MOBILE PHONE AND BARCODE SCANNER

MOHANAD ABDULHAMID¹, ACHIKI MATONGO²

¹AL-Hikma University, Iraq, ²University of Nairobi, Kenya
E-mail: moh1hamid@yahoo.com, researcher12018@yahoo.com

Abstract *With the advancement in microprocessor technology, mobile phones are becoming more preferable in our day to day lives as compared to personal computers since most phones are capable of functionalities as sophisticated as our PC's or laptops. Mobile phones nowadays have processor speeds as high as 1GHz and can have a storage capacity of about 32Gb and they also have long hours of battery life. In addition to these feature, there is the aspect of great portability. When all these are combined into one, the mobile phones can be viewed as a very powerful device that rivals PC's. Advancements of hardware also had made mobile phones much smaller and more efficient whereas including an increasing of peripherals number. The scope of this paper is basically to make a mobile phone application and how to interface the mobile phone to the scanner. The mobile application is to be developed in the Linux-based Android platform. An algorithm based on the operation principles is also to be developed during the work.*

Keywords: Mobile phone, barcode scanner, interface

1. INTRODUCTION

In 2007, a team of manufacturers of handset, wireless carriers, and developers of software program (notably, Google) formed the Open Handset Alliance (OHA), with the aim of growing the subsequent technology of wireless platform. Unlike current platforms, this new platform would be nonproprietary and based totally on open standards, which would lead to decrease improvement costs and accelerated profits. developers of mobile software would also have remarkable get entry to the handset features, permitting for higher innovation.

There are already several mobile platforms on the market nowadays, which include Symbian, Windows Mobile, BlackBerry, Java Mobile Edition (JME) but Android is considered the first complete, open, and free mobile platform that was made by Google and the OHA. The other systems offer some access to the hardware of device, but need writing complex C/C++ code and making heavy use of proprietary application programming interfaces (APIs) which are not easy to use. These proprietary operating systems which often prioritize native applications over the ones made by third parties and restrict communication amongst applications and native phone data.

Android is constructed on top of a strong and verified Linux kernel basis. Linux affords the hardware abstraction layer for Android, permitting Android to be ported to a wide form of systems in the future. Internally,

Android utilizes Linux for its memory management, processing management, networking, and different operating system offerings. Android is designed not just for cellular telephones but additionally for different hardware structures.

The aim of this paper is to develop a mobile phone application based on the Android platform. Here, a mobile phone and a barcode scanner are to be interfaced such that a mobile phone application is to be created and also an interface that receives input from a barcode scanner and display item details retrieved from a central database. Several researches that discuss this topic can be found in literatures [1-5].

2. DESIGN METHODOLOGY

In the design, the Android mobile phone is to be interfaced to the barcode scanner with the help of a Universal Serial Bus (USB) cable. The USB is designed with the goal of being simple to use and low cost of implementation. The USB is previously regarded as a high speed serial bus. Serial configuration is generally preferred as compared to parallel configuration owing to cheap cables and easier mode of implementation.

2.1 USB in mobile phone

The mobile phone basically has to be converted into a host device so as to see the barcode scanner as a peripheral and therefore the phone can retrieve whichever information the scanner acquires. The mobile phone basically does not have enough power to drive the USB peripheral and thus an external source of power for the USB is required. Mobile phones with USB-host capabilities is a major milestone in the advancement in mobile technology since pretty much any device can be interfaced with the mobile so long as it has a USB connector. Combining the extreme portability of the mobile phone and the versatility of the USB opens a wide range of applications.

The Android platform also does not support USB-host capabilities. Therefore, for the Android phone to support USB, drivers have to be written and patched to the Kernel of the phone. After this, the Android kernel is then recompiled so as to integrate the USB drivers into the platform.

Some mobile phone manufactures have been able to add USB-host capabilities to their products for example the Motorola DroidX, NEXUS 1 and Toshiba.

The possibilities for the interfacing of mobile phones and various peripherals are enormous and may prove to be very useful in various fields.

For example, a retailer might be able to request for more goods from a supplier by sending only the codes of the goods he needs rather than writing a whole bunch of details relating to the goods he wants. Since barcodes can be encoded with all the necessary information relating to a particular item, all the retailer and the supplier need is a means to access a central database through the mobile phone and the necessary services/orders are carried out.

Doctors may also be able to quickly reference a patient's details in a more efficient and fast way. This can all be done if the mobile phone is able to access the records in a remote location.

2.2 Implementation

In the implementation stage, the Android Software Development Kit (SDK) environment is used in the Eclipse platform. For the creation of databases, SQLite is used.

2.2.1 Database implementation

2.2.1.1 Preview of SQLite

SQLite is considered as a relational database management system same as Oracle, IBM's DB2 and Microsoft SQL server. SQLite had a reputation of being highly reliable and is the database system of choice for several consumer electronic devices, including several MP3 players, the iPhone and the iPod Touch and of course Android and Symbian phones. SQLite is commonly used because of the following features:

- It is open-source; it is placed in the public domain.
- It is lightweight and hence can be store within the memory allocation of the mobile phone.
- It requires no setup or administration. There is no server, no *config* file and no need for a database administrator unlike the standard SQL.

SQLite is differing from several traditional database engines in the usage of a loosely typed technique to column definition. In preference to requiring column values to conform to a single kind, the values in each row for every column are individually typed. As a result, there is no strict type checking whilst assigning or extracting values from every column within a row.

2.2.2 Barcode implementation

For the mobile phone to be able to read and decode barcodes, a ZXing zebra crossing (ZXing) library is used. ZXing is an open-source, multi-format 1D/2D barcode image processing library designed in Java. It is using the built-in camera on mobile phones to photograph and decode barcodes on the device, without communicating with a server. ZXing supports several barcode formats.

2.2.2.1 Encoding of barcode

The ZXing library encodes the barcode by considering that a "1" represents a "bar" or a "dark" section and a "0" represents a "light" or "space" section of the bar code. A double dark section is viewed as a 11 and a double space or light section is viewed as a 00.

Before the scanner encodes the value of the barcode, the checksum digit is firstly calculated and if it is valid, the code is scanned else an error is returned i.e. the barcode is viewed as an invalid code. Validity of the checksum digit is such that the check digit (i.e. last digit of the barcode) is cross-referenced with the value computed by the barcode scanner application. Therefore, the two values have to tally/match for a valid barcode.

Guard bars are usually encoded as dark-light-dark sections or bar-space-bars which can be viewed as 101. This applies for all the guard bars i.e. Left, Center and Right guard bars.

Steps for encoding/decoding the barcode are summarized as follows:

1. Compute the checksum for validity
2. Check the number system digit value and cross-reference the respective odd-even parity sequence with the parity encoding table.
3. Encode each number with the corresponding 10 sequence using the character set encoding table taking into account the respective parity of the characters whether even or odd parity.

2.2.2.2 Database design and program algorithm

The program follows the following steps during its implementation:

- The scanning application is first called into action via an intent.
- The symbology and barcode format are determined and the check digit is then confirmed for validity of the code.
- Scanning is then initiated.
- The database is then queried.
- The program then cross-references the bar code scanned to the codes in the barcoded field in the database.
- If the barcode code is found in the database, the corresponding image and details are displayed.
- The program then exits.

After the creation of the database, the next step is to make the Graphical User Interface (GUI). The GUI comprises of a scan button that is used to prompt for the opening of the barcode scanning application. Firstly, the main layout is designed using XML (Android uses XML for its GUI). The button is then placed on the linear layout of the GUI.

To navigate to the barcode scanning application, we have to call the ZXing library via intents. Once the barcode has been detected and recorded, another function is performed to record the scanned code on the main layout.

3. RESULTS

This section shows the results that are observed when the application is run. The default screen is as shown in Figure 1 when the application is started. On clicking the “Scan Barcode” button, the view on Figure 2 is shown. The red line is the one responsible for scanning the barcode i.e. the widths of the bars and spaces.

Figure 1. Initial screen

Figure 2. Barcode scanner view

Once the barcode is recognized and the value of the barcode determined, the product name and the image of the product are displayed as shown in Figure 3.

Figure 3. Displayed image and name

The GUI of the database that is created after scanning the barcode is as displayed in Figure 4. At the first instance of the scan, the barcode number that is displayed in Figure 3 is inserted/appended into the barcoded field. The corresponding product name is also displayed in the product field. For the image field, the path to where the camera stored the captured product image is referenced.

id	BarcodeId	Product	Image
1	87126037	Coca Cola	/sdcard/images/cocac
2	6923520260263	Edifier Speakers	/sdcard/images/eidife
3	8887290101219	MacCoffee 3 in 1	/sdcard/images/macC
4	6161100010017	Mumias Sugar 1Kg	/sdcard/images/mumi
5	4005808233502	Nivea Aqua Cool	/sdcard/images/nivea
6	6161100604605	Omo 1Kg	/sdcard/images/Omo
7	6161103210322	Ramtons Cordless Kettle	/sdcard/images/ramto
8	61601055	Vaseline Blue Seal	/sdcard/images/vaseli

Figure 4. Database created

Figure 5 shows how the application displays in the event that the barcode scanned is not found in the database.

Figure 5. Product not found

4. CONCLUSION

The main objective of the paper was to interface the mobile phone to a barcode scanner. In the paper, an alternative method of scanning was introduced. This is because the android platform is a new technology and does not currently support USB-hosting capabilities and thus the need to use an alternative way of scanning whatever barcode we have. Using the mobile phone camera, the work was a success since a barcode could be scanned and eventually retrieved from a central database as in the objective.

5. REFERENCES

- [1] E. Ohbuch, H. Hanaizumi, and L. Hock, "Barcode readers using the camera device in mobile phones", International Conference on Cyberworlds, Japan, 2004.
- [2] F. Reischach, S. Karpischek, F. Michahelles, and R. Adelman, "Evaluation of 1D barcode scanning on mobile phones", IEEE Conference on Internet of Things, Japan, 2010.
- [3] N. Hashim, N. Ibrahim, N. Saad, F. Sakaguchi, and Z. Zakaria, "Barcode recognition system", International Journal of Emerging Trends & Technology in Computer Science, Vol.2, Issue 4, PP.1-6, 2013.
- [4] P. Subpratatsavee, N. Janthong, P. Kuha, and C. Chintho, "HC2D barcode reader using embedded camera in Android phone", IEEE 11th International Joint Conference on Computer Science and Software Engineering, Thailand, 2014.
- [5] S. Son, and Y. Shin, "Design of smart shopping application using barcode scanning and location based coupon service", IEEE 8th International Conference on Grid and Distributed Computing, South Korea, 2015.