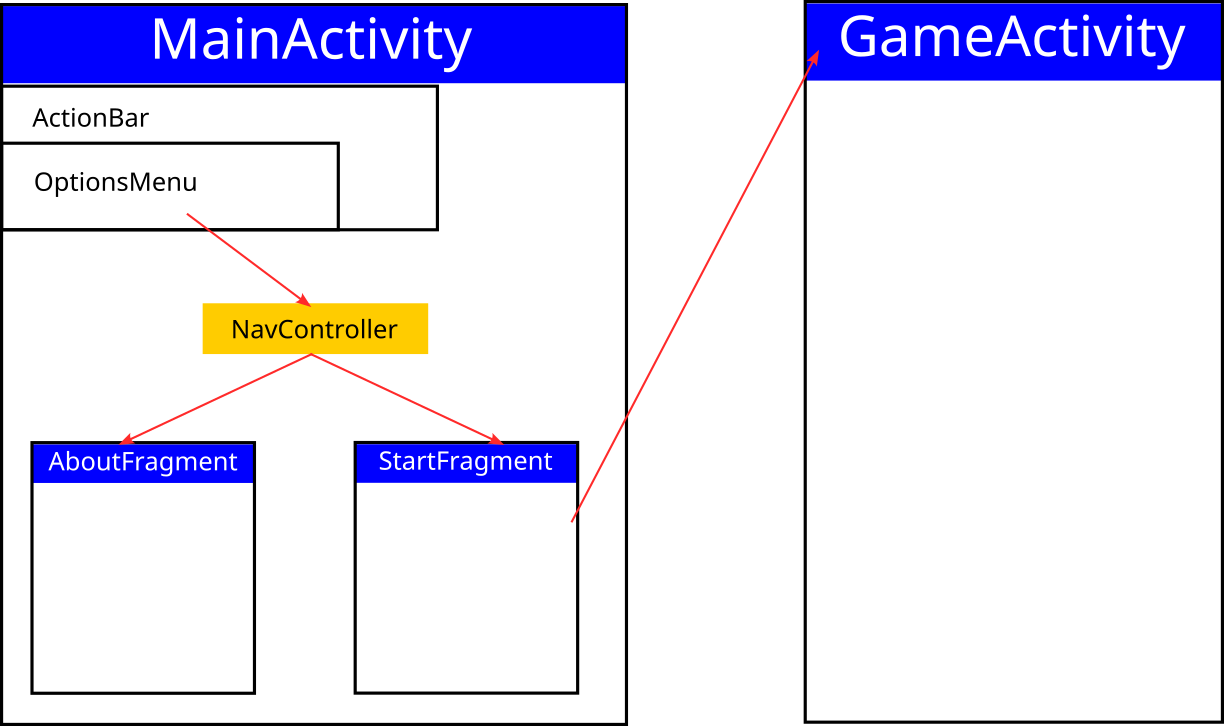
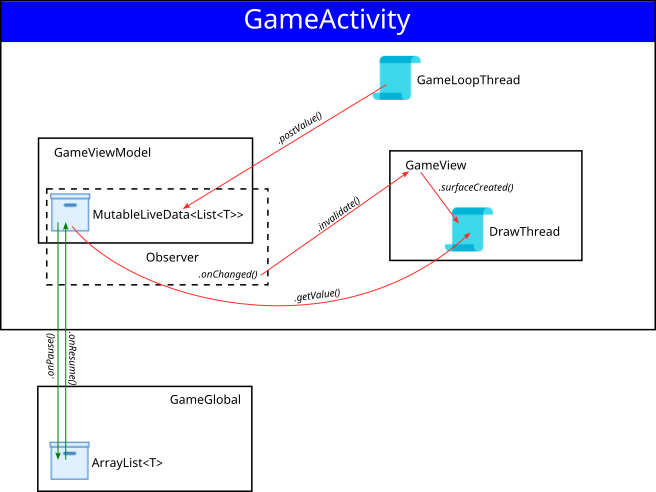
В качестве основы интерфейса взят стандартный шаблон Android приложения – Basic Activity. Структура интерфейса приложения состоит из основного Activity (MainActivity), в котором находится ActionBar с заголовком и меню, и навигационный контроллер (NavController), управляющий фрагментами (Fragments). Фрагмент StartFragment вызывается при запуске приложения и содержит в себе поля для ввода данных об игроке и насекомых, а также кнопку запуска игры. Фрагмент AboutFragment вызывается по нажатию пункта меню About и содержит информацию о приложении и ссылки на источники ресурсов. Само игровое поле реализовано в отдельном Activity (GameActivity), выводящемся в полноэкранном режиме.



*Рисунок 1. Элементы интерфейса приложения.*

Игровая механика, т.е. перемещение насекомых, обработка столкновений друг с другом и границами экрана, проверка попадания в насекомое, обратный отсчёт времени и подсчёт очков, реализована в отдельном потоке GameLoopThread. Этот поток запускается при активизации игрового поля (GameActivity.onResume()) и останавливается при его приостановке (GameActivity.onPause()).



*Рисунок 2. Взаимодействие потоков с данными раунда игры.*

Отрисовка объектов на экране также происходит в отдельном потоке. Применение фонового потока для рисования возможно благодаря использованию SurfaceView (класс GameView). При каждом создании поверхности рисования запускается фоновый поток DrawThread, выполняющий рисование насекомых и вывод обратного отсчёта и набранных очков в режиме OSD (наложение на игровое поле).

Очевидно, что фоновые потоки оперируют одним и тем же набором данных (коллекцией насекомых). Обеспечение доступа к ней реализовано посредством объекта класса ViewModel (GameViewModel). Отличительной особенностью данного класса является то, что его объект не уничтожается до тех пор, пока не будет окончательно уничтожено породившее его Activity. Внутри GameViewModel инкапсулированы объекты класса LiveData, непосредственно хранящие данные (точнее, MutableLiveData, предоставляющий метод postValue (T value) для потокобезопасного обновления данных). LiveData предоставляет механизм наблюдателя (Observer), позволяющий выполнять какое-либо действие при изменении данных.

Таким образом, получаем следующую схему взаимодействия потоков. Игровой процесс, выполняемый в потоке GameLoopThread, модифицирует коллекцию насекомых, инкапсулированную в объекте класса MutableLiveData. При этом, наблюдатель за данным объектом вызывает метод onChanged(), обнаружив изменения в данных. В данном методе вызывается перерисовка интерфейса (метод gameView.invalidate()). Соответствующий SurfaceHolder пересоздаёт поверхность, что, в свою очередь, порождает поток отрисовки DrawThread, где осуществляется доступ всё к той же коллекции насекомых и вывод их на экран.

Однако, приложение может быть приостановлено пользователем, при этом все потоки будут остановлены, а объект ViewModel, в конце концов, уничтожен. Чтоб не допустить потери данных незавершённого раунда игры, при остановке GameActivity данные сохраняются в глобальном объекте GameGlobal, а при создании GameActivity происходит их восстановление.

Помимо коллекции насекомых, к данным раунда игры относится также текущее значение таймера и набранные очки (эти данные не показаны на рисунке для упрощения).

Насекомые описаны классом Mob, который хранит информацию о местоположении насекомого (координаты x-y), векторе движения (в градусах, 0 – вертикально вверх, 180 – вертикально вниз), флаги состояния (жив/мёртв), и роде насекомого. Род описывается классом MobSpecies и в данной реализации имеет только одно значимое поле – изображение насекомого (паук или оса).

Жизненный цикл насекомого состоит из трёх стадий, управляемых двумя флагами (isAlive и isDying): насекомое живое (isAlive==true & isDying==false), насекомое умирает (isAlive==false & isDying==true) и насекомое мертво (isAlive==false & isDying==false). Переход от живого к умирающему происходит вызовом метода Kill(), переход от мертвого к живому – вызовом метода Revive(). Переход от умирающего к мертвому осуществляется последовательным вызовом метода DecreaseLife() до тех пор, пока количество этих вызовов не превысит значение заданной константы DYING\_DURATION. Практический смысл стадии умирания состоит в том, что в этот период вместо изображения насекомого, заданного его родом, выводится спрайт, символизирующий попадание (красное пятно).

При запуске раунда игры (т.е. создании GameActivity при пустом GameGlobal) генерируется коллекция насекомых, в случайном порядке расположенных по краям экрана. Род насекомого также выбирается случайно. Первоначальный вектор движения насекомого перпендикулярен краю экрана, у которого насекомое сгенерировано, ± случайное отклонение в пределах 45⁰. При генерации каждого очередного насекомого проверяется, не пересекаются ли его границы с уже существующими, т.е. предотвращается полное или частичное перекрывание объектов. Если совпадение границ обнаружено, предпринимается новая попытка генерации. Однако, количество таких попыток ограничено во избежание попадания в бесконечный цикл генерации при очень большом количестве насекомых, которые физически не могут разместиться без перекрытия.

Игра использует класс настроек (SharedPreferences) для сохранения своих параметров, таких как количество насекомых и рекорд очков, между запусками. Работа с настройками осуществляется в стартовом фрагменте StartFragment.

Также в приложении используются вспомогательные классы. Так, RandGenerator имеет два статических метода, для генерации случайного числа в заданном интервале и подбрасывания монеты с неравной вероятностью выпадения (это метод используется для принятия решения о «воскрешении» убитых насекомых).

Класс SoundManager является оберткой над объектом класса SoundPool и предоставляет статические методы для проигрывания звуков (реализованы два метода, звук попадания и звук промаха).

**Скриншоты приложения.**

**Список литературы**

1. Документация для разработчиков Android [Электронный ресурс] – Режим доступа: <https://developer.android.com/docs>

2. Start Android – учебник по Android для начинающих и продвинутых [Электронный ресурс] – Режим доступа: <https://startandroid.ru/ru/>

3. Руководство по языку программирования Java [Электронный ресурс] – Режим доступа: <https://metanit.com/java/tutorial/>