Experiences Teaching a Course in Android Game Development

Timothy E. Roden
Department of Computer Science
Lamar University
Beaumont, TX, USA
troden@lamar.edu

Rob LeGrand
Department of Computer Science
Angelo State University
San Angelo, TX, USA
rlegrand@angelo.edu

ABSTRACT
A one-semester undergraduate course in Android game development is described. The course has been taught six times at two different universities. We describe our experiences and lessons learned so far. Student feedback has shown the course to be popular and an important component in students' graduation portfolios, enabling many to get jobs immediately after graduation. Local industry has also responded well and has helped drive the outcomes of the course to some extent.

Categories and Subject Descriptors
K.3.2 [**Computing Milieux**]: Computers and Information Science Education [**Computer Science Education**]

General Terms
Design, Experimentation.

Keywords
Computer science education, game programming, game development, mobile application development.

1. INTRODUCTION
In 2008, the Computer Science (CS) Department at Angelo State University (ASU) began a new four-course elective sequence in computer game development. Three of the four courses emphasized PC game development while one course was devoted to the topic of handheld development. One of the most popular handheld gaming devices at that time was the Nintendo GameBoy. A wealth of online independently created tools enabled the instructor to successfully teach students how to program high-quality games for the GameBoy with minimum instruction [1]. By 2010 it was apparent that mobile gaming was moving increasingly away from dedicated consoles to general-purpose devices such as. In order to keep the content of the handheld game development course up to date, the instructor initially decided to transition the course to the iOS platform. Problems became immediately evident including the lack of on-campus labs with Macintosh computers, the restrictive licensing and distribution scheme of the platform and the fact that students and the instructor would be required to learn a new programming language: Objective C. It was decided to use Android as the replacement platform in the course. The reasons included: development could be done on PCs, of which both students and the university had in abundance, the programming language, Java, was already taught at the university, the development software was freely available and publishing and distributing applications had few restrictions. The primary instructor teaching the course at that time also believed, based on the information gathered at the GDC conference, that Android would eventually become the dominant operating system for mobile games, something not yet fully realized despite the fact Android now tops worldwide market share with 81% of mobile devices shipped in Q3 of 2013. The same report lists iOS devices at less than 13% market share [2].

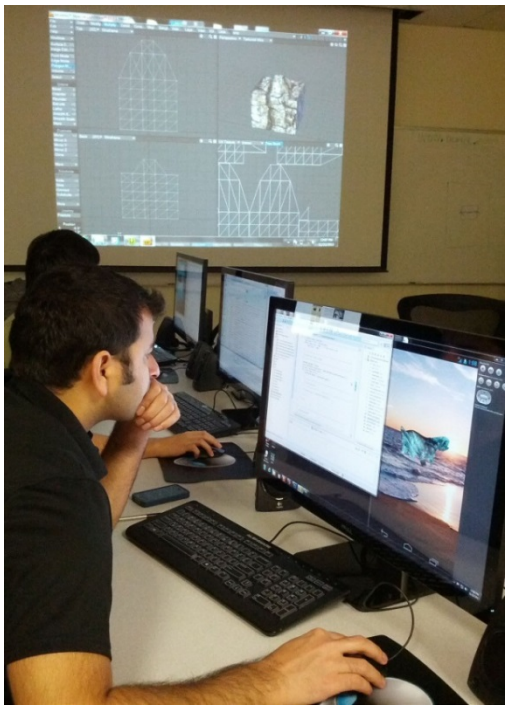

Figure 1. Students work simultaneously on both creating a 3D model and writing the code necessary to load the model into an Android application during class.

In spring 2011 the handheld game development course was taught for the first time using Android. Since then the course has been taught five other times, all using Android. In all instances, except one, the course has been taught in a dedicated game development lab, as shown in Figure 1.

2. RELATED WORK

Given the immense popularity of the Android platform and its usefulness as a gaming platform we expect many will adopt it as a teaching tool. Others have reported on similar forays into this new teaching method. Some of the observations are important to understanding the capabilities and limitations of the platform. For example, the constraints of the system have been noted, including the small screen size, small memory size, limitation of touch-only input, slower network data transmission rates and the possibility of frequent interruptions in network connectivity [5]. Students have not previously experienced most of these limitations in other programming courses. Still another important factor in game design is the fact a user's fingers may be covering a significant part of the small screen during gameplay which even further restricts the viewable screen space.

Another disadvantage reported, that we have also encountered, is the fragmentation issue [7]. The Android ecosystem is known as fragmented because of the many different devices with varying capabilities that run Android, as well as different versions of Android. This problem manifests when a student creates an app for the emulator, for example, and later tries to upload their app to a device and the app does not function correctly. We agree the best practice is to establish a baseline for the class each semester, including emulator settings and target versions of Android.

While it has been tried, we do not believe development, including code editing, on a physical device is a good idea due to the constrained environment. Instead, we recommend using a PC-hosted development environment. We agree that knowledge of object-oriented concepts is crucial for students in the course since the development language is Java and the Android SDK is heavily object-oriented [6].

A good list of topics for a course in mobile development has been defined in what was termed the "essence of mobile computing" [3]:

1) Mobility and pervasiveness
2) User interface and event-driven programming
3) Interruptions
4) Sensor-based input
5) Finite Resources
6) Low barrier to distribution of apps

While not specifically an outcome of our courses, we do try to emphasize to students that since mobile users will often have their device with them at all times, this opens up new opportunities for gameplay. As previously mentioned, designing the user interface in a mobile game can be quite challenging. We do stress that mobile games need to be interrupt-aware – a phone call or other event can interrupt an app at which point the app needs to gracefully pause and intelligently resume after the interruption is over. This point is not so much unlike traditional PC programming with the exception that in the resource-constrained mobile environment interrupting processes can often take resources away from the paused app which need to be reacquired upon resumption. While we have not utilized input methods in our classes besides touch input we intend to explore other methods in the future. Finally, since apps are easy to distribute we have typically made this a component of the course – showing students how to set up their development account and upload apps.

3. THE COURSE

The course has been taught six times between spring 2011 and fall 2013. In 2011 both instructors were at the same university. In 2012 an instructor moved to another university. So, the course has now been taught at two different universities. At both universities the course is 14 weeks long and is an elective that can be taken after students complete the first three required programming courses. Neither instructor has worked with a teaching assistant – all course grading was done by the instructors. The different versions of the course can be broken down into three distinct attempts: the first time the course was taught, later versions of the course at a university where Java is the primary programming language and another attempt of the course at a university where C++ is the primary language.

3.1 First Attempt

The handheld game development course was taught using Android for the first time in spring 2011. The first order of business was to teach students the operating system and device capabilities. Because there were many different Android devices on the market and since only two of twenty students in the class had an Android device the Android SDK emulator was chosen on the target platform. The emulator is a PC application that can run Android programs in a window on the PC desktop. A significant problem with the emulator is it is typically very slow to load and execute programs in addition to being functionally limited. For example, testing sensor features like the accelerometer and using multi-touch gestures is not possible.

The textbook chosen was a trade book that adequately described many of the important features of the operating system. For its chosen purpose in the class, which was to provide a broad overview of the system's capabilities, it was a good choice. On the other hand, topics in graphics programming were very limited in the book. At that time the instructor could not find an Android programming book more appropriate to game development. Since then, other books have been published that serve the needs of the course better including the current textbook in the course [8].

The preferred programming language for Android is Java. So, Java is used in the course. C++ can be used but is much more difficult due to significantly fewer resources available to students in terms of books, Internet resources and the lack of an integrated development environment specifically for C++ development on Android. Compounding the difficulty for students was the fact that the primary programming language at the university is C++. The instructor spent the first two weeks of class teaching Java using a set of slides that primarily compared the differences between the two languages and also highlighted the unique features of Java. Two programming exercises were given and students were required to use the freely available Eclipse integrated development environment (IDE) which, at this point in time, is still a requirement for Android development. A dedicated Android-specific IDE is under development by Google and we intend to use it once it has been fully released. While the instructor felt the time spent on Java was adequate, since students in the class were juniors and very familiar with programming, feedback from students at the end of the semester indicated they did not feel like two weeks was sufficient.

Following the chosen textbook, the Android operating system was thoroughly dissected and explained in detail including topics such as development tools, application life cycle, activities, user interface, intents, broadcast receivers, networking, file system, app preferences, database APIs, background services, telephony, audio, and accelerometer. For most of these major topics students were given an assignment. This set of topics occupied the majority of the semester.

Graphics programming was the final topic in the course. 2D graphics was covered using the Android Canvas class which contains a set of methods for 2D graphics rendering. In addition, OpenGL ES 1.x was covered to enable 3D rendering. Both topics included student assignments. In the case of 3D graphics, the assignment was the capstone of the course. The assignment had students create a slot machine app with rotating tumblers that were modeled as 3D objects with the object definitions written directly in code – no 3D modelling software was used.

Student feedback from this iteration of the course was generally negative. Students did not feel the topic of game development was adequately covered and many were also frustrated by the lack of Java experience and corresponding difficulties simply compiling simple programs. In retrospect, the many details of the operating system that were covered were not necessary in order to create simple games. There was a very positive impact from this first iteration of the course, however. Two local businesses who became aware the university was teaching Android development, subsequently engaged in sponsored research projects with the instructor and several of the students who had taken the course. Two of the students later went to work for one of the companies as general-purpose Android developers – not necessarily graphics-related apps only.

3.2 Teaching the Course at a Java School

The course has been taught three times at a university where Java is the primary programming language. The instructor's goal was to focus more exclusively on the topic of game development using both 2D and 3D graphics. With that in mind the instructor narrowed the topics related to the Android operating system to exclude many of the topics covered in the first attempt of the course. A new textbook was chosen specifically targeting Android game development with good results [8]. The textbook describes a framework for both 2D and 3D game development that hides and encapsulates many Android specific details allowing students to focus more easily on game design and implementation. Important topics of the operating system were still covered such as application life cycle, interruptions, intents, the event-driven programming model and the unique user interface model of touch only. Accelerometer input was also covered but only in passing since the target platform, up to now, has still been the Android emulator.

Since students already knew Java, no time was spent on that topic. Instead the course was divided roughly in half. The first half covered 2D game development while the second half covered 3D development. This corresponds with the textbook of which half is 2D and the remainder 3D. Most chapters include an author-created application project which can be downloaded from the Internet. The instructor used many of these sample projects as the basis for homework assignments in which students modify the projects to add additional or different functionality. Also, the instructor tried to insure that students read the textbook by assigning written homework for each chapter in the textbook consisting of 20-45 questions per chapter. Most of the answers to the written homework can be found in the textbook while only a few can be found on the Internet.

Students in the course are required to create all their own artwork. The textbook provides a good example since the author uses a doodle style – hand drawn artwork - the appearance of which might be associated with a child. For students who desired a higher level of art in the programs, the instructor gave in class presentations on the usage of a 2D paint and image editing program. For audio, the instructor also demonstrated the use of a freely available, but very useful audio editor.

The capstone of the first half of the course is a project students were required to complete individually. The assignment was to create a "shooting gallery" type game. In addition to a start screen, help screen and optional high scores screen, the main game screen was required to have ammunition, targets and a score visible in addition to an appropriate background image. Optionally, students could add explosions or other animations when targets were hit. User interface consisted of touching the target on the screen which both expended one unit of ammo and processed collision detection with a moving target to determine if the target was hit, in which case the target is then removed, score is adjusted and ammo decreased accordingly. Two examples of student work are shown in Figure 2.

During the second half of the course, students are introduced to OpenGL 1.x, a topic which is adequately, but not exhaustively covered in the textbook. The instructor also demonstrates the basics of 3D modelling so that students can create their own models for the projects. The capstone of the second half of the course is a project in which students can work in teams of two if they prefer. Unlike the narrowly defined scope of the mid-semester project, the final project is open ended, although students are required to submit and receive approval from the instructor for their game design. In almost all cases, students were overly ambitious in their goals requiring the instructor to help students scale back their idea into something that can be accomplished. A final presentation day is scheduled at the end of the semester. An example student presentation of a 3D app is shown in Figure 3.

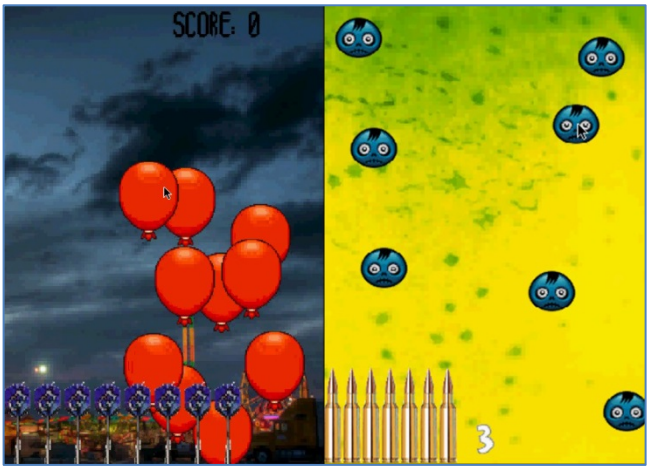

Figure 2. Two examples of 2D "shooting gallery" games created by students in the course.

Figure 3. A student presents a 3D game during class.

Student feedback from these iterations of the course has been much more positive. On feedback forms, it's not unusual for students to say it was the best course they had ever taken at the university and almost all wanted to learn more about Android by taking additional courses.

3.3 Teaching the Course at a C++ School

The Android course has since been taught again at the C++ school. This time, the instructor decided to spend almost a month introducing Java, emphasizing those aspects least like C++ (*e.g.*, arrays as objects) and most important to Android development (*e.g.*, memory management). Instead of lectures with slides, students were given assignments with webpage instructions (and sometimes some starter Java code) with class time to work on them in the Eclipse IDE. Student feedback on the course was very positive and indicated that the time spent on Java was worthwhile. The disadvantage was that less time was left for more advanced topics, especially 3D graphics.

4. RESULTS

Android has proven to be an efficient and useful tool in teaching the handheld game development course. Over six iterations of the course, the instructors have learned to keep the course content narrowly focused on the topic of graphics and game development. For most iterations of the course, the instructors have also tried to show students how to publish their games and add pop-up advertisements as a monetization method. So far we haven't had a student start their own company, but we have had multiple students gain employment upon graduation in Android development positions. We received unexpected attention from local industry which resulted in several sponsored projects which, in turn, influenced the course in small measure. For example one instructor feels it necessary to spend several lectures on the fragmentation issue since commercial applications will face this problem and it can be a difficult one. Basically, the instructor teaches students how to render their graphics to scale to any device – a topic not covered very well in the currently used textbook. The fragmentation issue is one of the biggest disadvantages of Android – the resulting code complexity can be both annoying and frustrating for students to deal with.

5. CONCLUSION

The experiences in the course have led the instructors to believe that because of the complexity of the underlying operating system and the continuing rise of Android as the world's most popular mobile operating system, the course is not sufficient to prepare students for careers that may involve Android development. One recommendation is to create an Android-only course to cover the operating system and desktop-type applications. Such a course could become a prerequisite for the handheld game development course. We would expect much higher quality work and even higher student satisfaction in the course, as well as preparing students better for employment.

The difficulties posed by using the Android emulator as the target platform need not be a problem any longer. As reported elsewhere, many students now have their own Android device, which is a benefit in such a course as ours [4]. A reasonably equipped Wi-Fi-only device can be purchased for under $200, which is equivalent to the cost of a high priced textbook. For future versions of the course, we may require students to have a physical device.

Regardless of which mobile platform is chosen to teach a course such as this there are tradeoffs. We chose Android because it offered more advantages than disadvantages as compared to another platform. Another university may well find that another platform is better suited to their curriculum.

6. REFERENCES

[1] Boudreaux, H., Etheredge, J., and Roden. T. Adding Handheld Game Programming to a Computer Science Curriculum. In *Proceedings of the 3rd International Conference on Game Development in Computer Science Education (GDCSE '08)* (Miami, Florida, February 28 – March 3, 2008), ACM Press, 2008, 16-20.

[2] Bradley, T. Android Dominates Market Share, But Apple Makes All The Money. http://www.forbes.com, retrieved: December 9, 2013.

[3] Burd, B., et. al. Educating for Mobile Computing: Addressing the New Challenges. In *Proceedings of the final reports on Innovation and Technology in Computer Science Education (ITiCSE-WGR '12)* (Haifa, Israel, July 3-5, 2012) ACM Press, 2012, 51-63.

[4] Fenwick, B., Kurtz, B., and Hollingsworth. J. Teaching Mobile Computing and Developing Software to Support Computer Science Education. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education (SIGCSE '11)* (Dallas, Texas, March 9–12, 2011), ACM Press, 2011, 589-594.

[5] Skelton, G., Jackson, J., and Dancer, F. Teaching Software Engineering Through the Use of Mobile Application Development. In *Journal of Computing Sciences in Colleges*, 28,5 (May 2013), 39-44.

[6] Tillmann, N., et. al. The Future of Teaching Programming is on Mobile Devices. In *Proceedings of the 17th annual ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '12)* (Haifa, Israel, July 3-5, 2012) ACM Press, 2012, 156-161.

[7] Werner, M. Teaching Graphics Programming on Mobile Devices. In *Journal of Computing Sciences in Colleges*, 28,6 (June 2013), 125-131.

[8] Zechner, M., and Green. R. *Beginning Android Games*. Second Edition. Apress. 2012.