Vanier College — Faculty of Science and Technology
Computer Science Department

Graphical and Wireless Programming

420-408-VA — Winter 2020

Section:	00001 & 00002	**Teacher:**	Sleiman Rabah
Ponderation:	3-4-4 (Theory-Lab-Homework)	**Office:**	K-316
		Hours:	By appointment
Pre-requisite:	420-300-VA (Data Structure)	**Phone:**	514.744.7500 x 8334
		E-mail:	MIO (Messaging in Omnivox)
Semester:	Winter 2020		

Course Description

This course is an introduction to the design and implementation of mobile graphical user interface (GUI), and wireless programming. This will include simple coverage of an adequate integrated development environment (IDE) supporting the main features needed to develop mobile applications including mobile phone emulators.
Topics covered in this course include, Android activities, fragments, dialog boxes, event-driven programming, custom widgets, SQLite databases, content providers, menus, messaging and networking, REST-based APIs and JSON, location-based services, and quick introduction to multimedia, graphics and animation.

Learning Objectives

1. Master the theoretical and practical concepts of mobile applications development using Android SDK (Standard Development Kit) and Android Studio IDE (Integrated Development Environment).
2. Use XML, Java, libraries and tools to implement GUI (Graphical User Interfaces) for mobile applications.
3. Design and implement a complete Android application that incorporates most of the following: SQLite databases, RESTful Web Services and JSON, fragments, multimedia, animation, location-based services and Google maps, etc.

COMPETENCIES RELATED TO THIS COURSE

The following competencies are addressed by this course. It should be noted that other courses develop some of these competencies in more details.

Number	Description
016X	Develop a user interface
016S	Use a structured programming language
016W	Produce algorithms
017B	Design and develop an application in a database environment
017C	Design and develop an application in a graphics environment
017D	Design and develop a hypermedia application within internal and global networks

Required Background and Prerequisite Knowledge

The practical programming components of this course require the use of the Java programming language and XML. Therefore, students are required to have a reasonably solid knowledge of both languages. The official prerequisite for this course is Data Structures (420-300-VA). Much of the material covered in Programming I & II as well as Data Structures is directly relevant for this course, therefore, it is critical that students understand and remember the knowledge they acquired in previous courses. Some of the essential concepts that students should know well to do well in this course include, but not limited, to the following:

- Core principles of Object-Oriented Programming (OOP)
- Client/server communication model, HTTP protocol and REST API. That is, implementing an HTTP client, creating HTTP requests and processing HTTP responses, etc.
- Event-driven programming model: generating and handling events in a GUI context
- Common data structures such as list, array, hash map, stack, queue, etc.
- Relational databases concepts. The design and implementation, as well as usae of relational database
- Data description languages: creating and parsing well-structured XML and JSON documents
- Declarative user interface programming approach using HTML

In summary, the important skills needed to do well in this course include: (1) good programming, debugging and problem solving skills (including the ability to understand, design, implement, and debug programs with non-obvious flow of control) (2) the ability to relate the theoretical material covered in class to the practical aspects of implementation, (3) time management, team work and excellent communication skills (any heated discussion between teammembers is strictly prohibited) and (4) the ability to abstract the knowledge learnt and apply it to a wide-range of problems (not necessarily related to Android programming, or even Computer Science for that matter).

Course Workload

This is a reasonably heavy course with several new concepts, and quite a lot of programming workload. Therefore, students should be prepared to spend adequate time and effort in this course. Moreover, it is assumed that students have a reasonably solid knowledge with object oriented programming. If judged necessary, a quick review of Java and OOP principles will be provided during the early lectures. Hence, it is very important that students attend the first week of classes.

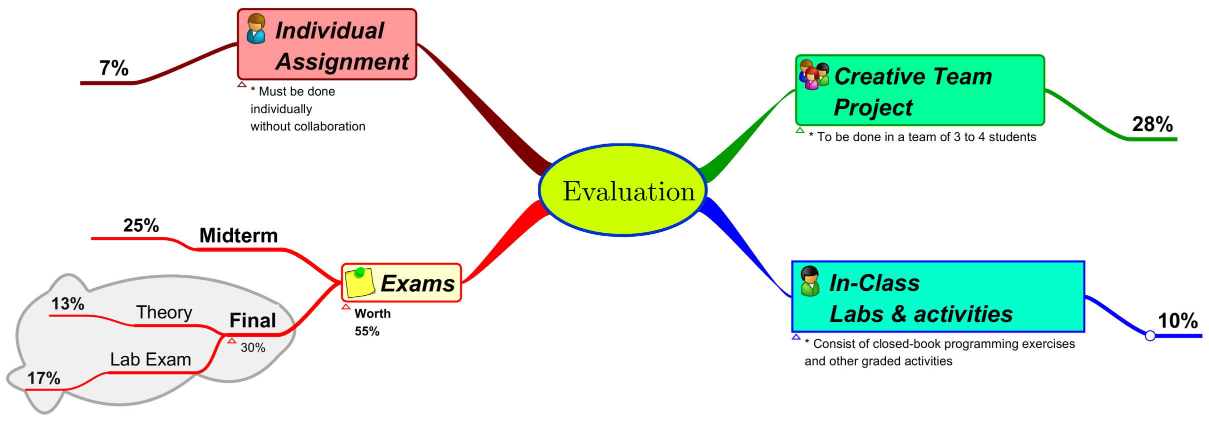
Course Materials

- **textbook:** there is no mandatory course required for this course. However, a list of recommended textbooks is provided in Section 20.
- **Online Resources:** such as APIs documentation and tutorials will be provided to students throughout the semester.
- **In-Class Notes:** additional resources such as learning strategies, in-class notes, as well as in-class activities to be done in groups will be also provided throughout the semester. Students are required to attend and take notes during lectures.

Teaching Methods and Course Structure

The course ponderation is 3-4-4. Course ponderation means three hours per week for learning new concepts and mobile programming topics during the lecture time; four hours per week for applying these concepts and do some in-class exercises in the computer lab; and finally, 4 hours per week to practice, do homeworks and try the examples given in class.

Grading scheme



Evaluation
Individual Assignment
7%
* Must be done individually without collaboration
Creative Team Project
28%
* To be done in a team of 3 to 4 students
Exams
Worth 55%
Midterm
25%
Final
30%
Theory
13%
Lab Exam
17%
In-Class Labs & activities
10%
* Consist of closed-book programming exercises and other graded activities

- ☞ Exams, quizzes, and in-class labs are **closed book**. Therefore, no documentation of any kind is allowed (unless previously specified by your teacher).
- ☞ Students will be informed of the exact date of each test at least one week in advance.
- ☞ To pass the course, **students must obtain:** 1) an overall passing grade **and** 2) a passing grade on the average of the tests (that is, 60% or higher).
- ☞ Late assignments are **NOT** accepted. Please do not send your late assignments by MIO.
- ☞ No makeup test will be given if students miss an exam, but students with justifiable absences (such as an emergency medical condition) must submit a medical note.
- ☞ Students handing in medical notes that are found to be fraudulent will face severe sanctions as stated in the Student Code of Discipline. The professional unsuitability policy could also be used to sanction the student.
- ☞ In case of a personal emergency, or difficulty in the course, do not hesitate to consult the teacher for help and suggestions.

Assignments and Team Project:

1. Assignments and in-class labs must be completed **INDIVIDUALLY** unless explicitly specified by your teacher.
2. In order to obtain full marks, regardless whether a graded component was implemented individually or in team, students must keep in mind the following:
   (a) Every individual student is responsible for understanding the entire assignment.
   (b) Students will be asked to demonstrate their assignments after the submission deadline. Students are expected to demonstrate complete understanding of the entire assignment.

Creative Team Project

In this course, students are going to learn many different technologies to build Android mobile applications. However, it can be de-motivating to build something to a specification, especially when students are learning something new and want to make something that has their own flavor.
To give students a chance to play around with the technologies they are learning, they will be asked to design and implement a mobile application from scratch, and entirely of their own design. Each week, students will be asked to meet some requirements. However, the requirements do not force students to have any particular content, features layout, color scheme, feel, etc – students will define those requirements themselves.

Required Software and Tools

- Java 7 or 8 (32 or 64 bits)
- OS: Windows 10
- Android Studio IDE (most recent version), you can get it from here
- Android Software Development Kit (SDK), it is embedded in Android Studio installer. You don't need to install it separately.
- Markdown for writing documentation.
- Distributed version control system.
- Bitbucket: a web-based version control repository hosting service.
- Trello: a Web-based project management system.
- A storage medium (a USB flash memory or any online free storage service such as GDrive or OneDrive) for storing and backing up your files.
- If you want to work at home on your assignments, students must install and use the **same versions** of the required software that are installed in our labs.

PRACTICE LABS

Android SDK and Android Studio are installed in the following labs: D-210, D-241, D-242, D-243 and D-207. Moreover, students can use the computer lab **D-207** to practice, try the tutorials, and work on their assignments.

Course Policies

Attendance

Students are expected to attend all meetings (including lab sessions). Classes will begin and usually end on time. Please do your best to get to class before the start of each and every meeting.

- The Android platform and Android Studio IDE are relatively new topics in the job market and in our Computer Science curriculum. In this regard, students will get many guidelines and help for learning Android mobile programming in class. Therefore, attendance is **strongly recommended** at both lecture and lab periods. Please refer to the Teaching Methodology Section 7.
- Whether they are present or not, students are responsible for all course materials covered: all topics discussed and instructions given during the lecture and labs periods. It is the student's responsibility to be aware of everything discussed in class that is considered being important for the course. Attendance is not part of the grading scheme, but students are responsible for **not missing any in-class assessment** (quizzes, assignments, exams).

Lateness Policy

- **Punctuality:** I do not like lateness. Students are expected to arrive on time at their scheduled classes. Students who are consistently late may be refused entry. You have **5 minutes** to arrive, after which the door will be closed. Late arrivals are disruptive to the learning of your peers. It is a sign of respect for the teacher, as well as the other students, to arrive to class on time, with the proper materials for that class.
- **No late work** will be accepted (assignments, in-class labs, team project). If a deadline approaches and students are not finished with their assigned work, students are encouraged to submit what they have completed by that time.
- **No extensions** to deadlines will be given nor special exemptions will be made for any student for any reason.

Class Policy and Class Guidelines

- Laptops are strictly prohibited in classroom during the lectures and labs.
- **Cell phones STRICTLY PROHIBITED during classes and labs**. Communication and electronic devices must be turned off () and placed in your bags (not on the desk just in front of you). The usage of any of such devices during the class will result in you being asked to immediately leave the class.
- **Headphones** should be removed all the time.
- **During lectures**, students must turn off their monitors and take notes. Using the computer during lectures without authorization is strictly not allowed.
- Video or audio recordings and taking photographs are NOT permitted.
- Persistent talking, whispering or any disruptive attitude will not be tolerated.
  - Students are expected to conduct themselves in a professional manner in both lab and theory periods. Students must be aware of the Student Misconduct in the Classroom (Policy number 7210-19) policy.
  - Do not "surf" the Web during lectures or labs, unless directed to by your instruc-

tor.

- Show respect for school property.
- Do not bring food and drink into the classroom or labs.
- Students are not allowed to install and use illegal software on the hard disk of any computer in the lab. In addition, students are strictly not allowed to watch videos or play games on these computers.
- The use of personal computers during lectures is strongly discouraged. Students are encouraged to take notes during lectures.

Lab Policy

During the lab periods you are expected to work on your assignments. Students are not allowed to watch videos, play games on lab's computers. Anyone caught playing games, watching videos or animes, installing or using illegal software, or eating in any of the labs will be fined $20. Please see the posted notices in lab D-207 for rules and consequences when those rules are not adhered to.

Collaboration Policy

Programming assignments must be completed individually. You may discuss an assignment in general terms with other students, including general discussion of how to approach the problem, but all code you submit must be your own. Any help you receive from classmates should be limited and should never involve details of how to code a solution. You must abide by the following:

- You may not work as a partner with another student on an assignment.
- You may not show another student your solution to an assignment, nor look at their solution.
- You may not have another person "walk through" an assignment, describe in detail how to solve it, or sit with you as you write it. You may also not provide such help to another student. This includes current or former students, tutors, friends, teachers, web site forums, or anyone else.
- Bear in mind that you will be asked to demonstrate your assignments.
- You may not post your homework solutions on a publicly accessible (non-password-protected) web server, during the course or after it has completed. Please discuss with your teacher about acceptable ways to show your work to others.
- Instead of sharing source code or solutions with a classmate, point them to other class resources such as lecture examples, the textbook, or contacting your teacher.
- **You must take reasonable steps to ensure that your work is not copied by others, such as by making sure to log out or lock shared computers, not leaving your external storage device without surveillance, not leaving printouts of your code in public places, and not emailing code to other students or posting it on the web or public forums.**

Information on College Policies

It is the student's responsibility to be familiar with and adhere to all Vanier College Policies. A summary of the course-level policies that apply in this and all other Vanier courses can be found under "*Course-Level Policies*" in Important Vanier Links on Omnivox, or on the **P**edagogical **S**upport and **I**nnovation (PSI) web page that is available at the following link: Policies.

Complete policies can be found on the Vanier College website, under Policies.

Your attention **is drawn in particular** to the following policies: policies on academic complaints; cheating and plagiarism; religious holy day absences; student misconduct in the classroom; **and**, student rights and responsibilities that are stated in IPESA Policy (Section 3.1).

Tentative Course Content

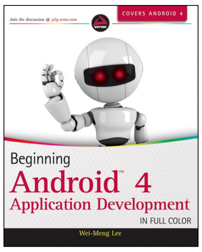
1. **Introduction**
   (a) Course outline discussion.
   (b) Quick Java review: Java basics, OOP-related concepts, GUI programming with Java Swing.
   (c) **Overview of the Android platform** – OS, versions, features, architecture, IDE, SDK, virtual device emulator (AVD), required development tools. Structure of an Android application. Installing Android Studio (*AS*). Getting started with *AS*: configuring devices (AVDs), creating a new Android project, building and running your new project, debugging with *AS* and troubleshooting the emulator.
   (d) Lab: Getting hands on AS. Create and run a very first android application.
2. **Exploring Android application basics**
   Application resources, files, and manifest.
   Understanding activities: Activity class, activity life cycle: life, stages and death, implementing and customizing your own activity. Understanding fragments and Intents.
   Lab: Chapters 1, 2 and 3
3. **Working with fragments and multiple activities**
   Passing intent data as parameters. Lab: Exercises
4. **Creating User Interfaces**
   Understanding graphical controls, layout management, responding to screen orientation changes. Using declarative-based approach to create GUIs. Event-driven programming. Understanding views, designing UIs with views, displaying pictures and menus with views.
   Labs: Create XML-based and Java-based user interfaces - Chapter 4, 5 & 6
5. **Working with menus**
6. **Creating Android multimedia applications**
   Built-in applications using ImageView and built-in date and time applications.
   Working with large applications (ex: data files, XML input data, etc.).
   Display pictures and play sound files.
   Labs: tutorials appearing on chapters 5 and 6 of the textbook
7. **Data persistence.**
   Loading and saving user preferences. Internal/external memory storage (files/SD Card). Write applications accessing SQLite databases and content providers.
   Lab: Tutorials appearing on chapters 7 and 8 of the textbook.
8. **Graphics and animation**
   Android Graphics features for drawing basic shapes (such as lines, rectangles, circles, and text) and for displaying simple animations. Basics on Android animation: drawable, transition, rotation, scale and alpha animation.
   Lab: Tutorials of the textbook (chapter 6) and animation tutorials (lecture notes)

9. **Location-based services** creating, customizing and using MAPs.
10. **Content providers** – sharing data in android, using a content provider. Creating and using your own custom content provider.
11. **Messaging and networking**
    Working with SMS messages. Consuming Web services using HTTP.
    Lab: Tutorials of the textbook - Chapters 9 and 11.
12. **Creative Team Project**
    Write a complete Android application using XML-based user interface, menus, SQLite databases, location-based services with Google maps, graphics, animation, media resources (sound or video) and SMS messages (throughout the semester).
13. **If time permits:**
    Working with Google maps: the Maps API key, adding markers, geocoding, and reverse geocoding.
    Location-based services. Overview of the Android security system. Encrypting devices and data.
    Lab: Tutorials of the textbook - Chapter 10.

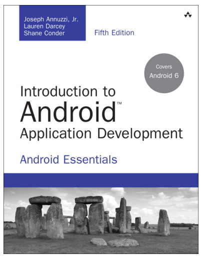
Appendix

List of Recommended Textbooks

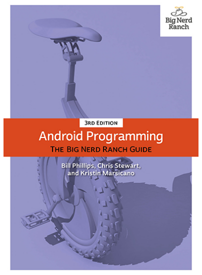
Title: Beginning Android Programming with Android Studio/4E
Author: Jerome F. DiMarzio
Publisher: John Wiley & Sons Inc. (WROX); year: 2016
ISBN-13: 978-1-118-70559-9
Available at: WROX's website and other vendors.
Paperback Price: US $39.99 | **E-book price:** US $25.99



Title: Beginning Android 4 Application Development
Author: Wei-Meng Lee
Publisher: John Wiley & Sons Inc. (WROX); year: 2012
ISBN-13: 978-1-118-19954-1
Available at: WROX's website and other vendors.



Title: Introduction to Android Application Development: Android Essentials/5E
Author: Joseph Annuzzi et al.
Publisher: Addison-Wesley Professional; year: 2016
ISBN-13: 978-0-13-438945-5
Available at: Pearson's website, Amazon and other vendors.



Title: Android Programming: The Big Nerd Ranch Guide/3E
Author: Bill Phillips et al.
Publisher: Big Nerd Ranch Guides; year: 2017
ISBN-13: 978-0-13-47060-5
Available at: Big Nerd Ranch's website and other vendors.

Evaluation Procedures

Course Component	**Overall Weight**	**Tentative Period**
In-Class Labs & activities	10%	(Week 1 to 14)
Individual Assignment	7%	(Week 3)
Creative Team Project	28%	(Week 1 to Week 14)
Midterm Exam	25%	Week 7 or 8
Final Exam	30%	**Week 15**
Theory (13%), practical exam (17%)		