ON THE IMPACT OF ANDROID API EVOLUTION ON EDUCATION MATERIALS

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

by

Kennedy Owen

June 2017

COMMITTEE MEMBERSHIP

TITLE:	On the Impact of Android API Evolution on Education Materials
AUTHOR:	Kennedy Owen
DATE SUBMITTED:	June 2017
COMMITTEE CHAIR:	David Janzen, Ph.D. Professor of Computer Science
COMMITTEE MEMBER:	Alexander Dekhtyar, Ph.D. Professor of Computer Science
COMMITTEE MEMBER:	Clark Turner, Ph.D. Professor of Computer Science

ABSTRACT

On the Impact of Android API Evolution on Education Materials

Kennedy Owen

The recent growing popularity of mobile devices has led to the establishment of several popular mobile platforms such as the Android operating system. To foster growth in this evolving market, Application Programming Interfaces (APIs) were created to enable developers to create mobile device applications that utilize mobile device features and functionality for personal or widespread commercial use. However, as a result of excessive device and API evolution, mobile development faces increasingly complex issues, including an alarmingly rapid decay of development resources.

This thesis conducts a case study around one such resource: a series of Android app development lab exercises used in an Android app development course taught at California Polytechnic State University, San Luis Obispo. First, these labs and their respective written guides were modernized and fitted for backwards-compatibility to better reflect newer Android devices and development tools at the time. The updated labs were subsequently used in the next course offering, with student lab feedback recorded for evaluation. Several years later, the apps from these new labs were further tested for abnormal behavior on a variety of Android devices. Results from analyzing all feedback and observations show that despite all measures taken to future-proof these labs, Android device and API evolution continues to vastly outpace third-party Android resources without frequent modernization and upgrades.

Keywords: Android, Cal Poly SLO, API, labs, Java, education

ACKNOWLEDGMENTS

I'd like to thank:

- Dr. Alexander Dekhtyar and Dr. Clark Turner, for demonstrating the true commitment of the Cal Poly Computer Science department by not only initially agreeing to serve on my thesis committee, but also upholding that agreement years later. I appreciate your dedication to and insight into my project.

- Dr. David Janzen, my project adviser. I truly appreciate your continued support and guidance for myself and my project. I can't thank you enough for your patience, encouraging words and feedback over the years that helped me finish this thesis. It was truly an honor to work with you on this project.

- Tiffany Yee, my girlfriend. For some mathematical assistance on this project, and for her love, trust and respect for me, which helped me continue and finish this project at my own pace. I love you dearly.

- My parents Bill and Caroline, and my brother Corey. For nearly three decades my family has giving me the environment and space to grow as an individual, provided immense guidance for me and supported my decisions throughout life. Much love and respect for you all.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

The popularity of mobile devices over the past several decades has spurred explosive growth and expansion of mobile device features and functionality. Companies that develop mobile devices have included more features such as built-in front- and rear-facing cameras, GPS trackers, light sensors, gyroscopes, expandable device storage options and fingerprint readers. Additionally, as a result of the advent and modernization of widespread Internet and personal mobile device usage within daily lives, modernized technologies such as cloud computing and digital content streaming have seen rapid growth and expansion. With these new technological developments over the last few decades, software developers, hobbyists and device software companies (henceforth collectively referred to as **mobile app developers**) began creating mobile software applications, or **mobile apps**, that run on these devices. These mobile apps directly utilize both hardware and software features and serve a variety of purposes, including (but not limited to) computer application replacements such as notepad editors or media players, simple single-purpose tools such as flashlights or document viewers, and complex programs that serve as mediums for GPS navigation, image editing, gaming or digital content consumption.

To lay a foundation to support such mobile app utility, mobile device platform creators have partnered with mobile device hardware and software developers to create Application Programming Interfaces (**API**s). APIs, such as the Android API[1], are a large set of instructions for a development system that allow mobile app developers access to and some degree of control over a mobile device's operating system, hardware

[1] **Introduction to Android** API guide page:
`https://developer.android.com/guide/index.html`

and software. APIs are usually established together with Software Development Kits (**SDK**s), which are a series of development tools that assist in developer navigation and utilization of the development system. APIs themselves expose low-level mobile device features (and various other outputs at the device's architectural level) in a simpler, higher-level layer to current and aspiring mobile app developers. Developers in turn create mobile apps using the API, utilizing various SDK tools as necessary along the way, by compiling code written in an Integrated Development Environment (**IDE**). This compiled code results in the creation of a mobile app package (for example, an .apk file in Android, colloquially referred to as an **APK**) that can be installed on a device by the creator for personal use, or published through an app store (such as the Google Play Store for Android) for public installation and use by any number of mobile device users. Due to the wide variety of mobile device types and versions, mobile APIs in particular effectively lower the barrier of learning mobile application development through the simplification of mobile device access and control. Most mobile device APIs are standardized, as well as regularly updated and maintained, by the companies that publish and license them for commercial and non-commercial use.

1.1 Mobile App Development Challenges

There are several challenges commonly encountered in mobile application development. One issue is continuous maintenance of mobile application code when API changes occur. Aside from mentions of new platform versions in newly released devices and preemptive published warnings of API changes by Google, there is currently no effective means of predicting when current or new API changes take place. This results in previously written mobile app code breaking or using old or unsafe procedures, requiring code in applications to be checked against new API releases on a

frequent basis. The frequency of API changes to account for the most recent mobile operating systems or API versions results in fewer and shorter-lasting development standards.

Another issue is the growth and expansion of mobile devices, which results in a mandatory cycle of phasing in and out old and new device features respectively. The issue is further complicated by widespread customization of mobile platforms by different mobile device manufacturers, which often results in inconsistent mobile app behavior across a variety of mobile devices. This situation has become increasingly common across all of mobile development, and becomes especially concerning in situations where apps exhibit different behavior in devices from the same device family (e.g. Samsung Galaxy S6 and Samsung Galaxy S7) or two identical devices running different software versions (e.g. two iPad Air devices, one with iOS version 10.0.1 and the other with iOS version 10.0.2).

These issues continue to plague mobile application developers in recent years, and negatively impact mobile application development resources, standards and curricula worldwide in a variety of contexts.

1.2 Project Overview

Education is one such impacted mobile development resource. To study the impact of the previously-discussed problems on mobile development in education, this project revolves around a case study involving third-party mobile app development resources for the Android platform. A series of exercises and resources (henceforth collectively referred to as the **old labs**) written by a former California Polytechnic State Uni-

versity, San Luis Obispo graduate student [18] were created for a Cal Poly Master's Degree thesis project; these labs were used in an upper-level Cal Poly Android app development course. For years, these labs had been actively used and referenced by developers and curricula worldwide. However, more modern analysis on these labs deemed them to be far outdated after being used for several years without major modifications.

This project focuses on these original labs as the crux of the case study. First, the outdated labs were updated and given specific support guidelines to isolate the study within a single snapshot in time. The updated exercises were then used for the same college-level mobile development course curriculum they were used for prior to updating, and the new exercises were evaluated in the form of post-completion student surveys. The new exercises were also tested on a variety of Android devices and device types to observe odd or improper app behavior on relatively newer Android devices. Finally, the results were analyzed to determine the overall effectiveness of the changes made, and to determine how the updated exercises handled the observed issues with the evolving Android devices and API they were written against. The project concludes with a discussion of improving and restructuring the course curriculum and lab exercises, with insight drawn from project observations and survey results.

Chapter 2

BACKGROUND

In order to properly construct the new labs for the case study, first a survey of the current state of Android development was conducted; this included an analysis of the current Android API and modern device usage. Afterwards, the original Android development course labs were examined to determine if lab upgrades or brand new exercises were appropriate for the current time frame. Before the new labs are explored in detail, however, a summary of Android device and OS version history, the Android API and the old labs will be discussed to provide background on the subject. This summary is not fully comprehensive on a general level, but is highly relevant to this paper's primary subject matter.

2.1 Android OS & Devices

Android, alongside iOS (created and maintained by Apple Inc.), is one of the all-time most popular and widely-used mobile operating systems (**OS**s) or **mobile platforms** worldwide with over 1 billion active users [14]. At the time of writing this paper, the most popular Android device family is the Samsung Galaxy S phone series [24], the latest release being the Galaxy S8 device collection. The Android operating system and accompanying development API have both evolved alongside new device production and functionality, and thus required investigation and analysis before creating new labs.

2.1.1 Android Pre-Tablet Era

Google's Android device legacy officially began in late 2008 with the release of the T-Mobile G1 [26]. The G1 was the first device to run the Android operating system, and featured core foundational device features such as a touch screen, a physical button navigation panel below the screen including a trackball, a 3-megapixel rear-facing camera and a full QWERTY slide-out keyboard. The G1 also came with a plethora of app support for apps such as Google Maps for GPS navigation [11], Amazon MP3 (now called *Amazon Music*) for digital music downloads [1], and the Google Android Market service (now called *Google Play Store*) for downloading, updating and managing apps on users' Android devices [9]. The first few iterations of the Android OS, specifically 1.0 (`BASE`), 1.1 (`BASE_1_1`), 1.5 (`CUPCAKE`) and 1.6 (`DONUT`) were established with the G1 as the primary device, adding a virtual keyboard and other software improvements to the G1 with each update. Notably, the G1 was the first device used in the Android app development course at Cal Poly, SLO as many of these devices were donated for the course by Google. The G1 was used predominantly in the first and second course offerings (discussed later in this chapter) by students who did not own or have access to their own personal Android device.

Over the next few years, more phone carriers and manufacturers began working together to produce new Android phones that evolved beyond the capabilities of the G1. Many of these commercially successful phones would eventually serve as the foundation of successful phone families for years to come. One such phone was the Motorola Droid released in late 2009, which was made available through Verizon Wireless [28]. The Droid was advertised as the first smart phone to run Android OS version 2.0 (`ECLAIR`), something the G1 did not support, and improved on prior features of Android phones such as the phone camera. The Droid also included Wi-Fi support,

external memory card support of up to 16 GB, a smart dictionary for search and typing that integrated the phone's Contacts list, and several new or updated apps such as Google Maps Navigation, Google Talk (now called *Hangouts*) for chat support [10] and an official mobile YouTube app for video consumption and uploading [12]. Verizon was also one of the first phone service provider companies to include several of their own pre-loaded applications on their supported phones, such as Verizon Wireless Visual Voice Mail on the Droid. Following the Droid were several more iterations by Motorola such as the Droid 2 and Droid 3, leading to the original Motorola Droid to be referred to as the **Droid 1**. In addition to `ECLAIR`, the major Android OS versions supported by the Droid 1 were 2.0.1 (`ECLAIR_0_1`), 2.1.x (`ECLAIR_MR1`) and 2.2.x (`FROYO`).

In 2010, the smartphone industry was growing and flourishing at a much greater rate than when the G1 first came out. In early 2010, Google released the Nexus S mobile phone, co-developed by Google and Samsung [5]. The Nexus S was advertised as the first smart phone to ship with Android OS version 2.3 (`GINGERBREAD`) and as a device that could read Near Field Communication (NFC) tags. In addition to various hardware and software upgrades and improvements, several new features supported by the Nexus S included gyroscope sensor support, the combination of dual front- and rear-facing cameras, and Internet-powered calling (as opposed to simply using telephone company data lines). Note that the Nexus S is a successor to the Nexus One phone, which was loaned to students for the course encompassing the project detailed later in this paper. Midway through the same year, Samsung released its own smart phone, the Galaxy S [21], which initially ran `ECLAIR_MR1` but quickly updated its user base to run Gingerbread. Samsung boasted that the Galaxy S had many new and improved features and software upgrades such as HD video playback and recording support, Augmented Reality (AR) in camera mode, a digital compass,

external micro USB 2.0 support, and its own patented pre-loaded Samsung apps such as Swype for ease of virtual keyboard word completion using the swiping patterns of a user's fingers. As mentioned above, the Galaxy S series would become one of the most prominent and popular Android phone series worldwide, as mentioned earlier.

2.1.2 Android Tablet Birth Era

While the smart phone market was erupting, a new market was emerging for a different subset of mobile devices: **tablets**. Tablets were larger smart devices with significantly bigger screens than most phones at the time (with an average of 9- to 10-inch screen size, compared to the current average smart phone's 4-inch screen size). They were designed with a larger profile and more powerful hardware than smartphones, to give users a much higher degree of mobile content interactivity and consumption in a device approximately between a smart phone and personal computer (more the former than the latter).

While supporting its currently widely successful Gingerbread Android OS for smart phones, Google made preparations to roll out a new Android tablet-exclusive OS, 3.0.x (`HONEYCOMB`). Honeycomb was not officially supported for smart phones, only tablets. Google worked with device manufacturers (Motorola and Asus to name a few) to create and release Android tablets in the hopes of creating a successful tablet experience against their largest competitor, Apple, which had announced its upcoming tablet device of choice, the iPad, in early 2010 [2]. Around the same time, Verizon Wireless announced the Honeycomb-driven Motorola Xoom [29]. The Xoom featured a 10.1-inch screen with a 1280x800 pixel display in 720p HD resolution, HDMI connectivity and front- and rear-facing cameras, as well as the same gyroscope, barometer, digital

compass and accelerometer support that recent smart phones supported. Identical or greater feature support, combined with Honeycomb's new tablet-specific interface, provided for a brand new Android user experience.

Over a year later, Asus announced their own tablet running Android Honeycomb, the Eee Pad Transformer in early 2011 [4]. Aside from expected hardware and software improvements, the Transformer was advertised with a full QWERTY keyboard hinged dock that allowed the tablet to double as a smaller laptop with extended battery life. Later versions of the Transformer (and other tablets like it made by other companies) would support newer versions of Honeycomb, Android 3.1.x (`HONEYCOMB_MR1`) and 3.2 (`HONEYCOMB_MR2`), which allowed for more input device support such as keyboards, game controllers, mice and digital cameras, as well as additional UI improvements. The Asus Transformer was immensely popular upon release and widely considered ahead of its time. However, non-tablet owners would never see these peripherals in the smart phone market, and no additional tablet-specific Android platform versions would be made by Google as of the time of writing this paper. It is worth noting that the Transformer's tablet-and-dock device model would resurface years later as a popular "mobile computing device" market option, succeeded by such devices as the Microsoft Surface tablet.

2.1.3 Post-Tablet Birth Era

With no new Android smart phone version updates since Gingerbread, and the introduction of a separate Android tablet version in Honeycomb, Google decided to combine its next Android OS release for phones and tablets into a unified solution that supported both types of devices. The next major Android version release, Android

4.0-4.0.2 (`ICE_CREAM_SANDWICH`, occasionally shortened to **ICS**) and its successor upgrade, Android 4.0.3-4.0.4 (`ICE_CREAM_SANDWICH_MR1`), added more user navigation control in the form of the standardized System Bar and Action Bar. The System Bar added virtual menu buttons as a complete alternative to physical device menu buttons. The Action Bar allowed for more app-specific behaviors embedded inside of notifications in a slide-down bar. ICS also introduced a spell checker for text entry, a voice input engine, facial recognition for stronger security, stronger cloud computing support and improved photo editing. These additions and improvements (and more) were now supported across both Android smart phones and tablets. Furthermore, since smart phones and tablets running ICS now supported a unified layout for users and developers, it was now easier for apps to be designed for, created and maintained across different Android device types. The first Android device that supported ICS was the Galaxy Nexus [22], published by Google and Samsung and made available in 2011. ICS would lay the foundation for multi-device app support and user experience for the next few years in Android history.

Following the significant success of Ice Cream Sandwich, Google's next Android OS version was 4.1-4.1.1 (`JELLY_BEAN`). Jelly Bean upgraded Android's graphics pipeline and rendering/drawing implementations to provide a faster visual experience, allowing for smoother open task navigation and additional accessibility and language support. It also expanded the Notification bar to allow developers to include additional data within notifications currently in the bar, and offered smarter app update methods. The first upgraded version of Jelly Bean was Android 4.2-4.2.2 (`JELLY_BEAN_MR1`), which allowed multiple users to share a single tablet using multiple user accounts, and supported device connection to external displays. The last upgraded version of Jelly Bean was Android 4.3 (`JELLY_BEAN_MR2`), which upgraded inherent Bluetooth connectivity support and streaming protocol support. The first Android devices to

support Jelly Bean were the Nexus 4 phone published by Google and LG, the Nexus 7 tablet published by Google and Asus, and the Nexus 10 tablet published by Google and Samsung [6]. All three devices were made available in late 2012, and the Nexus 7 had an additional model offering data capability alongside the default Wi-Fi capability. Targeting a phone (Nexus 4), portable tablet (Nexus 7) and large tablet (Nexus 10), Google aimed to ensnare and capture an even wider user market than before.

At the time of decision for updating the original lab exercises (see Section 2.6 for more detail), it was determined that Android OS versions 2.3.3-2.3.4 (`GINGERBREAD_MR1`) up to 4.2-4.2.2 (`JELLY_BEAN_MR1`) were the best versions to target due to current Android device usage statistics. The official Android Dashboards page[1] contains current Android device usage statistics. See Section 2.1.4 below for further detail on all Android versions released up to the time of writing this paper.

2.1.4 Android OS Version Details

For brevity and the limited scope of this project, details of important Android OS versions and device features have been summarized above as deemed appropriate for scope. A full list of Android OS/platform versions, names and release details is provided as follows in ***Figure 2.1*** for convenience, and is current as of the time of writing this paper (May 2017):

[1] Android Dashboards page:
`https://developer.android.com/about/dashboards/index.html`

Platform Version	API Level	VERSION_CODE	Notes
Android 7.1.1 Android 7.1	25	N_MR1	Platform Highlights
Android 7.0	24	N	Platform Highlights
Android 6.0	23	M	Platform Highlights
Android 5.1	22	LOLLIPOP_MR1	Platform Highlights
Android 5.0	21	LOLLIPOP	
Android 4.4W	20	KITKAT_WATCH	KitKat for Wearables Only
Android 4.4	19	KITKAT	Platform Highlights
Android 4.3	18	JELLY_BEAN_MR2	Platform Highlights
Android 4.2, 4.2.2	17	JELLY_BEAN_MR1	Platform Highlights
Android 4.1, 4.1.1	16	JELLY_BEAN	Platform Highlights
Android 4.0.3, 4.0.4	15	ICE_CREAM_SANDWICH_MR1	Platform Highlights
Android 4.0, 4.0.1, 4.0.2	14	ICE_CREAM_SANDWICH	
Android 3.2	13	HONEYCOMB_MR2	
Android 3.1.x	12	HONEYCOMB_MR1	Platform Highlights
Android 3.0.x	11	HONEYCOMB	Platform Highlights
Android 2.3.4 Android 2.3.3	10	GINGERBREAD_MR1	Platform Highlights
Android 2.3.2 Android 2.3.1 Android 2.3	9	GINGERBREAD	
Android 2.2.x	8	FROYO	Platform Highlights
Android 2.1.x	7	ECLAIR_MR1	Platform Highlights
Android 2.0.1	6	ECLAIR_0_1	
Android 2.0	5	ECLAIR	
Android 1.6	4	DONUT	Platform Highlights
Android 1.5	3	CUPCAKE	Platform Highlights
Android 1.1	2	BASE_1_1	
Android 1.0	1	BASE	

Figure 2.1: Android Version Info Chart

A version of the above table is live and updated by Google in the **API Levels** section on the **uses-sdk** documentation page[2]. [7].

2.2 Android API

The Android **Application Programmer Interface**, or API, is a powerful resource and instruction set that Android app developers use to construct official Android apps using programming code. As more Android devices are devised and released to the public, the Android API has evolved to allow developers to utilize new features in these devices (or utilize older features in newer, better ways).

The Android API is used in conjunction with a series of (mostly software) tools for use in Android app development that is collectively referred to as the Android Software Development Kit or Android **SDK**. One example of an SDK tool is the Android Debug Bridge, or **ADB**, which allows developer to connect a physical or virtual device to an Android development computer for device communication. Since the SDK is only primarily involved in Android development setup for labs covered in this project, it will not be covered in great detail in this paper.

2.3 API Levels

As stated previously, for any given Android OS version there is a specific version, or **level**, of the Android API that has been optimized by Google to most accurately and appropriately create apps for that OS/platform version. These versions of the

[2] **uses-sdk** documentation page:
`https://developer.android.com/guide/topics/manifest/uses-sdk-element.html`

Android API are individually referred to as **API levels**. The first version of the Android platform, 1.0 (`BASE`), which ran on the G1 phone, is matched by the very first Android developer API, API level 1. As another example, the Nexus S smart phone mentioned earlier runs Android platform version 2.3 (`GINGERBREAD`), which is matched by API level 9. The updated labs created for this project target Android platform versions 2.3.3-2.3.4 (`GINGERBREAD_MR1`) up to 4.2-4.2.2 (`JELLY_BEAN_MR1`), which corresponds to API levels 10 through 17. At the time of writing this paper, the Android API has a total of 25 levels with the latest version being Android 7.1.x (`N_MR1`), although this paper focuses primarily on API levels 10-17. For more detail on which API levels correspond to which Android platform versions, please refer to ***Figure 2.1*** shown in Section 2.1.4 above.

Each level of the Android API allows developers to create their own apps that can run on any Android device with a platform version that particular API level targets. Using the Android API, app developers not only enjoy a high degree of ease and freedom in designing the layout and behavior of apps, but they can utilize the device's hardware, software, and even other apps on the device while the app is running with relative ease. For instance, the Android app *Hangouts* (a messaging app that lets users interact with their friends) allows users to not only send data (MMS) messages and text (SMS) messages, but also easily access the device's camera through the phone's *Camera* app. This allows users of the Hangouts app to take pictures and videos to share with others all in one smooth experience, instead of having to do each step manually in separate apps. Thanks to the API, this smooth accessibility of components is preserved across most of the Android device feature spectrum for app developers.

NOTE: Formally published Android apps meant for widespread public use are typically published on and downloaded through the Google Play Store. This includes not only apps developed by Google or other commercial firms such as *Hangouts*, *Camera* or *Skype*, but also apps developed by users not affiliated with Google or other commercial firms (such as hobbyist developers) who elect to publish their app to the public.

To explain further, the Android API is a large set of instructions that communicates between an Android phone's physical and digital contents (hardware, software and some system apps such as *Camera*) and the user who is creating apps. An Android application developer can call upon any of these instructions inside the app they're developing whenever they have need for them. For example, to access the *Camera* app that is built into every Android phone, the user can utilize the **Camera** class inside the API. This aspect of the API and more will be discussed in greater detail in the next section.

These instructions also provide simplified and common methods for developers to create and modify certain aspects of an app, such as screen layout, power management or control over app orientation switching. The instructions also include interacting with certain device system settings such as power management. However, the instructions included in the Android API are limited so as to not allow developers full access to certain system settings, commands or any purposefully isolated apps on Android devices within their own apps; for example, a user-created app cannot command the device to power off, nor can it access or modify any other user-created app that the user has downloaded from the Google Play Store (only specific apps such as *Camera* that are meant for widespread access can be accessed from other apps). Only special commercial or system apps have the authority to power a device off. Apps published by hobbyists or non-commercial developers, as well as most apps in general, also can-

not access other apps published under the same conditions unless specifically allowed by an app's publisher.

2.4 API Content Introduction

This section describes the contents of the Android API in greater detail. Minor programming and software development knowledge is assumed for this section, although most terminology and topics will be described as generally as possible while preserving brevity.

Since the Android API changes often, the latest API level changes are reflected in the Android API documentation but almost all API levels are still supported to an extent. The API is changed and given a new version number when new Android platform versions are released. Throughout time, some aspects of the API have been modified, removed or **deprecated** (discussed several paragraphs below), so app developers targeting older API levels may use older methods or implementations in the API provided they are not targeting newer Android devices and OS versions. Likewise, app developers targeting newer API levels may have access to new methods or implementations in the API, but may or may not have the option to use older methods or implementations that are used in previous API levels; this is because Google may have decided that those older ways should not be supported on newer Android versions because they are less efficient, less secure or generally less compatible with newer devices for other reasons.

In other words...

- As Android devices and the Android OS/platform evolve over time, new ways of accessing or doing things in apps may be added to the Android API because they're more efficient, compatible and/or safer.

- Sometimes developers are limited to using old ways of accessing or implementing features or data in an Android app when targeting just older Android devices and platforms; this is because it is usually unreasonable to spend time making something very new work on an older device that was not designed with the new way in mind, and most consumers use newer devices rather than older ones.

2.5 API Contents

The Android API is largely based on the Java programming language. To describe the Android API contents, the Camera API class[3] will be primarily used as an example when qualified since one of the new labs created for this project utilizes this class, and cameras are very well-known and widely used hardware. Other classes may be brought up and are linked appropriately.

Please note that at the time of writing this paper, the Camera class has been deprecated in API level 21 in favor of the Camera2 class[4]. **Deprecation** indicates that something (usually a certain part or parts of a class, but sometimes an entire class or even an entire package) has been phased out in newer API levels, often replaced with something more modern or smartly-implemented, but it can still be utilized and

[3] **Camera** API class page:
`https://developer.android.com/reference/android/hardware/Camera.html`

[4] **Camera2** API Developer Reference page: `https://developer.android.com/reference/android/hardware/camera2/package-summary.html`

invoked in apps targeting older devices. The Android API always indicates deprecated aspects in the developer documentation and suggests newer alternatives when possible.

2.5.1 Packages & Classes

The Android API is organized as a series of **packages**. Packages are essentially folders or directories where the API contents for official components of Android development (and all instructions that allow developers to utilize it) live. Full package names are structured similarly to nested folders on a computer's file system, but use periods instead of slash characters to denote nesting. Full package names are usually organized by purpose; for instance, the `android.hardware` package contains hardware-specific classes related to the camera and sensors (which handle changes in the phone's accelerometer, gyroscope, ambient temperature, etc.), including the Camera class itself. A package can be thought of in simpler terms as a home for a collection of related classes. The hardware package summary page[5] contains more details on the structure and contents of the `android.hardware` package.

A **class** in the Android API is a grouping of instructions and information related to a single feature or component in or on an Android phone. As stated above, classes are sorted by category into packages based on their purpose. To use a class, it must be **imported** into developer app code by listing the exact location of the class first, and then it can be referred to subsequently in code by the class name itself. For example, since the Camera class is located in the `android.hardware` package, to use the Camera class an app developer would have to connect his or her app in code using

[5] `android.hardware` package summary page:
`https://developer.android.com/reference/android/hardware/package-summary.html`

the full location of the Camera class: `android.hardware.Camera`. Note also that a class can have other classes nested inside of it, but that is usually reserved for classes with very specific needs.

2.5.2 App Permissions & Manifest file

To create apps, developers use a combination of their own classes they create with specific behaviors and existing API classes. To use some API classes or features (including a device's camera), certain **permissions** may sometimes be required. Android users have likely seen permissions in action when installing and updating apps on the device. Apps that need to utilize certain features or functionality that may be considered personal to the user as decided by Google, such as dialing emergency contact numbers, modifying or adding to an inserted SD card, anything involving credit cards or monetary transactions, or taking pictures by directly using any device's camera, are required to prompt the user before installation or updates to warn users of these possible app behaviors. Permissions are declared in an app similarly to packages; for instance, the Camera permission is `android.permission.CAMERA`. For most purposes, Google has an official Android app permissions list with permissions that can be used for most general purpose developer needs, but app developers can also declare and use their own custom permissions. For more detail on permissions and for a full list of official Android-defined app permissions, please refer to the 'Permissions' section in the App Manifest API guide page[6].

Permissions are implemented on the code side inside of the app's **Manifest file**. Every Android app requires an App Manifest file in XML format (***AndroidManifest.xml***).

[6] **App Manifest** API Guide page:
`https://developer.android.com/guide/topics/manifest/manifest-intro.html`

In addition to permissions, the Manifest file contains various configurations and settings such as the app's version number, target API range, the list of used features in the app, and any other application-specific settings or code options such as **Activities** and **Intents** (discussed further in Chapter 3) that the app is using. An example Android app Manifest file is given below (***Figure 2.2***):

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="edu.calpoly.android.lab5"
        android:versionCode="1"
        android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="10"
              android:targetSdkVersion="17" />

    <permission android:name="edu.calpoly.android.lab5.permission.MAPS_RECEIVE"
                android:protectionLevel="signature"/>

    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"/>
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>

    <uses-feature android:name="android.hardware.camera" />

    <uses-feature android:glEsVersion="0x00020000"
                  android:required="true"/>

    <application android:allowBackup="true"
                 android:label="@string/app_name"
                 android:theme="@style/Theme.Sherlock.Light" >

        <activity android:name=".WalkAbout"
                  android:label="@string/app_name"
                  android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <meta-data android:name="com.google.android.maps.v2.API_KEY"
               android:value="AIzaSyCx4YfO9ZV4QHThWyxWlgnFWRslG1NdlWE" />
        <uses-library android:required="true" android:name="com.google.android.maps" />
    </application>
</manifest>
```

Figure 2.2: Walkabout Lab Android Manifest XML file

Note a distinction between *referencing* an Android device feature and *controlling* or *manipulating* an Android device feature. For example, the WalkAbout app (dis-

cussed later in this section and in Chapter 3) is a GPS app that tracks a user's local movements using Google Maps, and the app also allows the user to take pictures using a device's camera and place timestamps on the map where pictures are taken. Since a camera is being used for taking pictures, it is included in the Manifest file as a feature, `android.hardware.camera`. However, the Camera app permission, `android.permission.CAMERA`, is missing from the manifest file. This is because the WalkAbout app only uses the device's built-in *Camera* app to take pictures instead of directly invoking and controlling the device's physical camera. For a more detailed explanation of the Camera feature and permission declarations, see the 'Manifest declarations' section on the Camera API Guide page[7]. For more information on the App Manifest file and its general structure, see the App Manifest API Guide page[8].

2.5.3 API Level Differences

Each API level is thoroughly documented by Google in several ways: a less formal Platform Highlights notes page, and a formal API Differences Report that details relative changes between any given API level and the API level immediately before it. For example, for details on API level 19 (`KITKAT`, Android version 4.4), Google has provided a Platform Highlights page on KitKat[9] and an API Differences Report describing changes between API level 18 and 19 (`JELLY_BEAN_MR2`, Android version 4.3)[10]. Each API level's Platform Highlights and API Differences Report can be

[7] **Camera** API Guide page:
`https://developer.android.com/guide/topics/media/camera.html`

[8] **App Manifest** API Guide page:
`https://developer.android.com/guide/topics/manifest/manifest-intro.html`

[9] `KitKat` **Platform Highlights** page:
`https://developer.android.com/about/versions/android-4.4.html`

[10] API **Differences Report** page for API levels 18 and 19:
`https://developer.android.com/sdk/api_diff/19/changes.html`

viewed from the live API levels table on Google's uses-sdk documentation page[11], under the '**Notes**' and '**API level**' column entries in the table respectively.

Google started publishing API Differences Reports starting with API level 3. For convenience and perspective, a collection of API Differences Report details for Android API levels 3-17 are included below in Tables 2.1 through 2.15.

Table 2.1: API Differences Report change stats between API level 2→3

API level 2→3	**Additions**	**Changes**	**Removals**	**Sum**
Packages	6	36	0	42
Classes & Interfaces	67	181	0	248
Constructors	36	3	1	40
Methods	386	64	5	455
Fields	296	68	1	365
Total	**791**	**352**	**7**	**1150**
Overall Change				**4.30%**

[11] **uses-sdk** documentation page:
`https://developer.android.com/guide/topics/manifest/uses-sdk-element.html`

Table 2.2: API Differences Report change stats between API level 3→4

API level 3→4	Additions	Changes	Removals	Sum
Packages	4	26	0	30
Classes & Interfaces	22	73	2	97
Constructors	6	8	0	14
Methods	80	43	3	126
Fields	208	27	29	264
Total	**320**	**177**	**34**	**531**
Overall Change				**2.45%**

Table 2.3: API Differences Report change stats between API level 4→5

API level 4→5	Additions	Changes	Removals	Sum
Packages	3	30	0	33
Classes & Interfaces	85	128	0	213
Constructors	3	11	0	14
Methods	202	67	0	269
Fields	234	205	1	440
Total	**527**	**441**	**1**	**969**
Overall Change				**2.16%**

Table 2.4: API Differences Report change stats between API level 5→6

API level 5→6	Additions	Changes	Removals	Sum
Packages	0	4	0	4
Classes & Interfaces	0	4	0	4
Constructors	0	0	0	0
Methods	0	2	0	2
Fields	7	1	0	8
Total	**7**	**11**	**0**	**18**
Overall Change				**0.01%**

Table 2.5: API Differences Report change stats between API level 6→7

API level 6→7	Additions	Changes	Removals	Sum
Packages	1	11	0	12
Classes & Interfaces	5	21	0	26
Constructors	1	0	0	1
Methods	52	2	0	54
Fields	22	3	0	25
Total	**81**	**37**	**0**	**118**
Overall Change				**0.48%**

Table 2.6: API Differences Report change stats between API level 7→8

API level 7→8	Additions	Changes	Removals	Sum
Packages	11	40	0	51
Classes & Interfaces	60	122	0	182
Constructors	3	3	1	7
Methods	206	37	3	246
Fields	195	23	29	247
Total	**475**	**225**	**33**	**733**
Overall Change				**5.70%**

Table 2.7: API Differences Report change stats between API level 8→9

API level 8→9	Additions	Changes	Removals	Sum
Packages	4	50	0	54
Classes & Interfaces	79	165	9	253
Constructors	29	2	1	32
Methods	511	35	28	574
Fields	141	11	2	154
Total	**764**	**263**	**40**	**1067**
Overall Change				**2.93%**

Table 2.8: API Differences Report change stats between API level 9→10

API level 9→10	Additions	Changes	Removals	Sum
Packages	1	8	0	9
Classes & Interfaces	6	9	0	15
Constructors	0	0	0	0
Methods	8	0	2	10
Fields	10	0	0	10
Total	**25**	**17**	**2**	**44**
Overall Change				**0.65%**

Table 2.9: API Differences Report change stats between API level 10→11

API level 10→11	Additions	Changes	Removals	Sum
Packages	3	38	0	41
Classes & Interfaces	109	196	0	305
Constructors	20	5	1	26
Methods	431	100	26	557
Fields	619	36	0	655
Total	**1182**	**375**	**27**	**1584**
Overall Change				**2.56%**

Table 2.10: API Differences Report change stats between API level 11→12

API level 11→12	Additions	Changes	Removals	Sum
Packages	3	24	0	27
Classes & Interfaces	5	65	6	76
Constructors	4	1	6	11
Methods	75	27	2	104
Fields	87	9	0	96
Total	**174**	**126**	**14**	**314**
Overall Change				**1.15%**

Table 2.11: API Differences Report change stats between API level 12→13

API level 12→13	Additions	Changes	Removals	Sum
Packages	0	11	0	11
Classes & Interfaces	2	29	0	31
Constructors	0	0	0	0
Methods	22	12	0	34
Fields	68	1	0	69
Total	**92**	**53**	**0**	**145**
Overall Change				**0.04%**

Table 2.12: API Differences Report change stats between API level 13→14

API level 13→14	Additions	Changes	Removals	Sum
Packages	5	48	0	53
Classes & Interfaces	90	192	0	282
Constructors	19	2	0	21
Methods	285	94	29	408
Fields	405	34	16	455
Total	**804**	**370**	**45**	**1219**
Overall Change				**3.95%**

Table 2.13: API Differences Report change stats between API level 14→15

API level 14→15	Additions	Changes	Removals	Sum
Packages	0	21	0	21
Classes & Interfaces	12	34	0	46
Constructors	2	1	0	3
Methods	25	3	0	28
Fields	38	2	0	40
Total	**77**	**61**	**0**	**138**
Overall Change				**0.14%**

Table 2.14: API Differences Report change stats between API level 15→16

API level 15→16	Additions	Changes	Removals	Sum
Packages	4	39	0	43
Classes & Interfaces	57	211	0	268
Constructors	12	22	4	38
Methods	381	151	20	552
Fields	171	46	4	221
Total	**625**	**469**	**28**	**1122**
Overall Change				**2.52%**

Table 2.15: API Differences Report change stats between API level 16→17

API level 16→17	Additions	Changes	Removals	Sum
Packages	2	35	0	37
Classes & Interfaces	41	111	2	154
Constructors	2	1	0	3
Methods	150	37	19	206
Fields	155	69	3	227
Total	**350**	**253**	**24**	**627**
Overall Change				**1.26%**

The overall change percentage seems small for each API level despite the total number of changes, but this can be attributed to the ever-growing size of the API (changes represent a smaller piece of the entire API as the API grows). The overall change percentage between two adjacent API levels is calculated using the following formula, per the API Differences reports:

$$\text{Percentage difference} = \frac{100 * (\text{added} + \text{removed} + 2*\text{changed})}{\text{sum of public elements in BOTH APIs}}$$

Figure 2.3: Android API Differences Report overall change calculation

The total API difference between any two non-adjacent levels (such as API level 3 and API level 17) must be calculated differently since the API Differences Reports focus on relative change percentages and amounts between two adjacent levels. Unfortunately, the API Differences Reports do not include total numbers of packages, classes and other statistical indices; therefore, an accurate total API difference percentage between two non-adjacent levels cannot be easily calculated and produced.

2.6 Old Labs

As mentioned previously, a series of Android development lab exercises were created prior to the ones predominantly featured in this project. These labs (referred to as **old labs**) were created by James Reed, a Cal Poly M.S. Computer Science alumnus. Serving as the bulk of his Master's thesis work [18], these labs were used as a curriculum for Dr. David Janzen's *Android Application Development* course beginning in Cal Poly SLO's Winter 2010 quarter [19]. This initial version of the Android Application Development course would later be referred to as *Android App Course v1* (referred to in shorthand throughout the rest of this paper as **Course v1**) after newer versions of the course were taught with updated lab materials.

These labs were designed for students to learn the basics of Android application development, writing code in mostly empty coding projects (or **skeleton** projects)

provided as a standard development environment for all students when starting each lab exercise. The skeleton projects also (usually) provided **unit tests** with which students could test their code for proper functionality throughout the exercise, allowing students to self-test their progress on each lab for success or failure regarding code and app behavior expectations. Students used the Eclipse IDE program [27] to write, maintain and test their code.

The original lab exercises used in Course v1 have since been archived[12]. A general overview of these labs is shown below in ***Table 2.16***:

Table 2.16: Summary of Android App Course v1 lab content

Lab	**Topics**
Lab 1	Install tools, simple UI
Lab 2	More advanced UIs
Lab 3	More UI: Custom Views, Menus, Dialogs
Lab 4	Persistence: SharedPreferences, SQLite Databases
Lab 5	Maps, Camera, Files
Lab 6	ContentProviders, Services, Notifications, Threads
Lab 7	SMS, Accelerometer

[12]**Course v1** archive page: `https://sites.google.com/site/androidcoursearchive`

Each Course v1 lab exercise's learning goals are given below in order of presence in the course syllabus:

- **Lab 1** ("HelloWorld" app: Introduction to the Android programming environment and tools)
  - How to set up the development environment
  - How to create a 'Hello World' Android application
  - How to use the Android Emulator
  - How to install and run an application on a physical device
  - How to create a simple User Interface (UI)

- **Lab 2** ("JokeList" app: *Create a list of jokes in an app*)
  - How Views, View Groups, Layouts and Widgets work and relate to each other
  - How to declare layouts dynamically at runtime
  - How to reference resources in code and from other resource layout files
  - How to use Events and Event Listeners

- **Lab 3** ("JokeList 2.0" app: *Upgrade the previous lab's app to use more Android app features to populate and structure the joke list*)
  - How to declare layouts statically as an XML resource
  - How to create custom Views from scratch to suit a specific need
  - How to create Options and Context Menus
  - How to use Adapters and AdapterViews to bind a View class to data
  - How to establish HTTP connections
  - How to create Dialogs and Notifications

- **Lab 4** ("JokeList 3.0" app: *Upgrade the previous lab's app to use even more Android app features to better maintain the joke list*)
  - How to save and restore data as Application Preferences
  - How to save and restore data as Instance State
  - How to create an SQLite Database
  - How to manage database connections
  - How to insert, update, remove, and retrieve data from an SQLite Database
  - How to work with and manage Cursors
  - How to use CursorAdapters

- **Lab 5** ("WalkAbout" app: *Create a GPS path tracking app with picture-taking and map-marking functionality*)
  - How to incorporate Google Maps into an application
  - How to register for and receive GPS location information
  - How to draw graphics on the screen using the Canvas class
  - How to create Google Maps Overlays
  - How to use the Camera
  - How to write data to an SD card
  - How to create and delete private application files
  - How to launch and receive results from Activities

- **Lab 6** ("AppRater" app: *Create an app with a list of other apps to be rated by users*)
  - How to create and use ContentProviders
  - How to create and use Services
  - How to broadcast Intents
  - How to post notifications in the Notification Bar
  - How to respond to Intent broadcasts by creating and using BroadcastRecievers
  - How to perform work in a background thread

- **Lab 7** (Untitled app: *Create an app with text messaging, and experimenting with various other physical Android device features*)
  - How to send SMS text messages
  - How to listen for changes in the status of a sent SMS text message
  - How to register to receive information from a device's available sensors
  - How to monitor the motion of a physical device

Each lab consists of a programming exercise which results in a relatively simple, self-contained and fully-functioning Android app upon completion, and an accompanying lab writeup to help guide students through the exercise. At the end of every lab exercise in the course, a survey was conducted by students for James Reed's thesis project to verify the success of each lab; this tradition would continue to serve in all further course offerings as a general measure of lab success for the course. Except for Lab 1, each lab comes with a skeleton Eclipse project that provides a common structure to be used as a programmatical starting point. It is worth noting that for most course offering curricula, the final lab is a shorter and much more simplified lab exercise than most of the other labs despite being the final lab exercise in the curriculum; this is because students in the course are usually concurrently working on large team-based final app projects of choice.

A second iteration of these lab exercises was used for the Summer 2010 quarter at Cal Poly SLO, again for the same Android Application Development course taught by Dr. Janzen (appropriately dubbed *Android App Course v2* or **Course v2**) [20]. Course v2 used the same curriculum and number of labs as Course v1, but with minorly revised versions of James Reed's Course v1 lab exercises and writeups encountered after teaching Course v1 to students. The Course v2 curriculum and labs have also been archived[13].

Both Course v1 and Course v2 exercises were extremely well-received at their time of publish, both by students taking the course (as indicated by survey results) and worldwide by many instructors and self-taught developers alike. Dr. Janzen received hundreds of emails asking for further resources or permission to use the exercises in teaching programs outside of Cal Poly, SLO. However, as more advanced Android

[13]**Course v2** archive page: `https://sites.google.com/site/androidappcourse`

devices were released and the Android API continued to evolve, the Course v2 materials started to show signs of age despite the course's successes and ongoing high demand. Google, who had graciously gifted the school with Android phones for the course (so students who did not own an Android phone could have a device to test lab apps on), could not be counted on to continue providing newer devices to work on. Many aspects of the lab exercises started to become deprecated or replaced in the Android API, and newer core concepts and functionality would be missed out on in the older exercises. Furthermore, James Reed had already graduated from Cal Poly SLO and was no longer actively working on the lab exercises, and the course instructor could afford little time to spare between teaching other courses and helping students to significantly update the course content. It was clear that there was no distinct long-term solution yet for updating and maintaining the lab exercises for use in any educational setting.

Chapter 3

NEW LABS

This chapter deals primarily with the newest content for the Android Application Development course at Cal Poly SLO, specifically for the Spring 2013 course offering [17]. This iteration of the course is referred to as **Course v3** as it was the third time teaching the course with updated course materials. Course v3 contains significant changes to Course v2 relative to changes made between Course v1 and Course v2. These changes and additional content will be demonstrated and discussed in detail throughout this chapter.

3.1 Course v3 Considerations & Approach

As was made apparent at the end of the previous chapter, Course v2 was showing signs of age not even a year after release as new Android devices and API levels continued to hit the market and development scenes respectively. Android device manufacturers and Google itself showed no signs of slowing down on device production and changing the API. New features continued to be frequently added and iterated upon, and as such an immediate containment solution to this "moving target" issue was not apparent.

An additional problem to consider for Course v3 was students' limited access to Android devices. Students of the course who could use their own device had a variety of older and newer Android phones, but it was unacceptable to assume that all students owned personal Android devices to use throughout the course. This was further

complicated by the fact that phones provided by Google for previous iterations of the course were also significantly aged by this point; securing newer donated phones for the course could not be depended on. Android's emulated device implementation (Android Virtual Device, or **AVD**) was still not fully up to standards relative to physical devices at the time, and some lab exercises would not work at all on virtual devices (notably the WalkAbout app, which relies on GPS and moving around), eliminating virtual devices as a proper or even partial substitute. This created a wider variety of devices the course needed to account for in new lab exercise content.

Additionally, support for older Android devices had yet to be properly provided by Google. At the time, Android's support library[1] was very limited and not widely adopted by developers. The support libraries provided limited device compatibility features, and support library documentation was sparse and relatively unorganized; only several examples of support API library classes were viewable and runnable, and only via direct download of example code projects by interested developers. Thus, the official Android support library could not be relied on for maintaining lab exercises for older Android devices and API levels, so an out-of-the-box solution had to be properly researched.

Based on these identified problems, it was determined that Course v3 material would be designed around the following guidelines:

1. General lab exercise content format and project structure would remain identical or as similar to previous course incarnations when reasonable. Lab exercises would still contain unit tests for students to test their progress when reasonable.

[1] **Android support library** topic page:
`https://developer.android.com/topic/libraries/support-library/index.html`

2. The course would continue to emphasize mastering Android app development fundamentals over newer API and feature implementation developments, where reasonable (if a technique or aspect of the Android API previously taught in the course was deprecated and/or heavily reworked since Course v2, it would be most likely phased out of Course v3). Course v2 lab exercise content estimated to be less important from a fundamental perspective during Course v3's expected lifetime would likely not be carried over.

3. A new emphasis on app compatibility (mostly backwards) regarding devices and applications, to account for the wider variety of device platforms based on device usage statistics at the time. Most if not all new lab exercises would be designed to fit a specific range of Android APIs, and lab exercise content would be modified to account for extra project needs arising from backwards-compatibility needs.

4. Student lab exercises and final course projects (teamwork-based apps developed separately from the lab exercises) would continue to be emphasized for and evaluated upon completion on Android phone devices and phone emulators only, not tablets. However, for final projects, students would be encouraged to experiment beyond the Android development aspects taught in the lab exercises (including designing for tablets).

It was decided early during Course v3 planning that not all aspects of the course could be properly analyzed and factored into new course material until shortly before or even during the duration of the course, and compromises were made for course content to account for this at the start of the course. This had the added benefit of accounting as much as possible for API changes that could occur during the duration of the course (up until that lab was published and finalized in the course schedule). However, the easiest decision before the course began was to limit lab exercise de-

velopment to Android API levels 10-17. At the time, United States Android phone usage statistics indicated that Android 2.3.3-2.3.4 (`Gingerbread_MR1`) was the most used version across devices, and most newer phone devices were already running either the Ice Cream Sandwich range (Android 4.0-4.0.4) or the newest Android version range at the time, Jelly Bean (Android 4.1-4.3). Most tablet devices were running either Ice Cream Sandwich or the older Honeycomb range (Android 3.0.x-Android 3.2), and since Honeycomb was tablet-exclusive (and tablets were largely considered newer devices at the time), it was decidedly ignored in favor of Ice Cream Sandwich which covered both phones and tablets.

Following the decision to limit the Course v3 lab exercises to Android API levels 10-17, a backwards-compatibility implementation was chosen in **ActionBarSherlock** [30]. Written, published and maintained since 2012 by Jake Wharton, ActionBarSherlock (occasionally referenced as **ABS**) was essentially a standardization of the Ice Cream Sandwich UI and app layout (especially the **Action Bar**) that worked in apps targeting older devices with prior API version support. Emerging as a proven compatibility solution by a variety of Android app developers, including for API levels 10-17 that were chosen for the course, ABS was experimented with and eventually settled upon as the backwards-compatibility support library of choice for the new lab exercises in Course v3.

3.2 New vs Old Lab Objectives

Based on problems and considerations discussed above, newly updated and modified lab exercises were devised for Course v3 to replace the older Course v2 labs. These **new labs** were largely derivatives of the older labs, some with only minor changes.

However, others contained drastic updates, improvements and modifications over their predecessors.

The new lab exercises for Course v3 emphasized the following development objectives, shown side-by-side with the old lab objectives for comparison:

Table 3.1: New vs Old Labs 1 & 2 Objectives

New Labs	Old Labs
Lab 1 ("HelloWorld" app: Introduction to the Android programming environment and tools) • How to set up the Android Development Environment • How to create a basic "Hello World" Android Application containing a simple Graphical User Interface (GUI) • How the various parts and features of an Android Project work, and how they relate to each other • How to use the Android emulator • How to install and run an application on a physical Android device	**Lab 1** ("HelloWorld" app: Introduction to the Android programming environment and tools) • How to set up the development environment • How to create a 'Hello World' Android application • How to use the Android Emulator • How to install and run an application on a physical device • How to create a simple User Interface (UI)
Lab 2 ("JokeList" app: *Create a list of jokes in an app*) • How to create an Android Test Project and test Android Activities • How Views, View Groups, Layouts, and Widgets work and how they relate to each other • How to declare layouts dynamically at runtime • How to reference resources in code and from other resource layout files • How to use Android's system debug output monitor LogCat for debugging • How to use Events and Event Listeners	**Lab 2** ("JokeList" app: *Create a list of jokes in an app*) • How Views, View Groups, Layouts and Widgets work and relate to each other • How to declare layouts dynamically at runtime • How to reference resources in code and from other resource layout files • How to use Events and Event Listeners

Table 3.2: New vs Old Labs 3 & 4 Objectives

New Labs	Old Labs
Lab 3 ("JokeList 2.0" app: *Upgrade the previous lab's app to use more Android app features to populate and structure the joke list*) • How to declare layouts statically as an XML resource • How to create custom Views using existing components and show them in a scrollable ListView • How to add custom icons to a component and apply State Lists • How to use Adapters and AdapterViews to bind a front-end View class to its corresponding back-end data • How to create Toast Notifications • How to add backwards-compatible Android menus and nested menus using ActionBarSherlock (Action Bar) • How to add Contextual Menus for individual Views (Contextual Action Bar)	**Lab 3** ("JokeList 2.0" app: *Upgrade the previous lab's app to use more Android app features to populate and structure the joke list*) • How to declare layouts statically as an XML resource • How to create custom Views from scratch to suit a specific need • How to create Options and Context Menus • How to use Adapters and AdapterViews to bind a View class to data • How to establish HTTP connections • How to create Dialogs and Notifications
Lab 4 ("JokeList 3.0" app: *Upgrade the previous lab's app to use even more Android app features to better maintain the joke list*) • How to save & restore data as Application Preferences • How to save & restore data as Instance State • How to create, maintain and interact with Content Providers and tables • How to implement SQLite database helper classes to aid in table access, database initialization and minimal management • How to access and modify a SQLite database table using URIs and content provider database calls • What Cursors and CursorAdapters are and how they interact with Views • How to asynchronously load and automanage Cursors using LoaderManager and CursorLoader	**Lab 4** ("JokeList 3.0" app: *Upgrade the previous lab's app to use even more Android app features to better maintain the joke list*) • How to save and restore data as Application Preferences • How to save and restore data as Instance State • How to create an SQLite Database • How to manage database connections • How to insert, update, remove, and retrieve data from an SQLite Database • How to work with and manage Cursors • How to use CursorAdapters

Table 3.3: New vs Old Labs 5 & 6 Objectives

New Labs	Old Labs
Lab 5 ("JokeList 4.0" app: *Upgrade Lab 3's app to work with HTTP connections*) • How to establish HTTP connections • How to use AsyncTask	
Lab 6 ("WalkAbout" app: *Create a GPS path tracking app with picture-taking and map-marking functionality*) • How to display an interactive Google Map inside an application • How to register for an Android Google Maps API key online • How to enable Android Google Maps in an application project • How to register for and receive GPS location information • How to launch and receive outside Activities • How to change the way the Google Maps viewing camera looks at the map • How to draw basic shapes and Markers on a Google Map • How to use the Camera. • How to call the Camera Activity using an Intent • How to store pictures taken to internal storage (SD card, internal memory, etc.) • How to modify private application files for saving and loading data.	**Lab 5** ("WalkAbout" app: *Create a GPS path tracking app with picture-taking and map-marking functionality*) • How to incorporate Google Maps into an application • How to register for and receive GPS location information • How to draw graphics on the screen using the Canvas class • How to create Google Maps Overlays • How to use the Camera • How to write data to an SD card • How to create and delete private application files • How to launch and receive results from Activities

Table 3.4: New vs Old Lab 7 Objectives

New Labs	Old Labs
Lab 7 ("AppRater" app: *Create an app with a list of other apps to be rated by users*) • How to create and use Services using ServiceIntent • How to start and stop Services • How to perform repeated procedures using Timer and TimerTask • How to broadcast Intents • How to respond to Intent broadcasts using a BroadcastReceiver • How to post Notifications in the Notification area using NotificationManager and PendingIntent • How to create Notifications using a Builder • How to use and record changes in an interactive RatingBar • How to check for app install status on an Android device using the PackageManager	**Lab 6** ("AppRater" app: *Create an app with a list of other apps to be rated by users*) • How to create and use ContentProviders • How to create and use Services • How to broadcast Intents • How to post notifications in the Notification Bar • How to respond to Intent broadcasts by creating and using BroadcastRecievers • How to perform work in a background thread
	Lab 7 (Untitled app: *Create an app with text messaging, and experimenting with various other physical Android device features*) • How to send SMS text messages • How to listen for changes in the status of a sent SMS text message • How to register to receive information from a device's available sensors • How to monitor the motion of a physical device

It is important to note that the new Lab 5 ("JokeList 4.0" app) was designed by course instructor David Janzen as a short but informative exercise, thus bumping all

subsequent new labs up by one number. The contents of the old Lab 7 exercise were deemed not developed or important enough for the newer course, and thus it was cut for Course v3 instead of appearing in the curriculum as a Lab 8 exercise.

3.3 Course v3 Lab Exercises

Specific programmatic, structural or conceptual changes between old and new labs are further discussed below. Each lab's written guide (old and new) is linked to for side-by-side detailed lab exercise comparison at one's discretion. All images shown in the subsections below are taken from the respective lab's guide. Note that included images in the subsections below demonstrating the lab app may be a combination of screenshots of the same app instance taken in sequence at separate points in time and shown side-by-side in order, and are not intended to depict one app running concurrently on multiple devices.

In general, each Course v2 lab guide was significantly reworked and updated for Course v3 for improved ease of access and readability. A table of contents was added to most guides, and specific steps referring to computer lab machine setups (in Cal Poly SLO's computer labs) were omitted. Steps and explanations in each guide concerning Android API references were carefully researched and re-written to account for API changes or deprecations that occurred between Course v2 and Course v3. Old graphics depicting apps running in older versions of Android were completely replaced with app screenshots running in newer versions of Android. New explanations or graphics were added to each guide where appropriate to better illustrate and explain development concepts and commonly-encountered misconceptions or issues.

3.3.1 New Lab 1

HelloWorld app: Introduction to the Android programming environment and tools.

Lab Guides: Old[2] vs New[3]

Change summary: Aside from minor revisions and updates, the app structure and emphasized learning material covered in the lab generally follows that of Old Lab 1.

Notable changes:

- Required minimum Eclipse IDE [27] version updated to version 4.2 from version 3.4 (applies to all labs; included as a point for this lab as a big portion of the lab involves setting up the development environment and tools for all labs in the course).
- Additional expanded section on installing specific Android versions for older device/API backwards-compatibility in this and all future labs in the course. Course v3 labs targeted Android API levels 10-17, as opposed to Course v2's minimum target API level of 4 (Android OS version 1.6).
- Additional details on project file structure added, including details explaining new Android app development project files introduced by Google into the SDK since Course v2.
- Screenshots of app running on emulator and physical devices added.
- Additional steps for modifying an **Android Virtual Device** (or **AVD**) added.

[2] **Old Lab 1 guide** page: `https://sites.google.com/site/androidappcourse/labs/lab-1`

[3] **New Lab 1 guide** page: `https://sites.google.com/site/androidappcoursev3/labs/lab-1`

Lab & App Screenshots: The lab primarily consists of introductory learning exercises and explanations, mostly revolving around setting up the app `Activity` class layouts and behavior (***Figure 3.1*** below). The resulting final product consisted of an app allowing the user to enter their name and press a button to confirm, with their entered name displayed on a new screen (***Figure 3.2*** below).

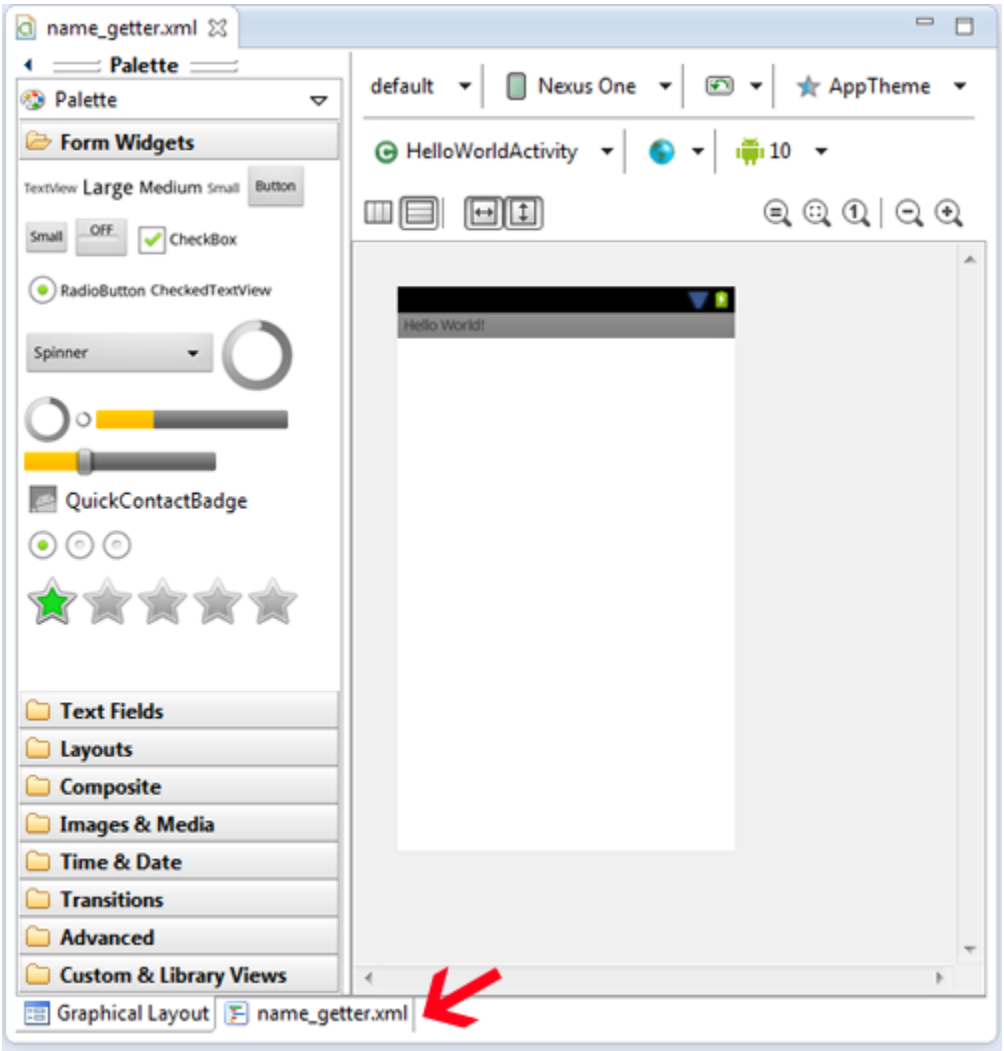

Figure 3.1: Lab 1: Hello World IDE app layout setup

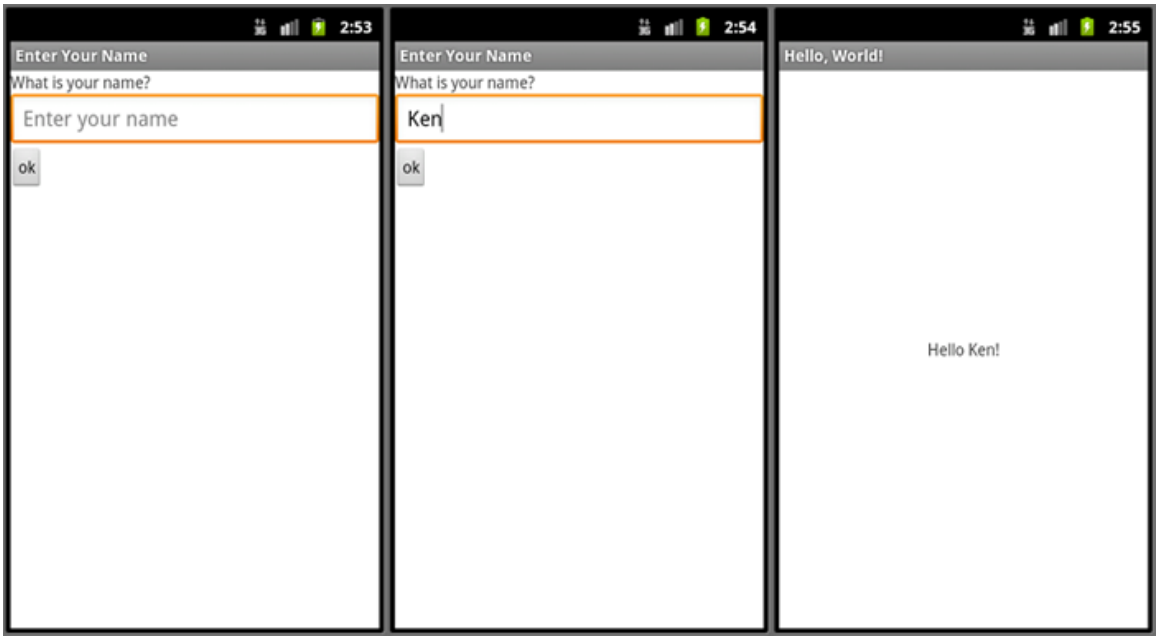

Figure 3.2: Lab 1: Hello World final app

3.3.2 New Lab 2

JokeList app: Create a list of jokes in an app.

Lab Guides: Old[4] vs New[5]

Change summary: Aside from minor revisions and updates, the app structure and emphasized learning material covered in the lab generally follows that of Old Lab 2.

Notable changes:

- Additional lab step and guide section added for creating Android Test Projects in Eclipse, as one of two proper methods for unit testing Android Eclipse app development projects. All content from the Course v2 guide concerning unit testing was consolidated inside this new Android Test Project guide section.
- Additional lab step and guide section describing the Android SDK's LogCat debugging framework, and how to debug live Android device apps from Eclipse.
- Added additional information regarding changing an app's text size as a step when setting up the `SimpleJokeList` class, and modifying test cases to match new text sizes.
- Additional images showing scrolling functionality of app's vertical `ScrollView` added.

[4] **Old Lab 2 guide** page:
`https://sites.google.com/site/androidappcourse/labs/lab-2-1`

[5] **New Lab 2 guide** page:
`https://sites.google.com/site/androidappcoursev3/labs/lab-2-1`

Lab & App Screenshots: The lab primarily consists of learning nested `View` layouts (***Figure 3.3*** below) and enabling outside input to them from the app's visual frontend and data backend (***Figure 3.4*** below). The exercise also introduced Android LogCat app debugging (***Figure 3.5*** below) and skeleton project unit tests for self-progress testing (***Figure 3.6*** below). The resulting final product consisted of an app allowing the user to pre-populate and add to a list of jokes in the scrolling nested app layout (***Figure 3.7*** below).

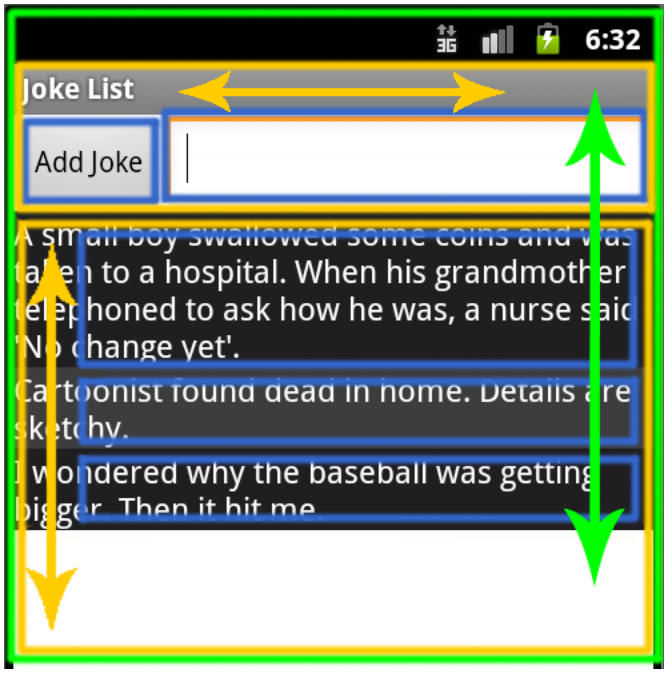

Figure 3.3: Lab 2: Joke List nested layout

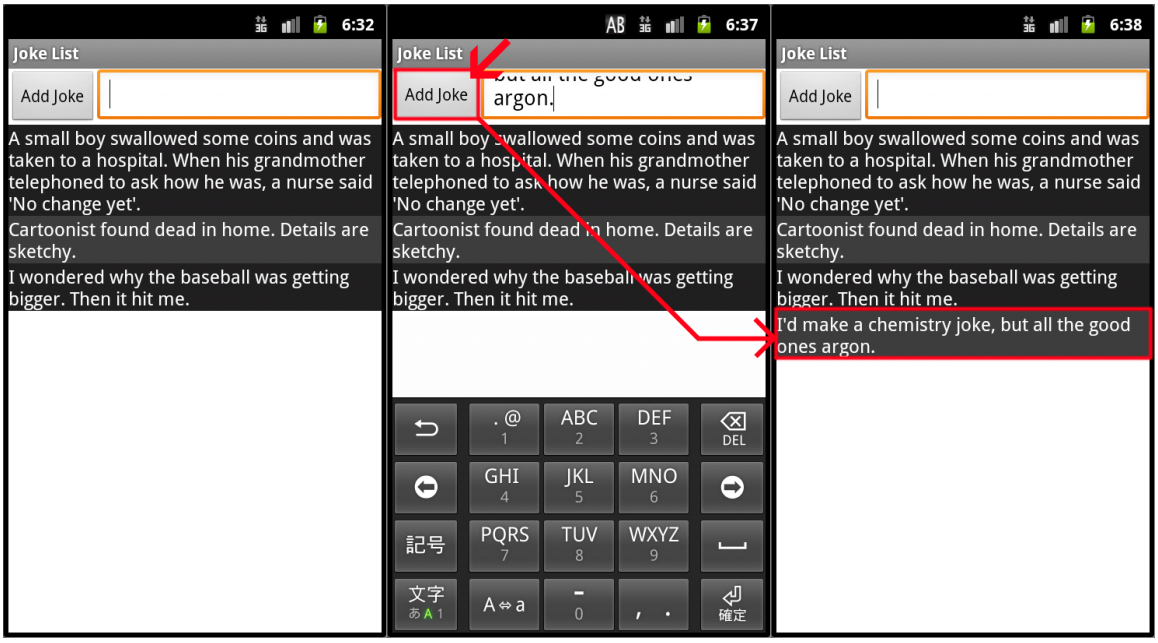

Figure 3.4: Lab 2: Joke List adding behavior

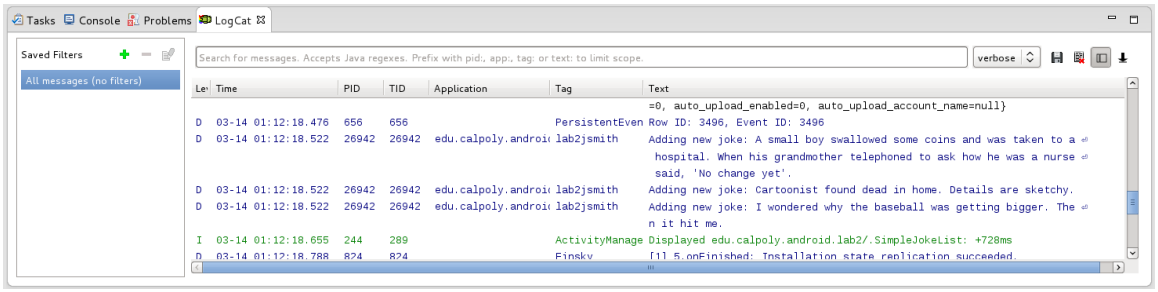

Figure 3.5: Lab 2: LogCat debug statements

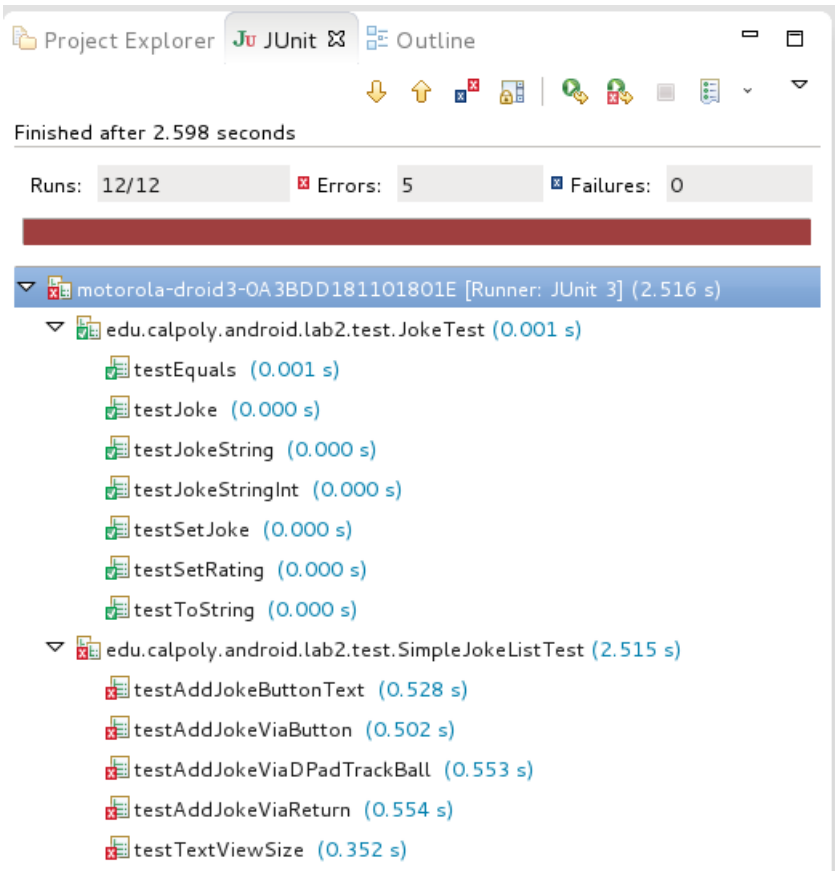

Figure 3.6: Lab 2: Running unit tests

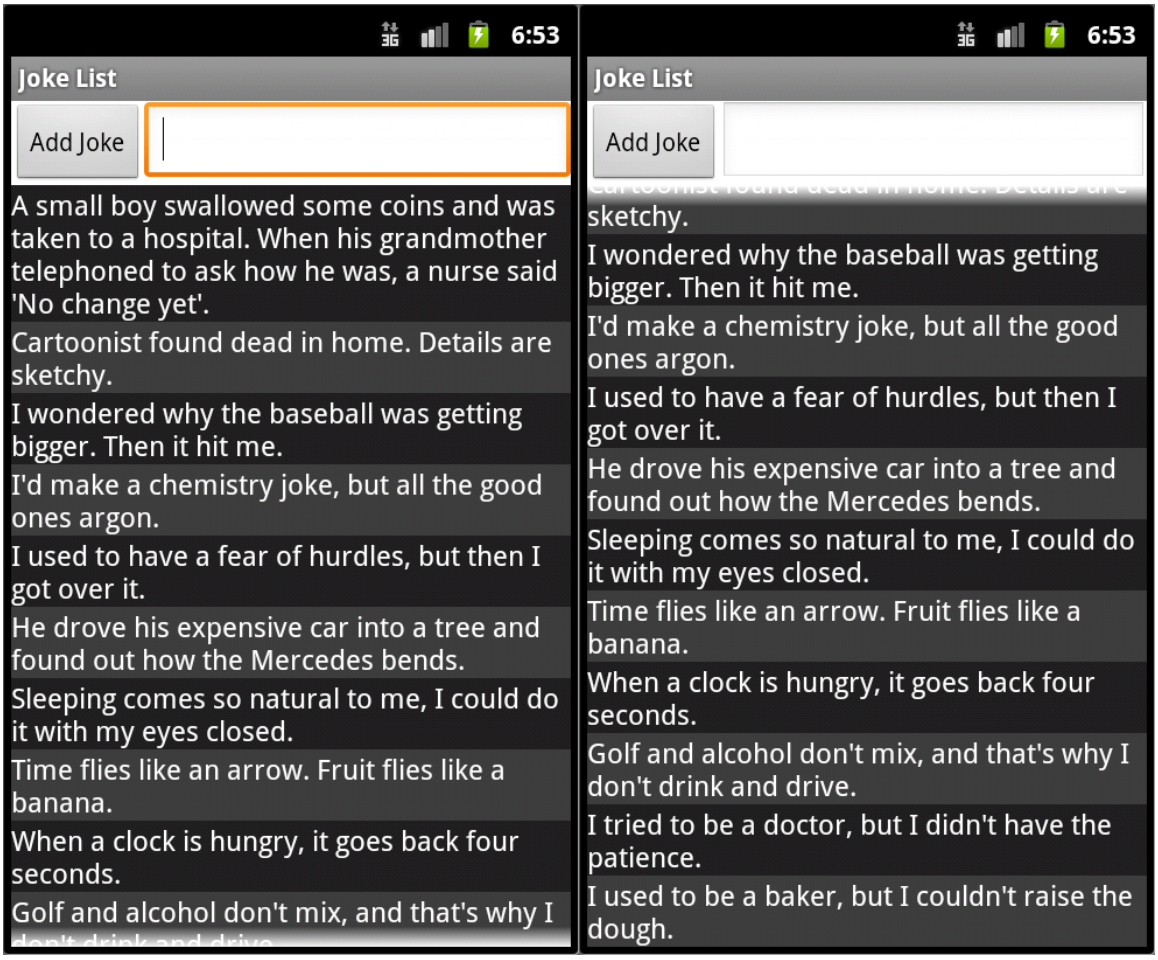

Figure 3.7: Lab 2: Final Joke List app

3.3.3 New Lab 3

JokeList 2.0 app: Upgrade the previous lab's app to use more Android app features to populate and structure the joke list.

Lab Guides: Old[6] vs New[7]

Change summary: Significant rework to the Options and Context Menu implementations, replaced with the ActionBarSherlock Action Bar and Context Menu (support implementation for backwards-compatibility). Moderate rework for removal of joke's current visual state (expanded or collapsed), replacing it with a new custom image and state machine implementation for joke rating. Apart from these changes, the app structure and emphasized learning material covered in the lab generally follows that of Old Lab 3.

Notable changes:

- New additional lab steps and guide sections for the more compatible Action Bar and Context Menu implementations provided by the third-party ActionBarSherlock support library, replacing the original Options and Context Menu implementations.
- Removal of original lab steps and guide sections to maintain current `JokeView` states as expanded or collapsed, primarily due to buggy behavior in the previous implementation and being deemed as a rather frustrating and unhelpful learning exercise.

[6] **Old Lab 3 guide** page: `https://sites.google.com/site/androidappcourse/labs/lab-3`
[7] **New Lab 3 guide** page:
`https://sites.google.com/site/androidappcoursev3/labs/lab-3`

- Removal of original lab step and guide section on establishing HTTP connections (moved to Lab 5) due to lab exercise overloading.
- New additional lab steps and guide sections for implementing custom images for joke ratings in the app's joke list using `StateList`s.

Lab & App Screenshots: The lab primarily consists of updating the previous lab's app to allow for joke ratings and joke list modifications and rating filters through the introduction of the backwards-compatible ActionBarSherlock Action Bar and Context Menu (***Figure 3.8*** and ***Figure 3.9*** below). The resulting final product consisted of an app allowing the user to pre-populate and freely add to, remove from, rate and sort a list of jokes in a refined app layout.

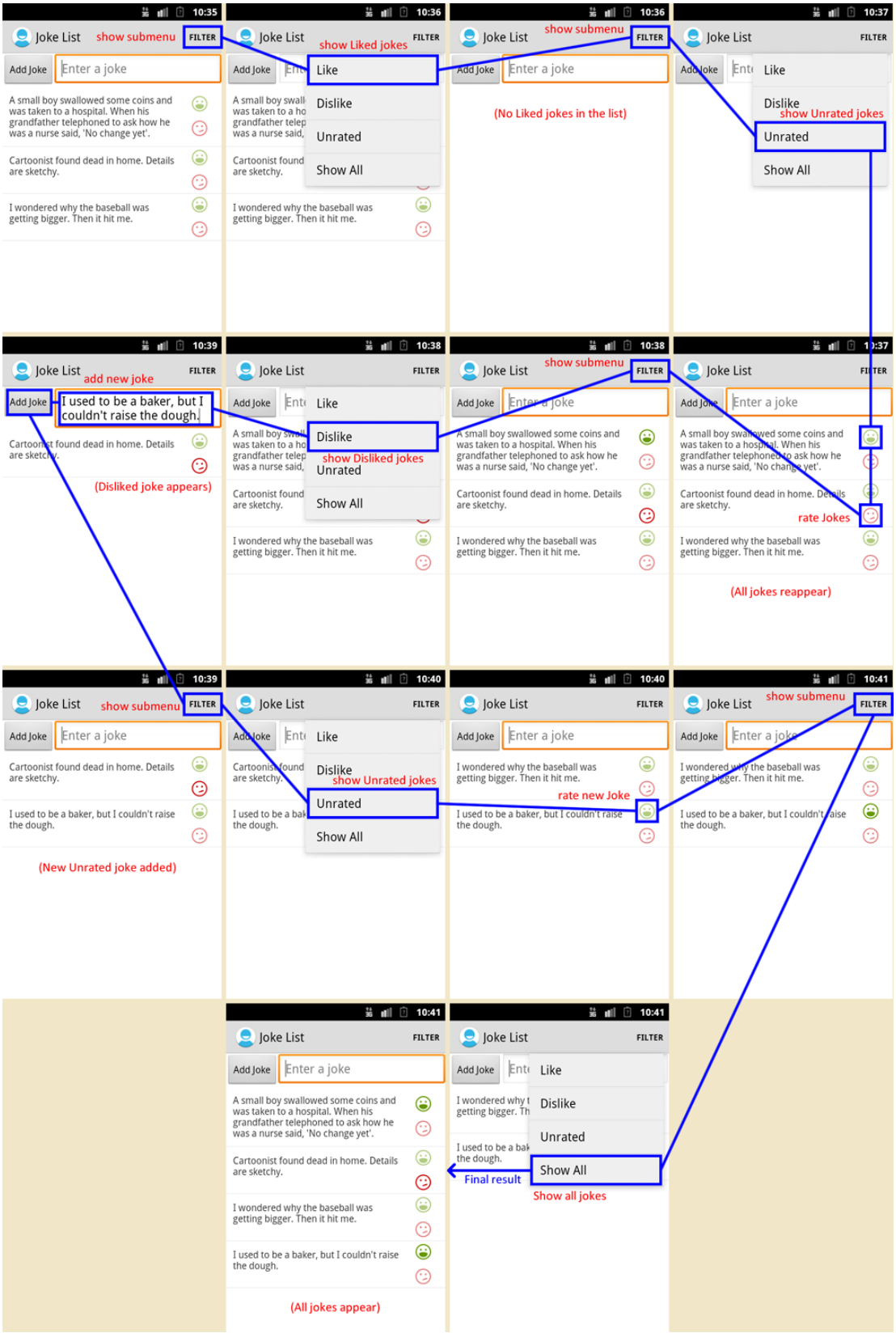

Figure 3.8: Lab 3: Joke List 2.0 list filter menu items storyboard

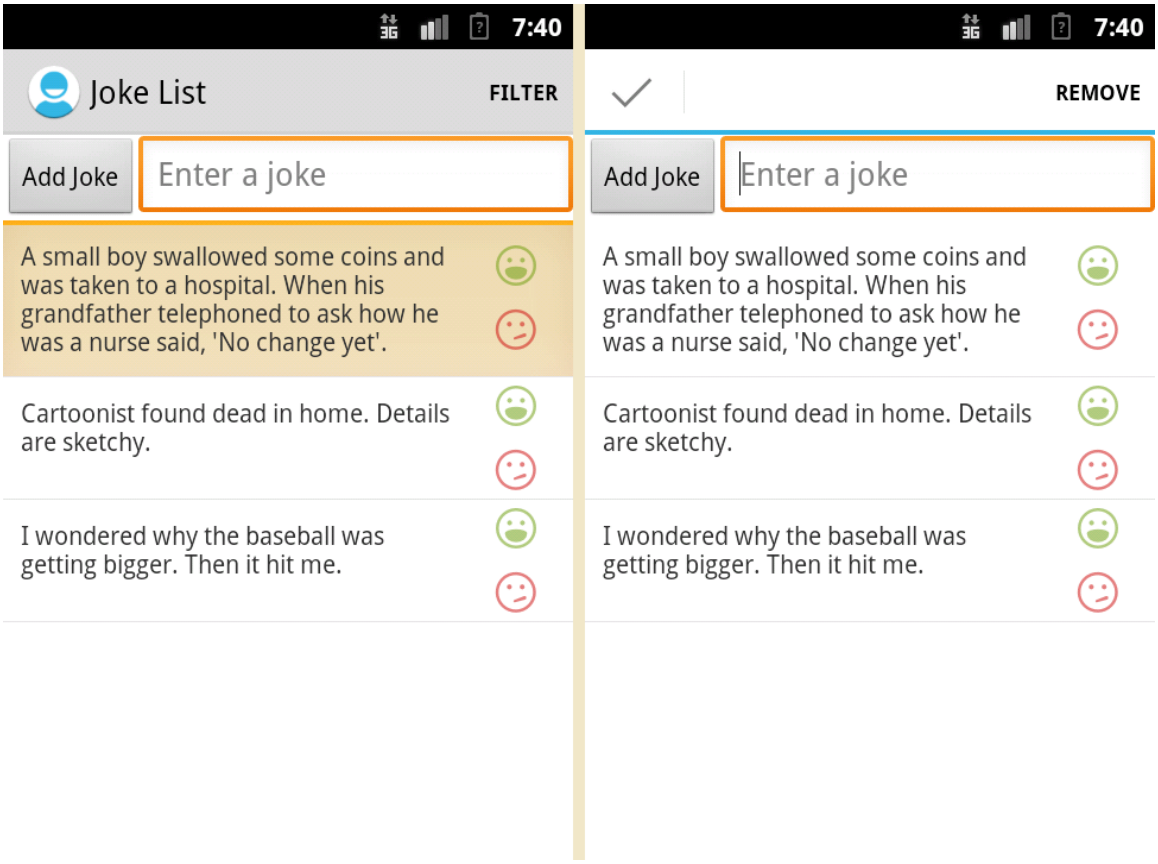

Figure 3.9: Lab 3: Joke List 2.0 context menu joke deletion

3.3.4 New Lab 4

JokeList 3.0 app: Upgrade the previous lab's app to use even more Android app features to better maintain the joke list.

Lab Guides: Old[8] vs New[9]

Change summary: Significantly reorganized and updated lab exercise; roughly to the same degree as New Lab 6 due to the sheer amount of content covered. The Old Lab 4 assignment and guide were deemed too light on detail and confusing as a lab assignment in Course v2. Android API deprecations at the time required further additions and complications to account for in the lab restructure. New Lab 4 was decidedly made into more of a teaching exercise than a self-learning lab in an attempt to fix the 'guess and check' methodology many students had resorted to for Old Lab 4, with significantly more database explanations provided. Only the general app content and instructional direction of Old Lab 4 were largely preserved in this new lab exercise.

Notable changes:

- New additional lab steps and guide sections for the more compatible Action Bar and Context Menu implementations provided by the third-party ActionBarSherlock support library, replacing the original Options and Context Menu implementations.

- New additional lab guide sections and images further describing database-related classes such as `ContentProvider`s and `ContentURI`s.

[8] **Old Lab 4 guide** page: `https://sites.google.com/site/androidappcourse/labs/lab-4`
[9] **New Lab 4 guide** page:
`https://sites.google.com/site/androidappcoursev3/labs/lab-4`

- Updated lab steps and guide sections to account for deprecated `CursorAdapter` class functionality, introducing the `LoaderManager` and `CursorLoader` classes.
- Significant restructuring and reorganization of lab steps and guide sections regarding joke database implementation, with new supporting explanatory content and images for additional context to accompany the new learning emphasis.
- Added new app data flowchart graphic to illustrate high-level relationships between content provider, loader and adapter components.

Lab & App Screenshots: This lab did not make many, if any, visual changes to the app relative to the previous lab app. Instead, it primarily updated the previous lab's app to allow for preservation of the app's data when switching app orientation or migrating away from the app on the device through the utilization of instance state and shared preferences (***Figure 3.10*** and ***Figure 3.11*** below, respectively). The lab mostly focuses heavily on database implementation, and walks developers through establishing a SQLite database for managing app joke storage and maintenance (sample app data flow diagram from guide shown in ***Figure 3.12*** below). The resulting final product is an app allowing the user to pre-populate and freely add to, remove from, rate and sort a list of jokes with data preserved across app sessions, with the app left visually identical to New Lab 3's final app.

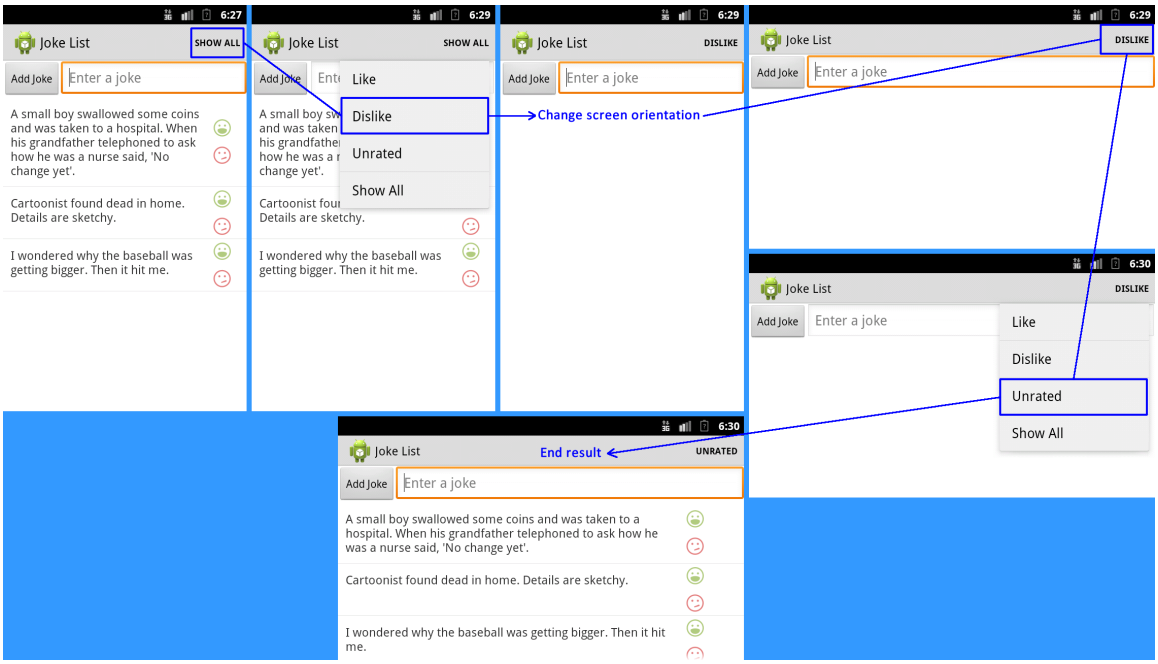

Figure 3.10: Lab 4: Joke List 3.0 instance state storyboard

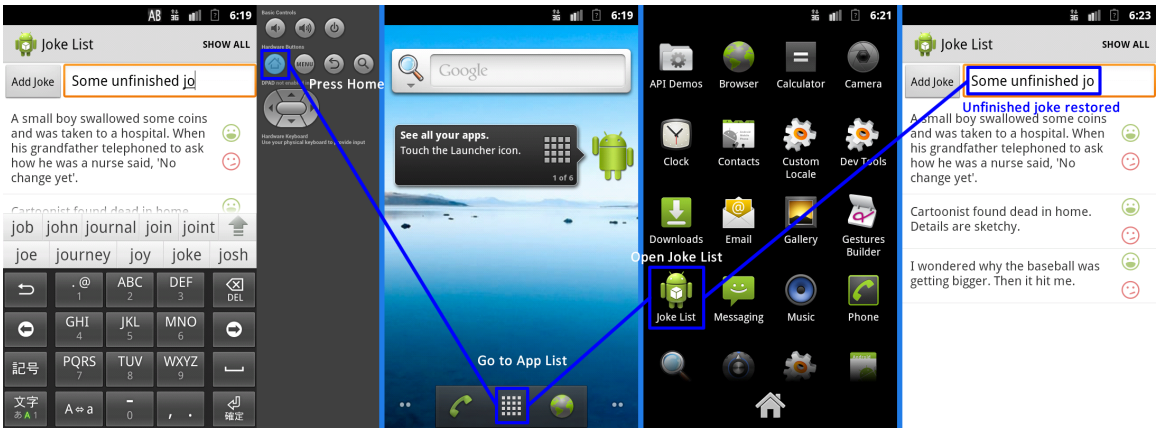

Figure 3.11: Lab 3: Joke List 3.0 shared preferences storyboard

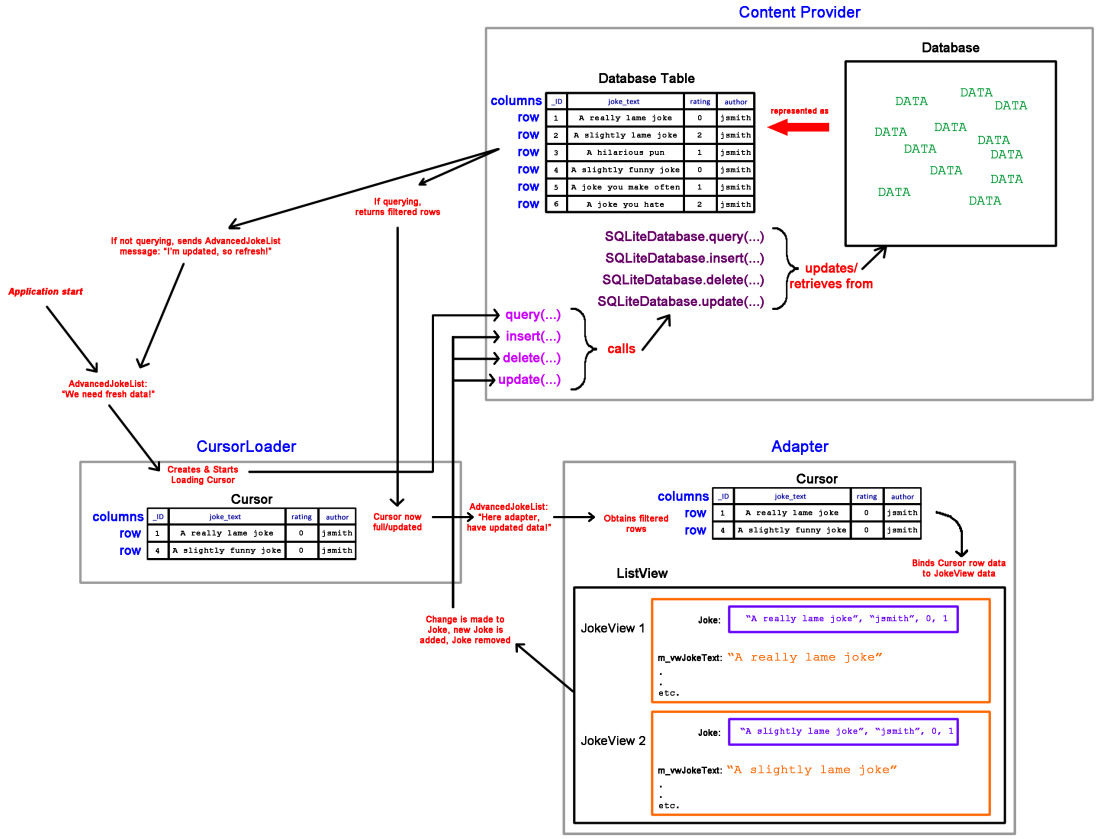

Figure 3.12: Lab 4: Data flow diagram

3.3.5 New Lab 5

JokeList 4.0 app: Upgrade Lab 3's app to work with HTTP connections.

Lab Guide: New[10]

Summary: This brand new lab was created and maintained by David Janzen partially through Course v3 as a spinoff of New Lab 3, and was meant to be a short side exercise to be worked on by students roughly halfway through the course curriculum near midterm time. The concepts discussed in this lab (Establishing HTTP connections, utilizing asynchronous tasks) were taken from Old Lab 3 in Course v2 and deemed important and relevant enough for this course, especially given the concepts covered in New Lab 7.

Lab & App Screenshots: The lab retools the menu from New Lab 3's app to allow for downloading and uploading of jokes from and to a web-hosted location using Android's asynchronous task implementation. Most of the changes are not visual, and instead modify the app's menus and underlying data handling to point to the specified web location as the primary joke list data source. Sample images of the app's download and upload features are shown in ***Figure 3.13*** and ***Figure 3.14*** below, respectively.

[10]**New Lab 5 guide** page:
`https://sites.google.com/site/androidappcoursev3/labs/lab5`

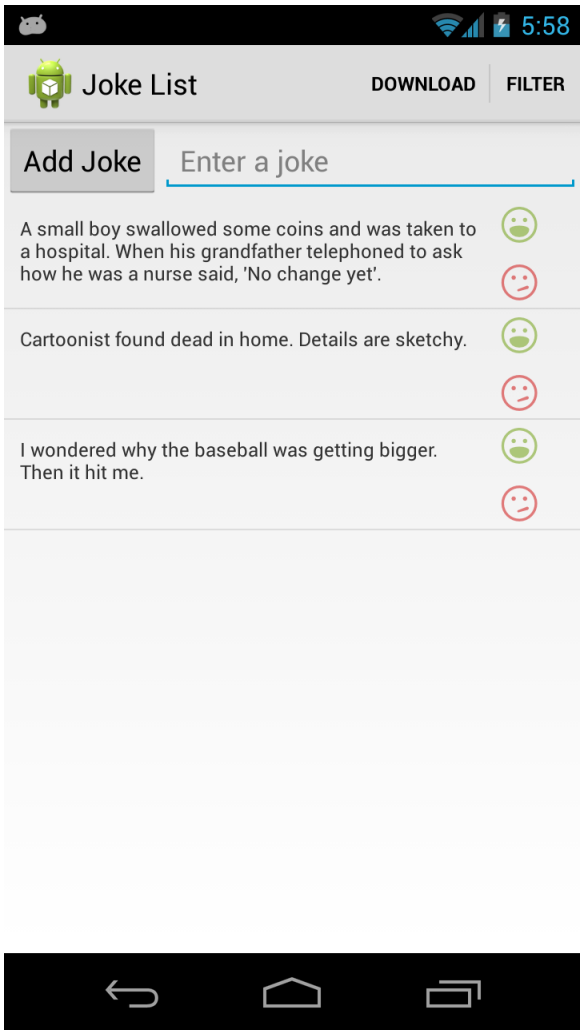

Figure 3.13: Lab 5: Joke List 4.0 joke download menu item

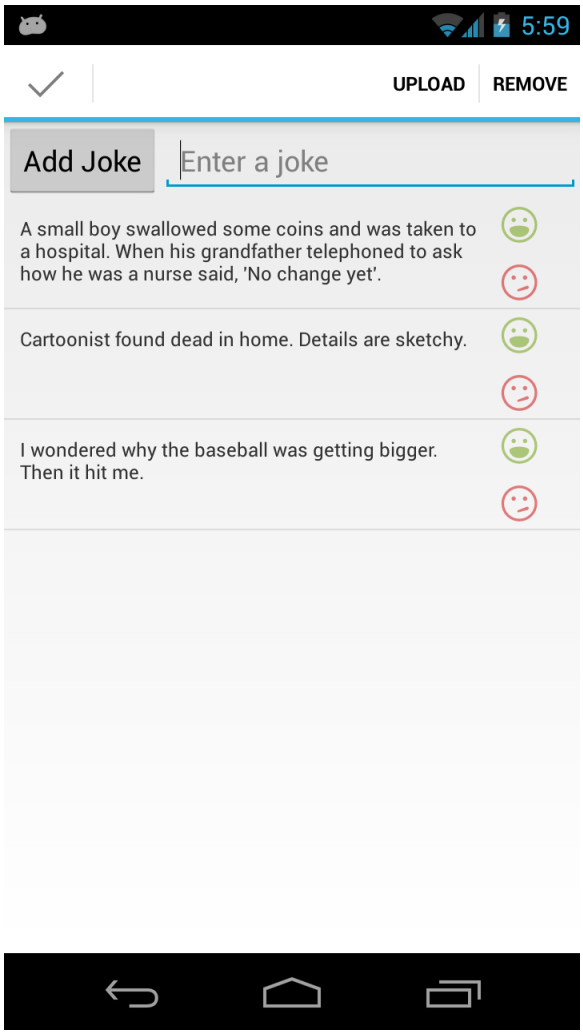

Figure 3.14: Lab 5: Joke List 4.0 joke upload context menu

3.3.6 New Lab 6

WalkAbout app: Create a GPS path tracking app with picture-taking and map-marking functionality.

Lab Guides: Old[11] vs New[12]

Change summary: Significantly reorganized and updated lab exercise; roughly to the same degree as New Lab 4 due to the sheer amount of content revisal and guide restructuring. Unlike New Lab 4, most changes present in New Lab 6 are due to API upgrades or deprecations; this exercise's direction was deemed appropriate for Course v3 and did not need a new heavy learning emphasis. The only concepts introduced to the new version of this lab (aside from backwards-compatibility support, API upgrades and deprecations) were additional discussions about device storage and utilizing the `Marker` class to add details at points on a Google Map. Only the general app content and instructional direction of Old Lab 5 were largely preserved in this new lab exercise.

Notable changes:

- New additional lab steps and guide sections for the more compatible Action Bar and Context Menu implementations provided by the third-party ActionBarSherlock support library, replacing the original Options and Context Menu implementations.

[11] **Old Lab 5 guide** page: `https://sites.google.com/site/androidappcourse/labs/lab-5`
[12] **New Lab 6 guide** page:
`https://sites.google.com/site/androidappcoursev3/labs/lab-6`

- Major refactor and reorganization of lab steps and guide sections for using the Google Maps v2 API, in place of the deprecated Google maps v1 API referenced in Old Lab 5.

- Additional inclusion of many new Android Maps app aspects in Maps API v2 that weren't present in (or have replaced deprecated aspects from) Maps API v1. The largest fundamental change is using Google `MapFragment` (specifically the Android support library's `SupportMapFragment` for this lab) class instances rendered over a `GoogleMap` object, instead of drawing maps using overlays like in Google Maps API v1. Other changes include the replacement of the deprecated `GeoPoint` class with the new `LatLng` class, utilizing the new `CameraUpdate` and `CameraUpdateFactory` classes for smooth map camera management and adjustment, and utilizing the new `Shape` interface to draw polylines instead of path overlays.

- Expanded sections to add more expository text on the Google Maps API (especially concerning API key retrieval and app inclusion), Google APIs console, and the Google Play and Android Support libraries to account for new Maps setup and app compatibility purposes.

- Expanded and updated lab step and guide section on using and setting up camera devices for picture taking, in particular to further describe camera image formats, virtual and physical device storage options, SD card storage and file storage complications.

- Added new lab step and guide section for using the `Marker` class to store data at map locations where pictures have been taken during map GPS recording sessions.

Lab & App Screenshots: The lab primarily consists of getting an app to properly utilize v2 of the Google Maps API, starting with a basic `GoogleMap` drawn inside the app (***Figure 3.15*** shown below). The lab then walks through drawing, saving and recording user-driven GPS paths (***Figure 3.16*** shown below), taking pictures through the app and storing them on the device (***Figure 3.17*** shown below), and finally using map `Markers` to indicate and provide extra data at locations where pictures were taken (***Figure 3.18*** shown below).

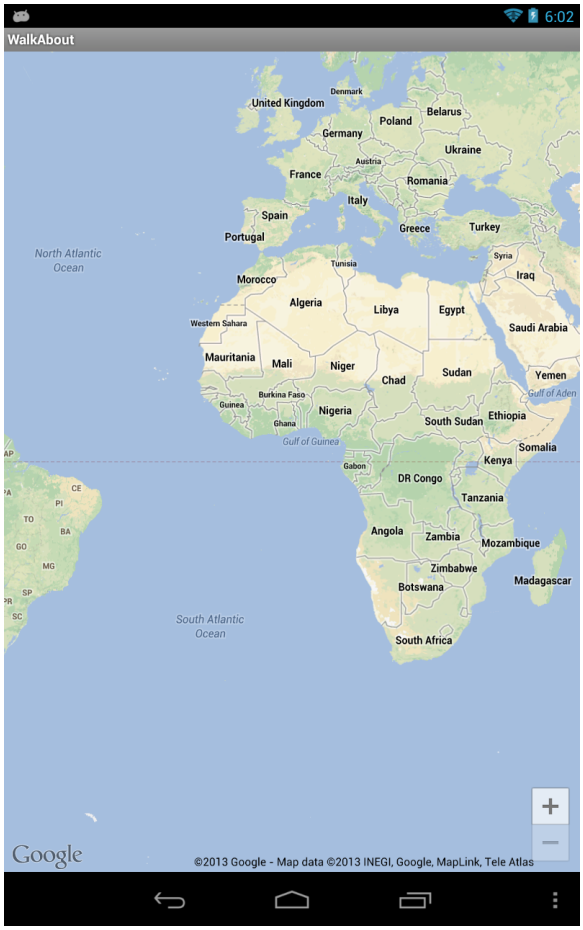

Figure 3.15: Lab 6: Walkabout basic Google Maps implementation

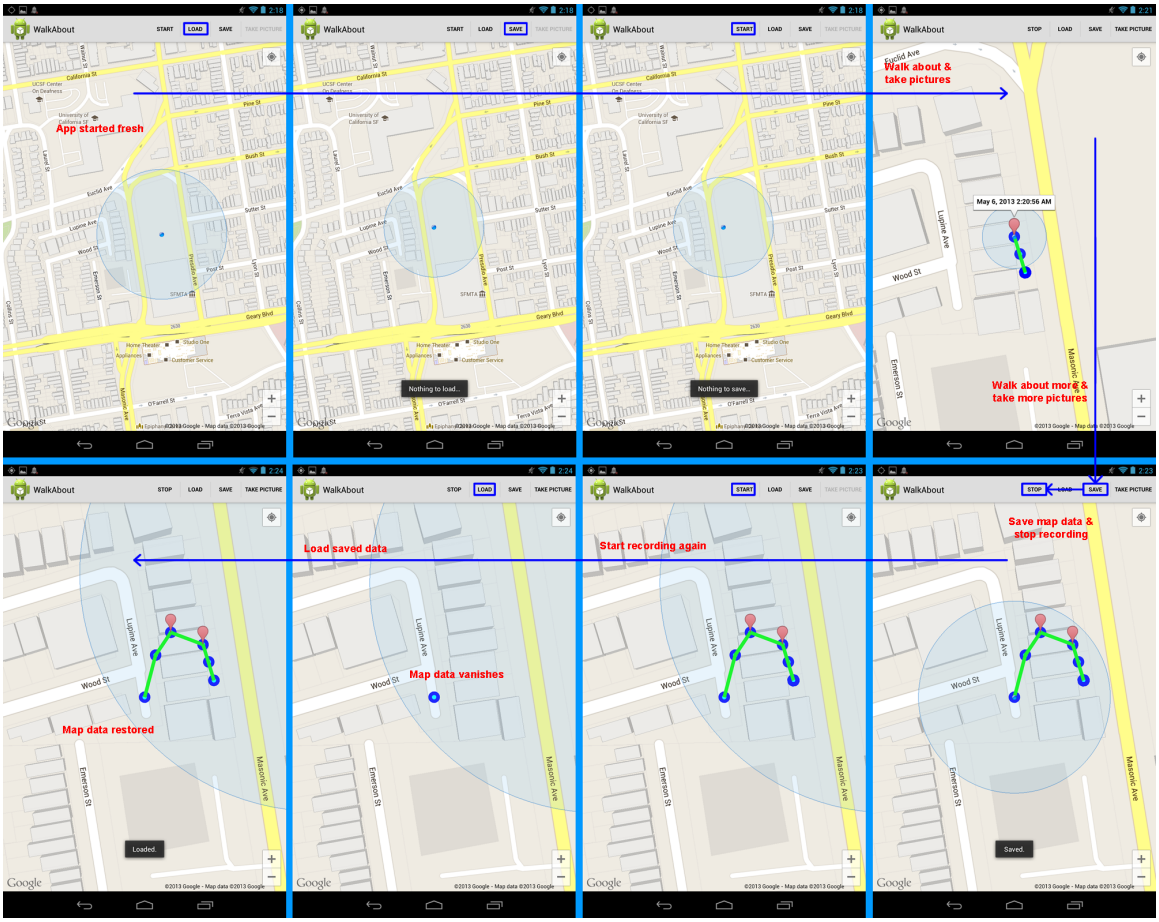

Figure 3.16: Lab 6: Walkabout GPS path save-load storyboard

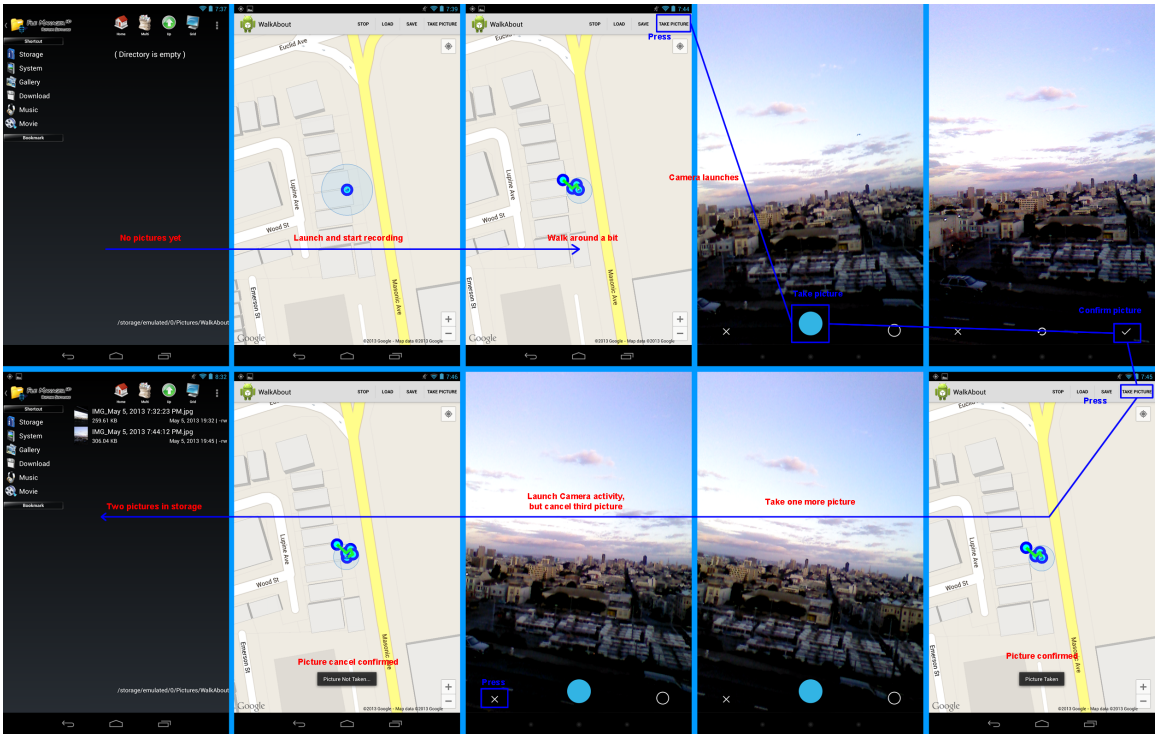

Figure 3.17: Lab 6: Walkabout GPS path & picture taking storyboard

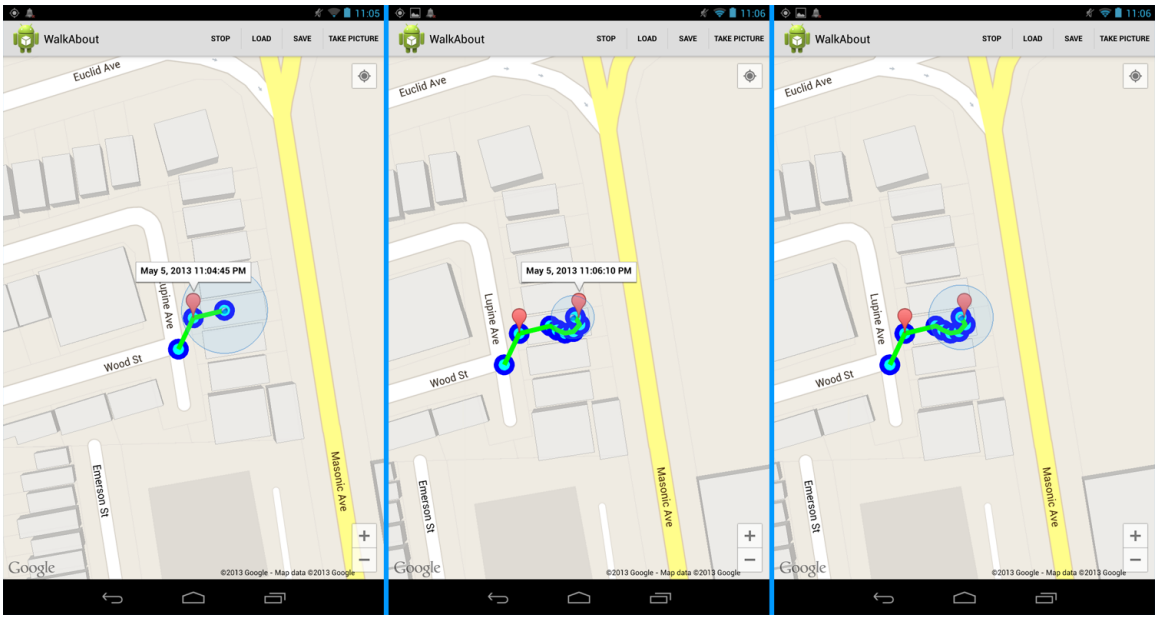

Figure 3.18: Lab 6: Walkabout GPS path markers

3.3.7 New Lab 7

AppRater app: Create an app with a list of other apps to be rated by users.

Lab Guides: Old[13] vs New[14]

Change summary: This lab was intentionally carried over from Course v2's version as a lighter exercise since students were mostly working on their final app projects at the same time. Old Lab 6 material that was already covered by other new labs was cut from New Lab 7 (specifically the exercise on database implementation). The biggest app change was relying on the `RatingBar` class for rating apps in the list instead of using a context menu with radio buttons. Otherwise, the app structure and emphasized learning material covered in the lab still generally follows that of Old Lab 6.

Notable changes:

- New additional lab steps and guide sections for the more compatible Action Bar and Context Menu implementations provided by the third-party ActionBarSherlock support library, replacing the original Options and Context Menu implementations.
- Removal of lab steps and guide sections covering database implementation, as these were sufficiently covered in New Lab 4. An implementation of the content provider and database needed for the exercise comes provided with this lab's skeleton project.

[13] **Old Lab 6 guide** page: `https://sites.google.com/site/androidappcourse/labs/lab-6`
[14] **New Lab 7 guide** page: `https://sites.google.com/site/androidappcoursev3/labs/lab-7`

- Removal of lab steps and guide sections covering rating an app in the list based on context menu options, replaced with lab step and guide section on using the `RatingBar` class for a more detailed standalone rating app.
- Updated lab step and guide section for starting app downloading service to use the `IntentService` class instead of the more basic `Service` class.
- Additional lab emphasis on Notification building and the `NotificationManager` and `PendingIntent` classes, given the advanced new notification bar in recent Android OS versions.
- Lab step and guide section on launching the Android Market/Google Play activity moved to the end of the lab.

Lab & App Screenshots: The lab primarily consists of setting up a list app so it downloads a list of app names and allows users to install and rate each app in the list from within the lab app. The initial list item layout was designed first (***Figure 3.19*** shown below). The lab then walks through getting, removing and notifying the user of apps in the list (***Figure 3.20*** shown below), and color-coding each app in the list based on its current install and `RatingBar` rating status (***Figure 3.21*** shown below).

Figure 3.19: Lab 7: AppRater list item XML layout

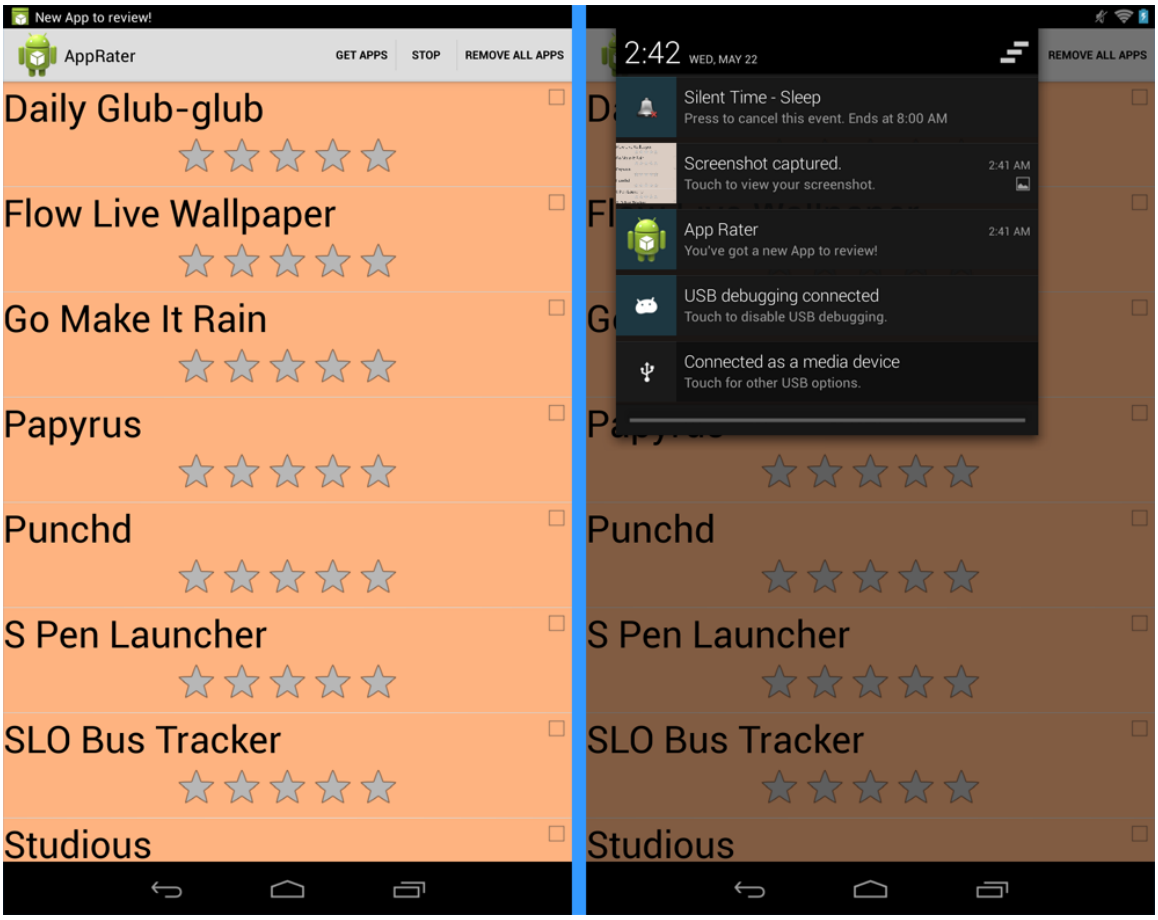

Figure 3.20: Lab 7: AppRater list and app download notification

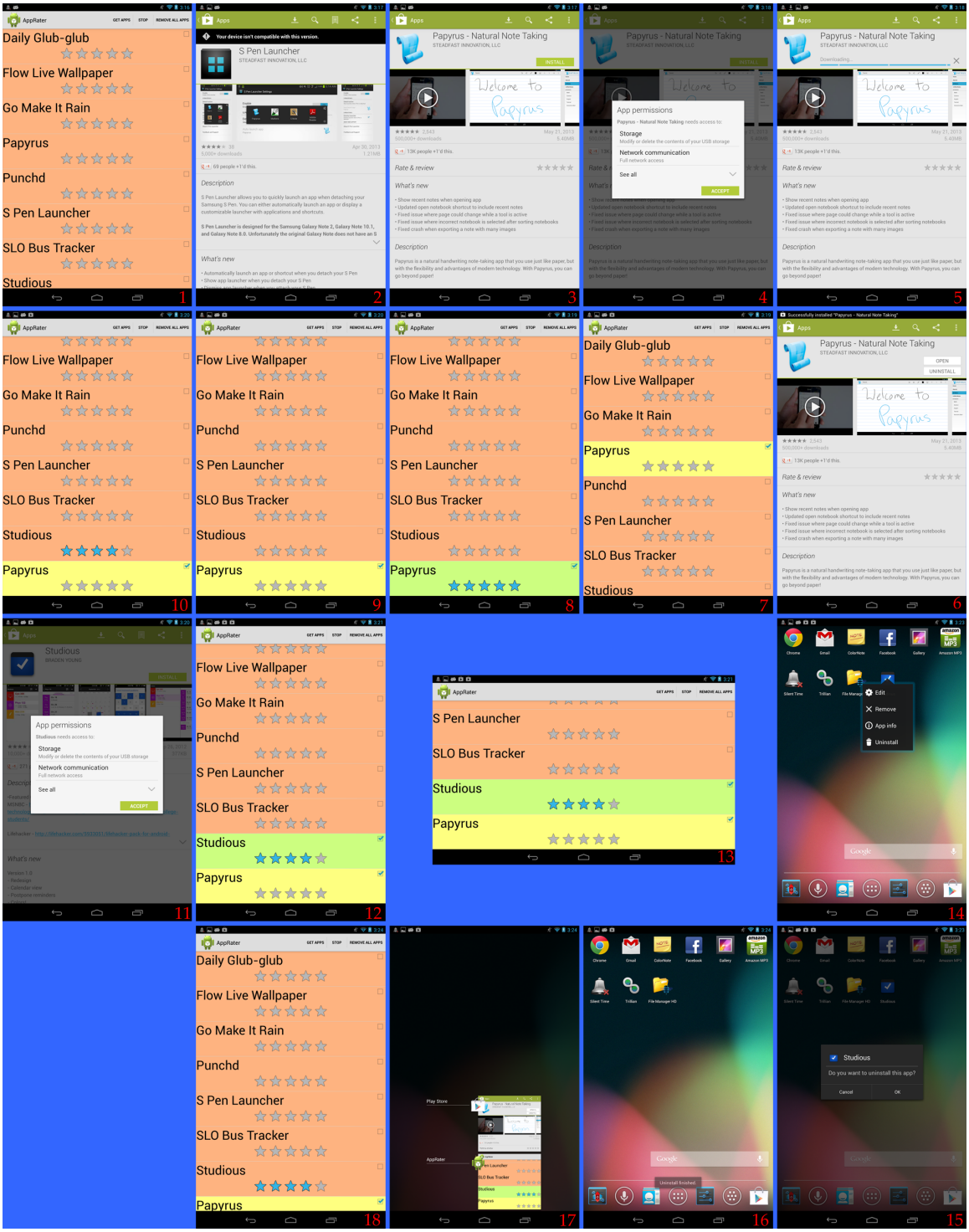

Figure 3.21: Lab 7: AppRater final app storyboard

3.3.8 Old Lab 7

Untitled app: Create an app with text messaging, and experimenting with various other physical Android device features.

Lab Guide: Old[15]

Summary: This lab was completely removed from Course v3, primarily due to general lack of depth and detail on the covered topics (text messaging and accelerometer). Instead, the portion of Old Lab 7 dealing with `PendingIntent`s was integrated into New Lab 7. Additionally, several concepts covered in Old Lab 7 were suggested to students as exploratory development exercises or possible inclusions in the Course v3 final project apps.

See the Old Lab 7 guide for images and more details on Old Lab 7.

[15] **Old Lab 7 guide** page: `https://sites.google.com/site/androidappcourse/labs/lab-7`

Chapter 4

EVALUATION

To properly evaluate the effectiveness of the new labs in Course v3, participating students were asked to complete surveys from SurveyMonkey [25] after working through each lab. This evaluation model was carried over from Course v1 and Course v2. Each survey URL was provided at the end of each hosted lab guide for easy and direct student participation after lab completion. Students were also asked to complete a final course survey near the end of course completion to gauge overall course effectiveness.

Additionally, in an attempt to measure the general degree of effort needed to convert Course v2 labs to Course v3 versions (and the general degree of change between versions), parallel lab key projects from old and new labs were quantitatively compared. Comparison measurements were calculated similarly to the API Differences Reports as weighted percentage changes across a multitude of statistical indices, with a focus on additions and removals.

Finally, for further lab effectiveness analysis, the major Course v2 and v3 lab apps were tested on a variety of Android devices running different versions of the Android OS/platform years after each course was completed. This brief test was executed to check for app errors or outdated/deprecated behavior resulting from the ongoing evolution of the Android API.

4.1 Survey Questions & Results Overview

Each survey consisted of several "rate the following aspects from 1-5" question sets, followed by several "essay box" open-ended questions. Each multiple choice question set posed a statement and offered **Strongly Disagree** (**1**), **Somewhat Disagree** (**2**), **Undecided** (**3**), **Somewhat Agree** (**4**) and **Strongly Agree** (**5**) as selectable ratings for assessing the statement. Additionally, each multiple choice question set provided an optional essay box below all statements in the set to allow for students that disagreed with any of the multiple choice statements to briefly indicate which ones and why. Each required essay box question prompted students to answer **'none'** or something similar if they had no outstanding or detailed answer to the question. Lastly, answers to all questions in each survey were required unless indicated below as optional.

Each new lab survey asked students the following questions pertaining to the lab:

1. Rate the statements below in regards to the lab as a whole:
   (a) (**Multiple choice**) I found the lab exercises challenging.
   (b) (**Multiple choice**) I found the lab exercises interesting.
   (c) (**Multiple choice**) It was clear to me what I was expected to do in the laboratory exercises.
   (d) (**Essay box, *optional***) If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

2. Rate the statements below in regards to the list of items in the 'Objectives' section of the lab:

   (a) (**Multiple choice**) I performed tasks that accomplished each of these objectives.

   (b) (**Multiple choice**) I gained enough knowledge about each of these objectives that I can comfortably perform tasks related to them independently.

   (c) (**Essay box, *optional***) If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

3. (**Essay box**) What did you like most about the lab?

4. (**Essay box**) What did you like least about the lab?

5. (**Essay box**) What part(s) of the lab did you find most important or interesting?

6. (**Essay box**) What could have been done better in the lab?

7. (**Essay box, *optional***) Any other comments.

Before continuing with the survey results, it is important to note two details. **Firstly**, the results of all surveys were *entirely* anonymous; all students completed each survey without providing any form of identification, in the hopes that students would be more honest with responses and ratings. This means that survey responses can only be tracked on a per-survey basis. It is therefore impossible to identify students based on their feedback across multiple surveys. As a result, these survey results are (and can only properly be) used purely for general evaluation and success of the labs and the overall course. **Secondly**, students were not *required* to complete each lab survey while taking the course; they were encouraged to complete surveys at the end of each lab, but full participation was not guaranteed. This resulted in a generally declining response rate throughout the duration of the course. Out of **33** total students enrolled

in Course v3, individual lab student survey response participation ranged from complete (33) to extremely partial (13). As such, surveys with lower participation rates not only contain more weighted responses (extreme/varied results) but also provide less overall and therefore weaker feedback for the respected lab.

With these two details in mind, survey results follow in the subsequent section.

4.2 Lab Survey Results & Analysis

All lab survey results are accompanied by results analysis. Some spelling or grammatical errors in student essay question answers were corrected. In the interest of saving space, not all essay question answers have been included; to estimate a well-rounded representation of all feedback for each essay question, generally more concise, detailed and critical student answers were chosen as objectively as possible based on lab difficulty and overall perceived student attitude towards the lab given all other answers in the survey.

4.2.1 Lab 1 Survey Results

Total survey responses: **33**

Survey response collection timeframe: **April 4-8, 2013**

1. Rate the statements below in regards to the lab as a whole:

Table 4.1: New Lab 1 Question 1 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I found the lab exercises challenging.	**18.18%** 6	**18.18%** 6	**9.09%** 3	**45.45%** 15	**9.09%** 3	33
I found the lab exercises interesting.	**0.00%** 0	**6.06%** 2	**0.00%** 0	**54.55%** 18	**39.39%** 13	33
It was clear to me what was expected to do in the laboratory exercises.	**0.00%** 0	**3.03%** 1	**6.06%** 2	**39.39%** 13	**51.52%** 17	33

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "The instructions made the lab straightforward and guided the user through the steps."
- "It wasn't challenging at all. But it's a good thing that it wasn't challenging, I needed an easing into everything."
- "I did not find this lab particularly interesting because I knew what the end

result was and it was not particularly interesting. However, I understand 'Hello World!' is a fundamental program and necessary to teaching."

- "Pretty standard intro lab, wouldn't make it harder."
- "Some of the steps assumed prior knowledge. While going on Google was relatively easy, a lot of stress came from feeling incompetent in 'simple' tasks."
- "I have already created an Android app that is on the market so this lab was a repeat of my previous experience with how to do all this. I wish I had this one to follow when I did it. Having everything in one spot makes it a lot better than finding everything everywhere."

2. Rate the statements below in regards to the list of items in the 'Objectives' section of the lab:

Table 4.2: New Lab 1 Question 2 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I performed tasks that accomplished each of these objectives.	**0.00%** 0	**0.00%** 0	**0.00%** 0	**24.24%** 8	**75.76%** 25	33
I gained enough knowledge about each of these objectives that I can comfortably perform tasks related to them independently.	**0.00%** 0	**3.03%** 1	**15.15%** 5	**39.39%** 13	**42.42%** 14	33

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "Some components were glossed over, it seemed. Additionally, there were a number of issues I had setting up the development environment which may make doing similar tasks difficult."
- "In the lab explanation, there was mention of dragging an editText object onto the Graphical Layout in Eclipse. After searching on both Eclipse on my desktop at home and Eclipse on the lab computers, I could not locate this object and ended up manually inserting it via text copied from the internet into the XML file."
- "I just have to go through and document the challenges I had."

3. What did you like most about the lab?

- "The lab identified the various files and responsibilities needed to create a full app."
- "What I liked most about the lab was how thoroughly it was written. Especially for an introductory lab to Android development, I found the additional explanations extremely useful."
- "I most liked the fact that we made a simple app, ported it to a mobile device and got it running. It's good to show a very simplified full process, from creation to publishing."
- "I liked the detailed explanation of all the files that Android generates."
- "It went through a lot of the basics of Android and provided a lot of links to various resources."
- "I liked that we're using Intents right off the bat, instead of just a generic hello world app."

4. What did you like least about the lab?

- "I had to google some things such as 'onClick' which might as well be included in the instructions."
- "There were a number of different issues I had setting up my development environment. Most of these center on the emulator not working (various components were missing that weren't immediately apparent or referenced anywhere I saw; for instance, ia32-libs is a dependency of the emulator), to Eclipse not showing the plugin options in all the appropriate screens, to various silent failures."
- "It was sometimes unclear when we were supposed to name a variable something specific, or whatever we wanted."
- "I somehow ended up with 2 XML files describing the Hello World activity, but only one was connected, so when I ended up editing the other one and no changes appeared, frustration ensued."
- "Sometimes the wording on certain directions was not 100% clear, but this helped with self learning to a degree."
- "The instructions for the external utilities (keystore, Eclipse) could have used more explanation or links. Mostly just stating that it is fine if you do not know about these things."
- "There was a lot of information covered and sometimes it was too much. If there is ever a part that is followed by 'more on this later' or something similar, save all the explanation until that point in time."
- "The uncertainty of 'If it doesn't work, just clean it!' seems odd, but sadly true."

5. What part(s) of the lab did you find most important or interesting?

- "Connecting the two screens/activities together."
- "I think that all of the additions we made to the initial HelloWorld app were

the most important and beneficial to understanding the Android development process."

- "The general overview of the different files/directories of the source was very helpful in understanding how the app worked, which I felt gave a fairly good grounding."
- "Overview of the files Android generates. I've done Android development before but I wasn't ever sure what all of them were."
- "The key signing which is still unclear to me, but that really should be researched on my own time."
- "Going through manipulating the XML files and setting up references to the strings."
- "The parts involving creating an Intent and storing and extracting information from a Bundle was the most important."
- "Learning how the XML and Java interact. Also, learning how to export your project so it can go up on the Google Play Store."

6. What could have been done better in the lab?

- "It was a really good first lab."
- "Ideally, there would be a troubleshooting guide which had fixes for a couple different issues. It took me a couple hours, in all, of searching for solutions to find the proper way to do it."
- "My only real comment is that since we didn't explicitly set IDs for the edittext and button in the EnterNameActivity, it took me a while to figure out why R.id.EditText1 was referencing the correct EditText. Other than that everything's fine."
- "More information about non-Android aspects of the project."
- "More coverage of the basic Android development tools like ADB, LogCat, etc.

Also, a little more guidance on keysigning would have been helpful."

- "A bit more detail towards the end. I appreciate the effort for trying to make me learn how to perform the necessary actions on my own."
- "I believe Eclipse has been updated since this tutorial. Some of the images do not match with the actual windows displayed."
- "Be more specific about what device settings to use for the AVD devices."

7. Any other comments:

- "This is NOT a short lab that can be completed in an hour, especially if you actually read all of your spec. I would have preferred more warning this lab would take hours. Seriously."
- "Very up to date! Followed most instructions exactly."
- "Overall, a good first lab, though perhaps a little more daunting that was originally implied."

4.2.1.1 Lab 1 Survey Results Analysis

This was a very important and foundational lab exercise, being the first in the course. At this point during the course, it was expected that students had little idea how novice or advanced their programming and engineering skills were in relation to the lab exercises (especially since no other Android courses were currently offered at Cal Poly, SLO). Combining this with the widely varying degrees of flexibility and progress for students in their college degree flowcharts at the time, this lab exercise and accompanying writeup were undoubtedly difficult to balance. This is confirmed by the varying degree of responses in the survey.

Survey results for questions 1 & 2 were unsurprising with little to no outliers, given the wide variety of student programming knowledge and experience. Some students found this introduction lab simple and easy, even citing prior Android development experience, while others indicated they experienced challenge stepping through the lab one step at a time. Overall, most students felt that Lab 1 was about right for an introduction lab, and were pleased that the lab guide included in-depth details covering a wide variety of Android development setup steps and components.

Survey results for questions 3-7 were also generally unsurprising. Aside from a decent number of varied comments on how the lab was too tough or too easy, most responses indicated that the lab was set up about right for an introduction lab, with many students figuring out issues on their own. However, several interesting survey responses stood out. To start, at least one response indicated that parts of the lab dealing with Android development setup and coding were already outdated, especially regarding a newer version of Eclipse. Furthermore, several students indicated a desire for a troubleshooting guide or more advanced Android topics such as the Android Debug Bridge (**ADB**)[1] and LogCat, the latter not being covered until Lab 2.

Overall the lab was mostly well-received, and students largely indicated an understanding of and appreciation for the materials covered. Some students were clearly hungry for more than a 'Hello World' app, although most appreciated the lab as more than just a simple one-screen 'Hello World' app.

[1] **Android Debug Bridge** page:
`https://developer.android.com/studio/command-line/adb.html`

4.2.2 Lab 2 Survey Results

Total survey responses: **29**

Survey response collection timeframe: **April 6-12, 2013**

1. Rate the statements below in regards to the lab as a whole:

Table 4.3: New Lab 2 Question 1 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I found the lab exercises challenging.	**0.00%** 0	**10.34%** 3	**17.24%** 5	**55.17%** 16	**17.24%** 5	29
I found the lab exercises interesting.	**0.00%** 0	**0.00%** 0	**0.00%** 0	**51.72%** 15	**48.28%** 14	29
It was clear to me what was expected to do in the laboratory exercises.	**0.00%** 0	**0.00%** 0	**6.90%** 2	**48.28%** 14	**44.83%** 13	29

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "I had issues getting the colors from colors.xml using Resources. The exact same code performed differently when copied to different methods in the project."
- "After knowing Eclipse and getting back into Java this lab was a lot easier."
- "When adding the listeners, it was not exactly clear that you needed to add the onKey listener to the editText. It would seem like you need to add it to the

view or the keyboard."

- "The exercises were still quite simple, and understandably so. We haven't gotten to any difficult aspects quite yet."

2. Rate the statements below in regards to the list of items in the 'Objectives' section of the lab:

Table 4.4: New Lab 2 Question 2 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I performed tasks that accomplished each of these objectives.	**0.00%** 0	**0.00%** 0	**0.00%** 0	**37.93%** 11	**62.07%** 18	29
I gained enough knowledge about each of these objectives that I can comfortably perform tasks related to them independently.	**0.00%** 0	**0.00%** 0	**10.34%** 3	**51.72%** 15	**37.93%** 11	29

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "If I have the lab instructions or access to Google, I could definitely do it."
- "Just have to document steps down for the nested layouts and listeners."
- "I will definitely use the lab as a reference, I couldn't memorize it all just from the lab exercise."

3. What did you like most about the lab?

- "Learning about layouts and declaring layouts dynamically at runtime and how to listen to different buttons, including hardware buttons."
- "Learning JUnit testing for Android."
- "Learning how to do everything by hand, no copy-paste."
- "Having the joke that I had written show up on the screen when I pressed the Enter/Return key on the Android device."
- "Builds nicely on the last lab."
- "I liked working with UI elements."

4. What did you like least about the lab?

- "Parts of it were frustrating because I have very little experience dealing with listeners."
- "It was boring. A joke list is not interesting. It also took a long time to complete, considering what the final result was."
- "It didn't seem like there was much of a use for dynamic layouts in this lab (except for the joke list view), it seems like it would have been a lot easier just to specify the add joke button and the joke edit text in the XML layout."
- "Outdated. Key listeners for virtual/soft keyboards should not be used."
- "The ambiguity about adding ENTER or CENTER key listeners – are these physical keys only? Should they work with virtual keyboards?"

5. What part(s) of the lab did you find most important or interesting?

- "I think it was good to learn about event listeners and the two main methods of implementing one."
- "I have experience in the listener pattern and basic UI design, so the lab wasn't challenging to me. However, it was useful to learn the API."

- "How to integrate keyboard/hardware buttons."
- "Setting up logcat for future use."
- "I liked defining our own color.xml. I also liked the challenge of trying to figure out methods on my own, like trying to add the key listeners."
- "The JUnit testing part was pretty interesting and definitely seems like something I will use later on."
- "Understanding how to use an Android Test Project, and starting early with good testing practices."
- "Understanding how to interface the activity with all the resources (e.g. how a resource creates a static member in the R class) as that was unfamiliar to me, but very useful."

6. What could have been done better in the lab?

- "There could have been less hand holding in this lab. I understand it is only the second lab, but for those that have written Java before (which should be a prerequisite for this course), many parts were unnecessarily cumbersome. I also would have preferred more freedom in the design of the app. I realize that can be difficult to automatically test, but why not have us write our own test cases or just demo the app to the professor in class?"
- "Dealing with the keyboard popups got annoying because I wasn't 100% sure what was causing them."
- "Possibly provide a little more explanation of static vs. dynamic layouts and why you may choose to use one over the other, e.g. limitations of static layouts, or could this app be accomplished using only static layouts, etc."
- "Nothing, I think it did a good job."
- "Device compatibility issues accounted for several hours of wasted time."

- "A hint on how to initialize LayoutParams in section 4.1.2. This is the only section I had trouble with, although it did force me to read the documentation."

7. Any other comments:

- "I liked the additional comments that guided us through the lab and explained what different components were for, or alternate ways to implement code."
- "I had no familiarity with hiding the keyboard and thus had no idea where that code you gave us was supposed to go."
- "Good lab. The lack of technical issues that arose in the first was very pleasant."

4.2.2.1 Lab 2 Survey Results Analysis

As the second lab exercise, it was assumed that students were generally more settled into Java programming and Android development. In particular, students were tasked with and could focus more on app development exercises rather than all of the Android app development setup steps covered in Lab 1. Despite the slightly stronger familiarity and understanding of Android concepts, however, student responses still varied greatly.

Survey results for questions 1 & 2 were generally expected, with a moderate increase in perceived lab difficulty and exercise interest compared to Lab 1. Students were overall slightly less certain about following and absorbing Lab 2's objectives compared to Lab 1's, and this was backed up by several responses indicating that the Android concepts covered in the lab were not a straightforward exercise that could be easily committed to memory after performing once.

Survey results for questions 3-7 were slightly surprising, particularly opinions that several parts of or the lab as a whole were outdated, boring or impractical. However, given the constant evolution of the Android API and devices, as well as Course v3's attempt to recycle core concepts from Course v2, this is understandable. Most positives with the lab concerned the introduction of dynamic layouts, as well as learning general Android development tools such as JUnit tests, Android Test Projects and LogCat. Students generally enjoyed this lab's expansion on Lab 1's brief introduction to managing app resources and layouts, which would form the foundation for the remaining labs. At least one comment alluded to frustration dealing with device compatibility issues, which would be covered and emphasized starting in Lab 3.

Overall, the lab was mostly well-received. Compared to Lab 1's responses, student reports of the lab's difficulty were less varied, with some students claiming the lab was too simple, but most results indicated a generally fair lab exercise outside of encountering a blocker or two. However, it is clear after analyzing survey results that this lab felt too plain, restrictive and outdated. Several student responses proved how behind Android lab resources were, stating that the inclusion of exercises in hooking up virtual device keyboards and buttons on older devices within the lab felt less relevant and more hurtful than helpful when it came to lab progress. Furthermore, several parts of the JUnit setup were pointlessly frustrating and restricting, particularly the arbitrary JUnit test regarding text font size (carried over from the same lab from Course v2) and the general feeling of restriction the required JUnit test-passing system brought about for students. Lastly, the lab's overall theme of building a joke list felt bland to some students.

4.2.3 Lab 3 Survey Results

Total survey responses: **18**

Survey response collection timeframe: **April 15-22, 2013**

1. Rate the statements below in regards to the lab as a whole:

Table 4.5: New Lab 3 Question 1 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I found the lab exercises challenging.	**0.00%** 0	**0.00%** 0	**0.00%** 0	**33.33%** 6	**66.67%** 12	18
I found the lab exercises interesting.	**0.00%** 0	**0.00%** 0	**0.00%** 0	**61.11%** 11	**38.89%** 7	18
It was clear to me what was expected to do in the laboratory exercises.	**11.11%** 2	**33.33%** 6	**11.11%** 2	**33.33%** 6	**11.11%** 2	18

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "Sometimes it was uncertain whether I had to implement interfaces or methods and when I was supposed to do them. The wording for many sections needs massive improvement."
- "Most of it was pretty good, but I think too much was expected for us to figure out on our own without guidance."

- "Implementing the long click ActionMode.Callback was difficult. I found the provided reference misleading because it used OnLongClickListener() rather than OnItemLongClickListener(). The wording of section 2.2.2 was confusing when I was trying to figure out which class to implement stuff in."
- "Some of the steps were not clear on what you were supposed to do because of how we were supposed to do things differently from how the Android documents indicated. For example, the menu and submenu initialization in onCreateOptionsMenu() was different from what the Android guide showed, and I had no idea how we were supposed to use those submenu and menu variables."
- "The entire menu section felt vague."
- "It was clear what the expected outcomes were but it was challenging figuring out how to get there sometimes."

2. Rate the statements below in regards to the list of items in the 'Objectives' section of the lab:

Table 4.6: New Lab 3 Question 2 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I performed tasks that accomplished each of these objectives.	**0.00%** 0	**0.00%** 0	**0.00%** 0	**44.44%** 8	**55.56%** 10	18
I gained enough knowledge about each of these objectives that I can comfortably perform tasks related to them independently.	**0.00%** 0	**0.00%** 0	**16.67%** 3	**50.00%** 9	**33.33%** 6	18

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "It was really interesting and cool but I can't say I could do it again from memory."
- "I would definitely need references to implement any of this again."

3. What did you like most about the lab?

- "Seeing nice UIs appear on my old phone!"
- "Using ActionBarSherlock to subvert Android API restrictions."
- "It taught me about the action bar and menu items as well as adapter views

and how to apply them to an adapter."

- "I feel like I am getting a much better understanding of the Android environment."
- "Learning about the different UI elements; Contextual Action Mode is cool."
- "Adding radio buttons with icons."

4. What did you like least about the lab?

- "Some of the ambiguity and legacy text in the lab."
- "It wasn't very exciting and it was very restrictive."
- "ActionBarSherlock has poor documentation."
- "Too many 'figure it out' parts in the guide."
- "Some parts were very annoying. Layouts needed to be a little different than you said in a few cases so that they actually worked. Also, the ListView caching bug."

5. What part(s) of the lab did you find most important or interesting?

- "Learning to use ActionBarSherlock, and being able to use that to modify other components in the view."
- "Actionbarsherlock was useful. And while I didn't like implementing it, knowing how to set a listener to individual units in a ListView will also be useful later."
- "Binding backend data to frontend views."
- "The parts where the ListView handles long clicks, and removing selected jokes."
- "Continuing to learn how the Java code and XML interact."
- "Learning how to create menus and using adapter views."

6. What could have been done better in the lab?

- "Removing the ambiguities and old parts of the lab would make it more clear."

- "Better explanation of how filtering works."
- "Overall, it's not that bad. However, small details such as typos and tests not always working made the lab take much longer than it should have."
- "Clearer instructions on parts that were not exactly clear, especially when the info is different from Google's Android guide."
- "Redo the whole context menu section, it is really vague and hard to understand what to do."
- "A bit more guidance."
- "Make it shorter."

7. Any other comments:

- "I hope Lab 4 doesn't take as long."

4.2.3.1 Lab 3 Survey Results Analysis

Lab 3 results indicated that this lab was a clear step up in difficulty relative to the first two lab exercises, for a variety of reasons. This was expected given the moderate degree of change the Course v2 version of this lab went through when becoming its Course v3 offering version. Notably, this was the first lab to introduce and strongly emphasize backwards-compatibility across a multitude of Android platforms using ActionBarSherlock.

Survey results for questions 1 & 2 reflected the step up in difficulty for this lab. However, according to several student responses, not all of the difficulty was due to the advancement of Android topics, but primarily caused by a less clear lab written guide; in particular, several lab guide sections were misleading or unclear, or even outdated relative to the official Android documentation at the time. Overall, survey results

indicated that the lab was a moderate to difficult challenge and boasted a variety of useful and challenging topics. Very few students indicated that this was a lab they could mostly do from memory.

Survey results for questions 3-7 were generally more positive than results for questions 1 & 2. Very few students gave significantly negative responses to these questions, mainly listing the lab as minorly annoying or lacking in detail, with the largest complaint coming back to the written guide's vagueness, lack of detail or outdated content. Students largely indicated appreciation for ActionBarSherlock and learning about Android backwards-compatibility support, citing recognition of their importance despite the rather simplistic lab steps and implementation. However, as will be revealed in later lab survey results, the potency of ActionBarSherlock would largely wear off after this lab was completed. This implies that the idea of app compatibility was considered more of an oversight by students, and possibly assumed to be a slick modern solution at the time without much other context for students to draw from (prior app compatibility issues).

Overall, while a longer and tougher lab compared to Lab 2, students were more receptive and positively critical of Lab 3. The average response indicated the feeling that the lab, despite presenting challenges and struggles, was still overall useful and a beneficial experience to work through. However, the lab was noted to cover outdated content and possess unclear guide information. It is clear that students wanted more out of this lab; it would not be unreasonable to either expand this lab in future iterations, with emphasis on polishing the lab guide students follow while attempting the exercise, or outright replace it with a different context than a joke list.

4.2.4 Lab 4 Survey Results

Total survey responses: **20**

Survey response collection timeframe: **April 29 - May 6, 2013**

1. Rate the statements below in regards to the lab as a whole:

Table 4.7: New Lab 4 Question 1 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I found the lab exercises challenging.	**0.00%** 0	**0.00%** 0	**0.00%** 0	**30.00%** 6	**70.00%** 14	20
I found the lab exercises interesting.	**0.00%** 0	**10.00%** 2	**10.00%** 2	**50.00%** 10	**30.00%** 6	20
It was clear to me what was expected to do in the laboratory exercises.	**25.00%** 5	**15.00%** 3	**15.00%** 3	**30.00%** 6	**15.00%** 3	20

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "This one was really hard to follow. Databases on Android are rather complex."
- "I found the description of URIs confusing. There were many points in the lab where one vital piece of information was missing that would make it possible to finish the lab."
- "The hairy nature of how to use databases in Android, and the sheer number

of classes needed, made it a bit difficult to keep track of why each piece was needed, which in turn made it difficult to understand what I had to do."

- "I get that you had to relay a lot of information, but that was just rough. Too much going on."
- "This lab didn't seem quite as interesting as other labs."

2. Rate the statements below in regards to the list of items in the 'Objectives' section of the lab:

Table 4.8: New Lab 4 Question 2 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I performed tasks that accomplished each of these objectives.	**0.00%** 0	**0.00%** 0	**5.00%** 1	**45.00%** 9	**50.00%** 10	20
I gained enough knowledge about each of these objectives that I can comfortably perform tasks related to them independently.	**10.00%** 2	**20.00%** 4	**30.00%** 6	**20.00%** 4	**20.00%** 4	20

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "Can comfortably perform tasks, but I still don't have a firm understanding of how to properly handle databases."
- "I fully completed the lab 3 times. I never managed to get the entire thing

working properly, and I have no idea why."

- "I will be able to use Bundles and SharedPreferences easily later on, but using a SQLite database will require me to look up either examples, diagrams, or tutorials. It will be much easier than it would have been going in with no introduction. It seems difficult to be able to teach database access in Android to the point where it is intuitive, but this was a good primer."
- "It seemed like a lot of the provided code was glossed over."
- "This lab was more of an exposure lab than a learning lab. I feel like I am aware of when this is good to use but would not be comfortable doing it on my own."
- "This lab was incredibly confusing."

3. What did you like most about the lab?

- "I love databases, so knowing them for Android is relative to my interests."
- "I liked the introduction to the database idea for persistence and how it interacts with the app."
- "I liked the story boards."
- "I feel like the quality of the Joke List app increased dramatically."
- "It did a good job explaining the many different steps that needed to be taken in order to set up a SQLite database in an Android app."
- "Seeing the various options developers have for storing data, and understanding what each one of them is used for."
- "Learning how to use URIs by ourselves."
- "Being able to store the Filter text."

4. What did you like least about the lab?

- "Very complex, too much handholding led to not understanding each of the components well enough."

- "Writing so much code before testing the application."
- "Reading blocks of code given to us to copy and paste. Just make it part of the stub."
- "That there was no way to test individual lab sections after Section 2."
- "The SQLite components were at times difficult to understand, and at some points I had to take what I was doing on faith of the lab."
- "The lab forced you to do a lot of coding without running your code, rather than develop incrementally. This made it hard to find bugs if your code didn't work right away at the end of the lab."
- "The vagueness on how to do certain items in the lab and how it is assumed we think like the author."
- "I don't think it was necessary to teach us about a SQLite database implementation in Android, since it isn't necessary in many apps. Most will have no reason for anything more than a key/value store."
- "How complicated it was to get such a simple database working."

5. What part(s) of the lab did you find most important or interesting?

- "Using URIs to communicate with a content provider."
- "Dealing with SQLite databases and ContentProviders."
- "All the parts that involved a SQLite database, which include the CRUD (Create, Read, Update, Delete) operations."
- "Understanding how SQLite databases work on Android is definitely the most important aspect of the lab."
- "I thought the sections on maintaining application state were really important."
- "The content provider and adapters."

6. What could have been done better in the lab?

- "Less hand holding so that we would need to know exactly what each component of the database interaction does."
- "The SQLite database diagram given could be more detailed. Introduce the diagram early, and then at each step of implementation refer to the diagram to show what part we are implementing."
- "I liked the flowchart you had at the end of the Lab writeup, but it would be great if you could also add a diagram detailing how all the classes we used in this lab work together to set up and use a SQLite database (like the one you made in class)."
- "Clearer instructions. There were some errors in the lab writeup - including Shift+F11 should be Ctrl+F11 to rotate in emulator."
- "Have more opportunities to test the application earlier in the lab."
- "In the future, when you say implement an interface, say whether we need to put 'implements <interface name>' or if you simply want us to implement the methods that are listed in <interface name>."
- "This is probably because of the recent update, but there is some critical conflicting information in the lab. The two example URIs don't match the URI pattern that the content provider is searching for, for example. Things like this are hard to notice for people using URIs for the first time."
- "More complete explanations, less copying and pasting, more checkpoints."
- "Perhaps the tricky web of classes needed for a database could have been explained better. There were also problems with the provided stub file."
- "Some parts could have been explained more thoroughly, in particular the cursors/cursor adapter."

7. Any other comments:

- "Exorbitantly long lab."
- "Please match your own code style - it's maddeningly annoying when you have conflicting styles in provided code."

4.2.4.1 Lab 4 Survey Results Analysis

Primarily covering persistence and SQLite databases, Lab 4 was a drastic step up from Lab 3 in terms of complexity and difficulty. It was decided before presenting this lab to students that a more explanatory approach would be attempted in this lab due to the advanced nature of its concepts and previously sparser exercises in the old lab version. While students generally recognized and understood how difficult this lab's subjects were based on responses, this lab was much less well-received than expected, primarily due to the attempt to change the lab's teaching angle while still preserving as much of Lab 4 from Course v2 in this new lab version for Course v3.

Survey results for questions 1 & 2 indicated great confusion and difficulty due to the complex nature of SQLite databases and persistence in general. Furthermore, students generally disliked the more explanation-heavy nature of the lab guide, with some responses indicating the feeling that the topic had to be forced to such a degree as an important aspect of Android app development. However, despite these comments students still responded somewhat positively on average for question 2, with a slight overall average of a "Somewhat Agree" response regarding levels of comfort with the lab tasks. This may be partly due to the more expository nature of the lab; however, students still had to complete and demonstrate the working lab app to the professor for credit so this may not have greatly influenced question 2 responses. It

was very clear to students after completing the lab that databases and persistence were important for Android apps, despite the confusion and frustration felt by many students (including one student who worked through the entire lab three times from scratch, yet still didn't get a fully working app at the end).

Survey results for questions 3-7 were mostly negative, with moderately more responses to questions 4 & 6 than questions 3 & 5. Students generally disliked the structure of the lab, with the biggest complaint regarding the lack of opportunities to test their app-in-progress throughout the lab. Some students felt that learning the intricacies of a database was forced, unfun and not very necessary; it can be generally inferred from these comments that students got frustrated with this lab to the point where they would much rather either be given a database to work with as a part of a larger lab with an explanation on the side, or be thrust into a more hands-on database learning exercise where they would be more on their own with significantly less exposition and hand-holding. Ironically, a later lab in the course (New Lab 7) would give students a working database. Students also pointed out various outdated or incorrect Android development information in the lab exercise and guide, further illustrating the struggle to constantly keep up with rapidly changing Android API and development standards.

Overall, while some students praised the evolution of the Joke List app or acknowledged how impressive the app had grown over the course of several labs, the general message concludes that the lab was too difficult, forced, expository or confusing. The collective student response to this lab was easily the least positive of all individual lab survey responses. This is not surprising, as the lab was not only anticipated to be one of the toughest labs for students to work through, but was also difficult to modify for the Course v3 curriculum by the lab's primary author. The chosen angle for the lab was too thinly split between exposition and exercise; thus, a safer angle was

taken for the written guide, which was largely reduced to explanations for the SQLite database aspects to avoid potential risk of confusing and misdirecting students even further. Furthermore, the notable lack of unit test opportunities in the lab exercise was frustrating to students. The included diagrams and informational tidbits (notably the URI section) were not enough to make up for the lack of student-driven learning exercises. Finally, students likely grew tired of evolving a single app concept (the same Joke List) over several labs, especially when combined with the complexity and exhaustive nature of this lab and this being the third simultaneous Joke List app exercise. In the future, a lab covering the ins and outs of databases should probably be carefully considered and even more carefully constructed, likely with a different context than a Joke List, to better instruct and enable students.

4.2.5 Lab 5 Survey Results

Total survey responses: **17**

Survey response collection timeframe: **April 29 - May 6, 2013**

1. Rate the statements below in regards to the lab as a whole:

Table 4.9: New Lab 5 Question 1 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I found the lab exercises challenging.	**5.88%** 1	**47.06%** 8	**11.76%** 2	**35.29%** 6	**0.00%** 0	17
I found the lab exercises interesting.	**0.00%** 0	**0.00%** 0	**0.00%** 0	**52.94%** 9	**47.06%** 8	17
It was clear to me what was expected to do in the laboratory exercises.	**0.00%** 0	**11.76%** 2	**0.00%** 0	**35.29%** 6	**52.94%** 9	17

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "The directions were for a context menu."
- "I didn't find the lab too challenging just because it was so short and the instructions were pretty straightforward."
- "URLs could have been explained more in class."
- "I don't think it was meant to be an especially challenging lab."

2. Rate the statements below in regards to the list of items in the 'Objectives' section of the lab:

Table 4.10: New Lab 5 Question 2 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I performed tasks that accomplished each of these objectives.	**0.00%** 0	**0.00%** 0	**0.00%** 0	**35.29%** 6	**64.71%** 11	17
I gained enough knowledge about each of these objectives that I can comfortably perform tasks related to them independently.	**0.00%** 0	**0.00%** 0	**5.88%** 1	**47.06%** 8	**47.06%** 8	17

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "It was nice having the ability to try to figure things out for ourselves but having something to fall back on if we got stuck (i.e. the previous lab and upload joke code)."

3. What did you like most about the lab?

- "Actually doing something that is used in the real world."
- "Learning how downloads work."
- "The length of the lab and how useful the information was within it."

- "It was only one topic and covered well. Short and to the point."
- "The simplicity (especially after Lab 4)."
- "The more hands-off approach, which gets students to learn the material on their own."
- "The fact that I wasn't up until 6am doing it."

4. What did you like least about the lab?

- "That the app crashed if you did not have the correct permissions in the Android manifest file."
- "Nothing, well done."
- "The fact that we had to use outdated instructions from a previous course lab."
- "Short, not enough interaction."
- "This lab referenced an older version of the lab and the instructions in the older lab didn't match up or were outdated."
- "Unclear how to implement download and upload menus."

5. What part(s) of the lab did you find most important or interesting?

- "Learning how to use AsyncTask."
- "The part of the lab that demonstrated how to upload and download jokes."
- "Being able to interact with a server and make POST/GET requests."
- "Connecting to the PHP backend."
- "It was cool to send web calls from within the app."
- "AsyncTasks. Awesome."

6. What could have been done better in the lab?

- "Include a more difficult concept in the lab, such as JSON deserialization or sharing Jokes through email."

- "Clearer instructions."
- "Mention that you can type the urls into a web browser to see data."
- "More stuff to do."
- "A better explanation of the progress dialog would have been nice."
- "Removed the suggestion to use InputStreamReader as it is not needed."
- "I feel that if Lab 5 is going to reference an older lab, the instructions in the referenced lab should reflect what Lab 5 is asking you to do."
- "Incorporate all the instructions into one web page, and update them. For instance, we don't have an onCreateContextMenu() function anymore. Not a huge deal, but annoying."

7. Any other comments:

- "Would liked to have seen where the downloaded data was coming from, or how to make a source to download data from."
- "Nothing, well done."

4.2.5.1 Lab 5 Survey Results Analysis

This lab was primarily a partial carry-over of one of Course v2's earlier labs (including the lab guide instructions). The lab was fully authored and administered by Dr. David Janzen, and meant to be a shorter exercise (since it was assigned closer to midterm exam time) on what were deemed to be important Android development topics regarding HTTP connections and asynchronous tasks.

Survey results for questions 1 & 2 were mostly positive as anticipated, although students appropriately did not find the lab challenging nor expected challenge out of the lab due to its simplicity and short length, especially compared to New Lab 4. The

most important feedback from students was likely the sense of relief that this lab was not building on a previous lab of an identical context, and thus felt more comfortable to work on without fear of getting very deep into the exercise and retracing steps too far back.

Survey results for questions 3-7 were mostly positive, with most negative responses highlighting the out-of-context experience dealing with outdated lab guide instructions. Students generally favored the lab and felt that the less expository angle of the exercise was very appropriate. Students thought the lab exercise was refreshing and more relevant due to covering the broader and more modern topics of server interaction and asynchronous tasks.

Overall, Lab 5 seemed extremely favored and appropriate for students at this point in the course. Several students actually felt the lab exercises and guide were *too* simple, seemingly wanting more to work on given the Android development concepts covered in the lab. Given the student feedback, a standalone and more in-depth lab revolving around this lab's concepts is a viable prospect for future iterations of this course, especially when factoring in student concerns over the exhausting and stale nature encountered in the previous three labs regarding building on the same app over and over. However, it should be noted that PHP is a slightly old web development technology package at the time of writing this paper; a cloud solution implementation should likely be considered for expanded lab exercise interaction.

4.2.6 Lab 6 Survey Results

Total survey responses: **13**

Survey response collection timeframe: **May 6-20, 2013**

1. Rate the statements below in regards to the lab as a whole:

Table 4.11: New Lab 6 Question 1 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I found the lab exercises challenging.	**0.00%** 0	**0.00%** 0	**7.69%** 1	**46.15%** 6	**46.15%** 6	13
I found the lab exercises interesting.	**0.00%** 0	**7.69%** 1	**7.69%** 1	**23.08%** 3	**61.54%** 8	13
It was clear to me what was expected to do in the laboratory exercises.	**7.69%** 1	**7.69%** 1	**7.69%** 1	**53.85%** 7	**23.08%** 3	13

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "I didn't find this lab as compelling as others."
- "I had major problems with saving the image format."

2. Rate the statements below in regards to the list of items in the 'Objectives' section of the lab:

Table 4.12: New Lab 6 Question 2 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I performed tasks that accomplished each of these objectives.	**0.00%** 0	**0.00%** 0	**0.00%** 0	**46.15%** 6	**53.85%** 7	13
I gained enough knowledge about each of these objectives that I can comfortably perform tasks related to them independently.	**0.00%** 0	**0.00%** 0	**7.69%** 1	**61.54%** 8	**30.77%** 4	13

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- ***No optional responses given.***

3. What did you like most about the lab?

- "Learning how to interact with Google APIs."
- "The content of the lab was interesting, and doing something besides another joke list iteration was refreshing."
- "Learning how to use the device camera."
- "I liked learning how to manipulate files in general."

- "Using Google Services."
- "Cool app that actually does something interesting. I could see how one might expand on this and actually publish it."

4. What did you like least about the lab?

- "The string parsing for saving/loading. And the fact that the map took forever to work because of an Android update that disabled a setting that was needed."
- "Registering to use Google Maps."
- "The GPS integration is really shoddy...it's probably not the lab's fault, but maybe you could mention it won't quite lock onto your position as well as the Google Maps app will."
- "Fails to account for the case where your app is killed off by the OS and gets restarted. You need to implement a workaround to get the desired functionality."
- "Getting the compass to show up."

5. What part(s) of the lab did you find most important or interesting?

- "How easy Google Maps is to use in Android app development."
- "How to get Google Map API authentication without mistakes."
- "Naming, saving and loading the camera image files."
- "Integrating the camera without implementing it ourselves was interesting."
- "The string parsing for saving/loading."
- "Working with the Google API. I've used it on webapps and it is powerful."
- "Maps and GPS integration."
- "The location services."

6. What could have been done better in the lab?

- "The setup instructions and storage section could have used more detail."
- "We already know Java. Making us do string parsing for 'x,x;' is a beginner exercise. It's not taxing or challenging, it's annoying."
- "Give us the code for parsing the files to be loaded. I spent a lot of time debugging non-Android code."
- "I wanted a few more 'on your own' parts."
- "Account for the case where the OS kills off your app in the background and is created from scratch instead of with onRestart()."
- "Clarification of the picture saving would have helped."
- "Better wording to account for crashing app due to limited memory. Also better wording in regard to the toggle start/stop button."

7. Any other comments:

- "This lab took me about 8 hours, but 2 were spent dealing with Android.R. In the end, I had to delete the project and start a new one and copy all the source code."
- "Unclear sometimes when to start or stop following the Google instructions vs the lab ones. Maybe list out each section in the Google instructions we should complete?"

4.2.6.1 Lab 6 Survey Results Analysis

With its wide range of objectives and largely new content, this lab was inherently designed as one of the largest in Course v2 and the same holds for this version in Course v3; this lab covered the most new development content (integrating maps and location services into an application, utilizing the Android device's camera and file

system, capturing and drawing map details on-screen) of all Course v3 labs. The general subject of the lab, the Google Maps API, was considered a strong and interesting Android development topic to teach and it was expected that students would appreciate such an importantly-perceived and refreshing topic. The lab was also completely independent from the Joke List series of labs (New Labs 2-5) and was largely revamped for the newest Google Maps API version at the time, hopefully providing both excitement and a breath of fresh air for students. It is safe to say that this was the Course v3 lab with the highest expectations for positive results.

Survey results for questions 1 & 2 largely aligned with these expectations. Almost all students indicated that the lab was interesting and useful, with 100% of responses believing that the lab objectives were reasonably covered throughout the exercise. Students also largely believed that the development processes covered in the labs were more useful, intuitive and memorable than those in other labs. One student even believed that this app could very well be legitimately published to the Google Play Store if it were expanded upon and polished more. Only one or two students indicated slight disagreement in answers to these questions, indicating a solid Android app development learning experience.

Survey results for questions 3-7 were not as overwhelmingly positive, mostly due to struggles students encountered with certain parts of the lab, but also partially caused by Android device system hitches such as new Android API updates or changes to how GPS works midway through the lab. The most commonly reported issues dealt with apps being killed off by the Android OS on test devices, and the perception of performing menial data parsing tasks this deep into an advanced engineering course (despite several responses indicating the opposite). However, little to no overall negative lab feedback indicates a solid lab experience. Students enjoyed learning the

details of Google Maps and the accompanying API, even those who had prior experience with it in other web contexts. Furthermore, while some students felt like there should have been more exploratory exercises in the lab, most enjoyed utilizing the Camera class without having to delve into the complexity of manually opening, controlling and closing the Android device camera. Few to no complaints from students regarding the lab guide (particularly regarding how outdated it might have been, or the nonstandard methods various Android devices would invoke regarding storing files in memory and file system navigation) round out an overall positive experience working through this lab.

Overall, given the effort spent to upgrade the lab, how far into the course the lab was assigned to students (after they had settled into Android app development) and the lab's general complexity and length, this was debatably the most positively-received lab in Course v3 given collective individual survey feedback. Students felt ready for this lab and largely enjoyed creating and learning about app features with Google Maps integration. The Android concepts and exercises covered in this lab are undoubtedly solid, and can safely carry through to any future course app iterations with the app and exercise largely intact (especially since Maps API v2 is still the latest version at the time of writing this paper). However, it may be beneficial to present this lab to students earlier, especially given the ideas and opportunities it may give students for their final course app projects.

4.2.7 Lab 7 Survey Results

Total survey responses: **13**

Survey response collection timeframe: **May 27 - June 10, 2013**

1. Rate the statements below in regards to the lab as a whole:

Table 4.13: New Lab 7 Question 1 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I found the lab exercises challenging.	**0.00%** 0	**0.00%** 0	**0.00%** 0	**53.85%** 7	**46.15%** 6	13
I found the lab exercises interesting.	**0.00%** 0	**7.69%** 1	**7.69%** 1	**53.85%** 7	**30.77%** 4	13
It was clear to me what was expected to do in the laboratory exercises.	**0.00%** 0	**15.38%** 2	**15.38%** 2	**53.85%** 7	**15.38%** 2	13

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "Not enough information about which variables to use in methods, had to guess on some."

2. Rate the statements below in regards to the list of items in the 'Objectives' section of the lab:

Table 4.14: New Lab 7 Question 2 Survey Responses

	Strongly Disagree	**Somewhat Disagree**	**Undecided**	**Somewhat Agree**	**Strongly Agree**	Total
I performed tasks that accomplished each of these objectives.	**0.00%** 0	**0.00%** 0	**7.69%** 1	**46.15%** 6	**46.15%** 6	13
I gained enough knowledge about each of these objectives that I can comfortably perform tasks related to them independently.	**0.00%** 0	**7.69%** 1	**7.69%** 1	**61.54%** 8	**23.08%** 3	13

If you disagree with any of the previous statements, please indicate which ones and briefly explain why.

- "Databases in Android have a lot of moving parts, and unfortunately without spending a lot of time on them it's difficult to grasp what each one does. This may require more in-class time in the future."

3. What did you like most about the lab?

- "Learning how easy Notifications are to make!"
- "Interesting and real world implementation."
- "Learning about Services."

- "Another (more relevant, I think) look at the SQLite stuff was priceless. Implementation of notifications was super useful as well."
- "Learning about Notifications and learning more about Intents."
- "Interacting with the Google Play Store."

4. What did you like least about the lab?

- "Some of the instructions were vague, which lead me in the wrong direction."
- "Somewhat blindly following the instructions when it came to using the database portion of the app."
- "Had to rely on StackOverflow for Intent help."
- "Working with the ListView is still a pain."
- "Trying to figure out why NotificationCompat.Builder.Build() wasn't working (it had to do with not having the latest versions of ActionBarSherlock and Android's support library)."
- "Working more with a database."
- "The Service part was slow."
- "Many sections were confusing, lab took a really long time to complete, too many sink-or-swim sections."

5. What part(s) of the lab did you find most important or interesting?

- "Learning how to use BroadcastReceiver and how to create Notifications."
- "Launching an Intent to install an app from the Google Play Store."
- "The Notifications section."
- "Services seem like a very powerful tool."
- "Notifications."

6. What could have been done better in the lab?

- "Be more abstract about what needs to be done with databases. It will force students to learn how to use them properly."
- "None. This was really well done."
- "Be more explicit instead of relying on prior lab knowledge."
- "Generally, the NotificationManager section was very vague. Adding that it must be obtained through a getSystemService() call and mentioning where to get the Notification ID would help."
- "Better explanation on the final portion of the lab."
- "We already learned databases, and I'm still not convinced they're really all that necessary in most apps. Why do we need to do more stuff with them?"
- "Clean up instructions, fewer sections that simply say "Go!" with no help."
- "Nothing, I think the lab is fine."

7. Any other comments:

- "Not sure what I was supposed to use that MARKET_REQUEST_CODE for... oh well."
- "Thanks for a great course! I thoroughly enjoyed it and feel prepared enough to code my own app."

4.2.7.1 Lab 7 Survey Results Analysis

As the final lab exercise in the course, Lab 7 took a slightly more instructional approach, especially since students were now working on team-based final course app projects and did not need to be stuck investing so much of their time on learning more fundamental Android concepts from the ground up. This lab strove to offer more insight on widely-used interactive Android app development components such

as Services, BroadcastReceivers and Notifications, and how to best utilize them (possibly in students' final course app projects). As such, students were largely not left to the task of setting up core lab components such as the database (which was provided wholly to students), and instead mostly left to figure out steps in each lab section. This was a direct inverse to Lab 4's approach: this final lab assumed and respected that students had matured in their knowledge of Android and their ability to research encountered issues for solutions largely on their own. This angle was generally well-received by students, who mostly identified and respected the lab's intentions.

Survey results for questions 1 & 2 were unsurprisingly largely positive. The few optional student responses addressed only minor problems encountered during the lab, with one response echoing concern over the complexity of databases generally shared by most students since Lab 4. While the lab was generally considered difficult by students, the perceived difficulty likely came from the much less expository nature of the lab exercise which largely required students to investigate encountered development problems on their own.

Survey results for questions 3-7 largely reflected the expected level of confusion and problems encountered by students while searching out solutions themselves. Only a few students indicated issues with sections of the written guide being misleading, most instead claiming the guide was too vague at times. The language used in most students' negative concerns indicated specific frustration with certain Android app components, with students listing trouble with certain API classes such as Intent, ListView and Service. Several of these concerns listed not only the problem encountered, but also the solution each student took to fix it, indicating that students were starting to assume more ownership over their work and encountered issues. On the positive side, students largely appreciated the relevance and real-world application

of the primary Android app components covered in the lab, especially Notifications, Services (and their relation to the IntentService class) and interacting more with the Google Play Store. There were also slightly more responses overall indicating the lab was good as-is compared to feedback for previous labs.

There are several intriguing takeaways from this lab's feedback. The first is the general lack of mention of the app itself by students. Students mostly if not exclusively commented on the key Android programming aspects covered in the lab and not the AppRater app itself, possibly indicating that students were becoming less attached to the lab exercises and apps themselves and more invested in learning the core app development concepts and utilities they could take away from the course. The RatingBar class, despite its limitations, was barely mentioned. It is also possible that this might be influenced by students' current course workload, as they were generally more invested in their team-based final course app project and thus felt less attachment to the lab and its resulting app. The second was the database-specific feedback; alongside several additional and unsurprising negative statements about the frustrations and continued use & emphasis of databases in this lab, several student responses indicated the opposite: the belief that continuing to look at databases was important, and that the lab should have been more abstract about databases to force students to learn them properly. The mixed feedback regarding databases in Lab 7 is very different from the overall negative Lab 4 feedback.

Overall, it is clear that the maturity of the students taking the course outweighed the problems and concerns they encountered with the lab. Thus, students generally appreciated the exercises covered in this lab more than those covered in previous lab exercises. The level of abstraction this lab embraced, coupled with the timing of the lab assignment coinciding with the much larger final course app project, felt appro-

priate based on student feedback. Despite the specific lack of feedback on the lab's app itself, little to no negative overall lab feedback indicates that the exercise was still appropriate and relevant for students. Future iterations of this lab likely should not change much (unless the lab was moved to and introduced earlier in the course), although movement away from databases and ListView is worth considering due to how widely they have already been used in the other course labs.

4.3 Final Course Survey Results

In addition to the lab surveys, an overall final course survey was sent out and completed by students at the end of the course. The questions for this survey were different from those presented to students in the lab surveys. Most questions asked students how comfortable they were with and interested in Android app development after going through the course curriculum, with several questions asking for student opinions on the quality and usefulness of the labs and suggested improvements for the course overall. Unlike all lab surveys, all students were required to complete the final course survey before finishing the course.

Final survey questions and answer options are listed below, accompanied by student survey results. This survey was taken around the time final course projects were due; note that students in Course v3 worked with students in a concurrent Graphic Communications course on their final team-based app projects.

Total survey responses: **26**

Survey response collection timeframe: **April 6-14, 2013**

1. How comfortable do you feel overall with Android development after completing the lab exercises?

Table 4.15: Final Course Survey Question 1 Responses

Answer Choices	**Responses**
Very comfortable	**30.77%** (8)
Comfortable	**57.69%** (15)
Neutral	**11.54%** (3)
Uncomfortable	**0.00%** (0)
Very uncomfortable	**0.00%** (0)
Total	**26**

2. How comfortable did you feel following the online lab writeups to complete the required tasks for each lab exercise?

Table 4.16: Final Course Survey Question 2 Responses

Answer Choices	**Responses**
Very comfortable	**15.38%** (4)
Comfortable	**53.85%** (14)
Neutral	**19.23%** (5)
Uncomfortable	**7.69%** (2)
Very uncomfortable	**3.85%** (1)
Total	**26**

3. How comfortable were you developing your own Android application project after working through the lab exercises?

Table 4.17: Final Course Survey Question 3 Responses

Answer Choices	Responses
Very comfortable	**38.46%** (10)
Comfortable	**53.85%** (14)
Neutral	**7.69%** (2)
Uncomfortable	**0.00%** (0)
Very uncomfortable	**0.00%** (0)
Total	**26**

4. How do you feel the course labs compare with the official Android development resources (documentation, references and guides)?

Table 4.18: Final Course Survey Question 4 Responses

Answer Choices	Responses
Course labs are much higher quality than the official Android resources	**15.38%** (4)
Course labs are somewhat higher quality than the official Android resources	**26.92%** (7)
Course labs are about the same quality as the official Android resources	**50.00%** (13)
Course labs are somewhat lower quality than the official Android resources	**7.69%** (2)
Course labs are much lower quality than the official Android resources	**0.00%** (0)
Total	**26**

5. Rate the course labs in terms of usefulness.

Table 4.19: Final Course Survey Question 5 Responses

	Not Useful	**Slightly Useful**	**Somewhat Useful**	**Very Useful**	**Extremely Useful**	Total
Lab 1: Hello World	**0.00%** 0	**19.23%** 5	**26.92%** 7	**26.92%** 7	**26.92%** 7	26
Lab 2: Joke List (views, listeners, dynamic layout)	**0.00%** 0	**3.85%** 1	**15.38%** 4	**42.31%** 11	**38.46%** 10	26
Lab 3: Joke List 2.0 (custom views, adapters, menus)	**0.00%** 0	**0.00%** 0	**0.00%** 0	**53.85%** 14	**46.15%** 12	26
Lab 4: Joke List 3.0 (content providers, databases)	**0.00%** 0	**7.69%** 2	**7.69%** 2	**23.08%** 6	**61.54%** 16	26
Lab 5: Joke List 4.0 (networking, AsyncTask)	**0.00%** 0	**3.85%** 1	**7.69%** 2	**38.46%** 10	**50.00%** 13	26
Lab 6: WalkAbout (maps, GPS)	**7.69%** 2	**7.69%** 2	**26.92%** 7	**23.08%** 6	**34.62%** 9	26
Lab 7: App Rater (services, broadcast receivers)	**3.85%** 1	**7.69%** 2	**19.23%** 5	**46.15%** 12	**23.08%** 6	26

6. How likely are you to continue with Android development after the course ends?

Table 4.20: Final Course Survey Question 6 Responses

Answer Choices	**Responses**
Very likely	**65.38%** (17)
Likely	**26.92%** (7)
Maybe	**7.69%** (2)
Unlikely	**0.00%** (0)
Very unlikely	**0.00%** (0)
Total	**26**

7. What additional Android content would you have liked to see covered in the lab exercises, or in a future new lab exercise? ***(Optional)***

- "I would like to see displaying images in a gridview covered in future exercises. Also more about server code, perhaps even writing your own. I would also like to see TDD, where students develop Android code using TDD."
- "More hardware-intensive labs that deal with getting data from the sensors of the device would be cool."
- "I'd like to see a larger focus on making sure that students learn how to use databases on Android devices. I'd also like to see some of the labs devoted as milestones towards students individual projects, so as to ensure they are working their way towards the end goal. Lastly, I would LOVE to see a lab devoted to understanding how drawables work, taking into account the mdpi, hdpi, xhdpi resolutions, as well as the differences between small, normal, large, etc. devices. This lab would ideally be done with the designers in the Graphic Communication class."
- "More on animation and sound."

- "I think that covering more UI design concepts specifically for Android would have been neat."
- "Database interactions, cloud resources."
- "More general UI concepts which are very common to all applications, such as true custom views (i.e. use of custom views in XML, custom attributes), styling features, etc."
- "Multi-touch gestures and how to define custom listeners. I don't think an entire lab would be needed for this, but maybe mention some references in lecture or put them on the PolyLearn site. I would have been interested in having a lab where the goal was to create an app that could work on both phone and tablet. I think that might help clear up some of my issues with Fragments and how to use the different image sizes."
- "Accelerometers, multitouch gestures."
- "How to enable swiping between screens."
- "I would have loved to see more labs based around game design."
- "A lab that has us write our own automated tests."

8. Please leave any additional comments about Android, the course or the labs. ***(Optional)***

- "Good course. Thanks!"
- "The labs were informative, but frustrating at times when steps were left out."
- "Before this course I would have never seen myself becoming an Android developer, let alone a game developer. However, having the ability to work a project of my choosing, and on a platform where ideas come to life very quickly, changed my mind drastically. Partnering with a graphic designer was a phenomenal experience, and has given me an immense respect for graphic designers and their abilities. I'm very glad to see this course become a full-fledged Android learning

experience. The labs themselves are helpful for getting started, but from my perspective they were a little too directed, telling students exactly what to do. This helped in some regards, but made it easier to just follow orders and not particularly learn the assignments. I'd like to see this course get 'harder' in the sense that the labs become more difficult, but the students learn the material from top to bottom. More examples and up-to-date information and resources would be a great supplement to difficult labs."

- "The labs were informative, but frustrating at times when steps were left out."
- "Labs need a major wording rework. Really difficult to figure out what they wanted from us on each lab."
- "Overall, this was a very fun course. It was an excellent experience, and I'm thrilled I got to take it. It tackles a huge area (no one could learn Android development fully in one quarter), and does it well. However, I felt that the combination of labs and the final course app project resulted in a lower quality final app than would otherwise be developed. I would choose to focus more heavily on the app, maybe having more milestones, and less on the labs."
- "Lab 4 was extremely frustrating to complete. I'm still not sure what I did wrong with it. Other than that, my main complaint with the labs is I wasn't always sure what file of the project I was supposed to be editing at any given time, especially when the class names were similar. Keep the ActionBarSherlock stuff, it was really useful. While I appreciated that we had the loaned Nexus One devices, the touch screen on them is really difficult to work with."
- "I thought the lecture went too fast for me to understand it."
- "I really enjoyed making the final project and working with the graphic designers."
- "Great class and the labs are very well documented. Though, some were a bit too long in my opinion."

4.3.1 Final Survey Results Analysis

In general, overall course survey results were extremely positive. The full maturation of students' programming skills and Android app development knowledge is apparent based on the feedback given. Students clearly indicated appreciation and respect for the course materials and lab exercises, and felt very confident about continuing with Android development after the conclusion of the course.

Survey results for question 1 indicated strong to above-average comfort with the topic of Android application development by the end of the course. Notably, not a single response indicated even mild discomfort working with Android app development, with only 3 of the 26 responses claiming a neutral feeling towards Android development.

Survey results for question 2 were slightly less positive compared to question 1's results. 3 of the 26 responses indicated discomfort when following the lab writeups, while 5 others felt neutral about following them. Over half of survey participants felt at least above-average comfort with the lab writeups. Overall the responses to this question were more positive than expected, given the somewhat high frequency of prior negative lab-specific responses indicating that writeups were too vague, misleading, uninformative or incorrect. This could be due to students feeling less pressure from the labs towards the end of the course, or maturation and realization of Android development as a hard-to-hit moving target (possibly leading to students giving more of a benefit of the doubt to the lab writeups).

Survey results for question 3 stood out as overwhelmingly positive, with only 2 of 26 responses indicating neutral feelings about the final team-based app project, and no negative responses. This is likely primarily due to students enjoying more freedom

with designing and working on their own apps after working entirely on self-contained lab exercises for much of the course, with students taking information and experiences from the latter and applying them to the former. These results also imply general satisfaction with and realization of value in the structure of the lab exercises, which gave students a solid development foundation earlier in the course for working on their own Android apps later in the course.

Survey results for question 4 were largely neutral; students generally felt that the course labs were overall slightly better in quality than official Google-maintained Android development resources. 2 of 26 responders believed the course labs were slightly lower in quality compared to official resources, while 11 of 26 believed the opposite. This range of feedback is not unexpected, as the lab exercises did not attempt to reinvent the Android app development materials and concepts for the Course v3 offering; the labs occasionally referred to official Android resources in the writeups, and the writeups could not all be updated with every change made to the official Android API or SDK. Because of this, some information in the written lab guides may have become outdated (even as soon as several weeks or even days after any given lab was finalized and assigned to students). This question's feedback is difficult to properly assess due to these factors, although the overall above-average quality rating of the labs relative to the official Android resources indicates that at their core, the written labs are still great centralized references for students that provide a satisfactory context for learning.

Survey results for question 5 were overall very positive yet somewhat surprising, particularly the extremely high usefulness score earned by Lab 4 and somewhat average usefulness score for Lab 6. Despite all of the frustrated comments made by students after working on Lab 4, this lab netted the highest number of Extremely Useful votes

of all of the labs. Lab 6, however, was expected to highly intrigue students and open their eyes up to the powerful Google Maps feature (a widely used feature for GPS navigation by itself). However, Lab 6 scored among the lowest in usefulness across all labs, likely due to the complexity of integrating Google Maps into an application and the narrow course opportunity to make use of it (due to students taking it as the course's penultimate lab). The overall highest rated lab individually, Lab 3, scored only positive to very positive usefulness ratings. This is somewhat surprising given that Lab 3 was one of the more altered and diverse lab exercises compared to the Course v2 version, covered a highly sought-after topic (custom views and app UI design) and introduced backwards-compatibility details to students in the course; the higher ratings may also reflect students' desire for more UI design coverage. Despite Lab 1's lowest overall rating by students in this survey question, its simplicity and coverage of introductory and setup material, which may also easily be covered by other websites or online guides discussing Android development, validates its low usefulness score.

Survey results for question 6 were outstandingly positive. Surprisingly, not a single student indicated that they would be unlikely to stop with Android app development after the course ended. This overall response is a clear indicator that students not only felt extremely educated in and prepared for the subject of Android app development throughout the course, but were inspired and passionate enough to continue exploring the subject afterwards. This is also possibly due to a realization of the importance and growth of Android development in the technology market in the form of personal or professional mobile development opportunities for their careers.

Survey answers to question 7 reflect students' desires for learning about a wider variety of Android development components and content. This is understandable due

to the generally self-contained nature of the lab exercises, particularly labs 2-5 which generally built on the same app over time, and the generally small difference in content and approach between Course v2 and Course v3. Students particularly wanted to see more lab coverage on mobile UI design, as well as device hardware and sensor utilization (accelerometer, touch interactivity, etc.). While students were encouraged to explore other possible Android app features when working on their team-based final course app projects, some important student feedback (including a comment from question 8) indicated wishes for more time on those final course projects, possibly even at the expense of several lab exercises throughout the course. This is valuable and very sensible feedback for future iterations of this course. Also, students once again pointed out in this survey that more emphasis on and freedom in working with unit tests would be helpful; if desired for future course iterations, this would be important to begin integrating no later than halfway through the course. Since Labs 6 and 7 were devoid of unit tests and are encountered much later in the course, either of those labs would be an ideal choice for allowing students to completely write their own unit tests.

Finally, survey answers to question 8 overall demonstrate the value and enjoyment students found in taking the course, and this is especially reflected in the outstandingly positive answers to this question submitted by several students. Negative feedback given for this question regarding the labs was understandable given the constant evolution of Android development over time (with Android API changes rolled out by Google even during the duration of the course), as well as the fact that not every single aspect of each lab exercise could be reasonably detailed and maintained in the lab guides during the course. The length of and material covered in each lab in the course was a guess based on estimated student knowledge progression on the subject of Android development at that point in the course, mixed with prior knowledge of general student expectations from previous course offerings that used largely similar

labs. This estimation was largely realized and accepted by students at the end of the course.

While it is nearly impossible to design lab exercises that all students can easily follow and grow with, especially due to the wide range of student developer experience and familiarity with Android/mobile development in general, the final overall course survey indicates that Course v3 was very satisfactory and valuable to the large majority of students.

4.4 Course Lab Keys

For each lab in all course offerings (v1-v3), a programming solution project (**lab key** project) was provided by the primary lab Android project and written guide author to the course instructor. These lab keys could be run as full app solutions by the course instructor through the Eclipse IDE to produce solution apps, which were used as comparisons against expected app behavior in student-completed lab app solutions. For labs where a mostly empty skeleton project was provided for students as a starting point, the skeleton project was usually the lab key project with most of the completed lab programming content stripped out. This created a controlled environment for each lab where students could develop and test their lab progress (if unit tests were provided) using the same app coding project structure; instead of creating their own project for each lab and working with varied app coding project structures, all students were afforded the same starting point.

Each lab key typically consists of lines of app code contained in a **src** folder, unit test code contained in a **test** folder, configurations and layouts contained in a **res** folder, and the **AndroidManifest.xml** file contents. The lab keys are mostly representative of the sum of a lab author's efforts in completing a self-contained lab exercise to assign to students. However, the lab keys for Course v2 could not be used for Course v3 since each lab was being modified for the new course offering's content, which required removing outdated or deprecated Android development implementations present in the old lab keys.

Due to the nature of the changes made to Course v2 labs for their respective Course v3 versions, and the overall total effort needed for each lab exercise (adding, subtracting or modifying lab features, components, unit tests and written guides; assisting

instructor and students on each lab), there is no true set of metrics for measuring the time and effort required to effectively convert the Course v2 labs into Course v3 labs and support them. In an attempt to measure this effort, the conducted analysis focused on the quantity of certain types and categories of Android app code in these lab key projects.

After all Course v3 lab content was finalized, the keys for most Course v2 and Course v3 labs were analyzed to obtain a relative degree of content change between them. The old-versus-new lab pairs chosen for this analysis were the six parallel labs primarily authored by the lab & written guide authors (James Reed for Course v2 and this paper's author for Course v3). The labs compared were:

1. Old Lab 1 vs New Lab 1 ("HelloWorld" app: *Introduction to the Android programming environment and tools*)
2. Old Lab 2 vs New Lab 2 ("JokeList" app: *Create a list of jokes in an app*)
3. Old Lab 3 vs New Lab 3 ("JokeList 2.0" app: *Upgrade the previous lab's app to use more Android app features to populate and structure the joke list*)
4. Old Lab 4 vs New Lab 4 ("JokeList 3.0" app: *Upgrade the previous lab's app to use even more Android app features to better maintain the joke list*)
5. Old Lab 5 vs New Lab 6 ("WalkAbout" app: *Create a GPS path tracking app with picture-taking and map-marking functionality*)
6. Old Lab 6 vs New Lab 7 ("AppRater" app: *Create an app with a list of other apps to be rated by users*)

The code metrics measured for each key project were:

- Classes (**src** + **test**)
- Packages (**src** + **test**)
- Methods (**src**)
- Methods (**test**)
- Unit test methods (**test**)
- Class variables (**src**)
- Class variables (**test**)
- Constants (**src**)
- Constants (**test**)
- Imports (**src**)
- Code lines (**src**)
- Method lines (**src**)
- Code lines (**test**)
- Method lines (**test**)
- Resource (layout/menu/values/etc.) lines (**res**)
- Manifest lines (**AndroidManifest.xml**)

Metric data was gathered using a combination of manual observation and the Metrics plugin for the Eclipse IDE.

4.4.1 Lab Key Metric Results

Metrics for each side-by-side lab key comparison are shown in tables below, one comparison per page. To roughly and accurately estimate the differences between similar labs, collected data was broken down by relative and weighted percentage by metric,

and then summed up and/or averaged to most accurately detail changes between old and new labs. This is similar to how the Android API Differences Report data is calculated, but comes with the caveat that only additions and subtractions are accounted for in final percentage calculations (not pure 1:1 changes). This means that, for example, if an old lab has 100 lines of code in it for a given metric, and the equivalent new lab also has 100 lines of code in it for the same metric, the calculations consider this a 0% change since the number of lines is identical (even if each of those 100 lines of code are completely different between both versions).

1 point per metric was considered as 1 weight, with all metrics weighed evenly based on reverse numeric magnitude relative to the total. Note that the weighted average listed in the bottom-right of each table (green text-colored cell) is the average weighted percent difference of change *between the old and new lab*, not the total average weighted percentage difference of *both labs*. A lab project key can *generally* be considered "stronger" if it has fewer classes, packages, total src methods, member variables, constants, imports and lines of code coupled with more total test methods, total unit test methods, method lines of code, layout/menu/values lines and Manifest file lines. Noticeably large metric differences between lab versions are bolded and highlighted in a light gold color, and descriptive commentary is added for each metric table as deemed appropriate.

Old Lab 1 vs New Lab 1:

Table 4.21: Old vs New Lab 1 Key Project data

Metric	Old	New	Wt. Old	Wt. New	Rel. Diff	Wt. Diff
Classes (src + test)	2	2	1.40%	0.78%	0.00%	0.00%
Packages (src + test)	1	1	0.70%	0.39%	0.00%	0.00%
Methods (src)	**3**	**5**	**2.10%**	**1.96%**	**66.67%**	**2.71%**
Methods (test)	0	0	0.00%	0.00%	0.00%	0.00%
Unit test methods (test)	0	0	0.00%	0.00%	0.00%	0.00%
Class variables (src)	1	2	0.70%	0.78%	100.00%	1.48%
Class variables (test)	0	0	0.00%	0.00%	0.00%	0.00%
Constants (src)	1	0	0.70%	0.00%	-100.00%	-0.70%
Constants (test)	0	0	0.00%	0.00%	0.00%	0.00%
Imports (src)	11	10	7.69%	3.92%	-9.09%	-1.06%
Code lines (src)	43	61	30.07%	23.92%	41.86%	22.60%
Method lines (src)	**15**	**26**	**10.49%**	**10.20%**	**73.33%**	**15.17%**
Code lines (test)	0	0	0.00%	0.00%	0.00%	0.00%
Method lines (test)	0	0	0.00%	0.00%	0.00%	0.00%
Resource lines (res)	**46**	**117**	**32.17%**	**45.88%**	**154.35%**	**120.47%**
Manifest lines	20	31	13.99%	12.16%	55.00%	14.38%
Total	**143**	**255**			**382.12%**	**175.05%**
Average					**23.88%**	**87.53%**

Two additional methods were added to New Lab 1, which is a bigger deal given that Lab 1 is a very small introductory lab with hardly any fundamental changes. The additional method lines of code and **res** folder content suggest that the lab materials were set up to be more verbose and slightly more content-heavy for students to learn more about UI materials/concepts compared to in Old Lab 1.

Old Lab 2 vs New Lab 2:

Table 4.22: Old vs New Lab 2 Key Project data

Metric	Old	New	Wt. Old	Wt. New	Rel. Diff	Wt. Diff
Classes (src + test)	4	4	0.76%	0.59%	0.00%	0.00%
Packages (src + test)	2	2	0.38%	0.30%	0.00%	0.00%
Methods (src)	13	13	2.48%	1.92%	0.00%	0.00%
Methods (test)	11	14	2.10%	2.07%	27.27%	1.14%
Unit test methods (test)	8	12	1.52%	1.78%	50.00%	1.65%
Class variables (src)	8	9	1.52%	1.33%	12.50%	0.36%
Class variables (test)	**8**	**0**	**1.52%**	**0.00%**	**-100.00%**	**-1.52%**
Constants (src)	**3**	**3**	**0.57%**	**0.44%**	**0.00%**	**0.00%**
Constants (test)	**6**	**0**	**1.14%**	**0.00%**	**-100.00%**	**-1.14%**
Imports (src)	15	18	2.86%	2.66%	20.00%	1.10%
Code lines (src)	138	149	26.29%	22.04%	7.97%	3.85%
Method lines (src)	77	84	14.67%	12.43%	9.09%	2.46%
Code lines (test)	131	179	24.95%	26.48%	36.64%	18.85%
Method lines (test)	**64**	**116**	**12.19%**	**17.16%**	**81.25%**	**23.85%**
Resource lines (res)	**17**	**46**	**3.24%**	**6.80%**	**170.59%**	**17.13%**
Manifest lines	20	27	3.81%	3.99%	35.00%	2.73%
Total	**525**	**676**			**250.31%**	**70.45%**
Average					**15.64%**	**35.23%**

This lab overall was a more detailed and efficient new lab key project than Old Lab 2. The same number of **src** folder constants was used, despite almost double the method lines of code contained in the newer version. This coupled with the significantly higher **res** folder content suggests that students were given more detailed and descriptive independent tasks (such as hooking up more UI in different contexts) in the newer lab.

Old Lab 3 vs New Lab 3:

Table 4.23: Lab Key Project comparison data between Old Lab 3 & New Lab 3

Metric	Old	New	Wt. Old	Wt. New	Rel. Diff	Wt. Diff
Classes (src + test)	17	20	0.84%	0.58%	17.65%	0.25%
Packages (src + test)	2	2	0.10%	0.06%	0.00%	0.00%
Methods (src)	36	28	1.77%	0.81%	-22.22%	-0.57%
Methods (test)	**33**	**59**	**1.62%**	**1.70%**	**78.79%**	**2.62%**
Unit test methods (test)	**23**	**41**	**1.13%**	**1.18%**	**78.26%**	**1.81%**
Class variables (src)	20	24	0.98%	0.69%	20.00%	0.34%
Class variables (test)	7	9	0.34%	0.26%	28.57%	0.17%
Constants (src)	10	8	0.49%	0.23%	-20.00%	-0.14%
Constants (test)	0	1	0.00%	0.03%	100.00%	0.03%
Imports (src)	43	29	2.12%	0.84%	-32.56%	-0.96%
Code lines (src)	431	358	21.21%	10.32%	-16.94%	-5.34%
Method lines (src)	250	217	12.30%	6.26%	-13.20%	-2.45%
Code lines (test)	**602**	**1400**	**29.63%**	**40.37%**	**132.56%**	**92.78%**
Method lines (test)	**398**	**1064**	**19.59%**	**30.68%**	**167.34%**	**84.12%**
Resource lines (res)	139	179	6.84%	5.16%	28.78%	3.45%
Manifest lines	21	29	1.03%	0.84%	38.10%	0.71%
Total	**2032**	**3468**			**585.12%**	**176.81%**
Average					**36.57%**	**88.41%**

Significantly more unit tests were added to the newer version of this lab. This was likely done due to the finicky nature of various Android UI elements visited throughout the lab exercise, in particular the text size of jokes in the joke list.

Old Lab 4 vs New Lab 4:

Table 4.24: Lab Key Project comparison data between Old Lab 4 & New Lab 4

Metric	Old	New	Wt. Old	Wt. New	Rel. Diff	Wt. Diff
Classes (src + test)	11	12	0.58%	0.38%	9.09%	0.09%
Packages (src + test)	3	2	0.16%	0.06%	-33.33%	-0.07%
Methods (src)	72	60	3.82%	1.91%	-16.67%	-0.95%
Methods (test)	**18**	**35**	**0.95%**	**1.12%**	**94.44%**	**1.95%**
Unit test methods (test)	**15**	**27**	**0.79%**	**0.86%**	**80.00%**	**1.32%**
Class variables (src)	28	25	1.48%	0.80%	-10.71%	-0.24%
Class variables (test)	**6**	**2**	**0.32%**	**0.06%**	**-66.67%**	**-0.25%**
Constants (src)	35	31	1.85%	0.99%	-11.43%	-0.32%
Constants (test)	2	0	0.11%	0.00%	-100.00%	-0.11%
Imports (src)	48	48	2.54%	1.53%	0.00%	0.00%
Code lines (src)	730	660	38.69%	21.05%	-9.59%	-5.73%
Method lines (src)	375	361	19.87%	11.51%	-3.73%	-1.17%
Code lines (test)	**210**	**914**	**11.13%**	**29.15%**	**335.24%**	**135.01%**
Method lines (test)	**129**	**748**	**6.84%**	**23.85%**	**479.84%**	**147.26%**
Resource lines (res)	186	179	9.86%	5.71%	-3.76%	-0.59%
Manifest lines	19	32	1.01%	1.02%	68.42%	1.39%
Total	**1887**	**3136**			**811.14%**	**277.58%**
Average					**50.70%**	**138.79%**

Despite the rather significant rework this lab underwent to become New Lab 4, most metrics remained roughly the same between lab versions. Unit tests received the majority of rework treatment, with almost double the number of unit tests in the newer version. This was likely done to more carefully and thoroughly test the greatly more extensive and in-depth database implementation.

Old Lab 5 vs New Lab 6:

Table 4.25: Lab Key Project comparison data between Old Lab 5 & New Lab 6

Metric	Old	New	Wt. Old	Wt. New	Rel. Diff	Wt. Diff
Classes (src + test)	**5**	**1**	**0.56%**	**0.15%**	**-80.00%**	**-0.57%**
Packages (src + test)	1	1	0.11%	0.15%	0.00%	0.00%
Methods (src)	29	16	3.27%	2.41%	-44.83%	-2.55%
Methods (test)	0	0	0.00%	0.00%	0.00%	0.00%
Unit test methods (test)	0	0	0.00%	0.00%	0.00%	0.00%
Class variables (src)	**19**	**7**	**2.14%**	**1.06%**	**-63.16%**	**-2.02%**
Class variables (test)	0	0	0.00%	0.00%	0.00%	0.00%
Constants (src)	**13**	**5**	**1.47%**	**0.75%**	**-61.54%**	**-1.37%**
Constants (test)	0	0	0.00%	0.00%	0.00%	0.00%
Imports (src)	**50**	**35**	**5.64%**	**5.28%**	**-30.00%**	**-3.27%**
Code lines (src)	**441**	**284**	**49.72%**	**42.84%**	**-35.60%**	**-32.95%**
Method lines (src)	**256**	**192**	**28.86%**	**28.96%**	**-25.00%**	**-14.46%**
Code lines (test)	0	0	0.00%	0.00%	0.00%	0.00%
Method lines (test)	0	0	0.00%	0.00%	0.00%	0.00%
Resource lines (res)	**47**	**79**	**5.30%**	**11.92%**	**68.09%**	**11.72%**
Manifest lines	**26**	**43**	**2.93%**	**6.49%**	**65.38%**	**6.16%**
Total	**887**	**663**			**-206.66%**	**-39.31%**
Average					**-12.92%**	**-19.65%**

Despite the rather significant upgrades and general rework this lab underwent to go from Old Lab 5 to New Lab 6, the lab was significantly scaled down code-wise and most recorded metrics reflect this. The results demonstrate how condensed yet smooth the app implementation became, as this lab was almost completely unchanged fundamentally. This was also the first lab since New Lab 1 to not have unit tests, which means all calculated weighted percentages are much heavier for this lab.

Old Lab 6 vs New Lab 7:

Table 4.26: Lab Key Project comparison data between Old Lab 6 & New Lab 7

Metric	Old	New	Wt. Old	Wt. New	Rel. Diff	Wt. Diff
Classes (src + test)	8	9	0.83%	0.71%	12.50%	0.19%
Packages (src + test)	**2**	**1**	**0.21%**	**0.08%**	**-50.00%**	**-0.14%**
Methods (src)	40	51	4.15%	4.04%	27.50%	2.25%
Methods (test)	0	0	0.00%	0.00%	0.00%	0.00%
Unit test methods (test)	0	0	0.00%	0.00%	0.00%	0.00%
Class variables (src)	**12**	**20**	**1.25%**	**1.58%**	**66.67%**	**1.89%**
Class variables (test)	0	0	0.00%	0.00%	0.00%	0.00%
Constants (src)	**33**	**90**	**3.43%**	**7.13%**	**172.73%**	**18.23%**
Constants (test)	0	0	0.00%	0.00%	0.00%	0.00%
Imports (src)	48	52	4.98%	4.12%	8.33%	0.76%
Code lines (src)	**497**	**604**	**51.61%**	**47.82%**	**21.53%**	**21.41%**
Method lines (src)	245	309	25.44%	24.47%	26.12%	13.04%
Code lines (test)	0	0	0.00%	0.00%	0.00%	0.00%
Method lines (test)	0	0	0.00%	0.00%	0.00%	0.00%
Resource lines (res)	**52**	**96**	**5.40%**	**7.60%**	**84.62%**	**11.00%**
Manifest lines	26	31	2.70%	2.45%	19.23%	0.99%
Total	**963**	**1263**			**389.23%**	**69.61%**
Average					**24.33%**	**34.81%**

The much greater number of member variables in the New Lab 7 code relative to Old Lab 6 suggests that the new lab was accessing and utilizing more specific Android API classes and components. The almost doubled **res** folder content in the newer version confirms this, as the higher resource factor indicates the use of additional UI components. No unit tests for this lab means all calculated weighted percentages are much heavier for this lab.

4.5 Course v2 & v3 Lab App Device Testing

Course v3 was conducted in Spring quarter of 2013, with its predecessor Course v2 conducted years before in the Summer quarter of 2010 and Winter quarter of 2011 (with the same Course v2 materials). To demonstrate lab content status and performance beyond each course's duration, the first 5 labs created and maintained by the primary lab authors (James Reed for Course v2, this paper author for Course v3) were tested on various Android devices. The goal of this test was to observe and document any obscure or broken lab app behavior, which may result from the advancement of Android device technology and Android API evolution discussed and observed so far throughout this paper. The devices in question ranged from the exact minimum API level (10) for Course v3's lab apps to somewhat newer Android platform versions relative to when Course v3 was conducted, including several devices outside of the specified maximum API level range (17). In particular, due to the backwards-compatibility precautions taken for all Course v3 apps, the hope was that the apps born from those newer labs would suffer very few irregularities and problems running on these devices.

This testing was conducted in 2014-2015. Efforts were made to test larger-scale labs with as similar app structure and behavior as possible considering both Course v2 and Course v3 lab objective and app structure; therefore, only the apps corresponding to the first five old and new lab combinations in each course were chosen for direct testing comparison. Notably, New Lab 7 (Old Lab 6) was skipped due to its similar composition relative to previous lab exercises.

4.5.1 Testing Parameters

The Course v2 & v3 lab combinations included in this test were:

1. **Old Lab 1 vs New Lab 1** ("HelloWorld" app: Introduction to the Android programming environment and tools)
2. **Old Lab 2 vs New Lab 2** ("JokeList" app: *Create a list of jokes in an app*)
3. **Old Lab 3 vs New Lab 3** ("JokeList 2.0" app: *Upgrade the previous lab's app to use more Android app features to populate and structure the joke list*)
4. **Old Lab 4 vs New Lab 4** ("JokeList 3.0" app: *Upgrade the previous lab's app to use even more Android app features to better maintain the joke list*)
5. **Old Lab 5 vs New Lab 6** ("WalkAbout" app: *Create a GPS path tracking app with picture-taking and map-marking functionality*)

The above apps were tested on the following devices:

1. **Nexus S**, Android v2.3.3 / API level 10 (`GINGERBREAD_MR1`), virtual phone device (AVD)
2. **Asus Eee Pad Transformer** version TF101, Android v3.2.1 / API level 13 (`HONEYCOMB_MR2`), physical tablet device
3. **Samsung Galaxy S3**, Android v4.4.2 / API level 19 (`KITKAT`), physical phone device
4. **Nexus 7**, Android v4.4.3 / API level 19 (`KITKAT`), physical tablet device
5. **Samsung Galaxy S5**, Android v5.0 / API level 21 (`LOLLIPOP`), physical phone device

Devices were chosen in the hopes of achieving reasonable device type and Android platform version variety, with some devices slightly outside the expected Course v3 Android API level range. Devices were also selected due to availability.

The test procedure for each app on each device involved five steps:

- Loading the solution (or **key**) version of the app's programming project in the Eclipse IDE, and making any changes necessary so the project could build error-free.
- If the app is already installed on the device, clear all device data for and uninstall the current version of the app on the device.
- Compiling the app's project key in Eclipse into an app package (.apk file) that gets freshly loaded and installed onto the device.
- Verify device can start up the app normally, then test the app's features and expected behavior within the scope of the app's corresponding lab, based on that app's lab guide.
- Recording any observed unexpected app visuals and behavior.

4.5.2 Testing Results

Resulting observations are shown in tables below, for each pair of old and new labs. If an observation is shared across all devices or both lab versions, it is listed under the ALL row entry or BOTH column entry, respectively. If there were no observations for a given lab version on a given device, the app status is listed as "Intended Behavior".

Old & New Lab 1:

Table 4.27: Old & New Lab 1 Observed App Behavior

Device	Old Lab	New Lab	Both
Nexus S	-	-	Intended Behavior
Asus Transformer	-	-	Intended Behavior
Samsung Galaxy S3	-	-	Clicking in text field in landscape mode (when screen rotation is enabled) opens expanded keyboard view with Done button.
Nexus 7	-	-	Intended Behavior
Samsung Galaxy S5	-	-	Clicking in text field in landscape mode (when screen rotation is enabled) opens expanded keyboard view with Done button.
All	-	-	-

Old & New Lab 2:

Table 4.28: Old & New Lab 2 Observed App Behavior

Device	Old Lab	New Lab	Both
Nexus S	-	-	Intended Behavior
Asus Transformer	• App is VERY pixelated. • Pushing Enter on-screen key in expanded keyboard view does not add a new joke to the list. • Pressing Add Joke button does not result in empty joke being added to the joke list.	Pushing Done button in expanded keyboard view when text field is empty does not add a new joke to the list, but still toggles the color of the next added joke as if one was added to the list.	-
Samsung Galaxy S3	-	Pressing Enter on-screen key in expanded keyboard view when text field is empty adds an empty joke to the joke list.	Clicking in text field in landscape mode (when screen rotation is enabled) opens expanded keyboard view with Done button. When Done is pressed, the joke gets added to the list without having to press Add Joke button (even empty jokes).
Nexus 7	App appears slightly pixelated.	Pressing Enter on-screen key in expanded keyboard view does not add a new joke to the list, instead it injects a newline into the text field.	-
Samsung Galaxy S5	-	Pressing Enter on-screen key in expanded keyboard view when text field is empty adds an empty joke to the joke list.	Clicking in text field in landscape mode (when screen rotation is enabled) opens expanded keyboard view with Done button. When Done is pressed, the joke gets added to the list without having to press Add Joke button (even empty jokes).
All	-	Textfield listener does not check for empty string.	Empty joke strings are not supposed to be allowed.

Old & New Lab 3, Part 1:

Table 4.29: Old & New Lab 3 Observed App Behavior, Part 1

Device	Old Lab	New Lab	Both
Nexus S	• Initially clicking joke text area to expand joke view results in + button change to - and removes "..." at end of long joke text, but does not properly expand joke to show buttons. Clicking - button immediately after this leaves button showing - but expands joke view to show buttons. • Clicking on some joke views (or their +/- buttons) cause other joke views to expand or contract. • Buttons don't always show when a joke view is in expanded mode. • Download Jokes menu option doesn't work. • Upload Joke to Server submenu option doesn't seem to work.	-	Pushing Enter on-screen key in expanded keyboard view when text field is empty adds a line break into the text field; pushing it again adds an "empty newline" joke to the joke list.
Asus Transformer	• App appears VERY pixelated. • Clicking on some joke views (or their +/- buttons) cause other joke views to expand or contract. • Buttons don't always show when a joke view is in expanded mode. • Download Jokes menu option doesn't work.	-	Pushing Enter on-screen key in expanded keyboard view when text field is empty adds a line break into the text field; pushing it again adds an "empty newline" joke to the joke list.
Samsung Galaxy S3	Downloading/scrolling through jokes sometimes results in previously expanded views showing vertical expansion, but no buttons.	-	• Pushing Enter on-screen key in expanded keyboard view when text field is empty adds a line break into the text field; pushing it again adds an "empty newline" joke to the joke list. • Clicking in text field in landscape mode (when screen rotation is enabled) opens expanded keyboard view with Next button. When Next is pressed, the joke gets added to the list without having to press Add Joke button.

The Lab 3 app behavior chart is continued on the next page.

Old & New Lab 3, Part 2:

Table 4.30: Old & New Lab 3 Observed App Behavior, Part 2

Device	Old Lab	New Lab	Both
Nexus 7	• App appears slightly pixelated. • Downloading/scrolling through jokes sometimes results in previously expanded views showing vertical expansion, but no buttons. • Pushing Enter on-screen key in expanded keyboard view when text field is empty adds a line break into the text field; pushing it again adds an "empty newline" joke to the joke list.	Pushing Enter on-screen key in expanded keyboard view when text field is empty adds a line break into the text field. Pressing it again duplicates the behavior.	-
Samsung Galaxy S5	• Downloading/scrolling through jokes sometimes results in previously expanded views showing vertical expansion, but no buttons. • Clicking in text field in landscape mode (when screen rotation is enabled) opens expanded keyboard view with Next button. When Next is pressed, the joke gets added to the list without having to press Add Joke button.	Clicking in text field in landscape mode (when screen rotation is enabled) opens expanded keyboard view with Done button. When Done is pressed, the joke gets added to the list without having to press Add Joke button. Expanded keyboard view does not disappear after Done is pressed.	Pushing Enter on-screen key in expanded keyboard view when text field is empty adds a line break into the text field; pushing it again adds an "empty newline" joke to the joke list.
All	• Clicking + button on any joke does not expand the joke view to full size; jokes only initially expand properly by clicking joke text area. • Joke minimization ("..." at end) when joke is collapsed doesn't work.	-	Empty joke strings are not supposed to be allowed.

Old & New Lab 4:

Table 4.31: Old & New Lab 4 Observed App Behavior

Device	Old Lab	New Lab	Both
Nexus S	Same issues as Lab 3.	-	Same issues as Lab 3.
Asus Transformer	Same issues as Lab 3.	-	Same issues as Lab 3.
Samsung Galaxy S3	Same issues as Lab 3.	Clicking in text field in landscape mode (when screen rotation is enabled) opens expanded keyboard view with Done button.	Same issues as Lab 3.
Nexus 7	Same issues as Lab 3.	Same issues as Lab 3.	-
Samsung Galaxy S5	Same issues as Lab 3.	• Clicking in text field in landscape mode (when screen rotation is enabled) opens expanded keyboard view with Done button. When Done is pressed, the joke gets added to the list without having to press Add Joke button. • Joke text in joke views is tiny.	Same issues as Lab 3.
All	Same issues as Lab 3.	• Same issues as Lab 3. • Force Stopping app and restarting it maintains database of jokes, but doesn't initially populate joke list with previous joke cursor/query result on startup.	Filtering jokes and changing their rating causes them to disappear from the currently displayed list. Same with adding a new joke while the list is currently filtered for anything but unrated/show all jokes.

Old Lab 5 & New Lab 6:

Table 4.32: Old Lab 5 & New Lab 6 Observed App Behavior

Device	Old Lab	New Lab	Both
Nexus S	• App loads, but with no working functionality. • App crashes upon pressing Start or Load options.	Clicking "get Google Play Services" button (see below) crashes app.	-
Asus Transformer	• Map squares do not load. • GPS does not work (likely due to Wi-Fi only capabilities of device, "Searching For GPS/Location" message). • All functions/menus work, but marker does not move based on location change.	Clicking "get Google Play Services" button (see below) takes player to the Google Play Store to install Google Play Services. However, device is "incompatible with this version" of Google Play Services.	-
Samsung Galaxy S3	• Map squares do not load. • GPS does not work (shows "Searching For GPS/Location" message). • App crashes upon pressing Start option. • Take Picture menu option does nothing.	Clicking "get Google Play Services" button (see below) takes player to the Google Play Store to install Google Play Services. App functions normally afterwards.	-
Nexus 7	• Map squares do not load. • GPS does not work, likely due to Wi-Fi only capabilities of device (shows "Searching For GPS/Location" message). • App crashes upon pressing Start option. • Take Picture menu option does nothing.	Clicking "get Google Play Services" button (see below) takes player to the Google Play Store to install Google Play Services. App functions normally afterwards.	-
Samsung Galaxy S5	• Map squares do not load. • GPS does not work (shows "Searching For GPS/Location" message). • All functions/menus work, but marker does not move based on location change. • Take Picture menu option does nothing.	Clicking "get Google Play Services" button (see below) takes player to the Google Play Store to install Google Play Services. App functions normally afterwards.	-
All	-	Popup appears on phone stating "WalkAbout won't run unless you update Google Play services" with button that allows the user to update Google Play services.	-

4.5.3 Testing Summary & Analysis

Device-specific app testing results were mostly expected, given the evolutionary nature of Android devices, platforms and APIs. The number of performance and visual oddities for a given app, considering its associated lab's complexity and level of behavior allowed, was unsurprising due to the course timeframe and lack of official/high quality resources available. It was therefore expected that many of the apps, especially those from Course v2, would struggle to run on newer devices.

Course v2 lab versions and their respective apps, having been finalized and left unchanged since 2010, would struggle to perform as expected or even completely fail to work on newer devices in 2014-2015. Course v2 apps were designed to work on devices with a minimum API level of 4 (Android 1.6, `DONUT`) with no maximum API level specified, meaning the apps were not planned to run on newer Android devices and API levels (nor were there many if any commonly used compatibility solutions at the time). As expected, several Course v2 apps encountered major behavioral failures, in particular Old Labs 3 & 4 missing much Joke List and menu functionality and Old Lab 5 ceasing to function or crashing completely (likely due to utilizing an older and much less supported Google Maps API version on most Android devices). Most Course v2 apps had at least 1 or 2 buttons completely lose their functionality, causing app slowdown or crashing. Several devices (especially the tablets) also rendered Course v2 apps in a pixelated manner, suggesting older API components, UI elements and sizes & dimensions used in the lab projects that are not ideal for the majority of new devices since Course v2 was taught (especially tablets, which had not been popularized at all yet). These UI issues and inconsistencies are further supported by the Course v3 survey result feedback asking for more device-specific UI design coverage in lab exercises.

Developed three years later, Course v3 lab versions and their respective apps were expected to greatly outperform Course v2 equivalents. This is primarily due to the new labs utilizing newer Android API levels (and putting a cap on the supported API device range by specifying a maximum API level) as well as accounting for compatibility on a variety of old and new devices through ActionBarSherlock. All 5 Course v3 apps, which allowed for a minimum API of 10-17 (Android 2.3.x, `GINGERBREAD_MR1`-Android 4.2.x, `JELLY_BEAN_MR1`), could successfully run and behave at least mostly properly on most if not all devices, with generally fewer observed issues on average compared to the same Course v2 apps. However, Course v3 apps generally suffered from similar specific interactivity inconsistencies and unwanted scenarios, such as dealing with the on-screen keyboard, due to the growing and ever-changing number of device interaction options at the time. This was likely due to a large focus on device input handling in early labs, but not in later labs. Thus, it seems reasonable to put less focus on handling many cases of device input and virtual controls in future iterations of the course, leaving it primarily up to students to learn the ins and outs of their lab app test devices. Most other observed Course v3 app issues were minor; as the Course v3 labs were formalized with a relatively large API level intended support range and app compatibility in anticipation of major development problems, this result was anticipated.

Lastly, while almost every old and new app combination shared a small set of minor problems on at least one of the test devices, the Joke List apps in particular seemed to contain incorrect implementation regarding behavior relative to what was expected of their related lab exercises. Blank jokes were allowed in several of the Joke list app versions, and a large mix of on-screen interactivity (such as combinations of entering jokes through the on-screen keyboard and in-app buttons) would cause unwanted or

odd behavior cases. These seem like incorrect lab key implementations and app behavior rather than Android evolution-related issues, and should be investigated and fixed for future course offerings.

An interesting final note: despite several devices running newer versions of Android (API level 19 for the Samsung Galaxy S3 and Google Nexus 7, API level 21 for the Samsung Galaxy S5) than allowed by Course v3 apps (API levels 10-17), those apps were able to install and run on those devices anyway. Upon further investigation, the official Android "uses-sdk" documentation page [7] has been updated since Course v3 was instantiated with the following warning and information about imposing a `maxSdkVersion` property in an Android development app project's **AndroidManifest.xml** file ***(Figure 4.1)***:

> **Warning:** Declaring this attribute is not recommended. First, there is no need to set the attribute as means of blocking deployment of your application onto new versions of the Android platform as they are released. By design, new versions of the platform are fully backward-compatible. Your application should work properly on new versions, provided it uses only standard APIs and follows development best practices. Second, note that in some cases, declaring the attribute can **result in your application being removed from users' devices after a system update** to a higher API Level. Most devices on which your application is likely to be installed will receive periodic system updates over the air, so you should consider their effect on your application before setting this attribute.

Introduced in: API Level 4

Future versions of Android (beyond Android 2.0.1) will no longer check or enforce the `maxSdkVersion` attribute during installation or re-validation. Google Play will continue to use the attribute as a filter, however, when presenting users with applications available for download.

Figure 4.1: Android App Manifest file maxSdkVersion property warning

This new information, coupled with new official Android backwards-compatibility documentation, suggests that Android's official app installation and development implementations/practices have been refactored by Google. Further research suggests that all Android apps made to run on more recent Android platforms (Android 3.0 / `HONEYCOMB` and newer) will in most cases seamlessly work on devices running that platform level and higher, invoking backwards-compatibility automatically if necessary. Official newer Android UI elements such as SearchView and the Action Bar were introduced in Android 3.0, and the entire point of ActionBarSherlock was to provide an unofficial backwards-compatible version of the Action Bar for previous versions. Additionally, official Android backwards-compatibility documentation states that the traditional menu system will now be used for apps target API versions below Android 3.0, thus solving the entire issue for almost all app and device cases. Extreme outlier cases aside, this new official app construction implementation effectively renders third-party backwards-compatibility systems (such as ActionBarSherlock, which was heavily utilized in Course v3 apps) completely obsolete.

Chapter 5

FUTURE WORK

While the overall results of this project generally indicate educational success within the intended scope, there are several general opportunities for improvement over the original project that should be considered for any future course iterations, assuming a fundamentally and structurally similar course.

5.1 Android Studio IDE

Android Studio is the official IDE for Android development [13] that was first released in May 2013 (version 0.1.x) and, at the time of writing this paper, is now up to version 2.3.2 (released April 2017). This new IDE, provided and maintained by Google, was based on a similar IDE, IntelliJ IDEA, which is provided and maintained by JetBrains s.r.o. [16]. Android Studio fully replaces the Eclipse IDE as the primary and official Android app development IDE. Such improvements over Eclipse include expanded Android app development support and advanced tools such as the ability to show new app code changes in an app already running on a device connected to Android Studio, a more expanded and feature-laden Android device emulator, simultaneous support for all Android devices, and wider version control support and integration. Android Studio aimed to preserve the familiarity of Android app development in Eclipse IDE by largely maintaining the same IDE structure, even offering an import conversion tool for previous Android Eclipse projects (including multiple related projects at once) as well as IntelliJ IDEA Android projects.

Android Studio's strong toolset creates good opportunities for new app concepts and exercises, particularly for UI design across multiple devices. It would not be unreasonable to not only convert all Android development in the course to this newer, more widely supported tool, but to also incorporate Android Studio's increased support for multiple devices into one or more lab exercises. This addresses the significantly increased consumer tablet usage since Course v3 transpired, and gives students increased insight into UI design for more than just Android phone devices (especially since Android Studio offers specific support for generating multiple APKs for different screen densities and Application Binary Interfaces, or **ABI**s, which decide how an app's machine code interacts with an Android device CPU architecture at runtime).

Additionally, Android Studio's inline debugging feature, which allows users to walk through live code execution of their app while it's running on a device, is a huge step up from the previous debugging method of printing out debug statements using LogCat, allowing for a much higher degree of app testing and code verification. This particular feature would significantly strengthen lab exercises once introduced to or discovered by students, allowing them to live debug lab apps to a large degree at almost any time without depending solely on the less convenient LogCat methods or included lab unit tests for debugging possible lab app code issues.

Lastly, the greater degree of accessibility in Android Studio's tools (including the configurable Gradle build system, Google Cloud Platform integration for creating and deploying Android app hosts, and additional support for higher-level device development using C/C++ code files and the Android Native Development Kit, or **NDK**) provides an increase in app development flexibility and customization, creating numerous course lab exercise extension opportunities. For instance, a hypothetical new lab exercise involving a simple interactive game may focus on utilizing the Android

NDK to maximize the app's performance on a variety of devices. This also creates an opportunity to move beyond a mostly Java-related programming course emphasis.

5.2 Additional Future Course Recommendations

While the Course v3 lab materials were recently used for two newer Android app development course offerings (taught in Spring and Fall 2016 at Cal Poly, San Luis Obispo), this iteration of the course was taught under different circumstances than before. These offerings were lectured by a former employee under Dr. David Janzen, Tony Lenz, who used the Course v3 lecture content and lab materials as a starting point. However, not much else is known about how these course offerings fared or their results, as Tony mostly taught the course independently. As a result, despite changing the Course v3 lab materials for a new course offering (the requirement to consider this course offering as *Course v4*, for the purposes of this paper), Tony's course teachings cannot be considered an official successor to Course v3.

For a future official course iteration (*Course v4*), there are several recommended content changes; these recommendations encompass what has already been discussed in the previous section about the Android Studio IDE and the survey results contained in the previous chapter. These change recommendations follow:

Firstly: more labs should take advantage of new feature support and built-in device sensors in newer devices and API levels. New development-side feature support has arguably been greatly improved by Google, and by the modernization and advent of newer devices running more stable, consistent and feature-heavy Android platform versions. Thus, there is not as much of a need to focus on preserving older

device features in the app course, which provides a great opportunity to cover newer, more interesting and exciting device features such as Near Field Communication (NFC) or scanning barcodes. Notably, there is no more need for manual backwards-compatibility integration (ActionBarSherlock in Course v3) in future course offerings since Google established an automatic backwards-compatibility for proper interface selection at, under and over Android API levels 11 (Android 3.0, `HONEYCOMB`). Furthermore, Google has provided stronger backwards-compatibility via updates to the Android Support Library and significantly upgraded compatibility documentation. Thus, less lab time needs to be dedicated to backwards-compatibility, and lab projects can do away with unofficially added compatibility support.

Additionally, the labs could also take more advantage of the built-in sensors in most if not all Android devices such as the motion & orientation sensors (accelerometer, gravity, gyroscope, rotational), environmental sensors (barometer, photometer, thermometer) and position sensors (orientation, magnetometer). Only one lab in Course v2 offered sensor exposure (Old Lab 7), but only to the point of demonstration (displaying device accelerometer readings) rather than real application. Based on student feedback, the Course v3 labs were generally too static, with WalkAbout being the only lab that got students to utilize their test devices physically. Better integration of sensors into lab exercises would make for more exciting and practical self-contained exercises, while simultaneously broadening student Android device knowledge.

Secondly: to address the impression students had that the labs weren't as hands-on or exciting as they had hoped. The course could benefit from having different and more varied lab exercises, covering a variety of other more interesting and practical topics. The Joke List apps in particular were not only a slightly missed opportunity to more deeply explore Android UI design, but also took up too many lab exercise

slots with an evolving yet largely redundant exercise set. For these labs, students understandably started to grow tired of the same app context over time. Additionally, several students showed an interest in more interactive or game-centric app exercises.

Several replacement lab app ideas are listed below:

- A themed for-fun survey (survey theme up to the student) with various UI input elements (radio buttons, drop-down menus) that could serve as an introduction to designing UIs for multiple devices and screen sizes. Once submitted, the survey would show a result in the form of a picture and brief explanation based on chosen answers. To expand this app, it could be backed by a database or push responses to a host, similar to New Lab 4 and New Lab 5 respectively, where all prior submissions and results can be viewed.

- A basic coloring/drawing app that responds to touch commands for drawing. To expand this app, the exercise could include loading photos from the device to draw over and saving creations to device storage, or add additional drawing tools. Students could also push creations to a server or share them with other users via NFC if possible.

- A simple interactive game that utilizes the device's accelerometer and/or gravity sensors to move something in-game around. This would likely be a more advanced exercise, although it could be expanded by adding more interactive gameplay elements or an NDK app device optimization component to the lab for better performance.

Thirdly: to the students' desire for more flexibility in lab exercises, and additionally wanting more time to work on the team-based final course app project. Due to the general shortage and limitation of unit tests in various labs, students expressed concern for trying to make their lab results pass specific unit tests at checkpoints during some labs, or being unable to run or write their own unit tests in others. It would make sense to dedicate some time in a future early or mid-course lab to discussing unit tests and how to write them; this would not only alleviate pressure on the lab author to assemble many unit tests to match a broad spectrum of cases that work on a variety of devices, but also allow students to experiment or even significantly utilize unit tests on their own. However, this may require a change to how course labs are graded. Also, survey results indicated that there were too many lab exercises in the course, leading to a lower quality or smaller final project app. Discounting New Lab 5 which was extremely short, it seems like a smarter idea in the future to have students work on one or two fewer labs throughout the course to provide more time and resources for the final project app, which is easily the largest and most complex project in the course curriculum due to students having more freedom to decide project content (as opposed to the lab exercises, which are generally self-contained). This may require significantly changing the lab exercise lineup, since fewer labs mean less opportunities to introduce new Android development content in course exercises (lecture may be less affected).

Finally, however, the biggest obstacle to upgrading or changing the course material and content is dedicated time. After several Android app course offerings, it is clear that simply maintaining and upgrading older lab contents to account for Android API and SDK changes is time-consuming enough on its own; due to the limited time to finalize the curriculum for Course v3, the author chose to hardly stray from the path laid out by Course v2 and expand lab content only moderately. Without sig-

nificant time and effort to redesign the course lecture and lab materials for newer, more relevant and involved Android exercises given student feedback and the evolution of Android devices and device types, fully revising course material with an appropriately-updated curriculum will remain a difficult task. Furthermore, maintaining such course materials once they are upgraded comes with the same problems. Android devices and development tools seem to always evolve to some degree beyond third-party references, meaning the hypothetical Course v4 and beyond will require constant upgrades, testing and maintenance to meet new standards or more modern components as time goes on.

Chapter 6

CONCLUSIONS

This project has been an exercise and experiment in modifying, creating and supporting materials and resources for an Android application development course taught at California Polytechnic State University, San Luis Obispo. Serving as a resource case study framed in a single snapshot in time, as well as building upon a solid foundation (Course v2), the Course v3 project proved effective for immediate course needs.

Before summarizing the project in its entirety, however, given the results of this case study and discussion of course recommendations in the previous chapter, it is worth exploring how to possibly change the underlying structure of the Android Application Development course. The project results obtained in this case study provide an opportunity to discuss how to effectively improve and change the course curriculum and structure, especially regarding the lab exercises, in order to prevent future course iterations (Course v4 and onward) from falling victim to the same problems that befell Course v3.

6.1 Course Restructure Discussion

While some elements of the course are required to change due to the evolving nature of Android devices, the API and official development resources, it is crucial to strengthen the course against these future evolutions. Based on this project's results, the app development course has overall been proven reliable and future-proof on a fundamental level. Lab objectives and exercise directions have generally stood the

test of time, with Course v3 survey results affirming this notion. However, regardless of how well core concepts have held up over time in each lab, many API- and software-dependent aspects of all lab exercises have been proven to be greatly at risk of requiring great degrees of change as time goes on. Combined with survey result feedback, this creates a case for course and lab curriculum restructuring to minimize how much change must be made for each lab between course offerings.

6.1.1 Fundamental Analysis & Recommendations

Firstly, an overall fundamental assessment of the course lab exercises. A general fundamental concept list from each new lab follows for convenience:

- **New Lab 1** (*Hello World*): Android development and IDE basics, setting up devices, app signing
- **New Lab 2** (*Joke List*): Nested Views and Widgets, dynamic layouts, accessing resources, debugging, events and event listeners
- **New Lab 3** (*Joke List 2.0*): Static layouts, data binding (Adapters), menus
- **New Lab 4** (*Joke List 3.0*): App persistence, databases (Cursors, content providers)
- **New Lab 5** (*Joke List 4.0*): HTTP connections, asynchronous tasks
- **New Lab 6** (*WalkAbout*): Google Maps, location services and navigation, Shapes and drawing, Camera, file storage
- **New Lab 7** (*AppRater*): More Widgets, Services, broadcasting intents and notifications, connecting to Google Play Store

Note that app compatibility (backwards-compatible Action Bar via ActionBarSherlock) is not included as it is not a fundamental development concept, but instead a special solution to a time-sensitive problem.

The most obvious concern is the relatively large number of labs given the course duration (roughly 10-12 weeks, as Cal Poly SLO is on the quarter system and the course does not span across multiple classes). A larger number of labs implies more effort needed to change between course offerings to keep up with API changes and deprecations. This concern is further compounded by the foundational expansion dependence of all Joke List labs, which take up over half of the exercises in the course and rely on being balanced and updated relative to each other. Additionally, students voiced valid concerns over the dense nature of the course based on general survey results, feeling like there was not enough time dedicated for the final team-based course app project. For example, New Lab 4, the most frustrating lab exercise due to its inclusion of a database implementation, seems difficult to include within a short quarter-based course where prior database knowledge is not assumed. Since the course is intended as a foundational springboard into the field of Android development, it therefore makes sense that future course offerings reduce the total number of labs in the curriculum by 1 or 2 (down to 5 or 6 labs instead of 7).

The next concern for the current course curriculum structure concerns the self-contained format and the app development experience of each lab. Further project results assessments, as well as student feedback and suggestions, both echo the stale nature of similarly-constructed labs (especially the layout exercises and UI structure of the Joke List and AppRater apps) and the exhaustive nature of building on the same app repeatedly. However, the core concepts covered by each of these labs are still relevant within the API and serve as solid foundational learning exercises. Since

each lab is intended to be a self-contained exercise, it seems unnecessary to have so many repeated design concepts and structural app similarity between all course lab exercises. For starters, without factoring in even more advanced UI concepts such as designing for different screen sizes and resolutions or significantly custom UI elements, there is too much general app composition overlap (e.g. the similarity between the Joke List `JokeViews` and AppRater `AppViews` and their simple vertical list structure). New Lab 7 is almost entirely rehashed content from previous labs, only generally introducing Notifications and ServiceIntents as new concepts. Since students are currently working on final course app projects while working on New Lab 7, and New Lab 7 was purposefully designed as a shorter and more concise exercise for students to account for this priority, it seems perfectly reasonable to move its newly-introduced core concepts into another lab and deprecate the lab altogether. Notifications and ServiceIntents seem simple and exciting enough to introduce earlier in the course, and thus in an earlier lab, not to mention their usages and implementations have been made much smoother in newer APIs after Course v2.

Finally, there is now a much larger selection of thoroughly maintained official Android development documentation and guidelines, rendering many introductory or step-by-step exercises redundant. New Lab 4 is one such affected example: not only does it seem more appropriate to leave databases as a possible exploratory exercise for final course app projects instead (especially given the generally frustrated student feedback on working with New Lab 4), but SQLite databases are now largely covered sufficiently by official developer documentation[1]. Conversely, the app persistence fundamentals covered early on in New Lab 4 could easily be moved into an earlier lab. Another example is New Lab 6; the entire process of obtaining a Google

[1] **Saving Data in SQL Databases** training page:
`https://developer.android.com/training/basics/data-storage/databases.html`

Maps API v2 key and app authorization, implementing a working Google Map with location services, drawing on the map with Shape paths and Markers, and interacting with and customizing a Google Map now seems to be completely detailed by official Google Maps API guides and tutorials (and are even all referenced from within a single guide page[2]). While Google Maps API v2 is still being used years later, this new documentation factor, combined with generally low student usefulness ratings for New Lab 6 in survey results, indicates that New Lab 6 is now largely overshadowed by new official documentation and not as interesting or exciting as initially thought for Course v3.

Given the above discussion, in order to minimize lab content change between future Android app development course offerings, it is advised to:

1. **First** read through current Android API reference guides, getting started guides and other official resources such as Best Practices[3] to see what Android concepts and standards are already covered in great detail. Compare the current course materials to what is currently available and well-documented to see what exercises can largely be stripped out, and identify sets of concepts and fundamentals that can be fitted around these resource-detailed fundamentals.

2. Stick closer to more abstract and fundamental API classes and implementations that have changed very little since early API levels, but use them in a more combinatorial or clever way that is not covered by modern API guides and official documentation. Focus more on *applications* of common programming concepts such as parsing and working with data files, rather than the concepts themselves.

[2]**Maps Android API** Getting Started guide page:
`https://developers.google.com/maps/documentation/android-api/start`

[3]**Best Practices** API guide page:
`https://developer.android.com/guide/practices/index.html`

3. Take much more advantage of physical device features and sensors, which will usually remain the same or similar over many API level iterations and will always be available in devices going forward.

4. Be wary of incorporating features or improvements that are new and likely to change due to developer or user demand, especially classes concerning app visuals (such as the PointerIcon class[4], at the time of writing this paper) and optimization implementations.

5. Aim for a total of 4-5 unique and interactive labs, with up to 2 labs that relate to each other at most (one building on the other). Consider making the Hello World lab smaller or more expedited (discussed more in the next subsection).

It is difficult to recommend exact API classes or implementations to include in new course lab exercises without considering or knowing how many labs the course will employ, as well as what the curriculum (lecture and lab) will contain and cover. However, based on survey results from this project, it is clear that students generally enjoy more interactive, modern and widely practical concept exercises (such as working with menus, Widgets, asynchronous tasks, HTTP connections, and referencing resources). Students would also very likely be more receptive to tasks and concepts that are relevant to typical modern mobile device usage on a daily basis due to personal familiarity; several examples include expanded exercises on saving and loading data from various local and web sources, communication with other devices, broadcasting custom notifications, and referencing other existing apps such as Camera within lab apps.

[4]**PointerIcon** API class page:
`https://developer.android.com/reference/android/view/PointerIcon.html`

6.1.2 Structural Analysis & Recommendations

Secondly, a higher-level assessment of course duration and purpose. The course can certainly benefit from changing on a fundamental concept level, but the overall course structure also warrants investigation based on the findings encountered throughout this project.

First, it is worth revisiting the purpose of the app course as a springboard for students to get introduced to and interested in the field of Android app development. The course should strive to show off the potential of Android, but not show or explain too much to students so there is room for self-exploration and growing self-interest in the subject. The Course v3 labs and written guides focused heavily on exposition in an attempt to sufficiently teach and explain Android concepts to students; as a result, there was less concern for students possibly growing overly dependent on the lab exercises instead of delving into the API and official resources to figure out Android development themselves. Therefore, an emphasis on more custom apps with more "clever" implementations should be stressed over a greater number of expository lab exercises (especially given the course density and official guide redundancy points given in the previous subsection). For example, New Lab 5, while a short and somewhat outdated (instruction-wise) lab, got students hungry for working with HTTP connections to connect to a server. Students realized the potential of this feature and how practical it was in 2013, and SQLite is still a viable database implementation option since such modern technology frameworks still do not change as quickly as mobile APIs and devices.

Another idea to promote this course's potential is to pair up students for one or more lab exercises, instead of just for the final course app. Based on survey results,

students generally seem more confident working on team-based apps; this is likely because they are more mature later in the course, and can lean on each other for ideas and assistance while working on implementing much more complex and freely chosen app concepts compared to the lab exercises. Pairing up students on one or more lab exercises would not only ease students into the final team-based course app project (which would prepare them for working in mostly larger teams), but also allows for more complex lab exercises to be balanced out by having two students working on an exercise together instead of just by themselves. It is also reasonable to pair students together for the Hello World lab to help students get up to speed on Android development and Java programming faster, but this comes with several grade-balancing risks and would have to be accounted for in the course curriculum (and comes with the assumption that students in this upper-division course can be trusted to work honestly and evenly with others on team-based assignments).

Finally, the duration of the course compared to the direction and mission of the course. Given the short duration of the course on the quarter system at Cal Poly SLO, students have found themselves squeezed through a dense curriculum. While students generally come out of the course feeling significantly prepared and versed in Android development, it is viable to entertain the idea of a multi-class Android development course that spans 2 quarters instead of 1. There will likely be more than enough demand to merit such a course expansion, but additional course content must be created, maintained and initially balanced for such a time and curriculum change. It is not clear or obvious, however, that extending the course will further develop the goal of the current course appropriately and in a desirable way. The second half of the course (quarter 2) would likely revolve around a much more applied series of exercises, possibly working more with students in other classes, the university or even interested company partners.

Overall, aside from fundamentally changing the labs (and adjusting the number of labs) as stated in the previous subsection, it is likely a safer and more efficient option to keep the course structure as similar as possible for future offerings. As an upper division course, the potential to give students Android-themed senior project or Master's thesis ideas (or other various personal/professional mobile development opportunities) after taking the course may be too valuable to move away from.

6.2 Project Summary

Android devices evolved over the years to meet growing consumer mobile device demand, including the addition of many new device features and types. New Android devices, particularly tablets, were born and began accompanying mobile phones in the mobile consumer market. To support this device evolution, Google not only modified and evolved its Android device OS, but also created and evolved several development tools for Android app developers. The most important of these tools was an API for developers used for making and publishing apps to the Android device app market. James Reed, a previous Cal Poly, SLO graduate student, created a series of Android app development learning exercises for a Master's thesis project based on these official Android resources in 2010. These lab exercises were used by Dr. David Janzen to teach core Android app development concepts to students in an upper-division course at the university. The first two iterations of this course, Course v1 and Course v2, were largely successful, with the labs used in these course offerings initially met with astounding educational and worldwide success.

However, increasingly problematic development issues began to emerge, and grew in magnitude as more Android devices, API levels and development tools were introduced. The lab exercises used in Course v1 and Course v2 quickly began to show age within a year of introduction, and with a lack of resources to upgrade them they became outdated and inaccurate largely within a year of being published. Furthermore, due to the lack of reliable, thorough and widely-supported official Android solutions in the earlier Android development years, device and developmental backwards-compatibility were minimally supported and became a problem for not only app developers who were not constantly updating apps to accommodate newer devices and API levels, but also those who used older devices trying to use those apps. As a third-party resource dependent on the API and official resources, Course v2 fell victim to this evolution; its lab exercises and materials became extremely outdated due to lack of support for newer Android OS versions and API levels.

With these issues in mind, an attempt was made for a new course offering, Course v3, to not only update and polish the Course v2 labs to better represent newer versions of Android and remove deprecated or unacceptably old lab content, but to also plan for this evolutionary Android trend. In particular, backwards-compatibility was emphasized and implemented in most new labs in the hopes of carrying over newer Android API UI elements (as well as changes to older ones) primarily through the use of ActionBarSherlock, while still utilizing these elements in newer API levels. Additionally, several significantly older Android concepts or component implementations found in the Course v2 labs were removed in favor of newly supported implementations or even outright removed and replaced by newer, more modern concepts or components. All lab guides were given more depth, references and details in the hopes of arming students with a better resource foundation. Despite moderate to large changes to the lab exercises and code projects for Course v3, the same lab core concepts were largely

carried over from Course v2, including a minimum target of only mobile devices due to the current lack of a large tablet prominence and extensive tablet development support.

While Course v3 was overall highly well-received, it encountered its fair share of issues reported by student lab and course surveys. These problems ranged from problems with overly complex lab content or written guides, to complaints about the outdated or stale nature of incorporated lab concepts, to not offering enough exciting and exploratory content. Additionally, the occasional response indicating missing or misleading bits of information in the written guides demonstrated that Android API contents, details and tools were already being revised and updated by Google; such information included in the Course v3 labs was already growing outdated and inaccurate due to the inability to change lab resources and guides to match every official Android change. The student responses were rounded out by very accurate and helpful suggested feedback for possible improvements or additions to the course and lab exercises.

This student survey feedback, combined with the results of Course v2 and v3 lab app device testing on a variety of modern-day devices that took place several years after Course v3's completion, indicated that Course v3 was overall a very positive iteration of the course and a strong entry into the Android Application Development course at Cal Poly, SLO. Finally, the findings resulting from the project's case study were used to pitch suggestions for new course developments and changes. Afterwards, possible course change ideas for future offerings on a structural and fundamental level were discussed for future work succeeding this project.

6.3 Final Project Conclusion

Overall, the project was a success for immediate course needs. Course v3 and the included lab exercises were generally very well-received with adequately appropriate modifications, updates and content introductions made by the author. Any changes made from Course v2 were generally taken well, with only generally minor negatives heard by students after lab completion. Students very highly rated the course, which left enough of an impact on the majority of students such that they would continue Android application development beyond the scope of the course. Despite the opportunities to add new features, lab content or even entirely new labs to the course curriculum, Course v3 can be considered an educational success, and a solid successor to Course v2 despite sticking mostly to the core concepts from Course v2.

However, the project fell victim to and greatly demonstrated the effects of Android device evolution, platform changes and development creep. The number of Android devices and device versions per manufacturer has not eased up since Course v3's debut, with the additional complication of manufacturers generally still adding their own spin to the Android OS provided to consumers on their devices. This implies that unique device behavior is still a widely varied and difficult problem to anticipate. Additionally, student survey feedback indicated that a handful of Android components, API solutions and development implementations were being changed or deprecated even during the duration of the course; this was after all Course v3 labs were finalized, up to and including the day each lab was released during the duration of the course.

Furthermore, most Course v3 lab apps encountered some degree of trouble when run on test devices years later; by itself this is a minimal issue, but these test devices were not running anywhere close to the newest versions of the Android platform at the time of testing. While the Course v3 lab apps were not tested on Android devices running relatively new Android versions from 2016-2017, the outdated trend that Course v2 found itself stuck in is likely to similarly impact Course v3, if it hasn't happened already. This is further supported by the deprecation of the Eclipse IDE and ActionBarSherlock. While Eclipse IDE Android projects are generally easy to port over to Android Studio, not only do the lab projects and guides assume Eclipse IDE is being used (incorporating various elements and shortcuts in Eclipse for Android development in the IDE) but there is a dependence on the ActionBarSherlock support library which will undoubtedly complicate that process. In particular, ActionBarSherlock became deprecated not just by Android's new backwards-compatibility support and documentation, but also due to becoming unsupported as of December 2014 (when it was discontinued and marked as discontinued by its author, Jake Wharton). Thus it is highly likely that the Action Bar and Context Menu implementations contained in the labs will clash with apps targeting significantly newer Android devices that do not have this problem. Thus, Course v3's backwards-compatibility implementation was effectively nullified for any future carryover.

The last concern for continued use of Course v3's content is its lack of newer, more practical content relative to Course v2. Due to time constraints and content agreement between Dr. Janzen and the Course v3 labs' primary author, a safer approach was chosen for Course v3. As such, many Course v2 labs and course content were fundamentally carried over relatively untouched for Course v3. The clear demand by students for more advanced, interesting and varied Android app development exercises and overall course content indicates that this safer approach was perhaps *too*

safe. Combined with ongoing Android device and developmental creep, this implies that if the course structure is kept relatively similar for future offerings, previously devoted levels of time and effort to maintain the course curriculum against modern Android standards is largely not enough to do future course iterations justice.

The final conclusion of this project is that Course v4 and onward require a great deal of time and effort to properly upgrade course content and lab exercises for future offerings. It is clear after this case study that Android evolution vastly eclipses and outstrips third-party resources without moderate to extreme dedication to making them easier to maintain. The course needs to not only become a more involved and detailed Android app development learning experience to better educate and prepare students, but also catch the course curriculum up with the newest Android devices, API levels, documentation and developmental tools that are constantly updated and iterated on by Google. This provides an opportunity to change the course in one or more ways. A variety of suggestions have been made over the previous chapters for improving future course iterations, not just on a material level but also on fundamental and structural levels. It is possible with carefully chosen course modifications and focal shifts to minimize the amount of content that must be changed in the course materials over time, which will hopefully make the course overall more future-proofed and effective in future offerings.

BIBLIOGRAPHY

[1] Amazon Mobile LLC. Official Android Amazon Music App Page. https://play.google.com/store/apps/details?id=com.amazon.mp3, 2016. [Online; accessed 16-Oct-2016].

[2] Apple Inc. Apple Launches iPad - Magical & Revolutionary Device at an Unbelievable Price. http://www.apple.com/pr/library/2010/01/27Apple-Launches-iPad.html, 2010. [Online; accessed 16-Oct-2016].

[3] Apple Inc. Apple Sells One Million iPads. http://www.apple.com/pr/library/2010/05/03Apple-Sells-One-Million-iPads.html, 2010. [Online; accessed 16-Oct-2016].

[4] Z. Epstein. ASUS Eee Pad Transformer offers Honeycomb and a convertible form factor. http://bgr.com/2011/03/25/asus-eee-pad-transformer-offers-honeycomb-and-a-convertible-form-factor, 2010. [Online; accessed 16-Oct-2016].

[5] Google Inc. Introducing Nexus S with Gingerbread. https://googleblog.blogspot.com/2010/12/introducing-nexus-s-with-gingerbread.html, 2010. [Online; accessed 16-Oct-2016].

[6] Google Inc. Nexus: The best of Google, now in three sizes. https://googleblog.blogspot.com/2012/10/nexus-best-of-google-now-in-three-sizes.html, 2011. [Online; accessed 7-Jan-2017].

[7] Google Inc. App Manifest API Guides - API Versions. https://developer.android.com/guide/topics/manifest/uses-sdk-element.html, 2016. [Online; accessed 16-Oct-2016].

[8] Google Inc. Google Maps Developer Home Page. https://developers.google.com/maps, 2016. [Online; accessed 16-Oct-2016].

[9] Google Inc. Google Play Store. https://play.google.com/store, 2016. [Online; accessed 16-Oct-2016].

[10] Google Inc. Official Android Hangouts App Page. https://play.google.com/store/apps/details?id=com.google.android.talk, 2016. [Online; accessed 16-Oct-2016].

[11] Google Inc. Official Android Maps App Page. https://play.google.com/store/apps/details?id=com.google.android.apps.maps, 2016. [Online; accessed 16-Oct-2016].

[12] Google Inc. Official Android YouTube App Page. https://play.google.com/store/apps/details?id=com.google.android.youtube, 2016. [Online; accessed 16-Oct-2016].

[13] Google Inc. Android Studio home page. https://developer.android.com/studio/index.html, 2017. [Online, accessed 20-May-2017].

[14] Google Inc. Google Play Console home page. https://developer.android.com/distribute/console/index.html, 2017. [Online; accessed 15-Jun-2017].

[15] A. Guenther et al. Cal Poly CSC Template. http://www.github.com/CalPoly, 2016. [Online; accessed 27-May-2017].

[16] JetBrains s.r.o. IntelliJ IDEA home page. https://www.jetbrains.com/idea, 2017. [Online, accessed 20-May-2017].

[17] K. Owen and D. Janzen. Android App Course v3. https://sites.google.com/site/androidappcoursev3, 2013. [Online, accessed 29-Jan-2017].

[18] J. Reed. Contextual Android Education. Master's thesis, California Polytechnic State University, San Luis Obispo, 2010. [Online; accessed 16-Oct-2016].

[19] J. Reed and D. Janzen. Android Course Archive. https://sites.google.com/site/androidcoursearchive, 2010. [Online, accessed 29-Jan-2017].

[20] J. Reed and D. Janzen. Android Course Archive 2. https://sites.google.com/site/androidappcourse, 2010. [Online, accessed 29-Jan-2017].

[21] Samsung Electronics Co. Ltd. Samsung Galaxy S Launches with Industry-leading Technological Innovations. https://www.samsungmobilepress.com/press/Samsung-Galaxy-S-Launches-with-Industry-leading-Technological-Innovations?2010-06-02, 2010. [Online; accessed 16-Oct-2016].

[22] Samsung Electronics Co. Ltd. Samsung and Google introduce Galaxy Nexus. https://www.samsungmobilepress.com/press/Samsung-and-Google-introduce-GALAXY-Nexus-1?2011-10-19, 2011. [Online; accessed 7-Jan-2017].

[23] Samsung Electronics Co. Ltd. Samsung Unveils Next Generation Smartphones at the Core of its Connected Galaxy Experience. https://www.samsungmobilepress.com/press/Samsung-Unveils-Next-Generation-Smartphones-at-the-Core-of-its–Connected-Galaxy-Experience?2016-02-22, 2016. [Online; accessed 16-Oct-2016].

[24] Samsung Electronics Co. Ltd. Samsung Galaxy S Series home page. http://www.samsung.com/us/mobile/phones/galaxy-s, 2017. [Online; accessed 15-Jun-2017].

[25] SurveyMonkey. SurveyMonkey home page. https://www.surveymonkey.com, 2017. [Online, accessed 26-Feb-2017].

[26] T-Mobile Inc. T-Mobile Unveils the T-Mobile G1 - The First Phone Powered by Android. https://newsroom.t-mobile.com/news-and-blogs/t-mobile-unveils-the-t-mobile-g1-the-first-phone-powered-by-android.htm, 2008. [Online; accessed 16-Oct-2016].

[27] The Eclipse Foundation. Eclipse IDE home page. https://eclipse.org, 2017. [Online, accessed 29-Jan-2017].

[28] Verizon Wireless. Hello Humans: Droid By Motorola Arrives Next Week. http://www.verizonwireless.com/news/article/2009/10/pr2009-10-27.html, 2009. [Online; accessed 16-Oct-2016].

[29] Verizon Wireless. Verizon Wireless And Motorola Mobility Announce Motorola XOOM Tablet On Nation's Largest And Most Reliable 3G Network. http://www.verizonwireless.com/news/article/2011/01/pr2011-01-05k.html, 2010. [Online; accessed 16-Oct-2016].

[30] J. Wharton. ActionBarSherlock home page. http://actionbarsherlock.com, 2012. [Online, accessed 8-Feb-2017].