International Journal of Multimedia and Ubiquitous Engineering
Vol.9, No.4 (2014), pp.187-198
http://dx.doi.org/10.14257/ijmue.2014.9.4.20

Research and Development of Mobile Application for Android Platform

Li Ma[1,2,3], Lei Gu[1,2] and Jin Wang[1,2,3]

[1]Jiangsu Engineering Center of Network Monitoring, Nanjing University of Information Science & Technology, Nanjing 210044
[2]School of Computer & Software, Nanjing University of Information Science & Technology, Nanjing 210044
[3]Key Laboratory of Meteorological Disaster of Ministry of Education Nanjing University of Information Science & Technology, Nanjing 210044

Abstract

Today, as the developing of hardware of mobile is getting better, the performance index is much higher than the actual requirements of the software configuration. Phone's features more depend on software. As the Android operating system is getting more popular, the application based on Android SDK attracts much more attention. But now, some of the Android application interface is too cumbersome, pop-up ads is overmuch and the function is too single, these cause some inconvenience to the users. This article presents the application by eliminating the redundancy. Three kinds of applications are developed base on Java and Android SDK --- Weibo client, video player and audio player. The audio player uses the ContentResolver and Curor to obtain music files and plays the music by using the Service Components to call the Media Player class in the background. The video player uses the Media Player class provided by Android SDK. This class loads the file through URL, realize the multimedia file parsing by calling the OpenCore Library, which is at the bottom of Android, through JNI and by calling the SurfaceFlinger interface to realize the video files' playback. The users' data is collected through the Sina open platform called by Sina client and the data will be returned under the format of JSON by the Sina server. The system uses the OAuth authentication method for user authorization to complete the login process. The specific functions of this system are developed based on Android Weibo SDK. The interfaces of these Android apps are pretty and the operation is smooth. What's more, the cumbersome interface and excessive advertising are eliminated, so that users are able to manipulate these apps more conveniently and smoothly.

Keywords: *Android, Weibo client, Video Player, audio player, Android SDK*

1. Introduction

In recent years, the emergence of smart phones has changed the definition of mobile phones. Phone is no longer just a communication tool, but also an essential part of the people's communication and daily life. Various applications added unlimited fun for people's lives. It is certain that the future of the network will be the mobile terminal.

Now the Android system in the electronics market is becoming more and more popular, especially in the smartphone market. Because of the open source, some of the development tools are free, so there are plenty of applications generated. This greatly inspired the people to use the Android system. In addition, it provides a very convenient hardware platform for developers so that they can spend less effort to realize their ideas. This makes Android can get further development [1-4].

As the smart phones and Android system getting popular, the operations like listening to music, watching videos, tweeting and some others can be moved from the computer to a phone now.

The applications on the market today are mostly commercial applications, and contain a large number of built-in advertising. If the user prefers to remove the built-in advertising, a certain price must be paid to reach that and this is not convenient. Meanwhile, because of the unfair competition of IT, many applications built illegal program to steal user information and cause some damage to user's personal privacy. Sometimes, users will pay more attention to the user experience of software. Therefore, the development of the application can not only be limited to the function, more attention should be paid to the user's experience. After studying some previous Android applications and access to large amounts of materials, we utilize the Java language, the Eclipse platform, Android ADT and the Android SDK to develop these three mobile applications. These systems have a nice interface and smooth operation. These Apps won't steal any personal information, but can exclude useless information and bring a wonderful user experience.

2. Android Architecture

We studied the Android system architecture. Android system is a Linux-based system, Use of the software stack architecture design patterns [1-2].

As shown in Figure 1, the Android architecture consists of four layers: Linux kernel, Libraries and Android runtime, Application framework and Applications [5-8].

Each layer of the lower encapsulation, while providing call interface to the upper.

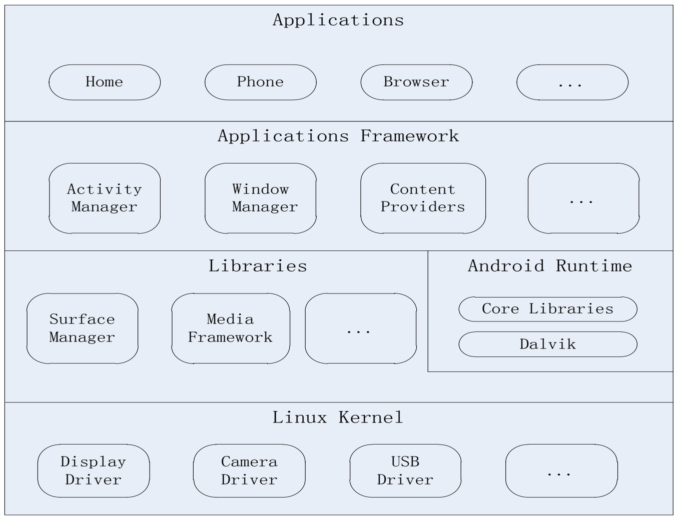


Figure 1. Android Architecture

2.1. Applications

Android app will be shipped with a set of core applications including client, SMS program, calendar, maps, browser, contacts, and others. All these application programs are developed in Java.

2.2. Application Framework

The developer is allowed to access all the API framework of the core programs. The application framework simplifies the reuse of its components. Any other app can release its functional components and all other apps can access and use this component (but have to follow the security of the framework). Same as the users can be able to substitute the program components with this reuse mechanism.

2.3. Libraries and Android Runtime

The library is divided in to two components: Android Runtime and Android Library. Android Runtime is consisted of a Java Core Library and Dalvik virtual machine. The Core Library provides Java core library with most functions. Dalvik virtual machine is register virtual machine and makes some specific improvements for mobile device.

Android system library is support the application framework, it is also an important link connecting between application framework and Linux Kernel. This system library is developed in C or C++ language. These libraries can also be utilized by the different components in the Android system. They provide service for the developers through the application framework.

2.4. Linux Kernel

The kernel system service provided by Android inner nuclear layer is based on Linux 2.6 kernel, Operations like internal storage, process management, internet protocol, bottom-drive and other core service are all based on Linux kernel.

3. Experimental Methods

3.1. Video Player

Video Player is achieved through the Eclipse platform. In order to develop android app, we will install a plug-in for Eclipse: Android Development Tools (ADT). Once installed, download Android SDK [10, 12], install and configure the SDK, then we can develop a video player.

Our research begins with the study of operating mechanism, Android platform media layer structure, xml customizable interface, Content Providers [9] achieves file scanning to get a list of media files, MediaPlayer class, file parsing, Surface Flinger interface. After that, we could develop an Android-based mobile video player. Realize media library, video player, file opening, audio, video, photographs and other functions. Figure 2 is system flow chart.

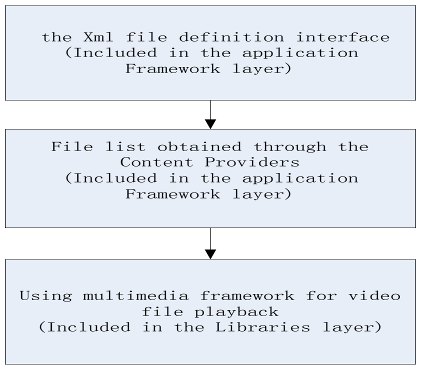

the Xml file definition interface (Included in the application Framework layer)
File list obtained through the Content Providers (Included in the application Framework layer)
Using multimedia framework for video file playback (Included in the Libraries layer)

Figure 2. System Flow Chart

The software interface is defined through XML files. XML layout files control view, is not only simple, but also isolated the View control logic from Java code and controlled by inserted into XML files. Reflects the MVC principle in a better way and also reflects the principle of separation of logic and views. This software obtains the list of media files by scanning through Content Providers. Content Providers is recognized as a bridge between the data storing and searching across programs. The function is to achieve data sharing among different Apps, it is the only way to share data with other apps. Figure 3 shows the media layer structure [13].

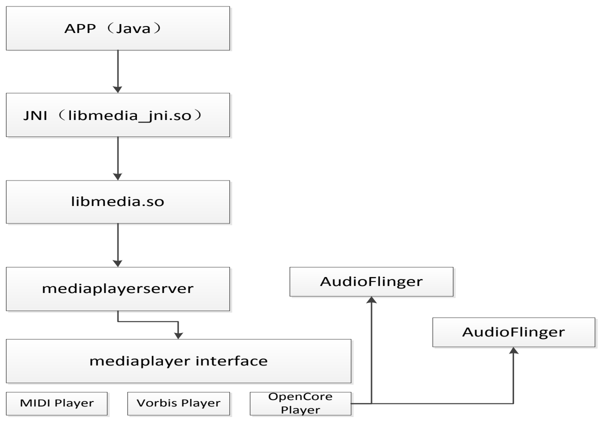


Figure 3. Media Layer Structure

The upper applications of Android-MediaPlayer are implemented by JAVA, realized logic processing. JAVA program realizes the playback of video file and online video by calling the underlying media library libmedi.so through JNI interface.

MediaPlayer can be roughly divided into two parts at run time: Client and Server. They are running in two separated processes. Binder used between them to achieve IPC communication. Mediaplayerservice in Figure 3 is a server-side implementation repository.

MediaPlayer calls media playback capabilities provided by Opencore [14] to implement video file playback, Opencore responsible media file format parsing, decoding audio and video data, and outputs the media data. Opencore calls SurfaceFlinger interface to realize the showing of video data and by applying AudioFlinger interface to realize the playback of audio data.

In the Android media layer, the most important class is MediaPlayer. MediaPlayer class and its associated structures are shown in Figure 4.

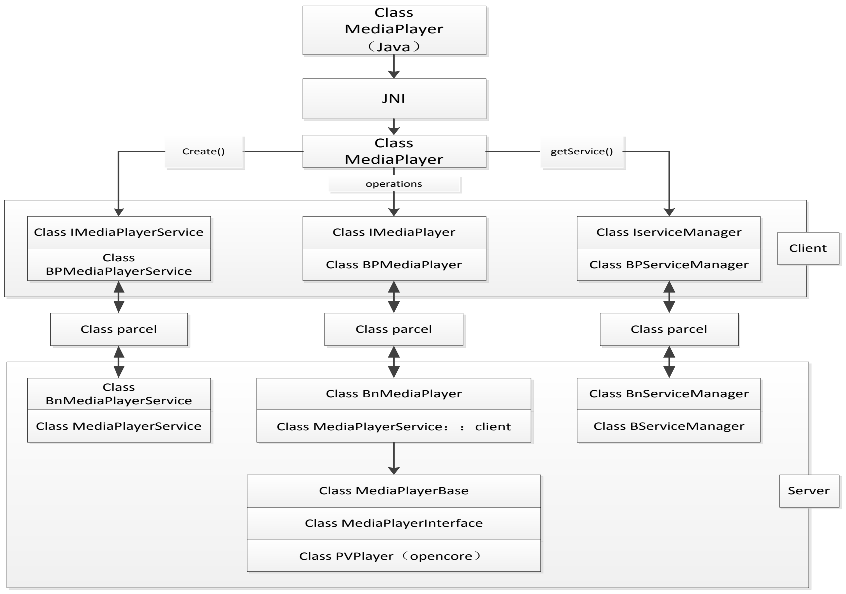


Figure 4. MediaPlayer Class Hierarchy Diagram

Upper JAVA program calls the underlying MediaPlayer class to implement Media streaming. First, the MediaPlayer class obtains a name for media.player services through IService _Manager getService interface. After that, all the operations are conducted through this MediaPlayer player and the interface is IMediaPlayer. All BpXXXX classes in Figure 4 are proxy classes, the responsibility is to realize message forwarding by sending the client requests to the service through the Binder mechanism. A corresponding BnXXXX subclass on the service side is responsible for implementing specific functions. The play of final broadcast media stream is achieved by calling the underlying Opencore libraries through MediaPlayerInterface interface. This class loads pre-played files through Uri， calling the OpenCore multimedia libraries to implement file parsing via JNI, by calling the SurfaceFlinger interface to realize the playing of video file, by calling the AudioFlinger interface to realize the playback of audio data. The software interface is simple, feature-rich, smooth operation and also by calling an external program to achieve audio and image playback. Figure 5 shows the video player interface.

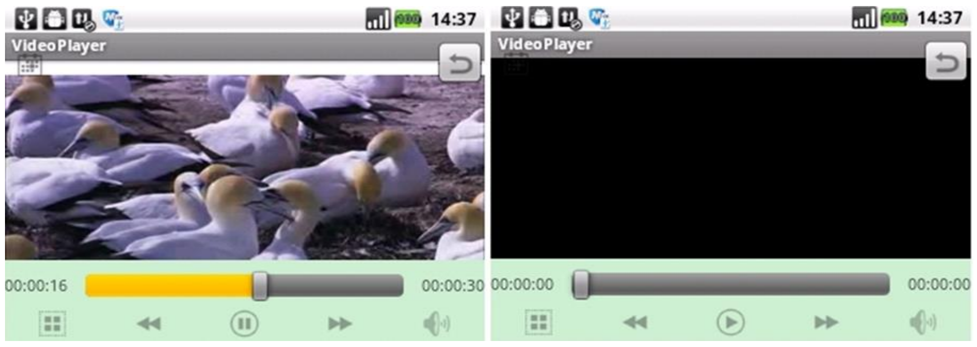


Figure 5. Video Player

3.2. Audio Player

The audio player development tool is the same as the one of video player. System structure and the process is the same as the process of video player. Also defines the interface in the Application Framework layer, and then acquires music files through ContentResolver in the Android Framework layer. Finally, plays the music by using the Service component calling the MediaPlayer class in the Libraries layer. The system structure is shown in Figure 6.

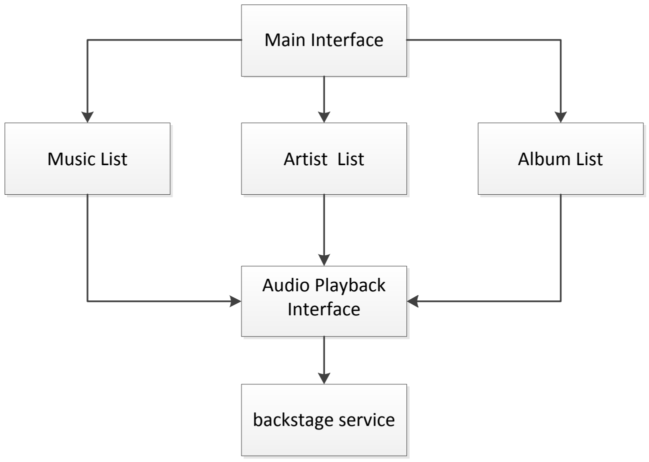


Figure 6. System Structure

The main interface module is the entrance of the application. Users will see the main interface modules after starting the application. The module itself does not reflect any of the information to the user, just call list module to display. Three lists are demonstrated: music list, album list and artists list. The main interface module is realized by calling MusicListActivity, AlbumListActivity and ArtistListActivity [15] module. The main interface logic diagram is shown in Figure 7.

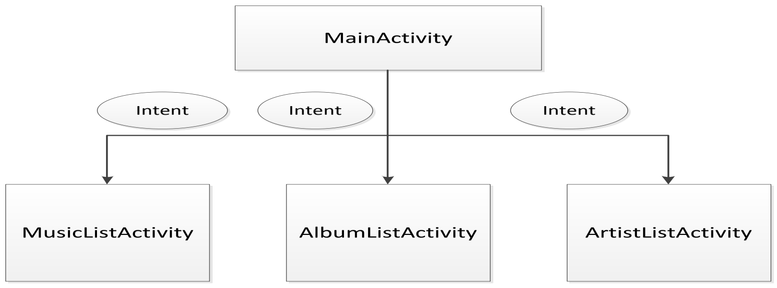


Figure 7. Main Interface Logic Diagram

Figure 8 is the specific flow chart.

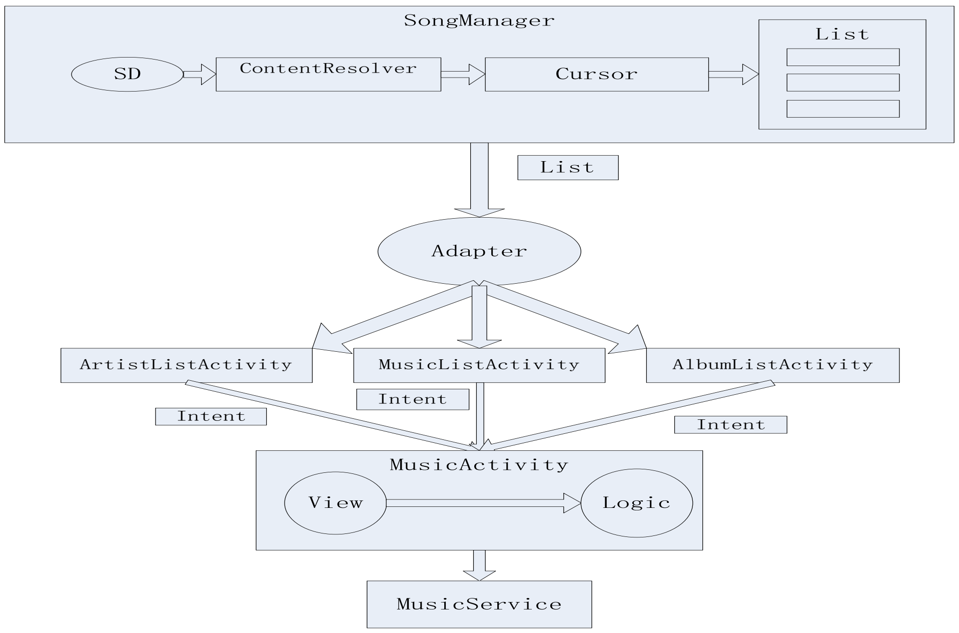


Figure 8. Audio Player Flow Chart

Audio file scanner module is responsible for scanning all the audio files on the SD card. The SongManager in this module is a class, this class has a static method to access to the SD card. The static method acquires the SD card audio resources by using Cursor class method provided by Android system, and will turns the received audio resources into a List class instance objects. The members of List are the JavaBean – Music used in the app. The List Array will eventually be returned to the other modules that they calling it. List module MusicListActivity, AlbumListActivity and ArtistListActivity module will call this module.

Adapter module is a tool that maps instance objects of the List<?> class to the ListView. The adapter module in this application will get a List < Music > instance objects that produced by SongManager module, and map it to the ListView view components in the MusicListActivity and other modules.

List module will not show to the user alone, but called by the main interface module. This module has three parallel parts: MusicListActivity, ArtistListActivity and AlbumListActivity. The function of them is to display the song list, artist and album list in the main interface. These parallel modules will call the static method in the SongManager class to acquire a List <Music> instance object, and then call the adapter module to handle the obtained List <Music> instance objects. Then, map List <Music> to its own ListView component. When the user selects an element of the ListView, this module will encapsulate the information into an intent object and sent it to music playback module.

Music playback module collects the intent sent from List module and analyzed it, then calls the background music services to play the audio file. The View components provides player with some basic functions, such as play, pause, fast forward, fast rewind, single play, random play, etc. This module will make the corresponding logical analysis after the users did operations to the components. Appropriate response and changes will be done according to the results analyzed. Audio player interface is shown in Figure 9.

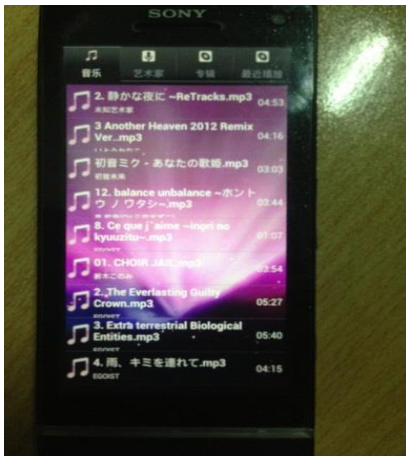

2. 静かな夜に ~ReTracks.mp3 04:53
3 Another Heaven 2012 Remix Ver..mp3 04:16
初音ミク・あなたの歌姫.mp3 03:03
12. balance unbalance ~ホント ウ ノ ワタシ~.mp3 03:44
8. Ce que j'aime ~ingri no kyuuzitu~.mp3
01. CHOIR JAIL.mp3
2. The Everlasting Guilty Crown.mp3 05:27
3. Extra terrestrial Biological Entities.mp3 05:40
4. 雨、キミを連れて.mp3 04:15

Figure 9. Audio player

3.3. Sina Weibo Client

The development tools of this system are the same as the ones of the video player and audio player, but except these tools, Sina open platform is also applied to acquire data and writing data.

In the design phase, logicality and scalability of the whole system are considered. Our system is divided into four layers, namely the UI layer, logic layer, Weibo interface layer, the network access layer. Figure 10 is system architecture.

UI layer is responsible for displaying the various forms of the system and the coordination between the various forms of invocation logic.

The logical layer core control scheduling module is used to access the data transferred by UI, tasks need to be performed are calling Weibo interfaces, accessing network data, returning message, refreshing UI, etc.

Sina Weibo interface layer (Sina API [11] encapsulated) provides various interfaces accessing to the open platform Sina Weibo and used to help the system access data and return data from Sina Weibo system.

Network Access Layer (Sina API [11] encapsulated) is responsible for the system and the server's network connection and data transfer.

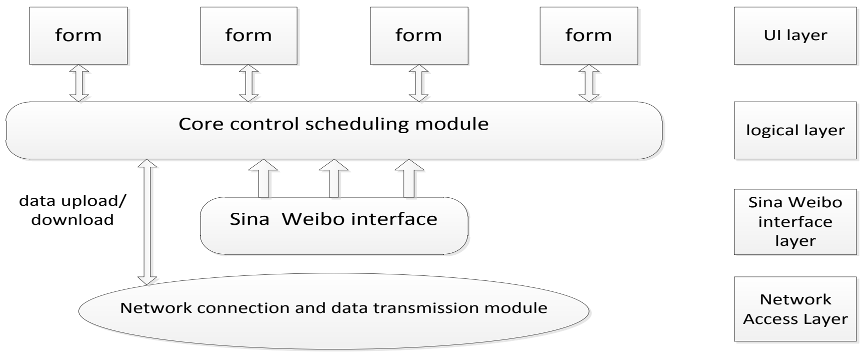

form
form
form
form
UI layer
Core control scheduling module
logical layer
data upload/ download
Sina Weibo interface
Sina Weibo interface layer
Network connection and data transmission module
Network Access Layer

Figure 10. System Architecture

Figure 11 is System flow chart.

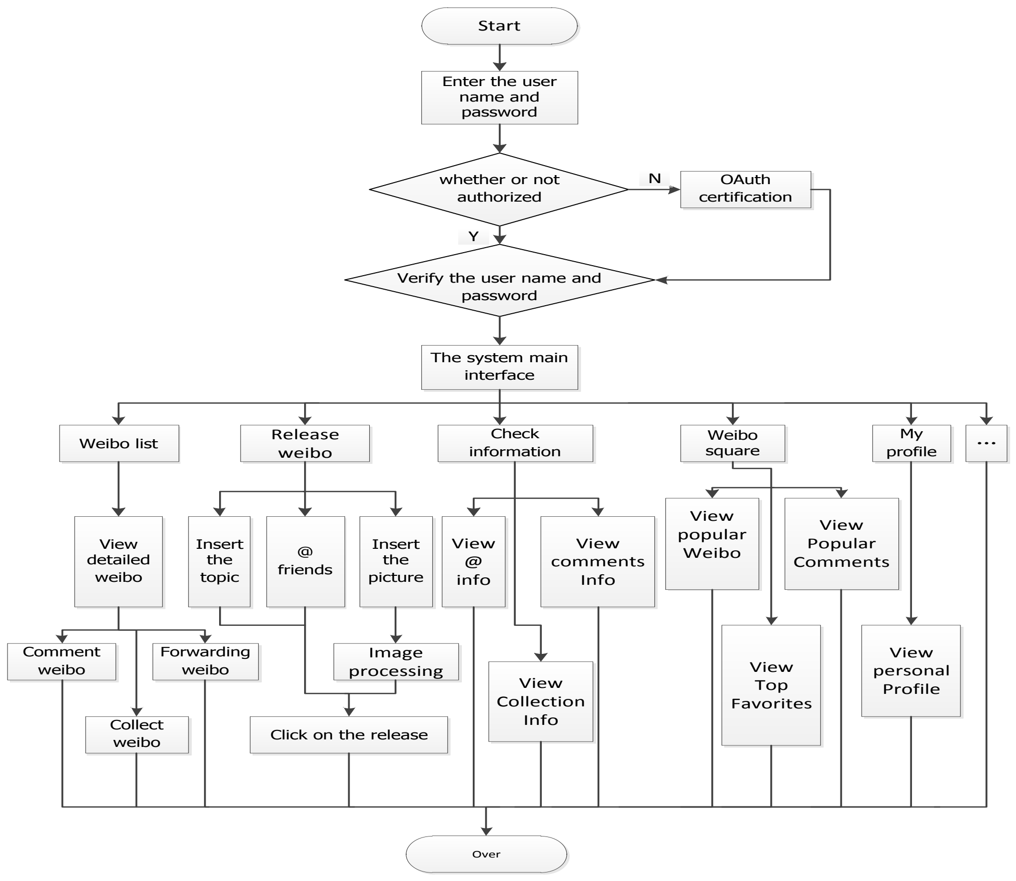


Figure 11. System Flow Chart

User login and authorization are processed by calling the pages provided by Sina, the system doesn't make intervention. Then, the third-party application can access the users' data without knowing the user account and password.

The system uses the Android Weibo SDK of Sina Weibo to assist the development, which has been integrated with an open source Java OAuth [11] authentication package: Signpost, developers have to register as a developer in Sina Weibo developer platform and obtain the corresponding App Key and App Secret to finish the guided registration.

Authentication needs to be done before Sina Weibo SDK interacts with the server. SDK provides a Weibo class, when start the program, the following code needs to be executed to create a Weibo object, and set App Key, App Secret and URL.

```
weibo = Weibo.getInstance();  // Create Weibo object
weibo.setupConsumerConfig(Consumer.consumerKey,
Consumer.consumerSecret);  // set App Key, App Secret
weibo.setRedirectUrl(Consumer.redirectUrl);  // Set the redirect URL
weibo.authorize(activity, new AuthDialogListenerImpl(activity));
```

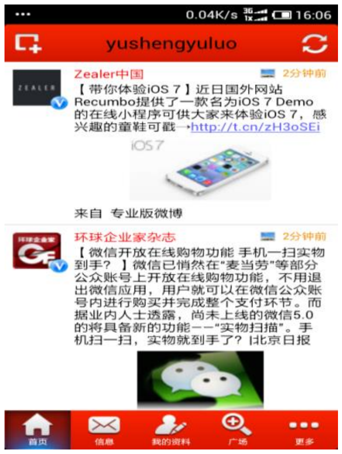
After executing the above code, the login interface will appear, enter user name and password, after that, click the Login button. If it is the first time of login, the Authorization page appears. The main screen will show up after the user login.

Specific functions of this system development are based on Android Weibo SDK, calling its wrapper classes to complete the corresponding task.

For example, the main interface is divided into three parts: the top is a toolbar, the middle area is a ListView, bottom is the button bar. Weibo List is in the middle of the main interface part, displayed through the ListView. As long as Weibo data was obtained from the service side, it will be shown directly on the ListView control through the Adapter.

The function of posting is achieved through WeiboManager.update method. This method can submit text and message containing pictures.

Browse Weibo interface is used to display all the information of Weibo and realize forwarding and commenting. The Weibo browse window class is WeiboViewer and the interface layout file is weibo_viewer.xml. When displaying the Weibo information, accessing the current Weibo Status objects and transfers them to the WeiboViewer, and then calls the loadContent method to load Weibo information. The completed client's main interface is shown in Figure 12.



Zealer中国 2分钟前
【带你体验iOS 7】近日国外网站Recumbo提供了一款名为iOS 7 Demo的在线小程序可供大家来体验iOS 7，感兴趣的童鞋可戳→http://t.cn/zH3oSEi
来自 专业版微博

环球企业家杂志 2分钟前
【微信开放在线购物功能 手机一扫实物到手？】微信已悄然在"麦当劳"等部分公众账号上开放在线购物功能，不用退出微信应用，用户就可以在微信公众账号内进行购买并完成整个支付环节。而据业内人士透露，尚未上线的微信5.0的将具备新的功能——"实物扫描"。手机扫一扫，实物就到手了？ |北京日报

Figure 12. The Client's Main Interface

4. Conclusion

The test involves three environments including hardware, software and network. Test hardware environment is Lenovo Y460 laptop and millet M1 phone; software environment is windows 7 and phone system environment is Android 4.0.3. Network environment is China Mobile which is 10M broadband, WIFI LAN and China Mobile GPRS network.

By testing each function on mobile phone and the computer simulator, the result showed that video player and audio player run well and no advertising. Sina weibo client can successfully complete OAuth2.0 certificate authority and login and collect the basic data of the user information from sina server and no redundant information. Expected effect is achieved after testing all the functions.

Since the Weibo client has to access to the network, when tested on an Android phone, the performance under the environment of WIFI network and mobile 2G GPRS network can meet the expected requirements.

Acknowledgements

This research work was supported by the Natural Science Foundation of China youth project (61103142). It was also supported by the by the Industrial Strategic Technology Development Program (10041740) funded by the Ministry of Trade, Industry and Energy (MOTIE) Korea.

References

[1] M. Butler, "Android: Changing the Mobile Landscape", Pervasive Computing, **(2011)**, pp. 4-7.
[2] B. Proffitt, "Open Android-For better and for worse", Spectrum, **(2011)**, pp. 22– 24.
[3] K. W. Tracy, "Mobile Application Development Experiences on Apple's iOS and Android OS", Potentials, **(2012)**, pp. 30 – 34.
[4] A. Shabtai, Y. Fledel, U. Kanonov, Y. Elovici, S. Dolev and C. Glezer, "Google Android: A Comprehensive Security Assessment", Security & Privacy, **(2010)**, pp. 35 – 44.
[5] A. Shabtai, Y. Fledel and Y. Elovici, "Securing Android-Powered Mobile Devices Using SELinux", Security & Privacy, **(2010)**, pp. 36 – 44.
[6] M. Song, J. Sun, X. Fu and W. Xiong, "Design and Implementation of Media Player Based on Android", WICOM, **(2010)**, pp. 1 – 4.
[7] D. Gavalas and D. Economou, "Development Platforms for Mobile Applications: Status and Trends", Software, **(2011)**, pp. 77 – 86.
[8] X. Zhao and D. Tian, "The Architecture Design of Streaming Media Applications for Android OS", ICSESS, **(2012)**, pp. 280 – 283.
[9] W. Enck, M. Ongtang and P. McDaniel, "Understanding Android Security", Security & Privacy, **(2009)**, pp. 50 – 57.
[10] Amirante, Castaldi, Miniero and Romano, "Meetecho Mobile: Accessing an IETF-compliant conferencing framework from cellular devices", Communications Magazine, **(2011)**, pp. 36 – 43.
[11] C. Wang, W. Duan, J. Ma and C. Wang, "The research of Android System architecture and application programming", ICCSNT, **(2011)**, pp. 785 – 790.
[12] J. P. Conti, "The androids are coming [Comms]", Engineering & Technology, **(2008)**, pp. 72 – 75.
[13] X. Zhao and D. Tian, "The architecture design of streaming media applications for Android OS", ICSESS, **(2012)**, pp. 280 – 283.
[14] Barbosa, Goncalves, Ribeiro and Costa, "Integration of SIP protocol in Android Media Framework", EUROCON, **(2011)**, pp. 1 – 4.
[15] S. Jin, H. Li and Y. Liu, "Research on media player based on Android", FSKD, **(2012)**, pp. 2326 – 2329.

Authors

Li Ma

She received her B.S. degree in 1985 from the Chengdu Institute of Meteorology and her Ph. D degree in 2011 from Nanjing University of Information Science and Technology. She is a professor and tutor for graduates in Nanjing University of Information Science and Technology. Her main research interests include image processing, pattern recognition, and meteorological information processing and data assimilation.

Lei Gu

He obtained his B.S. degree in the Computer and Software Institute from Nanjing University of Information Science and technology, China in 2012. Now, he is working toward the M.S. degree in the Computer and Software Institute. His main research interests include Data mining and cloud computing.

Jin Wang

He received the B.S. and M.S. degree in the Electronical Engineering from Nanjing University of Posts and Telecommunications, China in 2002 and 2005, respectively. He received Ph.D. degree in the Ubiquitous Computing laboratory from the Computer Engineering Department of Kyung Hee University Korea in 2010. Now, he is a professor in the Computer and Software Institute, Nanjing University of Information Science and Technology. His research interests mainly include routing method and algorithm design, performance evaluation and optimization for wireless ad hoc and sensor networks. He is a member of the IEEE and ACM.