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-ViewModel, 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-ViewModel, MVVM, Android Development, Kotlin |

Contents

List of Abbreviations

[1 Introduction 1](#_Toc3069330)

[2 Theoretical Background 1](#_Toc3069331)

[2.1 Android Fragment and Views 1](#_Toc3069332)

[2.1.1 Fragment 1](#_Toc3069333)

[2.1.2 ViewPager 2](#_Toc3069334)

[2.1.3 RecyclerView 6](#_Toc3069335)

[2.2 Android Components 7](#_Toc3069336)

[2.2.1 Activity 8](#_Toc3069337)

[2.2.2 Service 9](#_Toc3069338)

[2.2.3 Broadcast Receiver 9](#_Toc3069339)

[2.2.4 Content Provider 10](#_Toc3069340)

[3 Implementation 10](#_Toc3069341)

[3.1 Application Outline 10](#_Toc3069342)

[3.1.1 Description 10](#_Toc3069343)

[3.1.2 Material Design and Layout 10](#_Toc3069344)

[3.1.3 Development and Process 11](#_Toc3069345)

[3.2 Model View ViewModel (MVVM) 11](#_Toc3069346)

[3.3 Project Requirements 13](#_Toc3069347)

[3.3.1 Feature A 13](#_Toc3069348)

[3.3.2 Feature B 13](#_Toc3069349)

[3.3.3 Feature C 13](#_Toc3069350)

[3.4 Technology Stacks 14](#_Toc3069351)

[3.4.1 Data Flow 14](#_Toc3069352)

[3.4.2 Kotlin over java 14](#_Toc3069353)

[3.4.3 Unit Testing 15](#_Toc3069354)

[4 Results 15](#_Toc3069355)

[5 Conclution 15](#_Toc3069356)

[6 References 15](#_Toc3069357)

[References 16](#_Toc3069358)

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.

# 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 Components

The Android application is mainly composed of four components: Activity, Service, Broadcast Receiver and Content Provider [50]. The development of any Android application is inseparable from this. Four major components, in the realization process of the scenic intelligent tour guide service system, with the different functions of the four components, the functions of map display, visitor location monitoring, component communication and so on are realized. The relationship between the various components is shown in Figure 2.2:

### Activity

The most frequently used in Android, the most well-known is the Activity interface. Activity is all the interfaces that can be seen on the mobile phone. These interfaces basically belong to or depend on the Activity. The main role of Activity is to show the user information, provide user and system interaction interface, and monitor the interactive controls such as buttons on the interface.

In order to weaken the language of the admonition program and the characteristics of the process, the Android system designer designed the Activity component into a system control with a lifecycle form. The Activity mainly has seven lifecycles: onCreate, onStart, onResume, onPause, onStop, onDestroy, onRestart. These cycles are between the app running from the click and the last time the app is closed. When developers design the functionality of an app, they only need to match the life cycle according to the business to determine what needs to be done at different times.

----------------------------------------

An activity can be thought of as a user window that can be implemented programmatically. It is an important component of an Android application. It is the basis for all Android application development and provides a visual interface for interacting with users. In an activity, you can add multiple components and use different components to take care of different functions. Activities mainly include interface display and event processing. Developers can load all components of the user interface into the active lifecycle method onCreate, which is displayed on the device screen. Activities can handle the events of all components on the user interface, and complete the system functions by writing event listener code. For example, a developer can load a Button button on the interface and set the button's onClick event to implement other actions after the user clicks the button.

An Android application can have only one activity, or multiple activities can be combined. These activities form the activity stack. The current activity is at the top of the stack, and the previous activities are suppressed. These activities strictly follow the activity life cycle limits. In the actual development, the activity life cycle method should be focused on, and the implementation of each method directly affects the function and performance of the program.

### Service

The service component is similar to the active component and is the functionality required to complete the developer. The only difference is that the service component does not have a user interface. Developers only need to write the function code of the service, not involving the loading and display of the interface. The service can run long hours in the background, completing some of the developer's time-consuming operations that cannot be implemented in the activity. Other application components can launch a designed service, even when the user switches to another application, the service can still run in the background.

    Services can be divided into two types, startup and binding. Startup, when the active component starts the service by calling the startService method, the service is turned on, even if the activity that started it has been destroyed, the service can still run in the background; binding, when the active component is bound to the service by calling the bindService method, The service is in a bound state. Multiple components can be bound to the same service, and the service is destroyed when they are untied.

Developers generally perform long-running functions such as network upload and download, music playback, and user positioning that do not require an interface display through the service.

### Broadcast Receiver

A broadcast receiver is a component that handles external events of an application, similar to a service component, and has no user interface. After the broadcast receiver is defined, other applications call it according to its defined rules and send a broadcast to it. After receiving the broadcast, you can start an activity or a service to start the follow-up function. The use of broadcast receivers enables communication between different components within the same program and communication between different program components. An application can define multiple broadcast recipients, either dynamically by inheriting from the BroadcastReceiver class or statically registered in the AndroidManifest file.

### Content Provider

Content providers are used to share a given data set of one 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. The content provider includes the addition, deletion, and modification of data.

# Implementation

## Application Outline

### Description

### Material Design and Layout

In the era of disorder, Android is full of the style of the old Google period of freedom and laissez-faire. Android is like a wasteland opened by Google that everyone can use freely. Google does not impose any restrictions. Developers can upload any application designed according to their own ideas to the store without review. Android does not provide developers with a unified design specification in the era of disorder. The application products are mostly referenced to the materialized, flat design norm or self-contained style. The products of that period have no user experience, and people are forced to adapt to various Different interaction styles.

Google launched the new design language Material Design at I/O 2014. Google rethinks the user experience on the Android platform, trying to bring the physical world back through metaphors of the real world

The rules of the feed migrated to the digital world, and plans to extend the design language building to all platforms, including Android, Chrome OS and web pages, starting with the most popular Google Maps and Google Now. Clean typography and a simple unified layout provide a unified interactive experience for Google's many technology products, although they are not identical in form UI and physical world, but have the same interaction rules.

Material Design is based on three principles:

1. The entity sense metaphor is the central principle of Material Design. Starting from exploring the elements that make up the application, Google uses "paper" as the basis for Material Design to explore real paper. Material Design combines paper metaphor with ink coating metaphors. When the user touches a button, it will feed back to the user by expanding to a global mode.

### Development and Process

## Model View ViewModel (MVVM)

In the development process of a large software system, if you pay attention to the architecture of the program, you don't pay attention to the developer's modular design of the code, and do not pay attention to the decoupling of the function modules coupled by the program, which may be generated during the later iterative development and maintenance of the program. Some undetectable and difficult to locate errors, especially when the program reaches a certain scale, it will need to refactor the program. However, due to the coupling of function and business, the reconstruction will be difficult. Finally, the software can only be redesigned. To achieve, and this is a failed software development process. The best solution to the above problems is to design the system in the early development process, and to design functions and data for the functions and services that the software may have, thereby improving development efficiency and laying the foundation for future software maintenance.

As the software development process matures, many developers have summarized and created a number of framework design patterns suitable for large-scale software development processes, including MVC, MVP and MVVM. These three framework ideas were originally applied to Web applications. Mode, which shines in the era of rapid Internet development, the Android Framework incorporates these into the design of the system; MVP evolved from MVC, originally developed by IBM for object-oriented programming (java and C++) The framework model, after learning Android for a period of time, borrowed to structure the Android App architecture, and got a lot of praise; MVVM framework model was first proposed and applied by Microsoft in software development, with Google in 2015 developers After the Data Binding technology was proposed at the conference and the standard class library of Data Binding was provided to support Android development, many developers used this technology to introduce the MVVM architecture pattern into the mobile development field, and benefited from the introduction of MVVM after development. Data Binding is convenient, many Android developers Give a high rating.

Architecture Patterns The MVVM model was first proposed by Microsoft engineers, but at the 2015 Google Developers Conference, Google engineers introduced MVVM to the development of Android App. MVVM is called Model View ViewModel. MVVM is based on MVP, because Google easily transforms MVP into MVVM through Data Binding technology. The relationship between specific MVVMs is shown in Figure 2.3.

    MVVM is actually a further optimization of MVP. MVP is implemented by the Presenter layer when the data between the Model layer and the View is passed, but it is implemented by the ViewModel in MVVM, but the specific implementation process is realized by Data Binding technology. This is where Data Binding technology drives the implementation of the MVVM framework model. The Data Binding technology is that when the Model layer changes, the latest data is automatically filled in by the attributes written in the XML file in advance, without the developer taking the time to re-update the data for the View layer, thereby simplifying operations and improving development and testing efficiency.

## 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. Details of the reference Details of the reference Details of the reference Details of the reference Details of the reference.

**Title of the Appendix**

The contents of the appendix are placed here. Below are the instructions for removing and adding appendices in a way that maintains the headers and footers in their correct form.

Instructions for removing an unwanted appendix:

1. Select the entire page(s) that form the appendix and delete the contents by hitting the Delete key.
2. As you are in the beginning of the empty appendix page (see the image below), double-click the header of the empty page and press Link to Previous button in the ribbon. The following dialogue window opens:



Click Yes.

1. If necessary, make hidden format information visible by pressing .
2. Delete the section break immediately before the appendix to be removed (see image below).



Instructions for adding a new appendix:

1. Place the cursor at the end of the last appendix.
2. Select Page Layout from the menu bar. From the ribbon, select Breaks/Section Breaks/Next Page. This causes a new appendix to appear, but the appendix number in the header is not yet correct.
3. Double click the header of the new appendix with the wrong appendix number. If the option “Link to Previous” is selected, click the corresponding button to deselect it.
4. Replace the appendix number with the correct number.

Note that the appendices need to be updated in the table of contents manually.

**Title of the appendix**

The contents of the appendix are placed here.