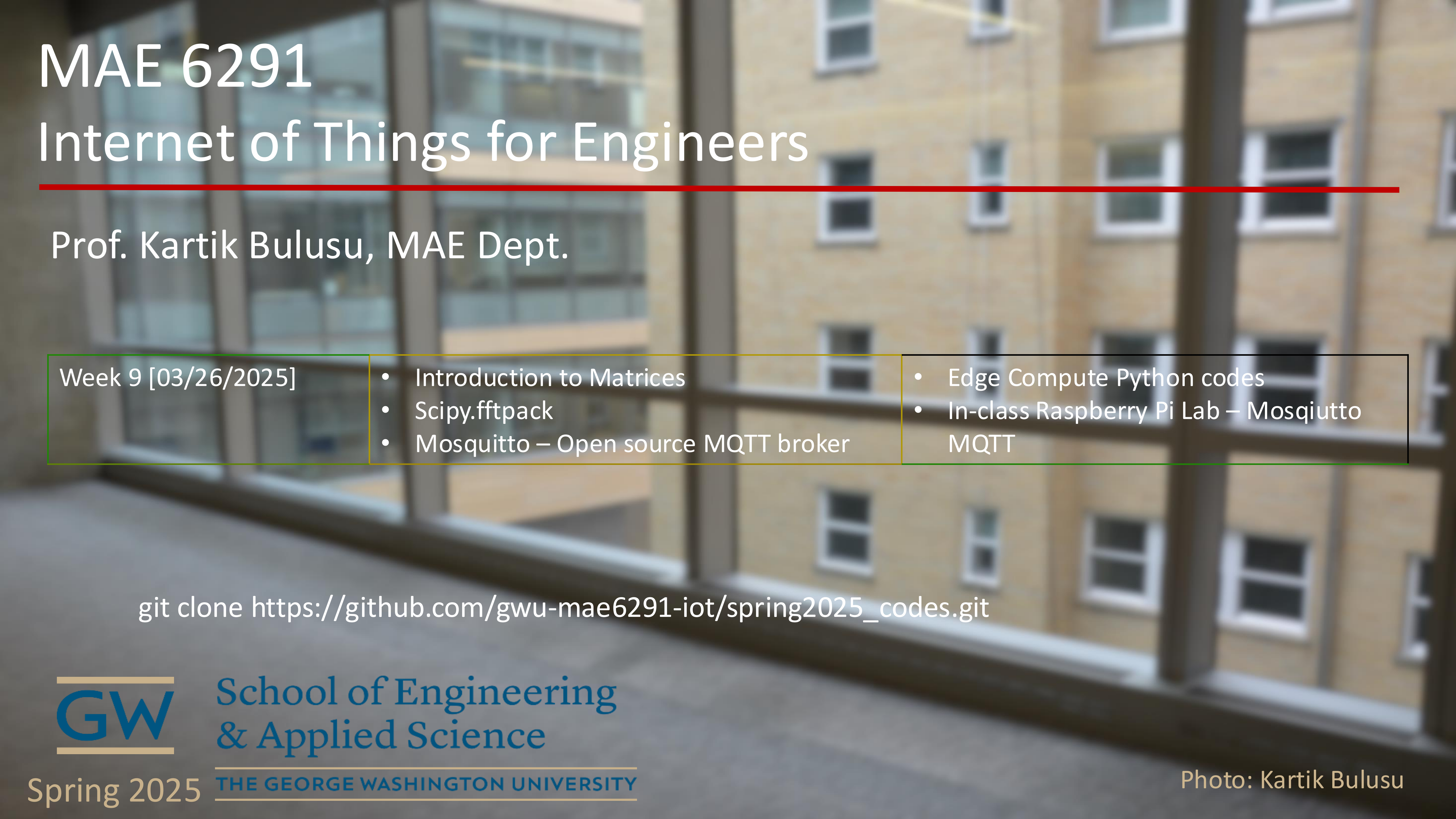


MAE 6291

Internet of Things for Engineers

Prof. Kartik Bulusu, MAE Dept.

Week 9 [03/26/2025]

- Introduction to Matrices
- Scipy.fftpack
- Mosquitto – Open source MQTT broker

- Edge Compute Python codes
- In-class Raspberry Pi Lab – Mosqiutto MQTT

```
git clone https://github.com/gwu-mae6291-iot/spring2025_codes.git
```

School of Engineering & Applied Science
THE GEORGE WASHINGTON UNIVERSITY

Spring 2025

Photo: Kartik Bulusu

Topics to be covered today

Hardware:
SenseHat

Edge computing on the Pi:
FFT + SciPy.fftpack

IoT Strategy:
Intro to MQTT

Expectations on student deliverables:

1. Executive summaries
2. Final project presentation
3. Final project demo
4. Final report in a conference-style template
5. Attendance and no extensions

Icon source:
https://thenounproject.com/icon/clean-energy-4339399/

Two questions up for discussion

1. How should we start perceiving an IoT system, physically ?

2. How / Where do we place the “thing” in that system ?

Keywords:
Small, functional, re-envisioning applications, efficient, sensor-driven, smart-sensors, connectivity, autonomy, data-driven, durability, fault tolerance, interoperability

Proximity to data, compute-power, network, distributed-network etc.

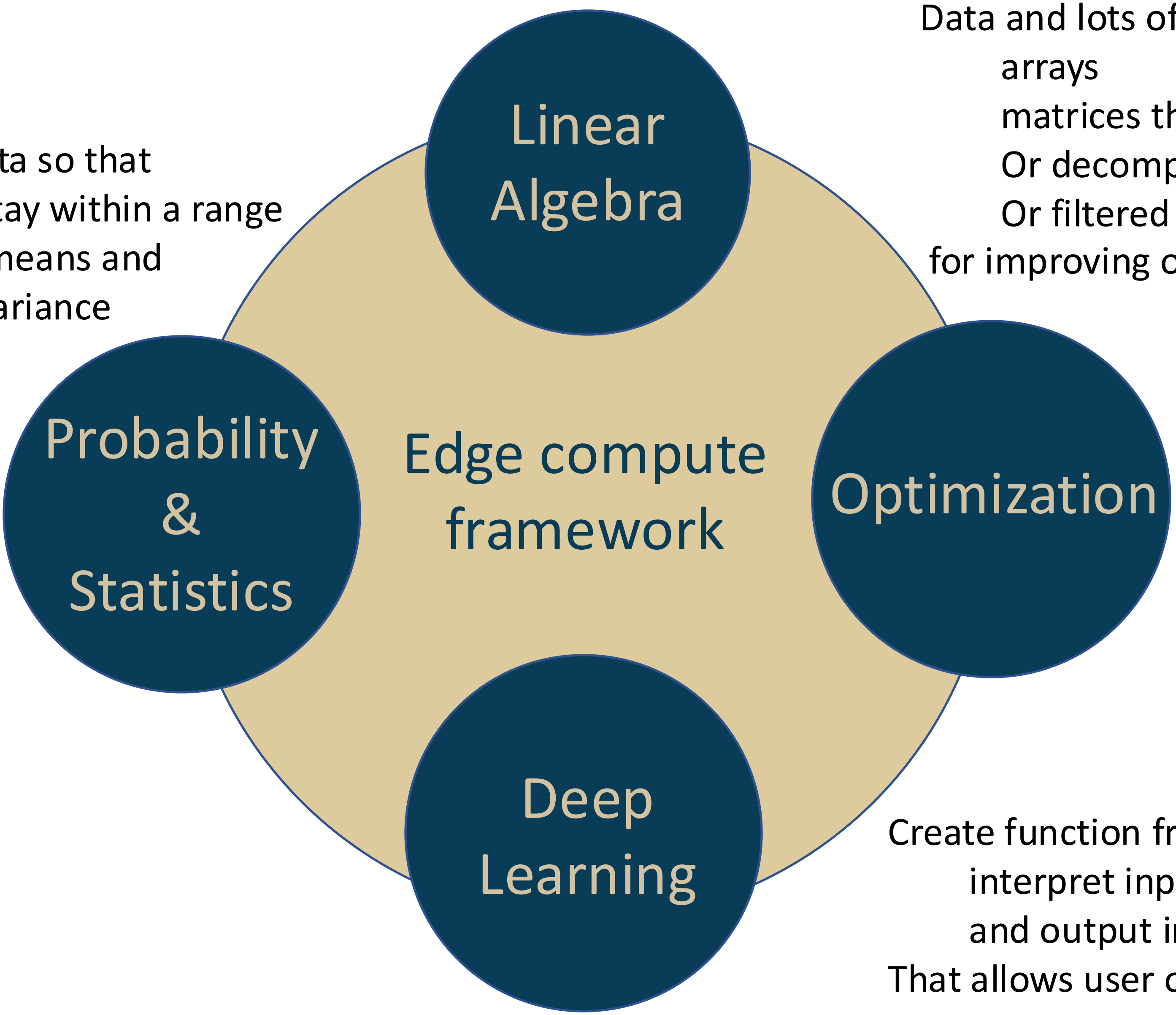
All activities today are a part of graded in-class lab
Download codes from github and demonstrate
[10 points]

Expanding the Edge framework to Fourier Analysis of data

Goal: To use scipy library for signal processing

Source:
https://youtu.be/Cx5Z-OslNWE
Strang, G., Linear Algebra and Learning from Data (2019)

Monitor data so that
they stay within a range
means and
variance



Data and lots of data
arrays
matrices that are special and can be factored
Or decomposed
Or filtered
for improving our understanding of the physical phenomena

Finding matrices
that transform data
and minimize errors
Memory intensive process that usually
happens at cloud-level

Create function from data at cloud-level
interpret input data at edge-level
and output information at edge-level
That allows user or system to take decisions

Source:
https://en.wikipedia.org/wiki/SciPy
https://scipy.org

Explore Signal Processing with Scipy- Python library

SciPy (pronounced /ˈsaɪpaɪ/ "sigh pie"[2]) is a free and open-source Python library used for scientific computing and technical computing.[3]

SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.

What is a Matrix ?

DATA

- Arranged in ROWS and COLUMNS
- Typically carries a MEANING

DATA

- Rectangular ARRAY of numbers

ARRAYS

- Two-dimensional arrays
- *m* rows and *n* columns

Source: http://giphy.com/search/matrix-gif

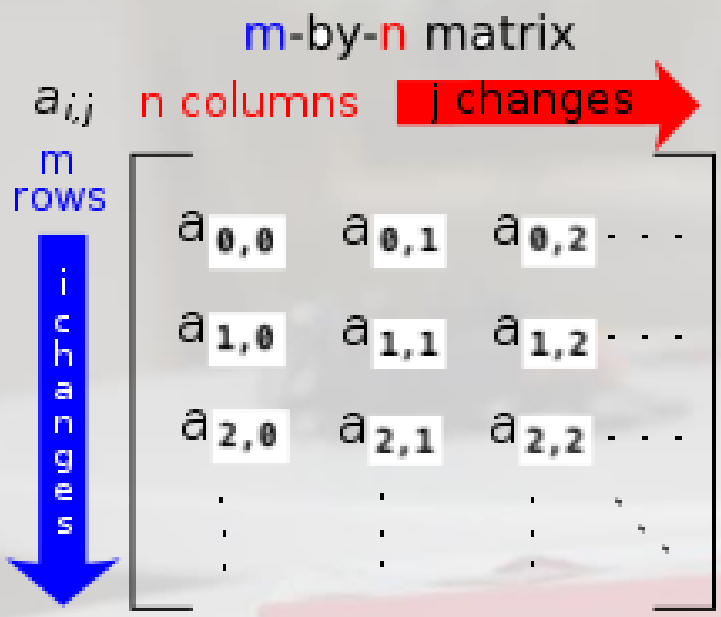
$$\begin{bmatrix} 1 & -4 \\ 9 & 6 \end{bmatrix}$$

$$\begin{bmatrix} 15 & 3 & 9 \\ 2 & 5 & 4 \end{bmatrix}$$

$$\begin{bmatrix} 11 & 7 \\ 4 & 2 \\ 6 & 9 \\ 3 & 1 \end{bmatrix}$$

Photo: Kartik Bulusu

Bookkeeping in a Matrix



$$\begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & \cdots \\ a_{1,0} & a_{1,1} & a_{1,2} & \cdots \\ a_{2,0} & a_{2,1} & a_{2,2} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Source: http://en.wikipedia.org/wiki/Matrix_(mathematics)

Python:

```
>>> import numpy as np
>>> A = np.matrix([[-1, 2],[3, 4]])
>>> A[0,0]
>>> A[0,:]
>>> A[:,0]
>>> A[1,0]
```

The ORDER of a matrix

- $A_{m\times n}$ *is* $m \times n$
- *Read as "m-by-n"*

a_{ij} *is called an ELEMENT*

- *at the ith row and jth column of A*

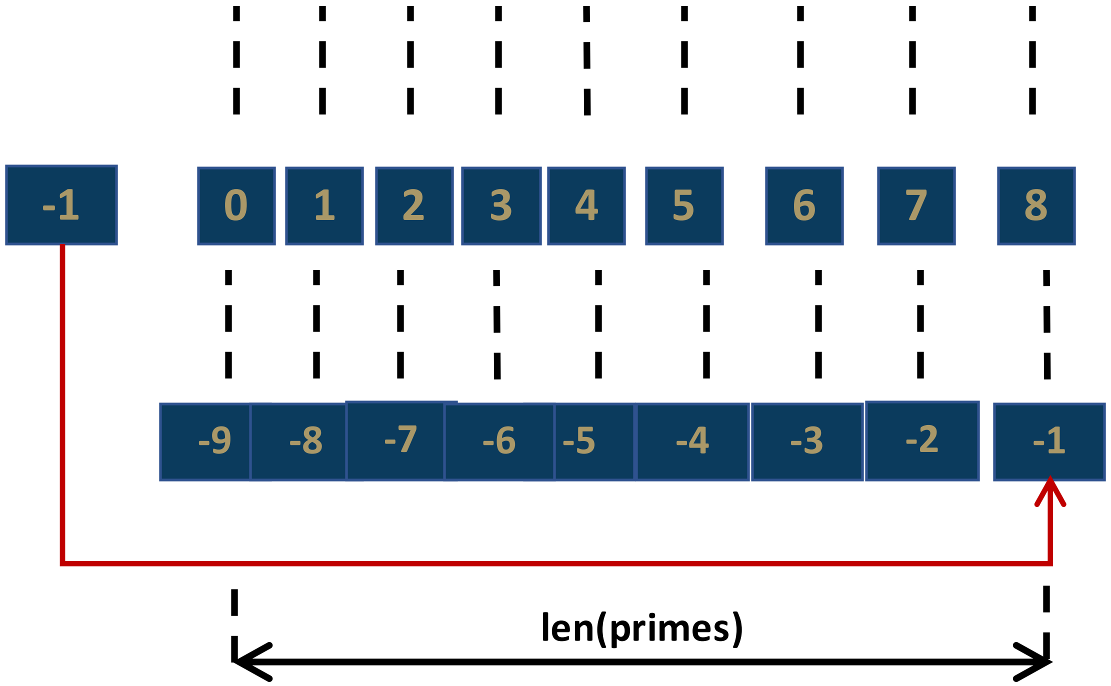
MATLAB:

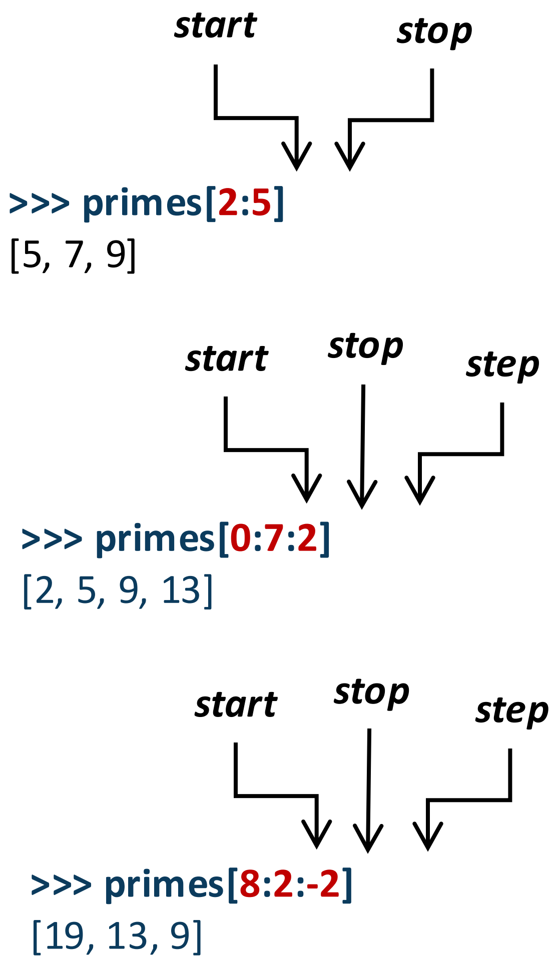
```
>> A = [-1 2; 3 4]
>> A(1,1)
>> A(1,:)
>> A(:,2)
>> A(2,1)
```

Indexing and Slicing Lists

Retrieve list-elements with a range of values

```
>>> primes = [2, 3, 5, 7, 9, 11, 13, 17, 19]
```





```
>>> primes[2:5]
[5, 7, 9]
```

```
>>> primes[0:7:2]
[2, 5, 9, 13]
```

```
>>> primes[8:2:-2]
[19, 13, 9]
```

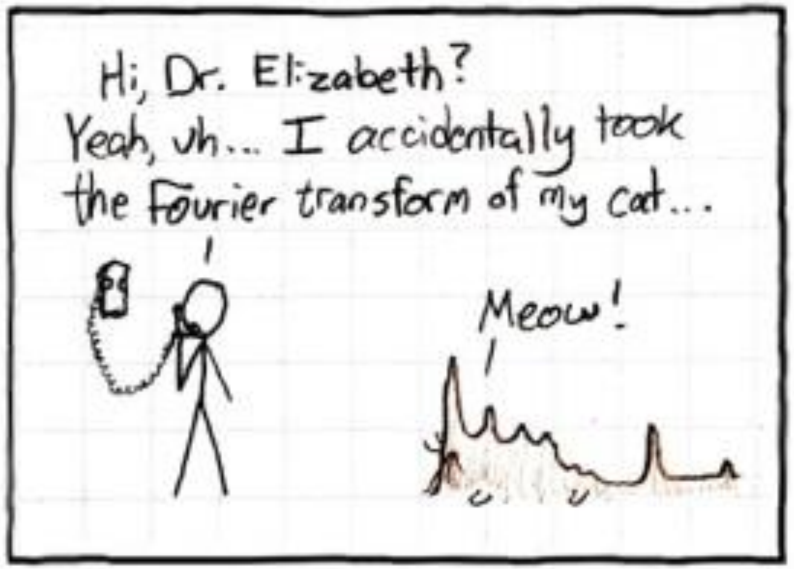
start: at the index value

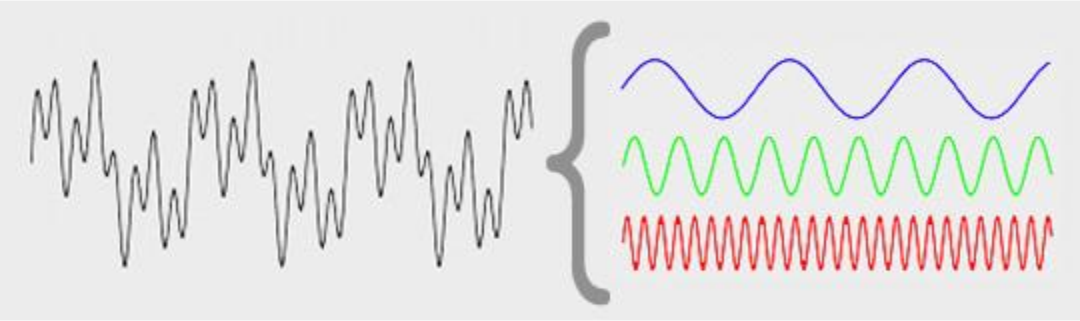
step: up or down at the increment value (default = 1)

stop: at the index value but not including it

Fourier transform: non-mathematical introduction

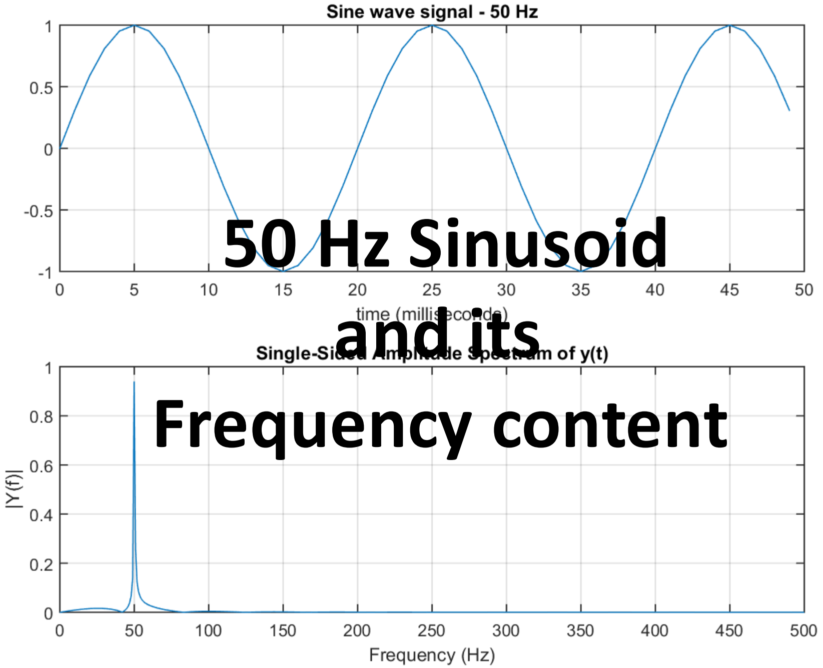
Seeing information in harmonics


Hi, Dr. Elizabeth?
Yeah, uh... I accidentally took the Fourier transform of my cat...
Meow!



Source:
1. http://www.blog.radiator.debacle.us/2013/02/narrative-systems-workflow-using.html
2. http://xkcd.com/26/
3. http://en.wikipedia.org/wiki/File:Joseph_Fourier_%28circa_1820%29.jpg

How do data/signals look in Fourier space ?

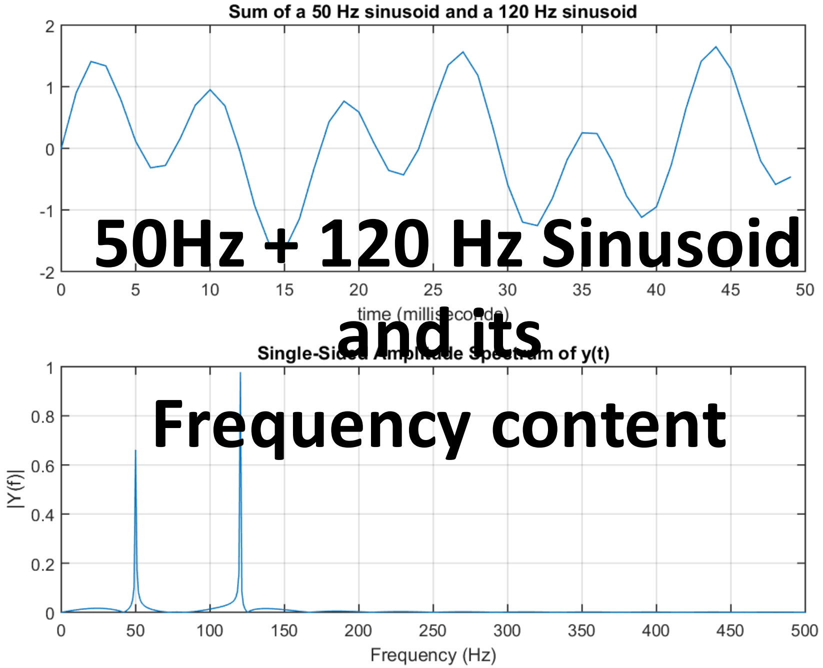

Sine wave signal - 50 Hz

Single-Sided Amplitude Spectrum of y(t)

50 Hz Sinusoid

and its

Frequency content


Sum of a 50 Hz sinusoid and a 120 Hz sinusoid

Single-Sided Amplitude Spectrum of y(t)

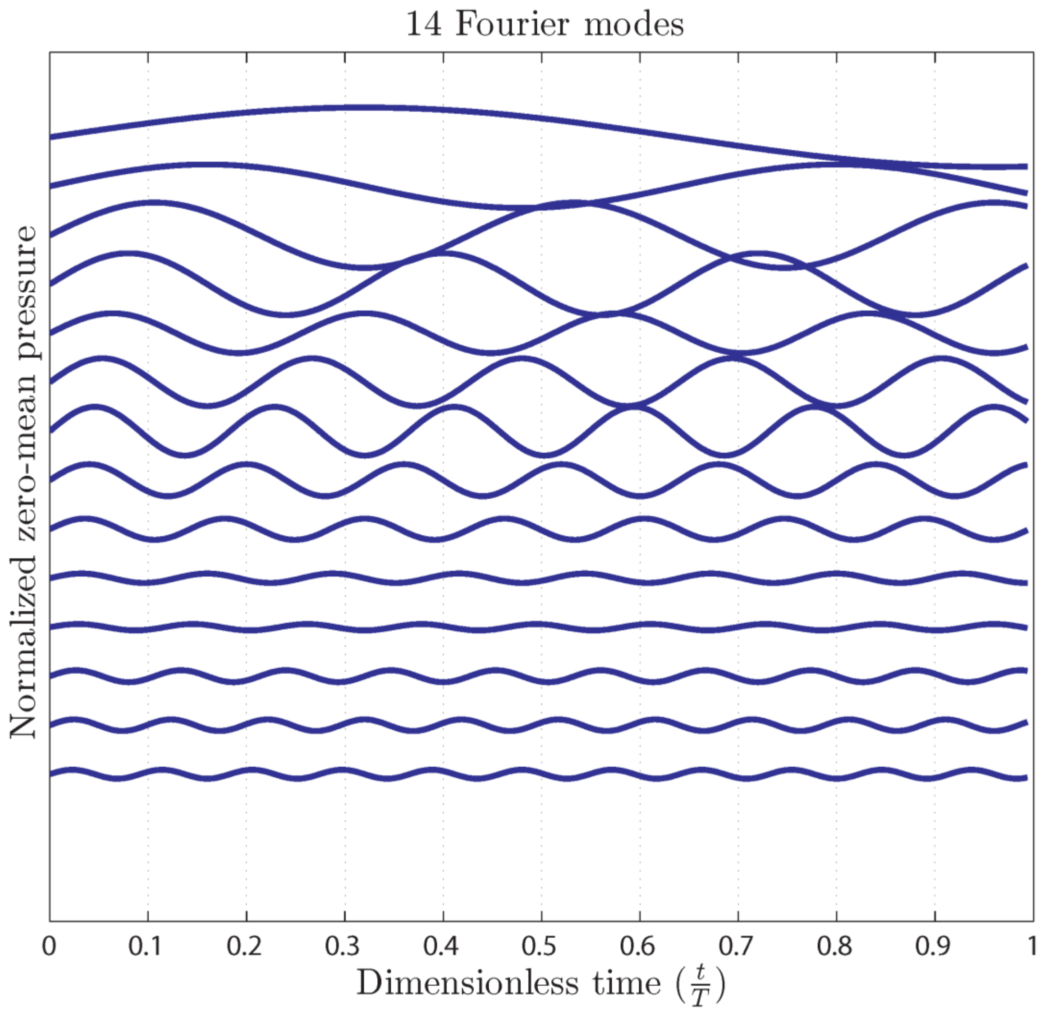
50Hz + 120 Hz Sinusoid

and its

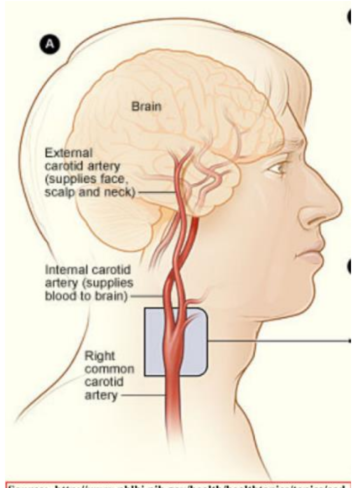
Frequency content

How do data/ signals look in Fourier space ?

Interpreting Fourier transform through an example



- **Decomposition of the pressure-time signal related to the carotid artery waveform**

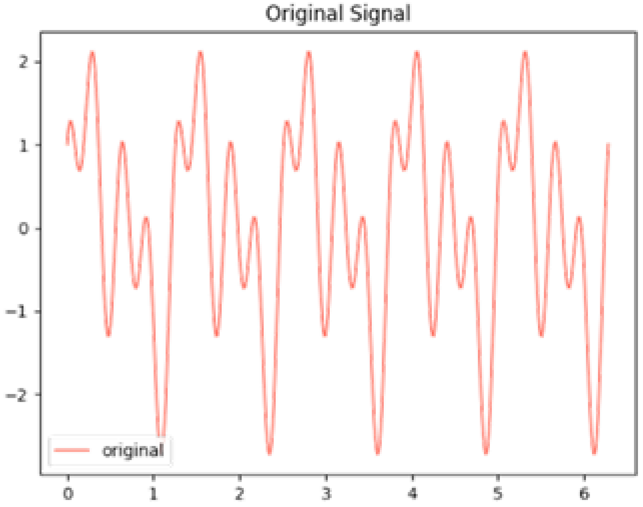

A
Brain
External carotid artery (supplies face, scalp and neck)
Internal carotid artery (supplies blood to brain)
Right common carotid artery

Source: http://www.nhlbi.nih.gov/health/healthtopics/topics/cad

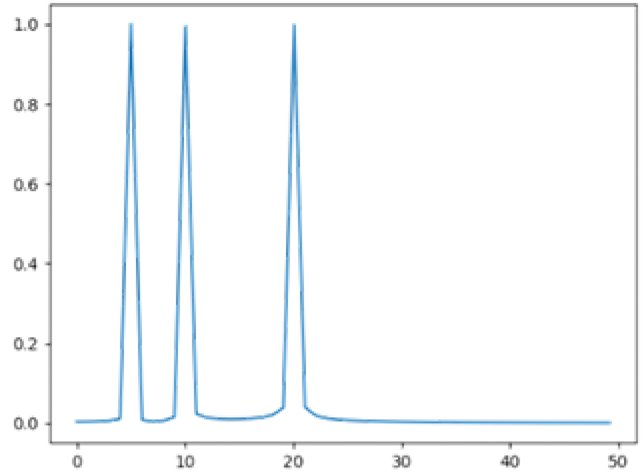
- **Summation of 14 wavetrains (Fourier modes) give the original waveforms**

Practical look at a code using scipy.fftpack library

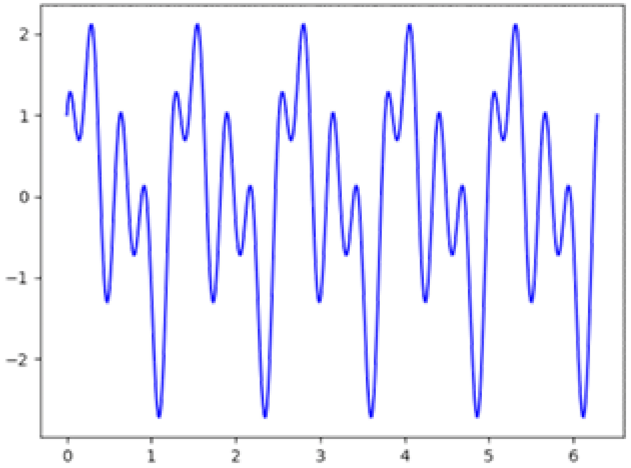
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.fftpack import fft, ifft, fftfreq

N = 1000
T = 1
x = np.linspace(0, 2*np.pi*N*T, N)
y1 = np.cos(20*x)
y2 = np.sin(10*x)
y3 = np.sin(5*x)

y = y1 + y2 + y3
# Produces an original signal
```


Original Signal

```
fy = fft(y)
# finds the fft
```



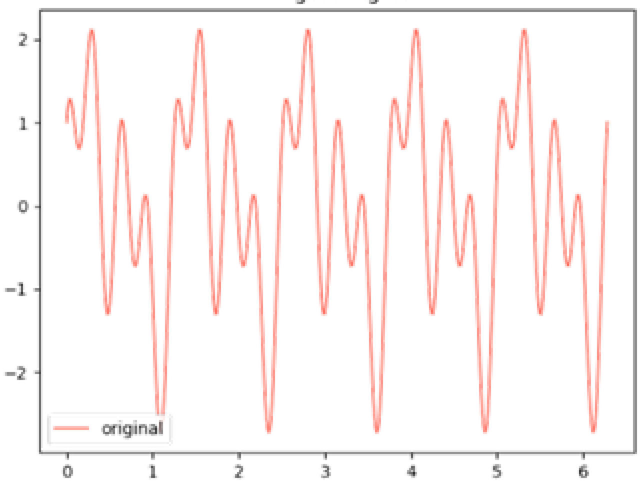
```
y4 = ifft(fy)
# finds the inverse fft
```



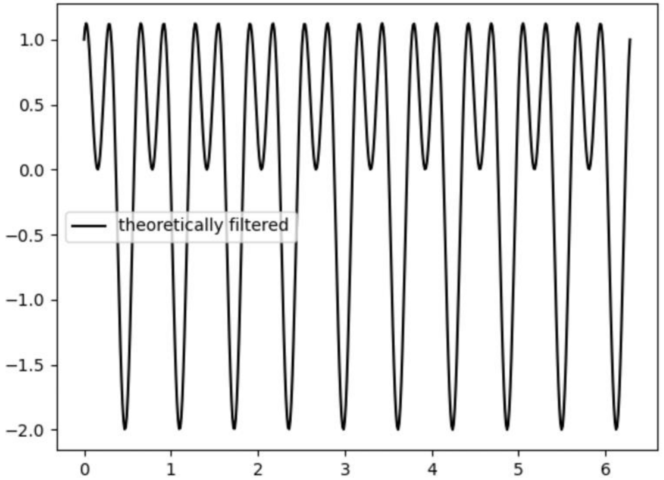
Practical look at Fourier filtering using scipy.fftpack

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.fftpack import fft, ifft, fftfreq

N = 1000
T = 1
x = np.linspace(0, 2*np.pi*N*T, N)
y1 = np.cos(20*x)
y2 = np.sin(10*x)
y3 = np.sin(5*x)

y = y1 + y2 + y3
# Produces an original signal
```

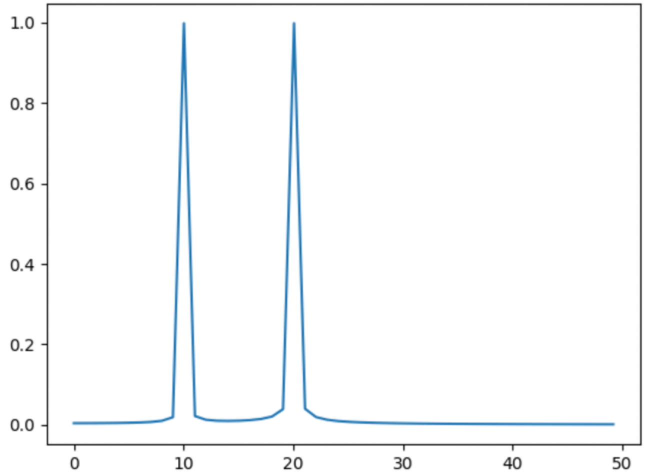
Original Signal



```
act = y1 + y2
# Produces a theoretically
# filtered signal
```

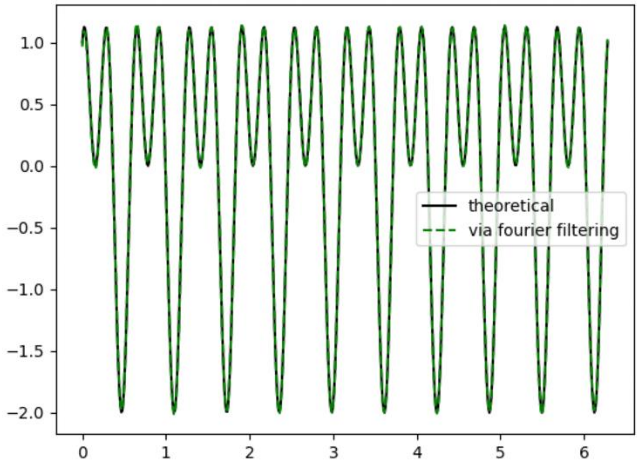


```
fy_act = fft(act)
# finds the fft
```



```
freqs = fftfreq(N)
nwaves = freqs*N # wave numbers

fft_vals = fft(y)

# Fourier filtering of 5 Hz signal
fft_new = np.copy(fft_vals)
fft_new[np.abs(nwaves)==5] = 0.0

# inverse fourier transform to
# reconstruct the filtered data
filt_data = np.real(ifft(fft_new))
```

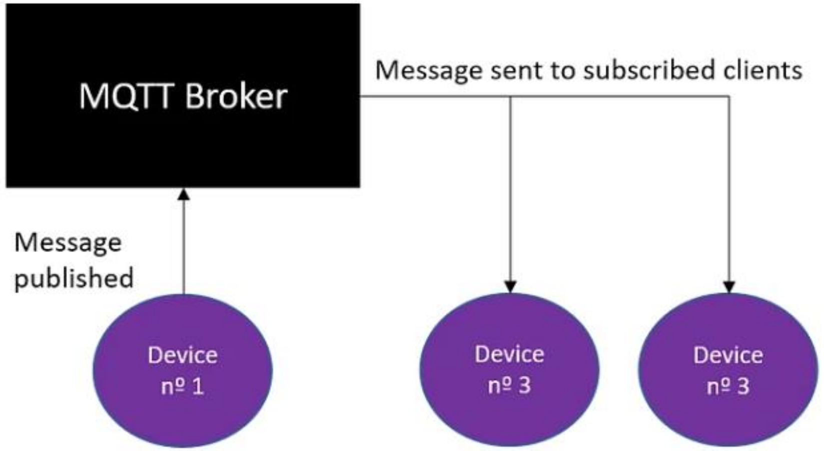
Data Filtering example



Explore MQTT Basics
Message Queuing Telemetry Transport

Goal: To understand how publishing and subscribing works practically

MQTT paradigm



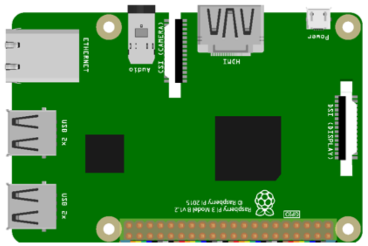
Example of the MQTT protocol

Source:
https://andre-benevides.medium.com/introduction-to-mqtt-and-configuration-of-a-mosquitto-broker-f0f7a7738bc8
https://learn.sparkfun.com/tutorials/introduction-to-mqtt#the-basics

Hardware

Broker

- The broker is the server
- It distributes the information to the interested devices connected to the server.



Client

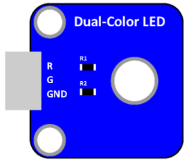
- The device that connects to broker to send or receive information.

SUNFOUNDER

Flame Sensor Module

SUNFOUNDER

PulseSensor Heart Rate Monitoring Sensor Module



SUNFOUNDER

Analog Hall Sensor Module

Messaging

Topic

- The name that the message is about.
- Clients publish, subscribe, or do both to a topic.

Publish

- Clients that send information to the broker to distribute to interested clients based on the topic name.

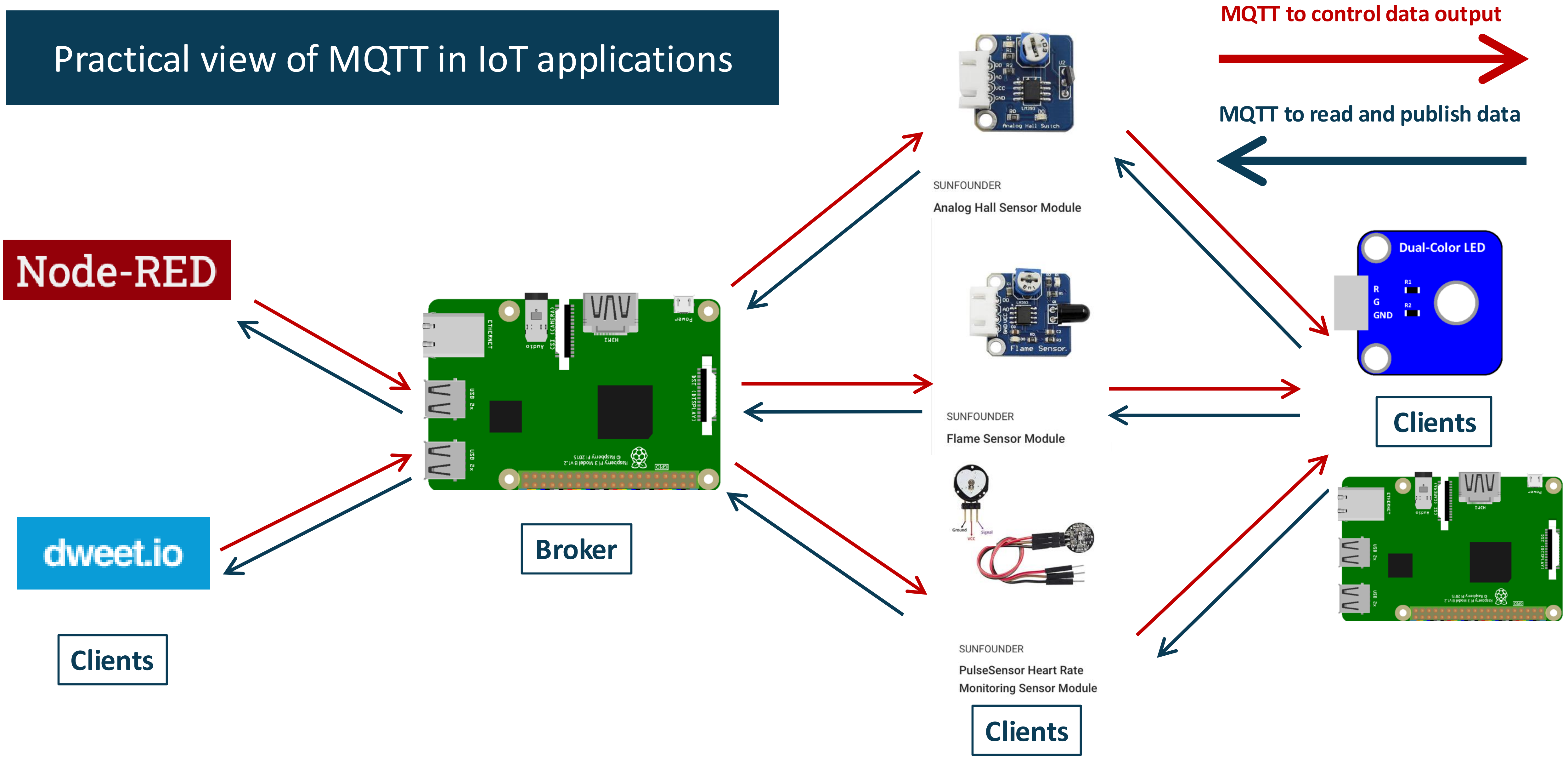
Subscribe

- Clients tell the broker which topic(s) they're interested in.

QoS

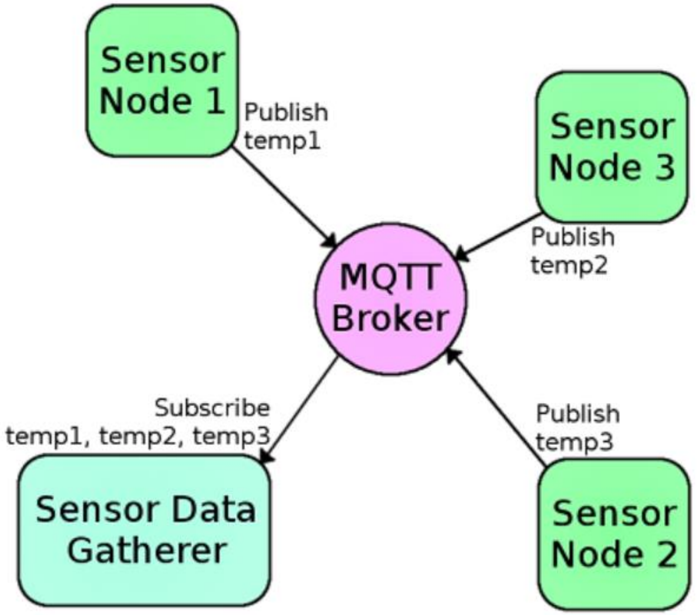
- Quality of Service to the broker
- Integer value ranging from. 0-2.

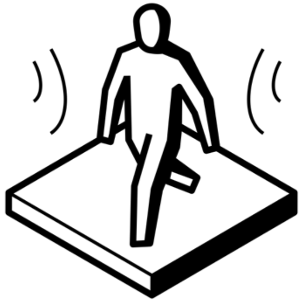
Practical view of MQTT in IoT applications


MQTT to control data output
MQTT to read and publish data
Node-RED
dweet.io
Clients
Broker
SUNFOUNDER
Analog Hall Sensor Module
SUNFOUNDER
Flame Sensor Module
SUNFOUNDER
PulseSensor Heart Rate Monitoring Sensor Module
Clients
Dual-Color LED
R
G
GND
Clients

MQTT paradigm

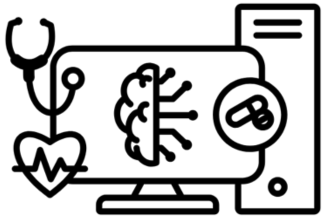
Source:
https://learn.sparkfun.com/tutorials/introduction-to-mqtt#the-basics
https://andre-benevides.medium.com/introduction-to-mqtt-and-configuration-of-a-mosquitto-broker-f0f7a7738bc8
Health monitoring by Sinta Maulana from Noun Project (CC BY 3.0)
motion sensor by Tomas Knopp from Noun Project (CC BY 3.0)
fire sensor by Carolina Cani from Noun Project (CC BY 3.0)
chat by Rolas Design from Noun Project (CC BY 3.0)




Created by Tomas Knopp from Noun Project

Created by Carolina Cani from Noun Project

Common usages


Created by Sinta Maulana from Noun Project

Created by Rolas Design from Noun Project

Source
https://mqtt.org/getting-started/ :
https://mosquitto.org
https://github.com/eclipse/mosquitto
https://projects.eclipse.org/projects/iot.mosquitto

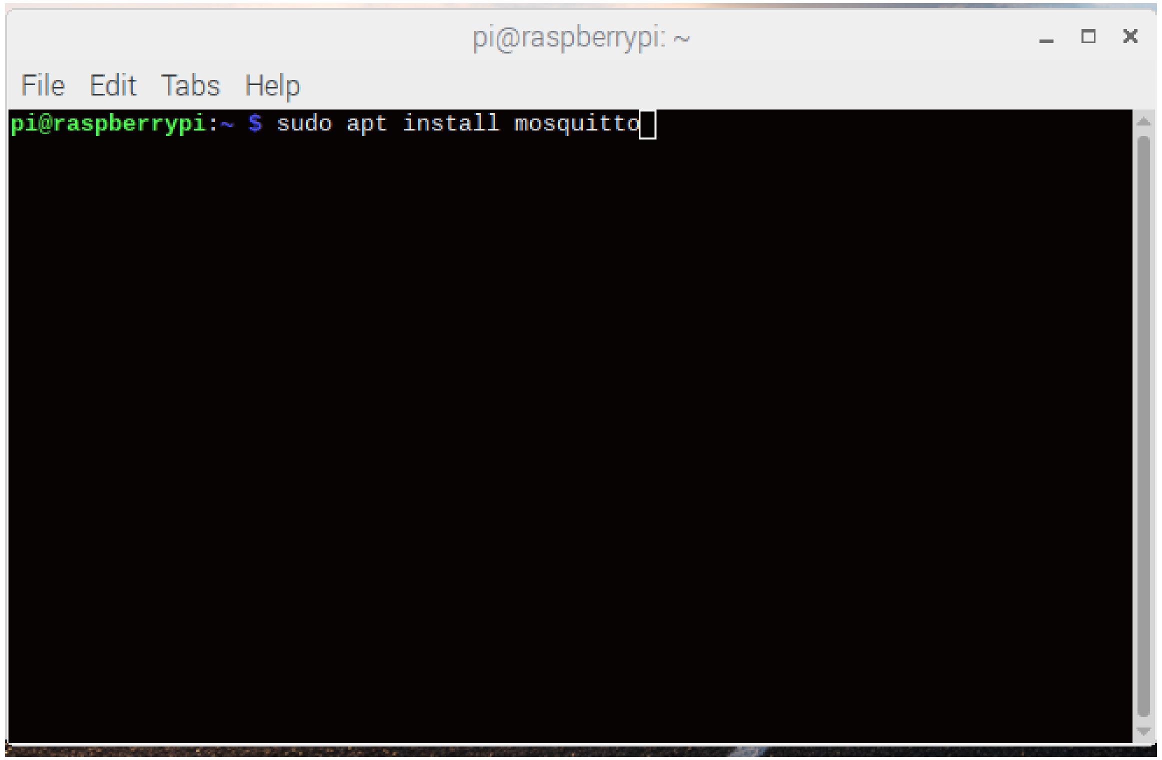
Eclipse Mosquitto - An open source MQTT broker

Eclipse Mosquitto provides a lightweight server implementation of the MQTT protocol that is suitable for all situations from full power machines to embedded and low power machines.

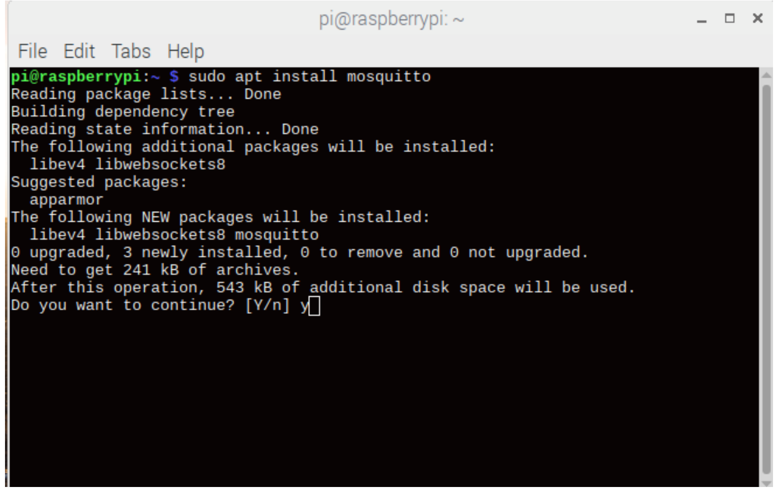
Sensors and actuators, which are often the sources and destinations of MQTT messages, can be very small and lacking in power. This also applies to the embedded machines to which they are connected, which is where Mosquitto could be run.

Step-1: Eclipse Mosquitto - An open source MQTT broker

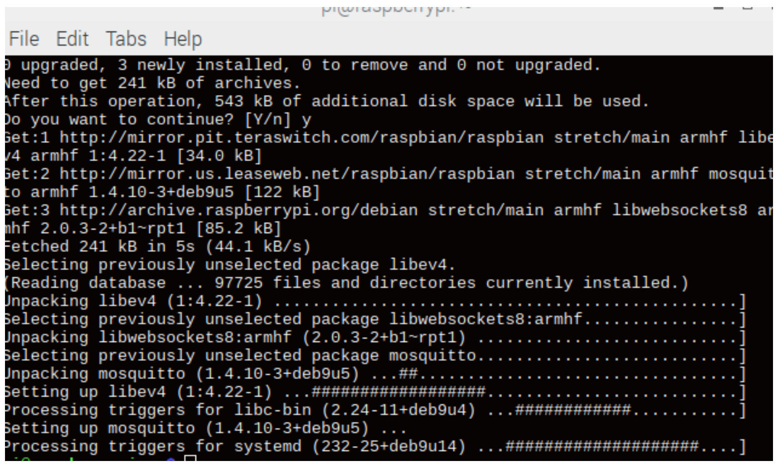
```
sudo apt-get update
sudo apt-get upgrade

sudo apt install mosquitto
```



```
pi@raspberrypi:~ $ sudo apt install mosquitto
```



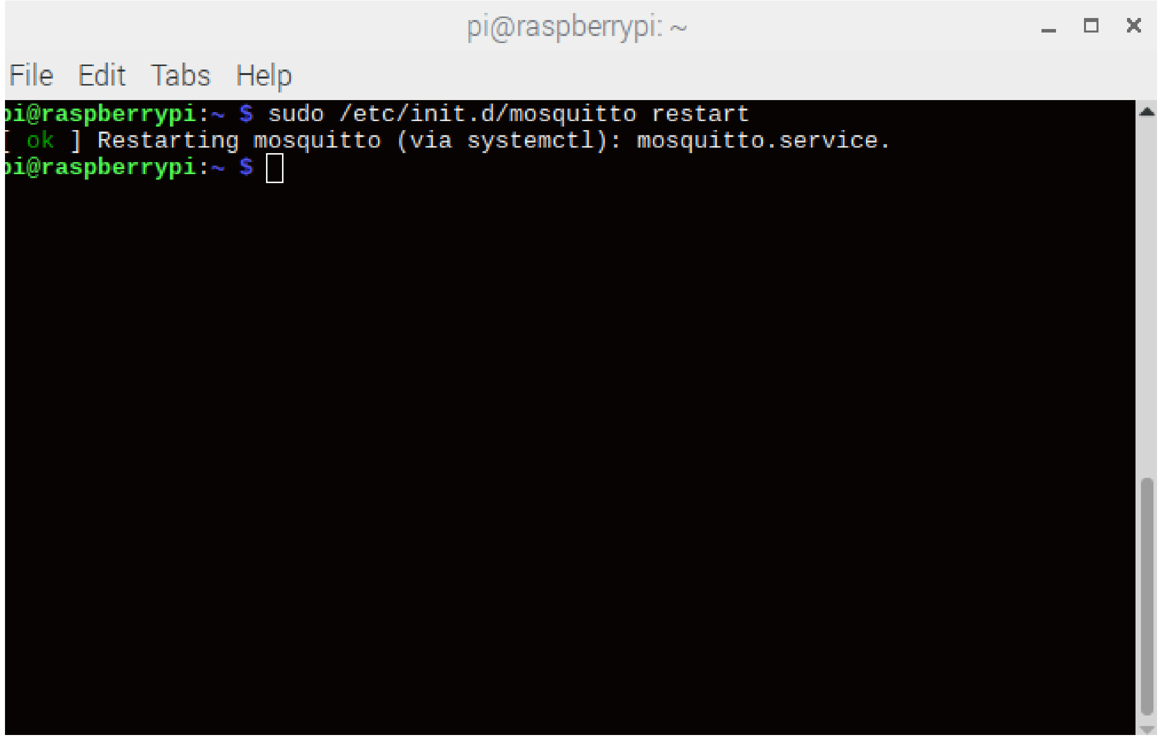
```
pi@raspberrypi:~ $ sudo apt install mosquitto
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libev4 libwebsockets8
Suggested packages:
  apparmor
The following NEW packages will be installed:
  libev4 libwebsockets8 mosquitto
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 241 kB of archives.
After this operation, 543 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```



```
) upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 241 kB of archives.
After this operation, 543 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://mirror.pit.teraswitch.com/raspbian/raspbian stretch/main armhf libe
v4 armhf 1:4.22-1 [34.0 kB]
Get:2 http://mirror.us.leaseweb.net/raspbian/raspbian stretch/main armhf mosquit
to armhf 1.4.10-3+deb9u5 [122 kB]
Get:3 http://archive.raspberrypi.org/debian stretch/main armhf libwebsockets8 ar
mhf 2.0.3-2+b1~rpt1 [85.2 kB]
Fetched 241 kB in 5s (44.1 kB/s)
Selecting previously unselected package libev4.
(Reading database ... 97725 files and directories currently installed.)
Unpacking libev4 (1:4.22-1) ...................................................]
Selecting previously unselected package libwebsockets8:armhf...................]
Unpacking libwebsockets8:armhf (2.0.3-2+b1~rpt1) ..............................]
Selecting previously unselected package mosquitto..............................]
Unpacking mosquitto (1.4.10-3+deb9u5) ...##....................................]
Setting up libev4 (1:4.22-1) ...##################.............................]
Processing triggers for libc-bin (2.24-11+deb9u4) ...############..............]
Setting up mosquitto (1.4.10-3+deb9u5) ...
Processing triggers for systemd (232-25+deb9u14) ...####################....]
```

Step-2: restart Mosquitto

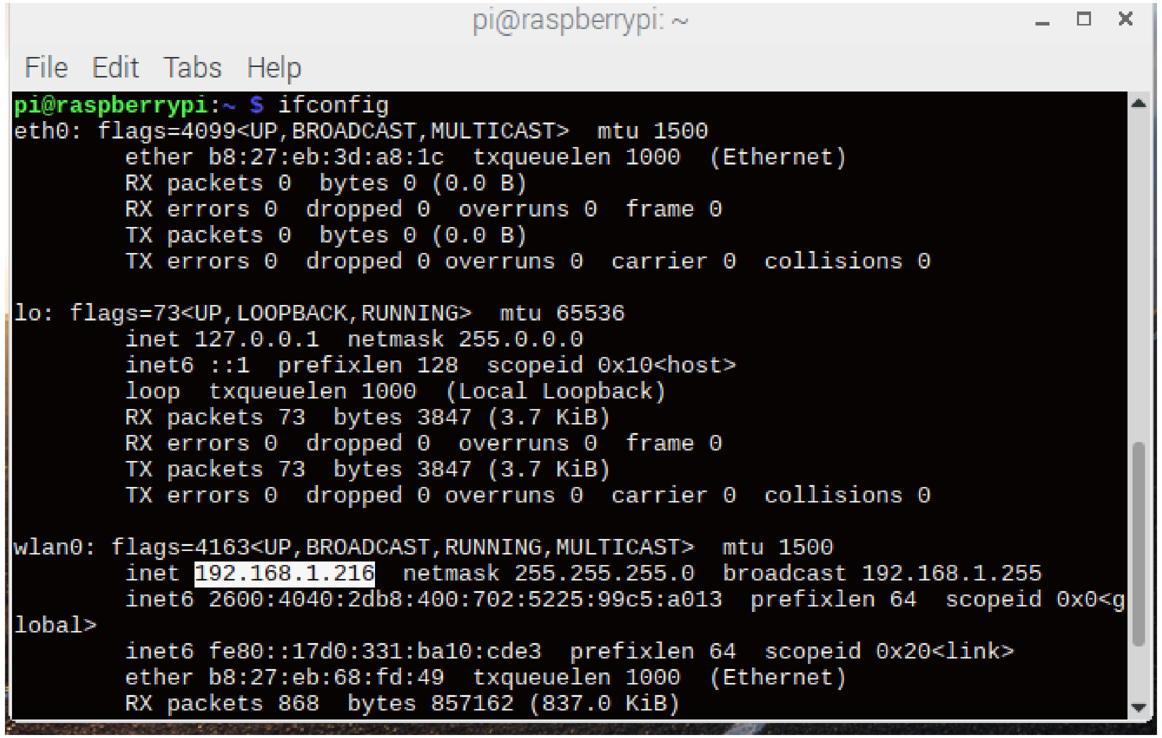
sudo /etc/init.d/mosquitto restart



```
pi@raspberrypi:~ $ sudo /etc/init.d/mosquitto restart
[ ok ] Restarting mosquitto (via systemctl): mosquitto.service.
pi@raspberrypi:~ $ 
```

Step-3: Get your IP address

ifconfig



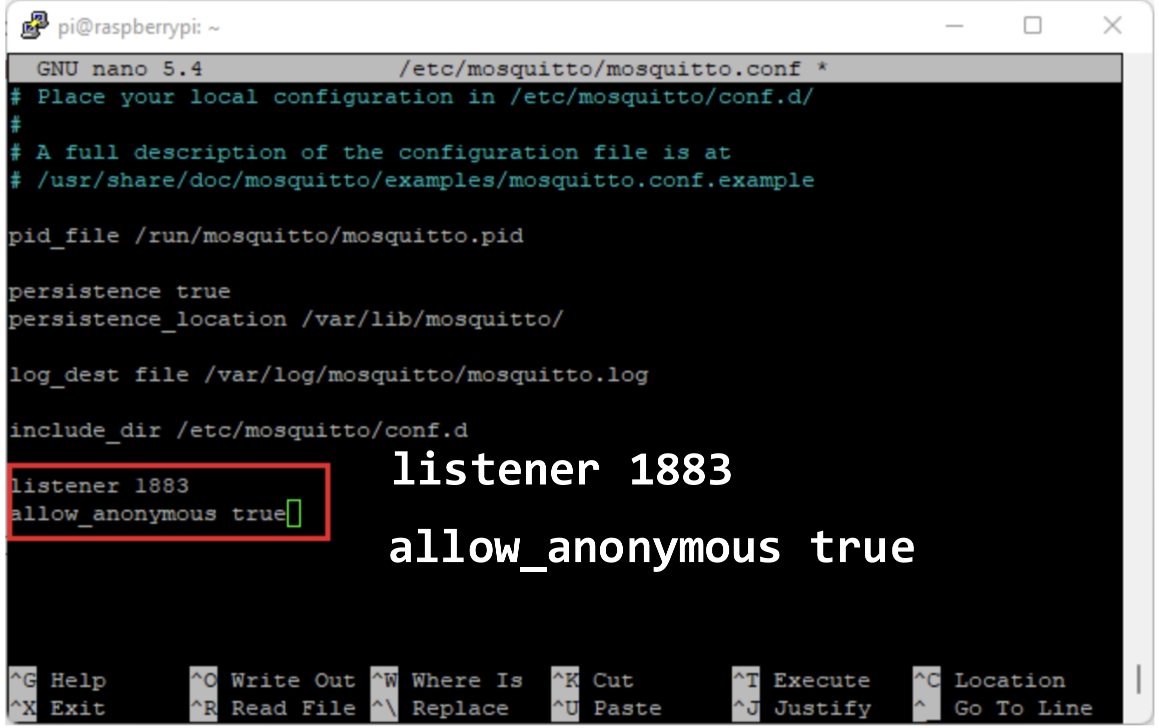
```
pi@raspberrypi:~ $ ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether b8:27:eb:3d:a8:1c  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 73  bytes 3847 (3.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 73  bytes 3847 (3.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.216  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 2600:4040:2db8:400:702:5225:99c5:a013  prefixlen 64  scopeid 0x0<g
lobal>
        inet6 fe80::17d0:331:ba10:cde3  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:68:fd:49  txqueuelen 1000  (Ethernet)
        RX packets 868  bytes 857162 (837.0 KiB)
```

hostname –I # this is also OK to use

Source:
https://randomnerdtutorials.com/how-to-install-mosquitto-broker-on-raspberry-pi/
https://forums.raspberrypi.com/viewtopic.php?t=331096

Step-4: Enable Remote Access to Mosquitto Broker (No Authentication)

```
sudo nano /etc/mosquitto/mosquitto.conf
```



```
  GNU nano 5.4                   /etc/mosquitto/mosquitto.conf *
# Place your local configuration in /etc/mosquitto/conf.d/
#
# A full description of the configuration file is at
# /usr/share/doc/mosquitto/examples/mosquitto.conf.example

pid_file /run/mosquitto/mosquitto.pid

persistence true
persistence_location /var/lib/mosquitto/

log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d

listener 1883
allow_anonymous true


^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

```
listener 1883

allow_anonymous true
```

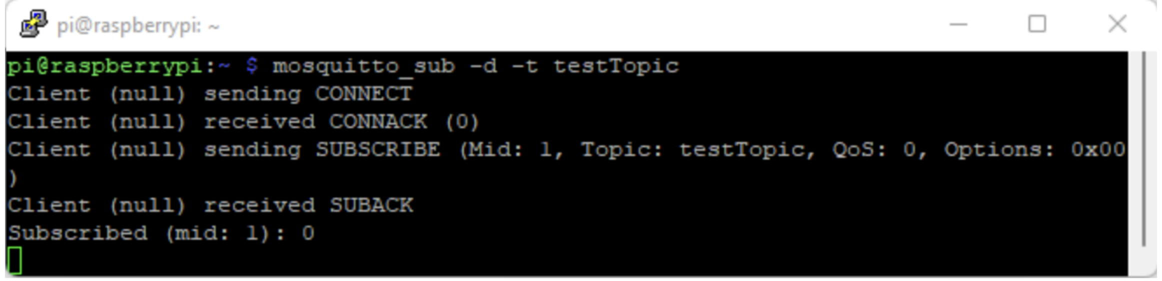
```
sudo systemctl restart mosquitto

sudo systemctl status mosquitto
```

Source:
https://randomnerdtutorials.com/testing-mosquitto-broker-and-client-on-raspbbery-pi/
https://mosquitto.org/man/mosquitto_pub-1.html
https://mosquitto.org/man/mosquitto_sub-1.html

Step-4: Publishing “Hello World!” Message to *testTopic* Topic

# Install mosquitto-clients
sudo apt install -y mosquitto mosquitto-clients

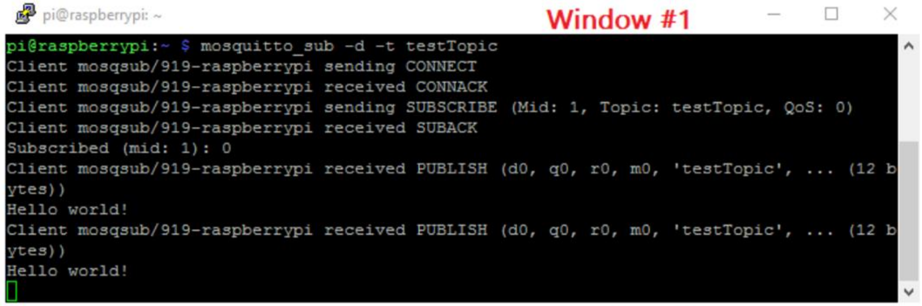
# Open a terminal window and type the following:
mosquitto_sub -d -t testTopic



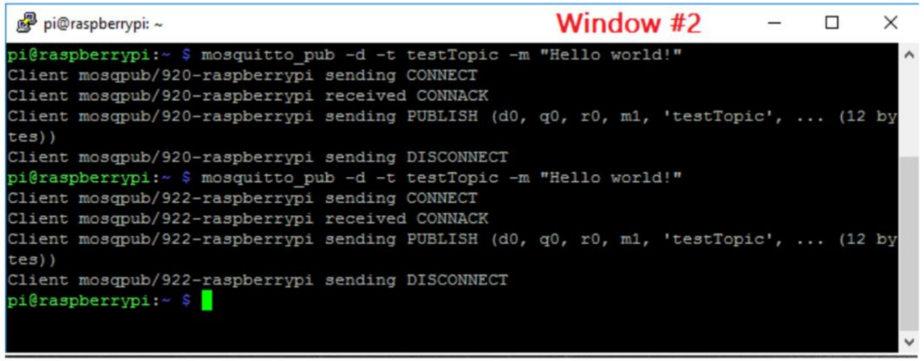
```
pi@raspberrypi:~ $ mosquitto_sub -d -t testTopic
Client (null) sending CONNECT
Client (null) received CONNACK (0)
Client (null) sending SUBSCRIBE (Mid: 1, Topic: testTopic, QoS: 0, Options: 0x00
)
Client (null) received SUBACK
Subscribed (mid: 1): 0
```

mosquitto_sub -v -t '#' -h <IP address>

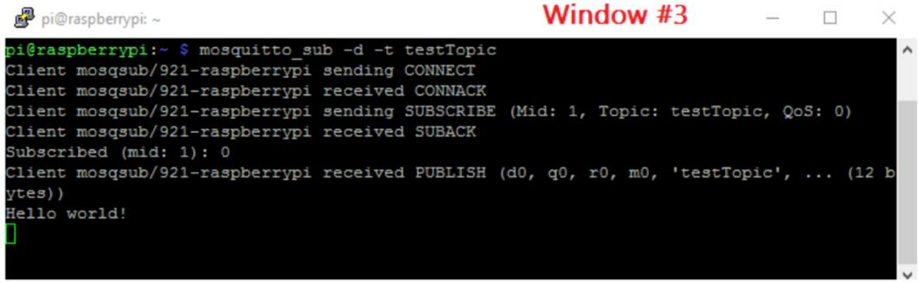
mosquitto_pub -d -t testTopic -m "Hello world!"



```
pi@raspberrypi:~ $ mosquitto_sub -d -t testTopic
Client mosqsub/919-raspberrypi sending CONNECT
Client mosqsub/919-raspberrypi received CONNACK
Client mosqsub/919-raspberrypi sending SUBSCRIBE (Mid: 1, Topic: testTopic, QoS: 0)
Client mosqsub/919-raspberrypi received SUBACK
Subscribed (mid: 1): 0
Client mosqsub/919-raspberrypi received PUBLISH (d0, q0, r0, m0, 'testTopic', ... (12 b
ytes))
Hello world!
Client mosqsub/919-raspberrypi received PUBLISH (d0, q0, r0, m0, 'testTopic', ... (12 b
ytes))
Hello world!
```



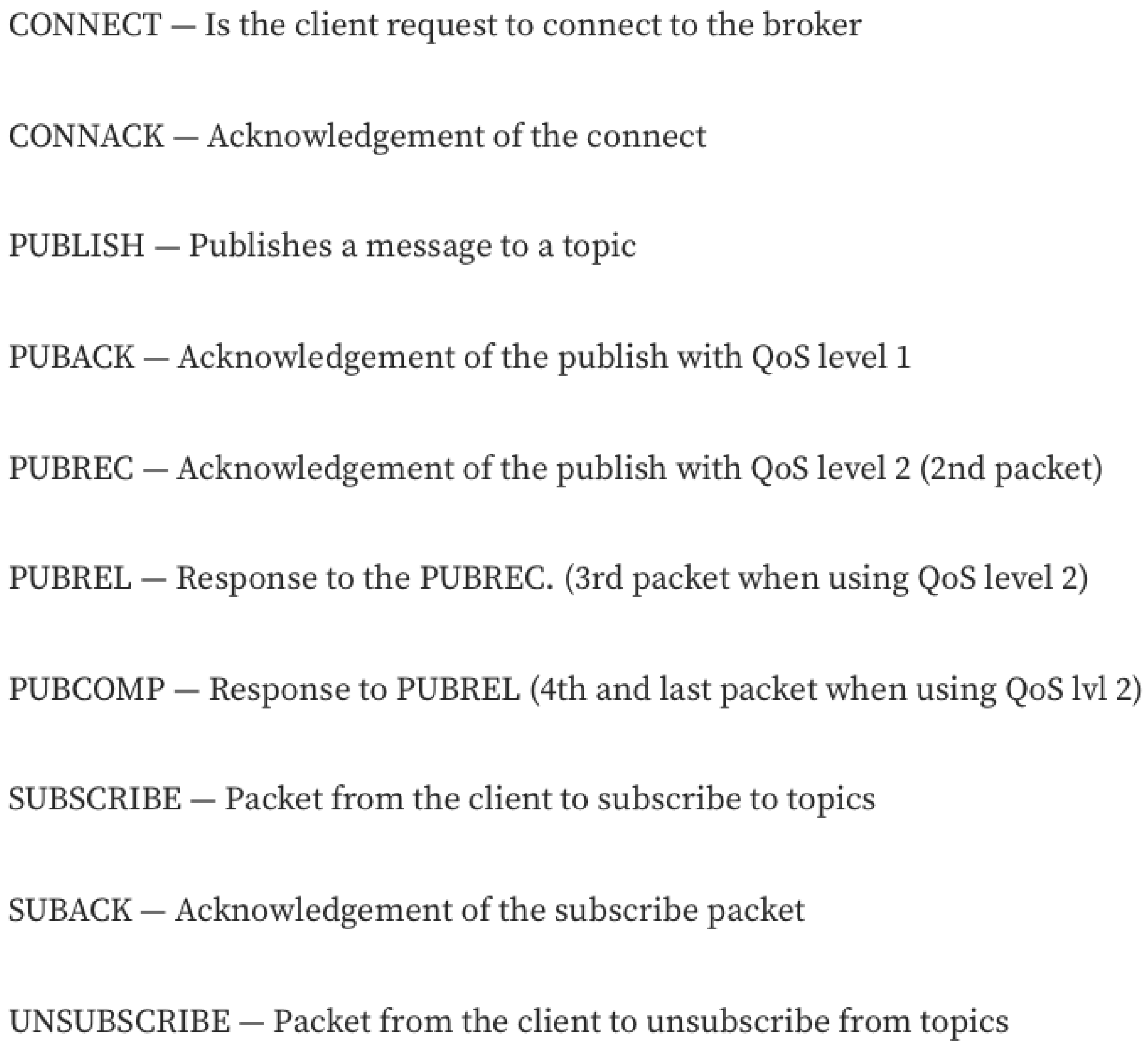
```
pi@raspberrypi:~ $ mosquitto_pub -d -t testTopic -m "Hello world!"
Client mosqpub/920-raspberrypi sending CONNECT
Client mosqpub/920-raspberrypi received CONNACK
Client mosqpub/920-raspberrypi sending PUBLISH (d0, q0, r0, m1, 'testTopic', ... (12 by
tes))
Client mosqpub/920-raspberrypi sending DISCONNECT
pi@raspberrypi:~ $ mosquitto_pub -d -t testTopic -m "Hello world!"
Client mosqpub/922-raspberrypi sending CONNECT
Client mosqpub/922-raspberrypi received CONNACK
Client mosqpub/922-raspberrypi sending PUBLISH (d0, q0, r0, m1, 'testTopic', ... (12 by
tes))
Client mosqpub/922-raspberrypi sending DISCONNECT
pi@raspberrypi:~ $
```



```
pi@raspberrypi:~ $ mosquitto_sub -d -t testTopic
Client mosqsub/921-raspberrypi sending CONNECT
Client mosqsub/921-raspberrypi received CONNACK
Client mosqsub/921-raspberrypi sending SUBSCRIBE (Mid: 1, Topic: testTopic, QoS: 0)
Client mosqsub/921-raspberrypi received SUBACK
Subscribed (mid: 1): 0
Client mosqsub/921-raspberrypi received PUBLISH (d0, q0, r0, m0, 'testTopic', ... (12 b
ytes))
Hello world!
```

Source:
https://andre-benevides.medium.com/introduction-to-mqtt-and-configuration-of-a-mosquitto-broker-f0f7a7738bc8

Types of MQTT messages



CONNECT — Is the client request to connect to the broker

CONNACK — Acknowledgement of the connect

PUBLISH — Publishes a message to a topic

PUBACK — Acknowledgement of the publish with QoS level 1

PUBREC — Acknowledgement of the publish with QoS level 2 (2nd packet)

PUBREL — Response to the PUBREC. (3rd packet when using QoS level 2)

PUBCOMP — Response to PUBREL (4th and last packet when using QoS lvl 2)

SUBSCRIBE — Packet from the client to subscribe to topics

SUBACK — Acknowledgement of the subscribe packet

UNSUBSCRIBE — Packet from the client to unsubscribe from topics

Explore SenseHat (The RPi companion sensor in the International Space Station

Example of who is using the sense HAT and where - Astro Pi

Source: https://youtu.be/kk_7KNuRLrk

What we will do today

- Co-work
  - Observe, ask and try in groups
- Write small program using Python
- Think about
  - Challenges, Opportunities, Gaps and Surprises

What we will learn today

- Communicate with the Sense HAT using Python
- Access the outputs of the Sense HAT
- Use the Sense HAT library to display messages and images
- Use loops to repeat certain code blocks