МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Кемеровский государственный университет»
Институт фундаментальных наук
Кафедра ЮНЕСКО по информационным вычислительным технологиям

ОТЧЕТ

по учебной практике, технологической (проектно-технологической) практике

проект «Cooler FPS»
(название проекта)

студентов 1 курса

Арышева Владимира Владимировича
(ФИО полностью)
Колесникова Алексея Леонидовича
(ФИО полностью)
Силивончик Анастасии Сергеевны
(ФИО полностью)

направление подготовки 02.03.03 Математическое обеспечение и администрирование информационных систем.
направленность (профиль) подготовки «Информационные системы и базы данных».

Руководитель практики:
канд. физ.-мат. наук, доцент
К.С. Иванов
(ученая степень, звание, должность, ФИО)
Зав. кафедрой ЮНЕСКО по ИВТ
доктор физ.-мат. наук, профессор
Ю.Н. Захаров
(ученая степень, звание, должность, ФИО)

Работа защищена с оценками:

Арышев В.В.	удовлетворительно	«_____»	2021 г.
(ФИО)	(оценка)		
Колесников А.Л.	удовлетворительно	«_____»	2021 г.
(ФИО)	(оценка)		
Силивончик А.С.	хорошо	«_____»	2021 г.
(ФИО)	(оценка)		

Кемерово 2021

Репозиторий

https://github.com/L1oid/EdPractice

Описание проекта

Название

Cooler FPS

Назначение проекта

Игра на Windows

Краткое описание

Шутер от первого лица - жанр компьютерных игр, в которых игровой процесс основывается на сражениях с использованием огнестрельного или любого другого оружия с видом от первого лица таким образом, чтобы игрок воспринимал происходящее глазами протагониста. Игра написана на языке программирования C#, используя игровой движок Unity.

Состав, актуальность темы, цели, задачи и план проекта

Состав группы участников проекта

	ФИО	Группа	Username	Роли
1	Арышев Владимир Владимирович	МОА-205	vovarishev	Программист, тимлид
2	Колесников Алексей Леонидович	МОА-205	L1oid	Программист
3	Силивончик Анастасия Сергеевна	МОА-205	Cheesemooncake	Программист

Актуальность:

С развитием компьютерной техники, компьютерные игры прочно вошли в нашу жизнь, как способов организации отдыха. С развитием электроники развиваются и вычислительные способности техники, и средства проектирования интерактивных пространств.

Лидерами на рынке создания компьютерных игр являются такие программные решения как: «Unreal Engine» и «Unity».

Выбором «Unity» как основного средства разработки связан с наличием у участников проекта опыты работы в данном программном решении.

Для ознакомления с большинством функций вышеописанного программного решения основной целью проекта является разработка игры.

Цель и задачи проекта

Изучить средства разработки компьютерных игр;

Изучить средства проектирования трёхмерного интерактивного пространства;
Изучить базовый синтаксис языка C#;
Выполнить проект на основе полученных навыков;
Презентовать результаты выполнения проекта;

Индивидуальные задачи участников

Арышев - Кодинг, разработка игры. Создание 3D моделей и их анимаций, проработка текстур мира игры. Создание локаций;

Колесников - Кодинг, разработка игры. Создание 3D моделей и их анимаций, проработка текстур мира игры. Создание локаций;

Силивончик - Кодинг, разработка игры. Создание 3D моделей и их анимаций, проработка текстур мира игры. Создание локаций;

Календарный план работы:

Изучение средств разработки компьютерных игр **2 февраля - 12 марта;**
Разработка и внедрение игровых механик **15 марта - 30 апреля;**
Разработка и отладка пользовательского интерфейса **3-7 мая;**
Тесты и отладка **10-17 мая;**
Составление отчетности - **24-31 мая;**

Средства разработки

Unity3D – игровой движок, выбран как самым популярный движок для разработчиков с низким уровнем подготовки; Использовалась бесплатная лицензия Unity Personal (условия использования лицензии);

Blender3D - работа с трёхмерными объектами (GNU лицензия);

Основным каналом связи между участниками выступал Discord;

Задачи между участниками распределяли с помощью сервиса Trello.

Инструкция по запуску

1. Зайти на сайт https://github.com/L1oid/EdPractice
2. Скачать папку ReleaseBuild
3. Зайти в папку и запустить файл CoolerKemSU FPS.exe

Ход работы

Система движения персонажа:

Система движения персонажа основана на добавлении скорости как силы, прилагаемой к физическому объекту. Так-же добавлена проверка на столкновения со стенами и плавное ускорение/замедление игрока.

```
private void MoveCharacter()
{
    var direction = new Vector3(input.Move, 0f, input.Strafe).normalized;
    var worldDirection = transform.TransformDirection(direction);
    var velocity = worldDirection * (input.Run ? runningSpeed : walkingSpeed);
    var intersectsWall = CheckCollisionsWithWalls(velocity);
    if (intersectsWall)
    {
        _velocityX.Current = _velocityZ.Current = 0f;
        return;
    }

    var smoothX = _velocityX.Update(velocity.x, movementSmoothness);
    var smoothZ = _velocityZ.Update(velocity.z, movementSmoothness);
    var rigidbodyVelocity = _rigidbody.velocity;
    var force = new Vector3(smoothX - rigidbodyVelocity.x, 0f, smoothZ - rigidbodyVelocity.z);
    _rigidbody.AddForce(force, ForceMode.VelocityChange);
}
```

Реализация прыжков:

Прыжки персонажа также основаны на добавление ускорения, но с проверкой нахождения на земле.

```
private void Jump()
{
    if (!_isGrounded || !input.Jump) return;
    _isGrounded = false;
    _rigidbody.AddForce(Vector3.up * jumpForce, ForceMode.Impulse);
}
```

Реализация стрельбы:

При стрельбе происходит проверка состояний (наличие боеприпасов, перезарядка и т.д.), запуск анимации и инициализация системы частиц.

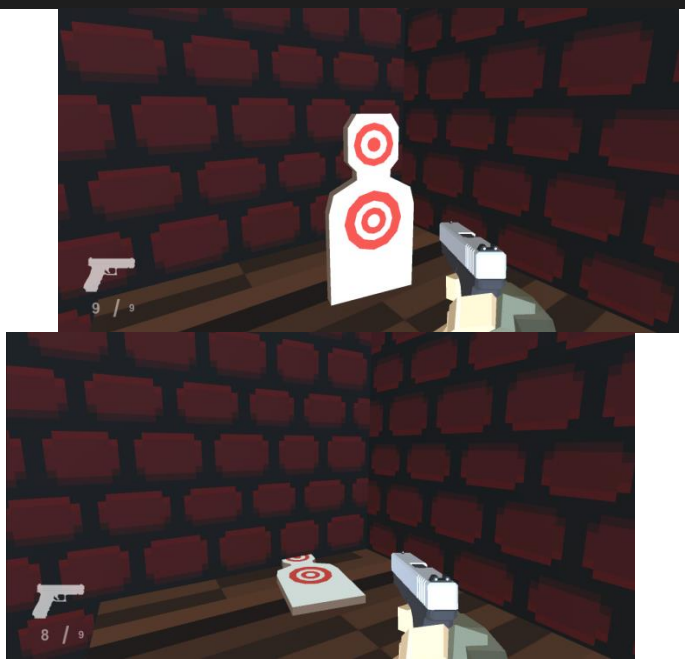
```
if (Input.GetMouseButtonDown (0) && !outOfAmmo && !isReloading && !isInspecting && !isRunning)
{
    anim.Play ("Fire", 0, 0f);

    muzzleParticles.Emit (1);

    currentAmmo -= 1;

    shootAudioSource.clip = SoundClips.shootSound;
    shootAudioSource.Play ();

    StartCoroutine(MuzzleFlashLight());

    if (!isAiming)
    {
        anim.Play ("Fire", 0, 0f);

        muzzleParticles.Emit (1);

        if (enableSparks == true)
        {
            sparkParticles.Emit (Random.Range (1, 6));
        }
    }
    else
    {
        anim.Play ("Aim Fire", 0, 0f);

        if (!randomMuzzleflash) {
            muzzleParticles.Emit (1);
        }
        else if (randomMuzzleflash == true)
        {
            if (randomMuzzleflashValue == 1)
            {
                if (enableSparks == true)
                {
                    sparkParticles.Emit (Random.Range (1, 6));
                }
                if (enableMuzzleflash == true)
                {
                    muzzleParticles.Emit (1);
                    StartCoroutine (MuzzleFlashLight ());
                }
            }
        }
    }
```

Добавление моделей:
Теперь игровые объекты не выглядят как прямоугольные параллелепипеды.

Реализация мишеней

При попадании по мишени, запускается анимация падения и звук попадания. Через некоторое время мишень поднимется.

```
private void Update () {

    randomTime = Random.Range (minTime, maxTime);

    if (isHit == true)
    {
        if (routineStarted == false)
        {
            gameObject.GetComponent<Animation> ().Play("target_down");

            audioSource.GetComponent<AudioSource>().clip = downSound;
            audioSource.Play();

            StartCoroutine(DelayTimer());
            routineStarted = true;
        }
    }
}

1 reference
private IEnumerator DelayTimer () {
    yield return new WaitForSeconds(randomTime);
    gameObject.GetComponent<Animation> ().Play ("target_up");

    audioSource.GetComponent<AudioSource>().clip = upSound;
    audioSource.Play();

    isHit = false;
    routineStarted = false;
}
```



Мишень до и после попадания.

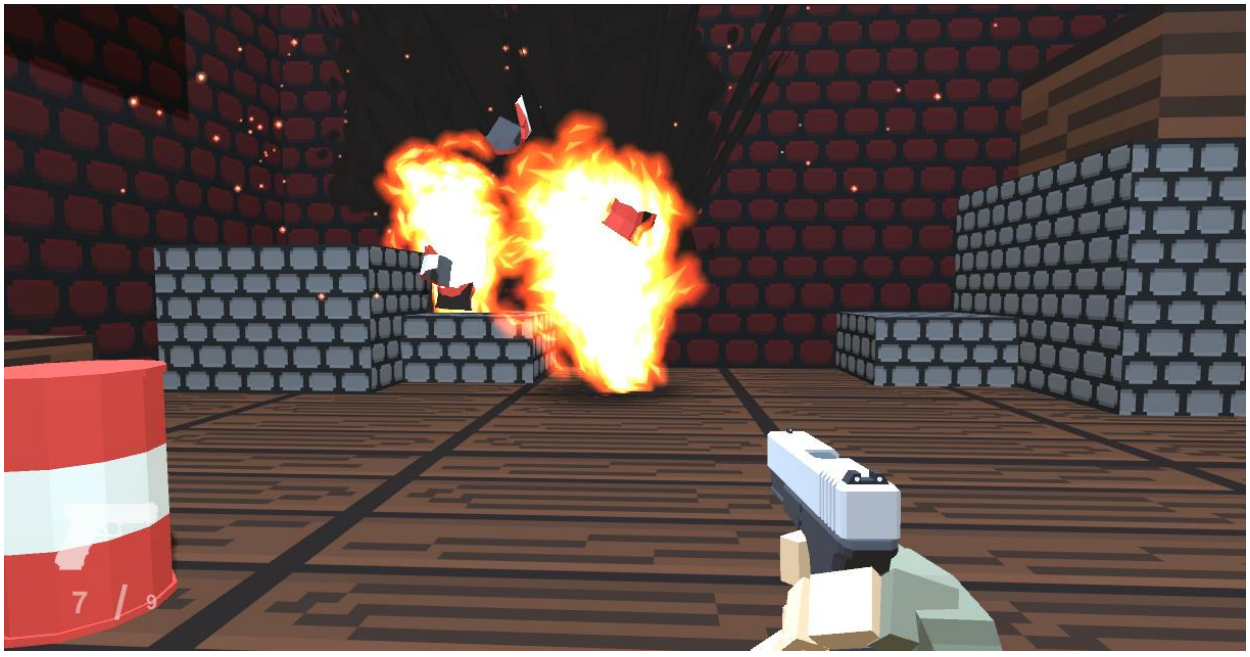
Проектирование помещений

Расставили игровые объекты по разным местам игровой сцены.

Реализация взрывов

При попадании по бочке происходит взрыв, реагирующий с другими игровыми объектами. Т.е. мишени падают, другие бочки взрываются. Проверка взаимодействия с объектами происходит в коллайдере-сфере, заданного радиуса.

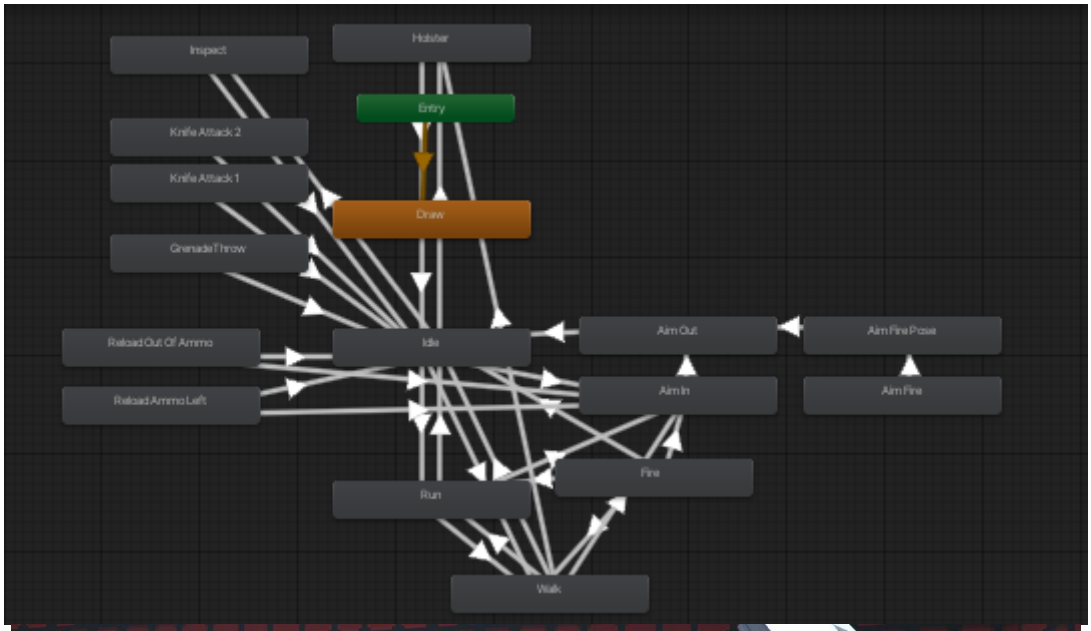
```
private IEnumerator Explode () {
    yield return new WaitForSeconds(randomTime);

    Instantiate (destroyedBarrelPrefab, transform.position,
                 transform.rotation);

    Vector3 explosionPos = transform.position;
    Collider[] colliders = Physics.OverlapSphere(explosionPos, explosionRadius);
    foreach (Collider hit in colliders) {
        Rigidbody rb = hit.GetComponent<Rigidbody> ();

        if (rb != null)
            rb.AddExplosionForce (explosionForce * 50, explosionPos, explosionRadius);

        if (hit.transform.tag == "ExplosiveBarrel")
        {
            hit.transform.gameObject.GetComponent<ExplosiveBarrelScript>().explode = true;
        }

        if (hit.transform.tag == "Target")
        {
            hit.transform.gameObject.GetComponent<TargetScript>().isHit = true;
        }
    }
```

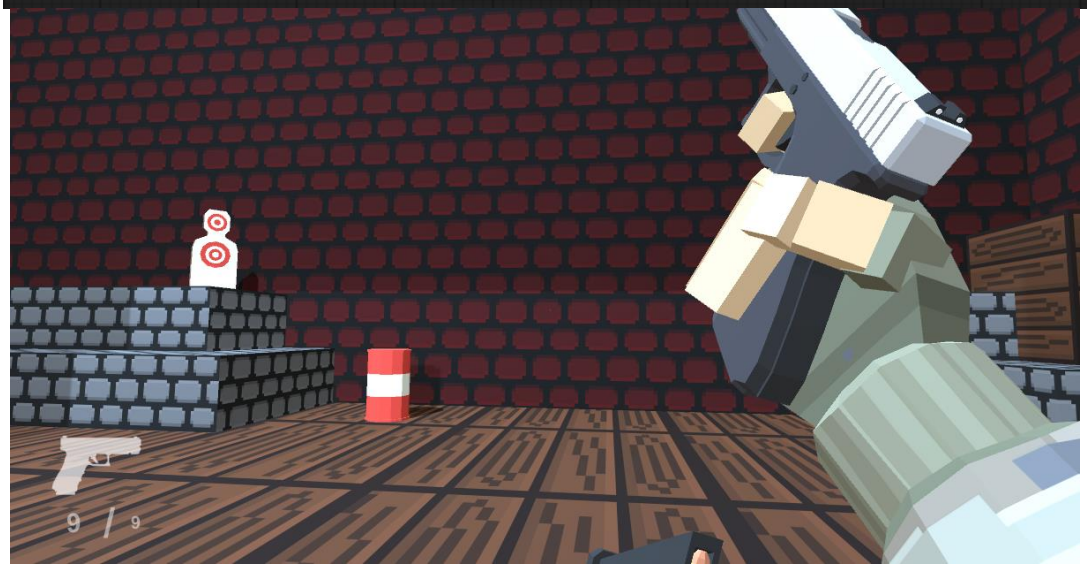


Красная бочка делает «БУМ»

Добавление и настройка анимации

Дерево анимации выглядит довольно запутанно, но имеет несколько групп. Сами анимации представлены на схеме как прямоугольники со скошенными краями.

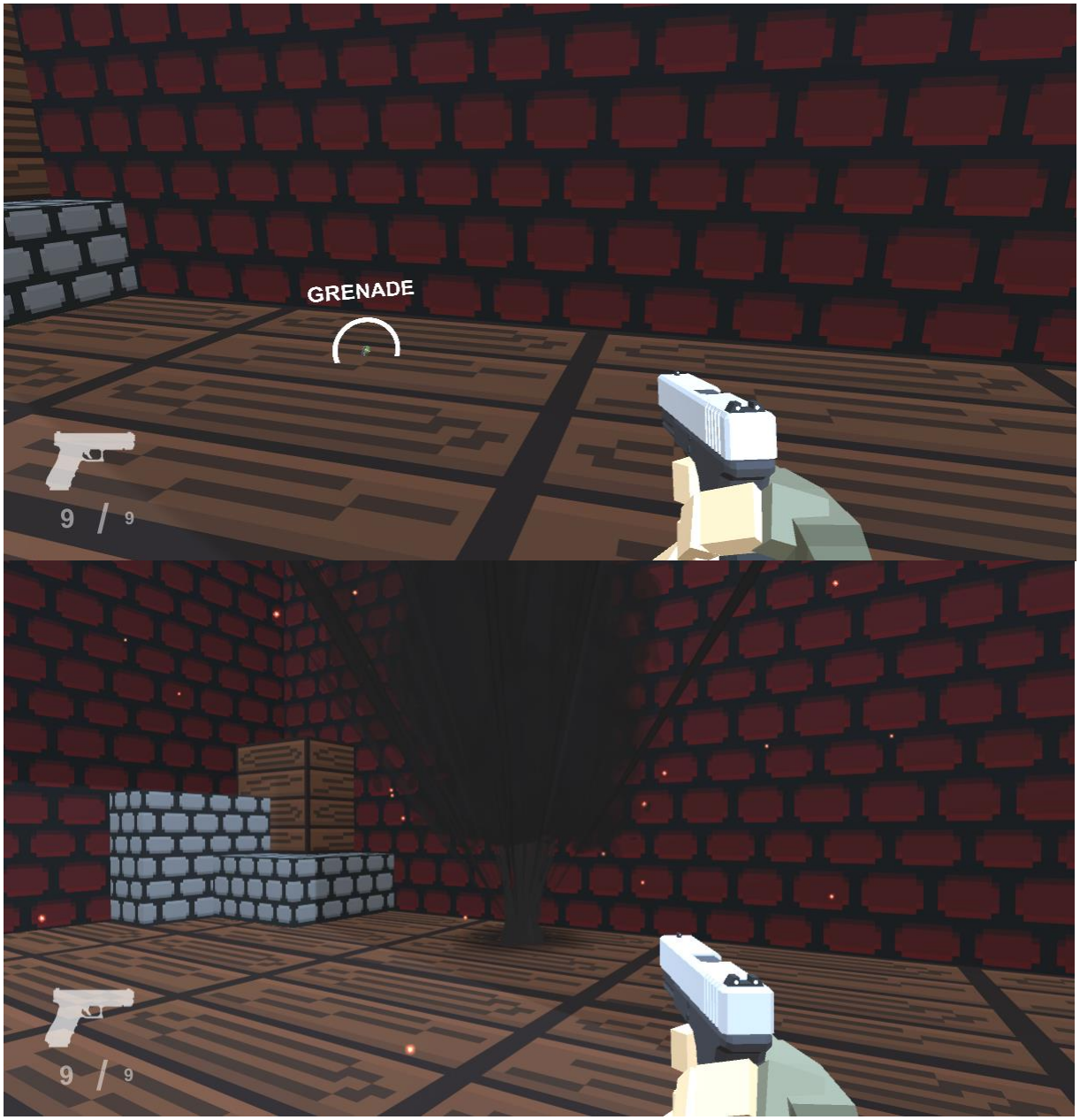




Один из кадров анимации перезарядки

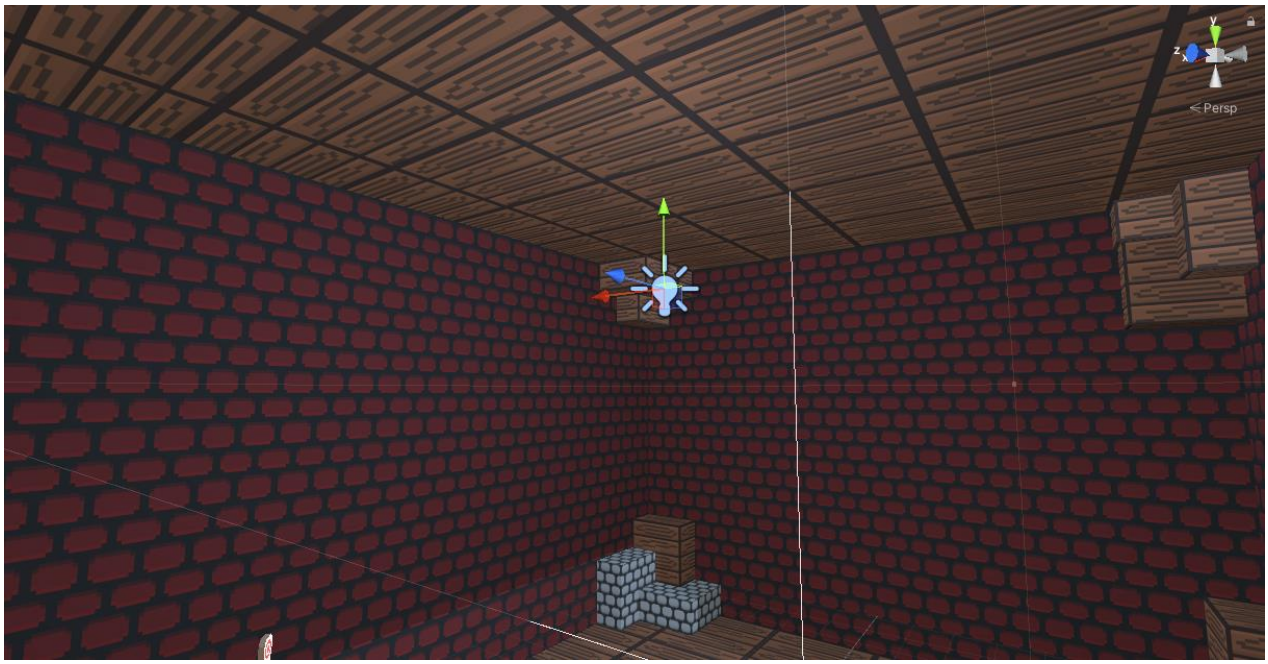
Реализация гранаты

Физика гранат схожа с физикой взрывающихся бочек и использует те-же функции.

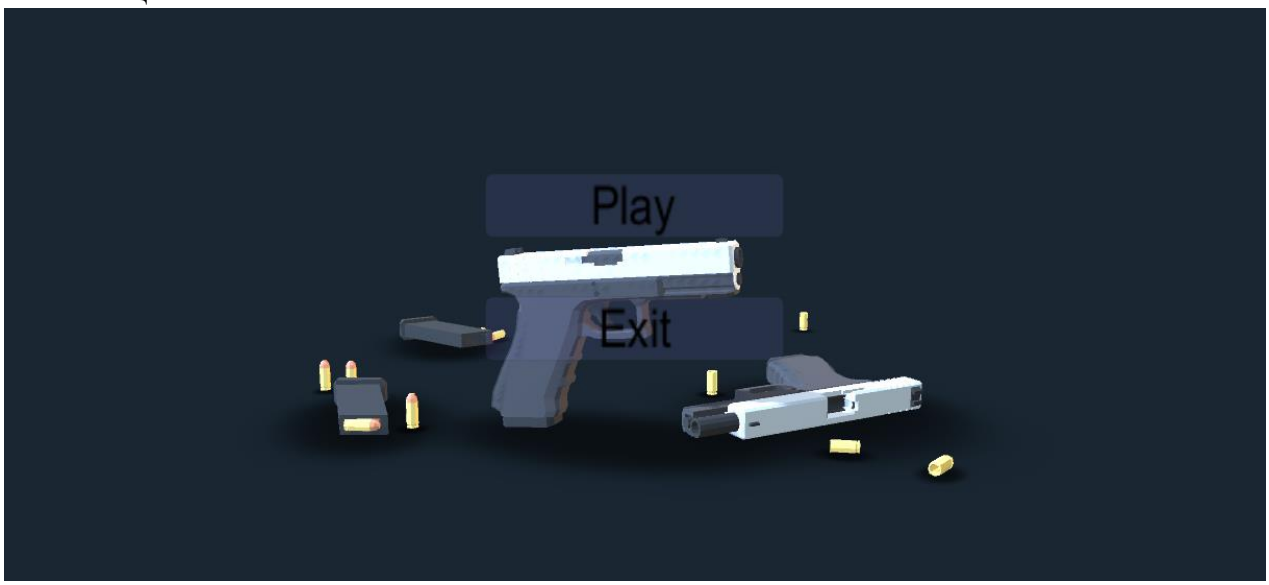


Граната!

Свет
Добавили и настроили простые источники света.



Главное меню
Реализация главного меню



Заключение

Проект завершен в сроки, план и задачи были выполнены. Участники справились со своими задачами и ролями, что и привело к успешному завершению проекта.

Литература

1. Unity в действии. Мультиплатформенная разработка на C#. 2-е межд. изд. — СПб.: Питер, 2019. — 352 с.
2. Unity и C#. Геймдев от идеи до реализации. 2-е изд. — СПб.: Питер, 2019. — 928 с.
3. Как создать внутриигровое меню в Unity [Электронный ресурс]. – Режим доступа: https://habr.com/ru/post/346370 (дата обращения: 17.12.2020).
4. Как создать внутриигровое меню в Unity [Электронный ресурс]. – Режим доступа: https://dtf.ru/gamedev/7227-orel-ili-reshka-sravnenie-unity-i-unreal-engine (дата обращения: 17.12.2020).
5. Trello — начало работы и скрытые фишки [Электронный ресурс]. – Режим доступа https://habr.com/ru/post/511446 (дата обращения: 17.12.2020).
6. Unity – Движение персонажа по вектору камеры (3D) [Электронный ресурс]. – Режим доступа https://pechenek.net/programming/c-sharp/unity-dvizhenie-personazha-po-vektoru-kameryi-3d/ (дата обращения: 17.12.2020).
7. Как запустить анимацию через скрипт C# [Электронный ресурс]. – Режим доступа https://ru.stackoverflow.com/questions/699004/unity-3d-5-5-1-Как-запустить-анимацию-через-скрипт-C (дата обращения: 17.12.2020).
8. В чем разница между Update и FixedUpdate в Unity, и стоит ли мне беспокоиться? [Электронный ресурс]. – Режим доступа https://qastack.ru/gamedev/73713/whats-the-difference-between-update-and-fixedupdate-in-unity-and-should-i-both (дата обращения: 17.12.2020).
9. Rigidbody [Электронный ресурс]. – Режим доступа https://docs.unity3d.com/ru/2019.4/Manual/class-Rigidbody.html (дата обращения: 17.12.2020).
10. Создание и уничтожение игровых объектов (GameObjects) [Электронный ресурс]. – Режим доступа https://docs.unity3d.com/ru/530/Manual/CreateDestroyObjects.html (дата обращения: 17.12.2020).