**Титульный лист**

**СОДЕРЖАНИЕ**

**ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ**

В настоящем курсовом проекте применяют следующие сокращения и обозначения:

ОС – операционная система

ПК – персональный компьютер

ПО – программное обеспечение

**ВВЕДЕНИЕ**

Актуальность курсового проекта заключается в том, что игры-стратегии помогают развить навыки аналитического мышления в режиме реального времени, посредством контроля ресурсов. Данная разновидность игр помогает интересно и с некоторой пользой провести время. Продумывание вариантов использования ресурсов позволяет развить способности к математическим расчётам.

Уже существуют проекты в жанре Tower defense, например BloonsTD. В сравнении с BloonsTD данный проект будет иметь пиксельный визуальный стиль, а тематика игры будет основана на фантастическом мире, наполненном магическики существами. Также башни будут иметь совершенно другой визуал и механики.

Целью курсового проектирования является разработка игры стратегии разновидности защита башни, в которой будет реализована покупка и размещение башен с целью уничтожения врагов. Процесс игры будет проходить в режиме выживания, где деньги будут накапливаться в зависимости от количества убитых врагов. После игры у пользователя будет возможность посмотреть количество побеждённых врагов, количество заработанных монет и потраченное время. В игре также должна быть реализована возможность менять следующие настройки: полноэкранный режим или в окне, яркость и контрастность.

Для достижения поставленной цели требуется решить следующие задачи:

* выбрать среду разработки и язык программирования для создания программного продукта,
* спроектировать интерфейс игры,
* написать код игры,
* выполнить тестирование и отладку готовой игры и внести корректировки на случай найденных неисправностей.

**1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ ДЛЯ РАЗРАБОТКИ ИГРЫ-СТРАТЕГИИ «МАГИЧЕСКИЕ БАШНИ»**

**1.1 Назначение и область применения**

Разрабатываемое программное обеспечение является игрой-стратегией «Магические Башни». Данная игра предназначена для симуляции тактик в режиме реального времени.

**1.2 Постановка задачи**

Требуется спроектировать и разработать игру-стратегию «Магические башни», которая должна содержать следующие функции:

* во время игры: возможность купить и разместить башню,
* настройка режима окна, яркости и контрастности,
* игровое меню с переходом к настройкам и главному меню,
* ознакомление с результатом игры,
* ознакомление с руководством пользователя.

**1.3 Описание алгоритма функционирования приложения**

Игра-стратегия «Магические башни» запускается двойным щелчком.

После запуска перед пользователем отображается главная сцена, на которой расположены название игры и кнопки: начать игру, продолжить игру, настройки~~,~~ помощь, выйти. При переходе на сцену настроек пользователь может изменить настройки игры такие как: режим окна, яркость и контрастность.

Для начала новой игры необходимо нажать на кнопку новая игра. Игру можно поставить на паузу, вызвав игровое меню с кнопками: главное меню, настройки, выйти, вернуться.

**1.4 Обоснование выбора ПО**

В ходе проектирования и разработки игры-стратегии «Магические башни» необходимо выбрать техническое и программное обеспечение для дальнейшей работы.

Для разработки кода игры-стратегии используется открытый кроссплатформенный 2D и 3D игровой движок Godot Engine.

* наличие встроенных средств компиляции и отладки,
* поддержка множества языков при разработке,
* единая среда разработки независимо от языка программирования,
* бесплатное приложение визуальная среда разработки,
* кроссплатформенность.

Для разработки кода программы используется плотно интегрированный в движок Godot язык программирования GDScript, который имеет ряд преимуществ:

* оптимизированность и плотная интеграция со средой разработки,
* межплатформенная поддержка,
* синтаксис похожий на Python.

Для создания элементов дизайна приложения используется программа aseprite, имеющий ряд преимуществ:

* простой интерфейс,
* ориентирован на пиксельную графику,
* большое количество профессиональных инструментов.

**2 Проектирование и разработка игры-стратегии «Магические башни»**

**2.1 Проектирование прототипа игры-стратегии «Магические башни»**

Для наглядного представления интерфейса программы спроектирован прототип. Прототип – это упрощённая модель интерфейсного приложения, которая отображает приблизительный вид создаваемого приложения. По стадиям готовности прототипы условно делят на три этапа:

* концептуальные,
* интерактивные,
* анимированные.

Для создания прототипа выбрана концептуальная модель. Концептуальный прототип представляет собой схематичное изображение будущих экранов.

В ходе проектирования разработано три прототипа сцен игры-стратегии «Магические башни»:

* прототип главной сцены, на котором показано название игры и область под кнопки меню,
* прототип сцены настроек, на котором показана область с настройками и кнопка перехода в главное меню.
* прототип сцены игры, на котором показана область под информацию игры, расположение кнопки старта игры, паузы и общая область игры.

Прототипы спроектированных сцен представлены в приложении А.

**2.2 Разработка блок-схемы**

В ходе проектирования была создана блок-схема, отображающая работу скрипта, отвечающего за постановку башен. Блок-схема представлена на рисунке 9.

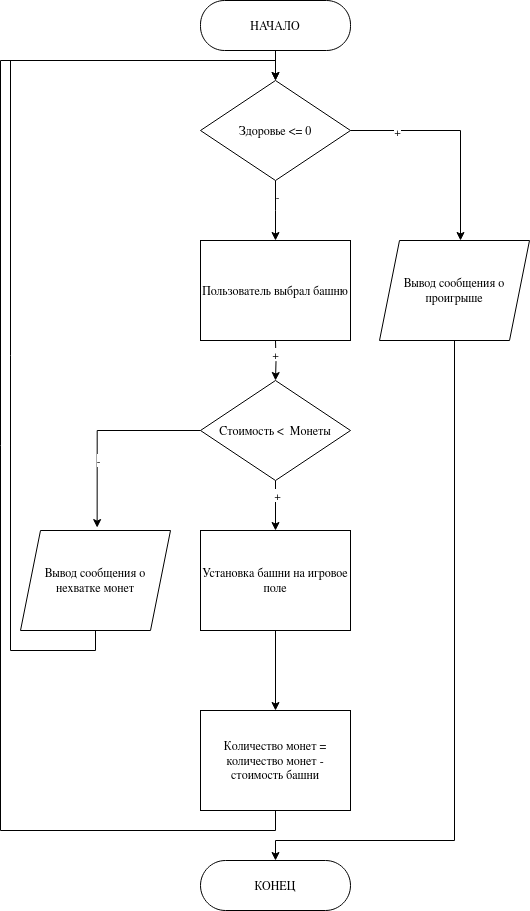


Рисунок 9 – Блок-схема игры-стратегии «Магические башни».

Также при проектировании была предусмотрена возможность сохранять прогресс и при повторном запуске предоставлять пользователю возможность продолжить ранее начатую игру.

Рисунок 10 – Блок-схема игры-стратегии «Магические башни».

**2.2 Разработка игры-стратегии «Магические башни»**

В процессе разработки игры-стратегии «Магические башни» создано несколько сцен, каждая из которых служит для выполнения функций, определённых в процессе проектирования. Главной сценой приложения является сцена «menu», которая открывается при запуске программы (рисунок 9). На сцене показаны кнопки для перехода на другие сцены.

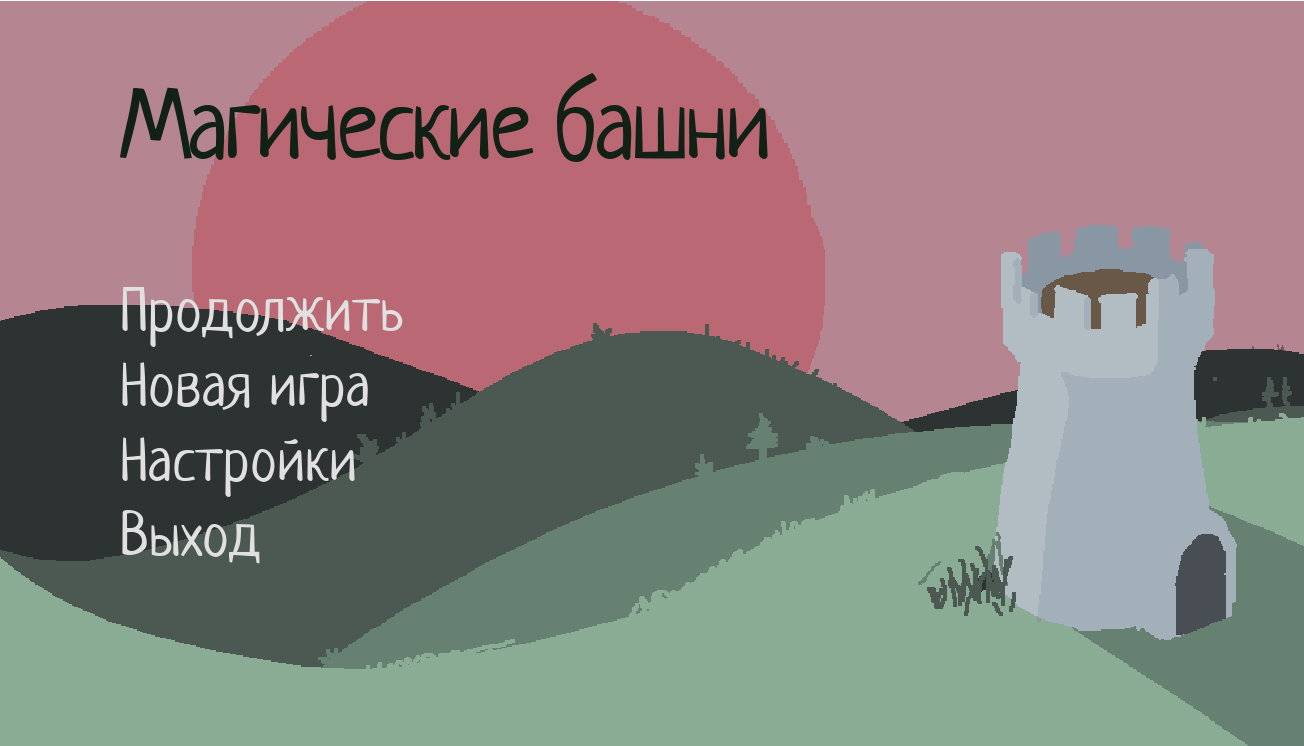


Рисунок 11 – Вид главной сцены игры-стратегии «Магические башни».

При нажатии на кнопку «Настройки» происходит переход на сцену настроек, показанную на рисунке 11. На данной сцене можно изменить режим окна, яркость и контрастность. Код метода save\_settings(), сохраняющего настройки в файл представлен на рисунке 10.

Листинг 1 – Игра-стратегия «Магические башни». Код метода save\_settings()

|  |
| --- |
| func save\_settings():  var f = File.new() //создание файла  f.open(settings\_file, File.WRITE) //открытие файла  f.store\_var(enable\_fullscreen) //настройка режима экрана  f.store\_var(brightness) //настройка яркости  f.store\_var(contrast) //настройка контрастности  f.close() //закрытие файла |

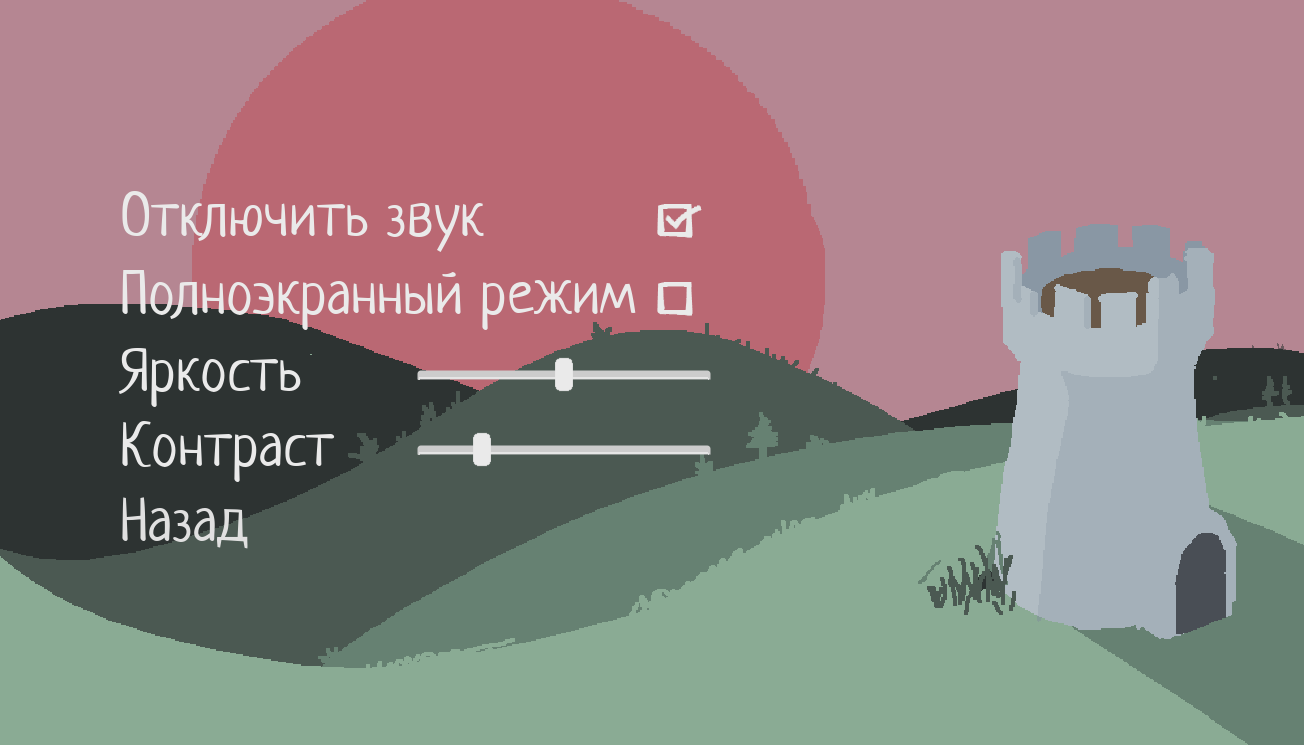


Рисунок 12 – Вид сцены настроек игры-стратегии «Магические башни».

При нажатии на кнопку «Новая игра» в главном меню происходит переход на сцену игры, показанную на рисунке 13. На данной сцене происходит игровой процесс, состоящий из покупки и постройки башен и уничтожения врагов с целью выживания. Код метода create(tower), создающий башню, представлен на рисунке 14.

Листинг 2 – Игра-стратегия «Магические башни». Код метода create(tower)

|  |
| --- |
| func create(tower):  t = tower.instance()//создание объекта башни  t.translation = pos//установка позиции  add\_child(t)//добавляет потомка на сцену  t.set\_owner(self)//задаёт владельца  t.set\_physics\_process(false)//отключает физические процессы в башне |



Рисунок 13 – Вид игровой сцены игры-стратегии «Магические башни».

В ходе разработки игры был создан класс Tower, содержащий свойства, отвечающие за тип пуль, выпускаемый башней, радиус атаки, скорость атаки, урон от атаки и стоимость башни. Также в класс были добавлены методы attack() выбирающий цель и атакующий её, метод \_on\_detectionArea\_area\_entered проверяющий столкновение пули с объектом, метод \_on\_detectionArea\_area\_exited фиксирующий выход пули из области объекта и метод \_on\_shot\_timer\_timeout отвечающий за частоту выстрела пуль. Листинг данного класса представлен в приложении Б.

Также был разработан класс Player содержащий количество здоровья игрока, количество доступного золота для покупки башен. Внутри класса Player содержатся методы get\_damage, срабатывающий при получении урона, метод add\_gold для увеличения количества золота при уничтожении врагов.

Листинг 3 – Игра-стратегия «Магические башни». Код класса Player.

|  |
| --- |
| signal health\_changed  signal gold\_changed  export(int) var health = 100 setget set\_health  export(int) var gold = 400 setget set\_gold  func get\_damage(amount):  health -= amount  emit\_signal("health\_changed")  func add\_gold(amount):  gold\_counter += amount  gold += amount  emit\_signal("gold\_changed") |

**2.3 Создание инсталляционного пакета для игры-стратегии «Магические башни»**

Для удобства установки и последующего использования программы создан инсталляционный пакет.

Инсталляционный пакет – это специально подготовленный запускаемый файл, предназначенный для установки программы на клиентские устройства.

Создание инсталляционного пакета проводилось

Создание инсталляционного пакета проводилось с помощью утилиты «Inno Setup Compiler».

Для создания инсталляционного пакета необходимо:

* запустить утилиту,
* выбрать способ создания нового файла установщика «create a new script file using the script wizard»,
* следуя указаниям мастера создания, указать имя установочного пакета, версию, издателя и веб-сайт издателя,
* на следующем шаге задать название папки, которая будет создана после установки и разрешение пользователю на редактирование этой папки,
* на следующем шаге выбрать исполняемый файл приложения и указать папки и файлы, которые будут установлены вместе с исполняемым файлом,
* далее задать имя приложения в стартовом меню и указать разрешения,
* далее добавить файл лицензии, информацию, показываемую перед и после установки,
* указать языки, с которыми будет работать установщик,
* далее задать папку, в которую будет создан установщик, имя установщика, файл иконки установщика и пароль для запуска установки, если нужен,
* после выполнения всех шагов мастера возможно сразу выполнить компиляцию и создание установочного файла. Либо просмотреть и отредактировать код,
* после компиляции будет создан инсталляционный пакет.

**2.4 Спецификация игры-стратегии «Магические башни»**

Составлена спецификация программы, в которой описываются файлы, задействованные в работе игры-стратегии «Магические башни», а также минимальные системные требования программы.

Для функционирования программы достаточны следующие программные и технические средства:

* операционная система Windows 7 и выше,
* оперативная память минимум 100мб,
* процессор с частотой не менее 1ГГц,
* объем свободного места на жёстком диске не менее 50Мб
* манипуляторы: клавиатура и мышь.

Для установки и последующего использования программы используется файл установщик с названием «Магические башни». После распаковки файлов на ПК установится исполняемый файл. Помимо исполняемого файла в конфигурацию пакета включены файлы, описанные в таблице 1.

Таблица 1 – Файлы игры-стратегии «Магические башни»

|  |  |  |
| --- | --- | --- |
| Обозначение | Наименование | Применение |
| Magical\_towers.exe | Исполняемый файл | Предназначен для запуска и работы игры |
| Magical\_towers.pck | Пакеты ресурсов | Файл для удобной модификации игры разработчиками |
| unins000.exe | Файл деинсталляции | Удаляет все файлы игры |

**2.5 Руководство пользователя для игры-стратегии «Магические башни»**

Для установки игры-стратегии «Магические башни» необходимо открыть установочный пакет, имеющий название «Магические башни». По окончанию процесса установки необходимо запустить приложение «Magical\_towers.exe».

При запуске программы открывается главная сцена (рисунок 15), на которой расположены кнопки для навигации. При нажатии кнопки «Начать игру» начинается игра. При нажатии на кнопку «Настройки» реализуется переход к сцене настроек. При нажатии на кнопку «Помощь» открывается панель с руководством пользователя. При нажатии на кнопку «Выйти» программа закрывается.

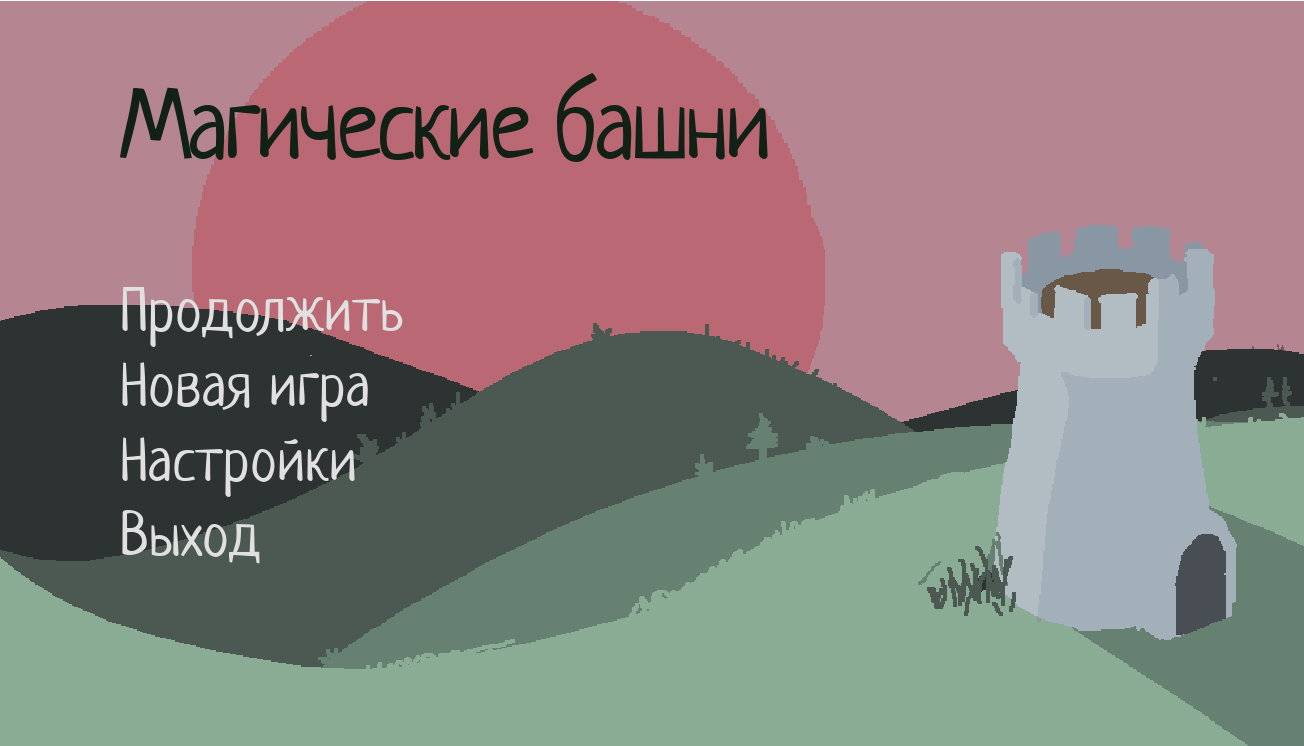


Рисунок 14 – Вид главной сцены игры-стратегии «Магические башни».

На сцене настроек, находятся элементы, показанные на рисунке 16. При их изменении меняются соответствующие настройки.

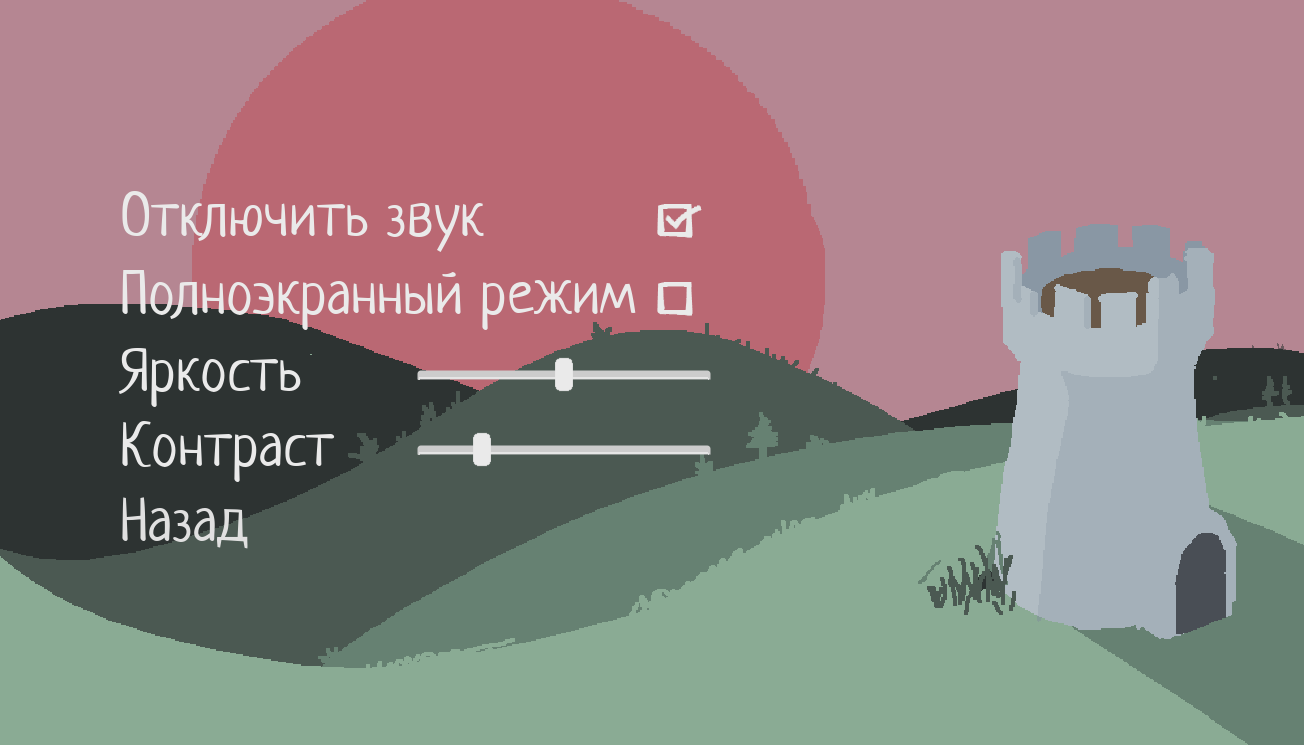


Рисунок 15 – Вид окна настроек игры-стратегии «Магические башни».

На сцене игра (рисунок 17), происходит сам игровой процесс. При нажатии на иконку башни, башня появляется на игровом поле, пользователь может её разместить по своему усмотрению. В левом верхнем углу отображается количество здоровья и денег. Справа находятся кнопки паузы, запуска волны и автоматического воспроизведения волн.



Рисунок 16 – Вид игровой сцены игры-стратегии «Магические башни».

В игровом меню есть кнопки «Главное меню», «Настройки», «Выйти», «Продолжить». При нажатии на кнопку «Главное меню» идёт возвращение к главной сцене, на кнопку «Настройки» открытие панели с элементами, показанными на рисунке 16, на кнопку «Выйти» выход из игры, на кнопку «Вернуться» закрытие панели меню и возобновление игры.

**3 ТЕСТИРОВАНИЕ ИГРЫ-СТРАТЕГИИ «МАГИЧЕСКИЕ БАШНИ»**

Для исключения ошибок и проверки соответствия программы поставленным требованиям необходимо провести тестирование. Тестирование программного обеспечения – проверка соответствия реальных и ожидаемых результатов поведения программы, проводимая на конечном наборе тестов, выполненном определённым образом.

В качестве метода тестирования выбран метод «чёрного ящика» [3], который предполагает тестирование со стороны пользователя программы.

Результат полного тестирования игры-стратегии «Магические башни» представлен в таблицах 2-4 по каждому модулю. В результате тестирования ошибки не выявлены, программа соответствует требованиям и функционирует корректно.

Таблица 2 – Результаты тестирования сцены «menu»

|  |  |  |
| --- | --- | --- |
| Исходные данные | Ожидаемый результат | Реальный результат |
| Нажатие на кнопку «Новая игра» | Перейдёт к сцене игры | Перешёл к сцене игры |
| Нажатие на кнопку «Настройки» | Перейдёт к сцене настроек | Перешёл к сцене настроек |
| Нажатие на кнопку «Помощь» | Откроется руководство пользователя | Открылось руководство пользователя |
| Нажатие на кнопку «Выйти» | Программа закроется | Программа закрылась |

Таблица 3 Результаты тестирования сцены «game»

|  |  |  |
| --- | --- | --- |
| Исходные данные | Ожидаемый результат | Реальный результат |
| Нажатие на кнопку «Пауза» | Откроется панель меню | Открылась панель меню |
| Нажатие на кнопку на панели меню «Настройки» | Откроется панель настроек | Открылась панель настроек |
| Нажатие на кнопку на панели меню «Выход в меню» | Перейдёт на главную сцену | Перешёл на главную сцену |

Продолжение таблицы 3

|  |  |  |
| --- | --- | --- |
| Исходные данные | Ожидаемый результат | Реальный результат |
| Нажатие на кнопку «Продолжить» на панели меню | Панель меню закроется | Панель меню закрылась |
| Нажатие на иконку башни | Создаться новая башня | Новая башня создалась |

Таблица 4 Результаты тестирования сцены «settings»

|  |  |  |
| --- | --- | --- |
| Исходные данные | Ожидаемый результат | Реальный результат |
| Нажатие на флажок «Отключить звук» | Отключится звук | Звук отключился |
| Нажатие на флажок «В окне» | Изменится режим окна | Изменился режим окна |
| Нажатие на кнопку «Назад» | Возвращение на главную сцену | Возвращение на главную сцену звуков |

**ЗАКЛЮЧЕНИЕ**

В процессе курсового проектирования была достигнута цель, которой является разработка Магические башни». В игре была создана возможность просмотра рейтинга пользователей и изменение следующих настроек: полноэкранный режим или в окне и громкость звука. Была разработана уникальная игра, отличающаяся от других игр в жанре tower defense, отличающаяся визуальным стилем, тематикой и механиками.

Для достижения поставленной цели были решены следующие задачи:

* проанализированы требования, отсортировать их по выполнимым и невыполнимым, по необходимым и ненужным. В итоге собрать информацию воедино и составить примерный план взаимодействия игры с пользователем;
* выбрана среда разработки и язык программирования для создания программного продукта;
* спроектирован интерфейс игры;
* разработана игра-стратегия;
* выполнено тестирование и отладка готовой игры и внесены корректировки на случай найденных неисправностей.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Документация Godot. – Текст : электронный // docs.godotengine.org. – URL: https://docs.godotengine.org/ru/stable/index.html (дата обращения 10.01.2025).
2. Тестирование методом черного ящика. – Текст : электронный // habr.com. – 2019. – URL: https://habr.com/ru/post/462837/ (дата обращения: 10.01.2025).
3. Найстром, Р. Шаблоны игрового проектирования / Р. Найстром – Москва : Бомбора, 2021. – 432 с. URL: https://martalex.gitbooks.io/gameprogrammingpatterns/content/ (дата обращения 25.12.2024) Режим доступа: для зарегистрир. пользователей. – Текст : электронный.
4. Джесси Ш. Искусство Геймдизайна. - 3-е изд. - Москва: Альбина Паблишер, 2020. - 640 с.
5. Pixel Art Tutorial: How to Create a Character Sprite // lospec.com URL: https://lospec.com/pixel-art-tutorials/how-to-create-a-character-sprite-sheet-by-etosurvival (дата обращения: 01.01.2025).
6. Bradfield C. Godot Engine Game Development . - Birmingham: Packt Publishing, 2018. - 298 с.
7. Лутц М. Изучаем Python. - 5-е изд. - Москва: Диалектика, 2019. - 833 с.
8. Павловская Т.А. Трофимов В.В. Основы алгоритмизации и программирования. - 5-е изд. - Москва: Юрайт, 2019. - 137 с.
9. Пиксель-арт: от черновика до игрового ассета // habr.com URL: https://habr.com/ru/articles/444738/ (дата обращения: 12.09.2024).
10. Diagram Software and Flowchart Maker // diagrams.net URL: https://app.diagrams.net/ (дата обращения: 20.01.2025).

ПРИЛОЖЕНИЕ A

(справочное)

Прототипы сцен



Рисунок Б.1 – Игра-стратегия «Магические башни». Прототип главной сцены



Рисунок Б.2 – Игра-стратегия «Магические башни». Прототип сцены настроек

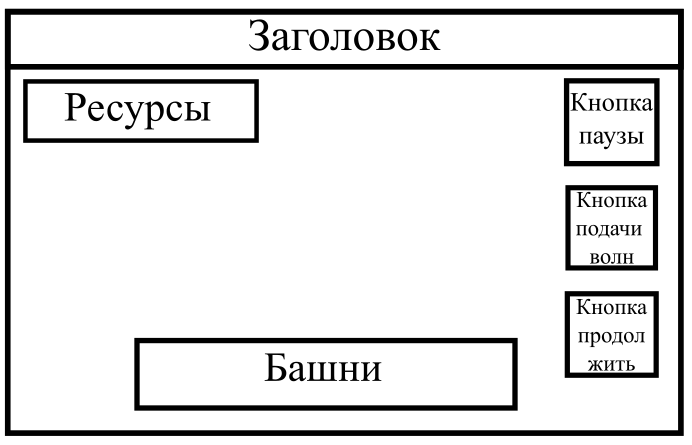


Рисунок Б.3 – Игра-стратегия «Магические башни». Прототип сцены игры

ПРИЛОЖЕНИЕ Б

(справочное)

Листинг класса Tower

Листинг 1 – Игра-стратегия «Магические башни». Код класса Tower.

|  |
| --- |
| class\_name Tower, "res://assets/towers/tower.png"  extends Spatial  export(PackedScene) var bullet  export(float) var attack\_radius = 5  export(float) var attack\_speed = 200  export(int) var attack\_damage = 20  export(int) var price = 0  var targets = []  var bullet\_load = true  onready var \_detection\_collision = get\_node("detection\_area/CollisionShape")  onready var \_building\_area = $building\_area  func \_ready():  get\_node("shot\_timer").wait\_time = 100/attack\_speed  var collision\_shape = \_detection\_collision.get\_shape()  collision\_shape.radius = attack\_radius  \_detection\_collision.set\_shape(collision\_shape)  func \_physics\_process(\_delta):  if !targets.empty() && is\_instance\_valid(bullet):  if bullet\_load == true:  attack()  bullet\_load = false  func attack():  var b = bullet.instance()  b.translation = to\_global($bullet\_pos.translation)  b.damage = self.attack\_damage  b.set\_target(targets.front())  b.look\_at\_from\_position(b.translation,  targets.front().get\_global\_transform().origin, Vector3.UP)  get\_parent().add\_child(b)  b.shoot()  func \_on\_detectionArea\_area\_entered(area):  if area.is\_in\_group("enemy"):  targets.append(area)  func \_on\_detectionArea\_area\_exited(area):  if area.is\_in\_group("enemy"):  targets.erase(area)  func \_on\_shot\_timer\_timeout():  bullet\_load = true  func is\_overlapping() -> bool:  return !\_building\_area.get\_overlapping\_areas().empty() || \  !\_building\_area.get\_overlapping\_bodies().empty()  """ setters getters """  func get\_price():  return price |

Сведения о самостоятельности выполнения работы

Работа выполнена мной самостоятельно.

Используемые в работе материалы и концепции из публикуемой литературы и других источников имеют ссылки на них.

Один печатный экземпляр работы и электронный вариант работы на цифровом носителе переданы мной на кафедру/отделение.

«\_\_\_» \_\_\_\_\_\_\_\_\_ 20\_\_\_г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_