# СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 3](#_heading=h.2jxsxqh)

[ВВЕДЕНИЕ 4](#_heading=h.3j2qqm3)

[1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 6](#_heading=h.1y810tw)

[1.1 Описание программируемой системы 6](#_heading=h.4i7ojhp)

[1.2 Обзор существующих решений 9](#_heading=h.2xcytpi)

[1.3 Требования к программируемой системе 11](#_heading=h.b07d0lefm6li)

[2 ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 12](#_heading=h.3whwml4)

[2.1 Диаграмма состояний системы 12](#_heading=h.2bn6wsx)

[2.2 Диаграмма классов системы 13](#_heading=h.qsh70q)

[2.3 Диаграмма последовательности системы 14](#_heading=h.1pxezwc)

[3 ПРАКТИЧЕСКАЯ ЧАСТЬ 15](#_heading=h.49x2ik5)

[3.1 Реализация требований к системе 15](#_heading=h.2p2csry)

[3.2 Функциональное тестирование программного продукта 17](#_heading=h.147n2zr)

[3.3 Инструкция по эксплуатации 22](#_heading=h.3o7alnk)

[ЗАКЛЮЧЕНИЕ 24](#_heading=h.23ckvvd)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 25](#_heading=h.ihv636)

[ПРИЛОЖЕНИЯ 27](#_heading=h.32hioqz)

[Приложение А – Исходный код программы 28](#_heading=h.nf3j29nmf0vz)

# ВВЕДЕНИЕ

Современный рынок разработки видеоигр требует от разработчиков не только мастерства в своей области, но и понимания смежных технологий. Создание современной игры-платформера в соответствии с поставленной задачей и темой позволяет приобрести практический опыт разработки современных видеоигр и взглянуть на классические задачи разработки игр под новым углом.

**Цель работы:** Создание игры-платформера для ПК на языке C++.

**Задачи работы:**

1. Описание концепции игры-платформера;
2. Изучение существующих игр-платформеров;
3. Составление требований к игре;
4. Разработка дизайн-документа игры;
5. Реализация игры в соответствии с требованиями;
6. Проведение тестирования игры;
7. Составление отчета по работе;
8. Презентация отчета и защита его.

**Объектом данного исследования** является Разработка игр-платформеров.

**Предметов исследования** в данной работе является Разработка игры-платформера для ПК на языке C++.

**В качестве основных методов исследования** применены анализ, синтез, сравнение и моделирование. Практическая реализация поставленной задачи соответствует основным парадигмам ООП, применяемым для разработки игр.

**Информационной базой** исследования являются открытые источники, а также материалы курса «Технологии индустриального программирования», доступные через систему дистанционного обучения РТУ МИРЭА.

В данном отчете будет представлен процесс разработки программного продукта, технологическое проектирование и описание системы. а также непосредственно результаты разработки.

# 1 Обзор существующих решений

Термином компьютерная игра обозначается компьютерная программа, которая служит для организации игрового процесса, связи с партнёром по игре, или сама выступающая в качестве партнёра [1].

Компьютерные игры классифицируют по нескольким основным признакам:

* жанр;
* количество игроков;
* визуальное представление;
* платформа.

Жанр игры определяется целью и основной механикой игры.

По количеству игроков игры разделяются на два вида:

* однопользовательские;
* многопользовательские.

По визуальному представлению компьютерные игры можно разделить на следующие виды:

* текстовые – минимальное графическое представление, общение с игроком проходит с помощью текста;
* 2D – все элементы отрисованы в виде двумерной графики (спрайтов);
* 3D – все элементы отрисованы в виде трехмерной графики (3D-модели).

По типу платформы:

* персональные компьютеры;
* игровые приставки/консоли;
* мобильные телефоны.

Общий алгоритм разработки компьютерной игры включает в себя 3 больших этапах [2]:

1. Проектирование.
2. Разработка.
3. Издание и поддержка.

На этапе проектирования определяются цель игры и средства ее разработки. При определении цели выделяются идея, жанр и сеттинг игры. Идея – это то, что будет побуждать игрока играть в создаваемую игру, и она очень тесно связана с жанром. Определив основные идеи игры, жанр будет подобран практически сразу. Определившись с жанром и идеей игры, следующим шагом будет выбор сеттинга. Сеттинг – это среда, в которой будет происходить основное действие игры. Он определяет место, время и условия действия. К средствам разработки в первую очередь относят программный код и игровой движок. От их грамотного выбора зависит как скорость самой разработки, так и работоспособность самого продукта в дальнейшем. Программный код в первую очередь зависит от платформы, для которой будет создаваться компьютерная игра.

Игровой движок отвечает за низкоуровневое описание физики объектов, правил рендеринга графики и другие. При выборе игрового движка первым делом смотрят на его доступность и уже сделанный выбор языка программирования.

После выбора цели игры и средств разработки, начинается второй этап реализации проекта – разработка. Разработка самый крупный и долгий этап реализации проекта, он включает в себя большое количество шагов, без которых невозможно создать рабочий продукт.

Игровая механика основывается на цели игры, она определяет все объекты и правила, по которым игрок будет взаимодействовать с ними. Обычно параллельно с разработкой игровой механики идет написание сюжета игры. Сюжет играет немаловажную роль, он определяет то, насколько будет игроку интересно играть в вашу игру. Сюжет представляют в двух вариантах: литературный и режиссерский сценарии. Литературный сценарий описывает основные события и персонажей игры, которые участвуют в игре. Режиссерский же представляет собой подробное описание уровней игры, событий, которые на этих уровнях происходят. Также на данном этапе начинается ранняя проработка графической составляющей и дизайна игры. На основе сюжета и заранее оговоренного дизайна, создаются ранние концепт-арты, на основе которых впоследствии будет проработан основной вид игры и персонажей. После разработки сюжета и игровой механики начинается самая важная часть – разработка самой игры.

При разработке игровых уровней изначально создается его упрощенный план, на котором схематично изображен сам уровень, а также изображены предметы, с которыми будет впоследствии взаимодействовать игрок. Следом после этого создается первая версия уровня.

Вскоре после создания первых уровней идет сборка первого прототипа игры, который называют альфа-версией игры. Она необходима для того, чтобы разработчик мог провести тестирование (альфа-тестирование) основной механики игры, и проверить насколько она соответствует заявленным требованиям. Если альфа-версия игры успешно проходит тестирование, наступает следующий этап разработки – проработка механики и объектов игры. На данном этапе идет доработка уровней и механики игры, и начинают добавлять первые сюжетные события в игру, такие как видеоролики, сюжетные диалоги и кат-сцены. Так же исправляются первые ошибки и неисправности в коде игры, которые были обнаружены при тестировании альфа-версии игры. После этого наступает этап создания второго прототипа игры, или, как принято говорить, бета-версии. Бета-версия служит для того, что протестировать игру на неисправности, фактически бета-версия представляет собой практически готовую игру. В ней могут отсутствовать какие-нибудь незначительные элементы, которые не влияют на геймплей игры.

Если игра проходит бета-тестирование, она отправляется на окончательную доработку и исправление критических ошибок, после чего идет сборка финальной версии игры и следом наступает релиз игры. После релиза игры последующая её поддержка. Поддержка заключается в выпуске патчей (файлов исправлений ошибок в готовом продукте) [3].

**Ori and the Blind Forest[4]**:

Ori and the Blind Forest предлагает уникальный искусственный стиль и захватывающий сюжет, но при этом предоставляет вызов в плане сложности. Если вы цените великолепную визуальную и звуковую атмосферу, глубокий сюжет и готовы к вызову в игровом процессе, Ori and the Blind Forest станет отличным выбором.

**Mark of the Ninja [5]:**

Mark of the Ninja — это выдающаяся игра в жанре стелс-платформера, предлагающая увлекательный геймплей, хороший сюжет и стильное исполнение. Если вам нравится стелс, и вы цените хорошо продуманные уровни, эта игра может предложить вам уникальный и захватывающий опыт.

**Ещё один пример:**

**…**

## 

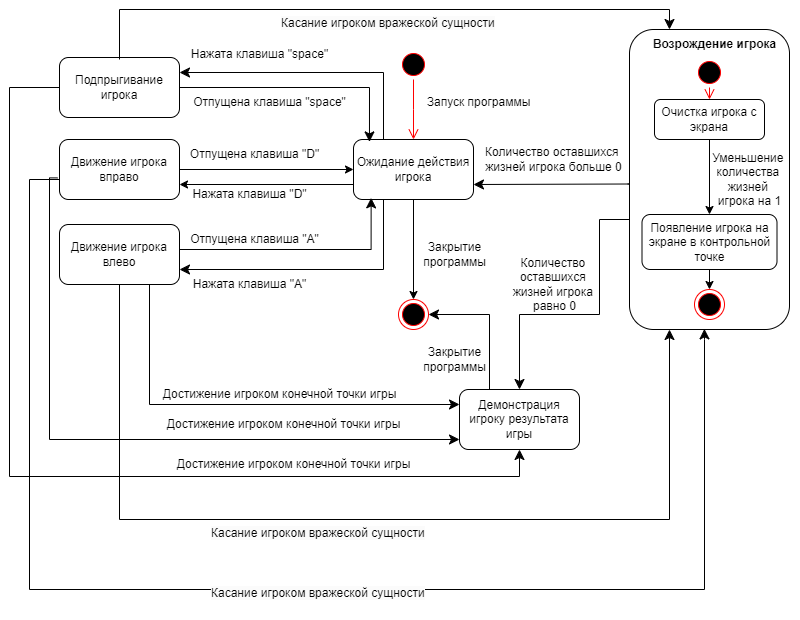
# 2. ОПИСАНИЕ ПРОГРАММИРУЕМОЙ СИСТЕМЫ

В таблице 2.1 представлены требования к программируемой системе.

*Таблица 2.1 – Требование к продукту*

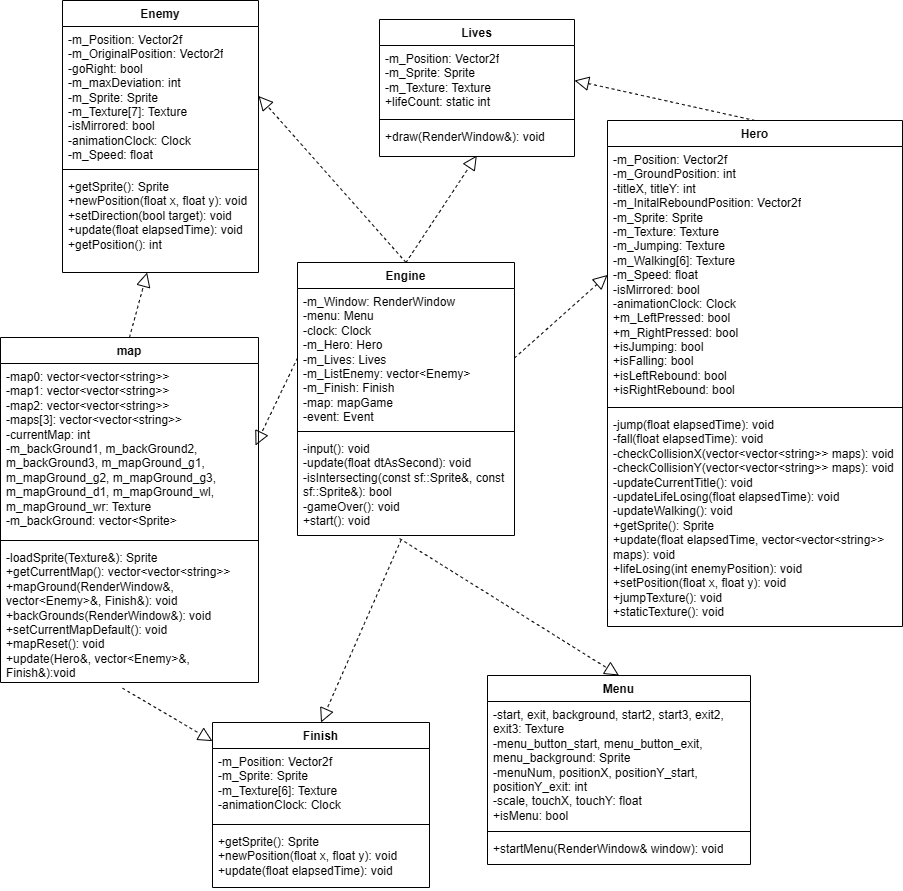
|  |  |  |
| --- | --- | --- |
| № | Требование | Значение |
| 1 | Язык программирования | C++ |
| 2 | Корректность работы | Приложение запускается и поддерживает стабильный цикл работы от момента старта до завершения |
| 3 | Применение принципов объектно-ориентированного программирования | При написании приложения, как минимум, были использованы классы в C++, объектный подход к проектированию системы, а также инкапсуляция |
| 4 | Интерфейс пользователя | Создан интерфейс пользователя, поддерживающий корректный пользовательский опыт и содержащий все необходимые пояснения к работе и эксплуатации |
| 5 | Инструкция по эксплуатации | Написана инструкция по эксплуатации, содержащая, в том числе, основные рекомендации по использованию и пояснения к возможным ошибкам в программе |
| 6 | Графическая библиотека | SFML |
| 7 | Управление вводом | Клавиатура ПК |
| 8 | Реализация анимационных эффектов | Интеграция спрайтовых анимаций для персонажей и интерактивных объектов для повышения визуальной аттрактивности и динамизма игрового процесса |
| 9 | Физический движок | Самописный |
| 10 | Система управления версиями | Использование системы управления версиями Git для отслеживания изменений в коде и ресурсах, обеспечивающее безопасность и возможность отката к предыдущим версиям |

Для описания работы системы были использованы диаграммы, описывающие работу системы в различных ее аспектах. Были спроектированы диаграмма состояний (Рисунок 2.1), диаграмма классов (Рисунок 2.2) и диаграмма последовательности (Рисунок 2.3).



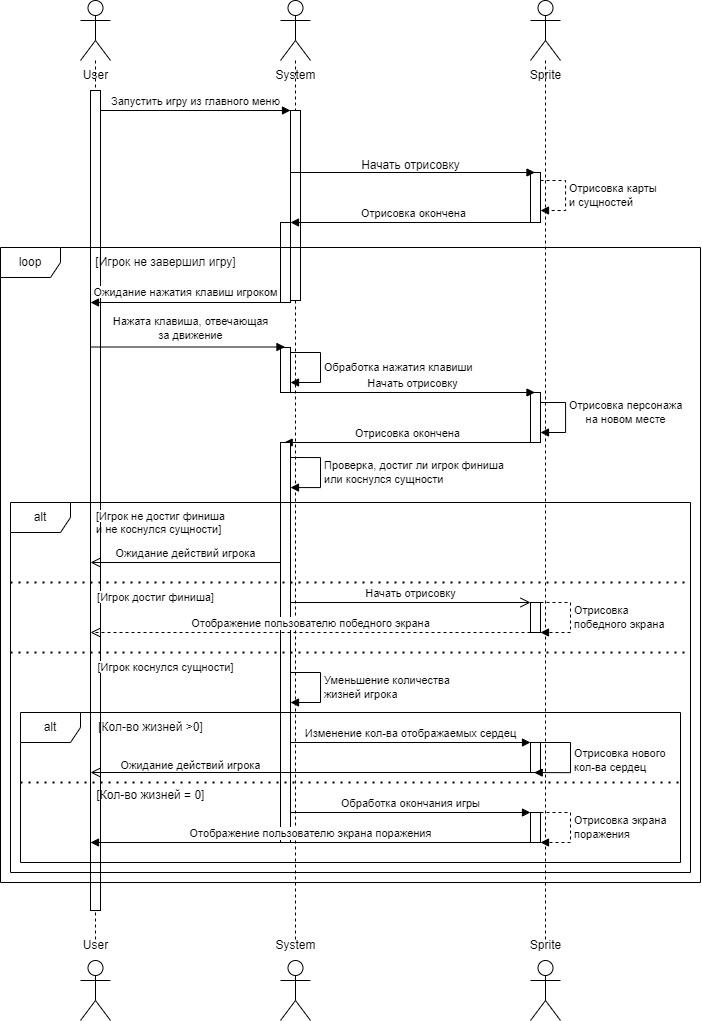
**Рисунок 2.1 – Диаграмма состояний системы**

ТУТ КОММЕНТАРИЙ ПО ДИАГРАММЕ СОСТОЯНИЙ



**Рисунок 2.2 - Диаграмма классов системы**

ТУТ КОММЕНТАРИЙ ПО ДИАГРАММЕ КЛАССОВ



**Рисунок 2.3 - Диаграмма последовательности системы**

ТУТ КОММЕНТАРИЙ ПО ДИАГРАММЕ ПОСЛЕДОВАТЕЛЬНОСТИ

# 3. результаты разработки программного продукта

**Реализация требований к системе**

1. **Язык программирования**

В рамках курсовой работы была разработана игра-платформер, полностью написанная на языке программирования C++. Увидеть реализацию данного требования можно обратившись к исходному коду программы (Приложение А).

1. **Корректность работы**

Обеспечена корректность функционирования приложения, выражающаяся в стабильности его запуска и поддержании непрерывного рабочего цикла вплоть до момента завершения сессии пользователя. Увидеть реализацию данного требования можно обратившись к скриншотам работы программы (Часть 3.2, Рисунки 3.1-3.6).

1. **Применение принципов объектно-ориентированного программирования**

В ходе разработки игры-платформера были применены ключевые принципы объектно-ориентированного программирования (ООП), включая инкапсуляцию, наследование и полиморфизм, что способствовало повышению модульности, расширяемости и поддерживаемости кодовой базы. Увидеть реализацию данного требования можно обратившись к исходному коду программы (Приложение А).

1. **Интерфейс пользователя**

Интерфейс пользователя был разработан с учетом удобства взаимодействия и интуитивной понятности, обеспечивая легкий доступ ко всем функциональным возможностям игры и комфортное управление игровым процессом. Увидеть реализацию данного требования можно обратившись к скриншотам работы программы (Часть 3.2, Рисунки 3.1-3.6).

1. **Инструкция по эксплуатации**

Для обеспечения понимания пользователем всех аспектов работы с игрой-платформером была подготовлена подробная инструкция по эксплуатации. Она содержит информацию о начале работы с игрой, описания интерфейса пользователя, правила и цели игры. Инструкция разработана с учетом принципов доступности и понятности для пользователей с различным уровнем опыта. Увидеть реализацию данного требования можно обратившись к специально отведенной части (3.3 Инструкция по эксплуатации) .

1. **Графическая библиотека**

Для реализации графического интерфейса и анимаций в игре использовалась графическая библиотека SFML (Simple and Fast Multimedia Library), что позволило обеспечить высокое качество визуализации и оптимизировать процесс разработки за счет удобства и гибкости предоставляемых инструментов. Увидеть реализацию данного требования можно обратившись к исходному коду программы (Приложение А).

1. **Управление вводом**

Управление вводом в приложении реализовано с использованием клавиатуры ПК, что обеспечивает точный и отзывчивый прием команд пользователя и способствует эффективному взаимодействию с игровым интерфейсом. Увидеть реализацию данного требования можно обратившись к исходному коду программы (Приложение А).

1. **Реализация анимационных эффектов**

Реализация анимационных эффектов была выполнена с применением современных техник графического дизайна, что позволило достичь плавности и реалистичности движений, а также улучшить визуальное восприятие и эстетическую привлекательность игры. Увидеть реализацию данного требования можно обратившись к скриншотам работы программы (Часть 3.2, Рисунки 3.1-3.6).

1. **Физический движок**

Для моделирования физики в игре был разработан собственный физический движок, что дало возможность точно настроить поведение объектов в соответствии с уникальными требованиями игры и обеспечило оптимальную производительность за счет специализированных оптимизаций. Увидеть реализацию данного требования можно обратившись к исходному коду программы (Приложение А).

1. **Система управления версиями**

Использование системы управления версиями GIT в процессе разработки игры позволило обеспечить эффективное сотрудничество в команде, удобное управление изменениями и сохранность кода, а также упростить процесс отладки и внедрения новых функций. Увидеть реализацию данного требования можно обратившись к проекту, расположенного на github.

**Функциональное тестирование программного продукта**

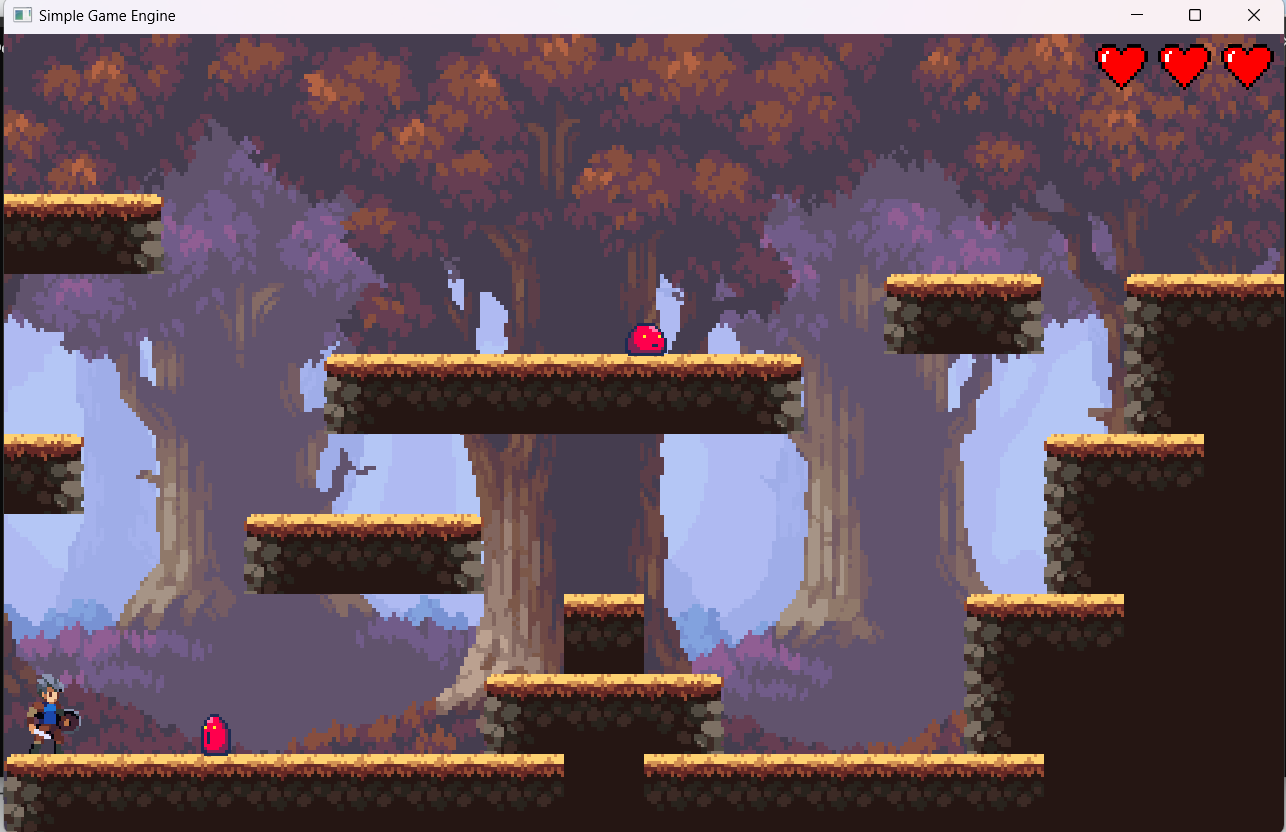
Работу игры на всех ее этапах координирует класс “Engine” (Приложение А, Листинги А.2-А.3), объект которого создается в главной функции (Приложение А, Листинг А.1).

Работа с программой начинается с вывода меню и выбора пользователем пункта меню (Рисунок 3.1). За вывод меню отвечает класс “Menu” (Приложение А, Листинги А.4-А.5).



**Рисунок 3.1 - Меню**

В случае выбора варианта “EXIT” программа завершает свою работу. В случае выбора варианта “PLAY” пользователь переходит непосредственно к игровому процессу (Рисунок 3.2).



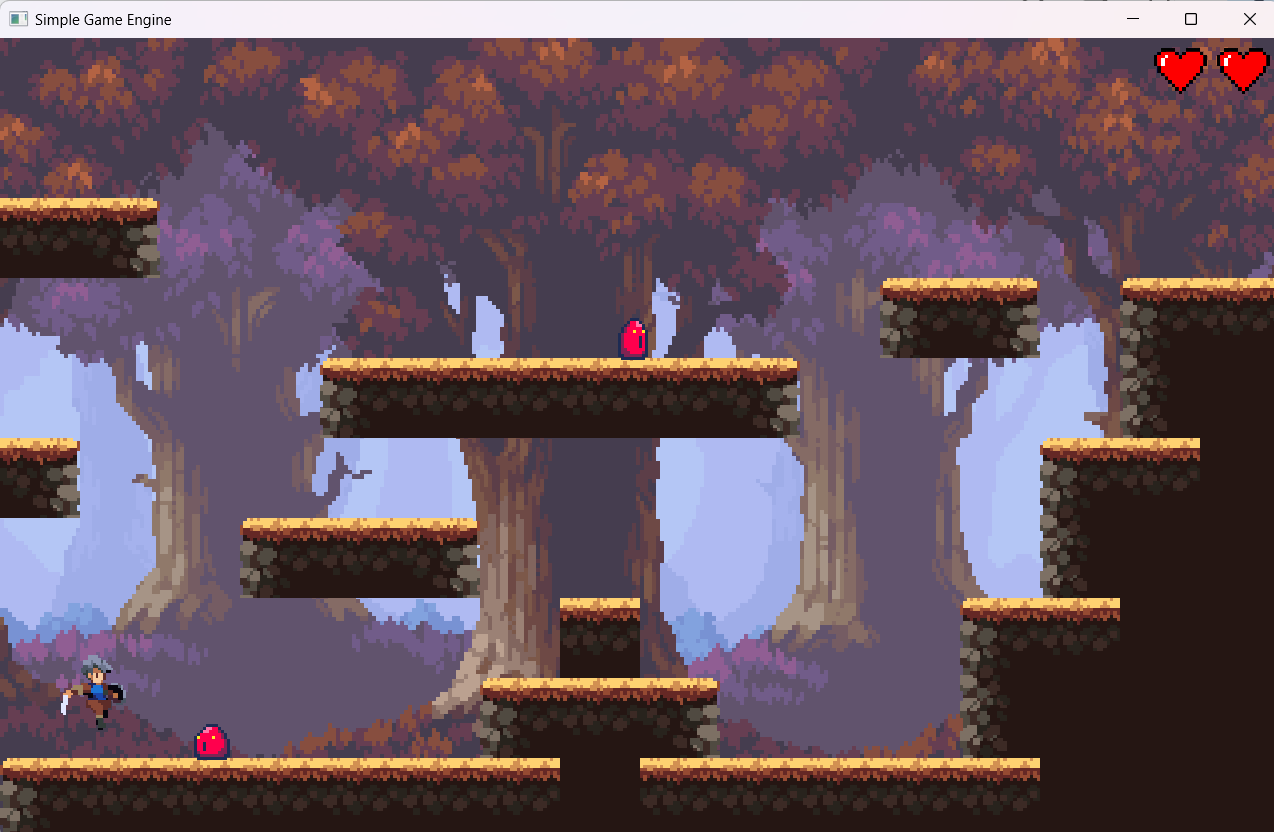
**Рисунок 3.2 – Начало игры**

Пользовательский персонаж появляется в левом нижнем углу карты и имеет возможность перемещаться влево (по нажатию клавиши “A”) и вправо (по нажатию клавиши “D”), а также обладает механикой прыжка (по нажатию клавиши “Space”). Обработка нажатий клавиш происходит в отдельной функции класса “Engine” (Приложение А, Листинги А.2-А.3), которая после обработки событий вызывает определенные функции в классе “Hero” (Приложение А, Листинги А.10-А.11), который непосредственно отвечает за движение персонажа.

Помимо спрайта персонажа, в игре присутствует спрайт врага (розовый слайм). Враг циклически перемещается влево и вправо на определенное расстояние от точки появления. Его механика прописана в отдельном классе “Enemy” (Приложение А, Листинги А.14-А.15).

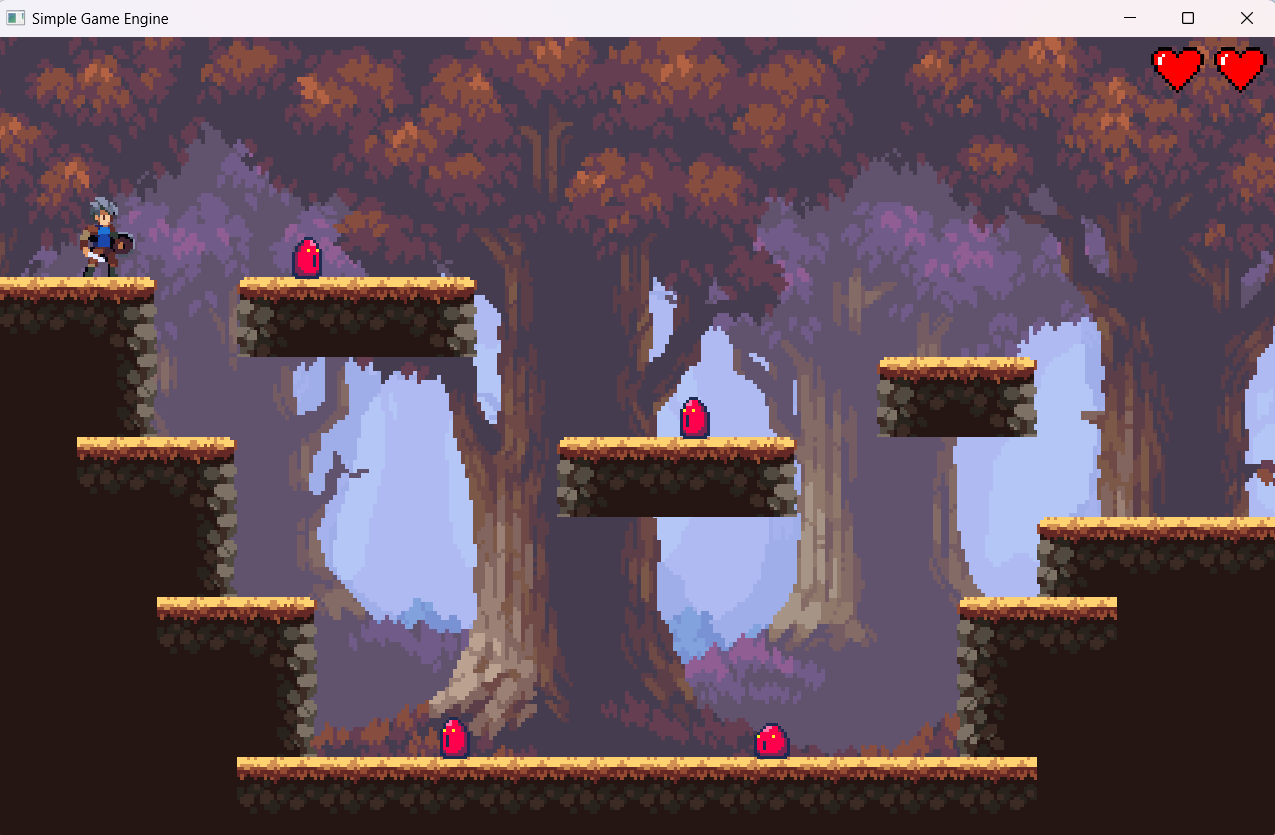
При столкновении спрайтов пользовательского персонажа и врага пользовательский персонаж теряет одну из трех своих жизней, за которые отвечает отдельный класс “Lives” (Приложение А, Листинги А.8-А.9). Если количество жизней оказывается равным нулю, то игра считается оконченной и игрок возвращается в меню (Рисунок 3.1).

Также после столкновения происходит отскок персонажа в противоположную от врага сторону (Рисунок 3.3). На протяжение отскока, в течение которого текстура игрового персонажа мерцает, получить урон еще раз невозможно.

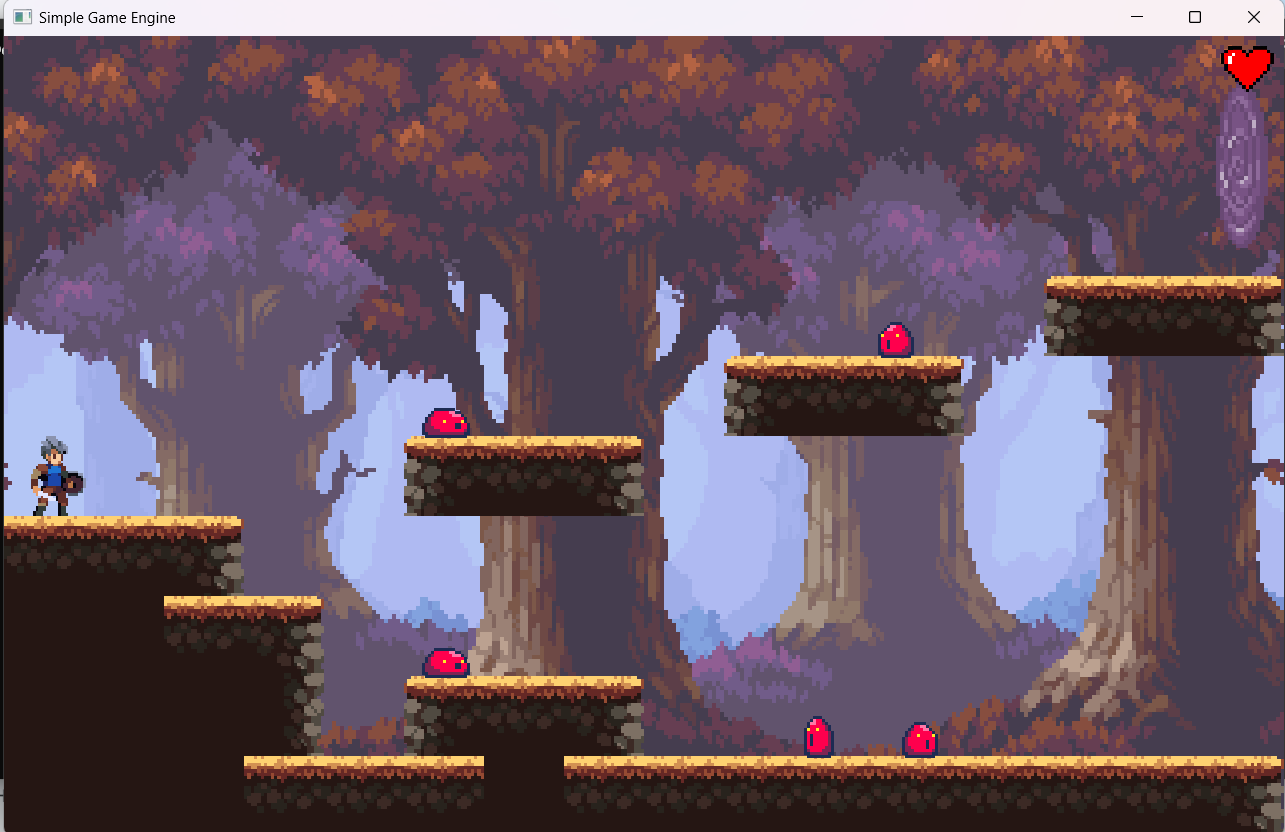


**Рисунок 3.3 – Отскок персонажа**

Вся игровая карта состоит из трех блоков, открывающихся игроку по мере прохождения (Рисунок 3.4 и Рисунок 3.5).

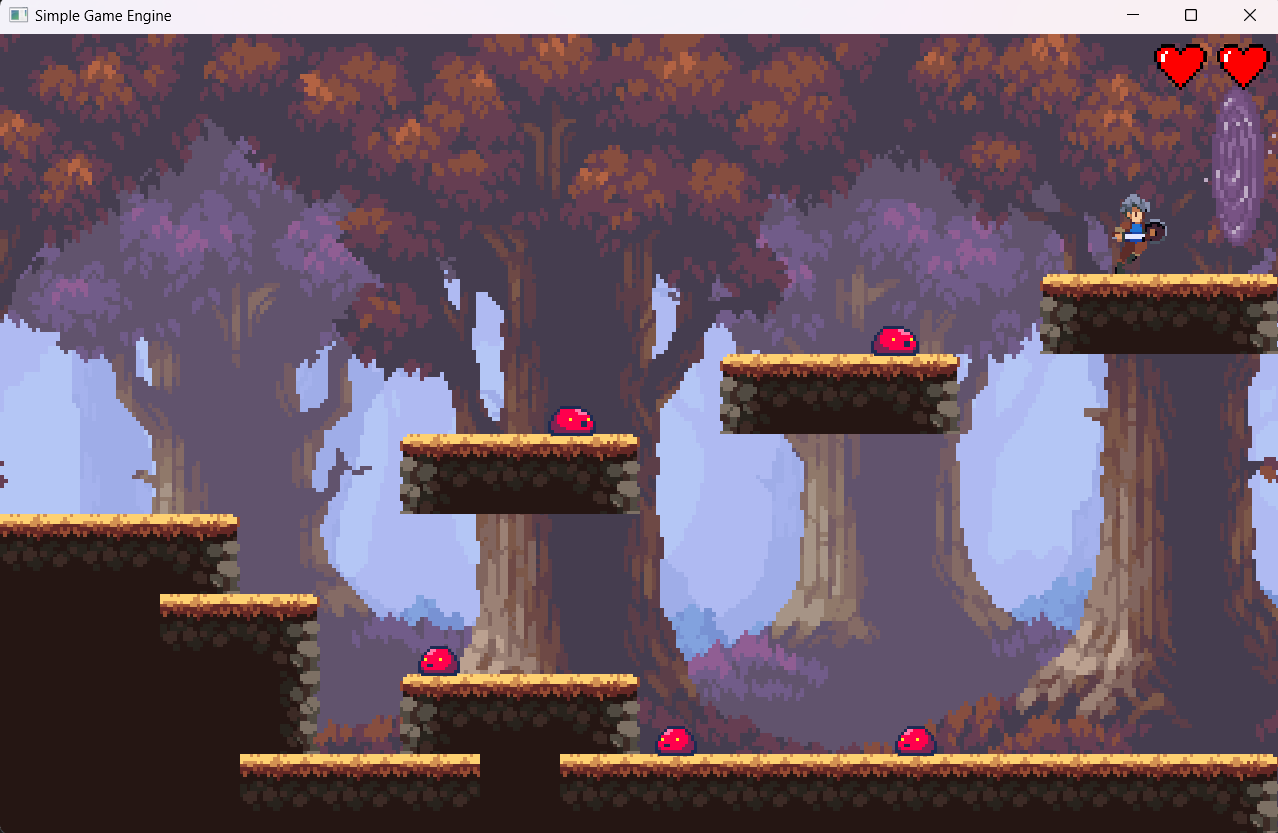


**Рисунок 3.4 – Переход в правую стену**

****

**Рисунок 3.5 – Выход из левой стены**

Каждый блок, кроме последнего, считается пройденным по достижении игроком крайней правой точки экрана на любой доступной высоте. Последний же блок считается пройденным по достижении определенного объекта на карте, текстура которого сделана в виде портала (Рисунок 3.6). Прохождение последнего блока карты считается окончанием игры, после чего пользователь возвращается обратно в меню (Рисунок 3.1).



**Рисунок 3.6 – Финиш игры**

**Инструкция по эксплуатации**

После запуска программы пользователь попадает в меню, где ему необходимо выбрать один из двух доступных вариантов. Для начала игры необходимо нажать левой кнопкой мыши на доступную на экране кнопку “PLAY”. Для выхода из программы необходимо нажать левой кнопкой мыши на доступную на экране кнопку “EXIT”.

При нажатии кнопки “PLAY” запускается непосредственно игровой процесс. Пользователю доступны следующие клавиши:

1. “A” - при нажатии на клавишу “A” игровой персонаж пользователя будет двигаться влево.
2. “D” - при нажатии на клавишу “D” игровой персонаж пользователя будет двигаться вправо.
3. “Space” - при нажатии на клавишу “Space” игровой персонаж пользователя будет подпрыгивать.

Конечной целью игрового процесса является достижение игровым персонажем конечной точки игры, расположенной на третьем блоке карты и сделанной в виде портала. Для прохождения блока карты пользователю необходимо двигаться по определенным текстурам, сделанным в виде земляных блоков, и избегать врагов, сделанных в виде розовых слаймов, до крайней правой точки экрана, после чего будет осуществлен переход персонажа на новый блок карты. При касании врага персонаж теряет одну из трех жизней, при потере всех трех жизней игра заканчивается.

После окончания непосредственно игрового процесса пользователь попадает обратно в начальное меню.

# ЗАКЛЮЧЕНИЕ

В результате выполнения курсовой работы была описана программируемая система, рассмотрены существующие решения-аналоги по теме, сформированы требования к системе, спроектированы диаграммы состояний, классов и последовательности самой системы. Сам программный продукты был разработан в соответствии с требованиями, протестирован, а также была написана инструкция по его эксплуатации.

Проект демонстрирует глубокое понимание объектно-ориентированного программирования, а также навыки работы с графической библиотекой SFML и системой управления версиями GIT. Разработанный продукт характеризуется стабильностью работы, качественной визуализацией и продуманным пользовательским интерфейсом.

В процессе разработки были решены сложные задачи, связанные с реализацией физического движка и анимационных эффектов, что позволило достичь реалистичности и плавности движений персонажей и объектов в игре. Использование GIT способствовало удобному управлению изменениями и сотрудничеству в команде.

Проект показал значительный потенциал для дальнейшего развития и улучшения, включая добавление новых функций и уровней, что может стать основой для будущих исследований и разработок в области создания игр. Работа над курсовой работой также способствовала повышению профессиональных компетенций в области программирования и дизайна игр.

Полученный опыт и знания будут служить твердой основой для дальнейшего обучения и профессионального роста в области разработки программного обеспечения и игровой индустрии.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Компьютерная игра / [Электронный ресурс] // Википедия : [сайт]. — URL: https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F\_%D0%B8%D0%B3%D1%80%D0%B0 (дата обращения: 09.03.2024).
2. Как создаются видеоигры: процесс разработки игры / [Электронный ресурс] // : [сайт]. — URL: https://itanddigital.ru/videogame#:~:text=%D0%A0%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0%20%D0%B2%D0%B8%D0%B4%D0%B5%D0%BE%D0%B8%D0%B3%D1%80%20%D0%BE%D0%B1%D1%8B%D1%87%D0%BD%D0%BE%20%D0%B4%D0%B5%D0%BB%D0%B8%D1%82%D1%81%D1%8F%20%D0%BD%D0%B0,%2C%20%D0%BF%D1%80%D0%BE%D0%B4%D0%B0%D0%BA%D1%88%D0%BD%20%D0%B8%20%D0%BF%D0%BE%D1%81%D1%82%2D%D0%BF%D1%80%D0%BE%D0%B4%D0%B0%D0%BA%D1%88%D0%BD.&text=%D0%97%D0%B4%D0%B5%D1%81%D1%8C%20%D0%BD%D0%B0%D1%87%D0%B8%D0%BD%D0%B0%D0%B5%D1%82%D1%81%D1%8F%20%D0%BA%D0%B0%D0%B6%D0%B4%D1%8B%D0%B9%20%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82. (дата обращения: 09.03.2024).
3. Семь этапов создания игры: от концепта до релиза / [Электронный ресурс] // Habr : [сайт]. — URL: https://habr.com/ru/companies/miip/articles/308286/ (дата обращения: 09.03.2024).
4. Ori and the Blind Forest / [Электронный ресурс] // Википедия : [сайт]. — URL: https://ru.wikipedia.org/wiki/Ori\_and\_the\_Blind\_Forest (дата обращения: 09.03.2024).
5. Mark of the Ninja / [Электронный ресурс] // Википедия : [сайт]. — URL: https://ru.wikipedia.org/wiki/Mark\_of\_the\_Ninja (дата обращения: 09.03.2024)

# ПРИЛОЖЕНИЯ

Приложение А – Исходный код программы

## Приложение А – Исходный код программы