**Building an Android application that uses Wikipedia Miner**

Shixuan Miao

Outline:

1. Introduction
2. Background

2.1 Android

2.2 The Wikipedia Miner Toolkit

1. Design
   1. High level design
   2. Detail design
2. Implementation
   1. Wikipedia Miner Web-Services
   2. Volley
   3. Coding
3. Conclusion and future work
4. References

3. Design

The following subsections cover the design of the project.

3.1 High level design

The basic requirement of this project is to build an Android application, which enables users to quickly search the definition of a keyword from Wikipedia.

Figure 3-1-1 shows the three layers of this project: Android application, Wikipedia Miner Web Services and Wikipedia Dataset. The Android platform (Version 4.2) was chosen to implement the client software for this project. The Wikipedia Miner Web Services (<http://wikipedia-miner.cms.waikato.ac.nz/services/>), which are hosted by the University of Waikato, provide a suit of XML-based services to extract information from Wikipedia. The Wikipedia Website provides the dataset for this project. The data format is JSON.

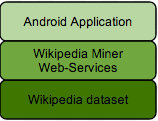


Figure 3-1-1 Project Structure

Figure 3-1-2 shows the project workflow. In this project, we use two Wikipedia Miner Web services: Search and exploreArticle. After the users input the keyword and click the search button, the application will send a request to the Search service. Then the sever side will return a set of response data, which includes a list of related Wikipedia articles’ titles. If the list is empty, it means a definition for the keyword cannot be found on Wikipedia. Next, the application will go back to the start screen. On the other hand, if matches have been found in Wikipedia, the application will display the list of Wikipedia article title keywords on the screen. After that the users choose the one title which they really want to know about. As before, the application will send a request to the exploreArticle service, this time with the article id. The server will return the short definition of what they are looking for. Finally, the definition will be displayed on the screen.

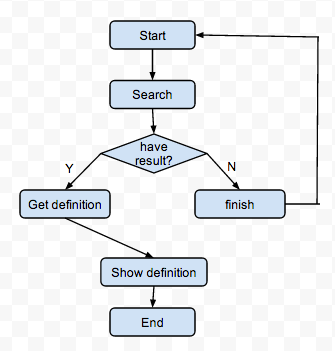


Figure 3-1-2 Project Workflow

3.2 Detailed Design

We now consider the detailed design of this program.

3.2.1 UI design

Figure 3-2-1 shows the UI design for this project. The UI has three parts: Main screen, search result list screen, and Wikipedia definition screen. The main screen has a text-field and a button. When the user inputs the keyword which they want to search for and clicks on the search button, the application will automatically switch to the search list screen. The relevant search results will be listed on the search result screen. After the user touches the specific word which they are looking for, the program will jump to the final definition screen. This definition screen will display the short definition of the keyword from Wikipedia.

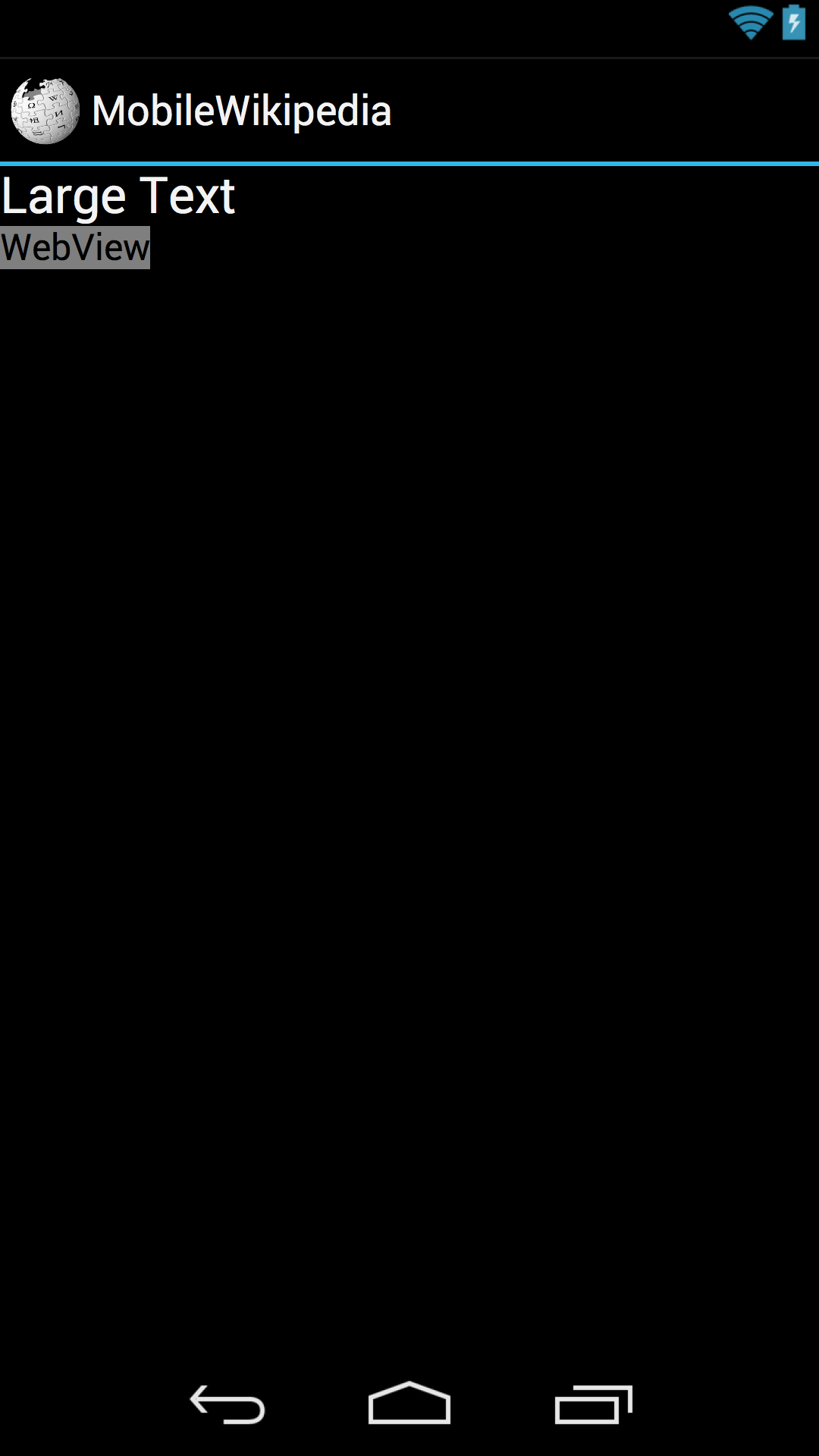
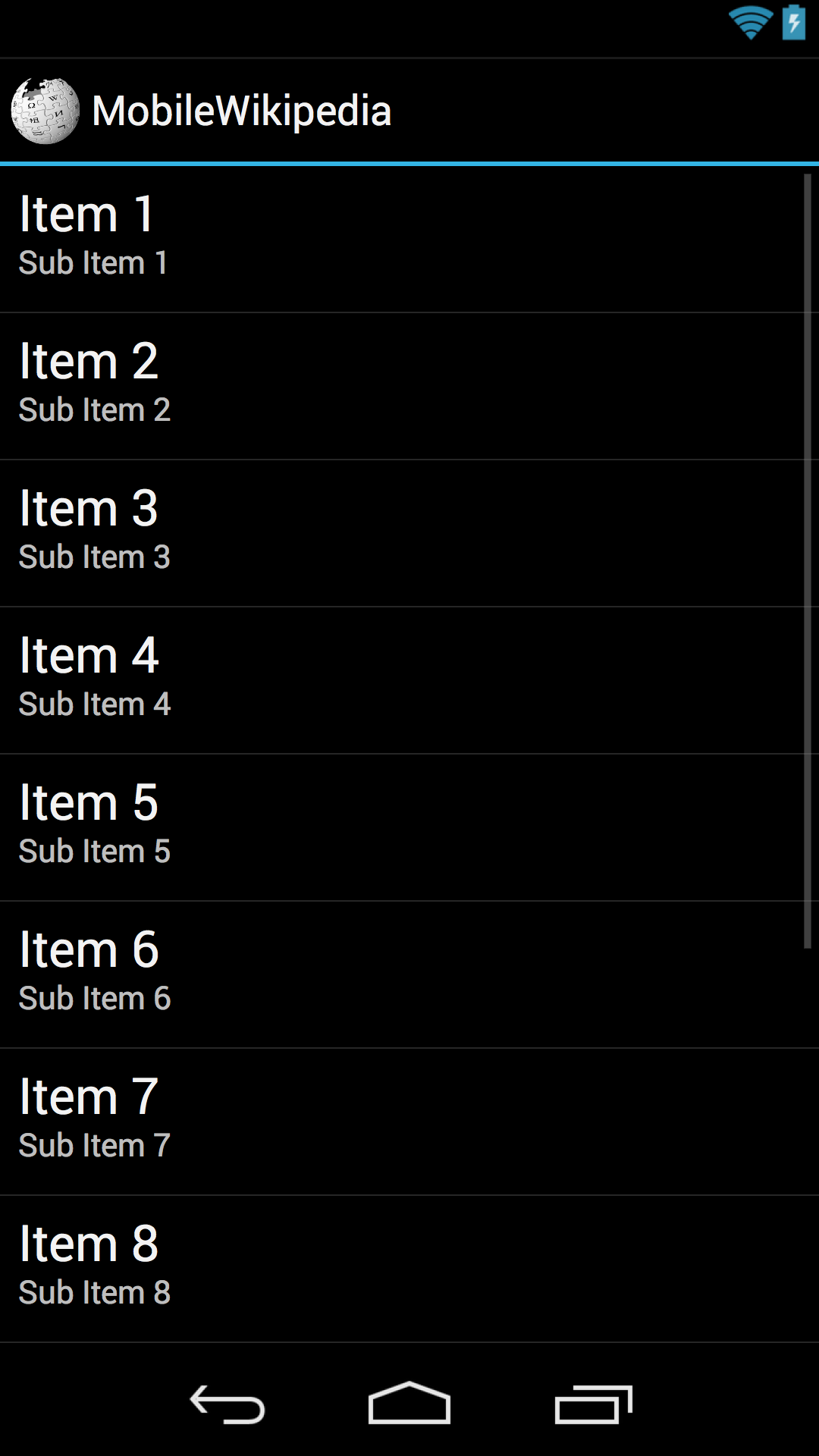
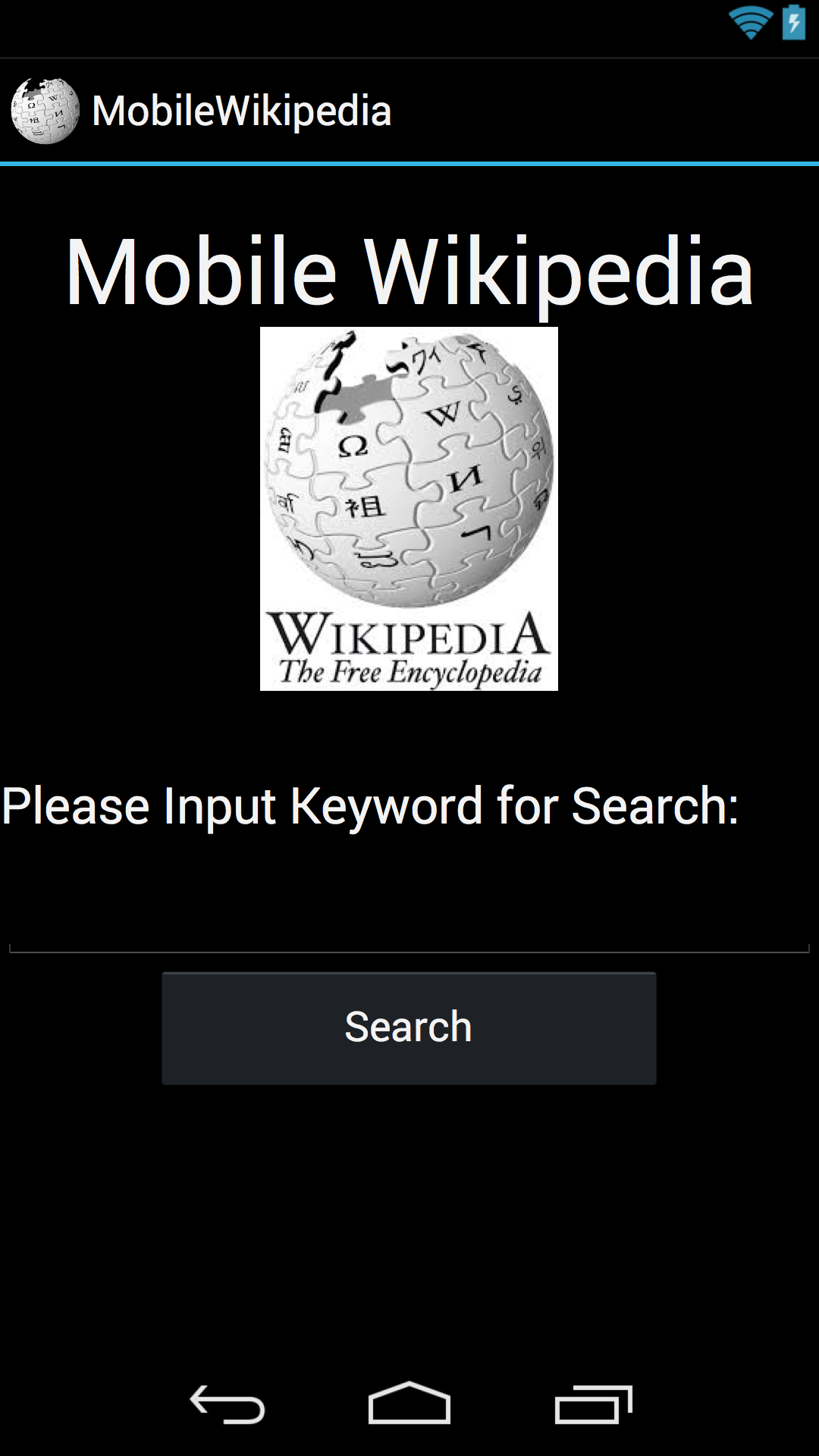


Figure 3-2-1 UI Design

3.2.2 Structure of Program

There are three activities and one fragment in this project: MainActivity, SearchResultActivity, WikiDefActivity and WikiMinerHttpClient. Each activity corresponds to one screen.

Figure 3-2-2-1 shows the structure of MainActivity. Class WikiMinerHttpClint is used for handle network communication. The addFragments() method is used to add the instance of class WikiMinerHttpClint to the Activity. The onClick() method is used to handle to onClick event of the button on the screen. The onWikiMinerExploreResponse() and onWikiMinerSearchResponse() methods are used to handle the response data from the background network component.



Figure 3-2-2-1 MainActivity

Figure 3-2-2-2 shows the structure of SearchResultActivity. The onCreate() method is used to list the search results on the screen.



Figure 3-2-2-2 SearchResultActivity

Figure 3-2-2-3 shows the structure of WikiDefActivity. Class WikiMinerHttpClint is used for handle network communication. The addFragments() method is used to add the instance of class WikiMinerHttpClint to the Activity. The onWikiMinerExploreResponse() and onWikiMinerSearchResponse() methods are used to handle the response data from the background network component.



Figure 3-2-2-3 WikiDefActivity

Figure 3-2-2-4 shows the structure of WikiMinerHttpClient. The getSearchResult() method is used to send search request to the Wikipedia Miner Web-Service server. The getExploreDef() method is used to send exploreArticle request to the server.



Picture 3-2-2-4 WikiMinerHttpClient



Figure 3-2-2-5 Activities workflow

Figure 3-2-2-5 shows the workflow of the activities of this project. The MianActivity is the entry point of the project. It uses getSearchResult() method to get search result and uses startActivity() method to invoke SearchResultActivity. Then it will invoke WikiDefActivity, and the WikiDefActivity will use getExploreDef() method to get Wikipedia definition. The components of this workflow will be discussed in detail in the next section.

4 Implementation

In the following, we will discuss the implementation of this project.

4.1 Wikipedia Miner Web-Service

Wikipedia Miner is an open source toolkit which enables you to set up your own web-services. In this project, we use the web-services which are hosted by the University of Waikato.

4.1.1 Search Service

The base service URL for the search service is: <http://wikipedia-miner.cms.waikato.ac.nz/services/search>?

Table 4-1-1 shows the parameters for the query. One can specify the query word and the response format.

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Value** |
| Query | String | Your query word |
| responseFormat | String | the format in which the response should be returned  possible values: direct json xml  default value: xml |

Table 4-1-1 Search Service Parameter

Let us consider an example request.

In this example, the query word is kiwi and the response format which we choose is json. The server will quickly return the result shown in Figure 4-1-1. The detail implementation will be discussed in next section.

Http Request:

<http://wikipedia-miner.cms.waikato.ac.nz/services/search?query=kiwi&responseFormat=json>

Result:

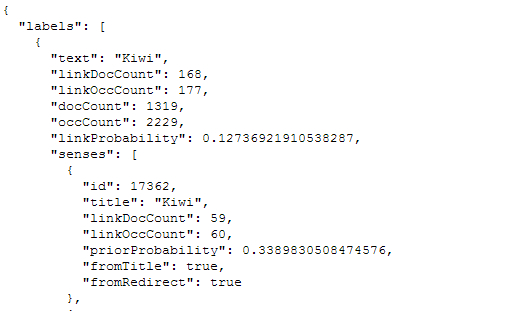


Figure 4-1-1 Search Result

4.1.2 exploreArticle Service

The base service URL for the exploreArticle service is:

<http://wikipedia-miner.cms.waikato.ac.nz/services/exploreArticle>?

Table 4-1-2 shows the parameters for the query. One can specify the query word and the response format. The definition is used to control whether the response data should include the definition or not.

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Value** |
| Id | Integer | The unique identifier of the article to explore |
| Definition | Boolean | true if a snippet definition should be returned, otherwise false |
| responseFormat | String | the format in which the response should be returned  possible values: direct json xml  default value: xml |

Table 4-1-2 exploreArticle Service Parameter

Let us consider another example request.

In this example, we use the article id 17362 which we got from the previous search example. The definition in the response result is what we are looking for. The detail implementation will be discussed in next section.

Http Request:

<http://wikipedia-miner.cms.waikato.ac.nz/services/exploreArticle?id=17362&definition=true&responseFormat=json>

Result:

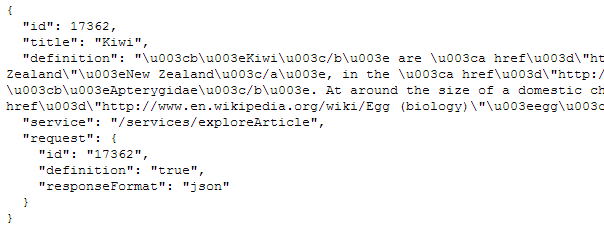


Figure 4-1-2 exploreArticle Result

4.2 Volley

Volley is an open source library which is made by Google and is provided by Google IO 2013. It makes networking for Android applications much easier and faster.

In this project, we use volley as our network component to post HTTP requests and handle the JSON response.

We use an Android Fragment to deal with network cases in this project. This fragment has an internal interface, WikiMinerListener, which lets the invoking method handle the response result. This is shown in Figure 4-2-1.

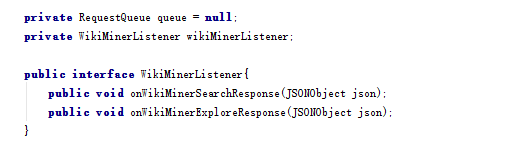


Figure 4-2-1 Internal interface

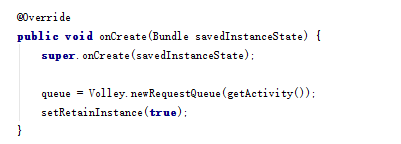


Figure 4-2-2 Declare request queue

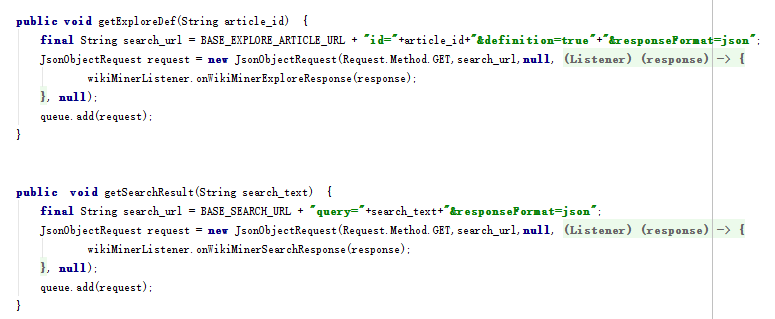


Figure 4-2-3 Post request methods

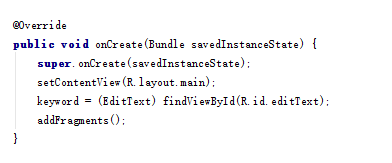
In this project, we just use the RequestQueue and JsonObjectRequest classes from Volley as shown in Figures 4-2-2 and 4-2-3. These are used to implement the getExploreDef() and getSearchResult() methods from Figure 3-2-2-5.

4.3 Coding of UI

According to the description earlier, we separate this project in two parts: UI and Networking.

The Networking part has already been discussed in Section 4.2

In the first screen of the UI, there are two key components: a text-field and a button. When the user clicks on the button, the application will call the onClick method. See Figure 4-3-1.



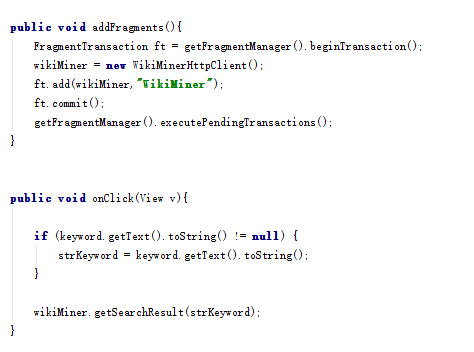


Figure 4-3-1 MainActivity

And we need to implement the method in which the WikiMinerListener is declared (see previous section). In this method, the onWikiMinerSearchResponse() method from Figure 3-2-2-5, we invoke the UI and pass the result by using the Intent class, see Figure 4-3-2

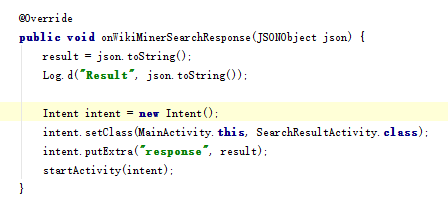


Figure 4-3-2 Method for handling the result

After we get the result JSON string from MainActivity, the SearchResultActivity displays the result in a Listview component and handles the onClick method. See Figure 4-3-3.

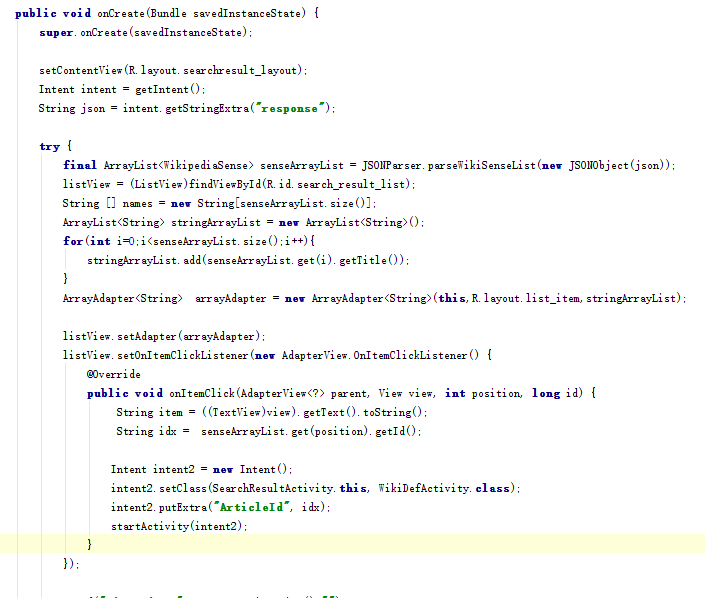


FIgure 4-3-3 SearchResultActivity

If the user touches on an item of the listview, the onClick method will be invoked and passes the article id to WikiDefActivity by Intent.

The WikiDefActivity only has a Text-View and a Web-View which are used to display title and definition of the article from Wikipedia. In this activity, we also use WikiMinerHttpClient to handle the network request and result. See Figures 4-3-4 and 4-3-5. The latter figure has the OnWikiMinerExploreResponse() method from Figure 3-2-25.

The onCreate() method is the override method from Android SDK Activity class. It is used to initial the created state of current Activity. In this case, we use it to set up background network component and catch the input parameter.

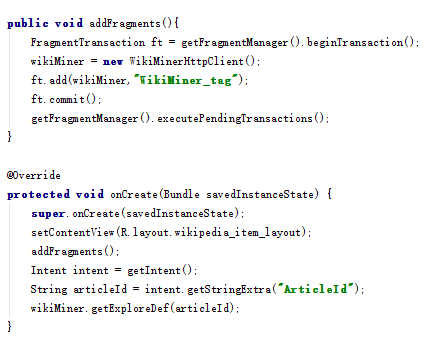


Figure 4-3-4 WikiDefActivity

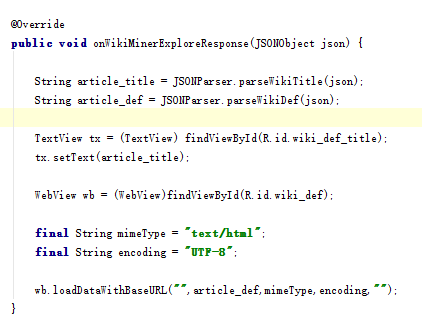


Figure 4-3-5 Method for displaying definition

5. Conclusion and future work

This paper has present a simple application for using Wikipedia Miner Web-services. It also shows a progress of creating an Android application.

In the future, we want to separate the application in two dependent parts: Service and Application.

The Service will run on the background and it provides more functions which using the Wikipedia Miner Web-services. It could be invoke by other Android application and return the response.

The application will invoke the service instead of using its own code.