2014 Hawaii University International Conferences
Science, Technology, Engineering, Math & Education
June 16, 17, & 18 2014
Ala Moana Hotel, Honolulu, Hawaii

A Modular Approach to Teaching Mobile APPS Development

Alrifai, Rad
Northeastern State University
Computer Science Department

Rad Alrifai
Computer Science Department
Northeastern State University

A Modular Approach to Teaching Mobile APPS Development

Synopsis:

To better relate mobile applications development to other topics from computer science and adapt to the continuous evolution in the mobile development environment, apps development course content is organized as a model consisting of five modules. Throughout these modules, mobile apps development is treated as an integral part of the computer science curriculum.

A MODULAR APPROACH TO TEACHING MOBILE APPS DEVELOPMENT

Rad Alrifai
Northeastern State University
Tahlequah, OK 74464, USA

ABSTRACT

Mobile devices are progressively replacing personal computers for many types of users, leading to a growing demand for mobile applications (apps) development. However, apps development requires rigorous programming and strong software development knowledge. Hence, apps development can be connected to other topics in computer science. To better relate mobile applications development to other topics from computer science and adapt to the continuous evolution in the mobile development environment, apps development course content is organized as a model consisting of five modules. Throughout these modules, mobile apps development is treated as an integral part of the computer science curriculum.

1. INTRODUCTION

Mobile devices continue to replace personal computers (PCs) for many daily computing uses. According to Reuters, sales of desktops and laptops will continue to decline. In 2013, the sale of PCs declined by 11.2% from its level in 2012. "Gartner forecast. Shipments of tablets are expected to rise 53.4 percent to 184 million. Traditional desktop and laptop computers will continue to decline "[8]. The increase in demand for mobile devices is leading to an increase in demand for apps development. According to the United States Bureau of Labor statistics, apps development is contributing to the increase in demand for employment in software development, "Employment of applications developers is projected to grow 28 percent, and employment of systems developers is projected to grow 32 percent. The main reason for the rapid growth is a large increase in the demand for computer software. Mobile technology requires new applications." [12]

However, several alternative apps development platforms and options are available for developing apps within each platform. Also, other challenges must be overcome when designing a new course on this topic. One challenge stems from the volatility and continuous revisions of the mobile development platforms. After considering various mobile apps platforms for a course on this topic, the Android platform was chosen. The Android operating system is widely used. It accounted for 79% of all worldwide smartphone operating systems in the second quarter of 2013 [4]. Furthermore, the code for the Android platform is available as an open source. Having the advantage of not being proprietary, the Android development environment is an attractive choice to use in a course on mobile apps development. Moreover, the Android developers' website [2] provides comprehensive resources, tutorials, and examples of using the Android Software Development Kit (SDK) for apps development.

After choosing the Android platform, the next consideration was whether to select HTML5 or the Android native code for apps development. On one hand, applications developed in HTML5 are easily portable to other platforms and hence have the potential to reduce development time in cross-platform development. Although native code tends to limit mobile applications to a specific mobile platform, the native code was chosen because of its underlying strengths. In

addition to being fast, reliable, secure, readable, and simple, native code development offers a large library, enforces strong typing, compiles into Dalvik byte-code, supports multithreading, enhances productivity, and improves apps performance. Also, native code has the advantages of providing better hardware access through application programming interfaces (APIs) and better support for implementing native widgets. Because of these advantages, many apps developers favor native applications development.

2. THE COURSE DESIGN

The growing demand for mobile devices is creating a growing demand for mobile applications development. Moreover, topics covered in apps development are closely linked to other computer science topics. Thus, a course has been developed with the intention to build on various topics taught in computer science and enhance apps development as an integral part of improving the quality of software development in general. The course applies graphical design to course objectives as they relate to the Android development platform. Furthermore, the course content is organized into a five-module model where each module covers a different area of apps development.

The course title is "Mobile Applications Development." It is designed as a three-hour undergraduate elective in computer science and has the following topics as prerequisites:

- Coding in C#, C++, or Java (preferred).
- Familiarity with using programming libraries.
- Knowledge of using the object-oriented paradigm.

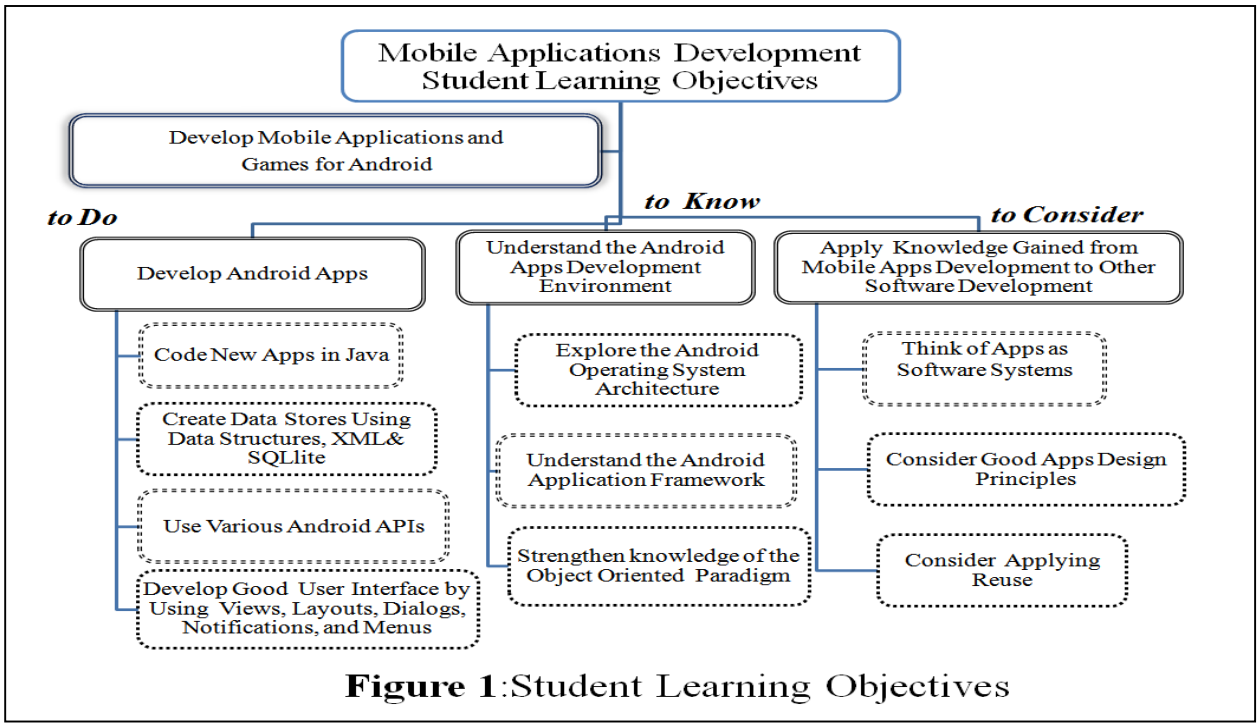

Figure 1:Student Learning Objectives

Figure 1 depicts the student learning objectives based on a modified form of the Graphic Display of Student Learning Objectives that was described in the Chronicle of Higher Education [5] with some additional details added. The course objectives are organized into three categories to reflect the three aspects of learning objectives as described in "How to Write Program

Objectives/Outcomes" [11]. Thus, the course learning objectives are organized into three categories: what the student should learn to do, know, or consider.

The layered approach has been widely used in computer science. Organizing a system into components is common in many areas of computer science including the OSI-RM, TCP/IP [3], software development life cycle, client/server architecture [6], and the model-view-controller (MVC) approach [9]. Thus, the modular approach will present mobile applications development in a manner that is familiar to computer science students. In addition, software design, coding, implementation, testing, and maintenance are easier, more flexible, and extendable with the modular approach. The modular design organizes course content into five modules. The first three modules, the presentation, data, and application logic, find root in both the three-tier client/server architecture [6] and the MVC approach. The application components separate data storage, business logic, and the visual representation of the data. This model for Android mobile application development is depicted in Figure 2. The application development components are grouped into five modules, as follows:

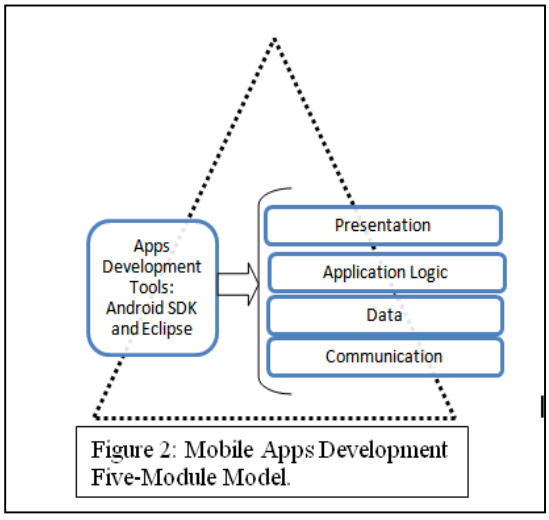

Figure 2: Mobile Apps Development Five-Module Model.

1. Apps development tools. Apps development tools consist of the Android SDK and Eclipse: The SDK and Eclipse are packaged as the Android Development Kit (ADT). ADT provides all the necessary tools for developing Android apps. The SDK consists of all tools needed for Android development, including a debugger, libraries, and emulator. Eclipse is a commonly used integrated development environment (IDE) plugin for ADT. The Android SDK includes a mobile device emulator that can be used to develop, test, and run apps on the computer rather than on a mobile device. The Android emulator imitates almost all the hardware and software features of a mobile device.
2. The presentation module. This module is concerned with designing user interface components, such as views, view groups (layouts), dialogs, notifications, and menus. The user interface design relies on storing interface parameters in XML documents. The user interface can be designed in Java, but it is commonly designed in XML by using either an XML editor or the Android's XML graphical editor. XML provides a simple mechanism for reusing various design elements, such as drawables, colors, and font styles, consistently in various views, screens, and apps.

3. The application logic module. Java Micro Edition (Java ME) is used for coding the Android apps. Native applications development requires the installation of the Java Development Kit (JDK) [7]. The Android system invokes the code from within an activity that is coded in Java. The Android's Java code can use many of the Java programming libraries. However, Android uses its own Java implementation for coding apps. Hence, the programming language used to program Android apps is often referred to as Java for Android. Java for Android stems from the Java programming language where the two are mostly similar; however, they have major differences, including:

- Android applications have a special life cycle [2]. Unlike regular Java programming where a program begins with a main () method, an Android app can start from an activity or other methods from several different stages in the life cycle, such as the onCreate () method.
- The Android runtime includes core Android libraries responsible for providing most of the functionalities available in the core Java programming language. Additionally, Android has its own specific libraries.
- The Android runtime uses the Dalvik Virtual Machine as an alternative to the Java Virtual Machine. The Dalvik Virtual Machine is optimized for use on mobile devices with limited resources such as power, CPU, and memory.
- Android does not support Java user interface libraries like java.awt (windowing and graphics) and java.swing (primary Java GUI widget toolkit). The widget package supplies most of the GUI elements in apps. Alternatively, the GUI elements can be coded in Java [2].

4. The data module. Generally, Android's data are stored in XML files and thus separated from code. Android's data include the values of different variables used in the Java code and values needed to instantiate widgets, thus increasing the readability and reusability of data when compared to hard-coded data. Also, the SQLite [10] database is used to store persistent application data. SQLite is a widely used open source general database layer written in C. SQLite requires very little memory and supports most basic SQL functions. The Android platform comes already packed with SQLite.

5. The communication module. This module consists of the set of APIs used for communication between the Android operating system and the various functions of other hardware and software components installed on mobile devices. The APIs provide the necessary classes through libraries supported at various API levels for Android. Newer API levels will support newer device functions that may not be supported by older devices. Separate APIs support different services like locations and maps and hardware components like the camera and media recorder.

The course design considers the rapidly evolving nature of mobile applications development and the Android platform. In addition, the course relates knowledge learned from apps development to other types of software development projects. Thus, the course is intended to teach core computer science topics that reinforce software development principles, while remaining relevant when developing apps for other platforms or software projects. Also, course topics are designed to accommodate continuous evolution in the mobile applications platform. To tap into prior knowledge to facilitate learning, the course design [1] adapts the modular approach, which is familiar to computer science students.

The course work includes six different homework assignments and a final project. The assignments are designed to gradually introduce students to various topics in mobile apps

development and aligned with the purpose of the corresponding modules of the introduced model. The first assignment introduces students to apps development tools and the Android development environment where students modify user interface and programming logic of existing apps. Subsequent assignments progressively introduce other modules from the mobile apps development architecture. The course covers all topics relevant to mobile apps development before the final project is assigned. For the final project, students develop an app for a simple test engine that requires understanding of all of the various modules and the corresponding topics covered in the course. The presentation of the final project is optional.

3. SUMMARY

Mobile applications development can be challenging. Even though the Android application development tools are widespread, the Android platform is undergoing rapid and continuous upgrades, which requires updating the developed apps and related course topics on apps development. Mobile applications development is an important topic to cover in computer science and it connects to many other topics, including the object-oriented paradigm, Java, programming, database management systems, and software engineering, that emphasize user interface design and human-computer interaction.

The course focuses on thoroughly covering the topics needed to develop Android apps and organizing these topics into five modules. Also, the development platform is adaptable, where upgrades to parts of the Android development platform will affect only the relevant modules in the introduced model. The design of this course attempts to cover mobile apps development by connecting course topics to topics from computer science. In addition, it attempts to apply relevant principles in organizing course objectives visually. While the Android development platform differs from other app development frameworks, many aspects of the presented approach can be applied to developing apps under other platforms and software projects.

4. REFERENCES

[1] Albert, R., Are you tapping into prior knowledge often enough in your classroom? July 19, 2011, http://www.edutopia.org/blog/prior-knowledge-tapping-into-often-classroom-rebecca-alber, retrieved October 24, 2013.
[2] Android Developers, http://developer.android.com/reference/android/package-summary.html, retrieved October 21, 2013.
[3] Braden, R., Editor, Internet Engineering Task Force, RFC 1122 - Requirements for internet hosts - communication layers, October 1989, http://www.faqs.org/rfcs/rfc1122.html, retrieved October 25, 2013.
[4] Gartner, Gartner says smartphone sales grew 46.5 percent in second quarter of 2013 and exceeded feature phone sales for first time, August 14, 2013, http://www.gartner.com/newsroom/id/2573415, retrieved October 19, 2013.
[5] Hara, B., Graphic display of student learning objectives, October 19, 2010, http://chronicle.com/blogs/profhacker/graphic-display-of-student-learning-objectives/27863, retrieved October 19, 2013.
[6] Microsoft Tech Notes, Chapter 7 - Client/server architecture, http://technet.microsoft.com/en-us/library/cc917543.aspx, retrieved October 19, 2013.
[7] Oracle, the Java tutorials, http://docs.oracle.com/javase/tutorial/, retrieved October 19, 2013.

[8] Reuters, Tablet demand drives rise in global smart device shipments: Gartner Oct 21, 2013, http://www.reuters.com/article/2013/10/21/net-us-gartner-2013devices-idUSBRE99K06Y20131021.
[9] Reenskaug, Trygve, the Model-view-controller (MVC) August 20, 2003, heim.ifi.uio.no/~trygver/2003/javazone-jaoo/MVC_pattern.pdfSahagian, J., retrieved October 19, 2013.
[10] SQLite.org, SQLite, http://sqlite.org/, retrieved October 21, 2013.
[11] University of Connecticut, Assessment primer: Goals, objectives and outcomes: How to write program objectives/outcomes, http://assessment.uconn.edu/docs/HowToWriteObjectivesOutcomes.pdf, retrieved October 24, 2013.
[12] United States Bureau of Labor Statistics, Occupational outlook handbook: Software developers, http://www.bls.gov/ooh/Computer-and-Information-Technology/Software-developers.htm#tab-6, retrieved October 24, 2013.