CMPS 312 Mobile Application Development

LAB 1: Setting Up the Android Development Environment and Version Control

---

Objective

1. Learn the basics of Git and GitHub
2. Set up your Android development environment, familiarize yourself with it,
3. Create and debug simple android applications.

Once you've completed this lab, you should have an initial working understanding of the tools developers use when they create Android apps. You should also know how to develop and debug a simple Android application. And how to use git and GitHub version control systems to manage and track changes in your code across versions.

Overview:

This lab is divided into two key parts, designed to provide you with essential skills for Android app development and effective version control using GitHub.

- **Part A: GitHub Introduction :** Get acquainted with GitHub, a collaborative software platform. Create a GitHub repository to house all your lab work, including in-lab assessments and exams submissions.
- **Part B: Android Development Setup :** Set up your Android development environment by installing Android Studio and related tools. Experiment with Android Virtual Devices, the emulator, and develop basic Android applications.

Feel free to ask for help during the lab. Our aim is to ensure you grasp GitHub and establish an efficient Android development environment for successful app creation.

PART A – GitHub

This section will guide you through essential GitHub operations, including handling merge conflicts, ensuring your ability to manage code collaboratively and effectively.

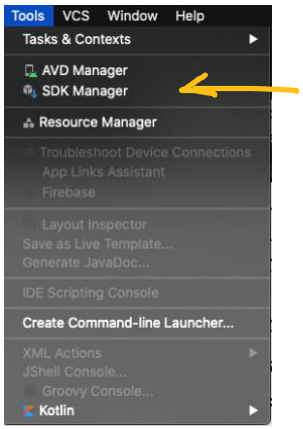
1. Download and Install GitHub Desktop app from https://desktop.github.com
2. While the download is underway, visit https://github.com to create a GitHub Account.
3. Online, create a GitHub repository and clone it to your local machine.
4. Introduce a new file to your repository and make your initial commit.

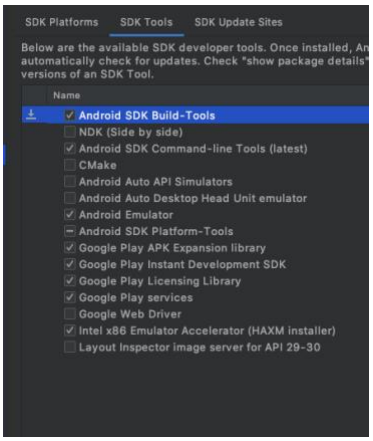
5. Employ the GitHub Desktop app to push your modifications to your GitHub repository.
6. Create a branch in your master repository, make alterations to your files.
7. Attempt to merge the branch into your master repository and address any merge conflicts that arise.
8. Successfully merge the branch into your master repository and push the resolved changes.
9. Initiate a pull request for review and integration, while documenting how you resolved the merge issues.

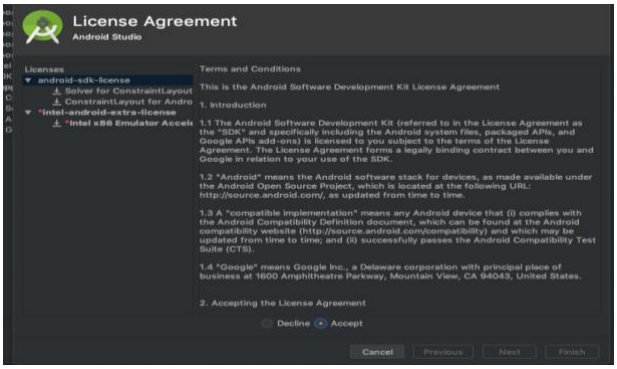
PART B – Android Development Setup

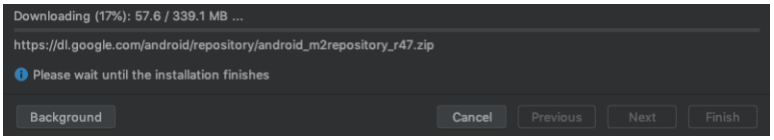
Downloading and Installing Android Studio

1. Download Android Studio from https://developer.android.com/studio/index.html.
2. Install Android Studio on Windows: Run the downloaded **.exe** file and follow the setup wizard for installation, including necessary SDK tools.
3. Install Android Studio on Mac: Open the **.dmg file**, drag Android Studio to Applications, launch it, and follow setup steps, including downloading required Android SDK components.
    a. **Optional**: On Mac, set Android Studio's Java environment path using: export JAVA_HOME=/Applications/Android\ Studio.app/Contents/JRE/JDK/Contents/Home/.
4. Configure SDK Manager: Open Android Studio, go to Tools → Android → SDK Manager, or use the SDK Manager icon if already open.
5. Launch SDK Manager and Install the following if they are not installed already.



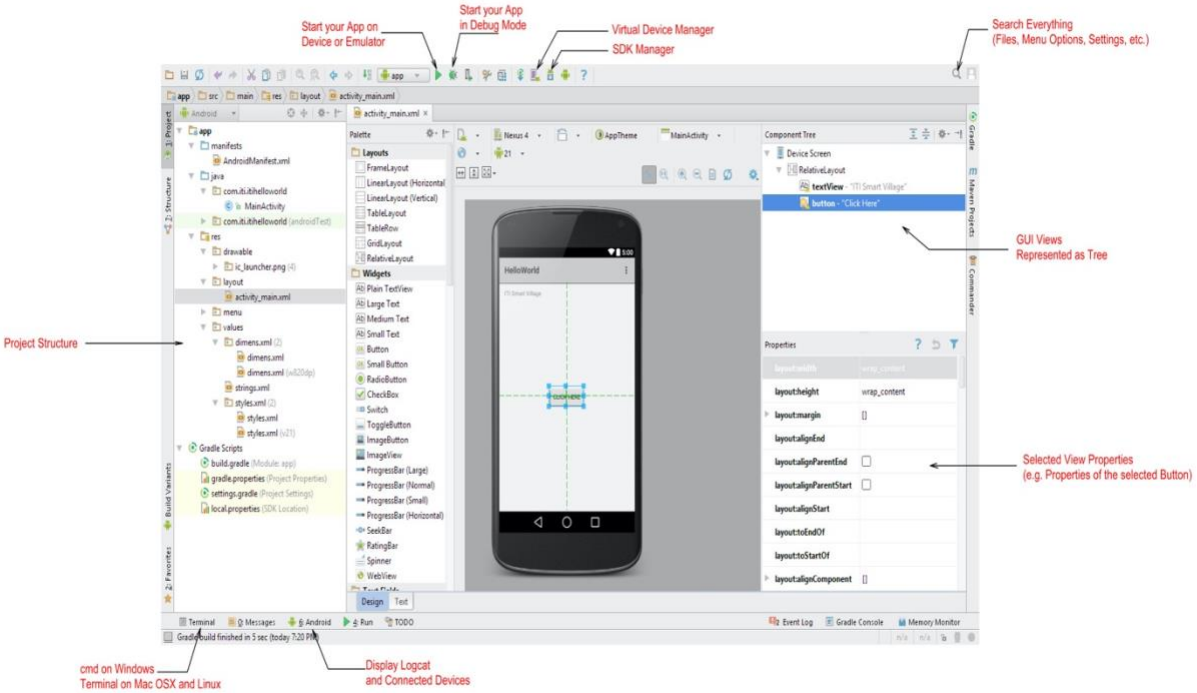






Configuring a project and understanding the IDE

1. Open Android Studio Application
2. Start New Android Studio Project and select Empty Activity
3. Make the application name: "Hello World."
4. Package Name: “com.cmps312.helloworld
5. Make sure the language is **Kotlin and not Java**
6. Choose the Minimum SDK API 16 (**Click on Help Me**)
7. Finish



Running the project

A. Using Real Physical Device

1. Plug the device through a USB port (Windows automatically downloads your drivers)
2. If a window did not download your device's driver, then go to Google "OEM Drivers from android developers" and follow the instructions)
3. Enable the developers' option on your physical phone by :
    i. Go to Settings => "About Your device" => "Build Number, "then press it Seven Times. This will enable the developer's option
    ii. Search for the developers' Options inside the settings
    iii. Tick the USB Debugging Enable box

B. Using Emulator

1. Click on the AVD Manager button on the top right corner of your android studio
2. At the bottom of the menu, click on "Create Virtual Device."
3. You will get three different choices
    i. You can define a new Hardware Profile – Look in the internet
    ii. You can Import a Profile from someone else's AVD
    iii. Use the predefined emulators
    iv. Combination (Best)

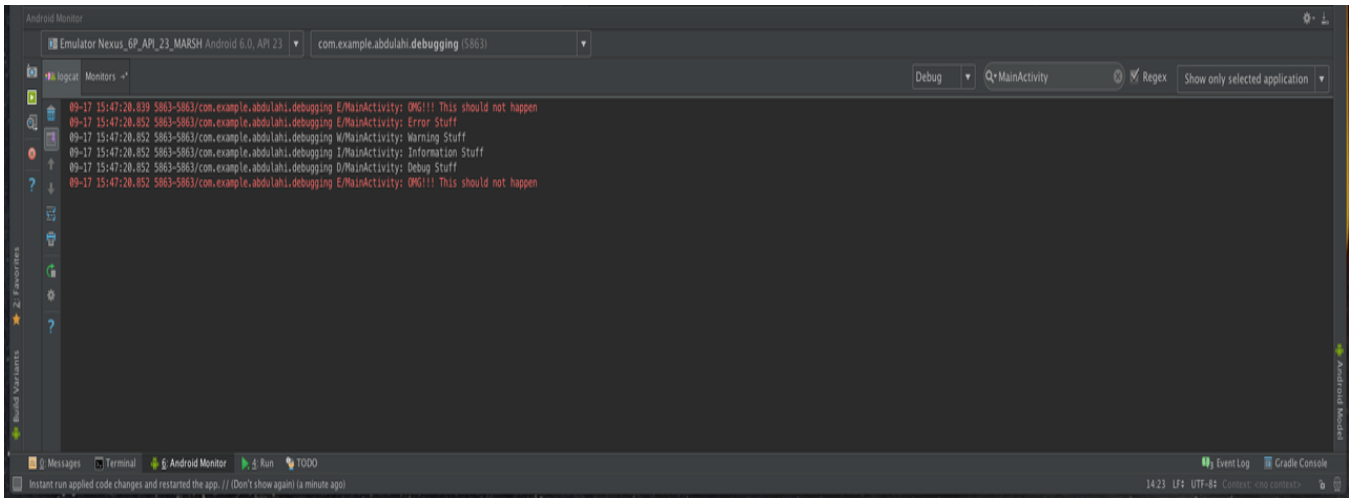
Run the application by pressing the RUN button on the top corner

Debugging Using System Logs

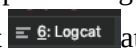
i. Log.v(); // Verbose (If you want to log everything in your app)
ii. Log.d(); // Debug (You want to view some variables values)
iii. Log.i(); // Info (Successfully downloaded the file/ connected)
iv. Log.w(); // Warning (When unexpected thing happened)
v. Log.e(); // Error (Something bad happened ლ(ಠ_ಠლ)
vi. Log.wtf()// (What a Terrible Failure) When something extremely bad happened ヽ(ಠ_ಠ)ノ

Add the above Logs to the Hello World Application

i. val tag = "**MyMainActivity**";
ii. Log.”**X**”(**tag**, "**MESSAGE**”); (Replace the X by the above Log type)
iii. Filter the different Log Messages by using the Android Monitor window



Capturing Screenshots

i. Run your App in Debug Mode.
ii. Click Logcat  at the bottom of the IDE.
iii. On the left corner, expand the two arrows and Click Screen Capture  on the left.

iv. Optional: To add a device frame around your screenshot, click Frame screenshot.
v. Click Save.

Capturing Video

i. Run your App in Debug Mode.
ii. Click Logcat  at the bottom of the IDE.
iii. Click Screen Record  on the left.
iv. Click Start Recording.
v. Interact with your app.
vi. Click Stop Recording.
vii. Enter a file name for the recording and click **OK**.