МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Автоматизированные системы управления»

ПРОГРАММИРОВАНИЕ ГРАФИЧЕСКИХ ПРИЛОЖЕНИЙ

Методические рекомендации к лабораторным работам для студентов специальности 6-05-0612-03 «Системы управления информацией» очной и заочной форм обучения

Могилев 2026

УДК 004.42
ББК 32.973-018
П78

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Автоматизированные системы управления» «16» декабря 2025 г., протокол № 5

Составители: д-р техн. наук, доц. А. И. Якимов;
ст. преподаватель Н. П. Скрылев

Рецензент канд. техн. наук, доц. В. В. Кутузов

Методические рекомендации к лабораторным работам по дисциплине «Программирование графических приложений» предназначены для студентов специальности 6-05-0612-03 «Системы управления информацией» очной и заочной форм обучения.

Учебное издание

ПРОГРАММИРОВАНИЕ ГРАФИЧЕСКИХ ПРИЛОЖЕНИЙ

Ответственный за выпуск	А. И. Якимов
Корректор	И. В. Голубцова
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 21 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

Введение

Целью преподавания дисциплины «Программирование графических приложений» является получение студентами навыков, необходимых для разработки графических приложений с использованием игрового движка Unreal Engine.

Цель методических рекомендаций – помочь студентам в самостоятельной подготовке и выполнений задания к лабораторным занятиям по дисциплине.

Порядок выполнения работы.

1 Изучить основные теоретические положения, сделав необходимые выписки в конспект.

2 Получить задание у преподавателя, выполнить типовые задания.

3 Оформить отчет.

Требования к отчету.

1 Цель работы.

2 Постановка задачи.

3 Результаты исследования.

4 Выводы.

1 Лабораторная работа № 1. Установка и настройка Unreal Engine и необходимых инструментов

Цель работы: изучение графических приложений, основных принципов работы Unreal Engine, его возможностей и архитектуры проекта.

Основные теоретические положения

Unreal Engine 4 (UE4) – это игровой движок и редактор, разработанный компанией Epic Games для создания игр и приложений. Чтобы установить UE4, необходимо создать новый аккаунт (учетную запись Epic) на сайте Epic Games, загрузить и установить Epic Games Launcher, в котором можно будет загрузить сам Unreal Engine.

При нажатии на ярлык Unreal Engine 4 откроется меню с предложением создать проект определенного типа. Выберите Games и нажмите кнопку Next (рисунок 1.1).

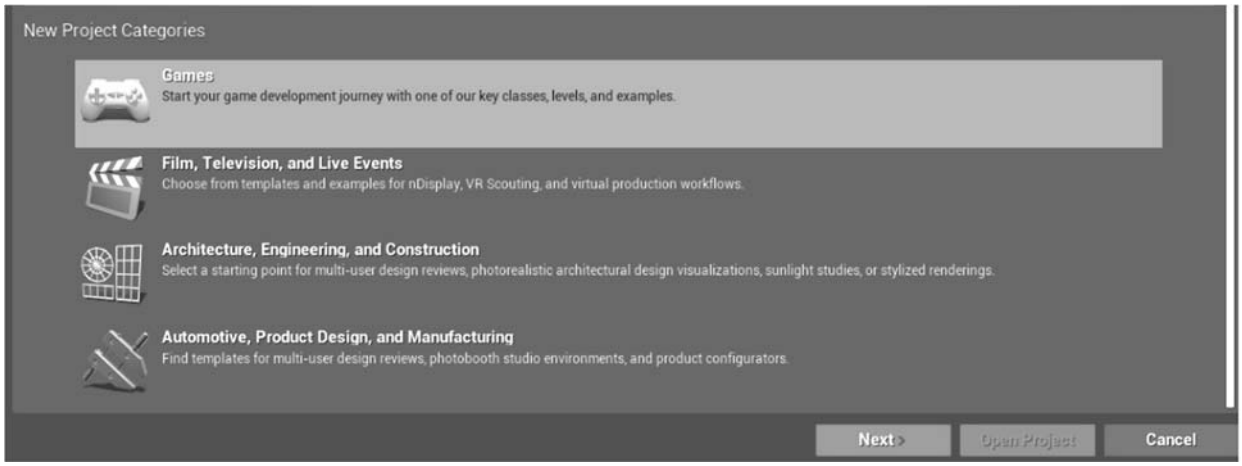

Рисунок 1.1 – Меню выбора типа проекта

Следующим шагом является выбор шаблона проекта. Темплейтом (шаблоном) является предустановленный набор ассетов (классов и компонентов), который будет встроен в проект. Для игр темплейты разделены на игровые жанры. Темплейт Blank означает, что будет создан «пустой» проект без предустановленных ассетов. Выберите его и нажмите кнопку Next (рисунок 1.2).

Последним шагом является выбор настроек проекта. Все настройки вы сможете изменить в ходе разработке игры, поэтому на данном этапе можно оставить все, как есть (рисунок 1.3).

После создания проекта он автоматически откроется. Окно Unreal Engine представлено на рисунке 1.4.

Окно **Viewport** – это окно, через которое вы редактируете сцену. Для того чтобы вращать камеру, необходимо удерживать **ПКМ** в окне Viewport, перемещая её. Для перемещения по уровню нужно зажать **ПКМ** и использовать клавиши **WASD**.

Рисунок 1.2 – Меню выбора шаблона

Рисунок 1.3 – Окно выбора настроек проекта

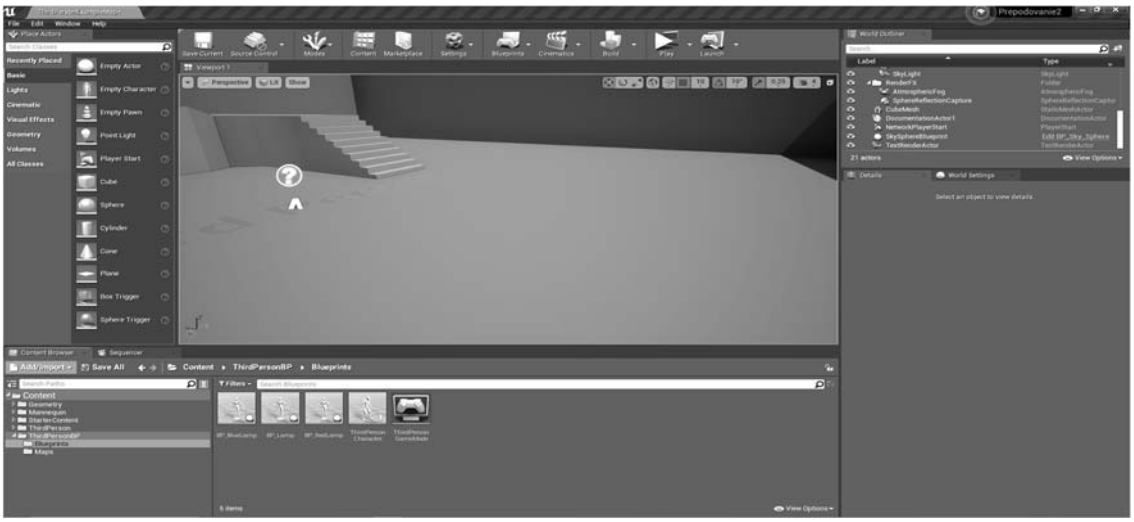

Рисунок 1.4 – Окно UE4 после запуска проекта

В окне **World Outliner** отображаются все объекты на текущем уровне. Вы можете упорядочить список, распределив связанные объекты по папкам, а также искать и фильтровать их по типам.

В окне **Details** отображаются все свойства выбранного объекта. Эта панель используется для изменения параметров объекта. Внесённые изменения повлияют только на выбранный экземпляр.

Окно **Place Actors** позволяет располагать на уровне различные типы объектов, такие как 3D-модели, источники освещения и камеры.

Content Browser – в этом окне отображаются все файлы проекта. Его можно использовать для создания папок и упорядочивания файлов. Здесь также можно выполнять поиск по файлам с помощью поисковой строки или фильтров.

Размещение объектов на сцене

Для того чтобы разместить объект на сцене, необходимо зажать на нужном объекте и вытащить его прямо в окно Viewport в то место, где вы хотите разместить объект. Так же можно сделать и с объектами из Content Browser (рисунок 1.5).

Рисунок 1.5 – Размещение объектов на сцене

Объекты на сцене можно редактировать, полностью изменяя их характеристики. Основными характеристиками любого объекта являются положение в пространстве, поворот и масштаб (Location, Rotation и Scale соответственно). Изменять эти характеристики можно при нажатии на объект в окне деталей, а также непосредственно в окне вьюпорта. При выборе объекта и нажатии на клавиши W, E и R будет происходить переключение режимов редактирования (рисунок 1.6).

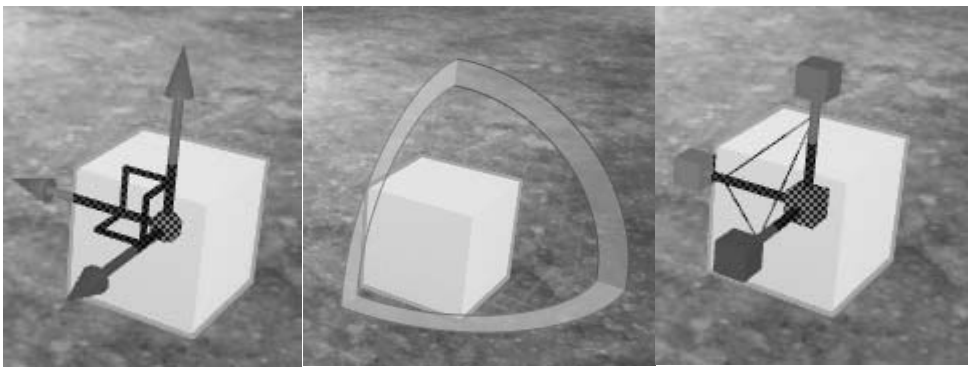

Рисунок 1.6 – Переключение режимов редактирования

Практические задания

Задание

Создать проект на основе одного из шаблонов. Составить композицию из объектов, находящихся во вкладках Basic (Cube, Sphere, Cylinder, Cone, Plane) и Lights (Point Light, Spot Light) окна Place Actor. Для составления композиции из 3D-объектов необходимо использовать масштабирование, поворот и изменение расположения, а для источников освещения – изменить их интенсивность и цвет.

Контрольные вопросы

1 Какова минимальная версия операционной системы для установки Unreal Engine?

2 Какие системные требования должны быть выполнены для оптимальной работы Unreal Engine?

3 Как загрузить Unreal Engine через Epic Games Launcher? Опишите последовательность действий.

4 В чем разница между руководством по установке и документацией по Unreal Engine?

5 Как настроить проект в Unreal Engine при его создании? Какие настройки наиболее критически важны?

6 Какие инструменты и плагины рекомендуется установить вместе с Unreal Engine для разработки?

7 Как проверить корректность установки Unreal Engine после завершения процесса?

8 Что такое Marketplace в Unreal Engine и как он может быть полезен разработчику?

9 Как выполнить обновление Unreal Engine до последней версии через Epic Games Launcher?

10 Каковы основные этапы, необходимые для настройки среды разработки, включая IDE и инструменты сборки?

2 Лабораторная работа № 2. Создание базового проекта и игрового пространства

Цель работы: изучение структуры проекта в Unreal Engine, включая объекты и компоненты, а также основных принципов работы с ними и взаимодействия между ними.

Основные теоретические сведения

Каждый игровой движок имеет скриптовый язык, позволяющий разработчикам добавлять и изменять функционал их игр. Некоторые движки используют существующую скриптовую среду, например LUA. Другие же создают собственную. UE4 предоставляет два способа создания контента: язык C++ и блюпринт (Blueprint). Система визуального программирования блюпринтов – это мощная и полнофункциональная среда разработки скриптов, работа с которой осуществляется посредством редактора. Она предоставляет художникам и дизайнерам возможность создавать полноценные игры, прототипировать идеи и изменять существующие элементы геймплея.

Blueprints

В UE4 существует пять типов блюпринтов.

Блюпринт уровня (Level Blueprint). Этот блюпринт используется для управления глобальными событиями на уровне. На одном уровне может быть только один блюпринт уровня. Он автоматически сохраняется при сохранении уровня.

Блюпринт-класс (Blueprint class). Это производный класс от другого существующего класса, созданного с помощью C++ или другого блюпринт-класса. Он используется для кодирования функционала актеров, помещенных на уровень.

Data-Only блюпринт (Data-Only Blueprint). Этот блюпринт хранит только измененные свойства унаследованного блюпринта.

Блюпринт-интерфейс (Blueprint Interface, BPI). Блюпринт-интерфейсы используются для хранения коллекции определенных пользователем функций, которые могут быть назначены другим блюпринтам. Блюпринт-интерфейсы позволяют другим блюпринтам обмениваться данными друг с другом.

Блюпринт-макрос (Blueprint Macros). Блюпринт-макросы – независимые графы часто используемых последовательностей нодов, которые могут многократно использоваться другими блюпринтами. Блюпринт-макросы хранятся в библиотеке макро-блюпринтов (Blueprint Macro Library).

Блюпринты уровней и блюпринт-классы – два наиболее часто используемых типа блюпринтов.

Интерфейс редактора блюпринтов

Чтобы открыть блюпринт уровня и увидеть интерфейс редактора (Blueprint Editor), выберите на панели инструментов редактора уровней пункт Blueprints ⇒ Open Level Blueprint (рисунок 2.1).

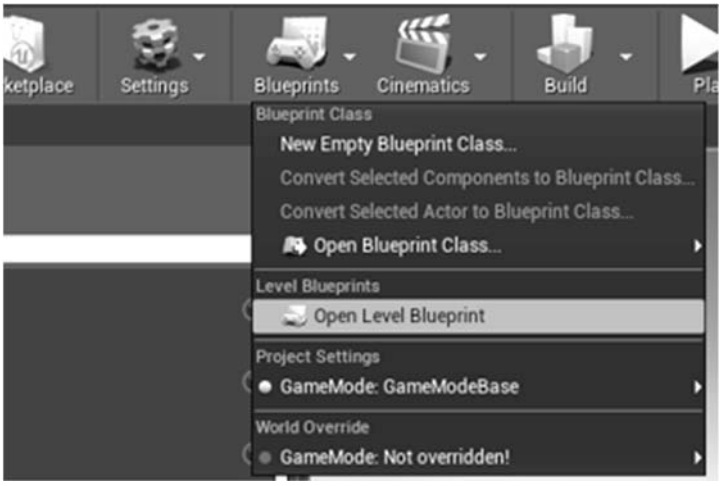

Рисунок 2.1 – Открытие блюпринта уровня

Интерфейс редактора блюпринтов содержит строку меню, панель инструментов для быстрого доступа к основным инструментам и операциям, панель Event Graph для проектирования скриптов, панель Details для отображения свойств, выбранных в редакторе блюпринта объектов, и панель My Blueprint, которая используется для управления и отслеживания графов нодов, функций, макросов и переменных, используемых в выбранном блюпринте. Возможности редактора блюпринтов выделены на рисунке 2.2 и описаны далее.

Панель инструментов. Панель инструментов содержит кнопки (с коротким описанием) для управления редактором блюпринтов.

Панель My Blueprint. Панель My Blueprint отслеживает все графы нодов, функции, макросы и переменные, которые используются в блюпринте. Каждая категория разделена заголовком, и справа от каждого заголовка есть символ «+», по которому можно щелкнуть, чтобы добавить элемент при необходимости. Вы можете использовать панель My Blueprint, чтобы добавить, переименовать или удалить все эти элементы.

Панель Details. Вы можете управлять свойствами добавленных в блюпринт компонентов, переменных и функций с помощью панели Details.

Панель Event Graph (граф событий). Панель Event Graph – это граф нодов по умолчанию, который используется для кодирования блюпринтов. Event Graph – это та область, в которой производится значительная часть работы при использовании редактора блюпринтов.

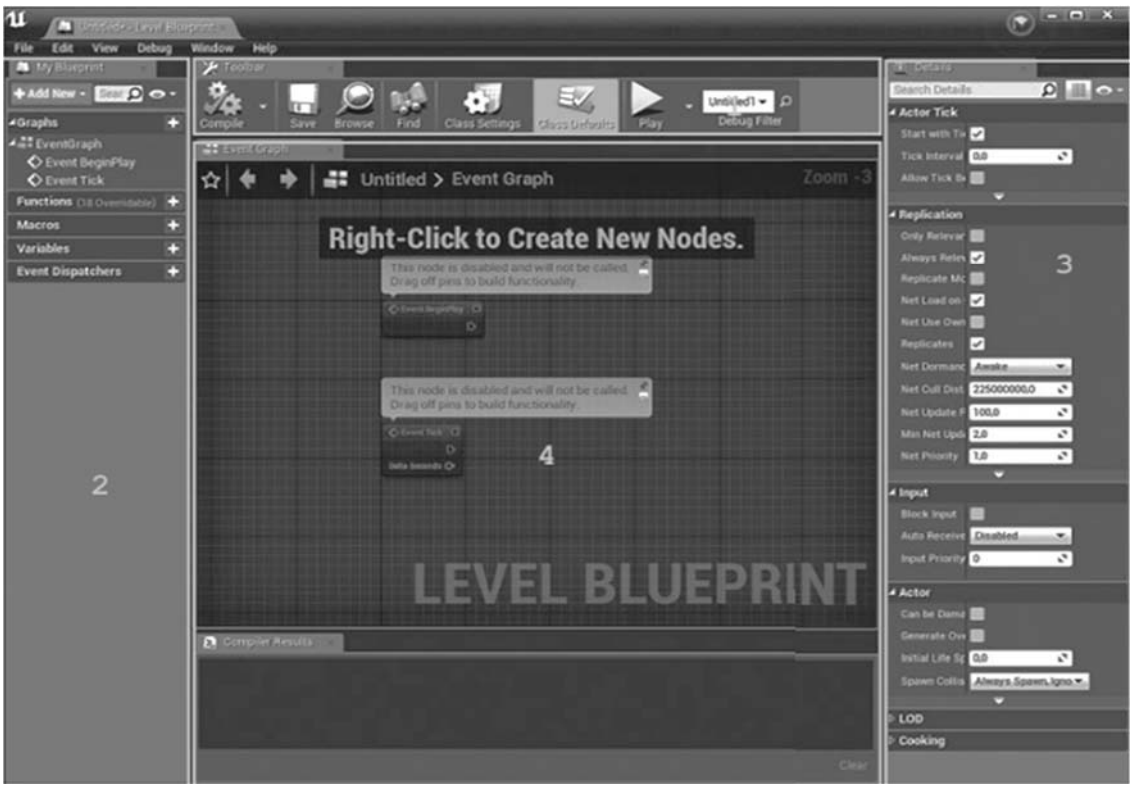

Рисунок 2.2 – Окно редактора блюпринта

Пример – Даны два числа A и B. Необходимо рассчитать их сумму, разницу и произведение. Результаты вывести на экран.

Для выполнения данного задания необходимо открыть Level Blueprint и добавить событие Event BeginPlay (в некоторых версиях UE оно уже добавлено в Event Graph по умолчанию). Для того чтобы добавить событие или функцию на Event Graph, необходимо вызвать контекстное меню поиска, нажав ПКМ в пустом месте полотна, и ввести в поиск название события или функции.

Создаем две новые переменные и указываем им в окне Details тип Integer (рисунок 2.3), а после перетаскиваем на Event Graph, выбирая пункт Get. После компиляции переменным можно указать значение по умолчанию в окне деталей.

Рисунок 2.3 – Создание переменных

Для того чтобы сложить две переменные, необходимо вызвать функцию сложения. Для этого достаточно ввести «+» в строку поиска и выбрать подходящее сложение (рисунок 2.4).

Далее необходимо в ее входные пины подключить уже добавленные переменные. Вывод сообщения на экран происходит с помощью функции Print String, которое необходимо добавить в Event Graph и соединить с событием Event

BeginPlay, чтобы вывести сообщение на экран после запуска уровня. Выходной пин функции умножения необходимо соединить с входным пином In String функции вывода на экран.

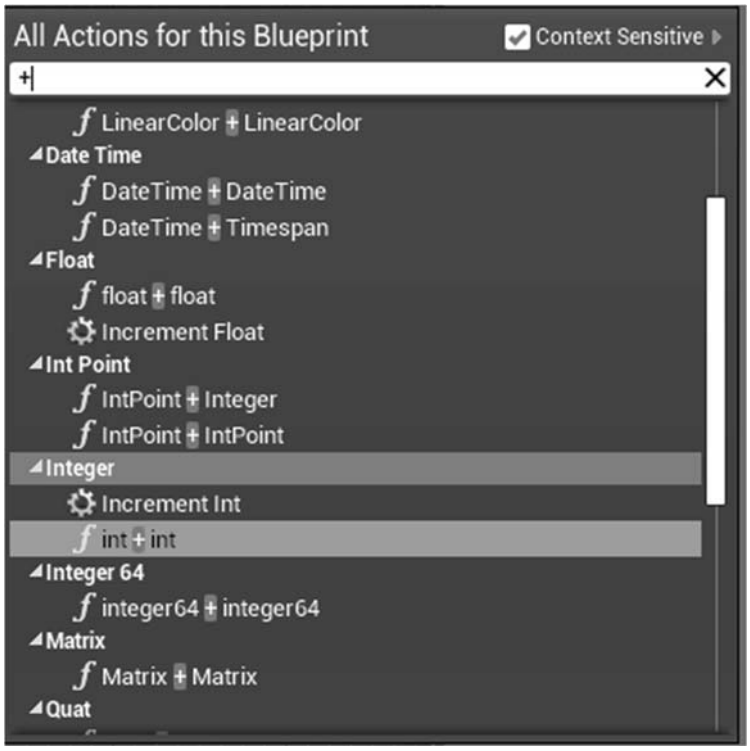

Рисунок 2.4 – Вызов функции сложения

Операция сложения двух переменных с последующим выводом результата на экран представлена на рисунке 2.5.

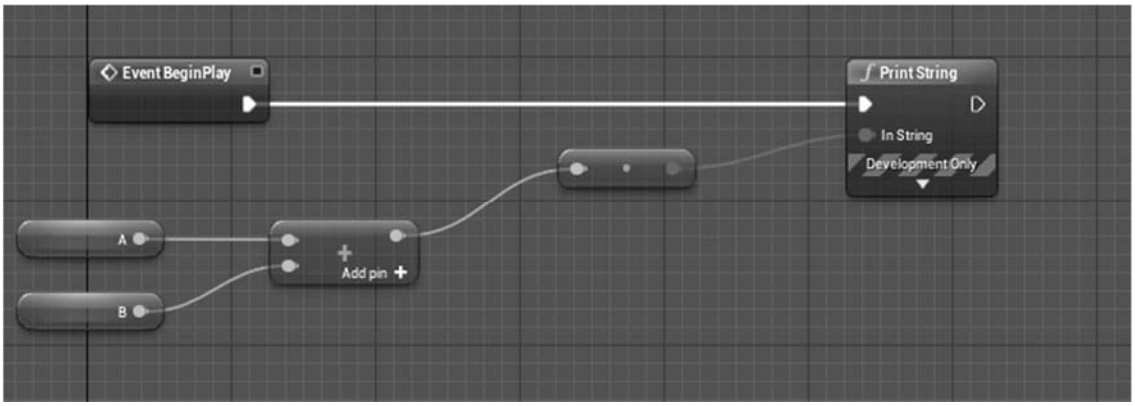

Рисунок 2.5 – Операция сложения

После запуска уровня результат сложения будет показан в левом верхнем углу вьюпорта.

Разницу и умножение двух переменных реализуем таким же образом (рисунок 2.6). Обратите внимание, что выходные пины можно использовать несколько раз, не добавляя в граф каждый раз ноды переменных.

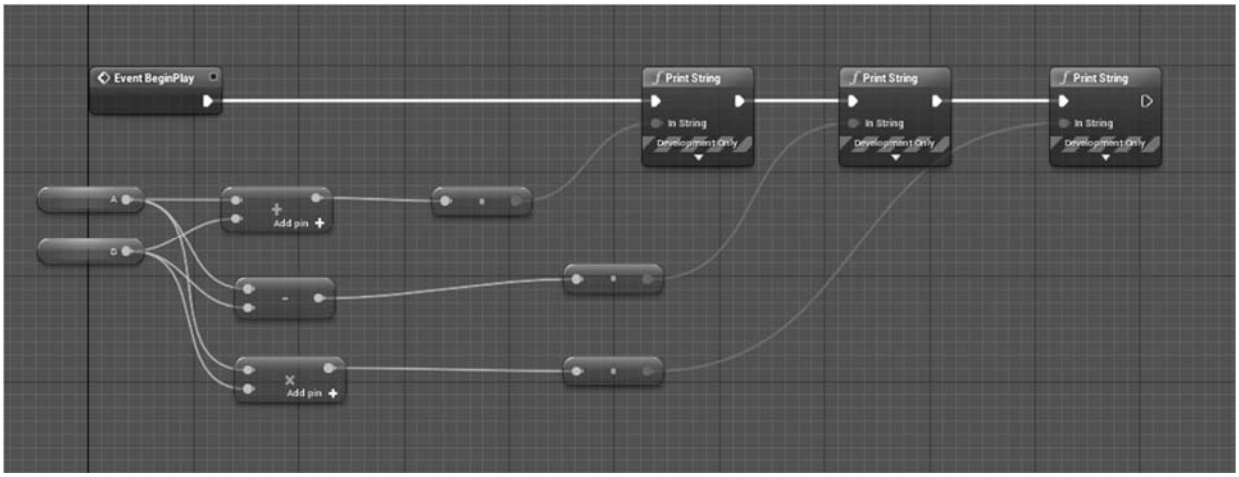

Рисунок 2.6 – Операции сложения, вычитания и умножения

Практические задания

Все задания необходимо выполнять в Level Blueprints в событии Begin Play. Вывод на экран производится функцией Print String. Переменные A, B и C должны быть типа Float. Числа выбираются произвольно. В задании 4 в массиве должны присутствовать отрицательные, положительные числа и 0.

Задание 1

Даны два числа A и B. Рассчитать C = (A + B) * (A – B).

Даны два числа A, B. Если A > B, то рассчитать C = A * B. Если нет – C = A.

Написать функцию, которая рассчитывает сумму и произведение двух чисел, принимая на вход эти два числа и отдавая на выход два рассчитанных.

Дан массив чисел из пяти элементов. Вывести на экран числа, которые больше 0.

Задание 2

Даны два числа A и B. Рассчитать C = (A + B) * (A – B).

Даны два числа A, B. Если A > B, то рассчитать C = A * B. Если нет – C = A.

Написать функцию, которая рассчитывает сумму и произведение двух чисел, принимая на вход эти два числа и отдавая на выход два рассчитанных.

Дан массив чисел из пяти элементов. Вывести на экран числа, которые больше 0.

Задание 3

Даны два числа A и B. Рассчитать C = A + B – (A * B).

Даны два числа A, B. Если A>B, то рассчитать C = 3 * A + B. Если нет – C = A – 2 * B.

Написать функцию, которая рассчитывает сумму и разность двух чисел, принимая на вход эти два числа и отдавая на выход два рассчитанных.

Дан массив чисел из семи элементов. Вывести на экран, есть ли в массиве 0.

Задание 4

Даны два числа A и B. Рассчитать C = B ^ 2 – 2 * A.

Даны два числа A, B. Если A > B, то рассчитать C = A + B. Если нет – C = B – A.

Написать функцию, которая рассчитывает сумму и частное двух чисел, принимая на вход эти два числа и отдавая на выход два рассчитанных.

Дан массив чисел из четырёх элементов. Вывести на экран числа, которые меньше 0.

Задание 5

Даны два числа A и B. Рассчитать C = (A – B) * B.

Даны два числа A, B. Если A > B, то рассчитать C = B. Если нет – C = A – B.

Написать функцию, которая рассчитывает разность и произведение двух чисел, принимая на вход эти два числа и отдавая на выход два рассчитанных.

Дан массив чисел из шести элементов. Вывести на экран кол-во положительных элементов.

Контрольные вопросы

1 Как создать новый проект в Unreal Engine? Перечислите шаги необходимой последовательности.

2 Каковы основные настройки, которые можно выбрать при создании нового проекта (например, шаблон, настройки модуля)?

3 Что такое уровень (level) в Unreal Engine и как его создать?

4 Как использовать инструмент Terrain для создания ландшафта в проекте?

5 Какие основные компоненты можно добавить в игровое пространство и как это делается?

6 Как импортировать 3D-модели и текстуры из внешних источников в проект Unreal Engine?

7 Что такое Blueprint и как он используется для создания логики в игровом пространстве?

8 Как настроить освещение в уровне для создания реалистичного игрового мира?

9 Как сохранить и протестировать изменения, внесённые в ваш проект и уровень?

10 Какие функции предоставляет панель деталей (Details Panel) и как она может помочь в настройке объектов на уровне?

3 Лабораторная работа № 3. Добавление и управление объектами в проекте

Цель работы: изучение процесса создания и управления объектами, включая создание, удаление, перемещение и манипуляцию с параметрами объектов.

Основные теоретические положения

Blueprint Class

Несмотря на то, что блупринты уровней удобны для создания цепочек событий, они связаны с уровнем, над которым вы в данный момент работаете. С другой стороны, блупринт-классы позволяют создать новые актеры, которые можно многократно использовать на любом уровне. Многократное использование ускоряет разработку, поскольку вам необходимо только один раз разработать функционал блупринт-класса, после чего вы сможете многократно использовать его на любом уровне.

Чтобы создать новый блупринт-класс, необходимо в пустом месте Content Browser нажать ПКМ и выбрать соответствующий вариант, как показано на рисунке 3.1.

Рисунок 3.1 – Создание нового блупринт-класса

В результате откроется окно с выбором родительского класса для созданного блупринта. Основной класс для всех объектов, размещаемых на сцене, – это Actor.

Окно редактора блупринт-класса (рисунок 3.2), в отличие от блупринта уровня, содержит в себе дополнительно панели Components, Viewport и Construction Script.

Панель Components. Содержит список всех компонентов блупринта и

используется для управления ими.

Панель Viewport. Вьюпорт отображает компоненты блупринта и используется для установки пространственных отношений компонента актера.

Construction Script. Скрипт конструирования – это уникальная функция, которая выполняется, когда экземпляр блупринта (актер) помещен на уровень.

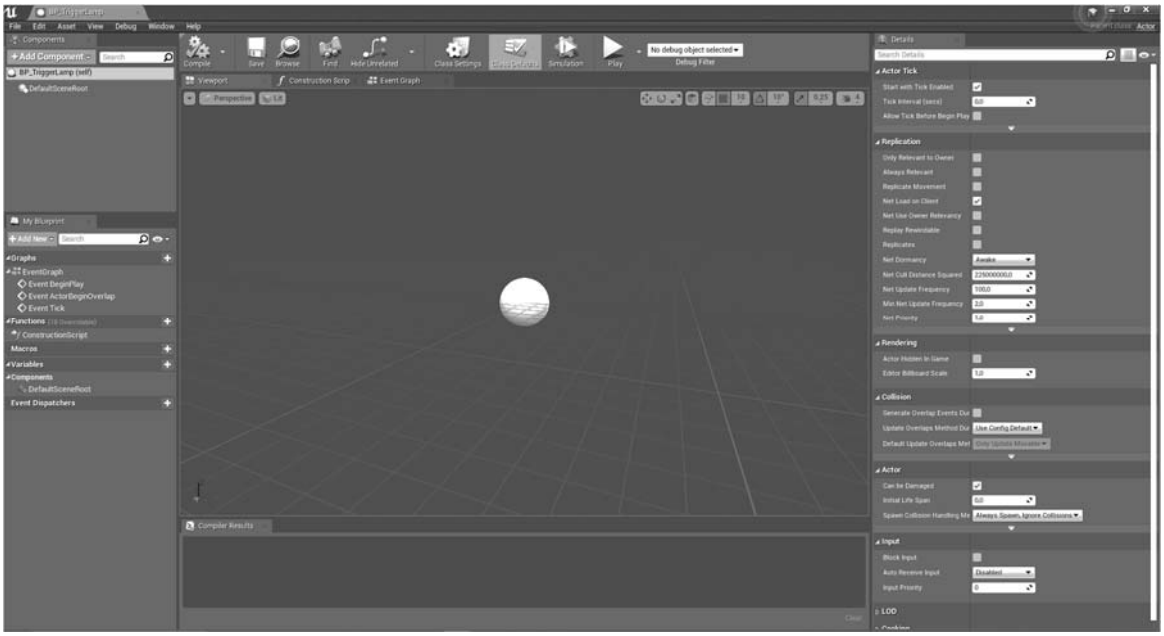

Рисунок 3.2 – Окно редактора блупринт-класса

Работа с компонентами

Для того чтобы добавить компонент в созданный блупринт, необходимо нажать на кнопку Add Component в окне Components и выбрать соответствующий тип компонента. Когда компонент добавлен в блупринт, появляется возможность редактировать его свойства в окне деталей, а также изменять его положение, поворот и масштаб во вьюпорте.

Программирование блупринтов

Когда вы производите программирование блупринта, у вас есть функции, имеющие своей целью весь актер, и функции, целью которых являются отдельные компоненты актера. Например, изменение положения актера на сцене и изменение положения его компонентов (рисунок 3.3).

Первая функция Set Actor Location имеет целью (пин Target) Self, что значит она применится для всего блупринта и переместит весь объект в заданные координаты. Вторая функция Set World Location требует, чтобы значение Target указывало на конкретный компонент блупринта, и она переместит только компонент Cube, а все остальные компоненты этого блупринта останутся на своих местах.

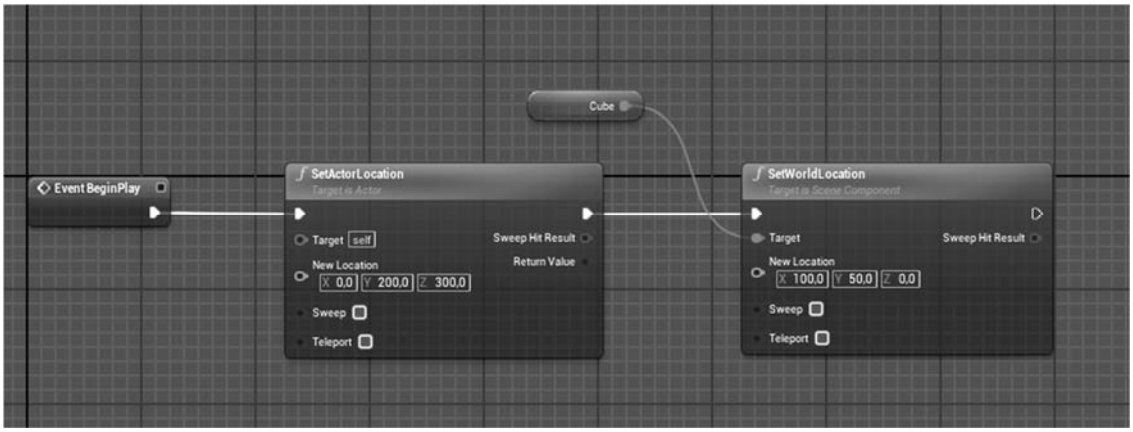

Рисунок 3.3 – Изменение положения объекта и его компонента

Пример – Необходимо создать новый блупринт-класс BP_Trigger с двумя компонентами: Box Collision и Point Light. При попадании персонажа в область триггера лампочка должна загореться, а при выходе – опять погаснуть.

Для удобства создадим проект на основе шаблона ThirdPerson, чтобы был уже существующий персонаж, который будет взаимодействовать с объектами на сцене.

Создадим новый блупринт-класс BP_Trigger и добавим нужные компоненты. Для этого во вкладке Components необходимо нажать на кнопку Add Components и ввести в поиске названия нужных компонентов Box Collision и Point Light. В окне деталей изменим масштаб компонента Box Collision, а компоненту Point Light поставим значение интенсивности на 0 (рисунок 3.4).

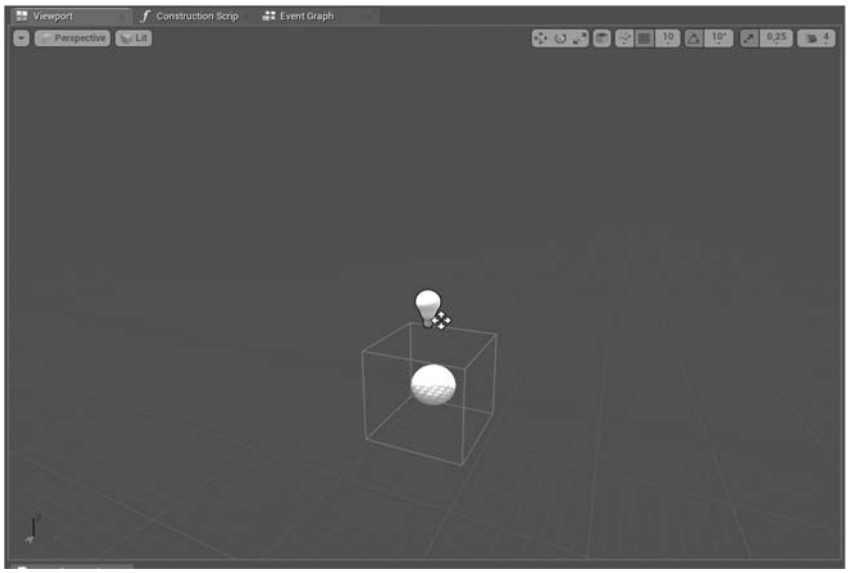

Рисунок 3.4 – BP_Trigger

Также у компонента Box Collision можно выключить свойство Hidden In Game для того, чтобы области триггера были видны в игре (по умолчанию он невидим). Это делается для облегчения тестирования работы логики триггера. После этого можно добавить BP_Trigger на сцену (рисунок 3.5).

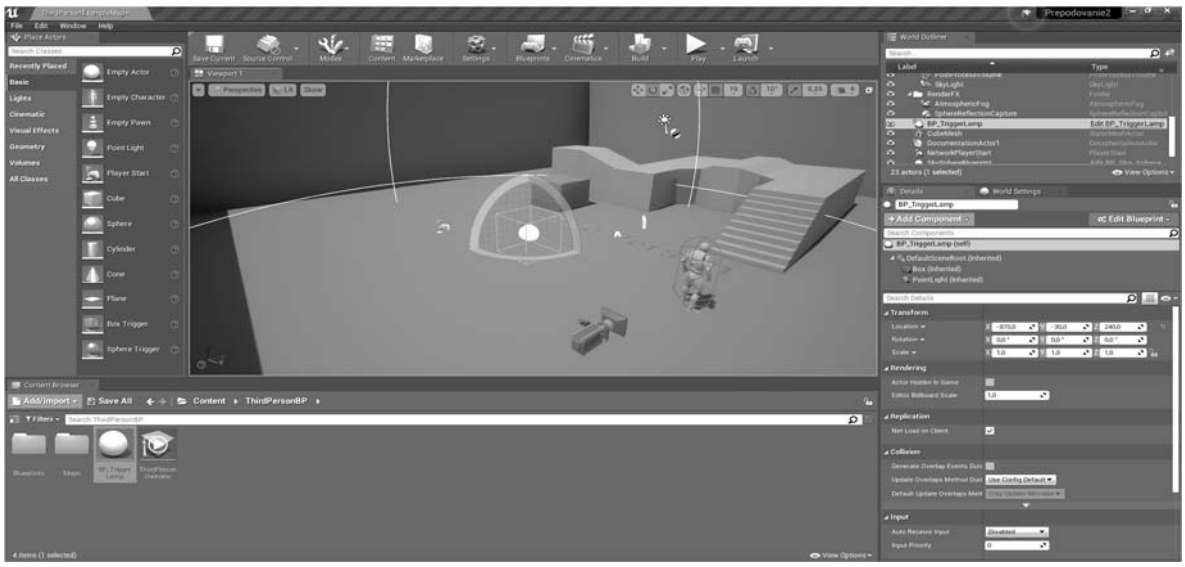

Рисунок 3.5 – Расположение объекта на сцене

После добавления блупринта на сцену можно вернуться к логике его работы. Для этого откроем Event Graph созданного блупринта. Обратите внимание, что в окне MyBlueprints появились переменные-ссылки на созданные ранее компоненты Box Collision и Point Light, с помощью которых мы можем редактировать их свойства в Event Graph. Если выбрать переменную Box, в окне деталей появится определенный набор событий, которые можно добавить в Event Graph. Для нашего примера добавим On Component Begin Overlap и On Component End Overlap (рисунок 3.6). Событие On Component Begin Overlap сработает автоматически при попадании в область Box Collision любого объекта на сцене, а событие On Component End Overlap при выходе из области соответственно.

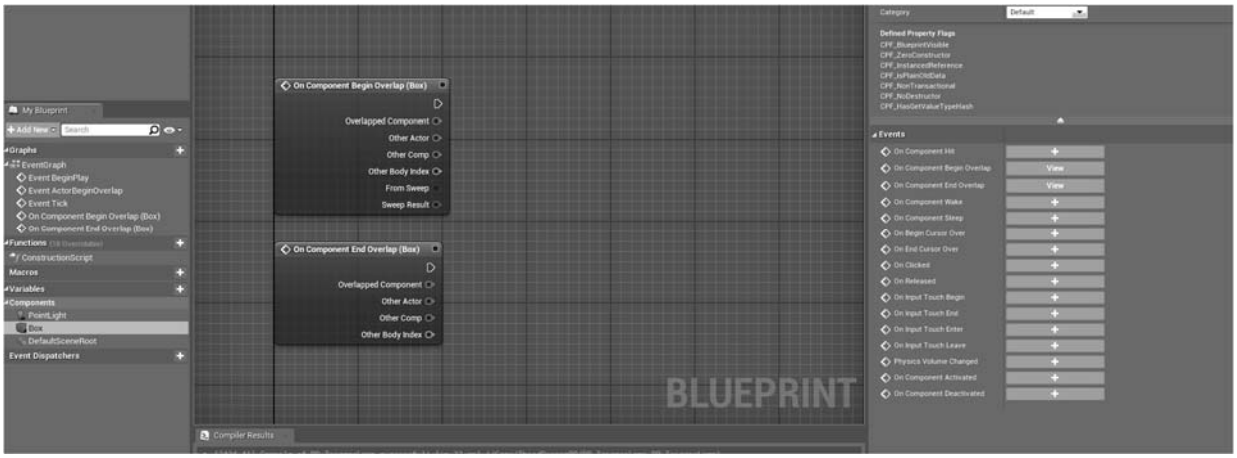

Рисунок 3.6 – Добавление событий для компонента Box Collision

Последнее, что осталось, – это добавить функции для изменения интенсивности лампочки и соединить их с соответствующими событиями. Для этого вызовем функцию Set Intensity (рисунок 3.7).

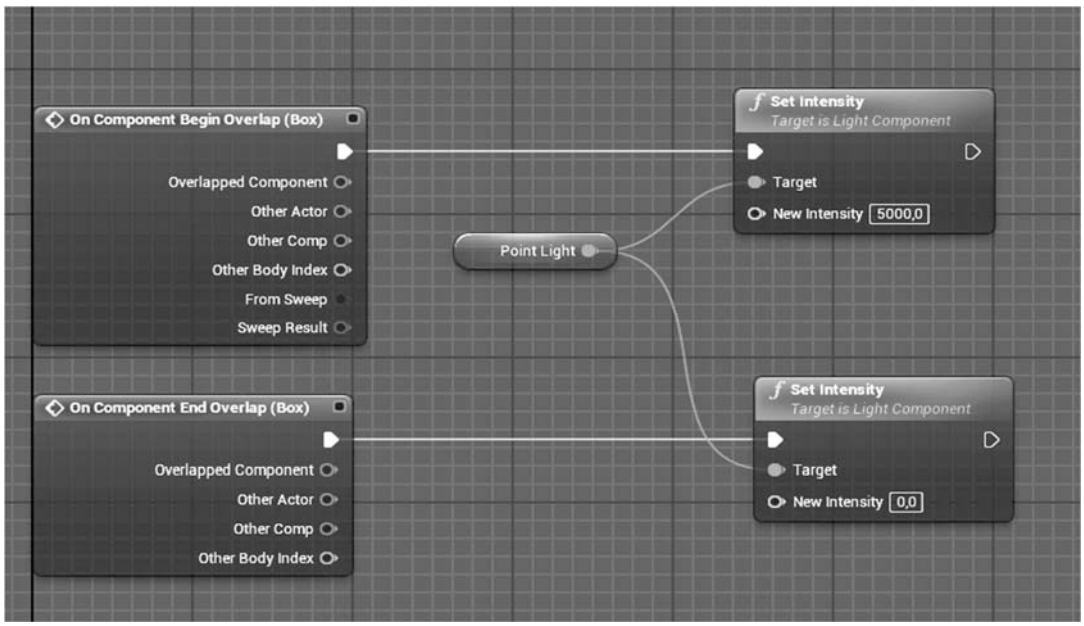

Рисунок 3.7 – Логика BP_Trigger

При запуске уровня и попадании персонажа в область триггера лампочка изменит свою интенсивность на 5000, а при выходе персонажа на 0, как показано на рисунке 3.8.

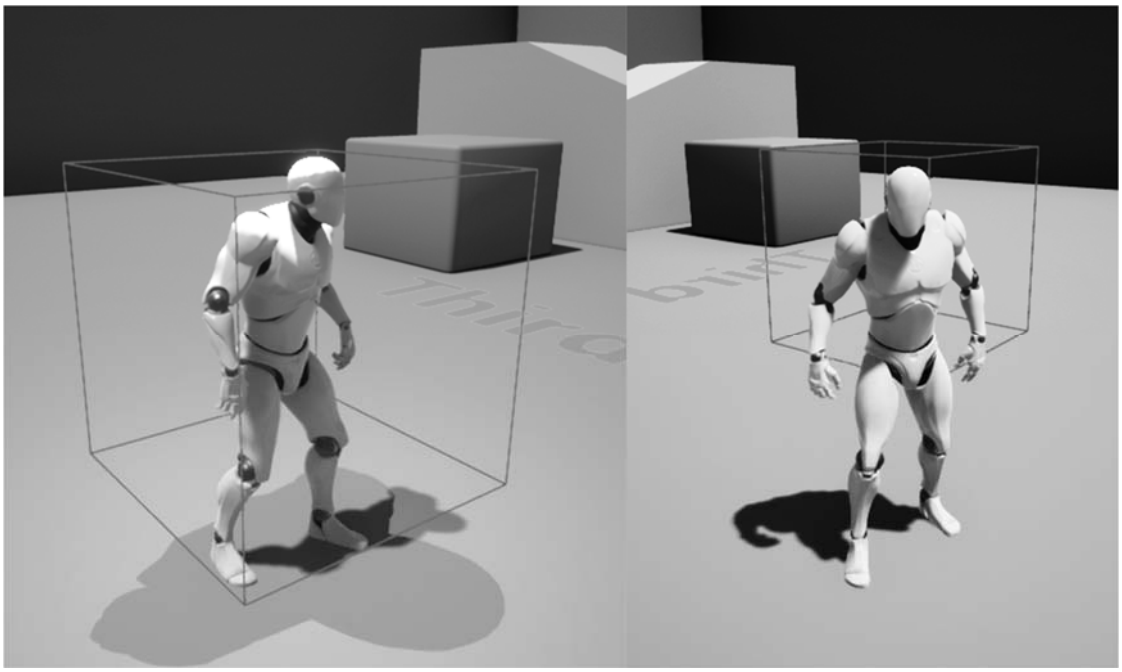

Рисунок 3.8 – Пример работы триггера

Однако стоит заметить, что этот триггер сработает при попадании любого объекта в область. Если же необходимо сделать так, чтобы только персонаж провоцировал работу событий, приходит на помощь функция Cast To.

Cast To – это функция, приводящая актера к определенному классу. При добавлении этой функции необходимо указать, к какому конкретно классу

должно быть осуществлено приведение типа. В случае с персонажем это класс ThirdPersonCharacter (рисунок 3.9).

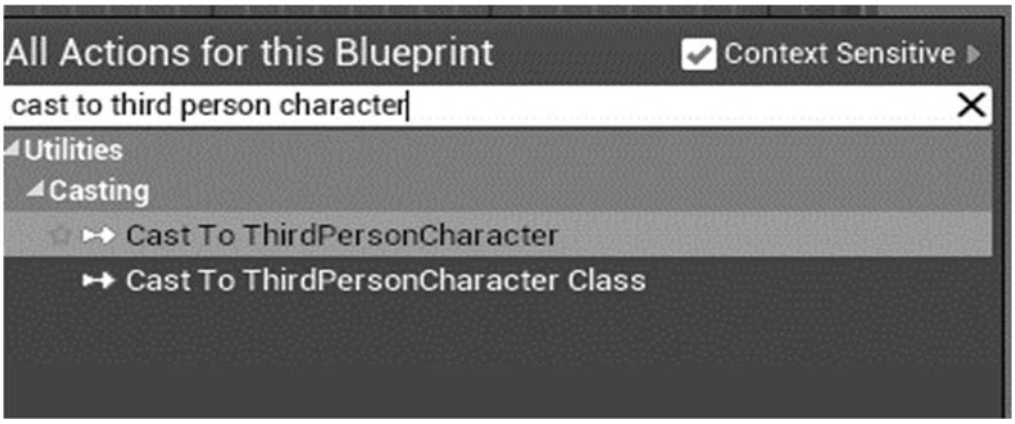

Рисунок 3.9 – Поиск функции Cast To

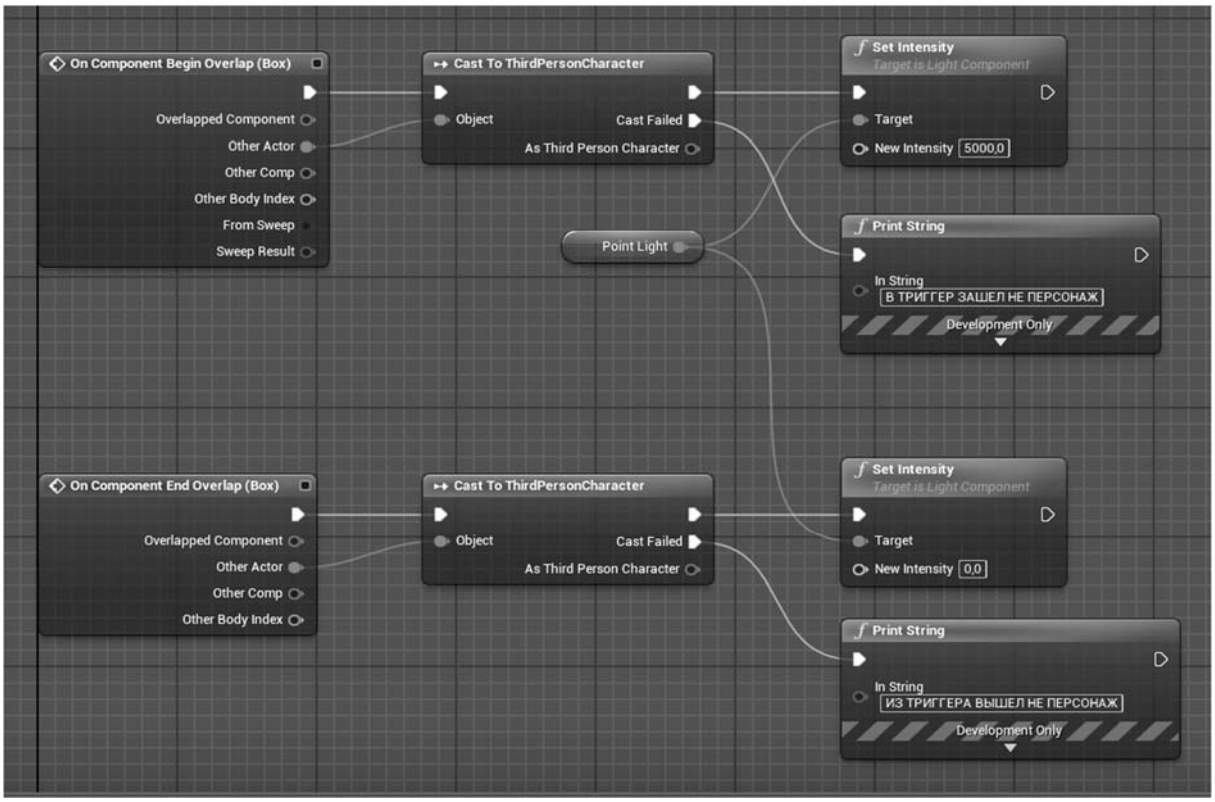

Рисунок 3.10 – Логика BP_Trigger после добавления проверок

У нее есть два выходных исполняемых пина и один пин с ссылкой на приведенный класс. После добавления необходимо соединить соответствующие пины в созданной нами логике работы триггера (рисунок 3.10).

Функция Cast To является не только возможностью проверки на принадлежность к классу объекта, а также и возможностью получить ссылку на другой блупринт-класс для дальнейшей работы с ним и его компонентами.

Практические задания

Задание 1

Разработать BP_Item(Sphere, Sphere Collision). Добавить него переменные Name(тип Text) и Count(тип Int). Переменные должны быть Instance Editable. Разместить несколько объектов на сцене. Изменить у объектов на сцене значение переменных. Когда персонаж заходит в область триггера, на экран должна вывестись информация о предмете (Name и Count через Print String).

Разработать блупринт BP_Door (Static Mesh, Box Collision). Если персонаж находится в границах триггера и нажимает на кнопку, дверь должна открыться (повернуть Static Mesh! при помощи функции Set World Rotation).

Задание 2

Разработать BP_Enemy(Skeletal Mesh, Sphere Collision). В деталях компонента Skeletal Mesh установить Mesh. Создать переменные Name (тип Text) и HP (тип Int). Переменные должны быть Instance Editable. Разместить несколько персонажей на сцене. Изменить у объектов на сцене значение переменных. Когда персонаж заходит в область триггера, на экран должна вывестись информация о персонажах (Name и HP через Print String).

Разработать BP_Item(Sphere, Sphere Collision). Если персонаж находится в границах триггера и нажимает на кнопку, предмет исчезает (функция Destroy Actor), а в персонаже увеличивается кол-во подобранных предметов (переменная типа Int).

Задание 3

Разработать BP_NPC (Skeletal Mesh, Sphere Collision). В деталях компонента Skeletal Mesh установить Mesh. Создать переменные Name (тип Text) и HasQuest?(тип Bool). Переменные должны быть Instance Editable. Разместить несколько персонажей на сцене. Изменить у объектов на сцене значение переменных. Если переменная Bool выставлена на True, то изменить материал Skeletal Mesh(Set material после Begin Play). Когда персонаж заходит в область триггера, на экран должна вывестись информация о персонаже (Name через Print String).

Разработать BP_Lamp (Spot Light) и BP_Switch (Cube, Box Collision). Если персонаж находится в границах триггера и нажимает на кнопку, в BP_Lamp включается источник света (Set Intensity) и загорается определенным цветом (Set Light Color).

Задание 4

Разработать BP_Item (Sphere, Sphere Collision). Добавить него переменные Name (тип Text) и Count (тип Int). Переменные должны быть Instance Editable. Разместить несколько объектов на сцене. Изменить у объектов на сцене значение переменных. Когда персонаж заходит в область триггера, на экран должна вывестись информация о предмете (Name и Count через Print String).

Разработать BP_RewardChest (Cube, Sphere Collision). Сделать компонент

Cube невидимым(свойство Visible в деталях). Если персонаж заходит в область триггера, Cube появляется (Set Visibility).

Контрольные вопросы

1 Что такое актер (Actor) в Unreal Engine и какую роль он играет в проекте?

2 Опишите процесс добавления нового объекта (актера) в сцену. Какие шаги необходимо выполнить?

3 Как можно импортировать 3D-модель в Unreal Engine? Какие форматы файлов поддерживаются?

4 Что такое Blueprint и как он используется для управления логикой объектов в Unreal Engine?

5 Как настроить свойства объекта (актера) в редакторе Unreal Engine? В каких вкладках находятся эти настройки?

6 Что такое компонент (Component) и как он влияет на поведение актеров в проекте? Приведите примеры компонентов.

7 Как можно создать и использовать пользовательские события в Blueprint? Как это может повлиять на управление объектами?

8 Как реализовать взаимодействие между объектами (актерами) в сцене? Приведите примеры методов взаимодействия.

9 Что такое события (Events) в Unreal Engine и как их использовать для управления логикой игры?

10 Как сохранить и загрузить состояние объектов в игре? Какие механизмы для этого предоставляет Unreal Engine?

4 Лабораторная работа № 4. Настройка материалов и текстур для объектов

Цель работы: изучение процесса создания и настройки материалов и текстур для объектов в Unreal Engine; применение различных текстурных процедур.

Основные теоретические положения

Материал (material) или шейдер (shader) – это совокупность текстур, векторов и других математических вычислений для описания поверхности и свойств ассетов в UE. Для создания материалов и текстур используется редактор материалов (Material Editor). Для создания материала необходимо выбрать соответствующий пункт в окне создания ассетов (рисунок 4.1).

При открытии созданного материала будет отображен редактор, состоящий из графа для редактирования, окна деталей, окна предпросмотра и окна с доступными элементами и функциями (рисунок 4.2).

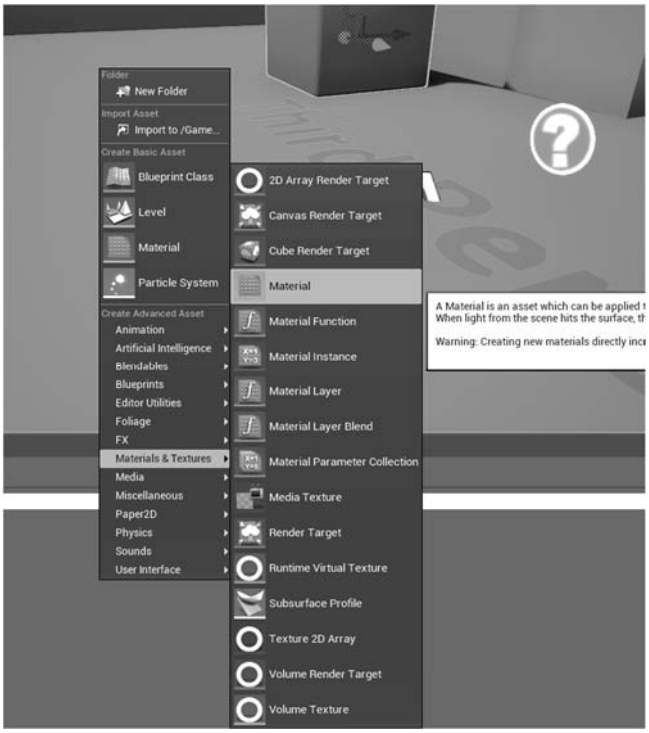

Рисунок 4.1 – Создание материала

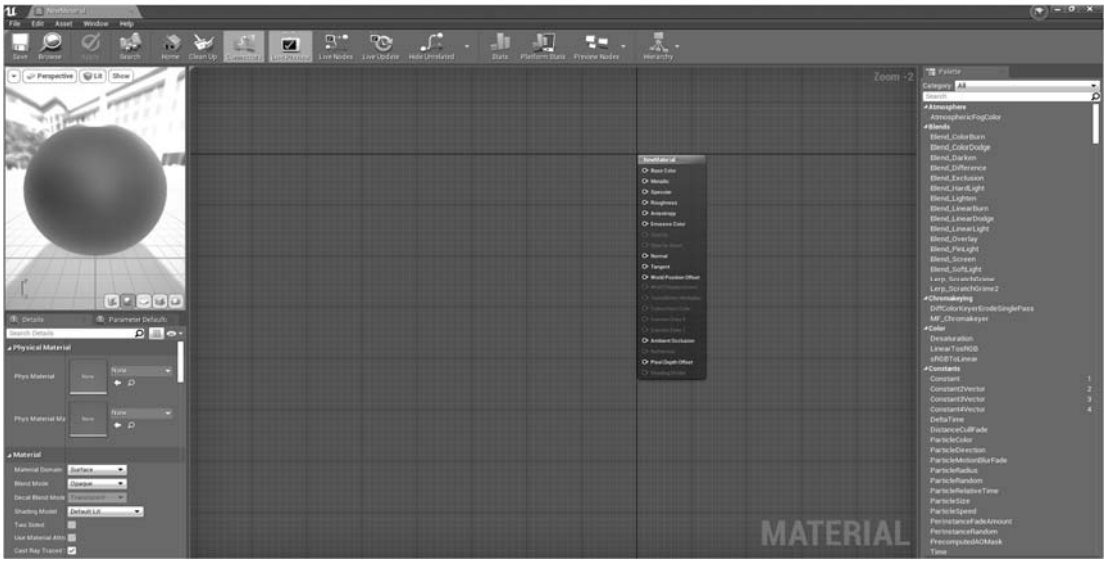

Рисунок 4.2 – Редактор материалов

В основе создания материалов лежит главная нода, содержащая входные параметры, которые контролируют финальное отображение материала на сцене. Далее будут рассмотрены основные свойства материала.

Base Color

Базовый цвет (Base color), который также иногда называют альбедо (albedo) или рассеивание (diffuse), ключевое описание цвета поверхности материала без учета всех деталей теней и освещения. По существу, входные данные базового цвета используют текстуру альбедо, которая представляет собой чистое значение

цвета создаваемого вами материала. Они не должны учитывать информацию о тени и освещении, а показывать только цвет, который вы хотите представить в материале. Это свойство может использовать входные данные текстур или даже простое векторное значение, которое является числом, представляющим чистый цвет (рисунок 4.3).

Metallic

Входные данные металлизированности (Metallic) материала используются, чтобы описать, является ли материал металлическим. Эти входные данные являются одним из наиболее простых для понимания элементов редактора материалов, а также одним из важнейших параметров для получения правильного отображения. Значение 0 используется для полностью неметаллических поверхностей, а значение 1 – для полностью металлических. Свойство отображено на рисунке 4.4.

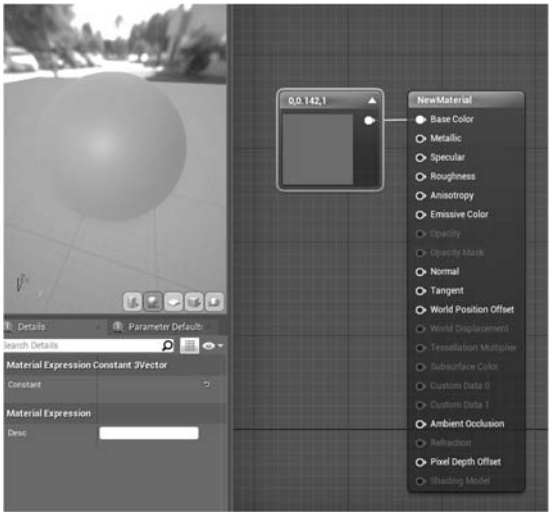

Рисунок 4.3 – Свойство Base Color

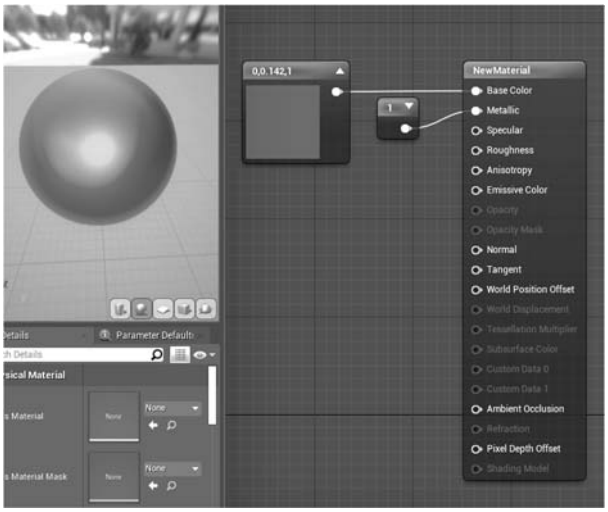

Рисунок 4.4 – Свойство Metallic

Roughness

Шероховатость (Roughness), глянец (gloss) или микроповерхностная детализация (micro surface detail), является одним из наиболее гибких аспектов PBR-системы. Такая текстура используется для представления шероховатости и изношенности поверхности создаваемого материала. Шероховатость изображает миниатюрные детали и описывает глянец или количество света, отражаемого с поверхности. К примеру, если вы создаете новый стальной металлический материал, металличность и альбедо материала будут довольно простыми, вам понадобится сделать небольшие изменения в цвете и колебании шума, но вы будете использовать шероховатость, чтобы описать все мелкие детали поверхности, такие как мелкие царапины, загрязнения или пыль. Свойство Roughness представлено на рисунке 4.5. Слева используется значение Roughness равное 0, а справа – 1.

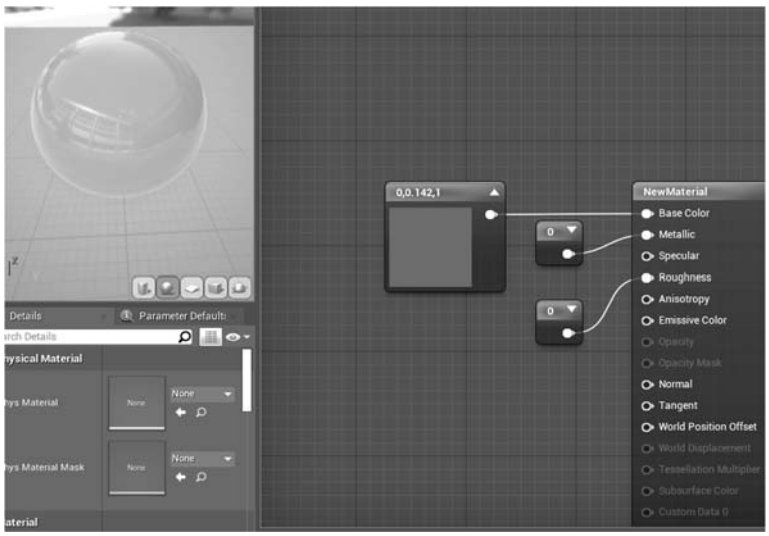

Рисунок 4.5 – Свойство Roughness

Normal Map

Использование текстуры с картой нормалей позволяет имитировать детали поверхности и рельеф. Каждый канал карты текстур (красный, зеленый и синий) составлен для представления различных углов направлений поверхности по цвету. Красный представляет ось X или направление света слева направо, сталкивающегося с поверхностью; зеленый – ось Y или направление сверху вниз; синий – ось Z или направление спереди назад. Свойство Normal представлено на рисунке 4.6.

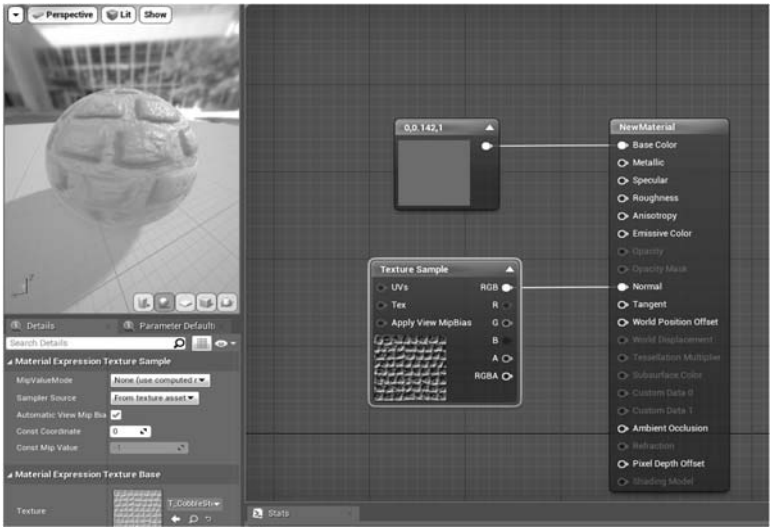

Рисунок 4.6 – Свойство Normal

Emissive Color

Свойство Emissive Color определяет эффект свечения материала. Все значения больше 1 будут создавать эффект свечения и HDR эффект. Свойство Emissive представлено на рисунке 4.7.

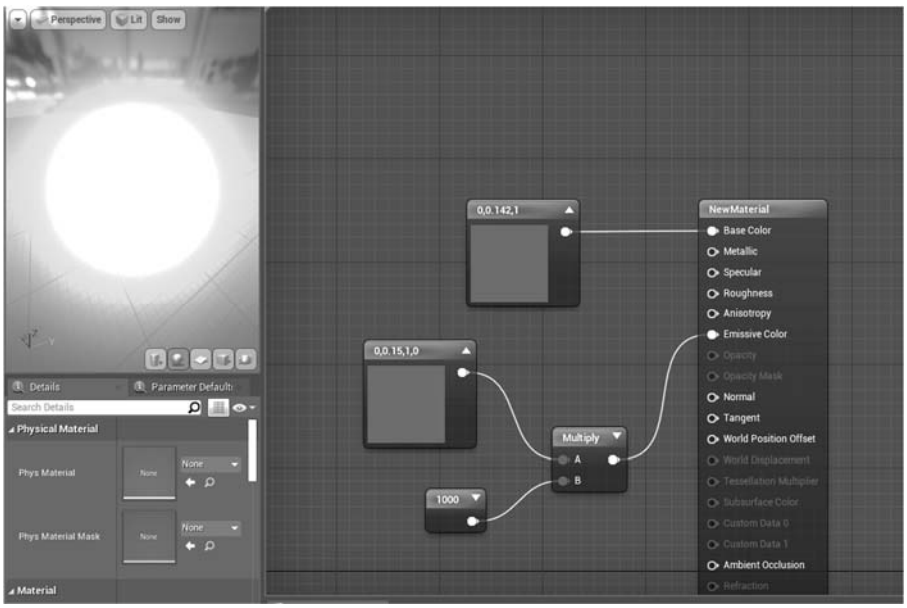

Рисунок 4.7 – Свойство Emissive Color

Ноды значений

Константы (constant values) – числа, которые могут задавать значения или цвета, в зависимости от количества используемых значений. Вы можете брать ноды значений из панели Palette и использовать их в материале в качестве входов.

Два наиболее часто используемых нода значений – это нод Constant и Constant3Vector. Нод Constant представляет единственное число или значение. Нод Constant3Vector представляет вектор или набор из трех чисел, каждое из которых представляет соответствующее значение RGB. Например, если нод Constant3Vector равен 1,4,6, это значит, что значение 1 для красного цвета, 4 – для зеленого и 6 – для синего. Константы представлены на рисунке 4.8.

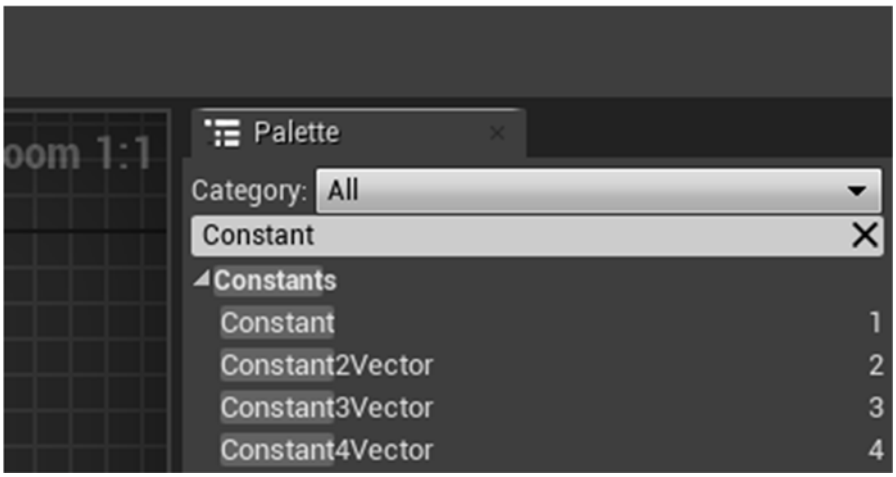

Рисунок 4.8 – Константы в окне Pallete

Использование текстур

Для того чтобы использовать заранее подготовленные текстуры, необходимо добавить ноду Texture Sample. При добавлении этой ноды необходимо в ее окне деталей указать ссылку на ассет текстуры, который должна использовать эта нода. Также можно перенести ассет текстуры из Content Browser непосредственно в окно редактора материалов, и нода Texture Sample добавится автоматически с необходимой ссылкой на ассет. Пример использования текстур представлен на рисунке 4.9.

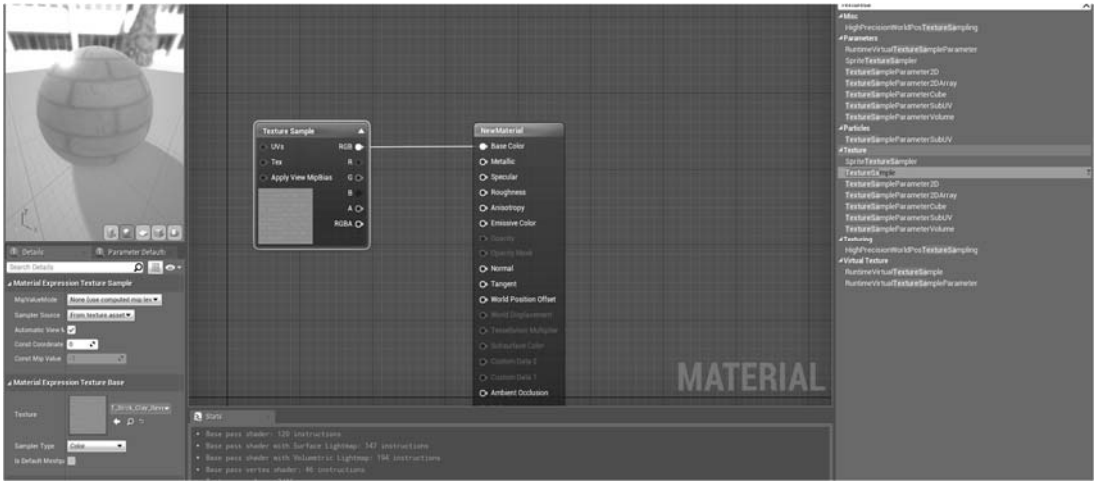

Рисунок 4.9 – Пример использования текстур

Экземпляры материалов

Экземпляры имеют ключевое значение для многократного использования материала без необходимости каждый раз создавать новый материал. Изменив некоторые ноды в материале, вы можете создать в нем особые ноды, способные динамично меняться. Например, если основной материал выкрашен нодом Constant3Vector в зеленый, превратив нод в параметр, вы сможете легко менять цвет другого экземпляра этого материала, вместо того чтобы создавать новый с нуля.

Чтобы работать с экземплярами, главный материал должен приспособиться к динамичной трансформации определенных нодов, превратив их в параметризированные ноды. В редакторе материалов основного материала щелкните ПКМ по любой константе-ноде и выберите вариант Convert to Parameter (конвертировать в параметр) в контекстном меню (рисунок 4.10).

Также можно изначально добавлять в материал параметры. Найти все доступные для создания параметры можно по соответствующему поиску в окне Palette (рисунок 4.11).

Чтобы создать экземпляр материала, необходимо щелкнуть ПКМ по материалу в окне Content Browser и выбрать вариант Create Material Instance в выпадающем списке (рисунок 4.12). Этот экземпляр материала использует параметры родительского материала, который был создан ранее.

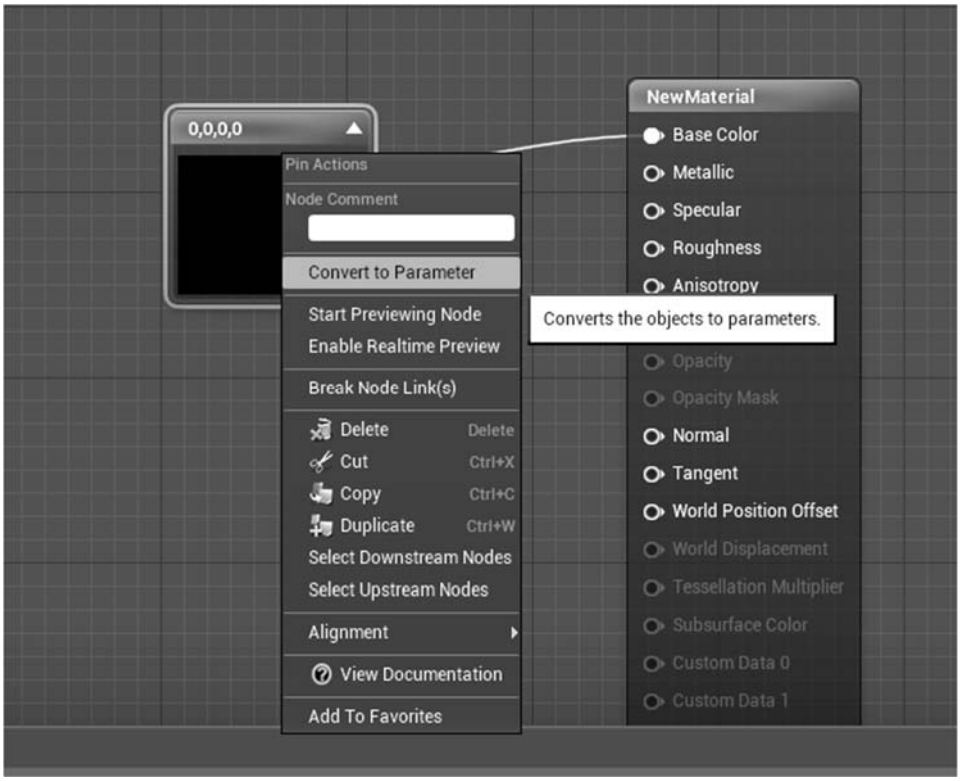

Рисунок 4.10 – Конвертация константы в параметр

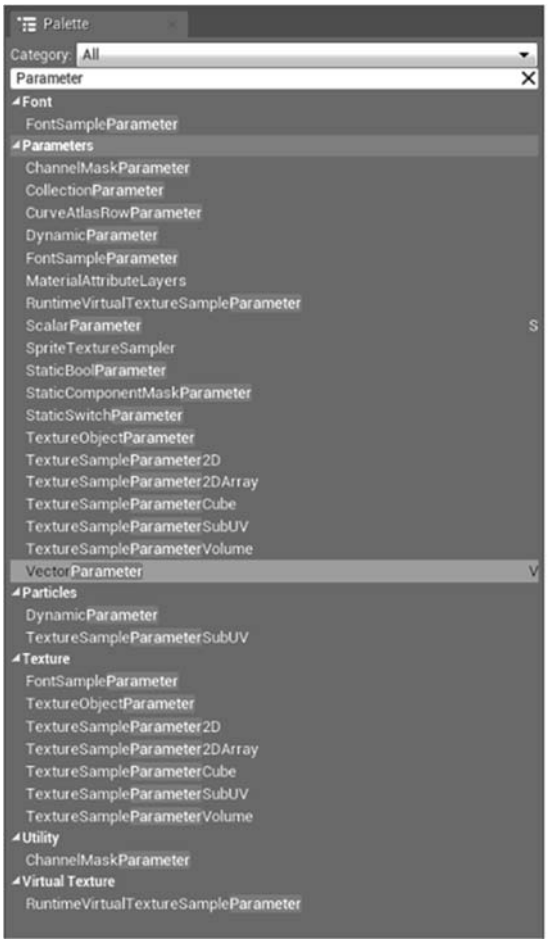

Рисунок 4.11 – Поиск параметров в окне Palette

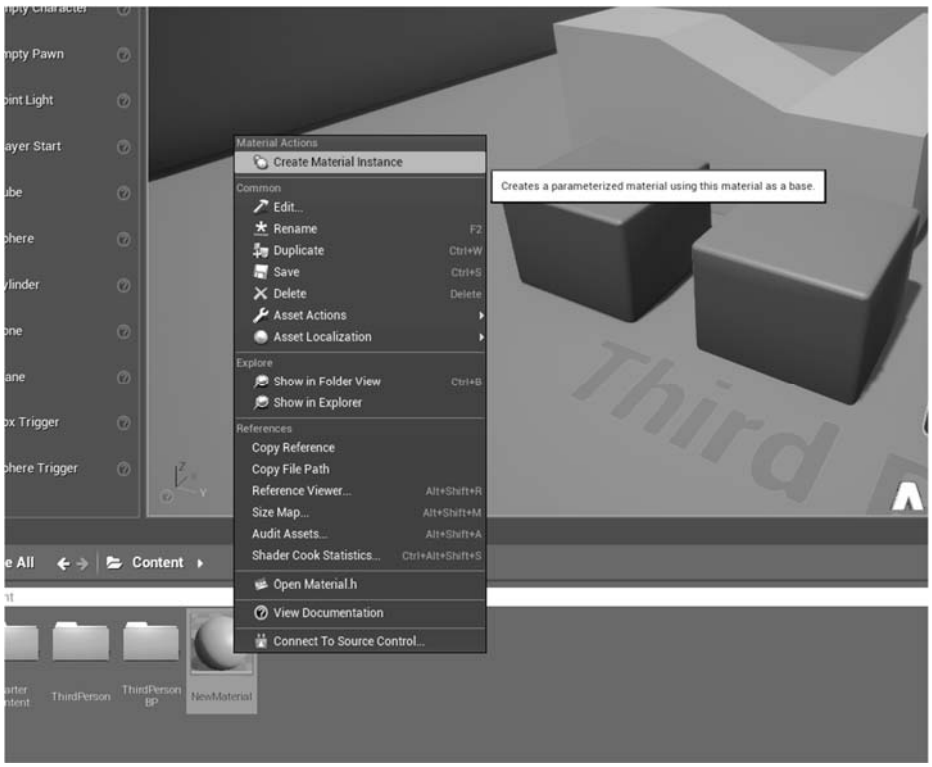

Рисунок 4.12 – Создание экземпляра материала

На рисунке 4.13 представлено окно редактирования экземпляра, в котором указаны параметры, добавленные в родительский материал.

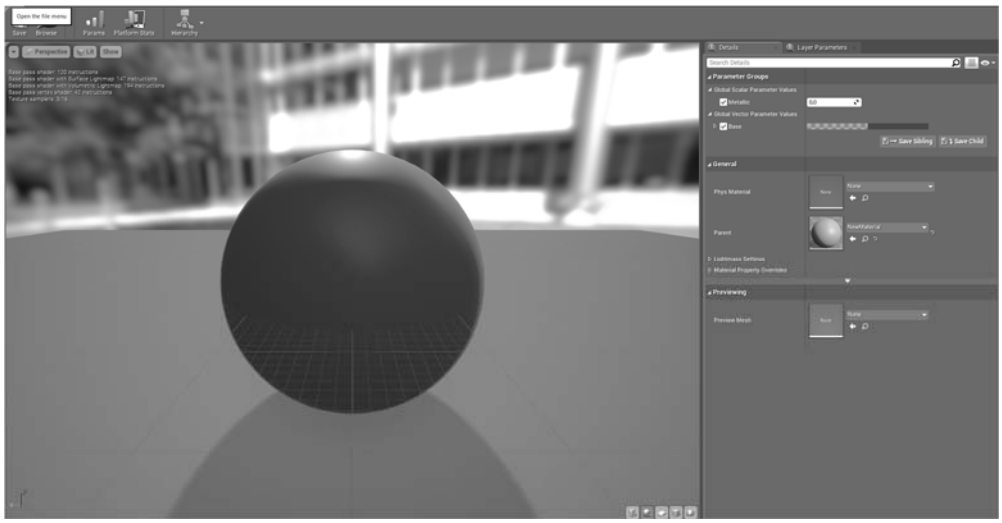

Рисунок 4.13 – Окно редактирования экземпляров материала

Практические задания

Задание

Для нескольких моделей необходимо создать материалы, используя различные текстуры. Все материалы должны иметь свойство редактирования степени растягивания по каждой стороне и возможность смешивания основного цвета с параметром. Все материалы должны иметь экземпляры с различными

значениями параметров. В случае отсутствия текстур для свойств добавить параметры. Ресурсы с моделями и текстурами по ссылке – https://drive.google.com/file/d/1Ahxcr8LBgGuvizHvzsTeNuzzDSLvjr6x/view?usp=sharing.

Можно использовать любые другие модели и текстуры.

Контрольные вопросы

1 Что такое материал (Material) в Unreal Engine и какова его основная функция?

2 Какой процесс необходимо выполнить для создания нового материала в Unreal Engine? Опишите шаги.

3 Что такое текстура (Texture) и как она используется в материалах? Каковы основные типы текстур в Unreal Engine?

4 Как импортировать текстуры в проект Unreal Engine? Какие форматы файлов поддерживаются для текстур?

5 Как использовать узлы (Nodes) в Material Editor для создания сложных материалов? Приведите примеры узлов, которые могут быть полезны.

6 Что такое UV-развертка (UV Mapping) и как она влияет на применение текстур к объектам?

7 Как настроить прозрачность материалов в Unreal Engine? Какие параметры необходимо изменить для достижения эффекта прозрачности?

8 Опишите процесс создания материала с использованием нормалей (Normal Maps). Как они влияют на рендеринг поверхности?

9 Как можно применить различные материалы к отдельным частям одного объекта? Какие подходы для этого существуют?

5 Лабораторная работа № 5. Добавление физической симуляции для объектов

Цель работы: изучение основ физической симуляции в Unreal Engine и применение полученных знаний для добавления реалистичного поведения объектов в графических приложениях. Студенты научатся использовать инструменты физики движка, а также интегрировать физические свойства в игровые объекты.

Основные теоретические положения

Физическая симуляция в Unreal Engine является ключевым компонентом, который позволяет разработать реалистичные взаимодействия между объектами в игре. Основной принцип системы физики основан на применении законов Ньютона и математических моделей для расчёта движения и столкновений объектов. Рассмотрим более подробно основные аспекты физической симуляции

в Unreal Engine.

Физические материалы. Физические материалы (Physical Materials) в Unreal Engine позволяют разработчику задавать специфику взаимодействия объектов с окружающей средой. Основные свойства физических материалов включают:

– плотность: определяет, насколько тяжёлым будет объект. Влияет на его поведение при столкновениях и взаимодействиях;

– сопротивление (Friction): указывает, насколько скользким будет объект при контакте с другими поверхностями. Увеличение этого параметра приводит к замедлению движения объектов при контакте;

– упругость (Restitution): отвечает за отскок объектов после столкновения. Чем выше значение, тем «упругее» будет объект.

Столкновения. Столкновения (Collisions) являются важным аспектом физической симуляции. Unreal Engine предлагает несколько типов столкновений:

– пробковые: столкновения, при которых объекты не проходят друг через друга.

– триггерные: позволяют отслеживать столкновения с объектами, не блокируя при этом их движение. Используется для активации событий.

– физические столкновения: реальные взаимодействия объектов, при которых расчёт физики влияет на движение и поведение.

Столкновения могут быть настроены через viewport в редакторе, где можно указать типы столкновений для каждого объекта.

Физические тела. Физические тела (Rigid Bodies) – это объекты, которые могут взаимодействовать с другими объектами через физику. В Unreal Engine существуют два типа физических тел:

1) статические тела (Static Bodies): объекты, которые не движутся. Они могут использоваться для создания поверхности, стен и других объектов, которые не должны изменять своё положение;

2) динамические тела (Dynamic Bodies): объекты, которые могут двигаться и реагировать на физические силы. Эти объекты могут быть подвержены столкновениям, гравитации и другим физическим эффектам.

Силы и моменты. Силы (Forces) играют важную роль в физической симуляции. В Unreal Engine вы можете применять различные виды сил к вашим объектам:

– сила (Force): постоянно применяемая сила, влияющая на движение объекта. Например, сила тяжести;

– импульс (Impulse): резкая сила, aplicada на короткий промежуток времени, вызывающая мгновенное изменение скорости объекта. Например, при ударе;

– torque: вращающая сила, влияющая на объект и вызывающая его вращение.

Анимация и физика. Unreal Engine также поддерживает интеграцию анимации и физики. С помощью Physics Constraints можно задавать ограничения на движение объектов, что позволяет создать более сложные системы, такие как полиэтиленовые эффекты, веревки и системы управления движением пер-

сонажей.

Примеры использования. Физическая симуляция может применяться во множестве игровых сцен:

– динамическое разрушение объектов: например, при стрельбе по стене объекты разлетаются на куски;

– взаимодействие с окружающей средой: например, при перемещении объектов или взаимодействии с изометрическими физическими элементами;

– игровая механика: включение механик, основанных на физическом взаимодействии, таких как гравитация, столкновения и отскоки.

Физическая симуляция в Unreal Engine является мощным инструментом для создания интерактивных и реалистичных игровых миров, позволяя разработчикам реализовывать увлекательные игровые механики и сценарии.

Практические задания

Задание 1. Создание физического материала

1 Создайте новый материал в Unreal Engine и настройте его физические свойства (плотность, трение, упругость).

2 Примените материал к простому объекту (например, кубу).

Задание 2. Настройка физических свойств объекта

1 Создайте новый статический или динамический объект в сцене.

2 Настройте параметры Rigidbody (масса, гравитация, трение).

3 Проверьте поведение объекта при взаимодействии с другими объектами.

Задание 3. Создание системы столкновений

1 Создайте два объекта с настроенными физическими материалами.

2 Добавьте обработчики событий для столкновения объектов.

3 Реализуйте изменение цвета объекта при столкновении.

Задание 4. Установка сил и их влияние

1 Создайте объект и добавьте к нему компонент Physics Force.

2 Реализуйте приложение силы на объект (например, при нажатии на клавишу).

3 Наблюдайте за поведением объекта при воздействии силы.

Задание 5. Создание физической симуляции в игре

1 Разработайте простую сцену с несколькими взаимодействующими объектами.

2 Настройте физические свойства и убедитесь, что объекты реагируют на физику реалистично.

3 Добавьте элементы игрового процесса, использующие физику (например, падение объектов).

Контрольные вопросы

1 Какова роль физических материалов в Unreal Engine?
2 Что такое статические и динамические тела в контексте физики?
3 Какие параметры физического тела можно настраивать?
4 Как работают силы в Unreal Engine?
5 Что происходит при столкновении двух физических объектов?
6 Какой инструмент в Unreal Engine используется для настройки столкновений?
7 Какие типы столкновений существуют в Unreal Engine?
8 Как можно изменить поведение объекта при взаимодействии с другими объектами?
9 В чем разница между физическими и статическими объектами?

6 Лабораторная работа № 6. Настройка поведения искусственного интеллекта для объектов

Цель работы: изучение базовых алгоритмов искусственного интеллекта в Unreal Engine и их применение.

Основные теоретические положения

Введение

В играх под AI (Artificial Intelligence, искусственный интеллект) понимается реализация интеллектуальных (кажущихся интеллектуальными) поведений для различных агентов. Типы AI могут существенно различаться:

– как AI обрабатывает задачи;
– как AI получает или воспринимает информацию о мире;
– как AI интерпретирует геометрическое пространство мира;
– как AI пытается повторять поведение, приближенное к человеческому поведению.

Главным компонентом любого игрового AI является формальная система принятия решений – по сути мозг AI. Этот «мозг» выполняет определенные действия, оценивая окружающую обстановку.

Создание AI включает в себя следующие шаги.

1 Создание класса неигрового персонажа.
2 Создание AI контроллера.
3 Добавление и настройка навигации.
4 Создание дерева поведения.

Создание неигрового персонажа и контроллера

Класс AIController является классом, в котором принято писать специфическую для неигрового персонажа логику. Часть нод, например ноды, связанные с созданием дерева поведения, можно создать только в классе AIController и наследуемых от него. Чтобы создать новый контроллер для персонажа, необходимо выбрать класс AIController в окне создания блупринт-класса (рисунок 6.1).

Рисунок 6.1 – Создание контроллера

После того, как контроллер создан, можно переходить к созданию персонажа. Обычно для объектов управляемого AI используются классы Pawn и Character. Класс Character наследуется от Pawn и есть смысл в его использовании, если ваш агент – это классический гуманоидный персонаж. Для создания класса персонажа необходимо выбрать пункт Character в окне создания блупринт-класс (рисунок 6.2).

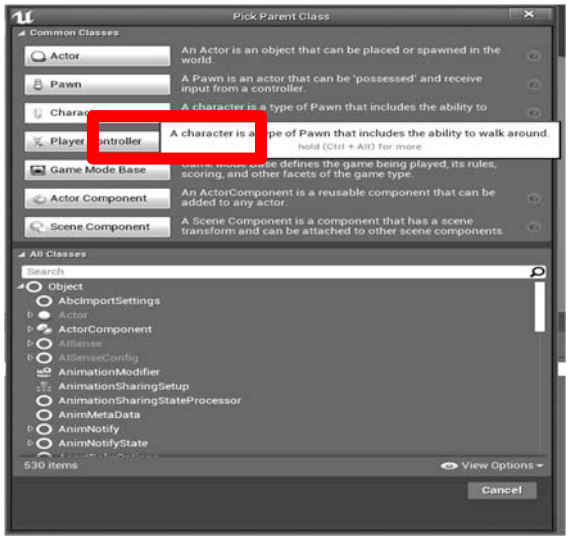

Рисунок 6.2 – Создание класса персонажа

После этого необходимо настроить меш персонажа. Для примера будет использоваться стандартный манекен. Настройка меша на рисунке 6.3.

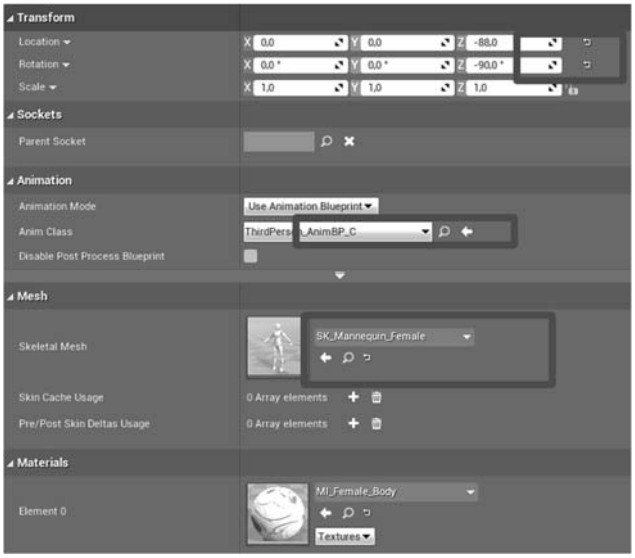

Рисунок 6.3 – Настройка меша

Последней настройкой персонажа является замена стандартного AIController на созданный вручную. Для этого необходимо выбрать нужный класс AIController Class в общих настройках класса персонажа (рисунок 6.4). После этого можно разместить созданного персонажа на сцену.

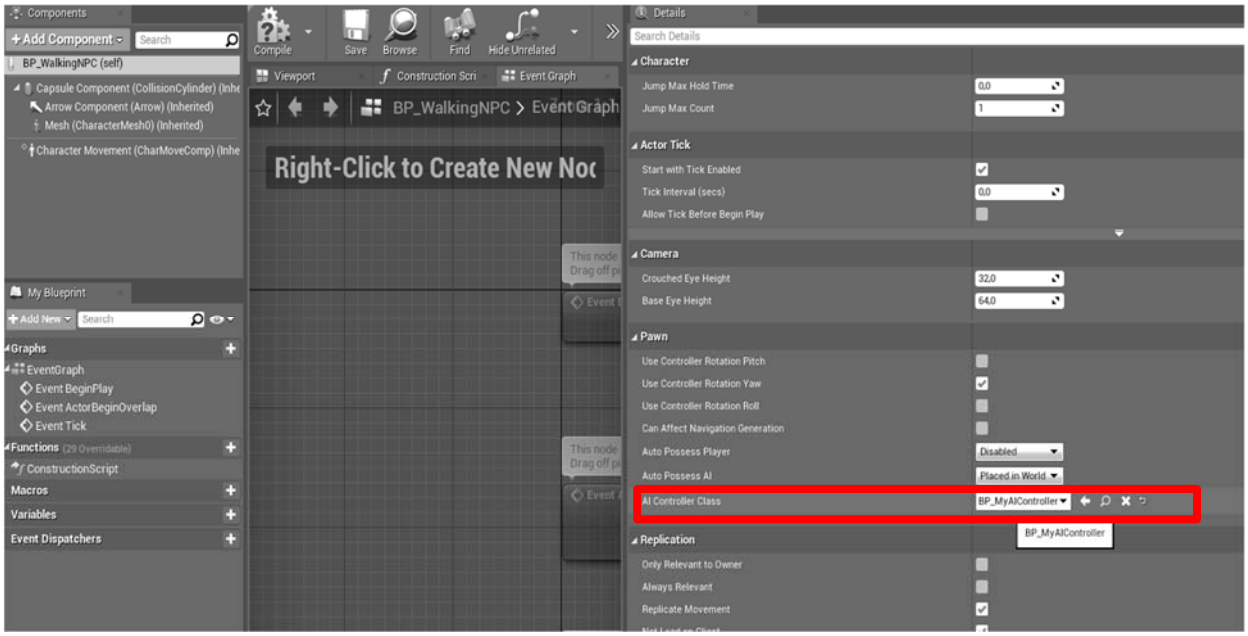

Рисунок 6.4 – Настройка контроллера для персонажа

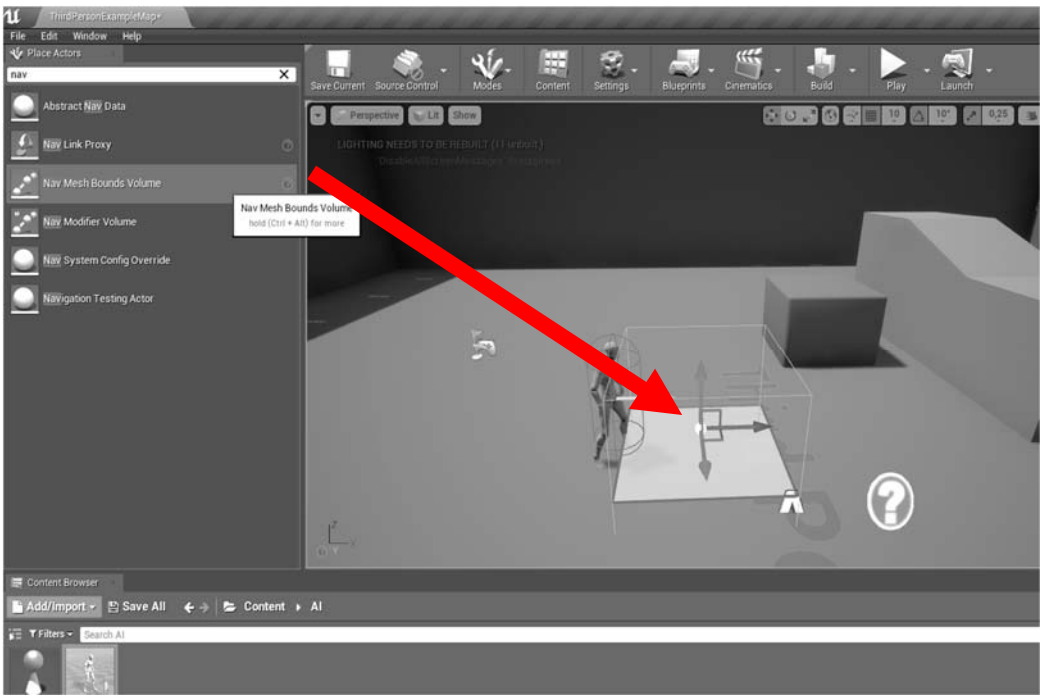

Рисунок 6.5 – Добавление Nav Mesh Bounds Volume

Добавление навигации

AI может перемещаться по миру, только если он понимает, как это делать. AI воспринимает мир вокруг себя в очень упрощенном формате. Процесс идентификации путей по локации называется Pathfinding. Три основных типа навигационных систем – Navigation Points, Navigation Grids и Navigation Meshes.

Для того чтобы добавить навигацию на уровень, необходимо поместить туда Nav Mesh Bounds Volume (рисунок 6.5). Вы можете изменить масштаб актора, чтобы покрыть нужную площадь уровня.

При нажатии на клавишу P будет подсвечена область Nav Mesh. Персонажи, управляемые AI, смогут перемещаться только по зеленой зоне. Также можно отобразить область Nav Mesh в окне «Show», выбрав Navigation (рисунок 6.6).

Рисунок 6.6 – Отображение навигации

Практические задания

Задание

Создать неигрового персонажа и дерево поведения. Необходимая модель поведения – патрулирование (движение к различным случайным точкам в определенном радиусе), следование за главным персонажем при его приближении и возвращение к патрулированию при его отдалении.

Контрольные вопросы

1 Что такое искусственный интеллект (ИИ) в контексте Unreal Engine и какие его основные компоненты?

2 Опишите, что такое поведенческое дерево (Behavior Tree) и как оно используется для создания сложного поведения ИИ.

3 Как реализовать навигацию для объекта с ИИ в игровом мире? Какие компоненты отвечают за навигацию?

4 Что такое Blackboard и как он взаимодействует с поведенческими деревьями?

5 Как настроить ИИ-агента для распознавания целей в игре? Какие методы могут быть использованы для этого?

6 Как использовать Blueprint для создания и настройки поведения ИИ? Какие узлы могут быть полезны для этой задачи?

7 Что такое системы триггеров (Trigger Systems) и как они могут быть использованы для взаимодействия ИИ с окружающей средой?

8 Как настроить реакции ИИ на события, происходящие в игре, например на звуковые или визуальные сигналы?

7 Лабораторная работа № 7. Создание пользовательского интерфейса и обработка ввода

Цель работы: создание пользовательских интерфейсов в Unreal Engine; размещение элементов интерфейса; настройка взаимодействия с пользователем и обработка ввода.

Основные теоретические положения

Создание виджет-блупринта

Unreal Motion Graphics UI Designer (UMG) – редактор в UE, который используется для создания пользовательских. UMG позволяет в интерактивном режиме размещать элементы интерфейса, называемые виджетами, и назначать блупринта заданную функциональность. Чтобы создать виджет, необходимо в окне создания ассетов выбрать Widget Blueprint в User Interface (рисунок 7.1).

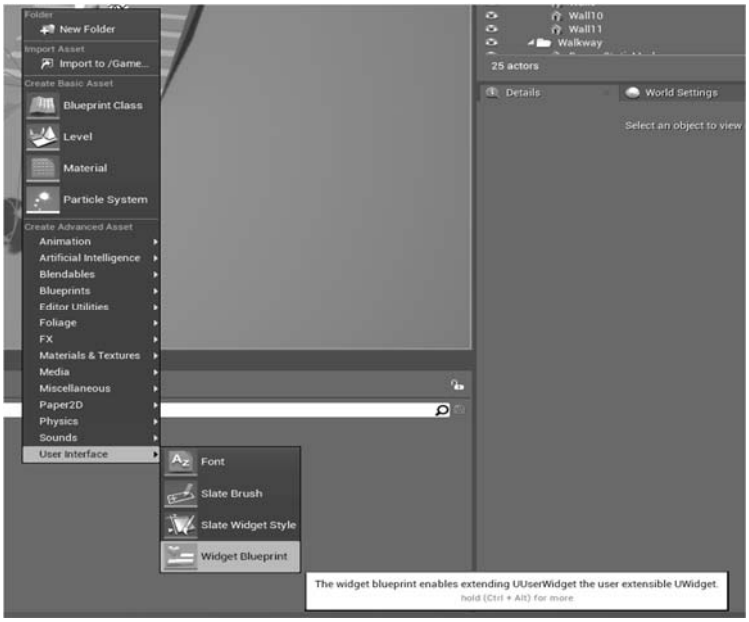

Рисунок 7.1 – Создание виджета

Навигация по интерфейсу UMG

Интерфейс UMG имеет два режима работы, приведенные на рисунке 7.2: режим конструктора Designer для размещения виджетов, например изображений или текста, и режим Graph для добавления функций в блупринт. При первом открытии UMG по умолчанию отображает режим конструктора.

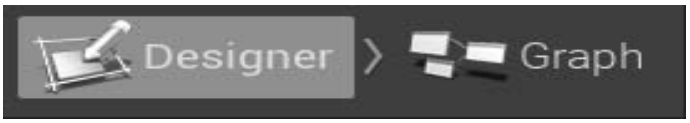

Рисунок 7.2 – Переключение между режимами

Режим конструктора

Режим конструктора представляет собой следующий набор окон.

Окно Palette. Содержит список всех стандартных виджетов, которые можно использовать, отсортированный по функциональности, а также виджетов, созданных пользователем.

Окно Hierarchy. Содержит виджеты, размещенные на вашем интерфейсе. На панели Hierarchy можно прикреплять виджеты друг к другу и отделять их друг от друга по мере необходимости.

Окно Details. Отображает свойства выбранных виджетов, размещенных на интерфейсе.

Окна Animations и Timeline. Используются для создания, управления и редактирования анимацией виджетов.

Окно Designer. Используется для создания интерфейса путем перетаскивания элементов с панели Palette.

Добавление виджета на экран

Для того чтобы созданный виджет отображался на экране, необходимо создать его при помощи функции Create Widget и выбрать необходимый класс виджета, а потом вывести при помощи функций Add To Viewport (рисунок 7.3). Эту комбинацию необходимо добавить в Begin Play любого класса, находящегося на сцене, например Third Person Character.

Функция Create Widget создает ссылку на виджет, что позволяет дополнительно записать эту ссылку в переменную, чтобы изменять созданный виджет из другого класса в дальнейшем.

В случае, если в проекте по умолчанию не отображается курсор мыши или пропадает после нажатия (например, в проекте от третьего лица), дополнительно необходимо изменить Input Mode на Game And UI, и отобразить курсор на экране с помощью Set Show Mouse Cursor (рисунок 7.4).

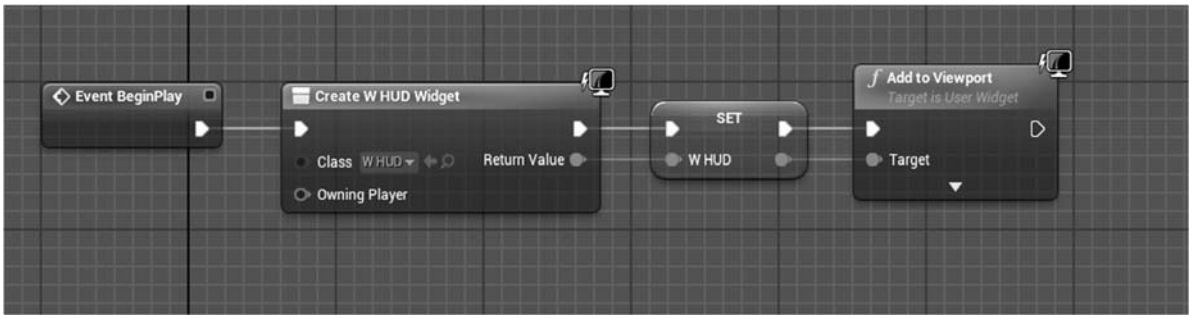

Рисунок 7.3 – Вывод виджета на экран

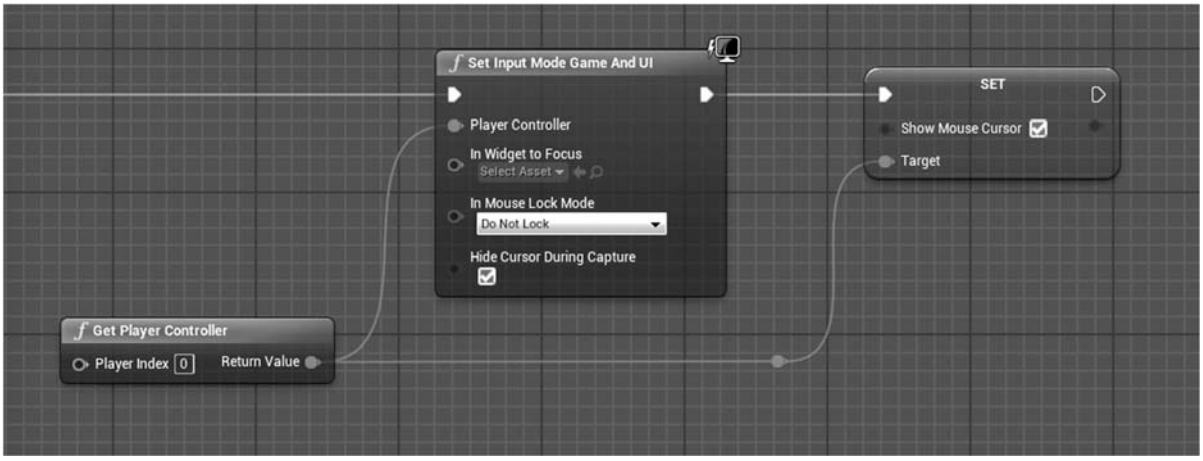

Рисунок 7.4 – Постоянное отображение курсора на экране

Переменные Expose On Spawn и Event Construct

При создании виджета и добавлении его на экран зачастую возникает необходимость отобразить какие-то значения в элементах виджета. Для этого можно использовать свойство переменных ExposeOnSpawn (рисунок 7.5) и Event Construct. Чтобы использовать это свойство, переменная также должна быть Instance Editable.

Свойство ExposeOnSpawn добавит эту переменную во входной параметр в функции Create Widget (рисунок 7.6) и появится возможность присвоить значение этой переменной в момент создания виджета.

Чтобы эта переменная сразу отобразилась в нужном нам элементе Text, достаточно записать значение переменной в текстовое поле после события Event Construct. По аналогии с BeginPlay и блупринтами данное событие срабатывает в момент создания виджета. Событие EventConstruct виджета представлено на рисунке 7.7.

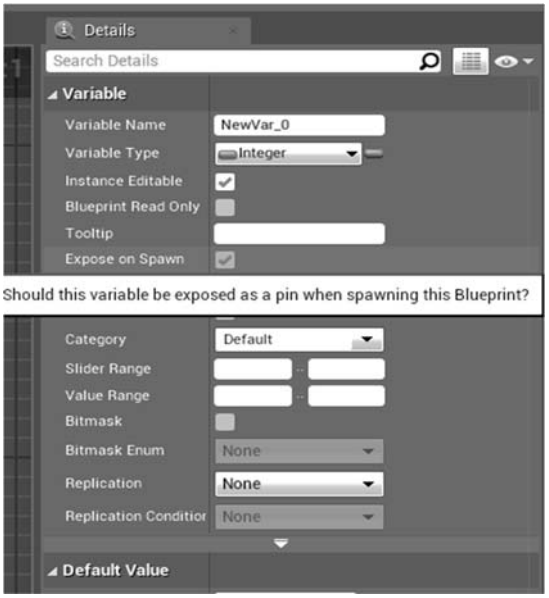

Рисунок 7.5 – Свойство Expose On Spawn

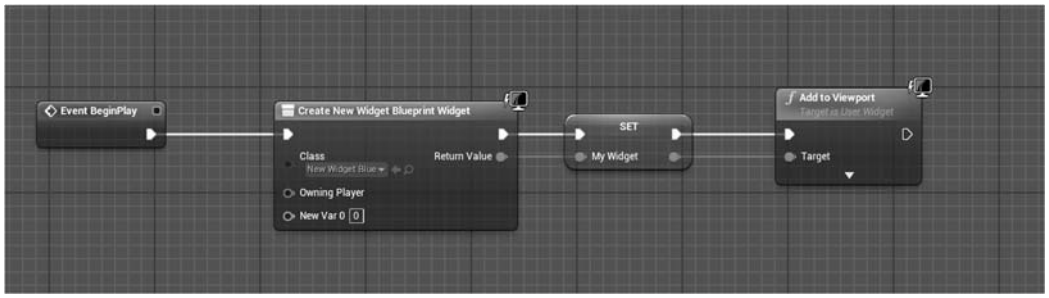

Рисунок 7.6 – Создание виджета с параметром

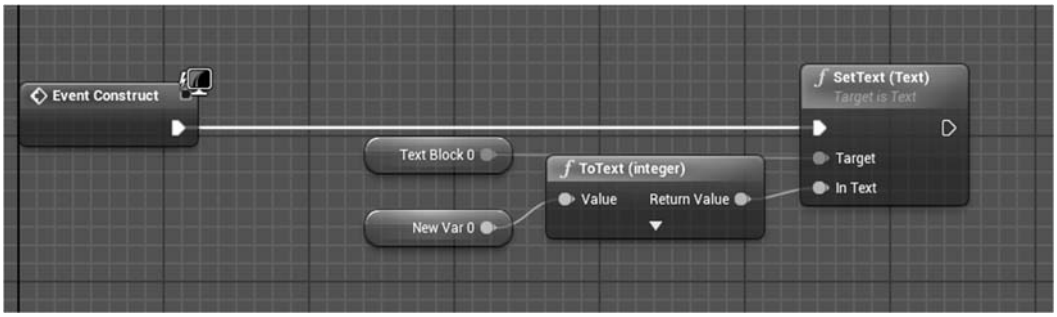

Рисунок 7.7 – Событие Event Construct

Практические задания

Задание 1

Создать виджет с тремя кнопками (элемент Button) и главным текстом (элемент Text). В кнопки дополнительно добавить элементы типа Text. При нажатии на каждую кнопку текст, записанный в элемент Text этих кнопок, должен отобразиться в главном элементе Text.

Задание 2

Разработать главное меню с визуальным оформлением. За основу можно взять любой уже имеющийся интерфейс меню из игр. Добавить все необходимые элементы и изменить им стиль. Функционал разрабатывать не нужно.

Контрольные вопросы

1 Что такое пользовательский интерфейс (UI) в Unreal Engine и какие его основные элементы?

2 Как создать новый виджет (Widget) в Unreal Engine? Опишите процесс создания и настройки виджета.

3 Как использовать Blueprint для привязки функциональности к элементам пользовательского интерфейса, таким как кнопки и текстовые поля?

4 Какие типы элементов UI доступны в Unreal Engine? Приведите примеры их использования.

5 Как настроить события (Events) для обработки ввода пользователя, например нажатия кнопок или перемещения мыши?

6 Что такое привязка данных (Data Binding) в контексте пользовательского интерфейса и как она может улучшить взаимодействие с игроком?

8 Лабораторная работа № 8. Создание ландшафта и настройка освещения в Unreal Engine

Цель работы: изучение основных принципов работы с ландшафтом, создание и настройка рельефа. Создание освещения, настройка источников света.

Основные теоретические положения

Введение

При работе с новым проектом вы можете столкнуться с тем, что статичные меши не покрывают пространство, которые вы хотели бы сделать доступным для исследования игроком, особенно если вы создаете свободные для путешествий внешние пространства. В таких ситуациях нужно использовать инструменты создания ландшафтов. Они довольно мощны и допускают создание мира, который может быть отредактирован посегментно, то есть вы можете производить

быстрое редактирование для расширения пространства игры и сделать игру с эффективным рендерингом.

Ландшафтные инструменты

Для создания и редактирования ландшафтов и их параметров в UE4 доступно множество инструментов. Доступ к элементам управления по умолчанию находится на панели Modes. По щелчку по кнопке Landscape (Ландшафт) откроется панель Landscape (рисунок 8.1). Вы также можете нажать комбинацию клавиш Shift + 2 для быстрого доступа к панели Landscape.

На панели Landscape доступны три главные вкладки:

1) Manage. Эта вкладка контролирует аспекты конструирования и управления ландшафтами;

2) Sculpt. Эта вкладка меняет пространство и форму ландшафтной геометрии;

3) Paint. Эта вкладка управляет типами материалов, примененными к поверхности ландшафтов.

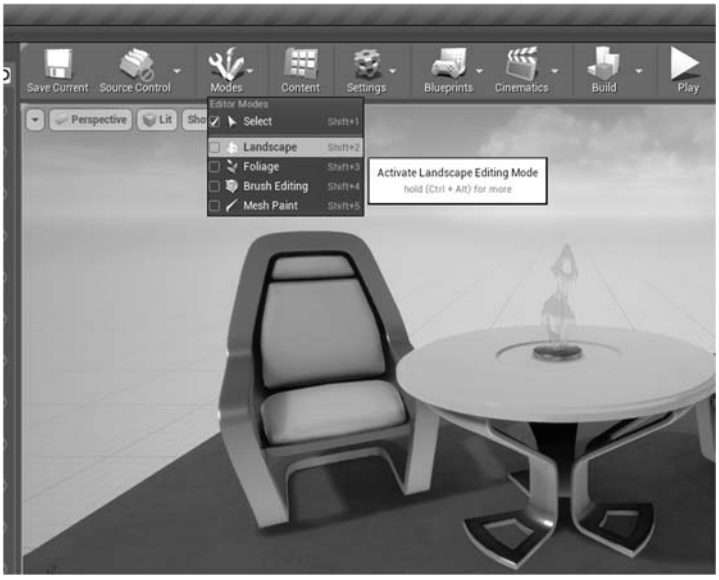

Рисунок 8.1 – Переход в режим редактирования ландшафта

Вкладки Sculpt и Paint затемнены и не могут быть использованы до тех пор, пока вы не создадите свой первый базовый ландшафт, где станут доступны эти опции.

Вкладка Manage

Первый раздел панели Landscape – это раздел управления (Manage). С его помощью вы можете создавать новые ландшафты и управлять существующими. Вы можете создать ландшафт двумя способами: новый ландшафт, основанный на наборе параметров, или ландшафт, основанный на импортированной карте высот.

Карта высот. Это текстура, предоставляющая информацию о разности высот, выполненная в оттенках серого. Белые поверхности на текстуре несут информацию ландшафту, что высота возрастает, в то время как черные пиксели сигнализируют о снижении высоты. Карта высот полезна, когда вы воссоздаете определенную локацию реального мира и полностью имитируете реальный ландшафт внутри игрового пространства. Также вы можете использовать карту высот как маску, чтобы уведомить UE4, какие области имеют указанные типы растительности или материала ландшафта.

Создание ландшафтов. При создании нового ландшафта у вас есть несколько начальных вариантов (рисунок 8.2).

Во-первых, необходимо определить, какой материал будет использоваться для ландшафта. Данный пункт можно оставить без выбора, а сам материал указать уже после создания ландшафта.

После материалов и слоев материалов следует настройка трансформации нового ландшафта. Здесь вы можете указать, куда поместить новый ландшафт в пространстве мира, а также его размер и угол поворота.

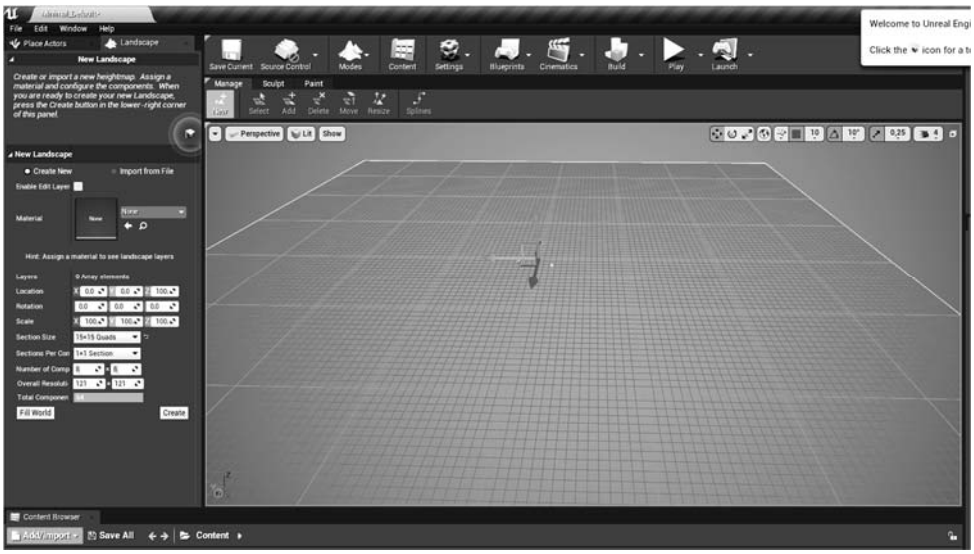

Рисунок 8.2 – Окно создания ландшафта

Следующий раздел элементов управления ландшафтами посвящен техникам уровней детализации (LOD, level of detailing) ландшафтов, которые вы можете изменять, чтобы ландшафт отображался быстро и эффективно. Этими элементами управления являются размер секции (Section Size) и количество секций в компоненте (Sections per Component). Они связаны с тем, сколько информации отображается игроку одновременно. Чем больше размер секции, тем меньше отображается внутри компонента или секции, что снижает нагрузку на процессор. Чем больше секций в компоненте, тем лучше UE4 распределяет и решает, какое качество отображать в каждой секции. В этом заключается тонкий баланс между обеспечением высокой частоты кадров и желаемым разрешением ландшафта.

Две последние настройки для создания нового ландшафта – это кнопки Fill

World и Create внизу панели Landscape. Кнопка Create подтверждает настройки и использует их для создания нового ландшафта. Кнопка Fill World создает ландшафт по всему доступному в текущий момент игровому пространству.

Управление ландшафтами

После создания нового ландшафта вам будут доступны новые опции для управления им. На вкладке Manage вы можете увидеть, что инструментом по умолчанию стала кнопка Selection, и выпадающее меню показывает новые варианты аспектов контроля только что созданного ландшафта. Некоторые из этих вариантов предназначены для удаления и добавления компонентов, эти варианты используются для изъятия секций из существующего ландшафта или добавления к нему новых (рисунок 8.3).

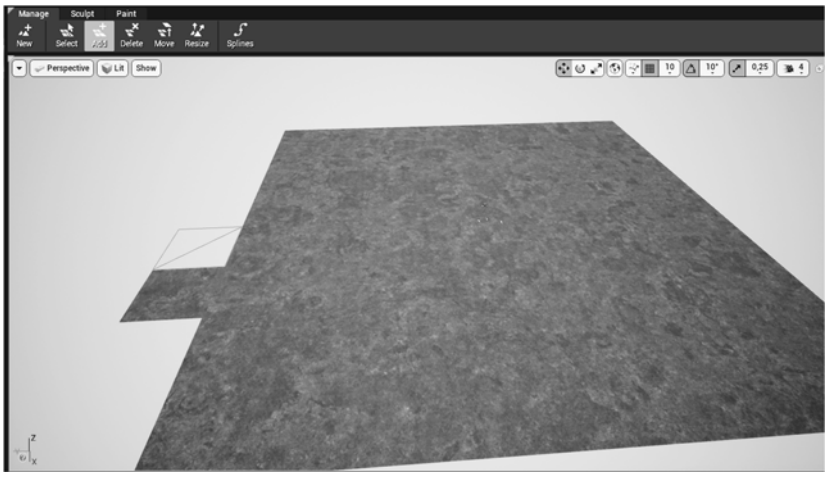

Рисунок 8.3 – Добавление компонентов в ландшафте

Вкладка Sculpt

После создания ландшафта вы можете начать создавать формы и объемы на ландшафте. Инструменты режима Sculpt представлены на рисунке 8.4.

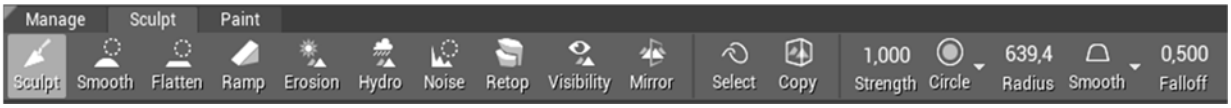

Рисунок 8.4 – Инструменты вкладки Sculpt

Sculpt. Этот инструмент скульптит выпуклые или вогнутые поверхности в ландшафтном меше. При применении этого инструмента с зажатой клавишей «Shift» будет происходить вдавливание ландшафта.

Smooth. Этот инструмент смягчает или сглаживает созданные инструментом Sculpt колебания высот между областями.

Flatten. Этот инструмент выравнивает ландшафт до высоты, указываемой при первом щелчке по ландшафту с активированным инструментом Flatten. Он

перемещает местность вверх или вниз, в зависимости от выбранного значения высоты.

Ramp. Этот инструмент соединяет две области, наклоняя ландшафт между двумя областями с постоянным изменением наклона между точками.

Hydro Erosion и Erosion. Эти инструменты моделируют общее изнашивание земли, которое происходит в мире, чтобы симулировать этот эффект на ландшафте игрового пространства.

Noise. Этот инструмент применяет общий шум к ландшафту и использует параметры настройки для определения объема и интенсивности.

Retopologize. Этот инструмент сокращает расстояние и различия между компонентами поверхности, чтобы уменьшить растяжение.

Visibility. Этот инструмент скрывает или делает видимыми выделенные поверхности в ландшафтном меше.

Mirror. Этот инструмент позволяет отразить элементы ландшафта.

Selection. Этот инструмент маскирует выделенные области ландшафтного меша.

Copy/Paste. Этот инструмент позволяет выделить секцию ландшафта, чтобы скопировать и перенести аналогичные параметры настройки высоты в другую секцию.

Настройки кисти

При работе с ландшафтом не малую роль играют не только выбранные инструменты редактирования, но и настройки кисти. Для кисти можно выбрать силу применения инструмента, тип кисти, радиус, тип затухания и его силу.

Существует 4 шаблона для типа кисти. Они представлены на рисунке 8.5:

Circle. Эта кисть является базовой. Она имеет форму окружности.

Alpha. Эта кисть использует указанную текстуру, как маску, использующую тона серого, как карта высот.

Pattern. Эта кисть использует повторяющийся шаблон.

Component. Эта кисть влияет на все элементы компонента редактируемой области.

Материалы ландшафта

Настройки ландшафтных материалов незначительно отличаются от настроек обычных материалов. Процесс создания материала для ландшафта тот же, что и для создания нового материала, за исключением, что смешивание слоев определяется с помощью специального нода в редакторе материалов, который называется Landscape Layer Blend. С этим нодом различные текстуры могут использоваться и распределяться в специальные слои, вызываемые в меню настроек ландшафта. После добавления этой ноды вы можете щелкнуть по симолу + на ноде, чтобы добавить слои (рисунок 8.6).

Вы можете добавлять и использовать множество слоев. Однажды помещенные в материал, текстуры, которые обычно объединены или смешаны в обычном материале, в данном случае направляются в слои нода Landscape Layer Blend

и затем помещаются в соответствующий вывод конечного материала, такой как базовый цвет или нормаль (рисунок 8.7).

Рисунок 8.5 – Типы кистей

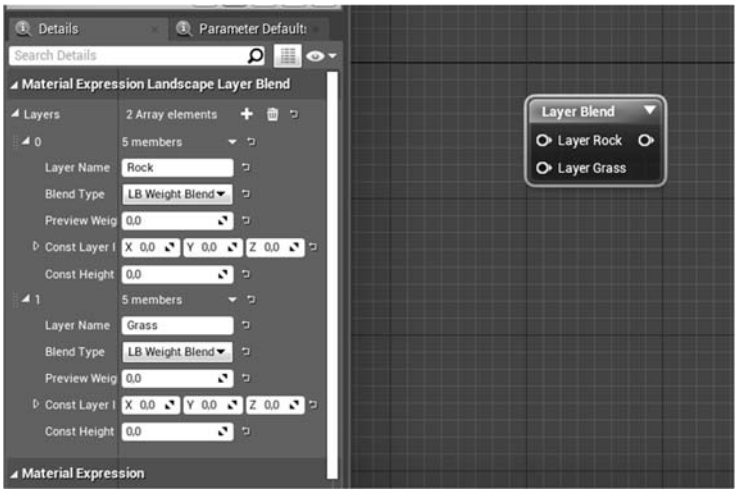

Рисунок 8.6 – Нода Landscape Layer Blend

Практические задания

Задание

Необходимо создать ландшафт, используя различные инструменты. При выдавливании элементов ландшафта необходимо использовать кисти с альфа-текстурой. Материал ландшафта должен включать в себя как минимум три слоя с различными настройками (цвет, растягивание и т. д.). На ландшафт необходимо добавить экземпляр данного материала. Добавить различные типы растительности. Ресурсы с альфа-текстурами, текстурами для покраски и растительностью можно найти по ссылке –https://drive.google.com/file/d/1gngmg9-0H4JgHN7cRS56qXDxIz6Ox8y/view?usp=sharing. Также можно использовать любые другие текстуры и модели на свой выбор.

Контрольные вопросы

1 Каковы основные инструменты для создания ландшафта в Unreal Engine?

2 Что такое Landscape Materials и как они влияют на визуализацию ландшафта?

3 Как создать и настроить текстуры для ландшафта?

4 Что такое слой (Layer) в контексте ландшафта и как он используются для управления различными материалами?

5 Каковы основные типы источников света доступны в Unreal Engine?

Список литературы

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

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