Государственное учреждение образования

«Лицей г.Новополоцка»

**Индивидуальный проект**

**секция «Информатика»**

**«Разработка игры жанра Tower Defense средствами C#»**

Автор работы:

Капустин Владислав Александрович,

10 класс ГУО «Лицей г.Новополоцка»,

Руководитель работы:

Коновалёнок Ольга Владимировна, учитель информатики, ГУО «Лицей г.Новополоцка»

2022г.

**Оглавление**

[Введение 4](#_Toc99448623)

[Работа над проектом «BaD» (Build and Defenсe) 5](#_Toc99448624)

[Заключение 8](#_Toc99448625)

[Список использованных источников 9](#_Toc99448626)

Приложение 10

# Введение

Сейчас трудно найти человека, который бы не играл в компьютерные игры. Есть противники компьютерных игр, есть ярые фанаты, есть пассивные игроки, но в любом случае равнодушных к играм нет. Игры уверенно вошли в нашу жизнь и уходить из нее не собираются. Ещё в 1938 году голландский философ Йохан Хёйзинга (1872−1945) в своём фундаментальном труде «Homo Ludens» («Человек играющий») предложил переосмыслить понятие культуры, исходя из наличия игрового элемента в культурной жизни. В связи со стремительным развитием цифровых технологий человек примеряет на себя все новые роли, в том числе и роли виртуального мира. [1]

Ключевой тренд 2021 года — кардинальное изменение отношения к играм, которые все меньше воспринимаются в качестве развлечения, помогающего скоротать вечерок, но все больше — как платформа для универсальной социализации, что может привести к появлению полноценных метавселенных (обширных виртуальных миров, которые дают игрокам широкие возможности для творчества и самореализации). [2]

Индустрия компьютерных игр – это сектор экономики, связанный с разработкой, продвижением и продажей игр. В неё входит большое количество специализаций, по которым работают тысячи специалистов по всему миру. Игровая индустрия тесно связана с производством центральных процессоров и других компонентов персональных компьютеров, так как игры зачастую требуют более высоких аппаратных мощностей, чем бизнес-приложения. В настоящее время игры являются движущей силой развития компьютерных технологий, которые впоследствии применяются и для других целей.

Для эффективного функционирования сферы игровой индустрии необходимы высококвалифицированные специалисты: программисты, геймдизайнеры, дизайнеры уровней, специалисты по звуку, копирайтеры, 3D- и 2D- художники и многие другие.

**Актуальность исследования** обусловлена тем, что игры на сегодняшний день чрезвычайно популярны. Широко распространено мнение, что видеоигры – приятный способ уйти от реальности, поэтому игроков часто обвиняют в жестокости, лени и пустой трате времени, эти споры бушуют много лет.

Цель: использование инструментов программы Unity для разработки игры в жанре Tower Defense.

Задачи: изучить возможности межплатформенной среды разработки компьютерных игр Unity; научиться создавать звуковые и анимационные эффекты; ознакомиться с языком программирования С#; создать игру в жанре Tower Defenсe.

Для реализации идеи игры была выбрана программа Unity. Данная программа позволяет создавать приложения, работающие на более чем 25 различных платформах, включающих ПК, игровые консоли, мобильные устройства, интернет-приложения и другие. Основными преимуществами Unity являются наличие визуальной среды разработки,  поддержки и модульной системы компонентов. При помощи этой программы написаны тысячи игр, приложений, математических моделей, которые охватывают множество платформ и жанров. При этом Unity используется как крупными разработчиками, так и отдельными разработчиками.

В качестве жанра игры выбран Tower Defense - он хорошо подходит для начинающих. Были определены цель и задачи проекта.

# Работа над проектом «BaD» (Build and Defenсe)

[3] Unity — межплатформенная среда разработки компьютерных игр, разработанная американской компанией Unity Technologies. Она позволяет создавать приложения, работающие на более чем 25 различных платформах, включающих персональные компьютеры, игровые консоли, мобильные устройства, интернет-приложения и другие.

Перед работой над проектом были выделены следующие этапы создания игры:

1.Разработка стратегии игры, сбор информации о жанре Tower Defense;

2.Построение локаций, объектов, противников;

3.Проектировка локаций с расположенными на них объектами (карта игры);

4.Написание скриптов (кода) на языке программирования C#;

5.Выставление и проверка игрового баланса;

6.Написание сюжета, сценария для игры;

7.Проектировке локаций с расположенными на них объектами, была создана карта игры;

8. Группировка всех составляющих игры, выполнение многократной проверки проекта и проработка финальных штрихов.

При принятии решения о создании игры было определено, что в качестве игрового жанра будет выбран Tower Defense.

Tower Defense (с [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) — «Башенная защита»), сокращённо TD – название жанра [компьютерных стратегических игр](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%B0%D1%82%D0%B5%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%B8%D0%B3%D1%80%D0%B0). Задача игрока в играх подобного жанра – расправиться с наступающими врагами, называемыми в некоторых играх «крипы» (от [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) [creep](https://ru.wiktionary.org/wiki/creep" \l "creep_(%D1%81%D1%83%D1%89%D0%B5%D1%81%D1%82%D0%B2%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5)) — «ползучая тварь»)[[1]](https://ru.wikipedia.org/wiki/Tower_Defense#cite_note-jay_flash-1), до того, как они пересекут карту, с помощью строительства башен, атакующих, когда враги проходят вблизи. Противники и башни обычно различаются по характеристикам и цене. Когда враги побеждены, игрок зарабатывает деньги или очки, которые используются для покупки или модернизации башен.

Подбор вида башен и их расположение является неотъемлемой стратегией игры. Обычно «ползучие твари» пробегают через подобие [лабиринта](https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D0%B1%D0%B8%D1%80%D0%B8%D0%BD%D1%82), что даёт игроку возможность стратегического размещения башен, но также существуют известные версии игры, названные линейными TD, где вместо лабиринтов используются прямые пути. В некоторых версиях игрок может сам выстраивать лабиринт из башен и блоков.

Для нашего проекта мы выбираем обычную версию жанра с модернизируемой башней. Радиус действия башни, уровень и мощность увеличиваются одновременно. Проект мы называем «BaD» (Build and Defence). Так как наша игра основывается на строительстве и защите от врагов.

Этапы создания локаций представлены в приложении 1 рис 1,2,3,4.

Первым делом мы прописали противников. Противники должны передвигаться по игровому полю и исчезать, если достигли финальной точки. Если противникам наносят урон башни, то они должны исчезать в конце при достижении отрицательного количества здоровья.

Далее нам был необходим спавнер, чтобы противники появлялись один раз в определенное время и проходили свой путь до финальной сцены игры. Для защиты нам нужна башня, чтобы она выслеживала противников и выстраивала защиту. Защитой будут выступать ядра. Они наносят урон противникам. В результате противостояния башни и противника у противника уменьшается здоровье до нуля, и он исчезает.

Оборона происходит по следующему правилу: башни можно ставить только на разрешенных территориях.

Для построения башен необходимы будут ресурсы. Без ресурсов пропадает интерес в игре.

В начале игры каждому игроку будут предоставлены начальные ресурсы, но в процесс игры их нужно постоянно пополнять. Башни за уничтожение противников будут зарабатывать ресурсы для игрока, чтобы строить башни и улучшать свою защиту. Игрок будет видеть количество доступных ему ресурсов для того, чтобы он понимал, что он может построить, а что нет.

Если противник проходит свой путь и не погибает, то игрок проигрывает.

Для победы нужно прожить некоторое количество времени, одновременно обороняясь. Добавим таймер, чтобы регулировать количество противников на игровом поле. Мы посчитали, что игрок не должен будет видеть таймер так как это создаст большее напряжение. Игрок будет задаваться вопросом: «Сколько мне осталось продержаться?». Текст в оформлении ярко желтый, чтобы игрок быстро распознавал информацию. Когда игрок выигрывает или проигрывает, его переводят на другой экран. Мы написали скрипт для перехода между сценами.

В игре есть царь. Он присылает различные послания, помогающие разобраться в игре. Для сцен обучения и послания от царя мы выбрали цвет пергамента, изменили шрифт на прописной и цвет текста изменен на черный.

Для сцен титров и предыстории был выбран черный фон с буквами желтого цвета на нем. Источником вдохновения в данном случае послужили «Звездные войны» и их дорожки повествования.

Для того чтобы наша игра была интересней нужно выставить в ней баланс.

Если победить двоих противников, то можно построить башню, но так исчезает интерес к игре. Интереснее, когда за трех противников выдается 60 единиц. Чтобы через двух противников у вас стало 100 единиц, и вы смогли себе позволить две башни.

Противники в нашей игре не будут использовать готовый скрипт для их волн (непрерывный поток) появления противников. Мы посчитаем с помощью математики, когда лучше всего пускать противников и какое время им выставлять в спавнерах. Допустим, выставляем первое появление на 3 с, на 6 с и на 9 с. Таким образом, будет непрерывный поток противников с возможностью отдыха для башен.

В нашей игре мы пытаемся реализовать придуманную нами историю про королевство под угрозой. Когда орда (наши противники) соберут 3 камня (кристалла), то весь мир игры исчезнет.

Мы выступаем в роли новичка инженера, вступившего в отряд рабочих в гильдии строителей. Его назначают инженером на первом его задании (обучении) на охрану одного из камней. Камень находится на поляне. Орда отстает от этого камня и идет к следующему, если игрок выполнит свое поручение. После этого игрока ждет повышение и награда, различные улучшения от гильдии. Если игрок не проведет защиту, то его ожидает суровое наказание от самого царя.

Следующее поручение защита кристалла в деревне. Здесь появляются новые враги, умеющие плавать и непредсказуемо двигаться. Появляется новый тип постройки поддоны и баллисты, орда отстает от этого камня и идет к следующему.

И последнее самое важное задание по защите городской церкви, содержащий последний камень. На этом этапе появляются новые враги, умеющие перепрыгивать через препятствия. Также мы создаем новый тип постройки: навесы и башни придворных магов.

Если игрок выполняет свое поручение, то в финале игры получает различные награды, почести, звание высшего инженера в королевстве.

Если игрок проигрывает, то получает суровое наказание от царя.

Вторым этапом постройки стала проектировка локаций с расположенными на них объектами. Мы добавили объекты на сцену: спавнеры противников, таймер со временем на прохождение.

Группировка всех составлявших игры стало следующим этапом разработки.

Выставим скрипты в соответствующие им объекты. Можно создать префаб (шаблон) одного объекта и не нужно будет каждый объект изменять отдельно, можно просто изменить.

Были разработаны дополнительные материалы. Для базовых противников (крипов) была создана капсула с двумя сферами и кубом. Капсулой я показал простое тело для более простой работы коллайдера (местом попадания)

Сферами же мы показали глаза, перекрасив их в красный цвет, так мы изобразили опасность. Внутри наших сфер(глаз) имеются сферы поменьше. сферы поменьше – это зрачки. Они добавлены для детализации нашего противника. (приложение1 рис.5)

Для первых башен мы выбрали пушечную башню применив ее как стандартную. Все характеристики башни средние, что можно увидеть при тесте игры. Баллиста медленная, но смертоносная с большим радиусом атаки. Башня магов самая быстрая в игре, но имеет наименьший радиус и урон. (приложение1 рис.6,7)

Одной из локаций стала деревушка. (приложение 1 рис 8)

Для третьей локации в игре был построен город. (приложение 1 рис 9,10)

[3] Приводим фрагмент кода для баллисты и башни магов (приложение 2). С остальными скриптами можно ознакомиться по ссылке: <https://disk.yandex.ru/d/hmQmwjVRq5Z-YQ> . Для запуска необходимо запустить файл BaD.exe

# Заключение

В ходе исследования нами выполнены все поставленные задачи: создание игры «BaD» (Build and Defence); создание звуковых и анимационных эффектов. Поставленной цели в своей работе мы достигли. Нам удалось, используя инструменты программы Unity разработать игру в жанре Tower Defense.

На сегодняшний день компьютерные игры достаточно прочно вошли в жизнь каждого современного человека. Хотим мы того или нет, но это свершившийся факт. Несмотря на это, споры вокруг компьютерных игр не утихают: сторонники говорят о развивающих качествах игр, противники – о вреде для психики, о возникновении компьютерной зависимости. Каждый из них по-своему прав. Суть этого спора кроется в афоризме древнего лекаря Парацельса: «Все есть лекарство, и все есть яд. Только доза делает вещество ядом или лекарством». Что же касается компьютерных игр, то все зависит от выбора игры и от места, которое игра занимает в жизни человека.

Пандемия коронавируса и повсеместный карантин обнаружили очевидную проблему: человечество оказалось не готово обеспечить бесперебойное функционирование ни спортивной, ни развлекательной индустрии — все очень быстро остановилось на неопределенный срок. И только игровая индустрия продолжает чувствовать себя сравнительно комфортно, что в очередной раз доказало: за виртуальной реальностью будущее. Исходя из этого и выходит актуальность разработок и продвижение новый игр.

# Список использованных источников

1. Homo Ludens; Статьи по истории культуры. / Пер., сост. и Х 35 вступ. ст. Д.В. Сильвестрова; Коммент. Д. Э. Харитоновича -М.: Прогресс - Традиция, 1997. - 416 с. ISBN 5-89493-010-3
2. https://trends.rbc.ru/trends/industry/cmrm/61c995ec9a79471845177afb
3. Тайнан,С. Геймдизайн. Рецепты успеха лучших компьютерных игр от Super Mario и Doom до Assassin’c Creed и дальше / Пер. с анг. М. Панина, А. Поповой. – СПб.: Питер, 2020. – 448 с.:ил.
4. Unity (игровой движок) [Электронный ресурс] // Википедия (свободная энциклопедия). – Режим доступа: <https://ru.wikipedia.org/wiki/Unity>. – Дата доступа:19.01.2022.
5. Мюллер, Д.П. С# для чайников/Д.П. Мюллер, Б.Семпф, Ч. Сфер. – Изд. Диалектика-Вильямс, 2019. – 608 с.

Приложение

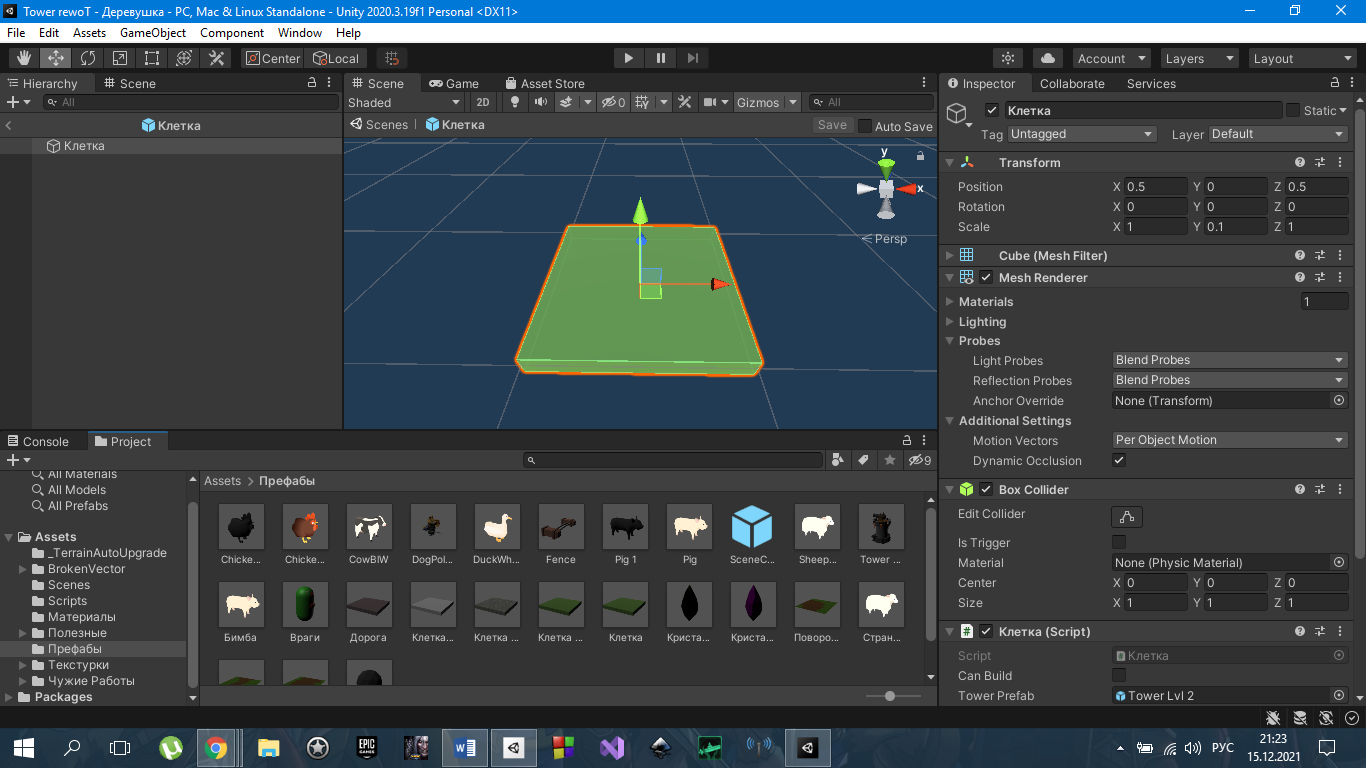


Рисунок 1 – Клетка с единичными размерами

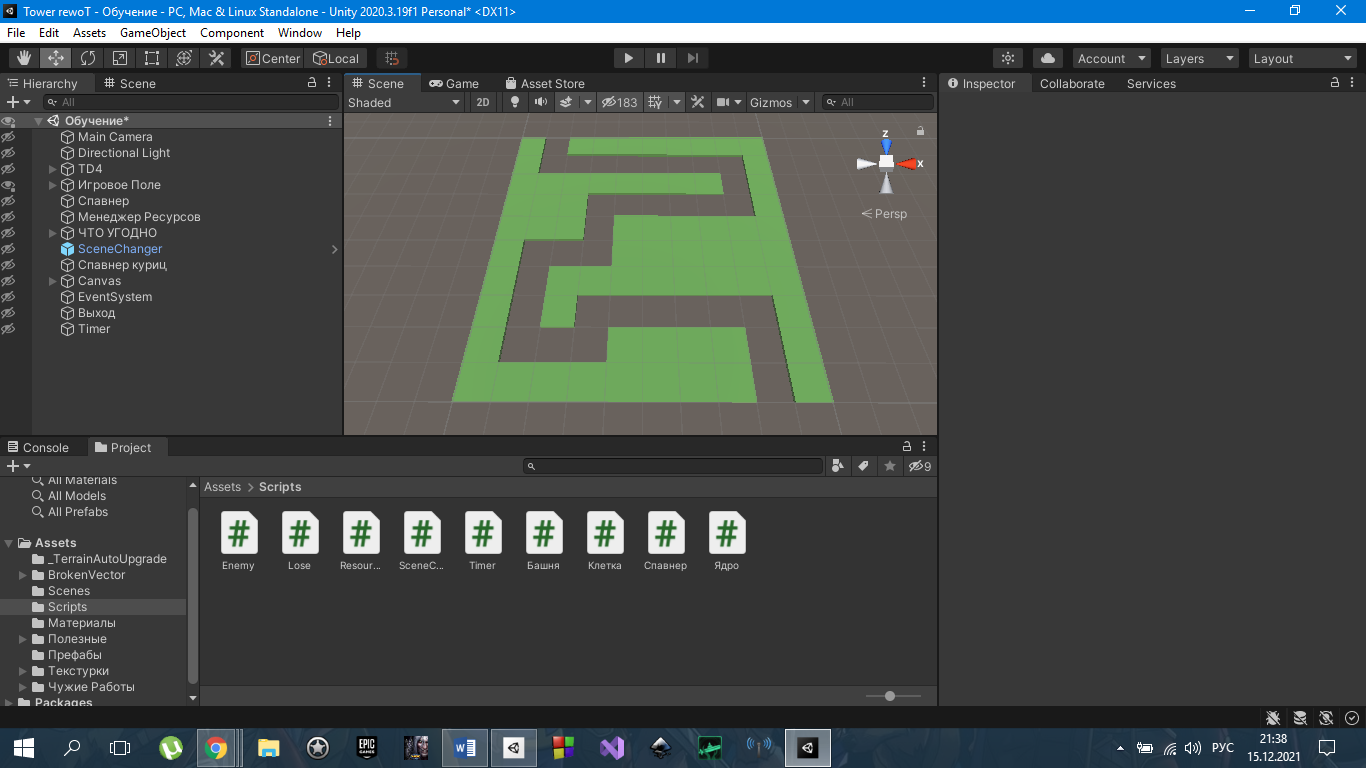


Рисунок 2 – Дорога на игровом поле

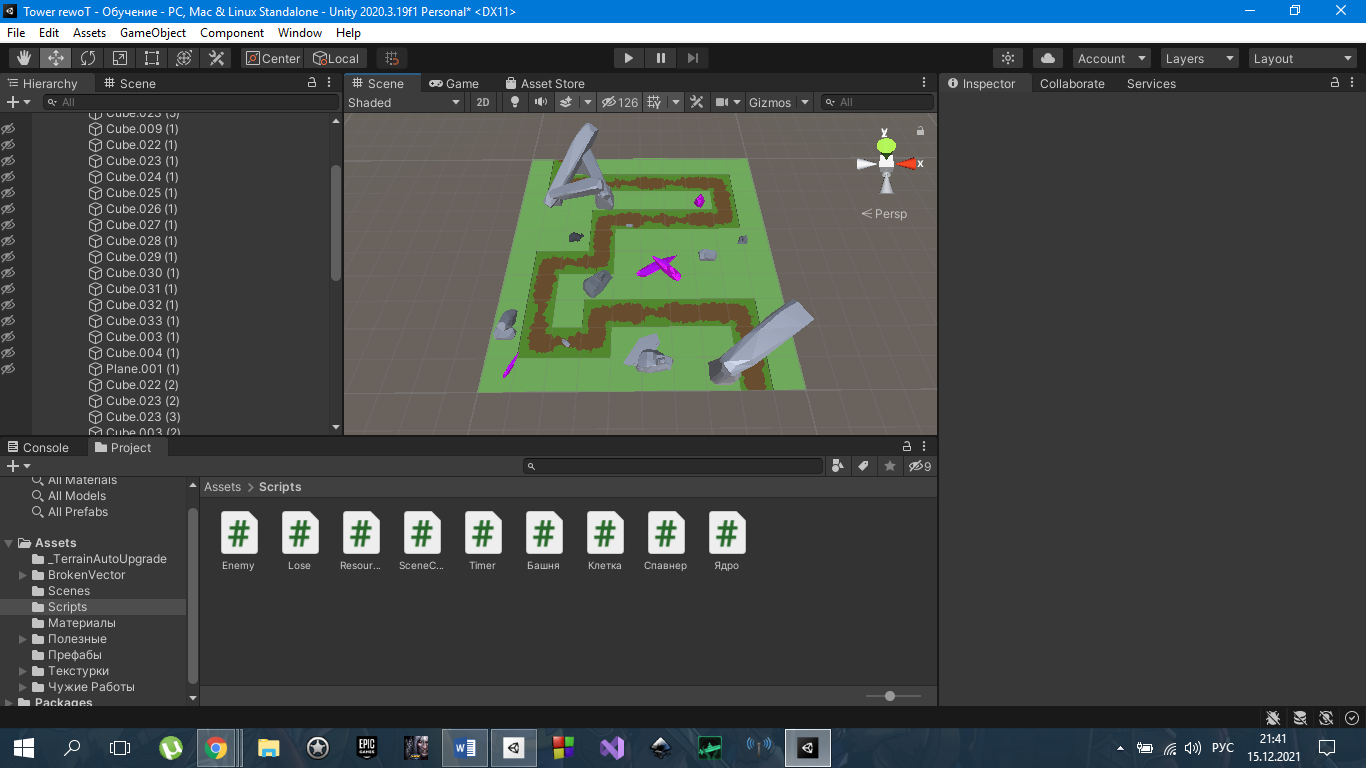


Рисунок 3 – Игровое поле с объектами

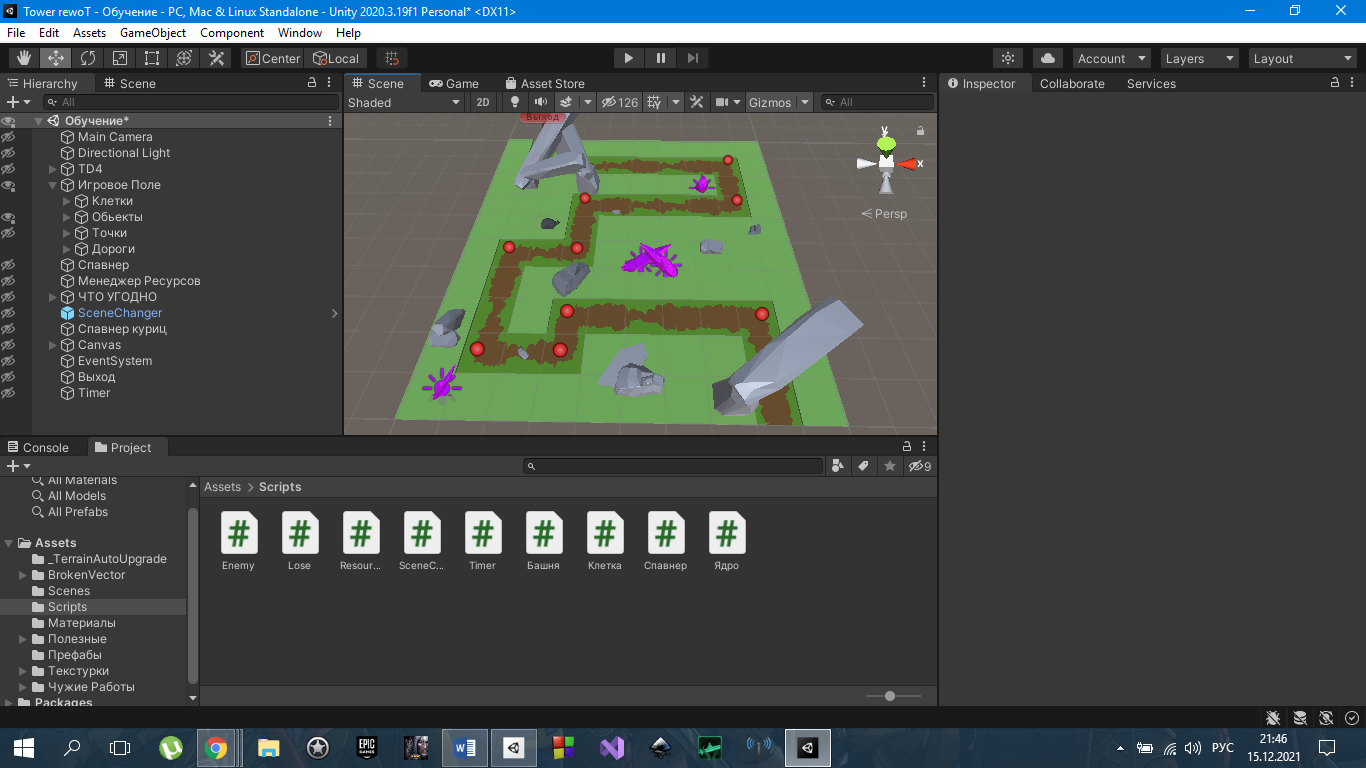


Рисунок 4 – Игровое поле с красными точками

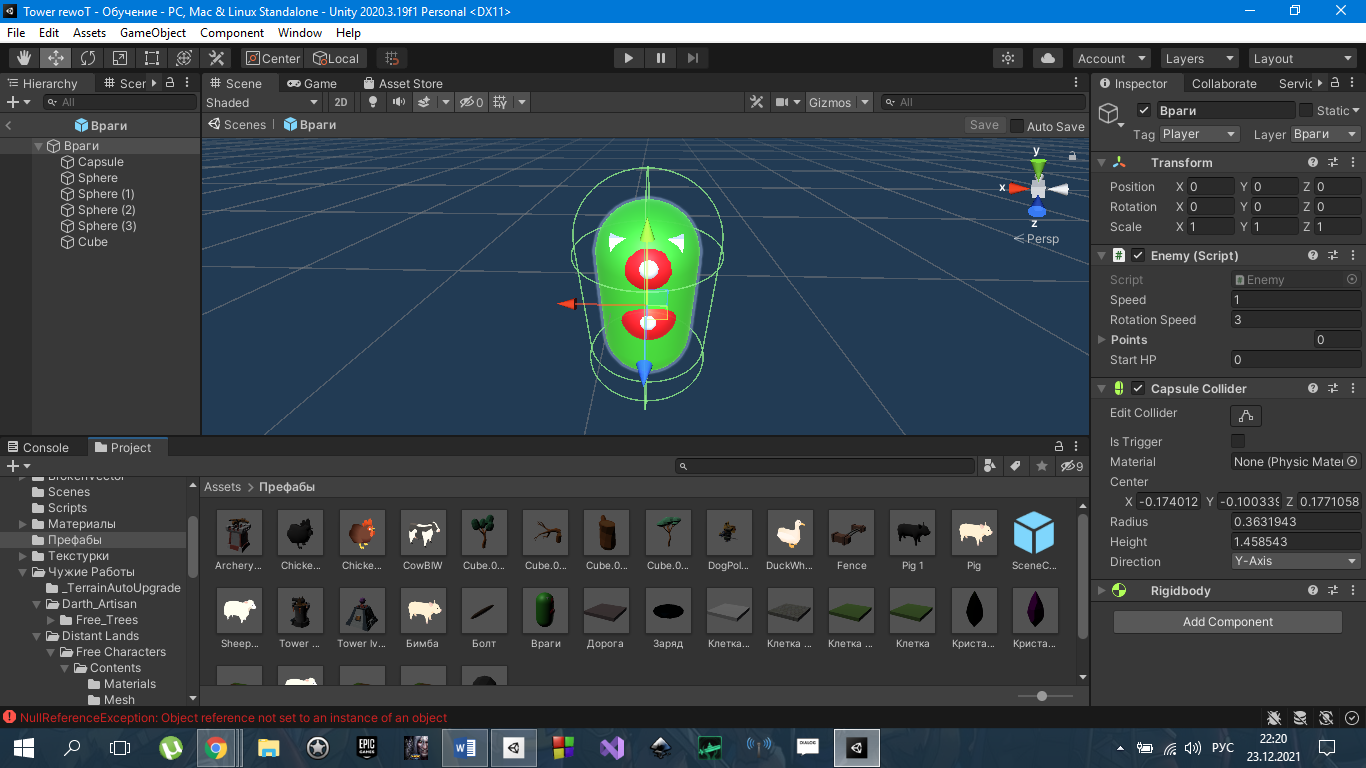


Рисунок 5 – Проектирование противника

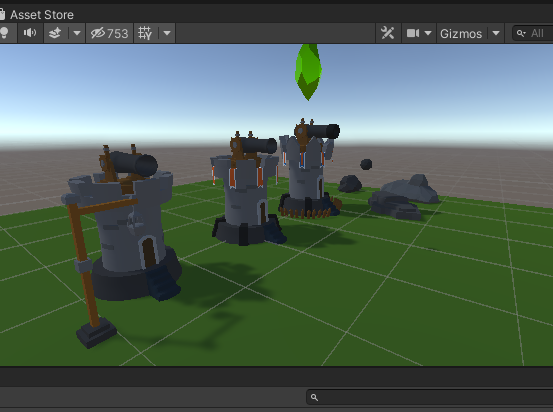


Рисунок 6 – Новые виды башен с улучшениями и новые локации



Рисунок 7 – Башни с баллистами

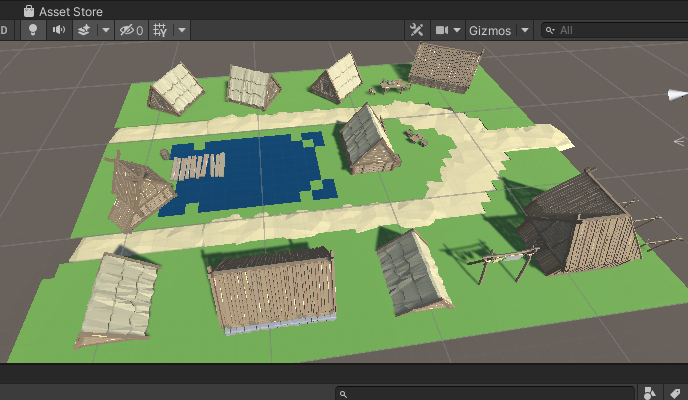


Рисунок 8 – Деревня

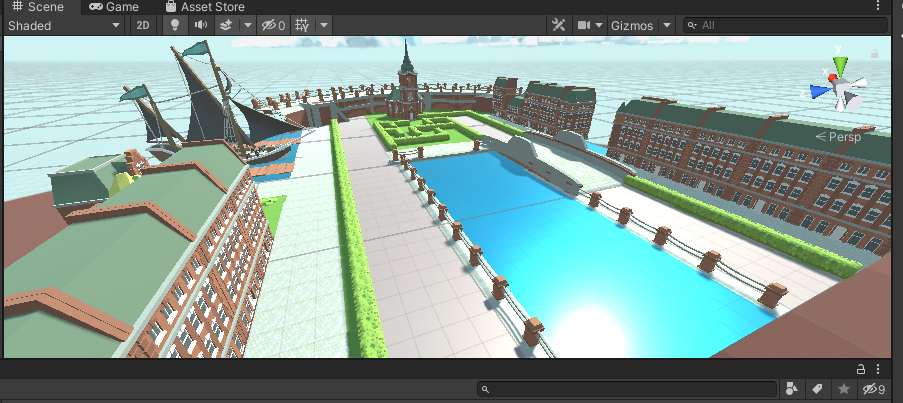


Рисунок 9 – Город

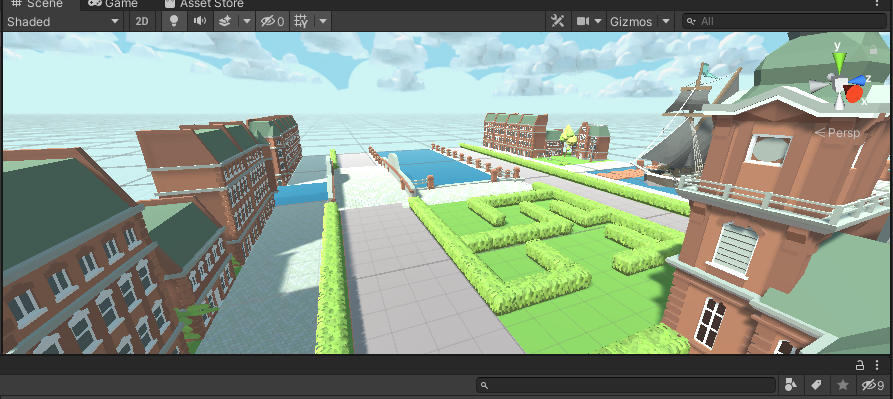


Рисунок 10 – Улицы города

Приложение 2

Баллиста

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Баллиста : MonoBehaviour

{

public Transform BulletPrefab;

public float FireDelay,Radius,Damage;

public LayerMask EnemyLayer;

private float firetimer;

private Transform Враги, Cylinder, ТочкаВыстрела;

void Start()

{ firetimer = FireDelay;

Cylinder = transform.GetChild(0);

ТочкаВыстрела = Cylinder.GetChild(0);

}

void Update()

{ if (firetimer > 0) firetimer -= Time.deltaTime;

else if (Враги)

Fire();

if (Враги)

{ Vector3 lookAt = Враги.position;

lookAt.y = Cylinder.position.y;

Cylinder.rotation = Quaternion.LookRotation(Cylinder.position - lookAt);

if (Vector3.Distance(transform.position, Враги.position) > Radius)

Враги = null;

}

else if (Враги == null)

{ Collider[] colls = Physics.OverlapSphere(transform.position, Radius, EnemyLayer);

if (colls.Length > 0)

Враги = colls[0].transform.GetChild(1);

}

}

void Fire()

{

Transform Ядро = Instantiate(BulletPrefab, ТочкаВыстрела.position, Quaternion.identity);

Ядро.LookAt(Враги);

Ядро.GetComponent<Ядро>().Damage = Damage;

firetimer = FireDelay;

}

}

Башня магов

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Башнямагов : MonoBehaviour

{

public Transform BulletPrefab;

public float FireDelay,Radius,Damage;

public LayerMask EnemyLayer;

private float firetimer;

private Transform Враги, Sphere, ТочкаВыстрела;

void Start()

{ firetimer = FireDelay; Sphere = transform.GetChild(0); ТочкаВыстрела = Sphere.GetChild(0);

}

void Update()

{ if (firetimer > 0) firetimer -= Time.deltaTime;

else if (Враги)

Fire();

if (Враги)

{ Vector3 lookAt = Враги.position;

lookAt.y = Sphere.position.y;

Sphere.rotation = Quaternion.LookRotation(Sphere.position - lookAt);

if (Vector3.Distance(transform.position, Враги.position) > Radius)

Враги = null;

}

else if (Враги == null)

{ Collider[] colls = Physics.OverlapSphere(transform.position, Radius, EnemyLayer);

if (colls.Length > 0)

Враги = colls[0].transform.GetChild(1);

} } void Fire() {

Transform Ядро = Instantiate(BulletPrefab, ТочкаВыстрела.position, Quaternion.identity);

Ядро.LookAt(Враги);

Ядро.GetComponent<Ядро>().Damage = Damage;

firetimer = FireDelay; } }