Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации
Бурятский институт инфокоммуникаций (филиал) Федерального государственного бюджетного образовательного учреждения высшего образования
«Сибирский государственный университет телекоммуникаций и информатики» в г. Улан-Удэ
(БИИК СибГУТИ)

Кафедра Информатики и вычислительной техники

Допустить к защите

Зав.каф. _____И.Б. Елтунова

РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА

По дисциплине: «Современные технологии программирования»

Разработка мобильной игры “Арканоид” на Java

Пояснительная записка

Студент _____ /Дондупова О.Б./

Факультет ИВТ Группа И-101

Руководитель _____ /Извеков Я.О. /

Улан-Удэ

2024 г.

СОДЕРЖАНИЕ

СОДЕРЖАНИЕ 2
ВВЕДЕНИЕ 3
ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 4
ПРАКТИЧЕСКАЯ ЧАСТЬ 5
2.1. Структура проекта 5
2.2. Описание используемых методов и алгоритмов 5
ЗАКЛЮЧЕНИЕ 10
СПИСОК ЛИТЕРАТУРЫ 11

ВВЕДЕНИЕ

Целью данной расчётно-графической работы является разработка игры «Арканоид» под Android и демонстрация функционала языка программирования Java.

«Арканоид» - аркадная игра является собирательным названием для класса подобных игр, история которого насчитывает более 50 лет.

Суть игры заключается в следующем: игрок направляет небольшую горизонтально двигающуюся платформу, чтобы не допустить касания мяча нижней стороны экрана. Цель игры заключается в том, чтобы выбить все блоки, которые находятся на противоположной стороне поля с минимально возможной потерей жизней. Реализации игры отличаются друг от друга. Некоторые игры имеют различные уровни сложностей, бонусы, выпадающие из разрушенного блока, также могут присутствовать блоки, разных уровней разрушения. Наличие или отсутствие какой-либо характеристики определяется только желанием разработчика.

Мобильное приложение реализовано с помощью технологий объектно-ориентированного программирования.

В данной реализации игры будет присутствовать механизм засекания времени игры, а также укрепленные блоки, которые разрушаются только со второго раза.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

«Арканоид» является представителем жанра «Аркады». Игры данного жанра имеют несложный игровой процесс, не меняющийся с течением времени. В большинстве аркадных игр для достижения результатов игроку нужно проявлять хорошую реакцию. Обычно в таких играх развита система бонусов: начисление очков, временное улучшение характеристик персонажа и т.д.

За счет простоты пользовательского интерфейса и правил игры, «Арканоид» быстро набрал популярность не только среди игроков, но и среди разработчиков. Данную игру часто выбирают в качестве учебного проекта, поскольку в процессе создания разработчик приобретает такие основные навыки, необходимые для создания игр, как:

- Использование объектно-ориентированного программирования на практике
- Отрисовка игрового интерфейса
- Обрабатывание действий игрока
- Обработка взаимодействия игровых объектов

Данная мобильная игра должна включать в себя следующий набор функциональных требований:

- Ряды блоков должны быть разных цветов для визуального разделения. Некоторые из них должны быть более прочными (ломаются от более одного попадания);
- Должны быть реализованы счетчики игровых жизней и заработанных очков;
- Должен быть реализован механизм счетчика времени, пройденного с начала игры;
- Скорость платформы при длительном нажатии игрока должна быть больше скорости платформы при одиночном нажатии на экран
- Угол отскакивания мяча должен изменяться в зависимости от скорости платформы

ПРАКТИЧЕСКАЯ ЧАСТЬ

Разработка проекта велась на языке программирования Java в среде разработки Android Studio.

2.1. Структура проекта

Приложение имеет 4 основных класса: AppView, Ball, Brick и Paddle (рисунок 1).

Класс AppView является управляющим классом и наследуется от класса SurfaceView, который предназначен для динамичной графики. Здесь прописана вся логика программы.

Класс Ball создает игровой мяч и описывает параметры игрового мяча, например, его размеры и скорость,

Класс Brick создает блок и содержит такие параметры, как цвет блока, его видимость, прочность и т.д.

Класс Paddle создает платформу и содержит такие параметры длина и высота платформы, ее скорость, состояние.

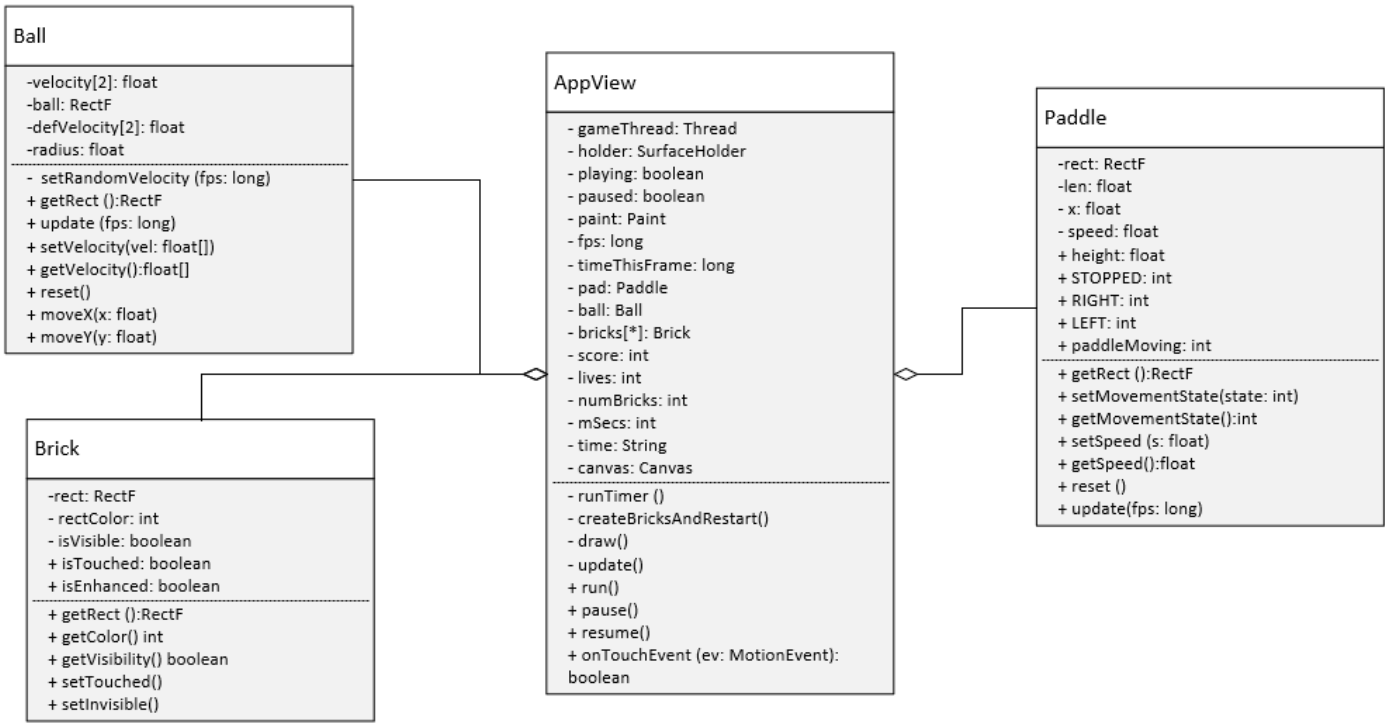

Рисунок 1. Диаграмма классов

2.2. Описание используемых методов и алгоритмов

Алгоритм работы программы представлен на рисунке 2.

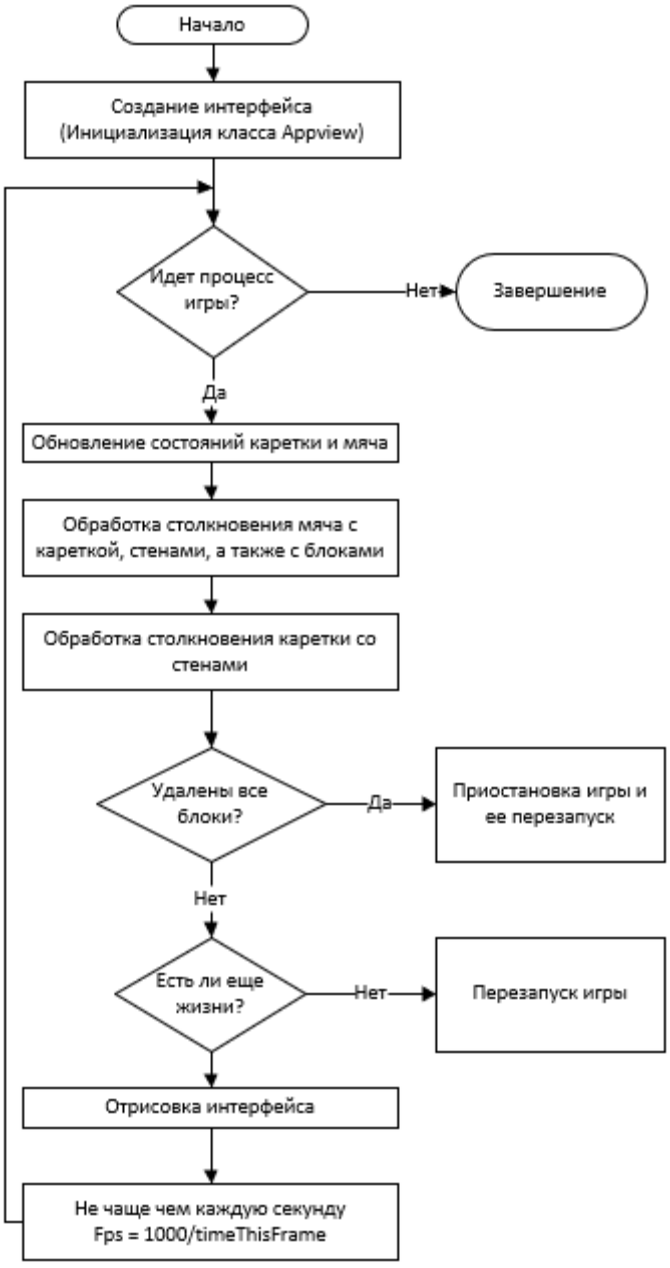

Рисунок 2. Блок-схема алгоритма

В MainActivity устанавливаем содержимое окна объект класса AppView. Конструктор класса инициализирует переменные, необходимые для отрисовки игры, создает мяч , платформу и блоки, а также запускает секундомер.

Ключевыми методами класса AppView являются run(), update(), draw(), отвечающие за логику программы.

Метод run() – в цикле while вызывает обновление состояния объектов (update()), и отрисовку интерфейса (draw()), вычисляя при этом не чаще 1 секунды частоту смены кадров, до тех пор, пока происходит процесс игры, т.е. пока playing = true.

Частота смены кадров используется для вычисления передвижения мяча и платформы и вычисляется по формуле:

$$\text{fps} = \frac{\text{Количество\_кадров}}{\text{Текущее\_время} - \text{время\_с\_начала\_отсчета}} = \frac{1000}{timeThisFrame}$$

Метод update() вызывает методы update() объектов Paddle и Ball, передавая вычисленную fps. Столкновение с блоками обрабатывается так (рис. 3):

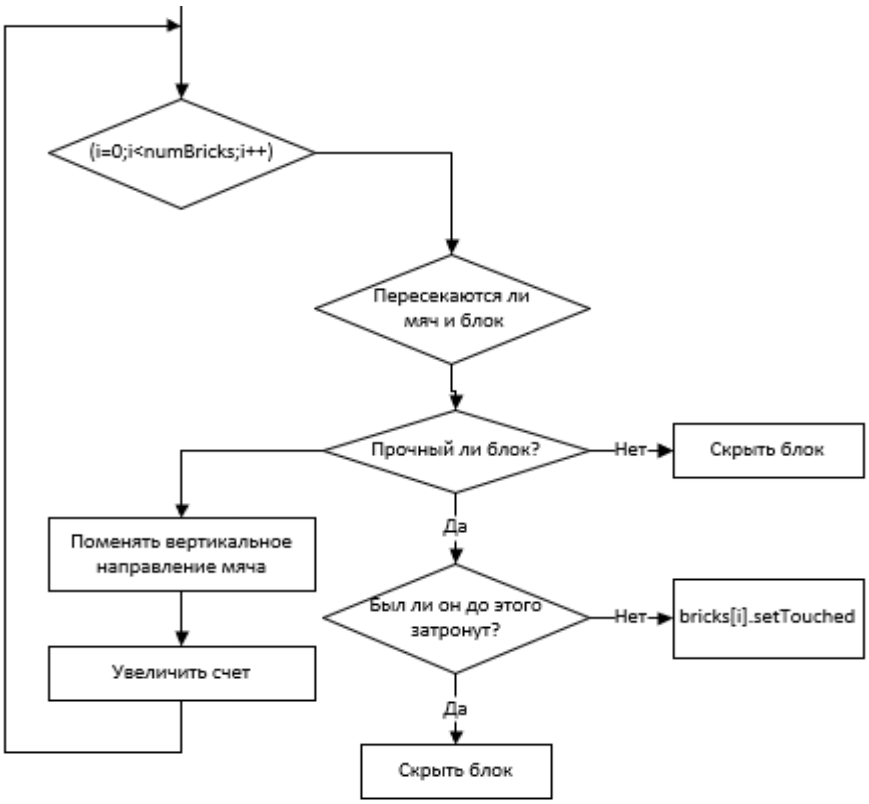

Рисунок 3. Блок-схема алгоритма обработки столкновения мяча с блоками

Столкновение мяча и платформы обрабатывается по следующему алгоритму (рис. 4):

Рисунок 4. Блок-схема обработки столкновения платформы и мяча

Отрисовка интерфейса производится следующим образом: холст блокируется объектом класса SurfaceHolder, который является интерфейсом для взаимодействия приложения с Surface. Далее рисуются объекты игры с учетом их обновленного расположения и состояния. После проверяется, уничтожил ли игрок все блоки или потерял все жизни. В конце холст разблокировывается и отправляется на экран.

Поскольку SurfaceView предоставляет отдельную область для рисования, действия с которой должны быть вынесены в отдельный поток приложения, то необходимо создать поток игры gameThread, а также дожидаться его в случае паузы игры и начала потока в случае возобновления.

Блоки разных цветов легко реализуемы с использованием дополнительного массива, хранящего значения цветов, который объявляется до создания блоков в методе createBricksAndRestart():

```
int[] colors = new int[]{Color.RED, Color.YELLOW, Color.GREEN};
```

Тогда при создании блока, в его конструктор передается значение цвета:

```
for (int row = 0; row < 3; row++) {
        bricks[numBricks] = new Brick(row, column, brickWidth, brickHeight,colors[row],
row == 2);
        numBricks++;
}
```

При этом в конструктор также передается логическое значение isEnhanced, значение true которого свидетельствует о том, что данный блок прочный (в данном случае isEnhanced = true, только когда блок принадлежит 3 ряду).

Интерфейс приложения перед запуском игры выглядит таким образом:

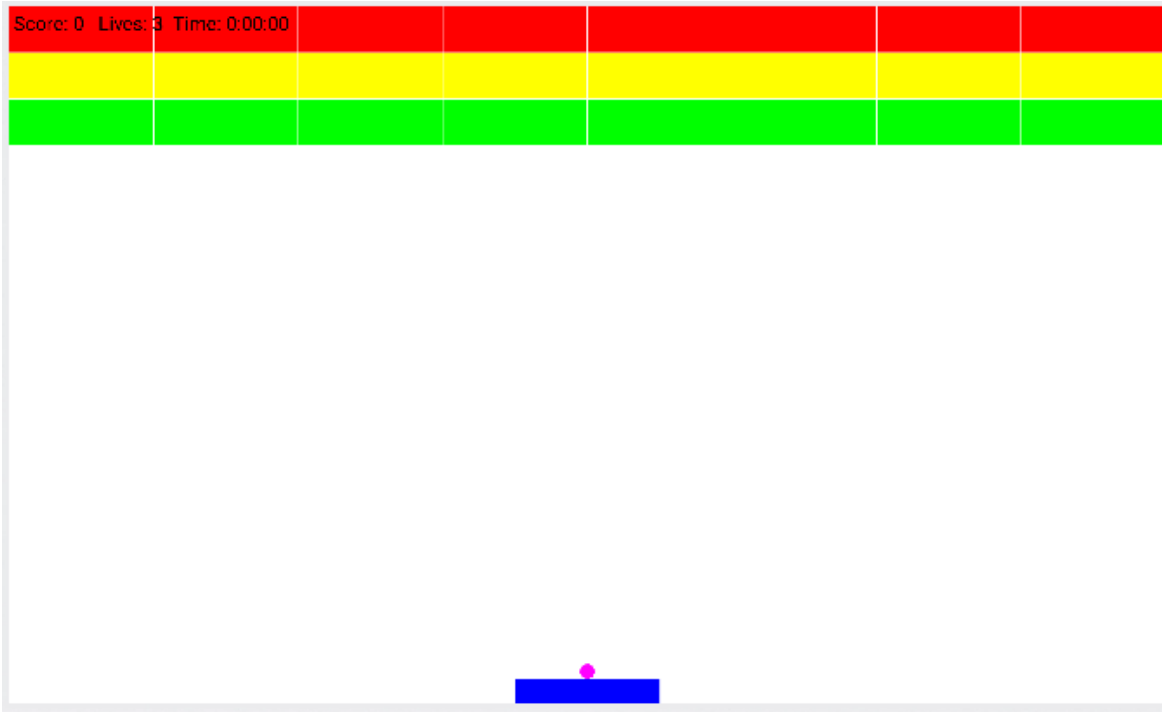

Рисунок 5. Начальный вид приложения

Реализованный функционал программы отлично демонстрирует рисунок 6. Вывод игрового счета, счетчика жизней и времени в левом верхнем углу. Блоки первого ряда снизу являются более прочными, чем остальные. При первом касании мяча, они меняют цвет на серый и разрушаются при втором касании.

Рисунок 6. Демонстрация работы программы

ЗАКЛЮЧЕНИЕ

В результате выполненной работы была реализована мобильная игра «Арканоид». Были представлены диаграмма классов и блок-схемы основных алгоритмов, образующих функционал программы.

В процессе разработки данной программы были приобретены навыки работы с потоками, генераторами рандомных чисел, SurfaceView, и таймером, также был освоен метод обработки нажатий пользователя, в том числе и длительных, что будет полезно в дальнейшей разработке программного обеспечения.

Таким образом, в ходе выполнения данной расчётно-графической работы был разработан программный продукт, в котором были учтен все функциональные требования, предъявленные в начале работы, также, при необходимости, программа может дополняться новыми компонентами.

СПИСОК ЛИТЕРАТУРЫ

1. Березовская Ю.В. Введение в разработку приложений для ОС Android : учебное пособие / Ю. В. Березовская, О. А. Юфрякова, В. Г. Вологдина [и др.]. — 3-е изд. — Москва : ИНТУИТ, Ай Пи Ар Медиа, 2021. — 427 с.— Текст: электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/102000.html (дата обращения: 10.12.2023). — Режим доступа: для авторизир. пользователей
2. Блох, Дж. Java. Эффективное программирование / Дж. Блох ; перевод В. Стрельцов ; под редакцией Р. Усманов. — 2-е изд. — Саратов : Профобразование, 2019. — 310 с — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/89870.html (дата обращения: 20.12.2023). — Режим доступа: для авторизир. пользователей
3. Колисниченко Д. Н. Программирование для Android. Самоучитель. - 3-е изд., перераб. и доп. - СПб.: БХВ-Петербург, 2021. - 288 с.: ил.
4. Нужный, А. М. Разработка мобильных приложений на языке Java с использованием Android Studio : учебное пособие / А. М. Нужный, Н. И. Гребенникова, В. В. Сафронов. — Воронеж :ВГТУ, ЭБС АСВ, 2020. — 93 с. — ISBN 978-5-7731-0906-8. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/111479.html (дата обращения: 20.12.2023). — Режим доступа: для авторизир. Пользователей
5. Льюис, Ш. , Данн М. Нативная разработка мобильных приложений / Льюис Ш. , Данн М. , пер. с англ. А. Н. Киселева. - Москва : ДМК Пресс, 2020. - 376 с. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : https://www.studentlibrary.ru /book/ISBN9785970608456.html (дата обращения: 18.12.2023). - Режим доступа : по подписке.