Scientific Research Publishing

Journal of Computer and Communications, 2018, 6, 92-100
http://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Design and Implementation of E-Commerce Platform Based on Android

Jie Shen, Guiling Sun*, Yangyang Li

College of Electronic Information and Optical Engineering, Nankai University, Tianjin, China
Email: 2120170359@mail.nankai.edu.cn, *sungl@nankai.edu.cn, liyangyang@mail.nankai.edu.cn

How to cite this paper: Shen, J., Sun, G.L. and Li, Y.Y. (2018) Design and Implementation of E-Commerce Platform Based on Android. *Journal of Computer and Communications*, **6**, 92-100.
https://doi.org/10.4236/jcc.2018.68007

Received: July 4, 2018
Accepted: August 28, 2018
Published: August 31, 2018

Abstract

This paper introduces the basic functions of the e-commerce system implemented on Android, including user management functions, product search functions, product browsing functions, and product tracking functions, etc. It is necessary to use technologies in Android in order to implement each module function. For example, network communication requires Android's network request technology and data analysis technology. The display of pictures requires the use of Android controls and cache technology. The traceability of products requires camera scanning technology. And RSA decryption technology and so on. Through the above thread pool technology and the use of caching mechanism, the user experience of UI will be improved and unnecessary network resources consumption will be avoided.

Keywords

Android, E-Commerce, Network Communication, Image Display, Product Traceability

1. Introduction

With the continuous development of mobile internet in recent years, more and more attention has been paid to the e-commerce platform and Android mobile users are also increasing.

First, in the e-commerce application, frequent requests will be initiated to the server, so that we will consume a lot of resources, and we know that Http communication can reduce a lot of resource consumption. At the same time, in order to improve performance, the thread pool technology is used in this system to manage each child thread that initiates a network request. In addition, considering that a large number of load pictures are required in the e-commerce platform, if a network pull is required for each request, it will cause application

delay and excessive consumption of traffic. Caching technology can help reduce the burden on mobile devices, improve the performance of applications, and at the same time reduce the consumption of traffic. Therefore, image caching technology is required in the picture display. Finally, because the product's traceability module contains all the information of the production process of the product, in order to encrypt the information of the two-dimensional code, we have enhanced the security through the encryption scheme of DES and RSA hybrid encryption algorithm. Therefore, we have designed and implemented our current Android-based e-commerce platform. As shown in Figure 1.

2. Network Module Implementation

Since the entire system adopts a C/S architecture, the network communication between the Android device and the server is indispensable. In the network communication, the way of network communication and the unity of data format are indispensable. Only when the specific communication method and data format are defined before the network communication, the client and server can communicate normally.

2.1. The Choice of Network Communication Method

Use HTTP communication method in Android. Http is a hypertext transfer

Figure 1. The concrete realization of the picture caching on the main page.

protocol, and Http communication mode uses a "request-response" mode to establish a connection. First the client initiates a data request. And then server responds passively and feeds back the data to the client. After the data is transmitted, the connection is automatically disconnected. Therefore, fewer resources are required. In the e-commerce system, the mobile device initiates a large number of network requests to the server to obtain product information. Therefore, using the Http communication method as the network communication method between the Android device and the server can reduce a lot of resource consumption. In our Android project development, there are two ways to achieve HTTP communication, and achieve data interaction with the server, one is using Http Client, and one is Http URL Connection. Each Http URL Connection instance can be used to generate a single request, but other instances can transparently share the underlying network that connects to the HTTP server. The process is shown in **Figure 2** [1].

2.2. Network Thread Management

Android uses the Reactor mode, or observer mode, because its read and write operations are non-blocking, allowing us to use only one thread to handle all IO events, this approach is synchronous. To improve performance, when a thread receives an event, it considers starting a new thread to handle it, and continues to wait for the next request. But the initiating network request must be in the child thread, network operations in the main thread will throw an exception and cause the program to crash. In e-commerce applications, frequent requests are initiated to the server. Creating a new thread for each request consumes a large amount of resources. Therefore, the thread pool technology is used in this system to manage each sub-thread that initiates a network request. There are several non-stop child threads and a task queue running in the thread pool. Each

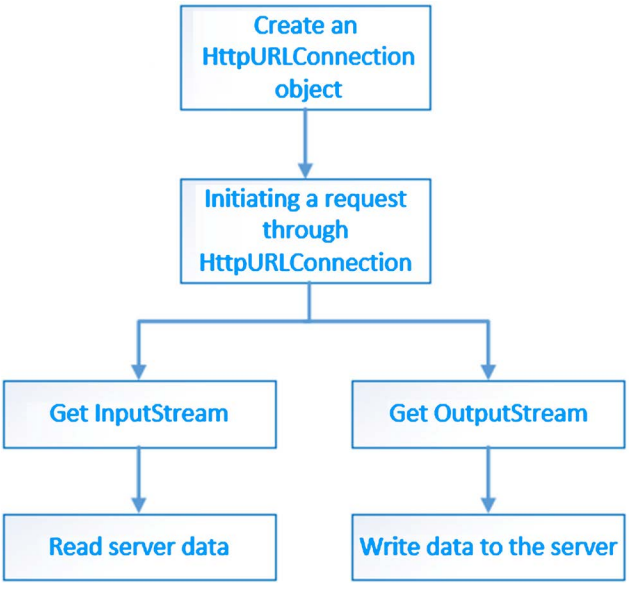

Figure 2. Http communication process in Android.

request is put into the task queue, and then the child threads continuously pull requests from the task queue to execute.

2.3. Json Data Analysis

In this e-commerce system, the data format used to communicate with the server is in the Json data format. Json is a form of data exchange in Google. It is independent of language and independent of platform. Google provides multilingual implementation: Java, c#, c++, go and python. Each implementation includes compiler and library files of the corresponding language. Because it is a binary format, data exchange is much faster than using XML. It can be used for data communication between distributed applications or data exchange in heterogeneous environments. As an excellent binary data transmission format, which is excellent in efficiency and compatibility, it can be used in many fields, such as network transmission, configuration file, data storage and so on [2] [3] [4] [5].

3. Implementation of Picture Display Module

In e-commerce applications, it is necessary to display a large number of pictures, and in a mobile device with limited resources and expensive traffic, if each request needs to be pulled through the network, it will cause the delay of application and excessive consumption of traffic. In order to solve the OOM exception generated by the network picture [6]. We can first adjust the size of the picture, and then recover the unused picture resources in time. Finally, we can help reduce the burden of mobile devices, improve the performance of the application and reduce the consumption of traffic by caching technology, so we need to use the picture caching technology in the picture display.

3.1. Adjust and Recycle Picture Data

First of all, in the image display of the network, many times the picture is limited, so we have to adjust the size of the picture to reduce the size of the resources. We can compress the bitmap through the "options. Injust Decode Bounds" method, so that the resources used in the pictures will be greatly reduced and convenient to load. Then we can recover some of the unnecessary picture resources in time. For example, we can recycle these resources through the "bitmap.recycle" method and release the space in time to avoid OOM anomalies.

3.2. Picture Caching Technology

Memory caching is used in Android, which is helpful to improve the speed of accessing new pictures recently, but there are still some shortcomings. If GridView components display large amounts of pictures, these images will quickly use the buffer space. When the cache space is full, the LruCache will release some old cached images, and we need to reload the old pictures when we use

them again [7]. And when our application is broken back to the background by an application such as the phone, it may be destroyed by the system to recover the memory, and when the application is reentered, all the caching data needs to be reloaded. This scenario requires a memory cache. The memory cache can persist these loaded images and shorten the time of loading pictures again when the data in the memory cache is no longer available. Of course, loading pictures from memory will be slower than loading from memory. Because the time to read memory is unpredictable, background threads should be used to load them. In addition to loading and parsing data from the thread pool, the Image Loader framework can be used to replace the original Image Request framework to achieve specific image loading. The constructor of the Image Loader framework has a cache parameter that displays default images without pictures, which consumes no traffic at this time and when the request fails Callbacks to the corresponding images also check if the local cache is available when the request is loaded successfully to further optimize the UI carton and avoid a large amount of traffic loss.The image cache loading is shown as shown in **Figure 3**.

In Android, a class of memory caching and disk caching based on the most recently used algorithm is provided. The memory cache class is the Disk LruCache class. Obviously, every time we update (called the put method) or access

Figure 3. Image cache loading display.

the node in the map (called the get method), the inner part of the Linked Hash Map will move the node to the tail of the list. Therefore, at the end of the linked list is the recently used node, where the head of the linked list is the most recently used node, when our cache space should be removed from time to time to remove the head node of the list, until there is room for the rest of the list. As you can see, the core functionality in DiskLruCache is completed, and what remains to be done in DiskLruCache is to define the total capacity of the cache space, the current capacity to save the data, and the put and get methods. The implementation process is shown in Figure 4.

4. Product Traceability Module

The function of product traceability module is to use the QR code information of the scanned product to contain the information of the production process in the information of the QR code. Encryption algorithm is used to encrypt plaintext files, and RSA encryption algorithm is used to encrypt the key. The work flow of the entire traceability function is shown in Figure 5.

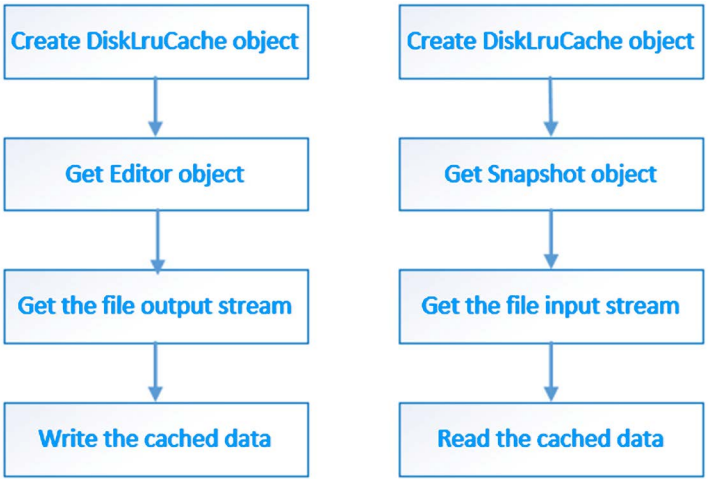

Figure 4. Write and read the disk cache data.

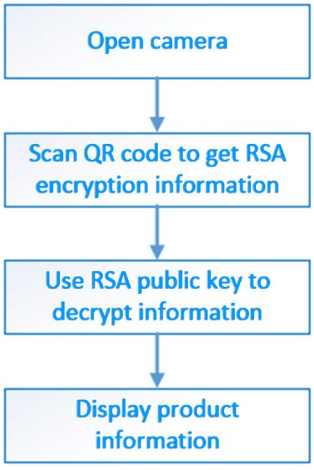

Figure 5. Commodity tracing function process.

4.1. The Realization of Camera Scan QR Code Function

The SDK in Android does not directly realize the class of QR code information using a camera, so it takes two steps to realize the function, one is to open the camera and the two is to analyze the information of the QR code by the image collected by the scanning camera. Google provides a class library that specializes in scanning and parsing QR codes. The method of using the class library is very concise [8]. Only the camera class Camera and the setting callback method "handle Decode ()" can open the camera and scan the QR code information and return the information to the callback method. But this method is not safe and easy to crack. Only through RSA encryption and other technologies to prevent malicious cracking or product information is modified, but also to achieve only professional reading equipment can read the encrypted information, to give users accurate traceability information. At the same time, it can activate the traceback code, only the traceback code can be read by the user. When the product is not sold, it can not read information even if it is illegally copied. The final implementation of a two-dimensional code is shown in Figure 6.

4.2. Implementation of Key Decryption Function of DES and RSA

1) Implementation steps of encryption scheme for DES and RSA hybrid encryption algorithm.

a) To generate DES encryption key K, in order to improve the security of data, each key K is used only once.

b) The key Ck used to encrypt plaintext is generated by using the public encryption

Figure 6. The final realization of the QR code platform.

key of RSA encryption algorithm to encrypt the key K of DES, and form the DES encrypted key Ck and save it.

c) Generate ciphertext C, use the generated key Ck to encrypt plaintext P DES and generate ciphertext.

2) The decryption principle of DES and RSA's hybrid encryption algorithm is decrypted by DES algorithm, and DES key is decrypted by RSA algorithm.

3) Specific implementation steps of decryption scheme based on DES and RSA hybrid encryption algorithm.

a) Get the encryption key Ck.

b) The key K used for DES decryption is generated. The key Ck of DES is decrypted by the decryption key of RSA, and the key K of DES decryption is formed.

c) Generate plaintext P.

4) The implementation steps of DES RSA mixed encryption algorithm

a) Private key generation.

```
openssl genrsa -out private.pem 1024
```

b) Convert the RSA private key into PKCS8 format

```
openssl pkcs8 -nocrypt -topk8 -in private.pem -out pkcs8.pem
```

c) Generate public key

```
openssl rsa  -in private.pem  -pubout -out public.pem
```

d) As follows: the export can be derived

```
[root@izm5e5fyju4es2mn4m9t66z ~]# ls
my-v1.1.3  pkcs8.pem  private.pem
[root@izm5e5fyju4es2mn4m9t66z ~]# openssl rsa -in ddmdd_a.key -pubout -out ddmdd_a.pub.pem
Error opening Private Key ddmdd_a.key
140329869866912:error:02001002:system library:fopen:No such file or directory:bss_file.c:402:fopen('ddmdd_a.key','r')
140329869866912:error:20074002:BIO routines:FILE_CTRL:system lib:bss_file.c:404:
unable to load Private Key
[root@izm5e5fyju4es2mn4m9t66z ~]# openssl rsa  -in private.pem  -pubout -out public.pem
writing RSA key
[root@izm5e5fyju4es2mn4m9t66z ~]# pwd
/root
[root@izm5e5fyju4es2mn4m9t66z ~]# ls
my-v1.1.3  pkcs8.pem  private.pem  public.pem
[root@izm5e5fyju4es2mn4m9t66z ~]#
```


5. Conclusion

The e-commerce functions implemented in this paper on the Android platform include not only commodity display, commodity traceability and other functions, but also optimization in the following aspects: caching technology is used in network communication request technology so that local caching can be used in the case of local loading to avoid. The repeated network requests optimize the user experience of mobile terminals. And the Json parsing technology is used to make the network request more convenient and fast. In addition, in the Android mobile side of a large number of image loading, the use of Image Loader cache loading framework instead of the original Image Request framework to load im-

ages and UI Carton to further optimize. Finally, in the traceability module of the product, the camera scanning technology and RSA decryption technology are adopted, and the user's UI experience is further optimized by creating a multi-tasking thread pool technology and the use of caching mechanism. At the same time, the unnecessary network requests are avoided maximally.

Acknowledgements

This work was partially supported by Tianjin Science and Technology Major Project (2017ZXHLNC00100) and supported by Tianjin Rural Working Committee and Tianjin Key Laboratory of Optoelectronic Sensor and Sensing Network Technology.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Cabir, Smartphone Malware (2004) http://www.f-secure.com/v-descs/cabir.shtml

[2] GDATA, Mobile Malware Report (Threat Report: Q4/2015) (2016) https://secure.gd/dl-us-mmwr201504

[3] Rieck, K., Holz, T., Willems, C., Düssel, P. and Laskov, P. (2008) Learning and Classification of Malware Behavior. *Proceedings of the* 5*th International Conference on Detection of Intrusions and Malware*, *and Vulnerability Assessment*, Paris, 10-11 July, 108-125. https://doi.org/10.1007/978-3-540-70542-0_6

[4] Griffin, K., Schneider, S., Hu, X. and Chiueh, T.-C. (2009) Automatic Generation of String Signatures for Malware Detection. *Proceedings of the* 12*th International Symposium on Recent Advances in Intrusion Detection*, Saint-Malo, 23-25 September 2009, 101-120. https://doi.org/10.1007/978-3-642-04342-0_6

[5] Joshua, A., Waziri, O.V., Abdullahi, M.B., Arthur, U.M. and Adewale, O.S. (2015) A Machine Learning Approach to Anomaly-Based Detection on Android Platforms. *International Journal of Network Security* & *Its Applications*, **7**, 15-35. https://doi.org/10.5121/ijnsa.2015.7602

[6] Allix, K., Bissyandé, T.F., Jérome, Q., Klein, J., State, R. and Traon, Y.L. (2016) Empirical Assessment of Machine Learning-Based Malware Detectors for Android. *Empirical Software Engineering*, **21**, 183-211. https://doi.org/10.1007/s10664-014-9352-6

[7] Georgiou, N., Konstantinidis, A. and Papadopoulos, H. (2016) Malware Detection with Confidence Guarantees on Android Devices. *Proceedings of the* 12*th IFIP WG* 12.5 *International Conference and Workshops, Artificial Intelligence Applications and Innovations*, *AIAI* 2016, Thessaloniki, 16-18 September 2016, 407-418. https://doi.org/10.1007/978-3-319-44944-9_35

[8] Amos, B., Turner, H. and White, J. (2013) Applying Machine Learning Classifiers to Dynamic Android Malware Detection at Scale. *Proceedings of the* 9*th International Wireless Communications and Mobile Computing Conference* (*IWCMC* 2013), Sardinia, 1-5 July 2013, 1666-1671. https://doi.org/10.1109/IWCMC.2013.6583806