UNIVERSIDAD
DE BURGOS

Pycom – multinetwork devices

Part One: MQTT via WiFi

José M. Cámara Nebreda. 2018

IoT devices on offer

- Development modules:
  - WiPy 3.0: WiFi & Bluetooth
  - LoPy: LoRa, WiFi & Bluetooth
  - SiPy: Sigfox, WiFi & Bluetooth
  - PoPy4: LoRa, Sigfox, WiFi & Bluetooth
  - Gpy: LTE-M, WiFi & Bluetooth
  - FiPy: LoRa, Sigfox, LTE-M, WiFi & Bluetooth
- Integrated circuits (OEM):
  - WO1: WiFi & Bluetooth
  - G01: LTE-M, WiFi & Bluetooth
  - LO1: LoRa, WiFi & Bluetooth
  - LO4: LoRa, Sigfox, WiFi & Bluetooth

FiPy

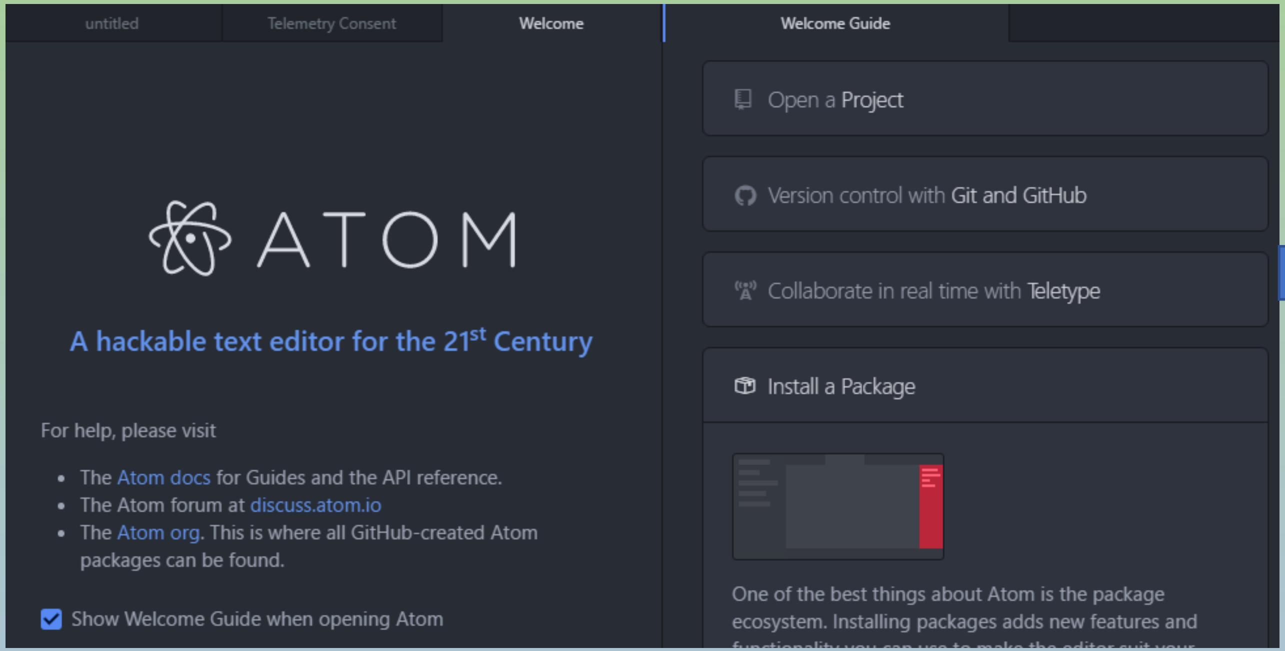
- Features:
  - ESP32 dual core microcontroller
  - USB host port
  - 8 MB flash memory
  - 4 MB RAM memory
  - 3v3 regulator

- Hardware requirements:
  - FiPy module
  - Host PC
  - USB o MicroUSB cable
  - Expansion board (recommended)

- Software requirements:
  - Text editor: ATOM recommended by manufacturer. It will perform as an IDE.
  - Pymakr Plugin for the editor to enable programing the device.

UNIVERSIDAD
DE BURGOS

Pymkr Installation



ATOM

A hackable text editor for the 21st Century

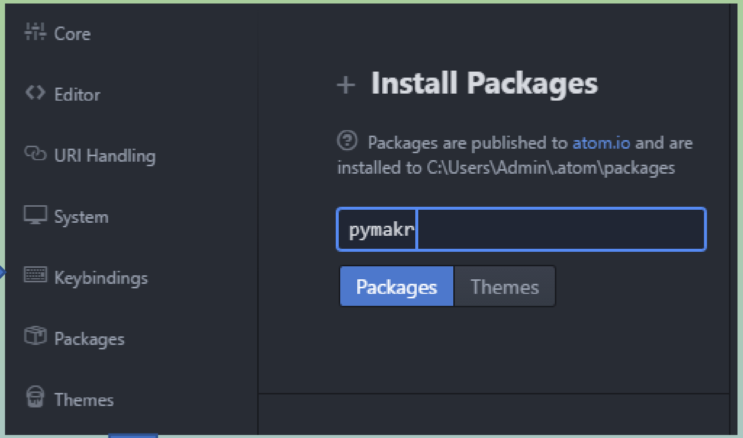
For help, please visit

- The Atom docs for Guides and the API reference.
- The Atom forum at discuss.atom.io
- The Atom org. This is where all GitHub-created Atom packages can be found.

☑ Show Welcome Guide when opening Atom

Open a Project
Version control with Git and GitHub
Collaborate in real time with Teletype
Install a Package

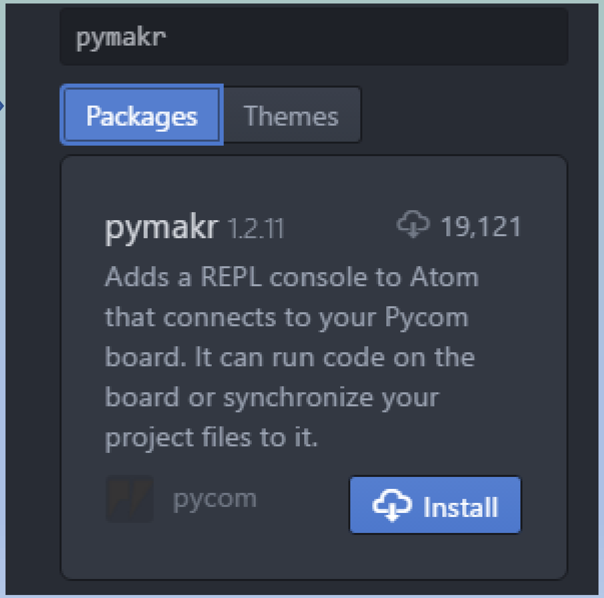
One of the best things about Atom is the package ecosystem. Installing packages adds new features and



+ Install Packages

Packages are published to atom.io and are installed to C:\Users\Admin\.atom\packages

pymakr


pymakr

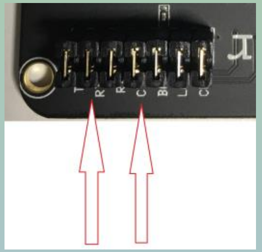
pymakr 1.2.11 19,121

Adds a REPL console to Atom that connects to your Pycom board. It can run code on the board or synchronize your project files to it.

For further information refer to: https://docs.pycom.io/chapter/gettingstarted/installation/pymakr.html

Start – up process

- Connect FiPy to expansion board and then to the host computer via USB cable.
- Remove RTS and CTS jumpers from the expansion board.



- Update the firmware using the pycom firmware update tool: https://docs.pycom.io/chapter/gettingstarted/installation/firmwaretool.html
- Connect to your FiPy board: https://docs.pycom.io/chapter/pymakr/installation/atom.html
- Try:

```
import pycom
pycom.heartbeat(False)
pycom.rgbled(0x00ff00)
```

this should turn the LED green

Wifi

- FiPy has a built-in wifi radio mudule.
- It can be configured for several operation modes:
  - AP (After power-up): this is the default mode. FiPy becomes an Access poit to where othre devices are connected. It can be found as “fipy-wlan-xxxx”. Its password is always www.pycom.io.
  - STA (station): the device is expected to connect to an external router. Three authentication modes are supported:
    - WPA personal: this mode enables FiPy to connect to your home network.
    - WPA Enterprise (EAP – TLS): this enables Fipy to connect to entreprise networks using certificates.
    - WPA Enterprise (EAP – TTLS or PEAP): to connect to Enterprise networks using user/passwork authentication.

Wifi examples

AP mode

```
CODE:
from network import WLAN
wlan = WLAN() # Default mode is AP
print("Network name: ",wlan.ssid())
print("IP address: ",wlan.ifconfig())
print("Autentication: ", wlan.auth())

OUTPUT:
Network name:  fipy-wlan-0090
IP address:  ('0.0.0.0', '0.0.0.0', '0.0.0.0', '0.0.0.0')
Autentication:  (3, 'www.pycom.io’)
```

WPA personal

```
CODE:
from network import WLAN
wlan = WLAN(mode=WLAN.STA)
print(‘Trying')
nets = wlan.scan()
for net in nets:
    if net.ssid == ‘my-wifi':
        print('Network found!')
        wlan.connect(net.ssid, auth=(net.sec, ‘wifi-key'), timeout=5000)
        print('connection established')
        break

OUTPUT:
Trying
Network found!
connection established
```

Wifi examples (II)

WPA Enterprise code

GENERAL CASE:

```
from network import WLAN
wlan = WLAN(mode=WLAN.STA)
wlan.connect(ssid=“mywifi”, auth=(WLAN.WPA2_ENT,
‘user’, ‘password’), identity=‘myid’,
ca_certs=‘/flash/cert/ca.pem’)
```

OUR CASE:

```
import pycom
import machine
Pycom.heartbeat(False)
from network import WLAN
wlan = WLAN(mode=WLAN.STA)
wlan.connect(ssid=“wifiubu”, auth=(WLAN.WPA2_ENT,
‘user’, ‘password’)
while not   wlan.isconnected():
            machine.idle()
pycom.rgbled(0x00ff00) # turn on the LED when connected
```

Output

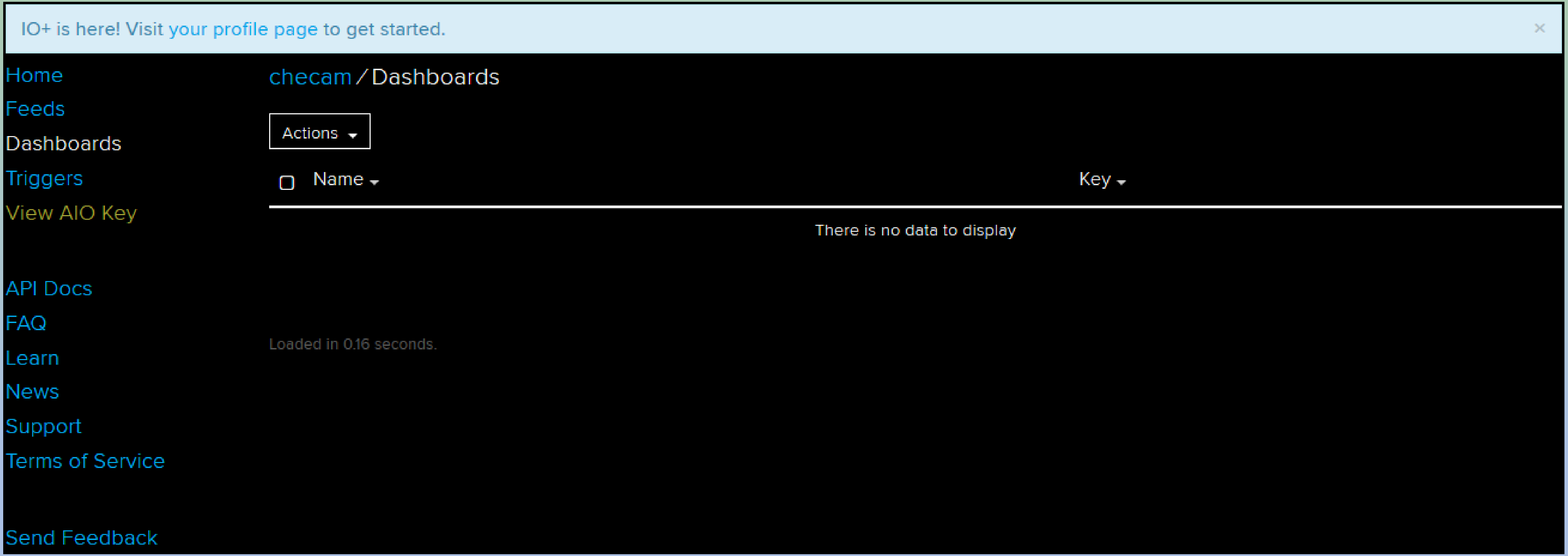
MQTT https://core-electronics.com.au/tutorials/internet-of-things-with-pycom-and-adafruit-io.html

- Application layer propocol
- Publisher/subscriber model
- Built on top of the protocol stack, provides a messaging mechanism to provide and obtain information in short pieces
- For testing, Pycom recommends the free Adrafruit services: https://io.adafruit.com/
- Sign in and you will gain access to you personal account (next slide)
- You will need to download the mqtt library mqtt.py at: https://github.com/pycom/pycom-libraries/tree/master/lib/mqtt
- https://docs.pycom.io/chapter/tutorials/all/mqtt.html

Adafruit account

Here you can manage (create, get, change) your feeds (input values, commands,..)

Here you can get your personal API key to include in your programming



IO+ is here! Visit your profile page to get started.

chececam / Dashboards

There is no data to display

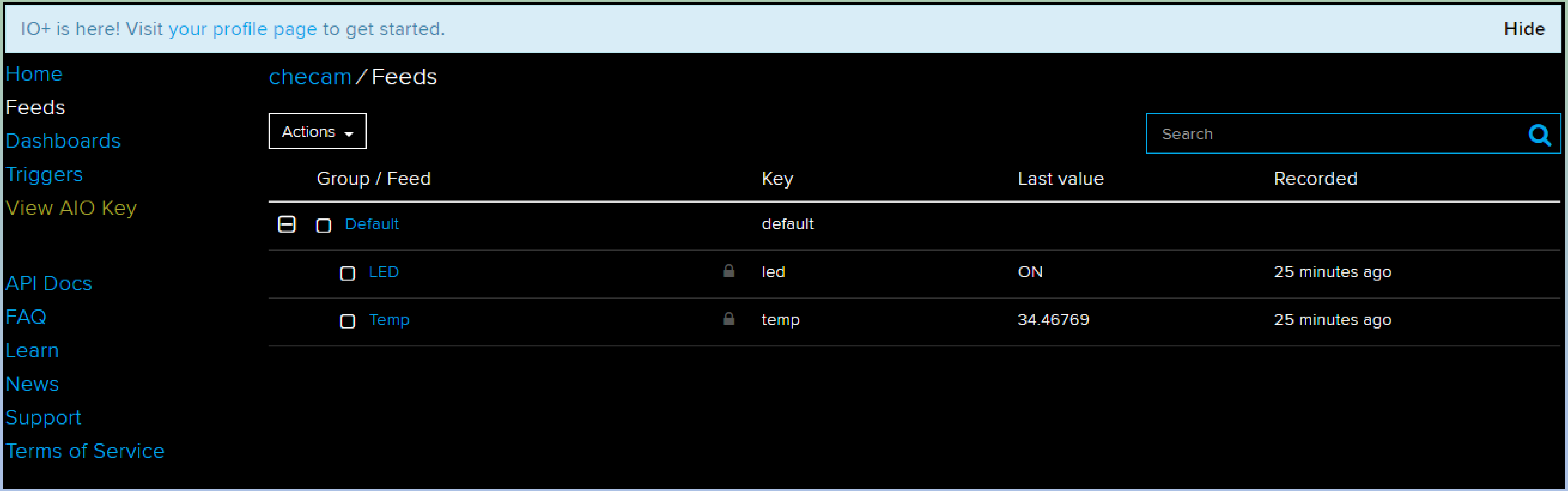
Loaded in 0.16 seconds.

UNIVERSIDAD DE BURGOS

Adafruit account

In “Actions” we create two new feeds:

- LED: meant to change the status of the onboard LED
- Temp: meant to display temperature readings from the PySENSE built in thermometer



IO+ is here! Visit your profile page to get started. Hide

Home
Feeds
Dashboards
Triggers
View AIO Key

API Docs
FAQ
Learn
News
Support
Terms of Service

checam / Feeds

Actions ▾

Search

Group / Feed	Key	Last value	Recorded
⊟ □ Default	default		
□ LED	led	ON	25 minutes ago
□ Temp	temp	34.46769	25 minutes ago

Feeds are input/output:

- When the Fipy publishes a new value, it is updated and displayed online
- When the Fipy subscribes, values changed online are received and read by Fipy
- Fipy can both publish and subscribe to the same feed

MQTT connection example

Code section

```
#mqtt section
print('Trying to connect Adafruit MQTT')
client = MQTTClient('FiPy', 'io.adafruit.com', port=1883, user='checam', password=‘xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx')
client.set_callback(sub_cb)
client.connect()
print('Connection successful')
client.subscribe(topic="checam/feeds/led")
client.subscribe(topic="checam/feeds/temp")
pycom.rgbled(0xff00ff)
```

Output

```
Trying to connect Adafruit MQTT
Connection successful
```

Mqtt library functionality

- connect(): sets up a socket to communicate with the server
- disconnect(): ends the socket
- publish(): sends information to a topic on the server
- subscribe(): subscribes to a topic on the server
- set_callback(): sets up the routine to which incoming subscribed messages are delivered
- wait_msg(): wait for subscribed messages to arrive. When arrived the message is processed and sent to a callback routine
- chk_msg(): checks whether a message is pending. If it is, proceeds as wait_msg()
- ping(): checks connection

UNIVERSIDAD
DE BURGOS

Pysense example

Pysense board provides several readings:

Temperature
Humidity
Pressure
Acceleration
Roll
Pitch
Light level
Altitude

Code section

```
#pysense section
mp = MPL3115A2(mode=ALTITUDE) # Returns height in meters. Mode may also be set to PRESSURE, returning a value in
Pascals
acc = LIS2HH12()
si = SI7006A20()
lt = LTR329ALS01()
while True:
    pitch = acc.pitch()
    roll = acc.roll()
    light = lt.light()
    print("Pitch & Roll: " + '{},{}'.format(pitch, roll))
    print("Light: " + '{}'.format(light))
    print("Altitude: " + str(mp.altitude()))
    print("Temperature: " + str(si.temperature())+ " deg C and Relative Humidity: " + str(si.humidity()) + " %RH")
    client.publish(topic="checam/feeds/temp", msg=str(si.temperature()))
    client.check_msg()
    time.sleep(5)
```

Pysense example

Pysense board provides several readings:

- Temperature
- Humidity
- Pressure
- Acceleration
- Roll
- Pitch
- Light level
- Altitude

Output

```
Pitch & Roll: -3.634626,110.4271
Light: (32, 27)
Altitude: 773.875
Temperature: 34.27464 deg C and Relative Humidity: 38.30771 %RH
```

Pysense library files

```
from LIS2HH12 import LIS2HH12
from SI7006A20 import SI7006A20
from LTR329ALS01 import LTR329ALS01
from MPL3115A2 import MPL3115A2,ALTITUDE,PRESSURE
```

Input feed

We use the onboard LED to show how commands can be received from the server and therefore from the remote user.

On the Pysense code example we had the line:

```
client.check_msg()
```

This MQTT method checks whether there is an incoming message:

In case there is no one, it does nothing.

If a message has arrived, it is delivered to the routine defined in the MQTT connection code example:

```
client.set_callback(sub_cb)
```

The routine sets what to do with the message:

```
def sub_cb(topic, msg):
   print(msg)
   print(msg.decode("utf-8"))
   if msg.decode("utf-8") == 'ON':
      pycom.rgbled(0xffffff)
   if msg.decode("utf-8") == 'OFF':
      pycom.rgbled(0x000000)
```

In this case t turns on and off the onboard LED depending on the text of the message

Micropython mqtt library issues

- Using the mqtt.py module we have encountered the following issue:
    - OSError -1 on method connect().
- For some reason, the getaddrinfo() function must be invoked within the socket.connect() method. To make that posible, we have added two new atributes at _init_(): self.server & self.port.
- We have replaced: self.sock.connect(self.addr) by self.sock.connect(socket.getaddrinfo(self.server,self.port)[0][-1])
- Connection is yet not always successful. A certain delay seems to be necessary in the form:

```
self.sock = socket.socket()
time.sleep(10)
self.sock.connect(socket.getaddrinfo(self.server,self.port)[0][-1])
```

References

- Although the contents of this presentation are original, most of the information, code and data included are based on the official Pycom website: https://pycom.io/
- A complete guide about the set-up, connection and programming of the development boards including FiPy can be found at: https://docs.pycom.io/