ECE 558 IoT MQTT Project

Assignment version 2.0: Raspberry Pi version
due to GitHub and Canvas (see Deliverables section for details)
on February 26, 2022 at 10:00 PM

Introduction
Project #2 requires you to write an Android-based control app communicating via MQTT with a RaspberryPi connected to sensors and simple electronic components. An MQTT server is hosted on your RaspberryPi, and both the Android and the RaspberryPi are MQTT clients. All the messages are sent over WiFi using the MQTT messaging protocol.

MQTT Overview
MQTT is a messaging protocol where all messages are sent to a central hub, known as the MQTT server or broker. Messages always have a topic. MQTT clients can either publish or subscribe to topics.

We will use the Mosquitto MQTT server (https://mosquitto.org/) and the Paho MQTT client (https://www.eclipse.org/paho/) libraries for this project. These packages from the Eclipse community are widely-used, open-source and have excellent “community” support. There are Paho MQTT client libraries for several programming languages, including Python and Java.

You will set up an MQTT server on the Raspberry Pi. You will then write two MQTT client applications: one in Python which will run on the Raspberry Pi, and one in Kotlin which will run on an Android. Each of these MQTT clients will both publish and subscribe to MQTT channels. Some of the MQTT messages will be commands, which your application will need to respond to appropriately.

The MQTT client on the RPi will **publish** topics for the following:
- Temperature
- Humidity
- Button Status (“pressed” or “not pressed”)

The MQTT client on the RPi will **subscribe** to the following topics:
- LED (turn on or turn off)
- interval (how often to take sensor readings)

The MQTT client on the Android will **publish** topics for the following:
- interval (user-entered value in the Android app)
- LED (the user can choose to turn LED on or off by pressing a button on the Android app)

The MQTT client on the Android will **subscribe** to the following topics:
- Button Status (“pressed” or “not pressed”)
- Temperature
- Humidity

GitHub and GitHub Classroom
Please turn in all code to GitHub Classroom. You may turn in your report (as a PDF or DOC file) to GitHub or to Canvas. You may wish to use GitHub for version control as you work. IntelliJ (and Android Studio which is based on IntelliJ) provides excellent integration with GitHub.

Deliverables
- Project Report (details below)
- Android Studio Project
- MQTT Client Program that runs on the RaspberryPi (written in Python)
- Video or live demo of your project

Project Report
Your project report can be brief, but it must include at minimum the following sections:
- Project Description, including which option you chose for MQTT server, which board you used, and which programming language you used to write the board's MQTT client program.
- Theory of Operation (https://en.wikipedia.org/wiki/Theory_of_operation)
- Link to your GitHub repository

Bill of Materials
- Android
- Android power supply
- Raspberry Pi version 3 or later
- Raspberry Pi power supply
- temperature/humidity sensor and connector
    - Sample project used the Adafruit AHT20 but you may use another if you prefer: https://www.adafruit.com/product/4209
- 8GB (or bigger) microSD card (to set up the Raspberry Pi for the first time)
- microSD card reader/converter to plug it into your computer (to set up the Raspberry Pi for the first time)
- red LED (or color of your choice, but change the value of the resistor accordingly)
- 330 Ω resistor
- push button
- breadboard
- (optional) USB to Serial debug/console cable. These cables provide a serial interface to the Raspian shell and console. https://www.adafruit.com/product/954

Task List
1. Procure the hardware and software needed for the project.
2. Bring up the Raspberry Pi. Follow one of the tutorials (links at the end of this document) to configure a headless Raspberry Pi.

3. Install the Mosquitto server, the MQTT client library for Python, the Python libraries for your temperature/humidity sensor, and your preferred Python development environment.
4. Wire the button, LED, and the temperature/humidity sensor hardware. There is a circuit diagram at the end of this document to get you started. Develop and run test programs to confirm that the devices are operating correctly.
5. Study and understand the command and topic specification provided in the project release.
6. Write the MQTT client to run on the RPi.
7. Verify (and debug if necessary) the MQTT clients using MQTT.fx, MQTT Explorer, or the command line MQTT Publish/Subscribe utilities. I would do this first on the RPi and then remotely with your PC connected to the RPI on your network.
8. Use an MQTT prototyping app like IoT MQTT Panel (there are several in the Google Play Store) to prototype your Android application. Prototyping with an Android IoT MQTT client API will help you debug and test the MQTT communication with the MQTT clients.
9. Develop an Android app that monitors and displays the sensors and controls the actuators. Integrate the Android app with the MQTT server and client running on the RPi. If you have bought up your system using an MQTT client app, your debug should be limited to problems in your application.
10. Prepare and submit your deliverables. Deliverables should include all the source code you developed, a Theory of Operation/Design report and a demo video. Since this is a team-based project, we expect you to do version control with GitHub and GitHub classroom.

Useful Links

- Digital to Analog Conversion with Raspberry Pi https://core-electronics.com.au/tutorials/digital-to-analogue-conversion-with-raspberry-pi.html
- How to Set Up a Headless Raspberry Pi, Without Ever Attaching a Monitor https://www.tomshardware.com/reviews/raspberry-pi-headless-setup-how-to,6028.html
- Setting up a Raspberry Pi Headless https://www.raspberrypi.org/documentation/configuration/wireless/headless.md
- Installing an MQTT server on Raspberry Pi https://pimylifeup.com/raspberry-pi-mosquitto-mqtt-server/
- How to change the mosquitto config file https://mosquitto.org/blog/2020/12/version-2-0-0-released/
- How to make a Mosquitto password file https://mosquitto.org/documentation/authentication-methods/
- MQTT with a Raspberry Pi and an Arduino https://www.youtube.com/watch?v=p3vJxGKWDIg
- Playing with MQTT by Android Part 1: https://www.youtube.com/watch?v=BAkGm02WBc0
- Part 2: https://www.youtube.com/watch?v=6AE4D8INs_U&t=3s
- Paho Android Service – MQTT Client Library Encyclopedia https://www.hivemq.com/blog/mqtt-client-library-enyclopedia-paho-android-service/

- Paho Python – MQTT Client Library Encyclopedia
  https://www.hivemq.com/blog/mqtt-client-library-paho-python/
- Hands-on MQTT Programming with Python by Gaston Hiller
  https://subscription.packtpub.com/book/application_development/9781789138542
- MQTT Essentials - A Lightweight IoT Protocol by Gaston Hiller
  https://subscription.packtpub.com/book/application-development/9781787287815
- Wiring an LED and push button on Raspberry Pi
  https://learn.sparkfun.com/tutorials/raspberry-gpio/all
- Wiring the AHT20 Temperature/Humidity sensor on Raspberry Pi
  https://learn.adafruit.com/adafruit-aht20/python-circuitpython

Notes about the Mosquitto Server on the Raspberry Pi
There is an update to Mosquitto which means that you have to make some changes to the mosquitto.conf file: https://mosquitto.org/blog/2020/12/version-2-0-0-released/

- It's best to make a new mosquitto.conf file and put it in the conf.d folder. Don't touch the original mosquitto.conf file.
- This is a mosquitto.conf file which worked for my application:

```
listener 1883 0.0.0.0
allow_anonymous true

per_listener_settings false
```

- Optional: create a password file following these instructions:
  https://mosquitto.org/documentation/authentication-methods/
  - If you do create a password file, add this line to mosquitto.conf:
    password_file /etc/mosquitto/passwordfile

Some useful terminal commands related to mosquitto:
- sudo systemctl status mosquitto : this tells if the mosquitto is running or not
- sudo systemctl stop mosquitto : this stops the mosquitto process
- mosquitto -c /etc/mosquitto/conf.d/mosquitto.conf : this command starts mosquitto up using your specified mosquitto.conf file instead of the default configuration settings.

Circuit Wiring Diagram