Introduction

I develop this application by using components including tailored ListView and corresponding Adapters, Drawer, Actionbar, Searchview, Preference framework, Fragments, Database, Provider, BitmapCache, Volley, Android Style, etc. These components consist of five distinctly different features. To deal with the network in terms of API, I use fragments without views to request and cache images. I use three strategies to save power. All API are tailored for this application. This application has been tested on Nexus 5 and Android version is 5.1 with API level 22. Android studio version is 1.2.1.1. A signed APK file is in the top level folder of the zipped file.

Five Features

1. **News List**

   API: http://wnews.timepic.net/wnews/api/newslist/category_id/0?key=hXVBQPSennzh46Xp

   Description: This is the main feature of this application, I use a tailored listView fragment to display data retrieved from API. In order to make this ListView highly performing, I used getView method and holder to reuse the cell in the NewsListAdapter. When the device is rotated, the corresponding data(selected position, news, etc.) will be stored and resumed. The cell itself contains TextViews and NetworkImageViews, When images are not ready or cannot be displayed, a default drawable image will be showed. All images are cached by BitmapCache.

2. **Search news by text**

   API: http://wnews.timepic.net/wnews/api/search/keyword/a?key=hXVBQPSennzh46Xp

   Description: I use Android SearchView to search news by sending data to the server. This Search also can give suggestions from what users have searched. This can be found in SearchRecentSuggestionsProvider and searchable.xml. When a search query is sent, I check the intent type in onNewIntent method to send a search request in MainActivity, then display retrieved data in the list.

3. **News categories**

   API: http://wnews.timepic.net/wnews/api/categories?key=hXVBQPSennzh46Xp

   Description: When the Application is launched at the first time, A database(wnews.db) and a table(news_categories) will be created in database/DatabaseHelper. Then I will request the server to download the latest categories then update this table by using NewsCategoriesDataSource. These categories will be showed in NavigationDrawer.

4. **User settings**

   API: http://wnews.timepic.net/wnews/api/settingsUpdate/?uid=0&settings={%22email%22:%22lishuzu@gmail.com%22,%22notifications_new_message%22:true,%22notifications_new_message_ringtone%22:%22content:\/\/media\/internal\/audio\/media\/17%22,%22notifications_new_message_vibrate%22:true}&key=hXVBQPSennzh46Xp

   Description: I use native Android PreferenceActivity and PreferenceFragment to display, edit and synchronise user settings with the server if the user logged in. Notice: once copy this link to a browser, user information will be updated to the parameters in this link. The test might be affected by this link.

5. **Login and Logout**

   API: http://wnews.timepic.net/wnews/api/login/?username=shuzuli&password=9162fa316df079ef770c95e05b708abb&key=hXVBQPSennzh46Xp

   Description: This application can be used without user login, but users can click the login button to in ActionBar. The test account is ***shuzuli***, password is ***1254291***. Once the user has logged in, all user information and settings will be downloaded and stored in SharedPreferences, I use utils/UserSession to manage user information and status. The log out menu shows on action bar. Once the user logs out, all information will be cleaned except the username which will still display in username textview. This offers convenience for second-time login, because, on the mobile device, It's not easy to type. Then, the login menu shows on action bar.

Power strategies

1. **Prefetch Data**

I apply prefetch data strategy in terms of images in the news list, I prefetch all images at the first time I request the API, Then put them into BitmapCache and ImageLoader, So these images won't be downloaded again when we display them anywhere in this application.

2. **Batch Transfers and Connections**

   I use Volley RequestQueue to reuse connections, which all requests will only create one connection. This queue will be kept in ClientFragment. I use setRetainInstance(true) in this fragment, so this fragment will not be destroyed. All requests are kept in this queue, when requests responded, it will call response methods in ApiResponseListener. When this fragment is stopped, I cancel all requests in onStop method to close the connections.

   I also bundled synchronisation of user settings with a getNewsList method at line 168 of MainActivity. Every time this application gets news, it will check whether settings are changed and the user logged in, if they are both true, the application synchronise settings with the servers.

3. **Reduce Connections**

   I download all news in a single GET request of the API (News list API), because I also design this API. If user click cell of the ListView, it will pass data to the NewsDetailActivity to show details of the news. In this NewsDetailActivity, it will not download news again which could reduce connections.

   I use checkCategoriesCache method at line 88 of MainActivity to refresh categories data every 24 hours, because categories are not changed frequently. This can reduce requests and data download.

   I also close the connections by using cancelAllRequests method of Volley, when the application are paused or exit or the device is rotated, for example, onStop method of MainActivity.

4. **DDMS Tool**

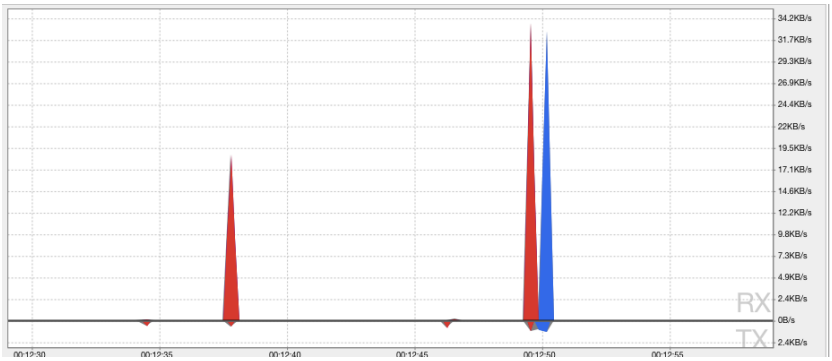

Figure 1. Tracking network usage with DDMS.

As Figure 1 shows, I use DDMS Tool to identify network problem, I optimise my application by using this tool.

Architecture

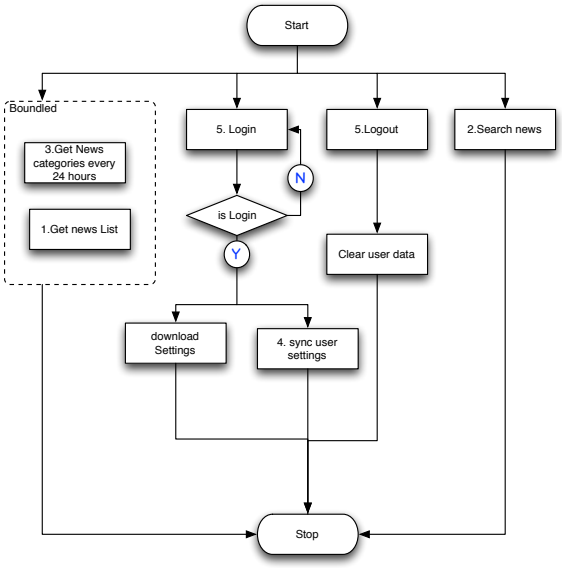

Figure 2: architecture of the app

As Figure 2 shows, the user can access Get news list, login and search news feature. Once the user logged in, the settings can be downloaded and uploaded. The menu on action bar can change depending the login status of the user.

To fulfil these strategies, they do have an effect on my architecture of this application. In order to put all requests into one queue, I have to retain ClientFragment all the time in my MainActivity. This could make this activity more big than I thought. To bundle requests as much as I can, I have to put synchronisation of user settings with download of news list together rather than putting it in onSharedPreferenceChanged method of PreferenceActivity, because if put it in onSharedPreferenceChanged method, it will sync, once settings have changed. This may make the device be full power state and drain more power. To reduce download and requests, I add on some conditions to check whether we really need to send a request. Then I cached what we got for reducing the frequency of connections.

Conclusion

In this essay, I design five different functions by using five API, then I applied three strategies on this application. To examine the network usage, I use DDMS tool. I illustrate the changes of structure of this application because of these strategies in terms of saving power.