Dartmouth CS 65/165

Smartphone Programming **Professor Andrew T. Campbell**

Want to be the next Nick D'Aloisio who at the ripe old age of 17 sold his smartphone app to Yahoo for $27 million?

Want to know the ins and outs of rolling your own Android app?

Want to develop your own app as part of a group project?

Then read on ....

[If you are not a Dartmouth student feel free to use the material found below. All the notes and code are online. Let me know what you think]

AndroiD

The coolest technology to emerge over the last decade is the smartphone -- it's had a huge impact on our lives. Today, over half of all Americans own a smartphone and I bet you are reading this webpage on your phone.

If you've been curious about programming phones and creating great apps then this course is for you. We'll focus on programming Android phones -- sorry iPhone.

The course has no exams or midterms -- it's all about programming. But there will be a lot of programming. The prerequisite for this course is Computer Science 10 (no exceptions given).

Because we plan to give each student a Google nexus 4 (thanks to the Neukom Institute and Computer Science department) for the duration of the course we have capped the course. However, if we exceed the cap we might be able to accommodate more students -- I'm assuming a good chunk of the class own an Android.

Lectures

- Location: 001 Rockefeller
- Monday Wednesday Friday 11:15-12.20 AM
- Tuesday 12:00-12:50 x-period
- Office hours: Sudikoff 260 Monday and Friday 4-5 pm

Teaching assistants

Fanglin.Chen.GR@dartmouth.edu. Office hours: 1 -2:30pm Monday and Tuesday and 5-6:30pm Friday - Sudikoff 220

Lixing.Lian.GR@dartmouth.edu. Office hours: 2-3:30 pm Tuesday, Wednesday, and Thursday - Sudikoff 202.

Both TAs will also cover lab hours Monday 6-9 PM in L001

What we'll teach you

The goal of this course is to teach students how to design, implement, test, debug and publish smartphone applications on java based android phones. Students will learn how to take their innovative ideas from conception to the android market through a series of rigorous hands-on programming assignments and group projects.

This is an introductory course aimed at undergraduate students (but graduate students are most

welcome) who have Java programming experience. However, there is a significant amount of programming in this course requiring a commitment on the part of the student.

Topics covered: the android development environment including eclipse, android SDK and phone emulator; key programming paradigms; UI design including views and activities; data persistence including SQLite; content providers; messaging and networking; phone sensors, location based services (e.g., Google Maps), background services; broadcast receivers; cloud programming using App Engine; and publishing applications to the android market.

Android programming concepts are reinforced through a set of thematic programming exercises that introduce these topics and incrementally allow the student to build a complex application; that is, programming labs form a set of components that collectively implement a continuous sensing application. The resulting phone app allows user to log their exercises (e.g., walks, runs) and display them on Google maps.

A key part of this course is group projects where students will work in small teams for joint problem solving.

Prerequisites

Computer Science 10 (no exceptions). Android is based on Java programming.

We are using Piazza for information dissemination and Q&A

Sign up to CS65 Piazza The best way to get answers to programming problems is crowdsourcing -- so the first time someone comes across a problem and we know the answer - everyone sees it. I encourage students to actively use piazza. It's a great system.

Grading

There are no exams or mid terms in this class -- it's all about hacking.

50% - Thematic programming exercises

There are 6 weekly programming assignments over the first 7 weeks. These labs are designed to help students learn the android programming environment and key programming paradigms. Assignments are done individually. Each lab will receive the same percentage of the grade.

Each student will demonstrate their standalone MyRuns5 (after lab 5 is complete). We will meet on the green to assess your app. You will be asked to demo two activities: walking around the green and running (if you don't feel up to running we will find someone to run with your phone).

Your MyRuns5 app should correctly capture the activity (walking and running) and the details of the exercise. Andrew should be able to see the exercised saved in your history tab displayed on Google Maps.

Labs are graded on completeness of the required features, the correctness of the functionality, and the robustness: **note, if your lab crashes it will not be graded and you will have to resubmit a working version with 20% penalty against your lab grade.**

20% - Short programming exercises

There will be a number of short programming exercises that will either be completed during a class period or submitted at midnight on the day that the assignment is made. Most of these will be groups of two. So find a coding buddy now. The grading is simple. If you get it functionally correct when we check the apk you get 100% else you get 50%. We will not return comments on this code. It either works or not.

Note that these coding exercises will not be scheduled in advance and can only be taken in class at the designated time.

30% - Group projects

Students will develop their own app.

Projects are made up of a small (2 people) teams and require strong collaboration and a problem solving mindset. The goals of this activity are to help you develop the confidence, skills, and habits necessary to write real phone apps while part of a multi-person team.

Each team member will get the same grade assuming all goes well. The grade will breakdown as follows:

Submission of programming assignments

We are using Subversion Version Control (SVN) for the submission of programming assignments. You should read those note before proceeding.

SVN is a tool for source code management. Your svn repository root is at https://svn.cs.dartmouth.edu/classes/cs65-S13/Your_Name/ so for me: https://svn.cs.dartmouth.edu/classes/cs65-S13/Andrew_Campbell

(Note, for classes after 2013 this is relevant: change cs65-s13 to the correct year and term for example W14 changes the svn commands below to cs65-W14)

How are we going to submit programming labs using SVN?

- Create a directory for each new lab in your root repository directory (viz. Lab1_submission, Lab2_submission, Lab3_submission, Lab4_submission, Lab5_submission, Lab6_submission) -- for example, https://svn.cs.dartmouth.edu/classes/cs65-S13/Andrew_Campbell
- Commit the following files for each lab.

Policy for late assignments

You should always aim to get your lab in on time.

```
Penalties: < 8 hours: 10%; < 24 hours: 20%; < 48 hours: 40%; more: no credit.
```

Under extenuating circumstances, alternative arrangements may be made with the instructor, if possible before the due date.

Our late policy is a bit harsh but here's the reasoning. Each lab builds on the previous lab so falling behind is a taxation unto itself. To balance this late policy I add this sweetner: **you get one free 12 hour pass and 1 free 24 hour pass without penalties**. These are atomic units -- you can't divide them up, before you ask. Take it when you like but put a note in your README.txt file and email the TA.

It is not good to fall behind because it limits the time you can spend on the next lab which comes out as you finish the currenmt one. Therefore, three final pieces of advice from CS1:

```
Don't fall behind in this course.
Don't fall behind in this course.
Don't fall behind in this course.
```

MyRuns project document

The MyRuns document captures the complete app specification and lab assignment information. Please read this a couple of times and refer back to it.

Week 1 (Jan 5-9)

We have updated these notes to Lollipop and the new Android Studio IDE. But you will need to take notes in class because in class I am using other but similar examples. Best to pull up your Studio and code as we go.

- Introduction
- Lecture 1: Getting started
- Lecture 2: The wonderful world of Android
- Lecture 3: Our First Android Application
- Lecture 4: Activity Lifecycle

Week 2 (Jan 12-16)

- OUT MyRuns_Lab_1: User Profile
- Lecture 5: A very cool activity lifecycle app to play with
- Lecture 6: User Interface I
- Lecture 7: User Interface II
- Lecture 8: Using the Camera and Data Storage

Week 3 (Jan 19-23)

- IN MyRuns_Lab_1: User Profile - Monday before midnight
- OUT MyRuns_Lab_2: User Interface
- Lecture 9: Fragments and ActionBars
- Lecture 10: Debugging
- Lecture 11: Dynamic Layouts using the Fragment Manager
- Lecture 13: Using PreferenceFragment to store user preferences
- Lecture 14: Customizing Dialogs with DialogFragment

Week 4 (Jan 26-30)

- IN: MyRuns_Lab_2: User Interface - Monday before midnight

Android programming books

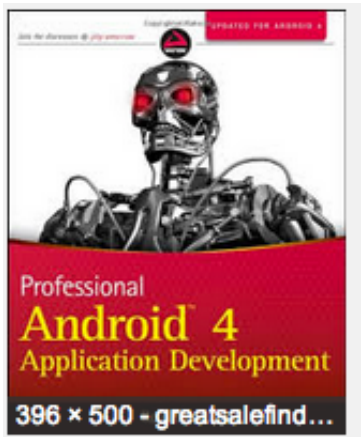
We do not recommend any text in particular because Android is such a moving target and many books are out of date soon after they are published. Android developer and other web resources are the most up to date. However, here are some suggestions if you would like a reference book.

The Busy Coder's Guide to Android Development

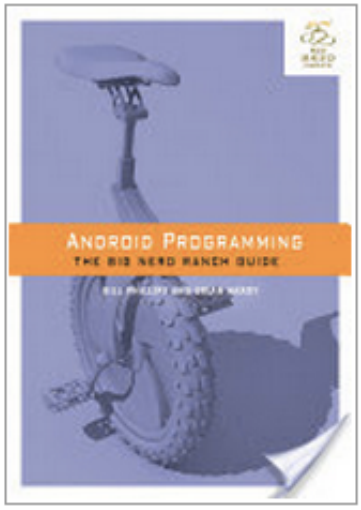
It is an online book $45.00.

I like the online book because you pay once and you get the revised versions for free (sort of). This is important because the APIs are still fluid. Many APIs are being added and depreciated at the same time. So the book at least quickly tracks major changes.

This is also excellent if now a little dated. The author is from the Android team so it represents a nice reference.



Finally, this book is not a reference but a boot camp. We will use some coded examples from this nice book:



Having all three books might seem excessive -- call me excessive.