

**ПРОЕКТНАЯ РАБОТА**

**Azamat adventure**

Выполнил: Конысбаев Санжар

Класс: 10F

Актобе, 2022

**Содержание**

[1. Анализ 3](#_Toc102716716)-7

[1.1 Природа проблемы 3](#_Toc102716717)

[Описание клиента и деятельности 3](#_Toc102716718)

[Используемые в настоящее время методы 3](#_Toc102716719)

[Происхождение данных 3](#_Toc102716720)

[1.2 Исследование и анализ 4](#_Toc102716721)-7

[Исследование 4](#_Toc102716722)

[Анализ](#_Toc102716728) 5-7

[План работы над проектом 7](#_Toc102716729)

[2. Проектирование 8](#_Toc102716730)-16

[2.1 Прототип решения 8](#_Toc102716731)-10

[2.2 Алгоритм работы программы 11](#_Toc102716732)-16

[2.3 Преимущества решения 16](#_Toc102716733)

[2.4 Ограничения решения 16](#_Toc102716734)

[3. Реализация 17](#_Toc102716735)-30

[3.1 Использованные структуры данных 17](#_Toc102716736)-18

[3.2 Входные и выходные данные 19](#_Toc102716737)

[3.3 Таблица идентификаторов 20](#_Toc102716738)-21

[3.4 Интерфейс программы 22](#_Toc102716739)-24

[3.5 Листинг программы 25](#_Toc102716740)-30

[4. Тестирование и отладка 31](#_Toc102716741)-32

[4.1 План тестирования 31](#_Toc102716742)

[5. Оценивание, документирование и внедрение 33](#_Toc102716743)

[5.2 Внедрение, оценивание клиента 33](#_Toc102716744)

# 1. Анализ

## 1.1 Природа проблемы

### Описание клиента и деятельности

Моим клиентом является Азамат Аманбай из 10F. За свою жизнь он попробовал множество довольно качественных игр, но в какой-то момент времени ему надоело следить за жизнью персонажа в игре, и он решил заказать игру с его участием. К его требованиям относятся наличие музыки, смешных персонажей и интересного дизайна.

Игра заключается в прохождении уровней, которые напрямую связаны с заказчиком, то есть уровнями будут являться кабинеты и комнаты Назарбаев Интеллектуальной школы. Каждый уровень имеет особенности и фичи, а также разный дизайн. Таким образом игра будет связана с реальной жизнью Азамата, которая полна увлекательных историй и происшествий.

### Используемые в настоящее время методы

Когда людям надоедает играть в видеоигры, они решают создавать их самим и не редко вставляют себя в главные роли. Но не каждому удается создать игру с первого раза. Таким образом они заказывают игры у мастеров.

### Происхождение данных

Ниже приведен примерный алгоритм курсов (Рис 1). Алгоритм состоит из нескольких этапов. Сначала пользователь вводит логин и пароль для входа в игру. Далее открывается главное меню, где вы можете нажать на кнопку настроек, закрытия игры, меню уровней, настроек и кнопки начала игры. После нажатии на кнопку меню появятся три уровня и пользователя будет выбор с какого уровня он может начать. После смерти появляется окно проигрыша, а после прохождения игры появляется победное окно.

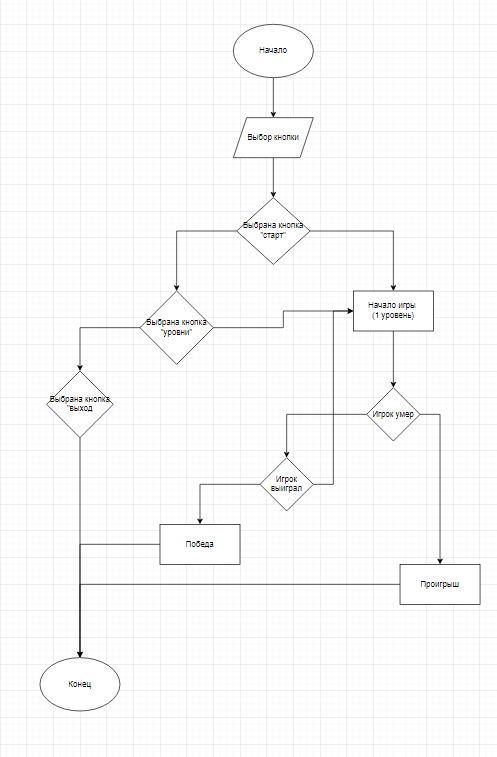


Рис 1. Алгоритм деятельности клиента

## 1.2 Исследование и анализ

### Исследование

При разработке проектов нужно было исследовать заинтересованность будущих клиентов, то есть узнать понравится ли идея и сюжет игры людям, а также путем голосования выбрать локации, врагов и персонажей.

#### 1. Хотели бы вы сыграть игру про вашу школу?

* Да!
* Нет(

#### 2. Хотели бы вы быть персонажем игры?

* Конечно
* НЕТ

#### 3. Выберите персонажа для игры

* Азамат
* Арсен
* Ержан

#### 4. Выберите боссов игры

* Шадияр Курашевич(учитель информатики)
* Альфия Зинетуловна(учитель математики)
* Владимир Иванович (учитель физкультуры)

#### 5. Выберите локации для игры

* Столовая
* Кабинет (предметный)
* Библиотека

### Анализ

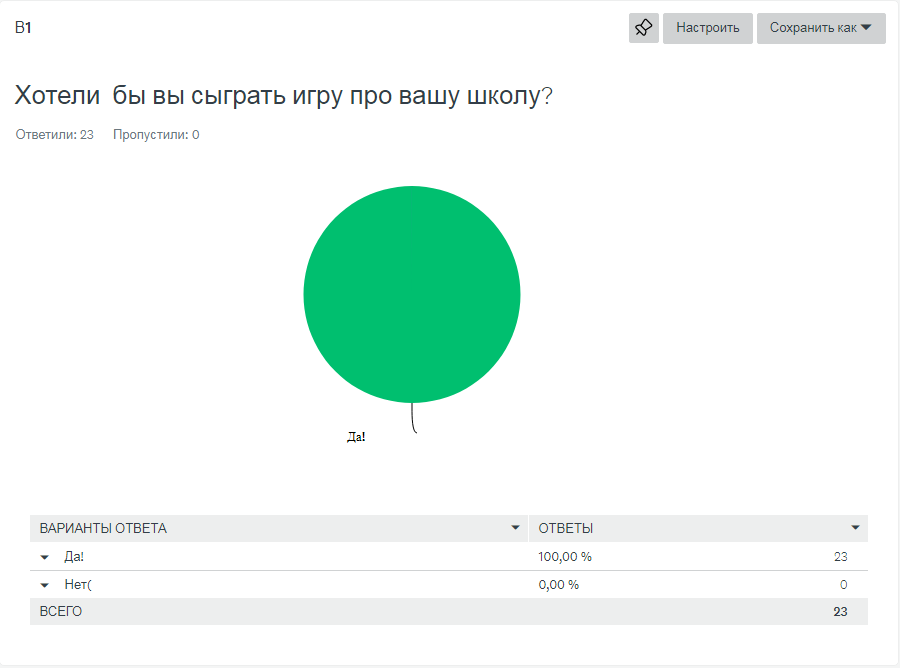


Рис. 2. Вопрос 1

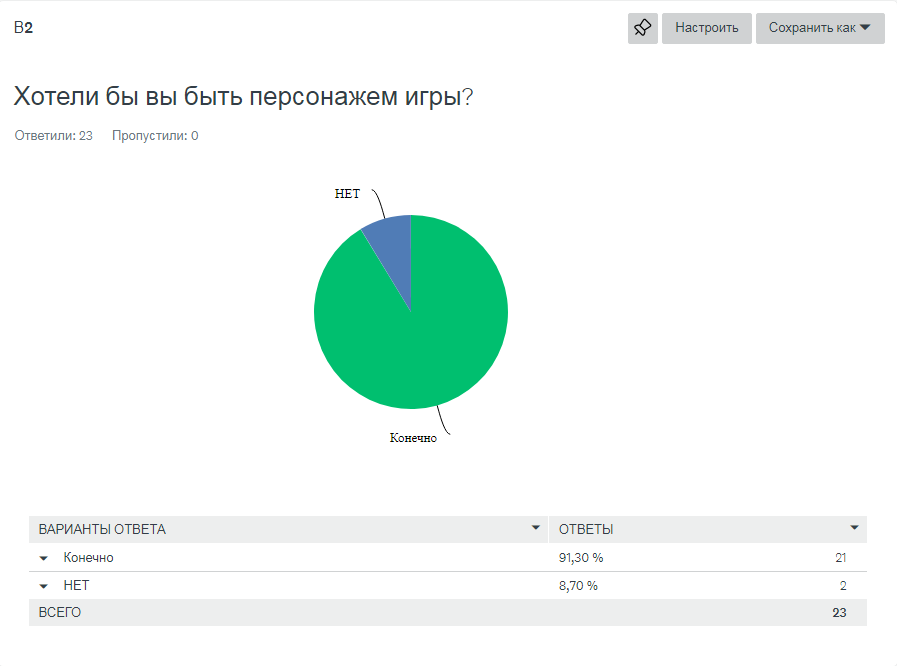


Рис. 3. Вопрос 2

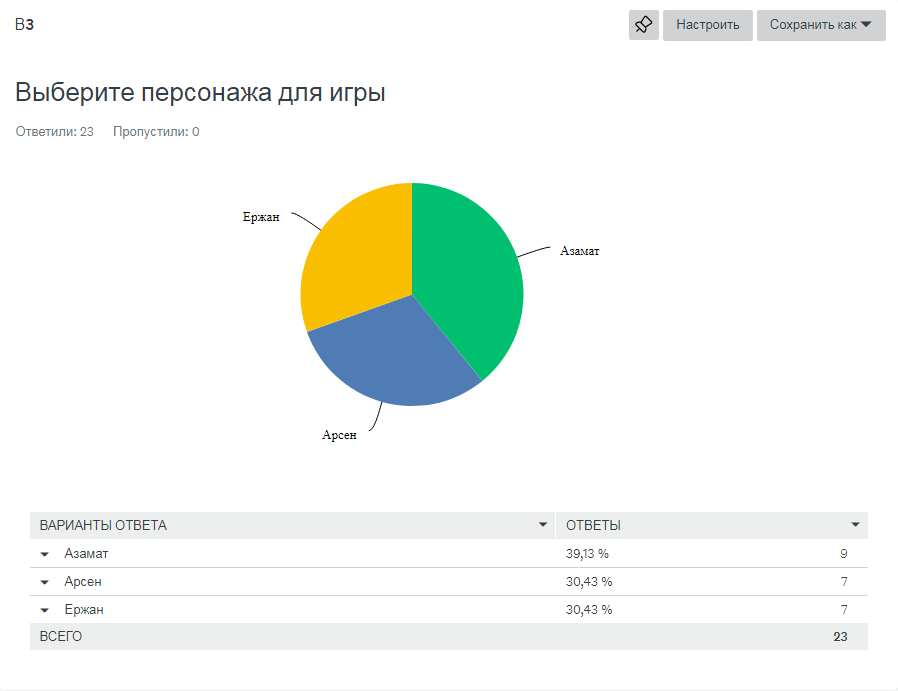


Рис. 4. Вопрос 3

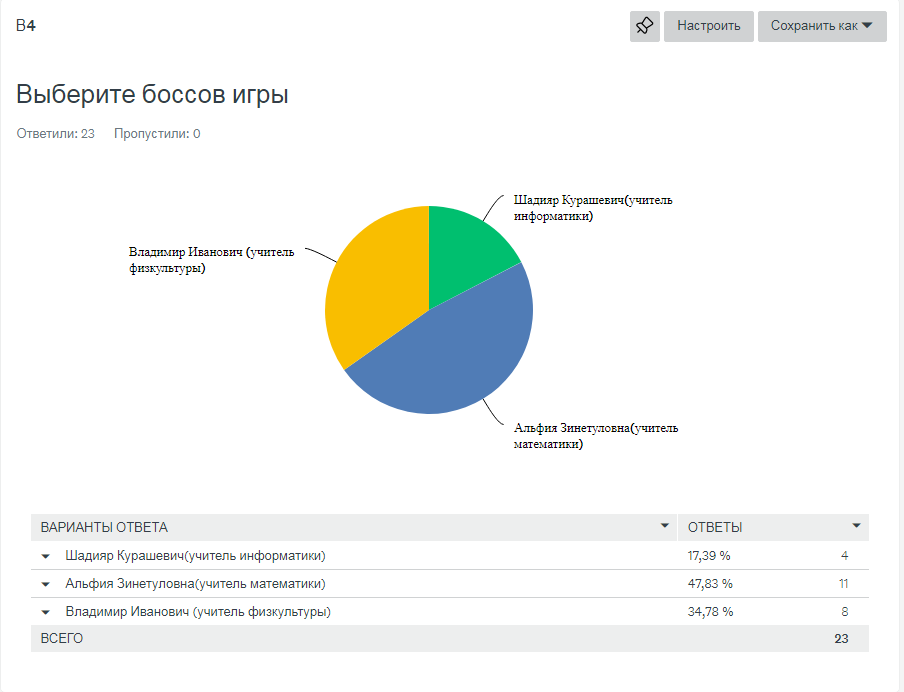


Рис. 5. Вопрос 4

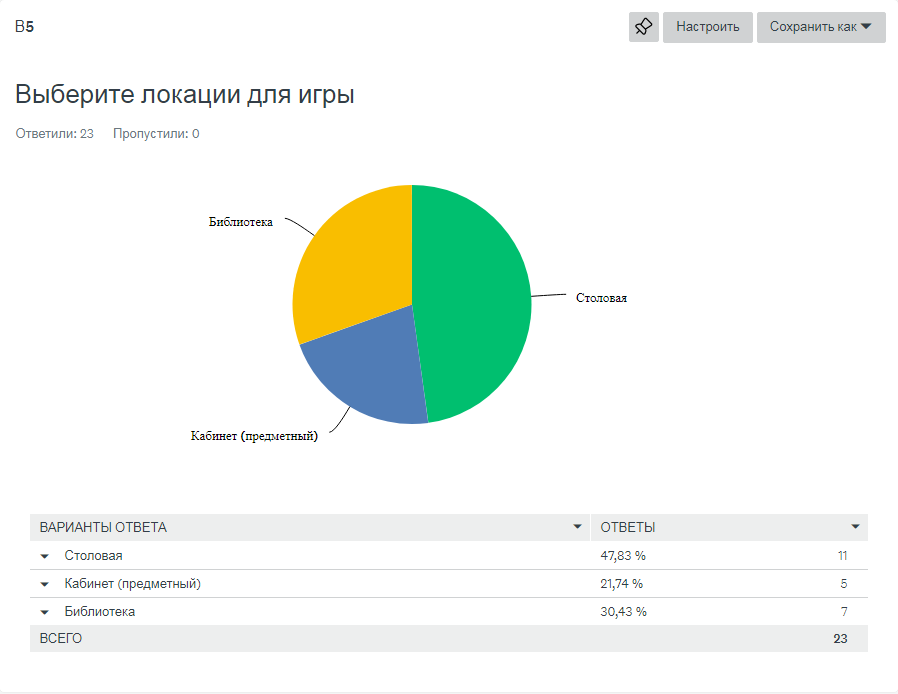


Рис. 6. Вопрос 4

По итогам опроса можно сделать вывод, что игра является востребованной и замысел понравился 100% прошедших опрос. Также прошедшие опрос хотят видеть Азамата в качестве главного героя, столовой в качестве локации и Альфии Зинетуловны в качестве антогониста. Однако учитель отказался от такого предложения и пришлось использовать одноклассников в качестве врагов.

### План работы над проектом

Для разработки проекта я выбрал программу Unity 2019, Word, GIMP, Draw.io и Paint. Я использовал Unity для написания кода, создание анимации объектов, GIMP и Paint для создания моделей персонажей и спрайтов для анимации, Draw.io для блок-схемы, а Word для документации игры. Для начала нужно будет создать спрайты и модельки, далее разработка самой игры, после разработки, я буду писать документации.

# 2. Проектирование

## 2.1 Прототип решения

Внизу на Рис. 2 приведен будущий интерфейс главного меню, через которое пользователь сможет начать игры или выбрать определённый уровень игры.

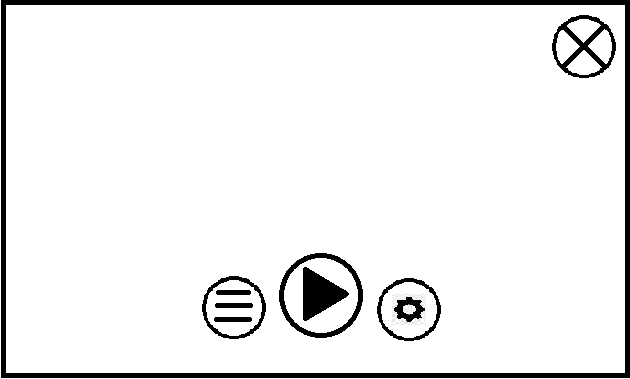


Рис. 7. Прототип сцены “\_MENU”

Далее и у пользователя будет выбор между кнопками. Кнопка настроек в меню не имеет функционала и нужна для эстетичности дизайна, а кнопка закрытия выходит из приложения. При нажатии на кнопку “start” (по центру), появляется сцена с игрой (Рис. 7). При нажатии на кнопку “LevelList” появится панель (Рис. 9). В панели будет выбор между 3 уровнями, соответственно названиям кнопок, то есть кнопка “level 1” открывает сцену “LEVEL1” (Рис. 7), кнопка “level 2” открывает сцену “LEVEL2” (Рис. 10), кнопка “level 3” открывает сцену “LEVEL3” (Рис. 11). Если игрок умирает, то появляется сцена “DeathScreen” (Рис. 12). При нажатии на “Esc” активируется панель паузы (Рис. 13), в которой вы можете выйти в меню или продолжить игру.

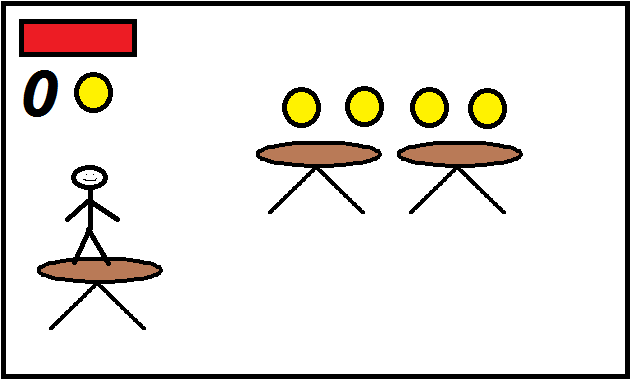


Рис. 8. Прототип сцены “LEVEL1”

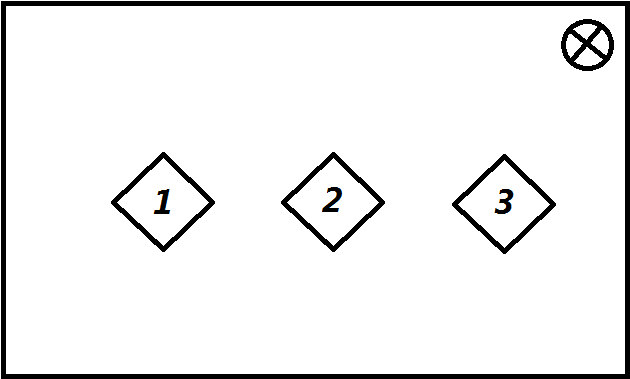


Рис. 9. Прототип панели “LevelsPanel”



Рис. 10. Прототип сцены “LEVEL2”

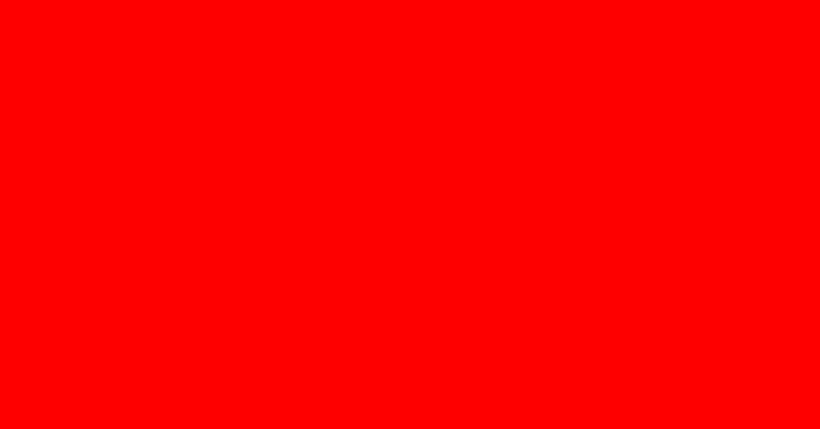


Рис. 11. Прототип сцены “LEVEL3”

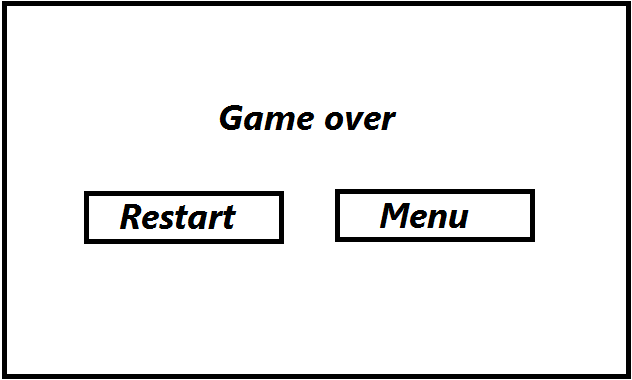


Рис. 12. Прототип сцены “DeathScreen”

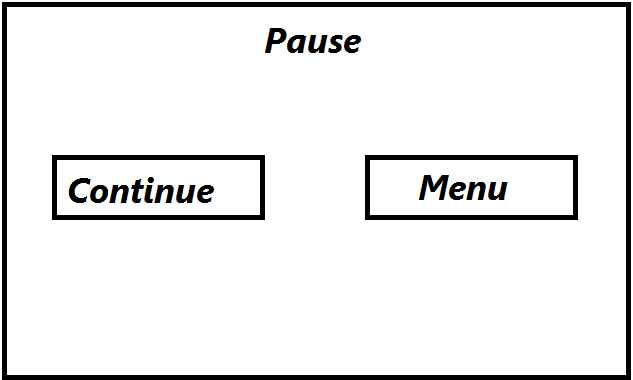


Рис. 13. Прототип панели “Pause\_menu”

## 2.2 Алгоритм работы программы

В моем проекте было использовано 11 различных скриптов. Скрипт “hero” является главным скриптом, так как он отвечает за движение и анимацию героя. Код “Camera” отвечает за динамичное движение камеры около главного героя. Скрипт “buttonSound” нужен для обеспечения звуковых эффектов в игре. Далее я использовал скрипт “Health” и “HealthBar” для системы жизней героя, а скрипты “Coin\_gain” и “Coin\_gainer” для счета собранных монет и их уничтожения. Скрипт “CollisionDamage” нужен для получения урона от врагов, код “Guard” является примитивным ИИ врага и скрипт “Ersul\_stol” нужен для особенности противника. Далее код “MENU” необходим для переключения между сценами, а скрипт “pause\_menu” для остановки игры.

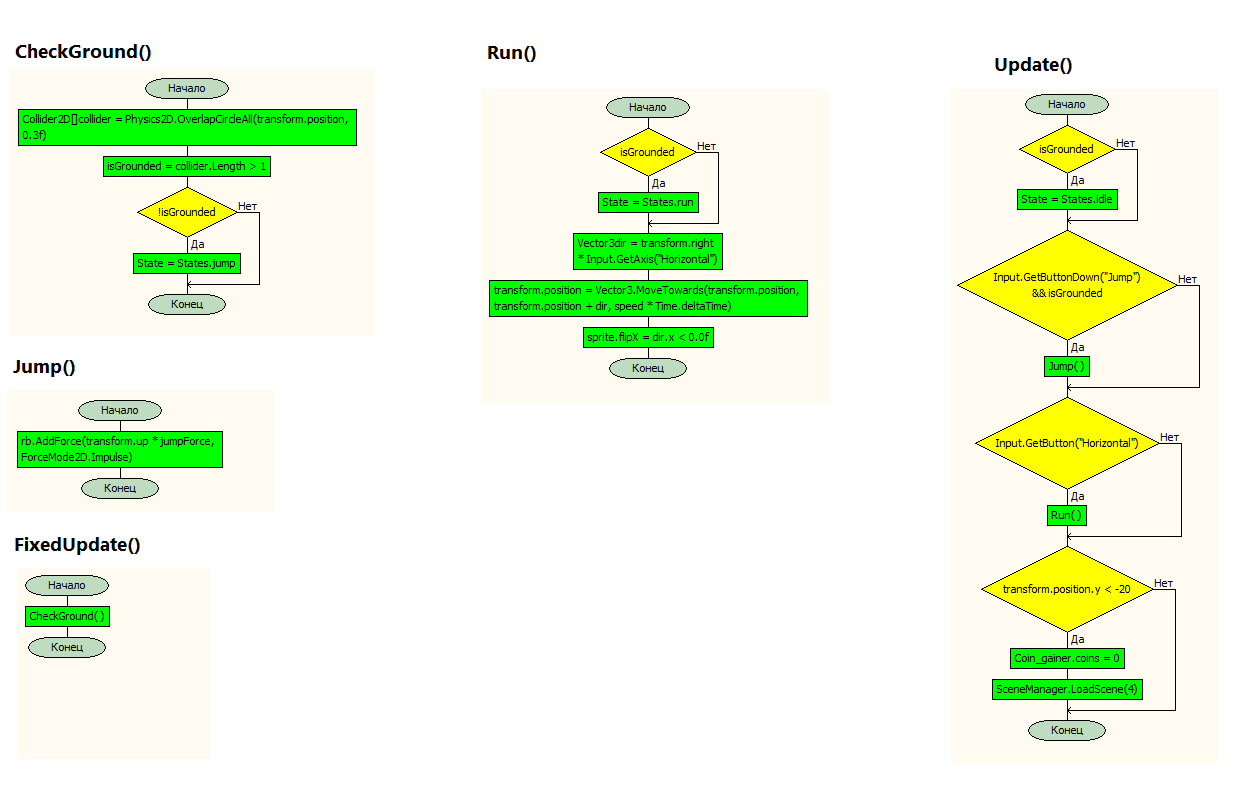


Рис. 14. Скрипт “hero”

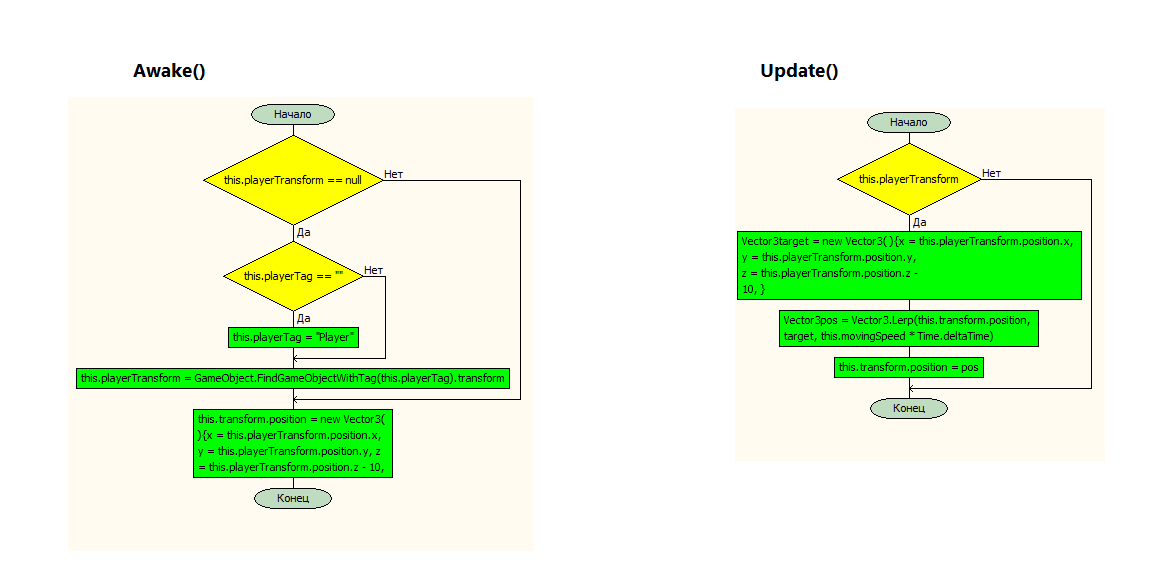


Рис. 15. Скрипт “Camera”

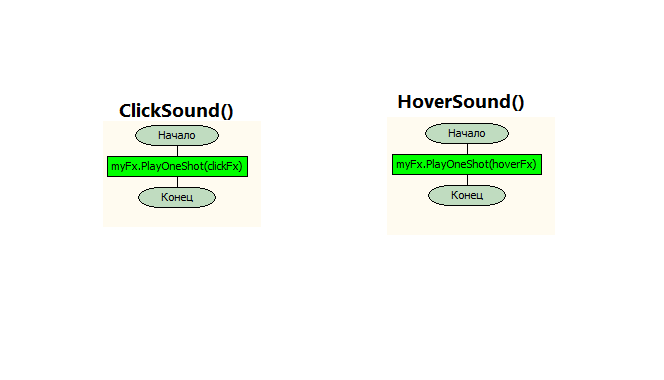


Рис. 16. Скрипт “buttonSound”

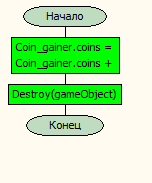


Рис. 17. Скрипт “Coin\_gain”

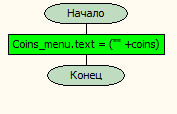


Рис. 18. Скрипт “Coin\_gainer”

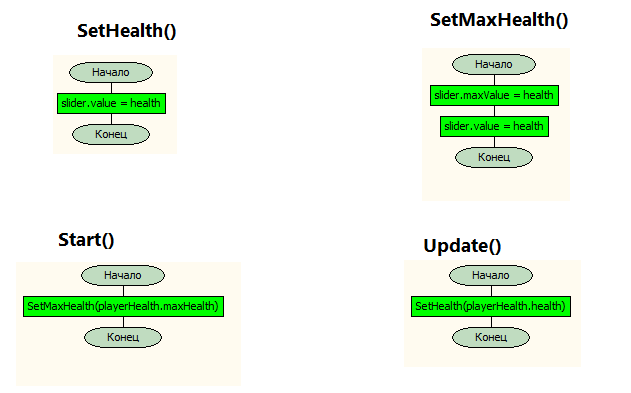


Рис. 19. Скрипт “HealthBar”

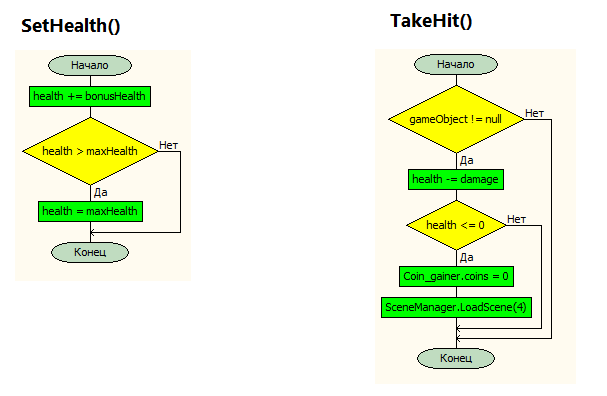


Рис. 20. Скрипт “HealthBar”

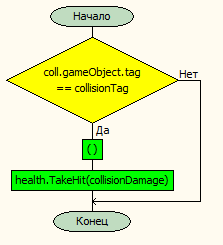


Рис. 21. Скрипт “CollisionDamage”

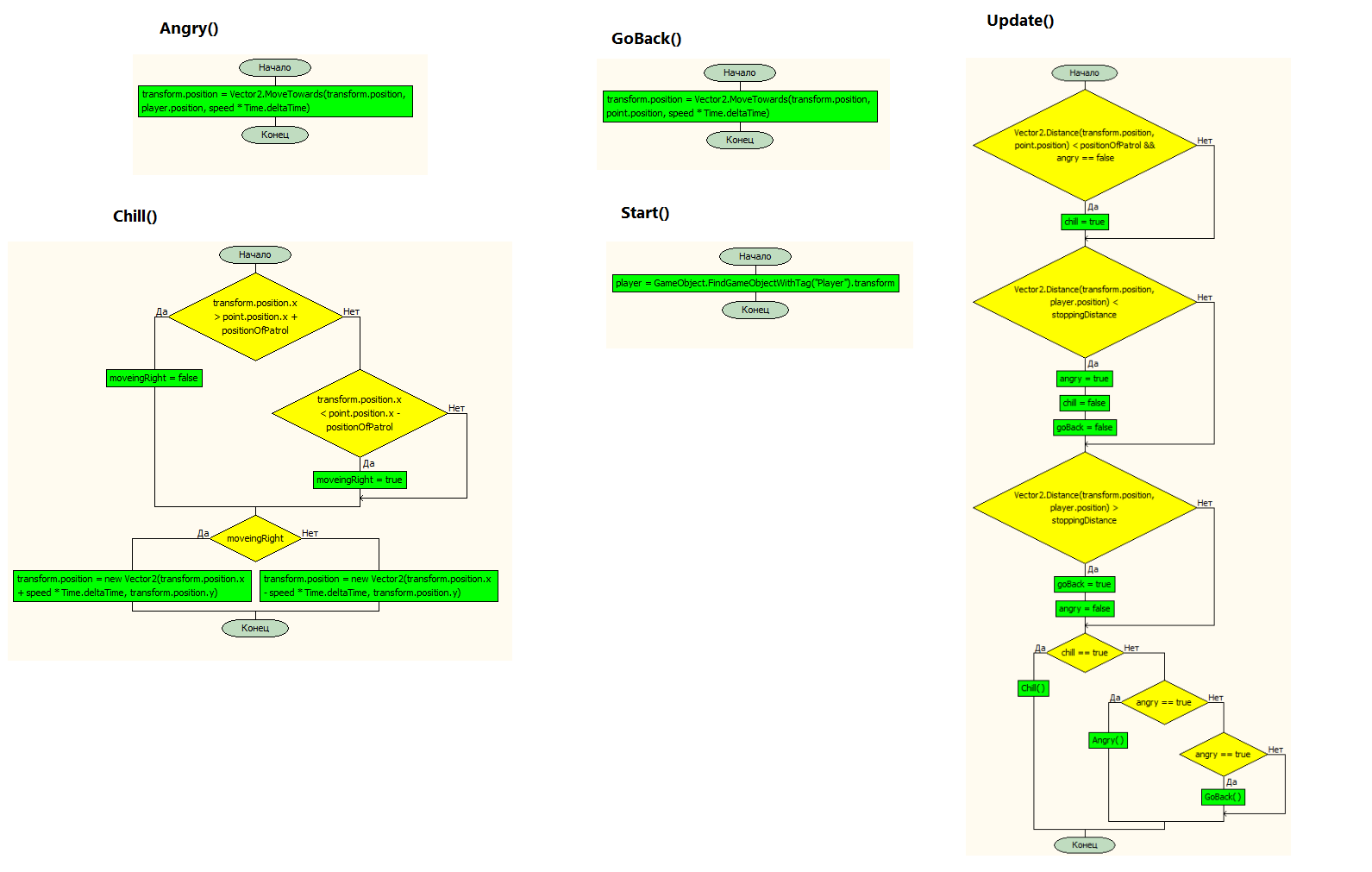


Рис. 22. Скрипт “Guard”

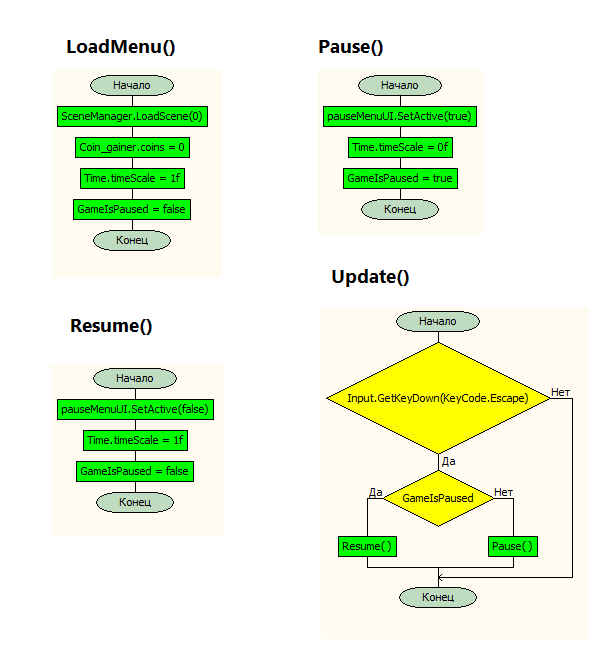


Рис. 23. Скрипт “Pause\_menu”

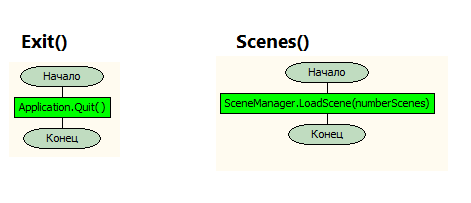


Рис. 23. Скрипт “MENU”

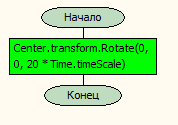


Рис. 23. Скрипт “Ersul\_stol”

## 2.3 Преимущества решения

Преимуществом данного решения состоится в том, что использую пользователь может играть безгранично и каждая игра будет отличаться от прошлой, так как проект включает в себя множество интересных возможностей, такие как музыка, уникальные враги, интересные анимации и множество забавных картинок.

## 2.4 Ограничения решения

Программа ограничена в плане добавления новых уровней, особенностей героя и остального. Также проект содержит несколько особенностей, при малейшем изменении которых повреждается сама игра.

# 3. Реализация

## 3.1 Использованные структуры данных

В ходе создания двумерной игры был использован массив коллайдеров, который имеет специфический тип данных, а также некоторое количество циклов.

Ниже приведены методы, которые необходимы для переходов между сценами, а также для выхода из приложения.

|  |  |
| --- | --- |
| Методы | Назначение |
| LoadScene() | Для открытия необходимой сцены |
| Quit() | Для закрытия приложения |

Таблица 1. Методы

Во время работы над игрой возникла проблема с прыжком персонажа, то есть персонаж мог совершать прыжки отталкиваясь от воздуха, что является багом игры. Для устранения возникшей проблемы я использовал следующий метод, который содержит массив:

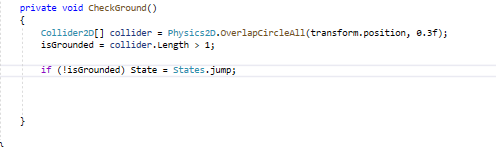


Рис. 24. Метод “CheckGround()”

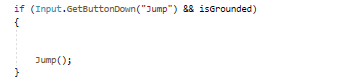


Рис. 25. Условие прыжка

Таким образом код высчитывает есть ли объект (имеющий компонент Box Collider 2D) под персонажем (Рис.24) и если логическая переменная isGrounded равна истене и нажата клавиша для прыжка, то игрок совершит прыжок. (Рис.25)

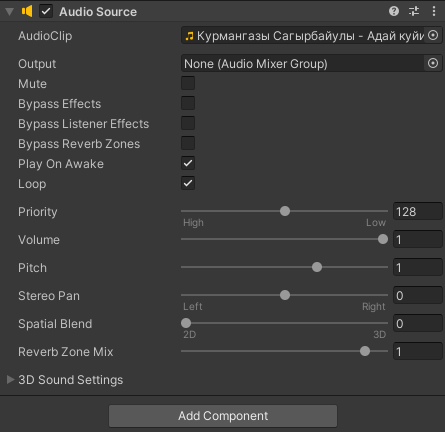


Рис. 26. Audio Source

Цикл в этом проекте используется для повторного исполнения фоновой музыки игры, так как игра не имеет временных рамок, а композиция длится не более 3 минут. (Рис. 26)

## 3.2 Входные и выходные данные

По существу, разработчики мелкомасштабных игровых проектов (platformer) не добавляют входные данные, так как это неправильно со стороны оформления и дизайна. К примеру, добавление окна регистрации в Tetris было бы не уместно. Однако я прогнулся под систему оценивания и добавил некоторое количество входных данных, которые портят игровую атмосферу.

|  |  |  |
| --- | --- | --- |
| *Название сцены* | *Входные данные* | *Комментарий* |
| \_MENU | Click Fx  Hovew Fx | Данные нужны для добавления музыки кнопкам при нажатии на них. То есть пользователь (разработчик) может добавить собственные звуки для кнопок |

Таблица 2. Таблица входных данных

|  |  |  |
| --- | --- | --- |
| *№ Сцены* | *Выходные данные* | *Комментарий* |
| LEVEL1 | Pause\_menu | При нажатии на клавишу “Esc” появится панель паузы, в которой заказчик сможет войти в режим паузы или выйти в главное меню |
| LEVEL1 | DeathScreen | После смерти главного героя появляется панель DeathScreen, в которой пользователь может начать игру сначала или выйти в главное меню |

Таблица 3. Таблица выходных данных

## 3.3 Таблица идентификаторов

|  |  |  |  |
| --- | --- | --- | --- |
| *Название скрипта* | *Имя переменной* | *Тип переменной* | *Комментарий* |
| Health, HealthBar, CollisionDamage | health | int | Отвечает за уровень жизни персонажей |
| Health, HealthBar | maxHealth | int | Переменная для определения максимального уровня жизни |
| Health | damage | int | Переменная, которая отвечает за урон, полученный от противника |
| Health | bonusHealth | int | Дополнительное здоровье (будет добавлено в следующем обновлении) |
| hero | speed | float | Переменная скорости персонажа |
| hero | jumpForce | float | Переменная силы прыжка персонажа |
| hero | isGrounded | bool | Переменная для определения наличия земли под героем в целях отмены двойного прыжка |
| Camera | moveingSpeed | float | Скорость движения камеры, которая следует за персонажем |
| Camera | playerTag | string | Переменная для отслеживания персонажа по тегу |
| pause\_menu | GameIsPaused | bool | Переменная, отвечающая за остановку игры |
| Coin\_gain, Coin\_gainer, hero | coins | int | Переменная, которая хранит количество собранных монет |
| MENU | numberScenes | int | Переменная, отвечающая за номера сцен |
| CollisionDamage | CollisionDamage | int | Переменная, которая отвечает за урон при столкновении |
| CollisionDamage | CollisionTag | string | Переменная, отвечающая за объект, который столкнется с препятствием |
| Guard | speed | float | Скорость врага |
| Guard | positionOfPatrol | int | Длина территории, по которой движется враг |
| Guard | moveingRight | bool | Направление врага |
| Guard | stoppingDistance | float | Дистанция, на которой враг отстает от главного героя |
| Guard | chill | bool | Враг находится в спокойном состоянии |
| Guard | angry | bool | Враг находится в агрессивном состоянии |
| Guard | goBack | bool | Враг идет обратно на охраняемую им территорию |

Таблица 4. Таблица идентификаторов

## 3.4 Интерфейс программы

По требованиям клиента, были составлены дизайны, связанные со школой и школьной атрибутикой. Также было использовано 5 сцен и 2 панели.



Рис. 27. Интерфейс “\_MENU”



Рис. 28. Интерфейс сцены “LEVEL1”



Рис. 29. Интерфейс панели “ LevelsPanel”



Рис. 30. Интерфейс сцены “LEVEL2”

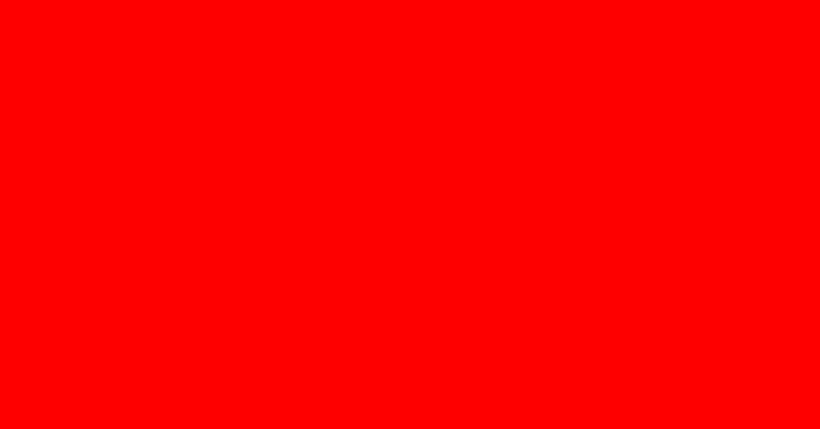


Рис. 31. Интерфейс сцены “LEVEL3”



Рис. 32. Интерфейс сцены “DeathScreen”

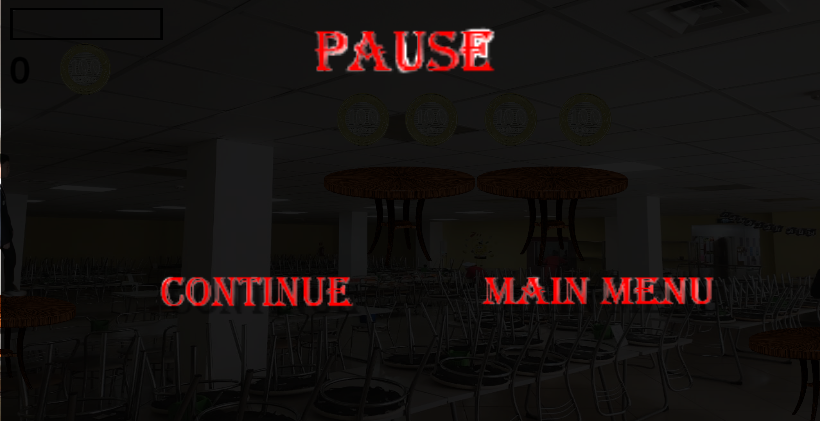


Рис. 33. Интерфейс панели “Pause\_menu”

## 3.5 Листинг программы



Рис. 34. Скрипт “hero”



Рис. 35. Скрипт “hero”



Рис. 36. Скрипт “Camera”

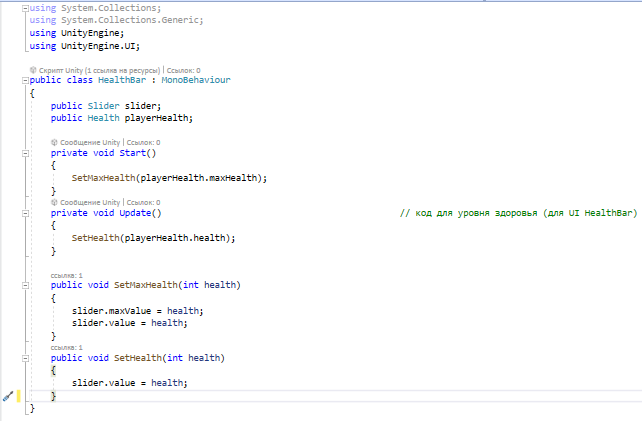


Рис. 37. Скрипт “HealthBar”

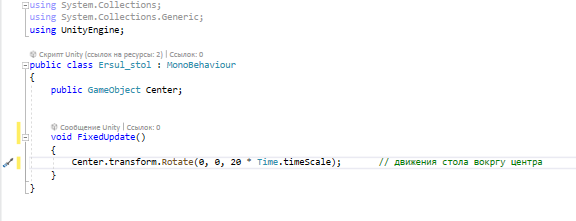


Рис. 38. Скрипт “Ersul\_stol”



Рис. 39. Скрипт “Health”

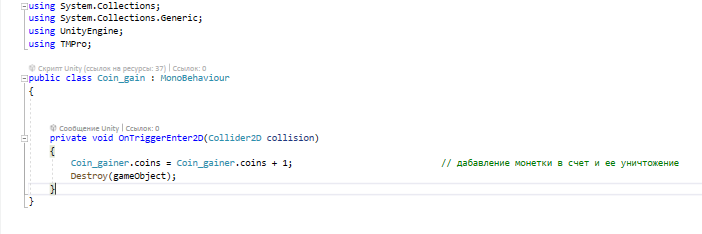


Рис. 40. Скрипт “Coin\_gain”



Рис. 41. Скрипт “Pause\_menu”

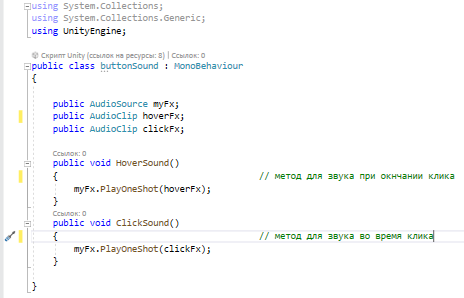


Рис. 42. Скрипт “buttonSound”

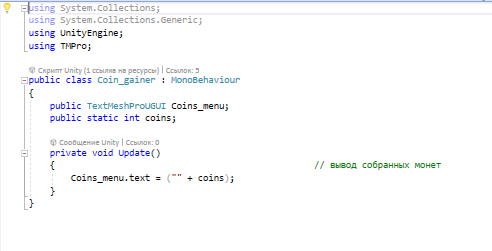


Рис. 43. Скрипт “Coin\_gainer”



Рис. 44. Скрипт “ColissionDamage”

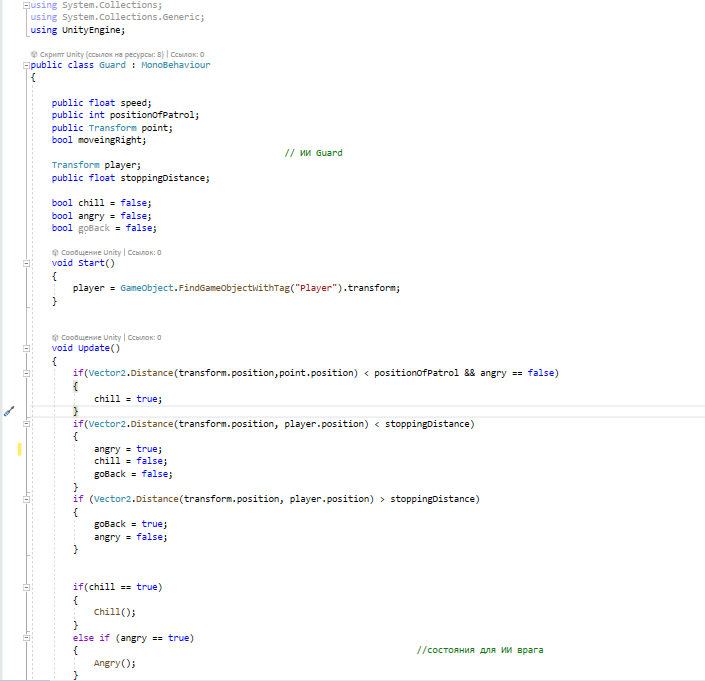


Рис. 45. Скрипт “Guard”

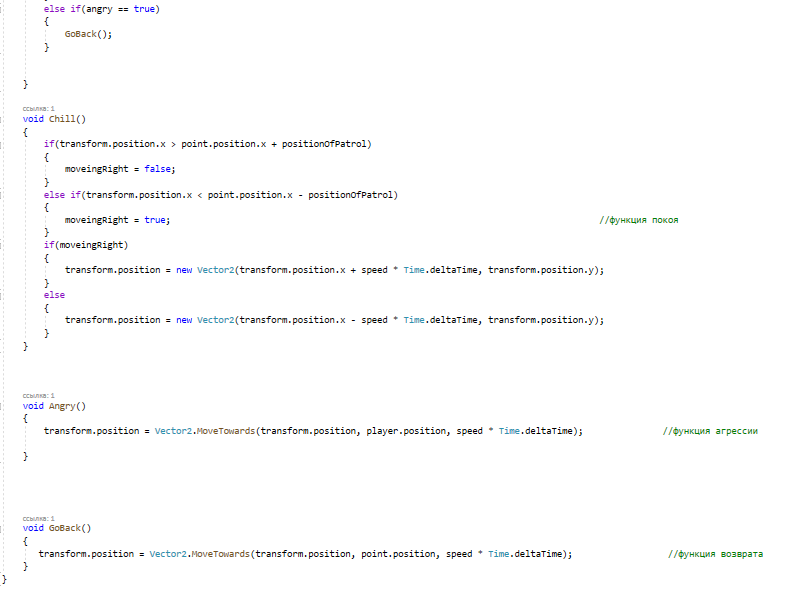


Рис. 46. Скрипт “Guard”

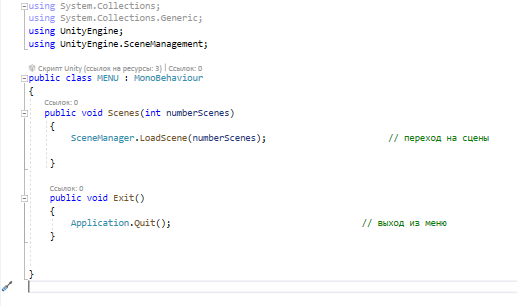


Рис. 47. Скрипт “MENU”

# 4. Тестирование и отладка

## 4.1 План тестирования

1. Тестирование методом чёрного ящика. Проверка работоспособности кнопок меню и управления, проверка слежения камеры за героем, а также других функций

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название сцены | Цель теста | Метод теста | Ожидаемый результат | Фактический результат |
| \_MENU | Проверить работоспособность всех кнопок в этой сцене | Нажать на кнопку “play”, “level panel”, “exit” | Появление сцены “LEVEL1” при нажатии на кнопку “play”, вывод панели “LevelsPanel” при нажатии на кнопку “level panel” и выход из игры при нажатии кнопки “exit” | Успешно. Программа вывела все необходимые сцены, панели, а также осуществляла выход из игры |
| \_MENU | Проверка перехода на уровни игры | Нажать на кнопки “Level 1”, “Level 2”, “Level 3” | При нажатии на соответствующую кнопку выводиться оподленный уровень игры | Успешно. Переход на уровни происходит без задержек и ошибок |
| LEVEL1 | Проверка управления героем | Нажать на клавиши “a”, “d”, “space” и “Esc” | При нажатии на клавишу “a” герой должен двигаться влево, “d” двигаться вправо, “space” совершать прыжок, и при нажатии на клавишу “Esc” будет происходить появление панели “Pause\_menu” | Успешно. Управление и выход в режим паузы работает корректно |
| LEVEL1 | Проверка получения урона | Игроку необходимо дойди до объектов (врагов) “guard” и “obstacle” и столкнуться с ними | Уровень здоровья, находящийся в объекте “HealthBar” уменьшится | Успешно, игрок получает урон от врагов и уровень жизни снижается |
| LEVEL1, DeathScreen | Проверка гибели игрока, то есть появление сцены “DeathScreen” | Игрок должен упасть в пустое пространство ниже точки, ординат которой равен -20 или понизить уровень здоровья до нуля в объекте “HealthBar” | Должна появиться сцена “DeathScreen” | Успешно. Сцена сразу же отобразилась после смерти игрока |
| LEVEL1 | Проверка сбора, счета и появления количества монет в Canvas | Столкнуться с объектом, к которому прикреплен скрипт “Coin\_gain”, к примеру объект “monteka(1)” | Объект “monteka(1)” должен исчезнуть, а в Canvas (TekstMeshPro) должно появиться количество собранных монет | Успешно. На счет добавилась одна монета. |

Таблица 5. Тестирование методом чёрного ящика

2. Тестирование методом белого ящика представляет собой метод, при котором тестирующий программу человек вводит три вида данных (нормальные, экстремальные и ошибочные). Таким образом при прохождении этого тестирования позволит разработчику увидеть ошибки в своем скрипте.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Сцена | Компонент | Нормальные | Экстремальные | Ошибочные |
| LEVEL1 | hero (GameObject) | Нажатие на кнопки управления “a”, “d”, “space” и кнопку “Esc”/ “a” – герой двигается на лево, “d” - герой двигается на право, “space” – герой совершает прыжок, “Esc” – происходит переход в режим паузы | Нажатие на “space” больше двух раз за секунду и зажатие данной клавиши / данное явление заставляет персонажа прыгать, отталкиваясь от воздуха, что не реалистично (баг). На данный момент баг исправлен, но персонаж близок к совершению полета при данных условиях | Нажатие на другие кнопки, по типу “1”, “2”, “Tab” / при нажатии данных клавиш в игре не будет происходить ничего |

Таблица 6. Типичные, экстремальные и неправильные значения

# 5. Оценивание, документирование и внедрение

**5.1 Руководство пользователю.**

**Инструкция по использованию игры**

1. Скачайте архив с файлами игры по ссылке.
2. Нажмите на активатор игры “menu2”.
3. Для начала игры нажмите центральную кнопу “play”, для выбора уровней нажмите кнопу “levels menu”, а для закрытия приложения нажмите на крест в правом верхнем углу.
4. Управление “a” – для движения влево, “d” – для движения вправо, “space” – для прыжка. Клавиша “Esc” для выхода в режим паузы.

## 5.2 Внедрение, оценивание клиента

Честно говоря, игра превзошла все мои ожидания. По началу я думал, что это будет обычный платформер с кривой графикой, так как разработчик изучал программу Unity 2019 не больше месяца. Однако не очень плавная графика сочеталась с сюжетом игры, удобное управление и интересные модельки персонажей и объектов, также фоновая музыка игры была очень колоритна. Я считаю, что проект является революционным в нашей школе, ведь он выделяется среди серой массы других приложений.

Клиент: Аманбай Азамат, 10F