2010

Building Applications for the Android OS Mobile Platform: A Primer and Course Materials

Victor Matos
Cleveland State University, v.matos@csuohio.edu

Rebecca Grasser
Lakeland Community College, bgrasser@lakelandcc.edu

Original Published Citation
Matos, V., Grasser, R. (2010). Building Applications for the Android OS Mobile Platform: A Primer and Course Materials. JCSC, 26(1), pp. 23-29.

BUILDING APPLICATIONS FOR THE ANDROID OS MOBILE PLATFORM: A PRIMER AND COURSE MATERIALS*

Victor Matos
Computer and Information Science Department
Cleveland State University,
Cleveland OH 44114
v.matos@csuohio.edu

Rebecca Grasser
Information Technology & Computer Science Department
Lakeland Community College
Kirtland, OH 44094
bgrasser@lakelandcc.edu

ABSTRACT

This paper has two parts: first a brief tutorial on Android OS and then we comment on our experience teaching a first offering of this material. The application in the tutorial is based on a case of reverse geo-coding (transforming a textual address to coordinates and producing its corresponding representation on a map). This example contains components normally included in a typical Android application, and although simple it is not trivial. We argue in favor of adopting a mobile programming experience in the CS curriculum. We believe students will find in Android a rich platform on which a variety of concepts, techniques, and resources can be combined to produce useful and marketable applications (a situation not often found in the CS curriculum). We provide suggestions on laboratory setups and some potential research projects.

Keywords: K.3.2 [Computers and Education]: Computer and Information Science Education - Computer science education, Curriculum. C.3 [Computer System Organizations] Special-Purpose and Application-Based Systems: Real –time and embedded systems.

1. INTRODUCTION

Cellular telephony has had a significant worldwide rate of acceptance, by year 2010 it is estimated that 3.5B of the 6.8B people in the planet will have access to a cell phone [4, 10]. Smartphone devices such as iPhone, Blackberry, and those that support the Android operating system are progressively making an impact on society. In addition to their support for voice and text exchange, smartphones are capable of executing sophisticated embedded software applications, as well as provide a simple link to the Internet and its resources.

It is reasonable to assume many of our students will engage in the development of the next generation of software applications for embedded and mobile devices. As educators we should recognize the importance of this reality and make our curriculum respond to the challenge. While introducing mobile computing is listed as optional in CS2008 [5], one of the outcomes of this area is to "Implement a simple application that relies on mobile and wireless data communications." To illustrate some concepts of Android development and to help students meet this outcome, we require the construction a simple mapping application, as well as a more complex application at the end of the term.

In the next section we take a brief look at the main components of the Android Operating System and the development of the simple mapping system. In the final section we discuss our experiences and provide guidelines for organizing a course on system development for Android-based devices.

2. ANDROID OS

Android OS includes a large set of features for supporting mobile applications. At the heart of Android there is the *Application Framework* enabling reuse and replacement of components. Android has its own virtual machine based on Linux but optimized for mobile devices.

The Android developer creates code solutions in Java, the development environment includes a device emulator, tools for debugging, memory and performance profiling, and a plugin for the Eclipse IDE [3]. Each Android application runs in its own Linux process. An application consists of a combination of components including: *Activities, Services, Broadcast Receivers,* and *Content Providers* [2]. *Activities* roughly correspond to a form or “screen”; they are invoked directly or indirectly using Intents. An *Intent* is a mechanism for calling a method, passing data, and receiving results. For instance, placing a phone call and showing a YouTube webpage can be done as follows:

```
Intent myActivity1 = new Intent (Intent.ACTION_CALL, Uri.parse("tel:555-1234"));
startActivity (myActivity1);
Intent myActivity2 =new htent (Intent.ACTION_VIEW, Uri.parse("http://www.youTube.com"));
startActivity (myActivity2);
```

A Service is an application component that runs in the background, has no GUI, and does not interact with the user for an indefinite period of time. For example the GPS hardware could be activated to collect positional information while other non-location based applications (such as music player) may work in the foreground without interacting with the GPS service. If an application wants to receive and respond to a global event, such as the phone ringing or an incoming text message, it must register a BroadcastReceiver. Finally, Content Providers globally expose and update persistent data as a simple table on a relational database [2].

Although Android is a real multi-tasking, multi-threaded OS, its activity manager is stack oriented. While the stack-oriented behavior is customizable, we illustrate the general concept with a basic example. Assume application AP1 (say, a music player) is running, the user (without stopping AP1) calls for a video-game AP2. Then the user requests a weather update calling AP3. Android pushes AP3 on top of AP2, which is already on top of AP1. Pressing the back button pops AP3 from the system's stack. The unfinished video-game AP2 is exposed. Pressing the back button one more time pops AP2 and AP1 is shown. Central to the Android OS architecture is the notion of application's life-cycle. Unlike most systems an Android application does not directly control itself. Its life span is determined by the OS after evaluating a number of factors including: how much of the app is running, how important these things are to the user, and how much overall memory is available in the system. The Android Developer's Guide [2] has a complete description of the stack process and the application life cycle.

3. ADONDE? - A MAPPING PRIMER

We use the project of a simple mapping system (called *Adonde?* in our lab write-up) to introduce students to the process of designing and building a mobile application. The project specifies that "*the user of this application willtype in a valid address and a map of the given place will be drawn on the phone's screen*" Figure 1-(*Left*) depicts on a road map the location of "Franklin College, IN". Observe there are view controls to zoom in and out. The user can also touch the screen and drag the map in any direction.

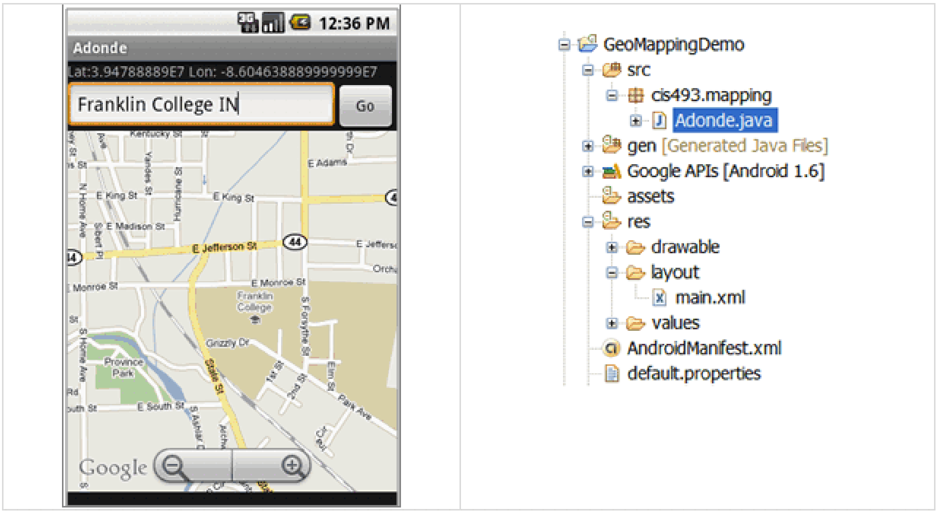

Figure 1: (Left) Screen-shot of the Android *Adonde* application showing a map and coordinates of "Franklin College IN". (Right) Structure of the app shown from Eclipse's Package Explorer

We start the lab by assuming the students have installed the most recent Android SDK [2], obtained a Google Maps API key [8], and their Eclipse IDE includes the ADT plugin [3]. The basic concepts of developing an Android application are discussed, including the file structure of the application. When a new Android application is created the Eclipse ADT provides the basic skeleton of the solution. The code is stored in a package inside the */src* folder; resources such as pictures, files, etc. are stored in the */res* subdirectory (see Figure 1-*Right*). In our example, the class *Adonde* extends Android's native MapActivity interface which provides means of displaying and navigating a map held in a MapView container. The GUI definition – holding the MapView widget- is stored as an XML document in the */res/layout/main.xml* file. The *AndroidManifest.xml* file contains a list of all the activities, and special permissions requested by the application.

Once the students have explored and understand the file layout, they are given the lab assignment to complete. Complete details on the coding of similar labs can be found at the university web site for this course [9].

4. COMMENTS ON OUR TEACHING EXPERIENCE

Our first offering of the Android OS course targeted a small class consisting of fourteen CS and engineering majors [9]. Students had already completed a course in data structures using Java; we did not assume a background in networking and databases. The semester course was delivered using traditional lectures, weekly lab assignments, and a major team/individual project presented to an audience at the end of the semester. The course materials and teaching resources were posted on the university website [9].

We were fortunate to have the assistance of local experts on mobile and embedded programming. We had three guest lectures on *iPhone OS* development during the semester. Although small coverage was given to this non-Android platform, it seemed to be beneficial as it provided an opportunity to compare and contrast features, and methodologies of these competing systems.

Android OS is a new platform and it is been constantly reviewed and updated; this makes the selection of a textbook very difficult. Excellent on-line documentation of the Android OS system is available at the Android Developers website [2]. We chose Mark Murphy's [11] text for the main reference as there are annual subscriptions available on his website. Finally, Abelson, Collins, et.al. [1] is a reasonable choice for an additional textbook. The instructor should be prepared to spend many hours writing (and updating) lecture notes and examples as the underlying structure of the platform changes and the product becomes more mature.

Students do not have to own an Android device to do their work. Almost all lab projects can be tested in the *Android Emulator*. Clearly some limitations exist, however the *Emulator Control Tool* provides support for telephony (sending and receiving phone calls and text-messages to other emulators) as well as GPS Location control. For equipping your laboratory we suggest acquiring a number of unlocked devices called *Android Developer Phones*(*ADP*) [2]. Currently only a few models of ADP are available from Android Market ($400.00-$600.00 per unit). If available, students can borrow the devices from the school lab, and insert their personal phone's SIM card to activate the ADP. It should be noted that not all applications require telephony as there are other ways (WiFi, for example) to provide students with the opportunities to meet the outcomes given in CS2008 [5].

While learning the foundations of Android development, our students worked in identifying a problem and producing a solution. Obviously it is very difficult to commit to a major application when still learning introductory material. We worked around this problem by selecting a number of solutions already in the Android market and re-engineering and modifying to local needs the chosen application. Students could work alone or in pairs, and were asked to select one of the projects from the given list. For instance, one of the projects called '*Go Cavs*' shows live scores for the local basketball team, as well as its schedule, team's web-site material, and so forth. Another interesting application called '*Ride & Roll*' displays on a Google map the location of the user, the closest bus/train station, and the current location of the bus or train. The goal is to tell the user when the next public transportation unit is approaching her location. Another project called '*McTalk*' uses the voice recognition unit to assemble and place an order at a McDonald's restaurant before approaching the drive-thru booth. As a final example, the application '*I am Ok*' [10] assembles a text-message containing text and GPS location of the sender and delivers to an emergency database from where data can be accessed by friends and family.

Some of these projects were not fully completed by the end of the semester; however the students had most of the conceptual design and GUI specifications to illustrate their application's functionality. The major impediment to project completion was time - the project proposal was simply too large to complete during the semester. We found this to be an important concept - as students learn to propose time requirements for a project, they need to understand how they work through complex undertakings. Each team was asked to formally present their work in the classroom. Students presented the proposal, the artifacts from the development process, the application, and thoughts for future work. Students were encouraged, but not required, to investigate the Android marketplace. While intellectual property was discussed during the course of the semester, the course curriculum concentrated on the development process. Any student work on the Android Marketplace was, by necessity, put there by the student on their own.

The reading of the course evaluation indicated that the material and format of the course raised the student's level of enthusiasm and participation. Students interacted in the classroom and worked in small teams. Overall there was a noticeable positive attitude, excitement, and sense of collaboration. Class discussion of team projects provided opportunities for self-criticism and self-improvement as well as peer evaluation.

The field of mobile application development is fertile ground for innovators. A few of the independent projects created in the course are now for sale in the Android Market. Most students commented about their willingness to continue 'polishing' their solution on their own time to make them be more attractive in the actual market.

5. CONCLUSION

In many traditional computer science programs, students mainly develop applications for desktop computing. Ideally you would like to prepare your students to be ready for as many as development platforms as possible. While there has been a recent addition of multimedia and robotics subjects in our curriculum[6], many of the students at our institution have not been fully exposed to skills that should transfer to employability. The addition of mobile computing to their academic program adds another level of authenticity c to their work [7].

The inherent complexities in developing mobile applications tests the student's ability to design, develop, test, and release an application that may be an order of magnitude more challenging than a desktop application. We believe mobile computing has become a necessary component in the education of CS majors. It does not matter what platform you choose; we tend to favor Android because of the openness of tools and resources, there is no cost to the student to test and distribute their code; however other alternatives are equally acceptable.

In our limited experience we saw students working with a great deal of enthusiasm in their individual projects. Perhaps the motivation comes from the novelty of the media and the opportunity for innovation. Getting ready to teach the material is not different from preparing other courses. Android development has a very low setup cost as all the tools are free, no special hardware is required, and the simulator is powerful enough to mimic almost all the functionality of the actual device. The strategy used in our first offering was depth in a single platform, we hope in the next offering to provide a survey of different mobile application environments to supplement the student's learning.

6. REFERENCES

[1] Abelson, W.F., Collins, C., Sen, R. *Unlocking Android - A Developer's Guide*. Manning Pub. April, 2009.

[2] Android Developer's Guide. http://developer.android.com/, Retrieved Feb 2010.

[3] Eclipse Foundation. Eclipse IDE – Plugin for Android. http:// www.eclipse.org. Retrieved Feb 2010.

[4] Central Intelligence Agency. 2008 World Factbook – Guide to Country Profiles - Telecommunications. https://www.cia.gov/library/publications/the-world-factbook/docs/profileguide.html. Retrieved March 2010.

[5] Computer Science 2008, An Interim Revision of CS 2001 (http://www.acm.org//education/curricula/ComputerScience2008.pdf). Retrieved May 2010.

[6] Guzdial, M., Ranum, D., Miller, B., Simon, B., Ericson, B., Rebelsky, S. A., Davis, J., Deepak, K., and Blank, D. 2010. Variations on a theme: role of media in motivating computing education. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education* (Milwaukee, Wisconsin, USA, March 10 - 13, 2010). SIGCSE '10. ACM, New York, NY, 66-67. DOI= http://doi.acm.org/10.1145/1734263.1734287

[7] Guzdial, M. and Tew, A. E. 2006. Imagineering inauthentic legitimate peripheral participation: an instructional design approach for motivating computing education. In *Proceedings of the Second international Workshop on Computing Education Research* (Canterbury, United Kingdom, September 09 - 10, 2006). ICER '06. ACM, New York, NY, 51-58. DOI= http://doi.acm.org/10.1145/1151588.1151597

[8] Obtaining a Google Maps API Key. http://code.google.com/android/add-ons/google-apis/mapkey.html. Retrieved Feb 2010

[9] Matos, V. CIS493 – Special Topics in Computer Science. http://grail.cba.csuohio.edu/~matos/notes/cis-493/Android-Syllabus.pdf. Retrieved February 2010.

[10] Matos, V., Blake, B. ‘I am OK’ - A Conceptual Model for a Global Emergency System and Its Societal Impact. *International Journal of Technology, Knowledge and Society*. 2(5), 7-18, 2007.

[11] Murphy, M.L. *The Busy Coder's Guide to Android Development*. CommonsWare Pub., 2009 (available at: http://commonsware.com/Android/index.html).