Муниципальное бюджетное общеобразовательное учреждение  
«Инженерный лицей №83 имени Героя Советского Союза Пинского Матвея Савельевича Уфимского государственного нефтяного технического университета»

городского округа город Уфа Республики Башкортостан

Индивидуальный проект  
по теме:  
«Создание игры ”Dungeon Escape” на языке программирования java при помощи фреймворка libGDX»

Обучающийся:

Гагарин Егор Александрович

Руководитель проекта:

Гильдин Александр Григорьевич,

учитель информатики

г. Уфа, 2023

**СОДЕРЖАНИЕ**

Введение . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

Глава 1 Планирование структуры и функционала мобильного приложения . . . 4

Глава 2 Выбор способа реализации мобильного приложения . . . . . . . . . . . . . . . 6

Глава 3 Разработка приложения . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

Глава 3.1 Создание окна главного меню игры . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

Глава 3.2 Создание окна настроек игры . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

Глава 3.3 Создание окна таблицы рекордов игры . . . . . . . . . . . . . . . . . . . . . . . . 10

Глава 3.4 Создание окна игры . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

Заключение . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12

Список литературы . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

Приложение . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

**ВВЕДЕНИЕ**

Мобильный телефон есть почти у каждого человека и мы пользуемся им ежедневно. Он позволяет не только искать различную информацию для учебы, но также может служить в качестве развлекательного средства для просмотра фильмов, прослушивания музыки или игр в различные мобильные приложения. Мы поставили перед собой **цель**, создать игровое приложение для мобильного телефона с операционной системой Android, посчитав **актуальной** тему создания мобильных приложений в современном мире, где с каждым годом количество мобильных устройств только увеличивается, а значит увеличивается и спрос на мобильные игры. Полученные в процессе разработки знания и навыки также могут пригодиться в дальнейшем не только нам, но и любым другим людям, желающим создать свое игровое приложение, поскольку проект находится в открытом доступе в облачной платформе хранения репозиториев GitHub.

Итак, для реализации поставленной нами цели, должны будут быть выполнены следующие **задачи**:

1. Изучить информацию по языкам программирования, подходящим для создания android приложений.
2. Выбрать наиболее подходящие инструменты для реализации задуманного функционала игры.
3. Спроектировать архитектуру приложения в соответствии с принципами ООП.
4. Реализовать разработанную нами архитектуру в соответствии с принципами ООП используя выбранный ранее язык программирования и инструменты разработки.
5. Протестировать приложение на наличие ошибок и некорректной работы при определённых сценариях использования и исправить соответствующие недочёты.

**ГЛАВА 1 ПЛАНИРОВАНИЕ СТРУКТУРЫ И ФУНКЦИОНАЛА ПРИЛОЖЕНИЯ**

Исходя из поставленной цели, мы разработали концепцию будущего игрового приложения. Смыслом игры будет выбраться из подземелья - лабиринта за минимальное количество ходов. При запуске приложения будет открываться главное меню с заставкой игры и кнопками «Начать», «Настройки» «Таблица рекордов».

При нажатии кнопки «Начать» откроется диалоговое окно с предложением выбрать имя персонажу, после нажатия кнопки подтверждения если имя выбрано корректно (в имени отсутствуют пробелы и его длина не нулевая и не превышает 10-ти символов) то запуститься сама игра. Игровое поле будет представлять собой двумерную сетку из различных клеток. Игрок сможет перемещаться между соседними клетками. Некоторые клетки будут представлять собой стены и в них игрок не сможет переместиться. Также в подземелье будут встречаться запертые двери, чтобы переместиться в которые необходимо сначала найти и активировать рычаг, который их откроет. Чтобы игра не была слишком легкой в подземелье будут встречаться враги - слаймы, которые будут пытаться атаковать игрока если он попадет в их поле зрения. Игрок также сможет атаковать слаймов в ответ. Для усложнения прохождения радиус атаки слайма будет увеличен до 2-х клеток, также его снаряды смогут перелетать двери и атаковать игрока за ними. Игрок может перемещаться в 4-х направлениях. Слаймы, в отличии от игрока, могут перемещаться в 8-ми направлениях. И игрок и слаймы не могут ходить сквозь стены. Управление будет осуществляться нажатием на соседние с игроком клетки (если игрок не находится в состоянии атаки и в клетку можно переместиться, то игрок двигается, если нельзя, то игрок остается на месте и ход не засчитывается, в состоянии атаки если в атакуемой клетке есть враг, то игрок совершает атаку и наносит ему урон, если клетка пуста, то ход не засчитывается), а также нажатием кнопок на экране, всего в игре будет 5 кнопок: кнопка подсказок (открывает окно с информацией о цели игры и управлении), кнопка карты (открывает окно с картой, изначально карта будет лежать на земле в подземелье — лабиринте и кнопка появится у игрока после того как он наступит на клетку с картой и поднимет ее), кнопка атаки (при ее нажатии она меняет свой цвет, что сигнализирует о том, что следующим ходом игрок не переместится, а выполнит атаку) и кнопка «сдаться» (записывает текущее состояния игрока в таблицу рекордов как проигрыш и возвращает его в главное меню. При нажатии этой кнопки появится диалоговое окно, предлагающее игроку вписать слово подтверждения этого действия). Игра будет пошаговой: после каждого хода игрока слаймы будут делать свой ход. Ходом будет считаться любое движение или атака, то есть просмотр карты или смена состояния атаки не будут считаться ходом. После того, как игрок дошел до выхода из подземелья или погиб открывается окно с изображением победы или поражения и предложением вернуться в главное меню. Имя игрока, количество ходов и успешность прохождения (игра закончилась победой или поражением) добавляется в таблицу рекордов.

При нажатии кнопки «Настройки» откроется окно настроек, где при помощи кнопок можно поменять язык (русский или английский), отключить автоотключение кнопки атаки (кнопка атаки будет деактивироваться после одной атаки) и очистить таблицу рекордов (при этом появится диалоговое окно, предлагающее игроку вписать слово подтверждения этого действия) а также вернуться в главное меню при помощи кнопки «Назад».

При нажатии кнопки «Таблица рекордов» откроется окно с таблицей рекордов. Таблица состоит из 3-х столбцов: «Имя» (имя игрока), «Ходы» (количество сделанных ходов) и «Итог» (Пройдено или не пройдено). Если количество записей в таблице превышает 5 то для корректного отображения данных появляются кнопки «Вверх» и «Вниз», позволяющие пролистать таблицу вверх или вниз. Когда положение таблицы конечное и дальше пролистывать уже некуда кнопки исчезают. Также можно вернуться в главное меню при помощи кнопки «Назад».

Чтобы игрок мог продолжить игру после закрытия приложения и его прогресс прохождения в игре сохранился нами будет реализована система сохранения. Поскольку игра пошаговая, рационально будет сохранять состояния только нестатичных объектов игрового поля (слаймов, дверей и рычагов, состояние и характеристики персонажа) после каждого хода персонажа.

В игре будут использоваться звуковые эффекты и фоновая музыка. Они будут взяты с бесплатного сайта Opengameart. Изображения для игры будут созданы нами самостоятельно. В качестве стиля изображений было решено использовать пиксель-арт, потому что он не требует высоких художественных навыков и позволяет получить оригинальные изображения одинакового качества. Для создания изображений нами будет использоваться бесплатный онлайн пиксельный редактор Piskelapp.

**ГЛАВА 2 ВЫБОР СПОСОБА РЕАЛИЗАЦИИ ПРИЛОЖЕНИЯ**

После планирования функционала приложения, мы можем выделить необходимые инструменты для его реализации. Так как игра будет предназначена для телефонов с операционной системой Android необходимо использовать совместимые с ней языки программирования. Самые популярные языки программирования Android приложений это Java, Kotlin и C#.

Java — статически типизированный объектно-ориентированный язык программирования общего назначения. Большим преимуществом Java является кросплатформенность: создатели языка реализовали принцип WORA (write once, run anywhere) что дословно переводится как «пиши один раз, запускай везде». Это возможно благодаря компиляции написанного на Java кода в байт-код. Этот формат исполняет JVM или виртуальная машина Java, являющаяся частью среды исполнения Java (JRE). Виртуальная машина не зависит от платформы поэтому достаточно установить ее на устройстве чтобы одну программу можно было запустить на любом устройстве.

Kotlin — статически типизированный объектно-ориентированный язык программирования, изначально создовался как улучшенная версия языка программирования Java. Также, как и Java он использует для кроссплатформенной работы JVM. Kotlin код может компилироваться в JavaScript. Однако поскольку Kotlin более молодой язык программирования, для него существует значительно меньше библиотек и фреймворков, чем для Java.

C# — статически типизированный объектно-ориентированный язык программирования общего назначения. Синтаксис этого языка программирования схож с синтаксисом Java, но он не поддерживается официальной средой разработки для Android приложений — Android Studio, в отличии от языков Java и Kotlin.

Мы будем использовать язык программирования Java, потому что он долгое время был основным официальным языком разработки приложений для OS Android, поэтому для него есть большое количество справочного материала, фреймворков и библиотек. Для написания кода мы будем использовать официальную среду разработки приложений для OS Android - Android Studio.

Так как нам понадобится работать с двумерной графикой, а платформой приложения будет Android – нам подойдут следующие фреймворки для языка программирования Java: Android Native, LibGDX.

Android Native – платформа разработки, которая входит в число самых популярных в мире платформ для разработки мобильных приложений. Этот фреймворк предоставляет библиотеки и инструменты разработчика, необходимые для создания, тестирования и отладки приложений для платформы Android. Главным его преимуществом является высокая скорость работы.

LibGDX - кроссплатформенный фреймворк для разработки игр и визуализации, основанный на языке программирования Java с некоторыми компонентами, написанными на C и C++ для повышения производительности определенного кода. Удобная система инициализации позволяет быстро создать шаблон для будущего приложения. Благодаря кроссплатформенности можно производить отладку кода на том же компьютере, который используется для написания кода, без необходимости постоянно производить отладку на мобильном устройстве с OS Android, что значительно ускоряет и упрощает разработку.

Мы будем использовать фреймворк LibGDX потому что он проще и удобнее, требует меньше действий для создания шаблона приложения и уже содержит библиотеку Gdx, которая будет использоваться для создания игры.

**ГЛАВА 3 РАЗРАБОТКА ПРИЛОЖЕНИЯ**

Разработка приложения будет происходить согласно принципам ООП. Необходимо разработать 13 классов. Из них 5 классов будут расширять интерфейс Screen библиотеки Gdx, и один класс будет расширять класс Game библиотеки Gdx.

Изображения всех визуальных составляющих мобильного приложения представлены в приложении к настоящему проекту.

**ГЛАВА 3.1 СОЗДАНИЕ ОКНА ГЛАВНОГО МЕНЮ ИГРЫ**

Разработку приложения мы начнем с главного меню. В нем используется 5 классов из 13 задуманных.

**Класс DungeonEscape.** Этот класс будет основным классом игры. Он расширяет класс Game библиотеки libGDX. В этом классе инициализируются все объекты классов Texture, Sound, SpriteBatch, BitmapFont, которые будут использоваться в игре, считываются данные из текстовых файлов, отвечающих за хранение координат объектов на карте, которые потом заполняются в двумерные массивы для использования в классе GameScreen.

Также в этом классе рассчитывается переменная size, отвечающая за определение размеров одной ячейки игрового поля.

На экране должна отображаться область 9 на 7 клеток, а также столбик из 7 клеток для кнопок, то есть всего (9+1)\*7=70 клеток.

Размеры одной квадратной клетки вычисляются по алгоритму:

size = height / 7;  
horizontal\_otstup = (width-size\*10)/2;  
vertical\_otstup = 0;  
if(horizontal\_otstup < 0){  
 horizontal\_otstup = 0;  
 size = width/10;  
 vertical\_otstup = (height - 7\*size)/2;  
}

Где horizontal\_otstup, vertical\_otstup - горизонтальный и вертикальный отступы, необходимые для отображения игрового поля ровно по середине экрана, максимально большими; height, width - высота и ширина экрана, size - полученное в результате значение. После инициализации всех переменных, создается новый объект класса MainMenuScreen, при инициализации которого при помощи this ему передаются ссылки на все ранее инициализированные переменные и вызывается метод setScreen класса Game, получающий в качестве входного параметра этот экземпляр класса MainMenuScreen и устанавливающий его в качестве активного интерфейса Screen.

**Класс MainMenuScreen.** Этот класс будет отвечать за создание и отображение главного меню игры, из которого можно начать игру, перейти в таблицу рекордов или в настройки. В зависимости от нажатой кнопки он создает новый объект класса RecordScreen, либо GameScreen, либо SettingsScreen, при инициализации которого при помощи this ему передаются ссылки на все ранее инициализированные переменные и вызывается метод setScreen класса Game, получающий в качестве входного параметра этот экземпляр класса RecordScreen, либо GameScreen, либо SettingsScreen и устанавливающий его в качестве активного интерфейса Screen. В случае с кнопкой начала игры сначала вызывается метод Gdx.input.getTextInput, который переопределяется при инициализации объекта класса TextInputListener. Этот метод при вызове запускает диалоговое окно с предложением выбора имени персонажа.

**Класс Animation.** Этот класс отвечает за создание анимации, ее отображения и смены кадров в ней.

**Классы Player и Slime.** Эти классы предназначены для создания и отображения персонажа и врага, а также за их взаимодействие. В главном меню будут созданы по одному экземпляру этих классов для экранной заставки.

**Класс Charge.** Этот класс отвечает за создание и отображение снаряда враги или игрока. Объект этого класса создается при инициализации персонажа или врага-слайма.

**ГЛАВА 3.2 СОЗДАНИЕ ОКНА НАСТРОЕК**

Теперь разработаем окно настроек. Помимо уже созданного класса DungeonEscape в нем будет использоваться класс SettingsScreen.

**Класс SettingsScreen.** Этот класс предназначен для создания и отображения окна настроек. Он расширяет класс ScreenAdapter библиотеки GDX. При инициализации объект этого класса устанавливается как активный объект Screen и получает возможность отображения на экране мобильного устройства. При отображении на экране выводятся кнопки выбора языка, выбора состояния автоотключения кнопки атаки (да или нет), кнопка очистки таблицы рекордов (при нажатии этой кнопки вызывается метод Gdx.input.getTextInput, который переопределяется при инициализации объекта класса TextInputListener. Этот метод при вызове запускает диалоговое окно с предложением ввести подтверждение очистки таблицы рекордов) и кнопка возврата в главное меню.

**ГЛАВА 3.3 СОЗДАНИЕ ОКНА ТАБЛИЦЫ РЕКОРДОВ**

Затем создадим окно таблицы рекордов. Помимо уже созданного класса DungeonEscape, для его работы будет использоваться класс RecordScreen.

**Класс RecordScreen.** Этот класс отвечает за отображение таблицы рекордов. Он расширяет класс ScreenAdapter библиотеки GDX. При инициализации объект этого класса устанавливается как активный объект Screen и получает возможность отображения на экране мобильного устройства.

**ГЛАВА 3.4 СОЗДАНИЕ ОКНА ИГРЫ**

Последним создадим окно самой игры. Помимо уже созданных классов Animation, Charge, DungeonEscape, Player и Slime необходимо реализовать классы Cage, Lever, WinScreen, DeathScreen, GameScreen.

**Класс Cage.** Этот класс отвечает за создание и отображение ячейки игрового поля.

**Класс Lever.** Этот класс отвечает за создание и отображение обьекта рычага и связанной с ним двери на игровом поле.

**Класс WinScreen.** Этот класс отвечает за отображение поздравления в случае победы игрока и записи результата прохождения в таблицу рекордов. Он расширяет класс ScreenAdapter библиотеки GDX. При инициализации объект этого класса устанавливается как активный объект Screen и получает возможность отображения на экране мобильного устройства. После нажатия кнопки возврата в главное меню он создает новый объект класса MainMenuScreen, при инициализации которого при помощи this ему передаются ссылки на все ранее инициализированные переменные и вызывается метод setScreen класса Game, MainMenuScreen и устанавливающий его в качестве активного интерфейса Screen.

**Класс DeathScreen.** Этот класс отвечает за отображение анимации смерти в случае проигрыша игрока и записи результата прохождения в таблицу рекордов. Он расширяет класс ScreenAdapter библиотеки GDX. При инициализации объект этого класса устанавливается как активный объект Screen и получает возможность отображения на экране мобильного устройства. После нажатия кнопки возврата в главное меню он создает новый объект класса MainMenuScreen, при инициализации которого при помощи this ему передаются ссылки на все ранее инициализированные переменные и вызывается метод setScreen класса Game, MainMenuScreen и устанавливающий его в качестве активного интерфейса Screen.

**Класс GameScreen.** Этот класс отвечает за сам игровой процесс: отображение игрового поля, персонажа и врагов, кнопок управления персонажем, рычагов и дверей. В этом классе реализованы функции перемещения персонажа, ответных действий врагов, сохранения промежуточного результата прохождения на случай закрытия приложения, проверки состояния кнопок, отображения подсказок в случае их использования, отображения карты и текущих характеристик игрока. При инициализации объект этого класса устанавливается как активный объект Screen и получает возможность отображения на экране мобильного устройства. После победы игрока (достижения клетки с выходом из подземелья-лабиринта) создается новый объект класса WinScreen, при инициализации которого при помощи this ему передаются ссылки на все ранее инициализированные переменные и вызывается метод setScreen класса Game, WinScreen и устанавливающий его в качестве активного интерфейса Screen. В случае же смерти игрока или нажатия кнопки «Сдаться» создается новый объект класса DeathScreen, при инициализации которого при помощи this ему передаются ссылки на все ранее инициализированные переменные и вызывается метод DeathScreen класса Game, WinScreen и устанавливающий его в качестве активного интерфейса Screen.

**ЗАКЛЮЧЕНИЕ**

По мере реализации настоящего проекта нами были изучены доступные средства проектирования приложений для операционной системы Android, выполняя поставленные задачиодну за другой, была достигнута цель проекта. Результатом проектной работы является готовый продукт - однопользовательская двумерная игра с реализованной системой сохранения конечных результатов а также промежуточного прогресса прохождения. Нами получен практический опыт создания приложений используя язык программирования java и фреймворк libGDX. Продукт разработки был загружен на облачную платформу хранения репозиториев GitHub и находится в открытом доступе, поэтому любой желающий сможет на его примере попробовать создать свое мобильное приложение, используя язык программирования java и фреймворк libgdx. В дальнейшем приложение может быть доработано и опубликовано на соответствующих интернет-ресурсах, также может рассматриваться его монетизация.

**СПИСОК ЛИТЕРАТУРЫ**

1. GitHub [Электронный ресурс]. - URL: hhttps://github.com/ (дата обращения: 30.09.2022).
2. It Школа Samsung [Электронный ресурс]. - URL: <https://myitschool.ru/> (дата обращения: 30.09.2022).
3. libGDX [Электронный ресурс]. - URL: http://www.libgdx.ru/ (дата обращения: 30.11.2022).
4. Piskelapp [Электронный ресурс]. - URL: <https://www.piskelapp.com/> (дата обращения: 30.12.2023).
5. Opengameart [Электронный ресурс]. - URL: https://opengameart.org/ (дата обращения: 20.02.2023).

**ПРИЛОЖЕНИЕ**

Рисунок 1. Сравнение ранней и последней версии визуальной составляющей игры.

|  |  |
| --- | --- |
|  |  |
| Ранняя версия игрового поля. | Последняя версия игрового поля. |

Рисунок 2. Главное меню игры.



Рисунок 3. Окно настроек игры.

Рисунок 4. Таблица рекордов.

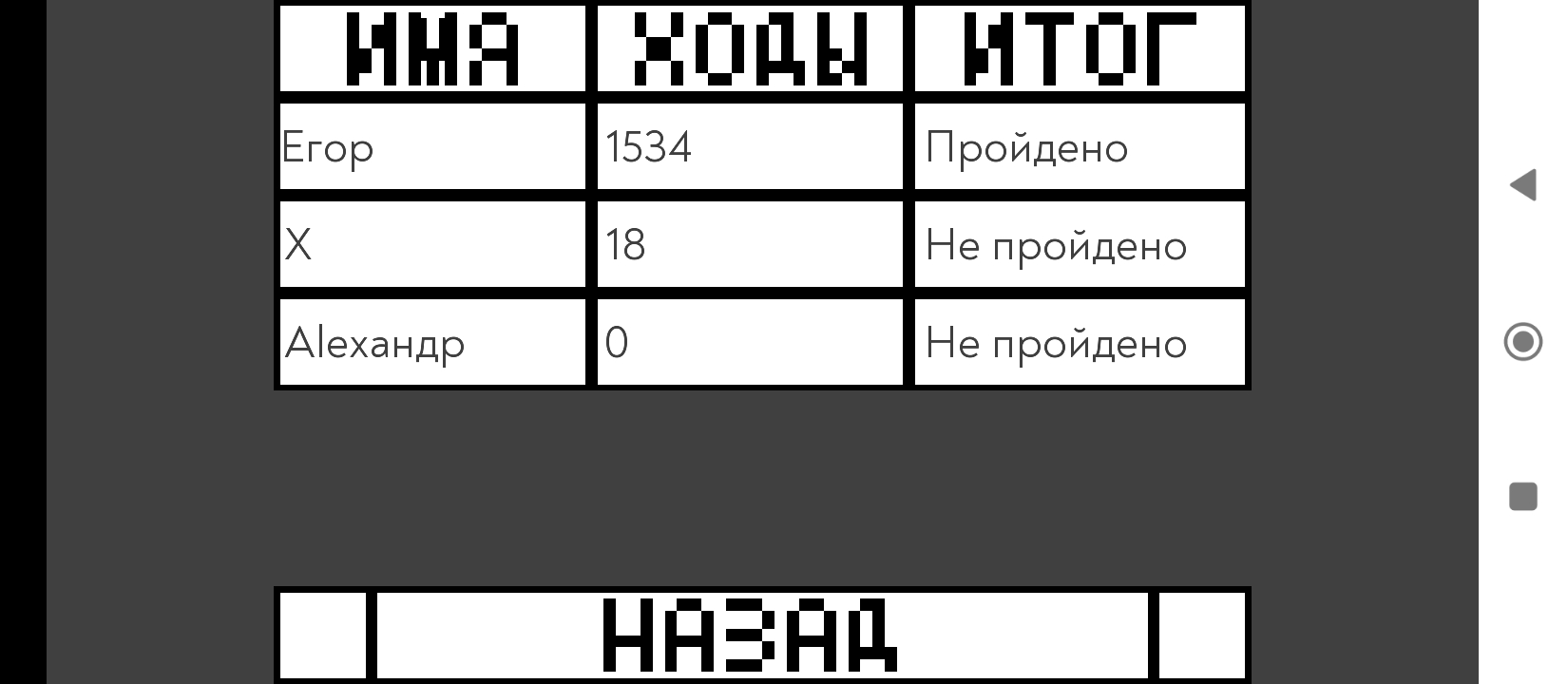


Рисунок 5. Победа игрока.

Рисунок 6. Проигрыш игрока.

