МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ КАЗАХСТАН
Satbayev University
Институт кибернетики и информационных технологий
Кафедра кибербезопасность, обработка и хранение информации

Баймолдина Анель Муратқызы

«Создание симуляции пространства обучения в условиях, приближенных к реальным ситуациям, с использованием VR технологий»

ДИПЛОМНАЯ РАБОТА

Специальность 5В070300 – Информационные системы

Алматы 2021

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ КАЗАХСТАН

Satbayev University

Институт кибернетики и информационных технологий

Кафедра кибербезопасность, обработка и хранение информации

ДОПУЩЕН К ЗАЩИТЕ

Заведующий кафедрой
КБОиХИ канд. техн. наук,
ассистент-профессор
_____Н. А. Сейлова
«28» 05 2021 г.

ДИПЛОМНАЯ РАБОТА

На тему: «Создание симуляции пространства обучения в условиях, приближенных к реальным ситуациям, с использованием VR технологий»

Специальность 5В070300 – Информационные системы

Выполнила: Баймолдина А.М.

Научный руководитель
магистр технических
наук, лектор
Аристомбаева М.Т.

«28» 05 2021 г.

Алматы 2021

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Satbayev University

Институт кибернетики и информационных технологий

Кафедра кибербезопасность, обработка и хранение информации

5В070300 – Информационные системы

УТВЕРЖДАЮ
Заведующий кафедрой
КБОиХИ канд. техн. наук,
ассистент-профессор
_____Н. А. Сейлова
«_28_»__05___20_21_г.

ЗАДАНИЕ
на выполнение дипломной работы

Обучающейся: Баймолдина Анель Муратқызы
Тема: «Создание симуляции пространства обучения в условиях, приближенных к реальным ситуациям, с использованием VR технологий»
Утверждена *приказом Ректора Университета № 2131-б от «24» 11 2020 г.*

Срок сдачи законченной работы «__» *мая 2021 г.*

Исходные данные к дипломной работе: результаты преддипломной практики, обзор предметной области, сбор теоретического материала.

Краткое содержание дипломной работы:

а) Анализ решений технологии виртуальной реальности;
б) Выбор средств проектирования и разработки VR-приложения;
в) Проектирование и разработка VR-приложения.
Рекомендуемая основная литература:___наименований

ГРАФИК

подготовки дипломной работы (проекта)

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечание
Исследование сферы технологий виртуальной реальности	12.02.2021	
Проектирование и разработка приложения	18.03.2021	
Создание скриптов и звукового сопровождения	20.04.2021	

Подписи

консультантов и нормоконтролера на законченную дипломную работу (проект) с указанием относящихся к ним разделов работы (проекта)

Наименование разделов	Консультанты Ф.И.О. (уч. степень, звание)	Дата подписания	Подпись
«Создание симуляции пространства обучения в условиях, приближенных к реальным ситуациям, с использованием VR технологий»			
Нормоконтролер			
Программная часть			

Научный руководитель:_____Аристомбаева М.Т.

Задание принял к исполнению обучающаяся_____Баймолдина А.М.

Дата "__"____2021 г.

АҢДАТПА

Бұл дипломдық жұмыстың мақсаты - оқу орындарында білім беруді жетілдіруге мүмкіндік беретін VR-оқыту қосымшасын жасау. Пәндік аймақты зерттеу және қолданыстағы шешімдерге талдау жасалды. Бағдарламаны әзірлеу Unity, Oculus (атап айтқанда OculusQuest2) және C# бағдарламалау тілі сияқты құралдарды қолданды.

АННОТАЦИЯ

Целью данной дипломной работы является создание обучающего VR-приложения, которое позволит усовершенствовать подачу знаний в учебных заведениях. Было проведено исследование предметной области и анализ существующих решений. Разработка приложения задействовала такие инструменты, как Unity, Oculus (в частности OculusQuest 2) и язык программирования C#.

SUMMARY

The purpose of this diploma work is to create a VR application, which could improve the process of giving knowledge in educational institutions. Subject area was surveyed and existing solutions were analyzed. Application development involved such instruments as Unity, Oculus (particularly OculusQuest 2) and programming language C#.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ……9

1 ИССЛЕДОВАНИЕ ПОНЯТИЯ И СФЕРЫ ПРИМЕНЕНИЯ ТЕХНОЛОГИЙ ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ……10

1.1 Преимущества и недостатки……..10

1.2 Анализ существующих решений……...11

2 РАЗРАБОТКА ПРОЕКТА……12

2.1 Постановка задачи……...12

2.2 Выбор инструментальных средств……12

2.2.1 Unity……...12

2.2.2 Язык программирования C#.....16

2.2.3 Oculus Quest 2……... 19

2.3 Средства проектирования информационной системы……21

3 РЕАЛИЗАЦИЯ ПРОЕКТА……22

3.1 Архитектура приложения……...22

3.1.1 Скрипты для «охраны безопасности жизнедеятельности (ОБЖ)»……..22

3.1.2 Скрипты для «виртуальной обсерватории»……...32

3.2 Звуковое сопровождение в приложении……...38

ЗАКЛЮЧЕНИЕ……..40

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ……..42

ПРИЛОЖЕНИЕ А……44

ПРИЛОЖЕНИЕ Б……..45

ВВЕДЕНИЕ.

Актуальность дипломного проекта «Создание симуляции пространства обучения в условиях, приближенных к реальным ситуациям, с использованием VR технологий» заключается в том, что виртуальная реальность (VR) на данный момент является активно развивающейся технологией, которая позволяет визуализировать воссозданный из виртуальных объектов мир и взаимодействовать с его окружением.

Обучающий материал со временем перетерпел значительные изменения и на данный момент воспроизводится как видео, которое наглядно показывает порядок действий или инструкций для той или иной задачи. Но что, если использовать в качестве обучения VR? Воспользовавшись его преимуществами, можно будет создать прекрасный материал для обучения в разных сферах.

Предметом исследования являются возможности виртуальной реальности и их применение в целях обучения. Таким образом затраты на воссоздания симуляции некоторых обучающих реквизитов могут значительно снизиться.

Целью данного проекта является создание функционального VR-продукта, который будет применяться в образовании для получения различных знаний.

Проанализировав различные варианты сценариев, мы остановились на двух – изучение космоса и основы безопасности жизнедеятельности, тем самым продемонстрировав два темпа развития событий – неторопливый, который заключён в освоении материала с помощью окружающих объектов и быстрый, который завязан на вариативности действий в ограниченный промежуток времени.

Во время создания проекта были проанализированы похожие VR-приложения:

- *«Virtual Speech»* – виртуальные курсы для улучшения социальных навыков на рабочем месте.
- *«Osso VR»* – симулятор операций для практики докторов и хирургов.
- *«VSTS»* – виртуальная военная подготовка.

Инструменты, которые были использованы в ходе реализации:

- «Unity» – межплатформенная среда разработки компьютерных игр, позволяющая создавать приложения, работающие на различных платформах, включающих персональные компьютеры, игровые консоли, мобильные устройства, интернет-приложения и другие;

- «OculusQuest» – шлем и контроллеры, среда работы которых была интегрирована в Unity;
- «3DS Max» - компьютерная графическая программа, которая направлена на создание 3D моделей, объектов, игр и изображений.

Тема дипломного проекта «Создание симуляции пространства обучения в условиях, приближенных к реальным ситуациям, с использованием VR технологий» была рассмотрена, также было проведено ознакомление с нюансами проекта, его предметной областью и выявлена его актуальность в современных реалиях.

1 ИССЛЕДОВАНИЕ ПОНЯТИЯ И СФЕРЫ ПРИМЕНЕНИЯ ТЕХНОЛОГИЙ ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ

1.1 Преимущества и недостатки

Виртуальная реальность предоставляет огромное количество преимуществ, среди которых:

- Улучшенная реальность: визуальная составляющая виртуальной реальности в разы лучше, чем в реальном мире. Данная технология используется при создании игр и позволяет пользователям ощутить себя в другом мире. Задействовав разные органы чувств и передавая вибрации через контроллеры, погружение в реальность VR-игры происходит на более глубоком уровне, давая отличный и насыщенный игровой опыт;

- Использование в разных сферах: широкий набор особенностей виртуальной реальности пользуется особой популярностью в военных, обучающих и здравоохранительных разработках. Также, это активно используется в авиации и архитектуре для наглядного просмотра готового продукта;

- Необыкновенный опыт для пользователя: после испытанного удовольствия от игры, в которой звуковое сопровождение, тактильное восприятие и способность взаимодействовать с окружением создаёт особенную реалистичность, пользователям хочется испытать это снова и снова. Особенно, такое восприятие можно объединить с созданием фильмов, где каждая сцена будет показана с разных углов, что воспроизводит ощущение интерактивности, как в играх;

- Детализированность: виртуальный мир даёт полное и детализированное видение окружения. Например, VR придаёт туристическим местам больше интереса и простоты, визуализируя больше деталей, что сподвигнет людей посетить их как можно скорее в реальной жизни;

- Связь с людьми: предоставляет возможность общаться с незнакомыми людьми, живущими на другом конце света, без боязни быть раскрытым. Примером этого служит VRChat, который является аналогом социальной сети, где пользователи могут примерить на себя различного рода модели их любимых персонажей из мультфильмов, сериалов и фильмов. [1]

Однако при этом существуют некоторые недостатки:

- Высокая стоимость: не каждый может позволить себе данную технологию;
- Нехватка живого общения: виртуальное общение не должно заменять реальное, к тому при такой коммуникации может выработаться чувствительность к обману;
- Ощущение никчёмности: временами у пользователей может возникнуть ощущение бесполезности в реальной жизни, поэтому они будут пытаться вернуться в виртуальную обстановку, чтобы сбежать от насущных проблем, что пошатнёт их ментальное здоровье;
- Зависимость от виртуального мира: это может вызвать серьёзные проблемы со здоровьем, начиная от ухудшения зрения до дезориентации и приступов; [2]
- Экспериментальный характер: пока ещё не существует полностью развитых VR-проектов, из-за серьёзных трудностей, которые возникают перед разработчиками.

1.2 Анализ существующих решений

После анализа различных обучающих приложений, были выявлены наиболее близкие по общей тематике варианты.

«Virtual Speech».

Данная технология включает в себя различные обучающие материалы. Созданная в формате онлайн курсов, она позволяет на наглядно смоделированных ситуациях обучить работников необходимым навыкам, которые важны для эффективной работы. Кроме того, пользователи могут присылать свои презентации и записи их речи, которые тут же будут рассмотрены и оценены по нескольким критериям.

«Osso VR».

Во время операций могут произойти непредвиденные ситуации, которые могут поставить под угрозу жизнь пациента. Чтобы снизить риск летального

исхода на хирургическом столе, данный симулятор позволяет проводить хирургам виртуальные операции, не боясь ошибок и учась на них.

«VSTS».

Военная подготовка крайне важна для каждого солдата. Однако с этой виртуальной обучающей программой, она станет ещё эффективнее, благодаря симуляции действий реального поля боя. Неожиданные развития событий и сложные условия могут улучшить реакцию бойцов и ускорить их процесс адаптации во время военных столкновений. [3]

2 ВЫБОР СРЕДСТВ ПРОЕКТИРОВАНИЯ И РАЗРАБОТКА ПРИЛОЖЕНИЯ

2.1 Постановка задачи

Главные задачи:

- Изучить особенности инструментальных средств и выбрать наиболее подходящие для цели варианты;
- Перенести выбранные сценарии реалий в виртуальную среду;
- На основе воссозданного дизайна выстроить работающую систему интерфейса и интерактивного взаимодействия с окружением;
- Внедрить звук для более полного игрового опыта.

2.2 Выбор инструментальных средств

Основными инструментами для разработки VR-приложения были Unity и OculusQuest 2. Выбор данных инструментов обоснован их наибольшей актуальностью на сегодняшний день и широким спектром возможностей, направленных на создание приложений в виртуальной среде.

2.2.1 Unity

Unity – универсальный движок, оснащённый встроенным редактором, который позволяет создавать игры для компьютеров, смартфонов, консолей и планшетов с любой операционной системой.

Главными преимуществами данного движка являются:

- Эффективен в рендеринге 2D и 3D моделей и изображений. Их итоговое качество сравнительно лучше, чем в других движках и приложениях;
- Бесплатность, которая не ограничивает список доступных функций;
- Кроссплатформенность позволяет создавать приложения и игры на нескольких платформах, таких как консоли, персональные компьютеры, браузеры и мобильные устройства; [4]

- Магазин ассетов предоставляет широкий выбор различных компонентов (звуки, физика, рендеринг, управление) для разработки приложения, среди которых можно найти не только хорошие платные ассеты, но и бесплатные;

- Простота и гибкость в использовании;

- Интегрированный редактор, поддерживающий такие языки программирования, как JavaScript и C#;

- Укладчик объёмного звука, который не представляет трудностей в применении;

- Интерактивные обучающие демо-игры, которые позволяют ознакомиться с как базовыми, так и дополнительными функциями Unity.

Однако вместе с этим существуют некоторые недостатки:

- Невозможность просмотреть исходный код для исправления багов;

- Занимает много памяти, что создаёт некоторые неполадки в системе;

- Для разработки масштабного проекта требуется оптимизация, доступная при приобретении полной (платной) версии движка;

- Создание текстур и ландшафтов необоснованно тратит больше усилий и времени. [5]

Интерфейс Unity.

Меню движка состоит из пяти окон:

- «Сцена» позволяет создавать визуальную составляющую игры, предоставляя возможность смотреть на объекты и напрямую взаимодействовать с ними (рисунок 2.1);

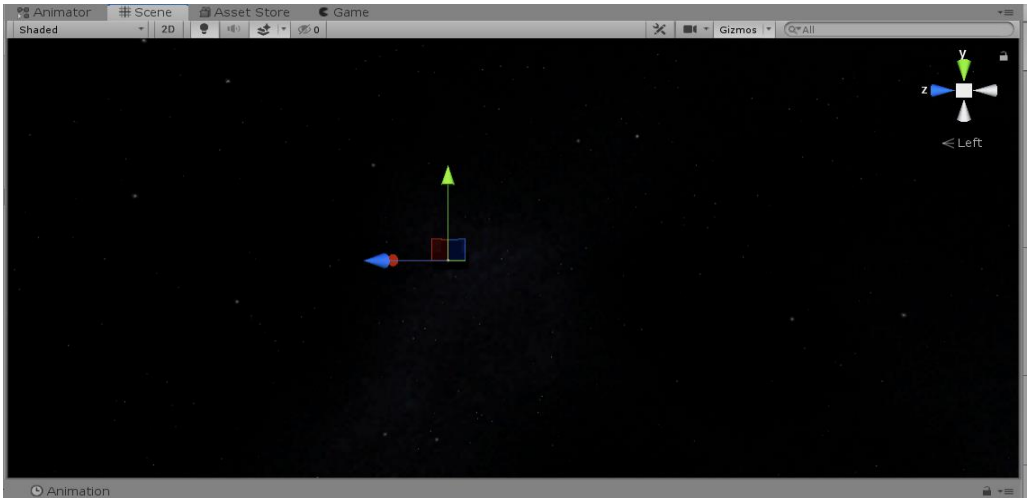


Рисунок 2.1 – Окно «сцены»

- «Проект» отображает все доступные ассеты (файлы, которые сохраняются на жёсткий диск), которые можно задействовать при разработке, ими могут быть скрипты, текстуры, 3D модели, аудио и сцены (рисунок 2.2);

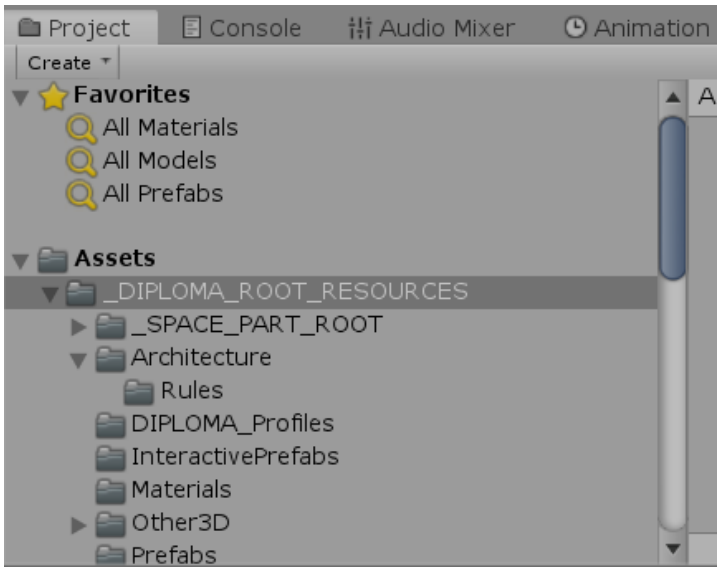


Рисунок 2.2 – Окно «проекта»

- «Иерархия» показывает список всех игровых объектов, задействованных в «сцене», их можно перемещать в разном порядке на своё усмотрение и группировать для создания «семей» (объект, находящийся во главе каждой такой группы в «иерархии» называется «родителем», а сгруппированные в нём элементы – «детьми») (рисунок 2.3);

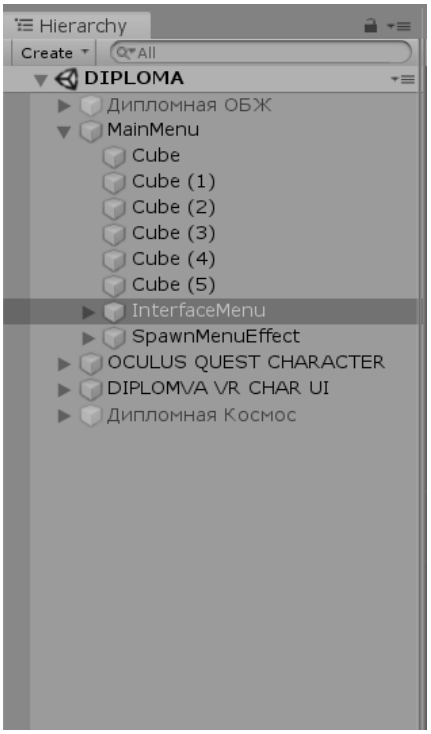


Рисунок 2.3 – Окно «иерархии»

- «Инспектор» отображает свойства игровых объектов и ассетов, которые были выбраны, его можно вызвать из «проекта» и «иерархии» (рисунок 2.4);

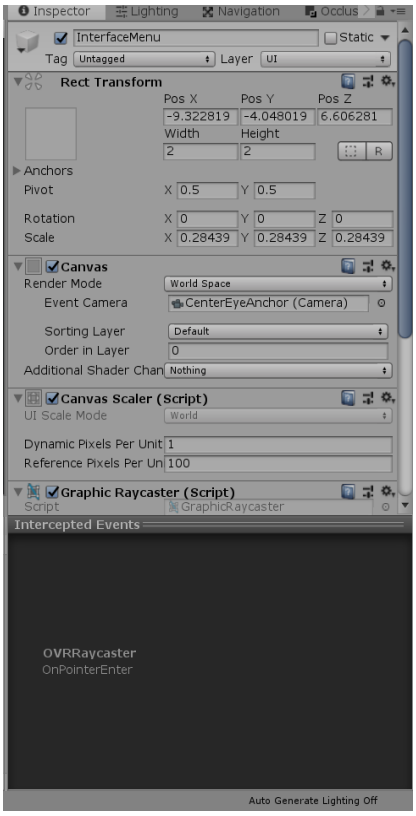


Рисунок 2.4 – Окно «инспектора»

- «Игра» позволяет посмотреть превью игры в редакторе Unity и протестировать его с помощью панели инструментов сверху, чьи кнопки отвечают за запуск теста игры, её приостановки и прокрутки по каждому фрейму (рисунок 2.5).

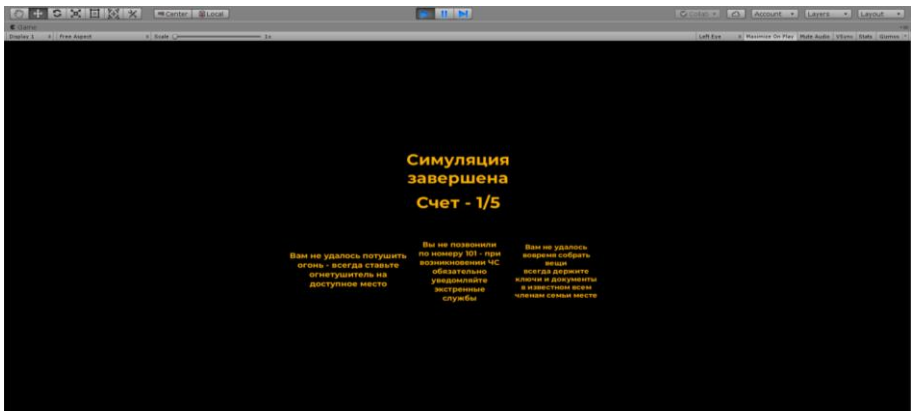


Рисунок 2.5 – Окно «игры»

Также, важно отметить, что в панели инструментов есть и другие функции, например, кнопки редактирования объектов, управления аккаунтами Unity, просмотра и отключения слоёв текстур на «сцене» и оформления окон в редакторе движка. [6]

2.2.2 Язык программирования C#

C# является одним из самых распространённых языков программирования в среде разработки игр и приложений. Поддержка на движках и разных платформах, объектная ориентированность, совместимость с более старыми версиями программ, управление памятью и умеренная скорость работы делают его отличным инструментом для создания скриптов. [7]

Основные составляющие C#:

- *Переменные* – хранят в себе значения и ссылки на объекты;
- *Функции* – коллекции в коде программы, которые сравнивают эти переменные и манипулируют ими;
- *Классы* – способ структурирования кода, который объединяет переменные и функции и присваивает их объекту.

Переменные вызываются методами «public» и «private», разница между ними в том, что переменная, вызванная первым методом, отображается на «сцене» у объекта и доступна другим скриптам и классам, а вторым методом – нет. Однако, «private» полезен тем, что делает код чище и понятнее, поскольку значения изменяются только внутри класса и нигде больше.

Тип переменной определяет вид значения (число, текст и сложные типы, такие как компоненты), который присваивается переменной.

Синтаксис переменной (Метод – тип – переменная) (рисунок 2.6):

```
5 public class DemoScript: MonoBehaviour {
6
7     public Light myLight;
8     private Light myOtherLight;
9
10 }
```

Рисунок 2.6 – Синтаксис переменной

Имя переменных всегда указывается со строчной буквы, но если это словосочетание, то второе или третье слово в имени пишется с прописной, например, myPosition.

После компиляции скрипта, переменные становятся видимыми в редакторе, в окне «инспектора» (рисунок 2.7).

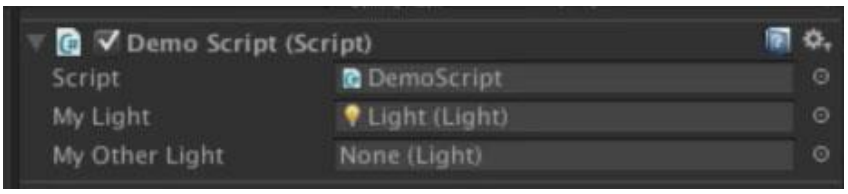


Рисунок 2.7 – Переменные в «инспекторе»

В нашем проекте это выглядит как на рисунке 2.8.

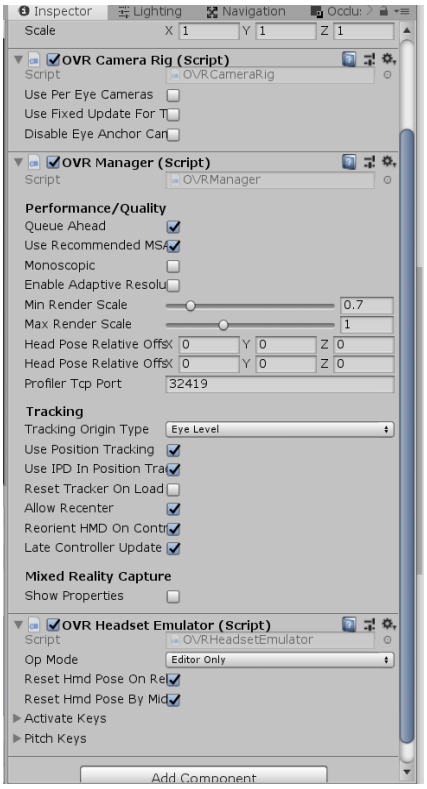


Рисунок 2.8 – Переменные в проекте

Функции созданы для манипуляции переменными (рисунок 2.9). Внутри Unity существуют несколько функций, которые запускаются автоматически.

```
/*
    Awake()
    Start()
    Update ()
    FixedUpdate()
    LateUpdate

*/
```

Рисунок 2.9 – Функции в Unity

Awake – вызывается единожды для того, чтобы создать экземпляр компонента. Пока игровой объект неактивен, призвать его не получится.

Однако, если он активен, но не включен, функция всё равно вызывается. В общих чертах, данная функция инициализирует все переменные, которым нужно приписать значение.

Start – призывается всего один раз и сразу после Awake, но для его работы нужно, чтобы игровой объект был включен.

Update – вызов приходится на каждый фрейм, причём логика кода работает непрерывно в виде анимаций или искусственного интеллекта.

FixedUpdate – позволяет подключить физику объектов.

LateUpdate – вызывается в конце фрейма (после Update), чтобы закрепить компоненты между собой, к примеру, камеру к объекту, чтобы смена позиций отображалась синхронно.

Синтаксис функции (тип возвращаемого значения – название функции со скобками – фигурные скобки) (рисунок 2.10):

```
13
14      void MyFunction () {
15
16      }
17 }
```

Рисунок 2.10 – Синтаксис функции

Название функции начинается с прописной буквы.

Классы являются коллекциями функций и переменных (рисунок 2.11).

```
6 public class DemoScript: MonoBehaviour {
7
8     public Light myLight;
9
10    void Awake () {
11        int myVar = AddTwo(9,2);
12        Debug.Log(myVar);
13    }
14
15    void Update () {
16        if (Input.GetKeyDown ("space")) {
17            MyFunction ();
18
19        }
20
21    }
22
23    void MyFunction () {
24
25        myLight.enabled = !myLight.enabled;
26    }
27
28    int AddTwo (int var1, int var2) {
29        int returnValue = var1 + var2;
30        return returnValue;
31
32    }
33
34 }
```

Рисунок 2.11 – Скрипт класса

Также дана возможность создать собственные классы, но для этого нужно сериализировать их в формат JSON, то есть превратить в поток байтов,

который можно будет десериализировать, чтобы заполнить игровой объект данными (рисунок 2.12). [8]

```
using UnityEngine;
using System.Collections;

[System.Serializable]
public class DataClass {
    public int myInt;
    public float myFloat;

}

public class DemoScript : MonoBehaviour {

    public Light myLight;
    public DataClass myClass;

    void Awake () {
        int myVar = AddTwo(9,2);
        Debug.Log(myVar);

    }

    void Update () {
        if (Input.GetKeyDown ("space")) {
            MyFunction ();

        }

    }

    void MyFunction () {
        myLight.enabled = !myLight.enabled;
    }
```

Рисунок 2.12 – Создание собственных классов

2.2.3 Oculus Quest 2

OculusQuest является гарнитурой виртуальной реальности. Состоит из шлема и контроллеров, которые надеваются на голову и руки и подключаются к персональному компьютеру, смартфонам и консоли с помощью кабеля. Обработка отображаемой картинки производится через видеокарту основного устройства и проецируется через шлем. Для освоения базовых функций, система данной гарнитуры знакомит пользователя с предназначением каждой кнопки и основными условиями безопасного использования OculusQuest.

Чтобы Oculus работал на Unity, необходимо скачать плагин, который позволит интегрировать его на движок. Для этого нужно зайти в Asset Store Unity и найти Oculus Integration Package. После скачивания важно импортировать данный плагин в открытый редактор движка. В результате, в окне «проекта» появится папка, хранящая в себе все необходимые компоненты для поддержки виртуальной реальности.

Будучи ассетом, интеграционный плагин Oculus, содержит в себе:

- Аудиоклипы, которые можно использовать в разработке игры;

- Редактор, который придаёт функциональность движку и расширяет возможности скриптов;

- Материалы, которые включают в себя элементы графического пользовательского интерфейса;

- Меши, чьё наличие требуется для определения границ действия VR;

- Плагины, которые дают доступ не только к ключевым действиям Oculus, но и к поддержке различных платформ;

- Префабы, которые добавляют компоненты виртуальной реальности в «сцену» (OVRCameraRig, OVRHandPrefab, OVRPlayerController);

- Ресурсы, в которых содержатся шейдеры, используемые в реальном времени;

- Сцены, где хранятся шаблоны сцен для демонстрации типичных концептов, присущих созданию VR-игры;

- Скрипты C#, которые связывают вместе фреймворк виртуальной реальности и компоненты Unity;

- Шейдеры, которые воспроизводят различного рода вычисления и работают через небольшие скрипты;

- Текстуры (ассеты изображений), которые требуются для некоторых скриптов.

Остановимся подробнее на префабах, скриптах и сценах.

Префабы - игровые объекты, которыми можно пользоваться неоднократное количество раз. Ими являются 3D фигуры, освещение, аудио и камера. Чтобы ими воспользоваться, достаточно переместить их в окно «сцены» движка.

В самой папке префабов содержатся такие файлы:

- OVRCameraRig – пользовательская камера виртуальной реальности, которая оптимизирует рендеринг для стереоскопического дисплея шлема OculusQuest, а также предоставляет доступ к OVRManager, то есть интерфейсу к гарнитуре VR;

- OVRHandPrefab – префаб, который даёт возможность сделать руки основным элементом управления игры;

- OVRCubemapCaptureProbe: позволяет в любой момент сделать 360-градусный скриншот рабочего приложения с перспективы камеры «сцены» и назначить эту функцию на определённую кнопку для активации;

- OVRPlayerController – возможность для игрока двигаться по виртуальному пространству, которая напрямую связана с OVRCameraRig, прикреплённой к виртуальному персонажу.

Скрипты же в основном открывают доступ к системе границ виртуального окружения (OVRBoundary.cs), тактильному отклику на специальном отслеживающем движения контроллере (OVRHaptics.cs),

унифицированной системе ввода для контроллеров и геймпадов Oculus (OVRInput.cs), скриптам, позволяющим корректно отображать объекты VR в линзах шлема. Помимо этого, существуют несколько скриптов для захвата и бросания игровых объектов (OVRGrabber.cs и OVRGrabbable.cs), переключения на моноскопический рендеринг (OVRMonoscopic.cs), сбрасывания на исходную позицию, чтобы изменить её показатели (OVRResetOrientation.cs) и смены режима потребления памяти при низком заряде устройства, с которого было запущено приложения (OVRModeParms.cs).

Сцены доступны в количестве трёх вариантов: первая сцена - «тривиальная», в которой есть один куб и простая камера Unity; вторая сцена – «кубы», состоящей из 3D массива кубов и камеры виртуальной реальности; третья сцена – «комната», в форме куба, заполненная кубами, которые управляются OVRPlayerController через OVRGrabber.cs и OVRGrabbable.cs. [9]

2.3 Средства проектирования информационной системы

Основой информационной системы данного проекта будет являться связь между преподавателем и обучающимся, реализованная через прикладное и системное программное обеспечение. Теоретически, прикладным программным обеспечением в данном случае может выступать онлайн-портал учебного заведения, в котором работает система предоставления материалов по различным предметам, а также контроля посещаемости и успеваемости. Системное программное обеспечение будет базисом данного портала, поскольку в нём будут храниться данные о преподавателях, студентам, имеющихся предметах и материалах.

VR-приложение может быть внедрено в такую непрерывную и рабочую информационную систему разными путями. Например, скачиваемый файл в отдельном окне портала в браузере. Другой пример - дополнительный способ закрепления знаний на основе пройденных тем в виде самостоятельной работы.

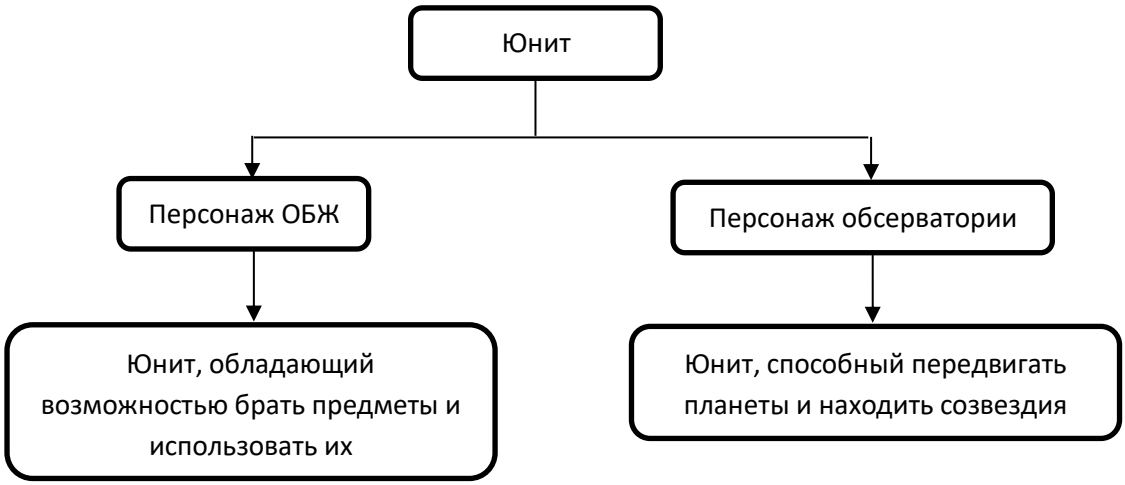
Аппаратным обеспечением, естественно, будут гарнитура виртуальной реальности вместе с имеющимся персональным компьютером. Для начала ознакомления с VR-проектом важно будет провести подробный инструктаж об использовании оборудования виртуальной реальности, предварительно ознакомить преподавателя с особенностями управления контроллеров и надевания шлема, чтобы он позже мог провести инструктаж обучающемуся,

которому предстоит выполнить задание в виртуальной обстановке. После окончания работы с симуляцией, обучающийся может отключить её и снять гарнитуру, чтобы затем получить оценку преподавателя.

Важно подметить, что вышеописанное является теорией, которую предстоит реализовать уже непосредственно при продвижении дипломного проекта как самостоятельного продукта в рамках стартапа.

3 РЕАЛИЗАЦИЯ ПРОЕКТА

3.1 Архитектура приложения



Архитектура приложения – это использование различных программных средств для создания приложения. Однако в данном дипломном проекте данная архитектура принимает несколько иное значение. Помимо использования Unity и интеграции в него Oculus, были написаны скрипты на C#, которые позволили оживить ранее созданный дизайн виртуального мира. То есть, архитектура этого приложения – задействование скриптов для создания интерактивности приложения.

В данном дипломном проекте был создан дизайн окружения дома для ОБЖ и космоса для виртуальной обсерватории, к которым позже были расписаны скрипты для интерфейса, взаимодействия с предметами и получения необходимой информации через текстовые подсказки и аудио-нарратив.

3.1.1 Скрипты для «охраны безопасности жизнедеятельности (ОБЖ)»

ОБЖ представляет собой ситуацию возникновения пожара, при которой, зная несколько правил техники безопасности, необходимо безопасно выбраться из «собственной» квартиры. Здесь предусмотрено поиск и использование предметов, тушение огня, освещение тёмного помещения, звонка в пожарную службу и выхода из квартиры. Установленный таймер позволяет осознавать важность времени в сложившихся условиях, в то время как счёт позволяет следить за прогрессом и в конце симуляции показывать действия, которые не были сделаны.

DestroyOnTime.cs

Данный код отвечает за уничтожение предметов через пять секунд после воздействия на него огня. Например, если персонаж не выберется из квартиры за отведённое время, всё вокруг него разрушится, и игра закончится.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class DestroyOnTime : MonoBehaviour
{
    void Start()
    {
        Destroy(this.gameObject, 5f);
    }
}
```

DiplomaSimController.cs

Это скрипт персонажа, чьи действия закреплены за контроллером, иными словами, все действия, которые будут производиться игроком, будут воспроизводиться в контексте данного кода. Также здесь хранятся ошибки, переходы между окном игрового пространства и завершающего экрана статистики.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class DiplomaSimController : MonoBehaviour
{
    public bool HaveKeys;
```

```
public bool Called;
public bool FireDestroyed;

[SerializeField] private GameObject MistakeKeys;
[SerializeField] private GameObject MistakeCalled;
[SerializeField] private GameObject MistakeFire;

[SerializeField] private GameObject endsimWindow;
[SerializeField] private GameObject player;
[SerializeField] private GameObject level;

public void TaskFinished(int number)
{
    if (number == 1)
    {
        FireDestroyed = true;
        MistakeFire.SetActive(false);
    }
    if (number == 2)
    {
        HaveKeys = true;
        MistakeKeys.SetActive(false);
    }
    if (number == 3)
    {
        Called = true;
        MistakeCalled.SetActive(false);
    }
}
public void TurnOnObject(GameObject obj)
{
    obj.SetActive(true);
}
public void TurnOffObject(GameObject obj)
{
    obj.SetActive(false);
}
public void TeleportPlayer(Transform pos)
{
    player.transform.position = pos.position;
    player.transform.rotation = pos.rotation;
}
public void EndSimulation()
{
```

```
        level.SetActive(false);
        endsimWindow.SetActive(true);
    }
}
```

FireLogic.cs

Данный код представляет логику огня, которая работает по типу увеличения единиц анимационных элементов пожара. Их уменьшение происходит только в случае тушения огнетушителем.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class FireLogic : MonoBehaviour
{
    [SerializeField] private ObjectsCounter root;
    [SerializeField] private float health = 100f;
    private void Start()
    {
        root.objectsAmount++;
    }
    private void OnTriggerStay(Collider other)
    {
        if (other.tag == "Flamestop")
        {
            health -= 0.5f;
            if (health <= 0)
            {
                root.objectsAmount--;
                root.CheckAmount();
                Destroy(this.gameObject);
            }
        }
    }
}
```

FlamestopperVR.cs

Скрипт огнетушителя, который позволяет брать этот предмет, запускать анимацию тушения огня и бросать.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class FlamestopperVR : MonoBehaviour
{
    [SerializeField] private GameObject Flamestopper;
    private OVRGrabbable ovrGrabable;
    public OVRInput.Button shootingButton;

    private void Start()
    {
        ovrGrabable = GetComponent<OVRGrabbable>();
    }
    private void Update()
    {
        if (ovrGrabable.isGrabbed && OVRInput.GetDown(shootingButton,
ovrGrabable.grabbedBy.GetController()))
        {
            Switch();
        }
    }
    private void Switch()
    {
        if (Flamestopper.activeSelf)
        {
            Flamestopper.SetActive(false);
        }
        else
        {
            Flamestopper.SetActive(true);
        }
    }
}
```

FlashlightVR.cs

Скрипт взаимодействия с фонариком.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
```

```
public class FlashlightVR : MonoBehaviour
{
    private Flashlight_PRO flashlight;
    private OVRGrabbable ovrGrabable;
    public OVRInput.Button shootingButton;

    void Start()
    {
        flashlight = GetComponent<Flashlight_PRO>();
        ovrGrabable = GetComponent<OVRGrabbable>();
    }

    // Update is called once per frame
    void Update()
    {
        if (ovrGrabable.isGrabbed && OVRInput.GetDown(shootingButton,
ovrGrabable.grabbedBy.GetController()))
        {
            flashlight.Switch();
        }
    }
}
```

InterfaceController.cs

Данный скрипт выводит информацию о таймере и счёте, которые отображаются игроку на экране.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class InterfaceController : MonoBehaviour
{
    [SerializeField] private Text scoreEndsimText;
    [SerializeField] private Text scoreText;
    public int score;
    [SerializeField] private int maxScore;
    TimerLogic timer;
    private void Start()
    {
        timer = GetComponent<TimerLogic>();
        UpdateScore();
    }
```

```
    public void UpdateScore()
    {
        scoreText.text = "Счет - " + score + "/" + maxScore;
        scoreEndsimText.text = "Счет - " + score + "/" + maxScore;
    }

}
```

TimerLogic.cs

Логика таймера, которая работает в формате минут и секунд, которые стремятся к нулю.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class TimerLogic : MonoBehaviour
{
    public GameObject timer;
    public Text TimerText;

    public float seconds;
    public float minutes;

    public GameObject additionalEventObject;

    private void Start()
    {
        StartTimeCounter();
    }

    public void StartTimeCounter()
    {
        StartCoroutine(time());
    }
    IEnumerator time()
    {
        yield return new WaitForSeconds(1);
        seconds--;
        if (seconds <= 0)
        {
            if (minutes > 0)
```

```
            {
                minutes--;
                seconds = 60;
            }
        }
        if (minutes < 3)
        {
            additionalEventObject.SetActive(true);
        }
        TimerText.text = minutes + "m : " + seconds + "s";
        if (minutes <= 0 && seconds <= 0)
        {
            timer.SetActive(false);
            this.enabled = false;
        }
        StartCoroutine(time());
    }
}
```

CheckIfGrabbed.cs

Проверка взятия предмета (например, работает с предметом «ключ»), после чего идёт запуск аудиофайла и завершение одного из заданий, которое засчитывается в счёте.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Oculus.Avatar;

public class CheckIfGrabbed : MonoBehaviour
{
    [SerializeField] private bool DestroyAfterGrab;
    [SerializeField] private InterfaceController root;
    [SerializeField] private DiplomaSimController main;
    [SerializeField] private int taskNumber;
    [SerializeField] private AudioSource taskSound;
    public OVRGrabbable neededObject;
    private void Update()
    {
        if (neededObject.isGrabbed)
        {
            root.score++;
```

```
            root.UpdateScore();
            if (DestroyAfterGrab)
            {
                Destroy(neededObject.gameObject);
                neededObject.GrabEnd(new Vector3(0, 0, 0), new Vector3(0, 0, 0));
            }
            taskSound.Play();
            main.TaskFinished(taskNumber);
            this.enabled = false;
        }
    }
}
```

NoObjectsInArea.cs

Если тэг объекта не обнаружен, то его логика не запускается.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class NoObjectsInArea : MonoBehaviour
{
    [SerializeField] private string findTag;
    [SerializeField] private bool stillHere;
    [SerializeField] private InterfaceController root;
    private void OnTriggerStay(Collider other)
    {
        if (other.tag == findTag)
        {
            stillHere = true;
        }
    }
    private void OnTriggerExit(Collider other)
    {
        if (other.tag == findTag)
        {
            stillHere = false;
        }
    }
}
```

ObjectIsInArea.cs

Если тэг объекта найден, то его логика запускается вместе с другими приписанными значениями. Скрипт ориентирован на игрока, особенно в момент приближения к двери (завершение игры).

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ObjectIsInArea : MonoBehaviour
{
    [SerializeField] private InterfaceController root;
    [SerializeField] private DiplomaSimController controller;
    [SerializeField] private bool lastTask;
    [SerializeField] private AudioSource additionalSound;
    private void OnTriggerEnter(Collider other)
    {
        if (other.tag == "Player")
        {
            root.score++;
            root.UpdateScore();
            if (lastTask)
            {
                controller.EndSimulation();
                additionalSound.Play();
            }
            Destroy(this.gameObject);
        }
    }
}
```

ObjectsCounter.cs

Считает количество объектов и добавляет/уменьшает счёт игрока.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ObjectsCounter : MonoBehaviour
{
    [SerializeField] private DiplomaSimController main;
    [SerializeField] private InterfaceController root;
    public int objectsAmount;
```

```
    [SerializeField] private int taskNumber;
    public void CheckAmount()
    {
        if (objectsAmount <= 0)
        {
            root.score++;
            root.UpdateScore();
            main.TaskFinished(taskNumber);
        }
    }
}
```

3.1.2 Скрипты для «виртуальной обсерватории»

Скрипты данного сценария рассматривают игровой опыт, заключённый в переносе планет Солнечной системы на правильные позиции, ознакомлению с фактами о них и нахождению созвездий в космическом пространстве.

Planet.cs

При соприкосновении контроллера к планете, над моделью планеты появляется её название, в противном случае, оно исчезает.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Oculus.Avatar;

public class Planet : MonoBehaviour
{
    public string PlanetName;
    [SerializeField] private SpaceInterfaceController root;

    private void OnTriggerEnter(Collider other)
    {
        if (other.tag == "Explorer")
        {
            root.CurrentPlanetText.text = PlanetName;
        }
    }
    private void OnTriggerExit(Collider other)
    {
        if (other.tag == "Explorer")
        {
```

```
            root.CurrentPlanetText.text = "";
        }
    }
}
```

Planetline.cs

Скрипт, который отвечает за расположение массива планет в одной линии.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Planetline : MonoBehaviour
{
    private LineRenderer planetside;
    [SerializeField] private List<Transform> planets;
    private void Start()
    {
        planetside = GetComponent<LineRenderer>();
    }
    void FixedUpdate()
    {
        planetside.SetPosition(0, planets[0].position);
        planetside.SetPosition(1, planets[1].position);
        planetside.SetPosition(2, planets[2].position);
        planetside.SetPosition(3, planets[3].position);
        planetside.SetPosition(4, planets[4].position);
        planetside.SetPosition(5, planets[5].position);
        planetside.SetPosition(6, planets[6].position);
        planetside.SetPosition(7, planets[7].position);
    }
}
```

PlanetsPlacer.cs

Расположение планет по порядку, то есть при приближении планеты к соответствующей полупрозрачной зоне, она встаёт на место и отрывается от контроллера игрока, отображая при этом информацию о счёте.

```
using System.Collections;
using System.Collections.Generic;
```

```
using UnityEngine;
using Oculus.Avatar;

public class PlanetsPlacer : MonoBehaviour
{
   [SerializeField] private SpaceInterfaceController root;
   [SerializeField] private bool passed;
   [SerializeField] private string neededName;
   [SerializeField] private AudioClip tellerClip;
   private void OnTriggerEnter(Collider other)
   {
      if (other.tag == "Planet")
      {
         other.gameObject.GetComponent<OVRGrabbable>().GrabEnd(new
Vector3(0, 0, 0), new Vector3(0, 0, 0));
         other.gameObject.GetComponent<OVRGrabbable>().enabled = false;
         other.gameObject.transform.position = this.transform.position;
         if (neededName ==
other.gameObject.GetComponent<Planet>().PlanetName)
         {
            passed = true;
            root.AddScorePlanets();
            root.FoundAStar("placed " + neededName);
            root.TellerSayStory(tellerClip);
         }
         root.totalTasks++;
         Destroy(this.gameObject);
      }
   }
}
```

SpaceInterfaceController.cs

Содержит в себе счётчики различного количества заданий и их выполнения, текст и его анимацию, названия и анимации планет, нахождение созвездий, звуки рассказчика (информация о правильно поставленной планете), завершения задачи и информации о прогрессе.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
```

```
public class SpaceInterfaceController : MonoBehaviour
{
    [SerializeField] private int maxTasks;
    public int totalTasks;
    [SerializeField] private Text scoreStarsText;
    [SerializeField] private Text scorePlanetsText;
    public int scoreStars;
    public int scorePlanets;
    [SerializeField] private int maxScoreStars;
    [SerializeField] private int maxScorePlanets;

    [SerializeField] private Text infoscreenText;
    [SerializeField] private Animator infoscreen;
    public Text CurrentPlanetText;

    public GameObject Planets;
    public GameObject PlanetsAnimation;
    public List<GameObject> AllPlanets;

    [SerializeField] private AudioSource teller;
    [SerializeField] private AudioSource completeTask;
    [SerializeField] private AudioSource info;

    TimerLogic timer;
    private bool finished;

    private void Start()
    {
        finished = false;
        timer = GetComponent<TimerLogic>();
        UpdateScore();
    }
    private void FixedUpdate()
    {
        if (!finished)
        {
            if (totalTasks >= maxTasks)
            {
                if (scorePlanets < 8)
                {
                    infoscreenText.text = "Planets are in wrong positions";
                    infoscreen.Play("Show");
                }
```

```
            if (scoreStars < 2)
            {
                infoscreenText.text = "Watch and learn more about stars";
                infoscreen.Play("Show");
            }
            if (scorePlanets >= 8 && scoreStars >= 2)
            {
                infoscreenText.text = "Good Work!";
                Planets.SetActive(false);
                PlanetsAnimation.SetActive(true);
                for (int i = 0; i<= AllPlanets.Count-1; i++)
                {
                    Destroy(AllPlanets[i]);
                }
                infoscreen.Play("Show");
            }
            finished = true;
        }
    }
}
public void TellerSayStory(AudioClip clip)
{
    teller.clip = clip;
    teller.Play();
}
public void UpdateScore()
{
    scoreStarsText.text = "Constellation found - " + scoreStars + "/" +
maxScoreStars;
    scorePlanetsText.text = "Planets placed - " + scorePlanets + "/" +
maxScorePlanets;
}
public void AddScoreStars()
{
    scoreStars++;
    UpdateScore();
    completeTask.Play();
}
public void AddScorePlanets()
{
    scorePlanets++;
    UpdateScore();
    completeTask.Play();
}
```

```
    public void FoundAStar(string starText)
    {
        infoscreenText.text = starText;
        infoscreen.Play("Show");
        info.Play();
    }
    public void SuccsessPlanets()
    {
        infoscreenText.text = "All planets placed correctly!";
        infoscreen.Play("Show");
        info.Play();
    }

}
```

StarsLogic.cs

Логика звёзд, то есть условия проявления созвездия на небе (при наведении на него головы).

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class StarsLogic : MonoBehaviour
{
    [SerializeField] private string StarName;
    [SerializeField] private Image ClearStar;
    [SerializeField] private SpaceInterfaceController root;
    private float state = 0f;
    private bool seen;
    private void Start()
    {
        seen = false;
    }
    /* private void Update()
     {

         if (state >= 1)
         {
             root.AddScoreStars();
             root.FoundAStar(StarName);
         }
```

```
    }*/
    public void LookAt()
    {
        //enable = true;
    }
    private void OnTriggerStay(Collider other)
    {
        if (other.tag == "Looker")
        {
            if (state < 1)
            {
                ClearStar.color = new Color(ClearStar.color.r, ClearStar.color.g,
ClearStar.color.b, state);
                state += 0.006f;
            }
            if (state >= 1 && !seen)
            {
                seen = true;
                root.AddScoreStars();
                root.FoundAStar(StarName);
                root.totalTasks++;
            }
        }
    }
}
```

3.2 Звуковое сопровождение в приложении

Были задействованы аудиофайлы в формате wav, которые примечательны тем, что дают более высокое качество звучания, а это важно, поскольку это улучшает восприятия виртуального мира.

В основном, можно поделить виды звукового сопровождения в данном дипломном проекте на три вида:

- Фоновая музыка, которая несёт в себе цель создания атмосферы, которая будет окружать пользователя во время игры;

- Нарратив, который призван дать конкретные знания в определённой теме. Например, в обсерватории можно узнать факты о планетах;

- Звуки предметов, которые привносят реализм в интерактивность окружающего пространства. Например, треск огня сигнализирует о возникшем пожаре, а звук огнетушителя – о тушении бушующего пламени.

Природа звука, воссозданная в данном проекте, была пространственной.

Пространственный звук – звук, передаваемый во всевозможных направлениях. Иными словами, направление звука не заключено в рамки стерео (левое и правое ухо), а расширено до верхней и нижней сторон. Это называется бинауральным синтезом, который симулируется с помощью задержек между приёмом звука и частотой его фильтрации.

Преимуществами пространственного звука являются усиление визуального эффекта и заполнение частей пространства, лежащих за гранью видимости, что улучшает погружение в виртуальный мир.

Для достижения пространственного звука, Oculus использует две технологии – Spatializer Component, который обрабатывает звук бинаурально, и Spatial Reverb, который создаёт отражение звука и его реверберацию в пространстве. [10]

ЗАКЛЮЧЕНИЕ

Исходя из введения были определены актуальность, предмет исследования, цель, проанализированы и выбраны два основных концептуальных сценария приложения.

Актуальность темы дипломного проекта заключалась в подчёркивании важности технологии виртуальной реальности как активно развивающейся и её сути как таковой.

Предмет исследования сводился к применению VR в обучении и уменьшению затрат по постройке симуляционных помещений.

Цель же проекта была раскрыта как создание продукта виртуальной реальности, который позволил бы обучать различного рода знаниям.

Концептуальными сценариями стали виртуальная обсерватория и ОБЖ, которые продемонстрировали два разных полюса динамики, заключённых в получении важных знаний наглядным образом.

В первом пункте обозревались решения виртуальной реальности и её технологии. Там были выявлены преимущества и недостатки данной технологии, а также схожие решения, которые определили вектор направления реализации проекта и его визуализации в виде задач.

Во втором пункте был рассмотрен выбор средств проектирования и разработки приложения. В нём раскрылись важные детали движка Unity, его сильные стороны, которые выделили этот движок среди альтернативных вариантов. Кроме того, был изучен язык программирования C#, произвелось ознакомление с его использованием в написании скриптов и гарнитура OculusQuest 2, на основе которой произошла разработка основного функционала проекта, с учётом её интеграции в Unity. Помимо этого, была предоставлена возможная теория использования данного приложения в уже теоретически существующей информационной системе учебного заведения.

Третий пункт затронул проектирование и разработку приложения, которая достигалась через понимание архитектуры проекта и его реализации с помощью скриптов и создания звукового сопровождения. Были написаны скрипты для «охраны безопасности жизнедеятельности» и «виртуальной обсерватории».

Скрипты для ОБЖ позволили воссоздать симуляцию пожара, в котором время оказалось ограниченным, каждый предмет по-своему важен и любое действие приводило к выполнению того или иного задания. Например, фонарик освещал помещение, огнетушитель тушил пожар, телефон вызывал пожарных, а ключ открывал дверь. Скрипты для обсерватории позволили расставить планеты по местам, услышать интересные факты о них. Помимо

того, была создана возможность увидеть созвездия при продолжительном просмотре на их область появления. Звуковая часть помогла улучшить погружение в виртуальный мир, понять атмосферу обстановки и представить себя на месте виртуального персонажа.

Практическая значимость исследования важных аспектов проекта позволила узнать лучше технологию виртуальной реальности, применить её для поставленной цели и раскрыть потенциал, заложенный в ней. Более того, был создан продукт, который, как планируется, будет развиваться и дальше, оснащаться новыми предметами для освоения обучающего материала и даже интегрирован в уже существующие системы, чтобы улучшить их качество работы.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Лиза Б. Преимущества и недостатки виртуальной реальности [Электронный ресурс]. / Б. Лиза – Wondershare Filmora, 2021. – Режим доступа: https://filmora.wondershare.com/virtual-reality/pros-cons-virtual-virtual.html
2. Дерек С. У виртуальной реальности есть проблемы. Вот какими методами разработчики игр пытаются их удалить [Электронный ресурс]. / С. Дерек – The Seattle Times, 2021. – Режим доступа: https://www.seattletimes.com/business/technology/virtual-reality-has-real-problems-heres-how-game-developers-seek-to-delete-them/#:~:text=Nausea%2C%20dizziness%2C%20disorientation%20and%20a,them%20that%20they%20are%20still
3. Софи Т. VR для корпоративных тренингов: существующие примеры использования виртуальной реальности [Электронный ресурс]. / Т. Софи – Virtual Speech, 2019. – Режим доступа: https://virtualspeech.com/blog/how-is-vr-changing-corporate-training
4. Анураг 13 плюсов и минусов выбора Unity, которые нужно знать [Электронный ресурс]. / Анураг – Newgenapps, 2018. – Режим доступа: https://www.newgenapps.com/blog/unity-3d-pros-cons-analysis-choose-unity/#:~:text=Pros%20of%20Using%20Unity%203D%3A&text=It%20is%20very%20effective%20while,platform%20development%20and%20multiplatform%20games
5. Разработка игры на Unity 3D: преимущества и недостатки [Электронный ресурс]. / Logic Simplified. – Режим доступа: https://logicsimplified.com/newgames/unity-3d-game-development-advantages-and-disadvantages/
6. Использование интерфейса Unity [Электронный ресурс]. / Unity Technologies, 2020. – Режим доступа: https://learn.unity.com/tutorial/using-the-unity-interface#5c7f8528edbc2a002053b6c9
7. Сэм С. Преимущества и приложения C# [Электронный ресурс]. / С. Сэм – Tutorials Point, 2018. – Режим доступа: https://www.tutorialspoint.com/Chash-Language-advantages-and-applications
8. Написание кода на C# в Unity для начинающих [Электронный ресурс]. / Unity. – Режим доступа: https://unity3d.com/learning-c-sharp-in-unity-for-beginners#:~:text=The%20language%20that's%20used%20in,variables%2C%20functions%2C%20and%20classes

9. Понимание компонентов интеграционного пакета Oculus [Электронный ресурс]. / Oculus. – Режим доступа: https://developer.oculus.com/documentation/unity/unity-utilities-overview/?device=QUEST
10. Звук в VR [Электронный ресурс]. / Unity Technologies, 2020. – Режим доступа: https://learn.unity.com/tutorial/unit-7-sound-in-vr#5df800f4edbc2a194605365f

ПРИЛОЖЕНИЕ А

Список звуков, задействованных в проекте (Рисунок 1-2):

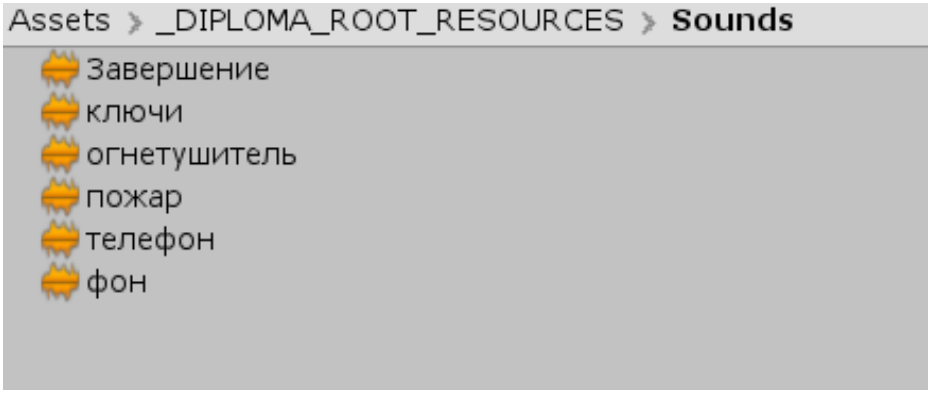


Рисунок 1 – Звуки для ОБЖ

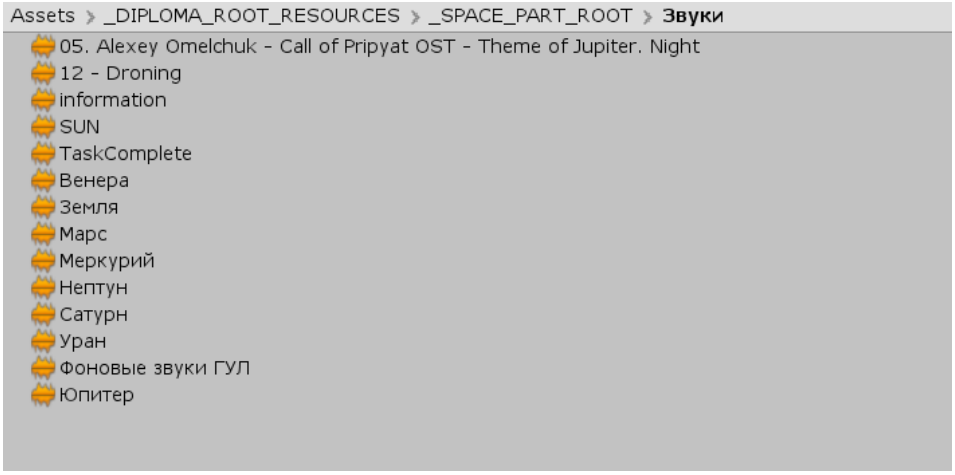


Рисунок 2 – Звуки для виртуальной обсерватории

ПРИЛОЖЕНИЕ Б

Листинг Б – Скрипт OVRGrabbable

```
using System;
using UnityEngine;

/// <summary>
/// An object that can be grabbed and thrown by OVRGrabber.
/// </summary>
public class OVRGrabbable : MonoBehaviour
{
    [SerializeField]
    protected bool m_allowOffhandGrab = true;
    [SerializeField]
    protected bool m_snapPosition = false;
    [SerializeField]
    protected bool m_snapOrientation = false;
    [SerializeField]
    protected Transform m_snapOffset;
    [SerializeField]
    protected Collider[] m_grabPoints = null;

    protected bool m_grabbedKinematic = false;
    protected Collider m_grabbedCollider = null;
    protected OVRGrabber m_grabbedBy = null;

        /// <summary>
        /// If true, the object can currently be grabbed.
        /// </summary>
    public bool allowOffhandGrab
    {
        get { return m_allowOffhandGrab; }
    }

        /// <summary>
        /// If true, the object is currently grabbed.
        /// </summary>
    public bool isGrabbed
    {
        get { return m_grabbedBy != null; }
    }

        /// <summary>
        /// If true, the object's position will snap to match snapOffset when grabbed.
```

```
        /// </summary>
    public bool snapPosition
    {
        get { return m_snapPosition; }
    }

        /// <summary>
        /// If true, the object's orientation will snap to match snapOffset when
grabbed.
        /// </summary>
    public bool snapOrientation
    {
        get { return m_snapOrientation; }
    }

        /// <summary>
        /// An offset relative to the OVRGrabber where this object can snap when
grabbed.
        /// </summary>
    public Transform snapOffset
    {
        get { return m_snapOffset; }
    }

        /// <summary>
        /// Returns the OVRGrabber currently grabbing this object.
        /// </summary>
    public OVRGrabber grabbedBy
    {
        get { return m_grabbedBy; }
    }

        /// <summary>
        /// The transform at which this object was grabbed.
        /// </summary>
    public Transform grabbedTransform
    {
        get { return m_grabbedCollider.transform; }
    }

        /// <summary>
        /// The Rigidbody of the collider that was used to grab this object.
        /// </summary>
    public Rigidbody grabbedRigidbody
```

```
    {
        get { return m_grabbedCollider.attachedRigidbody; }
    }

        /// <summary>
        /// The contact point(s) where the object was grabbed.
        /// </summary>
    public Collider[] grabPoints
    {
        get { return m_grabPoints; }
    }

        /// <summary>
        /// Notifies the object that it has been grabbed.
        /// </summary>
        virtual public void GrabBegin(OVRGrabber hand, Collider grabPoint)
    {
        m_grabbedBy = hand;
        m_grabbedCollider = grabPoint;
        gameObject.GetComponent<Rigidbody>().isKinematic = true;
    }

        /// <summary>
        /// Notifies the object that it has been released.
        /// </summary>
        virtual public void GrabEnd(Vector3 linearVelocity, Vector3
angularVelocity)
    {
        Rigidbody rb = gameObject.GetComponent<Rigidbody>();
        rb.isKinematic = m_grabbedKinematic;
        rb.velocity = linearVelocity;
        rb.angularVelocity = angularVelocity;
        m_grabbedBy = null;
        m_grabbedCollider = null;
    }

    void Awake()
    {
        if (m_grabPoints.Length == 0)
        {
            // Get the collider from the grabbable
            Collider collider = this.GetComponent<Collider>();
            if (collider == null)
            {
```

```
                throw new ArgumentException("Grabbables cannot have zero grab points and no collider -- please add a grab point or collider.");
            }

            // Create a default grab point
            m_grabPoints = new Collider[1] { collider };
        }
    }

    protected virtual void Start()
    {
        m_grabbedKinematic = GetComponent<Rigidbody>().isKinematic;
    }

    void OnDestroy()
    {
        if (m_grabbedBy != null)
        {
            // Notify the hand to release destroyed grabbables
            m_grabbedBy.ForceRelease(this);
        }
    }
}
```

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ КАЗАХСТАН

Satbayev University

Отзыв научного руководителя

Дипломная работа

Баймолдина Анель Муратқызы

5В070300 – Информационные системы

Тема: Создание симуляции пространства обучения в условиях, приближенных к реальным ситуациям, с использованием VR технологий

Групповая дипломная работа на тему «Создание симуляции пространства обучения в условиях, приближенных к реальным ситуациям, с использованием VR технологий» - представляет собой выпускную квалификационную работу по специальности 5В07300 – «Информационные системы», написанными Баймолдиной А.М. и Смагуловой Н.Д. Целью данной работы является создание обучающего VR-приложения, которое позволит усовершенствовать подачу знаний учебных заведениях.

Изучив и выбрав инструментальные средства, перенеся разработанные сценарии в виртуальную среду, снабдив их тех самым системой интерфейса и взаимодействия с окружением, а также внедрив звуковое сопровождение, студент Баймолдина А.М. полностью выполнила поставленные задачи в рамках своих обязанностей в дипломном проекте.

Дипломная работа состоит из 3 глав. В первой главе рассмотрены подробным образом преимущества и недостатки виртуальной реальности, а также проведён достоверный анализ существующих решений. Во второй главе были обозначены задачи и описаны основные инструментальные средства, которые были избраны для разработки проекта. Каждый инструмент был расписан лаконично, а также были приведены иллюстрации и примеры работы с ним. Третья глава соотносится с раскрытием основных действий, направленных на реализацию проекта. Два основных аспекта – написание скриптов и создание звукового сопровождения были описаны результативно, что позволило увидеть полный объём проделанной работы и оценить его надлежащим образом.

Дипломная работа на тему «Создание симуляции пространства обучения в условиях, приближенных к реальным ситуациям, с использованием VR технологий» выполнена Баймолдиной А.М. на хорошем уровне и может быть допущена к защите.

Научный руководитель
Магистр технических наук., Лектор

Аристомбаева М.Т.

«29» мая 2021

Протокол анализа Отчета подобия Научным руководителем

Заявляю, что я ознакомилась с Полным отчетом подобия, который был сгенерирован Системой выявления и предотвращения плагиата в отношении работы:

Автор: Баймолдина Анель Муратқызы
Название: Создание симуляции пространства обучения в условиях, приближенных к реальным ситуациям, с использованием VR технологий
Координатор: Аристомбаева Меруерт Тұрлұбекқызы
Коэффициент подобия 1: 0
Коэффициент подобия 2: 0
Коэффициент цитирования: 2,5

После анализа Отчета подобия констатирую следующее:

✓ ☐ обнаруженные в работе заимствования являются добросовестными и не обладают признаками плагиата. В связи с чем, признаю работу самостоятельной и допускаю ее к защите;

☐ обнаруженные в работе заимствования не обладают признаками плагиата, но их чрезмерное количество вызывает сомнения в отношении ценности работы по существу и отсутствием самостоятельности ее автора. В связи с чем, работа должна быть вновь отредактирована с целью ограничения заимствований;

☐ обнаруженные в работе заимствования являются недобросовестными и обладают признаками плагиата, или в ней содержатся преднамеренные искажения текста, указывающие на попытки сокрытия недобросовестных заимствований. В связи с чем, не допускаю работу к защите.

Обоснование:

Заимствования являются добросовестными и не обладают признаками плагиата, объясняются использованием общепринятой терминологией, а также названиями программ и файлов.

«28» мая 2021 г. Аристомбаева М.Т.,

Дата  Подпись Научного руководителя

Протокол анализа Отчета подобия заведующего кафедрой

Заведующий кафедрой заявляет, что ознакомился (-лась) с Полным отчетом подобия, который был сгенерирован Системой выявления и предотвращения плагиата в отношении работы:

Автор: Баймолдина Анель Муратқызы
Название: Создание симуляции пространства обучения в условиях, приближенных к реальным ситуациям, с использованием VR технологий
Координатор: Аристомбаева Меруерт Тұрлұбекқызы
Коэффициент подобия 1: 0
Коэффициент подобия 2: 0
Коэффициент цитирования: 2,5

После анализа Отчета подобия констатирую следующее:

✓ ☐ обнаруженные в работе заимствования являются добросовестными и не обладают признаками плагиата. В связи с чем, признаю работу самостоятельной и допускаю ее к защите;

☐ обнаруженные в работе заимствования не обладают признаками плагиата, но их чрезмерное количество вызывает сомнения в отношении ценности работы по существу и отсутствием самостоятельности ее автора. В связи с чем, работа должна быть вновь отредактирована с целью ограничения заимствований;

☐ обнаруженные в работе заимствования являются недобросовестными и обладают признаками плагиата, или в ней содержатся преднамеренные искажения текста, указывающие на попытки сокрытия недобросовестных заимствований. В связи с чем, не допускаю работу к защите.

Обоснование:

После анализа отчёта по плагиату и работы дипломника выявлено, что заимствования являются добросовестными и не обладают признаками плагиата, поскольку связаны с применением общеизвестных терминов и также названий общедоступных программ и файлов.

«28» мая 2021 г.

Дата Сейлова Н.А., подпись
зав.кафедрой