

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

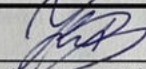
Направление подготовки: 09.03.01 «Информатика и вычислительная техника»
Профиль подготовки: «Программное обеспечение средств вычислительной техники и автоматизированных систем»

БАКАЛАВРСКАЯ РАБОТА

на тему

ИГРОВОЕ ПРИЛОЖЕНИЕ В ЖАНРЕ «РАННЕР»
НА ПЛАТФОРМЕ UNITY3D

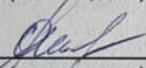
Студент


(подпись, дата)

31.05.2023

А.С. Корнаухова

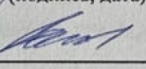
Руководитель


(подпись, дата)

31.05.2023

О.В. Юрова

Нормоконтролёр


(подпись, дата)

31.05.2023

А.С. Бычков

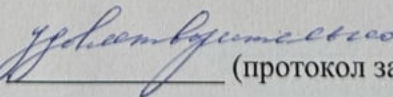
Работа допущена к защите (протокол зас. кафедры от 02.06.23 г. № 14)

Заведующий кафедрой
(подпись)


(фамилия, инициалы)

М.А. Митрохин

Работа защищена с отметкой



(протокол заседания ГЭК от

21.09.2023

№ 36)

Секретарь ГЭК


(подпись)

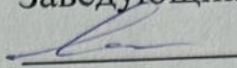
Е.И. Гурин
(фамилия, инициалы)

Пенза 2023

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
 М.А. Митрохин
« 26 » апреля 20 23 г.

**ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ**

1. Студент Корнаухова Анастасия Сергеевна гр 19ВВП1 факультета вычислительной техники.

2. Тема работы «Игровое приложение в жанре «раннер» на платформе Unity3D»

Тема утверждена приказом ПГУ № 458/о от "26" апреля 2023 г.

3. Руководитель бакалаврской работы к.т.н., доцент, каф. ВТ ПГУ Юрова О.В.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

Реализовать игровое приложение на платформе *Unity3D*, предоставляющее следующее:

- возможность передвижения персонажа влево/вправо;
- контроль за уровнем энергии;
- возможность пополнения энергии;
- систему подсчета очков рейтинга занесения их в таблицу рекордов.

Также провести тестирование и отладку каждого элемента приложения.

5. Перечень подлежащих разработке вопросов

Требуется произвести анализ предметных областей: изучить существующие аналоги, выделить их достоинства и недостатки. Выбрать язык программирования, который подойдет для написания логики приложения. Определить архитектуру приложения и вспомогательные средства разработки.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Изучение предметной области	50	26.04.2023-30.04.2023	<i>фиа</i>
2.	Анализ требований	10	01.05.2023-02.05.2023	<i>фиа</i>
3.	Выбор технологий, применяемых при проектировании и разработке	50	03.05.2023-07.05.2023	<i>фиа</i>
4.	Описание алгоритмов	40	08.05.2023-11.05.2023	<i>фиа</i>
5.	Реализация игрового приложения на платформе Unity3D	150	12.05.2023-29.05.2023	<i>фиа</i>
6.	Оформление пояснительной записки	24	30.05.2023-31.05.2023	<i>фиа</i>

Дата выдачи задания "26" апреля 2023 г.

Руководитель бакалаврской работы _____ *фиа* Юрова О.В.
(подпись, дата)

Задание к исполнению принял студент _____ *Корнаухова* Корнаухова А.С.
(подпись, дата)

Работу к защите допустить _____ *Финонова* Фионова Л.Р.
Декан факультета (подпись, дата)

Реферат

Бакалаврская работа содержит 96 с., 15 рис., 1 табл., 4 источн. и 3 прил.
ИГРОВОЙ ПРОЕКТ, *UNITY*, *VISUAL STUDIO*, ЯЗЫК *C#*, ИНФОРМАЦИОННЫЕ
ТЕХНОЛОГИИ, КОМПЬЮТЕРНАЯ ГРАФИКА

Цель работы – реализация игрового приложения. Для достижения данной цели были определены следующие задачи:

- провести анализ предметной области;
- провести проектирование приложения;
- провести функциональное тестирование.

В результате выпускной квалификационной работы было создано игровое приложение на основе *Unity3D*.

					ПГУ1.090301.10.001 ПЗ		
Изм.	Лист	№ докум.	Подпись	Дата			
Разраб		Корнаухова А.С.			Лит.	Лист	Листов
Провер.		Юрова О.В.				2	96
Н. Контр		Бычков А.С.			ФВТ, каф. ВТ гр. 19ВВП1		
Утверд		Митрохин М.А.					
					Игровое приложение в жанре «раннер» на платформе <i>Unity3D</i> Пояснительная записка		

Содержание

Перечень сокращений и обозначений.....	5
Введение.....	6
1 Мобильная игра в жанре «раннер» на платформе Unity3D.....	8
1.1 Анализ предметной области.....	8
1.1.1 Анализ рынка.....	8
1.1.2 Анализ аналогичных проектов.....	9
1.1.3 Анализ существующих решений для реализации проекта.....	13
1.1.4 Постановка задачи.....	15
1.1.4.1 Общие сведения.....	15
1.1.4.2 Диаграмма вариантов использования.....	15
1.2 Выбор средств.....	16
1.2.1 Программные средства, используемые при разработке системы.....	16
1.2.2 <i>Unity Editor</i>	17
1.2.3 <i>Visual studio</i>	18
1.3 Реализация проекта.....	19
1.3.1 Разработка графического оформления.....	19
1.3.2 Разработка игры.....	21
1.3.3 Тестирование.....	29
1.3.4 Технические характеристики.....	32
2 Безопасность жизнедеятельности.....	34
2.1 Вредоносные факторы.....	34
2.2 Требования к помещению.....	35
2.3 Требования к помещению.....	36
2.4 Требование к рабочему месту.....	36
2.5 Требования к организации работы.....	35
Заключение.....	39
Список используемых источников.....	40
Приложение А — <i>UML</i> диаграммы.....	41

Приложение Б — Листинг программы.....	45
Приложение В — Презентация.....	85

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		4

Перечень сокращений и обозначений

В настоящей работе применяют следующие сокращения и обозначения

Игрок – человек, играющий в видеоигры

Монетизация – совокупность элементов игры, прямо или косвенно влияющих на игровой процесс, ведущих к увеличению продолжительности игровой сессии, способствующих вовлеченности игрока и, таким образом, предоставляющих игроку возможность потратить деньги внутри игры

ПК – персональный компьютер

ПЭВМ – персональная электронно-вычислительная машина

Раннер – жанр мобильных игр, в которых игрок контролирует персонажа, который бежит по бесконечно генерирующемуся уровню, избегая препятствий и собирая бонусы

API (Application Programming Interface) – интерфейс программирования приложений

AR (Augmented Reality) – технология, которая позволяет добавлять в реальный мир дополнительные элементы, создавая таким образом гибридную среду

IDE (Integrated Development Environment) – интегрированная среда разработки.

VR (Virtual Reality) – технология, которая позволяет создавать компьютерную симуляцию окружающей среды, в которой пользователь может взаимодействовать с объектами и перемещаться по пространству.

UML (Unified Modeling Language) – унифицированный язык моделирования

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		5

Введение

Мы живем в то время, когда почти у каждого человека есть смартфон, ведь он стал неотъемлемой частью нашей жизни. В наше время даже у детей, начиная с самого раннего детства, появляются телефоны личного пользования.

Со временем все больше людей приобретают новейшие смартфоны для личного пользования, поскольку производители телефонов стараются делать свои товары доступнее в ценовом сегменте и более мощными по сравнению с предыдущими моделями. Раньше телефон использовался только как средство связи, а не для развлечения, но теперь ситуация изменилась. С каждым годом появляется все больше приложений и игр для развлечения. Именно поэтому работа над разработкой игр для мобильных телефонов сегодня является хорошо оплачиваемой.

С развитием технологий и увеличением числа пользователей смартфонов, рынок мобильных игр стал очень перспективным. Сейчас игры для мобильных устройств занимают лидирующие позиции в мировой индустрии развлечений. В 2020 году объем рынка мобильных игр составил более 77 миллиардов долларов, а к 2023 году он может достигнуть 106 миллиардов долларов.

Причиной такого роста является не только увеличение числа пользователей смартфонов, но и улучшение качества графики и звука в играх, а также возможность играть в любое время и в любом месте. Большинство игр для мобильных устройств бесплатны, но содержат внутриигровые покупки, что позволяет разработчикам получать доход.

Таким образом, разработка игр для смартфонов является актуальной и перспективной областью, которая привлекает все больше внимания и инвестиций.

Аудитория мобильных игр очень широка и разнообразна. Среди нее можно выделить как детей и подростков, так и взрослых людей всех возрастов и профессий. Многие люди играют в мобильные игры для развлечения и отдыха, а другие – чтобы развивать свои навыки и умения.

Среди аудитории мобильных игр можно выделить следующие группы:

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		6

- дети и подростки. Эта группа является одной из наиболее активных пользователей мобильных игр. Они играют в различные жанры игр, такие как аркады, головоломки, спортивные игры и т.д.;

- молодые люди. Эта группа также очень активна в использовании мобильных игр. Они часто играют в игры, которые позволяют им общаться с друзьями и участвовать в онлайн-соревнованиях;

- взрослые люди. Эта группа также является активными пользователями мобильных игр. Они часто играют в стратегические и игры-симуляторы, которые позволяют им развивать свои навыки и умения;

- пожилые люди. Эта группа пользователей мобильных игр намного меньше, но они также используют мобильные игры для развлечения и отдыха.

Раннер (англ. *runner*) – это жанр мобильных игр, в которых игрок управляет персонажем, который бежит по бесконечно генерирующемуся уровню, избегая препятствий и собирая бонусы. В раннерах игрок должен быстро реагировать на изменения ситуации на экране, чтобы не попасть в ловушку или не столкнуться с препятствием.

Игры-раннеры остаются актуальными благодаря своей простоте и доступности. Они не требуют сложного управления и могут быть играны даже новичками в мире видеоигр. Кроме того, они предлагают быструю и захватывающую игровую механику, что делает их привлекательными для широкой аудитории. Некоторые игры-раннеры также имеют элементы социальной игры, что позволяет игрокам соревноваться с друзьями и другими игроками по всему миру. В целом, игры-раннеры остаются популярными и актуальными в игровой индустрии.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		7

1 Мобильная игра в жанре «раннер» на платформе Unity3D

1.1 Анализ предметной области

1.1.1 Анализ рынка

Анализ рынка мобильных игр показывает, что он постоянно растет и развивается. Сегодня мобильные игры являются одним из самых популярных видов развлечения. Они доступны для загрузки на большинстве мобильных устройств и предлагают широкий выбор жанров и типов игр.

Среди наиболее популярных жанров мобильных игр можно выделить аркады, головоломки, стратегии, спортивные и симуляторы. Каждый из этих жанров имеет свои особенности и привлекает свою аудиторию.

Одним из главных факторов успеха мобильных игр является их доступность. Игры можно загрузить бесплатно, а затем приобретать внутриигровые элементы за реальные деньги. Это позволяет игрокам получать удовольствие от игры без необходимости тратить большие суммы денег.

Также важным фактором успеха мобильных игр является их социальный аспект. Многие игры предлагают возможность соревноваться с друзьями и общаться с ними внутри игры. Это создает дополнительный стимул для игры и повышает ее интересность.

Однако на рынке мобильных игр есть и некоторые проблемы. Одна из них – высокая конкуренция. Сотни тысяч игр доступны для загрузки, и каждый день появляются новые. Это означает, что разработчикам нужно постоянно работать над улучшением своих игр и искать новые способы привлечения игроков.

Также существует проблема монетизации игр. Некоторые игры предлагают игрокам слишком много платных элементов, что может отталкивать пользователей. Другие игры, напротив, предлагают слишком мало возможностей для покупок, что может снижать доходы разработчиков.

В целом, рынок мобильных игр продолжает развиваться и привлекать все большее количество пользователей. Однако для успешного выхода на этот рынок

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		8

нужно учитывать множество факторов и постоянно работать над улучшением своих игр.

1.1.2 Анализ аналогичных проектов

Subway Surfers

Игра в жанре раннер, разработанная студией *Kiloo* и *SYBO Games*. Игрок управляет персонажем по имени Джейк, который бежит по рельсам метро, избегая препятствий и собирая монеты. В игре есть несколько персонажей, которых можно разблокировать за монеты или купить за реальные деньги.

Графика *Subway Surfers* выполнена в ярком и красочном стиле, а музыкальное сопровождение состоит из зажигательных ритмов. Игра имеет множество локаций, каждая из которых представляет собой город со своими достопримечательностями и особенностями.

В игре есть различные бонусы и улучшения, которые помогают игроку преодолеть препятствия и собрать больше монет. Например, можно использовать скейтборд или реактивный ранец, чтобы быстрее перемещаться по уровню.

Subway Surfers является одной из самых популярных игр в жанре раннер на мобильных устройствах. Она доступна для скачивания на платформах *iOS*, *Android* и *Windows Phone*.

Данную игру можно увидеть на рисунке 1.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		9



Рисунок 1 — Subway Surfers

Достоинства игры *Subway Surfers*:

- простота управления. Игра подходит для всех возрастов и уровней игровой подготовки;
- красочная графика и дизайн. Визуальное оформление привлекает внимание и создает атмосферу города;
- бесконечный режим игры. Игрок может играть бесконечно долго, не зависимо от того, насколько хорошо он играет;
- регулярные обновления. Разработчики регулярно выпускают обновления с новыми персонажами, локациями и заданиями;

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		10

- возможность соревноваться с друзьями. Игроки могут соревноваться с друзьями, пытаясь набрать больше очков.

Недостатки игры *Subway Surfers*:

- монотонность. Игра может быстро надоесть из-за однообразных заданий и локаций;

- необходимость покупать игровые элементы. Для того, чтобы получить доступ к новым персонажам и улучшениям, игрокам нужно платить реальные деньги;

- неудобство при игре на маленьких экранах. Из-за большого количества движений и скорости игры, на маленьких экранах может быть сложно управлять персонажем;

- наличие рекламы. В игре присутствует много рекламных материалов, которые могут раздражать игроков;

- высокая нагрузка на батарею. Из-за большого количества движений и графики, игра может быстро разряжать батарею устройства.

Temple Run

Игра в жанре раннер, разработанная студией *Imangi Studios*. Игрок управляет персонажем, который бежит по древнему храму, избегая препятствий и собирая монеты. В игре есть несколько персонажей, которых можно разблокировать за монеты или купить за реальные деньги.

Графика *Temple Run* выполнена в реалистичном стиле, а музыкальное сопровождение состоит из мистических звуков. Игра имеет множество локаций, каждая из которых представляет собой древний храм со своими опасностями и загадками.

В данной игре также есть различные бонусы и улучшения, помогающие игроку преодолеть препятствия и собрать больше монет. Например, можно использовать магические кристаллы или силу рун, чтобы ускорить бег и увеличить количество монет.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		11

Temple Run является одной из самых популярных игр в жанре раннер на мобильных устройствах. Она доступна для скачивания на платформах *iOS*, *Android* и *Windows Phone*.

Скриншот из данной игры предоставлен на рисунке 2.



Рисунок 2 — *Temple Run*

Достоинства игры *Temple Run*:

- простота управления. Игра подходит для всех возрастов и уровней игровой подготовки;
- красочная графика и дизайн. Визуальное оформление привлекает внимание и создает атмосферу джунглей;
- бесконечный режим игры. Игрок может играть бесконечно долго, не зависимо от того, насколько хорошо он играет;
- разнообразие персонажей и локаций. Игрок может выбирать из разных персонажей и локаций, каждый из которых имеет свои особенности;
- возможность соревноваться с друзьями. Игроки могут соревноваться с друзьями, пытаясь набрать больше очков.

Недостатки игры *Temple Run*:

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		12

- монотонность. Игра может быстро надоесть из-за однообразных заданий и локаций;
- необходимость покупать игровые элементы. Для того, чтобы получить доступ к новым персонажам и улучшениям, игрокам нужно платить реальные деньги;
- неудобство при игре на маленьких экранах. Из-за большого количества движений и быстроты игры, на маленьких экранах может быть сложно управлять персонажем;
- наличие рекламы. В игре присутствует много рекламных материалов, которые могут раздражать игроков;
- высокая нагрузка на батарею. Из-за большого количества движений и графики, игра может быстро разряжать батарею устройства.

1.1.3 Анализ существующих решений для реализации проекта

Сегодня доступно множество инструментов для создания игр, как платных, так и бесплатных. Они варьируются от простых библиотек для популярных языков программирования до мощных редакторов с расширенным функционалом. Чтобы создать высококачественную игру, необходимо правильно выбрать инструмент для ее разработки.

Давайте рассмотрим наиболее популярные игровые платформы с обширным функционалом для разработки игр на популярных языках программирования и с положительными отзывами.

1. *Unity* — это одна из самых популярных платформ для разработки игр, которая поддерживает различные языки программирования, включая *C#*, *JavaScript* и *Boo*. *Unity* обеспечивает полный контроль над графикой, звуком и физикой игры, а также предоставляет множество инструментов для создания игровых механик и управления игровым процессом.

2. *Unreal Engine* — это еще одна популярная платформа для разработки игр, которая использует язык программирования *C++*. *Unreal Engine* обладает высоким уровнем графической детализации и поддерживает различные платформы, включая ПК, консоли и мобильные устройства.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		13

3. *Godot* — это бесплатная и открытая платформа для разработки игр, которая поддерживает языки программирования *C#* и *GDScript*. *Godot* предоставляет мощный редактор с интуитивно понятным интерфейсом и широким набором инструментов для создания игровых механик и управления игровым процессом.

4. *Construct* — это простой в использовании конструктор игр, который не требует знания программирования. *Construct* позволяет создавать игры с помощью перетаскивания объектов и настройки их свойств. Конструктор поддерживает множество платформ, включая ПК, консоли и мобильные устройства.

5. *RPG Maker* — это специализированная платформа для создания ролевых игр, которая предоставляет готовые шаблоны и инструменты для создания игровых механик и сюжета. *RPG Maker* поддерживает язык программирования *Ruby* и имеет большое сообщество пользователей, которые делятся своими проектами и опытом.

Выбор инструмента для разработки игры зависит от многих факторов, таких как опыт разработчика, требования проекта и целевая платформа. Однако, любой из перечисленных инструментов может стать отличным выбором для создания высококачественной игры.

Решение использовать *Unity* для разработки игры было обусловлено несколькими факторами:

- кроссплатформенность: *Unity* позволяет создавать игры для различных платформ, включая мобильные устройства, компьютеры и консоли;
- легкость использования: *Unity* имеет интуитивно понятный интерфейс и прост в освоении даже для новичков в разработке игр;
- большое сообщество: *Unity* имеет огромное сообщество разработчиков и пользователей, которые могут помочь с решением проблем и поделиться опытом;
- большой выбор ресурсов: *Unity* имеет огромную библиотеку ресурсов, таких как готовые модели, текстуры, звуки и другие элементы, которые могут быть использованы в игре;
- возможности для разработки *VR* и *AR* приложений: *Unity* является одной из лучших платформ для разработки виртуальной и дополненной реальности приложений;

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		14

- поддержка множества языков программирования: *Unity* поддерживает множество языков программирования, включая *C#, JavaScript, Boo* и другие;
- интеграция с *Visual Studio*: *Unity* интегрируется с *Visual Studio*, что облегчает разработку скриптов и приложений.

Исходя из вышеперечисленного: *Unity* является мощной и универсальной платформой для разработки игр и приложений, которая предоставляет разработчикам широкие возможности и инструменты для создания высококачественных продуктов.

1.1.4 Постановка задачи

1.1.4.1 Общие сведения

Задачей данной выпускной квалификационной работы является разработка игрового приложения в жанре «раннер», в которой выделено следующее:

цели игры:

- предоставить игрокам увлекательное времяпрепровождение;
- проехать как можно большее расстояние, избегая препятствий и собирая кофе;
- набрать наибольший результат.

Функции игры:

- управление мотоциклом, нажимая на экран устройства и изменяя направление движения вправо или влево;
- сбор кофейных кружек, чтобы оставаться бодрым и не засыпать за рулем;
- избегание различных препятствий на дороге;
- система подсчета очков и занесение их в рейтинг рекордов.

Поставленные задачи включают в себя взаимодействие с *API Unity*, проектирование архитектуры игрового приложения, разработку логики функций и тестирование всех заявленных функциональностей.

1.1.4.2 Диаграмма вариантов использования

UML-диаграмма вариантов использования (рисунок 3). была разработана в ходе анализа [1].

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		15

Данное игровое приложение направлено на использование одним человеком. Игрок управляет персонажем на мотоцикле.

Человек на мотоцикле – игровой объект, управляемый игроком, который перемещается на экране по горизонтали. Управление осуществляется с помощью свайпа влево-вправо (для перемещения), нажатия на экран (для подачи звукового сигнала).

Также в главном меню игрок может выполнить такие действия, как:

- начать игру: приступить к игровому процессу;
- посмотреть таблицу рекордов: в процессе игры игрок зарабатывает очки, которые записываются в конце игры. Данные сохраняются в таблицу.

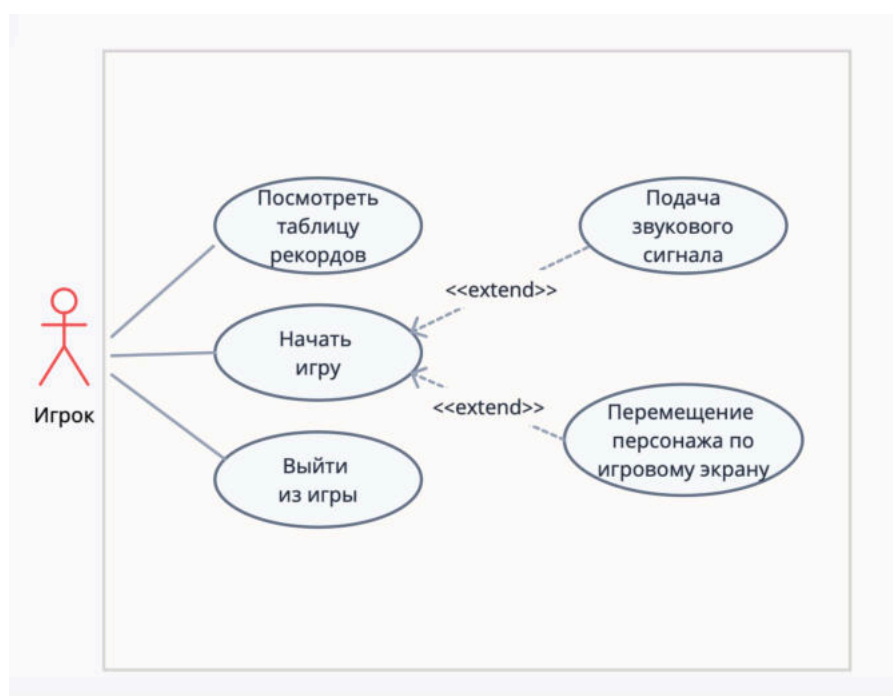


Рисунок 3 — Диаграмма вариантов использования

1.2 Выбор средств

1.2.1 Программные средства, используемые при разработке системы

Для разработки игры на Unity необходимы следующие среды:

- *unity editor*. Основная среда разработки, которая позволяет создавать и редактировать игровые объекты, скрипты, материалы и т.д.;

- *visual studio* или другой редактор кода для написания скриптов на языке *C#*;

- *3D*-моделирование и анимация. Для создания и анимации игровых объектов необходимо использовать специализированные программы, такие как *Blender*, *Maya* или *3ds Max*;

- графический дизайн. Для создания интерфейса, иконок и других элементов необходимо использовать графические редакторы, такие как *Adobe Photoshop* или *GIMP*;

- звуковая обработка. Для создания звуковых эффектов и музыки необходимо использовать звуковые редакторы, такие как *Audacity* или *Adobe Audition*.

1.2.2 *Unity Editor*

В *Unity Editor* используется для создания и редактирования игровых объектов, скриптов, материалов и других элементов игры. Он предоставляет инструменты для создания *2D* и *3D* графики, настройки физики, анимации, звуков и эффектов.

Основные работы, которые можно выполнять в *Unity Editor*:

- создание и редактирование игровых объектов: *Unity Editor* позволяет создавать различные игровые объекты, такие как персонажи, препятствия, окружение и т.д. Вы можете настраивать их свойства, добавлять компоненты и привязывать скрипты;

- написание скриптов: *Unity Editor* имеет встроенный редактор скриптов, который позволяет создавать и редактировать скрипты на языках *C#* и *JavaScript*. Вы можете использовать скрипты для управления поведением игровых объектов, создания логики игры, обработки пользовательского ввода и т.д.;

- создание и настройка материалов: *Unity Editor* позволяет создавать и настраивать материалы, которые определяют внешний вид игровых объектов. Вы можете настраивать текстуры, цвета, отражения и другие параметры материалов;

- работа с анимациями: *Unity Editor* имеет инструменты для создания и настройки анимаций игровых объектов. Вы можете создавать анимации движения, поворота, масштабирования и других параметров;

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		17

- настройка физики: *Unity Editor* позволяет настраивать физические свойства игровых объектов, такие как масса, трение, гравитация и т.д. Вы можете использовать физические свойства для создания реалистичных эффектов и поведения игровых объектов;

- работа со звуком: *Unity Editor* позволяет добавлять звуковые эффекты в игру и настраивать их параметры. Вы можете использовать звуковые эффекты для создания атмосферы игры и улучшения игрового опыта.

В целом, *Unity Editor* является основной средой разработки для создания игр на платформах *Windows, Mac OS, Linux, iOS, Android* и других. Он предоставляет разработчикам множество инструментов и возможностей для создания высококачественных игр.

1.2.3 *Visual studio*

Visual Studio — это интегрированная среда разработки (IDE), которая используется для написания кода на различных языках программирования, включая *C#, C++, Visual Basic* и другие. Она интегрируется с *Unity Editor* и используется для создания и редактирования скриптов, которые управляют поведением игровых объектов в *Unity*.

Основные работы, которые можно выполнять в *Visual Studio*:

- написание кода: *Visual Studio* предоставляет разработчикам удобный интерфейс для написания и редактирования кода на различных языках программирования. Вы можете использовать его для написания скриптов, которые управляют поведением игровых объектов в *Unity*;

- отладка кода: *Visual Studio* имеет встроенные инструменты для отладки кода, которые позволяют разработчикам искать и исправлять ошибки в своем коде. Вы можете использовать отладчик для пошагового выполнения кода, проверки значений переменных и т.д.;

- работа с проектами: *Visual Studio* позволяет создавать и управлять проектами, которые содержат все файлы и настройки, необходимые для разработки приложения. Вы можете использовать его для создания новых проектов, добавления файлов, настройки параметров проекта и т.д.;

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		18

- интеграция с системами контроля версий: *Visual Studio* интегрируется с различными системами контроля версий, такими как *Git*, *SVN* и другие. Это позволяет разработчикам управлять своим кодом и совместно работать над проектом;

- работа с плагинами: *Visual Studio* поддерживает плагины, которые добавляют дополнительные функции и возможности в *IDE*. Вы можете использовать плагины для улучшения работы с кодом, автоматизации задач и т.д.

В целом, *Visual Studio* является необходимым инструментом для разработки скриптов и приложений на платформе *Unity*. Он предоставляет разработчикам мощный набор инструментов и возможностей для создания высококачественных игр и приложений.

1.3 Реализация проекта

1.3.1 Разработка графического оформления

Для начала следует выделить такие понятия, как спрайт (*sprite*), префаб (*prefab*) и тайл (*tile*).

Спрайт в *Unity* — это 2D-изображение, которое используется для создания графических элементов игры, таких как персонажи, объекты окружения, интерфейс и т.д. Спрайты могут быть созданы в любом графическом редакторе и импортированы в *Unity*, где они могут быть использованы как отдельные элементы или объединены в анимации. Каждый спрайт имеет свои собственные свойства, такие как размер, масштаб, положение и т.д.

Префабы в *Unity* — это готовые к использованию объекты, которые можно создавать и настраивать заранее, чтобы использовать их в игре многократно. Префабы могут содержать любые компоненты, такие как модели, анимации, скрипты и другие элементы.

Префабы могут быть созданы из любого объекта в *Unity*, просто перетаскивая его в окно "Префабы". Затем можно настроить свойства префаба и сохранить его для использования в будущем.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		19

Префабы позволяют значительно ускорить процесс разработки игры, так как разработчик может создавать и настраивать объекты заранее и использовать их в любом месте игры. Это также помогает снизить количество дублированных объектов в игре и упрощает процесс обновления и изменения объектов.

Префабы могут использоваться для создания уровней игры, интерфейса пользователя, персонажей, оружия и многого другого. Они являются важным инструментом для разработки игр в *Unity*.

Тайл в *Unity* — это спрайт, который используется для создания повторяющихся элементов в игре, таких как фон, текстуры земли и стен, узоры и т.д. Тайлы обычно имеют одинаковый размер и могут быть объединены в большие карты, чтобы создать большие уровни или миры.

Для создания графического оформления игры были использованы различные элементы, доступные в Asset Store. Это включает анимации движения главного персонажа, препятствия и фоновое изображение, которое отображается за уровнем. Все эти элементы были интегрированы в *Unity* и использованы для создания игрового мира.

Также были использованы тайлы для создания текстур земли, стен и других поверхностей. Эти тайлы были объединены в большие карты, чтобы создать уровни игры. Были настроены свойства каждого тайла, такие как размер, текстура, масштаб и поворот, чтобы создать разнообразие визуальных элементов.

Для создания интерфейса игры были использованы графические элементы, такие как кнопки, текстовые поля и изображения. Эти элементы были созданы в *Unity* с помощью встроенных инструментов для *GUI*.

В целом, графическое оформление игры было создано с использованием различных элементов и инструментов, доступных в *Unity* и *Asset Store*. Было уделено внимание деталям и разнообразию визуальных элементов, чтобы создать привлекательный и интересный визуальный опыт для игроков (рисунок 4).

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		20



Рисунок 4 — Фоновое изображение уровня и используемые префабы

1.3.2 Разработка игры

Разработка данного игрового приложения велась на языке *C#* в среде разработки *Visual Studio*.

Игра состоит из двух сцен: игрового поля и меню. Иерархия объектов игрового поля представлена на рисунке 5.

					ПГУ1.090301.10.001 ПЗ	Лист
						21
Изм.	Лист	№ докум.	Подпись	Дата		

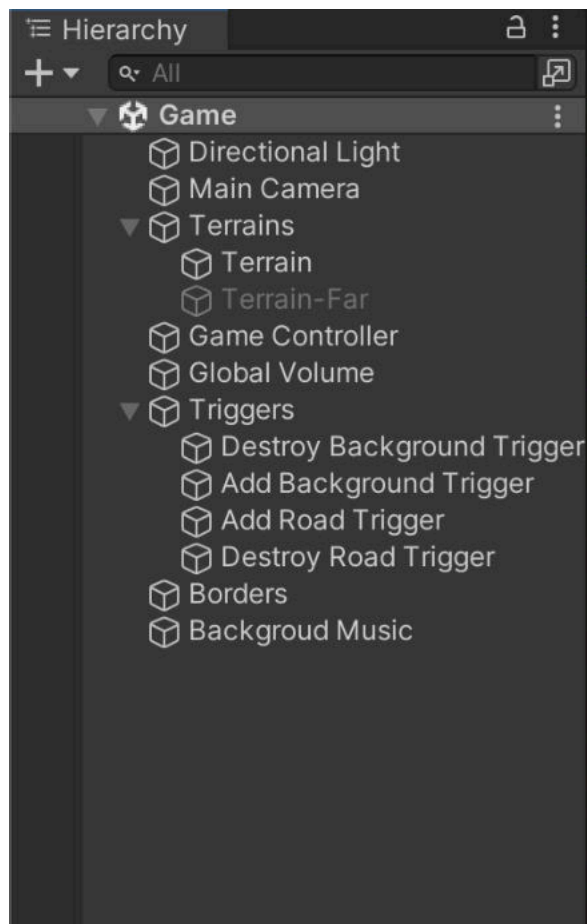


Рисунок 5 — Иерархия объектов

Программирование в *Unity3D* состоит из созданий пользовательских классов, которые в свою очередь подключаются к игровым объектам в качестве компонентов. Все такие классы наследуются от класса *MonoBehaviour*.

В данном проекте были спроектированы следующие классы:

Класс *GameController*:

- управляет игровым процессом;
- содержит ссылки на настройки игры и данные;
- использует библиотеку *Entitas.Unity* для работы с сущностями игры;
- имеет метод *LoadScene* для загрузки новой сцены;
- в методе *Awake* устанавливает целевую частоту кадров в зависимости от платформы;
- в методе *Start* инициализирует системы и контексты игры, устанавливает настройки и данные;

- в методе *Update* выполняет системы игры.

Библиотека *Entitas.Unity*:

- используется для работы с сущностями игры;
- позволяет создавать и изменять сущности, добавлять и удалять компоненты, обрабатывать события и т.д..

Классы *PlayerSetup*, *UISetup*, *RoadSetup*, *BackgroundSetup*, *GameData*:

- содержат настройки и данные для игры.

Класс *Contexts*:

- представляет собой контейнер для хранения контекстов игры.

Класс *Systems*:

- представляет собой контейнер для систем игры;
- инициализирует системы и контексты при запуске игры.

Класс *InputHandler*:

- отвечает за обработку ввода игрока с помощью *Unity Input System*.

Класс *Static*:

- является статическим и содержит методы для сохранения и загрузки статистики игры.

Классы *AddBackgroundCollision* и *AddRoadCollision* представляют собой компоненты *Unity*, которые используются для обработки столкновений объектов в игре.

Класс *UnityRigs* представляет собой компонент *Unity*, который содержит ссылки на компоненты *RigBuilder*, *Transform* и *Animator*.

Диаграммы используемых классов в скриптах предоставлены на рисунках 6–10:

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		23

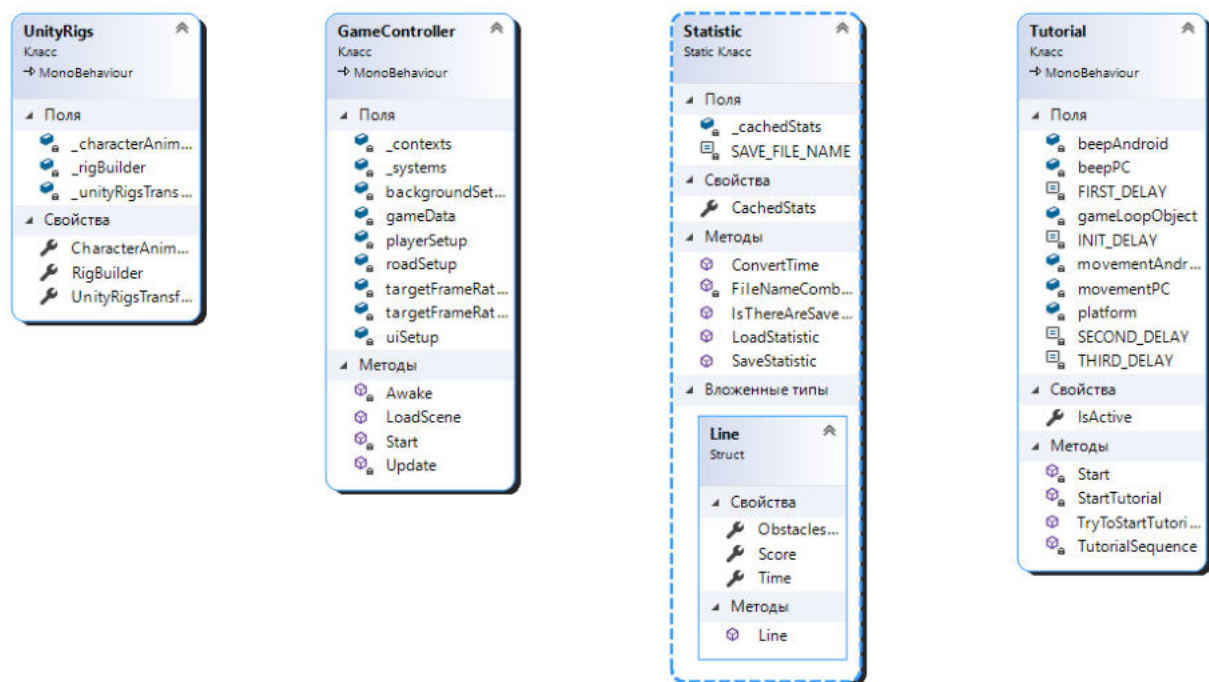


Рисунок 6 — Диаграмма основных скриптов

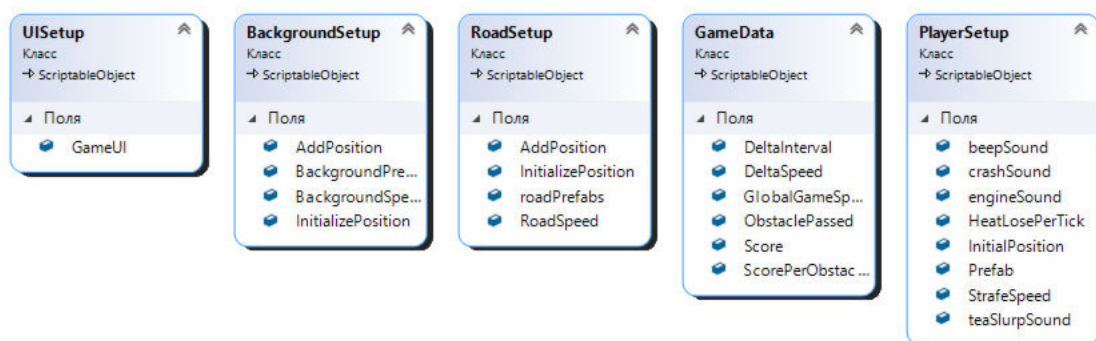


Рисунок 7 — Диаграмма *setup*-скриптов

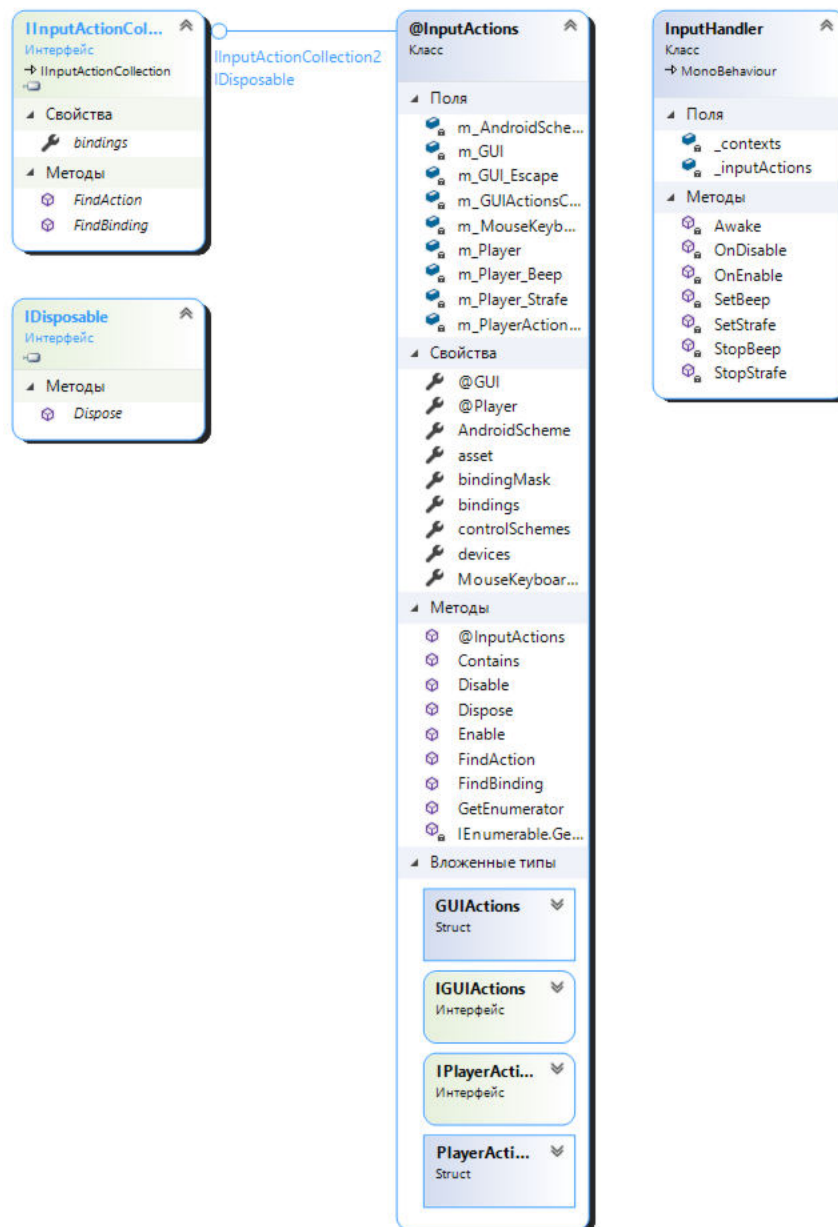


Рисунок 8 — Диаграмма *input*-скриптов

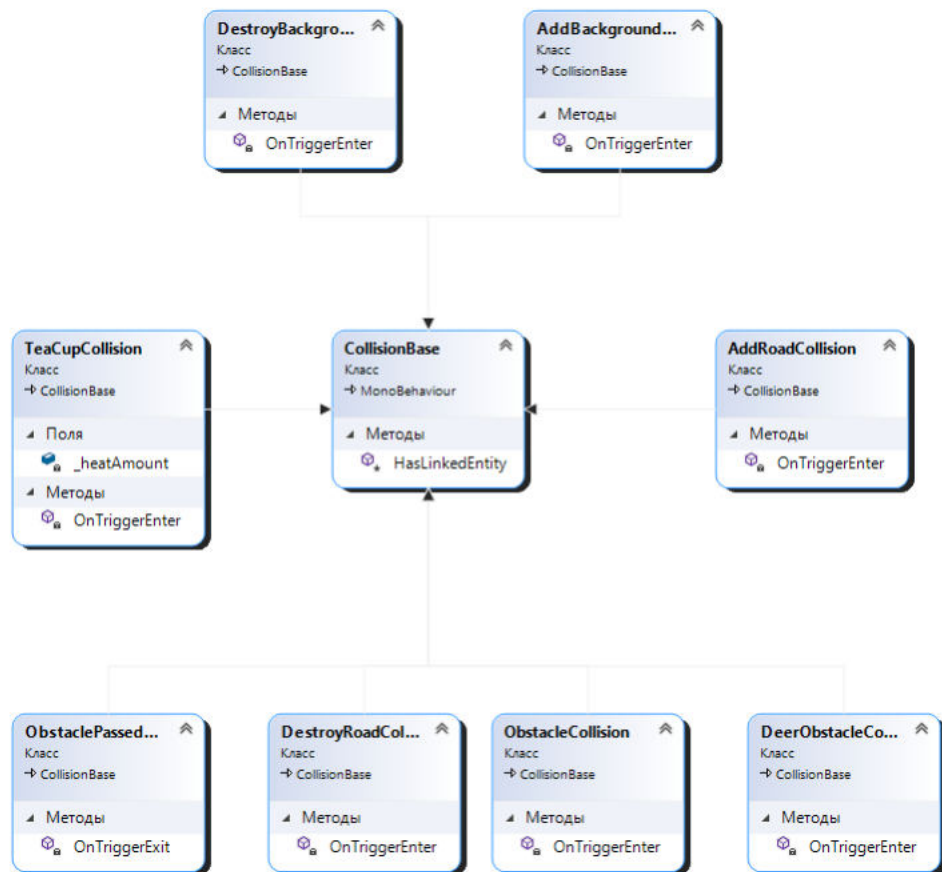


Рисунок 9 — Диаграмма *collisions*-скриптов

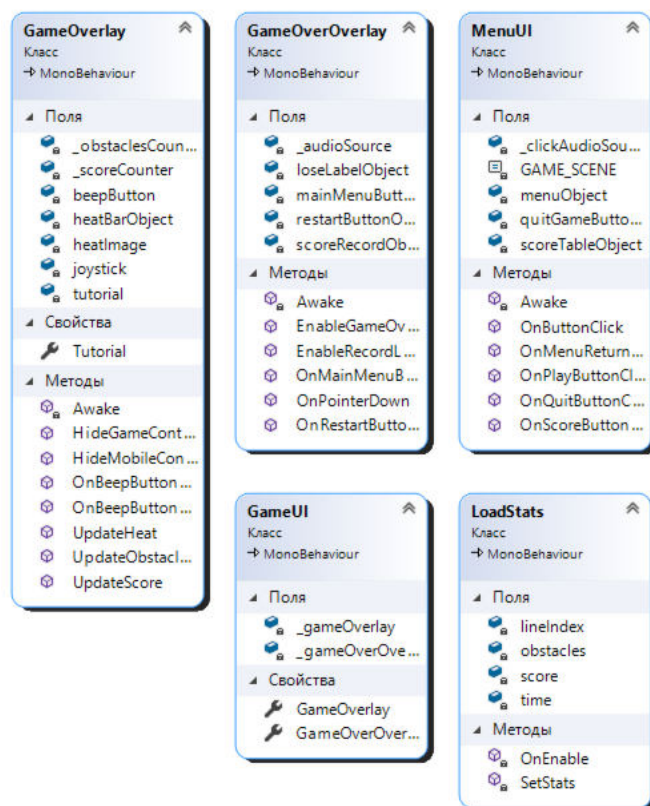


Рисунок 10 — Диаграмма *UI*-скриптов

После создания скриптов нужно перейти к созданию анимации [2]. Работать с анимацией следует в окне Аниматора (рис. 11). Здесь требуется настроить переходы между различными анимациями и установить связи между ними.

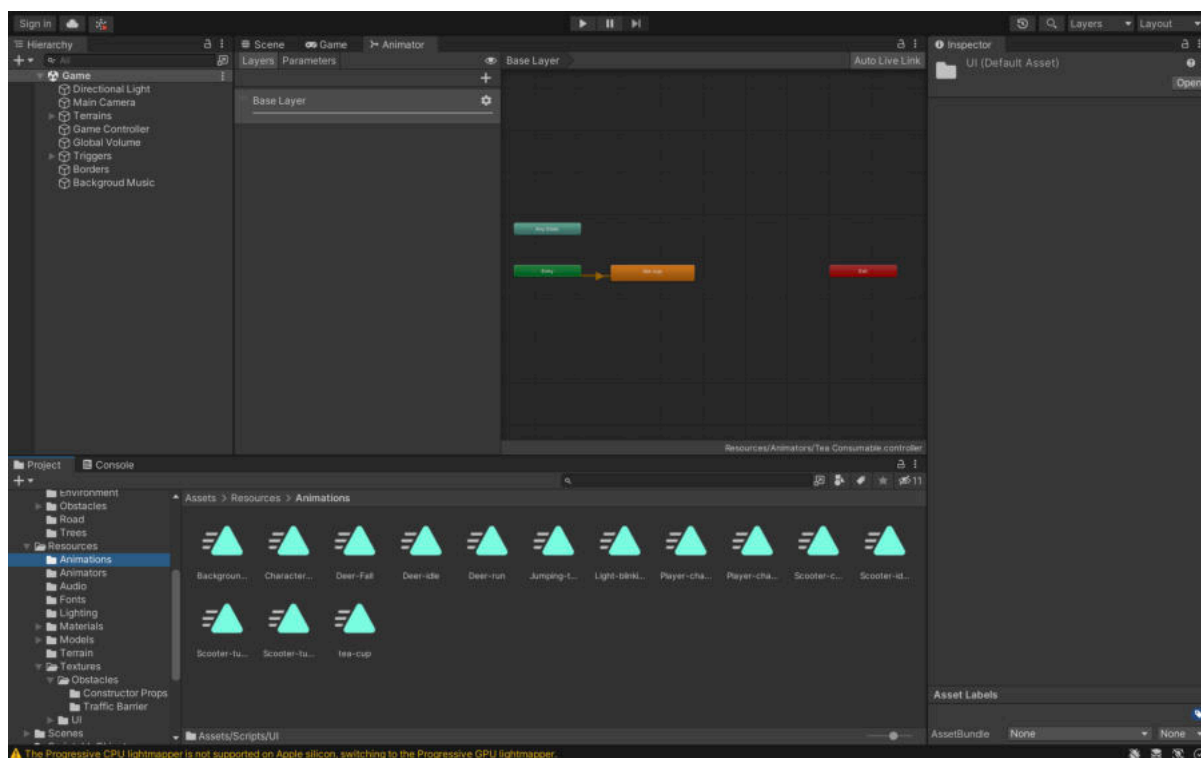


Рисунок 11 — Окно Аниматора и список анимаций

Далее проверить работает ли проект можно, нажав кнопку, которая обозначает запуск проекта, данная кнопка находится в верхней части экрана. Рядом с ней находятся кнопки паузы и окончания теста игры.

Окно запущенного проекта в *Unity* предоставлен на рисунках 12–15.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		27

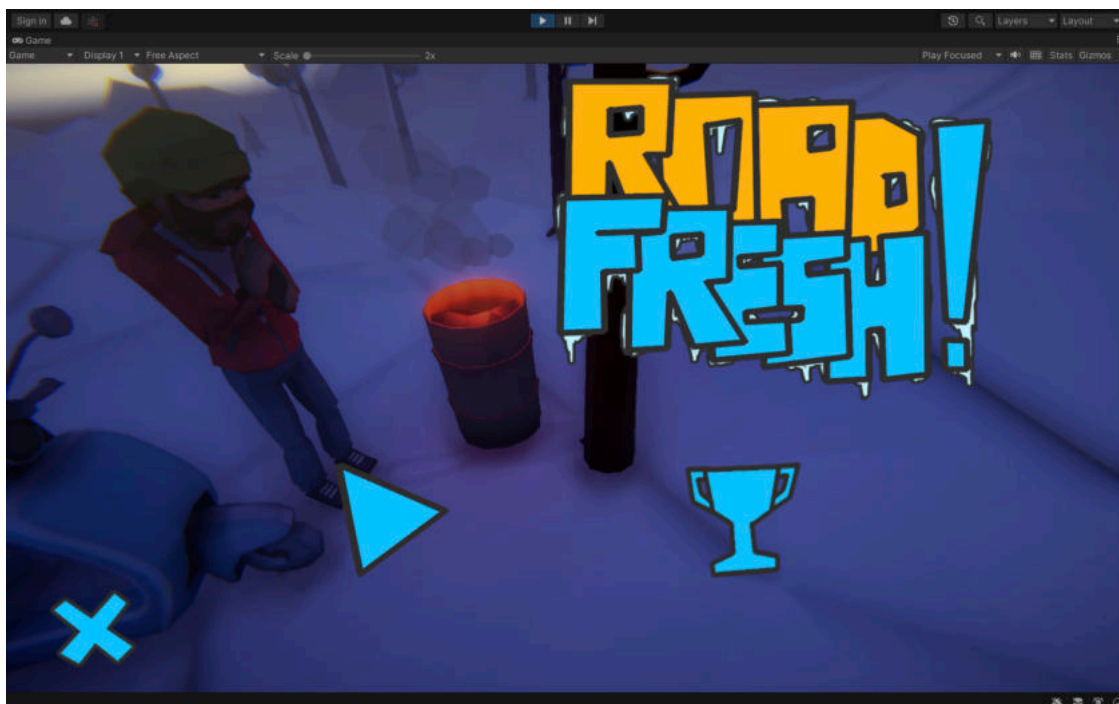


Рисунок 12 — Меню



Рисунок 13 — Начало игры

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		28



Рисунок 14 — Проигрыш игрока

UML-диаграмма деятельности предоставлена на рисунке 15.

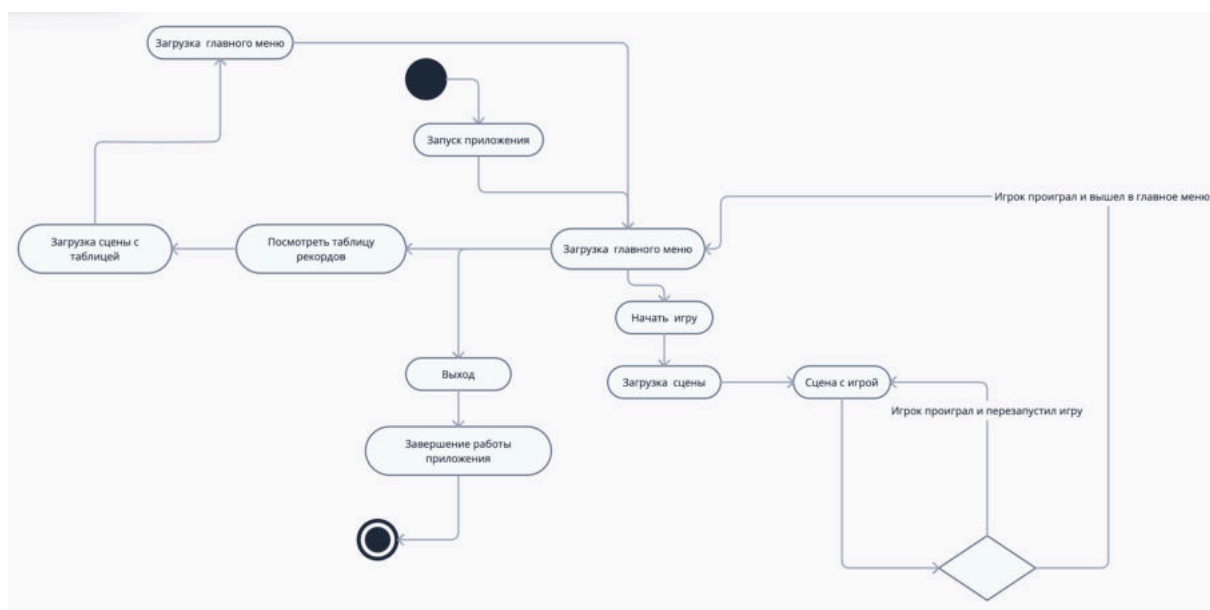


Рисунок 15 — Диаграмма деятельности

1.3.3 Тестирование

Данный тест приложения является проверкой на соответствие основным функциональным требованиям. Тест-план игрового приложения предоставлен в таблице 1.

Таблица 1 — Тест-план игрового приложения

№	Назначение	Действие	Ожидаемый результат	Полученный результат	Итог
1	Проверка правильной реакции персонажа на команды перемещения.	Используя кнопки передвижения, проехать часть игрового уровня.	Персонаж будет передвигаться в правильном направлении, анимация проигрывается так, как положено.	Персонаж передвигается в правильном направлении, анимация проигралась так, как положено.	Пройден.
2	Проверка корректного использования звукового сигнала.	Используя кнопку подачи сигнала, подать сигнал, тем самым спугнув животное.	Игрок отпугивает животное звуковым сигналом.	Игрок смог отпугнуть животное звуковым сигналом.	Пройден.
3	Проверка корректного использования игрового предмета.	Игрок проезжает через игровой предмет в виде чашки с кофе.	Когда игрок проезжает через чашку, проигрывается анимация. Шкала энергии восстанавливается.	Проехав через чашку, проигралась анимация. Шкала энергии восстановилась.	Пройден.

№	Назначение	Действие	Ожидаемый результат	Полученный результат	Итог
4	Проверка на правильную реализацию события проигрыша игрока.	Позволить игроку врезаться в препятствие	Когда игрок врезается в препятствие, проигрывается анимация падения игрока. Появляется меню выбора действий.	После того, как игрок врезался в препятствие, проигралась анимация падения игрока. Появилось меню выбора действий.	Пройден.
5	Проверка на правильную работу меню проигрыша.	1) Проиграть и перезапустить игру. 2) Проиграть и выйти в главное меню.	При нажатии определенной кнопки игра перезапускается или выходит в главное меню.	В зависимости от выбора игра перезапускается, либо игрок выходит в главное меню.	Пройден.
6	Проверка на правильную работу главного меню.	1) Запустить игру. 2) Открыть таблицу рекордов.	При нажатии определенной кнопки игра запускается, открывается таблица	В зависимости от выбора запускается игра, открывается таблица,	Пройден.

№	Назначение	Действие	Ожидаемый результат	Полученный результат	Итог
		3) Выйти из игры.	рекордов или выходит из игры в целом.	либо закрывается игра	
7	Проверка на правильную работу набора игровых очков.	Игрок проезжает мимо предмета, являющимся событием начисления очков.	Миновал предмет, начисляются 100 игровых очков, которые суммируются на протяжении всей игры.	Счет увеличился на 100 очков.	Пройден.

Данное тестирование показывает, что на данном этапе реализации проекта, игра соответствует функциональным требованиям выделенных на этапе постановки задачи.

1.3.4 Технические характеристики

В первую очередь, надо брать во внимание, что данная работа не является полностью готовой к релизу мобильной игрой, и оказывается прототипом, который является примерно ранней альфа-версией. Это связано с тем, что производство высококачественного продукта требует огромных затрат времени, поэтому на данный момент трудно определить, какие системные требования будут необходимы для будущего проекта, так как многие элементы еще не реализованы. Мы имеем только основные элементы игровой механики, поэтому технические характеристики для мобильной игры могут значительно измениться в будущем.

Однако, в настоящее время мы можем предположить, что для корректной

работы игры вам понадобится телефон с операционной системой не ниже *Android 5.0 или iOS 9.0*, процессором не ниже четырехъядерного с тактовой частотой от 1.5 ГГц, графическим процессором не ниже *Adreno 405* или *Mali-T760* и не менее 2 Гб оперативной памяти. Однако, эти требования могут измениться в будущем в зависимости от развития проекта.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		33

2 Безопасность жизнедеятельности

2.1 Вредоносные факторы

Работа за компьютером может негативно сказаться на здоровье человека, поэтому необходимо соблюдать ряд мер безопасности, чтобы минимизировать риски. Эти меры включают в себя регулярные перерывы, правильную осанку, использование специальных программ и настроек, а также занятия физическими упражнениями. Кроме того, необходимо соблюдать правила электробезопасности и использовать специальные средства защиты, такие как защитные очки и наушники. Все эти меры помогут сохранить здоровье и работоспособность при работе за компьютером.

Некоторые из вредных факторов трудового процесса при работе за компьютером могут включать в себя:

- монотонность работы: длительное время, проводимое за компьютером, может привести к усталости и раздражению;
- неправильная осанка: длительное время, проводимое в неправильной позе, может привести к боли в спине, шее и плечах;
- освещение: плохое освещение может привести к напряжению глаз и усталости;
- шум: шум от компьютера или других источников может привести к утомлению и стрессу;
- электромагнитные поля: компьютеры могут создавать электромагнитные поля, которые могут быть вредны для здоровья;
- радиация: компьютеры могут излучать радиацию, которая может быть вредна для здоровья;
- вибрация: некоторые компьютеры могут создавать вибрацию, которая может привести к утомлению и боли в руках и запястьях;
- пыль: компьютеры могут создавать пыль, которая может быть вредна для здоровья, особенно для людей с аллергиями или астмой;

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		34

- клавиатура и мышь: повторное нажатие клавиш клавиатуры или движение мыши может привести к боли в руках и запястьях;
- электрические удары: неправильное использование компьютера или нарушение правил электробезопасности может привести к электрическому удару.

2.2 Требования к помещению

Для обеспечения безопасности и комфорта при работе за компьютером необходимо соблюдать следующие требования к помещению:

- освещение: помещение должно быть хорошо освещено, чтобы предотвратить напряжение глаз;
- вентиляция: помещение должно иметь хорошую вентиляцию, чтобы предотвратить накопление пыли и улучшить качество воздуха;
- расположение компьютера: компьютер должен быть расположен на стабильной поверхности и находиться на расстоянии от глаз не менее 50 см.;
- мебель: рабочее место должно быть оборудовано удобной мебелью, которая поддерживает правильную осанку;
- электрическая безопасность: все электрические устройства должны быть безопасно подключены и заземлены;
- шум: помещение должно быть защищено от шума, который может повлиять на концентрацию и вызвать стресс;
- регулярные перерывы: необходимо делать регулярные перерывы, чтобы избежать усталости и напряжения;
- удобный доступ к необходимым материалам: все необходимые материалы, такие как документы или инструменты, должны быть удобно расположены в пределах досягаемости;
- устойчивость мебели: мебель должна быть устойчивой, чтобы предотвратить ее переворачивание или падение;
- предотвращение пыли: помещение должно регулярно убираться, чтобы предотвратить накопление пыли и улучшить качество воздуха.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		35

2.3 Требования к помещению

Освещение является важным фактором при работе за компьютером, поэтому необходимо соблюдать определенные требования для обеспечения комфортных условий труда:

- освещение должно быть достаточным и равномерным, чтобы предотвратить напряжение глаз и усталость. Недостаточное освещение может привести к ухудшению зрения и снижению производительности;
- источник освещения не должен создавать бликов на экране компьютера, так как это может вызвать дискомфорт и затруднить чтение информации;
- освещение должно быть регулируемым, чтобы можно было подстроить его под индивидуальные потребности пользователя. Каждый человек имеет свои предпочтения относительно яркости и цветовой температуры света;
- цветовая температура света должна быть комфортной для глаз. Рекомендуется использовать свет с теплой цветовой температурой (около 3000K), так как он более приятный для глаз и создает более расслабленную атмосферу;
- при необходимости можно использовать дополнительное местное освещение для улучшения видимости определенных зон на рабочем месте. Например, можно использовать направленный светильник для освещения документов или рабочей поверхности;
- рекомендуется использовать натуральное освещение, если это возможно, чтобы улучшить качество воздуха в помещении и снизить затраты на электроэнергию. Но необходимо учитывать, что солнечный свет может создавать блики на экране компьютера, поэтому его следует ограничивать или использовать специальные покрытия на окнах.

2.4 Требование к рабочему месту

Для создания комфортного и эффективного рабочего места программиста необходимо учитывать несколько требований. Важно обеспечить правильную эргономику мебели и оборудования, а также поддерживать оптимальные условия

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		36

освещения, звукоизоляции и температуры в помещении. Рабочий стол должен быть чистым и организованным, а индивидуальные потребности программиста должны быть учтены. Регулярные перерывы и физические упражнения также важны для поддержания здоровья.

Площадь рабочего места пользователя ПК с ЭЛТ-дисплеем должна составлять не менее 6 м², для ПК с плоским дисплеем - 4,5 м². Ежедневная влажная уборка и проветривание помещения также являются важными условиями для комфортной работы. Шумящее оборудование должно быть размещено вне рабочих мест сотрудников, чтобы избежать превышения нормативных уровней шума.

Хорошее освещение рабочего стола является одним из ключевых требований. Рабочие столы должны быть размещены таким образом, чтобы естественный свет падал преимущественно слева от мониторов. Расстояние между рабочими столами должно быть не менее 2,0 м, а расстояние между боковыми поверхностями видеомониторов - не менее 1,2 м.

Конструкция рабочего стола должна обеспечивать оптимальное размещение используемого оборудования. Высота рабочей поверхности стола должна составлять 725 мм, а рабочая поверхность стола - ширину 800-1400 мм и глубину 800-1000 мм. Рабочий стол должен иметь пространство для ног высотой не менее 600 мм, шириной - не менее 500 мм, глубиной на уровне колен - не менее 450 мм и на уровне вытянутых ног - не менее 650 мм.

Конструкция рабочего стула или кресла также является важным условием для комфортной работы. Рабочий стул или кресло должны быть подъемно-поворотными, регулируемые по высоте и углам наклона сиденья и спинки, а также расстоянию спинки от переднего края сиденья. Клавиатуру следует располагать на расстоянии 100-300 мм от края стола, а экран видеомонитора должен находиться на расстоянии 600-700 мм от глаз пользователя.

В целом, создание комфортного и эффективного рабочего места для программиста является важным условием для повышения производительности и поддержания здоровья.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		37

2.5 Требования к организации работы

Работа с компьютером должна быть организована в зависимости от вида и категории трудовой деятельности. Все виды трудовой деятельности можно разделить на три группы: считывание информации с экрана монитора с предварительным запросом (группа А), ввод информации (группа Б) и творческая работа в режиме диалога с ПЭВМ (группа В). Основная работа с ПЭВМ должна занимать не менее 50% времени в течение рабочей смены или рабочего дня, если выполняются работы, относящиеся к разным видам трудовой деятельности.

Для каждой группы трудовой деятельности установлены три категории тяжести и напряженности работы с ПЭВМ. Максимальные значения соответствуют III категории работы с ПЭВМ, которые определяются по суммарному числу считываемых или вводимых знаков за рабочую смену или по суммарному времени непосредственной работы с ПЭВМ за рабочую смену.

Для предотвращения утомления пользователей ПЭВМ рекомендуется чередовать работы с использованием ПЭВМ и без него. Если работа требует постоянного взаимодействия с компьютером, то рекомендуется делать перерывы на 10-15 минут через каждые 45-60 минут работы. Непрерывная работа без перерыва не должна превышать 1 час.

При работе с ПЭВМ в ночную смену (с 22:00 до 6:00) продолжительность регламентированных перерывов желательно увеличивать на 30%, независимо от категории и вида трудовой деятельности.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		38

Заключение

В рамках выпускной квалификационной работы было создано игровое приложение, которое может работать на разных платформах благодаря использованию среды *Unity*.

В рамках работы был проведен анализ существующих игровых приложений, которые имеют схожие функциональные возможности с созданным приложением. Для построения приложения были найдены готовые решения, которые были адаптированы к требованиям проекта. Также было проведено проектирование приложения, в результате которого была определена окончательная структура приложения и его функциональные возможности. Благодаря использованию среды *Unity*, приложение может работать на разных платформах, что делает его более удобным для пользователей. Все это было реализовано с помощью языка программирования *C#*.

После этого было проведено функциональное тестирование, в результате которого было установлено, что разработанное игровое приложение и все входящие в его состав объекты работают корректно. Таким образом, можно утверждать, что приложение соответствует всем требованиям качества программного обеспечения и готово к использованию.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		39

Список использованных источников

1. Хабгуд, Леоненков, А.В. Объектно-ориентированный анализ проектирования с использованием *UML* и *IBM Rational Rose*: учеб. пособие Текст. / А.В. Леоненков. М.: Интернет-Ун-т информ. технологий: БИНОМ, Лаборатория знаний, 2006. — 320 с.
2. Торн А. Основы анимации в *Unity* / А. Торн, В. Ратц, С. Джозеф. – Москва: Наука и техника, 2016. – 175 с. – ISBN 978-5-97060-377-2.
3. Бонд Д. Г. *Unity* и *C#*. Геймдев от идеи до реализации. 2-е изд. / Д. Г. Бонд. – Санкт-Петербург : Питер, 2021. – 928 с.
4. Ламмерс, К. Шейдеры и эффекты в *Unity*. Книга рецептов. Как с помощью шейдеров и постэффектов добиться потрясающей картинки в проектах на *Unity* / К. Ламмерс. – Россия : ДМК Пресс, 2016. – 274 с.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		40

Приложение А

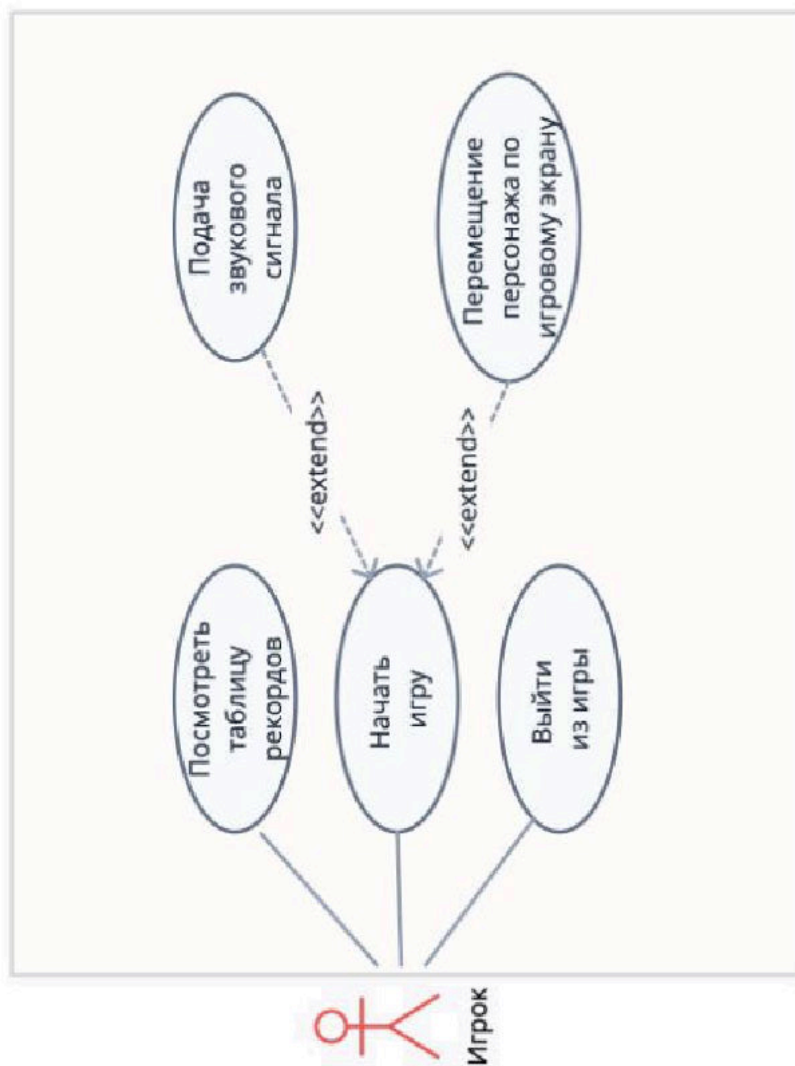
(обязательное)

Мобильная игра в жанре «раннер» на платформе *Unity3D*

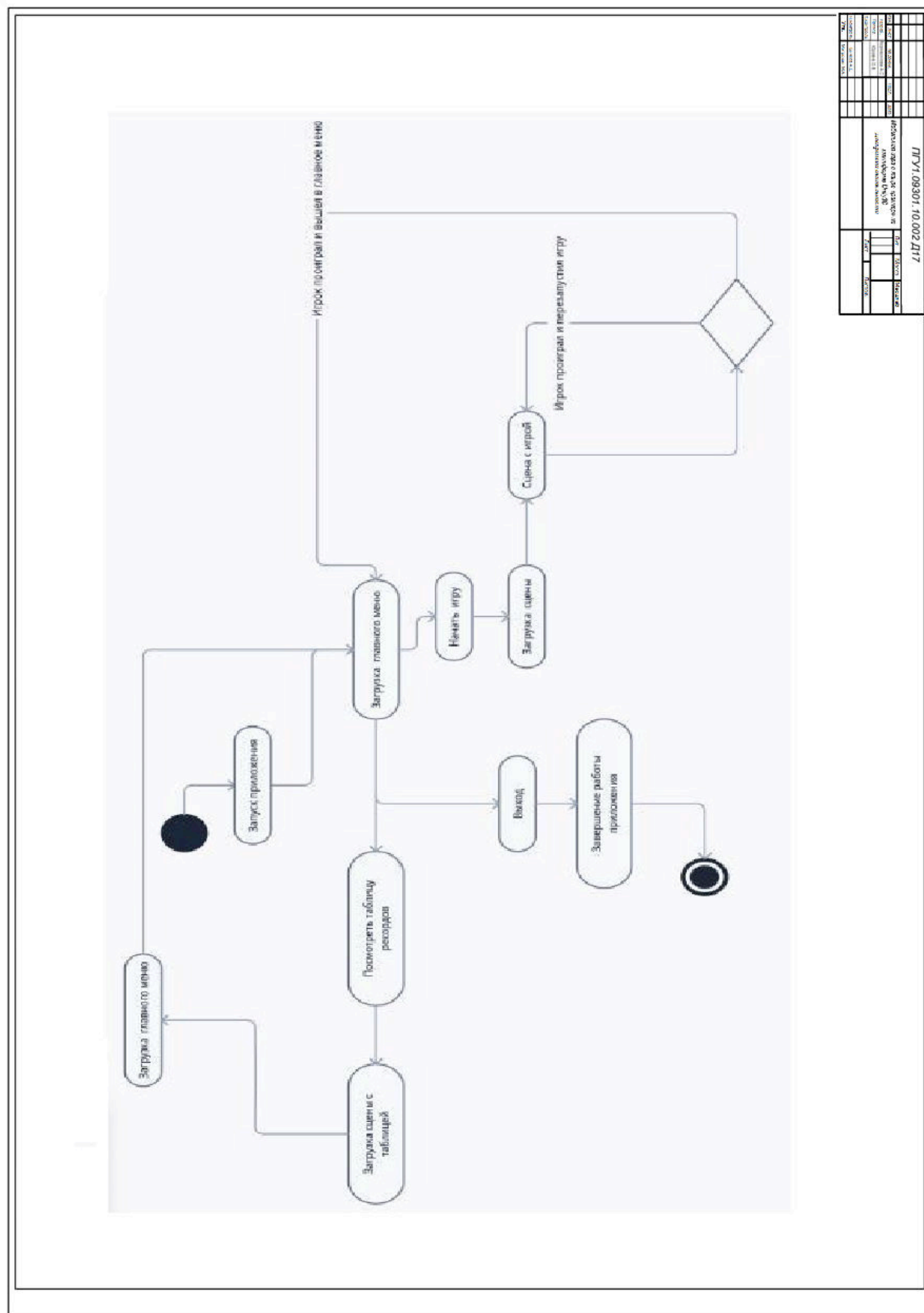
UML диаграммы

					ПГУ1.090301.10.001 ПЗ	Лист
						41
Изм.	Лист	№ докум.	Подпись	Дата		

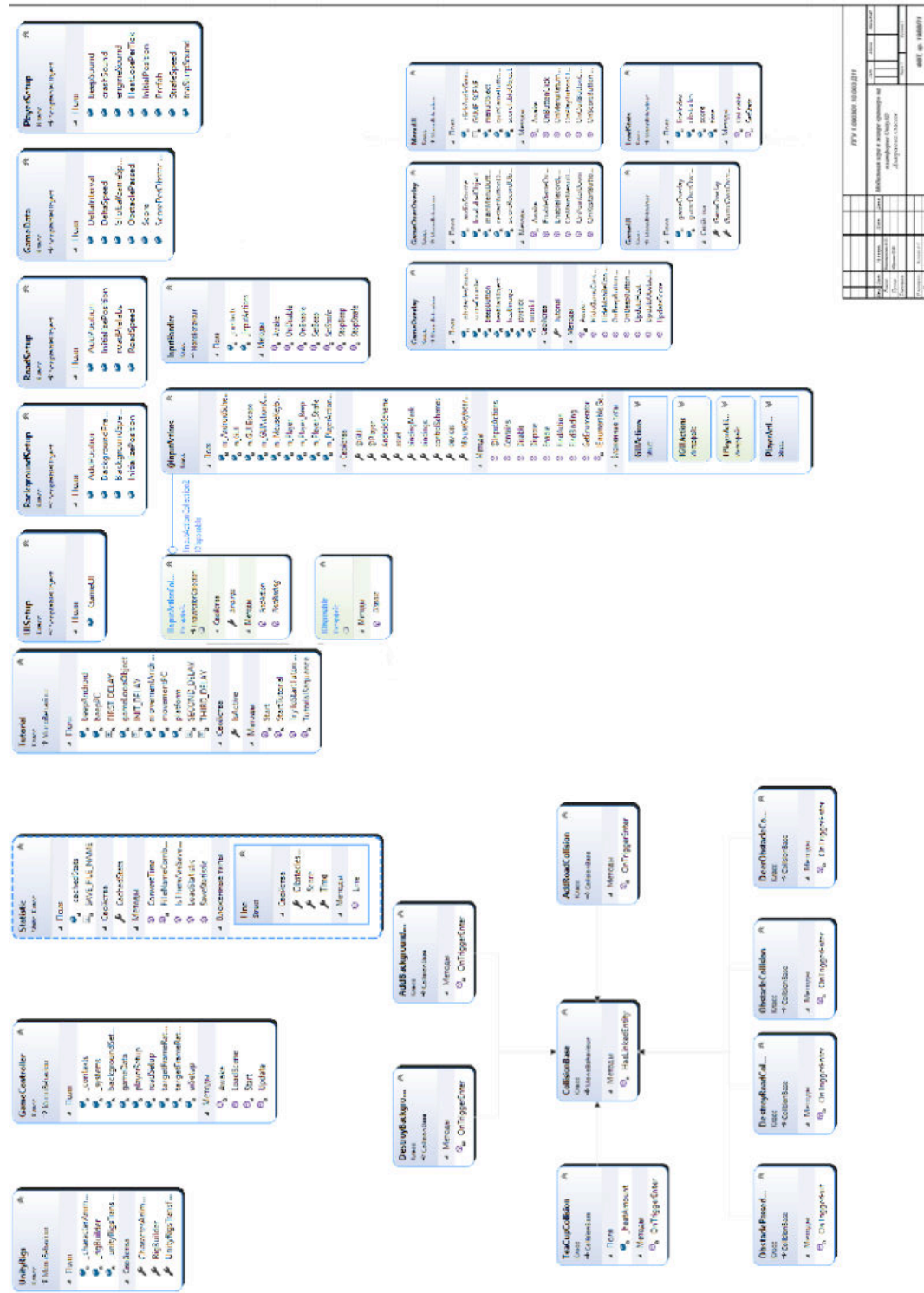
Приложение А. 1 – *UML* – диаграмма прецедентов

[illegible]

Приложение А.2 – *UML* – диаграмма деятельности



Приложение А.3 – *UML* – диаграмма классов



Приложение Б

(обязательное)

Мобильная игра в жанре «раннер» на платформе *Unity3D*

Листинг программы

Файл *GameController.cs*

```
using Entitas;  
using Entitas.Unity;  
using RoadFresh.Initialize.Setups;  
using UnityEngine;  
using UnityEngine.SceneManagement;  
  
public class GameController : MonoBehaviour  
{  
    [SerializeField]  
    private PlayerSetup playerSetup;  
    [SerializeField]  
    private UISetup uiSetup;  
    [SerializeField]  
    private RoadSetup roadSetup;  
    [SerializeField]  
    private BackgroundSetup backgroundSetup;  
    [SerializeField]  
    private GameData gameData;  
    [SerializeField]  
    private int targetFrameRateWindows;  
    [SerializeField]  
    private int targetFrameRateAndroid;  
  
    private Contexts _contexts;  
    private Systems _systems;
```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		45

```

public void LoadScene(string sceneName)
{
    var allEntities = _contexts.game.GetEntities();
    _systems.DeactivateReactiveSystems();
    _systems.ClearReactiveSystems();
    foreach (GameEntity entity in allEntities)
    {
        if (entity.hasView)
        {
            entity.view.value.Unlink();
            entity.Destroy();
            continue;
        }
        entity.Destroy();
    }
    SceneManager.LoadScene(sceneName);
}

private void Awake()
{
    if (Application.platform == RuntimePlatform.Android)
        Application.targetFrameRate =
targetFrameRateAndroid;
    else
        Application.targetFrameRate =
targetFrameRateWindows;
}

private void Start()
{

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		46

```

        _contexts = Contexts.sharedInstance;
        _systems = new GameSystems(_contexts);
        _contexts.game.SetGameController(this);
        _contexts.game.SetPlayerSetup(playerSetup);
        _contexts.game.SetRoadSetup(roadSetup);
        _contexts.game.SetBackgroundSetup(backgroundSetup);
        _contexts.game.SetGameData(gameData);
        _contexts.game.SetUISetup(uiSetup);
        _systems.Initialize();
    }

    private void Update()
    {
        _systems.Execute();
    }
}

```

Файл *Static.cs*

```

using System;
using System.Collections.Generic;
using System.IO;
using System.Runtime.Serialization.Formatters.Binary;
using UnityEngine;

namespace RoadFresh.Stats
{
    public static class Statistic
    {
        private static List<Line> _cachedStats;

        public static List<Line> CachedStats
        {
            get { return _cachedStats; }
        }
    }
}

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		47

```

    }

    private const string SAVE_FILE_NAME = "statistic.dat";
    public static List<Line> LoadStatistic()
    {
        var fileName = FileNameCombine(SAVE_FILE_NAME);
        List<Line> statistic;
        BinaryFormatter formatter = new BinaryFormatter();
        FileStream stream = File.Open(fileName,
FileMode.Open);

        statistic = formatter.Deserialize(stream) as
List<Line>;

        stream.Close();
        stream.Dispose();
        _cachedStats = statistic;
        return statistic;
    }

    public static void SaveStatistic(List<Line> statistic)
    {
        using (FileStream stream =
File.Create(FileNameCombine(SAVE_FILE_NAME)))
        {
            BinaryFormatter formatter = new
BinaryFormatter();

            formatter.Serialize(stream, statistic);
            stream.Close();
        }
        _cachedStats = statistic;
    }

    private static string FileNameCombine(string
saveFileName)
    {

```

					ПГУ1.090301.10.001 ПЗ	Лист
						48
Изм.	Лист	№ докум.	Подпись	Дата		

```

        var filename =
Path.Combine(Application.persistentDataPath, saveFileName);

        return filename;
    }

    public static bool IsThereAreSaveData()
    {
        if (!File.Exists(FileNameCombine(SAVE_FILE_NAME)))
        {
            return false;
        }
        return true;
    }

    public static string ConvertTime(float time)
    {
        int _seconds = (int)time % 60;
        int _minutes = (int)time / 60;
        int _hours = _minutes / 60;

        if (_hours == 0)
        {
            if (_minutes == 0)
            {
                return $"00:00:{_seconds}";
            }
            else
            {
                return $"00:{_minutes}:{_seconds}";
            }
        }
        else
        {

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		49

```

        return $"({_hours}:{_minutes}:{_seconds}";
    }
}

[Serializable]
public struct Line
{
    public Line(float score, float time, float
obstaclesCount)
    {
        Score = score;
        Time = time;
        ObstaclesCount = obstaclesCount;
    }

    public float Score { get; }
    public float Time { get; }
    public float ObstaclesCount { get; }
}
}
}

```

Файл *Tutorial.cs*

```

using RoadFresh.Stats;
using System.Collections;
using UnityEngine;

public class Tutorial : MonoBehaviour
{
    [SerializeField]
    private GameObject gameLoopObject;
    [SerializeField]
    private GameObject movementPC;

```

					ПГУ1.090301.10.001 ПЗ	Лист
						50
Изм.	Лист	№ докум.	Подпись	Дата		

```

[SerializeField]
private GameObject movementAndroid;
[SerializeField]
private GameObject beepPC;
[SerializeField]
private GameObject beepAndroid;
private RuntimePlatform platform;

private const int INIT_DELAY = 0;
private const int FIRST_DELAY = 3;
private const int SECOND_DELAY = 4;
private const int THIRD_DELAY = 5;

public bool IsActive { get; private set; } = false;

private void Start()
{
    TryToStartTutorial();
}

public void TryToStartTutorial()
{
    platform = Application.platform;
    if (!Statistic.IsThereAreSaveData())
        StartCoroutine(StartTutorial(platform));
}

private IEnumerator StartTutorial(RuntimePlatform platform)
{
    IsActive = true;
    switch (platform)
    {

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		51

```

        case RuntimePlatform.Android:

StartCoroutine(TutorialSequence(movementAndroid, beepAndroid,
gameLoopObject));

            break;

        case RuntimePlatform.WindowsPlayer:

            StartCoroutine(TutorialSequence(movementPC,
beepPC, gameLoopObject));

            break;

        case RuntimePlatform.WindowsEditor:

            StartCoroutine(TutorialSequence(movementPC,
beepPC, gameLoopObject));

            break;

    }

    yield return null;

}

private IEnumerator TutorialSequence(GameObject movement,
GameObject beep, GameObject gameLoop)
{

    yield return new WaitForSeconds(INIT_DELAY);
    movement.SetActive(true);
    yield return new WaitForSeconds(FIRST_DELAY);
    movement.SetActive(false);
    beep.SetActive(true);
    yield return new WaitForSeconds(SECOND_DELAY);
    beep.SetActive(false);
    gameLoop.SetActive(true);
    yield return new WaitForSeconds(THIRD_DELAY);
    gameLoop.SetActive(false);
    IsActive = false;
    yield return null;

}

}

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		52

Файл *UnityRigs.cs*

```
using UnityEngine;

using UnityEngine.Animations.Rigging;

namespace RoadFresh.View
{
    public class UnityRigs : MonoBehaviour
    {
        [SerializeField]
        private RigBuilder _rigBuilder;
        [SerializeField]
        private Transform _unityRigsTransform;
        [SerializeField]
        private Animator _characterAnimator;

        public RigBuilder RigBuilder
        {
            get { return _rigBuilder; }
        }

        public Transform UnityRigsTransform
        {
            get { return _unityRigsTransform; }
        }

        public Animator CharacterAnimator
        {
            get { return _characterAnimator; }
        }
    }
}
```

					ПГУ1.090301.10.001 ПЗ	Лист
						53
Изм.	Лист	№ докум.	Подпись	Дата		

Файл *AddBackgroundCollision.cs*

```
using Entitas.Unity;
using UnityEngine;

namespace RoadFresh.GameLoop.Collisions
{
    public class AddBackgroundCollision : CollisionBase
    {
        private void OnTriggerEnter(Collider other)
        {
            if (HasLinkedEntity(other.gameObject))
            {
                var entity =
                (GameEntity)other.gameObject.GetEntityLink().entity;
                if (entity.isBackground)
                {
                    var collisionEntity =
                Contexts.sharedInstance.game.CreateEntity();

                collisionEntity.AddAddBackgroundCollision(other.gameObject,
                gameObject);
                }
            }
        }
    }
}
```

Файл *AddRoadCollision.cs*

```
using Entitas.Unity;
using UnityEngine;

namespace RoadFresh.GameLoop.Collisions
{
```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		54

```

public class AddRoadCollision : CollisionBase
{
    private void OnTriggerEnter(Collider other)
    {
        if (HasLinkedEntity(other.gameObject))
        {
            var entity =
(GameEntity)other.gameObject.GetEntityLink().entity;
            if (entity != null && entity.isRoad)
            {
                var collisionEntity =
Contexts.sharedInstance.game.CreateEntity();

collisionEntity.AddAddRoadCollision(other.gameObject, gameObject);
            }
        }
    }
}

```

Файл *CollisionBase.cs*

```

using Entitas.Unity;
using UnityEngine;

namespace RoadFresh.GameLoop.Collisions
{
    public class CollisionBase : MonoBehaviour
    {
        protected bool HasLinkedEntity(GameObject gameObject)
        {
            var entityLink = gameObject.GetEntityLink();
            if (entityLink != null)
                return true;
        }
    }
}

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		55

```

        return false;
    }
}

```

Файл *DeerObstacleCollision.cs*

```

using Entitas.Unity;
using UnityEngine;

namespace RoadFresh.GameLoop.Collisions
{
    public class DeerObstacleCollision : CollisionBase
    {
        private void OnTriggerEnter(Collider other)
        {
            if (HasLinkedEntity(other.gameObject))
            {
                var entity =
(GameEntity)other.gameObject.GetEntityLink().entity;
                if (entity.isPlayer)
                {
                    entity.isFell = true;
                    var deerAnimator =
GetComponent<Animator>();
                    deerAnimator.SetTrigger("Fall");
                }
            }
        }
    }
}

```

Файл *DestroyBackgroundCollision.cs*

```

using Entitas.Unity;

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		56

```

using UnityEngine;

namespace RoadFresh.GameLoop.Collisions
{
    public class DestroyBackgroundCollision : CollisionBase
    {
        private void OnTriggerEnter(Collider other)
        {
            if (HasLinkedEntity(other.gameObject))
            {
                var entity =
(GameEntity)other.gameObject.GetEntityLink().entity;
                if (entity.isBackground)
                {
                    var collisionEntity =
Contexts.sharedInstance.game.CreateEntity();

collisionEntity.AddDestroyBackgroundCollision(other.gameObject,
gameObject);

                }
            }
        }
    }
}

```

Файл *DestroyRoadCollision.cs*

```

using Entitas.Unity;
using UnityEngine;

namespace RoadFresh.GameLoop.Collisions
{
    public class DestroyRoadCollision : CollisionBase
    {
        private void OnTriggerEnter(Collider other)

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		57

```

        {
            if (HasLinkedEntity(other.gameObject))
            {
                var entity =
(GameEntity)other.gameObject.GetEntityLink().entity;
                if (entity.isRoad)
                {
                    var collisionEntity =
Contexts.sharedInstance.game.CreateEntity();

collisionEntity.AddDestroyRoadCollision(other.gameObject,
gameObject);
                }
            }
        }
    }
}

```

Файл *ObstacleCollision.cs*

```

using Entitas.Unity;
using UnityEngine;

namespace RoadFresh.GameLoop.Collisions
{
    public class ObstacleCollision : CollisionBase
    {
        private void OnTriggerEnter(Collider other)
        {
            if (HasLinkedEntity(other.gameObject))
            {
                var entity =
(GameEntity)other.gameObject.GetEntityLink().entity;
                if (entity.isPlayer)
                    entity.isFell = true;
            }
        }
    }
}

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		58

```

    }

    }

}

```

Файл *ObstaclePassedCollision.cs*

```

using Entitas.Unity;
using UnityEngine;

namespace RoadFresh.GameLoop.Collisions
{
    public class ObstaclePassedCollision : CollisionBase
    {
        private void OnTriggerExit(Collider other)
        {
            if (HasLinkedEntity(other.gameObject))
            {
                var entity =
                (GameEntity)other.gameObject.GetEntityLink().entity;
                if (entity.isPlayer)
                {
                    var collisionEntity =
                Contexts.sharedInstance.game.CreateEntity();

                collisionEntity.AddObstaclePassedCollision(other.gameObject,
                gameObject);
            }
        }
    }
}

```

Файл *TeaCupCollision.cs*

```

using Entitas.Unity;
using UnityEngine;

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		59

```

namespace RoadFresh.GameLoop.Collisions
{
    public class TeaCupCollision : CollisionBase
    {
        [SerializeField]
        private float _heatAmount;
        private void OnTriggerEnter(Collider other)
        {
            if (HasLinkedEntity(other.gameObject))
            {
                var entity =
(GameEntity)other.gameObject.GetEntityLink().entity;
                if (entity.isPlayer)
                {
                    var playerHeat = entity.heat.value +
_heatAmount;

                    entity.ReplaceHeat(playerHeat);
                    entity.slurpAudioSource.value.Play();
                    Destroy(gameObject);
                }
            }
        }
    }
}

```

Файл *InputActions.cs*

```

//-----
-----

// <auto-generated>

//      This code was auto-generated by
com.unity.inputsystem:InputActionCodeGenerator

//      version 1.4.4

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		60

```

//      from Assets/Scripts/Input/InputActions.inputactions
//
//      Changes to this file may cause incorrect behavior and
will be lost if
//      the code is regenerated.
// </auto-generated>
//-----
-----

using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine.InputSystem;
using UnityEngine.InputSystem.Utilities;

public partial class @InputActions : IInputActionCollection2,
IDisposable
{
    public InputActionAsset asset { get; }
    public @InputActions()
    {
        asset = InputActionAsset.FromJson(@"{
""name"": ""InputActions"",
""maps"": [
    {
        ""name"": ""Player"",
        ""id"": ""876e9367-e4de-49d3-9fcf-16b10fddb20"",
        ""actions"": [
            {
                ""name"": ""Strafe"",
                ""type"": ""Value"",
                ""id"": ""440b5a45-e0eb-45c2-a938-
fc5795ff25d0"",
                ""expectedControlType"": ""Axis"",

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		61

```

        "processors": "",
        "interactions": "",
        "initialStateCheck": true
    },
    {
        "name": "Beep",
        "type": "Button",
        "id": "65b9aa39-a8c0-4d5f-afe7-
eeeea1b5f82b",
        "expectedControlType": "Button",
        "processors": "",
        "interactions": "",
        "initialStateCheck": false
    }
],
"bindings": [
    {
        "name": "1D Axis",
        "id": "d7d37cbe-e66d-4c7c-b5e5-
9e7d285600e1",
        "path": "1DAxis",
        "interactions": "",
        "processors": "",
        "groups": "",
        "action": "Strafe",
        "isComposite": true,
        "isPartOfComposite": false
    },
    {
        "name": "negative",
        "id": "595527ea-ac8b-41db-ad3b-
6c65060b878a",
        "path": "<Keyboard>/a",

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		62

```

        "interactions": "",
        "processors": "",
        "groups": "Mouse & Keyboard",
        "action": "Strafe",
        "isComposite": false,
        "isPartOfComposite": true
    },
    {
        "name": "positive",
        "id": "efc2d256-5bc9-45d6-b6e8-
2dfa4148dcad",
        "path": "<Keyboard>/d",
        "interactions": "",
        "processors": "",
        "groups": "Mouse & Keyboard",
        "action": "Strafe",
        "isComposite": false,
        "isPartOfComposite": true
    },
    {
        "name": "1D Axis Android",
        "id": "c6fa3dbd-502b-4bc6-969d-
cd44c8882ad2",
        "path": "1DAxis",
        "interactions": "",
        "processors": "",
        "groups": "",
        "action": "Strafe",
        "isComposite": true,
        "isPartOfComposite": false
    },
    {
        "name": "negative",

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		63

```

e3c78926bcea"",
        ""id"": ""61a5672d-ea15-4d32-ae2-
        ""path"": ""<AndroidJoystick>/stick/left"",
        ""interactions"": "",
        ""processors"": "",
        ""groups"": ""Android"",
        ""action"": ""Strafe"",
        ""isComposite"": false,
        ""isPartOfComposite"": true
    },
    {
        ""name"": ""positive"",
        ""id"": ""66d76f8f-816d-4a09-8a5b-
feea25a13db9"",
        ""path"":
""<AndroidJoystick>/stick/right"",
        ""interactions"": "",
        ""processors"": "",
        ""groups"": ""Android"",
        ""action"": ""Strafe"",
        ""isComposite"": false,
        ""isPartOfComposite"": true
    },
    {
        ""name"": "",
        ""id"": ""7c19a8b7-1d35-4f20-8532-
279dfd3d6f7d"",
        ""path"": ""<Keyboard>/space"",
        ""interactions"": "",
        ""processors"": "",
        ""groups"": ""Mouse & Keyboard"",
        ""action"": ""Beep"",
        ""isComposite"": false,
        ""isPartOfComposite"": false
    }

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		64

```

        }
    ]
},
{
    ""name"": ""GUI"",
    ""id"": ""e252ac6d-5b11-43dc-b4ca-1166e3c5fc3f"",
    ""actions"": [
        {
            ""name"": ""Escape"",
            ""type"": ""Button"",
            ""id"": ""dd256637-3179-46d5-aff3-
d093d80c6cb4"",
            ""expectedControlType"": ""Button"",
            ""processors"": "",
            ""interactions"": "",
            ""initialStateCheck"": false
        }
    ],
    ""bindings"": [
        {
            ""name"": "",
            ""id"": ""7dd06625-81a5-49db-a989-
978dab9b59d1"",
            ""path"": ""<Keyboard>/escape"",
            ""interactions"": "",
            ""processors"": "",
            ""groups"": ""Mouse & Keyboard"",
            ""action"": ""Escape"",
            ""isComposite"": false,
            ""isPartOfComposite"": false
        }
    ]
}

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		65

```

],
""controlSchemes"": [
    {
        ""name"": ""Mouse & Keyboard"",
        ""bindingGroup"": ""Mouse & Keyboard"",
        ""devices"": [
            {
                ""devicePath"": ""<Keyboard>"",
                ""isOptional"": false,
                ""isOR"": false
            },
            {
                ""devicePath"": ""<Mouse>"",
                ""isOptional"": false,
                ""isOR"": false
            }
        ]
    },
    {
        ""name"": ""Android"",
        ""bindingGroup"": ""Android"",
        ""devices"": [
            {
                ""devicePath"": ""<AndroidJoystick>"",
                ""isOptional"": false,
                ""isOR"": false
            },
            {
                ""devicePath"": ""<Touchscreen>"",
                ""isOptional"": false,
                ""isOR"": false
            }
        ]
    }
]

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		66

```

        ]
    }
}

}");

    // Player
    m_Player = asset.FindActionMap("Player",
throwIfNotFound: true);

    m_Player_Strafe = m_Player.FindAction("Strafe",
throwIfNotFound: true);

    m_Player_Beep = m_Player.FindAction("Beep",
throwIfNotFound: true);

    // GUI
    m_GUI = asset.FindActionMap("GUI", throwIfNotFound:
true);

    m_GUI_Escape = m_GUI.FindAction("Escape",
throwIfNotFound: true);
}

public void Dispose()
{
    UnityEngine.Object.Destroy(asset);
}

public InputBinding? bindingMask
{
    get => asset.bindingMask;
    set => asset.bindingMask = value;
}

public ReadOnlyArray<InputDevice>? devices
{
    get => asset.devices;
    set => asset.devices = value;
}

```

					ПГУ1.090301.10.001 ПЗ	Лист
						67
Изм.	Лист	№ докум.	Подпись	Дата		

```

        public ReadOnlyArray<InputControlScheme> controlSchemes =>
asset.controlSchemes;

        public bool Contains(InputAction action)
        {
            return asset.Contains(action);
        }

        public IEnumerator<InputAction> GetEnumerator()
        {
            return asset.GetEnumerator();
        }

        IEnumerator IEnumerable.GetEnumerator()
        {
            return GetEnumerator();
        }

        public void Enable()
        {
            asset.Enable();
        }

        public void Disable()
        {
            asset.Disable();
        }

        public IEnumerable<InputBinding> bindings =>
asset.bindings;

        public InputAction FindAction(string actionNameOrId, bool
throwIfNotFound = false)

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		68

```

        {
            return asset.FindAction(actionNameOrId,
throwIfNotFound);
        }

        public int FindBinding(InputBinding bindingMask, out
InputAction action)
        {
            return asset.FindBinding(bindingMask, out action);
        }

// Player
private readonly InputActionMap m_Player;
private IPlayerActions m_PlayerActionsCallbackInterface;
private readonly InputAction m_Player_Strafe;
private readonly InputAction m_Player_Beep;
public struct PlayerActions
{
    private @InputActions m_Wrapper;
    public PlayerActions(@InputActions wrapper) { m_Wrapper
= wrapper; }

    public InputAction @Strafe =>
m_Wrapper.m_Player_Strafe;

    public InputAction @Beep => m_Wrapper.m_Player_Beep;

    public InputActionMap Get() { return
m_Wrapper.m_Player; }

    public void Enable() { Get().Enable(); }
    public void Disable() { Get().Disable(); }
    public bool enabled => Get().enabled;

    public static implicit operator
InputActionMap(PlayerActions set) { return set.Get(); }

    public void SetCallbacks(IPlayerActions instance)
    {

        if (m_Wrapper.m_PlayerActionsCallbackInterface !=
null)

        {

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		69

```

        @Strafe.started -=
m_Wrapper.m_PlayerActionsCallbackInterface.OnStrafe;

        @Strafe.performed -=
m_Wrapper.m_PlayerActionsCallbackInterface.OnStrafe;

        @Strafe.canceled -=
m_Wrapper.m_PlayerActionsCallbackInterface.OnStrafe;

        @Beep.started -=
m_Wrapper.m_PlayerActionsCallbackInterface.OnBeep;

        @Beep.performed -=
m_Wrapper.m_PlayerActionsCallbackInterface.OnBeep;

        @Beep.canceled -=
m_Wrapper.m_PlayerActionsCallbackInterface.OnBeep;
    }

    m_Wrapper.m_PlayerActionsCallbackInterface =
instance;

    if (instance != null)
    {
        @Strafe.started += instance.OnStrafe;
        @Strafe.performed += instance.OnStrafe;
        @Strafe.canceled += instance.OnStrafe;
        @Beep.started += instance.OnBeep;
        @Beep.performed += instance.OnBeep;
        @Beep.canceled += instance.OnBeep;
    }
}

}

public PlayerActions @Player => new PlayerActions(this);

// GUI
private readonly InputActionMap m_GUI;
private IGUIActions m_GUIActionsCallbackInterface;
private readonly InputAction m_GUI_Escape;
public struct GUIActions
{
    private @InputActions m_Wrapper;

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		70

```

        public GUIActions(@InputActions wrapper) { m_Wrapper =
wrapper; }

        public InputAction @Escape => m_Wrapper.m_GUI_Escape;
        public InputActionMap Get() { return m_Wrapper.m_GUI; }
        public void Enable() { Get().Enable(); }
        public void Disable() { Get().Disable(); }
        public bool enabled => Get().enabled;

        public static implicit operator
InputActionMap(GUIActions set) { return set.Get(); }

        public void SetCallbacks(IGUIActions instance)
        {
            if (m_Wrapper.m_GUIActionsCallbackInterface !=
null)
            {
                @Escape.started -=
m_Wrapper.m_GUIActionsCallbackInterface.OnEscape;

                @Escape.performed -=
m_Wrapper.m_GUIActionsCallbackInterface.OnEscape;

                @Escape.canceled -=
m_Wrapper.m_GUIActionsCallbackInterface.OnEscape;
            }
            m_Wrapper.m_GUIActionsCallbackInterface = instance;
            if (instance != null)
            {
                @Escape.started += instance.OnEscape;
                @Escape.performed += instance.OnEscape;
                @Escape.canceled += instance.OnEscape;
            }
        }
    }

    public GUIActions @GUI => new GUIActions(this);
    private int m_MouseKeyboardSchemeIndex = -1;
    public InputControlScheme MouseKeyboardScheme
    {

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		71

```

        get
        {
            if (m_MouseKeyboardSchemeIndex == -1)
m_MouseKeyboardSchemeIndex = asset.FindControlSchemeIndex("Mouse &
Keyboard");

            return
asset.controlSchemes[m_MouseKeyboardSchemeIndex];
        }
    }

    private int m_AndroidSchemeIndex = -1;
    public InputControlScheme AndroidScheme
    {
        get
        {
            if (m_AndroidSchemeIndex == -1)
m_AndroidSchemeIndex = asset.FindControlSchemeIndex("Android");

            return asset.controlSchemes[m_AndroidSchemeIndex];
        }
    }

    public interface IPlayerActions
    {
        void OnStrafe(InputAction.CallbackContext context);
        void OnBeep(InputAction.CallbackContext context);
    }

    public interface IGUIActions
    {
        void OnEscape(InputAction.CallbackContext context);
    }
}

Файл InputHandler.cs

using UnityEngine;

namespace RoadFresh.Controls.Input
{

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		72

```

public class InputHandler : MonoBehaviour
{
    private InputActions _inputActions;
    private Contexts _contexts;

    private void Awake()
    {
        _contexts = Contexts.sharedInstance;
        _inputActions = new InputActions();
    }

    private void OnEnable()
    {
        _inputActions.Player.Strafe.performed +=
callbackContext => SetStrafe(callbackContext.ReadValue<float>());
        _inputActions.Player.Strafe.canceled +=
callbackContext => StopStrafe();
        _inputActions.Player.Beep.performed +=
callbackContext => SetBeep();
        _inputActions.Player.Beep.canceled +=
callbackContext => StopBeep();
        _inputActions.Enable();
    }

    private void OnDisable()
    {
        _inputActions.Disable();
    }

    private void SetStrafe(float direction)
    {
        _contexts.game.ReplaceStrafeInput(direction);
    }
}

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		73

```

        private void StopStrafe()
        {
            _contexts.game.ReplaceStrafeInput(0);
        }

        private void SetBeep()
        {
            _contexts.game.ReplaceBeep(true);
        }

        private void StopBeep()
        {
            _contexts.game.ReplaceBeep(false);
        }
    }
}

```

Файл *BackgroundSetup.cs*

```

using Entitas.CodeGeneration.Attributes;
using System.Collections.Generic;
using UnityEngine;

```

```

namespace RoadFresh.Initialize.Setups
{
    [CreateAssetMenu]
    [Game, Unique]
    public class BackgroundSetup : ScriptableObject
    {
        public List<GameObject> BackgroundPrefabs;
        public float BackgroundSpeed;
        public Vector3 InitializePosition;
    }
}

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		74

```

        public Vector3 AddPosition;
    }
}

```

Файл *GameData.cs*

```

using Entitas.CodeGeneration.Attributes;
using UnityEngine;

namespace RoadFresh.Initialize.Setups
{
    [CreateAssetMenu]
    [Game, Unique]
    public class GameData : ScriptableObject
    {
        public float DeltaSpeed;
        public float DeltaInterval;
        public float ScorePerObstacle;
        public float GlobalGameSpeed;
        public float Score;
        public float ObstaclePassed;
    }
}

```

Файл *PlayerSetup.cs*

```

using Entitas.CodeGeneration.Attributes;
using UnityEngine;

namespace RoadFresh.Initialize.Setups
{
    [CreateAssetMenu]
    [Game, Unique]
    public class PlayerSetup : ScriptableObject
    {
        public GameObject Prefab;
    }
}

```

					ПГУ1.090301.10.001 ПЗ	Лист
						75
Изм.	Лист	№ докум.	Подпись	Дата		

```

        public float StrafeSpeed;
        public float HeatLosePerTick;
        public Vector3 InitialPosition;
        public AudioClip beepSound;
        public AudioClip teaSlurpSound;
        public AudioClip engineSound;
        public AudioClip crashSound;
    }
}

```

Файл *RoadSetup.cs*

```

using Entitas.CodeGeneration.Attributes;
using System.Collections.Generic;
using UnityEngine;

namespace RoadFresh.Initialize.Setups
{
    [CreateAssetMenu]
    [Game, Unique]
    public class RoadSetup : ScriptableObject
    {
        public List<GameObject> roadPrefabs;
        public float RoadSpeed;
        public Vector3 InitializePosition;
        public Vector3 AddPosition;
    }
}

```

Файл *UISetup.cs*

```

using Entitas.CodeGeneration.Attributes;
using UnityEngine;

namespace RoadFresh.Initialize.Setups
{

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		76

```

        [CreateAssetMenu]
        [Game, Unique]
        public class UISetup : ScriptableObject
        {
            public GameObject GameUI;
        }
    }

```

Файл *GameOverlay.cs*

```

using System;
using TMPro;
using UnityEngine;
using UnityEngine.UI;

namespace RoadFresh.View.UI
{
    public class GameOverlay : MonoBehaviour
    {
        [SerializeField]
        private TextMeshProUGUI _scoreCounter;
        [SerializeField]
        private TextMeshProUGUI _obstaclesCounter;
        [SerializeField]
        private Image heatImage;
        [SerializeField]
        private GameObject heatBarObject;
        [SerializeField]
        private GameObject joystick;
        [SerializeField]
        private GameObject beepButton;
        [SerializeField]
        private Tutorial tutorial;
    }
}

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		77

```

public Tutorial Tutorial { get => tutorial; }

private void Awake()
{
    if (Application.platform !=
RuntimePlatform.Android)
        HideMobileControlls();
}

public void UpdateScore(float value)
{
    _scoreCounter.text = Convert.ToString((int)value);
}

public void UpdateObstaclesCount(float value)
{
    _obstaclesCounter.text =
Convert.ToString((int)value);
}

public void UpdateHeat(float value)
{
    heatImage.fillAmount = value;
}

public void HideGameControlls()
{
    heatBarObject.SetActive(false);
    HideMobileControlls();
}

public void HideMobileControlls()
{

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		78

```

        joystick.SetActive(false);
        beepButton.SetActive(false);
    }

    public void OnBeepButtonPress()
    {
        Contexts.sharedInstance.game.ReplaceBeep(true);
    }

    public void OnBeepButtonRelease()
    {
        Contexts.sharedInstance.game.ReplaceBeep(false);
    }
}
}

```

Файл *GameOverOverlay.cs*

```

using TMPro;
using UnityEngine;
using UnityEngine.SceneManagement;

namespace RoadFresh.View.UI
{
    public class GameOverOverlay : MonoBehaviour
    {
        [SerializeField]
        private GameObject loseLabelObject;
        [SerializeField]
        private GameObject scoreRecordObject;
        [SerializeField]
        private GameObject restartButtonObject;
        [SerializeField]

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		79

```

private GameObject mainMenuButtonObject;

private AudioSource _audioSource;
private void Awake()
{
    _audioSource = GetComponent<AudioSource>();
}

public void EnableGameOverUI()
{
    loseLabelObject.SetActive(true);
    restartButtonObject.SetActive(true);
    mainMenuButtonObject.SetActive(true);
}

public void EnableRecordLable()
{
    scoreRecordObject.SetActive(true);
}

public void OnRestartButtonClick()
{
    Contexts.sharedInstance.game.isSceneReloading =
true;
}

public void OnMainMenuButtonClick()
{
    Contexts.sharedInstance.game.isReturnMenu = true;
}

public void OnPointerDown()
{

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		80

```

        _audioSource.Play();
    }

}

```

Файл *GameUI.cs*

```

using Entitas.CodeGeneration.Attributes;
using UnityEngine;

namespace RoadFresh.View.UI
{
    [Game, Unique]
    public class GameUI : MonoBehaviour
    {
        [SerializeField]
        private GameOverlay _gameOverlay;
        [SerializeField]
        private GameOverOverlay _gameOverOverlay;

        public GameOverlay GameOverlay
        {
            get { return _gameOverlay; }
        }

        public GameOverOverlay GameOverOverlay
        {
            get { return _gameOverOverlay; }
        }
    }
}

```

Файл *LoadStats.cs*

```

using System.Collections.Generic;
using TMLPro;

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		81

```

using UnityEngine;

namespace RoadFresh.Stats
{
    public class LoadStats : MonoBehaviour
    {
        [SerializeField]
        private int lineIndex;
        [SerializeField]
        private TextMeshProUGUI score;
        [SerializeField]
        private TextMeshProUGUI obstacles;
        [SerializeField]
        private TextMeshProUGUI time;

        private void OnEnable()
        {
            if (Statistic.CachedStats != null &&
Statistic.CachedStats.Count >= lineIndex)
            {
                SetStats(Statistic.CachedStats);
            }
            else if (Statistic.IsThereAreSaveData())
            {
                SetStats(Statistic.LoadStatistic());
            }
        }

        private void SetStats(List<Statistic.Line> statistic)
        {
            score.text = statistic[lineIndex -
1].Score.ToString();

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		82

```

        obstacles.text = statistic[lineIndex -
1].ObstaclesCount.ToString();

        time.text =
Statistic.ConvertTime(statistic[lineIndex - 1].Time);

    }

}

```

Файл *MenuUI.cs*

```

using UnityEngine;
using UnityEngine.SceneManagement;

namespace RoadFresh.View.UI
{
    public class MenuUI : MonoBehaviour
    {
        [SerializeField]
        private GameObject menuObject;
        [SerializeField]
        private GameObject scoreTableObject;
        [SerializeField]
        private GameObject quitGameButtonObject;

        private AudioSource _clickAudioSource;

        private const string GAME_SCENE = "Game";
        private void Awake()
        {
            if (Application.platform ==
RuntimePlatform.Android)

                quitGameButtonObject.SetActive(false);

            _clickAudioSource = GetComponent<AudioSource>();
        }
    }
}

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		83

```

        public void OnButtonClick()
        {
            _clickAudioSource.Play();
        }

        public void OnPlayButtonClick()
        {
            SceneManager.LoadSceneAsync (GAME_SCENE);
        }

        public void OnScoreButtonClick()
        {
            menuObject.SetActive (false);
            scoreTableObject.SetActive (true);
        }

        public void OnMenuReturnButtonClick()
        {
            menuObject.SetActive (true);
            scoreTableObject.SetActive (false);
        }

        public void OnQuitButtonClick()
        {
            Application.Quit();
        }

    }
}

```

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		84

Приложение В

(обязательное)

Мобильная игра в жанре «раннер» на платформе *Unity3D*

Презентация

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		85

Тема работы: «Мобильная игра в жанре «раннер» на платформе Unity3D»

Автор работы: студентка группы 19ВВП1 Корнаухова А.С.
Научный руководитель: к.т.н., доцент каф. ВТ ПГУ Юрова О.В.

Пенза 2023

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		86

Актуальность работы

Смартфоны стали неотъемлемой частью нашей жизни, а игры для мобильных устройств занимают лидирующие позиции в мировой индустрии развлечений. Рынок мобильных игр перспективен и быстро растет благодаря увеличению числа пользователей смартфонов, улучшению качества графики и звука в играх, а также возможности играть в любое время и в любом месте. Большинство игр для мобильных устройств бесплатны, но содержат внутриигровые покупки, что позволяет разработчикам получать доход. Разработка игр для смартфонов является актуальной и перспективной областью, которая привлекает все больше внимания и инвестиций.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		87

Цель и объект работы

Данная дипломная работа посвящена разработке игрового приложения в жанре «раннер», которое предоставляет игрокам увлекательное времяпрепровождение и целью которого является проехать как можно большее расстояние, избегая препятствий и собирая кофе. В игре реализовано управление мотоциклом, сбор кофейных кружек, избегание препятствий на дороге и система подсчета очков. В рамках работы было выполнено взаимодействие с платформой Unity, проектирование архитектуры игрового приложения, разработка логики функций и тестирование всех заявленных функциональностей.



Аналоги

Subway Surfers

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ1.090301.10.001 ПЗ

Лист

89

Аналоги

Temple Run



					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		90



Стек технологий

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		91

Диаграмма вариантов использования

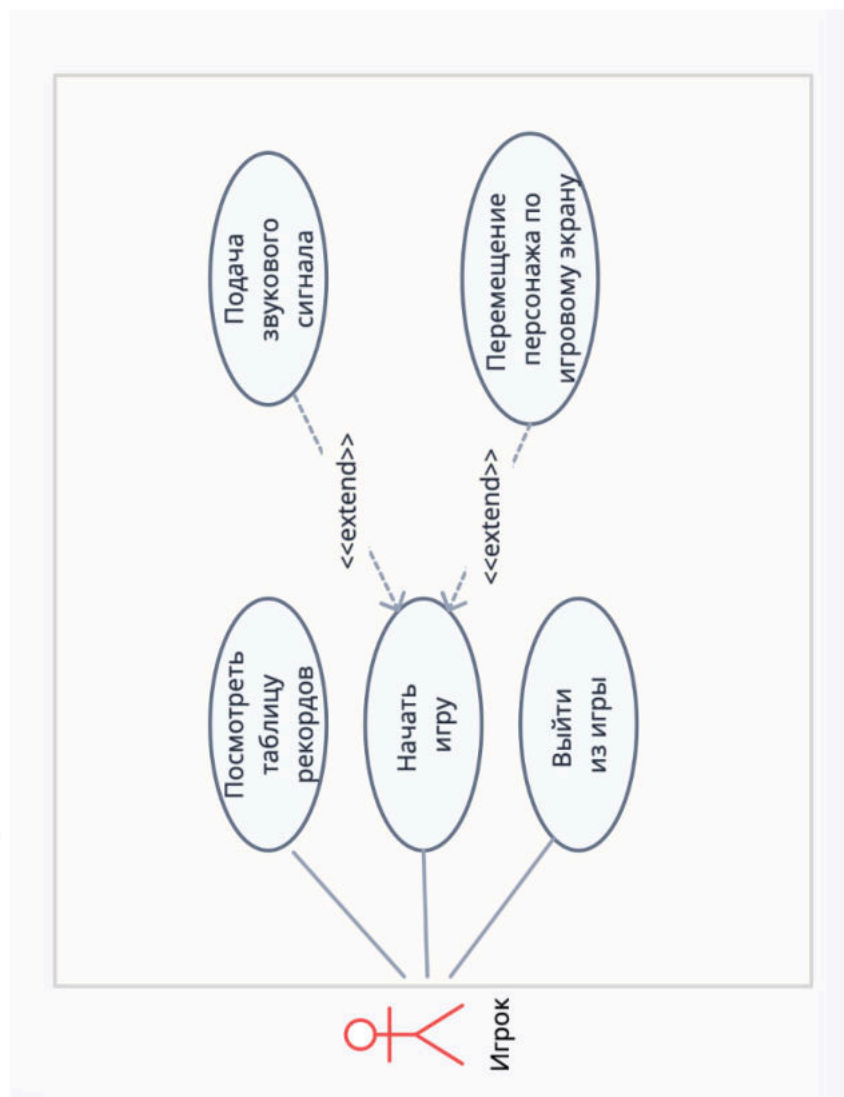
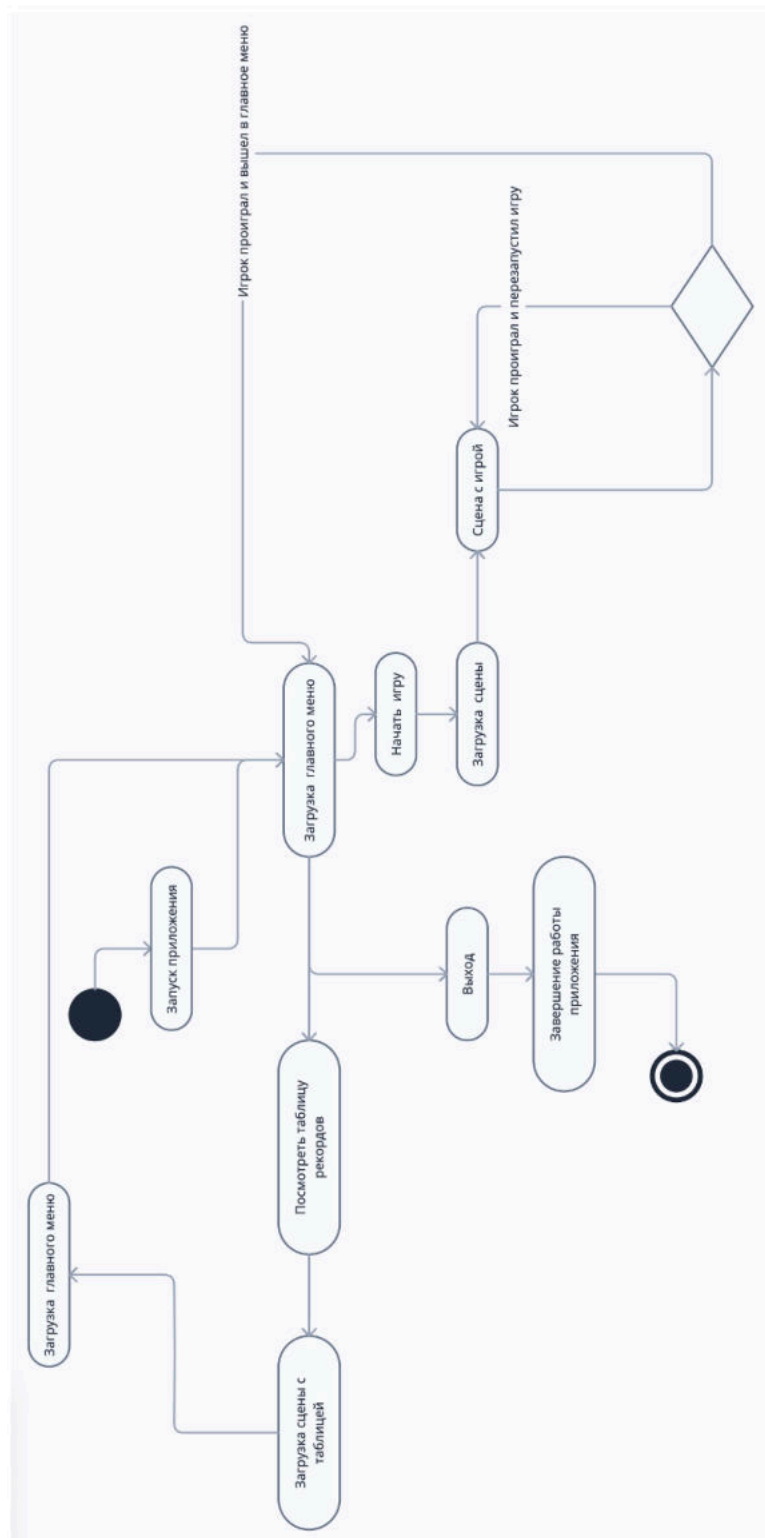
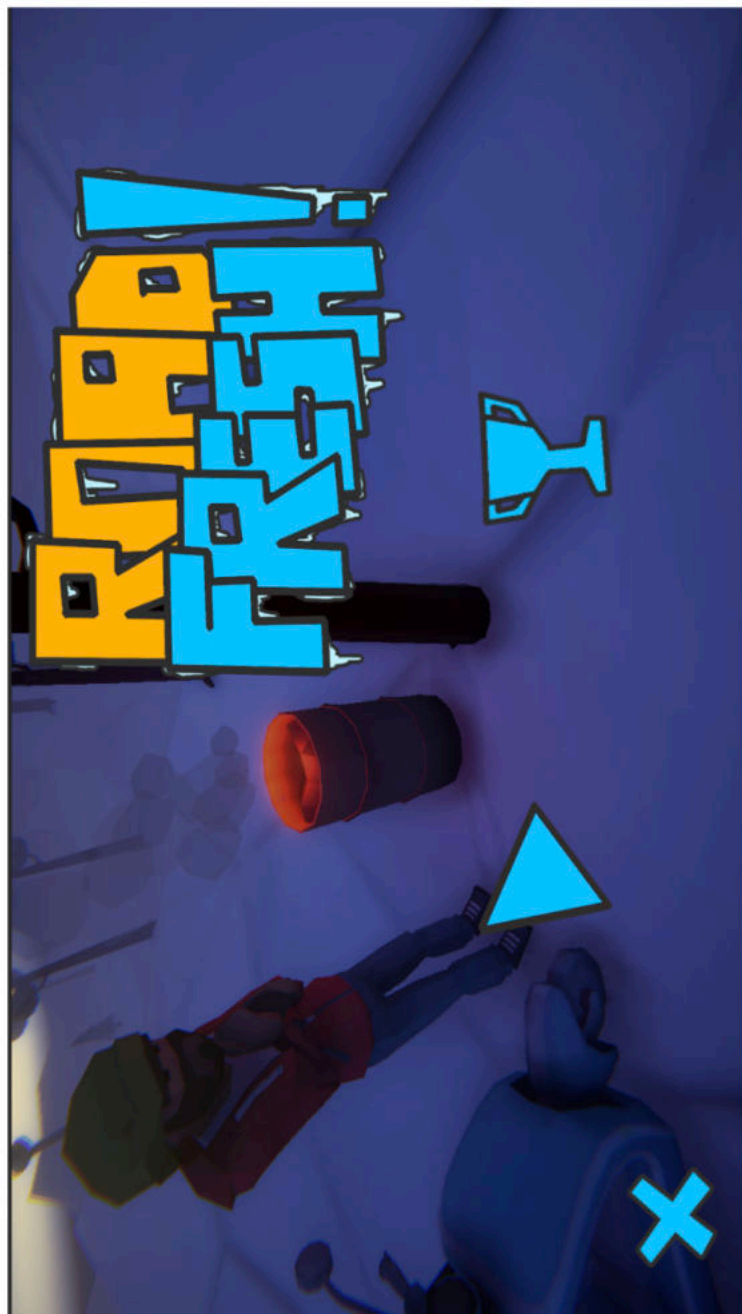


Диаграмма деятельности



Демонстрация



Изм.	Лист	№ докум.	Подпись	Дата

ПГУ1.090301.10.001 ПЗ

Лист

94

Заключение

В рамках работы было создано игровое приложение на основе среды Unity, которое может работать на разных платформах. Был проведен анализ существующих приложений и найдены готовые решения, а также проведено проектирование и функциональное тестирование. Разработанное приложение соответствует всем требованиям качества программного обеспечения и готово к использованию.

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		95

Спасибо за внимание!

					ПГУ1.090301.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		96

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

«Утверждаю»

Заведующий кафедрой

М.А. Митрохин

« 26 » апреля 20 23 г.

**ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ**

1. Студент Корнаухова Анастасия Сергеевна гр 19ВВП1 факультета вычислительной техники.

2. Тема работы «Мобильная игра в жанре «раннер» на платформе Unity3D»

Тема утверждена приказом ПГУ № 458/о от "26" апреля 2023 г.

3. Руководитель бакалаврской работы к.т.н., доцент, каф. ВТ ПГУ Юрова О.В.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

Реализовать игровое приложение на платформе *Unity3D*, предоставляющее следующее:

- возможность передвижения персонажа влево/вправо;
- контроль за уровнем энергии;
- возможность пополнения энергии;
- систему подсчета очков рейтинга занесения их в таблицу рекордов.

Также провести тестирование и отладку каждого элемента приложения.

5. Перечень подлежащих разработке вопросов

Требуется произвести анализ предметных областей: изучить существующие аналоги, выделить их достоинства и недостатки. Выбрать язык программирования, который подойдет для написания логики приложения. Определить архитектуру приложения и вспомогательные средства разработки.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Изучение предметной области	50	26.04.2023- 30.04.2023	
2.	Анализ требований	10	01.05.2023- 02.05.2023	
3.	Выбор технологий, применяемых при проектировании и разработке	50	03.05.2023- 07.05.2023	
4.	Описание алгоритмов	40	08.05.2023- 11.05.2023	
5.	Реализация игрового приложения на платформе Unity3D	150	12.05.2023- 29.05.2023	
6.	Оформление пояснительной записки	24	30.05.2023- 31.05.2023	

Дата выдачи задания "26" апреля 2023 г.

Руководитель бакалаврской работы _____ Юрова О.В.
(подпись, дата)

Задание к исполнению принял студент _____ Корнаухова А.С.
(подпись, дата)

Работу к защите допустить
Декан факультета _____ Фионова Л.Р.
(подпись, дата)