**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ  
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук  
Образовательной программы бакалавриата «Программная инженерия»

|  |  |
| --- | --- |
| СОГЛАСОВАНО  Научный руководитель, доцент департамента  программной инженерии  факультета компьютерных наук, канд. техн. наук    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.Л. Макаров  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г. | УТВЕРЖДАЮ  Академический руководитель  ОП «Программная инженерия» профессор департамента программной инженерии, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В. В. Шилов «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г. |

|  |  |
| --- | --- |
| *Подп. и дата* |  |
| *Инв. № дубл.* |  |
| *Взам. инв. №* |  |
| *Подп. и дата* |  |
| *Инв. № подл* |  |

**2D-игра “Adventure”  
 на Unity**

**Пояснительная записка**

**ЛИСТ УТВЕРЖДЕНИЯ**

**RU.17701729.****04.01-01 81 01-1-ЛУ**

Исполнитель:  
студент группы БПИ198  
\_\_\_\_\_\_\_\_\_\_ / Здор А. М. /  
«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

**Москва 2020**

УТВЕРЖДЕН

RU.17701729.04.01-01 81 01-1-ЛУ

|  |  |
| --- | --- |
| Подп. и дата |  |
| Инв. № дубл. |  |
| Взам. инв. № |  |
| Подп. и дата |  |
| Инв. № подл |  |

**2D-игра “Adventure”  
 на Unity**

**Пояснительная записка**

**RU.17701729.04.01-01 81 01-1**

**Листов 36**

**Москва 2020**

**Содержание**

[1. ВВЕДЕНИЕ 3](#_Toc40800002)

[1.1 Наименование программы 3](#_Toc40800003)

[1.2 Документ, на основании которого ведется разработка 3](#_Toc40800004)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 4](#_Toc40800005)

[2.1 Назначение программы 4](#_Toc40800006)

[2.2 Краткая характеристика области применения 4](#_Toc40800007)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 5](#_Toc40800008)

[3.1 Постановка задачи на разработку программы 5](#_Toc40800009)

[3.2 Описание алгоритма и функционирования программы 5](#_Toc40800010)

[3.3 Описание и обоснование выбора метода организации входных и выходных данных. 13](#_Toc40800011)

[3.4 Описание и обоснование выбора состава технических и программных средств. 14](#_Toc40800012)

[4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 15](#_Toc40800013)

[4.1 Предполагаемая потребность 15](#_Toc40800014)

[4.2 Технико-экономические показатели, обосновывающие преимущество выбранного варианта технического решения. 15](#_Toc40800015)

[5. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ 16](#_Toc40800016)

[ПРИЛОЖЕНИЕ 1 ТЕРМИНОЛОГИЯ 18](#_Toc40800017)

[ПРИЛОЖЕНИЕ 2 ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ 19](#_Toc40800018)

[ПРИЛОЖЕНИЕ 3 ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ МЕТОДОВ И СВОЙСТВ 21](#_Toc40800019)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 36](#_Toc40800020)

1. ВВЕДЕНИЕ
   1. Наименование программы

Наименование темы разработки – «2D-игра “Adventure” на Unity»

Наименование темы разработки на английском языке – «Adventure 2D Unity Game»

Краткое наименование программы – «Adventure».

* 1. Документ, на основании которого ведется разработка

Программа разрабатывается на основании приказа Национального исследовательского университета "Высшая школа экономики" декана **факультета компьютерных наук И.В. Аржанцева "Об утверждении тем, руководителей курсовых работ студентов образовательной программы «Программная инженерия» факультета компьютерных наук" № 2.3-02/1112-04 от 11.12.2019.**

1. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ
   1. Назначение программы
      1. Функциональное назначение

Программа представляет собой 2D игру жанра “Runner” и несет развлекательную функцию. Ее задача – предоставить пользователю (игроку) возможность прохождения бесконечной полосы препятствий с целью установления новых рекордов и сбора монет для последующей прокачки персонажа. Вследствие того, что игрок имеет мало времени на принятие решений, игра развивает реакцию.

* + 1. Эксплуатационное назначение

Программа может эксплуатироваться на любом PC на OC Windows, который соответствует «Требованиям к составу и параметрам технических средств».

Потребителем является любой пользователь PC, без специальной подготовки, который хочет развлечься, скоротать время и любит игры жанра “Runner”

* 1. Краткая характеристика области применения

Программа будет использоваться в индустрии развлечений, для проведения времени, отдыха, снятия эмоционального напряжения и стресса, развития реакции и умения быстро принимать решения.

1. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ
   1. Постановка задачи на разработку программы

Программа должна реализовывать следующие функции:

* Прокачка персонажа.
* Бесконечная случайная генерация ландшафта из готовых кусков.
* Сохранение информации о пользователе:
  + - * 1. Количество заработанного золота.
        2. Рекорд (дистанция лучшего пробега).
        3. Прокаченные способности.
* Возможность начать забег.
* Управление персонажем с помощью клавиатуры.
* Генерация монет и бонусов.
* Генерация ловушек и враждебных существ.
* Возможность регулировать громкость.
* Отображение информации об игроке во время забега:

1. Текущая дистанция.
2. Количество золота собранного в данном забеге.
3. Количество жизней персонажа.

* Возможность поставить паузу во время забега.
* Следование камеры за игроком во время забега
* Смерть персонажа при выходе за границы уровня
* Смерть персонажа при окончании здоровья
* Работа магнита, если он прокачан
  1. Описание алгоритма и функционирования программы
     1. Описание игрового процесса

Чтобы передвигаться игрок должен использовать стрелки на клавиатуре. Во время забега необходимо перемещаться строго по каменным платформам, в противном случае игрок упадет и забег закончится. Во время бега игрок может собирать монеты, жизни и открывать сундуки в которых лежат монеты. Игрок должен стараться обходить шипы, привидений, уворачиваться от лазеров, снарядов пушек. При соприкосновении с снарядом пушки, лазером, шипом, призраком игрок теряет одну жизнь и на 1 секунду становится неуязвимым (неуязвимость сопровождается эффектом мигания главного героя). Если вы упали или вас догнал демон который гонится за вами, игра заканчивается в независимости от количества ваших жизней. Полученные деньги игрок может потратить на прокачку. Есть два типа прокачки: увеличение максимального количества жизней и увеличение дальности действия магнита.

* + 1. Описание функционирования различных меню в игре

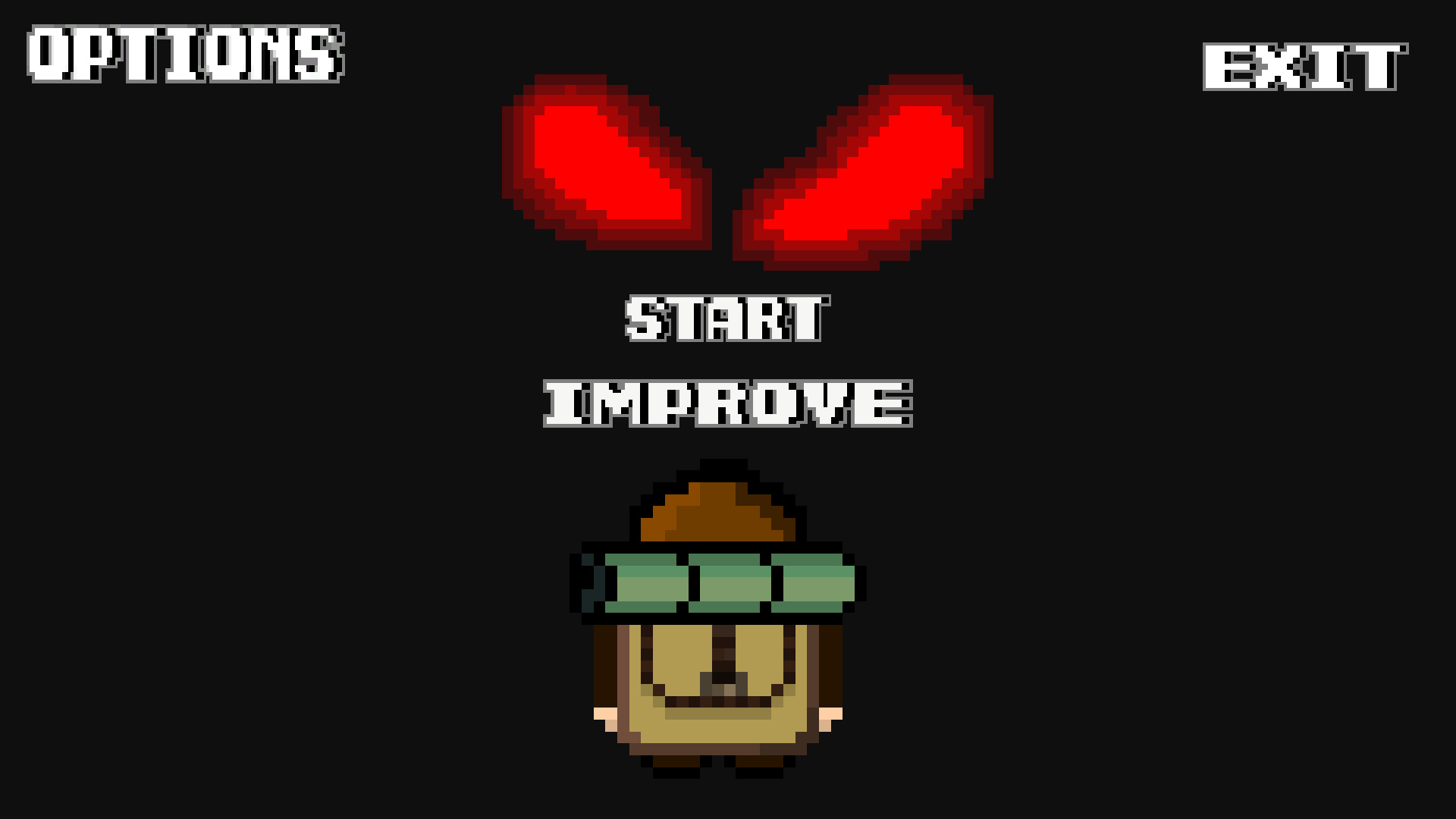
При запуске игры игрок видит главное меню (рис. 1). Оно выглядит следующим образом: 

Рисунок 1. Главное меню.

1. START (начать новый забег);
2. IMPROVE (в этом пункте игрок может потратить деньги на улучшения);
3. OPTIONS (здесь игрок может изменить настройки звука);
4. EXIT (выход из игры);

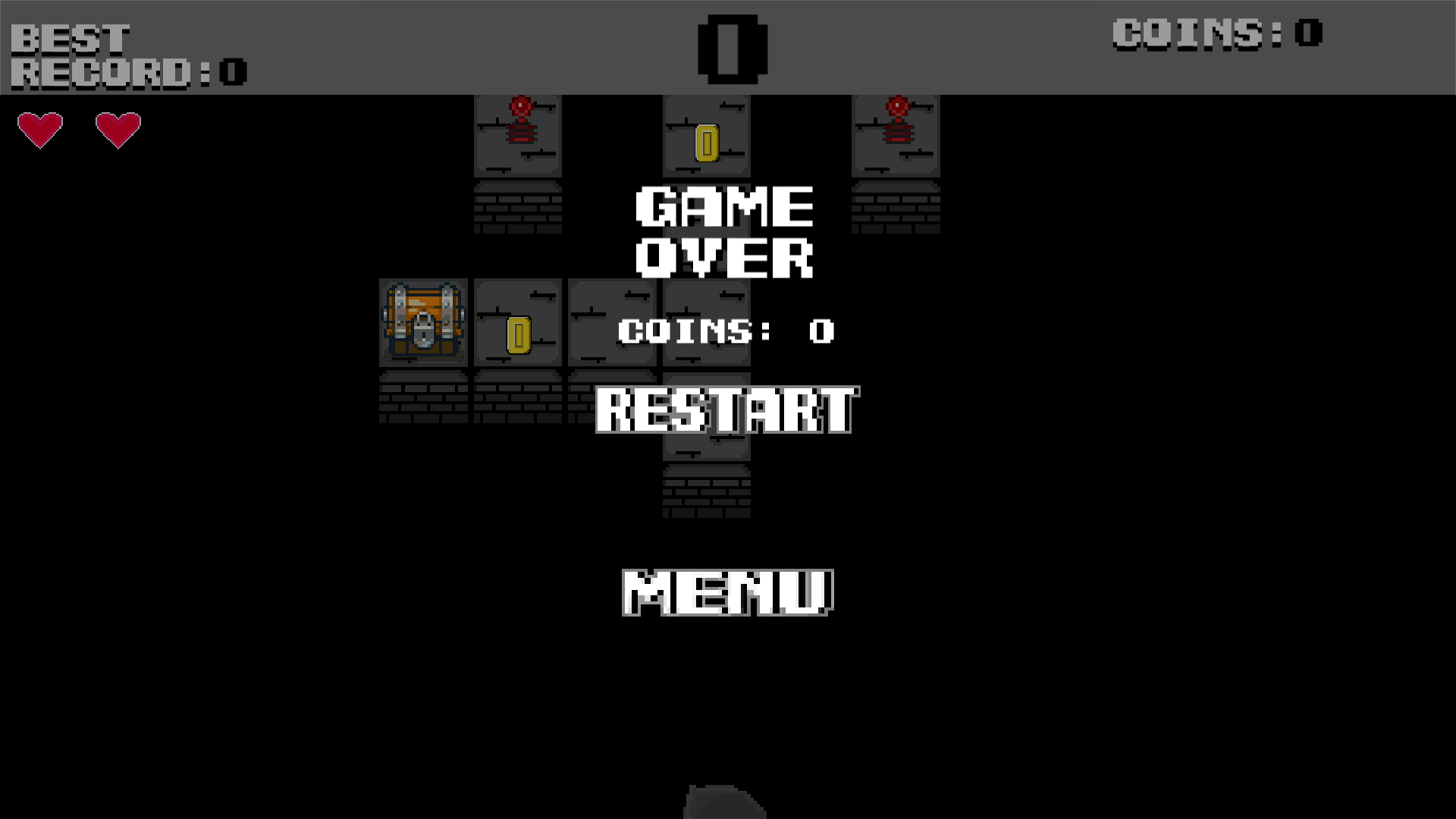
Меню настроек выглядит следующим образом: 

Рисунок 2. Меню настроек.

1. Master Volume (слайдер общей громкости);
2. Sounds Volume (слайдер громкости звуков);
3. Music Volume (слайдер громкости музыки);
4. Back (возвращение к главному меню игры);

Во время игры по нажатию клавиши Escape игра останавливается и перед игроком появляется меню паузы:Рисунок 3. Меню паузы.

1. Resume (продолжить игру, также можно повторно нажать Escape);
2. Options (меню настроек звука, выглядит также как и в главном меню рис.2);
3. Menu (выход в главное меню);

После смерти игрока появляется меню конца игры (рис. 4). Оно выглядит следующим образом: Рисунок 4. Меню конца игры.

1. Restart (перезапуск забега);
2. Menu (главное меню);
   * 1. Описание функционирования персонажа

Игрок состоит из BoxCollider2D, RigitBody2D в режиме Kinematic и дочернего объекта Magnet. Также на игрока повешены следующие скрипты:

1. PlayerController отвечает за управление персонажем, его передвижением и обработкой смерти;
2. Player\_money отвечает за счет монет собранных в раунде;
3. Player\_score отвечает за счет очков в раунде;
4. HealthSystem отвечает за обработку получения урона и отображение жизней;
5. Falling отвечает за анимацию падения персонажа;

При нажатии одной из стрелок скрипт PlayerController высчитывает новое положение персонажа в которое необходимо переместиться. После этого начинается движение. Во время движения игрок не может отдавать команды персонажу. Как только конечная точка достигнута, персонаж переводится в положение ожидания команды от игрока. Благодаря этому реализовано перемещение по клеткам игрового поля. Также при нажатии стрелок персонаж определяет направление движения и меняется соответствующая анимации (GoUp, GoDown, GoLeft, GoRight). При перемещении проверяется, что персонаж перешёл на клетку пола (посредствам взаимодействия коллайдера пола и игрока), если игрок не попал на клетку пола, то проигрывается метод Play() из скрипта Falling, отвечающий за анимацию падения. При получении урона игрок теряет одну жизнь и становится неуязвимым на 1 секунду. Неуязвимость сопровождается эффектом моргания персонажа, который реализован следующим образом. В методе Update() производится счет таймера и вычисление значения альфа канала спрайта персонажа как функция синуса деленная на 2 и сдвинутая вверх на 1. Таким образом функция принимает значения от 1 до 0. Затем в функцию подставляется значение таймера до множенное на 4\*PI.

Код:

// Если игрок жив и неуязвим.

if (CanBeDamaged == false && health > 0)

{

// Тик таймера.

timer -= Time.deltaTime;

// Изменение альфа канала по синусоиде для эффекта

// моргания персонажа.

Color tmp = GetComponent<SpriteRenderer>().color;

GetComponent<SpriteRenderer>().color = new Color(tmp.r, tmp.g, tmp.b, 0.5f \* Mathf.Sin(timer \* Mathf.PI / 0.25f) + 0.5f);

}

Похожим образом устроен Falling. В нем с тиком таймера меняется положение персонажа по оси Y и значение альфа канала.

Код:

void Update()

{

// Проигрывается ли анимация и не закончилась ли она.

if (IsPlaying && timer > 0)

{

// Тик таймера.

timer -= Time.deltaTime;

// Перемещаем игрока по Y вниз.

gameObject.transform.position -= new Vector3(0, Time.deltaTime\*5f, 0);

// Уменьшаем значение альфа канала.

Color tmp = GetComponent<SpriteRenderer>().color;

GetComponent<SpriteRenderer>().color = new Color(tmp.r, tmp.g, tmp.b, tmp.a - 0.03f);

}

}

Магнит является дочерним объектом персонажа который имеет свой BoxCollider2D и скрипт Magnet. Когда коллайдер монеты касается коллайдера магнита она добавляется в лист монет. В методе Update() происходит проход по всем монетам листа и изменение их координат по средствам функции Mathf.Lerp. Таким образом все монеты попавшие в поле магнита, притягиваются к игроку.

* + 1. Описание функционирования врагов и алгоритма их поведения.

Привидение состоит из BoxCollider2D и скрипта Enemy. В скрипте есть список с точками, по которым должен двигаться враг. Точки расположены таким образом, что прямая между любыми соседними точками параллельна осям координат. В Udpate() враг движется к текущей точке, как только он ее достигает, то меняет точку и направлении движение и соответственно анимацию движения. Как только враг достигает последней точки списка, он начинает обходить точки в обратном порядке. Если коллайдер призрака соприкасается с коллайдером игрока, то вызывается метод скрипта HealthSystem GetDamage().

* + 1. Описание алгоритма демона бегущего за персонажем.

Демон состоит из BoxCollider2D и скрипта Death. В скрипте в методе Update() устанавливается позиция демона по X такая же как и у камеры, для того чтобы демон был посередине экрана. После чего проверяется, если персонаж оторвался от демона по оси Y больше чем на delta, демон телепортируется в позицию по Y = y(персонажа) – delta. И производится движение демона по оси Y. Кроме того при приближении демона к игроку возникает звуковой эффект погружения под воду, он устроен следующим образом. Если расстояние до персонажа меньше 7 то значение Cutoff freq у фильтра LowPass который повешен на миксер Master изменяется следующим образом:

master.audioMixer.SetFloat("MasterLowPassFreq", Mathf.Lerp(300, 5000, (Vector2.Distance(player.transform.position, transform.position) - 3) / 4f));

* + 1. Описание алгоритма генерации карты.

Вся карта состоит из заранее построенных блоков дороги – префабов. На сцене самой игры есть объект LevelGenerator, на котором висит одноименный скрипт. Изначально строится 7 кусков дороги (каждый кусок выбирается случайным образом из листа с префабами), затем как только персонаж отходит от первого куска более чем на delta первый кусок буфера удаляется и добавляется новый в начало карты. Чтобы понять, куда ставить новый блок, на каждом префабе висит скрипт BlockInfo, который содержит информацию о смещении первого элемента пола префаба относительно последнего по X и по Y.

* + 1. Описание алгоритма слежения камеры за персонажем.

На камеру повешен скрипт Camera, который отвечает за слежение. В методе Update() проверяется на сколько отошел персонаж от камеры, если он отдалился больше чем на delta, камера начинает двигаться к игроку с помощью функции Mathf.Lerp. Таким образом создается эффект плавного перемещения камеры за персонажем.

Код:

void Update()

{

// Если игрок жив.

if (!IsDie)

{

// Если игрок выбежал за максимальное отклонение.

if (Mathf.Abs(Vector3.Distance(player.transform.position + t, gameObject.transform.position)) > delta)

{

// Перемещаем камеру к игроку.

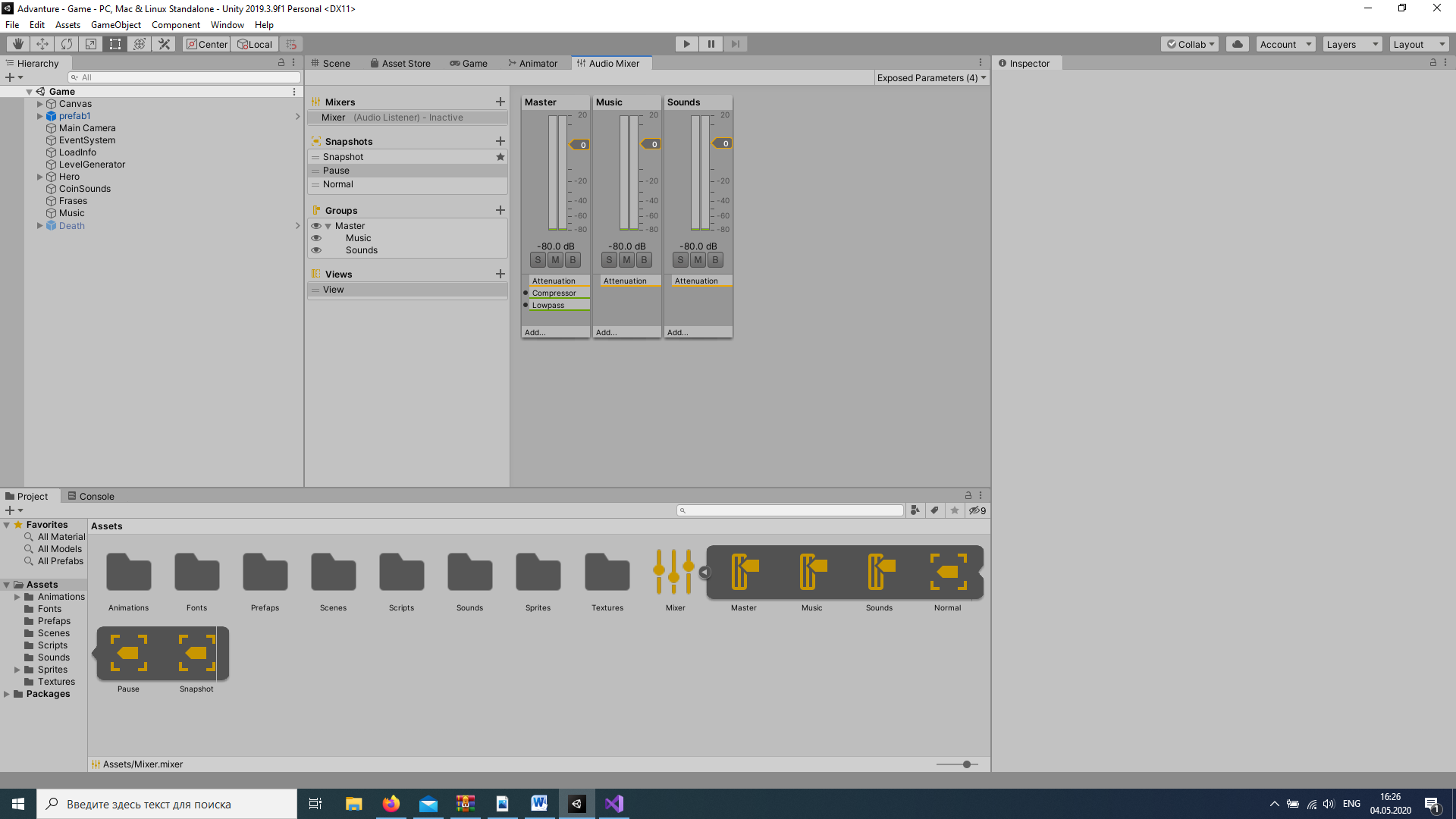
transform.position = Vector3.Lerp(gameObject.transform.position, player.transform.position + t, koef);

}

}

}

* + 1. Устройство звука.

В игре существует две аудио группы (дочерние группы Master) Music и Sounds. Все AudioSource подключены к одной из этих групп. И при изменении слайдеров громкости меняются соответствующие настройки громкости групп. Кроме того в игре созданы два снапшота звука Normal и Pause. Во время игры стоит Normal, а при открытии меню паузы или конца игры стоит Pause при котором все звуки становятся более тихими. Рисунок 5. Звук в игре.

* 1. Описание и обоснование выбора метода организации входных и выходных данных.
     1. Описание метода организации входных и выходных данных.

Все данные о прогрессе игрока: уровень прокачки, кол-во монет, рекорд; сохраняются в файл save.gamesave посредствам механизма сериализации данных. Загрузка данных происходит из того же файла с помощью десириализации. За механизм сохранения и загрузки отвечает скрип SaveLoadManager. Во время игры информация хранится в публичных статических полях класса Info.

* + 1. Обоснование выбора метода организации входных и выходных данных.

Метод организации входных и выходных данных, описанный в п. 3.3.1., выбран поскольку является простым в реализации и защищает информацию от ручного изменения путем открытия файла с сохранением в текстовом редакторе.

* 1. Описание и обоснование выбора состава технических и программных средств.

Список технических средств для корректной работы программы:

1. Мышь и клавиатура
2. Звуковая плата
3. Динамики или наушники
4. Процессор с поддержкой набора инструкций SSE2
5. Оперативная память 256 МБ
6. Не менее 1,5GB свободного места на жёстком диске
7. Графическая карта: DX10 или DX11 или DX12
8. USB разъем 2.0

Список программных средств для корректной работы программы:

1. Операционная система Windows 7 (SP1+) или Windows 10, 64-бит версий
2. DirectX 10 или более поздняя версия
3. Microsoft .NET Framework 4.6 или более поздняя версия

Клавиатура и мышь необходимы для осуществления ввода информации.

Звуковая плата и динамики (наушники) необходимы для прослушивания музыки и звуков.

1,5 GB на диске необходимы для корректной работы OC и .NET Framework

При кол-ве ОЗУ меньше указанного могут возникнуть проблемы, которые могут привести к падению программы.

USB 2.0 требуется для поставки программы на компьютер.

Остальные требования взяты с сайта Unity и необходимы для корректной роботы игрового движка.

1. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ
   1. Предполагаемая потребность

Программа будет использоваться в индустрии развлечений, для проведения времени, отдыха, снятия эмоционального напряжения и стресса, развития реакции и умения быстро принимать решения.

* 1. Технико-экономичес­кие показатели, обосновывающие преимущество выбранного варианта технического решения.

Игра была разработана с использованием игрового движка Unity 2D. Преимущество данного варианта технического решения состоит в том, что он абсолютно бесплатный в отличии от других аналогов. Unity распространяется бесплатно для физических лиц с условием, что оборот или объем привлеченных инвестиций не превышает 100 тыс. $ за последние 12 месяцев. Кроме того у Unity есть очень подробная документация и в силу распространенности движка, в сети можно найти множество уроков по работе с данным программным пакетом.

Существуют зарубежные аналоги данной программы. Например, ReDungeon. Преимуществом данной программы над ReDungeon в том, что разработка была абсолютно бесплатной, а так же программа обладает дополнительными игровыми функциями такими как, бонус здоровья или возможность прокачки магнита.

1. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ
2. ГОСТ 19.101-77 Виды программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.102-77 Стадии разработки. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.103-77 Обозначения программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.104-78 Основные надписи. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.105-78 Общие требования к программным документам. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.603-78 Общие правила внесения изменений. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
11. ГОСТ 15150-69 Машины, приборы и другие технические изделия. Исполнения для различных климатических районов. Категории, условия эксплуатации, хранения и транспортирования в части воздействия климатических факторов внешней среды. – М.: Изд-во стандартов, 1997.
12. Unity Learn URL: <https://learn.unity.com/tutorials> (дата обращения: 27.11.2019)
13. Unity Documentation URL: <https://docs.unity3d.com/ru/530/Manual/> (дата обращения: 27.11.2019)
14. System requirements for Unity 2019.3 <URL:https://docs.unity3d.com/Manual/system-requirements.html> (дата обращения 04.05.2020)

# ПРИЛОЖЕНИЕ 1 ТЕРМИНОЛОГИЯ

Демон – игровой объект, который движется вслед за персонажем по оси Y.

Runner – это жанр игр, где персонаж постоянно бежит/летит/ползет/прихрамывает вперед по теоретически бесконечному игровому миру.

Призрак(привидение) – враждебный NPC.

# ПРИЛОЖЕНИЕ 2 ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ

|  |  |
| --- | --- |
| Классы | Назначение |
| BlockInfo | Класс для хранения смещения координат начала и конца данного префаба. |
| Camera | Класс для слежения камеры за игроком. |
| Chest | Класс сундука, обработка открытия и информация о содержимом. |
| Coin | Класс монеты, обработка подбора, обработка взаимодействия с магнитом, информация о ценности. |
| Colya | Класс кольев, обработка нанесения урона игроку. |
| Death | Класс демона, движение за игроком, обработка убийства игрока. |
| Enemy | Класс врага, движение между точками, обработка столкновения с игроком. |
| Falling | Класс анимации падения игрока, после выхода за пределы поля. |
| Floor | Класс ячейки пола, выставляет координату по Z относительно координаты по Y. |
| FraseScript | Класс фразы при побитии рекорда, воспроизведение фразы. |
| GameOverMenu | Класс меню окончания игры, обработка нажатий на кнопки изменение звуковых настроек. |
| Gun | Класс пушки, периодическая стрельба объектом GunAmmo. |
| GunAmmo | Класс снаряда пушки, движения, обработка касания игрока. |
| HealthBonus | Класс бонуса жизней, обработка подбора игроком. |
| HealthImproving | Класс улучшения жизней, обработка попытки улучшить кол-во жизней. |
| HealthSystem | Класс системы жизней игрока, отображение информации о кол-ве жизней, обработка получения урона и лечения. |
| ImproveSceneMeneger | Класс управления сценой улучшения, обработка нажатий кнопок и обновление текстовой информации на сцене. |
| Info | Статический класс для хранения информации об игроке во время работы программы. |
| LaserGun | Класс лазерной пушки, анимация, периодическая работа, обработка касания игрока. |
| LevelGenerator | Класс генерации раунда. |
| LoadInfoGame | Класс загрузки информации об игроке при запуске программы. |
| Magnet | Класс магнита, обработка захваченных в поле монет, добавление и удаление монет из поля. |
| MagnetImproving | Класс улучшения магнита, обработка попытки улучшения магнита. |
| MainMenu | Класс главного меню, обработка работы кнопок и слайдеров. |
| Obstacle | Класс препятствия, отталкивает игрока на начальную позицию при попытке пройти сквозь. |
| PauseMenu | Класс меню паузы, обработка работы кнопок. |
| Player\_money | Класс отвечающий за подсчет денег игрока во время забега. |
| Player\_score | Класс отвечающий за подсчет очков игрока. |
| PlayerController | Класс игрока отвечающий за управление, движение, смерть персонажа. |
| SaveLoadManager | Класс отвечающий за сохранение и загрузку информаций об игроке в и из памяти компьютера |
| SetInfoGame | Класс установки текущего рекорда при начале раунда. |
| SoundColya | Класс отвечающи за компенсацию положения объекта звука по оси Z из-за смещения клеток пола. |
| SoundScript | Класс отвечает за воспроизведения звука монеты и изменения ее pitch при сборе монет подряд. |
| Teleport | Класс телепорта, отвечает за обработку телепортации игрока. |

# ПРИЛОЖЕНИЕ 3 ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ МЕТОДОВ И СВОЙСТВ

Описание полей методов и свойств класса BlockInfo.cs.

|  |  |  |  |
| --- | --- | --- | --- |
| Поля | | | |
| Имя | Модификатор доступа | Тип | Назначение |
| X | public | int | Длинна префаба по x. |
| Y | public | int | Длинна префаба по y. |

Описание полей методов и свойств класса Camera.cs.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | |
| Имя | | Модификатор доступа | | | Тип | Назначение |
| player | | public | | | GameObject | Игрок. |
| koef | | public | | | float | Коэффициент для смещения камеры по Lerp. |
| delta | | public | | | float | Расстояние на котором перестает действовать Lerp. |
| IsDie | | public | | | bool | Мертв ли игрок. |
| t | | private | | | Vector3 | Смещение камеры по Z относительно player. |
| Методы | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение |
| Start | private | | void | - | | Установка начального положения камеры. |
| Update | private | | void | - | | Слежение за игроком. |

Описание полей методов и свойств класса Chest.cs.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | |
| Имя | | Модификатор доступа | | | Тип | Назначение |
| animator | | public | | | Animator | Аниматор сундука. |
| bc | | private | | | BoxCollider2D | Коллайдер сундука с триггером. |
| cost | | private | | | int | Награда за открытие. |
| Методы | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение |
| Start | private | | void | - | | Инициализация коллайдера и аниматора. |
| OnTriggerEnter2D | private | | void | Collider2D col | | Обрабатывает взаимодействие игрока с триггером. |

Описание полей методов и свойств класса Coin.cs.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | |
| Имя | | Модификатор доступа | | | Тип | Назначение |
| red\_coin | | public | | | Sprite | Картинка красной монеты. |
| green\_coin | | public | | | Sprite | Картинка зеленой монеты. |
| cost | | private | | | int | Ценность монеты. |
| Методы | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение |
| Start | private | | void | - | | Установка ценности монеты и ее картинки в зависимости от очков игрока. |
| OnTriggerEnter2D | private | | void | Collider2D col | | Обработка взаимодействия монеты с игроком или магнитом. |

Описание полей методов и свойств класса Colya.cs.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | |
| Имя | | Модификатор доступа | | | Тип | Назначение |
| bc | | public | | | BoxCollider2D | Коллайдер кольев. |
| start\_delay | | public | | | float | Начальная задержка перед работой. |
| timer | | private | | | float | Таймер для цикличной работы кольев. |
| IsAttack | | private | | | bool | Активны ли колья. |
| animator | | private | | | Animator | Аниматор кольев. |
| source | | public | | | AudioSource | Звук кольев. |
| Методы | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение |
| Start | private | | void | - | | Установка таймера, инициализация аниматора и триггера. |
| Update | private | | void | - | | Циклическая работа кольев. |
| OnTriggerEnter2D | private | | void | Collider2D col | | Обработка взаимодействия с игроком. |

Описание полей методов и свойств класса Deth.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| player | | public | | | PlayerController | | Игрок. |
| camera | | public | | | GameObject | | Камера. |
| master | | public | | | AudioMixerGroup | | Миксер Master (главная аудио группа). |
| speed | | public | | | float | | Скорость демона. |
| delay | | public | | | float | | Фора для игрока. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Инициализация камеры, игрока. Установка позиции демона с учетом форы. | |
| Update | private | | void | - | | Движение демона за игроком и изменение аудио-эффектов в зависимости от расстояния до игрока. | |
| OnTriggerEnter2D | private | | void | Collider2D col | | Обработка взаимодействия демона с игроком. | |

Описание полей методов и свойств класса Enemy.cs.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | |
| Имя | | Модификатор доступа | | | Тип | Назначение |
| points | | public | | | Transform[] | Точки по которым враг передвигается при патруле. |
| current\_point | | private | | | int | Точка к которой в данный момент передвигается враг. |
| IsForward | | private | | | bool | Прямой порядок обхода или обратный. |
| animator | | public | | | Animator | Аниматор врага. |
| direction | | private | | | Dir | Направление в котором двигается враг. |
| Методы | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение |
| Start | private | | void | - | | Установка начальной позиции и направления. Запуск анимации. |
| Update | private | | void | - | | Обход точек. |
| OnTriggerEnter2D | private | | void | Collider2D col | | Обработка взаимодействия врага с игроком. |

Описание полей методов и свойств класса Falling.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| timer | | private | | | float | | Таймер для проигрывания анимации падения. |
| IsPlaying | | private | | | bool | | Проигрывается ли анимация. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Инициализация таймера. | |
| Update | private | | void | - | | Проигрывание анимации в зависимости от IsPlaying. | |
| Play | public | | void | - | | Запуск анимации падения. | |

Описание полей методов и свойств класса Floor.cs.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| Start | private | void | - | Установка координаты по z равную координате по y для изометрии. |

Описание полей методов и свойств класса FraseScript.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| source | | private | | | AudioSource | | Фраза. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Инициализация source. | |
| Play | public | | void | - | | Запуск source. | |

Описание полей методов и свойств класса GameoverMenu.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | | Назначение |
| total\_coins | | public | | Text | | | Счетчик общего количества монет. |
| gameoverMenuUI | | public | | GameObject | | | Меню конца игры. |
| puseMenu | | public | | GameObject | | | Меню паузы. |
| Normal | | public | | AudioMixerSnapshot | | | Снапшот звука во время игры. |
| OnPaused | | public | | AudioMixerSnapshot | | | Снапшот звука на паузе. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Аргументы | Назначение | |
| GameOverMenu | public | | void | | - | Открытие меню конца игры. | |
| Restart | public | | void | | - | Перезапуск забега. | |
| MainMenu | public | | void | | - | Выход в главное меню. | |

Описание полей методов и свойств класса Gun.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | | Назначение |
| ammo | | public | | GameObject | | | Снаряд. |
| shotDir | | public | | Transform | | | Направление выстрела. |
| timer | | private | | float | | | Таймер для циклической стрельбы. |
| Player | | private | | Transform | | | Игрок. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Аргументы | Назначение | |
| Start | private | | void | | - | Инициализация Player и timer. | |
| Update | private | | void | | - | Циклическая стрельба. | |
| Shoot | private | | void | | - | Выстрел. | |

Описание полей методов и свойств класса GunAmmo.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | | Назначение |
| speed | | public | | float | | | Скорость снаряда. |
| destroyTime | | public | | float | | | Время до уничтожения. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Аргументы | Назначение | |
| Start | private | | void | | - | Вызов функции DestroyAmmo через delayTime. | |
| Update | private | | void | | - | Движение снаряда. | |
| DestroyAmmo | private | | void | | - | Уничтожение объекта чнаряда. | |
| OnTriggerEnter2D | private | | void | | Collider2D col | Обработка взаимодействия снаряда с игроком. | |

Описание полей методов и свойств класса HealthBonus.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | | Назначение |
| player\_hp | | private | | HealthSystem | | | Жизни игрока. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Аргументы | Назначение | |
| Start | private | | void | | - | Инициализация player\_hp. | |
| OnTriggerEnter2D | private | | void | | Collider2D col | Обработка взаимодействия бонуса с игроком. | |

Описание полей методов и свойств класса HelthImproving.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | | Назначение |
| bars | | public | | Sprite[] | | | Картинки с разной степенью заполнения полоски прокачки. |
| costs | | private | | int[] | | | Цены на прокачку. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Аргументы | Назначение | |
| Start | private | | void | | - | Инициализация цен и картинки текущего уровня прокачки. | |
| Improve | public | | void | | - | Улучшение хп. | |

Описание полей методов и свойств класса HealthSystem.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | | Назначение |
| CanBeDamaged | | public | | bool | | | Уязвим ли герой в данный момент. |
| health | | public | | int | | | Текущее здоровье. |
| numberOfLifes | | public | | int | | | Максимальное количество жизней. |
| lifes | | public | | Image[] | | | Картинки сердечек. |
| fullLife | | public | | Sprite | | | Спрайт заполненного сердца. |
| emptyLife | | public | | Sprite | | | Спрайт пустого сердца. |
| timer | | private | | float | | | Таймер для неуязвимости после получения урона. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Аргументы | Назначение | |
| Start | private | | void | | - | Инициализация жизней и максимального количества хп. | |
| Update | private | | void | | - | Обработка неуязвимости при получении урона и отображение количества сердец. | |
| GetDamage | public | | void | | - | Получение урона игроком. | |
| AddHP | public | | void | | - | Исцеление, при сборе HealthBonus. | |

Описание полей методов и свойств класса ImproveSceneManager.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | | Назначение |
| coins | | public | | Text | | | Счетчик монет. |
| cost\_magnet | | public | | Text | | | Цена прокачки магнита. |
| cost\_hp | | public | | Text | | | Цена прокачки хп. |
| improve\_health | | public | | Button | | | Кнопка прокачки хп. |
| improve\_magnet | | public | | Button | | | Кнопка прокачки магнита. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Аргументы | Назначение | |
| Update | private | | void | | - | Обновление стоимостей прокачки и количества денег. | |
| MainMenu | public | | void | | - | Переход в главное меню. | |

Описание полей методов и свойств класса Info.cs.

|  |  |  |  |
| --- | --- | --- | --- |
| Поля | | | |
| Имя | Модификатор доступа | Тип | Назначение |
| total\_coins | public | int | Общее количество монет. |
| best\_record | public | int | Лучший рекорд. |
| numberOfLifes | public | int | Максимальное количество хп. |
| costs | public | int[] | Цены прокачек. |
| magnetLevel | public | float | Мощность магнита. |

Описание полей методов и свойств класса LaserGun.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | Назначение | |
| animator | | public | | Animator | | Аниматор пушки. | |
| start\_delay | | public | | float | | Начальная задержка перед циклической работой. | |
| timer | | private | | float | | Таймер для циклической работы. | |
| IsPrepare | | private | | bool | | Готовится ли пушка или стреляет. | |
| bc | | private | | BoxCollider2D | | Триггер пушки. | |
| source | | public | | AudioSource | | Звук пушки. | |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Аргументы | | Назначение |
| Update | private | | void | | - | | Циклические выстрелы. |
| Start | private | | void | | - | | Инициализация коллайдера, аниматора и таймера. |
| OnTriggerEnter2D | private | | void | | Collider2D col | | Взаимодействие с игроком. |

Описание полей методов и свойств класса LevelGenerator.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | | Назначение |
| Roads | | public | | GameObject[] | | | Префабы кусков дороги. |
| StartBlock | | public | | GameObject | | | Стартовый кусок дороги. |
| X | | private | | float | | | Координаты следующего префаба. |
| Y | | private | | float | | |
| Ylast | | private | | float | | | Координата по Y последнего блока дороги. |
| ColSpawn | | private | | int | | | Размер буфера дороги. |
| WasStartBlockDeleted | | private | | bool | | | Уничтожен ли стартовый блок. |
| PlayerTransf | | public | | Transform | | | Игрок. |
| CurrentBlocks | | private | | List<GameObject> | | | Буфер префабов дороги. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Аргументы | Назначение | |
| Start | private | | void | | - | Инициализация X,Y и заполнение буфера. | |
| Update | private | | void | | - | Вызов функции CheckForSpawn. | |
| CheckForSpawn | private | | void | | - | Проверка, можно ли ставить следующий блок и удалять последний. | |
| SpawnBlock | private | | void | | - | Спавн нового префаба дороги. | |
| DestroyBlock | private | | void | | - | Удаление последнего блока дороги. | |

Описание полей методов и свойств класса LoadInfoGame.cs.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| Start | private | void | - | Загрузка сохранения игры. |

Описание полей методов и свойств класса Magnet.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| CoinsField | | private | | | List<GameObject> | | Монеты которые были захвачены полем магнита. |
| player | | private | | | GameObject | | Игрок. |
| power | | public | | | float | | Мощность магнита. |
| bc | | public | | | BoxCollider2D | | Коллайдер магнита. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Инициализация игрока и радиуса магнита. | |
| Update | private | | void | - | | Притягивание монет из поля к игроку. | |
| AddToField | public | | void | GameObject obj | | Добавление монеты в поле. | |

Описание полей методов и свойств класса MagnetImproving.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| bars | | public | | | Sprite[] | | Картинки прокачки магнита. |
| costs | | private | | | int[] | | Стоимость прокачки. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Инициализация стоимостей и уровня прокачки. | |
| Improve | public | | void | - | | Улучшение магнита. | |

Описание полей методов и свойств класса MainMenu.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| mixer | | public | | | AudioMixer | | Миксер звука. |
| option\_menu | | public | | | GameObject | | Меню настроек. |
| menu | | public | | | GameObject | | Меню. |
| slider\_music | | public | | | Slider | | Слайдер громкости музыки. |
| slider\_master | | public | | | Slider | | Слайдер общей громкости. |
| slider\_sound | | public | | | Slider | | Слайдер громкости звуков. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Инициализация слайдеров. | |
| PlayGame | public | | void | - | | Начать игру. | |
| Improve | public | | void | - | | Перейти к улучшениям. | |
| Exit | public | | void | - | | Выход из игры. | |
| Options | public | | void | - | | Открытие меню настроек. | |
| Back | public | | void | - | | Закрыть меню настроек. | |
| ChangeTotalVolume | public | | void | float volume | | Изменение общей громкости. | |
| ChangeSoundVolume | public | | void | float volume | | Изменение громкости звуков. | |
| ChangeMusicVolume | public | | void | float volume | | Изменение громкости музыки. | |

Описание полей методов и свойств класса Obstacle.cs.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| OnTriggerEnter2D | private | void | Collider2D col | Взаимодействия игрока с препятствием. |

Описание полей методов и свойств класса PauseMenu.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | Тип | | | Назначение |
| GameIsPaused | | public | | bool | | | Стоит ли игра на паузе. |
| pauseMenuUI | | public | | GameObject | | | Родительский объект меню паузы. |
| Normal | | public | | AudioMixerSnapshot | | | Снапшот звука во время игры. |
| OnPaused | | public | | AudioMixerSnapshot | | | Снапшот звука на паузе. |
| mixer | | public | | AudioMixerGroup | | | Миксер(Master). |
| slider\_music | | public | | Slider | | | Слайдер громкости музыки. |
| slider\_master | | public | | Slider | | | Слайдер общей громкости. |
| slider\_sound | | public | | Slider | | | Слайдер громкости звуков. |
| settings\_menu | | public | | GameObject | | | Меню настройки звука. |
| menu | | public | | GameObject | | | Меню паузы. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | | Аргументы | Назначение | |
| Start | private | | void | | - | Установка слайдеров. | |
| Update | private | | void | | - | Контроль нажатия ESC. | |
| Resume | public | | void | | - | Продолжение игры. | |
| Pause | public | | void | | - | Установка паузы. | |
| MainMenu | public | | void | | - | Возвращение в главное меню. | |
| Options | public | | void | | - | Переход в меню настроек. | |
| BackButton | public | | void | | - | Возвращение из меню настроек. | |
| ChangeTotalVolume | public | | void | | float volume | Изменение общей громкости. | |
| ChangeSoundVolume | public | | void | | float volume | Изменение громкости звуков. | |
| ChangeMusicVolume | public | | void | | float volume | Изменение громкости музыки. | |

Описание полей методов и свойств класса player\_money.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| coins | | public | | | int | | Деньги заработанный в текущем забеге. |
| TextMoney | | public | | | Text | | Счетчик монет. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Инициализация coin. | |

Описание полей методов и свойств класса player\_score.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| current\_score | | public | | | int | | Очки в данном забеге. |
| TextScore | | public | | | Text | | Счетчик очков. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Инициализация current\_coin. | |

Описание полей методов и свойств класса PlayerController.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| IsRecordWasBitten | | private | | | bool | | Был ли побит рекорд. |
| IsOnGround | | private | | | bool | | Находится ли персонаж на земле. |
| IsMoving | | public | | | bool | | Движется ли игрок. |
| IsGamePaused | | public | | | bool | | Находится ли игра на паузе. |
| IsGameOverMenuOpen | | public | | | bool | | Открыто ли меню конца игры. |
| gameoverMenu | | public | | | GameoverMenu | | Меню конца игры. |
| animator | | public | | | Animator | | Аниматор персонажа. |
| step | | private | | | Vector2 | | Шаг перемещения героя. |
| new\_position | | public | | | Vector2 | | Позиция в которую перемещается персонаж. |
| start\_position | | private | | | Vector2 | | Начальная позиция при загрузке уровня. |
| up\_vector | | private | | | Vector2 | | Единичные вектора для всех направлений перемещения. |
| down\_vector | | private | | | Vector2 | |
| left\_vector | | private | | | Vector2 | |
| right\_vector | | private | | | Vector2 | |
| direction | | private | | | Dir | | Направление персонажа. |
| timer | | private | | | float | | Таймер задержки после смерти. |
| koef | | public | | | float | | Коэффициент Lerp для перемещения. |
| delta | | public | | | float | | Радиус прекращения действия Lerp. |
| rb | | public | | | RigidBody2D | | Физическое тело. |
| bc | | public | | | BoxCollider2D | | Коллайдер персонажа. |
| magnet | | public | | | GameObject | | Магнит. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Инициализация полей. | |
| Update | private | | void | - | | Перемещение, считывание клавиш. | |
| EndGame | public | | void | - | | Таймер конца игры. | |
| Move | public | | void | - | | Движение игрока. | |
| GetInput | private | | void | - | | Считывание нажатий. | |
| OnTriggerEnter2D | private | | void | Collider2D col | | Проверка нахождения на полу. | |
| Die | public | | void | int code | | Смерть персонажа. | |
| GoToStart | public | | void | - | | Возвращение к позиции до движения. | |

Описание полей методов и свойств класса SaveLoadManager.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| filePath | | private | | | string | | Путь к файлу сохранения. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Создание файла сохранения если его нет. | |
| SaveGame | public | | void | - | | Сохранение игры. | |
| LoadGame | public | | void | - | | Загрузка игры. | |

Описание полей методов и свойств класса SetInfoGame.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| record | | public | | | Text | | Рекорд. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Инициализация record. | |

Описание полей методов и свойств класса SoundColya.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| pref | | public | | | GameObject | | Пол к которому прикреплён звук. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Компенсация сдвига пола по Z. | |

Описание полей методов и свойств класса SoundScript.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| coins | | private | | | AudioSource | | Звук монетки. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| Start | private | | void | - | | Инициализация coins. | |
| GetCoinSound | public | | void | - | | Проигрыш coins с изменением pitch. | |

Описание полей методов и свойств класса Teleport.cs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| Имя | | Модификатор доступа | | | Тип | | Назначение |
| finishPoint | | public | | | Floor | | Точка куда телепортируется игрок. |
| startPoint | | public | | | Floor | | Точка откуда телепортируется. |
| Методы | | | | | | | |
| Имя | Модификатор доступа | | Тип | Аргументы | | Назначение | |
| OnTriggerEnter2D | private | | void | Collider2D col | | Процесс телепортации. | |

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |