



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ

Робототехника и комплексная автоматизация (РК)

КАФЕДРА

Системы автоматизированного проектирования (РК6)

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

НА ТЕМУ:

*«Методика разработки игрового жанра RTS-проекта на
3D-движке Unreal Engine 5»*

Студент РК6И-71Б

(Подпись, дата)

Симуныю Э.Р.

И.О. Фамилия

Руководитель курсового проекта

(Подпись, дата)

Витюков Ф.А.

И.О. Фамилия

2023 г.

**Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)
(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ
Заведующий кафедрой РК6
А.П. Карпенко

« _____ » 20 ____ г.

**З А Д А Н И Е
на выполнение курсового проекта**

по дисциплине Модели и методы анализа проектных решений

Студент группы РКБИ-71Б

Симунью Эльвина Рувимбо
(Фамилия, имя, отчество)

Тема курсового проекта Методика разработки проектов в жанре RTS (Real Time Strategy) в 3D-движке Unreal Engine 5

Направленность КП (учебный, исследовательский, практический, производственный, др.) учебный
Источник тематики (кафедра, предприятие, НИР) кафедра

График выполнения проекта: 25% к 5 нед., 50% к 11 нед., 75% к 14 нед., 100% к 16 нед.

Задание. Разработайте систему выбора и отмены выбора персонажей игроков. Добавьте возможность подсчитывать и показывать собранные ресурсы и перемещать камеру в область выбранного игрока. Создавайте 3D-модели персонажей и зданий для размещения на карте.

Оформление курсового проекта:

Расчетно-пояснительная записка на 32 листах формата А4.

Перечень графического (илюстративного) материала (чертежи, плакаты, слайды и т.п.):

3 графических листа

Дата выдачи задания « » 202 г.

Руководитель курсовой работы

(Подпись, дата)

Витюков Ф.А.

И.О. Фамилия

Студент

(Подпись, дата)

Симунью Э.Р.

И.О. Фамилия

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

АННОТАЦИЯ

В данной статье рассматриваются подходы к разработке и анимации в Unreal Engine 5 для проектов жанра RTS (стратегия в реальном времени). Разработка системы выбора и отмены выбора персонажей игроков. Реализация анимации перемещения персонажей из одной точки в другую и даже выполнения таких действий, как добыча полезных ископаемых и нападение. Также описаны этапы оптимизации 3D-моделей зданий для дальнейшего использования в движке. Проведен анализ эффективности использования блюпринтов, проведено сравнение производительности реализующих его компонентов в Unreal Engine 5.

В расчёто-пояснительной записке 32 страницы, 28 рисунок, 3 графических листа.

СОДЕРЖАНИЕ

АННОТАЦИЯ	3
ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ	5
ВВЕДЕНИЕ	6
1.1 Ключевые особенности игр RTS:	7
2. Создание проекта и импорт активов.....	9
2.1 сцена с инструментом ландшафта	10
2.2 Перенос текстуры из Quixel Bridge в Unreal Engine	13
3 Blueprints.....	14
3.2 Создание зданий	16
3.3 Индикатор раздела	17
3.4 Модель игрового персонажа.....	19
3.4.1 Используйте компонент Blueprint, чтобы активировать наклейку выбора.	21
4 Входы и логика для отправки движения юнитам.....	22
5. Добавьте анимацию при движении и добыче полезных ископаемых.	23
6. Использование Game mode для хранения важной информации об игре	25
7. Выбор области RTS	27
8. Реализация панели "Здоровье"	28
9. Контроллер искусственного интеллекта	30
ЗАКЛЮЧЕНИЕ.....	31
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ.....	32

ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ

UE5 – трёхмерный движок Unreal Engine 5.

AActor (actor) – один из основных классов в UE5. Представляет собой любой объект, который может быть размещен на уровне.

UStaticMeshComponent (SMC) – компонент в UE5, позволяющий использовать Static Mesh.

Полигон – многоугольник, являющийся базовым компонентом 3D-сетки. Основные типы: треугольник (tri), четырёхугольник (quad) и n-gon (5 или более вершин).

Polycount – количество полигонов модели.

High-poly модель – максимально детализированная версия модели. Имеет высокий polycount, в основном используется для дальнейшего запекания текстур.

Low-poly модель – модель, содержащая относительно низкое количество полигонов, при этом сохраняющая основные геометрические свойства объекта.

Ретопология – процесс изменения топологии 3D-сетки модели с целью упрощения/улучшения.

Текстура – изображение, накладываемое на поверхность 3D-модели. Может содержать различные свойства поверхности, например: цвет, жёсткость (roughness), смещение (displacement), направление нормалей (normal map), и т.д.

Blueprint – a high level, visual scripting system that provides an intuitive, node-based interface that can be used to create any type of script events in the Unreal editor.

LOD (Level of Detail) – техника, позволяющая подменять разные по детализации версии модели в зависимости от дистанции между камерой и объектом.

CPU (Central Processing Unit) – центральный процессор.

GPU (Graphics Processing Unit) – графический процессор (видеокарта).

FPS (Frames per second) – количество кадров в секунду.

RHI – Rendering Hardware Interface в Unreal Engine 5.

ВВЕДЕНИЕ

При создании интерактивных элементов пользовательского интерфейса одной из основных задач является достижение удобства потенциального пользователя при взаимодействии с конечным продуктом. Оно, в свою очередь, состоит из множества аспектов. В контексте компьютерных игр можно выделить такие основные аспекты удобства, как простота управления, интуитивно понятный дизайн и комфортная работа.

Для реализации простой и понятной схемы управления необходимо четко понимать специфику продукта, продумывать различные сценарии использования, исследовать предметную область и рынок, анализировать качество уже изобретенных решений.

В данной статье рассматривается разработка некоторых элементов для проекта в жанре rts(real timestrategy).

1. Обзор RTS (Real Time Strategy)

Жанр игр «Real Time Strategy» (RTS) на протяжении десятилетий очаровывал геймеров своим захватывающим игровым процессом и стратегическими задачами. В этих играх игроки берут на себя роль командира, направляя армии, управляя ресурсами и принимая важные решения в режиме реального времени.

1.1 Ключевые особенности игр RTS:

Геймплей в реальном времени. В отличие от пошаговых стратегических игр, действия в стратегиях в реальном времени разворачиваются в реальном времени, требуя от игроков быстрого мышления и быстрого принятия решений.

Управление ресурсами. Эффективное управление ресурсами, такими как минералы, золото или энергия, имеет решающее значение в играх RTS. Игроки должны собирать ресурсы, строить здания и производить юниты, чтобы расширять свою армию и доминировать на поле битвы.

Строительство базы. В стратегиях в реальном времени очень важно построить сильную и хорошо защищенную базу. Игроки строят различные структуры для производства юнитов, исследования технологий и укрепления своих позиций от атак противника.

Управление юнитами: в стратегиях в реальном времени представлено огромное количество уникальных юнитов, каждый из которых имеет свои сильные и слабые стороны. Игроки должны стратегически размещать и маневрировать своими отрядами, чтобы получить преимущество над противниками.

Многопользовательские сражения. Один из самых захватывающих аспектов стратегий в реальном времени — многопользовательские сражения. Игроки могут проверить свои навыки против друзей или соревноваться в онлайн-матчах, разрабатывая стратегии и перехитрывая противников со всего мира.

Популярные серии игр RTS:

- StarCraft
- Age of Empires
- Warcraft III
- Command & Conquer

2. Создание проекта и импорт активов

Чтобы загрузить нужные нам активы, мы сначала открываем лаунчер эпических игр, переходим к образцам и в устаревших образцах выбираем образцы стратегических игр. Образцы стратегических игр поставляются с кодом C++ ai и множеством других вещей, нам нужны только активы, которые включают в себя здания и текстуры камней.

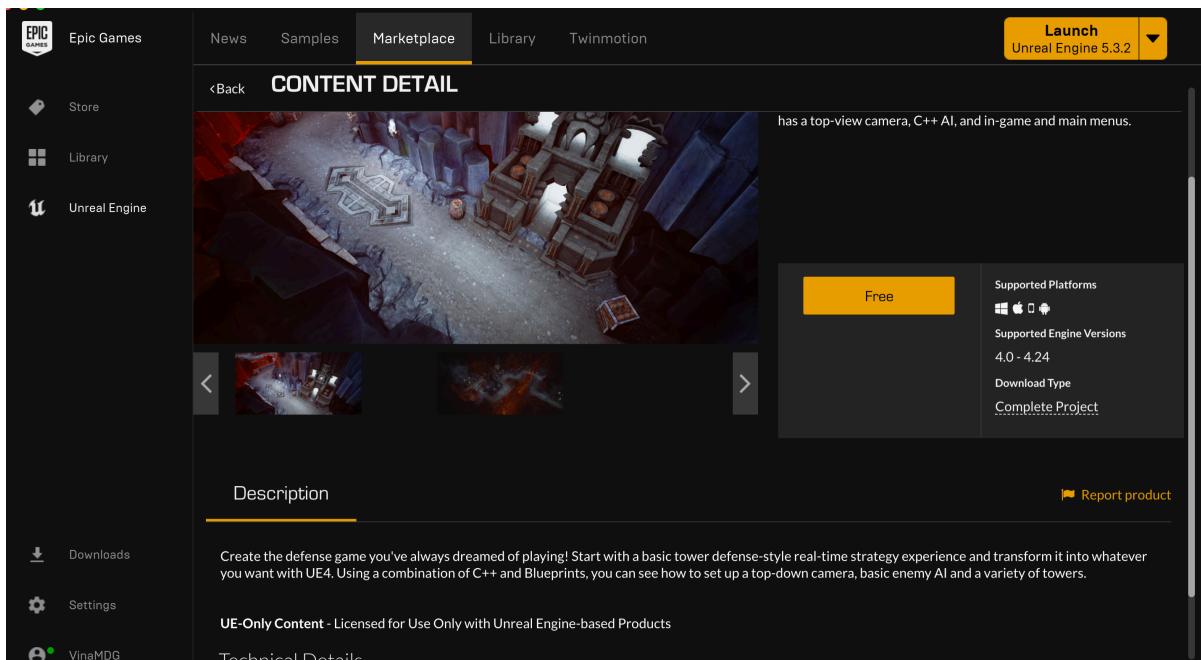


Рисунок 1. Epic Games Launcher

Чтобы создать проект в Unreal Engine, мы нажимаем кнопку запуска, как показано на рисунке выше, и следуем следующим настройкам:

- В меню выбора проекта выберите <Игры> (рисунок 2);
- В следующем меню выберите пункт Пустой (рисунок 2);
- В последнем меню вам нужно выбрать, нужен ли проекту чертеж или C++.

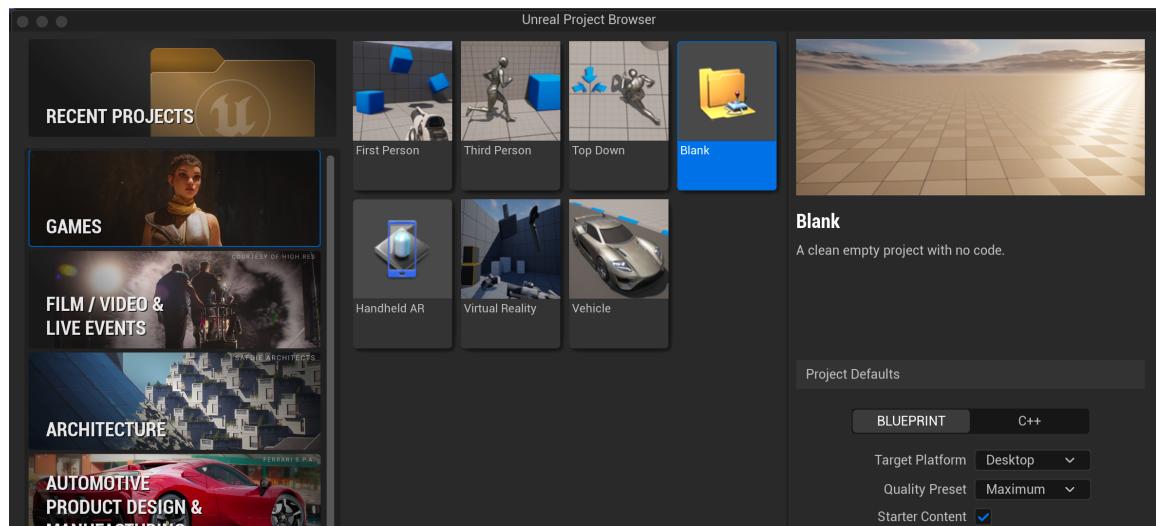


Рисунок 2. Создание проекта

2.1 сцена с инструментом ландшафта

Чтобы создать сцену, вы создаете пустой открытый мир из меню <Файл> и выбираете <Новый уровень>, затем выбираете <Пустой открытый мир> (рисунок 3).

После этого вы создаете новую папку в <content>, куда мы поместим ресурсы. создай быстро эту сцену создай свет

Затем создайте световой люк и поместите атмосферный свет, затем объемное облако, высокий туман и атмосферу неба. Чтобы начать редактирование ландшафта, редактор должен находиться в ландшафтном режиме, после этого выбираем создать (картинка 4)

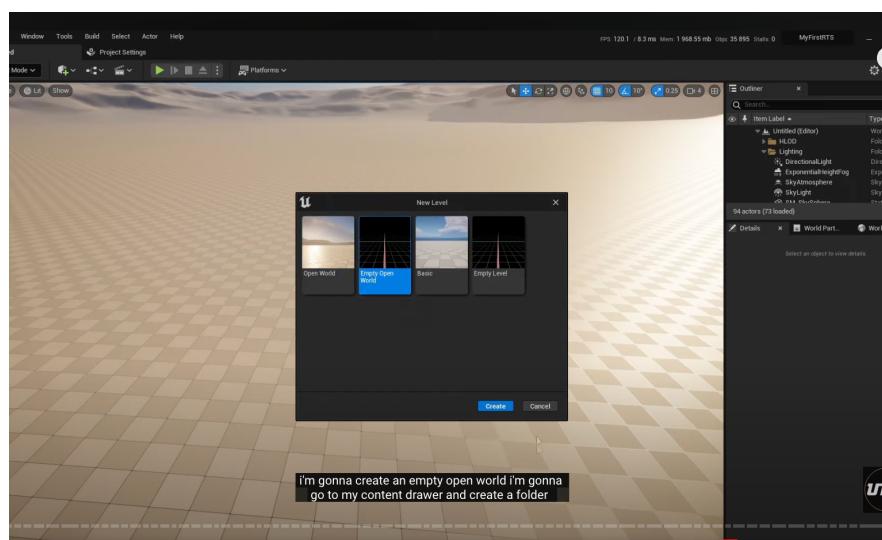


Рисунок 3. Создание проекта

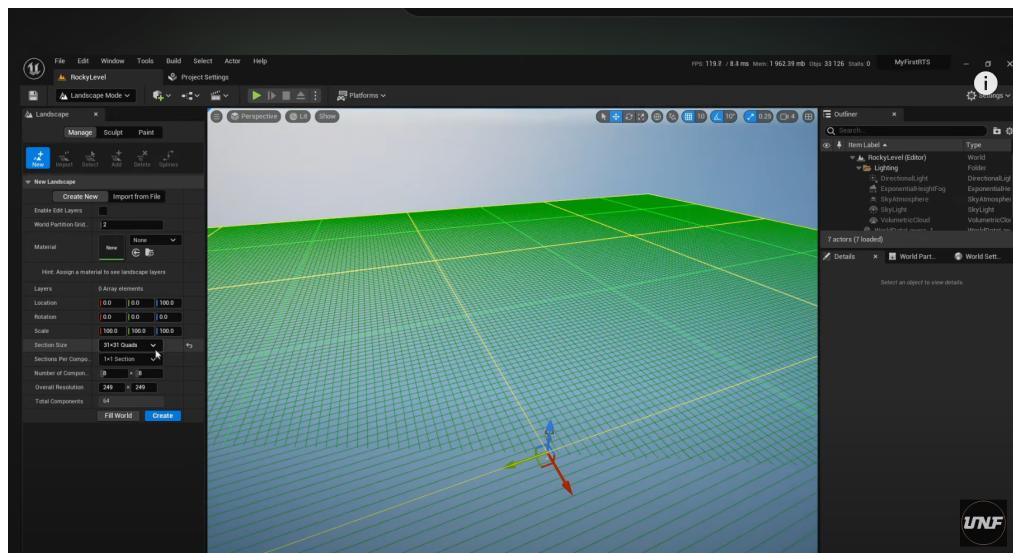


Рисунок 4. Landscape mode

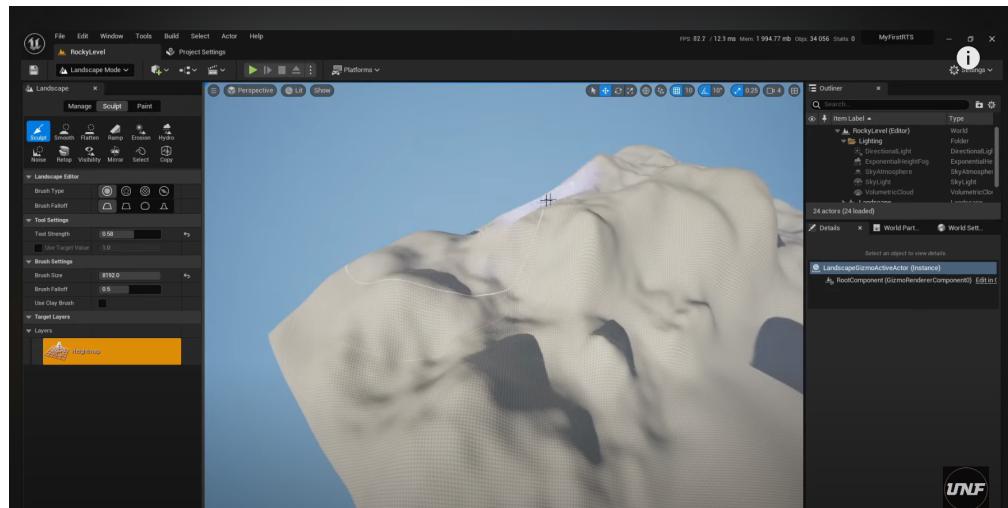


Рисунок 5. Landscape mode

Используем инструменты для лепки, чтобы создать горы, ландшафты и холмы (рисунок 5).

Возвращаемся в <<Режим выбора>> в редакторе, ресурсы, которые мы скачали из образцов в создаваемую нами игру, будут находиться в папке <Strategy game Assets > в <<browser content >>.

Затем мы перетаскиваем нужную модель на карту и изменяем ее размер (Рисунок 6).



Рисунок 6. Select mode

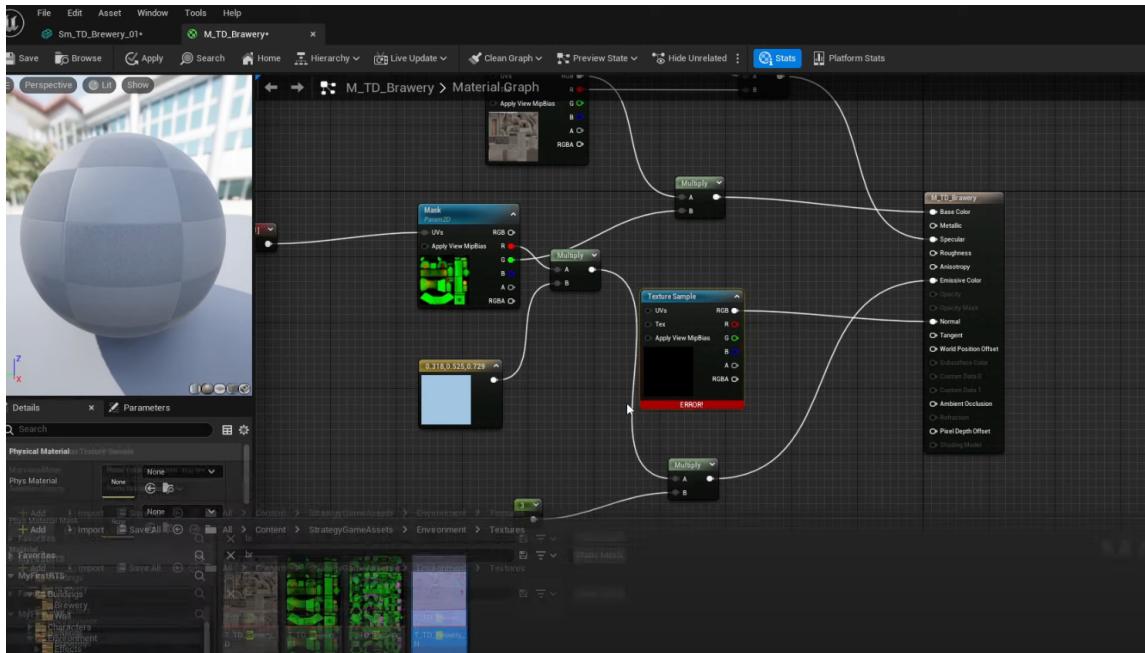


Рисунок 7. Landscape mode

Редактируем текстуру здания в меш-Blueprint.

2.2 Перенос текстуры из Quixel Bridge в Unreal Engine

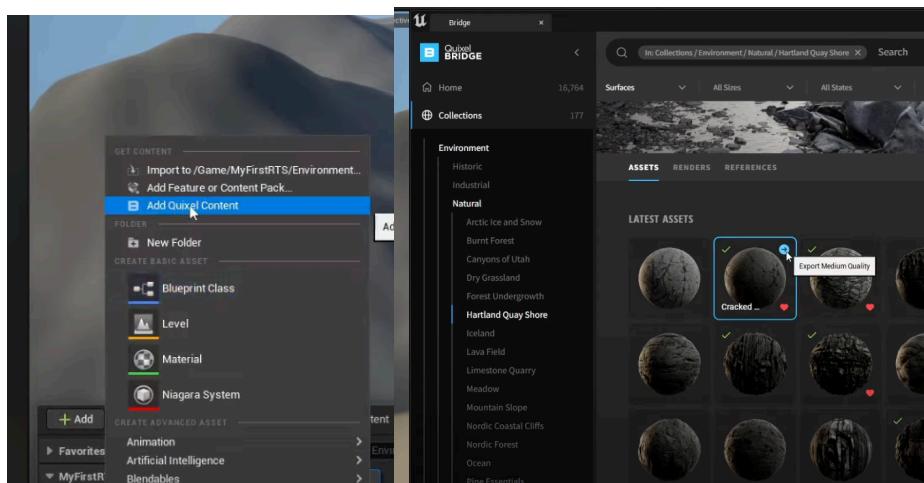


Рисунок 8. Quixel Bridge

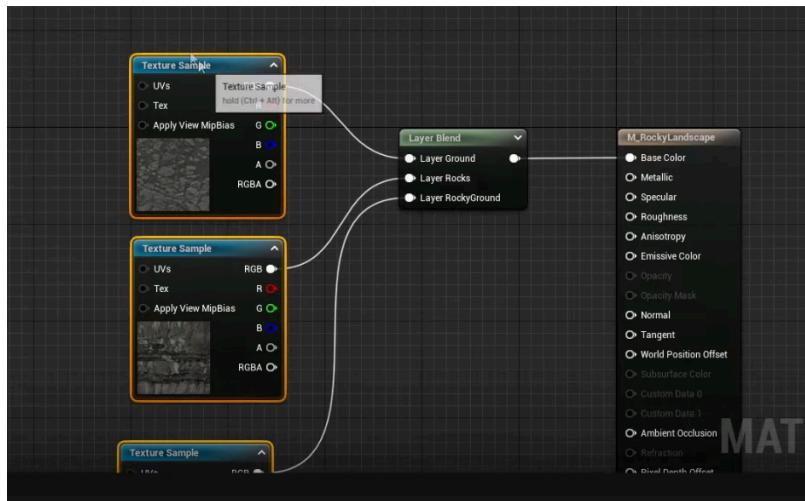


Рисунок 9. Материалы из Quixel, подключенные к материалу ландшафта.

Чтобы перенести текстуру, вы выбираете <<add>> в браузере контента, затем выбираете «Добавить содержимое Quixel» (рис. 8), затем появляется всплывающее меню Quixel, оттуда вы выбираете нужный материал и текстуры. Они будут автоматически добавлены в проект, затем мы перетаскиваем ресурсы в редактор, подключаем узлы ресурсов к модели (Рисунок 9).

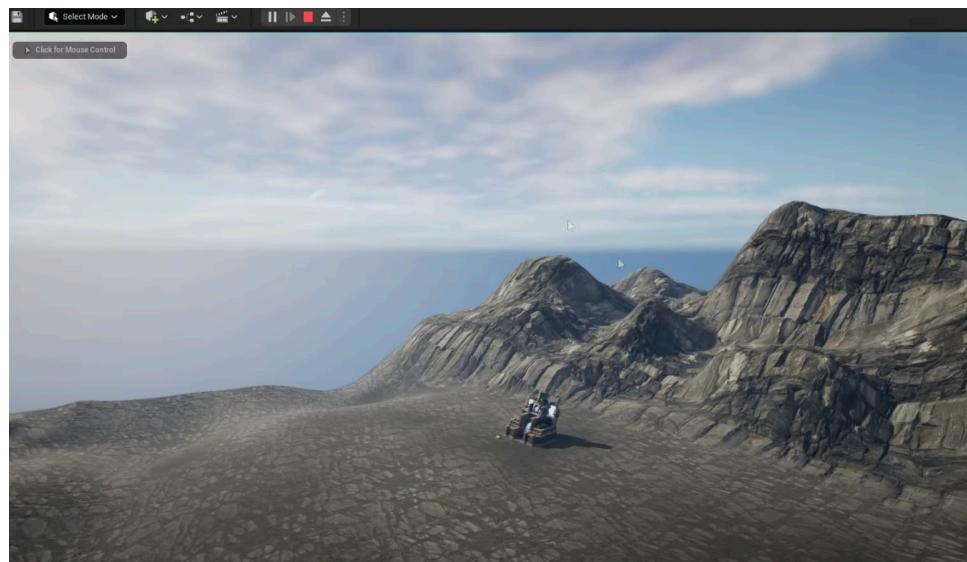


Рисунок 10.

3 Blueprints

Blueprint создаются внутри Unreal Editor визуально, а не путем ввода кода, и сохраняются как ресурсы в пакете контента. По сути, они определяют новый класс или тип Actor, который затем можно разместить на картах как Instances, ведущие себя как любой другой тип Actor.

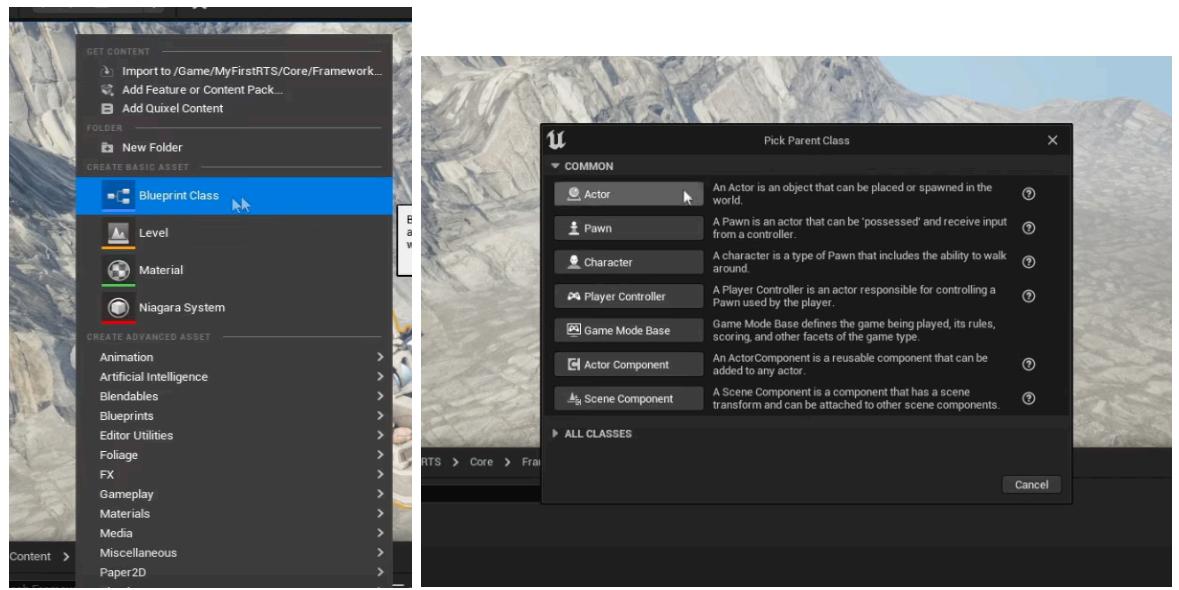


Рисунок 11.

Первоначально нам нужно создать классы инфраструктуры, необходимые для функционирования нашей игры. Для этого мы создаем классы Blueprints внутри Unreal Engine для создания взаимодействия между объектами в игре.

Для этого мы щелкаем правой кнопкой мыши по созданной нами папке с именем framework и затем выбираем класс Blueprint. Чтобы добавить функциональность, каждый Blueprint будет производным от другого класса. Существуют разные типы родительских классов в Blueprint, как показано на (рис. 10).

Тип класса	Описание
Actor	это объект, который можно разместить или создать в мире.
Pawn	это Actor, которым можно «овладеть» и получать данные от Контроллера.
Character	то Pawn , которая включает в себя способность ходить, бегать, прыгать и многое другое.
Playercontroller	это actor, ответственный за управление пешкой, используемой игроком
Game mode	режим определяет игру, ее правила, подсчет очков и другие аспекты типа игры.

3.1 Камера игрока

Поскольку в игре мы собираемся много перемещаться и приближать камеру к зоне игрока, для реализации этого мы создаем камеру в классе PAWN BLUEPRINT.

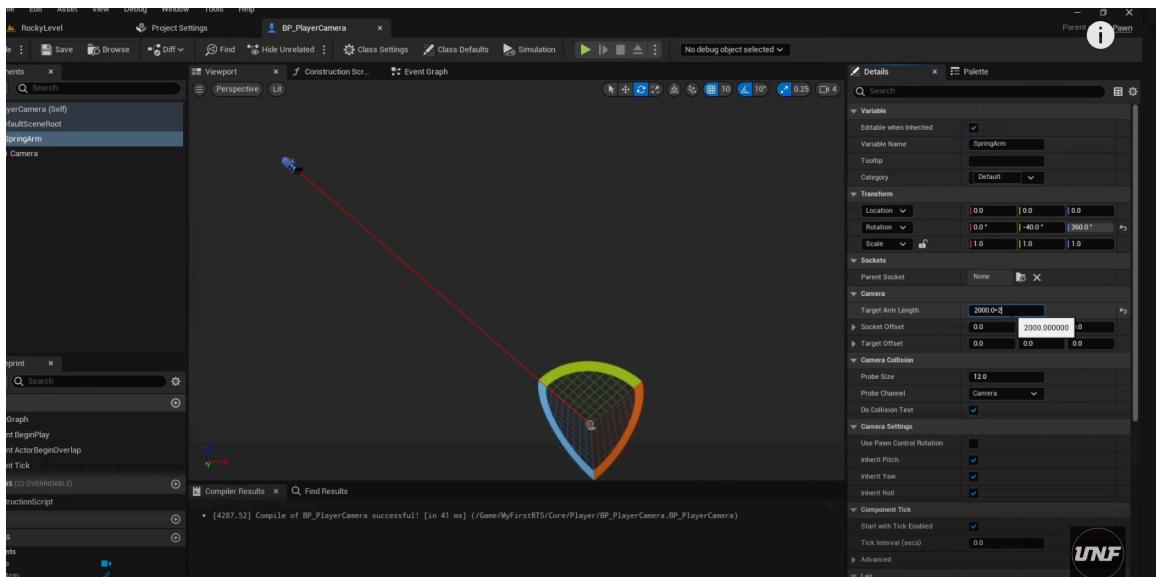


Рисунок 11. камера BLUEPRINT в редакторе BLUEPRINT

Мы можем контролировать расстояние от земли на чертеже. В BLUEPRINT мы создаем функции, которые будут сравнивать и вычислять положение мыши, чтобы камера следовала за мышью, когда она достигнет края.



Рисунок 12. камера BLUEPRINT в редакторе BLUEPRINT показ функций

3.2 Создание зданий

Для наших зданий мы будем использовать Actor BLUEPRINT, мы собираемся использовать Static Mesh и включить концепцию программирования,

называемую наследованием, поэтому мы можем создать базовый класс, а затем создать дочерний класс. В панели содержимого, где находится здание BLUEPRINT, мы щелкаем по нему правой кнопкой мыши и создаем класс BLUEPRINT на основе этого здания BLUEPRINT.

3.3Индикатор раздела

В нашей игре нам нужно показать, когда выбрано здание, и для этого использовалась декаль, которую можно выбрать с помощью значка прямоугольника в редакторе (рисунок 13). Затем мы создаем экземпляр материала для наклейки, который придаст ей форму и цвет.

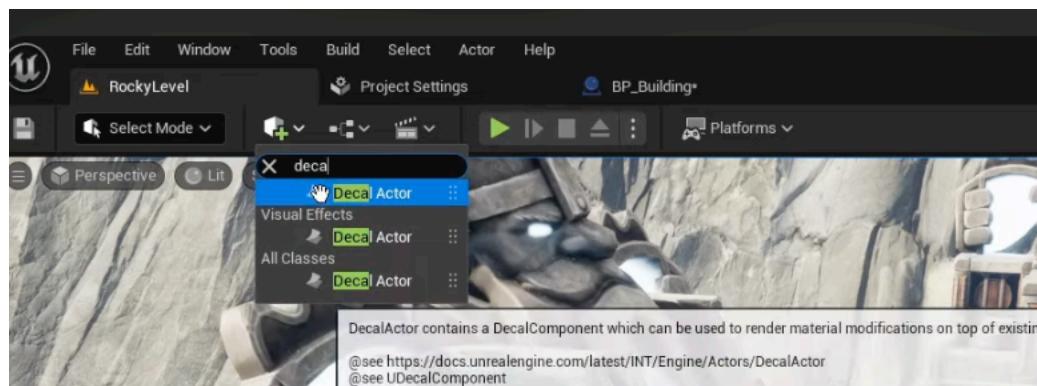


Рисунок 12. Выбор decal

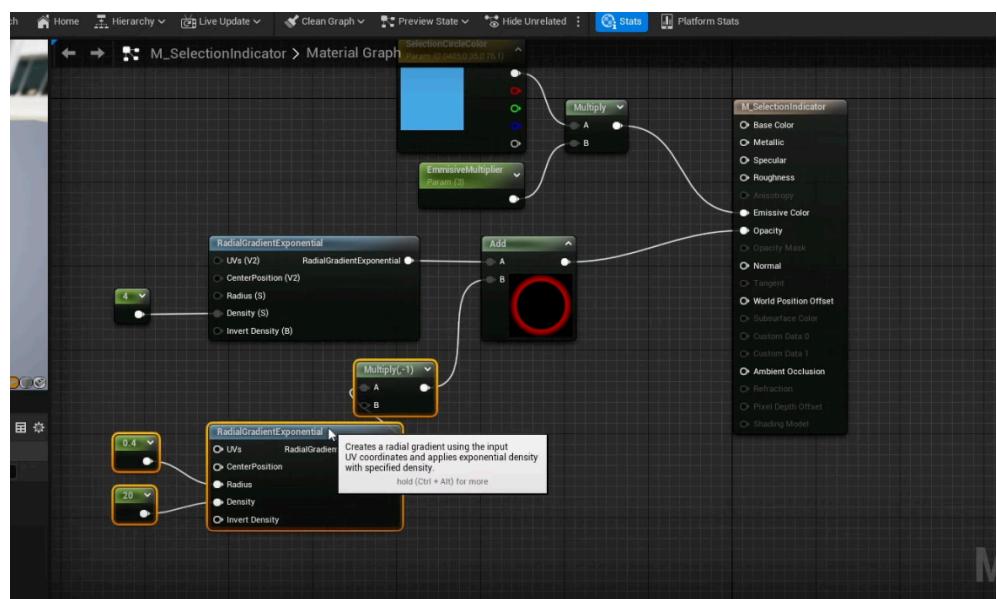


Рисунок 13. Редактирование формы и цвета Decal

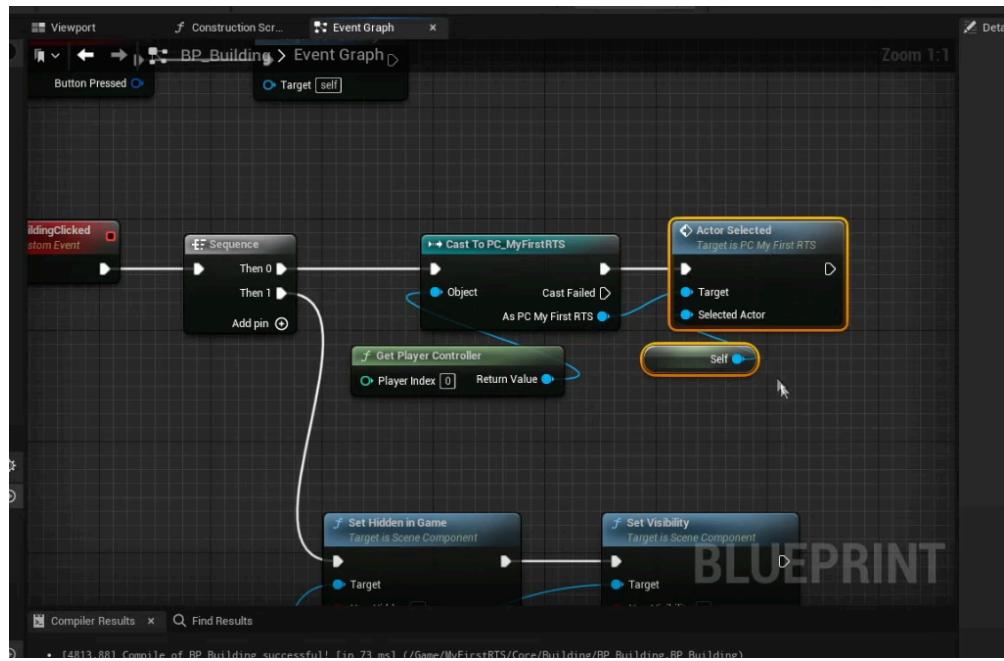


Рисунок 14. Редактирование формы и цвета Decal

Реализуя индикатор выбора, мы добавляем Декаль в BLUEPRINT здания из видового экрана и делаем то же самое с персонажами. Мы создаем такие функции, как «Выбрать событие», которые будут вызывать декаль в Player controller Blueprint всякий раз, когда выбирается здание. (Рисунок 14) .

Классы, расширяющие функциональность базовых классов игрока ACardPPawn – класс, производный от APlayerPawn. Добавляет поддержку актуального в контексте задачи управления игрока. Доступен выбор, добавление, удаление карты; приближение к выбранной части поля, просмотр карт в выбранной группе;

1. APlayerController – класс, производный от APlayerController. Позволяет добавлять UMG-виджеты, задает пользовательские настройки графики и ввода;
2. APlayerCameraManager – класс, производный от APlayerCameraManager. Обновляет значение FOV (Field of View) в зависимости от соотношения сторон окна приложения.

3.4 Модель игрового персонажа

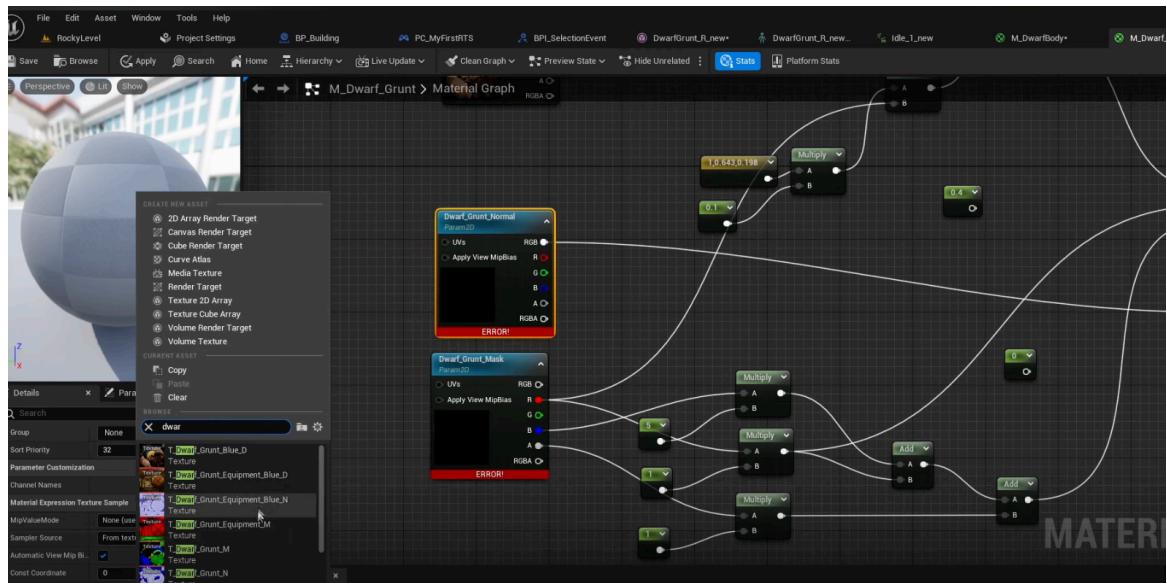


Рисунок 15. Редактирование формы и цвета для 3D модели персонажа



Рисунок 16. 3D модели персонажа

Создайте класс проекта под названием `playercontroller`, `playercontroller` будет работать как мозг для любого персонажа, которого мы можем использовать внутри unreal, поэтому всякий раз, когда вы скажете ему,

куда идти, он так и сделает. Модифицируем материал Instance модели персонажа, добавляем текстуру и цвет персонажа.(Рисунок 15)

Для нашего персонажа мы создаем компонент актера Blueprint и назовем его компонентом юнита, и мы увидим, как мы используем этот компонент для большинства вещей.

нам нужно правильно создать этого работника, поэтому давайте создадим рабочий блок BLUEPRINT. Теперь у нас будет много рабочих, у нас будут разные типы модулей. AGameModeBase – базовый класс игрового режима. На данный момент используется для тестирования текущего решения. Создает два APlacementManager'a: собственный и вражеский, а также инициализирует их. Хранит в себе указатели на APlayerPawn, ACardPlayerController и APlacementManager игроков в случае их существования. базовый класс игрового режима. На данный момент используется для тестирования текущего решения. Создает два APlacementManager'a: собственный и вражеский, а также инициализирует их. Хранит в себе указатели на APlayerPawn, ACardPlayerController и APlacementManager игроков в случае их существования.

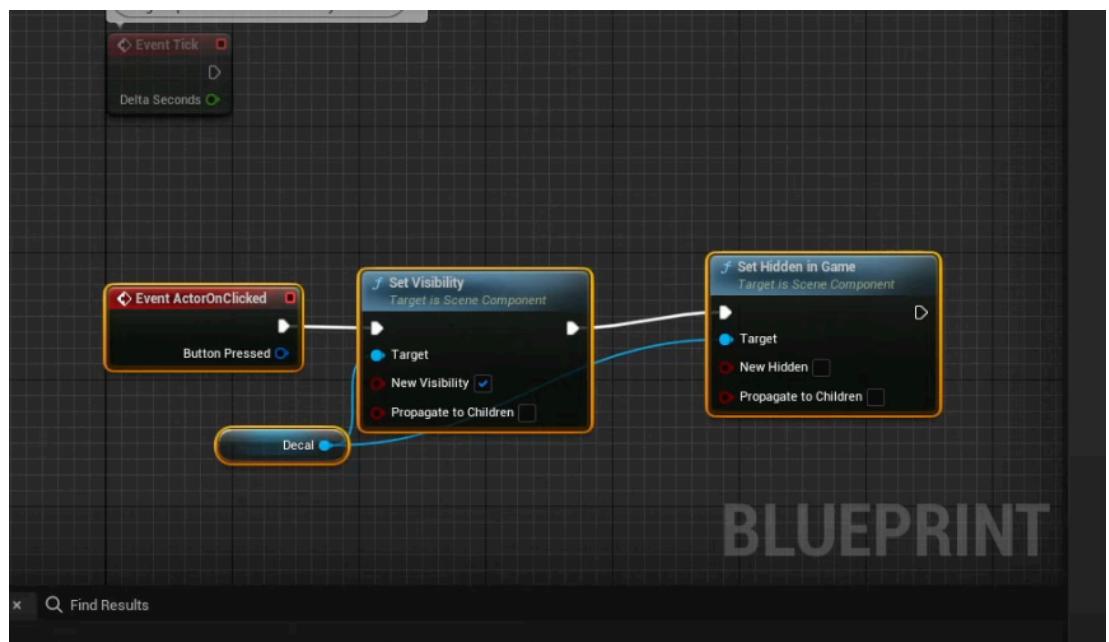


Рисунок 17. Редактирование формы и цвета Decal для 3D модели персонажа

3.4.1 Используйте компонент Blueprint, чтобы активировать наклейку выбора.

Чтобы активировать декаль, мы используем функцию Get Component by Class в BLUEPRINT для наших основных функций в игровом режиме. Она ищет массив компонентов и возвращает первый встреченный компонент указанного класса.

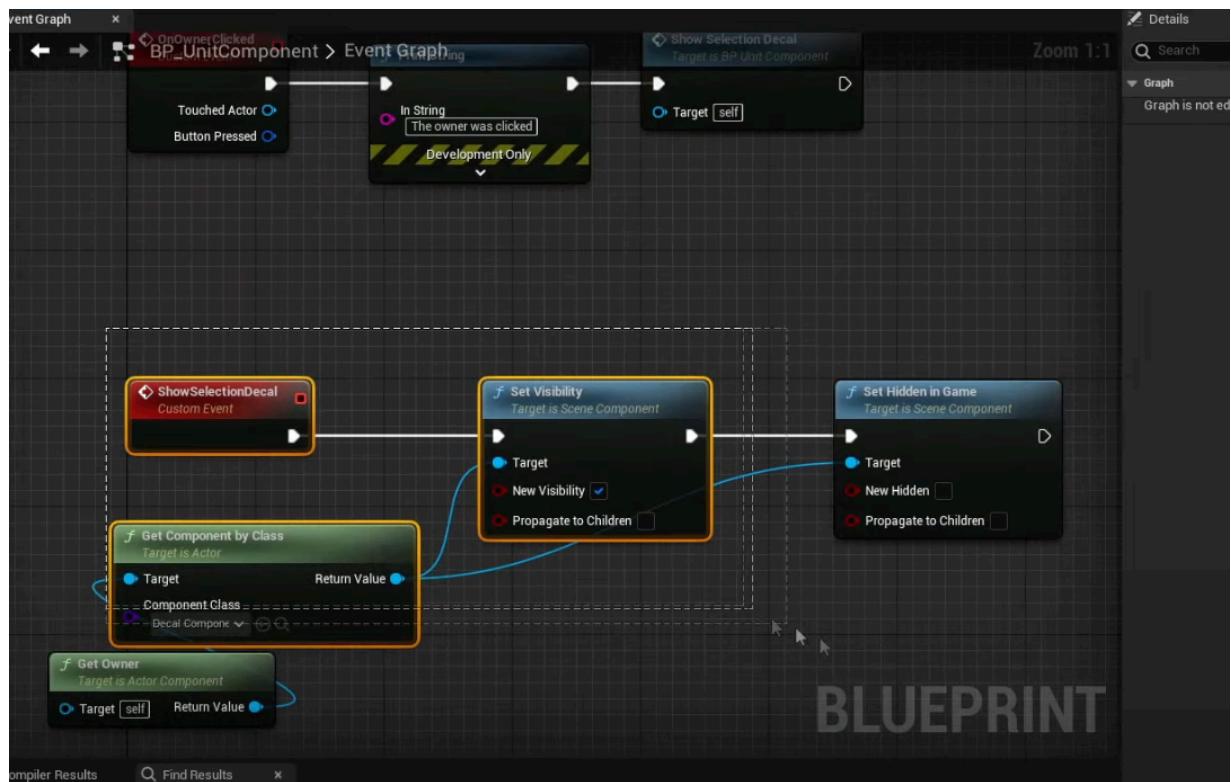


Рисунок 18. функций Get Component by Class

Изначально для каждого игрока создается менеджер выкладки (APlacementManager), а в нем два ряда, нижний из которых унаследован своими правилами расположения карт (композитный ряд). Каждый из рядов имеет ограничение по числу групп, но не по количеству однотипных карт в группе. Более того, число рядов не ограничено лишь двумя: при необходимости можно добавить дополнительные, изменив масштаб всех рядов в менеджере. На рис. 3 изображён пример реализации с использованием дополнительных рядов:

4 Входы и логика для отправки движения юнитам

Группировка карт происходит их типу. Группы не имеют строгого ограничения по количеству карт в них. Для удобства восприятия групп и работы с ними используются следующие приёмы:

- изначально задается ограничение по количеству отрисовываемых карт в группе. Если текущее число карт не превышает соответствующее значение, добавление новой карты явно отразится в изменении внешнего вида группы. В противном случае, рядом с ней появится счётчик, показывающий актуальное количество карт;
- поскольку реальное число карт в группе может отличаться от количества видимых, необходимо добавить возможность выбора из нее элементов. Для этого создан дополненный вариант ряда, имеющий задний фон. В случае раскрытия группы для выбора он появляется над менеджером с исходной группой и перекрывает его собой (рис. 20).

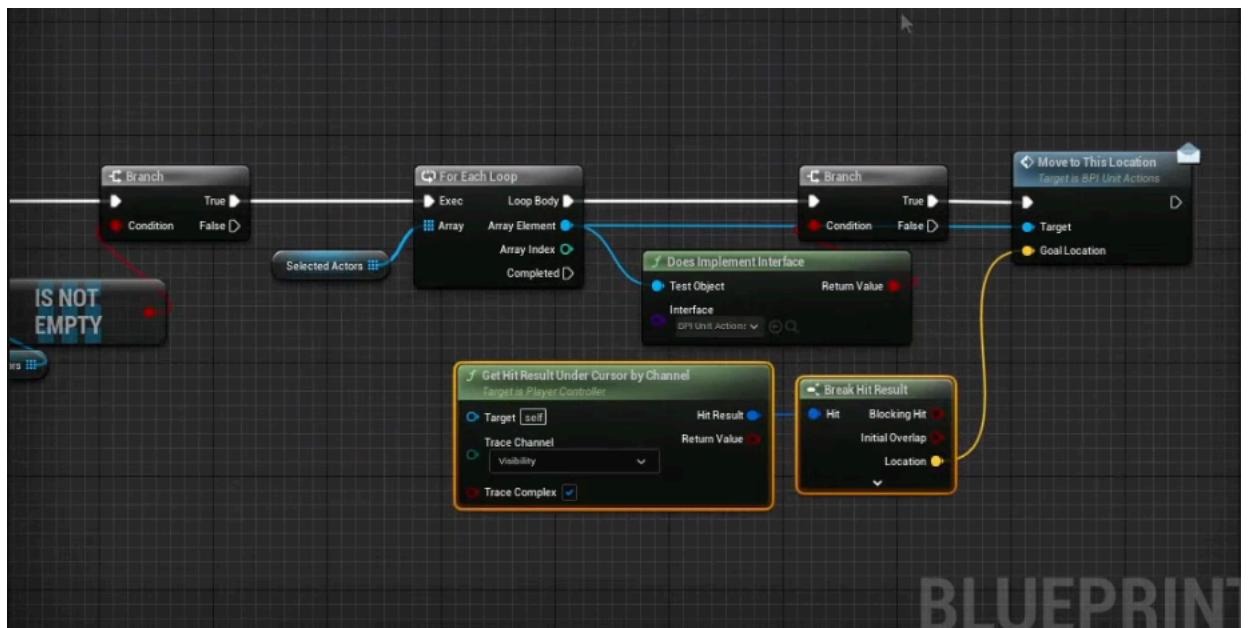


Рисунок 19. Функции для отправки сообщения отряду о необходимости
перемещения

5. Добавьте анимацию при движении и добыче полезных ископаемых.

Монтажи анимации — это мощные ресурсы в Unreal Engine, которые позволяют разработчикам комбинировать анимацию и управлять ею в одном активе. Они предоставляют универсальный набор инструментов для создания сложных последовательностей анимации, управления воспроизведением анимации и интеграции анимации в Blueprints и Animation Blueprints. В этом обзоре мы рассмотрим ключевые особенности и возможности анимационных монтажей в Unreal Engine.

Ключевые особенности анимационных монтажей:

Управление анимацией: Animation Montages позволяют разработчикам управлять воспроизведением анимации, включая запуск, остановку, наложение и зацикливание анимации. Они предоставляют возможность упорядочить и синхронизировать несколько анимаций для персонажей или объектов.

Интеграция чертежей: монтажи анимации можно легко интегрировать в чертежи и чертежи анимации, что позволяет разработчикам запускать и контролировать последовательности анимации с помощью визуальных сценариев. Это обеспечивает гибкость в создании динамичной и отзывчивой анимации игрового процесса.

Смешение анимации: монтажи поддерживают смешивание различных анимаций, обеспечивая плавные переходы и плавное смешивание состояний анимации. Это позволяет разработчикам создавать естественные и реалистичные движения персонажей.

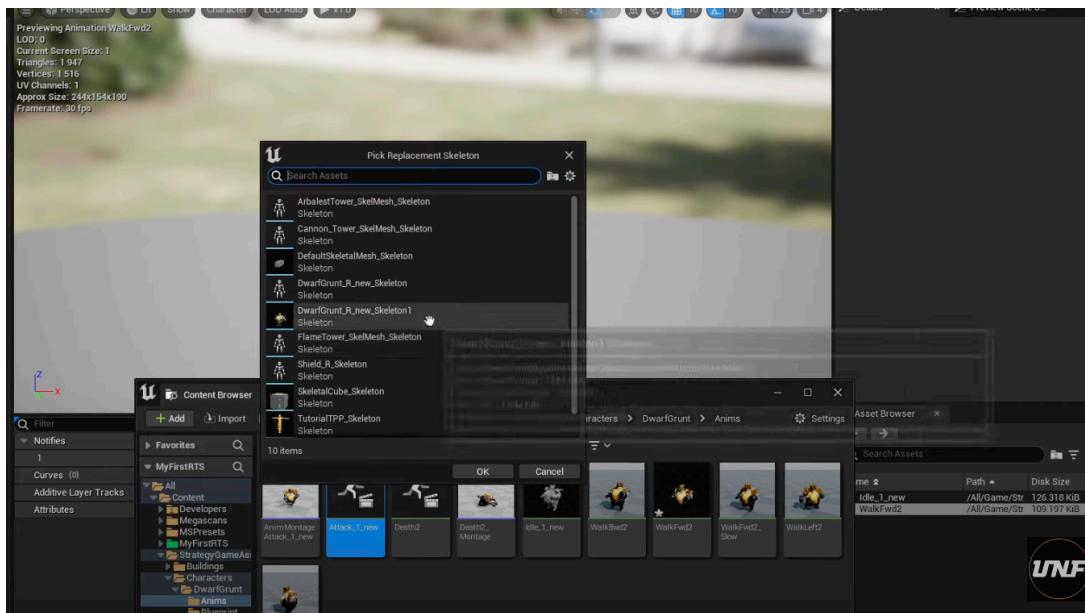


Рисунок 20. Выбор анимации для бега

Разделы анимации. Монтаж можно разделить на разделы, которые представляют собой определенные части анимационной последовательности. Эти разделы можно использовать для запуска событий, применения игровой логики или синхронизации других игровых систем с анимацией. Уведомления об анимации: Animation Montages поддерживают уведомления об анимации, которые представляют собой события, запускаемые в определенных точках во время воспроизведения анимации. Уведомления можно использовать для запуска звуковых эффектов, эффектов частиц или других игровых событий.

Дочерние монтажи: Монтажи могут быть организованы иерархически, при этом дочерние монтажи наследуют анимацию от родительских монтажей. Это позволяет создавать модульные настройки анимации и упрощает управление сложными анимационными системами. **Настройка и редактирование:** Unreal Engine предоставляет комплексный редактор анимационных монтажей, позволяющий разработчикам редактировать и настраивать монтажи визуально. Сюда входит настройка времени анимации, настройки смешивания и управление дорожками анимации.

6. Использование Game mode для хранения важной информации об игре

Он контролирует различные аспекты, включая присоединение игроков, приостановку игры, переходы между уровнями, условия победы и многое другое. Игровой режим действует как мост между игровым миром и игроком, управляя логикой игры и обеспечивая основу для игровой механики.

В игру мы добавили деятельность по добыче полезных ископаемых и собираемся поместить в наш план функции и переменные, которые помогут нам хранить информацию в игре и отображать ее. Мы импортируем значки, которые будут показывать накопление в нашем интерфейсе.

Виджеты помещаются в Blueprint, виджеты — это основная форма, которую мы можем легко создать. Пользовательский интерфейс внутри нереальных движков должен что-то показывать. Мы решаем размер виджета, который мы хотим, чтобы он закрывал экран, поэтому перетаскиваем панель холста. Это холстовые панели, которые можно менять в том размере, который нам нужен.(Рисунок 21)

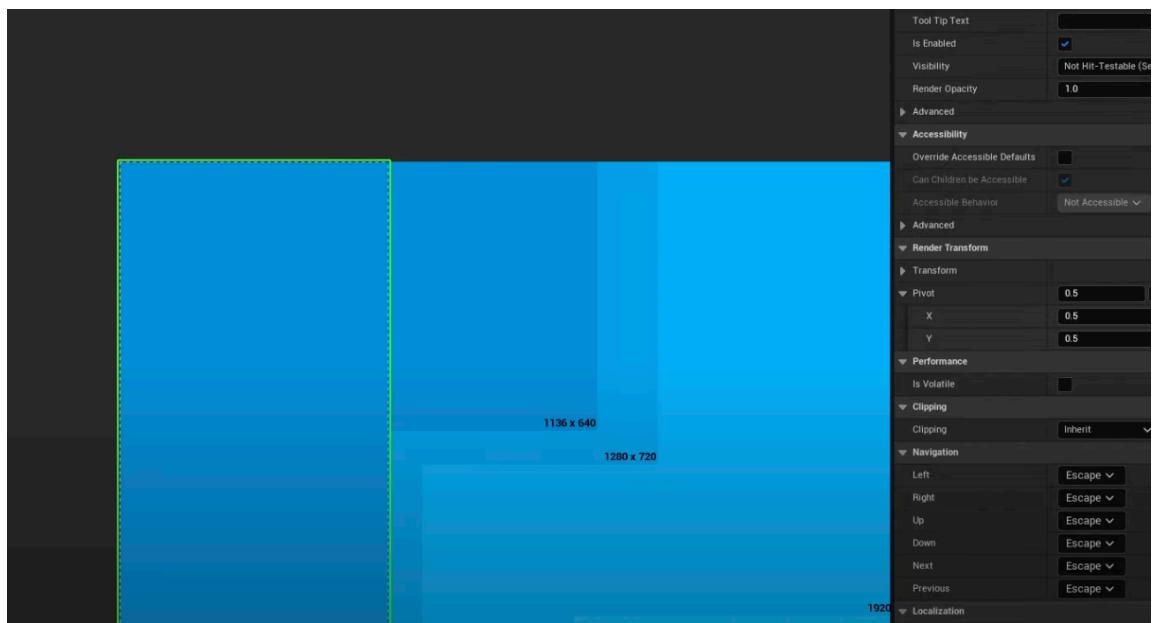


Рисунок 21. Выбор размера виджета для установки в игру

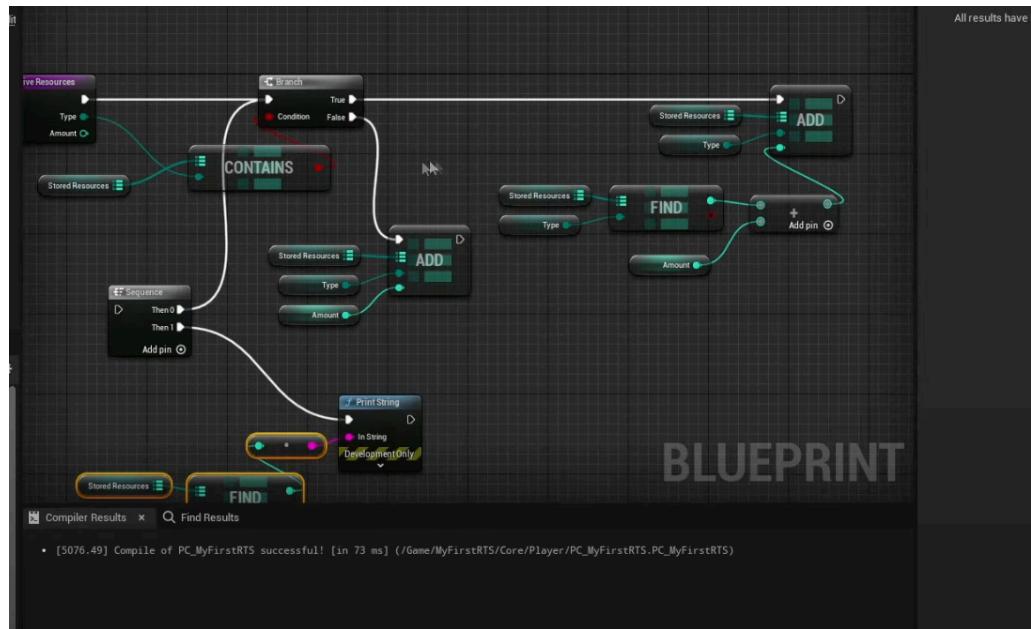


Рисунок 22. Функции, отвечающие за хранение информации



Рисунок 23. Результат

Однако при использовании общего пула инстансов HISMC будет прикреплен к APlacementManager'у, а не к конкретному ряду. В таком случае при скейлинге ряда (ARow) с инстансами желаемый результат не будет достигнут. Для решения данной проблемы создадим дополнительные USceneComponent'ы для каждой карты в ряду, прикрепим их к корневому компоненту ряда. Рассчитанные положения карт в ряду применим к новым USceneComponent'ам, а затем обновим значения абсолютных transform'ов инстансов HISMC до

текущих значений абсолютных transform'ов соответствующих USceneComponent'ов.

7. Выбор области RTS

Этот тип выбора предполагает удержание комбинации клавиш и перетаскивание мышью для определения поля. Все Актеры внутри поля будут выбраны или отменены в зависимости от комбинации клавиш и кнопки мыши, которая удерживается во время перетаскивания мыши.

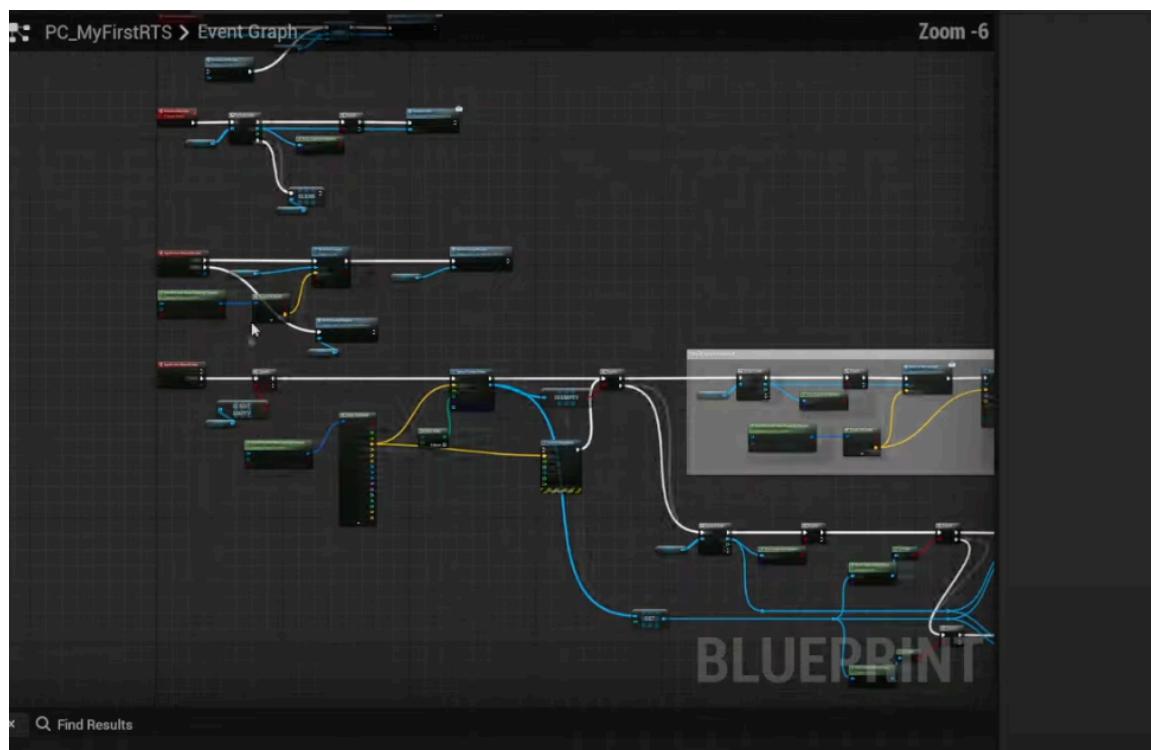


Рисунок 24. Визуализация использования функций для вызова Marquee

В данном случае анимация перемещения имеет фиксированную длительность, не зависящую от расстояния. Для реализации такого анимированного перехода необходимо иметь:

1. исходный transform;
2. текущий transform (в начальный момент времени равен исходному);
3. целевой transform (рассчитывается заранее).

Скорость интерполяции местоположения можно задать как:



Рисунок 25. Визуализация Marquee в игра

8. Реализация панели "Здоровье"

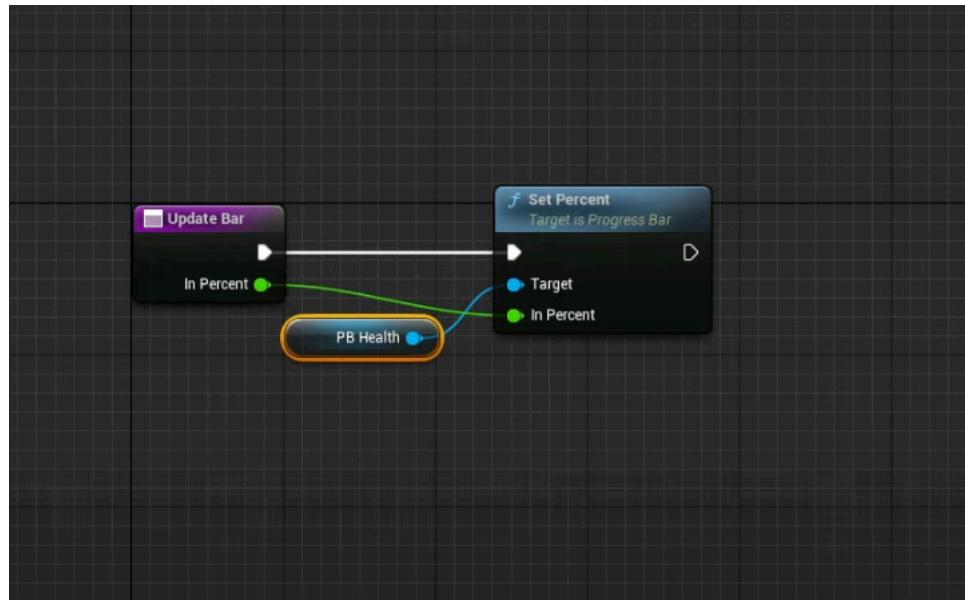


Рисунок 26. Реализация панели "Здоровье" в Blueprint

Поскольку модели карт в рамках одного менеджера одинаковы, для оптимизации их отрисовки можно применить инстансинг. Основная его цель – понижение количества draw calls (вызовов отрисовки). CPU зачастую является

бутылочным горлышком, поскольку при большом количестве простых для исполнения draw calls GPU справляется со своими задачами слишком быстро.

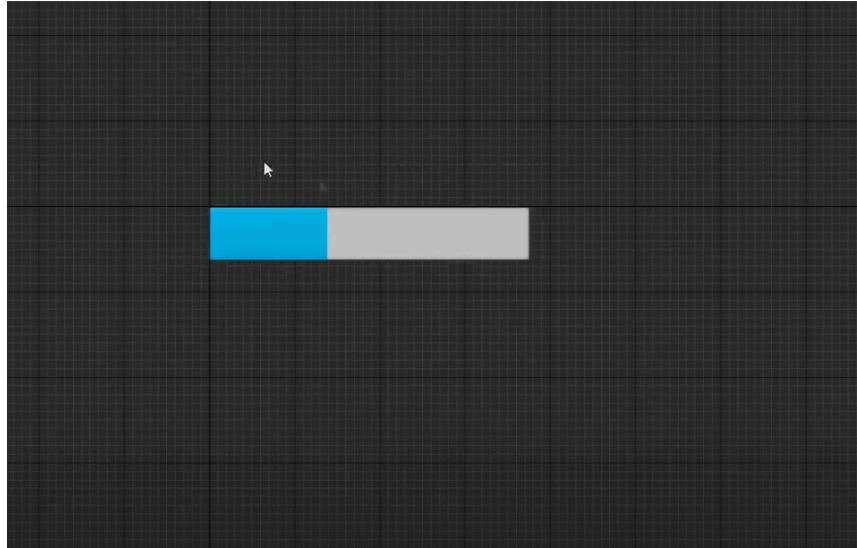


Рисунок 27. Реализация панели "Здоровье" в Blueprint

Инстансинг частично решает данную проблему: рендеринг однотипных объектов происходит за один проход, тем самым понижая нагрузку на CPU.

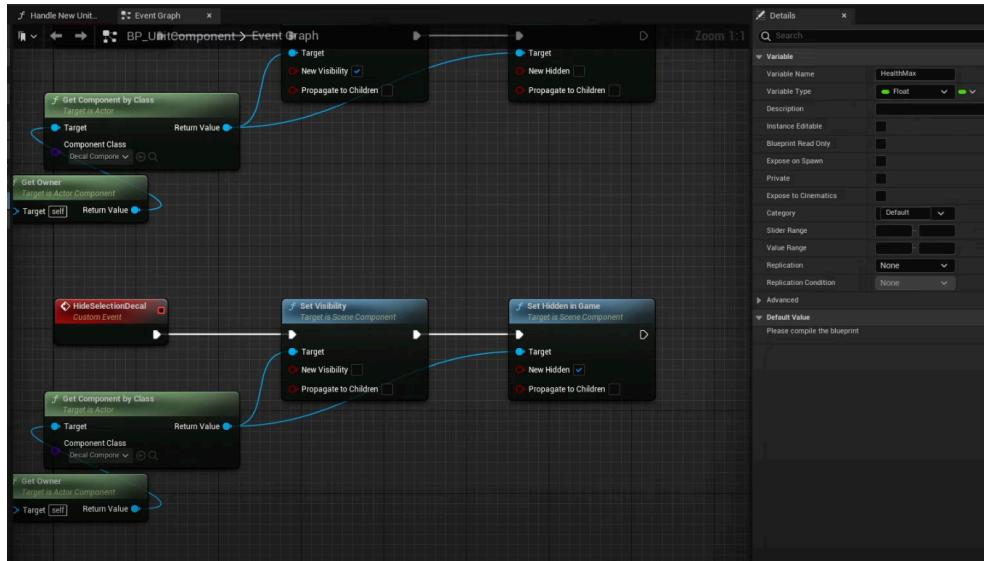


Рисунок 28. Функции для подключения панели здоровья к здоровью

В рамках работы был проведен анализ производительности с инстансингом и без. Для сравнения было измерено:

- количество кадров в секунду (FPS);
- количество отрисованных примитивов сцены (треугольников);

- количество вызовов отрисовки (draw calls).

9. Контроллер искусственного интеллекта

Контроллер ИИ в Unreal Engine — это важнейший компонент, который управляет поведением и принятием решений неигровых персонажей (NPC) или объектов искусственного интеллекта (ИИ) в игре. Он позволяет разработчикам создавать интеллектуальный и отзывчивый ИИ, который может перемещаться по игровому миру, взаимодействовать с объектами и другими персонажами и принимать решения на основе логики игры.

Perception: AI Controller incorporates Perception components such as Sight Sense, Hearing Sense, and Damage Sense to enable AI characters to perceive and react to the game environment. These components provide the AI with sensory input, allowing them to detect and respond to stimuli like player presence, sounds, or damage.

Navigation and Pathfinding: AI Controller includes built-in navigation and pathfinding systems that allow AI characters to navigate the game world efficiently. The NavMesh system provides a precomputed navigation mesh that AI characters can use to find paths and avoid obstacles. Developers can also fine-tune AI movement behaviors to ensure smooth and realistic navigation.

AI Communication: AI Controller facilitates communication between AI characters, enabling them to coordinate actions, share information, and work together as a group. This can be useful for implementing team-based AI behaviors, such as squad tactics or cooperative tasks.

Blueprint Integration: Unreal Engine allows developers to create AI Controllers using Blueprints, providing a visual scripting interface for AI behavior. This allows for rapid prototyping and easy iteration, as well as the ability to extend and customize AI behavior without writing code.

ЗАКЛЮЧЕНИЕ

В результате работы приобрел навыки работы с Unreal Engine 5 и приобрел опыт использования Blueprints. Овладел базовыми навыками 3D-моделирования, текстурирования. Овладел базовыми навыками анимации в Unreal Engine. В процессе работы выполнены следующие задачи:

1. реализована система выбора и отмены выбора зданий и персонажей;
2. добавлена возможность удалять карты с поля боя;
3. добавлена возможность выбирать карты из группы;
4. добавлена возможность масштабирования и перемещения камеры в зону игрока;
5. реализована анимация персонажей;
6. создана пейзажная сцена;
7. для персонажей созданы 3D модели;
8. реализован AI-контроллер.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Unreal Engine 5 Documentation // Unreal Engine Documentation URL: <https://docs.unrealengine.com/>. Дата обращения: 23.04.2024;
2. YouTube UNF GAMES // YouTube, UNF GAMES URL: <https://www.youtube.UNF GAMES>. Дата обращения: 23.04.2024;
3. Modeling – Blender Manual // Blender Manual URL: <https://docs.blender.org/manual/en/latest/modeling/index.html>. Дата обращения: 18.12.2023;
4. Мирмап // Wikipedia, the free encyclopedia URL: <https://en.wikipedia.org/wiki/Мирмап>. Дата обращения 05.04.2024;
5. Level of Detail (computer graphics) // Wikipedia, the free encyclopedia URL: [https://en.wikipedia.org/wiki/Level_of_detail_\(computer_graphics\)](https://en.wikipedia.org/wiki/Level_of_detail_(computer_graphics)). Дата обращения: 15.04.2024;