Муниципальное автономное общеобразовательное учреждение
Средняя школа № 8

Проектная работа по информатике

«Создание игры с применением игровой среды Unity и языка программирования c#»

Выполнил: ученик 10 «Б» класса
Бушуев Дмитрий
Руководитель: Кустова Ю.Е.

Актуальность темы учебного исследования

Сфера видеоигровых развлечений может повлиять на выбор профессии в будущем.

В настоящее время видеоигру трудно отличить от качественного фильма.

Данное направление сегодня развивается быстрыми темпами, является популярным и требует специалистов, которые получают достойную заработанную плату.

Эти причины подтверждают актуальность темы проекта.

Цель и задачи исследовательской работы:

Цель работы – разработать свою видеоигру

Задачи:

- познакомиться с литературой об игровой среде Unity, языке программирования c#;
- рассмотреть синтаксическое строение языка c#;
- рассмотреть функционал игровой среды Unity;
- создать игру;
- сделать выводы о проделанной работе;

Гипотеза

С помощью игровой среды Unity и языка программирования C#, я смогу сделать свою игру

Введение

C# – объектно-ориентированный язык программирования, разработан в 1998 году группой инженеров компании Microsoft.

Разрабатывался как язык программирования прикладного уровня для CLR и, зависит от ее возможностей. Последняя версия языка вышла 10 ноября 2020 года.

Unity — среда разработки компьютерных игр, позволяющая создавать приложения, работающие на более чем 25 различных платформах, включающих персональные компьютеры, игровые консоли, мобильные устройства, интернет-приложения и другие.

Создание игры

Для создания игры я выделил следующие этапы:

1. сформулировать сценарий игры;
2. создать игровой мир;
3. создать таймер круга (с возможностью отображения лучшего круга);
4. провести эксперимент по запуску видеоигры;
5. провести анализ и корректировку, оптимизируя игру;

Этап 1. Сценарий игры

В сценарий игры я заложил сюжет о начинающем пилоте, который начинает свою карьеру с картинга. В недалеком будущем он решает подписать контракт с гоночной командой.

Но есть условие со стороны гоночной команды, предъявляемое каждому кандидату, необходимо поставить лучшее время на трассе Autodromo de Nationale Monza на Porche 911 GT-3R.

Этап 2. Создание игрового мира

На 2 этапе загружаются необходимые ассеты из магазина Unity в проект (Asset store, магазин для скачивания 3D моделей, текстур и скриптов).

Далее создание очертания трека (модель дороги переносится из Assets-Track-Prefabs в 3D пространство и объект «Terrain» для расстановки деревьев. Результат данного этапа работы можно посмотреть в приложении 1

Приложение 1

Этап 3. Создание таймера круга

На 3 этапе создавались таймер круга с применением языка программирования C#, скрипт и триггер для системы лучшего круга.

Последовательно внедрялись скрипты с именами «LapTimeManager», «LapComplete» и «HalfPointTrigger».

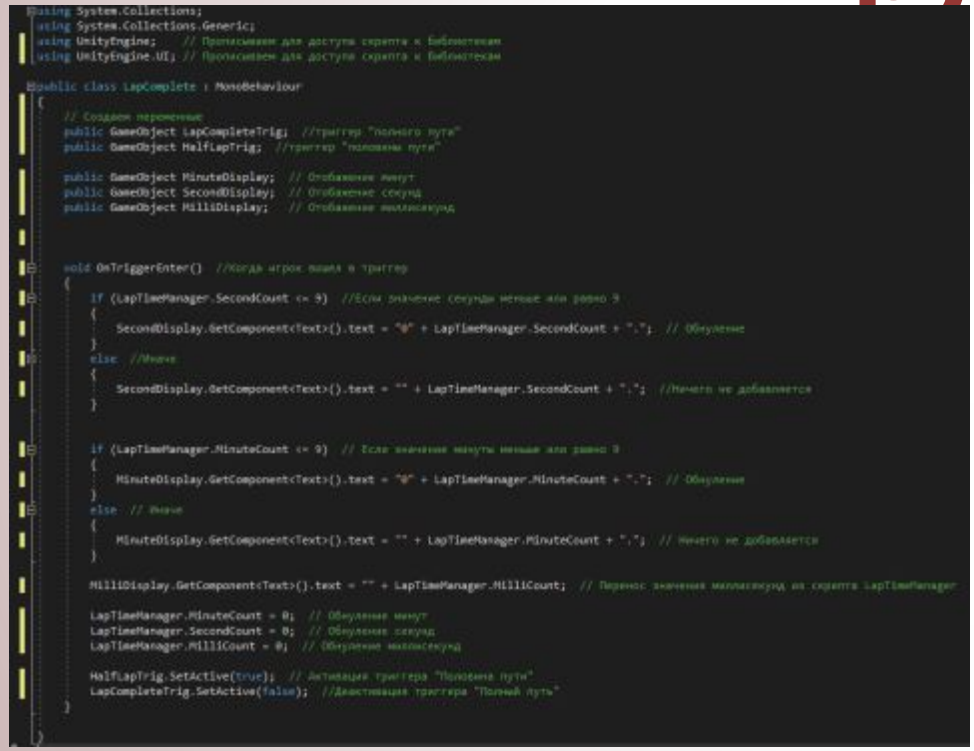
Unity автоматически в файле сделал необходимую заготовку. Результат данного этапа можно наблюдать в приложении 2.

Этап 4. Создание таймера круга

пример скрипта

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;    // Прописываем для доступа скрипта к библиотекам
using UnityEngine.UI;// Прописываем для доступа скрипта к библиотекам

public class LapTimeManager : MonoBehaviour
{
    // Создаем переменные
    public static int MinuteCount;  // Счет минут
    public static int SecondCount;  // Счет секунд
    public static float MilliCount; // счет миллисекунд
    public static string MilliDisplay; // Отображение миллисекунд


    public GameObject MinuteBox; // Отобажение минут
    public GameObject SecondBox; // Отображение секунд
    public GameObject MilliBox;  // Отображение миллисекунд

    // Update is called once per frame
    void Update()
    {
        MilliCount += Time.deltaTime * 1000;  // Начинается отсчет времени
        MilliDisplay = MilliCount.ToString("F0");
        MilliBox.GetComponent<Text>().text = "" + MilliDisplay; //Доступ к элементу Text в Unity

        if (MilliCount >=1000)  // Если значение миллисекунды больше или равно 1000
        {
            MilliCount = 0;  //Обнуление миллисекунд
            SecondCount += 1;  // Прибавление значения в секунду на 1 единицу
        }

        if (SecondCount <= 9)  // Если значение секунды меньше или равно 9
        {
            SecondBox.GetComponent<Text>().text = "0" + SecondCount + ".";  // Обнуление
        }
        else  // Иначе
        {
            SecondBox.GetComponent<Text>().text = "" + SecondCount + ".";  // Ничего не добавлябется
        }

        if (SecondCount >=60) // Если значение секунды меньше или равно 9
        {
            SecondCount = 0;  // Значение секунд обнуляется
            MinuteCount += 1; // Прибавление значения в минуту на 1 единицу
        }

        if(MinuteCount <=9)  // Если значение секунды меньше или равно 9
        {
            MinuteBox.GetComponent<Text>().text = "0" + MinuteCount + ":";  // Обнуление
        }
        else  // Иначе
        {
            MinuteBox.GetComponent<Text>().text = "" + MinuteCount + ":";  // Ничего не добавляется
        }

    }
}
```

Этап 4. Создание таймера круга



Скрипт LapComplete

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
5   public class HalfPointTrigger : MonoBehaviour
6   {
7       // Создаем переменные
8       public GameObject LapCompleteTrig;  // Создание триггера "Полный путь"
9       public GameObject HalfLapTrig;      // Создание триггера "Половина пути"
10
11      void OnTriggerEnter()  // Когда игрок вошел в триггер
12      {
13          LapCompleteTrig.SetActive(true);  // Активация триггера "Полный путь"
14          HalfLapTrig.SetActive(false);     // Деактивация триггера "Половина пути"
15      }
16  }
```

Скрипт HalfPointTrigger

Приложение 2

Этап 4. Эксперимент

После того, когда все основные элементы видеоигры созданы, был проведен пробный запуск.

В ходе тестирования видеоигры были обнаружены проблемы с производительностью, а именно с низкой частотой кадров, что подтвердила программа «MSI Afterburner». И я решил оптимизировать видеоигру (Данную проблему вы можете наблюдать в приложении 3).

Приложение 3

Этап 5. Оптимизация видеоигры

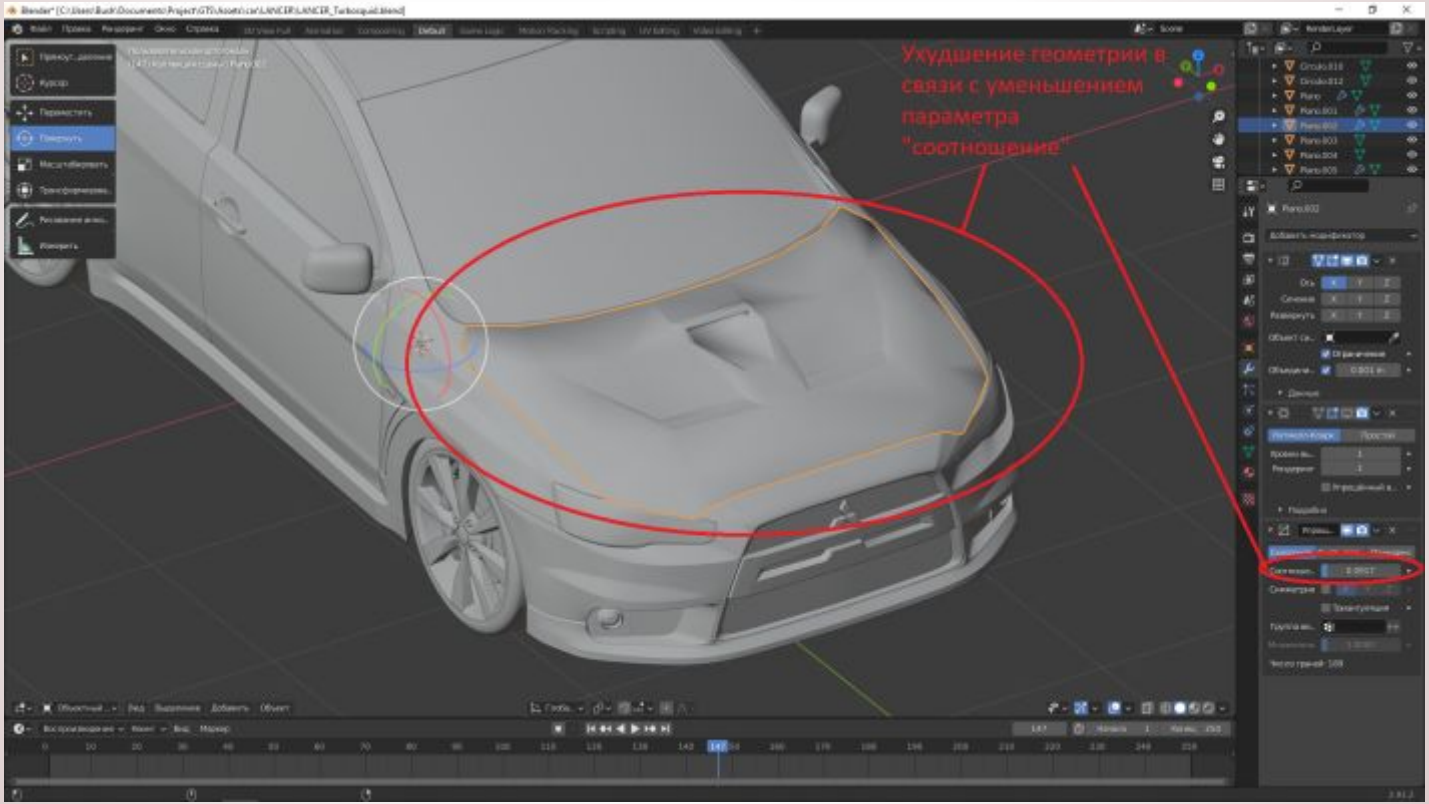
Оптимизация моделей с помощью Blender

В настоящее время 3D модели распространяются в очень высоком качестве геометрии, что очень нагружает видеоадаптер.

По этой причине была создана LOD система. Она переключает модели при сближении игрока к объекту, с более низко качественной к более качественной и наоборот.

После импортирования модели я уменьшил параметр «соотношение». Что влияет на качество геометрии следующим образом - чем меньше параметр, тем лучше, но при маленьком значении качество геометрии может быть настолько низким, что объект может пропасть (приложении

Приложение 4



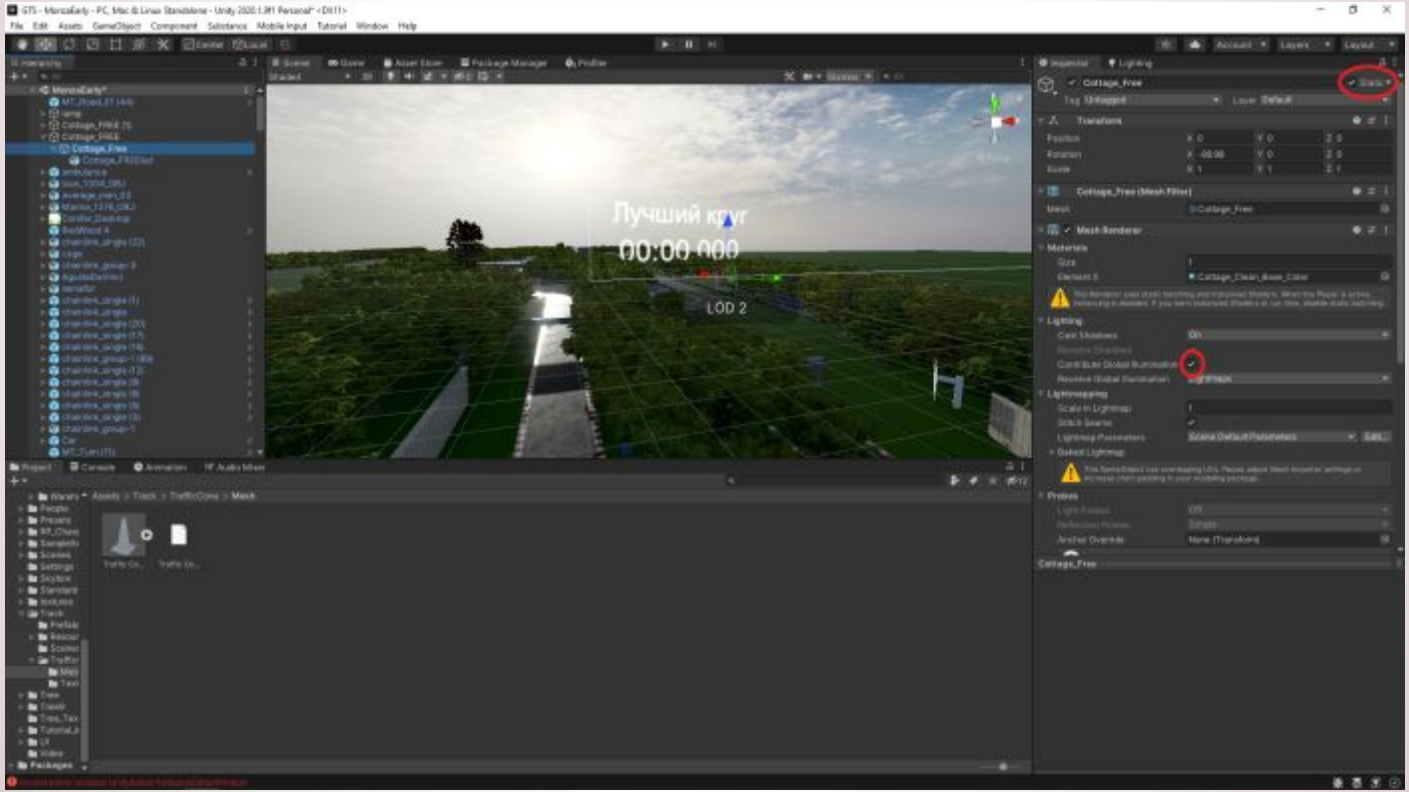
Этап 5. Оптимизация видеоигры

оптимизация освещения Static batching

Static batching позволил уменьшить количество вызовов отрисовки для геометрии любого размера, но применимо лишь для статичных объектов.

Результат данного этапа работы можно посмотреть в приложении 5

Приложение 5



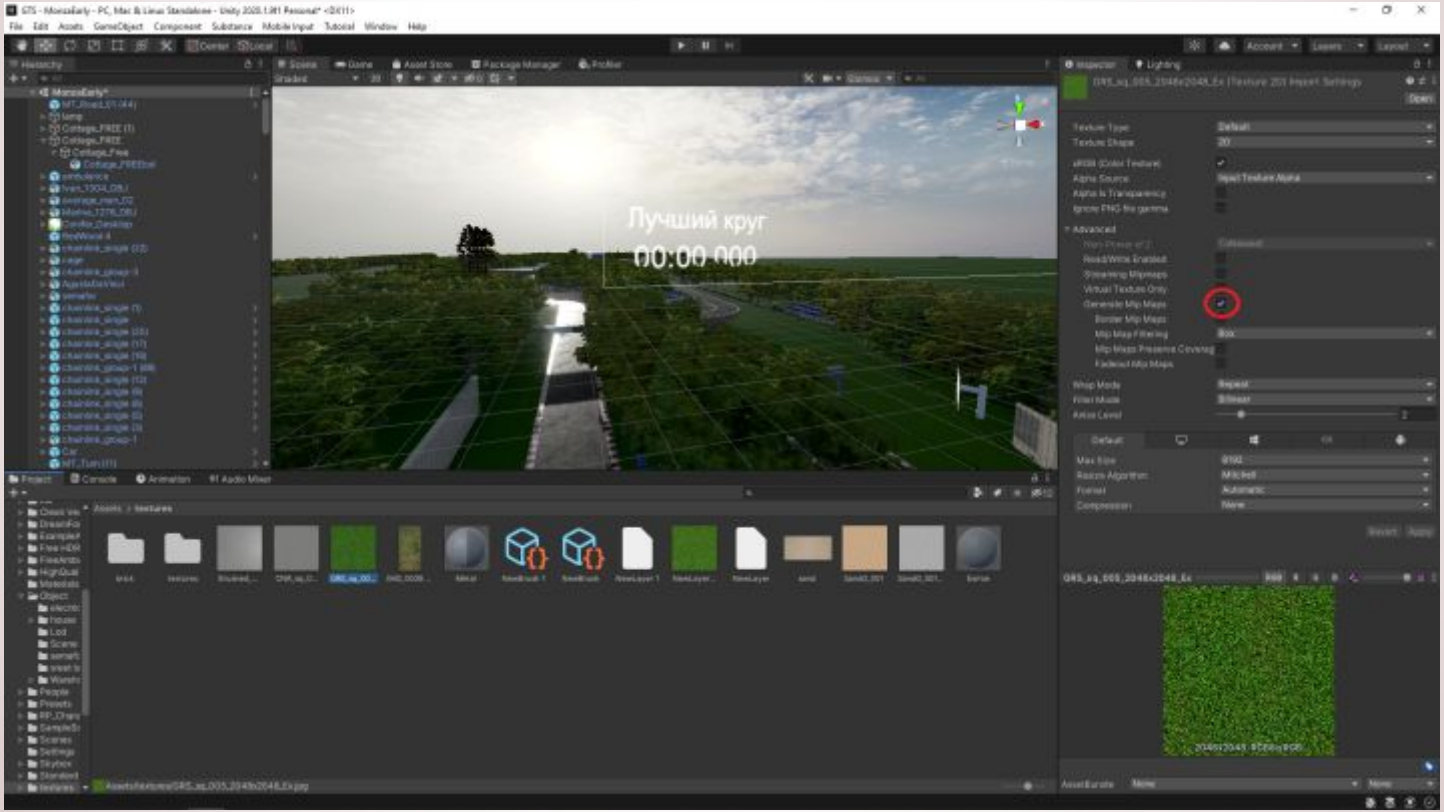
Этап 5. Оптимизация видеоигры

Использование Mip Map текстур

Я произвел сжатие текстур, что позволило графическому ускорителю использовать для маленьких треугольников текстуры пониженного разрешения.

Результат данного действия работы можно посмотреть в приложении 6

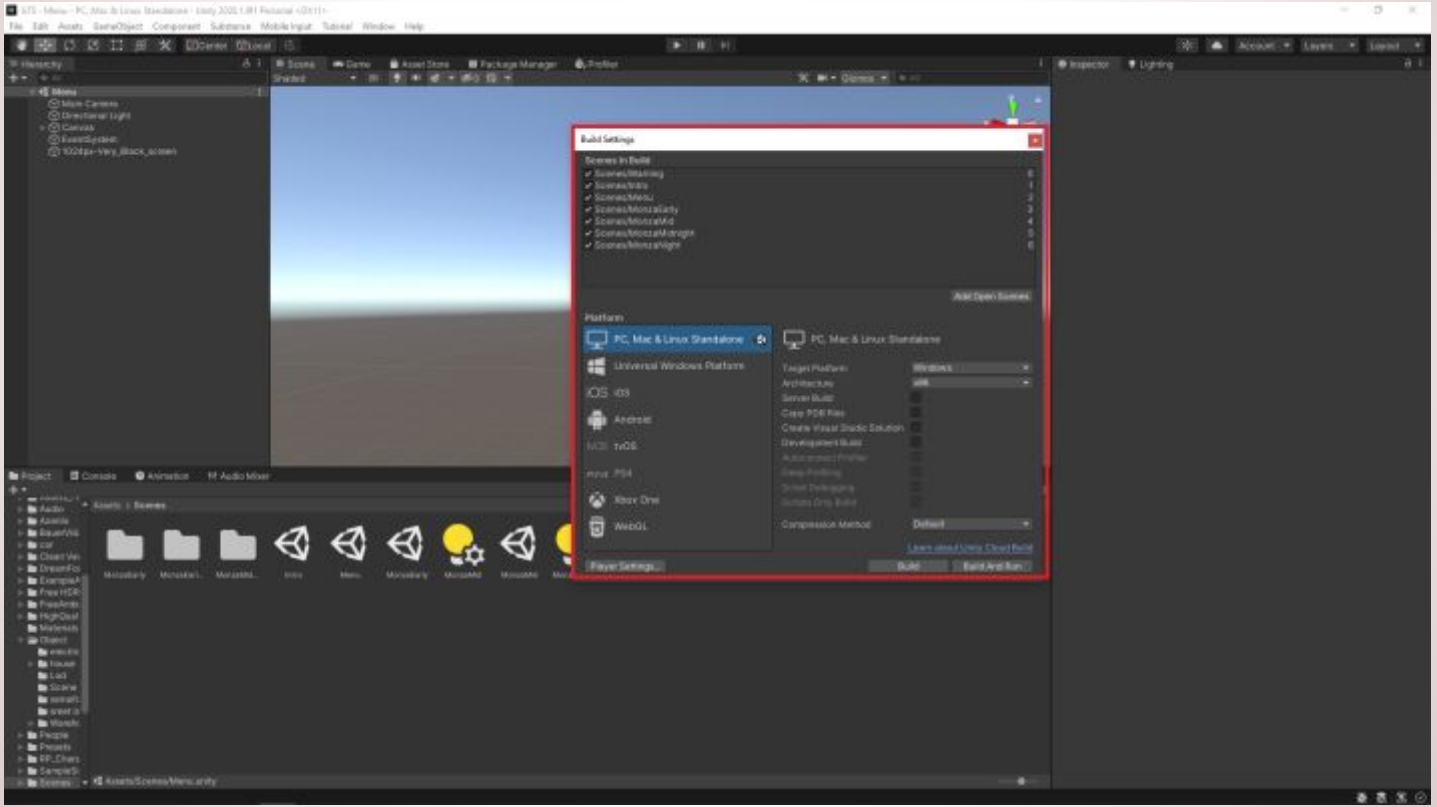
Приложение 6



Этап 6. Сборка игры и ее запуск

На последнем этапе я выполнил сборку игры и проверил ее работоспособность. Для этого запустил файл и попробовал найти ошибки в своей игре. Запуск игры не подтвердил ошибок (Приложение 7)

Приложение 7



Анкетирование

По окончанию работы мне стало интересно мнение моих одноклассников по поводу качества моей видеоигры, и я решил провести анкетирование.

Я сформулировал следующие вопросы и предложил шкалу оценивания

1. Оценка визуального исполнение игры (1-5 баллов)
2. Оценка игровой механики, возможности в игре (1-5 баллов)
3. Оценка технического состояния игры (отсутствие «багов», ошибок) (1-5 баллов)
4. Оценка качества интерфейса игры (1-5 баллов)
5. Оценка качества работы игры на вашем компьютере (1-5 баллов) (Высокая частота кадров)
6. Оценка звукового оформления игры (1-5 баллов)

Анкетирование

7. Заинтересовались ли вы работой в информационной сфере

Ответы 1 – да, 2 - нет

8. Оценка информативности данного проекта по созданию компьютерной видеоигры (1-5 баллов)

9. Пробовали ли вы найти работу в сфере информационных технологий

Ответы 1 – да, 2 - Нет

10. Итоговая оценка данной видеоигры (1-10 баллов)

В результате мной была сформулирована таблица

Номер вопроса	1	2	3	4	5	6	7	8	9	10
Процентное соотношение	80%	40%	95%	80%	75%	90%	50%	65%	70%	65%

Заключение

В результате работы над проектом мной были детально изучены: теоретические основы языка программирования C#; изучены основы создания видеоигрового мира; способы оптимизации видеоигры;

В ходе исследования я подтвердил гипотезу и пришел к следующим выводам:

1. Чем сложнее и качественнее видеоигра, тем больше времени нужно потратить на ее разработку;

2. Оптимизация видеоигры является важной частью работы при ее разработке;

3. Легче и быстрее разработать игру в игровой среде, чем с помощью одного языка программирования;

Список использованной литературы

Интернет ресурсы

1. https://docs.unity3d.com/Manual/index.html
2. http://unity3d.ru/distribution/index.php
3. https://ru.wikipedia.org
4. https://docs.microsoft.com/ru-ru/dotnet/csharp