GitHub Username: studenttwok

DMR-MARC Reader

Description

http://www.dmr-marc.net/ is a website that provides information of radio hams and repeaters using DMR. Hams can talk to each other by setting their mobile radio frequency to the TX and RX frequency of repeaters. The repeater information required in the setup process is provided as a large database by DMR-MARC. This app is going to make use of that database, to provide a download once, use everywhere experience for DMR Hams.
http://www.hamqth.com/ is a website to provides basic info of Hams. The app can query that website on request to get the info of interested hams.

Intended User

Radio Hams using DMR system, who are looking forward to get the repeaters info around the earth easily, and know who are using DMR as well.

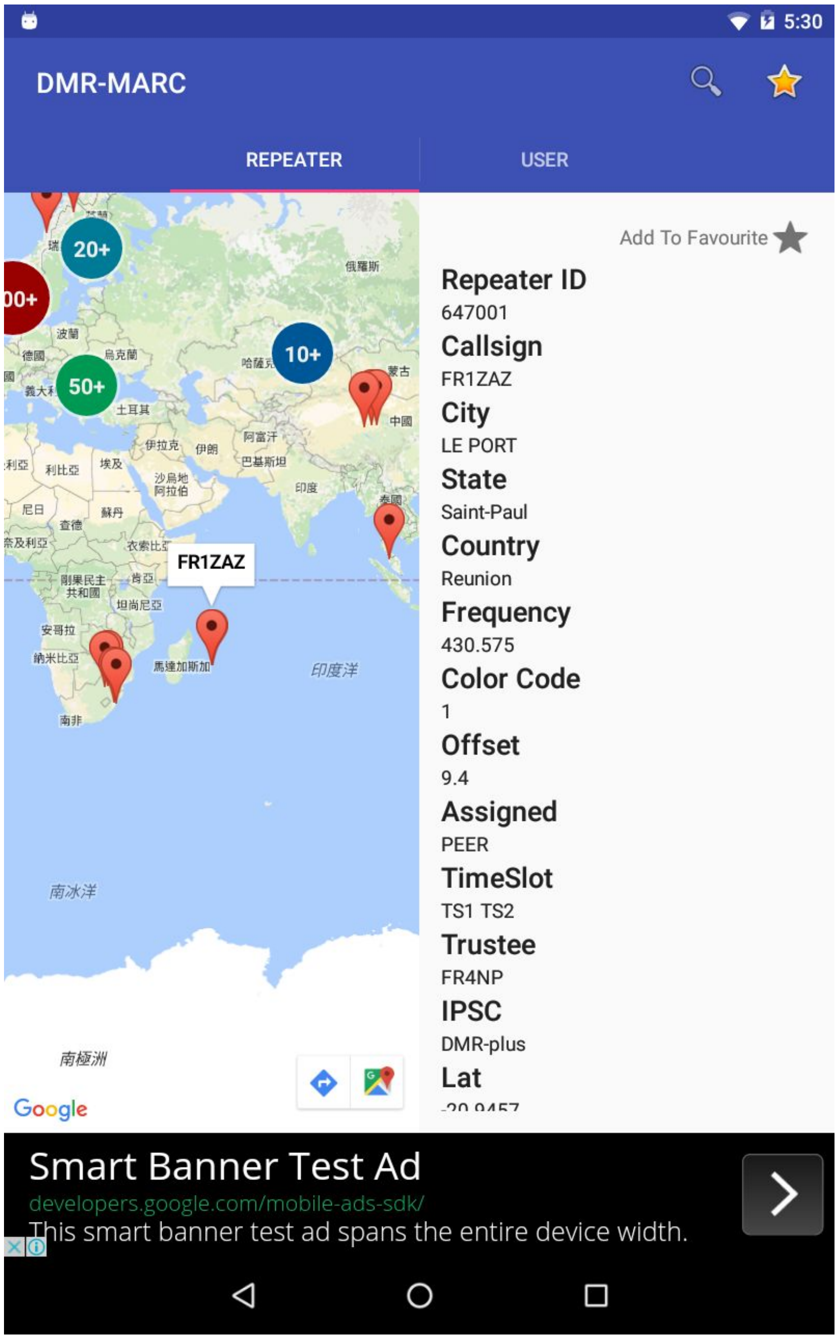
Features

1. User is able to explore repeats in the world map.
2. User is able to get the parameters of each repeater.
3. User is able to search repeaters and users keyword
4. User is able to read the basic info of an DMR hams
5. User is able to ‘star’ their favourite repeaters, so they can access them quickly.
6. Home widget is provided to show their favourite repeaters info.
7. Repeater and user info can be shared.
8. Most of the app data stores in local and core function requires no internet connection to work.

User Interface Mocks

These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Home Screen - Repeaters



DMR-MARC

Add To Favourite

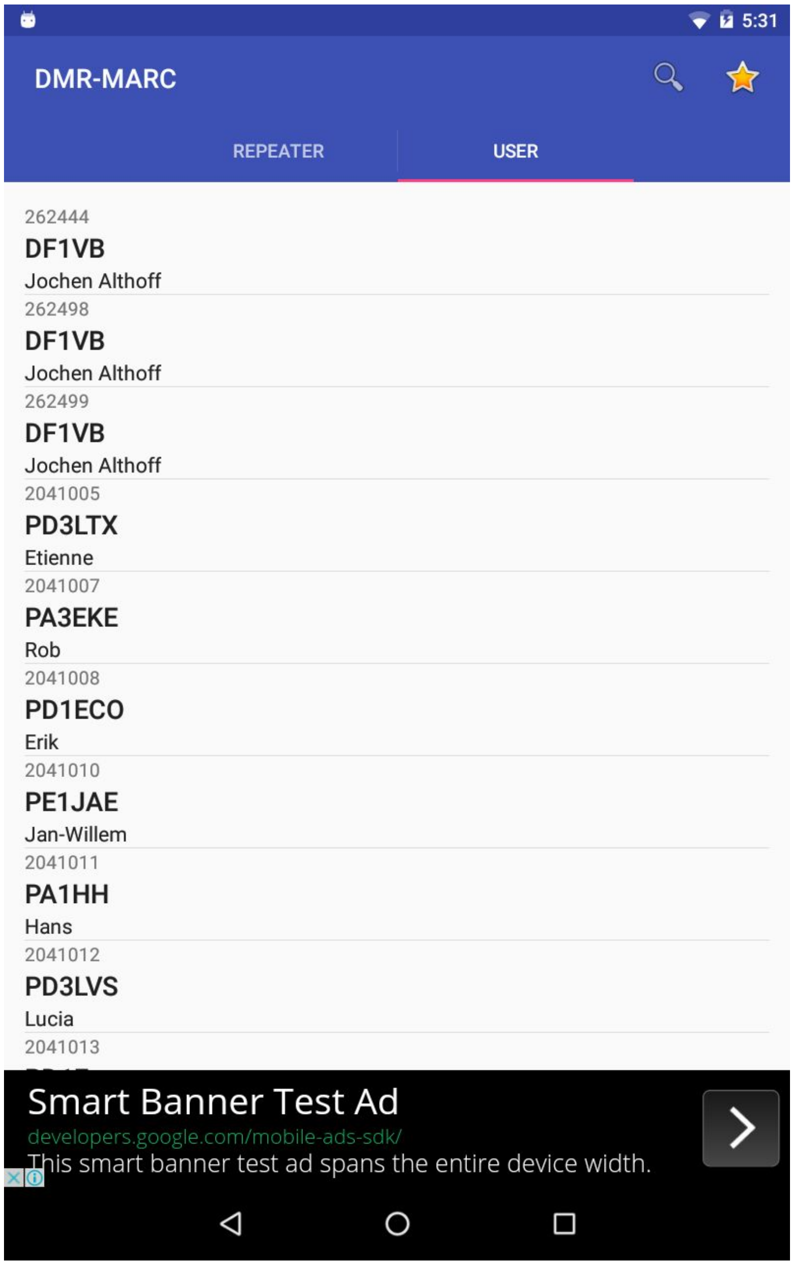
Repeater ID
647001
Callsign
FR1ZAZ
City
LE PORT
State
Saint-Paul
Country
Reunion
Frequency
430.575
Color Code
1
Offset
9.4
Assigned
PEER
TimeSlot
TS1 TS2
Trustee
FR4NP
IPSC
DMR-plus
Lat
-20.9457

Smart Banner Test Ad
developers.google.com/mobile-ads-sdk/
This smart banner test ad spans the entire device width.

This screen shows the repeater information. while use click on the callsign of the repeaters, details of the repeater will be shown on the right, or as a dialog, depends on screen size. User can ‘star’ or ‘unstar’ the repeaters on the top right hand star control.

Keyword search and favourite list can also be performed on the top right hand action buttons.

Home Screen - Users

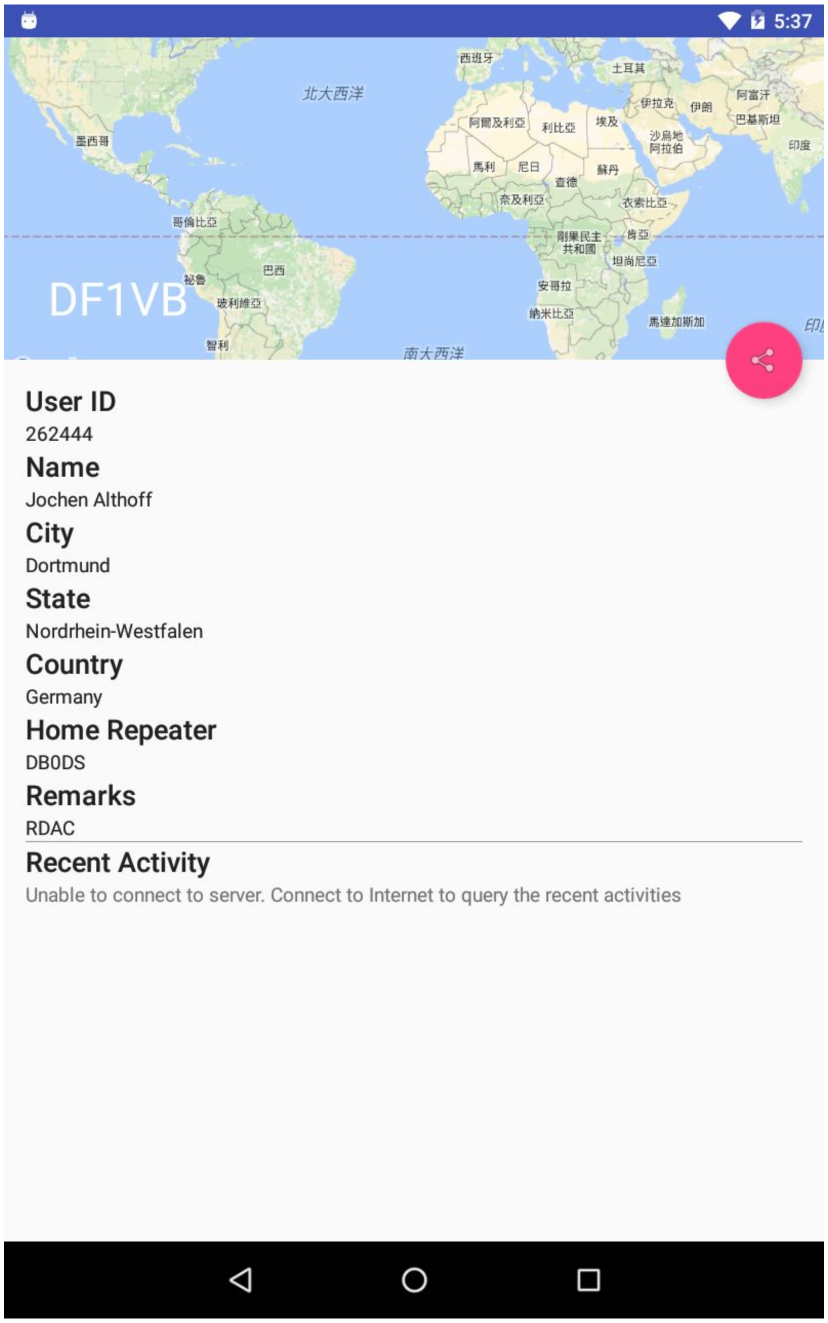


262444
DF1VB
Jochen Althoff
262498
DF1VB
Jochen Althoff
262499
DF1VB
Jochen Althoff
2041005
PD3LTX
Etienne
2041007
PA3EKE
Rob
2041008
PD1ECO
Erik
2041010
PE1JAE
Jan-Willem
2041011
PA1HH
Hans
2041012
PD3LVS
Lucia
2041013

Smart Banner Test Ad
developers.google.com/mobile-ads-sdk/
This smart banner test ad spans the entire device width.

This tab shows the users who are registered in DMR-MARC. They are all DMR user.

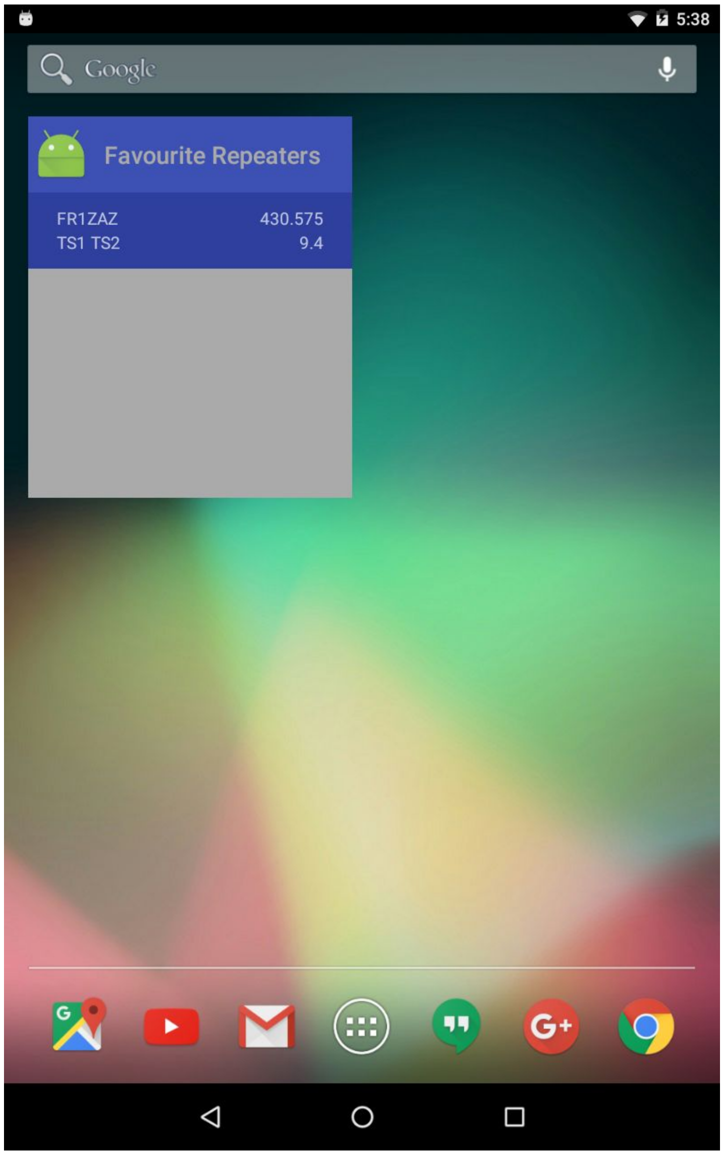
User Details



User ID
262444
Name
Jochen Althoff
City
Dortmund
State
Nordrhein-Westfalen
Country
Germany
Home Repeater
DB0DS
Remarks
RDAC
Recent Activity
Unable to connect to server. Connect to Internet to query the recent activities

When click on the user from the previous user list page, this activity will be shown. It will display the information from DMR-MARC, at the same time, if there is internet connection, it will try to fetch the user activity log and location from hamqth.com.

Widget showing favourite repeaters



The widgets can show the list of repeaters that user have been stared.
Click on the item will show the repeater information.

Key Considerations

How will your app handle data persistence?

One time database download task is required. The data will that be store in local sqlite database. Data provided is used as a data accessing layer to encapsulate the sqlite specific operation.

Two tables will be create, one store repeaters and one store users data. Favourite data is store as a column in repeater tables.

Describe any corner cases in the UX.

If internet connection is down while the app is requesting data from the internet, suitable dialog is needed to alert the user. For example, alert dialog will be shown when the init database cannot be downloaded at the first run.

Describe any libraries you'll be using and share your reasoning for including them.

Volley will be used in this project to make the API requests.
OK Http is used to provide effective http client library. Colley cannot handle that job because the size of database is too big for it.
A maidenhead library is ported from an objective c. The library if found on github as one single source and I rewrite it in Java. The library is used to perform coordinate system conversion.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

In Android Studio, create a new Android project with empty Activity template.
Setting API level to the latest one(API 23), and set the minSDK version to API 15(Android 4.0.4).
Include the Volley library by adding:

```
compile 'com.mcxiaoke.volley:library:1.0.19'
```

to app/build.gradle

This is an unofficial mirror of Volley project anyway(https://github.com/mcxiaoke/android-volley)

Also ok http client with the following code:

compile 'com.squareup.okhttp:okhttp:2.6.0'

Google play Service SDK(AdMob and Map), android support library and design support library are used, which is added the same way as above.

Task 2: Implement UI for Each Activity and Fragment

Layout files table here:

Name	Description	Alternative version
activity_home	Layout of HomeActivity Consists on viewpager and adView only	-
activity_main	Layout for MainActivity, consists of progressbar	-
activity_repeaterdetail	Layout for RepeaterDetailActivity, as a holder of RepeaterDetailFragment	-
activity_userdetail	Layout for UserDetailActivity	-
appwidget_main	Main layout file for app widget	-
appwidget_main_item	the listitem required for the app widget	-
cardview_userdetail	the cardview item required in UserDetailActivity	-
content_userdetail	the main content of UserDetailActivity	-
fragment_repeaterdetail	Layout for RepeaterDetailFragment, it can be used as in form of DialogFragment	-

fragment_repeatermap	Layout for RepeaterMapFragment	A master-detail layout for device have the smallest width 600dp
fragment_userdetail	Layout for UserDetailFragment	-
fragment_userlist	Layout for UserListFragment	-
fragment_userlist_item	List item used in UserListFragment	-

Material Design is the basic principle of the above UI design.

Task 3: About data

Content Provider is created to handle the API request based on data request by the application.
ShareActionProvider is used to share data outside this application.
The mechanism is like what we had done in the previous Nanodegree project.
An Intent Service is written to perform download task.

Task 4: Core Logic

Program the Activity and Fragment logic as designed.
Besides that, App Widget is needed to implement as well.

Tables on some of the key components

Name	Task
HomeActivity	For data checking
HomeActivity	Main Activity that user will interact the most. Contains a viewpager for user to view repeaters and users list(Fragment)
UserDetailActivity	Showing the user detail
DMRContract	Data contract
DMRDbHelper	SQLite database implements
DMRProvider	Encapsulation of Sqlite operations

AppWidgetProvider	Appwidget controller
AppWidgetRemoteViewService	Logics on appwidget data retrivating
Constant	Constas used in application

Task 5: Polishing

Adding back the content description for some UI controls.
Test for LTR layout.
Shared Element transection can be added.
Error checking on different using scenario.

Task 6: Deploy

Build the release APK using gradle script.