Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Институт компьютерных и инженерных наук
Кафедра информационных и управляющих систем
Направление подготовки 09.03.01 – Информатика и вычислительная техника

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой
_____ А.В. Бушманов
«_____»_____2024 г.

БАКАЛАВРСКАЯ РАБОТА

на тему: Разработка игрового приложения «BeLive» средствами среды Unity

Исполнитель
студент группы 0103-об _____ (подпись, дата) Д.А. Радченко

Руководитель
доцент _____ (подпись, дата) Т.А. Галаган

Консультант:
по безопасности и экологичности
доцент, канд. техн. наук _____ (подпись, дата) А.Б. Булгаков

Нормоконтроль
инженер кафедры _____ (подпись, дата) В.Н. Адаменко

Благовещенск 2024

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Институт компьютерных и инженерных наук

Кафедра информационных и управляющих систем

УТВЕРЖДАЮ

Зав. кафедрой
_____ А.В. Бушманов

«_____»_____2023 г.

ЗАДАНИЕ

К выпускной квалификационной работе студента Радченко Д.А.

1. Тема выпускной квалификационной работы: Разработка игрового приложения «BeLive» средствами среды Unity.

2. Срок сдачи студентом законченной работы: _____

3. Исходные данные к бакалаврской работе: отчет о прохождении преддипломной практики, специальная литература, нормативные документы.

4. Содержание выпускной квалификационной работы: анализ предметной области, unity, c#, проектирование игрового приложения, спрайты, тайлы, uml-диаграммы, объектно-оринтированное программирование.

5. Перечень материалов приложения: техническое задание, UML-диаграмма классов

6. Консультанты по выпускной квалификационной работе: Консультант по безопасности и экологичности: доцент, канд. техн. наук А.Б. Булгаков

7. Дата выдачи задания: 02.10.2023 года.

Руководитель выпускной квалификационной работы: доцент Т.А. Галаган

Задание принял к исполнению: 02.10.2023 г.: _____
(подпись студента)

РЕФЕРАТ

Бакалаврская работа содержит 88 с., 38 рисунков, 2 приложение, 28 источников.

АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ, UNITY, C#, ПРОЕКТИРОВАНИЕ ИГРОВОГО ПРИЛОЖЕНИЯ, СПРАЙТЫ, ТАЙЛЫ, UML-ДИАГРАММЫ, ОБЪЕКТНО-ОРИНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ.

Цель работы: Разработка 2D игрового приложения на платформе Unity с использованием языка программирования C# и инструментов разработки Visual Studio.

Выполнение проекта включает два этапа: Первым этапом является исследование предметной области. На втором этапе выполняется проектирование основных игровых механик и компонентов, таких как система управления персонажем, взаимодействие с объектами и базовая анимация, разработка игрового приложения.

Игровые механики программируются на языке C# в среде разработки Visual Studio. Среди них - управление персонажем, обработка столкновений, взаимодействие с игровыми объектами, а также реализация игровых правил и логики. Особое внимание уделено разработке физики игры, которая обеспечивает реалистичность движений и взаимодействий.

Результатом выполнения работы является полнофункциональное 2D игровое приложение, разработанное на платформе Unity с использованием языка C#. Это приложение обеспечивает увлекательный геймплей и позволяет пользователю погрузиться в игровую среду, созданную на основе тщательно проработанных визуальных и механических компонентов.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Во второй половине XX века компьютеры и компьютерные игры начали своё стремительное развитие, оказав значительное влияние на общество и став неотъемлемой частью жизни. Начав своё путь с простых механических устройств и примитивных аркадных автоматов. Компьютеры и игры на их основе превратились в мощные интерактивные системы, которые сегодня используются для множества целей – от образования и работы до отдыха и развлечений.

Современные компьютерные игры представляют собой сложные и многоуровневые системы, которые предлагают пользователю не только развлечение, но и способствуют развитию когнитивных способностей, социального взаимодействия и творческого мышления. Жанр RPG, или ролевые игры, является одним из наиболее популярных в индустрии компьютерных игр благодаря своему глубокому погружению в виртуальные миры, обширным возможностям для кастомизации персонажей и захватывающим сюжетным линиям.

Цель данной дипломной работы заключается в создании компьютерной игры в жанре RPG, что представляет собой сложную и актуальную задачу в сфере разработки игр. Исследование направлено на анализ компьютерных игр как объекта изучения и выявление наиболее эффективных средств и технологий для их создания.

Основная цель исследования – разработка уникальной и захватывающей RPG-игры, использующей передовые технологии и методы разработки. Для достижения этой цели были поставлены следующие ключевые задачи:

Анализ жанра RPG – провести всесторонний анализ существующих игр в жанре RPG для выявления основных механик и характеристик, что позволит сформировать чёткие функциональные требования к разрабатываемой игре.

Изучение инструментов разработки – сравнить различные платформы

и инструменты для разработки игр, чтобы выбрать наиболее подходящие средства, обеспечивающие необходимый уровень гибкости и функциональности.

Разработка дизайна игрового мира – создать детализированные концепции существ, персонажей и окружения. Визуализировать эти элементы в графическом редакторе для обеспечения уникальности и привлекательности визуального стиля игры.

Создание прототипа игры – разработать рабочий прототип игры, включающий основные игровые элементы и интерфейс, что позволит демонстрировать основные концепции и механики игры.

Тестирование и отладка – провести тщательное тестирование прототипа для выявления и устранения технических и игровых недочётов, гарантируя высокое качество и стабильность конечного продукта.

Разработка RPG-игры представляет собой многослойную задачу, требующую глубоких знаний в области программирования, дизайна и психологии игр. Эта дипломная работа нацелена на углубление понимания этих процессов и вклад в развитие игровой индустрии, предлагая игрокам новые захватывающие приключения в виртуальных мирах. Результаты исследования помогут расширить границы жанра RPG и предложат пользователям уникальный игровой опыт.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛОСТИ

1.1 Анализ и общая характеристика предметной области

Игра является многогранным и многоаспектным феноменом, охватывающим различные составляющие человеческой и животной жизни. Она включает в себя элементы развлечения и отдыха, но её значимость выходит далеко за пределы этих функций. Рассмотрим предметную область игры и её ключевые аспекты.

Игра выполняет множество функций, каждая из которых имеет важное значение для комплексного развития индивида:

когнитивное развитие:

– игровая деятельность стимулирует мозговую активность;

– способствует формированию и укреплению нейронных связей;

– улучшает память, внимание, логическое мышление и способность к решению проблем.

физическое развитие:

– игры, требующие физической активности, улучшают координацию, баланс, силу и выносливость;

– развивают моторику и физическую подготовленность.

повторение и закрепление:

– игровые процессы обеспечивают повторение определённых действий;

– это приводит к автоматизации навыков и улучшению исполнения;

– способствует формированию долговременных паттернов поведения и рефлексов.

социальная интеграция:

– игры с элементами группового взаимодействия развивают коммуникативные навыки;

– способствуют умению работать в команде и социальной адаптации;

– помогают изучать социальные нормы и роли в обществе.

Игра применяется в самых разнообразных сферах человеческой деятельности. Рассмотрим основные из них:

образование и педагогика:

– игры служат педагогическим инструментом, облегчающим усвоение информации и развитие когнитивных навыков;

– в этнопедагогике игры используются для передачи культурных ценностей, обычаев и традиций, способствуя социализации и культурной адаптации.

психология и психотерапия:

– игровые методики применяются для диагностики, коррекции и поддержки психического здоровья;

– помогают преодолевать коммуникативные и поведенческие барьеры;

– используются в лечении психологических расстройств и улучшении эмоционального состояния.

бизнес и менеджмент:

– деловые игры и симуляции являются инструментами стратегического планирования и принятия решений;

– моделируют экономические и управленческие процессы, обучая анализу рынка, прогнозированию и оптимизации бизнес-процессов;

– способствуют развитию лидерских качеств, умению работать в команде и принимать решения в условиях неопределенности.

здравоохранение и реабилитация:

– игровые технологии и методы используются для восстановления двигательных функций, когнитивных способностей и эмоционального состояния;

– применяются в реабилитации после травм, операций или в процессе лечения хронических заболеваний.

Игра является мощным инструментом для всестороннего развития и социальной адаптации как у людей, так и у животных. Она выполняет критически важные функции в когнитивном, физическом и социальном развитии, обеспечивая индивиду возможности для личностного и коллективного роста. В разных областях деятельности игра находит своё уникальное применение, выходя за рамки простого развлечения и став значимым компонентом образовательных, лечебных, управленческих и культурных процессов.

Военные симуляции и стратегические игры являются инструментом для отработки военных доктрин, тактик и стратегий. Они создают условия, максимально приближенные к реальным боевым операциям, позволяя военнослужащим тренировать реакцию на различные сценарии без риска для жизни. Такие игры способствуют развитию стратегического мышления, оперативного планирования и координации действий в рамках военных подразделений.

Игра, как многомерный социально-культурный феномен, охватывает широкий спектр дисциплин и подходов в своем изучении. В рамках научного дискурса, терминология и классификация игр представляют собой следующие концепции:

Игрология – это междисциплинарное поле исследований, занимающееся анализом игровых процессов и их влиянием на индивидуальное и коллективное поведение. Игрология изучает структурные характеристики игр, их функциональное значение в социальных и культурных контекстах, а также роль игры в формировании идентичности и передаче культурных кодов.

Игрография – этот термин относится к систематическому описанию игр, включая их правила, механизмы, дизайн и визуальные аспекты. Игрография занимается каталогизацией игровых элементов и их классификацией, что позволяет ученым и разработчикам игр лучше понимать их структуру и динамику.

Психологическая и культурологическая антропология игры – д анная область исследований фокусируется на изучении игры как важного компонента культурного наследия и средства психологического развития. Она

рассматривает игры как инструмент социализации, через который передаются культурные ценности, обычаи и знания. В психологическом аспекте, антропология игры исследует влияние игровой деятельности на формирование личности, когнитивное развитие и эмоциональное благополучие.

Эти направления представляют собой лишь часть более широкой академической дискуссии о роли игры в человеческом обществе и культуре, подчеркивая ее значимость как инструмента обучения, развития и адаптации.

Таким образом, игра является неотъемлемой частью человеческой культуры и общества, выполняя ряд важных функций, от обучения и развития до социальной адаптации и стратегического планирования. Игры и игровая деятельность продолжают эволюционировать, адаптируясь к меняющимся условиям и потребностям общества.

Игра, может быть, и фактически является предметом изучения различных наук, например, биологии, физиологии и т. д. Понятие «игра» включает в себя огромный спектр представлений, и разные авторы по-своему подходят к трактовке этого определения. Но как бы различные авторы ни трактовали термин «игра», она всегда являлась одной из ведущих форм развития психических функций человека и способом реального познания мира. Компьютерная игра – игра, являющаяся компьютерной программой и использующая мультимедийные возможности компьютера.

2D игры представляют собой увлекательный жанр компьютерных игр, где все визуальные объекты существуют в двухмерном пространстве. Такие игры могут варьироваться от простых платформеров до сложных стратегий и RPG, каждая из которых отличается уникальным стилем и игровыми механиками.

Жанр является ключевым элементом любой игры и определяет её основные сюжетные и стилистические признаки. Например, в платформерах игроку предстоит управлять персонажем, который прыгает по платформам, избегает препятствий и сражается с врагами, в то время как в стратегических играх

основной акцент делается на планировании и управлении ресурсами.

Не менее важным компонентом является сеттинг, который определяет время и место, в которых разворачиваются события игры. Сеттинг может значительно влиять на атмосферу игры и её восприятие игроками. Например, действие может происходить в научно-фантастическом будущем, средневековом фэнтезийном королевстве или современном мегаполисе.

Игровая механика включает в себя правила и ограничения, определяющие, какие действия могут совершать игроки и как они взаимодействуют с игрой и друг с другом. Механики могут быть простыми и интуитивно понятными, либо сложными и глубокими, требующими стратегического мышления и мастерства.

Разработка компьютерной игры – это сложный и многогранный процесс, который объединяет множество дисциплин, включая искусство, программирование, аудиодизайн и интерактивный дизайн. Создание игры требует не только технических навыков, но и глубокого понимания повествования, психологии и эстетики. Разработчики постоянно балансируют между креативностью и техническими ограничениями, чтобы создать интересный и увлекательный игровой проект.

Таким образом, 2D игры, независимо от их жанра и сеттинга, продолжают развиваться и удивлять игроков по всему миру благодаря богатству игровых механик и креативным подходам к их разработке.

Рисунок 1 – Схема этапов разработки

Этапы разработки:

Ключевые роли в команде разработчиков:

– геймдизайнеры: отвечают за концепцию и дизайн игры, разработку правил и механик;

– художники и аниматоры: создают визуальный стиль игры, персонажей и анимацию;

– программисты: пишут код, который превращает идеи дизайнеров и художников в работающую игру;

– аудиодизайнеры: работают над звуковым оформлением, включая музыку, звуковые эффекты и озвучивание;

– тестировщики: ищут и документируют ошибки для их последующего исправления командой разработчиков;

– менеджеры проекта: координируют работу команды, следят за соблюдением сроков и бюджета.

Технологии и инструменты:

– игровые движки: Unity, Unreal Engine, Godot и другие, предоставляющие инструменты для создания и оптимизации игровых процессов;

– языки программирования: C++, C#, Python и другие, используемые для написания игрового кода;

– графические редакторы: Blender, Maya, Photoshop для создания визуального контента;

– системы управления версиями: Git, SVN для координации работы над проектом и отслеживания изменений.

Разработка компьютерной игры – это многогранный и сложный процесс, требующий слаженной работы целой команды профессионалов с различными навыками. От зарождения идеи до финального релиза, каждый этап разработки играет ключевую роль в создании высококачественного и увлекательного продукта, который способен удовлетворить ожидания игроков. Современные технологии и инструменты предоставляют разработчикам возможность воплощать самые смелые идеи, делая процесс создания игр одним из самых инновационных и динамично развивающихся направлений в сфере индустрии развлечений.

Особенно интересным феноменом в мире разработки игр являются инди-разработчики – это энтузиасты, которые работают самостоятельно или в небольших командах без финансовой поддержки крупных издательств. Такие игры зачастую более простые и менее бюджетные по сравнению с проектами из крупных студий, однако они могут предложить уникальные геймплейные механики и оригинальные идеи. Истинная сила инди-игр заключается в креативности и свободе, позволяющей экспериментировать с различными концепциями. Результатом этих усилий становятся оригинальные и увлекательные проекты, которые часто находят свою аудиторию.

Инди-игры обычно распространяются через интернет или различные сервисы цифровой дистрибуции и могут быть как бесплатными, так и продаваться за небольшую плату. Несмотря на ограниченные ресурсы и небольшую команду, некоторые инди-игры достигают огромного успеха. Так, например, игра Minecraft стала одной из самых продаваемых игр в мире и вошла в книгу рекордов Гиннеса как самая успешная инди-игра.

Таким образом, и крупные студии, и инди-разработчики вносят свой вклад в разнообразие и инновации в игровую индустрию. Благодаря совместным усилиям профессионалов и энтузиастов, игроки по всему миру могут наслаждаться множеством уникальных и увлекательных игровых произведений.

Рисунок 2 – Скриншот игры Minecraft

RPG – один из жанров компьютерных игр. Основой игрового процесса является отыгрывание определенной роли. Игрок управляет определенным героем, имеющим свои характеристики, умения и навыки, которые могут меняться в течение игры.

Тайловая графика представляет собой метод создания игровой графики из отдельных изображений, называемых тайлами, которые обычно имеют прямоугольную форму. Преимущества данного подхода включают в себя простоту использования: благодаря набору тайлов можно легко и быстро собирать сложные изображения и уровни, что значительно упрощает процесс

разработки.

Однако одной из главных проблем тайловой графики является её монотонность, обусловленная повторением одних и тех же элементов. Эта повторяемость может снижать визуальное разнообразие и делать игровой мир менее интересным. Для решения этой проблемы разработчики применяют несколько эффективных методов.

Первый метод заключается в создании нескольких версий одного тайла с незначительными изменениями. Это могут быть небольшие различия в текстуре, цвете или дополнительных деталях. Перемешивание таких вариаций помогает скрыть схожесть тайлов и создать иллюзию разнообразия, делая игровой мир более живым и динамичным.

Ещё один способ справиться с монотонностью – использование клеточной механики игры, где каждый тайл имеет определенное значение. Например, тайл с изображением цельных кирпичей может обозначать непроходимую стену, а тайл с потрескавшимися кирпичами – разрушаемую преграду. Таким образом, повторяемость становится частью геймплейной логики и воспринимается игроками как естественный элемент игрового мира.

Тайловая графика является мощным инструментом в арсенале разработчиков, позволяющим создавать разнообразные и богатые миры с минимальными затратами усилий. Благодаря различным техникам улучшения визуального разнообразия, этот метод продолжает оставаться актуальным и широко используемым в игровой индустрии.

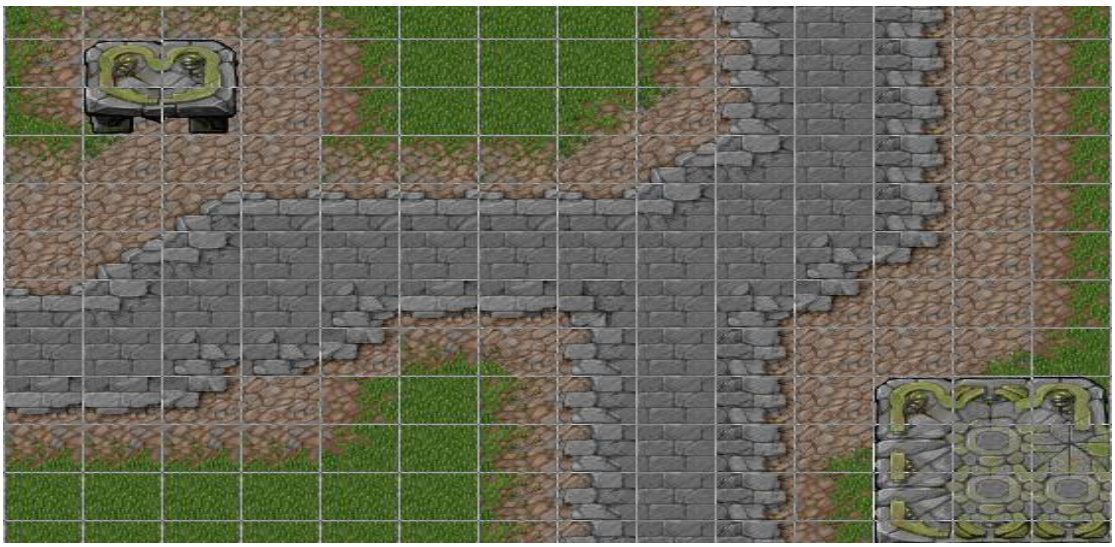

Рисунок 3 – Пример использования тайловой графики

ASCII – «Американский Стандартный Код для информационного обмена» (American Standard Code for Information Interchange). Был разработан в начале 60-х годов 20-го века как стандартная кодировка для компьютеров и аппаратных устройств. Данная кодировка использовалась как для передачи данных, так и для создания изображений (рисунок 3) из символов в так называемой ASCII-графике (или «псевдографике»), что помогало создавать псевдографические изображения для отображения их там, где невозможно отобразить обычную графику.

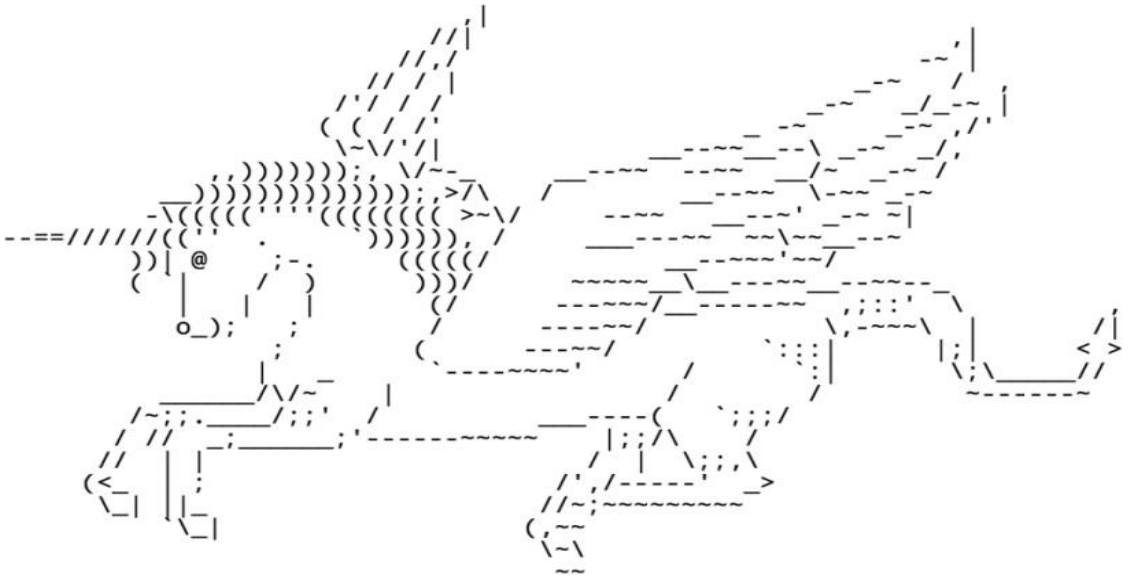

Рисунок 4 – Пример псевдографики

1.2 Анализ существующих разработок

Случайная генерация игрового окружения добавляет элемент неопределенности и уникальности к каждой игровой сессии. Системы генерации контента создают различные комбинации локаций, врагов и предметов, что делает каждое прохождение игры неповторимым. Это стимулирует игроков к повторным прохождениям и глубокому исследованию, так как каждый раз они сталкиваются с новыми вызовами.

Необратимость действий игрока в RPG имеет долгосрочные последствия. Ошибки могут быть фатальными, и часто нет возможности отменить их без перезапуска игры. Это заставляет игроков тщательно взвешивать свои решения и стратегии, придавая игре большую глубину и реализм.

Полная доступность игровых возможностей с самого начала, в отличие от многих других жанров, где функционал игры раскрывается постепенно, является еще одной характеристикой RPG. В таких играх игрокам часто предоставляется полный набор возможностей с начала игры, что позволяет им экспериментировать и разрабатывать собственные стратегии без искусственных ограничений.

Самостоятельное исследование мира способствует более глубокому погружению в игру и развитию навыков решения проблем. Игроки должны самостоятельно находить решения и пути для преодоления препятствий, что делает игровой опыт более насыщенным и увлекательным.

Полная свобода действий в RPG жанре часто отказывается от линейного прохождения в пользу открытого мира и нелинейной структуры. Игроки могут свободно выбирать, какие задания выполнять, когда и как взаимодействовать с игровым миром, что создает ощущение истинной свободы и влияния на мир игры.

Эти особенности делают RPG жанр одним из самых гибких и разнообразных в игровой индустрии, предлагая игрокам уникальный опыт и глубокое погружение в каждой игре.

Одним из примеров классического RPG является Pixel Dungeon – кроссплатформенная игра, выпущенная в 2013 году. Эта игра представляет собой классический рогалик, соответствующий всем канонам родоначальника жанра. В Pixel Dungeon 25 уровней, каждый пятый из которых заканчивается битвой с боссом. Конечной целью игры является поиск амулета Йендора, как в оригинальном Rogue.

Игрок перед началом игры может выбрать один из четырех классов персонажей – воин, маг, охотник или вор. Каждый класс отличается начальными параметрами и стартовой экипировкой, что позволяет разнообразить игровой опыт.

В игре всего три параметра героя – здоровье, сила (влияющая на эффективность экипировки) и голод. На каждом уровне герой должен найти переход на следующий уровень. При исследовании локаций герой может встретить врагов, которых в игре насчитывается 25 видов. После их смерти могут выпадать различные полезные предметы.

Среди предметов есть зелья и свитки, эффект которых можно узнать только после их использования, а также одежда (доспехи и кольца), оружие ближнего боя, метательное оружие, волшебные палочки и еда. Еда необходима для удовлетворения голода персонажа, иначе он не будет восстанавливать здоровье при перемещении, а будет получать урон. Один из свитков позволяет накладывать на вещи чары, дающие бонусы.

Pixel Dungeon является ярким примером того, как случайная генерация окружения, необратимость действий и разнообразие игровых возможностей могут создать увлекательную и глубоко проработанную игру, которая продолжает привлекать игроков долгие годы после своего выпуска.

Рисунок 5 – Скриншот игры Pixel Dungeon

При убийстве врагов персонаж получает опыт, повышающий уровень персонажа (что дает больший запас здоровья и очко силы).

Иногда на уровне встречаются магазины, где герой может потратить добытое золото на различные предметы.

Сама игра имеет простое и понятное управление, но при этом высокую игровую сложность. В игре имеется большое количество достижений.

DRL (ранее DoomRL) – это кроссплатформенная игра, выпущенная в 2013 году, выполненная в сеттинге вселенной игр Doom. Игра предлагает множество элементов, позаимствованных из оригинальных игр серии, включая врагов и вооружение. При запуске игроку предлагается выбрать один из нескольких уровней сложности, а затем – один из трёх классов персонажей, каждый из которых обладает уникальными характеристиками.

Основной игровой процесс включает в себя исследование 25 уровней, не считая секретных, с целью уничтожения врагов и выполнения поставленных задач. Главный герой – последний выживший из отряда космического спецназа, отправленного на завод на планете Фобос, спутнике Марса, для проверки сигнала бедствия. Действия игры разворачиваются именно здесь, соответствуя оригинальной вселенной Doom.

Управление в игре простое и интуитивно понятное, но сама игра отличается высокой сложностью, требуя от игрока стратегического мышления и

взвешенных решений. Убийство врагов приносит персонажу опыт, который повышает его уровень, увеличивая запас здоровья и давая очки силы. Иногда на уровнях попадаются магазины, где герой может потратить накопленное золото на различные полезные предметы.

DRL также включает в себя большое количество достижений, добавляя дополнительный уровень челленджа и разнообразия в игровой процесс. Игра предоставляет много возможностей для персонализации и адаптации под стиль каждого игрока, что делает её особенно привлекательной для любителей тактических игр и вселенной Doom.

Рисунок 6 – Скриншот из игры Doom

Отличием от оригинального Doom является то, что в DRL (рисунок 6) боевая система имеет большое тактическое значение, в то время как Doom является очень динамичным шутером.

Одна игровая сессия (полное прохождение) DRL не очень долгая, относительно многих игр данного жанра. Игра имеет простое и понятное управление.

Рисунок 7 – Скриншот игры DRL

Powder (рисунок 7) – идейный продолжатель NetHack. Разработчик игры хотел сделать игру, похожую на NetHack, но при этом для системы GameBoy Advance. Выбор платформы и ее портативность сыграли большую роль в истории развития и популярности игры.

Рисунок 8 – Скриншот игры Powder

Так как игра разрабатывалась для платформы GameBoy Advance (рисунок 8), можно отметить, что в игре крайне простое и понятное управление, поскольку консоль обладает малым количеством кнопок.

Рисунок 9 – Консоль GameBoy Advance

Powder – это захватывающая roguelike игра, в которой игроку предстоит пройти через 25 уровней, чтобы победить могущественного монстра по имени Baezl'bub и завладеть его сердцем. Персонаж имеет два основных параметра – жизнь и мана. За уничтожение монстров герой получает опыт, необходимый для повышения уровня. При достижении нового уровня игра предлагает выбрать одного из нескольких богов, которому можно последовать, каждый из которых дарует уникальные бонусы.

В игре представлено большое количество различных предметов, способностей и заклинаний, что делает каждую партию уникальной и разнообразной.

Elona, выпущенная в 2008 году, представляет собой roguelike JRPG, в которой основной акцент сделан на ролевую составляющую. В отличие от классических roguelike игр, смерть в Elona не является необратимой, она лишь снижает имеющиеся ресурсы и способности. Случайная генерация в игре осуществляется только в части локаций.

Elona предлагает обширное количество квестов и персонажей, что роднит её с традиционными RPG играми больше, чем с roguelike. Игрок управляет персонажем, который попадает в страну Северный Тайрис, периодически охваченной болезнью, заставляющей живых существ мутировать. Эти мутанты становятся одними из врагов, с которыми приходится сталкиваться главному герою.

Обе игры предоставляют увлекательный опыт для игроков, любящих

погружаться в уникальные миры с множеством возможностей для исследования и развития персонажа.

Рисунок 10 – Скриншот игры Elona

Для исследования в Северном Тайрисе персонажу доступны несколько городов и ряд других локаций. Есть возможность создать свои собственные локации. Дополнительно доступны как заранее созданные, так и процедурно генерируемые подземелья, на нижнем уровне которых есть боссы, дающие игроку награду за победу над ними.

Хоть в игре и есть основное задание, необходимое для прохождения игры, разработчик задумал игру именно не как сложную и запутанную, а простую и забавную, в которой игрок может спокойно исследовать мир в свое удовольствие.

Игроку доступно огромное множество разнообразных навыков и умений для освоения, среди которых есть как боевые и магические навыки, так и различные профессии, в том числе шитье одежды, рыболовство, садоводство, шахтерское дело и другие.

Поскольку игра представляет собой, прежде всего PRG, то игровая сессия может затянуться на многие часы.

Lost Labyrinth – данная игра (рисунок 10) была создана в 2001 году.

Данная игра имеет возможность игры вчетвером на одном компьютере.

Рисунок 11 – Скриншот игры Lost Labyrinth

Lost Labyrinth – это уникальная roguelike игра, отличающаяся от других представителей жанра своими механиками и необычным подходом к генерации уровней. При создании персонажа игроку предоставляется возможность выбрать несколько навыков из огромного списка, включающего как положительные, так и отрицательные навыки. Эти навыки напрямую влияют на последующую генерацию уровней и стратегию прохождения игры.

Герой имеет более десяти различных характеристик, среди которых, помимо стандартных здоровья и маны, также присутствуют восприятие, голод, жажда и нагрузка. Цель игры – найти девять фрагментов Посоха, пройдя через множество различных локаций.

Во время исследования мира Lost Labyrinth игрок может встретить торговцев, у которых можно приобрести снаряжение или еду, что помогает продвинуться дальше по уровню. Игра отличается отсутствием получения опыта за убийство врагов; вместо этого, игрок получает опыт только при переходе на следующий уровень.

Игра также выгодно выделяется своей скоростью: средняя игровой сессия занимает всего 10-40 минут, что делает её идеальным выбором для тех, кто хочет получить быстрый и увлекательный игровой опыт.

Lost Labyrinth поражает своим разнообразием навыков и характеристик,

наряду с уникальной механикой генерации уровней, что делает каждое прохождение по-настоящему уникальным и захватывающим.

1.3 Анализ средств разработки и обоснование выбора технологии проектирования для всех элементов проекта

Unity 2018 – мощный и гибкий кроссплатформенный игровой движок, разработанный компанией Unity Technologies, который предоставляет разработчикам обширные возможности для создания разнообразных интерактивных приложений. Этот движок подходит для разработки видеоигр, тренировочных симуляций и проектов виртуальной реальности.

Одной из ключевых особенностей Unity 2018 является его универсальность. Двигатель поддерживает работу как с 2D, так и с 3D графикой, что позволяет создавать визуально насыщенные и технологически сложные проекты. Дополнительно предусмотрены инструменты для анимации, которые упрощают процесс создания движущихся объектов и персонажей.

Unity 2018 также включает в себя мощные средства для работы со звуком. Это позволяет разработчикам добавлять в проекты разнообразные звуковые эффекты и музыкальные сопровождения, что значительно увеличивает уровень погружения пользователя в игру или симуляцию.

Ещё одним важным аспектом движка является физический движок, который симулирует реальные физические явления. С помощью этого инструмента разработчики могут внедрять в свои проекты реалистичное взаимодействие объектов, что делает работу с физикой интуитивной и простой.

Благодаря своей кроссплатформенности, Unity 2018 позволяет создавать приложения для множества устройств и операционных систем, включая ПК, консоли, мобильные устройства и системы виртуальной реальности. Это делает его идеальным выбором для команд, стремящихся разработать продукт, доступный для широкого круга пользователей.

Таким образом, Unity 2018 представляет собой мощное решение для разработки интерактивных приложений, объединяя в себе инструменты для

работы с графикой, анимацией, звуком и физикой. Его гибкость и кроссплатформенные возможности открывают новые горизонты для разработчиков, способствуя созданию высококачественных и инновационных продуктов.

Кроссплатформенность:

Одной из ключевых особенностей Unity является его кроссплатформенность. Unity поддерживает огромное количество платформ:

– десктопные ОС: Windows, macOS, Linux;

– мобильные ОС: Android, iOS, Fire OS;

– консоли: PlayStation 4, PlayStation Vita, Xbox One, Nintendo 3DS, Nintendo Switch;

– виртуальная реальность: Oculus Rift, Steam VR, Gear VR, PlayStation VR;

– телевизионные ОС: Android TV, Samsung SMART TV, tvOS.

Это означает, что разработчики могут создавать игры и приложения, которые будут работать на всех этих платформах без необходимости переписывания кода для каждой из них.

Системные требования:

Unity 2018 имеет относительно низкие системные требования, что делает его доступным для широкого круга разработчиков:

– операционная система: Windows 7/8/10 (64-бит) или Mac OS X 10.9+;

– процессор: с поддержкой набора инструкций SSE2;

– видеокарта: с поддержкой DirectX 10.

Низкие системные требования позволяют использовать Unity на большинстве современных компьютеров, что снижает барьеры для входа в индустрию разработки игр.

Инструменты и возможности:

Unity предлагает разнообразные инструменты для разработки:

– редактор Unity – интуитивно понятный интерфейс и возможность визуального программирования (через систему Unity Visual Scripting);

– Asset Store – магазин активов, где разработчики могут покупать и продавать готовые ресурсы, такие как модели, текстуры, скрипты и инструменты;

– системы физики и анимации – реалистичное моделирование физики и сложные анимационные системы для создания живых и убедительных игровых миров;

– поддержка многопользовательских игр – встроенные инструменты для создания многопользовательских игр и поддержки сетевого взаимодействия.

Unity 2018 остается одним из самых популярных выборов среди разработчиков игр благодаря своей универсальности, доступности и мощному набору функций. Он позволяет создавать игры и приложения высокого качества, которые могут достигать широкой аудитории благодаря поддержке множества платформ. В то же время, низкие системные требования делают его доступным для начинающих разработчиков и небольших студий, что способствует демократизации процесса создания игр.

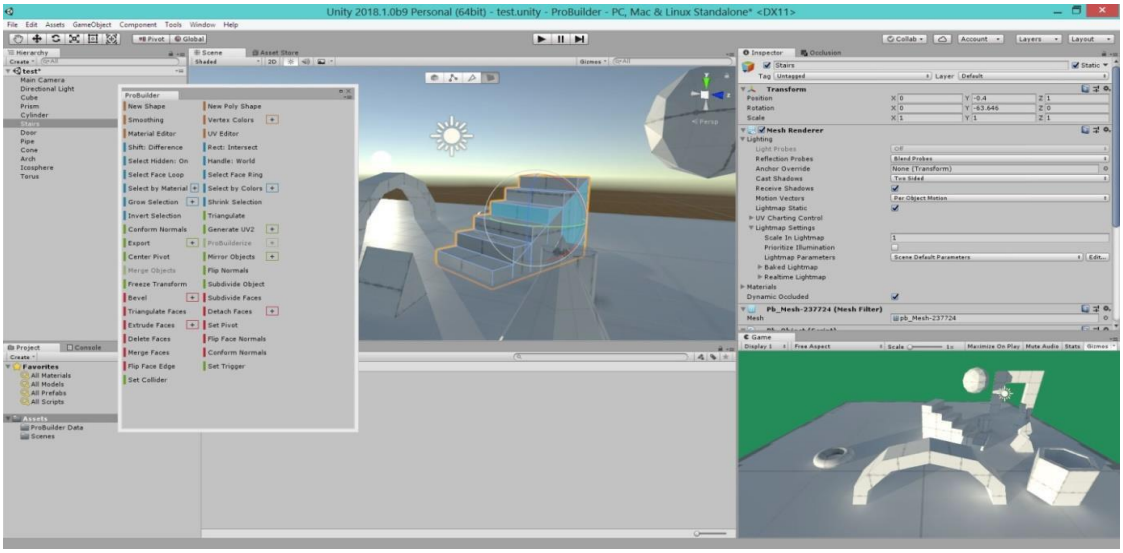

Рисунок 12 – Интерфейс Unity 2018

Движок Unity (рисунок 12) обладает удобным и понятным интерфейсом и имеет богатый функционал.

Имеет инструменты, как для художников-разработчиков (например, Timeline для разработки анимационных сцен, Cinemachine – набор «умных» и динамических камер, Progressive Lightmapper для работы с освещением и

отличная поддержка программы Autodesk Maya для работы с 3D-анимацией и моделированием), так и для программистов-разработчиков. Движок позволяет вести разработку как 2D, так и 3D проектов.

Имеется поддержка движков Box2D (движок для работы с двумерными объектами) и NVIDIA PhysX (движок для работы с физикой объектов).

Данный движок имеет отличную оптимизацию, что сказывается на скорости и качестве проекта.

В Unity поддерживает такие языки программирования как C# и UnityScript (основанный на JavaScript).

Unity имеет свой собственный магазин Asset Store, в котором представлен огромный каталог материалов для разработки, как платных, так и бесплатных: можно найти все необходимое – рисунки, модели, скрипты, различные дополнения и расширения и прочее.

Имеется возможность разрабатывать сетевые игры с помощью сервиса Unity Multiplayer. Данные игры используют серверы Unity Matchmaker, что сильно упрощает задачу соединения пользователей.

Unity поддерживает одновременную работу над одним проектом несколькими разработчиками.

На официальном сайте движка можно найти огромное количество документации (рисунок 13), в том числе и обучающей, что, несомненно, является неплохим подспорьем начинающего разработчика.

Обучающая информация представлена как отдельными уроками, подробно разбирающими какие-либо темы, так и проектами по созданию небольших игр.

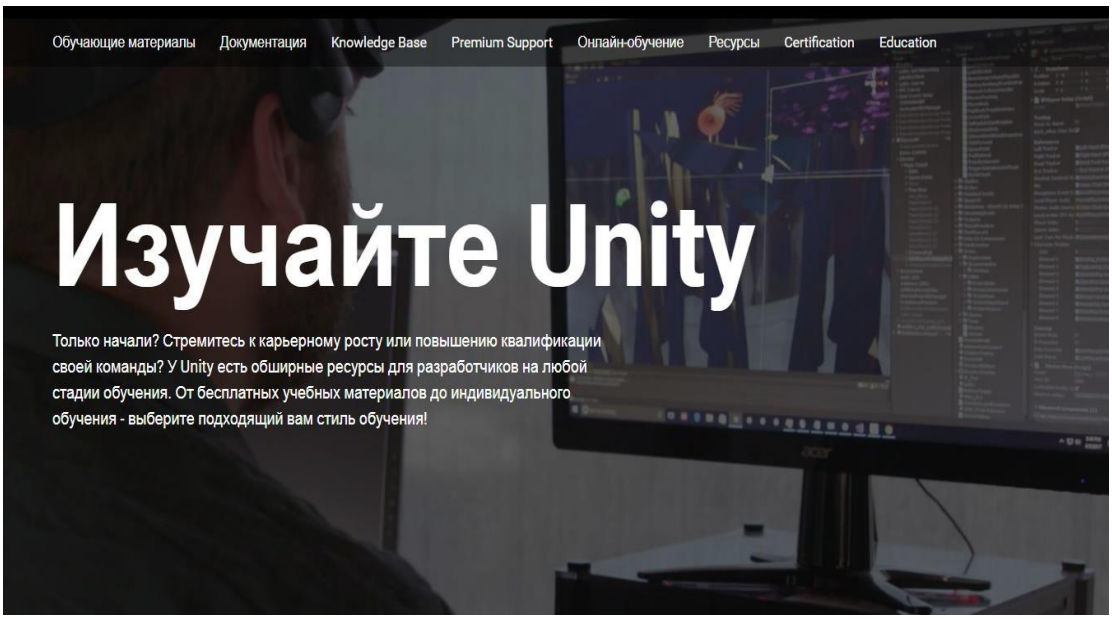

Рисунок 13 – Раздел официального сайта движка с документацией

Поскольку это один из самых распространенных игровых движков, в интернете представлено большое количество неофициальной обучающей информации.

Данный движок имеет 3 версии – Personal (бесплатно), Plus (35$ в месяц), Pro (125$ в месяц). Данные версии отличаются лишь предлагаемым

функционалом и максимально допустимым годовым доходом (100 тыс. долларов, 200 тыс. долларов и без ограничений соответственно).

Движок является очень универсальным и позволяет использовать его для разработки самых разнообразных продуктов.

CryEngine V – самая последняя версия игрового движка от компании Crytek. Данный движок может похвастаться отличной графикой. Первая версия движка CryEngine использовалась компанией при разработке первой части игры Far Cry.

Рисунок 14 – Скриншот игры Far Cry

CryEngine V хоть и является кроссплатформенным движком, поддерживает всего лишь 4 платформы – Windows, Playstation 4, Xbox One, Oculus Rift.

Минимальные системные требования не очень высокие, хоть и выше, чем у Unity: Windows 7/8/10 64x, процессор Intel Dual-Core от 2GHz (Core 2 Duo и выше)/AMD Dual-Core от 2GHz (Phenom II X2 и выше), оперативная память 4 GB, видеокарта NVIDIA GeForce 450 и выше/AMD Radeon HD 5750 и выше, поддержка DirectX 11.

Движок имеет простой и понятный интерфейс (рисунок 16), во многом схожий с интерфейсом Unity.

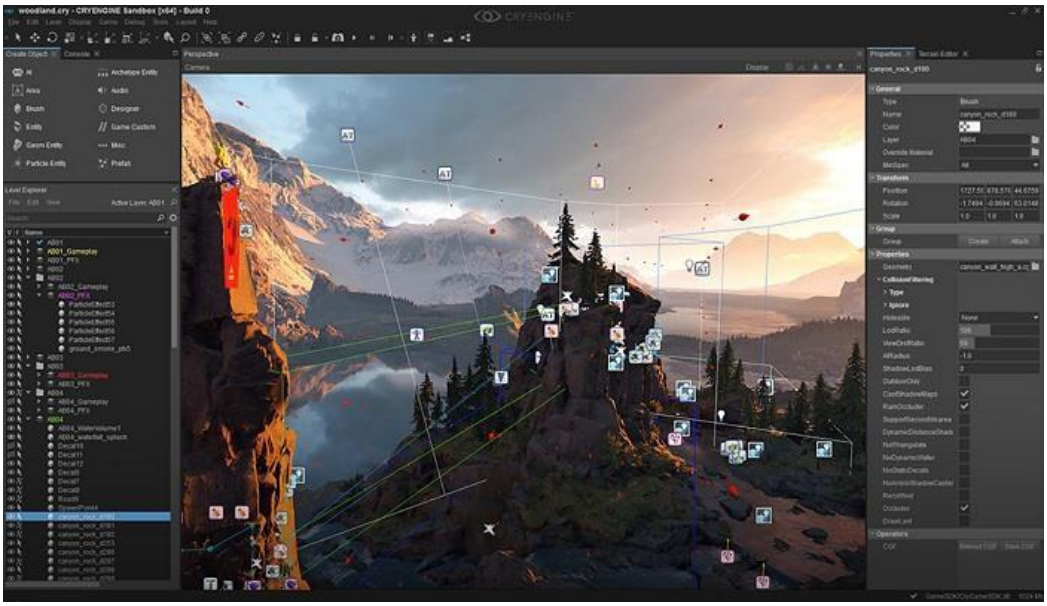

Рисунок 15 – Интерфейс CryEngine V

Движок CryEngine делает большой упор на визуальную составляющую

проекта, поэтому имеет большое количество функций и инструментов для достижения превосходного качества изображения. Имеется поддержка DirectX 12.

Physically Based Rendering (PBR) представляет собой метод визуализации, который стремится к точному моделированию взаимодействия света с материальными поверхностями, опираясь на законы физики. Этот подход позволяет создавать изображения с высокой степенью реалистичности, учитывая такие факторы, как освещение, материалы и их свойства. В контексте компьютерной графики, PBR обеспечивает следующие преимущества:

– реалистичное взаимодействие света и материалов – используя физически корректные алгоритмы, PBR имитирует способы, которыми свет отражается, преломляется и поглощается различными материалами, что приводит к более правдоподобному отображению объектов;

– динамическая Обработка Водных Каустик – реализация водных каустик в реальном времени позволяет добиться эффекта реалистичного отображения световых паттернов, создаваемых водной средой, что значительно улучшает визуальное восприятие водных поверхностей;

– эффективное сглаживание – сглаживание изображения снижает визуальную пикселизацию, обеспечивая более гладкие и четкие края объектов, что улучшает общее визуальное качество сцены;

– теневые карты с расширенными настройками – использование теневых карт с широким спектром параметров настройки позволяет добиться точного и гибкого контроля над отображением теней, что способствует созданию более глубокой и объемной визуализации сцены.

– HDR-кривая для настройки освещения – применение HDR-кривой обеспечивает расширенный динамический диапазон освещения, позволяя детализировать темные и светлые участки изображения, тем самым улучшая визуальное различение тонов.

Таким образом, система Physically Based Rendering является ключевым

элементом современной компьютерной графики, обеспечивающим создание визуально богатых и убедительных цифровых сцен.

В движке присутствует встроенный редактор материалов. Система CryEngine Sandbox предлагает большое количество инструментов для быстрого создания игровых уровней и миров. Редактор TrackView представляет собой встроенный инструмент редактирования видеозаписей для создания интерактивных сцен с зависящим от времени управлением объектами и событиями. Инструмент Designer Tool является редактором моделей и позволяет экспортировать созданные объекты во внешние инструменты

Хотя CryEngine и поддерживает языки программирования C++ и Lua, в нем доступна система визуальных скриптов Flowgraph, которая позволяет создавать и контролировать игровую логику и события без необходимости писать скрипты вручную.

Движок имеет большое количество настроек анимации персонажей, в том числе параметрическую скелетную анимацию. Имеется встроенная расширенная система искусственного интеллекта, позволяющая настроить реалистичное поведение неигровых персонажей.

CryEngine представляет инструментарий для анализа производительности игры – можно отслеживать проблемы производительности прямо во время игры, а инструмент Statoscope (рисунок 16) позволяет получать всю информацию по расходу ресурсов компьютера в графическом виде.

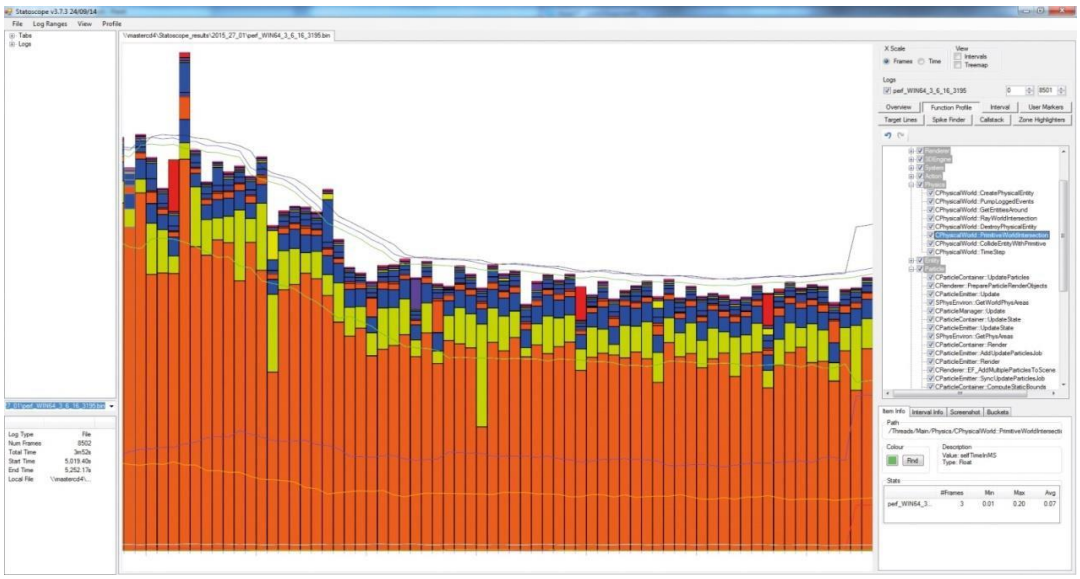

Рисунок 16 – Инструмент Statoscope

Проекты, созданные на CryEngine, имеют открытый исходный код.

На официальном сайте имеется обучающая информация и магазин с различными материалами для разработки – модели, скрипты, анимации, звуки и прочее (как платные, так и бесплатные).

Движок распространяется со свободной лицензией, однако, если продукт, созданный на движке, приносит прибыль более 5 тысяч долларов в год, то 5 % прибыли отчисляется Crytek.

Данный движок является отличным инструментом для разработки 3D проектов.

Unreal Engine 4 – игровой движок, разработанный компанией Epic Games. Первая версия Unreal Engine была разработана для игры этой компании под названием Unreal (рисунок 17).

Рисунок 17 – Скриншот игры Unreal

Движок Unreal Engine поддерживает следующие платформы – Windows, Mac OS, Android, iOS, Nintendo Switch, Linux, PS4, Xbox One, Oculus Rift, PlayStation VR, Samsung Gear VR, Viveport, Daydream, HTML5.

Данный движок обладает следующими рекомендуемыми системными требованиями: Windows 7/8/10 64x или Mac OS 10.13 или Linux Ubuntu 15.04, процессор Intel Quad-core или AMD (2.5 GHz и выше), оперативная па- мять 8 GB (16 GB для Linux), видеокарта с поддержкой DirectX 11 (для Windows) или Metal 1.2 Compatible Graphics Card (для Mac OS) или NVIDIA GeForce 470 GTX и выше (для Linux). Минимальные требования ограничены лишь версией операционной системы.

Движок имеет удобный, настраиваемый интерфейс (рисунок 18), сильно отличающийся от интерфейсов предыдущих движков.

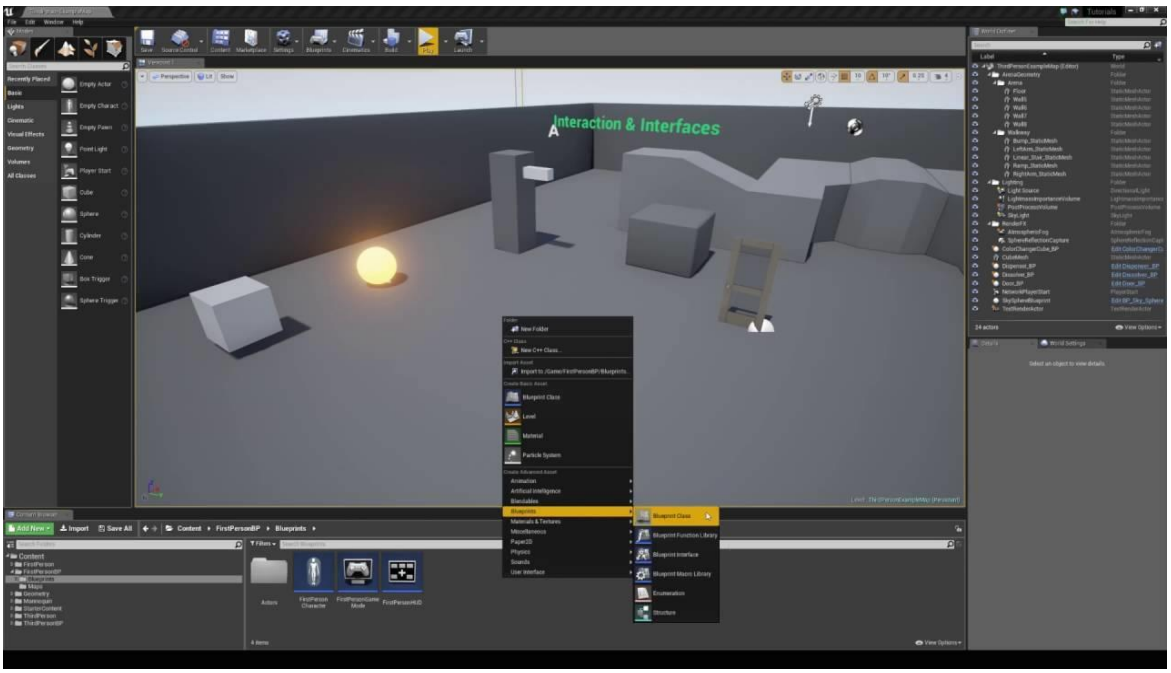

Рисунок 18 – Интерфейс Unreal Engine 4

Движок позволяет добиться фотореалистичной графики с помощью большого количества настроек рендера, динамических теней, отражений каналов освещения.

Инструментарий движка позволяет существенно упростить работу с ландшафтом и растительностью.

Имеется полный доступ к исходному коду на языке C++.

Благодаря системе визуальных скриптов Blueprint есть возможность управлять игровой логикой без написания скриптов вручную, хоть такая возможность и имеется (на том же языке C++).

Встроенный редактор визуальных эффектов Cascade позволяет настраивать системы частиц с использованием самых разных модулей. В движок встроен редактор материалов, использующий искусственное затенение и дающий беспрецедентный контроль над внешним обликом персонажей и объектов.

Используется обширный набор анимационных инструментов, позволяющих редактировать сетку и анимацию персонажей. Кроме того, получившийся результат можно сразу же посмотреть и оценить.

Для создания анимированных сцен и видеороликов используется

инструмент Sequencer, позволяющий настраивать освещение, камеру и персонажей.

Unreal Engine имеет широкую поддержку разработки мультиплеерных игр. Данный движок поставляется с масштабируемой и проверенной архитектурой клиент/сервер.

Unreal Engine позволяет работать в режиме виртуальной реальности, с расширенными элементами управления движением. Благодаря тесному сотрудничеству компании Epic Games с мировыми лидерами в области аппаратного и программного обеспечения, Unreal Engine обеспечивает высокое качество взаимодействия с системами виртуальной и дополненной реальности.

Расширенная система искусственного интеллекта позволяет настраивать реалистичное поведение неигровых персонажей.

На официальном сайте имеется официальная документация и обучающая информация. Имеется магазин с различными материалами для разработки – модели, скрипты, анимации, звуки, плагины и прочее (как платные, так и бесплатные).

Данный движок наиболее подходит для разработки 3D игр, нежели 2D, однако имеет широкий выбор платформ, как для разработки, так и для готовых проектов.

Так как для реализации данные проекта не требуется подключение каких-либо дополнительных инструментов, главными критериями выбора среды разработки являются следующие: бесплатная лицензия, наличие обучающей документации, низкие системные требования, удобный инструментарий для разработки 2D проектов. Данным критериям более всего соответствует движок Unity.

Для реализации программного кода в движке Unity используется Microsoft Visual Studio 2017 (рисунок 19), который имеет возможность интеграции в Unity.

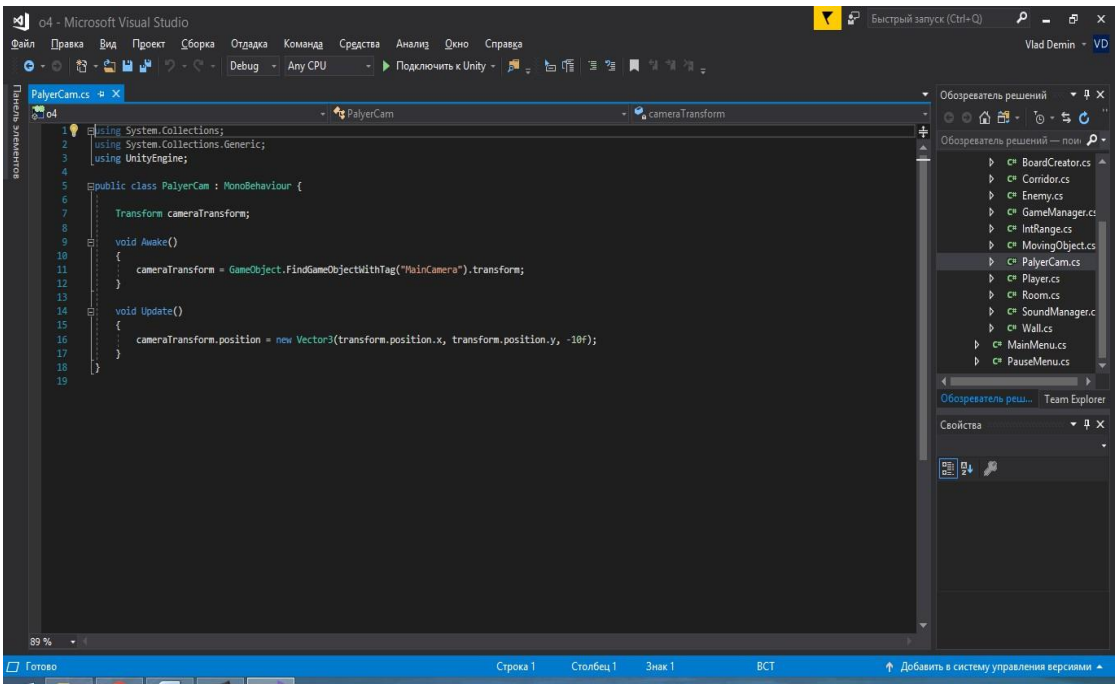

Рисунок 19 – Интерфейс Microsoft Visual Studio 2017

Visual Studio 2017 – программный продукт компании Microsoft, включающий в себя интегрированную среду разработки программного обеспечения. Включает в себя редактор исходного кода и имеет возможность рефакторинга кода. Обладает редактором форм для упрощения создания графического интерфейса приложений. Имеется возможность расширенной настройки продукта под свои нужны, доступно подключение различных сторонних дополнений.

Данный продукт является удобным инструментом программирования и отлично подходит для написания кода.

Одним из важнейших этапов разработки компьютерной игры является работа с визуальной частью. В этом могут помочь графические редакторы. Есть большое количество различных графических редакторов, обладающих различным функционалом и предназначенных для различных задач, но среди них имеется один, который одновременно обладает внушительным инструментарием, но при этом является легким в изучении.

Adobe Photoshop CC 2017 – растровый графический редактор, распространяемый компанией Adobe Systems. Данный редактор поддерживает как растровые, так и векторные форматы изображений и поддерживает все самые

распространенные цветовые модели (RGB, SMYK, LAB и другие). Имеется поддержка графических планшетов и расширенный функционал работы как с графикой, так и со скриптами и анимацией. Adobe Photoshop распространяется по платной лицензии (от 1288 рублей в месяц).

Поскольку данный графический редактор еще и изучался во время обучения в университете, именно он был выбран для работы с графической частью игры.

1.4 Общий алгоритм реализации проекта

Этап 1 – разработка концепта игры. На данном этапе выбирается жанр игры, ее сеттинг, ее вида (2D/3D), вида камеры (от первого лица/от третьего лица), пишется сценарий. На этом же этапе, выбирается среда разработки игры, и платформа и определяется целевая аудитория.

Этап 2 – написание скриптов. Сюда входят скрипты процедурной генерации и обработчика уровня, скрипт передвижения (персонажа и врагов), скрипт персонажа и врагов (взаимодействие героя с бонусами и врагами).

Этап 3 – отрисовка графики. Рисуются тайлы для генерации уровня и спрайты бонусов, спрайты героя и врагов, с последующей их анимацией.

Этап 4 – тестирование продукта. Здесь происходит проверка работоспособности продукта и отдельных его механик.

Этап 5 – запаковка проекта и выпуск игры.

2 ПРОЕКТНАЯ ЧАСТЬ

2.1 Функционал проекта

В данном проекте 2D-игры реализован следующий функционал:

Начало игры – пользователь может начать новую игру или продолжить существующую, выбрав соответствующую опцию в меню.

Управление персонажем – пользователь может управлять движением персонажа влево или вправо.

Атака – пользователь может атаковать врагов или другие объекты в игре.

Сбор предметов – пользователь может собирать различные предметы, разбросанные по игровому миру.

Взаимодействие с врагами – персонаж пользователя может взаимодействовать с врагами, что может включать атаки, избегание и другие действия.

Управление атрибутами персонажа – пользователь может просматривать текущие атрибуты своего персонажа, такие как здоровье, уровень, сила и другие характеристики.

Пользователь может изменять атрибуты персонажа, возможно, повышая уровень или улучшая характеристики.

Управление инвентарем – пользователь может просматривать текущий инвентарь, видя все собранные предметы, может использовать предметы из инвентаря, что может влиять на атрибуты персонажа или предоставлять другие бонус, так же можно выбрасывать ненужные предметы из инвентаря, освобождая место для новых.

Этот функционал обеспечивает полноту игрового процесса и позволяет пользователям погружаться в игровой мир, управлять своим персонажем и взаимодействовать с окружением.

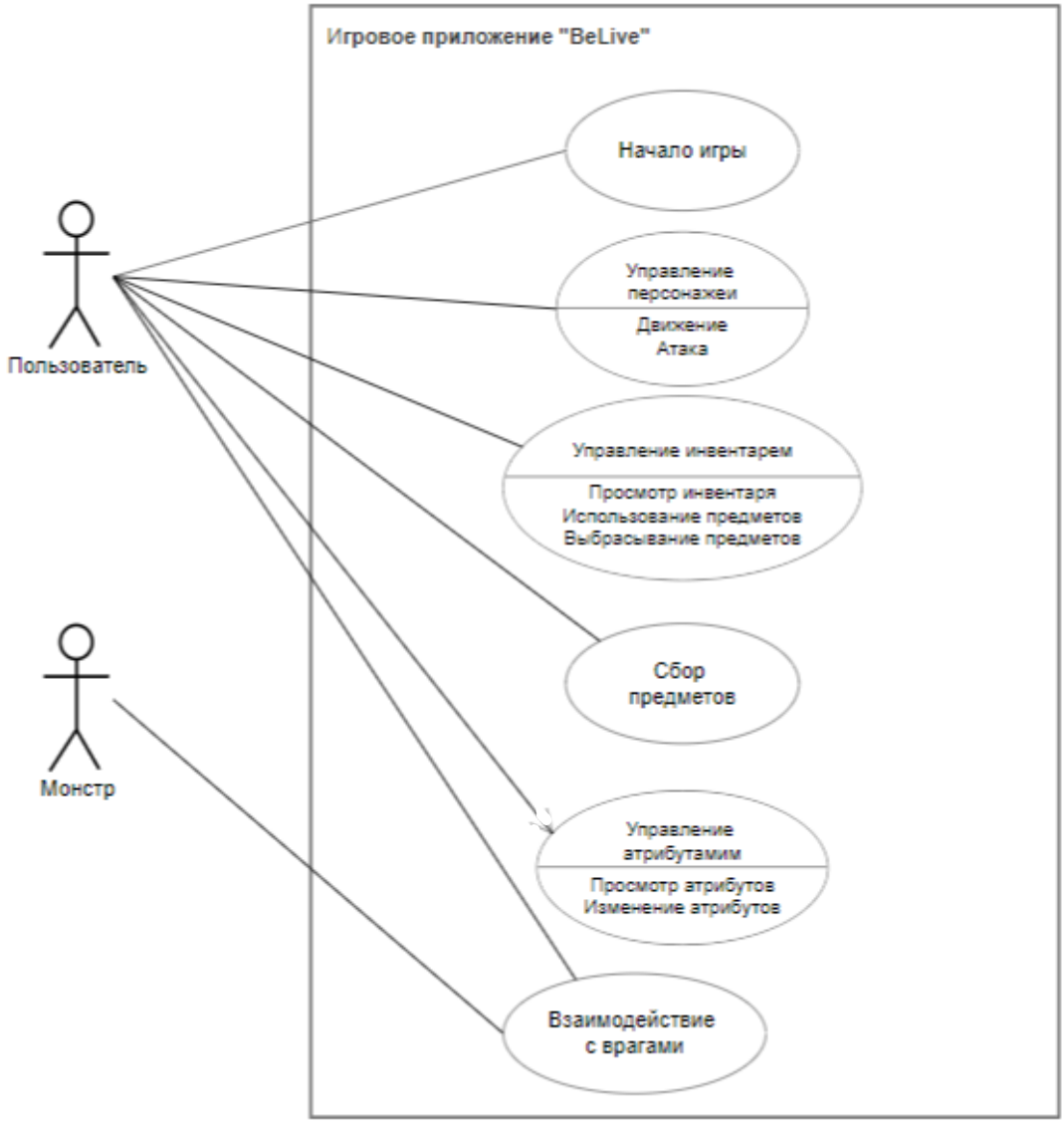

Рисунок 20 – Диаграмма прецедентов

2.2 Характеристика потенциальной аудитории потребителей проекта, ориентированность работы

Характеристика целевой аудитории является одним из значительных этапов разработки любого проекта, так как это поможет разработчику лучше понять целевую аудиторию и, следовательно, более внимательно отнестись к тем или иным деталям разрабатываемого продукта. Поэтому для проекта была описана его целевая аудитория.

Типичный представитель целевой аудитории разрабатываемой выглядит так: подросток 11–19 лет, преимущественно мужского пола (поскольку жанр ролевых игр более распространен у лиц мужского пола) доход низкий

или отсутствует. Место жительство любое – от небольших населенных пунктов до крупных городов. Род занятий – учеба или работа. Свободное время предпочитает проводить, играя в игры или сидя в интернете. Готовы узнавать/пробовать новое.

Поскольку жанр RPG является довольно специфичным и не самым распространенным, его целевая аудитория не так обширна, как у более популярных жанров. Также этому способствует довольно высокая сложность игр данного жанра, игр в жанре RPG с низкой сложностью крайне мало. Именно поэтому целевой аудиторией разрабатываемой игры были выбраны людей, которые хотят познакомиться с данным жанром. Разрабатываемая игра не обладает такой высокой сложностью, как большинство представителей данного жанра, но при этом позволяет понять основные механики, присущие жанру RPG.

Поскольку игровые сессии крайне быстрые, то данная игра может приглянуться людям с небольшим количеством свободного времени. Игра обладает пиксельной графикой, что может привлечь любителей проектов такого вида.

В общем, данный проект может заинтересовать самую разнообразную аудиторию.

2.3 Методы разработки дизайна окружения и персонажей

Перед полноценным началом разработки, разрабатывается непосредственно сам дизайн видеоигры.

2.3.1 Сайт для разработчиков игровых приложений «itch.io»

"Itch.io" удобная платформа для независимых разработчиков игр. Будучи одним из лидеров в индустрии инди-игр, "itch.io" позволяет разработчикам по всему миру напрямую продвигать и продавать свои игры, избегая необходимости работать с посредниками. Одной из ключевых функций "itch.io" является возможность предварительного тестирования игр: игроки могут скачать демоверсии, чтобы оценить игру до покупки полной версии, что помогает им сделать осознанный выбор. Сайт также предлагает доступ к бесплатным

ассетам от профессиональных дизайнеров, которые я использовал при создании своей игры. Широкий спектр игр, доступных на платформе, позволил мне разработать концепцию графики для моего проекта и изучить современные игровые механики, особенно в жанре платформеров.

"Itch.io" не просто способствует коммерческому успеху, но и служит важным местом для обмена опытом и идеями между разработчиками, а также платформой для получения отзывов от игроков, что является критически важным для улучшения и доработки игр. Платформа предоставляет гибкие инструменты для настройки страницы игры, позволяя разработчикам точно передать свое видение и привлекать заинтересованную аудиторию.

2.3.2 Приложение для работы с пиксельной графикой

После анализа разнообразных инструментов для создания пиксельной графики, выбор пал на Aseprite в качестве основного средства для дизайна персонажей и мира игры. Программное обеспечение выделяется эффективностью и обширным набором функций, позволяющих реализовать уникальные и яркие визуальные образы.

Создание цветовой палитры стало первым шагом, задающим тон всей игре.

Акцент на яркости и насыщенности цветов призван привлечь внимание игроков и задать уникальный стиль проекту. Затем началась разработка главного героя игры.

С использованием Aseprite был создан первый кадр персонажа, начиная с базовых пиксельных квадратов и постепенно объединяя их в единую форму. Добавление контуров и деталей придало персонажу реалистичность и выразительность.

Для анимации персонажа использовался функционал анимации в Aseprite, создавая серию кадров, которые оживляют героя, позволяя ему двигаться и взаимодействовать в игровом мире. Добавление нескольких кадров с разными позами создало иллюзию движения.

После завершения работы над персонажем началась разработка окружения. С помощью Aseprite были разработаны разнообразные элементы ландшафта, включая деревья, камни и траву, используя те же методы, что и для персонажа, чтобы достичь детализированного и живописного вида.

В итоге, благодаря Aseprite, создан уникальный и живописный игровой мир, который не только привлекает внимание игроков, но и придает игре особенный, запоминающийся стиль. Этот мир не является просто фоном для действия; он становится частью повествования, добавляя глубину и контекст к приключениям героя. Такой подход к дизайну способствует созданию более погруженного и цельного игрового опыта.

2.4 Жизненный цикл проекта, описание поэтапной реализации проекта с указанием средств реализации

2.4.1 Этап эскизного проектирования и разработки элементов дизайна

Рисунок 21 – Схема этапов жизненного цикла

Первым этапом в разработке игры стало эскизное проектирование. На этом этапе необходимо определить сеттинг, то есть выбрать сюжетную тему или мир, в котором будет разворачиваться действие игры. В качестве сеттинга было выбрано приключение в средневековом фэнтезийном мире, что является популярным выбором для большинства классических RPG игр.

Многие из известных классических RPG игр использовали либо

псевдографический стиль, либо простую и примитивную графику. В связи с этим было принято решение использовать в нашей игре пиксельную графику. Такой выбор обусловлен несколькими факторами. Во-первых, пиксельная графика пользуется значительной популярностью среди современных игроков. Во-вторых, она визуально напоминает графику игр жанра roguelike, что добавляет ей особого шарма. Решено было использовать разрешение 16x16 пикселей, благодаря чему все тайлы и объекты будут иметь размеры 16 на 16 пикселей, обеспечивая стильный и ретро-атмосферу.

Для главного героя было решено выбрать рыцаря. Это классический персонаж, который прекрасно вписывается в средневековый фэнтезийный мир. Для создания анимации получения урона было важно, чтобы персонаж выглядел естественно и реалистично. По этой причине был выбран образ рыцаря без шлема, чтобы лучше передать эмоции и реакции на боевые действия. В ходе работы было создано несколько набросков рыцаря, а также различные варианты его лица и оружия, чтобы выбрать наиболее подходящий внешний вид.

Таким образом, на этапе эскизного проектирования были определены ключевые аспекты будущей игры: выбраны сеттинг, графический стиль и главный герой. Эти решения заложили основу для дальнейшего развития проекта и позволили сформировать уникальную концепцию игры, которая сочетает элементы классических RPG с современными визуальными тенденциями.

Рисунок 22 – Лист анимации персонажа

Фэнтезийный сеттинг позволяет использовать различных мифических существ, поэтому в качестве врагов были выбраны следующие существа – зомби, привидение и гигантская змея. Для них были сделаны наброски, по которым были созданы спрайты врагов.

Для данных спрайтов были нарисованы кадры анимации простоя и атаки.

Рисунок 23 – Спрайты змеи, зомби, привидения

Были отрисованы тайлы окружения (стены и пол) трех видов (рисунок 28) для создания разного внешнего вида уровней: каменное подземелье, песчаное подземелье, ледяное подземелье, и тайл перехода на следующий

уровень.

Данные тайлы являются бесшовными, это позволяет составлять из них локации любого размера, и при этом стыки спрайтов не будут бросаться в глаза

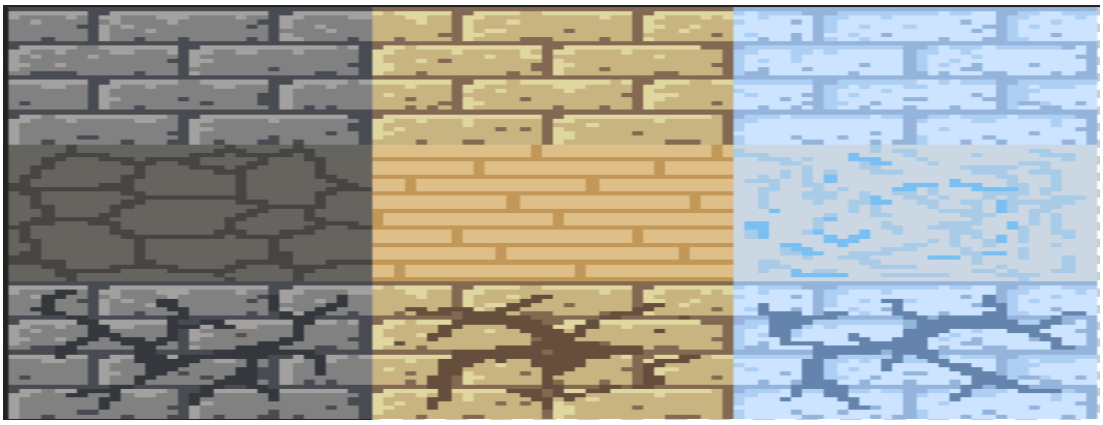

Рисунок 24 – Тайлы уровней трех видов

Необходимо на данном этапе описать сюжет. Игры в жанре roguelike, как правило, не имеют глубокого сюжета и обычно его описание ограничено кратким введением игрока в события игры в самом начале. Такой же подход к рассказу сюжета был использован и в данном проекте.

Игроку предстоит взять на себя роль искателя приключений, отправляющегося на поиски волшебного артефакта Кубка Света (рисунок 24), который находится в глубинах подземелий и, дающий владельцу невиданную мощь.

Использованный в игре шрифт имитирует шрифты аркадных автоматов и является пиксельным шрифтом, что в целом вписывается в стилистику игры.

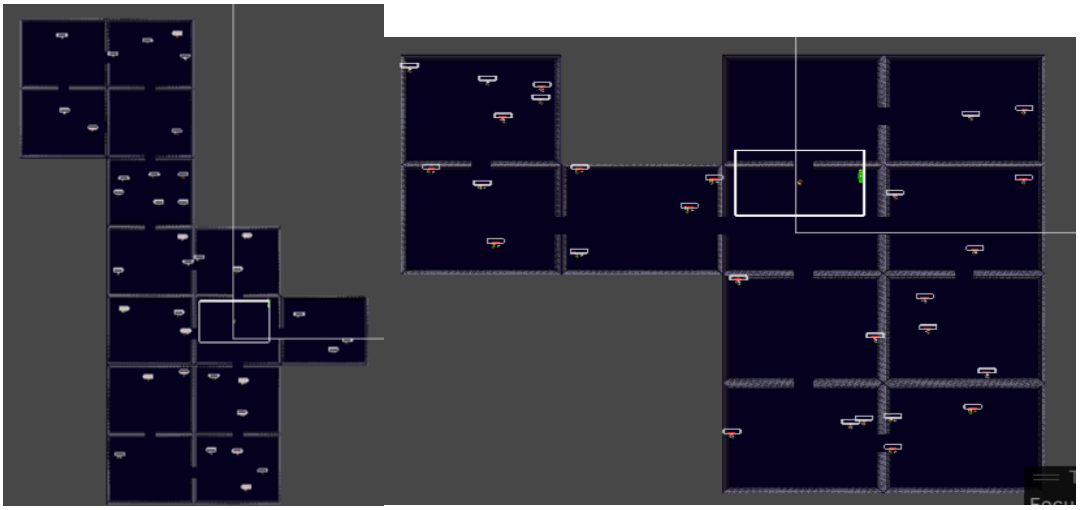

Рисунок 25 – Примеры создания процедурно-генерируемого лабиринта

Создание процедурной генерации мира позволяет создавать новые миры без обязательного вмешательства разработчика, что позволяет разнообразить игру.

2.4.2 Этап работы с визуальной частью игры

После импорта спрайтов в игровой движок необходимо с помощью движка нарезать лист спрайтов на отдельные спрайты, а из них создать игровые объекты – префабы (Prefabs). Для создания героя и врагов необходимо создать пустой объект (рисунок 25), выделить кадры одной из анимаций и перетащить их в созданный пустой объект.

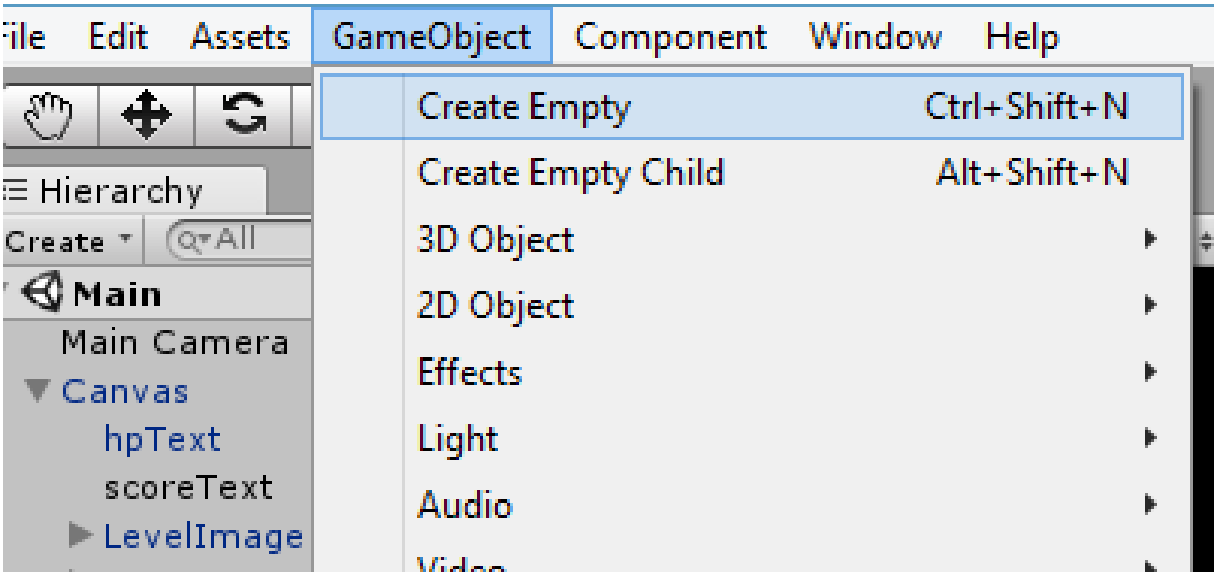

Рисунок 25 – Создание пустого объекта

Движок предложит сохранить анимацию, вместе с анимацией для объекта создается контроллер анимации (рисунок 26), позволяющий настроить переходы анимации из одного состояния в другое.

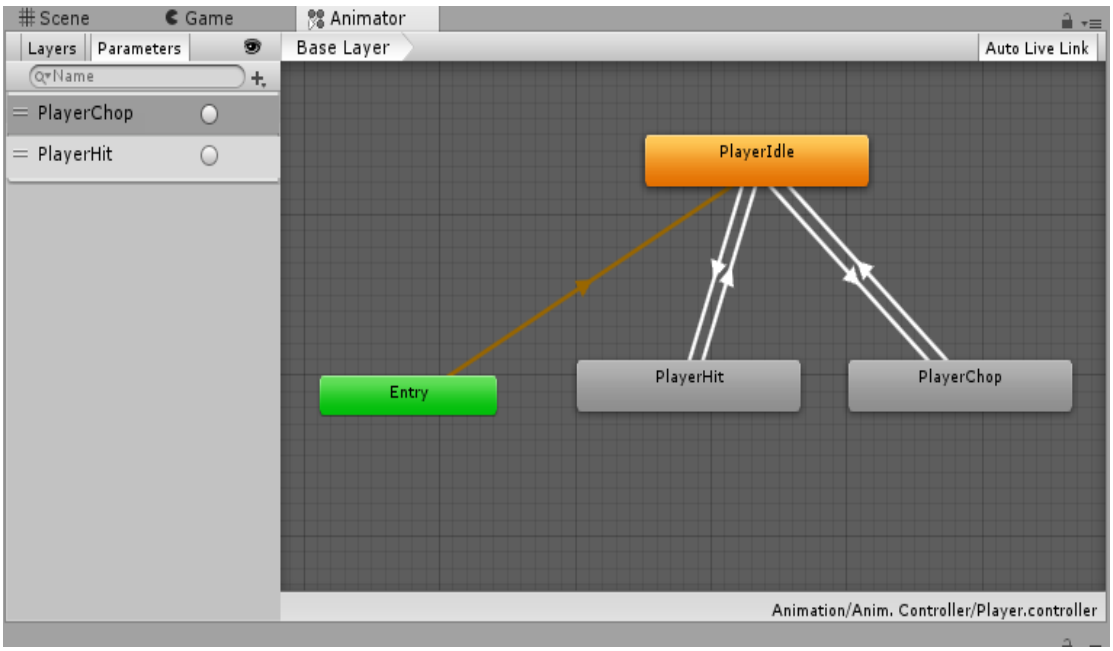

Рисунок 26 – Контроллер анимации объекта Player

Затем необходимо добавить объектам компоненты Rigidbody 2D и Box Collider 2D (рисунок 33). Компонент Rigidbody 2D позволяет настроить перемещение объекта, а Box Collider 2D отвечает за настройку коллизии объекта (то есть столкновение с другими объектами), что позволит объектам – взаимодействовать друг с другом.

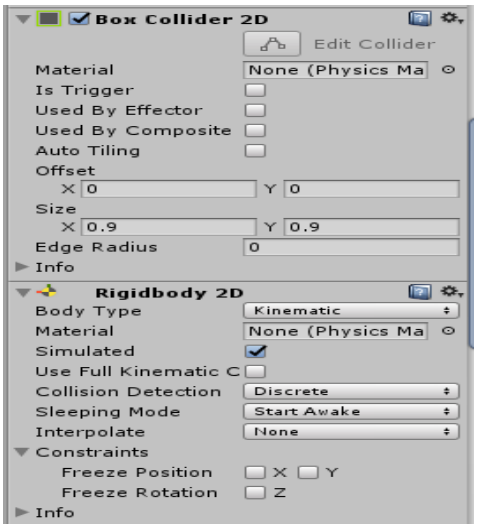

Рисунок 27 – Компоненты Box Collider 2D и Rigidbody 2D объекта Player

Для создания тайлов нужно также создать пустой объект, но добавить ему компонент Sprite Renderer (рисунок 27). В данном компоненте необходимо выбрать спрайт нужного тайла. Если подразумевается взаимодействие с

объектом, то необходимо еще добавить компонент Box Collider 2D.

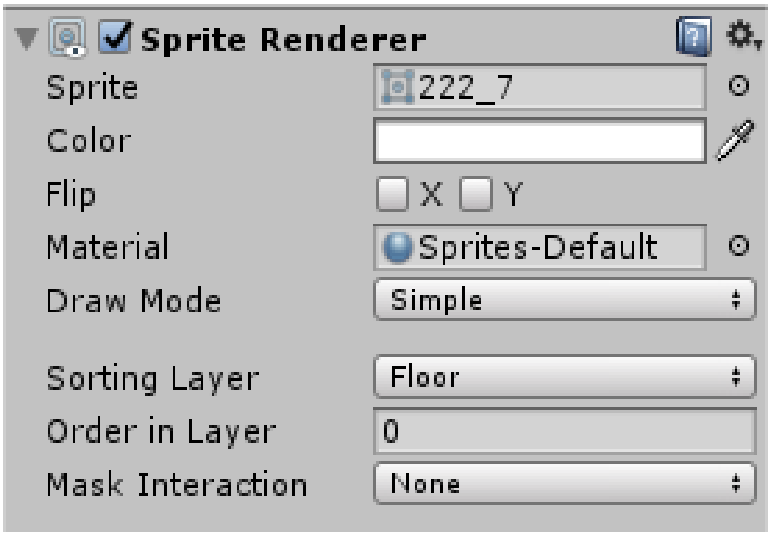

Рисунок 28 – Компонент Sprite Renderer тайла пола Floor 1

Были созданы элементы интерфейса – количество жизней героя, количество набранных очков (рисунок 28).

Рисунок 29 – Элемент интерфейса персонажа

Были созданы инвентарь и окно статистики персонажа, настроены взаимодействие данных элементов между собой (рисунок 29).

Рисунок 30 – Инвентарь и окно статистики персонажа.

Был создан элемент интерфейса Canvas (рисунок 31), в котором были добавлены текстовые поля (для каждого значения свое поле), обновляемые с помощью скриптов.

Рисунок 31 – Элемент интерфейса Canvas

С помощью элемента было настроено и отображение экрана уровня и меню паузы (рисунок 32).

Рисунок 32 – Меню паузы игры

2.4.3 Этап программирования скриптов

Среда разработки Unity хоть и обладает массой различных компонентов с огромным количеством настроек, их функционала явно недостаточно для реализации задуманных механик, поэтому необходимо обратиться к программированию скриптов. Поскольку средой разработки является Unity, то программирование скриптов будет осуществляться на языке программирования C#.

Игра использует один и тот же объект GameManager, в котором генерируется каждый новый уровень.

К объекту GameManager прикреплены 2 скрипта: GameManager и BoardCreator.

Скрипт GameManager отвечает за переходы между уровнями, создание листа врагов и пошаговость игры. Данный скрипт содержит следующие методы:

– Awake. Метод вызывается перед загрузкой игры, загружает лист врагов для данного уровня. Вызывает метод InitGame для первого отображения экрана уровня (надпись «Уровень 1»);

– OnLevelWasLoaded. Метод считает уровни и вызывает метод InitGame для отображения следующих экранов уровня, начиная со второго;

– InitGame. Данный метод отвечает за вывод на экран названия уровня в соответствующий текстовый элемент интерфейса (LevelText) во время

смены уровня игры и очищает лист врагов;

– HideLevelImage. Скрывает экран смены уровней;

– Update. Метод вызывается каждый кадр игры и вызывает интерфейс IEnumerator, отвечающий за поочередное передвижение врагов;

– AddEnemyToList. Метод добавляет в лист врагов, генерируемых с помощью скрипта Enemy.

– GameOver. Отвечает за вывод на экран сообщения о проигрыше игрока, показывая при этом небольшую статистику по прохождению (количество пройденных уровней и набранных очков).

Кроме данных методов, скрипт GameManager содержит интерфейс IEnumerator под именем MoveEnemies. Данный интерфейс после хода игрока

позволяет каждому врагу из листа сделать ход. После хода каждого врага право хода переходит к игроку.

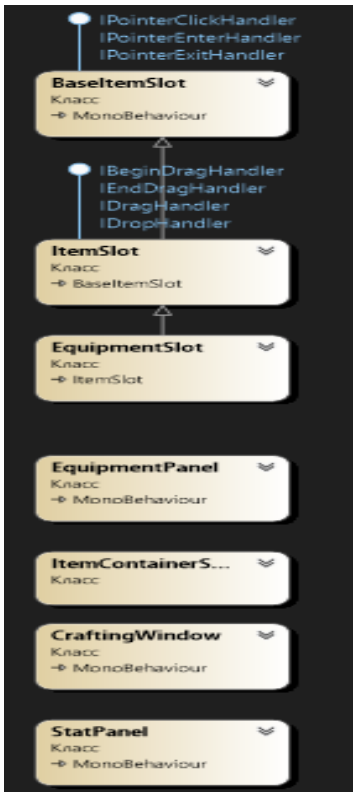

Рисунок 33 – UML-диаграмма классов слотов для хранения предметов.

Скрипт EquipmentPanel отвечает за применения эффектов снаряжения и визуальное отображение.

Скрипт StatPanel отвечает за подсчет эффектов снаряжения и

отображение статистики.

Скрипт Inventory отвечает за эффекты предметов в инвентаре персонажа и их визуальное отображение.

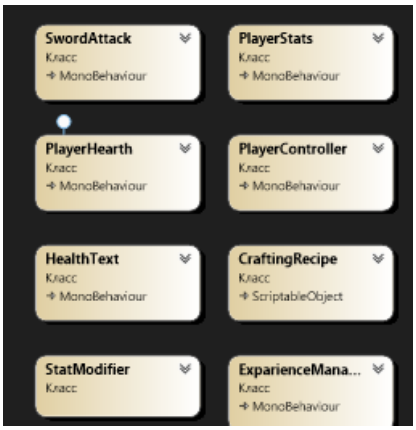

Рисунок 34 – UML-диаграмма классов персонажа

Скрипт Character отвечает за персонажа определяет его основные атрибуты и увеличивает их в зависимости от снаряжения:

– Start – во время загрузки уровня загружает данные об игроке (жизни, очки и т.д.);

– OnDisable – сохраняет информацию об игроке во время переходов между уровнями.

Скрипт PlayerStats отвечает за работу с атрибутами персонажа.

Скрипт PlayerController отвечает за передвижение игрока:

– Update – проверяет каждый кадр, наступила ли очередь игрока ходить, запрещает игроку двигаться по диагонали;

– AttemptMove – дочерний метод скрипта MovingObject. Метод позволяет игроку перемещаться, воспроизводит звук перемещения игрока.

Скрипт SwordAttack отвечает за атаку персонажа:

– EnemyAttack – дочерний метод скрипта MovingObject. Метод отвечает за атаку игроком врагов;

– OnTriggerEnter2D – данный метод отвечает за срабатывание триггеров при взаимодействии с различными игровыми объектами (например, зелья лечения или золото).

Скрипт HealthBar отвечает за здоровье персонажа:

– LoseHP. При получении урона игроком вычитает здоровье игрока;

– UpdatehpText. Реагирует на изменения количества здоровья игрока и изменяет ответственный за вывод на экран количества здоровья элемент интерфейса на соответствующее количество;

– UpdateScoreText. Аналогичен предыдущему, за исключением того, что отвечает не за здоровье игрока, а за количество набранных им очков;

– CheckIfGameOver. При снижении здоровья игрока до нуля вызывает экран с соответствующим сообщением;

– UpdataBaseHealth – изменение максимального значения здоровья персонажа в зависимости от атрибутов.

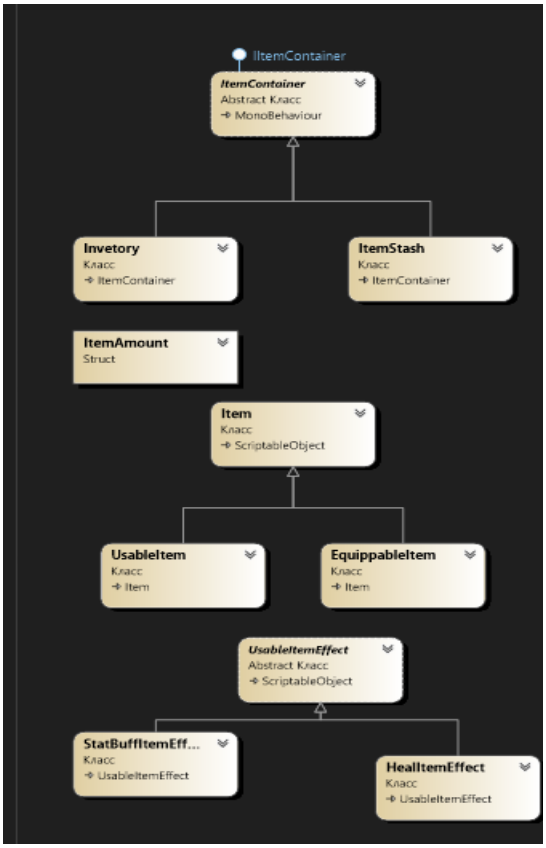

Рисунок 35 – UML-диаграмма классов инвентаря и предметов.

На данном рисунки представлена схема наследования слотов предметов.

Скрипт BaseItemSlot отвечает за отображение объекта.

Скрипт ItemSlot отвечает за возможные действия с предметами.

Скрипт IItemConteiner – это свойство контейнера, определяющее размер

ячейки, ID и количество хранимых объектов.

Скрипт ItemConteiner отвечает за визуальное отображение объектов.

Скрипт Monster отвечает за врагов и содержит следующие методы:

– Start – добавляет врагов в лист, содержащийся в скрипте GameManage;

– Awake – содержит ссылку на объект GameManager;

– AttemptMove – дочерний метод скрипта MovingObject. Определяет будет ли ходить каждый отдельно взятый враг (или сколько ходов этот враг пропустит);

– MoveEnemy – выделяет героя целью врага и заставляет двигаться врага в направлении героя;

– OnCantMove – дочерний метод скрипта MovingObject. Отвечает за атаку врагом героя;

– TakeDamage – следит за изменением здоровья врага при получении им урона от героя. При достижении нулевой отметки здоровья уничтожает этот объект с игрового поля.

Скрипты Slime, Volk, Skelet, Prizrak отвечают за отдельные особенности каждого вида монстров.

Скрипт SpawnerRooms отвечает за генерацию игрового уровня и всех объектов на нем. Содержит следующие классы:

– Awake – содержит ссылку на GameManager;

– Start – определяет, какой из трех типов уровней будет создан. Запускает генерацию уровня;

– CreateRoomsAndCorridors – создание массивов с комнатами и коридорами, генерируемыми в скриптах Room и Corridor соответственно и построение подземелья из этих массивов;

– SetTilesValuesForRooms. Задает клеткам комнаты тип тайла пола;

– Floor;

– SetTilesValuesForCorridors. Задает клеткам коридора тип тайла по-

– ла Floor;

– InstantiateTiles. Генерирует тайлы на уровне, соответствующего вида тайла (пол или стена) из массива;

– InstantiateOuterTiles. Создает дополнительную не разрушаемую стену из тайлов вокруг игрового поля;

– InstantiateVerticalOuterWall – генерация тайлов на вертикальных неразрушаемых стенах;

– InstantiateHorizontalOuterWall – генерация тайлов на горизонтальных неразрушаемых стенах;

– InstantiateFromArray – выдает случайный объект или тайл из списка доступных;

– LayoutObjectsAtRandom – метод, использующийся для случайной генерации юнитов и собираемых объектов на уровне;

– CreateEnemies – с помощью метода LayoutObjectsAtRandom генерирует какие враги и в каком количестве появятся на уровне;

– CreatePotions – с помощью метода LayoutObjectsAtRandom генерирует какие зелья и в каком количестве появятся на уровне;

– CreateGold – с помощью метода LayoutObjectsAtRandom генерирует каком количестве золото появится на уровне;

– CreateAllObjectsInRoom – создает врагов, зелья и золото на игровом поле.

Скрипт Wall отвечает за отображение тайлов поврежденной стены при повреждении стены игроком.

Скрипт IntRange генерирует случайные диапазоны чисел, использующиеся для генерации игрового поля.

Скрипт MovingObject содержит родительские классы, связанные с перемещением объектов и использующиеся в скриптах Player и Enemy.

Скрипт PlayerCam отвечает за наблюдением камеры за персонажем игрока. При помощи данного скрипта герой всегда находится в центре экрана.

Скрипт SoundManager отвечает за воспроизведение звуков и музыки во

время игры.

Скрипт PauseMenu отвечает за срабатывание кнопок в меню паузы. Скрипт MainMenu отвечает за срабатывание кнопок в главном меню.

2.3.2 Этап тестирования

Этап тестирования является ключевым в процессе разработки игры, так как он позволяет не только проверить функциональность и стабильность проекта, но и выявить ошибки, которые могли остаться незамеченными. Кроме того, важно дать возможность потенциальным пользователям ознакомиться с игрой, чтобы оценить интуитивность интерфейса и общую доступность игры.

В процессе первичного тестирования были обнаружены следующие проблемы:

– анимация удара и получения урона персонажем "Player" непреднамеренно повторялась, вместо того чтобы проигрываться единожды;

– система не засчитывала игроку очки за уничтожение врагов;

– происходила одновременная генерация двух объектов "Player" и "Exit";

– несколько незначительных ошибок возникали при генерации уровней.

После исправления выявленных недочетов, игра была повторно запущена, и данные ошибки больше не проявлялись.

Далее, игра была предоставлена для внешнего тестирования различными категориями пользователей, включая как опытных геймеров, так и тех, кто редко или вовсе не играет в видеоигры. Это позволило оценить удобство использования и понятность игрового процесса для широкого круга игроков.

Результаты тестирования показали, что управление и основные игровые механики были интуитивно понятны всем участникам, и дополнительных ошибок в ходе этого этапа выявлено не было. Это свидетельствует о том, что игра готова к дальнейшему распространению и может предложить пользователям качественный и плавный игровой опыт.

2.4.4 Запаковка и выпуск игры

Финальным этапом разработки компьютерной игры является ее запаковка. Во время подготовки к сборке игры необходимо очистить проект от ненужных файлов – это необходимо для того, чтобы итоговый продукт занимал меньше места на жестком диске.

После очистки проекта необходимо открыть настройки проекта Build Settings. В окне настроек проекта необходимо выбрать сцены, которые будут в итоговой игре и желаемую платформу. В данном случае – Windows.

Остается лишь нажать кнопку Build, после чего движок начнет сборку проекта. Можно нажать кнопку Build And Run и тогда проект запустится сразу же после сборки.

После завершения сборки будет доступен готовый продукт, который можно запустить с помощью файла с разрешением .exe.

Не стоит забывать, что для некоторых платформ будет нужно дополнительно доработать проект. Например, для платформ с сенсорным управлением (телефоны или планшеты) необходимо адаптировать проект под сенсорное управление. Но несмотря на это, возможность адаптации на другие платформы является неоспоримым плюсом.

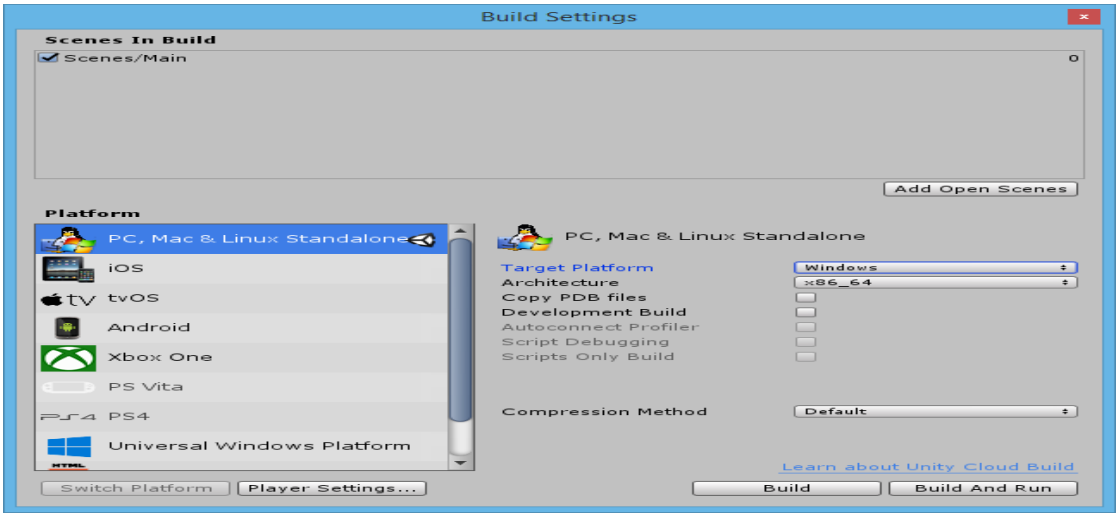

Рисунок 36 – Окно настройки проекта.

3 БЕЗОПАСТНОСТЬ И ЭКОЛОГИЧНОСТЬ

3.1 Безопасность

3.1.1 Вирусы и злонамеренное ПО

Вирусы и вредоносное ПО (Malware) – это зловредные программные коды, созданные с целью повредить систему, украсть данные или нарушить работу игрового приложения. В контексте разработки игры на Unity, вирусы и вредоносное ПО могут включать в себя заражение игровых файлов, внедрение зловредного кода в модификации или патчи, а также манипулирование активами игры.

Воздействие на игровое приложение:

– инфекция активов игры**:**

Заражение игровых файлов – игровые файлы, такие как текстуры, модели и другие ресурсы, могут быть заражены вредоносным кодом, который при запуске игры может исполняться и наносить вред системе или красть данные.

Изменение функциональности – вредоносные коды могут изменять или добавлять функции в игровые файлы, создавая новые уязвимости или осуществляя несанкционированные действия во время работы игры.

Внедрение бэкдоров – вредоносные коды могут создавать скрытые входы (бэкдоры), которые позволят злоумышленникам получить несанкционированный доступ к игровым серверам или клиентам.

– влияние на производительность**:**

Замедление работы игры – вредоносные коды могут потреблять значительные системные ресурсы, вызывая снижение производительности игры, увеличение времени загрузки и снижение FPS (кадров в секунду);

Вылеты и сбои – неоптимизированные или намеренно вредоносные коды могут вызывать вылеты игры, сбои серверов или полные крахи системы, приводя к потере прогресса и необходимости перезапуска.

Площадка распространения:

– пиратские копии игры:

Нелегальные версии – пиратские копии игры, распространяемые через неофициальные каналы, могут содержать встроенное вредоносное ПО, заражающее системы пользователей при установке и запуске;

Распаковка и рипы – в процессе взлома и распаковки оригинальных файлов могут быть внедрены вирусы и вредоносные компоненты.

– модификации и патчи:

Неофициальные моды – модификации игры, создаваемые и распространяемые сообществом, могут содержать вредоносные коды, если они были загружены с ненадежных источников.

Патчи – неофициальные патчи и исправления могут содержать вредоносное ПО, особенно если они предлагают сомнительные улучшения или исправления к игре.

– онлайн-компоненты:

Сетевые взаимодействия – вредоносные коды могут распространяться через сетевые компоненты игры, такие как мультиплеерные режимы или системы распределенной авторизации и обновлений.

Социальная инженерия – пользователи могут быть обманом склонены к скачиванию и установке вредоносных компонентов через социальную инженерию (например, фишинговые письма или поддельные ссылки в игровых чатах).

Методы защиты:

– регулярные обновления и патчи:

Частые обновления – постоянно обновляйте игровое приложение, устраняя известные уязвимости и внедряя новые защитные меры.

Автоматические обновления – использование автоматических обновлений для оперативного внедрения исправлений без необходимости участия пользователя.

Валидация файлов – внедрение механизмов проверки целостности файлов (например, систем хэширования и подписки), чтобы убедиться, что игровые файлы не были изменены или заменены.

– антивирусное ПО:

Интеграция антивирусной защиты на стадии разработки – используйте антивирусные решения во время разработки и тестирования игры для предотвращения внедрения вирусов в процессе создания.

Регулярное сканирование – проводите регулярные антивирусные сканирования на игровых серверах и системах разработки, предотвращая распространение вредоносных кодов.

Рекомендации для пользователей – рекомендуйте пользователям устанавливать и регулярно обновлять антивирусное программное обеспечение на своих системах.

– защищенные каналы для распространения:

Проверенные платформы – распространяйте игру исключительно через проверенные и надежные платформы (например, Steam, PlayStation Store, Google Play), обеспечивающие защиту от вредоносного ПО.

Шифрование данных – используйте шифрование для защиты данных во время их передачи по сети, предотвращая перехват и изменение файлов.

Цифровая подпись приложений – подписывайте игру цифровыми сертификатами, которые подтверждают подлинность и целостность файлов, предотвращая их несанкционированное изменение

3.1.2 Фишинг

Фишинг – это тип кибератаки, при которой злоумышленники пытаются получить личные и конфиденциальные данные, такие как логины, пароли и финансовую информацию, выдавая себя за доверенные источники. В контексте игровых приложений на платформе Unity, фишинг может проявляться в виде поддельных веб-ресурсов, писем электронной почты или внутриигровых сообщений, которые выглядят официально, но на самом деле предназначены

для кражи данных пользователей.

Меры борьбы с фишингом:

– обучение игроков основам кибергигиены – информирование игроков о том, как распознавать подозрительные сообщения и веб-сайты;

– проведение информационных кампаний и создание руководств и FAQ (Часто задаваемых вопросов) по вопросам безопасности;

– использование внутриигровых уведомлений для предупреждения игроков о текущих фишинговых атаках.

Разработка и внедрение системы проверки подлинности сообщений внутри игры:

– внедрение механизмов, которые помогут игрокам идентифицировать официальные сообщения. Например, использование цифровых подписей или водяных знаков, которые трудно подделать.

– создание и поддержка официальных каналов коммуникации представленные в пунктах а-б:

а) обеспечение игроков доступом к проверенным и достоверным источникам информации, таким как официальный сайт игры, форумы и социальные сети. Это помогает снизить вероятность того, что игроки станут жертвами фишинга, зная, где искать достоверную информацию;

б) также разработчики могут применять технологии машинного обучения для анализа и выявления фишинговых попыток, а также внедрять многофакторную аутентификацию для усиления безопасности учетных записей. Регулярные аудиты безопасности и тесты на социальную инженерию помогут оценить устойчивость игрового комьюнити к фишинговым атакам и улучшить защитные меры.

Эти действия позволят создать эффективную защиту от фишинговых атак и обеспечить сохранность конфиденциальной информации игроков.

3.1.3 Несанкционированный доступ

Несанкционированный доступ к информационным системам и данным

является одной из наиболее распространённых угроз безопасности в современных организациях. Этот аспект затрагивает широкий спектр действий, направленных на получение доступа к ресурсам системы без должного разрешения. В рамках данной главы рассмотрим ключевые методы защиты от несанкционированного доступа и их значение для общей безопасности системы.

– аутентификация и авторизация:

Аутентификация представляет собой процесс подтверждения подлинности пользователя, устройства или другой сущности, претендующей на доступ к системе. Одним из стандартных методов аутентификации является использование паролей, но в современных условиях этого может быть недостаточно. Более надежными методами являются:

Многофакторная аутентификация (MFA): требует предоставления нескольких доказательств подлинности, таких как что-то, что пользователь знает (пароль), что-то, что он имеет (смартфон для получения одноразового кода), и что-то, что он является (биометрические данные).

Биометрическая аутентификация: использует уникальные физиологические характеристики пользователя, такие как отпечатки пальцев, распознавание лиц или сканирование радужки глаза.

Авторизация осуществляется после успешной аутентификации, и она определяет, какие ресурсы и данные может использовать конкретный пользователь. Это достигается путем настройки ролей и прав доступа на уровне системы.

– шифрование данных

Шифрование играет ключевую роль в защите данных как в процессе их передачи, так и при хранении. Использование современных алгоритмов шифрования, таких как AES (Advanced Encryption Standard), обеспечивает защиту данных от перехвата и компрометации. Шифрование может быть реализовано на различных уровнях:

Шифрование транспортного уровня (TLS) – защищает данные во время

их передачи по сети, обеспечивая их конфиденциальность и целостность.

Шифрование на уровне данных– данные шифруются непосредственно в базе данных или на уровне файловой системы, что предотвращает их прочтение даже при физическом доступе к носителю данных.

– межсетевые экраны и системы обнаружения вторжений – межсетевые экраны (брандмауэры) обеспечивают фильтрацию трафика между различными сегментами сети на основе настроенных правил доступа, что предотвращает несанкционированные попытки подключения к внутренним ресурсам. Системы обнаружения и предотвращения вторжений (IDS/IPS) позволяют выявлять подозрительную активность и предотвращать потенциальные атаки в реальном времени;

– журналирование и мониторинг – ведение журналов событий (логирование) и постоянный мониторинг системы позволяют отслеживать все попытки доступа и выявлять потенциально вредоносные действия. Анализ логов помогает в оперативном выявлении инцидентов безопасности и проведении расследований после их обнаружения;

– обновление и патчинг программного обеспечения – уязвимости в программном обеспечении являются одной из основных причин несанкционированного доступа. Регулярное обновление ОС и приложений, а также своевременное применение патчей, устраняющих известные уязвимости, значительно повышают уровень безопасности системы;

– обучение и осведомленность сотрудников – человеческий фактор остается одним из самых уязвимых звеньев в системе безопасности. Регулярное обучение сотрудников методам защиты информации, формирование культуры безопасности и повышение осведомленности о современных угрозах способствуют предотвращению ошибок, которые могут привести к несанкционированному доступу;

В заключение, защита от несанкционированного доступа требует комплексного подхода, включающего современные технические решения,

организационные меры и постоянное повышение осведомленности пользователей. В условиях возрастающих угроз кибербезопасности такой подход позволяет существенно уменьшить риски и обеспечить надежную защиту информации.

3.1.4 Утечка данных

Утечка данных означает несанкционированный доступ, использование или раскрытие защищённой информации, что может обернуться серьёзными последствиями, такими как юридические, финансовые и репутационные проблемы для организации и её клиентов.

Методы предотвращения утечек данных представлены в пунктах а-з:

а) принципы безопасного программирования:

Поддержание целостности входящих данных – внедрение механизмов валидации и очистки входящих данных для предотвращения атак типа SQL-инъекция и кросс-сайтовый скриптинг.

Управление доступом и сессиями – введение политик минимальных привилегий и использование надёжных способов аутентификации и авторизации.

Обработка ошибок и исключений – разработка логирования и отчётности, исключающих утечку конфиденциальной информации через сообщения об ошибках.

б) защита серверов для хранения данных:

Физическая и программная защита – обеспечение многоуровневой безопасности серверов с использованием брандмауэров и систем обнаружения вторжений.

Регулярное обновление и патчинг – постоянное внедрение обновлений безопасности для операционных систем и приложений для устранения известных уязвимостей.

в) регулярные аудиты безопасности:

Идентификация уязвимостей – применение автоматизированных

инструментов и ручных проверок для выявления слабых мест в безопасности инфраструктуры.

Соблюдение стандартов: Проверка соответствия актуальным нормативам и стандартам безопасности данных.

г) политика безопасности данных – создание документированных инструкций по обработке, хранению данных и действиям в случае инцидентов утечки данных.

д) обучение сотрудников – разработка и проведение образовательных программ для сотрудников с целью информирования о возможных угрозах и методах предотвращения утечек данных.

е) мониторинг и реагирование на инциденты:

Настройка инструментов мониторинга для выявления необычных паттернов поведения, указывающих на возможную утечку данных.

Формирование команды для быстрого реагирования на инциденты безопасности.

ж) шифрование – использование современных алгоритмов шифрования для обеспечения конфиденциальности данных на стадии хранения и передачи.

з) резервное копирование и восстановление – разработка стратегии резервного копирования и восстановления данных для предотвращения потерь информации в случае утечек.

Комплексное применение этих мероприятий формирует многоуровневую защиту от утечек данных, обеспечивая сохранение конфиденциальности и целостности информационных активов организации.

3.2 Экологичность

Энергоэффективность – использование оборудования, которое соответствует стандартам Energy Star или директивам Европейского Союза по энергоэффективности (EU Ecodesign Directive), позволяет снижать энергетические затраты при разработке. Это достигается благодаря применению компонентов с низким энергопотреблением и алгоритмов управления энергией, которые

адаптируют потребление энергии в зависимости от нагрузки на систему.

Системы управления энергией – автоматизированные системы управления энергопотреблением, такие как Advanced Configuration and Power Interface (ACPI), дают возможность компьютерам и серверам переходить в режимы с пониженным энергопотреблением при отсутствии активности, что помогает сократить расходы на энергию и продлить срок службы оборудования.

Минимизация отходов представлены в пунктах а-в:

а) внедрение цифровых инструментов для документации и коммуникации исключает необходимость в использовании бумажных носителей, что уменьшает объем отходов и снижает экологическое воздействие их утилизации.

б) политика устойчивого потребления – разработка политики устойчивого потребления, основанной на принципах циклической экономики (circular economy), стимулирует повторное использование и переработку материалов, что минимизирует количество отходов и оптимизирует использование ресурсов.

в) утилизация ресурсов:

– применение лицензий, которые можно передавать или продлевать, уменьшает необходимость в приобретении новых копий программного обеспечения, что снижает производственные отходы и углеродный след;

– использование облачных решений для хранения данных и совместной работы устраняет потребность в физических носителях, таких как жесткие диски и оптические носители, что способствует уменьшению энергопотребления и сокращению отходов.

Социальная ответственность представлена в пунктах а-д:

а) поддержка экологических инициатив:

Создание и реализация программ по посадке деревьев и созданию зеленых зон в городских и пригородных районах. Это не только улучшает внешний

вид территорий, но и способствует улучшению микроклимата и снижению загрязнения воздуха.

Внедрение мер по очистке водоемов, установке фильтрационных систем на предприятиях и проведение экологических аудитов для оценки и уменьшения воздействия на окружающую среду.

б) волонтерская деятельность:

– организация таких мероприятий, как дни чистоты и уборка мусора в общественных местах и национальных парках, способствует формированию экологической культуры и повышению осведомленности среди населения;

– создание возможностей для участия сотрудников в волонтерских программах, включая предоставление административного отпуска для участия в экологических проектах и партнерство с некоммерческими организациями и экологическими фондами.

в) корпоративная культура устойчивого развития:

– проведение семинаров, тренингов и мастер-классов для сотрудников по вопросам экологии, устойчивого развития и экологически ответственного поведения в повседневной жизни;

– введение корпоративных стандартов и политик, направленных на сокращение использования пластика, бумаги и других материалов, негативно влияющих на окружающую среду.

г) обучение и развитие:

– экологическое образование: Организация программ для сотрудников по повышению осведомленности о проблемах окружающей среды и методах устойчивого развития способствует формированию экологически ответственного подхода к профессиональной деятельности;

– внедрение принципов экологического дизайна в процесс разработки позволяет создавать продукты, которые учитывают жизненный цикл ресурсов и минимизируют экологический отпечаток.

д) экологические достижения:

– проведение мероприятий по уменьшению выбросов углекислого газа, таких как оптимизация логистики и использование возобновляемых источников энергии, что приведет к значительному снижению углеродного следа проекта;

– внедрение комплексной системы управления отходами, включая меры по сбору, сортировке, переработке и безопасной утилизации, демонстрирует приверженность принципам устойчивого развития и заботу об окружающей среде.

3.3 Чрезвычайные ситуации

Чрезвычайная ситуация (ЧС) – это непредвиденное событие или обстоятельство, которое нарушает нормальный ход разработки или эксплуатации программного обеспечения, вызывает значительные задержки, финансовые потери или ухудшение качества продукта. ЧС могут включать технические проблемы, угрозы безопасности данных, негативные отзывы пользователей, юридические и этические вопросы, а также трудности с маркетингом и продажами.

В процессе разработки игрового приложения на платформе Unity могут возникать различные чрезвычайные ситуации, которые способны повлиять на ход проекта и его конечное качество. Рассмотрим основные типы таких ситуаций, их причины и методы эффективного реагирования на них.

Технические проблемы:

Сбои и ошибки в коде:

– причины:

– недоработки и баги, возникающие при написании кода;

– коллизии при объединении различных фрагментов кода, особенно в командных проектах;

– Недостаточное тестирование на разных этапах

разработки;

– методы решения:

– регулярное тестирование: Внедрение автоматизированных систем тестирования, таких как unit-тесты, и интеграция систем непрерывной доставки (CI/CD) для раннего обнаружения и исправления ошибок;

– контроль версий: Использование систем контроля версий (например, Git) для управления изменениями, проведения код-ревью и минимизации рисков коллизий;

– отладка: Проведение регулярных сеансов отладки с использованием инструментов Profiler и Console в Unity для выявления и устранения ошибок.

Проблемы с производительностью:

– причины:

– неоптимизированный код и ассеты;

– высокие требования к ресурсам на конечных устройствах;

– неэффективное управление памятью и вычислениями;

– методы решения:

– оптимизация ассетов: сжатие текстур и аудио, использование более легких моделей и оптимизация скриптов;

– профилирование – профилирование приложения для выявления узких мест и их последующая оптимизация;

– LOD системы – использование Level of Detail (LOD) систем для снижения нагрузки на GPU путем динамического изменения уровня детализации объектов.

Безопасность данных:

Уязвимости в безопасности:

– причины:

– недостаточное внимание к защите данных при

разработке;

– использование устаревших или уязвимых библиотек;

– методы решения:

– принципы безопасного кодирования: Внедрение лучших практик кодирования для защиты информации и проверки вводимых данных;

– обновление библиотек: Регулярное обновление используемых библиотек и фреймворков.

Взломы и утечки данных:

– причины:

– недоработки в системе безопасности;

– отсутствие многоуровневой защиты;

– методы решения:

– шифрование данных: Применение современных методов шифрования для защиты данных при передаче и хранении;

– многоуровневая защита: Внедрение фаерволов, антивирусов и проведение регулярных пенетрационных тестов.

Пользовательский опыт:

Негативный отзыв:

– причины:

– неудобный интерфейс;

– технические проблемы и баги, влияющие на игровой процесс;

– методы решения:

– система обратной связи: Создание внутри приложения системы сбора отзывов и предложений;

– аналитика пользователя: Анализ поведения

пользователей для выявления и устранения проблем UX/UI.

Проблемы доступности

– причины:

– не значение потребностей пользователей с ограниченными возможностями.

– отсутствие адаптации интерфейса для всех категорий пользователей.

– методы решения:

– инклюзивный дизайн: Разработка с учетом руководств по доступности, таких как WCAG.

– тестирование доступности: Проведение тестирования с участием пользователей с ограниченными возможностями.

Юридические и этические вопросы:

Нарушение авторских прав:

– причины:

– использование нелицензированных ассетов или контента.

– методы решения:

– аудит контента: Проведение регулярных аудитов контента на соответствие авторским правам.

– лицензирование: Использование только лицензированного или открытого исходного кода.

Этические дилеммы:

– причины:

– недостаточное внимание к этическим аспектам при разработке.

– методы решения:

– этический контроль: Включение специалиста по

этике в команду и проведение обучающих семинаров по этическим вопросам.

Маркетинг и продажи:

Низкие продажи:

– причины:

– недостаточная проработка маркетинговой стратегии.

– невнимательность к потребностям целевой аудитории.

– методы решения:

– маркетинговая стратегия: Исследование рынка и целевой аудитории, создание точечных маркетинговых кампаний.

– анализ рынка: Регулярный анализ рынка и адаптация продукта под текущие тенденции и запросы.

Учитывая все вышеперечисленные аспекты, можно разработать и внедрить стратегии для предотвращения и решения чрезвычайных ситуаций при создании игрового приложения на Unity. Это поможет не только создать высококачественный продукт, но и обеспечить его стабильность, безопасность и популярность среди пользователей.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной работы была успешно достигнута основная цель – разработка уникальной и захватывающей компьютерной игры в жанре RPG, использующей передовые технологии и методы разработки. Проведённое исследование позволило глубже понять процессы создания игр и внести вклад в развитие игровой индустрии.

На начальном этапе работы был проведён всесторонний анализ существующих RPG-игр, что позволило выделить ключевые механики и характеристики жанра. Это исследование стало основой для формирования чётких функциональных требований к разрабатываемой игре и помогло определить, какие элементы делают RPG-игры популярными и привлекательными для игроков.

Изучение инструментов разработки и сравнение различных платформ позволили выбрать наиболее подходящие средства для создания игры. В результате использовались инструменты, обеспечивающие необходимый уровень гибкости и функциональности, что позволило реализовать все задуманные концепции и механики на должном уровне.

Большое внимание было уделено разработке дизайна игрового мира. Были созданы детализированные концепции существ, персонажей и окружения, а их визуализация в графическом редакторе обеспечила уникальность и привлекательность визуального стиля игры. Это помогло создать богатый и захватывающий виртуальный мир, в который игроки смогут погружаться с удовольствием.

Разработанный прототип игры включал основные игровые элементы и интерфейс, что позволило эффективно демонстрировать ключевые концепции и механики игры на практике. Это стало важным этапом для тестирования и отладки, которые проводились тщательно и выявили все технические и

игровые недочёты. В результате были устранены все существенные проблемы и обеспечено высокое качество и стабильность конечного продукта.

Таким образом, данная работа успешно реализовала поставленные задачи и предоставила новый взгляд на процесс разработки RPG-игр. Результаты исследования не только продемонстрировали возможность создания уникальной и захватывающей игры, но и предложили пути для дальнейшего развития жанра, способствуя расширению границ интерактивных развлечений и предложениям уникального игрового опыта для пользователей. Всё это подтверждает значимость и актуальность выполненного исследования в контексте современного игрового процесса.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Бонд, Д.Г. Unity и C# Геймдев от идеи до реализации / Д.Г. Бонд. – Санкт-Петербург: Питер, 2019.

2 Бацеко, Р.В. Платформа Unity как средство разработки игр в жанре головоломка / Р.В. Бацеко – Краснодар, 2023 – 30 стр.

3 Виды компьютерных игр / URL: https://wobla.ru/infomat/games.aspx# (дата обращения: 13.04.2024)

4 Головоломка (жанр компьютерных игр) / URL: https://ru.wikipedia.org/wiki/Головоломка_(жанр_компьютерных_игр) (дата обращения: 10.02.2023).

5 Гейг, М. Разработка игр на Unity за 24 часа / М. Гейг. – Москва : Бомбора, 2020. – 466 с.

6 Денисов, Д. В. Разработка игры в Unity. С нуля до реализации / Д. В. Денисов. – ЛитРес:Самиздат, 2021.

7 Использование Unity для разработки приложений / URL: https://androidtools.ru/coding/ispolzovanie-unity-dlya-razrabotki-prilozhenij/ (дата обращения: 11.01.2023).

8 Ильин, В. Основы создания 2D персонажа в Unity 3D / URL: https://habrahabr.ru/post/211472/ (дата обращения: 15.03.2024).

9 Коситова, Я. С. История создания жанра игр «головоломка» и перспективы развития на медиа рынке 21 века/ Я. С. Коситова. – Ростов-На-Дону, 2019.

10 Официальный сайт Unity. / URL: https://unity3d.com/ru (дата обращения: 20.03.2023).

11 Официальный сайт Visual Studio. / URL:https://www.visualstudio.com-/ru/vs/ (дата обращения: 20.03.2023).

12 О Godot Engine / URL: https://docs.godotengine.org/ru/stable/about/introduction.html# (дата обращения: 10.04.2023).

13 Официальный сайт компании Unreal Engine / URL: https://www.unr-ealengine.com/en-US/download (дата обращения: 25.03.2023).

14 Официальный сайт компании Godot / URL: https://godote-ngine.o-rg/download/windows/ (дата обращения: 10.02.2023).

15 Основы анимации в Unity: пер. с англ. Р.Рагимова. – М.: ДМК Пресс, 2016.– 176 с.

16 Перельман, Я.И. Дом занимательной науки: Лабиринты / Я. И. Перельман. – Проспект, 2022 г.

17 Рамбо, Дж. UML 2.0. Объектно-ориентированное моделирование разработка/ Дж. Рамбо. – СПб.: Питер, 2007. – 544 с.

18 Торн, А. Основы анимации в Unity: учебное пособие/ А. Торн – ред.: Д. Мовчан, переводчик: Р. Рагимов. – Москва: ДМК, 2016 – 176с.

19 Термин «компьютерная игра»: опыт междисциплинарного анализа /URL:https://cyberleninka.ru/article/n/termin-kompyuternaya-igraopytmezhdist-siplinarnogo-analiza/viewer (дата обращения: 15.02.2023).

20 Хокинг Дж. Unity в действии. Мультиплатформенная разработка на C#: учебное пособие/ Джозеф Хокинг – Санкт-Петербург: Питер, 2016 – 336с.

21 Что такое игровой движок? / URL: https://club.dns-shop.ru/blog/t-64-videoigryi/34701-chto-takoe-igrovoi-dvijok/ (дата обращения: 20.11.2022).

22 Что такое Visual Studio? / URL: https://learn.mi-crosoft.com/ru-ru/visualstudio/get-started/visual-studioide?vie-w=vs-2022 (дата обращения: 10.02.2023).

23 Asperite / URL: https://www.aseprite.org/ (дата обращения:29.01.2023).

24 Adobe Photoshop / URL: https://www.adobe.com/products/photoshop/ (дата обращения: 29.01.2023).

25 C# Programming Guide / URL: https://msdn.microsoft.com/en us/library/67ef8sbd.aspx (дата обращения: 10.02.2023).

26 The evolution of video gaming and content consumption / URL:https://www.pwc.com/us/en/industry/entertainmentm+edia/publictions/asets/p-wc-video-gaming-and-content-consumption.pdf (дата обращения: 20.11.2023).

27 Unity Manual. / URL: http://docs.unity3d.com/Manual/index.html (дата обращения: 10.12.2022).

28 Godot / URL: https://godotengine.org (дата обращения: 20.11.2023).

ПРИЛОЖЕНИЕ А

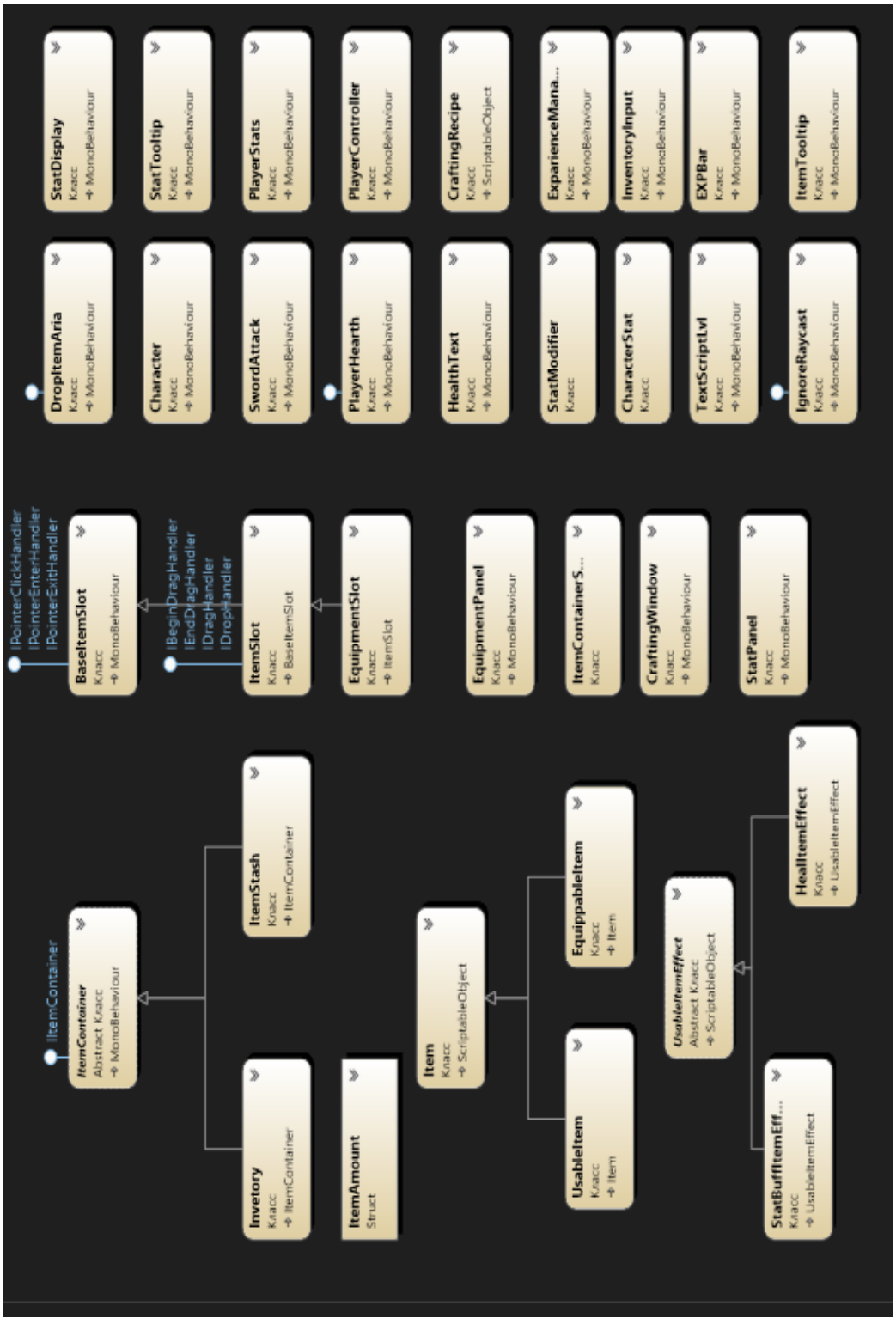

Рисунок А1 – UML-диаграмма классов приложения «BeLive» часть 1

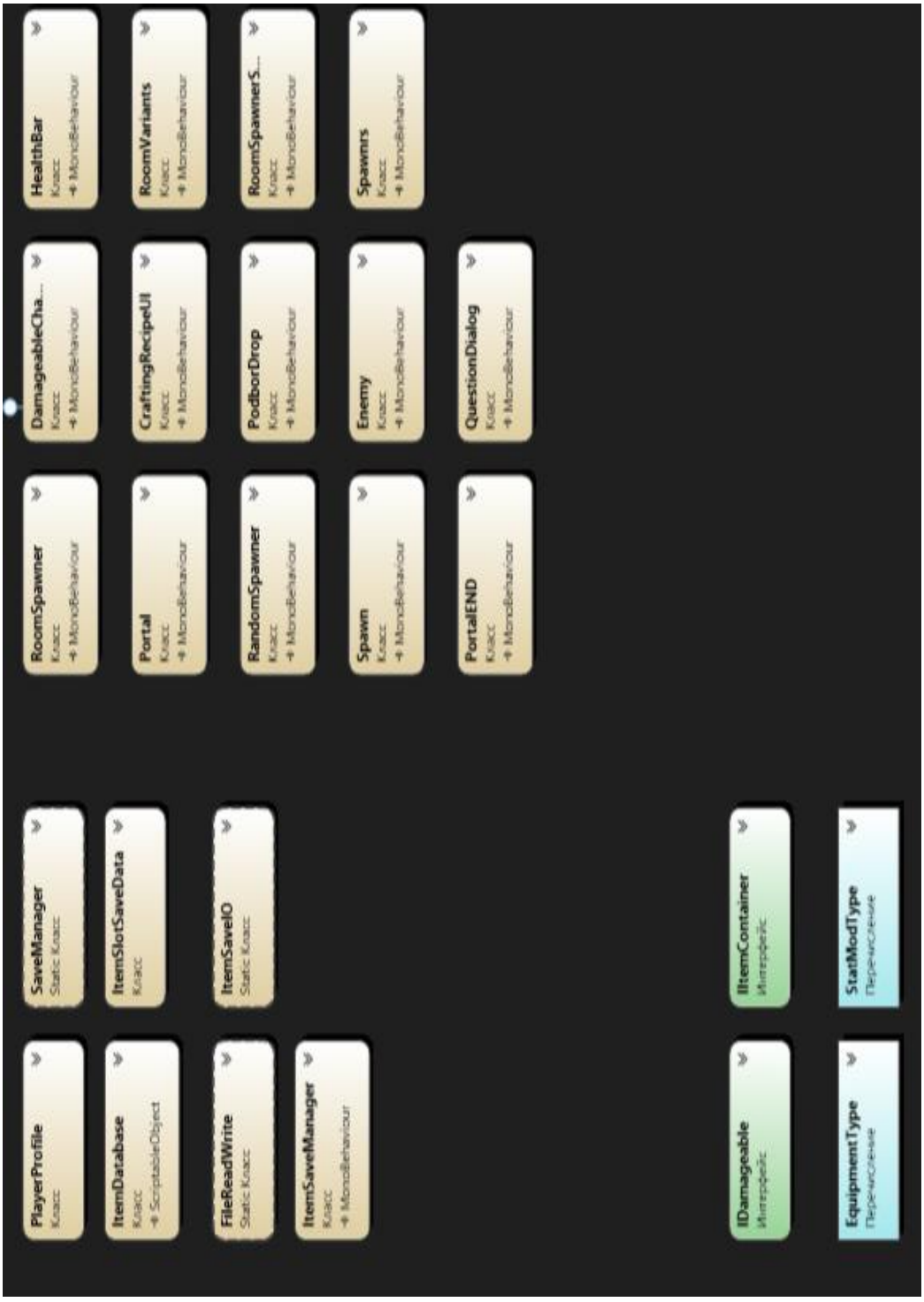

Рисунок А2 – UML-диаграмма классов приложения «BeLive» часть 2

ПРИЛОЖЕНИЕ Б

Техническое задание

1 ВВЕДЕНИЕ

1.1 Название проекта

Разработка 2D игры "BeLive" на Unity

1.2 Разработчик

Разработчик: студент 0103-об Института компьютерных и инженерных наук.

1.3 Перечень документов

Деятельность проекта осуществляется на основании следующих документов:

Закон РФ от 19.04.1991 N 1032-1 (ред. от 08.12.2020) "О занятости населения в Российской Федерации";

Трудовой кодекс РФ;

Налоговый кодекс РФ.

1.4 Плановые сроки работы

Начало: 1 декабря 2023 года

Окончание: 1 июня 2024 года

2 НАЗНАЧЕНИЕ И ЦЕЛИ СОЗДАНИЯ

2.1 Назначение игры

«BeLive» – это 2D игра в жанре платформер с элементами RPG, предназначенная для развлечения и развития логического мышления у игроков. Игра предоставляет захватывающие уровни, увлекательный сюжет и разнообразные механики.

2.2 Цели создания игры

Перед разработкой были поставлены следующие ключевые цели:

Обучение навыкам разработки 2D-игр с использованием Unity.

Создание качественного игрового продукта для демонстрации.

Развитие навыков программирования и использования игровых движков.

Привлечение внимания студентов и преподавателей к проекту через интересный и интерактивный контент.

3 ХАРАКТЕРИСТИКИ ОБЪЕКТА АВТОМАТИЗАЦИИ

3.1 Описание системы

Разрабатываемая система представляет собой игровое приложение, реализованное на движке Unity. Основные компоненты системы включают:

Игровой движок (Unity): Обеспечивает графику, физику и логику игры.

Управление персонажем: Механики движения и взаимодействия с окружением.

Система уровней: Набор игровых уровней с различной сложностью и задачами.

Система инвентаря: Управление предметами и их использование.

Интерфейс пользователя (UI): Окна меню, статистика игрока, инвентарь и прочее.

3.2 Автоматизация процессов

Использование автоматизации в игре включает:

Создание и управление игровыми объектами: Автоматическое создание и удаление объектов на уровне, настройка сценариев для игрового процесса.

Управление поведением NPC (неигровых персонажей): Реализация ИИ для NPC, создание сценариев для взаимодействия с персонажем.

Автоматическое сохранение и загрузка прогресса игрока: Реализация функций автосохранения и загрузки данных игрока при старте игры.

Путь прохода NPC и врагов: Использование pathfinding алгоритмов для определения пути NPC и врагов.

4. Требования к системе

4.1 Требования к системе в целом

Безопасность данных: Система должна обеспечивать сохранность данных сохранений и настроек пользователя.

Удобный интерфейс: Интуитивно понятный интерфейс для всех категорий пользователей.

Оптимизация процессов: Оптимизация для стабильной работы на различных устройствах.

Автоматизация: Включение автоматических функций, таких как автосохранение.

Гибкость: Возможность адаптации и расширения.

4.2 Требования к структуре и функционированию системы

4.2.1 Навигация и интерфейс

Четкое меню.

Подразделы: "Новая игра", "Продолжить", "Выход".

Легкость в навигации по игровым экранам и интерфейсам.

4.2.2 Информация об игре

Описание управляемых персонажей.

Описание механик и целей игры.

Поддержка различных типов управления (клавиатура/геймпад).

4.2.3 Интерфейс пользователя

Отображение текущих заданий, инвентаря, статуса персонажа.

Настройка графики, звука и управления.

Удобное управление инвентарем и взаимодействие с предметами.

4.2.4 Перспективы развития и модернизация

Добавление новых уровней и персонажей: Расширение игрового контента путем добавления новых уровней и персонажей с уникальными способностями.

Разработка онлайн-режима: Возможность проведения многопользовательских сессий, где игроки смогут соревноваться или сотрудничать.

Интеграция с социальными сетями: Встроенные функции для обмена успехами, публикации скриншотов и достижений в социальных сетях.

Внедрение дополнительных языков: Перевод интерфейса и контента на несколько языков для расширения аудитории.

4.3 Требования к надежности

Надежность системы имеет ключевое значение для успешного выполнения проекта. Включает в себя:

Безопасность:

Защита данных игрока: Информация о прогрессе и настройках должна быть защищена от потери данных и несанкционированного доступа.

Доступность:

Стабильная работа: Игра должна быть доступна для пользователей круглосуточно без сбоев и подлагиваний.

Резервное копирование:

Регулярное создание копий данных игры: Реализация системы резервных копий, чтобы игроки могли восстановить свои данные в случае сбоя.

Обновление и поддержка:

Регулярные обновления: Включает установку последних обновлений для устранения багов и добавления нового функционала.

Масштабируемость:

Система должна быть готова к увеличению нагрузки и расширению, особенно если планируется разработка дополнительного контента или онлайн-режима.

Отказоустойчивость:

Включение резервных серверов и использование облачных технологий

для обеспечения бесперебойной работы даже при возможных сбоях.

Тестирование и отладка:

Регулярное тестирование игры для выявления и исправления ошибок до их появления у конечных пользователей.

5 Состав и содержание работ по созданию системы

Разработка 2D игры «BeLive» требует выполнения нескольких последовательных этапов. Каждый этап имеет свои ключевые задачи и результаты.

Этапы разработки:

1. Исследование и анализ:

1.1. Изучение компонентов игры и существующих аналогов:

Анализ текущих решений на рынке.

Выделение ключевых механик и разработка уникальных фич.

1.2. Определение технических требований:

Определение основной концепции игры.

Формулирование требований к функциональности.

2. Составление технического задания:

2.1. Уточнение требований заказчика:

Совместное обсуждение с преподавателем и руководителем курсового проекта.

2.2. Определение используемых средств разработки:

Выбор движка (Unity), инструментов разработки, языков программирования (C#).

3. Разработка системы:

3.1. Проектирование логики игры:

Создание схемы игровой механики.

Разработка алгоритмов поведения NPC и врагов.

3.2. Создание уровней и объектов:

Дизайн уровней и размещение игровых объектов.

Внедрение механик взаимодействия с пользователями.

4. Тестирование:

4.1. Проверка работы всех компонентов игры:

Тестирование на различных устройствах.

Исправление выявленных багов и ошибок.

4.2. Исправление выявленных ошибок:

Выявление и устранение проблем в системе до выпуска конечного продукта.

5 СОГЛАСОВАНИЕ С ЗАКАЗЧИКОМ

5.1 Презентация и проверка соответствия требованиям

Демонстрация игры заказчику.

Совместное обсуждение и доработка проекта в соответствии с пожеланиями и замечаниями.

5.2 Внесение необходимых правок по замечаниям заказчика

Исправление выявленных замечаний.

Финальная проверка перед сдачей проекта.

6 ПОРЯДОК КОНТРОЛЯ И ПРИЁМКИ СИСТЕМЫ

6.1 Контроль на всех этапах разработки

Заказчик принимает участие на всех стадиях контроля и приемки, что гарантирует соблюдение всех требований и спецификаций.

6.2 Проверка системы на соответствие требованиям ТЗ

При приёме системы заказчик должен:

Ознакомиться с документацией и руководством пользователя.

Присутствовать на тестировании системы для проверки её работоспособности.

Оценить соответствие функционала игры техническому заданию.

6.3 Тестирование функционала игры

Все тесты проводится в условиях реальной работы, включая:

Игровой процесс.

Взаимодействие с интерфейсом.

Проверку всех предусмотренных автоматизированных процессов.

6.4 Приёмка системы

Во время финальной приемки заказчик должен:

Подтвердить выполнение всех предъявленных требований.

Принять систему к эксплуатации.

Убедиться в полной работоспособности игры и отсутствии критических багов.