— жесткий диск с объемом памяти не менее 128 ГБ свободного дискового пространства;

— видеокарта с объемом оперативной памяти 1024 МБ и поддержкой стандарта OpenGL 3.3;

— монитор с минимальной матрицей 1024x768@60Hz 24bit

— манипулятор типа мышь PS/2 или USB;

— клавиатура стандартная PS/2 или USB (101/102 клавиши);

— операционная система Windows 8.1 и выше.

А.3.6 Требования к информационной и программной совместимости

А.3.6.1 Требования к исходным кодам и языкам программирования

Для компиляции проекта необходимо иметь компилятор, поддерживающий C++17 или выше. CMake версии 3.10 или выше.

А.3.6.2 Требования к программным средствам, используемым программой

— операционная система: Windows 8.1 и выше, Linux, Mac OS X;

— драйвер видеокарты, совместимый с OpenGL 3.3 и выше.

А.3.6.3 Требования к защите информации и программ

Требования к защите информации и программ не предъявляются.

А.3.7 Требования к маркировке и упаковке

Требования к маркировке и упаковке отсутствуют.

А.3.8 Требования к транспортировке и хранению

Требования к транспортировке и хранению не предъявляются.

А.4 Требования к программной документации

Программная документация должна состоять из следующих документов:

— описание программы;

— программа и методика испытаний;

— руководство пользователя.

Содержание и структура программной документации соответствует требованиям ЕСПД.

А.5 Технико-экономические показатели

Технико-экономические показатели отсутствуют.

А.6 Стадии и этапы разработки

Разработка программы включает в себя следующие стадии:

— анализ исходных данных и постановка задачи проектирования;
— разработка и утверждение технического задания;
— разработка структуры приложения;
— разработка отдельных модулей системы;
— интегрирование модулей в систему;
— тестирование системы;
— отладка системы;
— разработка программной документации.

А.7 Порядок контроля и приемки

Контроль выполнения осуществляется руководителем еженедельно. Так же для контроля выполнения требований технического задания необходимо провести испытания. Порядок и состав испытания определяются программой и методикой испытаний. Контроль и приемка программного обеспечения осуществляются в соответствии с программой и методикой испытаний, разработанной по ГОСТ 19.301-2000 «Программа и методика испытаний. Требования к содержанию и оформлению».

Основным методом испытания программы будет визуальный контроль выполнения программой требующихся функций.

ПРИЛОЖЕНИЕ Б

(обязательное)

Программа и методика испытаний

Б.1 Объект испытания

Объектом испытания является программное обеспечение «Графическое приложение для разработки шейдерных программ с использованием визуального программирования»

Б.2 Цель испытаний

Целью испытаний является проверка корректности работы программы, а также проверка выполнения функций, указанных в техническом задании. При выявлении несоответствий в работе программы, ошибок в программе или программной документации требуется доработка программного обеспечения и (или) документации.

Б.3 Состав предъявляемой документации

Для проведения испытаний программы должна быть предоставлена следующая техническая документация:

— техническое задание с перечислением всех требований, предъявляемых к программе;

— описание программы;

— руководство оператора программы;

— программа и методика испытаний.

Программная документация должна быть оформлена в соответствии со следующими нормативными документами:

— ГОСТ 19.201-78 ЕСПД. Техническое задание. Требования к содержанию и оформлению;

— ГОСТ 19.301-2000 ЕСПД. Программа и методика испытаний. Требования к содержанию, оформлению и контролю качества;

— ГОСТ 19.402-2000 ЕСПД. Описание программы. Требования к содержанию, оформлению и контролю качества;

— ГОСТ 19.505-79 ЕСПД. Руководство оператора. Требования к содержанию и оформлению.

Б.4 Требования к программе

Основным требованием к разработанному программному обеспечению является корректное выполнение функций, установленных техническим заданием.

Б.5 Средства и порядок проведения испытаний

Все испытания проводились на стационарном компьютере со следующей конфигурацией:

- ОС Windows 10 64 bit;
- AMD Athlon II X3 435;
- 4 GB ОЗУ;
- 128 GB HDD.

Порядок проведения испытаний:

- проверка работоспособности модуля редактирования;
- проверка работоспособности модуля компилирования;
- проверка работоспособности модуля визуализации.

Б.6 Методы испытания

Тестирование программы выполняется в соответствии с порядком проведения испытаний, указанном в пункте Б.5. Будет проведено функциональное тестирование и тестирование всех модулей. Для каждого испытания разработаны проекты тестов, которые описаны в таблице Б.1.

Таблица Б.1 – Проекты функциональных тестов

Требование	**Идентификатор системного тестового случая**	**Входные данные теста**	**Цель проведения теста**
Модуль редактирования	Т.1	Нет	Проверка работоспособности редактора
Модуль компиляции	Т.2	Результат работы модуля редактирования	Проверка работоспособности компилятора
Модуль визуализации	Т.3	Результат работы модуля компилирования	Проверка работоспособности визуализатора

Б.6.1 Тестирование графического интерфейса

Специальные тесты для графического интерфейса пользователя не разрабатываются. Причина состоит в том, что применение интерфейса подразумевается во всех тестах свойств. Если какой-либо из этих тестов завершается неудачно, то причина будет связана либо с графическим интерфейсом пользователя, либо с функциональными возможностями, которые доступны через этот интерфейс.

Б.6.2 Тестирование выходных данных

В разделе представлены процедуры для всех тестов, приведенных в таблице Б.1.

Тестовый случай Т.1 Модуль редактирования

Случай 1:

1 Развернуть ветвь в магазине узлов.

2 Взять узел и потянуть его на сцену.

3 Отпустить узел над сценой.

Ожидаемый результат: узел с указанным именем и типами будет создан на сцене в месте его отпускания.

Случай 2:

1 Добавить ещё один узел на сцену

2 Попробовать соединить узлы друг с другом.

Ожидаемый результат: узлы будут соединены между собой в случае, если это возможно

Случай 3:

1 Создать переменную.

2 Добавить её на сцену.

3 Сделать её типа GET или SET.

Ожидаемый результат: Будет создана переменная, а её узел будет помещён на сцену.

Случай 4:

1 Выделить созданную переменную в списке переменных.

2 Нажать на кнопку изменения имени.

3 Ввести новое имя.

4 Сохранить изменения.

Ожидаемый результат: имя переменной должно измениться везде, даже на узлах.

Случай 5:

1 Выбрать ранее переименованную переменную.

2 Удалить эту переменную.

Ожидаемый результат: Переменная и все её узлы будут удалены.

Случай 6:

1 Добавить на сцену несколько узлов.

2 Соединить их между собой так, чтобы каждый узел определял тип следующего узла.

3 Удалить один из узлов.

Ожидаемый результат: все последующие узлы потеряют подключения к невалидному порту узла.

Тестовый случай Т.2 Модуль компилирования

Случай 1:

1 Начать компиляцию подготовленного проекта.

2 Дождаться завершения.

3 Проверить вывод компилятора.

Ожидаемый результат: компилятор должен собрать рабочую программу.

Случай 2:

1 Начать компиляцию подготовленного проекта.

2 Попытаться прервать компиляцию.

Ожидаемый результат: компиляция должна завершиться досрочно.

Случай 3:

1 Подготовить данные для компиляции с ошибкой.

2 Выполнить компиляцию.

Ожидаемый результат: Компиляция завершиться с ошибкой.

Случай 4:

1 Подготовить данные с лишними элементами.

2 Выполнить компиляцию.

Ожидаемый результат: Программа завершится с ошибкой, если эта переменная будет мешать работе компилятора.

Тестовый случай Т.3 Модуль визуализации

Случай 1:

1 Выбрать пункт меню «Load model».

2 Выбрать загружаемый объект.

3 Дождаться загрузки.

Ожидаемый результат: на сцене должен появится загруженный объект.

Случай 2:

1 Навести курсор на изображение визуализируемой сцены.

2 Нажать левую кнопку мыши.

3 Потянуть курсор мыши в любом направлении, удерживая клавишу мыши.

Ожидаемый результат: угол поворота камеры должен измениться.

Случай 3:

1 Навести курсор на изображение визуализируемой сцены.

2 Начать вращать колёсико мыши на себя и от себя.

Ожидаемый результат: Камера будет приближаться и отдаляться от объекта.

Случай 4:

1 Подготовить тестовую сцену в редакторе.

2 Выполнить компиляцию компилятором.

Ожидаемый результат: результат компиляции должен быть автоматически применён к объекту, установленному в сцене.

ПРИЛОЖЕНИЕ В

(обязательное)

Описание программы

Анотация

Данный документ является описанием программы для разработки шейдерных программ с использованием визуального программирования Документ включает в себя следующие разделы:

— общие сведения;

— функциональное назначение;

— описание логической структуры;

— вызов и загрузка;

— входные данные;

— выходные данные.

В.1 Общие сведенья

На основании технического задания была разработана программа для разработки шейдерных программ с использованием визуального программирования.

Разработанное программное обеспечение является настольным кроссплатформенным приложением, реализованным на языке программирование C++ с использованием Qt framework версии 5.12. В качестве среды разработки использовалась IDE Qt Creator версии 4.8.1. Система сборки, использованная при разработке – CMake версии 3.14.

Для запуска программы необходима система с установленными бинарными файлами библиотеки Qt.

В.2 Функциональное назначение

Данное программное обеспечение выполняет три основные функции:

— предоставляет инструмент разработки шейдерной логики;

— компилирует результат разработки шейдерной программы;

— отображает результаты компиляции.

В.3 Описание логической структуры

Основными задачами, которые ставились при проектировании программного средства, являются:

— выбор набора компонент для реализации создания и редактирования логики шейдерных программ;

— проектирование удобной подсистемы компилирования шейдерной логики;

— проектирование графического интерфейса для визуализации результатов работы;

— выбор минимальных характеристик технических средств и операционной системы;

— выполнение функций в соответствии с эксплуатационными документами.

В запущенном программном обеспечении пользователь может построить логику шейдера и увидеть результат его работы.

В.4 Вызов и загрузка

Для запуска разработанной программы необходимо перейти в каталог с скомпилированным исполняемым файлом программы и запустить его любым способом запуска исполняемых файлов, доступной в используемой операционной системе. После успешного запуска пользователь увидит рабочий диалог.

В.5 Входные данные

В качестве входных данных программа принимает пользовательские проекты и модели. Пользовательские проекты используются для восстановления прогресса работы. Пользовательские модели используются для их визуализации в окне визуализации с целью продемонстрировать результат работы разработанного шейдера.

В.6 Выходные данные

Выходными данными являются сохраняемые пользователями файлы проектов и экспортируемые скомпилированные шейдерные программы.

ПРИЛОЖЕНИЕ Г

(обязательное)

Руководство пользователя

Анотация

Данное руководство предназначено для пользователей, работающих с графическим приложением для разработки шейдерных программ с использованием визуального программирования. Документ включает в себя следующие разделы:

— назначение программы;

— условия выполнения программы;

— установка и конфигурирование системы;

— выполнение программы;

— сообщения оператору.

Г.1 Назначение программы

Данное программное обеспечение предназначено для разработки шейдерных программ с помощью визуального подхода к программированию.

Г.2 Условия выполнения программы

Данное программное обеспечение будет успешно функционировать на любой системе, поддерживающей:

— Qt 5.11 Framework;

— OpenGL 3.3.

Дополнительными требования:

— 2 Гб оперативной памяти;

— 200 Мб свободного дискового пространства.

Г.3 Установка и конфигурирование системы

Для установки данного программного обеспечения на компьютер необходимо скопировать или скомпилировать исполняемый файл программы.

Если программа устанавливается в среде ОС Windows, необходимо убедится, что рядом с исполняемым файлом присутствуют все динамические библиотеки, необходимые для работы программы.

Если программа устанавливается в UNIX подобных системах, то пользователю необходимо убедится, что у целевой системы установлен Qt 5.11 Framework.

Г.4 Выполнение программы

При запуске программы пользователь видит диалоговое окно открытия или создания проекта, как показано на рисунке Г.1.

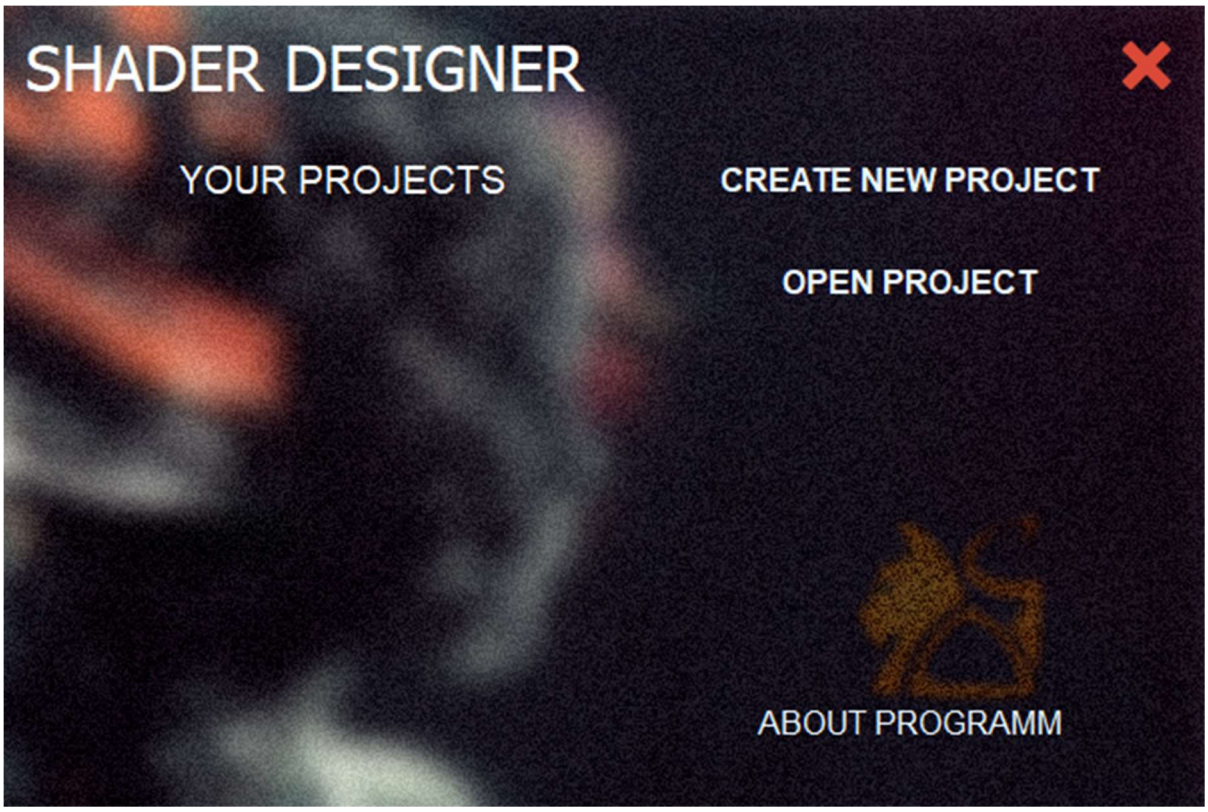

Рисунок Г.1 – Диалог запуска проекта

После выполнения желаемого действия открывается главное окно приложение, включающая в себя все доступные компоненты управления. На рисунке Г.2 изображено главное окно программы.

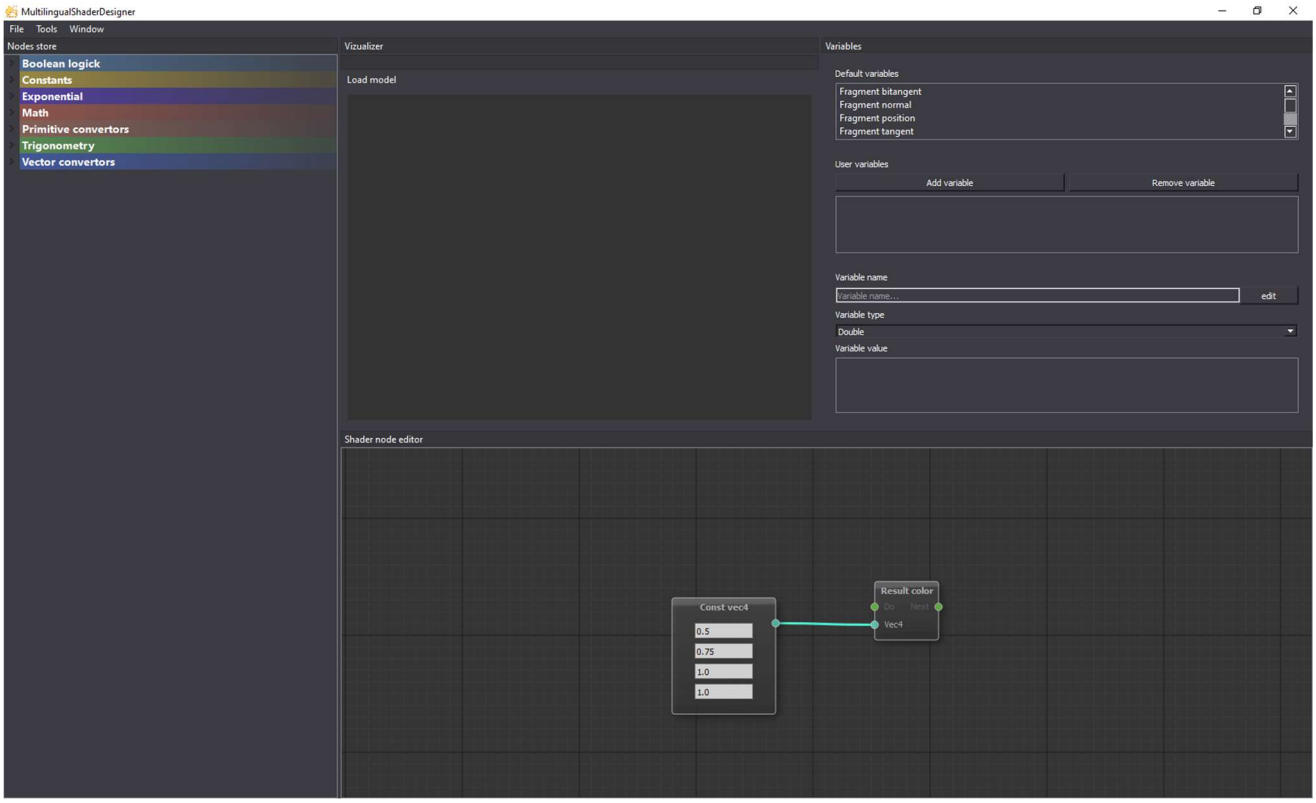

Рисунок Г.2 – Главное окно программы

Для создания первой шейдерной программы необходимо раскрыть одну из ветвей в магазине узлов и перетянуть один из узлов на графический редактор. В результате графический редактор может выглядеть так же, как на рисунке Г.3.

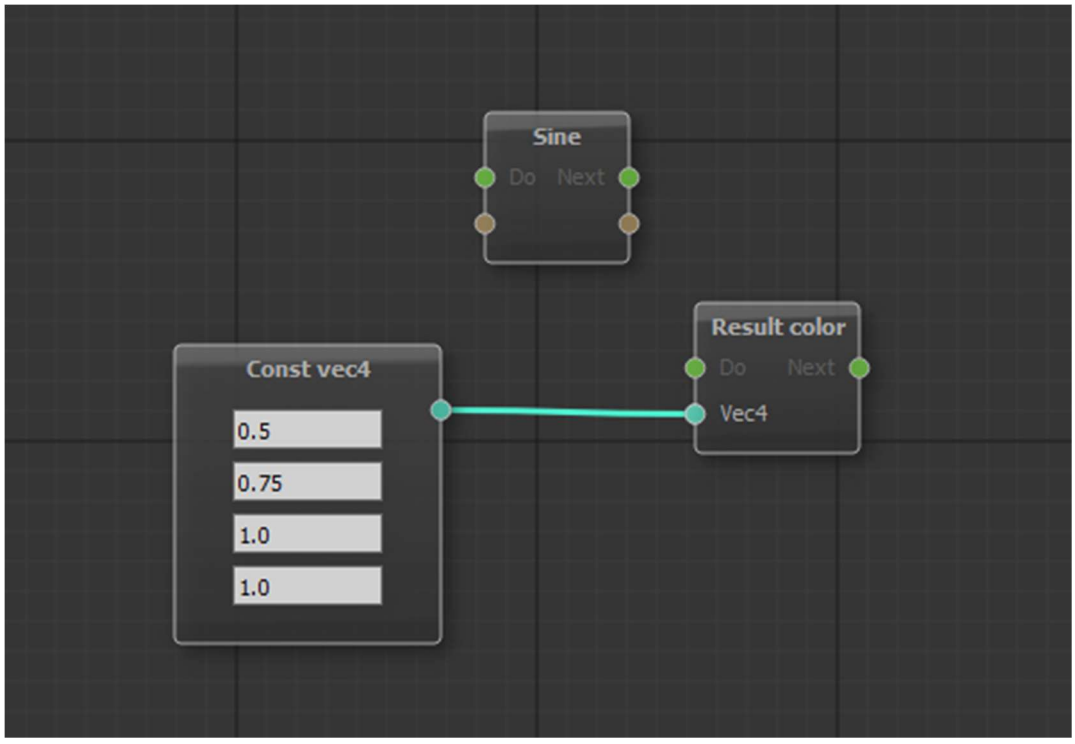

Рисунок Г.3 – Пример расположения нового узла на графическом редакторе

После того, как новый узел был добавлен, пользователь может установить новые соединения между узлами, покликав на окончания соединений между узлами. В результате управления узлами у пользователя может получиться результат, близкий к результату, изображенном на рисунке Г.4.

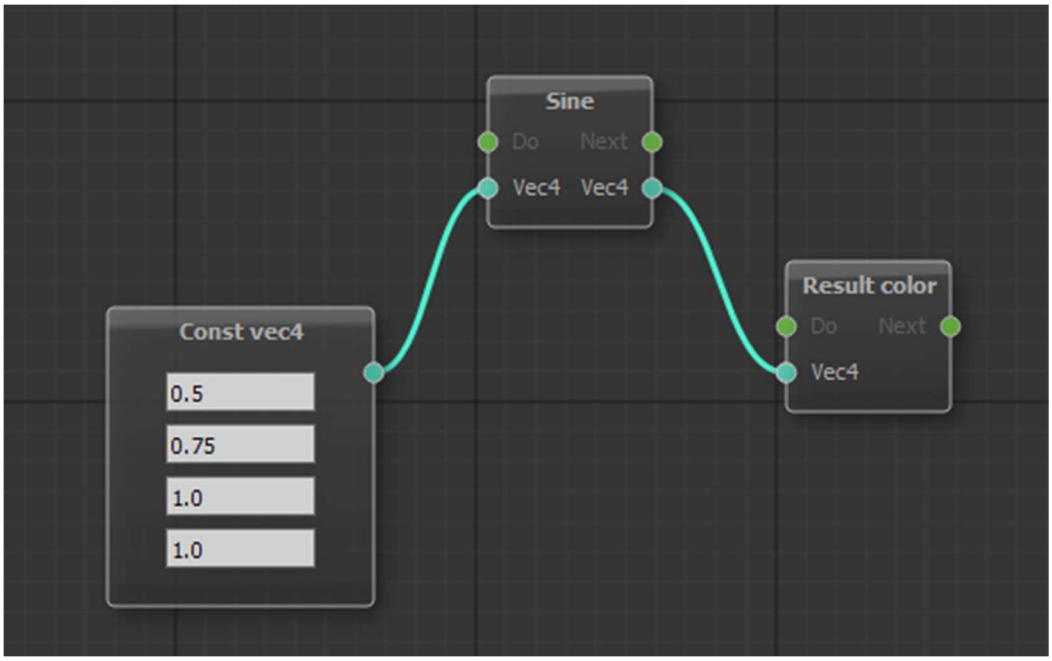

Рисунок Г.4 – Результат соединения узлов

После успешного соединения узлов необходимо установить модель в сцену и скомпилировать шейдерную программу. В результате получится результат, аналогичный результату на изображение Г.5.

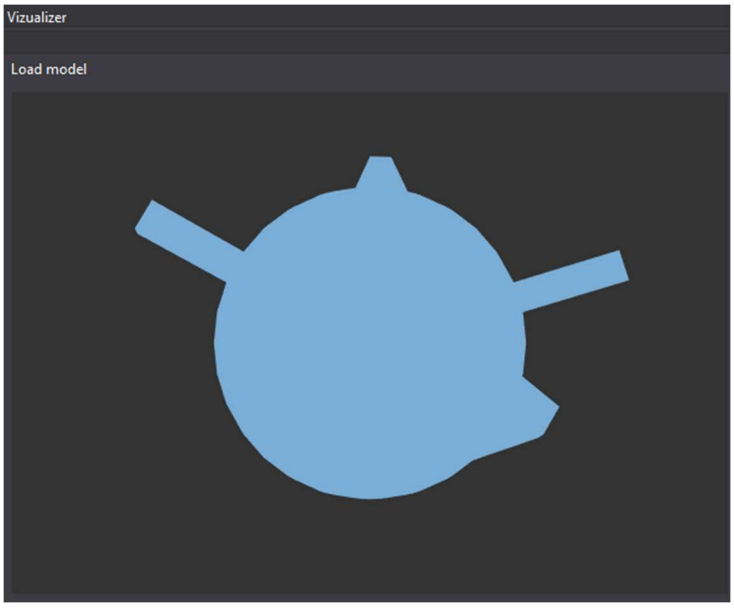

Рисунок Г.5 – Результат применения скомпилированного шейдера

Г.5 Сообщения оператору

Во время работы программы, пользователь может выполнять действия, которые могут так или иначе повлиять на процесс работы программы. Чтобы избежать случаев неожиданного завершения программы или тому подобных случаев, необходимо сообщать пользователю о его ошибках, в идеале, не допуская таких действий со стороны пользователя.

Если пользователь начнёт выполнять действия, которые могут привести к ошибкам, то программа сообщит пользователю информацию о том, что он сделал не так. На рисунке Г.6 представлен диалог с сообщением об ошибке во время создания проекта.

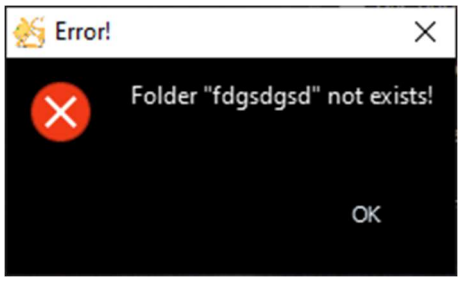

Рисунок Г.6 – сообщение об ошибки во время создания проекта

ПРИЛОЖЕНИЕ Д

(справочное)

Диаграммы вариантов использования

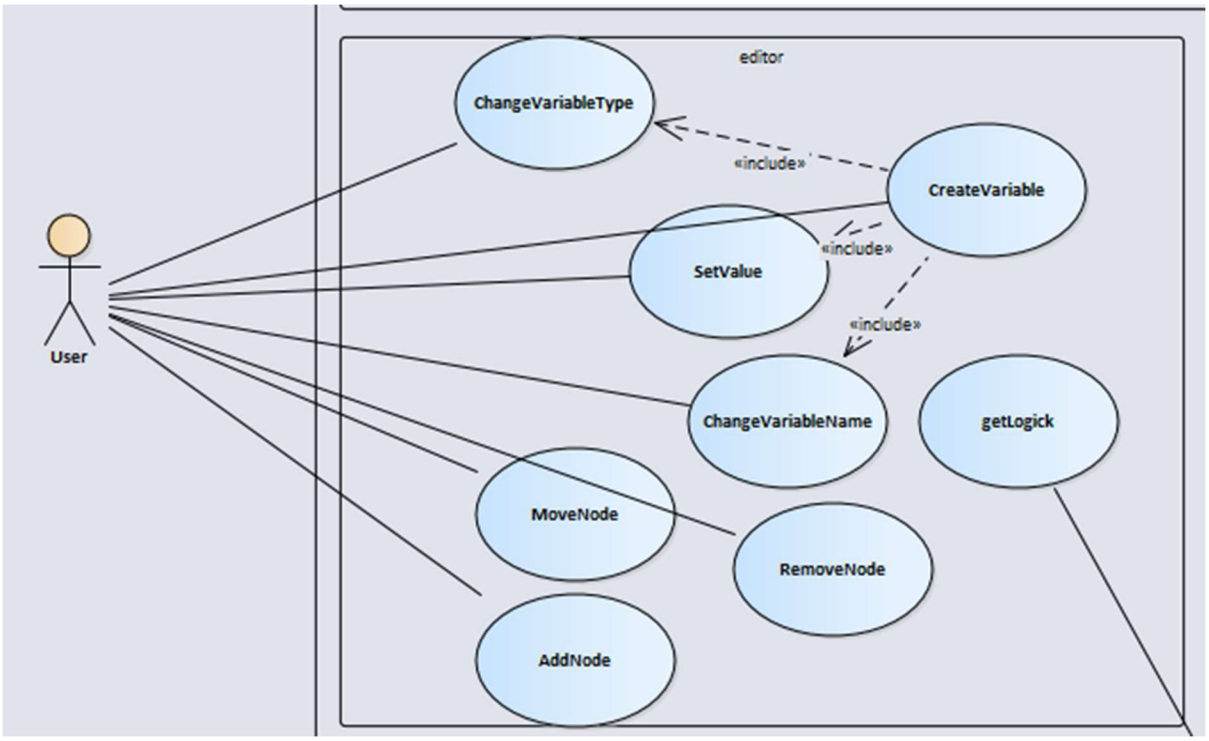

Рисунок Д.1 – Диаграмма вариантов использования модуля редактирования

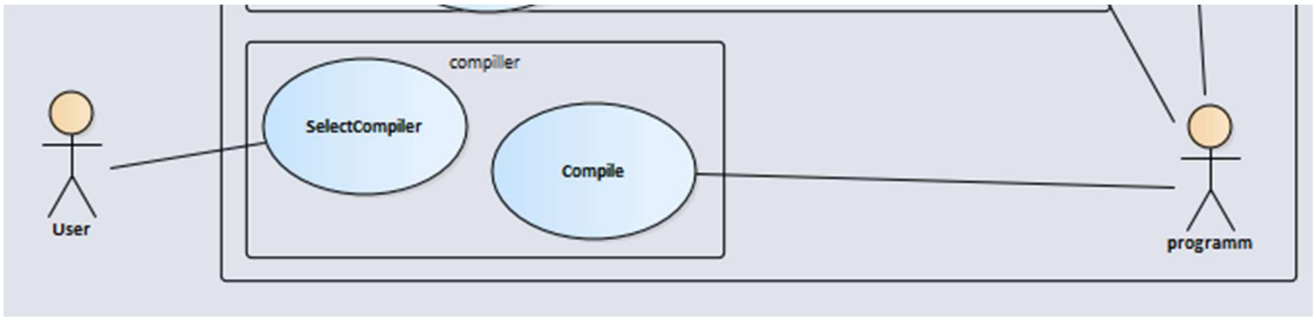

Рисунок Д.2 – Диаграмма вариантов использования для модуля компилирования

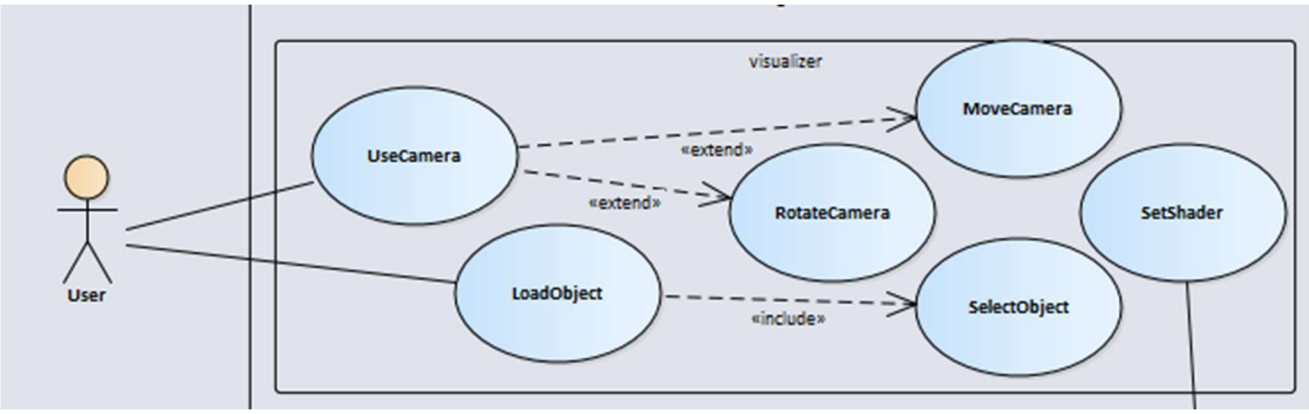

Рисунок Д.3 – Диаграмма вариантов использования для модуля визуализации

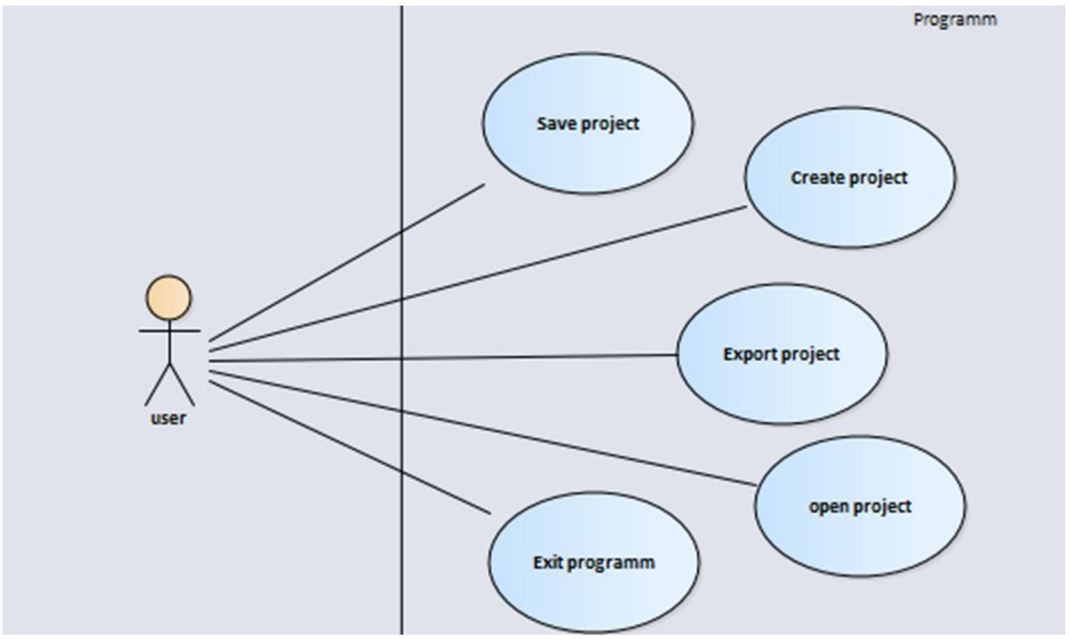

Рисунок Д.4 – Диаграмма вариантов использования модуля программы

ПРИЛОЖЕНИЕ Е

(справочное)

Диаграммы классов

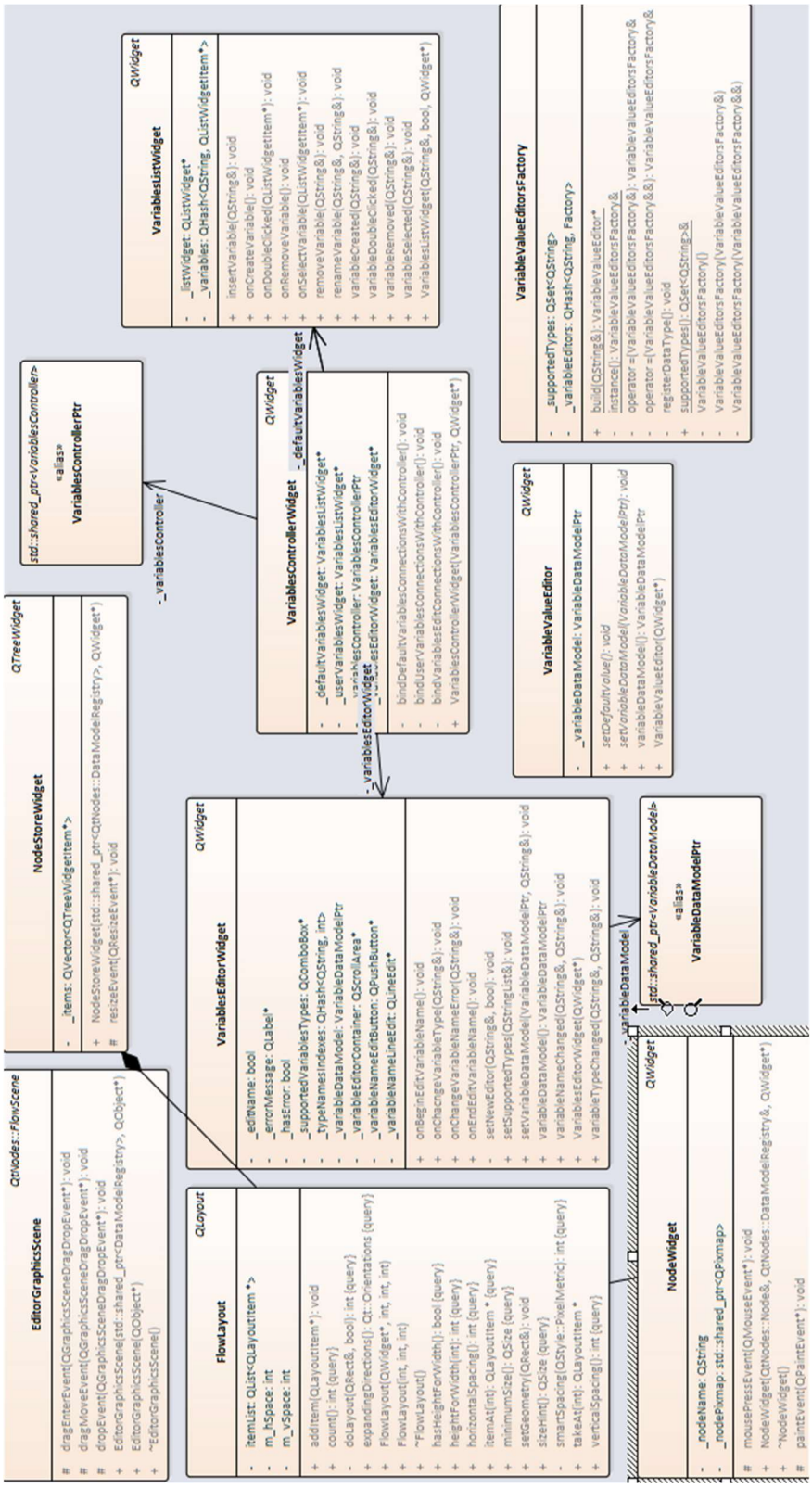

Рисунок Е.1 – Диаграмма классов модуля редактирования

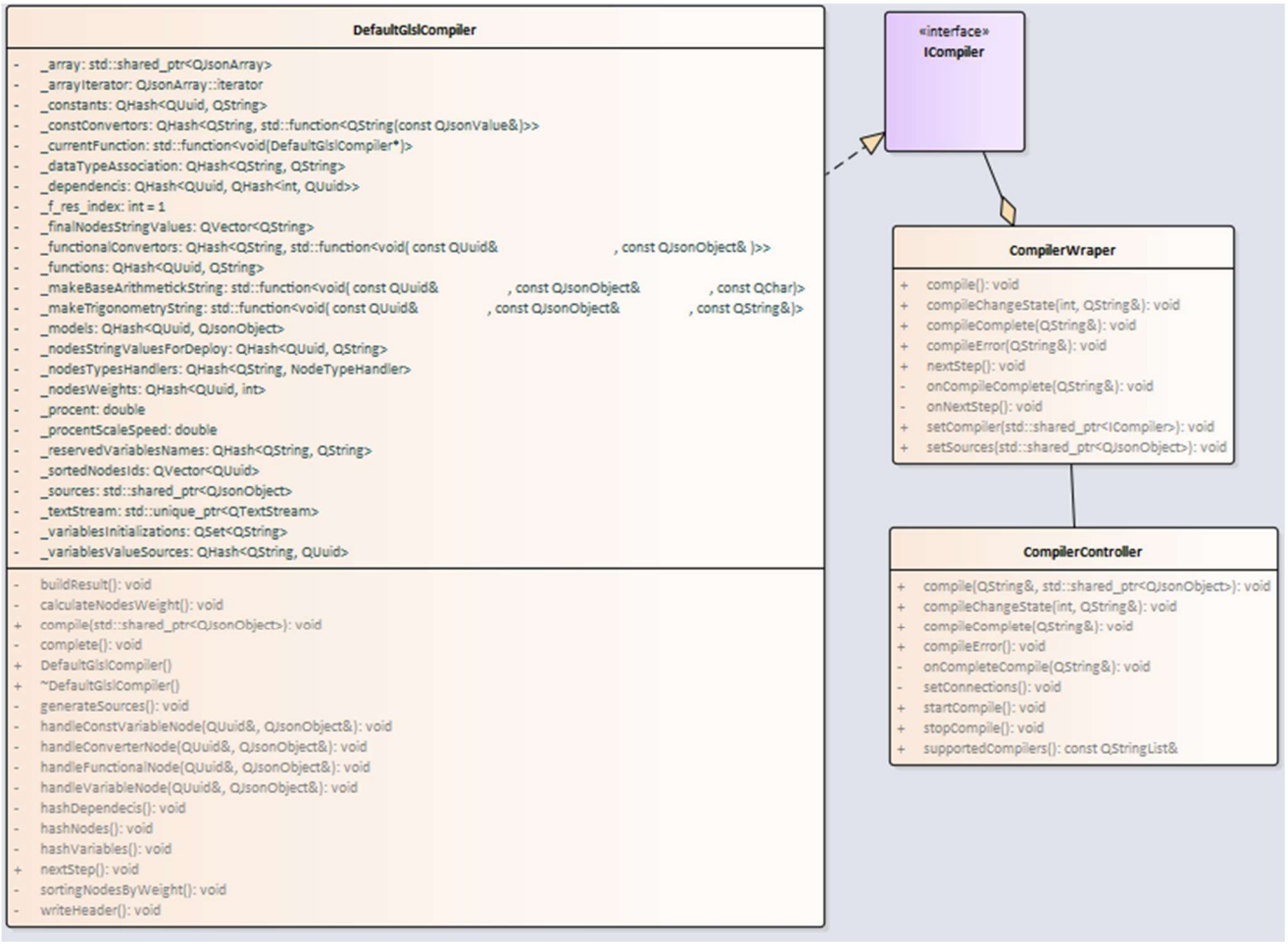

Рисунок Е.2 – Диаграмма классов компилятора

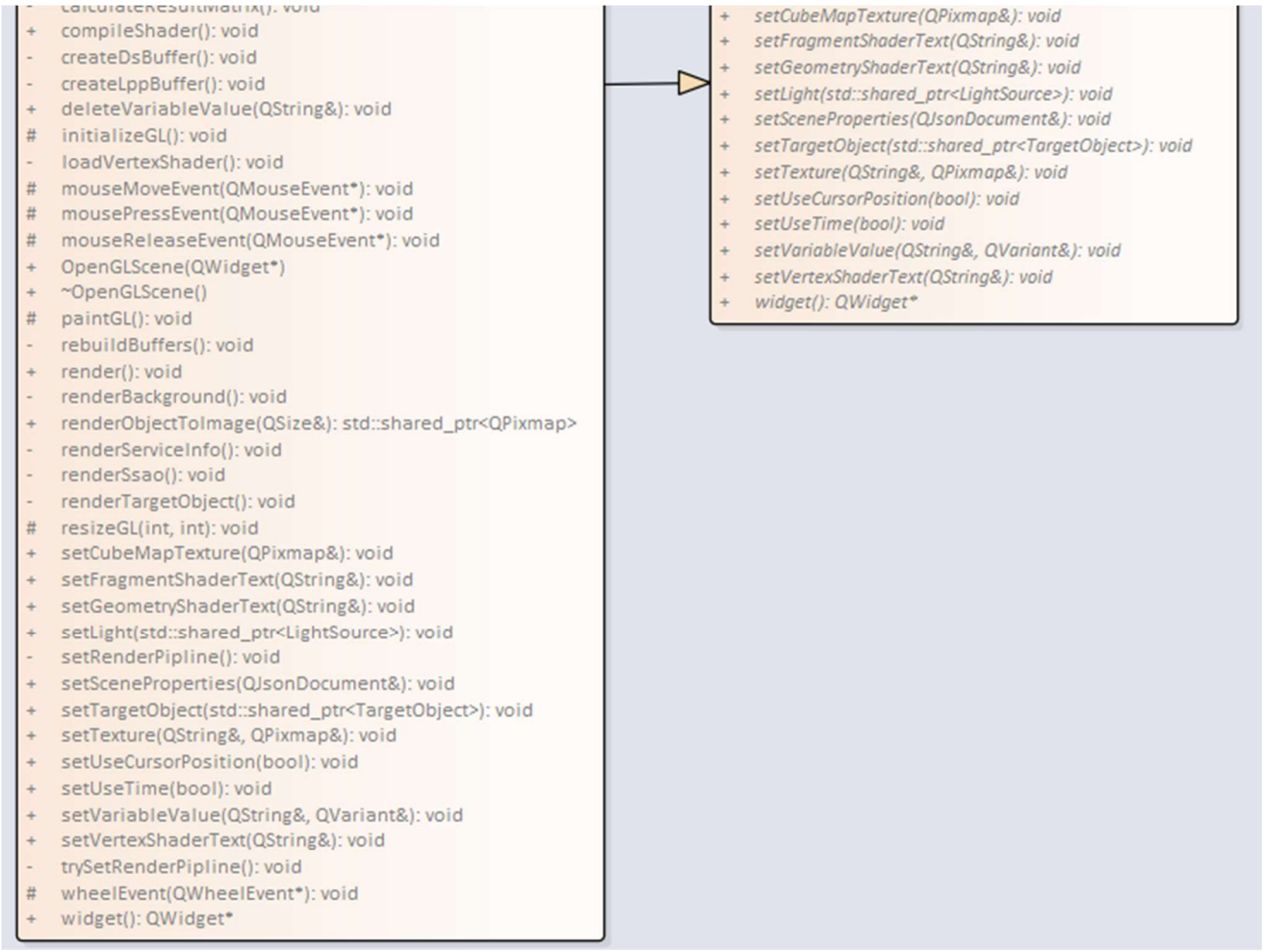

Рисунок Е.3 – Диаграмма классов модуля визуализации

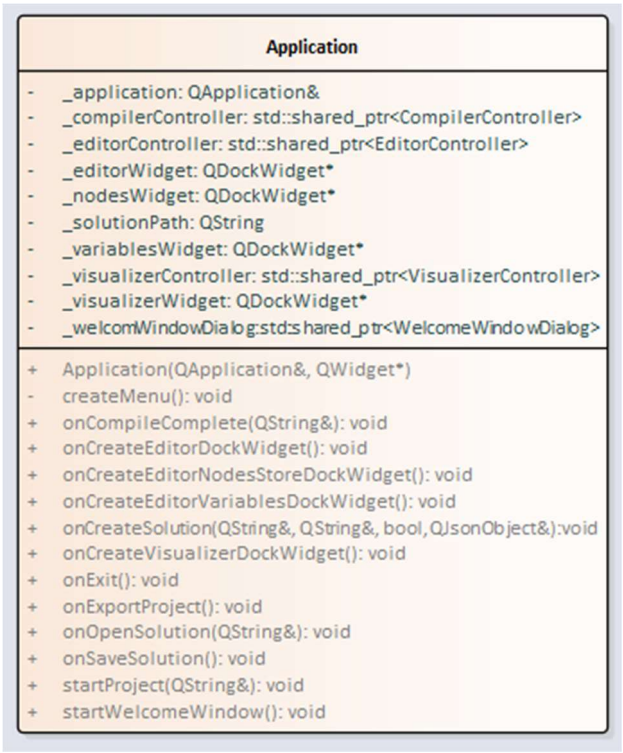

Рисунок Е.4 – диаграмма классов модуля программы

ПРИЛОЖЕНИЕ Ж

(справочное)

Диаграмма развёртывания приложения