Networking for Software Developers

This is a group lab that will contribute towards your final project. One person will submit this work to the group dropbox.

This is a very short and almost trivial lab.

And yes this is the last lab from your instructor.

# Lab 12 – Send and receive messages via MQTT.

You will implement a client to send structured data to a server. You will also implement a client that will receive the structure data.

You will create three separate files for this lab: a subscriber, a publisher and a util. subscriber.

### Wk12a\_util.py

This file will have the following:

1. A start\_id that will be initially set to 111 or your favourite number. This will be used to number the payload.
2. A create\_data() function that will create and return a dict. This will be the payload.
3. A print\_data() function that will take a dict and print the parts in a human-readable format.

The following is a sample of the data I used. You should not use mine, you must come up with something of comparable complexity.

{

'id': 112, #sequence number

'patient': patient, #name of patient

'time': time.asctime(), #time this was generated

'heart\_rate': int(random.gauss(80, 1)), #heart rate

'respiratory\_rate': int(random.gauss(12,2)), #respiratory rate

'heart\_rate\_variability': 65, #???

'body\_temperature': random.gauss(99, 0.5), #temperature

'blood\_pressure': { #subkey

'systolic': int(random.gauss(105,2)), #systolic pressure

'diastolic': int(random.gauss(70,1)) #diastolic pressure

},

'activity': 'Walking' #activity

}

We can use a class to model our data, but that would be an overkill. Maybe a python dataclass would be a better fit. However a python dict is able to satisfy all of our needs and it is also light weight.

### Wk12a\_publisher.py

This file will have the following:

1. Call the above function to obtain a dict.
2. Convert the above dict to a string. Use json.dumps().
3. Create a client.
4. Connect to the server.
5. Publish to the server.
6. Print a message.
7. Close the connection.

It might be advisable to do multiple transmissions (maybe 10) and sleeping at the end of each cycle.

### Wk12a\_subscriber.py

This file will have the following:

1. Create a client.
2. Assign the on\_messege delegate to the function in Step 7.
3. Connect to the server.
4. Subscribe to the required topic.
5. Print a message.
6. Invoke the client loop\_forever() method.
7. Create a function to do the following: (see signature in text or ppt slide)
   * Decode the message.
   * Convert the decoded string to a dict. Use the json.loads() function.
   * Call the function in the first file to print the dictionary.

## Due:

See course shell

#### Submission

1. You will follow the normal naming pattern for your code file e.g. group\_«your\_group\_number»\_util.py e.g. group\_1\_util.py.
2. Must be uploaded to course dropbox.
3. This is due by the end of week 11.

#### Rubrics

|  |
| --- |
| util |
| [Start\_id] 1/1 |
| [Create\_data] 8/8 |
| [Print\_data] 3/3 |
| [Coding style] 2/2 |
| [Sub-total] 14/14 |
| Publisher |
| [Call create\_data] 2/2 |
| [Convert payload to string] 2/2 |
| [Create client] 2/2 |
| [Connect to server] 2/2 |
| [Publish] 1/1 |
| [Print feedback] 1/1 |
| [Close connection] 1/1 |
| [Repeat] 2/2 |
| [Sub-total] 13/13 |
| Subscriber |
| [Create client] 2/2 |
| [Wireup handler ] 2/2 |
| [Connect to server] 2/2 |
| [Subscribe to topic] 1/1 |
| [Print message] 1/1 |
| [Invoke the loop\_forever] 1/1 |
| [Handler: Decode message] 2/2 |
| [Handler: Convert to dict] 1/1 |
| [Handler: Print message] 1/1 |
| [Sub-total] 13/13 |
| [Total] 40/40 |