МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Автоматизированные системы управления»

КОМПЬЮТЕРНАЯ ГРАФИКА

Методические рекомендации к курсовому проектированию для студентов направлений подготовки 09.03.01 «Информатика и вычислительная техника» и 09.03.04 «Программная инженерия» очной формы обучения

Могилев 2022

УДК 004.92
ББК 32.973-02
К63

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Автоматизированные системы управления» «30» августа 2022 г., протокол № 1

Составители: канд. техн. наук, доц. В. А. Широченко;
д-р техн. наук, доц. А. И. Якимов

Рецензент Ю. С. Романович

Методические рекомендации к курсовому проектированию предназначены для студентов направлений подготовки 09.03.01 «Информатика и вычислительная техника», 09.03.04 «Программная инженерия».

Учебно-методическое издание

КОМПЬЮТЕРНАЯ ГРАФИКА

Ответственный за выпуск	А. И. Якимов
Корректор	Т. А. Рыжикова
Компьютерная верстка	Н. П. Полевничая

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 26 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.
Пр-т Мира, 43, 212022, г. Могилев.

Содержание

1 Цель курсового проектирования

Целью курсового проектирования является разработка десктопного или мобильного игрового приложения с использованием игровых движков Unity или Unreal Engine, выполненного в 2D- или 3D-графике.

В ходе выполнения курсового проекта решаются следующие основные задачи:

– изучить основные методы разработки игровых приложений;

– изучить современные технологии разработки игровых приложений;

– овладеть навыками разработки 2D-спрайтов и 3D-моделей;

– научиться применять способы создания фотореалистических изображений;

– узнать основные функциональные возможности современных игровых движков, организацию диалога в игровых движках;

– научиться ориентироваться в классификации современных моделей создания игровых приложений;

– научиться реализовывать доступный и удобный для игрока пользовательский интерфейс в игровых приложениях;

– научиться реализовывать построение «открытых» графических систем, 2D- и 3D-моделирование;

– овладеть навыками разработки мобильных и десктопных игровых приложений.

2 Описание интегрированной среды разработки игры

При первом запуске игрового движка Unreal Engine отобразится окно Project Browser (рисунок 1).

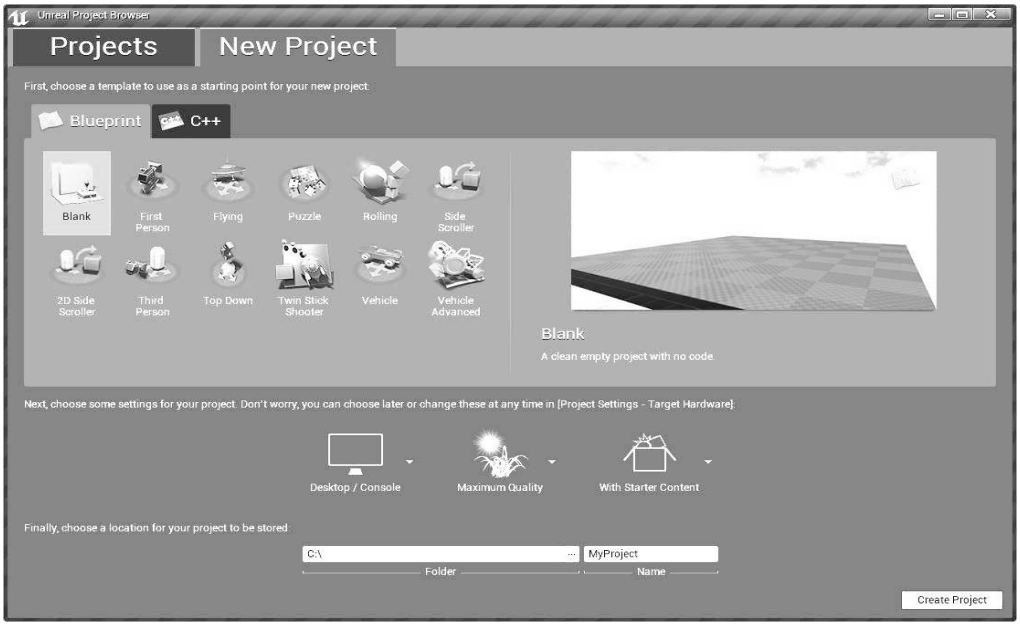

Рисунок 1 – Окно Project Browser

Project Browser является основным хранилищем проектов. В нем можно создать новый проект, открыть уже существующий проект с компьютера. Также имеется поддержка проектов «Примеры», в которых можно оценить возможности и работу Unreal_Engine_4.

В окне Projects (рисунок 2) будет показан весь список проектов, когда хоть один будет создан. Изначально здесь будет приведен список всех проектов в папке установки. Двойной клик по иконке открывает проект.

Рисунок 2 – Окно Projects

Также вы можно нажать Browse и выбрать проект на компьютере. Для этого нужно выбрать *.uproject файл.

Вкладка New Project (рисунок 3) позволяет создать новый проект из существующих заготовок. Шаблон Blank создает полностью пустой проект.

Рисунок 3 – Окно New Project

Всего имеется два типа заготовок: C++ и язык визуального программирования Blueprint. C++ означает, что логика проекта будет писаться на C++. Другой тип означает, что для логики будет предоставлен визуальный скриптовый язык Blueprint. В процессе разработки можно совмещать два типа.

Чтобы создать новый проект, необходимо:

– выбрать заготовку из списка;
– установить настройки проекта;
– ввести название проекта и путь к нему;
– нажать Create.

После создания проекта откроется редактор и запустится стандартный уровень.

При создании проекта имеется возможность установить его начальные настройки графики и контента. После создания проекта эти настройки можно изменить в настройках проекта (рисунок 4).

Проект будет оптимизирован для ПК или консолей.

Оптимизирует проект для запуска на мобильных платформах.

Опция позволяет создать проект с высокими настройками графики. Используется при создании проекта для ПК или консолей.

Опция позволяет создать проект с низкими настройками графики. Используется при создании проекта для мобильных платформ.

Включает в проект стартовый контент. Не рекомендуется включать, если создается мобильная игра, т. к. данный контент очень сильно влияет на размер.

Не включает в проект стартовый контент.

Рисунок 4 – Содержимое настроек проекта

В основу редактора Unreal Engine 4 входит окно редактора уровня (рисунок 5). Данный режим доступен по умолчанию и отображается в качестве самого главного редактора, поэтому изучение его является неотъемлемой частью изучение движка.

Рисунок 5 – Редактор уровней

Редактор уровней предоставляет весь основной функционал для проектирования и создания уровней в проекте. В данном редакторе можно просматривать, редактировать и создавать уровни, располагая на нем объекты, а также манипулируя ими и изменяя их свойства.

Все сцены, которые создаются в Unreal Engine 4, интерпретируются как уровни. Под данным термином можно представлять 3-мерное окружение, в котором игрок существует и взаимодействует с данным миром. Всё, что помещено на уровень, будь то модель, персонаж, источник света, называются Actor (объект, размещаемый на сцене). Техническим языком термином Actor определяется всё, что имеет позицию, поворот и масштаб.

Процесс создания уровней состоит в основном из расположения различных объектов на сцену. Это могут быть обычные декорации из мешей, источников света и т. д. Также на уровне могут располагаться игроки, автомобили, частицы и все, что можно увидеть глазами.

Список панелей окна Unreal Engine 4 представлен на рисунке 6.

Панель вкладок находится вверху основного окна и отображает единственную вкладку с названием открытого уровня. Вкладки остальных окон могут быть помещены на данную панель для более удобной навигации между разными редакторами. Панель имеет сходство с панелью вкладок в веб-браузере.

Имя вкладок отображает название открытого уровня или редактируемого ассета (используемый ресурс, например 3D-модель) в случае, если вкладка принадлежит другому окну.

1 – панель вкладок и меню; 2 – Toolbar; 3 – Modes; 4 – Content Browser; 5 – вьюпорты (по умолчанию развернут режим перспективы); 6 – Scene Outliner; 7 – Details

Рисунок 6 – Панели Unreal Engine 4

Справа на панели вкладок отображается название проекта, в котором производится работа в текущий момент.

Toolbar, как и в многих программах, – это меню с быстрым доступом к наиболее используемым функциям редактора Unreal Engine 4 (рисунок 7).

Сохраняет уровень. Сохраняет только уровень без остального контента.

Открывает Content Browser.

Открывает Магазин, где можно скачать примеры работ или купить контент.

Открывает меню Quick Settings, которое предоставляет доступ к самым важным настройкам редактора уровней.

Открывает настройки уровня в окне Details.

Рисунок 7 – Содержимое панели Toolbar

Предоставляет быстрый доступ к функциям создания и редактирования Blueprint’ов.

Позволяет создать или отредактировать существующую Matinee последовательность.

Производит операции предпросчета. Данная операция просчитывает все самые сложные аспекты уровня, например, такие как статическое освещение.

Компилирует и перезагружает код C++. Данная кнопка доступна лишь, если проект выполняется на C++ и установлена Visual Studio 2015.

Запускает режим симуляции уровня.

Запускает режим игры в нормальном состоянии. Нажав на стрелочку рядом, можно установить дополнительные настройки.

Запускает уровень на любом из поддерживаемых устройств. Список устройств можно увидеть, раскрыв меню кликом по стрелочке справа.

Окончание рисунка 7

Вьюпорт – панель, где просматривается и редактируется уровень напрямую.

Панель содержит в себе несколько вьюпортов, которые могут быть скрыты или раскрыты в случае, если нужны будут ортографические виды, для манипулирования объектами.

Панель Details содержит информацию, утилиты и функции, специфичные для выбранного объекта во вьюпорте. Также содержит ячейки для указания данных для перемещения, вращения и масштабирования выбранных объектов, обеспечивает быстрый доступ к дополнительным параметрам в зависимости от типа выбранного объекта. Панель Details позволяет просмотреть используемый на объекте материал, если такой имеется, и быстро открыть его для редактирования.

Объекты могут иметь понятные названия, устанавливаемые непосредственно в редакторе. Имена можно использовать, чтобы вызвать эти объекты или найти их, используя поисковую строку на панели World Outliner.

Чтобы изменить имя объекта, просто следует ввести его название в текстовую строку сверху на панели Details.

Параметры, отображаемые на панели Details, можно отфильтровать, используя поисковую строку (рисунок 8). Одновременно с тем, как вводится текст, параметры объекта автоматически фильтруются в соответствии с текстом.

Свойства могут быть включены или отключены. Свойство можно изменить, только если оно включено. По умолчанию все свойства включены до тех пор,

пока они имеют условие редактирования. Свойства с условием редактирования полагаются на значение другой переменной, чтобы определить, когда они включены и могут быть изменены или отключены.

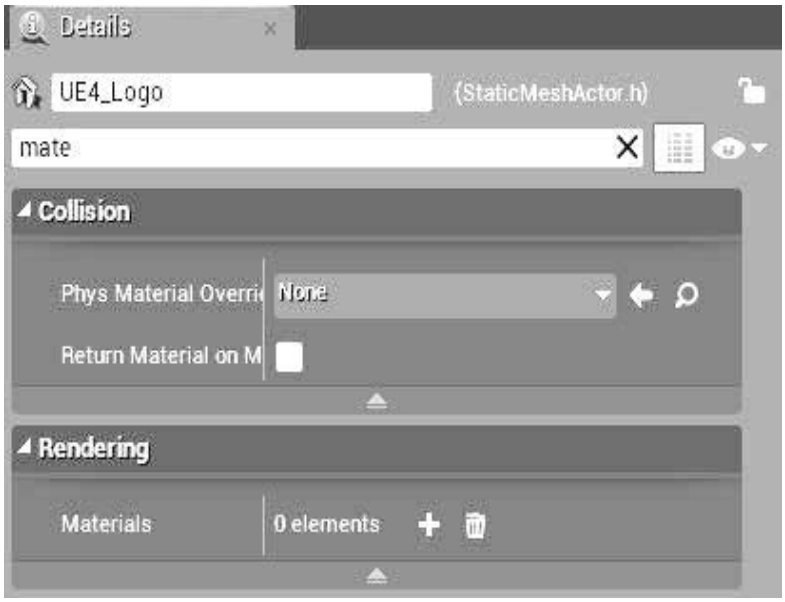

Рисунок 8 – Поисковая строка окна Details

В некоторых случаях изменение условий используется, чтобы определить, когда свойство перепишет некоторые другие значения или не окажет никакого эффекта. В других случаях определенные свойства могут просто не оказывать влияния, пока не встретятся с некоторым условием. Например, можно иметь группу свойств, которые относятся к непрямому (рассеянному) освещению и булевскому свойству, которое глобально переключает, когда непрямое (рассеянное) освещение включено или выключено. Каждое свойство в группе может быть обусловлено глобальным переключателем, поэтому они включены только тогда, когда используется рассеянное освещение.

Все свойства на панели Display отображаются в категориях. Как правило, это категории Rendering, Lighting, Collision и т. д.; они определяют, как свойство инициализируется в коде, и используются в качестве средства организации связанных с ним свойств на группы.

Вкладка Scene Outliner (рисунок 9) отображает список всех объектов на уровне в древовидной системе (для групп). Объекты могут быть вызваны через данную панель. Также можно использовать выпадающее меню Info, чтобы отобразить дополнительные колонки информации об объектах.

Размещение объектов на сцене происходит с помощью перетаскивания определенного объекта из окна Content Browser в любое место окна Viewport (рисунок 10).

Выбор объекта на сцене выполняется простым нажатием на необходимый объект.

После размещения объекта на сцене его можно трансформировать.

Инструмент трансформации используется для управления и манипуляций в

трехмерном пространстве UE4. Существует три типа трансформации: перемещение (Move), масштабирование (Scale) и поворот (Rotate).

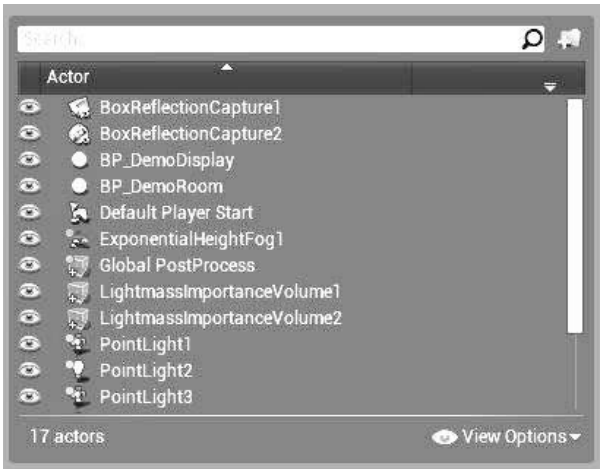

Рисунок 9 – Вкладка Scene Outliner

Рисунок 10 – Размещение объекта на сцене

Трансформация перемещения (Move) – один из наиболее часто используемых инструментов трансформации в UE4. Этот инструмент позволяет перемещать actor в трехмерном пространстве из одного местоположения в другое. Положение координат для каждого Actor основано на якорной или исходной точке Actor, в которую он был помещен в UE4 изначально. Чтобы переместить Actor внутри сцены, необходимо сначала выбрать его, а затем использовать инструмент Move (переместить) трансформации или нажать клавишу W.

Масштабирование (Scale) позволяет увеличивать и уменьшать размеры Actor по осям X, Y или Z. Можно совершать его с сохранением пропорций или с различными коэффициентами масштабирования по осям. Когда Actor помещается на сцену прямо из панели Content Browser, масштаб равен 1 по всем осям. Чтобы изменить масштаб, сначала следует выбрать Actor и затем выбрать инструмент трансформации Scale (масштабировать) или нажать клавишу R. Перемещая любой из ориентированных на направления манипуляторов, можно масштабировать Actor в любом направлении. Выбрав белый куб в центре, можно масштабировать Actor пропорционально по всем осям. Наконец, выбрав одну из

панелей, соединенных с двумя кубами направлений, можно масштабировать Actor пропорционально по двум осям.

Поворот (Rotate) – это последний вид трансформации, который можно использовать для управления Actor внутри игрового мира. Поворот в UE4 производится так же, как в большинстве 3D-программ: установкой градусов поворота. Полный поворот составляет 360 град. Полный поворот может происходить по любой из трех осей: X, Y или Z.

Блупринты – это скриптовая система в Unreal Engine 4, которая представляет собой визуальный интерфейс для создания элементов геймплея.

С помощью блупринтов, разработчики могут создавать такие вещи, как:

– игровые режимы – устанавливать правила игры, изменять поведение игры в общем плане;

– игроки – назначать игроков, придавать им особые черты и вид;

– камеры – создавать виды для обзора и изменять свойства камер в реальном времени;

– управление – назначать кнопки для управления персонажем, автомобилем или внешнем уровнем;

– вещи – оружие, подбираемые предметы и прочее;

– окружение – создание случайно генерированного окружения.

Блупринты являются типом ассетов, которые предоставляют пользователям интуитивную систему для создания специальных типов объектов, которые можно поставить на сцену. Также блупринты используются для создания игровой логики уровня или логики уровня, при этом не пишется ни строчки кода, что облегчит написание логики не только программистам, но и дизайнерам, кто не силен в программировании.

Блупринты используют встроенную в редактор систему для визуального построения логических последовательностей. Соединяя ноды (блоки, события, функции и переменные), возможно создать достаточно сложные логические элементы, которые будут создавать геймплей игры.

Два самых используемых типов блупринтов – Level Blueprint и Class Blueprint.

Level Blueprint используется для манипулирования объектами на сцене в процессе игры.

Class Blueprint позволяет создать сложные объекты для последующего размещения на сцене, такие как открывающиеся двери, ящики с предметами, кнопки и т. п. На изображении сверху напольная кнопка и дверь являются разными блупринтами и содержат определенный скрипт для взаимодействия с игроком, проигрывания анимации и звука, открытия дверей и т. д.

Игровые персонажи тоже являются классовыми блупринтами, с помощью которых можно создавать все нужные элементы и логику будущего персонажа. Можно устанавливать параметры камеры, устанавливать управление персонажем, включая мышь и даже сенсорные экраны, и создавать вещи, на которые персонаж способен.

При создании блупринта персонажа (Character Blueprint) будут уже заготов-

ленные настраиваемые свойства для передвижения, прыжка, плавания и падения. Все, что нужно, – это добавить управление и определить, как персонаж будет вести себя.

Создавая любую логику с помощью блупринтов, работют с редактором блупринтов. Редакторы бывают различных типов в зависимости от редактируемого блупринта.

Вся функциональность блупринтов использует различного рода элементы, которые и составляют окончательный вариант создаваемого блупринта.

Для того чтобы создать блупринт-класс, необходимо нажать правой кнопкой мыши в пустом месте окна Content Browser и выбрать пункт Blueprint Class (рисунок 11).

Рисунок 11 – Создание блупринт-класса

При двойном нажатии на созданный блупринт-класс откроется окно редактирования класса (рисунок 12).

Основные элементы Blueprints:

– переменные;
– события;
– функции;
– макросы;
– диспетчеры событий;
– соединительные провода.

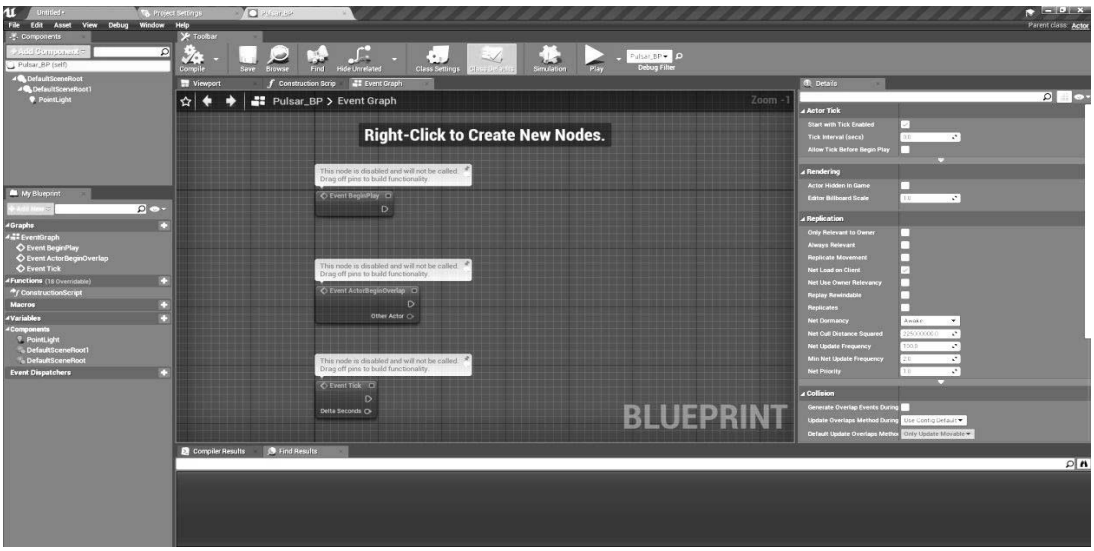

Рисунок 12 – Окно редактирования блупринт-класса

Переменные – это параметры, которые хранят в себе какое-либо значение, или же содержат в себе отсылку к объекту. Данные параметры могут быть доступны как внутри блупринта, так и из внешней среды для того, чтобы разработчик мог контролировать копию блупринта, которая была помещена на сцену.

Переменные отображаются в виде закругленных прямоугольников, которые отображают цвет типа переменной и её имя.

Переменные могут быть созданы с разным типом, включая типы с данными, такие как булевские, целые числа или текст, и типы отсылки к объектам. Также могут быть созданы массивы для каждого типа переменных.

После создания переменной ее можно использовать. При перетаскивании из списка переменных непосредственно в граф появится контекстное меню с выбором установки или получения значения переменной.

Events – события, которые определяют начало конкретной логической последовательности. Они позволяют блупринту исполнять какие-либо действия при выполнении события, например, при старте игры, при рестарте уровня или получении урона.

События создаются непосредственно в Graphs (графах). Существуют события, которые вызываются при реагировании на определенные обстоятельства (например, Event BeginPlay сработает сразу же после начала уровня), и пользовательские события, которые разработчик может вызвать в нужный момент времени. У событий могут быть входы, но нет выходов.

Функции – это узловые графики, принадлежащие определенному блупринту, которые могут быть выполнены или вызваны из этого или другого блупринта. Изначально у функции есть только входные параметры.

Для того чтобы вызвать функцию, можно перетащить ее из списка функций на граф или написать ее имя в поиске.

Blueprint Macros – это то же самое, что и свернутый график нодов. Они имеют контакт входа и выхода, соединенные внутренними блоками. Вызов макроса происходит таким же образом, как и вызов функции.

Основные операторы, использующиеся в Blueprints:

– арифметические операторы (+, –, /, *, ++, – –, % и т. д.);

– операторы сравнения (<, >, <=, >=);

– логические операторы (and, or, not, nor, xor, nand);

– операторы равенства (equal, not equal).

Вызывать эти операторы можно так же, как и любую функцию, нажав ПКМ в любом пустом месте графа и вписав название оператора в окне поиска. Причем, если нужно сложить два числа, достаточно просто нажать +, и появится набор операторов сложения, разделенных по типу слагаемых (int + int, float + float и т. д.).

Операторы ветвления.

Оператор ветвления **Branch** работает от входного параметра типа bool. В случае, если Condition принимает значение True, будет исполняться код, стоящий после True (соответственно, если False – то False). В данном случае на экран будет выводиться строка Yes (рисунок 13).

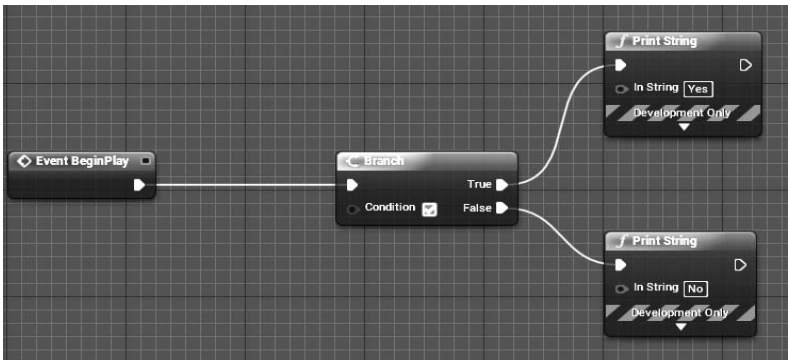

Рисунок 13 – Оператор ветвления Branch

Оператор ветвления **Switch** работает похожим образом. Различие только в том, что на вход подается не bool, а переменные типа int, string, name или enum (рисунок 14).

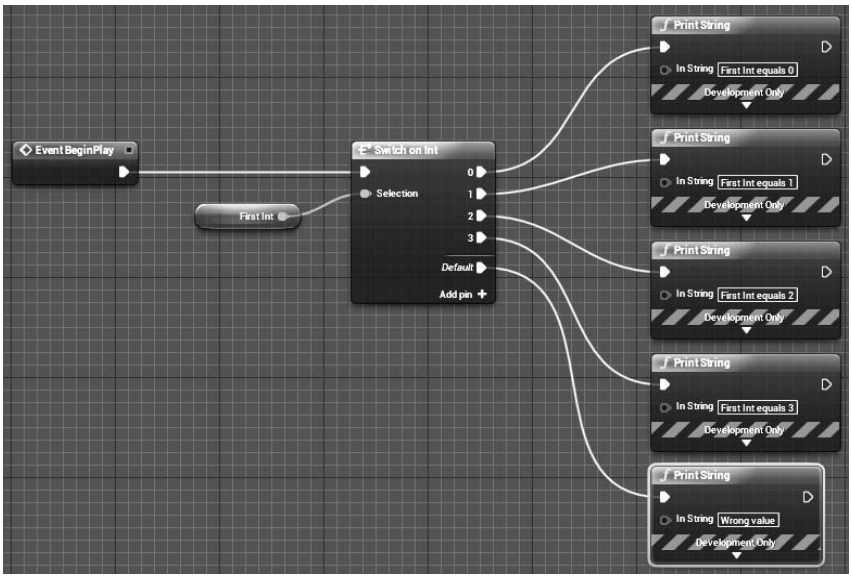

Рисунок 14 – Оператор ветвления Switch

Циклы.

Цикл WhileLoop – цикл будет выполняться до тех пор, пока истинно (True) значение Condition. Как только оно становится ложным (False), цикл будет остановлен (рисунок 15).

Рисунок 15 – Цикл WhileLoop

Данный блок имеет узлы, представленные в таблице 1.

Таблица 1 – Узлы WhileLoop

Название узла	Описание
Основной вход	Запускает выполнение блока
Loop Body	При работе цикла на этот узел будет поступать сигнал
Completed	После завершения цикла на этот узел поступит сигнал
Condition	Пока значение верно, цикл будет работать

ForLoop – узел работает как стандартный цикл (рисунок 16). Выходной узел срабатывает для каждого значения в промежутке, указанном в параметрах блока.

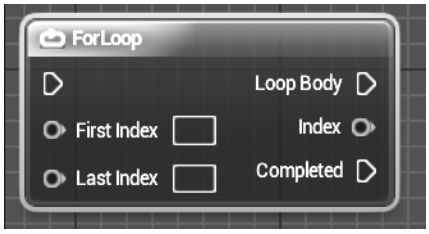

Рисунок 16 – ForLoop

Данный блок имеет узлы, представленные в таблице 2.

ForLoopWithBreak – цикл работает до тех пор, пока будет выполняться условие либо пока в узел Break не придет сигнал.

Циклы **ForEachLoop** и **ForEachLoopWithBreak** работают так же, как и ForLoop и ForLoopWithBreak, за тем исключением, что в них не вводится начальный и конечный индексы, а указывается, по какому массиву необходимо пройтись в цикле. Выполнятся они будут столько раз, сколько элементов в массиве.

Таблица 2 – Узлы ForLoop

Название узла	Описание
Входной узел	Запускает данный блок
First Index	Начальное значение цикла
Last Index	Последнее значение цикла
Loop Body	Срабатывает при каждом круге цикла
Index	Текущий индекс
Completed	После завершения работы цикла срабатывает данный узел

3 Организация курсового проектирования

Основные руководящие данные для выполнения проекта оформляются кафедрой в задании по курсовому проектированию, утверждаемом заведующим кафедрой.

Во время выдачи задания студент и руководитель проекта уточняют график индивидуальных консультаций по проекту, обмениваются адресами электронной почты.

Законченный и оформленный курсовой проект предоставляется руководителю для рецензирования. Срок составления рецензии не должен превышать семи дней. В рецензии преподаватель должен отметить каждую ошибку и неточность с указанием, в чем заключается сущность ошибки. Недопустима расстановка вопросительных и других знаков без соответствующих разъяснений. Все исправления в тексте и замечания на полях рецензируемой работы необходимо писать чернилами, отличными от чернил, которыми написана работа.

В рецензии должен быть представлен подробный анализ недостатков и ошибок, уровень соответствия, конкретно и четко сформулированы все требования, которые должен выполнить студент. Курсовой проект направляется на доработку, если количество ошибок и погрешностей позволяет отнести его к низкому уровню соответствия. Допустимые погрешности и ошибки при определении учебных достижений представлены в таблице 3.

Таблица 3 – Допустимые погрешности и ошибки при определении учебных достижений студентов

Шкала соответствия	Уровень соответствия	Балл	Количество ошибок / погрешности несущественные / существенные
Соответствие	Высокий	5	3/2/0
	Средний	4	6/3/2
	Минимально необходимый	3	7/4/3
Несоответствие	Низкий	2	8/5/4

При повторном рецензировании преподаватель должен проверить исправление его предыдущих замечаний. Указание новых замечаний не допускается. Если проект удовлетворяет требованиям, предъявляемым к нему, он допускается к защите, о чем руководитель делает надпись на чертежах и записке. Защита проекта производится специальной комиссией из 2–3 человек при непосредственном участии руководителя курсового проектирования в присутствии студентов данной группы. Защита состоит в коротком докладе студента (8…10 мин) по выполненному проекту и в ответах на вопросы. Студент должен при защите проекта дать все объяснения по существу проекта.

Погрешностями при определении учебных достижений считаются:

– неточные выражения в пояснительной записке;

– нерациональные, но правильные приемы, используемые для решения поставленных задач;

– незначительные погрешности при отображении графических объектов.

К несущественным ошибкам относятся:

– неточности определения характеристик и параметров объектов;

– нерациональный способ решения задачи или план ответа (нарушение логики изложения материала, подмена основных понятий второстепенными);

– отсутствие ссылок на использованные источники;

– несоблюдение требований ГОСТа и небрежное оформление пояснительной записки и графического материала.

К существенным ошибкам относятся:

– подмена понятий в изложении основных понятий;

– незнание фундаментальных понятий разработки игровых приложений с использованием выбранного игрового движка;

– неумение в ответе объяснить материал, делать выводы и обобщения;

– неумение письменно оформить материал;

– неумение применять теоретические знания для разработки игровых приложений;

– отсутствие полного функционала игрового приложения.

4 Содержание курсового проекта

Курсовой проект состоит из пояснительной записки и графического материала. Структура пояснительной записки приведена в таблице 4. Основными требованиями к пояснительной записке являются четкость и логическая последовательность изложения материала, убедительность аргументации, краткость и ясность формулировок. В тексте записки не должно быть общих фраз, очевидных выводов и т. п. Объем пояснительной записки – не менее 20 страниц текста.

Перечень графических материалов проекта указан в таблице 5. Рекомендуется выполнять графическую часть на листе формата А3. Объем графического материала должен быть не менее двух листов.

Таблица 4 – Структура пояснительной записки

Наименование раздела	Рекомендуемый объем, стр.
Титульный лист	1
Задание на проектирование	1
Содержание	1
Введение	1
1 Постановка задачи	1
2 Выбор и обоснование технологии разработки	3
3 Проектирование системы	5
4 Реализация системы	10
5 Результаты тестирования системы	3
Заключение	1
Список использованной литературы	1

Таблица 5 – Структура графического материала

Наименование	Формат	Количество листов
Сценарий игрового приложения	А3	1
Результаты работы игрового приложения	А3	2

Листы графической части должны содержать сценарий и результаты тестирования работы игрового приложения.

5 Оформление курсового проекта

Оформление курсового проекта должно соответствовать требованиям ГОСТ 2.105–95. Текстовая часть пояснительной записки выполняется либо чертежным шрифтом по ГОСТ 2304–81 с высотой букв не менее 5 мм либо машинным способом шрифтом Таймс с высотой букв 14 пунктов через одинарный интервал.

Все листы пояснительной записки, включая графики, схемы, таблицы, должны содержать стандартную рамку и быть пронумерованными. Титульный лист не нумеруется, но при нумерации страниц он считается первым.

При использовании научно-технических положений, определений, формул, стандартов и других данных необходимо делать ссылку на источник, указывая его номер в списке литературы. Номер источника заключается в квадратные скобки (например, ссылка на второй источник в списке литературы: [2]). Список литературы составляется либо по алфавиту, либо по мере появления ссылок в тексте пояснительной записки и оформляется в соответствии с ГОСТ 7.1–2003.

Формулы, иллюстрации и таблицы нумеруются в пределах раздела. Например, седьмая формула второго раздела нумеруется так: (2.7).

Обозначения переменных и параметров, принятых в формулах, должны быть расшифрованы сразу после написания формулы. При этом указываются единицы измерения переменных и параметров.

Рисунки, графики и таблицы сопровождаются наименованиями, отображающими их содержание (например, Рисунок 1 – Диаграмма состояний игрового приложения). Если на одном рисунке изображено несколько графиков различных процессов, то каждый график должен иметь отдельное обозначение, которое необходимо расшифровать в поясняющих данных к рисунку. Поясняющие данные помещаются под рисунком перед его наименованием.

Рисунки, графики и схемы можно помещать либо на листах, содержащих текст пояснительной записки, если они незначительны по размеру, либо на отдельных листах, которые располагаются сразу после первой ссылки на них в тексте.

Размещаемые в тексте перечисления требований, указаний, положений и т. п. следует обозначать строчными буквами со скобкой, например: а) первый элемент; б) второй элемент; и т. д., записывать с малой буквы и разделять между собой символом точки с запятой.

Графическая часть курсового проекта выполняется в соответствии с требованиями ГОСТ 19.701–90 в среде инженерной графики Visio, AutoCAD или другого программного обеспечения, согласованной с руководителем. При этом наименьшая величина высоты отдельного символа составляет 10 мм, а наименьшее расстояние между блоками – 5 мм. При этом допускается такое отклонение в соотношениях геометрической формы изображений, которое не затрудняет определения назначения блока при чтении схемы. Пояснительная записка должна помещаться в жесткую обложку для курсовой работы с титульным листом без основной надписи.

6 Разбивка этапов курсового проекта

Разбивка этапов курсового проекта, определение количества минимальных и максимальных баллов за каждый из них производится преподавателем. Примерный перечень этапов выполнения курсового проекта и количества баллов за каждый из них представлен в таблице 6.

Максимальные баллы начисляются за выполненные досрочно (точно в срок) разделы. За выполнение необязательных подразделов начисляются дополнительные баллы. Минимальные баллы начисляются при несвоевременном выполнении разделов (без соблюдения календарных сроков выполнения курсового проекта) и наличии ошибок.

Итоговая оценка курсового проекта представляет собой сумму баллов за выполнение и защиту курсового проекта и выставляется в соответствии со шкалой, приведенной в таблице 7 (по пятибалльной системе).

Таблица 6 – Разбивка этапов курсового проекта

Этап выполнения	Минимум	Максимум
Теоретические исследования проблемы, постановка задачи	9	15
Практические исследования	9	15
Разработка программного обеспечения	9	15
Оформление пояснительной записки	9	15
Итого за выполнение курсовой работы	36	60
Защита курсовой работы	15	40

Таблица 9 – Разбивка этапов курсового проекта

Оценка	Отлично	Хорошо	Удовлетворительно	Неудовлетворительно
Баллы	87–100	65–86	51–64	0–50

7 Методические указания к выполнению курсового проекта

7.1 Оформление содержания

Содержание пояснительной записки размещается сразу после бланка задания на курсовой проект. Оно включает наименования разделов и подразделов пояснительной записки с указанием номера страницы. Сквозная нумерация ее листов выполняется с титульного листа (на нем номер не ставится). Содержание фактически расположено на третьей странице пояснительной записки. При оформлении электронного варианта текста рекомендуется использовать иерархическую структуру заголовков, автоматическую расстановку номеров страниц и вставку автосодержания.

7.2 Оформление введения

Введение оформляется на отдельной странице. Объем его не должен превышать четыре процента от общего состава пояснительной записки. В тексте введения описывается, к какой области относится данная работа и на основе какого документа производится ее выполнение.

7.3 Постановка задачи

Формулируя постановку задачи курсового проекта, необходимо определить её цель и задачи. Здесь следует описать системы, которые будут реализованы в игровом приложении, а также требования к этим системам. Данные требования должны быть выставлены с учетом целевой платформы разрабатываемого приложения. Далее следует описать тип и стиль графики, которая будет использоваться в курсовой работе, а также требования к пользовательскому интерфейсу.

7.4 Выбор и обоснование технологии разработки

В разделе необходимо произвести анализ существующих технологий разработки игровых приложений, создания 2D-спрайтов и 3D-моделей. Далее следует описать преимущества и недостатки каждой технологии, их направленности и ограничения. Обоснование выбора технологии следует составлять на основе поставленной задачи. Для реализации игрового приложения необходимо использовать игровой движок Unity и язык программирования C# либо игровой движок Unreal Engine версии 4 или 5 с использованием языков программирования Blueprints и C++. В программном модуле должны быть реализованы все системы, выбранные при постановке задачи.

Для создания 2D-спрайтов допускается использовать любой графический редактор.

Для создания 3D-моделей необходимо использовать редакторы трехмерной графики Autodesk Maya, 3ds Max или Blender.

7.5 Проектирование системы

Раздел посвящается разработке алгоритмов функционирования игрового приложения. Здесь описывается сценарий игры, логика ее прохождения. Следует описать, какие действия может совершать пользователь в игровом приложении, какие цели ставятся перед пользователем и условия достижения конечного результата. В данном пункте составляются диаграммы состояний игрового приложения. Должны быть представлены рисунки с прототипированием пользовательского интерфейса. Необходимо составить и описать требования к графическим объектам игрового приложения, их качеству и уровню детализации.

7.6 Реализация системы

В разделе необходимо описать сущности, разработанные при создании игрового приложения.

Следует описать поэтапное создание 3D-моделей и 2D-спрайтов, а также представить иллюстрации конечного результата. Создание 3D-моделей происходит в два этапа: создание форм и текстурирование. В описании этапов следует указать, какие технологии применялись для создания моделей и текстур.

Необходимо описать создание игровой локации, настройка освещения и постобработка. При описании следует указать, какие объекты использовались для настройки освещения, какие характеристики источников освещения и постобработки присутствуют в игровом приложении.

Далее необходимо описать разработанные классы игрового приложения с листингом или иллюстрациями функций и методов. Каждый разработанный метод должен содержать его задачу и пояснение работы.

Также следует описать разработанный пользовательский интерфейс, его функции и задачи.

7.7 Результаты тестирования

В разделе приводится описание результатов, полученных при выполнении программы, с иллюстрациями работы игрового приложения. Необходимо представить иллюстрации игрового процесса с пояснениями, результаты прохождения игрового приложения, а также работу пользовательского интерфейса. Результатом успешного тестирования является выявление ошибок, полученных на этапе разработки, и их исправление.

7.8 Оформление заключения

В заключении необходимо сформулировать и проанализировать полученные результаты в ходе выполнения курсовой работы. Объем заключения не должен превышать четырех процентов от общего состава пояснительной записки.

7.9 Оформление списка используемых источников

Оформление списка источников литературы, используемых при выполнении курсового проекта, производится в соответствии с требованиями ГОСТ 7.1–2003 *Библиографическое описание документа. Общие требования и правила составления*.

Список литературы

1 **Куксон, А.** Разработка игр на Unreal Engine 4 за 24 часа / А. Куксон, Р. Даулингсока, К. Крамплер. – Москва: Эксмо, 2019. – 528 с.

2 **Загарских, А. С.** Введение в разработку компьютерных игр / А. С. Загарских, А. А. Хорошавин, Э. Э. Александров. – Санкт-Петербург: ИТМО, 2019. – 79 с.

3 **Хокинг, Дж.** Unity в действии. Мультиплатформенная разработка на C# / Дж. Хокинг. – 2-е изд. – Санкт-Петербург: Питер, 2019. – 352 с.

4 **ГОСТ 2.105–95.** Единая система конструкторской документации. Общие требования к текстовым документам. – Взамен ГОСТ 2.105–79, ГОСТ 2.906–71; введ. с 01.07.96. – Москва: Изд-во стандартов, 2006. – 29 с.

5 **ГОСТ 7601–78.** Физическая оптика. Термины, буквенные обозначения и определения основных величин. – Введ. 01.01.80. – Москва: Изд-во стандартов, 2006. – 18 с.

6 **ГОСТ 14686–69.** Средства измерения световых величин. Термины. – Введ. 01.01.70. – Москва: Изд-во стандартов, 1976. – 7 с.

7 **ГОСТ 7.1–2003.** Библиографическое описание документа. Общие требования и правила составления. – Взамен ГОСТ 7.1–84, ГОСТ 7.16–79, ГОСТ 7.18–79, ГОСТ 7.34–81, ГОСТ 7.40–82; введ. 01.07.2004. – Москва: Изд-во стандартов, 2004. – 169 с.

8 **ГОСТ 19.701–90.** Единая система программной документации. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения. – Введ. 01.01.92. – Москва: Изд-во стандартов, 2010. – 23 с.