

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Хакасский технический институт – филиал ФГАОУ ВО
«Сибирский федеральный университет»

Кафедра прикладной информатики, математики и естественно-научных
дисциплин

УТВЕРЖДАЮ
Заведующий кафедрой
_____ Е. Н. Скуратенко
подпись
«____» _____ 2022 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.03 Прикладная информатика

Разработка демо-версии компьютерной игры в жанре «Инди»

Руководитель _____ доцент, канд. физ.-мат. наук М. А. Бурева
подпись, дата

Выпускник _____ К. В. Танков
подпись, дата

Консультанты
по разделам:

Экономический _____ Е. Н. Скуратенко
подпись, дата

Нормоконтролер _____ В. И. Кокова
подпись, дата

Абакан 2022

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Хакасский технический институт – филиал ФГАОУ ВО
«Сибирский федеральный университет»

Кафедра прикладной информатики, математики и естественно-научных
дисциплин

УТВЕРЖДАЮ
Заведующий кафедрой
_____ Е. Н. Скуратенко
подпись
« ____ » _____ 2022 г.

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы**

Абакан 2022

Студенту Танкову Константину Вячеславовичу

Группа ХБ 18-03

Направление 09.03.03 Прикладная информатика

Тема выпускной квалификационной работы: Разработка демо-версии компьютерной игры в жанре "Инди".

Утверждена приказом по институту № 208 от 14.04.2022 г.

Руководитель ВКР: М. А. Буреева, доцент, канд. физ.-мат. наук, ХТИ – филиал СФУ

Исходные данные для ВКР: заказ ХТИ – филиала СФУ.

Перечень разделов ВКР:

1. Анализ процесса создания компьютерной игры в рамках обучения по направлению подготовки 09.03.03 Прикладная информатика.
2. Разработка демо-версии компьютерной игры.
3. Оценка экономической эффективности проекта.

Перечень графического материала: нет

Руководитель ВКР _____ М. А. Буреева
подпись

Задание принял к исполнению _____ К. В. Танков
подпись

«14» апреля 2022 г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка демо-версии игры в жанре «Инди» содержит 77 страниц текстового документа, 51 иллюстрацию, 14 таблиц, 8 формул, 28 использованных источников.

РАЗРАБОТКА, КОМПЬЮТЕРНАЯ ИГРА, UNITY, MICROSOFT, MICROSOFT VISUAL STUDIO CODE, AUDACITY, ИГРОВАЯ ИНДУСТРИЯ, STEAM, ЖАНР.

Объект выпускной квалификационной работы: процесс разработки компьютерных игр.

Предмет выпускной квалификационной работы: процесс разработки компьютерной игры в жанре инди на платформе Unity.

Цель выпускной квалификационной работы заключается в разработке демо-версии компьютерной игры в жанре "Инди".

Создание качественной компьютерной игры требует от разработчика наличия самых разнообразных навыков: от программирования до обработки графики и звука. Большинство этих навыков соответствуют компетенциям выпускника по направлению подготовки 09.03.03 Прикладная информатика.

Первый раздел посвящен анализу предметной области. Рассмотрено соответствие навыков разработки компьютерной игры компетенциям выпускника по направлению подготовки 09.03.03 Прикладная информатика, кратко описана история развития игровой индустрии, проведен обзор подобных программных продуктов, выбрана среда разработки.

Во втором разделе был описан процесс разработки демоверсии компьютерной игры. Она включает три игровых уровня, главное меню и меню выбора уровней.

В третьем разделе проведена оценка экономической эффективности проекта.

SUMMARY

The theme of the graduation thesis is «Development of Demo Version of Indie Genre Game». It consists of 77 pages, 51 figures, 14 tables, 8 formulae, 28 reference items.

DEVELOPMENT, COMPUTER GAME, UNITY, MICROSOFT, MICROSOFT VISUAL STUDIO CODE, AUDACITY, GAMING INDUSTRY, STEAM, GENRE.

The object of the thesis: the development of computer games.

The subject of the thesis: the development of a computer game in the “Indie” genre on the “Unity” platform.

The purpose of the thesis is to develop a demo version of a computer game of the "Indie" genre.

Creating a high-quality computer game requires the developer to have a wide variety of skills: from programming to processing graphics and sound. Most of these skills correspond to the competencies of a graduate of the 09.03.03 “Applied Informatics” Training Program.

The first section is devoted to the analysis of the subject area. The correspondence of computer game development skills to the competencies of a graduate in the field of study 09.03.03 “Applied Informatics” has been considered, the history of the development of the gaming industry has been briefly described, a review of similar software products has been carried out, and a development environment has been selected.

The second section describes the process of developing a demo version of a computer game. It includes three game levels, the main menu and the level selection menu.

The third section assesses the economic efficiency of the project.

English language supervisor

signature, date

N.V. Chezybaeva
(full name)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	8
1 Анализ процесса создания компьютерной игры в рамках обучения по направлению подготовки 09.03.03 Прикладная информатика	9
1.1 Описание соответствия навыков разработки компьютерной игры компетенциям выпускника по направлению подготовки 09.03.03 Прикладная информатика.....	9
1.2 Начало игровой индустрии.....	12
1.3 Жанры компьютерных игр.....	12
1.4 Описание идеи разрабатываемой игры	14
1.5 Описание платформы Steam	14
1.6 Сравнительный анализ и выбор инструментальных средств разработки ..	17
1.7 Моделирование процесса разработки компьютерной игры	24
1.8 Выводы по разделу «Анализ процесса создания компьютерной игры в рамках обучения по направлению подготовки 09.03.03 Прикладная информатика».....	29
2 Разработка демоверсии компьютерной игры.....	29
2.1 Установка программного обеспечения	29
2.2 Разработка игровых уровней	33
2.3 Разработка меню.....	51
2.4 Выводы по разделу «Разработка демоверсии компьютерной игры»	56
3 Оценка экономической эффективности проекта.....	56
3.1 Подбор оборудования и программного обеспечения для реализации проекта.....	56
3.2 Капитальные затраты	60
3.2.1 Затраты на проектирование	60
3.3 Эксплуатационные затраты	64
3.4 Расчет совокупной стоимости по методике ТСО	64
3.5 Оценка рисков проекта	67

3.6 Оценка экономического эффекта от разработки демоверсии.....	68
3.7 Выводы по разделу «Оценка экономической эффективности проекта»....	71
ЗАКЛЮЧЕНИЕ	73
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	74

ВВЕДЕНИЕ

В наше время все больше и больше людей играют в компьютерные игры. Люди играют в компьютерные игры с разными целями. Кто-то хочет просто расслабиться и весело провести время, а кто-то хочет достигнуть определенного уровня и выступать на мировых соревнованиях, при этом не имея специальных навыков на входе. Благодаря тому, что игровая индустрия имеет низкий порог входа, независимые разработчики и огромные компании, специализирующиеся на разработке и издательстве компьютерных игр, могут комфортно себя ощущать на рынке и занимать свою нишу. Вследствие этого на рынке существует несчетное множество игр, и каждый разработчик может реализовать даже свою самую смелую идею, а каждый игрок может найти для себя именно то, что он и хотел.

Объект выпускной квалификационной работы: процесс разработки компьютерных игр.

Предмет выпускной квалификационной работы: процесс разработки компьютерной игры в жанре «инди» на платформе Unity.

Цель выпускной квалификационной работы заключается в разработке демо-версии компьютерной игры в жанре "Инди".

Цель выпускной квалификационной работы предполагает решение таких задач:

- 1) Выполнить анализ предметной области.
- 2) Провести анализ инструментальных средств разработки.
- 3) Разработать демо-версию компьютерной игры.
- 4) Провести оценку экономической эффективности программного продукта.

1 Анализ процесса создания компьютерной игры в рамках обучения по направлению подготовки 09.03.03 Прикладная информатика

1.1 Описание соответствия навыков разработки компьютерной игры компетенциям выпускника по направлению подготовки 09.03.03 Прикладная информатика

Выпускники направления подготовки "Прикладная информатика" востребованы на различных предприятиях, начиная от специалистов, обслуживающих информационные системы предприятий, и заканчивая узкими специалистами, такими как C# Backend-разработчик или 1С-разработчик. Профессия геймдизайнер является очень разносторонней, и имеет много ответвлений:

- геймдизайнер кор-геймплея(отвечает за основные игровые механики);
- геймдизайнер мета-геймплея(отвечает за все происходящее вокруг основного геймплея);
- нарративный геймдизайнер(отвечает за создание сюжета и персонажей);
- технический геймдизайнер(отвечает за работу механик и скриптинг);
- дизайнер уровней(отвечает за создания уровней).

В больших студиях для каждого направления, есть отдельный человек.

Инди-разработчик, который занимается разработкой в одиночку, должен совместить в себе все эти навыки. Для развития профессиональных навыков геймдизайнера требуются профессиональные компетенции. В соответствии с ФГОС ВО по направлению подготовки 09.03.03 Прикладная информатика (уровень бакалавриат), утвержденным приказом Министерства образования и науки Российской Федерации от 19 сентября 2017 г. № 922 [25], выпускник должен обладать общепрофессиональными и профессиональными

компетенциями для получения документа об образовании. Разработка игры сориентирована на получение данных компетенций, в том числе с профессиональными стандартами "Программист" и "Специалист по информационным системам". В таблице 1 представлено соотношение вышеуказанных компетенций с результатами проделанной работы.

Таблица 1 – Компетенции

Компетенции	Реализация в проекте
Способность анализировать социально-экономические задачи и процессы	С помощью системного анализа был проведен анализ рынка и спрогнозирован успех проекта
Способность использовать основные законы естественнонаучных дисциплин и современные информационно-коммуникационные технологии в профессиональной деятельности	В разработке используется современное программное обеспечение, такое как Unity, Audacity, Krita.
Способность проводить обследование организаций, выявлять информационные потребности пользователей, формировать требования к информационной системе	Для разработки компьютерной игры был проведен анализ целевой аудитории и выявлены их потребности.
Способность разрабатывать, внедрять и адаптировать прикладное программное обеспечение	Итогом разработки является готовый программный продукт.
Способность выполнять технико-экономическое обоснование проектных решений	В процессе разработки было написано техническое задание к игре.
Способность проводить описание прикладных процессов и информационного обеспечения решения прикладных задач	Для разработки игры была создана модель в нотации IDEF3.
Способность программировать приложения и создавать программные прототипы решения прикладных задач	Для разработки использовался язык программирования C#.

Окончание таблицы 1

Способность проводить тестирование компонентов программного обеспечения ИС	Компьютерная игра перед релизом пройдет стадию тестирования для выявления ошибок.
Способность осуществлять тестирование компонентов информационных систем по заданным сценариям	В компьютерной игре стадию тестирования должен пройти каждый ее уровень.
Способность осуществлять презентацию информационной системы и начальное обучение пользователей	Компьютерная игра будет иметь обучающий уровень для тех, кто запускает игру впервые.
Способность осуществлять и обосновывать выбор проектных решений по видам обеспечения информационных систем	Для разработки были выбраны определенные информационные системы, их описание и анализ аналогов был описан в выпускной квалификационной работе.
Способность проводить оценку экономических затрат и рисков при создании информационных систем	В выпускной квалификационной работе была описана экономическая часть с обоснованием затрат и рисков.
Способность анализировать рынок программно-технических средств, информационных продуктов и услуг для создания и модификации информационных систем	Проведен анализ рынка программно-технических средств и проведена оптимизация минимальных системных требований для запуска игры.
Способность готовить обзоры научной литературы и электронных информационно-образовательных ресурсов для профессиональной деятельности	Во время разработки компьютерной игры было изучено множество информационно-образовательных ресурсов.

Как следует из анализа соответствия навыков работы и требований к освоению компетенций, разработка демоверсии компьютерной игры позволяет согласиться с утверждением: "Разработка игры отвечает требованиям для получения общепрофессиональных и профессиональных компетенций в соответствии с ФГОС ВО "Прикладная информатика". Более того, данные компетенции могут послужить базой для освоения профессии геймдизайнера.

1.2 Начало игровой индустрии

В 1960 году компанией Digital Equipment Corporation был выпущен первый компьютер серии PDP. Он получил название PDP-1. Спустя 2 года, в 1962 году для него была разработана первая компьютерная игра – Space War. Space War – одна из самых важных и влиятельных игр в ранней истории компьютерных игр. Она стала популярной в 1960-х годах настолько, что была установлена практически на всех компьютерах PDP-1. Space War портировалась на другие компьютерные платформы, а впоследствии была воссоздана на более современных языках программирования для эмуляторов компьютера PDP-1. Игра непосредственно вдохновила многих на создание других электронных игр, примерами среди которых являются первые коммерческие игры для аркадных автоматов Galaxy Game и Computer Space, появившиеся в 1971 году. Помимо этих, было оказано влияние на более поздние, например на Asteroids 1979 года.

В 2007 году Space War вошла в десятку самых важных компьютерных игр всех времен, положивших начало созданию игрового канона в Библиотеке Конгресса США. С момента выхода этой игры, игровая индустрия активно развивалась и прошла много стадий своего развития [10].

Сейчас видеоигры доступны абсолютно каждому и благодаря современным технологиям игровая индустрия и по сей день стремительно развивается.

1.3 Жанры компьютерных игр

Существуют компьютерные игры разных жанров, и каждый жанр можно разделить на несколько поджанров. Ниже представлена лишь малая часть жанров и поджанров, так как их существует огромное множество: стратегия, стратегия в реальном времени, пошаговая стратегия, шутер, FPS шутер, арена

шутер, приключение, RogueLike, SideScroller, RunandGun, файтинг, SurvivalHorror, инди.

Если игру тяжело отнести к какому-либо жанру, и по совместительству игра разработана независимым разработчиком, то игру причисляют к жанру – инди-игра [8]. "Инди" в этом слове является сокращением от английского слова "independent", которое переводится как независимый. Так игру "BlindDrive" практически невозможно причислить к определенному жанру. Несмотря на то, что в игре необходимо вести автомобиль, игра не подходит по нескольким параметрам к жанру гонки, и является инди-игрой. Целью игры является вести автомобиль с завязанными глазами, а жанр гонка подразумевает элемент соревнования, где игрок ведет машину на скорость.

Существуют игры с элементами нескольких жанров, которые могут принадлежать каждому из них. Такие проекты причисляют либо к одному из жанров, который в игре является основным, либо сразу ко всем, присутствующим в игре, если они в равной мере составляют геймплей проекта. Также существует множество игр, которым сложно определить их жанр, так как игра просто не подходит под определение ни к одному из них. В качестве примера можно привести игру "ForHonor". Несмотря на то, что проект является файтингом, в нем отсутствует большинство ключевых особенностей данного жанра. Зато в ней присутствуют элементы из жанров МОВА и стратегия. Файтинг подразумевает бой между двумя соперниками в ограниченном пространстве, а ForHonor – игра, в которой 10 игроков, разделенные на две команды, должны сразиться между собой на большом игровом поле.

Игровая индустрия насчитывает огромное множество жанров и новые жанры появляются и по сей день. Для разработки был выбран жанр инди так, как он не ограничивает разработчика в своих идеях, и позволяет разработчику занять свою нишу.

1.4 Описание идеи разрабатываемой игры

Разрабатываемая игра будет в жанре "Инди", так как она не будет принадлежать к какому-либо другому жанру игры, и будет иметь уникальный игровой процесс. По сюжету главный герой -бывший взломщик, который на одном из заданий потерял зрение. Из-за того, что он в жизни больше ничего не умеет, он решает восстановить свою репутацию, и с нуля учиться вскрывать замки, но на этот раз, используя только свой слух. Игра будет представлять собой взлом различных видов замков, которые игрок должен будет взламывать, ориентируясь на звук в своих наушниках.

Игра предоставит уникальное взаимодействие с игроком, используя только звуковое сопровождение, что позволит игроку научиться концентрироваться не только на визуальных образах, тем самым позволяя ему глубже погрузиться в игровой процесс, и самому представить происходящее у себя в голове. Благодаря минимальному количеству графики, в данную игру сможет играть большее количество игроков. Она сможет запускаться на слабых компьютерах, и тем самым можно увеличить охват потенциальной аудитории.

1.5 Описание платформы Steam

Игра будет распространяться посредством платформы Steam [17]. Steam является онлайн-сервисом цифрового распространения компьютерных игр и программ, разработанный и поддерживаемый компанией Valve. Существует множество сервисов распространения компьютерных игр и программ, такие как UPlay [19], Battle.net [4], но эти платформы предназначены для распространения только собственных игр этих компаний. Steam же является платформой, где опубликовать свой труд может каждый. В 2012 году Steam запустили проект, который должен быть направлен на поддержку инди разработчиков: SteamGreenlight.

SteamGreenlight является сервисом, где каждый инди разработчик может выложить свою игру. Пользователь может заплатить взнос в размере 100 долларов и выкладывать неограниченное количество игр. Затем, спустя определенное время, Steam составляет топ 100 игр, и пользователи голосованием выбирают лучшие из них для публикации в Steam. Топ составлялся на основе голосов пользователей, которые заходили на страницу игры в SteamGreenlight и голосовали либо "за", либо "против". Сервис просуществовал до 2017 года и был закрыт из-за факторов, которые, наоборот, только мешали разработчикам.

Во-первых, сервис имел страничную систему, и если после публикации вашей игры, множество других людей выкладывало свои, то ваша игра опускалась в самый низ и переходила на другую страницу, и пользователи уже ее не видели, так как большинство людей не смотрят следующие страницы, а смотрят только то, что расположено на первой, главной странице. Плюс к этому голосование пользователей за выход игры в топ 100 не имели должного эффекта. Голоса "против" не учитывались в системе, а наоборот поднимали рейтинг игры за счет посещения ее страницы и проявления на ней активности, и игра получала зеленый свет. Также многие разработчики занимались ботоводством. Ботоводство – имитация действий пользователей, посредством компьютерных программ [24]. Люди просто покупали или создавали ботов для голосований и посещений страницы, тем самым поднимая ее рейтинг. Так же во время работы сервиса, Steam представила коллекционные карточки. Коллекционные карточки — это виртуальные карточки, которые выпадают во время игры. Их можно превратить в игровые значки или поменяться с другими пользователями. Некоторые решили это монетизировать. Они создавали игры-пустышки, в которых не было никакого контента, и создавали карточки для этих игр. Затем создавали или покупали ботов, на правах разработчика давали им коллекционные карточки, и боты сами продавали их на торговой площадке. Благодаря этому, люди могли даже ни разу не видеть игру, а ее разработчик

богател на продаже карточек. Из-за всех этих факторов SteamGreenlight был закрыт, и анонсирован SteamDirect [16].

"Добро пожаловать в Steam Direct! Это новая система публикации игр в Steam, которая должна сделать этот процесс упорядоченным, прозрачным и доступным для новых разработчиков со всего света" -написано на главной странице SteamDirect. Данный сервис имеет определенные отличия от SteamGreenlight. Во-первых, взнос в размере 100 долларов больше не является разовым, и разработчик должен платить этот взнос за каждую выкладываемую игру. Во-вторых, этот взнос возмещается в виде выплаты после того, как валовая выручка от продажи продукта в Steam составит как минимум 1000 долларов. В-третьих, возможность добавить коллекционные карточки появляется у разработчика спустя время, когда игра набирает определенную популярность. Данные факторы ограничили поток выкладываемых игр благодаря тому, что для получения прибыли, необходимо было создавать более качественные продукты и заниматься их распространением, что отпугнуло разработчиков, желавших заработать легкие деньги.

Так как продажа игр является средством получения заработка, то с этого заработка необходимо платить налоги. Размер налогов зависит от статуса разработчика. Если он зарегистрирован как физическое лицо, то он должен платить 13% от дохода. Если зарегистрирован индивидуальный предприниматель (ИП), то нужно платить 6% с продаж плюс социальные отчисления. Если зарегистрировано общество с ограниченной ответственностью (ООО), то необходимо платить 6% с продаж, и различный налог. Также необходимо иметь долларовый счет в банке, так как Steam предоставляет вывод средств только в долларах. Для регистрации в SteamDirect необходимо ввести информацию об ИП или ООО. Затем нужно оплатить взнос. После этого необходимо указать налоговую информацию о разработчике. После всех выполненных действий нужно заполнить форму W-8BEN. Затем заявка рассматривается администраторами Steam, это занимает около суток.

После этого разработчик получает доступ к своему кабинету SteamDirect и может выкладывать свою игру.

Разрабатываемая демоверсия игры будет содержать три уровня различной сложности. Это необходимо для того, чтобы показать будущему игроку развитие персонажа, что главный герой не стоит на месте, а совершенствует свои навыки. Демоверсия будет длиться около 20 минут. Этого времени будет достаточно для того, чтобы показать игроку большинство геймплейных особенностей игры.

1.6 Сравнительный анализ и выбор инструментальных средств разработки

Для создания компьютерной игры необходим игровой движок. Движок игры – это ее основное ядро, базовое программное обеспечение, на основе которого строятся все остальные составляющие игры. Существует множество различных игровых движков. Ниже рассмотрены одни из самых популярных игровых движков.

Unity [20] – межплатформенная среда разработки компьютерных игр, разработанная американской компанией Unity Technologies, в которой объединены различные программные средства, используемые при создании ПО – текстовый редактор, компилятор, отладчик и так далее. При этом, благодаря удобству использования, Unity делает создание игр максимально простым и комфортным, а мультиплатформенность движка позволяет разработчикам охватить как можно большее количество игровых платформ и операционных систем. Выпуск Unity состоялся в 2005 году. Логотип Unity представлен на рисунке 1.

К достоинствам движка относят:

- мультиплатформенность;
- удобный интерфейс, представленный на рисунке 2;

– большая библиотека ассетов и плагинов, фрагмент которой представлен на рисунке 3.

К недостаткам относятся:

– плохая оптимизация больших пространств;

– большие затраты памяти.



Рисунок 1 – Логотип Unity

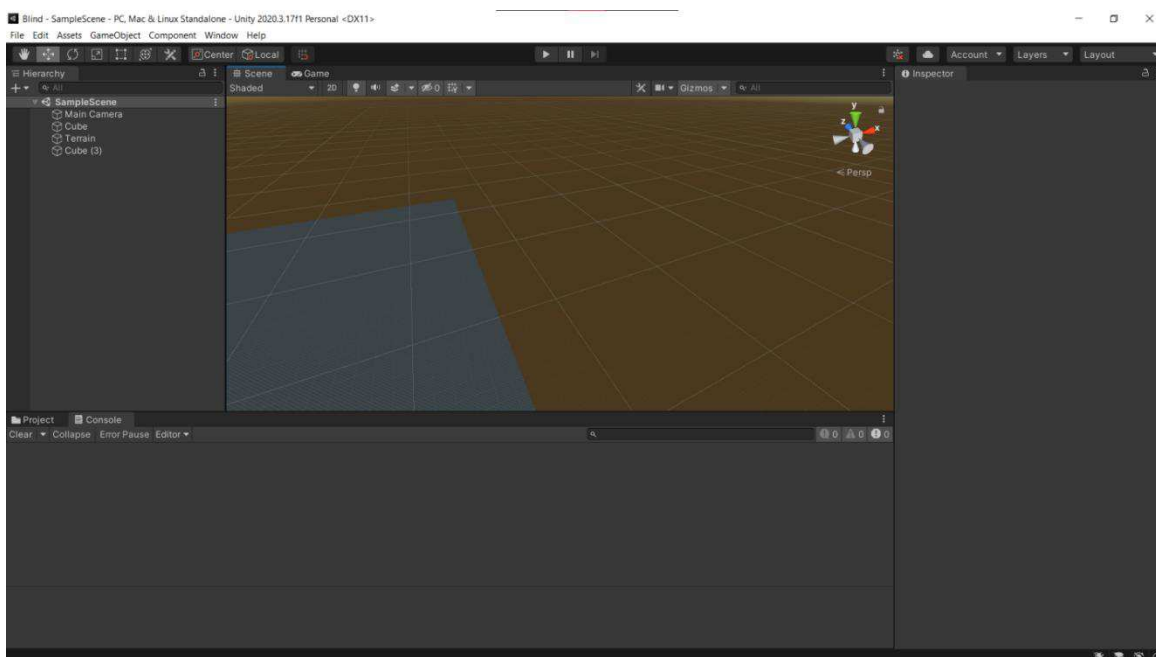


Рисунок 2 – Интерфейс Unity

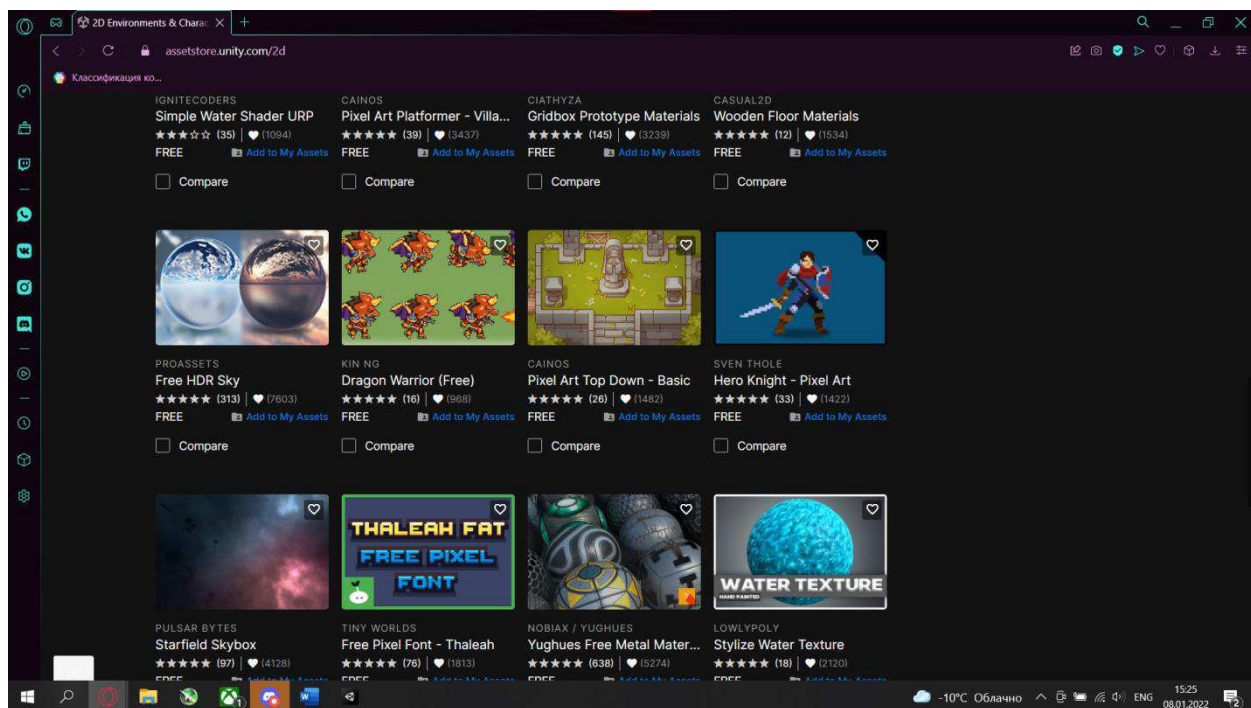


Рисунок 3 – Контент из Unityassetsstore

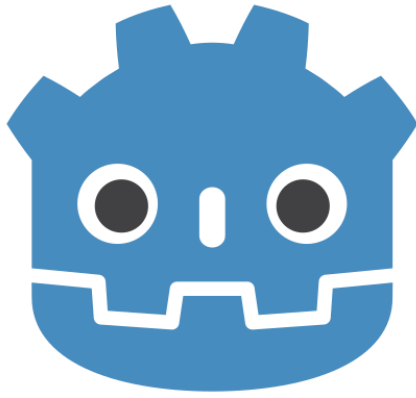
Godot engine [11] – открытый, кроссплатформенный 2D и 3D игровой движок, выпущенный в 2016 г. под лицензией MIT, который разрабатывается сообществом Godot Engine Community. Среда разработчика работает на Linux, OS X, Windows, BSD и Haiku и может экспортировать игровые проекты на ПК, консоли, мобильные и веб-платформы. Движок в большей степени ориентирован на 2D. Логотип Godotengine представлен на рисунке 4. Интерфейс и магазин представлены на рисунках 5 и 6.

К достоинствам движка относят:

- большая документация;
- небольшой размер игр;
- поддержка нескольких языков программирования: GDScript, C#, C++.

К недостаткам относятся:

- отсутствие большинства функций в связи с новизной движка;
- мало обучающего контента.



GODOT

Game engine

Рисунок 4 – Логотип Godot engine

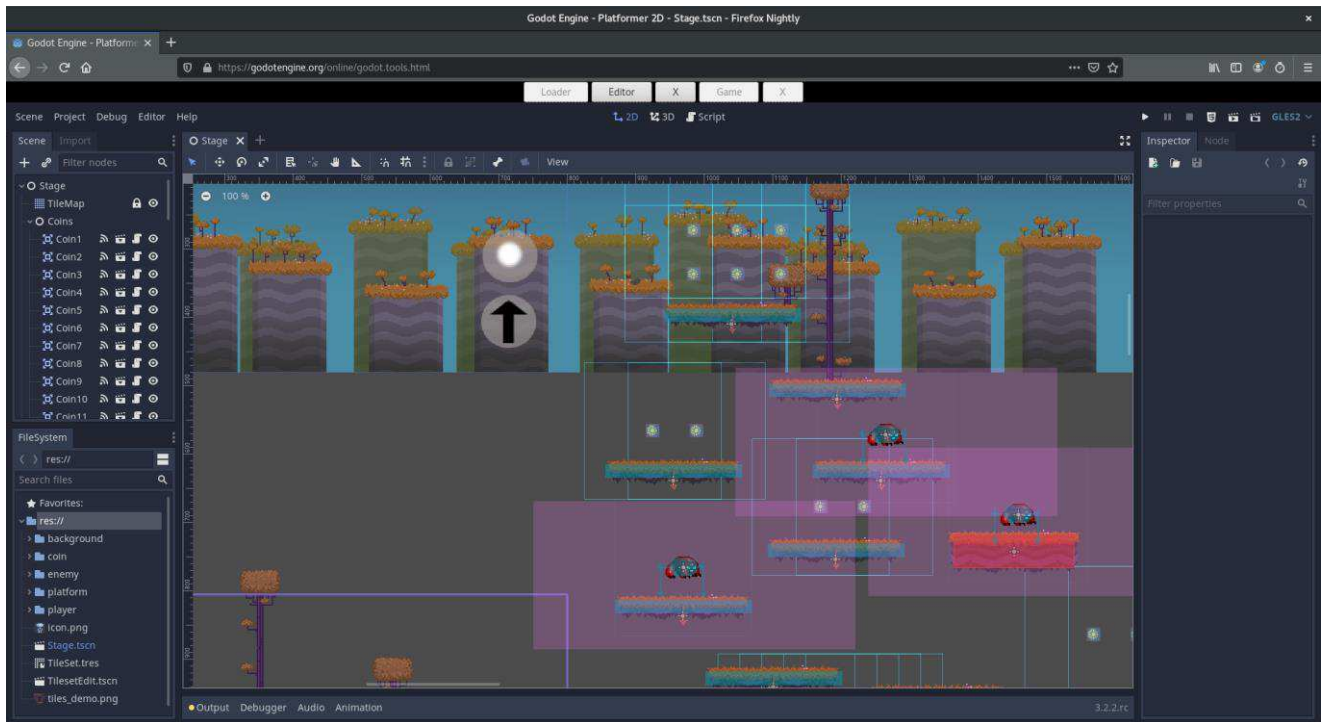


Рисунок 5 – Интерфейс Godot engine

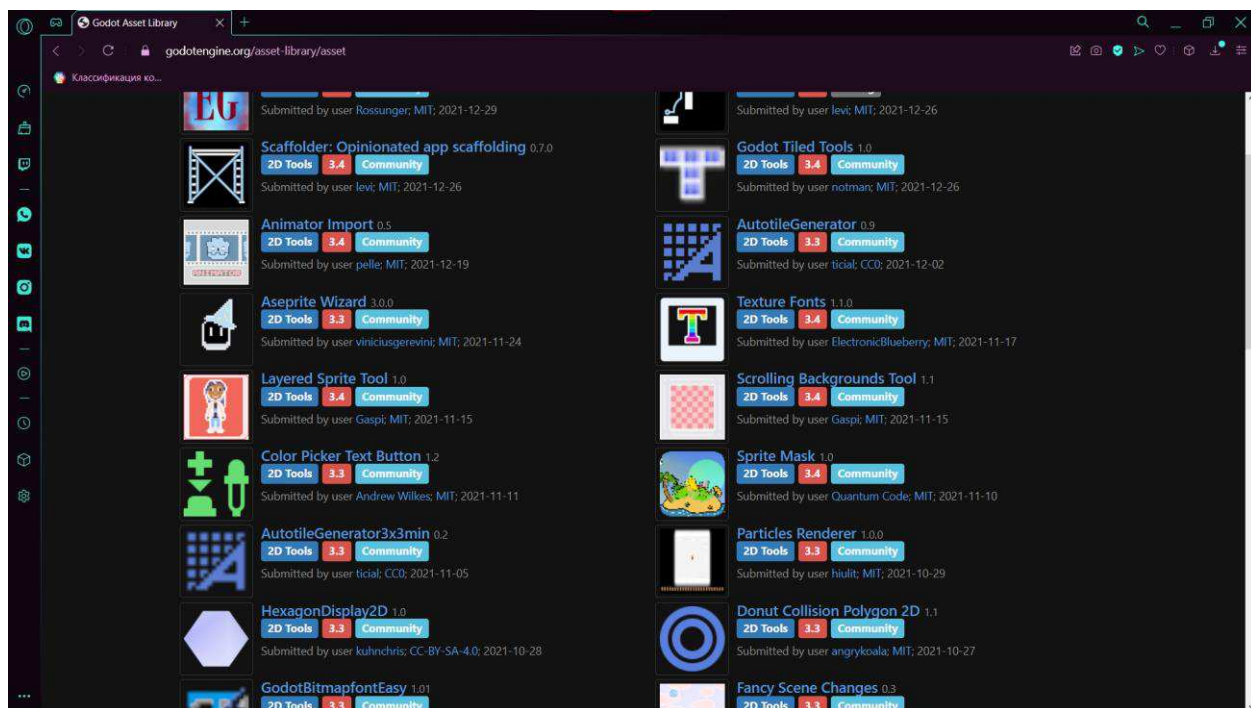


Рисунок 6 – Контент из Godotassetlibrary

Unreal Engine [21] – игровой движок, разрабатываемый и поддерживаемый компанией Epic Games [9]. Первой игрой на этом движке был шутер от первого лица Unreal, выпущенный в 1998 году. Хотя движок первоначально был предназначен для разработки шутеров от первого лица, его последующие версии успешно применялись в играх самых различных жанров, в том числе в стелс-играх, файтингах и массовых многопользовательских ролевых онлайн-играх. В прошлом движок распространялся на условиях оплаты ежемесячной подписки. С 2015 года Unreal Engine бесплатен, но разработчики использующих его приложений обязаны перечислять 5% роялти [28] от общемирового дохода с некоторыми условиями. Логотип Unreal Engine представлен на рисунке 7.

К достоинствам движка относят:

- графика;
- качественные ассеты, представленные на рисунке 9.

К недостаткам относится:

- вес движка и игр;

– громоздкий интерфейс, представленный на рисунке 8.



Рисунок 7 – Логотип Unrealengine

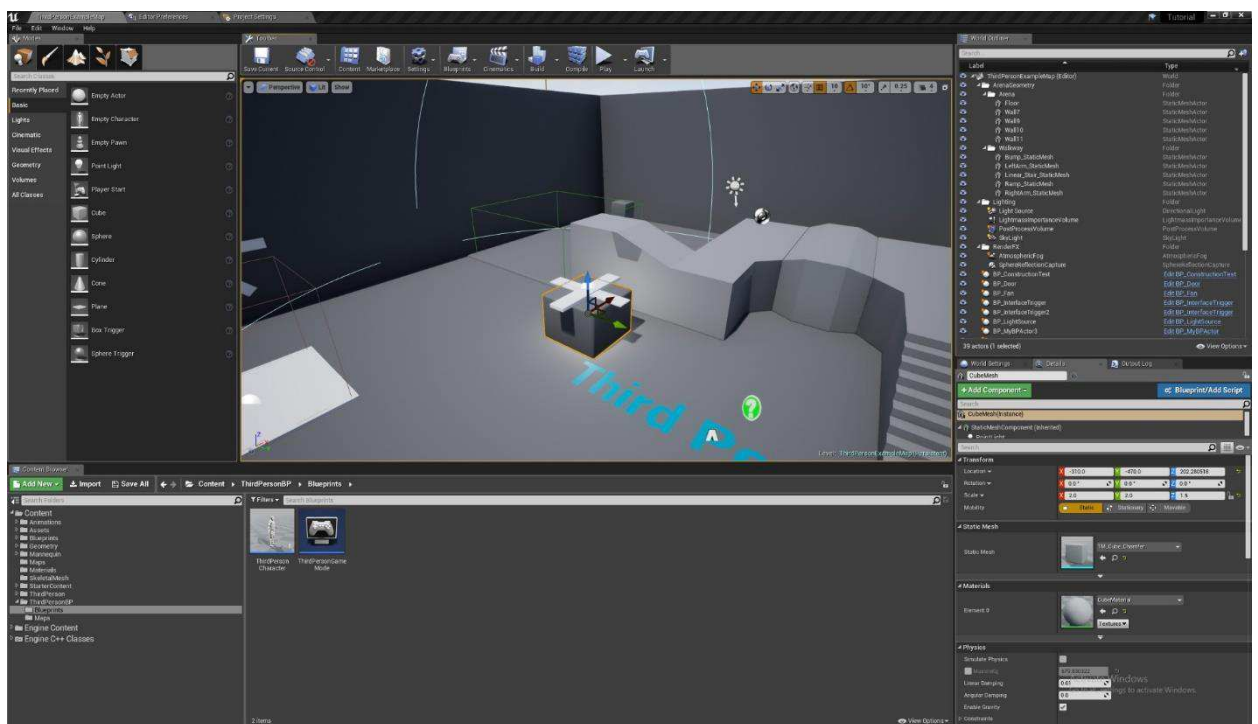


Рисунок 8 – Интерфейс Unreal engine

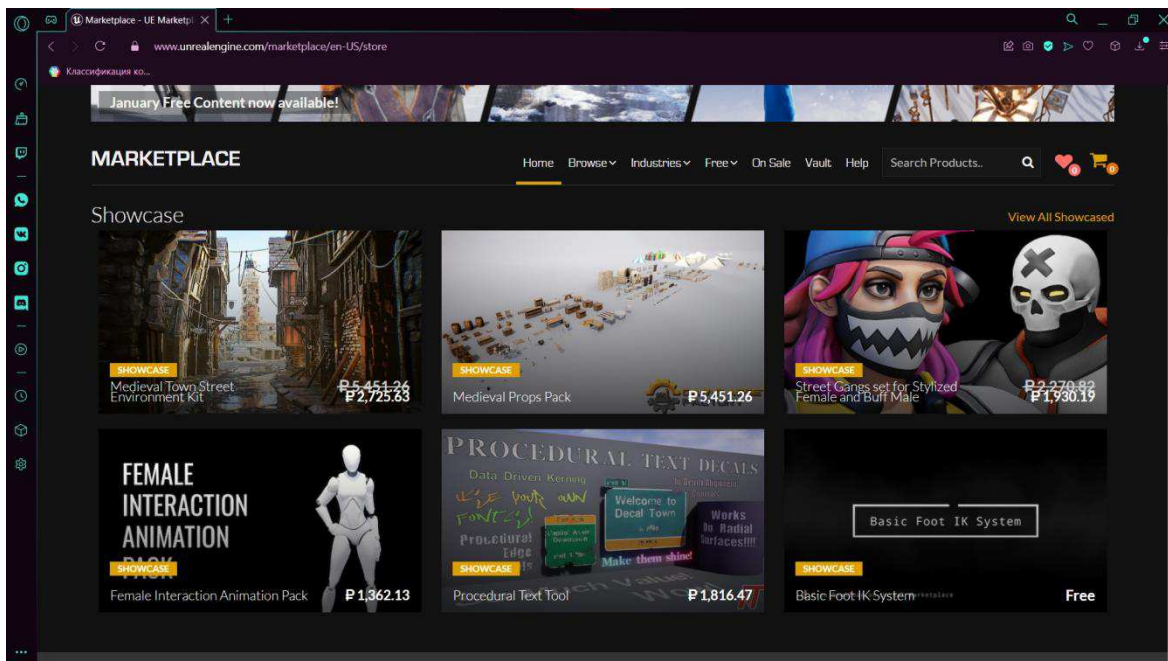


Рисунок 9 – Контент UEMarketplace

После проведенного анализа вышеперечисленных движков определены критерии для оценивания по пятибалльной шкале, где 2 – низкий уровень оценки и 5 – самый высокий. Субъективная оценка, основанная на работе с каждым из игровых движков, представлена в таблице 2.

Таблица 2 – Сравнение игровых движков

Критерии	Игровые движки		
	Godot	Unity	Unreal engine
Интерфейс	5	5	2
Доступность бесплатного контента	5	4	5
Порог входа	4	4	2
Скорость устранения ошибок	5	3	3
Популярность	2	5	4
Количество обучающего контента	3	5	4
Наличие шаблонов для графики	2	5	4
Среднее значение	3,7	4,4	3,4

На основе анализа табличных данных лучшим вариантом выбора игрового движка является Unity. И помимо того, что он набрал наибольший средний балл, он удовлетворяет наиболее важным критериям, таким как: количество обучающего контента и порог входа.

Несмотря на то, что Unity имеет встроенную IDE, для более комфортной работы следует установить VisualStudioCode. Интегрированная среда разработки или IDE – система программных средств, используемая программистами для разработки программного обеспечения. Так как в Unity используется язык программирования C# и присутствует интеграция с VisualStudioCode, то это IDE является наилучшим решением.

Для подходящей работы со звуком были рассмотрены такие программы как: Audacity и Adobe Audition. Они в полной степени удовлетворят необходимым потребностям, таким как: обрезка звука, настройка тональности звука и экспорт звуков в необходимом формате. Но так как Audacity является бесплатной, то для работы со звуком была выбрана именно она.

1.7 Моделирование процесса разработки компьютерной игры

IDEF3 —методология моделирования и стандарт документирования процессов, происходящих в системе. Представленная модель показывает действия игрока, зашедшего в игру. После попадания в главное меню игрок может перейти в настройки, начать новую игру и загрузить сохранение, если он уже играл и желает продолжить с последнего сохранения. В настройках игрок может изменить положение звука, в случае если у пользователя звук отображается некорректно и звук, который должен быть в правом наушнике исходит из левого. Также игрок может изменить общую громкость игры.

Как только игрок начинает новую игру, сначала необходимо настроить ориентацию и громкость звука. Затем игрок переходит к обучению, и он должен, исходя из звука в его наушниках, нажать кнопку, соответствующую направлению звука. Если игрок нажимает неправильные кнопки, то он

возвращается обратно к обучению. Если же игрок делает все правильно, то он переходит на тестовый уровень.

Он должен вставить натяжитель в замок, нажав определенную клавишу, и другой клавишей, ориентируясь на звук, по очереди выставлять штифты в замке до определенного уровня. Натяжитель – это устройство, которое позволяет вскрыть замок, выставляя штифты в замке в нужном порядке.

При неправильном прохождении уровня игрок вернется к этапу обучения, а при правильном уровне закончится, и игрок сможет сохранить свой прогресс для дальнейшего прохождения, без необходимости проходить обучение, и выйти из игры. Модель IDEF3 представлена на рисунке 10.

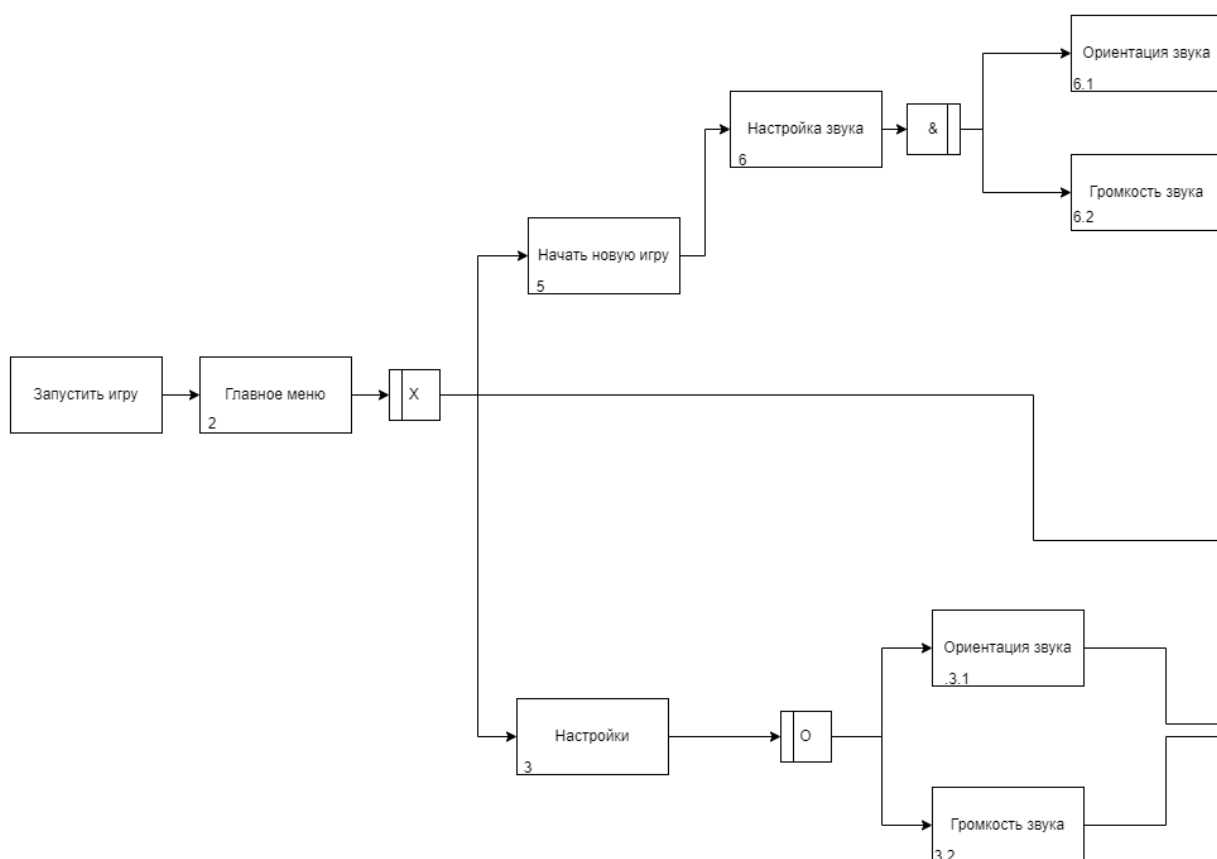


Рисунок 10 – Модель IDEF3, лист 1

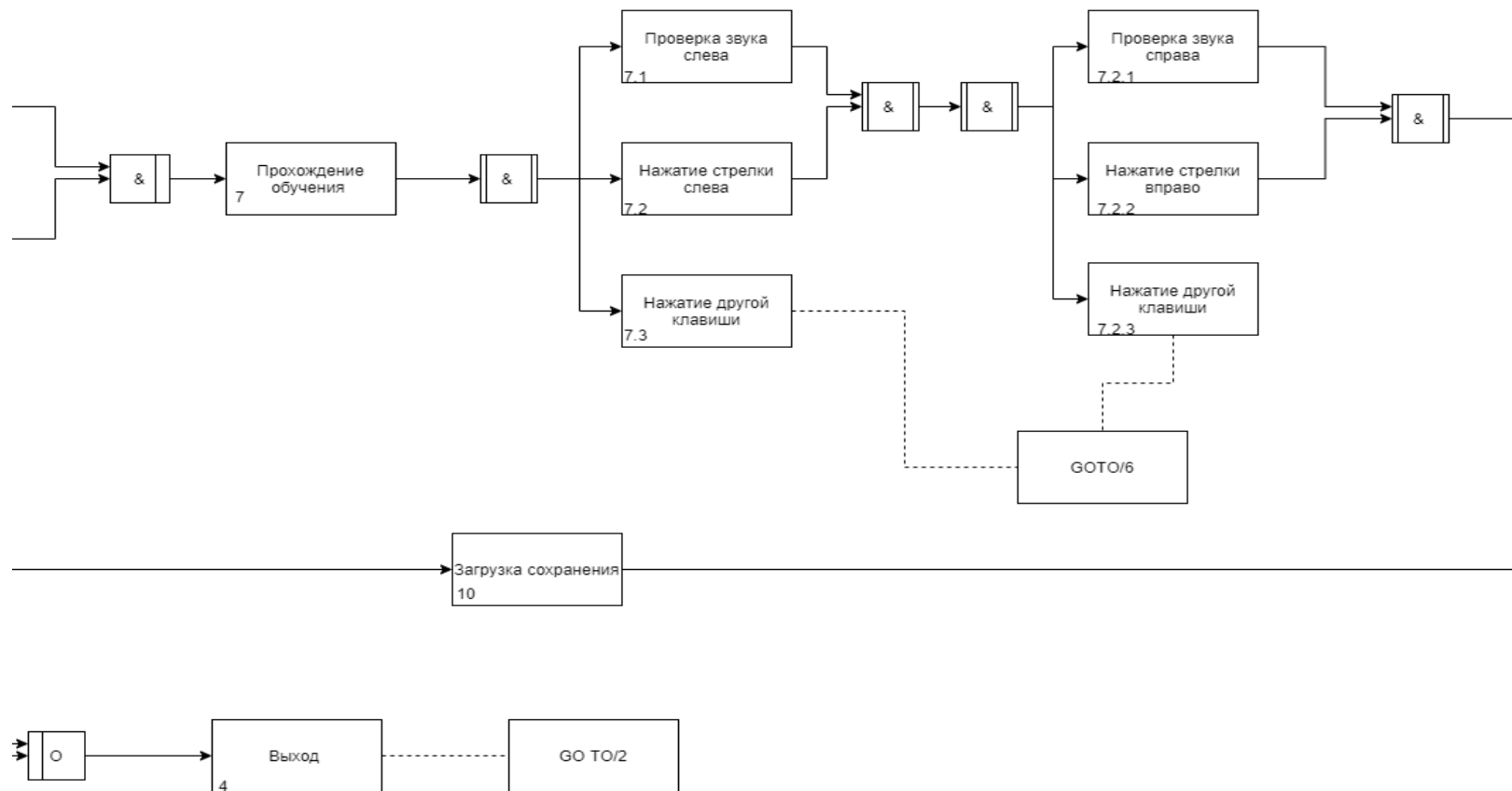


Рисунок 10, лист 2

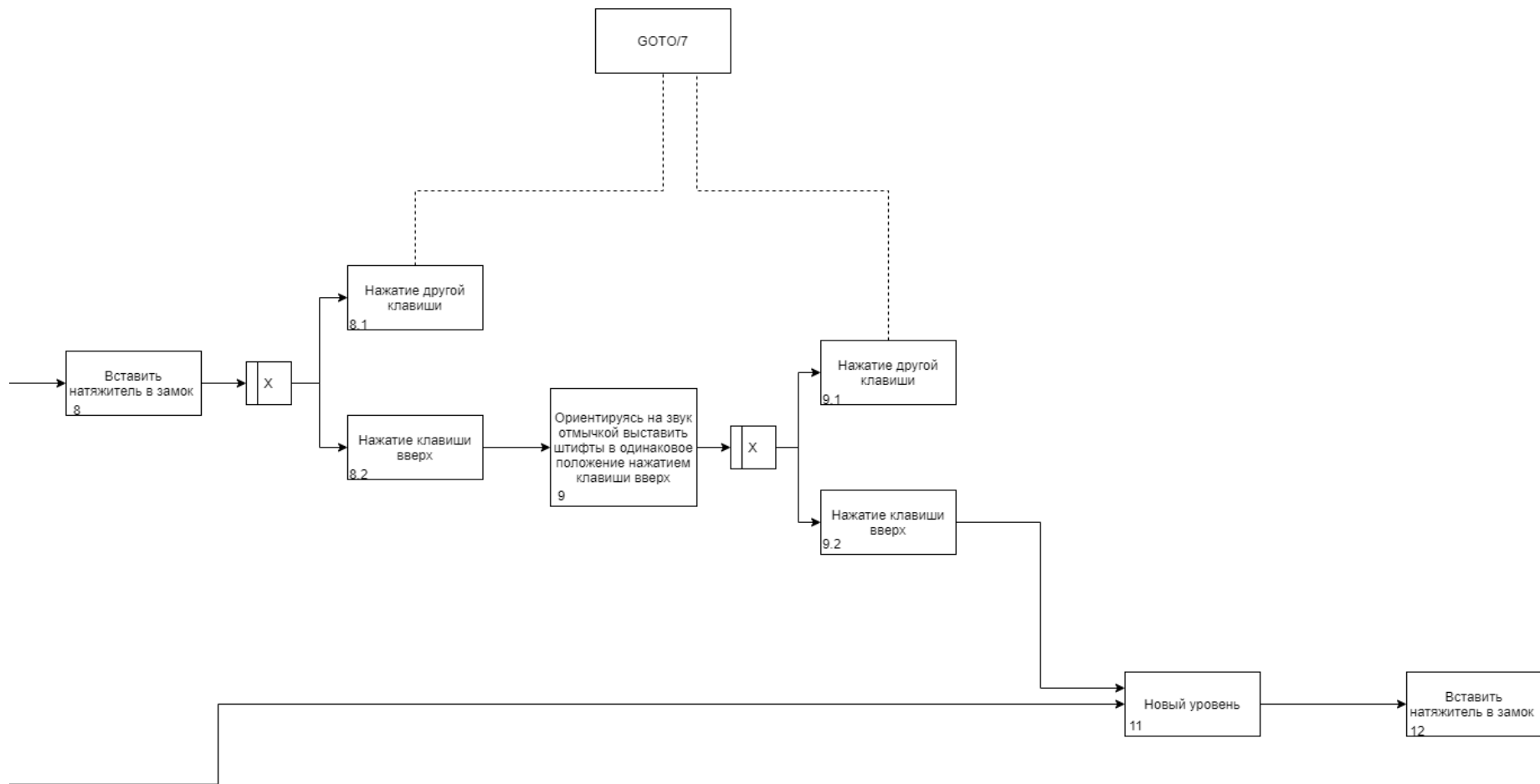


Рисунок 10, лист 3

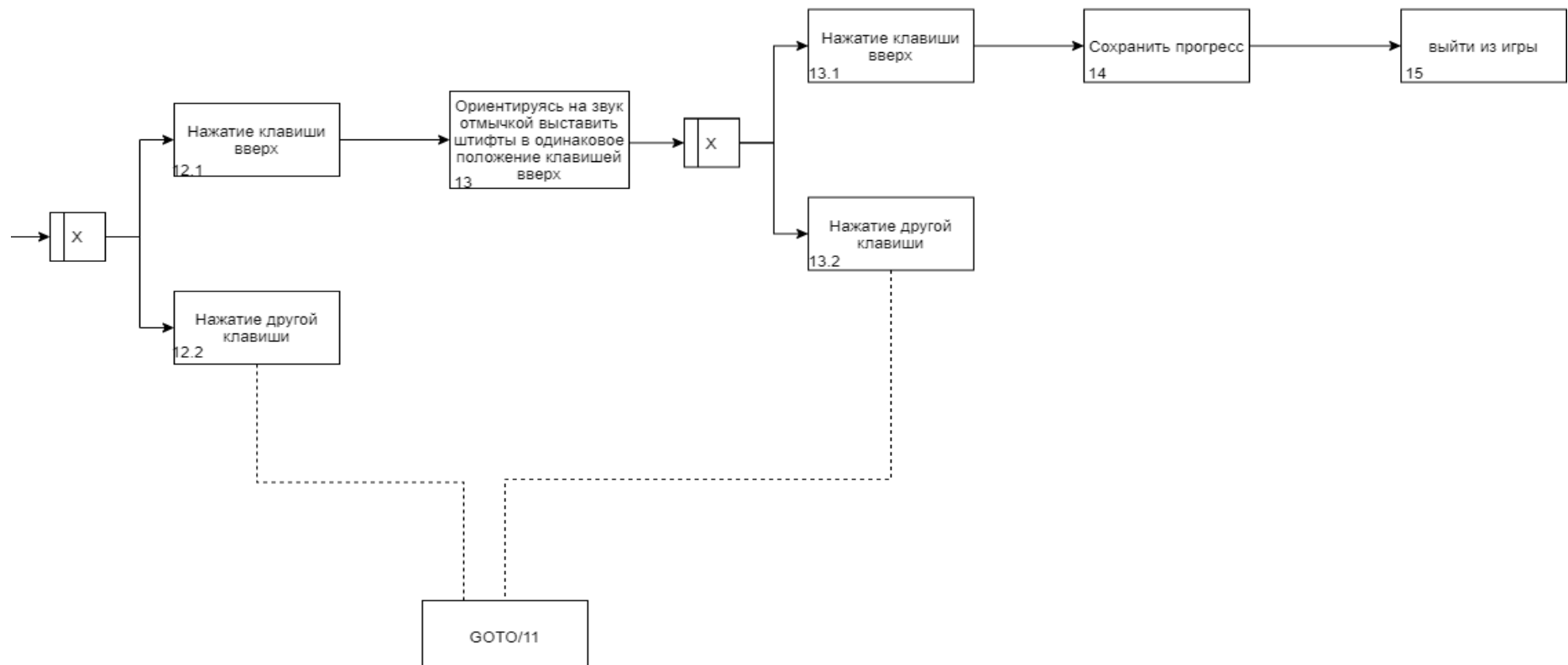


Рисунок 10, лист 4

1.8 Выводы по разделу «Анализ процесса создания компьютерной игры в рамках обучения по направлению подготовки 09.03.03 Прикладная информатика»

В данном разделе была определена актуальность выбранной темы с точки зрения соответствия навыков разработки компьютерных игр компетенциям выпускника направления 09.03.03 Прикладная информатика. Описаны этапы развития игровой индустрии, а также основные жанры компьютерных игр. Рассмотрена платформа для дальнейшего продвижения игры. Был проведен анализ средств разработки, и было выбрано следующее программное обеспечение: игровой движок Unity, IDE Microsoft Visual Studio Code и приложение Audacity. Был определен концепт разрабатываемой игры, построена модель в нотации IDEF3, которая показывает процесс игры со стороны игрока.

2 Разработка демо-версии компьютерной игры

2.1 Установка программного обеспечения

Перед началом разработки было скачано специальное программное обеспечение. Первым делом скачали игровой движок Unity с официального сайта [20]. Для этого перешли на официальный сайт Unity и выбрали тариф для физического лица "personal", что показано на рисунке 11.

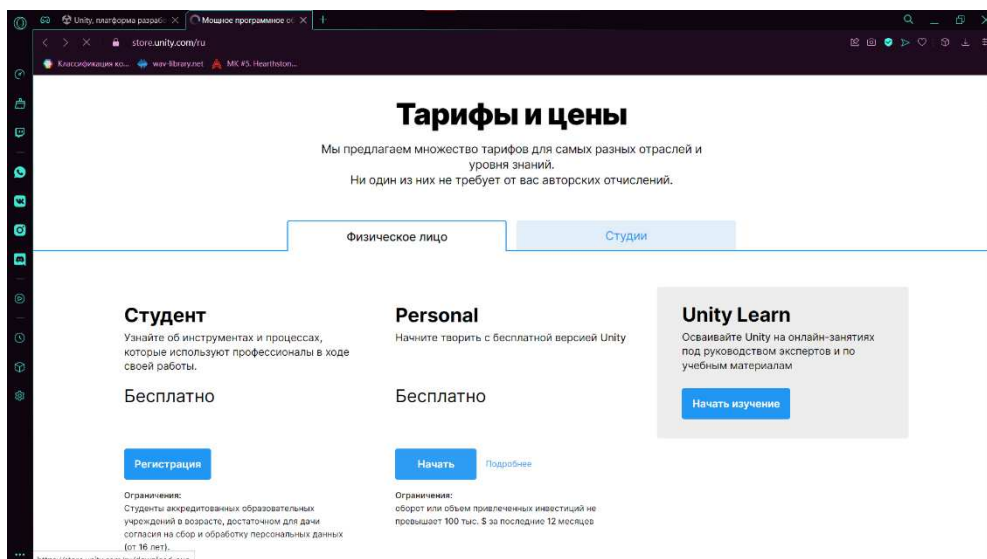


Рисунок 11 – Тарифы Unity

После этого выбрали версию операционной системы и установили Unity, что показано на рисунке 12.

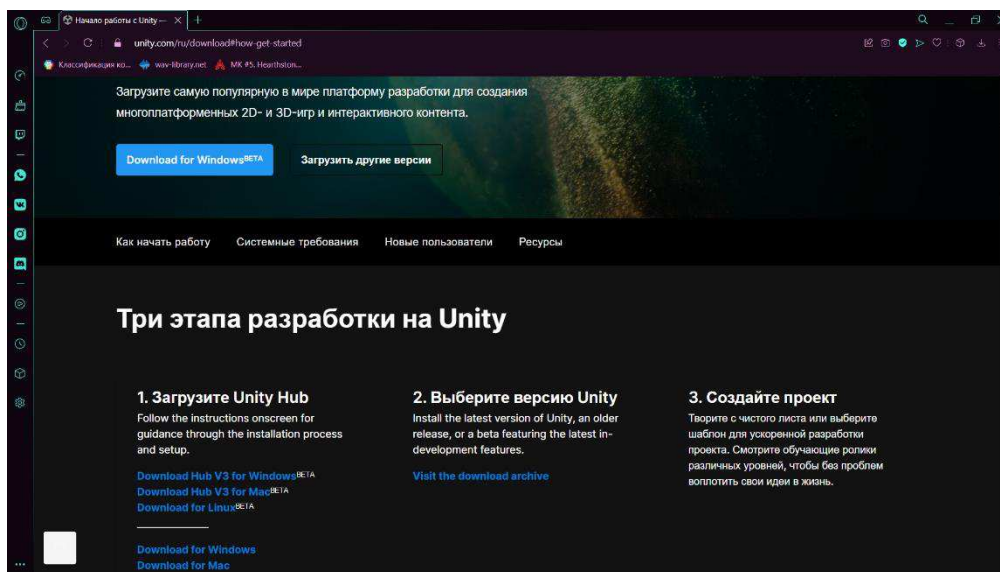


Рисунок 12 – Загрузка Unity

После того, как Unity был скачан и установлен, зарегистрировали Unity ID, как показано на рисунке 13.

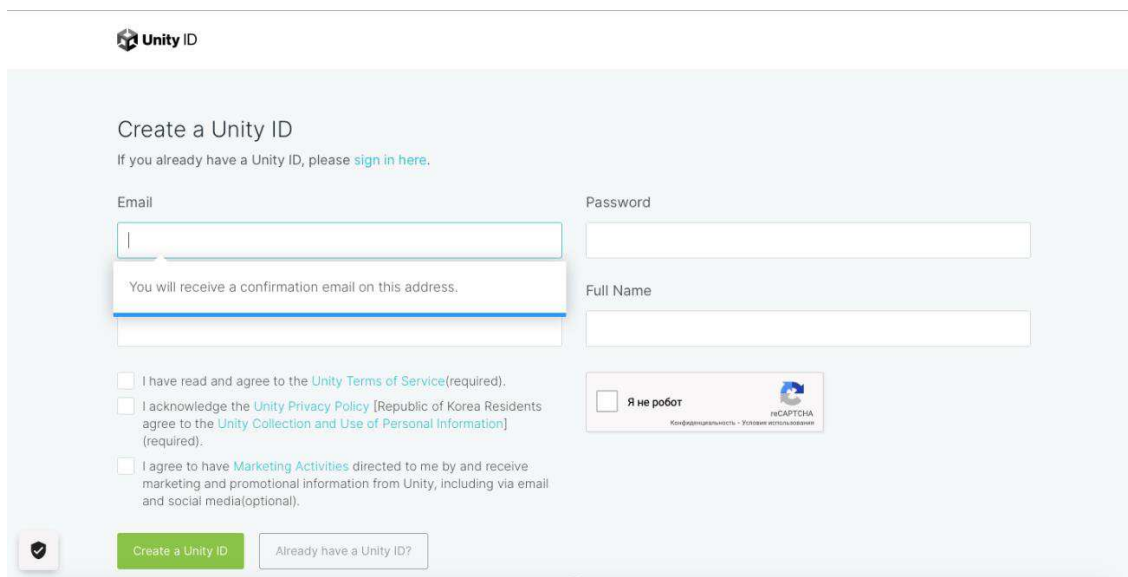


Рисунок 13 – Регистрация UnityID

По завершении регистрации был продемонстрирован вступительный рекламный ролик и открылся UnityHub, который позволяет просматривать существующие проекты, версии самого Unity и обучающие материалы. Окно UnityHub показано на рисунке 14.

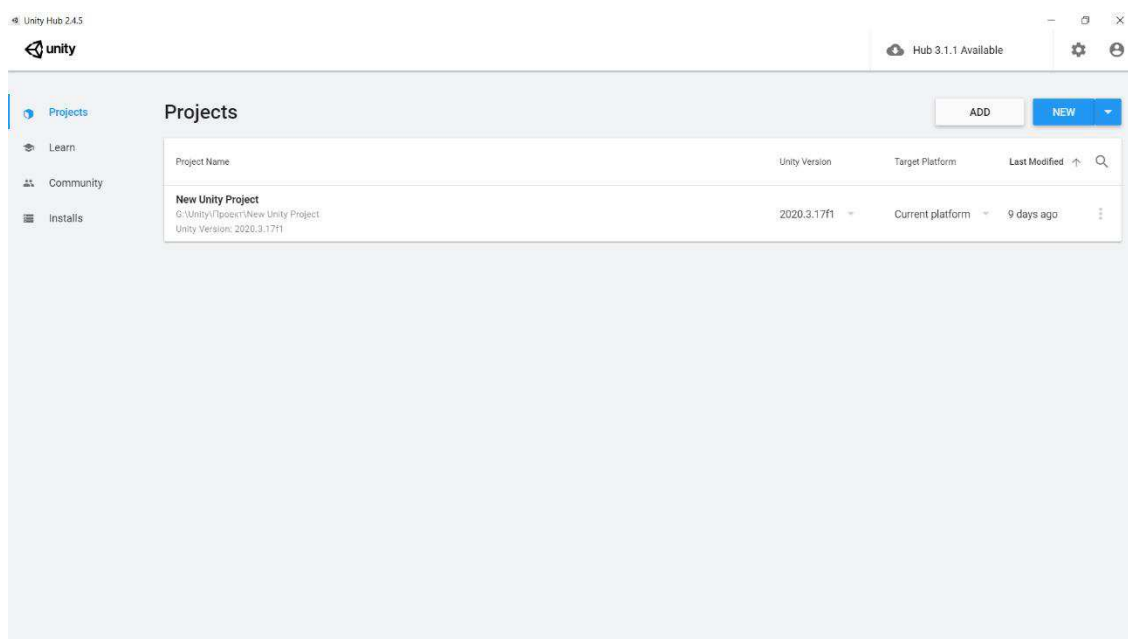


Рисунок 14 – Окно UnityHub

VisualStudioCode является бесплатно распространяемым программным обеспечением и оно было скачано с официального сайта [22] после выбора операционной системы, что показано на рисунке 15.

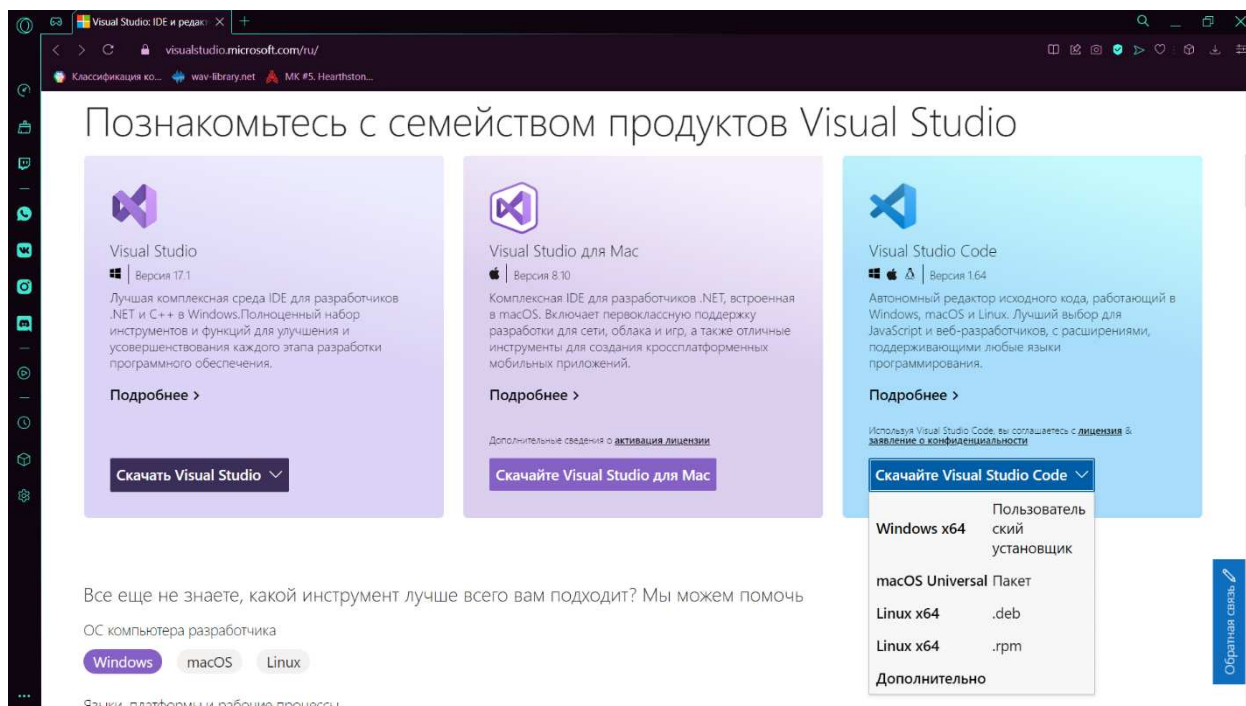


Рисунок 15 – Установка VisualStudioCode

Для работы со звуком необходимо бесплатное программное обеспечение Audacity. Оно в большей мере удовлетворяет всем необходимым требованиям для работы со звуком. Установка Audacity показана на рисунке 16.

После установки приложения не требуется дополнительная регистрация или настройка его функционала, можно сразу приступить к обработке звука.

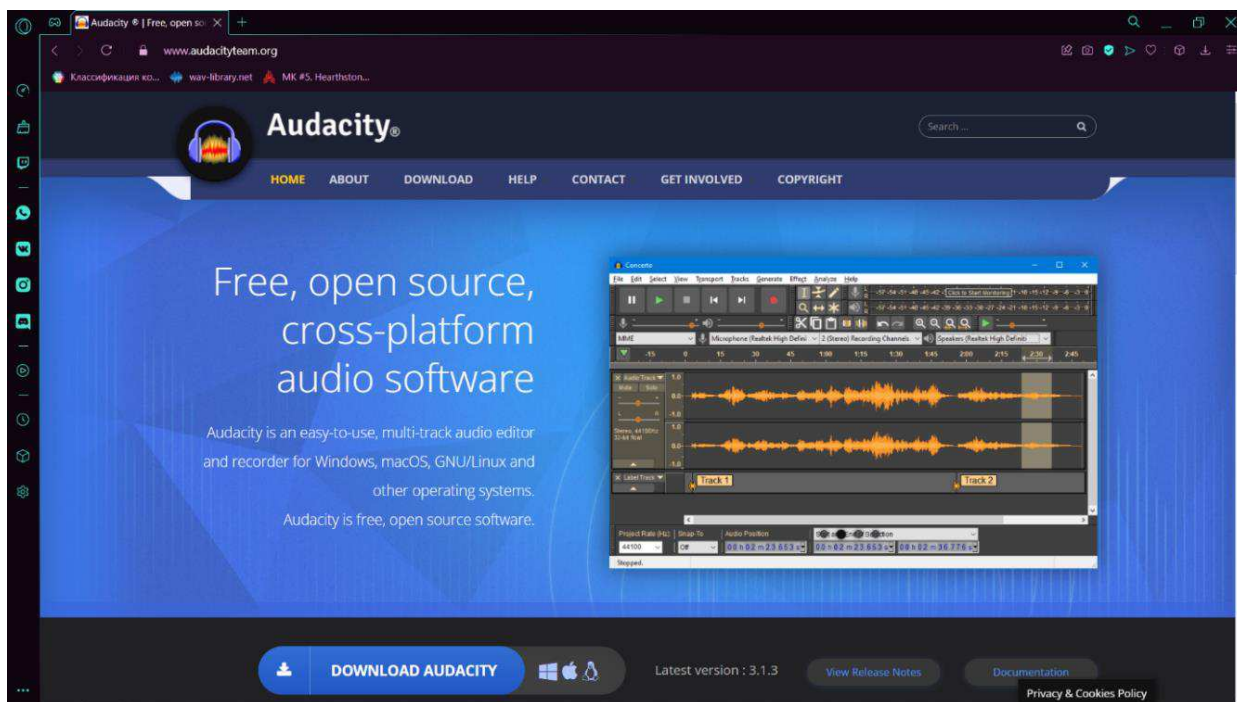


Рисунок 16 – Официальный сайт Audacity

2.2 Разработка игровых уровней

После установки всего необходимого ПО приступили к созданию проекта в Unity. Для этого открыли UnityHub и нажали на кнопку "New". Затем открылось дополнительное окно, и Unity предложил выбрать шаблон создаваемой игры, ее название и расположение. Создание проекта показано на рисунке 17.

Выбираемый шаблон зависит от жанра будущей игры. Так как разрабатываемая игра рассчитана в основном на звуковое восприятие, то вполне подойдет шаблон 2D.

Наименование проекта и его расположение пользователь выбирает по своему усмотрению.

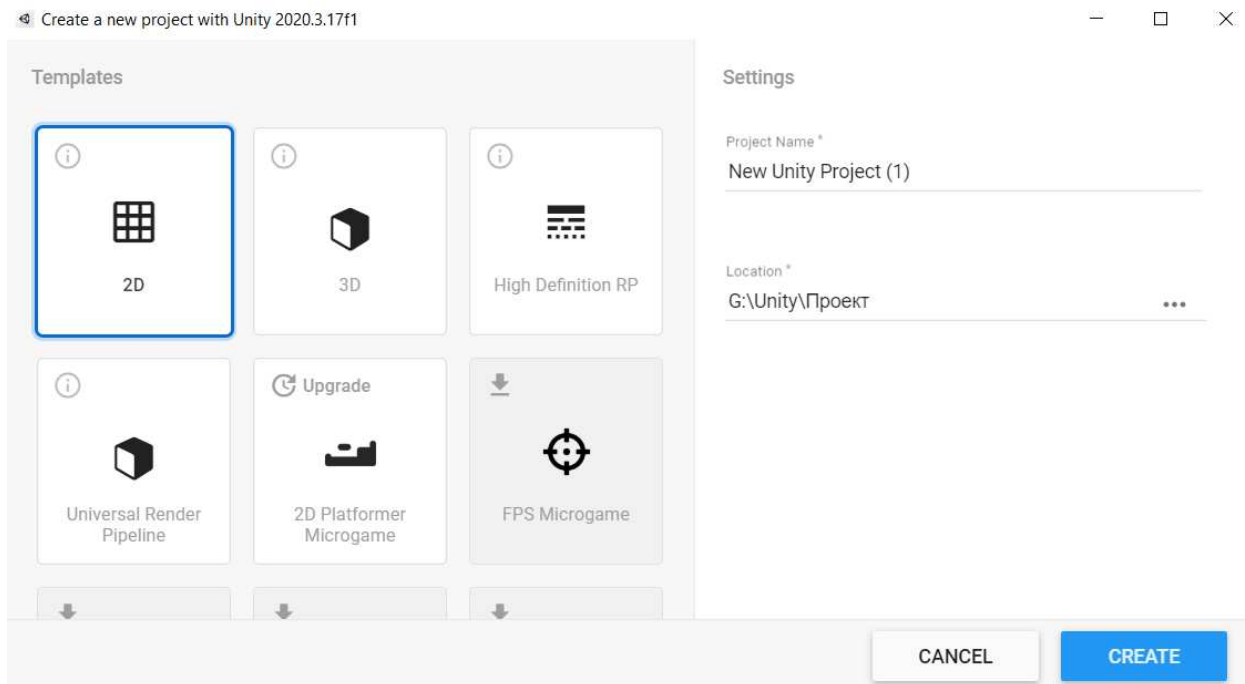


Рисунок 17– Создание нового проекта в Unity

После выбора шаблона нажали кнопку "Create", после чего открылся пустой проект в Unity, как показано на рисунке 18.

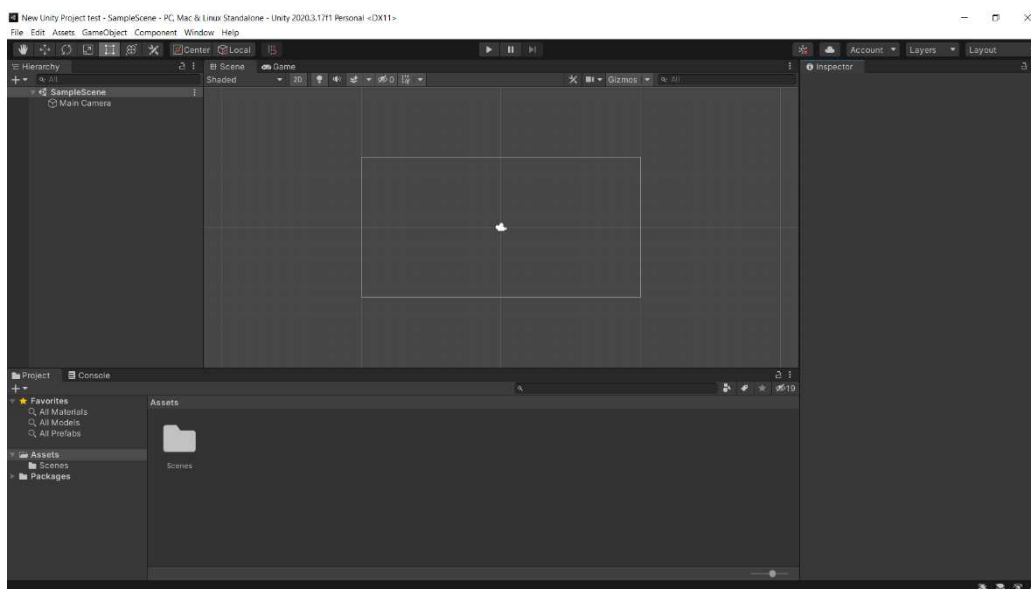


Рисунок 18 – Пустой проект в Unity

В первую очередь необходимо найти настройку "ExternalScriptEditor", которая находится в "Edit" -> "Preferences" -> "ExternalTools" и выбрать ранее установленный VisualStudioCode, как показано на рисунке 19.

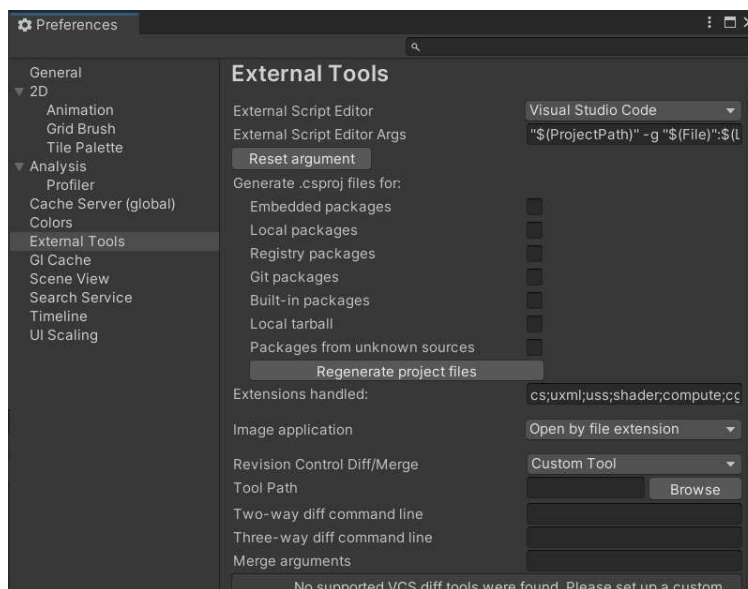


Рисунок19 – Настройка"External Script Editor"

Это нужно для того, чтобы создаваемые скрипты открывались сразу в VisualStudioCode. Из-за особенностей игры перед созданием необходимо было подготовить звуковое сопровождение, которое использовалось на первом уровне. Для этого запустилиAudacity. Интерфейс программы представлен на рисунке 20.

Интерфейс приложения оформлен на русском языке. Стиль меню напоминает меню MSOffice. Поэтому работа в приложении не вызывает затруднений даже у неопытного пользователя.

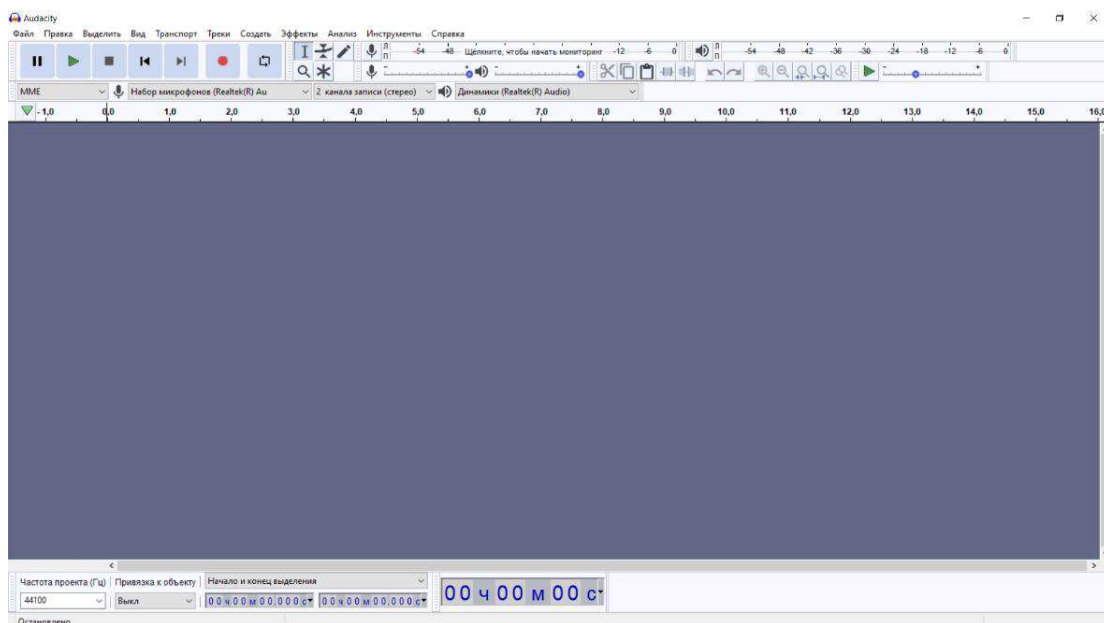


Рисунок 20 – Интерфейс Audacity

Затем перешли на сайт <https://wav-library.net> для подбора необходимых звуковых дорожек. Сайт представлен на рисунке 21.

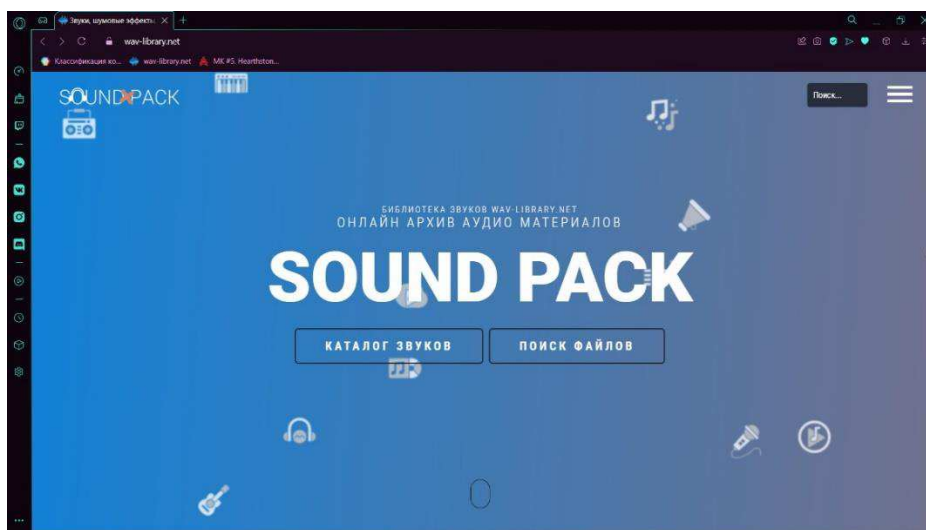


Рисунок 21 – Сайт <https://wav-library.net>

После этого нажали кнопку "Каталог звуков" и выбрали подходящую категорию, как показано на рисунке 22.

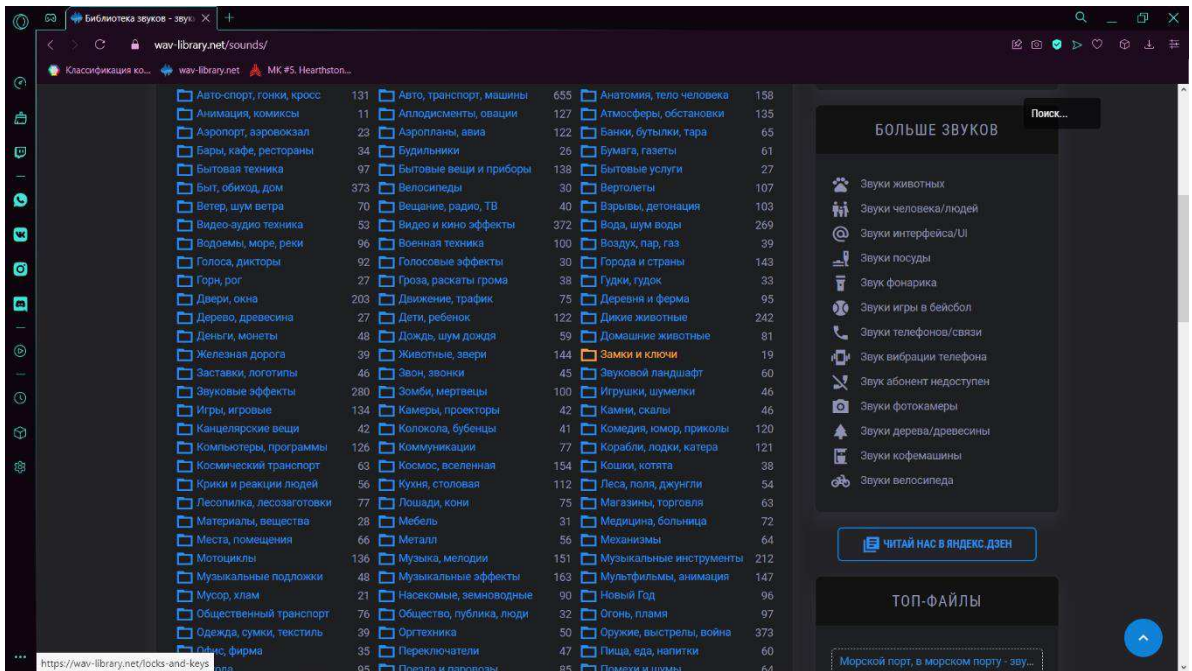


Рисунок 22 – Категории звуков

После выбора категории прослушали предлагаемые сервисом звуки и скачали подходящие. Перечень необходимых для разрабатываемой игры звуков представлен на рисунке 23.

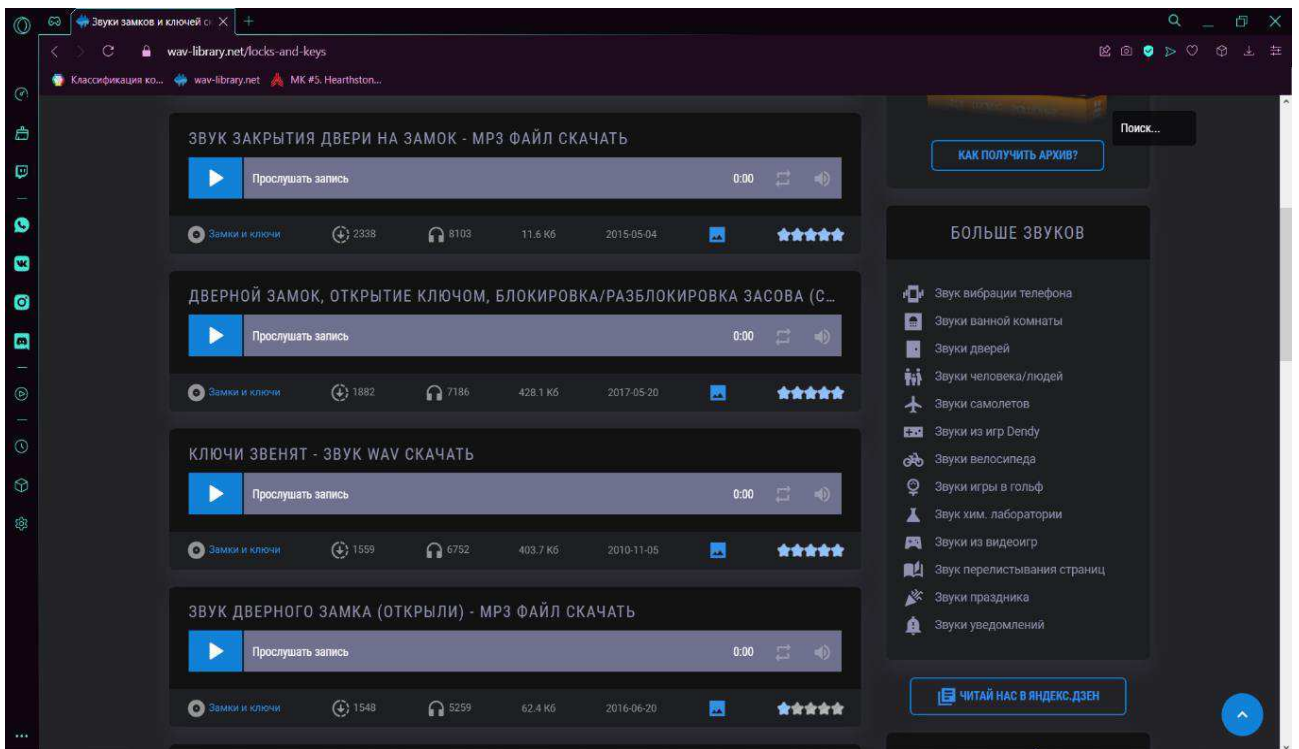


Рисунок 23 – Предлагаемые звуки

После загрузки всех необходимых звуков отредактировали каждый звук в Audacity. Для этого необходимо переместили звуковой файл в программу для создания ее стереотрека, как показано на рисунке 24.

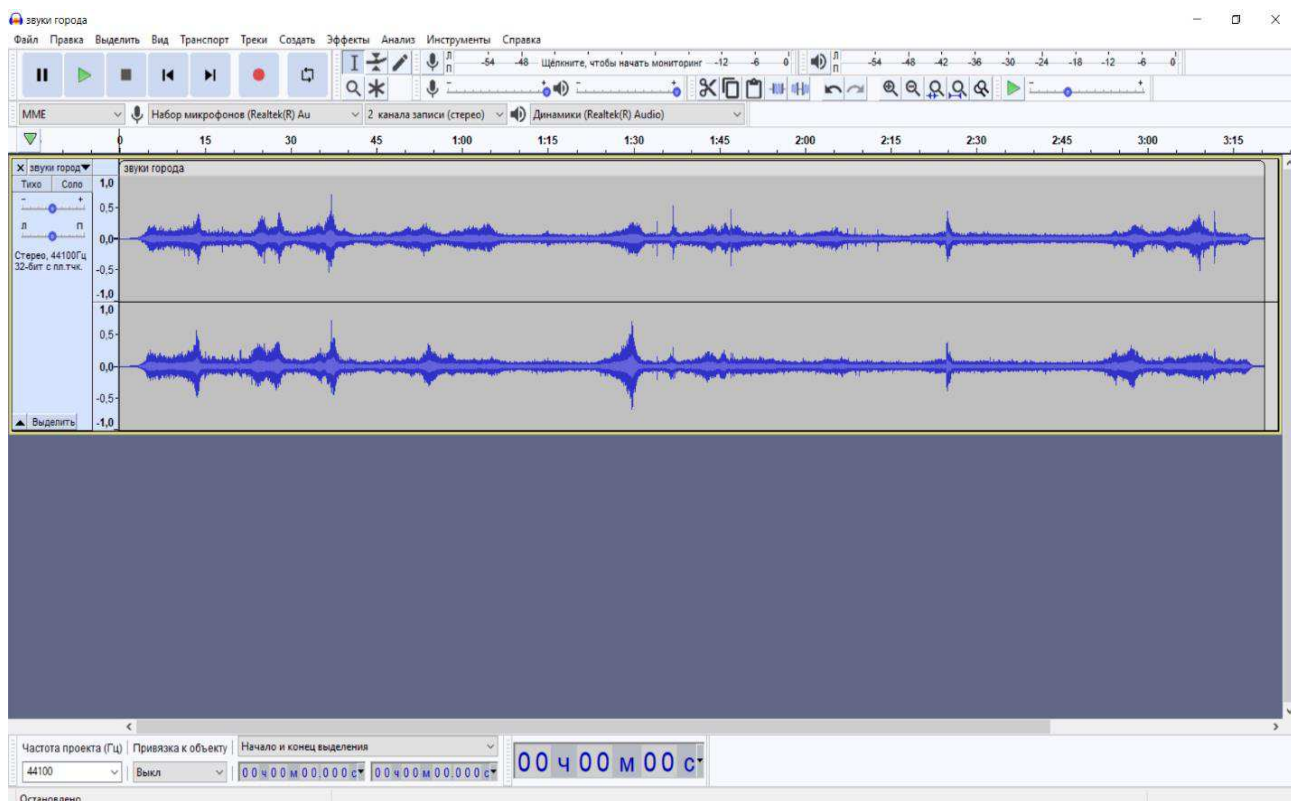


Рисунок 24– Звуковая дорожка в Audacity

Затем прослушали дорожку, вырезали нужную часть на отдельный стереотрек, удалили ненужное и экспортировали оставшийся стереотрек в формате wav. Это пришлось проделать со всеми скачанными звуками. Затем готовые обрезанные звуки были добавлены в проект. Для этого перенесли все звуки в область Assets в Unity, как показано на рисунке 25.

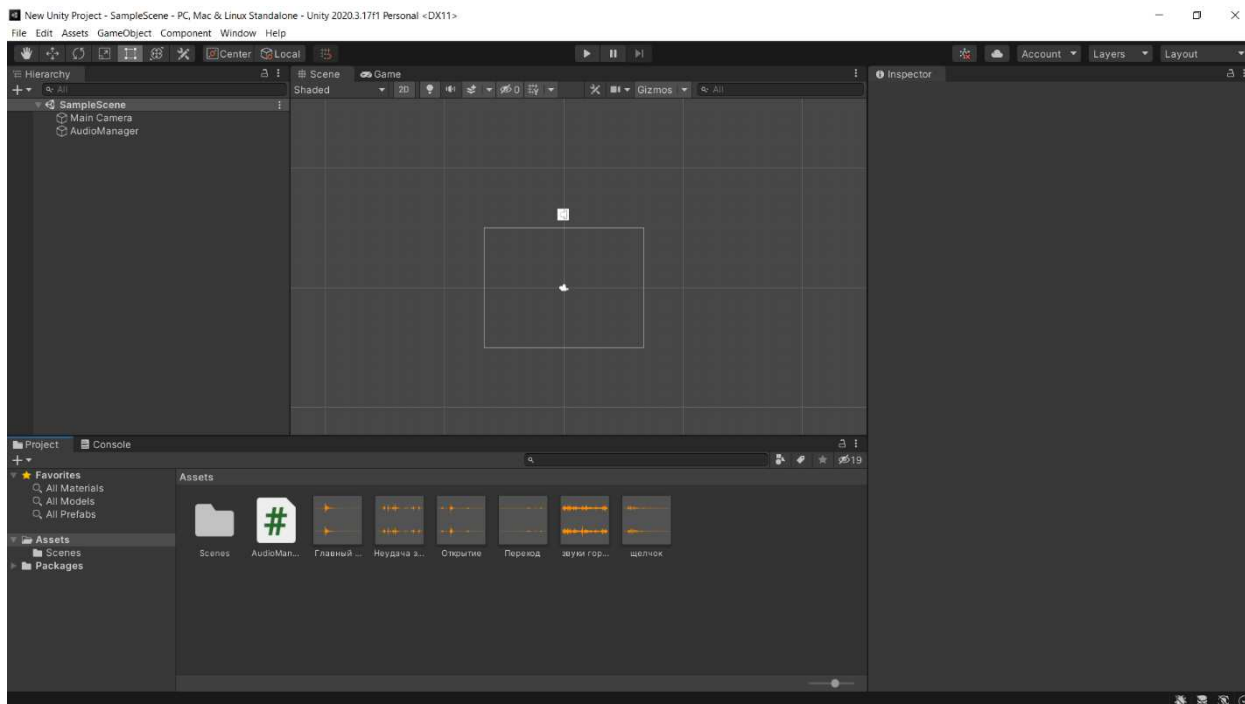


Рисунок 25 –Окно с необходимыми звуковыми файлами в проекте на Unity

После загрузки всех необходимых звуков на пустой сцене был создан объект. В Unity есть 7 видов объектов: 2D объект, 3D объект, эффект, аудио, видео, свет и интерфейс. Для работы со звуком был выбран объект аудио: AudioSource. В нем можно настроить громкость, ориентацию, тональность и дистанцию, на которой слышно звук. Объект представлен на рисунке 26.

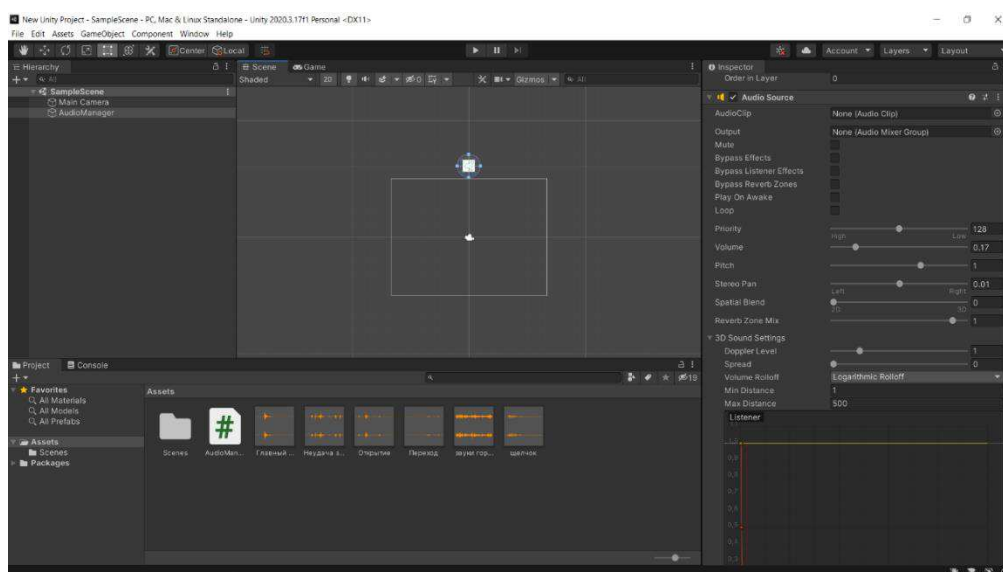
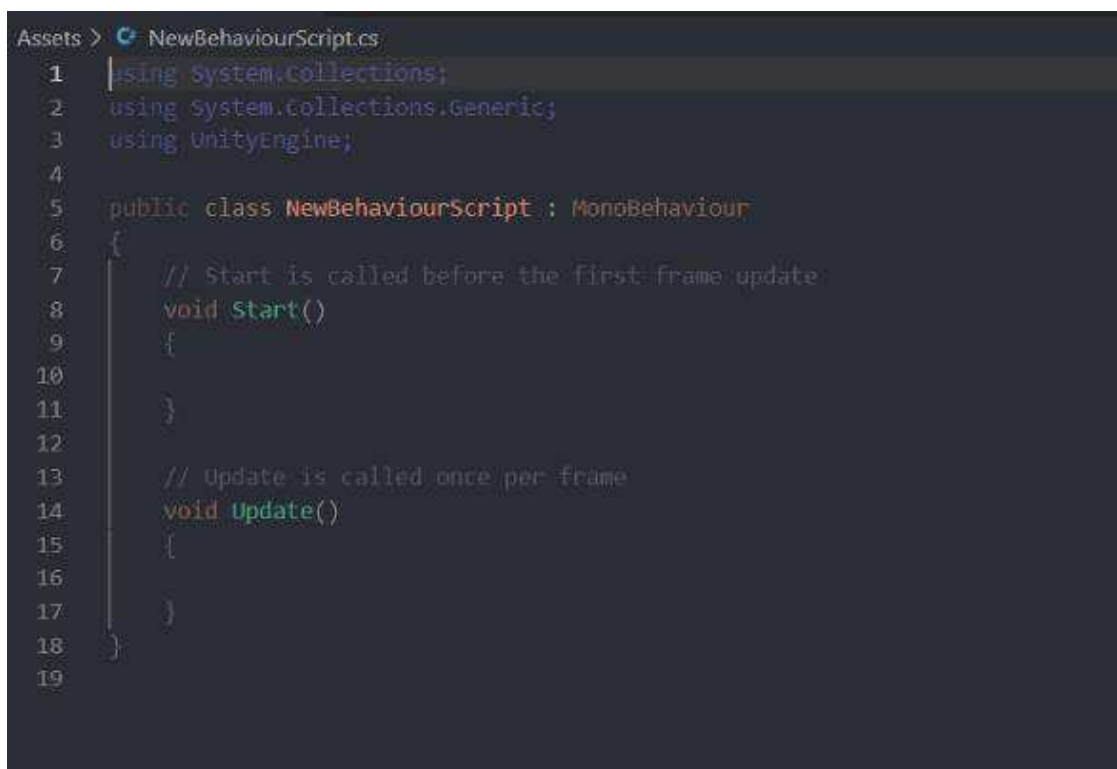


Рисунок 26 – Объект AudioSource

Затем в созданном объекте, в области Assets необходимо было создать скрипт. Для этого нужно нажать правой кнопкой мыши на пустое место в области, нажать кнопку "Create" и выбрать "C# Script". После создания скрипта необходимо было его назвать и теперь его можно открывать. Новый скрипт открывается в VisualStudioCode, как показано на рисунке 27.



```
Assets > NewBehaviourScript.cs
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  public class NewBehaviourScript : MonoBehaviour
6  {
7      // Start is called before the first frame update
8      void Start()
9      {
10
11     }
12
13     // Update is called once per frame
14     void Update()
15     {
16
17     }
18 }
19
```

Рисунок27 – Скриптв Visual Studio Code

При создании нового скрипта Unity сам создает его структуру и некоторые из главных методов. Методы в Unity отвечают за последовательность выполнения кода в игре. Выделяют такие основные методы как: Awake, Start, FixedUpdate, Update, OnDisable, OnEnable и OnDestroy. Awake отвечает за события, которые произойдут, как только объект появится на сцене. Все, что находится в методе Start, произойдет в момент запуска скрипта. Update и FixedUpdate выполняют одинаковую функцию и срабатывают в каждый момент времени с той разницей, что FixedUpdate срабатывает через определенный промежуток времени, а время срабатывания Update зависит от скорости обновления экрана. OnDisable и

OnEnable срабатывают, когда объект не виден игроку и когда виден соответственно. OnDestroy срабатывает в момент уничтожения объекта. Более подробная структура и последовательность выполнения методов показана на рисунке 28.

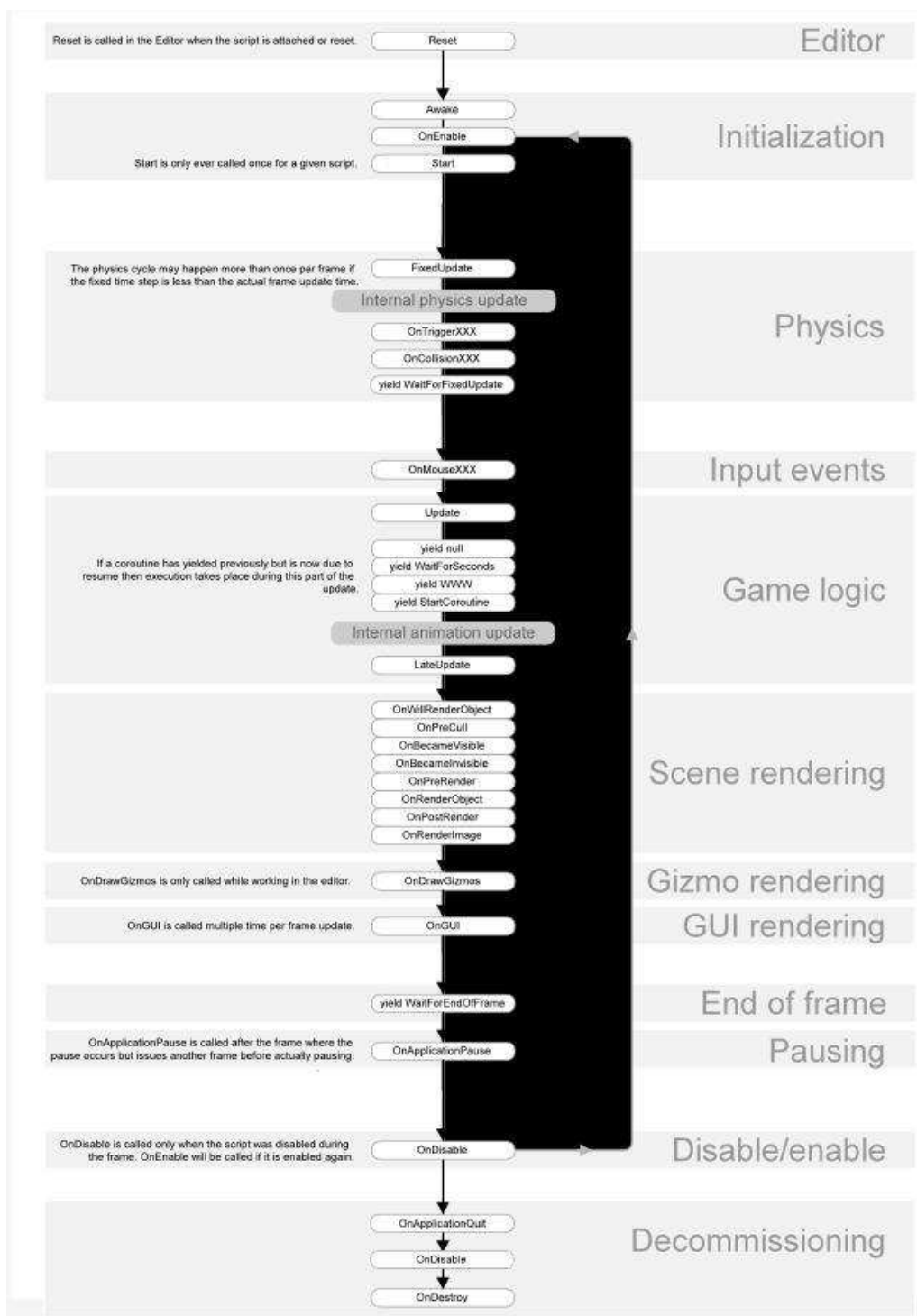


Рисунок 28 –Последовательность срабатывания методов

Также Unity предлагает различные варианты описания переменных. При объявлении переменной достаточно указать тип `Public` для того, чтобы данную переменную можно было редактировать не только через скрипт, но и в самом Unity.

Так как объект `AudioSource` не позволяет сразу использовать добавленные звуки по нажатию клавиши, это необходимо было исправить через скрипт. Для этого пришлось создать несколько переменных, которые отвечают за редактирование звука через инспектор Unity и количество добавленных звуков, что показано на рисунках 29 и 30.

```
public AudioClip[] audioClips;  
  
public static AudioManager instance;  
  
private AudioSource player;
```

Рисунок 29 – Создание переменных

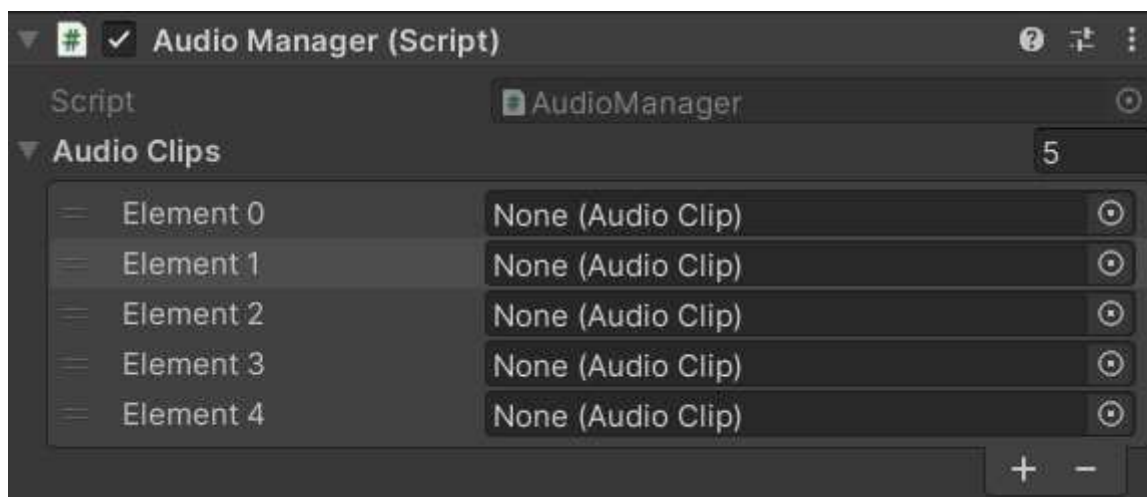


Рисунок 30 – Переменные, отображаемые в Unity

После создания переменных, в них можно сразу добавлять загруженные в Unity звуки, как показано на рисунке 31.

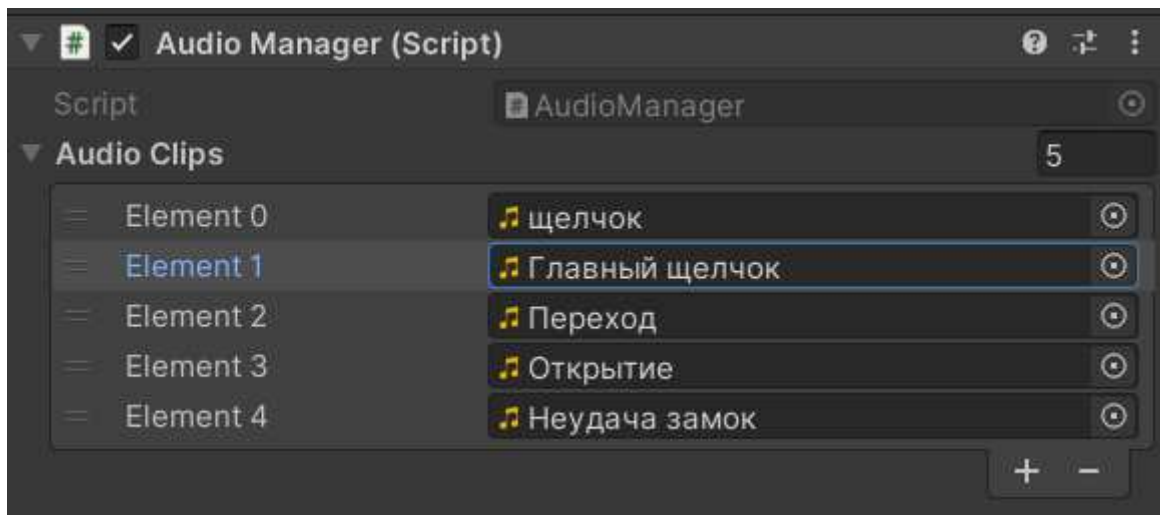


Рисунок 31 – Добавленные звуки

После этого можно начинать создавать структуру уровня, а добавленные звуки пригодятся позже. Для начала были созданы все необходимые переменные. Для первого уровня было создано 6 переменных, отвечающих за положение штифта в замке и 6 переменных для подсчета нажатий игрока. Создание переменных и присваивание им значений показано на рисунках 32 и 33.

```
int sh1;  
int sh2;  
int sh3;  
int sh4;  
int sh5;  
int sh6;  
int x1;  
int x2;  
int x3;  
int x4;  
int x5;  
int x6;  
int s;  
int t;
```

Рисунок 32 – Создание переменных

```
void Start ()
{
    sh1 = Random.Range(2,6);
    sh2 = Random.Range(2,6);
    sh3 = Random.Range(2,6);
    sh4 = Random.Range(2,6);
    sh5 = Random.Range(2,6);
    sh6 = Random.Range(2,6);

    x1 = 0;
    x2 = 0;
    x3 = 0;
    x4 = 0;
    x5 = 0;
    x6 = 0;
    s = 0;
    t = 0;
}
```

Рисунок 33 – Присваивание значений переменным

Присваивание значений переменным происходит в методе Start, так как присваивание должно происходить единожды.

Также необходима еще одна переменная для перехода от одного штифта к другому, и она будет срабатывать, когда игрок решил, что выставил штифт в нужное положение, и решит перейти к следующему.

После создания всех необходимых переменных приступили к сборке первого уровня. Он заключается в открытии навесного замка, представленного на рисунке 34.



Рисунок 34 – Навесной замок

Для этого необходимо было настроить взаимодействие игрока с переменными. Игрок будет выставлять штифт в нужное положение путем нажатия клавиши "вверх". Для считывания клавиш используется элемент (`Input.GetKeyDown (KeyCode.UpArrow)`). `Input.GetKeyDown` отвечает за нажатие клавиши. Если же изменить это на `Input.GetKeyUp`, то Unity будет считывать момент, когда игрок не нажал на клавишу, а отпустил ее. `KeyCode.UpArrow` отвечает за то, какую именно кнопку нажимает игрок.

Используя все вышеперечисленное, был написан программный код для установки первого штифта. На рисунке 35 показан код, который выставляет штифт в нужное положение и воспроизводит разные звуки в зависимости от места штифта.

```

Assets > AudioManager.cs
78
79     if (Input.GetKeyDown (KeyCode.Return))
80     {
81         |   if (AudioManager.GetClips ().Length > 3)
82         |   {
83         |       AudioManager.PlaySound (AudioManager.GetClips () [2]);
84         |   }
85         |   s = s + 1;
86     }
87
88
89
90
91
92     if (s == 0)
93     {
94
95         |   if (Input.GetKeyDown (KeyCode.UpArrow))
96         |   {
97
98
99         |       x1 = x1 + 1;
100
101         |       if (x1 == sh1)
102         |       {
103         |           |   if (AudioManager.GetClips ().Length > 1)
104         |           |   {
105         |           |       AudioManager.PlaySound (AudioManager.GetClips () [1]);
106         |           |   }
107         |           |   }
108         |           |   }
109         |           |   else
110         |           |   {
111         |           |       |   if (AudioManager.GetClips ().Length > 0)
112         |           |       |   {
113         |           |       |       AudioManager.PlaySound (AudioManager.GetClips () [0]);
114         |           |       |   }
115         |           |       |   }
116         |           |       |   }
117         |           |       |   }

```

Рисунок 35 – Код первого штифта

Аналогичные действия были проделаны с остальными штифтами, заменяя переменные.

На втором уровне используется круглый, механический, кодовый замок и ПИН-код. Пример подобного замка представлен на рисунке 36.



Рисунок 36 – Круглый, механический, кодовый замок

Игроку необходимо будет три раза повернуть кольцевой механизм замка, подсчитать количество щелчков при каждом повороте и ввести эти значения друг за другом без пробелов как ПИН-код в текстовое поле. Для реализации этого алгоритма были созданы три переменные, отвечающие за начальное положение, и три переменные, отвечающие за конечное положение. Это показано на рисунке 37.

```
sh1 = Random.Range(5,12);  
sh2 = Random.Range(5,12);  
sh3 = Random.Range(5,12);  
  
x1 = 0;  
x2 = 0;  
x3 = 0;
```

Рисунок 37 – Создание переменных

После этого, как и на первом уровне, прописали изменение переменных по нажатию определенных клавиш.

Затем для ввода ПИН-кода добавили специальное поле InputField, что показано на рисунке 38.

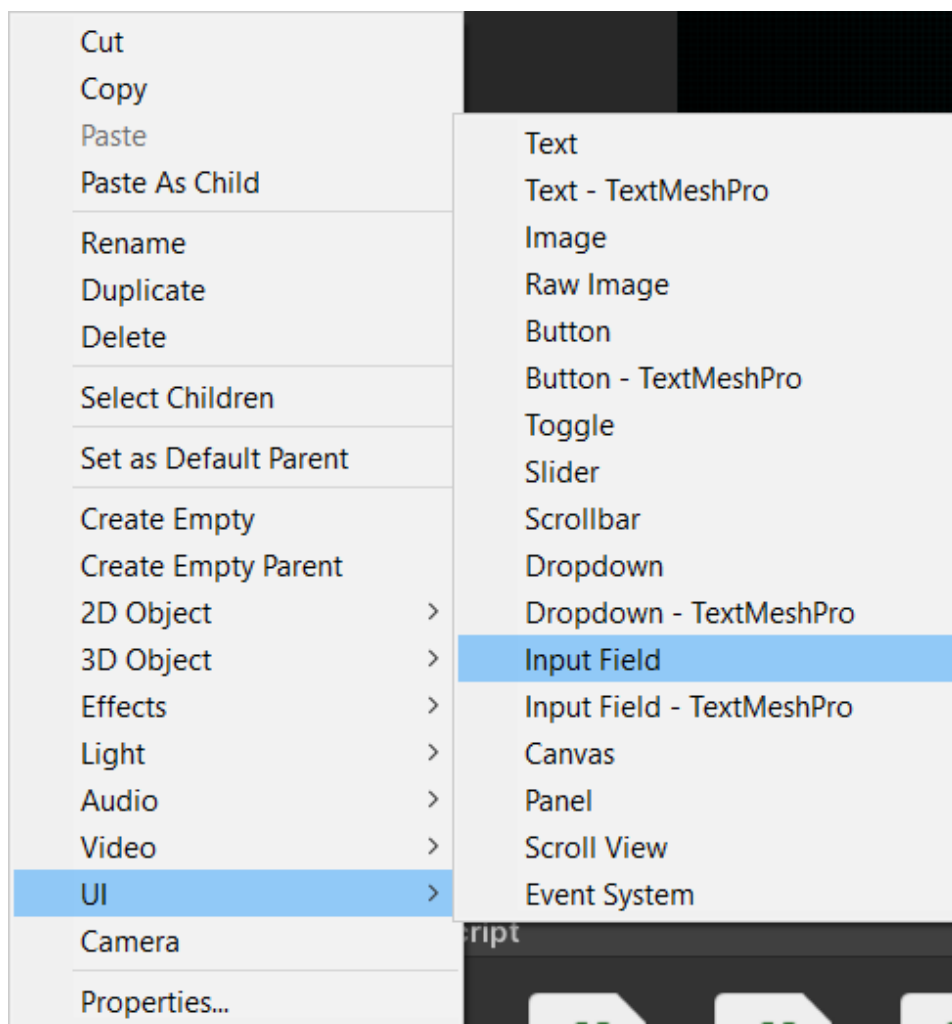


Рисунок 38 – Добавление поля InputField

Для корректной его работы, добавили кнопку, при нажатии на которую происходит считывание данных из поля и их сравнение. Для этого был написан скрипт, который воспроизводится при нажатии на кнопку. Чтобы не создавать отдельный скрипт, в уже существующем скрипте создали метод, к которому будет обращаться кнопка. На рисунке 39 представлен написанный метод.


```
Assets > Script > AudioManager3.cs
259     }
260 }
261
262
263
264
265 }
266
267 public void Con()
268 {
269     Debug.Log(InputNum.text);
270     string number = InputNum.text;
271     Debug.Log(number);
272     string str = shs.ToString();
273     if (number == str)
274     {
275         SceneManager.UnloadSceneAsync(4);
276         SceneManager.LoadScene(6);
277     }
278     else
279     {
280         SceneManager.UnloadSceneAsync(4);
281         SceneManager.LoadScene(5);
282     }
283
284
285
286 }
287
288
289 private void Update ()
290 {
291
292     if (t==0)
293     {
294         zamok();
295     }
296     else
297     {
298
```

Рисунок 39 – Метод для вызова кнопки

Затем в настройках кнопки добавили объект, к которому прикреплен скрипт, и выбрали созданный метод как показано на рисунке 40.

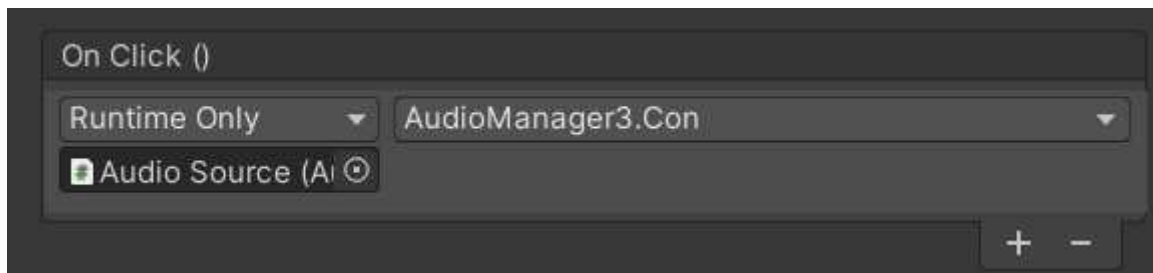


Рисунок 40 – Настройка кнопки

Теперь при нажатии на кнопку скрипт проверяет введенные данные игрока и завершает уровень.

Третий уровень заключается в прохождении небольшого лабиринта. Для его создания было принято решение использовать массив, где нулем обозначены границы лабиринта, единицами – маршрут, а двойкой- выход из лабиринта. На рисунке 41 представлен пример созданного массива.

```
int[,] Array = new int [,]
{
    {0,0,0,2,0,0,0},
    {0,0,0,1,1,1,0},
    {0,0,0,0,0,1,0},
    {0,0,1,1,1,1,0},
    {0,0,1,0,0,0,0},
    {0,0,1,0,0,0,0},
    {0,0,0,0,0,0,0}
};
```

Рисунок 41 – Пример созданного массива

Затем задали начальное положение игрока, и прописали перемещение по массиву. На рисунке 42 представлено перемещение по массиву и выход из него при достижении цифры 2.

```
Assets > Script > AudioManager2.cs
117     }
118     x = x + 1;
119 }
120 else
121 {
122     if (AudioManager2.GetClips ().Length > 0)
123     {
124         AudioManager2.PlaySound (AudioManager2.GetClips () [0]);
125     }
126 }
127 Debug.Log(Array[y,x]);
128 }
129 if (Input.GetKeyDown (KeyCode.RightArrow)){
130     x = x + 1;
131     if (Array[y,x] == 0)
132     {
133     {
134         if (AudioManager2.GetClips ().Length > 1)
135         {
136             AudioManager2.PlaySound (AudioManager2.GetClips () [1]);
137         }
138         x = x - 1;
139     }
140     else
141     {
142         if (AudioManager2.GetClips ().Length > 0)
143         {
144             AudioManager2.PlaySound (AudioManager2.GetClips () [0]);
145         }
146     }
147     Debug.Log(Array[y,x]);
148 }
149 if (Array[y,x] == 2){
150     SceneManager.UnloadSceneAsync(3);
151     SceneManager.LoadScene(6);
152 }
153
154
155 }
156
```

Рисунок 42 – Перемещение и выход из лабиринта

2.3 Разработка меню

После создания уровней было создано главное меню игры. Для этого в иерархии создали новую сцену и добавили на нее кнопки. Чтобы кнопки работали, был написан скрипт, который отвечает за перемещение между сценами. Скрипт представлен на рисунке 43.

```
Scene.cs x
Assets > Script > Scene.cs
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.SceneManagement;
5
6 public class Scene : MonoBehaviour
7 {
8     public void NextLevel(int _sceneNumber)
9     {
10         SceneManager.LoadScene(_sceneNumber, LoadSceneMode.Additive);
11     }
12 }
13
14
15
```

Рисунок 43 – Скрипт перехода между сценами

Этот скрипт создает глобальную переменную, к которой можно обращаться вне скрипта. Затем был создан объект и к нему добавлен этот скрипт, как показано на рисунке 44.

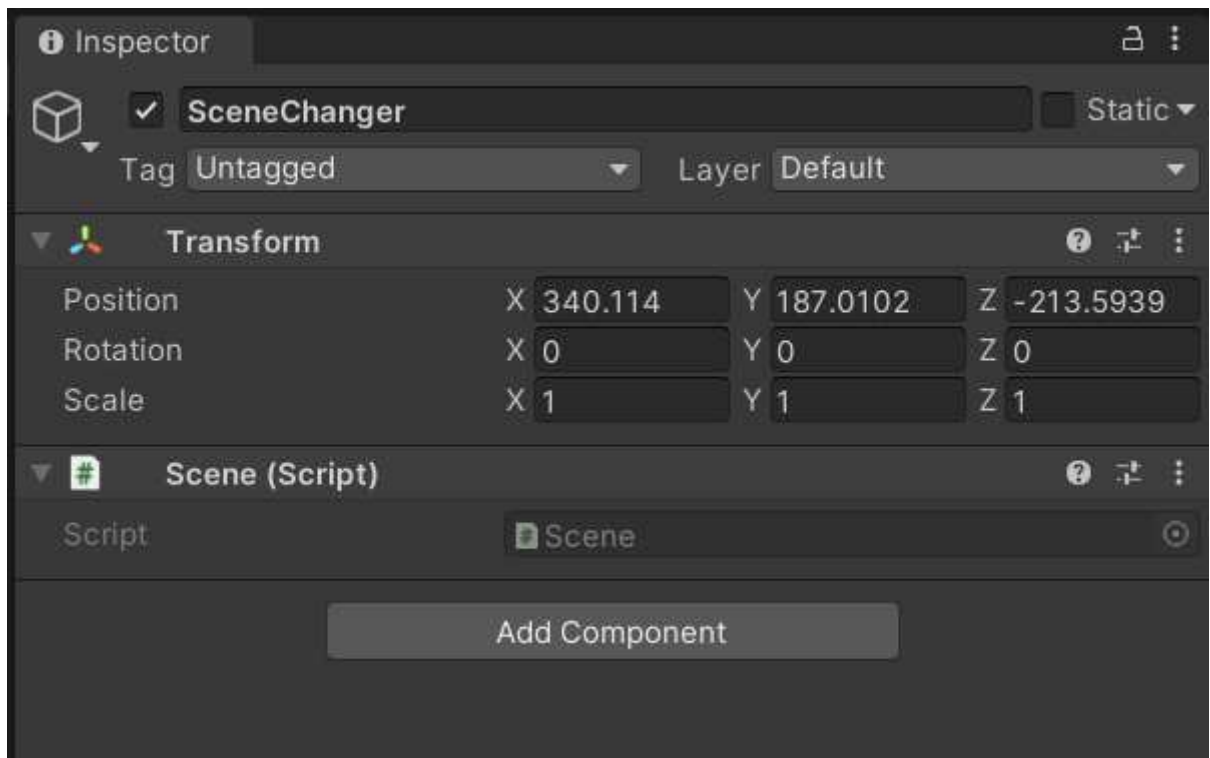


Рисунок 44 – Объект с импортированным к нему скриптом

У каждой добавленной кнопки есть поле, где можно прописать действие, которое будет происходить при нажатии. В это поле перенесли созданный ранее объект со скриптом и в выпадающем меню выбрали глобальную переменную из скрипта, как показано на рисунке 45.

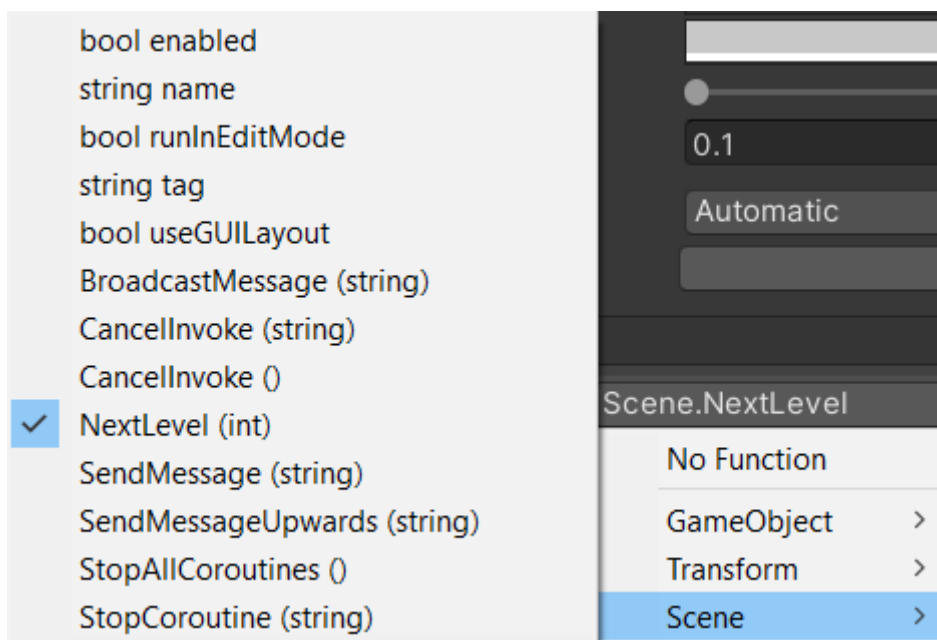


Рисунок 45 – Выбор переменной

Затем все сцены добавили в настройку File>BuildSettings и запомнили их номера, как показано на рисунке 46.

При добавлении сцен каждая сцена получает строго определенный порядковый номер. В процессе разработки при оформлении переходов между сценами пользователю необходимо учитывать этот порядок. Указанный порядковый номер является уникальным идентификатором соответствующей сцены.

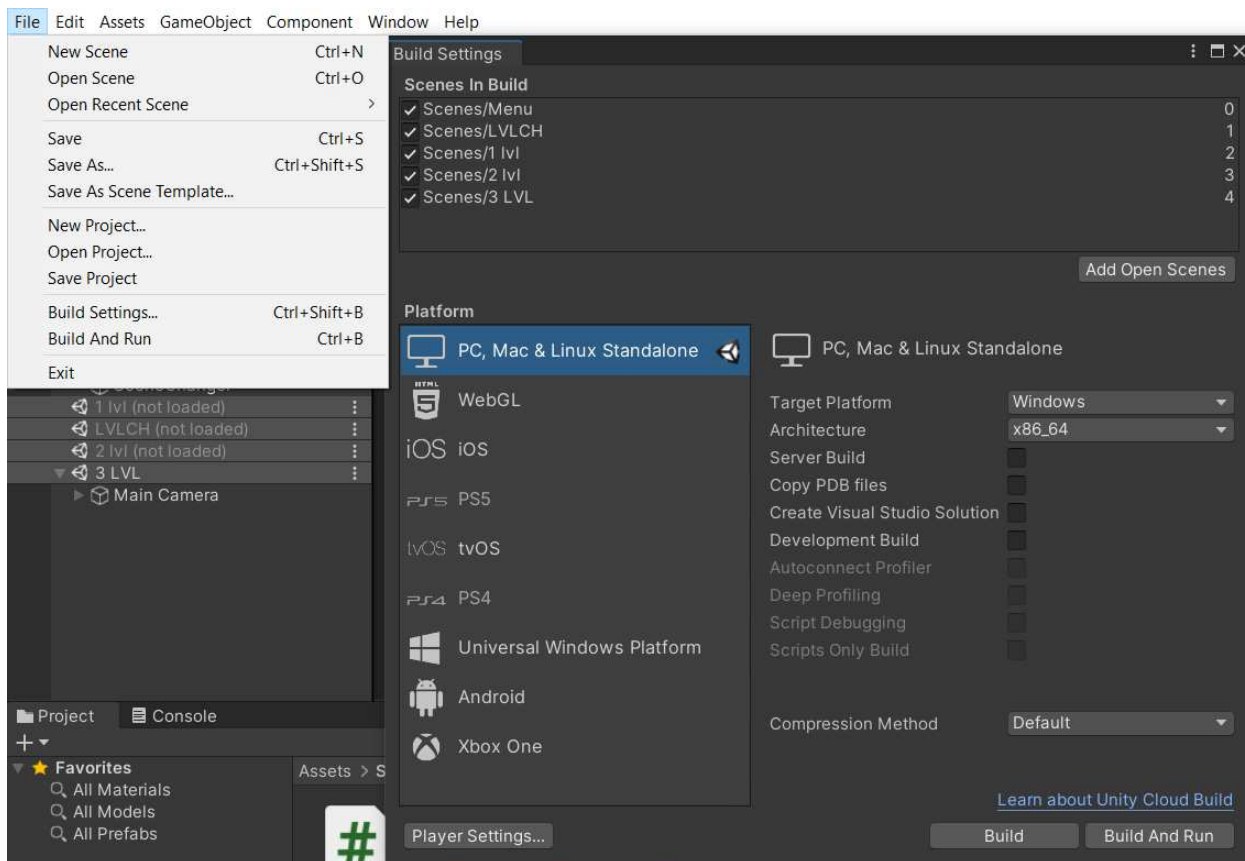


Рисунок 46– Настройка BuildSettings

Затем в поле выбора сцены при нажатии на кнопку ввели число, соответствующее порядку в настройке, как показано на рисунке 47.

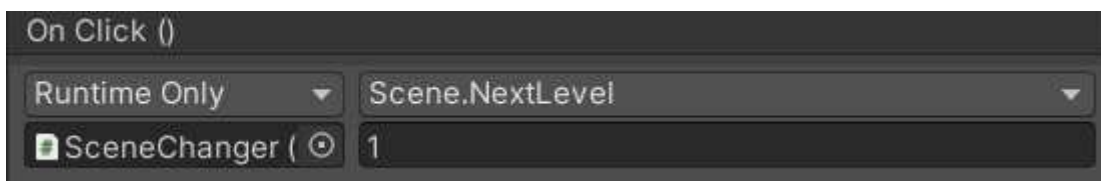


Рисунок 47 – Настройка кнопок основного меню

Теперь при нажатии на кнопку происходит смена сцены в игре. После этого также настроили остальные кнопки.

На рисунке 48 представлено готовое главное меню.

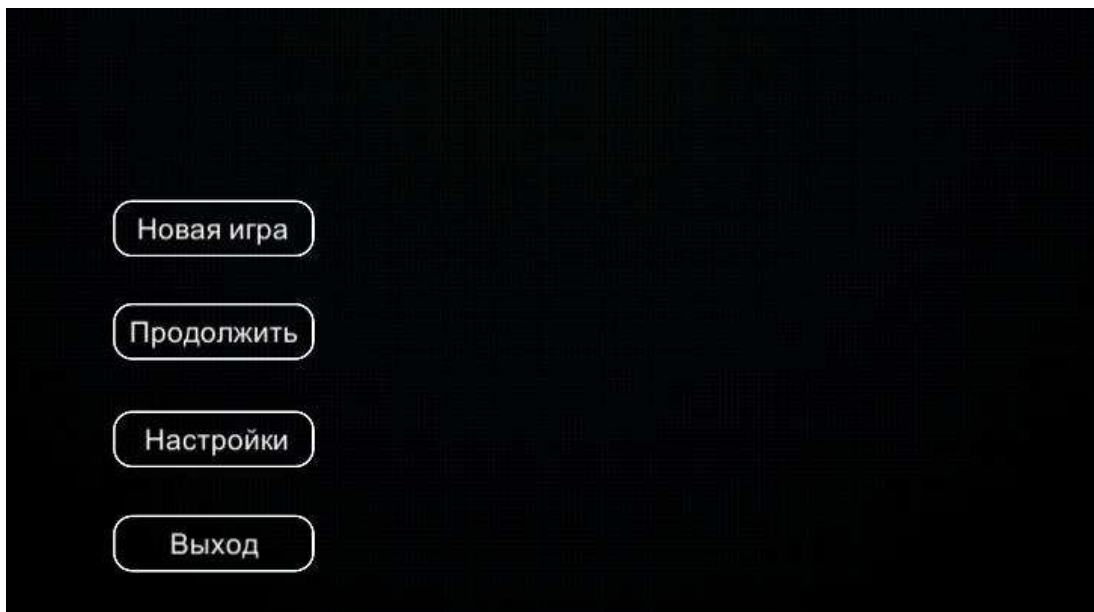


Рисунок 48 – Главное меню

Аналогичным образом было создано меню выбора уровней, представленное на рисунке 49.

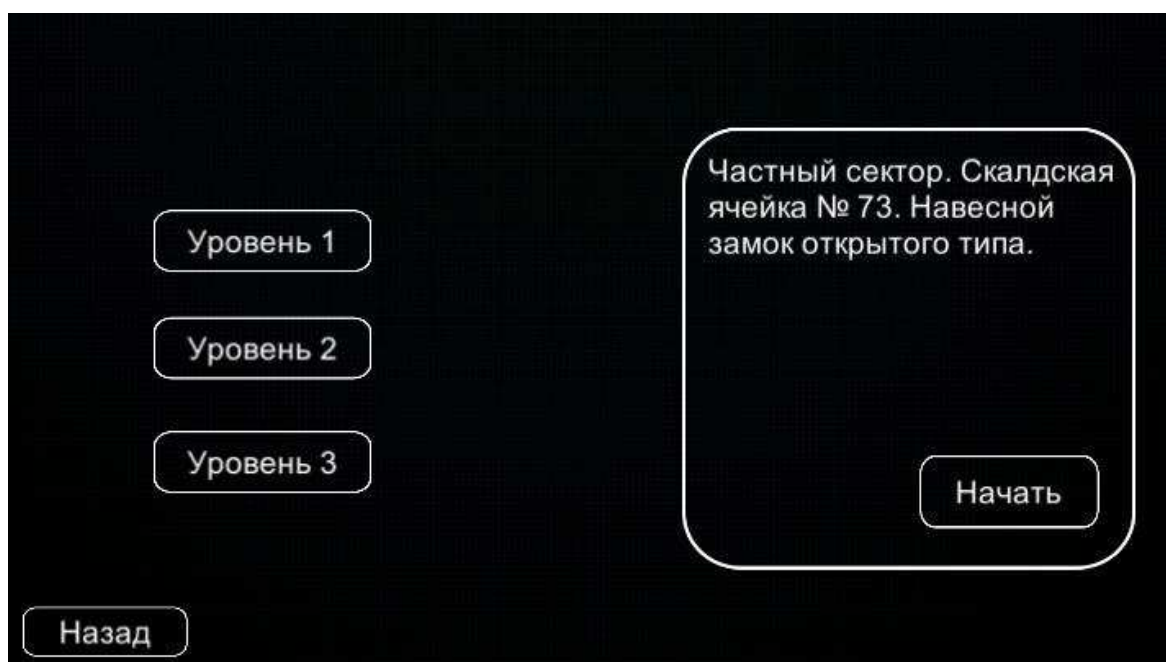


Рисунок 49 – Меню выбора уровней

2.4 Выводы по разделу «Разработка демо-версии компьютерной игры»

В данном разделе был описан процесс установки необходимого программного обеспечения и настройки его параметров. На платформе Unity средствами языка программирования C# было создано главное меню игры, меню выбора уровней и три игровых уровня. Представлены программные коды кнопок и игровых элементов, позволяющие управлять действиями игрока.

3 Оценка экономической эффективности проекта

3.1 Подбор оборудования и программного обеспечения для реализации проекта

Для разработки компьютерной игры необходимо подобрать определенное программное обеспечение и комплектующие для персонального компьютера. Комплектующие компьютера представлены в таблице 3.

Таблица 3 – Комплектующие компьютера

Комплектующие	Название	Цена в руб.
Материнская плата	GIGABYTE H310M S2 2.0	4 150
Процессор	Core i9-9900K BOX	37 999
Видеокарта	MSI GeForce RTX 2070 Super GAMING TRIO [RTX 2070 SUPER GAMING TRIO]	45 999
Оперативная память	AMD Radeon R9 Gamer Series [R9S432G3206U2K] 32 ГБ	10 999
Кулер	Cooler Master MasterLiquid 240P MIRAGE	11 999
Блок питания	Chieftec Element 700W [ELP-700S]	4 650

Окончание таблицы 3

Жесткий диск	960 ГБ SSD-накопитель Qumo Novation 3D [Q3DT-960GAEN]	8 350
Корпус	DEEPCOOL MATREXX 50 ADD-RGB 3F [DP-ATX-MATREXX50-AR-3F-US] черный	4 599
Звуковая карта	ASUS STRIX SOAR	6 799
Монитор	Samsung F27T350FHI [LF27T350FHIXCI]	14 699
Клавиатура	Redragon Visnu	2 699
Мышь	A4Tech Bloody W60 Max красный	2 599
Проводная гарнитура	2.1 F&D F550X	5 199
Итоговая стоимость		160 740

Средний срок эксплуатации данного компьютера составляет 7 лет с учетом изнашивания отдельных его элементов, постоянного развития технологий и, как следствие, устаревания данного оборудования.

Расчет амортизации необходим для того, чтобы узнать сумму затрат на оборудование на период разработки игры.

Для расчета амортизации был выбран способ списания стоимости по сумме чисел лет срока полезного использования. Расчет производился по формуле

$$A = C \times \text{ЧЛ} \div \sum \text{ЧЛ} \div 12, \quad (1)$$

где A – амортизация;

C – начальная стоимость;

ЧЛ – число лет, остающихся до конца срока полезного использования;

$\sum \text{ЧЛ}$ – сумма чисел лет срока полезного использования.

Ускоренный способ выплаты позволяет адаптироваться для изменчивого рынка. Срок эксплуатации 7 лет (84 м). Расчет амортизации представлен в таблице 4.

Таблица 4 – Расчет амортизации

Год	Проценты выплат в год	Ежегодная амортизация в рублях	Ежемесячная амортизация в рублях
2022	25%	40 185,00	3 348,75
2023	21%	34 444,29	2 870,36
2024	18%	28 703,57	2 391,96
2025	14%	22 962,86	1 913,57
2026	11%	17 222,14	1 435,18
2027	7%	11 481,43	956,79
2028	4%	5 740,71	478,39
Итого		160 740,00	

Для создания игры потребуется специальное программное обеспечение, представленное в таблице 5.

Таблица 5 – Необходимое программное обеспечение

Программное обеспечение	Цена в рублях
Unity	Бесплатно
Krita	Бесплатно
Операционная система Microsoft Windows 10 Home	14 000
Visual studioCode	Бесплатно
Kaspersky cloud free	Бесплатно
Audacity	Бесплатно

Операционная система будет использоваться на протяжении всего жизненного цикла компьютера, следовательно, срок амортизации – 7 лет.

Для расчета амортизации программного обеспечения был выбран линейный способ. Расчет был произведен по формуле

$$A = C \div \text{СПИ} \div 12, \quad (2)$$

где A – то же, что и в формуле (1);

C – то же, что и в формуле (1);

СПИ – срок полезного использования в годах.

В таблице 6 представлен расчет амортизации программного обеспечения.

Таблица 6 – Расчет амортизации программного обеспечения

Год	Проценты выплат в год	Ежегодная амортизация в рублях	Ежемесячная амортизация в рублях
2022	14.3%	2000	167
2023	14.3%	2000	167
2024	14.3%	2000	167
2025	14.3%	2000	167
2026	14.3%	2000	167
2027	14.3%	2000	167
2028	14.3%	2000	167
		14 000	

Сроки разработки демоверсии игры были рассчитаны, исходя из анализа длительности демоверсии. Сроки представлены в таблице 7.

Таблица 7 – Сроки разработки демоверсии

Вид деятельности	Длительность в днях
Подготовка, анализ рынка	5
Разработка	30
Тестирование	15
Доработка	15

3.2 Капитальные затраты

Капитальные затраты не носят разовый характер и вычисляются по следующей формуле

$$K = K_{\text{пр}} + K_{\text{ТС}} + K_{\text{ЛС}} + K_{\text{ПО}} + K_{\text{ИО}} + K_{\text{ОБ}} + K_{\text{ОЭ}}, \quad (3)$$

где $K_{\text{пр}}$ – затраты на проектирование ИС;
 $K_{\text{ТС}}$ – затраты на технические средства управления;
 $K_{\text{ЛС}}$ – затраты на создание линий связи локальных сетей;
 $K_{\text{ПО}}$ – затраты на программные средства;
 $K_{\text{ИО}}$ – затраты на формирование информационной базы;
 $K_{\text{ОБ}}$ – затраты на обучение персонала;
 $K_{\text{ОЭ}}$ – затраты на опытную эксплуатацию.

3.2.1 Затраты на проектирование

Затраты на проектирование рассчитываются по следующей формуле

$$K_{\text{пр}} = K_{\text{ЗП}} + K_{\text{ИПС}} + K_{\text{СВТ}} + K_{\text{Проч}}, \quad (4)$$

где $K_{\text{ЗП}}$ – затраты на заработную плату;
 $K_{\text{ИПС}}$ – затраты на инструментальные программные средства;
 $K_{\text{СВТ}}$ – затраты на средства вычислительной техники;
 $K_{\text{Проч}}$ – прочие затраты.

$K_{\text{ЗП}}$ – это величина фонда оплаты труда (ФОТ), который включает в себя заработную плату, а также дополнительные средства для обязательных отчислений во внебюджетные фонды (пенсионное, социальное и

медицинское страхование). Величина данных отчислений равна 30,2% от фонда заработной платы за весь период работы.

Заработная плата рассчитывается на срок работы на два месяца. В заработной плате уже учтены районный и северный коэффициент. Заработная плата разработчика на unity – 40 000 руб.

$$40\,000 * 2 = 80\,000 \text{ руб.}$$

В фонд оплаты труда входят отчисления во внебюджетные фонды:

$$K_{зп} = 80\,000 \text{ руб.} * 1,302 = 104\,160 \text{ руб.}$$

$K_{ипс}$ рассчитывается, исходя из стоимости использования программного обеспечения.

Амортизационная стоимость программного обеспечения на месяц рассчитана ранее, на два месяца составит:

$$K_{ипс} = 167 * 2 = 334 \text{ руб.}$$

$K_{свт}$ это затраты на оборудование за период работы над проектом. 3 348 руб. это ежемесячные амортизационные отчисления за первый год использования. Так как компьютер будет использоваться 2 месяца

$$K_{свт} = 3\,348 \text{ руб.} * 2 = 6\,696 \text{ руб.}$$

Расчет прочих расходов. На прочие расходы принято выделять до 5% от суммы проектных затрат. Которые включают в себя затраты на электроэнергию, канцелярские расходы, оплату связи и прочее:

$$K_{проч} = (104\,160 + 334 + 6696) * 5\% = 5\,559 \text{ руб.}$$

В таблице 8 и на рисунке 50 представлена структура проектных затрат.

Таблица 8 – Структура проектных затрат

Показатель	Затраты в рублях
$K_{зп}$	104 160
$K_{ипс}$	334
$K_{свт}$	6 696
$K_{проч}$	5 559
$K_{пр}$	116 749

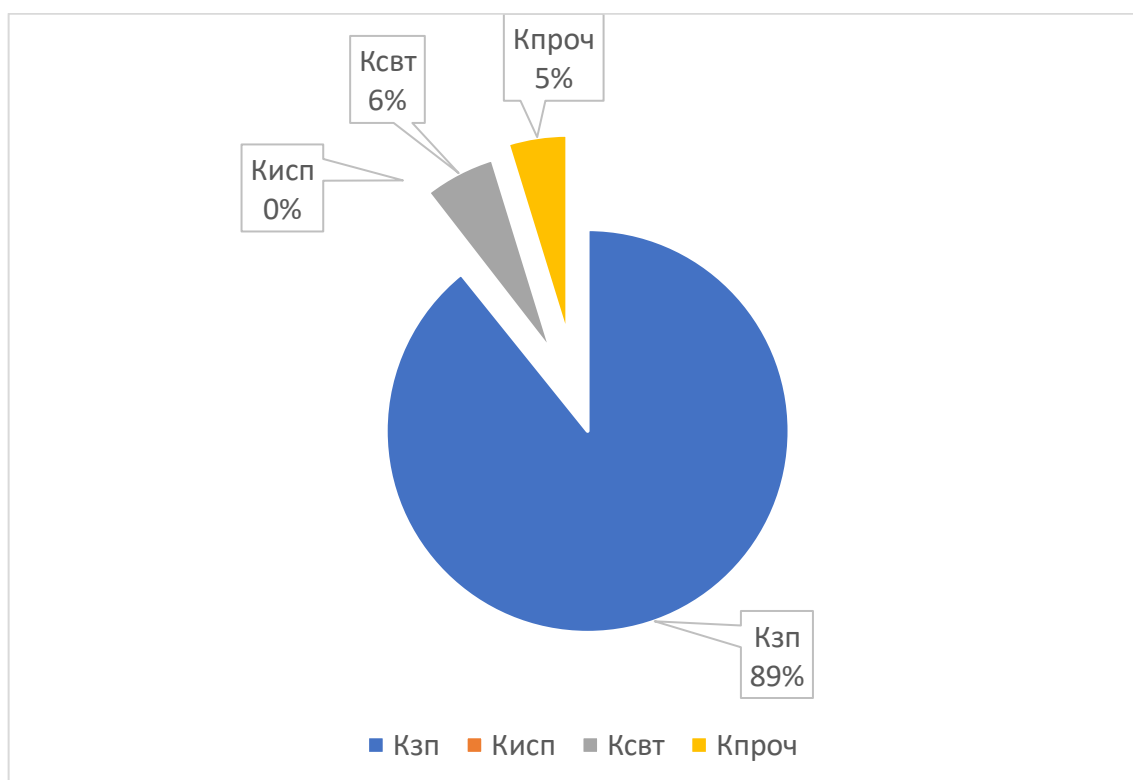


Рисунок 50 – Структура проектных затрат

Данные в таблице 8 и на рисунке 50 показывают, что большую часть проектных затрат составляют затраты на заработную плату разработчика.

$K_{тс} = 0$ – нет необходимости в закупке дополнительного оборудования, так как готовый проект будет запускаться на компьютерах пользователей.

$K_{лс} = 0$ – так как продукт распространяется посредством сети Интернет то этим программным продуктом может пользоваться любой, у кого есть доступ к сети интернет.

$K_{по} = 7\,460$ рублей (100\$, курс 10.02.22) – так как для внедрения необходимо выложить готовый программный продукт в онлайн-сервис цифрового распространения компьютерных игр Steam.

$K_{ио} = 0, K_{об} = 0, K_{оэ} = 0$, так как для обслуживания проекта и обучения работы с проектом нет необходимости в дополнительном персонале.

Исходя из всего вышеперечисленного,

$$K = 116\,749 \text{ руб.} + 7\,460 \text{ руб.} = 124\,209 \text{ руб.}$$

В таблице 9 и на рисунке 51 представлена структура капитальных затрат. Найденные ранее проектные затраты и составляют основную долю капитальных затрат.

Таблица 9 – Структура капитальных затрат

Показатель	Затраты в рублях
$K_{пр}$	116 749
$K_{тс}$	0
$K_{лс}$	0
$K_{по}$	7 460
$K_{ио}$	0
$K_{об}$	0
$K_{оэ}$	0

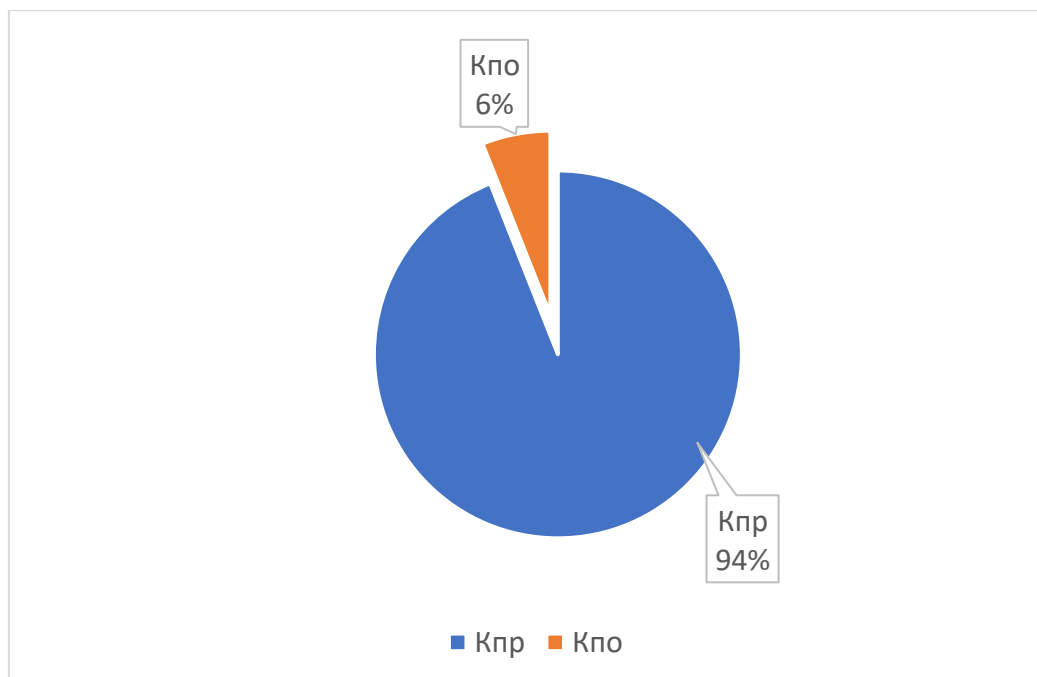


Рисунок 51 – Структура капитальных затрат

3.3 Эксплуатационные затраты

Так как итогом разработки является готовый проект, нет необходимости в его содержании и эксплуатации. В эксплуатационные затраты войдут затраты на рекламу и продвижение игры. Затраты представлены в таблице 10.

Таблица 10 – Прочие затраты

Платформа	Срок	Цена в рублях
ВКонтакте	4 месяца	38 700
ТикТок	4 месяца	38 700
Всего		77 400

3.4 Расчет совокупной стоимости по методике TCO

TCO (Total Cost of Ownership – совокупная стоимость владения) – методика, разработанная в конце 80-х годов XX века компанией Gartner

Group для расчета финансовых затрат на владение компьютерами. Она используется для того, чтобы рассчитать финансовые затраты на ИТ. Показатель совокупной стоимости владения ИС рассчитывается по формуле

$$TCO = DE + IC1 + IC2, \quad (5)$$

где DE (direct expenses) – прямые расходы;

$IC1$ (indirect costs) – косвенные расходы первой группы;

$IC2$ – косвенные расходы второй группы.

Для данного проекта косвенные расходы не являются значимыми, следовательно $TCO \approx DE$

$$DE = DE_1 + DE_2 + DE_3 + DE_4 + DE_5 + DE_6 + DE_7 + DE_8, \quad (6)$$

где DE_1 – капитальные затраты;

DE_2 – расходы на управление ИТ;

DE_3 – расходы на техническую поддержку АО и ПО;

DE_4 – расходы на разработку прикладного ПО внутренними силами;

DE_5 – расходы на аутсорсинг;

DE_6 – командировочные расходы;

DE_7 – расходы на услуги связи;

DE_8 – другие группы расходов.

Исходя из вышеперечисленных расчетов, получим:

$$DE_1 = 124\,209 \text{ руб.}$$

$DE_2 = 0$, так как итогом работы является готовый проект, и нет необходимости в его обслуживании, следовательно, нет необходимости содержать дополнительных сотрудников.

$DE_3 = 0$, так как готовый проект не нуждается в обслуживании, следовательно, нет необходимости в дополнительном оборудовании.

$DE_4 = 0$, так как результатом работы является готовый продукт, то нет необходимости в создании дополнительного программного обеспечения.

$DE_5 = 0$, так как при разработке проекта нет необходимости в привлечении сторонних лиц для консультации.

$DE_6 = 0$, так как сотрудникам не нужно ездить в командировки, следовательно, на это нет затрат.

$DE_7 = 0$, так как для общения с заказчиком будет использована сеть интернет, данные затраты входят в прочие при расчете DE_1 .

Предполагаемые расходы на продвижение игры (реклама) включены в состав прочих затрат:

$$DE_8 = 77\,400 \text{ руб.}$$

Косвенные затраты в проекте, в соответствии с анализом рисков проекта, приняты равными нулю.

$$IC_1 = 0$$

$$IC_2 = 0$$

Структура ТСО представлена в таблице 11 и на рисунке 37.

Таблица 11 – Расчет ТСО

Показатель	Затраты в рублях
DE_1	124 209
DE_2	0
DE_3	0
DE_4	0
DE_5	0

Окончание таблицы 11

DE_6	0
DE_7	0
DE_8	77 400
$IC1$	0
$IC2$	0
Всего	201 206

3.5 Оценка рисков проекта

Реализационный риск может иметь разную степень вероятности, так как от разных рисков могут быть разные воздействия на проект.

Риск, связанный с характеристиками проекта, имеет низкую вероятность, так как все возможные факторы учитываются на стадии планирования.

В таблице 12 приведен перечень рисков и путей их решения.

Таблица 12 – Анализ рисков

№п.	Группы рисков	Перечень рисков проекта	Уровень влияния риска на проект	Вероятность риска	Возможность предотвращения или снижения риска
1	Реализационный риск	Невостребованность продукта на рынке	Низкий	Средняя	Тщательное исследование рынка
2	Риск, связанный с характеристиками проекта	Поломка компьютера	Высокий	Низкая	Использование только необходимого ПО
3	Реализационный риск	Слив пиратской копии игры	Низкий	Средняя	Использование защиты от взлома
4	Реализационный риск	Провал рекламной компании	Низкий	Низкая	Тщательное исследование рынка
5	Риск, связанный с характеристиками проекта	Закрытие платформ, связанных с разработкой проекта	Высокий	Средняя	Поиск альтернативных платформ

Тщательное исследование рынка включает в себя анализ концептов популярных игр, изучение целевой аудитории игры, анализ современных маркетинговых стратегий продвижения игры. Во избежание взлома необходимо использовать технологии защиты от несанкционированного взлома, например Denuvo. При закрытии платформ, связанных с разработкой проекта, необходимо искать их аналоги, например: Steam – EpicGameStore, Unity – Godotengine, Patreon – Boosty. Во избежание поломки компьютера разработчик должен пользоваться только необходимым программным обеспечением, чтобы избежать случайного попадания вирусных программ на устройство.

В результате исследования рисков было выявлено, что разрабатываемый проект относится к группе среднего риска, значит дополнительные средства на преодоление рискованных ситуаций не требуются ($IC1$ и $IC2 = 0$).

3.6 Оценка экономического эффекта от разработки демо-версии

Представленный проект является демо-версией игры с целью оценки покупательского спроса и получения возможности прибыли с будущего проекта – разработки полноценной игры. Для сбора средств необходимо разместить демоверсию на краудфандинговой платформе, данная практика получения средств для разработки игры является широко распространенной среди независимых разработчиков.

Для обоснования вышесказанного необходимо рассчитать прогнозные величины возможного дохода, складывающейся из стоимости будущей полной версии игры и количества продаж копий игры.

Для определения стоимости игры были проанализированы игры с примерно подобным концептом разрабатываемой игры.

В таблице 13 представлено сравнение игр по таким критериям как: сюжет, геймплей, реиграбельность, атмосфера, стоимость.

Таблица 13 – Сравнение игр

Игра	Сюжет	Геймплей	Реиграбельность	Атмосфера	Стоимость в рублях
Blind Drive	6	4	4	6	259
Apotheorasis	5	5	3	7	205
Dark echo	3	7	5	6	79
Разработанная игра	6	7	4	7	349

С учетом того, что платформа Steam берет 30% с продаж. Игра будет продаваться за 349 рублей.

Для определения прогноза продаж полной версии игры, необходимо изучить данные ранее проанализированных игр и выяснить примерное количество их продаж.

Сайт SteamSpy позволяет получить информацию о количестве проданных копий в первый год после релиза. Данные представлены в таблице 14.

Таблица 14 – Количество купивших игру

Игра	Количество купивших игру
Blind Drive	10 000
Dark echo	35 000
Apotheorasis	10 000
Среднее количество игроков	19 000

На основе установленной стоимости и прогнозируемого количества продаж копий составлены пессимистичный прогноз на уровне одного процента продаж аналогичных проектов, оптимистичный на уровне 30 процентов.

Необходимо посчитать среднее значение по трем играм. После этого, на основе полученных данных составляется пессимистичный и

оптимистичный прогноз продаж будущей игры. Он рассчитывается по формуле

$$K \div 100 \times 30 \times Ц, \quad (7)$$

где K – количество купивших игру;

$Ц$ – цена за игру.

А пессимистичный подразумевает, что продажи составят лишь 1%. Он рассчитывается по формуле

$$K \div 100 \times Ц \quad (8)$$

Пессимистичный прогноз:

$$\frac{19000}{100} \cdot 245 = 46\,550 \text{ руб.}$$

Оптимистический прогноз:

$$\frac{19000}{100} \cdot 30 \cdot 245 = 1\,396\,500 \text{ руб.}$$

Исходя из всех вышеперечисленных расчетов, можно сделать вывод, что при необходимых затратах в 201 206 рублей и выполнении условий оптимистичного прогноза игра окупается более чем в 6 раз. Так же при коммерческом успехе игры будет заработан имидж, который поднимет продажи для следующих проектов.

При оценке интереса к демоверсии игры можно судить о развитии пессимистичного или оптимистичного прогноза для полной версии игры. В случае оптимистичного прогноза потребуются дополнительные небольшие

затраты для достижения устойчивого положительного эффекта. Так как игра рассчитана на звуковое сопровождение, главным ее элементом будет являться профессиональная озвучка внутриигровых персонажей. Готовая игра будет распространяться с локализацией для таких языков как: русский, английский, испанский, немецкий, французский, итальянский, китайский, японский, корейский. Для этого необходимо будет перевести сценарий на эти языки, и заказать его озвучку. Полная озвучка на одном языке будет стоить около 6 тысяч рублей, следовательно, озвучка на всех языках будет стоить около 54 тысяч.

Так же размещение демоверсии игры может привлечь потенциальных игроков и инвесторов, что поможет продвижению игры на первоначальной стадии после релиза.

Разработка демоверсии игры не является объектом прямого получения дохода, данный проект позволяет получить возможность для реализации полноценного проекта с получением прогнозируемых доходов при пессимистичном прогнозе

При пессимистичном прогнозе доход составит 46 550 руб., что не позволит окупить затраты на создание демоверсии игры, которые составили 201 206 руб. В случае реализации такого прогноза, в будущем следует проводить более глубокий анализ рынка игровой индустрии и провести анализ ошибок, допущенных в текущей разработке.

При оптимистичном прогнозе доход составит 1 396 500 руб., что позволяет окупить все затраты на разработку и вложить оставшиеся средства в будущий проект.

3.7 Выводы по разделу «Оценка экономической эффективности проекта»

В данном разделе были посчитаны затраты на реализацию проекта по методике ТСО, которые составили 201206 рублей. Также была найдена

стоимость будущей игры и спрогнозированы доходы при пессимистичном и оптимистичном итоге продаж игры. Были определены возможные риски реализации проекта и описаны пути их решения.

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной квалификационной работы была проанализирована предметная область и были сделаны следующие выводы:

- игровая индустрия стремительно развивается;
- порог входа как для игроков, так и для разработчиков очень низкий;
- существование множества различных игровых жанров дает свободу разработчикам в реализации своих идей.

Определен жанр и концепт разрабатываемой игры. В отличие от большинства других игр, создаваемая игра будет основана не на визуальной составляющей, а на звуковой. Большая часть взаимодействия игры с игроком будет производиться путем различных звуков.

Подходящей средой разработки был выбран игровой движок Unity, так как он удовлетворяет наиболее важным критериям таким как: количество обучающего контента и порог входа.

Демоверсией является три готовых игровых уровня, главное меню и меню выбора уровней. Демоверсия была написана на игровом движке Unity с поддержкой языка программирования C#. Для создания уровней были найдены и отредактированы необходимые звуки и написаны скрипты, отслеживающие нажатия пользователя на клавиатуре, и управляющие добавленными звуками.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. 3DNews: Valve будет бороться с поддельными коллекционными карточками Steam: официальный сайт. – 1997. – URL: <https://3dnews.ru/952383/valve-budetborotsya-s-poddelnimi-kollektsionnimi-kartochkami-steam> (дата обращения: 21.03.2022).
2. Adobe: официальный сайт. – 1985. – URL: <https://www.adobe.com/ru/products/audition.html> (дата обращения: 21.03.2022).
3. Audacity: официальный сайт. – 2000. – URL: <https://www.audacityteam.org> (дата обращения: 01.04.2022).
4. Blizzard: официальный сайт. – 1991. – URL: <https://www.blizzard.com/ru-ru/> (дата обращения: 21.03.2022).
5. DevTribe: Закрытие SteamGreenlight и восход SteamDirect: официальный сайт. – 2017. – URL: <https://devtribe.ru/p/media/steam-greenlight-and-steam-direct> (дата обращения: 21.03.2022).
6. DNS: Операционная система Windows 10: официальный сайт. – 1998. – URL: <https://www.dns-shop.ru/product/2314d81a413d3330/operacionnaa-sistema-microsoft-windows-10-home/characteristics/> (дата обращения: 28.03.2022).
7. DTF: Valve объявила о закрытии программы Steam Greenlight: официальный сайт. – 2016. – URL: <https://dtf.ru/gameindustry/4296-valve-obyavila-o-zakrytii-programmy-steam-greenlight> (дата обращения: 19.03.2022).
8. DTF: Инди игры: жанр, которого не существует: официальный сайт. – 2016. – URL: <https://dtf.ru/games/225632-indi-igry-zhanr-kotorogo-ne-sushchestvuet> (дата обращения: 21.03.2022).
9. EpicGamesStore: официальный сайт. – 2018. – URL: <https://store.epicgames.com/ru/> (дата обращения: 21.03.2022).
10. Gamer: Spacewar. Первая компьютерная игра в истории - экскурсия в прошлое: официальный сайт. – 2009. – URL:

<http://www.gamer.ru/everything/spacewar-pervaya-kompyuternaya-igra-v-istorii-ekskursiya-v-proshloe> (дата обращения: 21.03.2022)

11. Godot: официальный сайт. – 2007. – URL: <https://godotengine.org> (дата обращения: 21.03.2022).

12. HeadHunter: Разработчик на Unity: официальный сайт. – 2000. – URL: https://abakan.hh.ru/search/vacancy?clusters=true&no_magic=true&ored_clusters=true&enable_snippets=true&salary=&text=Unity+c%23&from=suggest_post (дата обращения: 28.03.2022).

13. ITisgood: Что такое ИДЕ (Интегрированная среда разработки): официальный сайт. – 2001. – URL: <https://itisgood.ru/2019/10/01/chto-takoe-ide-integrirovannaja-sreda-razrabotki/> (дата обращения: 27.03.2022).

14. Kaspersky: официальный сайт. – 1997. – URL: <https://www.kaspersky.ru/free-antivirus> (дата обращения: 01.04.2022).

15. Playground: Мнение: почему закрытие SteamGreenlight - хорошая новость: официальный сайт. – 2001. – URL: https://www.playground.ru/misc/news/mnenie_pochemu_zakrytie_steam_greenlight_horoshaya_novost-237240 (дата обращения: 21.03.2022).

16. SteamDirect: официальный сайт. – 2017. – URL: <https://partner.steamgames.com/steamdirect> (дата обращения: 21.03.2022).

17. Steam: официальный сайт. – 2003. – URL: <https://store.steampowered.com/?l=russian> (дата обращения: 21.03.2022).

18. SteamSpy: официальный сайт. – 2015. – URL: <https://steamspy.com> (дата обращения: 04.04.2022).

19. Ubisoft: официальный сайт. – 1986. – URL: <https://www.ubisoft.com/ru-ru/> (дата обращения: 21.03.2022).

20. UnityStore: Тарифы и цены: официальный сайт. – 2005. – URL: https://store.unity.com/ru?_gl=1*4uyp4y*_gcl_aw*R0NMLjE2NDE2MzAwNDMuQ2owS0NRaUFpZVdPQmhDWUFSSXNBtMNPdzB4bnM2dUF0bmIzODJCUmFwM2czRXpDN3JlbnNpSIZBWDJlVgwbDN1SkVTNFQ1NERQV183c2FBdE9CRUFMd193Y0I.#plans-individual (дата обращения: 01.04.2022).

21. UnrealEngine: официальный сайт. – 2004. – URL: <https://www.unrealengine.com/en-US> (дата обращения: 21.03.2022).
22. VisualStudio: официальный сайт. – 1997. – URL: <https://visualstudio.microsoft.com/ru/vs/> (дата обращения: 01.04.2022).
23. Академик: IDE: официальный сайт. – 2000. – URL: <https://dic.academic.ru/dic.nsf/ruwiki/40574> (дата обращения: 27.03.2022).
24. Ботоводство: Викисловарь: официальный сайт. – 1980. – URL: <https://ru.wiktionary.org/wiki/ботоводство> (дата обращения: 21.03.2022).
25. Гарант: официальный сайт. – 1990. – URL: [https://base.garant.ru/71787548/#:~:text=изменениями%20и%20дополнениями\),Приказ%20Министерства%20образования%20и%20науки%20РФ%20от%2019%20сентября%202017,%20\(с%20изменениями%20и%20дополнениями](https://base.garant.ru/71787548/#:~:text=изменениями%20и%20дополнениями),Приказ%20Министерства%20образования%20и%20науки%20РФ%20от%2019%20сентября%202017,%20(с%20изменениями%20и%20дополнениями) (дата обращения: 04.04.2022).
26. Консультант Плюс: Налоговый кодекс Российской Федерации: официальный сайт. – 1992. – URL: https://www.consultant.ru/document/cons_doc_LAW_19671/ (дата обращения: 03.04.2022).
27. Русские блоги: Часто используемые методы Unity API и примечания по классам: официальный сайт. – 1999. – URL: <https://russianblogs.com/article/17361441397/> (дата обращения: 28.03.2022).
28. Франшиза: Роялти: что это такое и какой размер приемлемый: официальный сайт. – 2001. – URL: <https://store.epicgames.com/ru/> (дата обращения: 28.03.2022).

Выпускная квалификационная работа выполнена мной самостоятельно.
Использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

Отпечатано в одном экземпляре.

Библиография 28 наименований.

Один экземпляр сдан на кафедру.

« ____ » _____ 2022 г.

_____ Танков Константин Вячеславович
подпись


Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Хакасский технический институт – филиал ФГАОУ ВО
«Сибирский федеральный университет»

Кафедра прикладной информатики, математики и естественно-научных
дисциплин

УТВЕРЖДАЮ

Заведующий кафедрой


Е. Н. Скуратенко

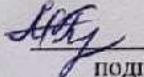
подпись

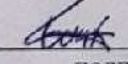
« 17 » 06 2022 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.03 Прикладная информатика

Разработка демо-версии компьютерной игры в жанре «Инди»

Руководитель  17.06.22 доцент, канд. физ.-мат. наук М. А. Буреева
подпись, дата

Выпускник  17.06.22 К. В. Танков
подпись, дата

Консультанты
по разделам:

Экономический  17.06.22 Е. Н. Скуратенко
подпись, дата

Нормоконтролер  17.06.22 В. И. Кокова
подпись, дата

Абакан 2022