Санкт-Петербургский государственный политехнический университет

Институт компьютерных наук и технологий

**Кафедра «Компьютерные системы и программные технологии»**

**КУРСОВОЙ ПРОЕКТ**

**Android приложение “Пазл”**

по дисциплине «Проектирование мобильных приложений»

Выполнил

студент

гр.3530901/70202 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д. В. Павлов

(подпись)

Преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. Н. Кузнецов

(подпись)

«\_\_\_» декабря 2019 г.

Санкт-Петербург

2019

Санкт-Петербургский государственный политехнический университет

**ЗАДАНИЕ**

**НА ВЫПолнение Курсового проекта**

студенту группы 3530901/70202 Павлову Даниилу Вячеславовичу

(номер группы) (ФИО)

***1. Тема проекта (работы):*** Android приложение “Пазл”

***2. Срок сдачи законченного проекта (работы)*** « » декабря 2019 г.\_\_\_\_\_\_\_

***3. Исходные данные к проекту (работе)***: условие задания для выполнения курсового проекта

***4. Содержание пояснительной записки:*** введение, основная часть, заключение, список использованных источников, приложения.

***Дата получения задания***: « » октября 2019 г.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_А. Н. Кузнецов\_\_

*(подпись) (инициалы, фамилия)*

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_Д. В. Павлов\_

*(подпись студента) (инициалы, фамилия)*

\_\_\_\_\_\_\_\_\_\_\_

*(дата)*

**СОДЕРЖАНИЕ**

|  |  |  |
| --- | --- | --- |
| Введение………………. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 4 |
| Выполнение работы…………………………………………………… | | 5 |
| 1. Activities ………………………………................................... | 5 | |
| 1.1. HomeActivity....................................................................... | 5 | |
| 1.2. AboutActivity........................................................................ | 6 | |
| 1.3. LevelChoiceActivity............................................................. | 7 | |
| 1.4. Standard & Hard Activities…..………………....................... | 8 | |
| 1.5. PuzzleActivity....................................................................... | 9 | |
| 1.6. WinActivity…………………............................................... | 10 | |
| 2. Прочие классы............................................................................. | 11 | |
| 2.1. ImgViewModel………………………………………………  2.2. Adapter……………………………………………………….  2.3 MusicService………………………………………………….  2.4 HomeWatcher………………………………………………… | 11  11  11  11 | |
| Тестирование…………………………………………………………...12  Заключение…………………………………………………………….13  Список используемых источников…………………………………...14 |  | |
|  | |  |
|  | |  |
|  | |  |
|  | |  |

**введение**

Темой данного курсового проекта является написание мобильного приложения “Пазл” на Android. Мною было обозначено несколько требований для реализуемого приложения:

1. Футбольная тематика (от собираемых картинок с игроками FC Chelsea, до цветового оформления приложения в цветах клуба, а также иконка);
2. 2 уровня сложности – standard (пазл собирается из 9 деталей) и hard (пазл собирается из 16 деталей). Каждый уровень сложности включает в себя 3 картинки и все они отличаются друг от друга. Картинки можно менять в свободном порядке (пример – левый верхний квадрат можно поменять местами с правым нижним);
3. Внедрение собственной написанной композиции, которая будет играть на фоне при переходе в любое activity и контролируется из домашней.

Далее в отчете будут приведены другие немаловажные особенности и дополнения.

Среда разработки – Android Studio 3.5.3.

Минимальный поддерживаемый API – 21.

**Выполнение работы**

В данном разделе будет приведено описание всех компонентов приложения, их роль и особенности.

1. Activities

1.1. HomeActivity

Стартовое activity. Из него мы можем попасть либо в раздел выбора уровней (LevelChoice Activity), либо в информацию о приложении (About Activity). Переходы реализованы с помощью интентов и совершаются при нажатии на соответствующие кнопки, а выход из приложения возможен при нажатии на аппаратную кнопку back (лишь в этом Activity).   
В данном Activity используется Toolbar, который содержит item sound. Он отвечает за воспроизведение музыки, которая включается сразу же при запуске приложения. Item сопровождается соответствующим drawable ресурсом (Рис.1.1.1):

  
Рис.1.1.1.

Отключение воспроизведения музыки реализуется тапом по иконке, после чего она меняет своей drawable ресурс (Рис.1.1.2):

  
Рис.1.1.2.

При сворачивании приложения или переходе в Tasks музыка уходит на паузу, а при нажатии на item останавливается совсем\начинается воспроизведение с самого начала. Так как Toolbar реализован лишь в данной Activity, то, соответственно, включить\отключить музыку можно лишь здесь и данный выбор будет соответственно использован в последующих activities (либо воспроизводим, либо нет). Подробнее механизм работы воспроизведения будет разобран в описании класса MusicService.

Внешний вид иконки приложения представлен на Рис.1.1.3.:  


Рис.1.1.3.

Внешний вид Activity Home представлен на Рис.1.1.4.:



Рис.1.1.4.

1.2. AboutActivity

Здесь содержится информация об организации, руководителе и разработчике проекта. Также указана версия приложения. Возврат производится с помощью аппаратной кнопки back, как и во всех других последующих activities. Лишь в Home нажатие аппаратной кнопки back приведет к завершению приложения.

Внешний вид Activity About представлен на Рис.1.2.1.:

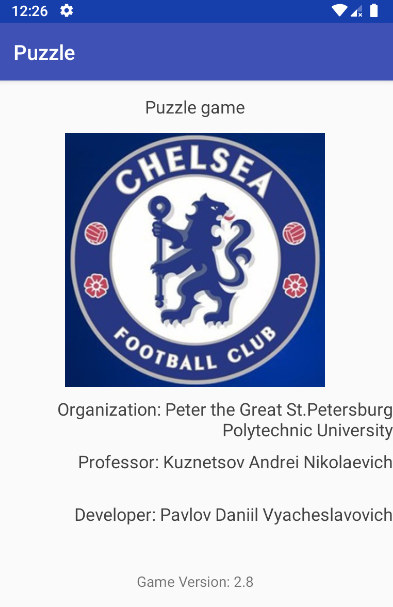


Рис.1.2.1.

1.3. LevelChoiceActivity

Activity выбора уровня. Можем собирать пазл в стандартном режиме (9 кусочков), либо в сложном (16 кусочков).

Внешний вид Activity LevelChoice представлен на Рис.1.3.1.:



Рис.1.3.1.

1.4. Standard & Hard Activities

Визуально данные activities выглядят одинаково, различия лишь в передаваемой интентами информацией.

Примеры интентов для перехода на 2 уровень standard и hard режимов представлены в Листингах 1.4.1-1.4.2 соответственно:

Листинг 1.4.1.(Standard)

Intent toS2;  
toS2 = new Intent(Standard.this, Puzzle.class).  
 setFlags(Intent.*FLAG\_ACTIVITY\_REORDER\_TO\_FRONT*);  
toS2.putExtra("strName", "2");  
startActivity(toS2);

Листинг 1.4.2.(Hard)

Intent toH2;  
toH2 = new Intent(Hard.this, Puzzle.class).  
 setFlags(Intent.*FLAG\_ACTIVITY\_REORDER\_TO\_FRONT*);  
toH2.putExtra("strName", "5");  
startActivity(toH2);

Соответствующие values будут обрабатываться в Activity Puzzle, о котором пойдет речь далее.

Внешний вид Activities Standard & Hard представлен на Рис.1.4.1.:

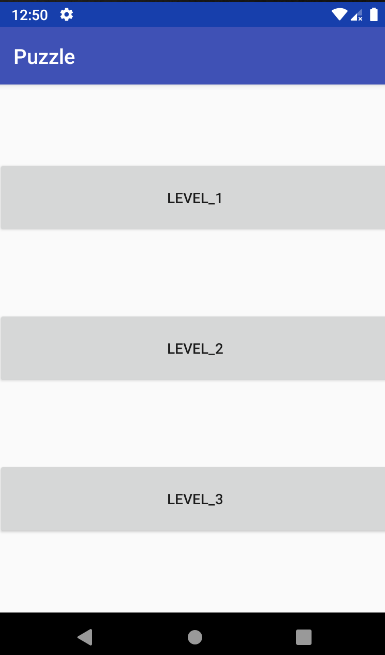


Рис.1.4.1.

1.5. PuzzleActivity

В данном Activity происходит сама игра. Рассмотрим принцип работы на примере “уровень standard 1”.

Как только мы выбрали уровень и нажали соответствующий button, происходит генерация игры. Выше было описано, что каждый интент передает определенный value, в зависимости от выбранного уровня. В данном случае мы передаем значение “1”. Присвоение индексу значения представлено в Листинге 1.5.1.:

Листинг 1.5.1.

Bundle bundle = getIntent().getExtras();  
load(bundle.getString("strName"));  
*index* = Integer.*parseInt*(bundle.getString("strName"));

Метод load отвечает за создание массива маленьких картинок, в соответствии с выбранным индексом, и загрузку целой картинки. Пример для значения “1” представлен в Листинге 1.5.2.:

Листинг 1.5.2.

public void load(String choice) {  
 switch (Integer.*parseInt*(choice)) {  
 case 1:  
 level1 = new int[]{R.drawable.*p1\_1*, R.drawable.*p1\_2*,  
 R.drawable.*p1\_3*, R.drawable.*p1\_4*,  
 R.drawable.*p1\_5*, R.drawable.*p1\_6*,  
 R.drawable.*p1\_7*, R.drawable.*p1\_8*, R.drawable.*p1\_9*};  
 imgShow.setImageResource(R.drawable.*p1*);  
 break;

Была установлена кнопка, с помощью которой можно показывать результат, который должен получится, либо игровое поле. При нажатии надпись на ней соответственно меняется на hide или show.   
Также был задействован класс Chronometer, который выполняет функцию секундомера и показывает текущее затраченное время на прохождение уровня.

Разрезанные картинки устанавливаются в gridView, количество столбцов и рядов в котором зависит от полученного с интентом value (если меньше 4, то делаем 3 колонки, иначе 4). Сама установка происходит с помощью класса Adapter.

Перемещение картинок реализовано в методе swapPrepare и там же проверяется сравнение идеального варианта с текущим (элементы массива сравниваются друг с другом). Как только оба варианта совпадают, секундомер останавливается и с него считывается накопленное значение, так как далее оно будет показано в WinActivity. К текущему же Activity будет применен finish(), так как надобность в нем пропадет.

Также в этом Activity использовались альтернативные ресурсы. Чтобы кнопка и текстовое поле секундомера не разлетались по экрану при повороте, были соответствующим образом оформлены portrait и landscape layouts для activity\_puzzle.xml.

Внешний вид Activity Puzzle в portrait и landscape ориентациях представлен на Рис.1.5.1-4.

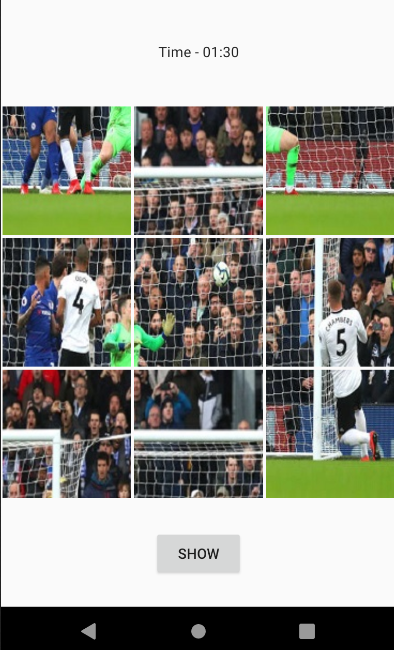
 

Рис.1.5.1. Рис.1.5.2.

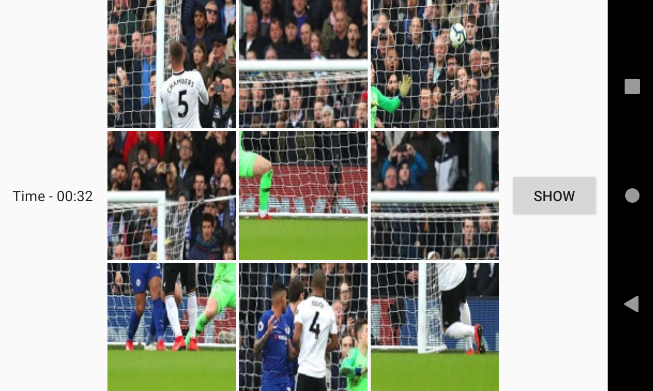
 

Рис.1.5.3. Рис.1.5.4.

1.6. WinActivity

В данное Activity мы попадаем, как только собираем картинку. В зависимости от наличия интернет-соединения, либо его отсутствиям, пользователь либо увидит картинку (загружается по url с помощью библиотеки Picasso), либо увидит надпись “Unfortunately, you don’t have an internet connection and we can’t show your reward ☹”. Использование библиотеки реализует загрузку изображения в фоновом потоке, поскольку тяжеловесные вычисления нельяз представлять в Main Thread.

Сохранение картинки происходит во ViewModel. Нам не придется загружать ее из интернета при изменении конфигурации устройства.

Внешний вид WinActivity представлен на Рис.1.6.1-2.

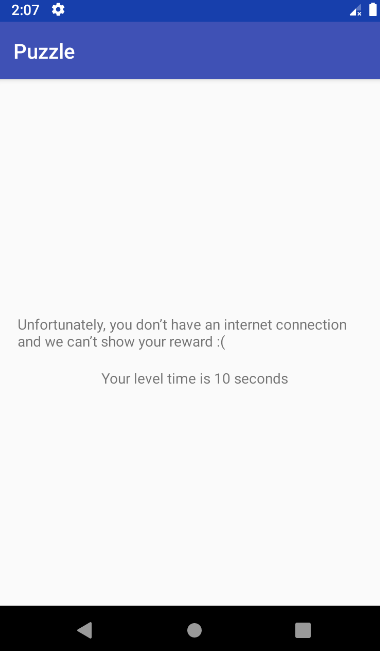
 

Рис.1.6.1. Рис.1.6.2.

2. Прочие классы

2.1. ImgViewModel

Во ViewModel будут храниться следующие поля:

1. MutableLiveData<Bitmap> \_image – здесь будет храниться наша картинка.
2. MutableLiveData<Boolean> \_spinner – булевая переменная, обозначающая необходимость отображения на экране кольца загрузки. Когда мы начинаем скачивать картинку, должен показываться на экране. После загрузки становится невидимым.

Здесь находится метод loadImage, используемый в WinActivity для загрузки приложения.

2.2. Adapter

Используется для форматирования игрового поля в Activity Puzzle.

2.3. MusicService

Был задействован BoundService. Выбор пал на него, потому что он позволяет передавать ответ тому, кто его вызывал. Started сервисы отправляют Broadcast, ничего не зная о своих клиентах.

Данный сервис отвечает за воспроизведение музыки в каждом из Activity. В зависимости от булевой переменной soundIsOff мы будем либо привязывать сервис к Activity (doBindService()), либо отвязывать (doUnbindService()), если изначально был привязан. Также здесь прописаны методы pauseMusic() & resumeMusic(), чтобы при сворачивании приложения музыка уходила на паузу (а не начиналась с начала после возвращения в приложение).

2.4. HomeWatcher

Данный класс был добавлен для того, чтобы, когда вызывается onPause() при переходе в другое Activity, музыка не уходила на паузу, тем самым исключаются прерывания в приблизительно 1 секунду.

В activities будут использоваться методы startWatch() & stopWatch(), где 1 отвечает за мониторинг ситуации в Activity, а 2 будет останавливать этот мониторинг.

ТЕСТИРОВАНИЕ

Было использовано 3 подхода:

1)Monkey Test - программа, которая работает на эмуляторе или устройстве и генерирует псевдослучайные потоки пользовательских событий, таких как касания. Программа является довольно-таки опасной для реального устройства, поэтому применялась исключительно на эмуляторах.   
Тестирование проходило успешно, однако, стоит учесть, что все же это не очень хороший выбор, так-как действия данной программы хаотичны.

2)Тестирование знакомыми – данный подход был поделен на 2 части: приложение предлагалось одногруппникам \ людям, которые знакомы с android разработкой; рядовым пользователям.  
Данный вариант тестирования ошибок на финальном этапе также не выявил.

3)App Crawler – самый полезный вариант из 3. Данная программа производит разные действия с приложением, но, в отличии от 1 подхода, все действия не случайны. Сканирование прекращается автоматически: когда больше нет уникальных действий, которые нужно выполнить; когда происходит сбой приложения или истекло заданное вами время ожидания.

Тестировать с помощью app Crawler просто, потому что это не требует от разработчика написания кода. Более того, вы можете запускать его на разных устройствах для поиска сбоев, визуальных проблем или проблем с производительностью. Данный вариант тестирования тоже был пройден успешно.

В ходе всех 3 подходов было проверено воспроизведение музыки: работает корректно и без пауз при переходе между activities; при сворачивании приложения уходит на паузу, после возвращения в него музыка продолжается с момента установки; при отключенном воспроизведении музыки она нигде не воспроизводится. При поворотах экрана музыка тоже работает корректно, пазл не разрушается и сохраняет все части на своих местах.  
Также приложение успешно прошло испытание на устройствах с разными расширениями экрана, все поля отображаются корректно и на своих местах.

**Заключение**

В ходе выполнения курсового проекта были получены навыки разработки мобильного приложения, а в данном случае – пазла.

Работа включает в себя почти все темы, рассмотренные в курсе, кроме RecyclerView (не было необходимости в его применении), что является, по моему мнению, несомненным плюсом.

К изначально мною обозначенным требованиям, описанным во введении, я смог добавить секундомер, WinActivity (в которым при наличии интернета мы загружаем картинку и сохраняем во ViewModel, а также пишем за сколько прошли уровень), возможность показать пользователю исходную картинку, а также вынес в ToolBar отключение/включение музыки.

В силу отсутствия компьютера, на курсовой проект ушло всего 2 недели, да и то далеко не все дни удавалось им заниматься. В связи с этим присутствуют такие недочеты, как отсутствие SharedPreferences (позволили бы проще управлять воспроизведением музыки по всему приложению), все картинки для 6 уровней находятся изначально в проекте в целом и разрезанном виде (нет базы данных), ну и само разрезание картинки не делается автоматически. Однако, за такой короткий срок все равно удалось реализовать очень многое.

Ссылка на репозиторий GitHub: https://github.com/DaniilPavlov/Courseproject

**Список используемых источников**

1. Android Developers. URL: https://developer.android.com/ ;
2. Stack Overflow. URL: https://stackoverflow.com/ .
3. <https://github.com/andrei-kuznetsov/android-lectures>