Software Engineering Large Practical

Design Document

Ramona Comanescu (s1427590)

November 2016

Contents

1 Introduction

This document describes the software design of Grabble, as presented in the requirements document. The high level activity interaction, utility classes and storage decisions will be outlined.

2 Technologies used

The application was built using API 21, targeting devices with Android 5 (Lollipop).
The external libraries used are: Google Maps, Google Play Services, Google Firebase and XML Parser.

Google Maps The application relies on Google Maps for displaying a map of the Edinburgh Campus.

```
compile 'com.google.maps.android:android-maps-utils:0.4+'
```

Firebase is a Google library that provides cloud storage, NoSQL database access and user authentication.

```
compile 'com.google.firebase:firebase-database:9.4.0'
compile 'com.google.firebase:firebase-auth:9.4.0'
```

Google Services are required by Firebase.

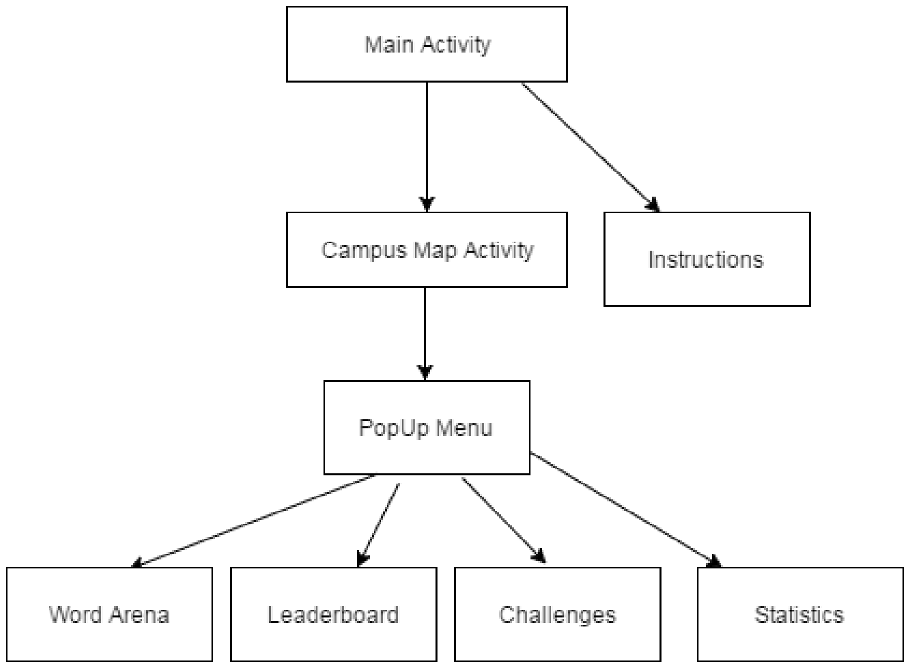
```
compile 'com.google.android.gms:play-services:9.6.1'
apply 'com.google.gms:google-services:3.0.0'
```

XMLPull This library allows parsing the kml files containing the letter markers.

```
import org.xmlpull.v1.XmlPullParser;
import org.xmlpull.v1.XmlPullParserException;
```

3 Overview

Figure 1: Activities interaction



3.1 Main Requirements

The main requirements of the game (collecting letters and constructing words) are met by Campus Map Activity and Word Arena Activity.

Campus Map loads the markers for the current day of the week and saves the state of the map so the same marker can not be collected twice.

Word Arena allows constructing words with the collected letters and validates them against a locally stored dictionary. Each letter has a different value and a final score is calculated.

3.2 Bonus Features

The main features that should add value to the user experience are:

Login This feature allows the user to personalize the game with their unique email address.

Leaderboard This feature introduces the idea of competition into the game, displaying other users' scores against the current player.

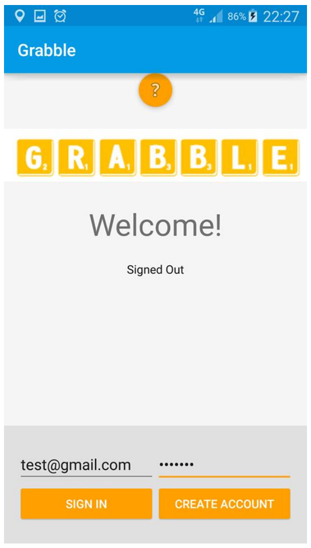
Challenges This feature adds further goals to the game, apart from collecting letters. The user is presented with some challenges to complete in exchange for extra points.

Statistics This feature allows the user to visualize their achievements: past challenges, all their words and their score.

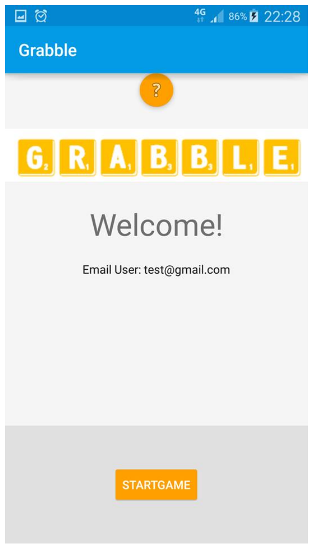
4 Activity interactions and implementations

4.1 Main Activity (Sign in)

The Main Activity presents a sign in interface. There is also an instructions button that presents a pop up with details of how to play the game.



(a) Sign in



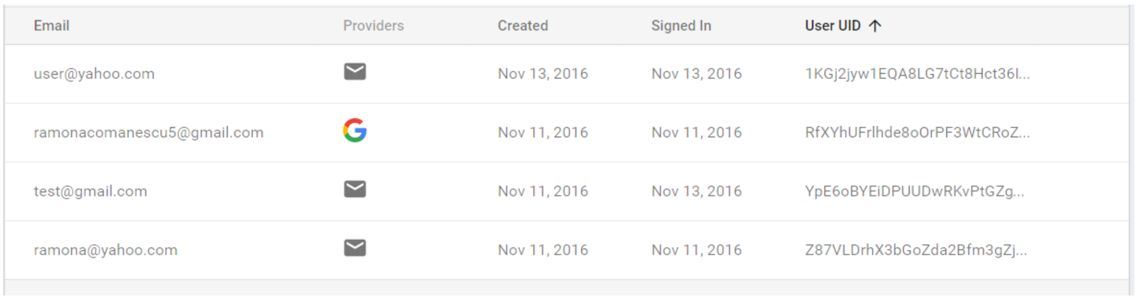
(b) Start

Figure 2: Main Activity

If it is the first time launching the game, the user is presented with a form where they can input an email and password and create an account. After creating an account, the user can Sign in and Start the game.

For account management, an external (Google) Firebase library is used, because it provides cloud user authentication and NoSQL database management. Each account is stored in Firebase.

Figure 3: Registered accounts

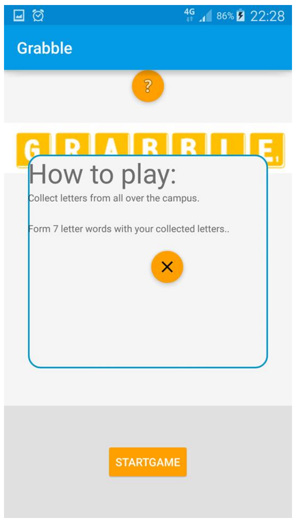


Email	Providers	Created	Signed In	User UID ↑
user@yahoo.com	✉	Nov 13, 2016	Nov 13, 2016	1KGj2jyw1EQA8LG7tCt8Hct36l...
ramonacomanescu5@gmail.com	G	Nov 11, 2016	Nov 11, 2016	RfXYhUFrlhde8oOrPF3WtCRoZ...
test@gmail.com	✉	Nov 11, 2016	Nov 13, 2016	YpE6oBYEiDPUUDwRKvPtGZg...
ramona@yahoo.com	✉	Nov 11, 2016	Nov 11, 2016	Z87VLDrhX3bGoZda2Bfm3gZj...

Note: The design of the game does not allow the user to sign it with multiple accounts on the same device, because most of the game data is stored locally. The implementation can be extended so that all the database operations can be performed in the cloud to allow saving the state of more than one account.

By pressing the Question mark button, an Instructions pop up is launched, displaying game play and rules.

Figure 4: Instructions



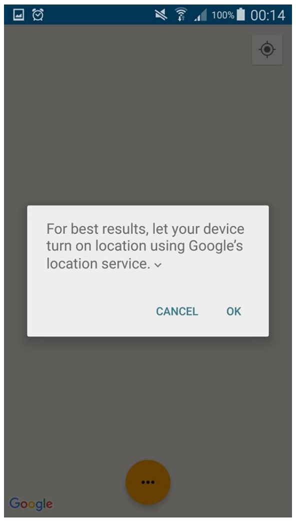
22:28
Grabble
?
GRABBLE
How to play:
Collect letters from all over the campus.
Form 7 letter words with your collected letters..
STARTGAME

By pressing Start Game, a new activity is initialized: Campus Map Activity.

4.2 Campus Map Activity

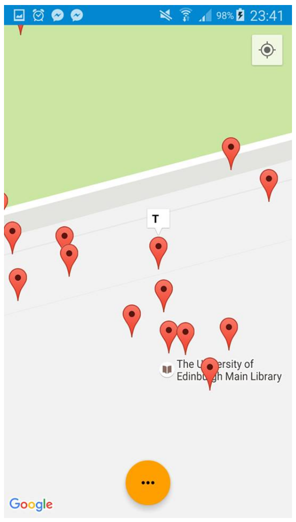
Campus Map first checks if Internet and location are enabled and displays a prompt to enable them if they are not.

Figure 5: GPS prompt


For best results, let your device turn on location using Google's location service.

It then proceeds to load all the markers on the map and shows them around the user's position. Zoom is disabled so the user can only see letters close to their position, for a more interesting game play.

Figure 6: Campus Map with markers



After loading the markers for the first time in the day, they can be saved in Shared Preferences, using their coordinates as a hashcode. This decision has been taking after experimenting with both a database and Shared Preferences.

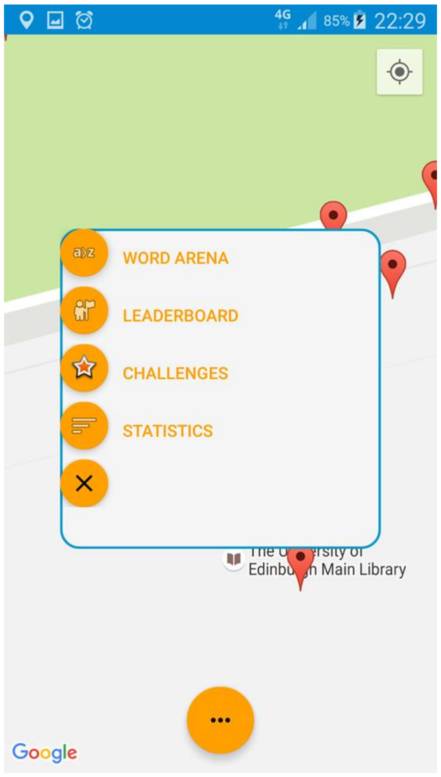
The last time when the markers were downloaded is stored and checked against the current date. If the markers have already been downloaded in the current day, they are just loaded from memory, otherwise they are requested from the inf.ed webpage. When the user comes back on the same day, already collected letters will not appear again.

Each marker has a letter name as a description. By clicking on it, the user can collect a letter, if their position is close enough to the letter. The marker is then removed from the map and from Shared Preferences. The collected letters are then saved in Shared Preferences, with the letter name as the key and with a counter that can increase as they are collected or decrease as they are used.

4.3 Pop Up Menu

Campus Map Activity has a pop up menu (...) that gives the user access to the rest of the game: Word Arena, Leaderboard, Challenges and Statistics.

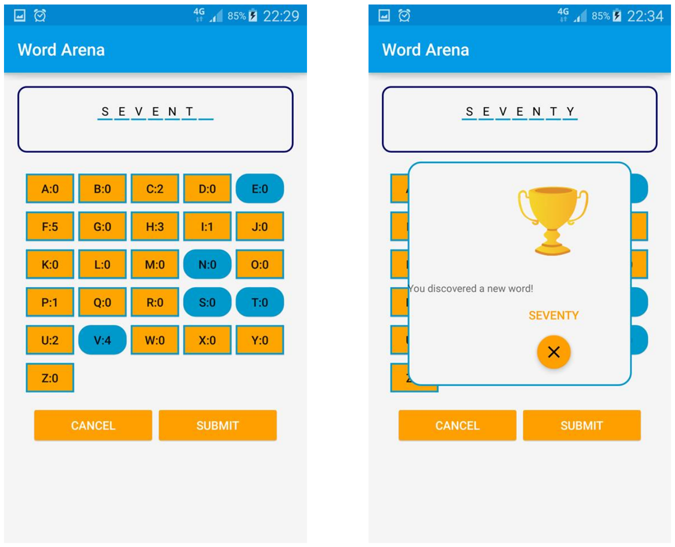
Figure 7: Pop Up Menu



Each button takes the user to a different part of the game (Activity).

4.4 Word Arena

Word arena contains all the letters that the user has collected.



You discovered a new word!

SEVENTY

(a) Word Arena (b) New Word

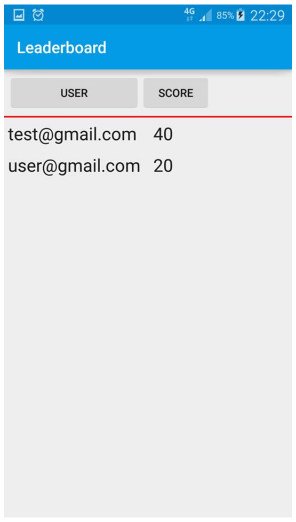
Figure 8: Word Arena Activity

Letters can be selected in order to create a new word. After submitting the word, it is checked against the dictionary and if it is accepted, the user is displayed the number of points than they earned. The dictionary file is stored locally, its content is read line by line and loaded into memory as a hashmap, for faster lookup of words. User's total score is updated in the Firebase database.

4.5 Leaderboard Activity

This activity connects to the Firebase instance in order to get all the scores from the database and order them from highest to lowest.

Figure 9: Leaderboard



test@gmail.com	40
user@gmail.com	20

4.6 Challenges

This activity displays a set of challenges that users can complete for bonuses:

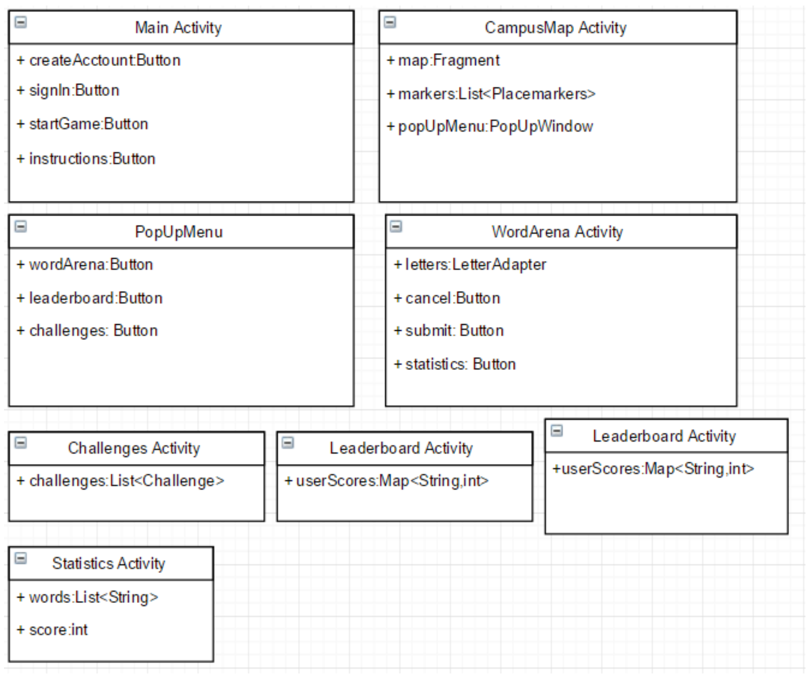
- Play the game 2 days in a row
- Have more than 100 letters in inventory
- Have 5 or more of each letter
- Create at least a word that starts with each letter of the alphabet
- Achieve a score of 2000

Every time a new word is created, the application checks whether a new challenge has been completed.

4.7 Statistics

In this activity, the user can see all the words created in the past and their score.

Figure 10: Activity fields



5 Utility classes

The applications's activities are implemented using the following helper classes:

5.1 Download Letters, KxmlParser and Placemark

When launching the Campus Map activity, the DownloadLetters class will asynchronously download an .kxml document which will be parsed by the Kxml-Parser. A placemark class is needed for working with the results:

```
Placemark(String name, String description, String lat, String lng)
```

5.2 LetterAdapter

This class is used for displaying a grid with all the collected letters in Word Arena activity.

5.3 ScoreAdapter

This class is used for displaying a table with users and their scores in the Leaderboard activity.

5.4 Challenge

Challenge class encapsulates the details of a challenge(id, completed, points worth)

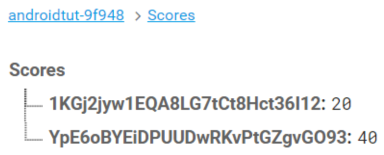
6 Storage

6.1 Cloud storage

In order to keep track of all the players and their scores, a Firebase instance is used, which gives the app user authentication and database access, which can then be used to construct a leaderboard.
A score Database saves each user's score, using an unique id for each user. The database structure is:

UserUid	Score

Figure 11: Score database



```
androidtut-9f948 > Scores

Scores
├─ 1KGj2jyw1EQA8LG7tCt8Hct36I12: 20
└─ YpE6oBYEiDPUUDwRKvPtGZgvGO93: 40
```

6.2 Local storage

Local storage is used for saving the map state, saving collected letters, saving completed challenges and saving completed words. Shared Preferences has proven to be sufficient for the task (without slowing down the game), since the values to be stored are primitive.
The dictionary file is stored on the device.

7 Conclusion

Most of the presented design has been implemented and tested, resulting in a functional application with intuitive interface and appropriate activity interaction.