Setting up Android Studio/Testing

# Installing Android Studio

* Go to <https://developer.android.com/studio/> and click “Download Android Studio”. Install Android Studio.
* Follow the video for installing here: <https://developer.android.com/studio/install> .

# Installing Git

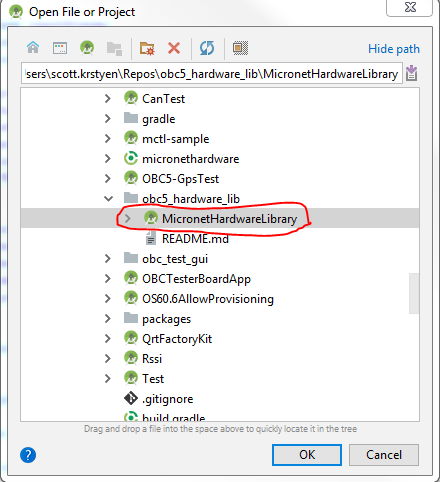
* Go to <https://git-scm.com/downloads> and click “Windows” to download Git.
* Run .exe file to install Git. “Next” through all prompts. Then hit “Install”.

# Download the sources for the OBC5 Micronet Hardware Library

* Click Windows Key + R. Type “cmd” and hit enter.
* Change directory to the place where you want to download the sources.
  + Typing “dir” will display the files/folder in your current directory.
  + Typing “cd..” will take you up a level out of your current folder.
  + If there is a folder in the folder you currently are in (for example if the folder is called “myFolder”, then type “cdmyFolder” to go into that folder.
* In cmd enter “git clone <https://github.com/Micronet-Ltd/obc5_hardware_lib.git>”. The sources will be downloaded to a folder named “obc5\_hardware\_lib” in the folder you are currently in.
* In cmd enter “cd obc5\_hardware\_lib” to go into the folder you just downloaded.
* In cmd enter “git checkout develop” to switch to the development branch from the master branch.

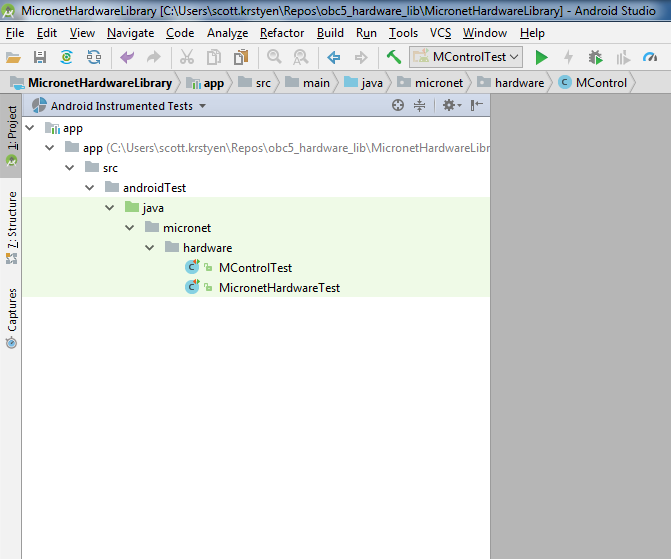
# Open the Project with Android Studio

* In Android Studio, click to open a new project.
* Navigate to the folder where the “obc5\_hardware\_lib” folder is located. In the folder there will be a project folder called “MicronetHardwareLibrary”, select it.



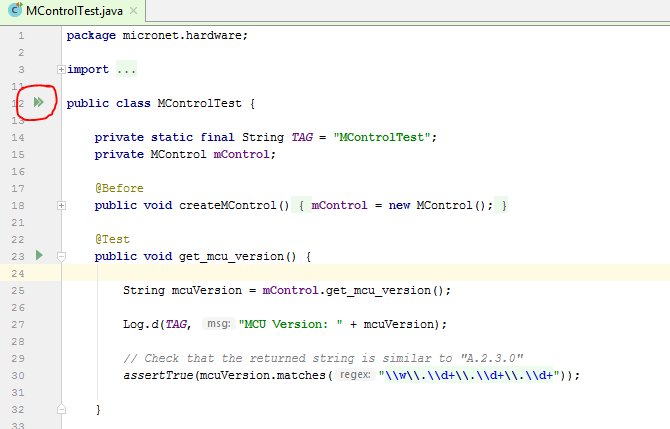
* Click “Ok” to open the project. Android Studio will take a few minutes to load the project.
* You might be prompted to download extra files for Android Studio. You don’t need to download any emulators, but download the other updates (usually something like build tools or sdk versions).
* From the top toolbar, click “Build”->”Make Project”. The project should build without errors. If there are errors, then it is likely that Android Studio will direct you to download some files you need. If you can’t get the project to build without errors then send me an email with a description and screenshot of the errors. If we need to, we can also set up a team viewer to help set it up.

# Opening the Android Tests

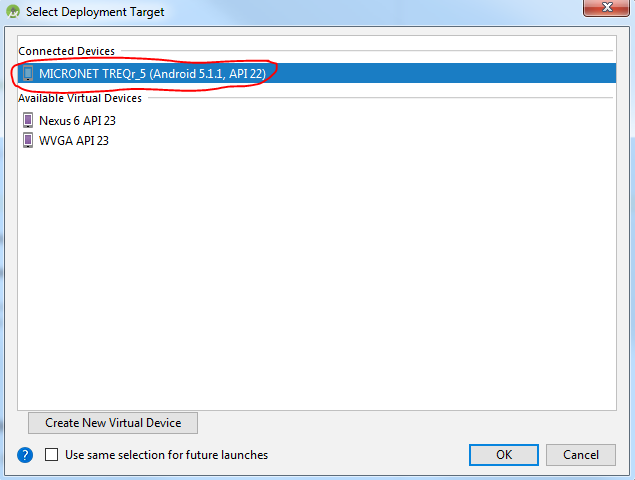
* In the top-left corner of the screen, make sure that this dropdown selector is set to “Android Instrumented Tests”. If it is not, then click it and select “Android Instrumented Tests”.
* Right below the dropdown, click the arrow to the left of the “app” folder. Then click the arrows all the way down to the bottom level.  
  
* Double click on “MicronetHardwareTest” to display it.
* MicronetHardwareTest contains test functions for the OBC5 Micronet Hardware Library where the documentation can be found at the root of the “obc5\_hardware\_lib” folder in the docs folder in “MH\_JavaDoc.zip”. I’ll also attach the Javadoc to the email I send. The Micronet Hardware library uses the MControl library so that is why there are two test files. You should just need to test the Micronet Hardware Library, you shouldn’t need to use the MControl Test file.

# Running the Android Tests

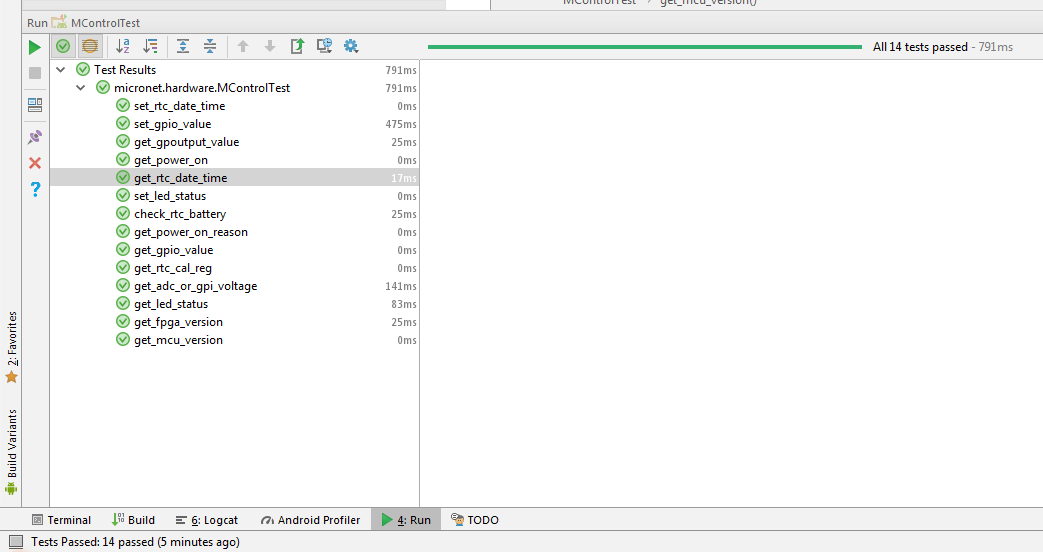
* To run a particular test file, open it the main editor. Near the top of the file next to the line numbers by the class name will be a double green arrow.



* Click the double arrow and select “Run”. You will be prompted which device you want to run it on. Make sure that your device is connected over ADB through the device’s WIFI Hotspot and select that device in the prompt.



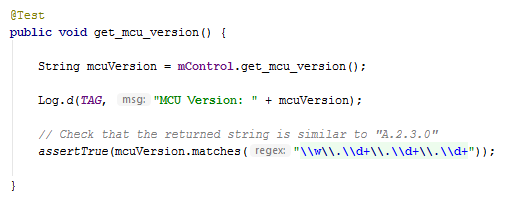
* The results will appear on the bottom half of the window. Individual tests are displayed on the left and they will have a green arrow if they passed and a red if they failed.



* The tests are still in development. Most of the tests don’t do anything yet, so by default they will pass.

# Understanding and Creating Android Tests

* In each test file there are individual tests defined by the functions in the file.



* Each test must be designed as above with a different name; the inside of the function should be different because that is what happens when the test is run.
* Asserts are used to make the test pass or fail. If the test is empty, then it will pass by default. Above you can see that if the return String mcuVersion matches a certain regex then the test will pass, but if it doesn’t then it will fail. There are various types of assert functions that can be found here <https://developer.android.com/reference/junit/framework/Assert> .
* In most of the assert functions you can also add a message that will be printed out if the test fails.