Hao Zhang

Design and Implementation of Social Event Application Based on Android

Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology

Bachelor’s Thesis

30 April 2019

|  |  |  |
| --- | --- | --- |
| Author  Title  Number of Pages  Date | Hao Zhang  Design and Implementation of Social Event Application Based on Android  xx pages + x appendices  30 April 2019 | |
| Degree | Bachelor of Engineering | |
| Degree Programme | Information Technology | |
| Professional Major | Mobile Solutions | |
| Instructors | Kari Salo, Head of Degree Programme | |
| Over the years, more and more tourists come to Helsinki to travel, in order to let more people know about the events happening in Helsinki area, with the help of the “City of Helsinki” organization, a social events viewing application based on Android platform is born. The main goal of this thesis is to produce a public events information platform based on Android to make it easier for people to find Activities happening in the moment or in the future and make people participate in Activities, integrate into local life and learn about Helsinki culture.  During the development process, back-end data is provided from the organization's open data which covers public data in the Helsinki region. The major application case used in this thesis is a completed social event application written in Kotlin and the specific location of the event will be marked on Google Maps. Besides, RecyclerView is wildly used in this application to display specific event information, such as date, price, event publisher and so on.  Design pattern, as an essential part of computer science, is beneficial for keeping projects architecture scalable and testable. This thesis introduces Model-View-View-Model, a design pattern encouraged for Android development. In addition, MVVM design pattern will be demonstrated along with the extracted code from the application case.  In summary, this thesis implements a social event application based on MVVM design pattern and the UI of application conforms to the “Material Design” specification. | | |
| Keywords | | Material Design, Model-View-View-Model, MVVM, Android Development, Kotlin |

Contents

List of Abbreviations

[1 Introduction 1](#_Toc3413395)

[2 Theoretical Background 1](#_Toc3413396)

[2.1 Android Fragment and Views 1](#_Toc3413397)

[2.1.1 Fragment 1](#_Toc3413398)

[2.1.2 ViewPager 2](#_Toc3413399)

[2.1.3 RecyclerView 6](#_Toc3413400)

[2.2 Android Four Main Components 7](#_Toc3413401)

[2.2.1 Activity 8](#_Toc3413402)

[2.2.2 Service 9](#_Toc3413403)

[2.2.3 Broadcast Receiver 10](#_Toc3413404)

[2.2.4 Content Provider 11](#_Toc3413405)

[3 Application Case 12](#_Toc3413406)

[3.1 Application Description 12](#_Toc3413407)

[3.2 Model View View-Model (MVVM) 13](#_Toc3413408)

[3.3 UI Design 16](#_Toc3413409)

[4 Implementation 16](#_Toc3413410)

[4.1 Project Requirements 16](#_Toc3413411)

[4.1.1 Feature A 16](#_Toc3413412)

[4.1.2 Feature B 16](#_Toc3413413)

[4.1.3 Feature C 16](#_Toc3413414)

[4.2 Technology Stacks 18](#_Toc3413415)

[4.2.1 Data Flow 18](#_Toc3413416)

[4.2.2 Kotlin over java 18](#_Toc3413417)

[4.2.3 Unit Testing 19](#_Toc3413418)

[5 Results 19](#_Toc3413419)

[6 Conclution 19](#_Toc3413420)

[7 References 19](#_Toc3413421)

[References 20](#_Toc3413422)

Appendices

Appendix 1. Title of the Appendix

Appendix 2. Title of the Appendix

List of Abbreviations

ORM Object-relational mapping. The set of rules for mapping objects in a programming language to records in a relational database, and vice versa.

DBMS Database management system. Software for maintaining, querying and updating data and metadata in a database.

[注意修改每个图片的索引]

# Introduction

Write the introduction of your thesis here. Use the Body text style for normal text. Likewise, use styles Heading 1, Heading 2, and Heading 3 for headings. As you apply the predefined styles, the text will automatically format correctly: the line spacing will be 1.5, both edges will be justified, and the text will be hyphenated.

The last line of a paragraph can be left hanging, that is, it does not have to reach the right margin.

Begin a new paragraph at the left margin, that is, do not indent the first line.

# Theoretical Background

## Android Fragment and Views

Android contains multiple views for user interface components, such as text-view, image-view, viewpager, recycler-view and so on. Since ViewPager and recycler-view are widely used in the application case of the thesis, therefore these two views will be discussed in this chapter along with android Fragment.

### Fragment

One of the powerful features added to Android 3.0 (API 11) is Fragment, which is designed to provide more dynamic and flexible UI design support for large screens such as tablets. Fragments must always be managed by the Activity , and their lifecycle is directly affected by the lifecycle of the host Activity [X].

For example, when an Activity is paused, all of its Fragments are paused as well; when the Activity is destroyed, all Fragments are also destroyed. However, when the Activity is running, each Fragment can manipulate independently, such as adding or removing them. Since each Fragment is able to designed as a modular and reusable Activity component, therefore, compared to Activity , Fragment takes up less memory space. The diagram below shows the process when a Fragment is added to an Activity :

1. Workflow of adding a Fragment to Activity . Copied from [X]

In the first step, the Activity gets a reference to the Fragment. After that, the Activity gets a reference to the View-Group where the Fragment will be rendered inside of it. Secondly, the Activity adds the Fragment and the Fragment starts to generate its view based on its independent XML file. Finally, return the created Fragment view to the Activity and the Fragment view will be inserted into the View-Group parent for displaying.

### ViewPager

ViewPager is a component of SupportV4 and it extends from the View-Group class, hence, ViewPager can be regarded as a view container. In practical working, ViewPager is most often used in conjunction with Fragment, which is a convenient way to supply and manage the lifecycle of each page [X].

1. Guide pages on Android application

ViewPager, as a view container, has multiple views inside of it which  allows the user to flip left and right through pages of data as seen in figure 2. Other than making guide pages, the most commonly used of ViewPager is to make an automatic slide show to display the hottest events or advertising exhibitions.

1. Layout of MainActivity

In the application case of this thesis, ViewPager is used to combine with Fragments as the main interface framework. As observed in Figure 3, the layout of the main screen is not very complicated. There is a tab bar at the top, and the rest of the page is filled with a ViewPager which will be filled with four Fragments.

1. Main interface framework of the application case

The ViewPager in the main page just serves as a container, mainly responsible for dynamically loading the four Fragments and each Fragment will display distinct page. As shown in the Figure 4, inside the setupViewPager() function that does the following things: instantiate a PagerAdapter.object and by invoking addFragment() function to insert 4 Fragments to the ArrayList that inside the adapter. By assigning the customized ViewPager adapter object to the adapter property of the viewpager, finally, hook up the PagerAdapter to the ViewPager,

The ViewPager class requires an adapter class to provide data to it, as observed in Figure 4, the data source used in PagerAdapter is an ArrayList to carry data items. There is an inner-class that extends the FragmentPagerAdapter abstract class and implements the getItem() method to supply instances of Fragment as new pages. The ViewPager adapter also requires to override the getCount() function, which returns the amount of pages the adapter will generate. In addition, ViewPager has built-in swipe gestures to transition through pages, and they display screen slide animations by default, so there is no need to define extra animation.

### RecyclerView

RecyclerView is one of the components in Android Lollipop 5.0, with the advent of the RecyclerView, it made its way officially. Compared with the classic ListView, the RecyclerView is more advanced, powerful and flexible enhancement of ListView especially in the aspect of displaying a scrolling list of elements based on large data sets or data that frequently changes [X].

1. Three extensions for RecyclerView

RecyclerView provides a less coupled way to implement what ListView can do, Therefore, it is widely used in the application case of this thesis, there are three main advantages of RecyclerView:

1. RecyclerView itself does not care about where and how the item is displayed, there is a object called layoutManager, as a massive enhancement brought to the RecyclerView, provides a list with different structures. In a ListView, the only type of view available is the vertical ListView. There is no official way to even implement a horizontal ListView. However in RecyclerView,it offers many customization options, such as LinearLayoutManager, StaggeredLayoutManager and GridLayoutManager. If these above layout does not suit project’s need, developers are uble to create their own layout structure by extending the RecyclerView.LayoutManager abstract class.
2. RecyclerView itself does not care about the effect of item addition and deletion animation. The tranditional ListView is lacking in support of good animations, but the RecyclerView uses an animator to alter its appearence. Developers can define their own animator object by extending RecyclerView.ItemAnimator. With the RecyclerView.ItemAnimator class, animating the views becomes easier and intuitive. On the other hand, animation makes the display of list elements more in line with Material Design specifications.
3. RecyclerView itself just care about how to take the view, the views in the list are represented by view holder objects. These objects are instances of a class you define by extending RecyclerView.ViewHolder. Each view holder is in charge of displaying a single item with a view. In a ListView, it was recommended to use the ViewHolder pattern but it was never a compulsion. In case of RecyclerView, this is mandatory using the RecyclerView.ViewHolder class. When implementing a RecyclerView this class is used to define a ViewHolder object which is used by the adapter to bind ViewHolder with a position. In this way, RecyclerView avoids a heavy operation of finding views by ids every time.

## Android Four Main Components

The Android application is mainly composed of four components: Activity, Service, Broadcast Receiver and Content Provider. The development of Android application is inseparable from these four major components.

1. Relationship between Android four major components

In the process of implementing the application case, with the help of the four components. Functionality such as map display, user location monitoring, components communication and so on are realized. The relationships between components are shown in Figure 6. Each component will be specifically introduced in the following paragraphs.

### Activity

The most frequently used in Android is the Activity component, which is one of the most basic modules in Android. Activity takes care of generating a window for placing UI with setContentView(View) method. Activities are usually presented to the user as a full screen window. These interfaces basically belong to or depend on the Activity . In the android program, the Activity generally represents a screen of the mobile phone. If compare a mobile phone to a web browser, the Activity is equivalent to a web page. Similar to the Input, H1, Span and other elements contained in a HTML page, Button, Checkbox and other controls can be added in the Activity. From another perspective, the concept of Activity and the concept of the web page is similar to each other.

Generally, an android application is composed of multiple Activities, and multiple Activities can jump between each other. For example, pressing a Button may jump to other Activities. When a new Activity is opened, the previous Activity is placed in a paused state and pushed into the top of the Activity stack. The user can return to the previously opened Activity by the back button on the phone. There is a choice of Activities that are not necessary to keep, thus, those Activities can be selectively removed from the Activity stack. A little different from web page jumping, jump between Activities may return values. For example, from Activity A to Activity B, then when Activity B finishes running, it may give Activity A a return value. As shown in Figure 6, Launch an Activity for which the previous Acticity would like a result when it finished.

1. Start a Activity and wait for a result

In addition, the Activity component is designed to have a system control in the form of a lifecycle. The Activity mainly has: onCreate, onStart, onResume, onPause, onStop, onDestroy, and onRestart lifecycle methods. When developers design the functionality of an application, they only need to match the lifecycle according to the business to determine what needs to be done at different lifecycle methods.

### Service

Service is a solution for running programs in the background process. It is very suitable for tasks that do not need to interact with users but also run for a long time. The operation of the service does not depend on any user interface, but can be used to interact with other components, even if the current application is switched to the background, or the user opens another application, The service is still up and running because the service is not running in a separate process, but rather on the application process in which the service was created. When an application process is killed, all services that depend on it will stop running.

The service can be used in applications with multiple occasions, such as detecting changes in the file on the SD card, or recording changes in the location of your geographic information in the background. The commonality of the above cases is that these operations do not require an interface display through the service.A typical case is music application in which an Activity starts a service running on user interaction, with the service probably downloading music from the web server. The user can continue to interact with the Activity while the service runs since it executes in the background.

### Broadcast Receiver

Broadcast Receiver is a component that can be regarded as a messaging system across applications and outside of the normal user flow. Similar to the Service component, it has no user interface. The principle behind of Broadcast Receiver is kind of like publish-subscribe pattern in Youtube. In youtube, one user can subscribe to multiple videos to get an update reminder. Similarly, the user can also post videos and be subscribed to by others.

1. Example of detecting battery change using Broadcast Receiver

Back to adnroid, after the broadcast receiver is defined by one application, other applications call it according to its defined rules and send a broadcast to it. After receiving the broadcast, the received data can start an Activity or a Service to start the follow-up function. In additon, an application can define multiple broadcast recipients, either dynamically by inheriting from the BroadcastReceiver class as illustrated in Figure 7 or statically registered in the AndroidManifest file.

### Content Provider

In Android, the protection of data is very strict. Except for the data placed in the SD card, the database, files and other contents held by an application are not allowed to be directly accessed. Android certainly does not really make every application an "island", it has a ”window” for all applications to interact with data from other application, this ”window” is the Content Provider.

1. illustration of migrating content provider storage [X]

The Content Provider is an access plan for third-party application data provided by Android. Content providers are used to share a given data set of an application to other applications, and other applications can obtain data from a content provider. The relational database SQLite provided in Android system will create its own data set for each application level, and only the content provider can share data between each application. Content Provider is able to modify application data store implementation without affecting other existing applications that rely on accessing data. In this case, only Content Provider is affected, not the application that accesses it. For instance, the SQLite database can be replaced with an alternate storage, as shown in Figure 7 [X].

# Application Case

## Application Description

The application case of this paper is the result of my participation in an innovation project, which is a joint event organized by Helsinki Metropolia University of Applied Science and City of Helsinki. The City of Helsinki organization supplies backend database called “Open Data” and API interface to provide data for this project and our project team is mainly responsible for front-end development based on the Android platform.

The core idea of the application is to provide a platform for people living in Helsinki or traveling in Helsinki to find real-time public events. The project adopts the MVVM framework design, and the client interface design follows the specification of material design, MVVM architecture and Material Design are further discussed in the following sections. Moreover, the client is programmed with Kotlin and uses some third-party libraries, such as Retrofit, Google-Map and so on. The main functions of the project are as follows:

1. List upcoming events and display specific event information such as time, location, price, publisher, etc.
2. Show the specific location of the event as a marker on the map
3. Filter result by category, age, price, postal code, date and so on
4. Create peronal event and post it to Open Data
5. Search for upcoming events by specific time or other criteria

## Model View View-Model (MVVM)

In the process of large-scale software system development, if developers do not pay attention to the architecture of the program, the modular design of the code and the decoupling of the function modules, this may lead to some undetectable and difficult to locate errors. Especially these undetectable problems occur when the program reaches a certain scale, Due to the high degree of coupling between functions and services, it can lead to difficulties in reconstruction, finally, it can only be achieved by redesigning the software. In order to improve development efficiency and lay the foundation for future software maintenance. it is crucial to define a reasonable program architecture in the early development process.

The MVVM framework model was first proposed by Microsoft and applied in software development. But MVVM entered and popularized in the Android Development after Data Binding technology was introduced by Google at the 2015 Developer Conference and Google launched the standard Data Binding library for Android development. Data Binding, as its name suggests, Data is bound to an element to receive real-time change notifications, In Android, when the Model layer changes, the latest data is automatically populated into the View layer by pre-writing the attributes in the XML file as shown in Figure 10. This avoids the developer taking the time to re-update the data by findViewById method for the View layer, which simplifies operations and improves development and testing efficiency. Thanks to the convenience of Data Binding, developers have used this technology and brought the MVVM architecture pattern into mobile development.

1. Example of Data Binding

The MVVM design pattern, as shown in Figure 11, is similar to the well-known MVC pattern to some extent because Model layer and View layer are relatively identical. The only difference is C (controller) and VM (view model).

1. View

The View layer is only responsible for UI-related work, UI and data are strictly separated. It does not perform logical processing and update the UI in the Activity or Fragment. UI renew is implemented by changing the data source in the View-Model layer. Simply put: In View layer, the main role of View is informing the View-Model about the user’s actions

1. Vew-Model

The View-Model layer does exactly the opposite of the View layer. View-Model only does things related to business logic and data source. It does not do anything related to UI. The View-Model layer does not hold any reference to the UI element and it is unable to update the UI by reference to the UI element. Simply put: View-Model is not tied to the view, the main role of the View-Model layer is to wrap the model and arrange observable data needed by the view.

1. Model

The biggest feature of the Model layer is that it is assigned the responsibility of data acquisition. View-Model is used with Model to capture and save data. The Model provides a data acquisition interface for the ViewModel to invoke, and through data transformation and manipulation and finally mapping to the properties of a UI element of the View layer.

1. The relationships between View, View-Model and Model (Copied from David Britch (2017) [X])

In summary, Data and UI are independent of each other in the MVVM framework, the View-Model only needs to focus on the data and business logic. The View layer is focused on the UI, and the two are completely isolated. Both the unit test of the UI and the unit test of the business logic are low-coupling. This greatly improves the testability of the application. In addition, due to the low coupling of the MVVM framework, team development is more convenient, such as one developer handling business and data, and another developer responsible for specialized UI processing.

## UI Design

In the rapid development of mobile development, interface design is getting more and more attention, especially after Google released Material Design on Google I/O conference in 2014. Application’s interface design is kind of like the industrial design in industrial products, which is an important selling point of the product. The criteria for verifying an interface are neither the opinion of a project development team leader nor the result of a project member's vote, but the user's feelings. An application with a reasonable and pretty interface will not only bring a comfortable visual enjoyment, but also bring people closer to the product. The UI design of the application case uses Material Design as guideline to create a consistent interface and user experience.

# Implementation

## Project Requirements

### Feature A

### Feature B

### Feature C

#### Google Map调用的实现

The Google Maps API is a personalized map programming API provided by Google Inc. for map software developers based on various platforms. Given the open source and free features of the Google Maps API, using the Google Maps API, developers can embed all the data from Google Maps into their own development software and use Google Maps data to provide location services to users without having to It takes a lot of manpower, material and financial resources to build your own map server.

In addition to helping developers embed maps into web applications, the Google Maps API allows developers to use Javascript scripts for application development, add annotations and paths to maps, and other layer overlays. Click on the action and display a bubble alert window with content information. The operability of the Google Maps API gives mobile phone users a sense of software use, so the travel information service application developed this time uses the Google Maps API to achieve the goal of personalized maps.

Specific steps to invoke the implementation of the Google Maps API;

1. Create the project and select "Goolge Apis" for the SDK.

2. Modify the Android Manifest.xml file.

This is because the data about the map needs to be obtained by moving the path: <uses-permission android; name="android.permission.INTERNET>

3. Create a MapView to create a MapView according to the prompts in the map. The apiK value used in the android system is the key value of the previous application.

4, to achieve Map Activity

Map Activity is mainly used to manage the MapView created before. Of course, this part of the program needs to come from the Map Activity class to implement the MapView function.

MapView can mainly provide W kinds of ground circumference, and their respective setting methods are as follows. mMapView.setTraffic(true);//Set to traffic mode mMapView.setSate...te(tme);//Set to satellite mode mMapView.setStreetView(false);//Set to Street View mode 42 Chapter 5 Travel Information Service Application Software The set Built Zoom Controls method can be used to reduce and enlarge the map display. 5, Ovelay use Overlay can be used in advance on the map to express their own language, that is, text records. Before doing this work, you need to use the transformation between the coordinates to achieve the goal.

## Technology Stacks

### Data Flow

### Kotlin over java

The app with the Android operating system is developed in the Java language, which is an object-oriented programming language. Since Java was created, it has been honed for many years and the use of many developers, feedback, maintenance, Java is now very mature, with many excellent features compared with other languages, such as Java is a simple Object-oriented languages no longer use C++'s multi-inheritance, operator overloading and other language features that are difficult to understand. Java also has distributed features. Java supports network programming. Java guarantees through language-level exception trapping mechanism. Its robustness, the Java language is also portable, because the Java language provides the specified basic type data, the Java language provides multi-threading at the same time, and provides a complete set of mechanisms for multi-threading to ensure the correctness of the data. With the further development and improvement of Java, the performance of Java is getting higher and higher on different platforms, and the speed of running on some platforms has not lost C++.

### Unit Testing

# Results

# Conclution

# References

References

Layout of this page in the number (Vancouver) referencing system:

1. Official documentation of Fragment [online] Google Corporation; URL: [https://developer.android.com/guide/components/Fragments](https://developer.android.com/guide/components/fragments) Accessed 23 February 2019
2. Anupam Chugh. Android Fragment Lifecycle [online]; URL: [https://www.journaldev.com/9266/android-Fragment-lifecycle](https://www.journaldev.com/9266/android-fragment-lifecycle) Accessed 22 February 2019
3. Official documentation of ViewPager [online] Google Corporation; URL: <https://developer.android.com/reference/android/support/v4/view/ViewPager> Accessed 23 February 2019
4. Create a List with RecyclerView [online] Google Corporation; URL: <https://developer.android.com/guide/topics/ui/layout/recyclerview> Accessed 23 February 2019
5. Content providers [online] Google Corporation; URL: <https://developer.android.com/guide/topics/providers/content-providers.html> Accessed 23 February 2019
6. The Model-View-View-Model Pattern; 2017. [online] Microsoft; URL: <https://docs.microsoft.com/en-us/xamarin/xamarin-forms/enterprise-application-patterns/mvvm> Accessed 23 February 2019
7. Create a List with RecyclerView [online] Google Corporation; URL: <https://developer.android.com/guide/topics/ui/layout/recyclerview> Accessed 23 February 2019
8. Create a List with RecyclerView [online] Google Corporation; URL: <https://developer.android.com/guide/topics/ui/layout/recyclerview> Accessed 23 February 2019
9. Create a List with RecyclerView [online] Google Corporation; URL: <https://developer.android.com/guide/topics/ui/layout/recyclerview> Accessed 23 February 2019
10. Create a List with RecyclerView [online] Google Corporation; URL: <https://developer.android.com/guide/topics/ui/layout/recyclerview> Accessed 23 February 2019