к сцене на dist единиц |
| moveBackward | public | | void | | float dist | | Отдаляет камеру от сцены на dist единиц |
| moveLeft | public | | void | | float dist | | Двигает камеру в левую сторону от сцены на dist единиц |
| moveRight | public | | void | | float dist | | Двигает камеру в правую сторону от сцены на dist единиц |
| moveUp | public | | void | | float dist | | Двигает камеру вверх на dist единиц |
| moveDown | public | | void | | float dist | | Двигает камеру вниз на dist единиц |
| clear | public | | void | |  | | Очищает граф сцены |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс** ru.dansstuff.simpleopengl.window.**OpenGLTestFrame extends JFrame** | | | | | | | |
| **Поля** | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | Описание | |
| @G @S width | | private | | int | | Ширина окна | |
| @G @S height | | private | | int | | Высота окна | |
| @G @S canvas | | private | | GLViewerCanvas | | Экземпляр обертки над OpenGLViewer | |
| @G @S currentObject | | private | | GLObject | | Текущий объект, над которым ведутся манипуляции | |
| **Методы** | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Параметры | | Описание |
| initWindow | private | | void | |  | | Выставляет свойства окна |
| initMenuBar | private | | void | |  | | Добавляет меню в окно |
| getLoadSaveMenu | private | | javax.swing.JMenu | |  | | Возвращает меню работы с файлами |
| getSceneHandlingMenu | private | | javax.swing.JMenu | |  | | Возвращает меню работы со сценой |
| getObjectHandlingMenu | private | | javax.swing.JMenu | |  | | Возвращает меню работы с объектами |

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопроводительного документа и дата | Подпись | Дата |
| измененных | замененных | новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |