The Things Network.cat

Taller **bàsic** - fem un termòstat amb una **Lopy4**

Xarxa comunitària per l'Internet de les Coses

@ttncat
thethingsnetwork.cat

@thethingsntwrk
thethingsnetwork.org

Introducció

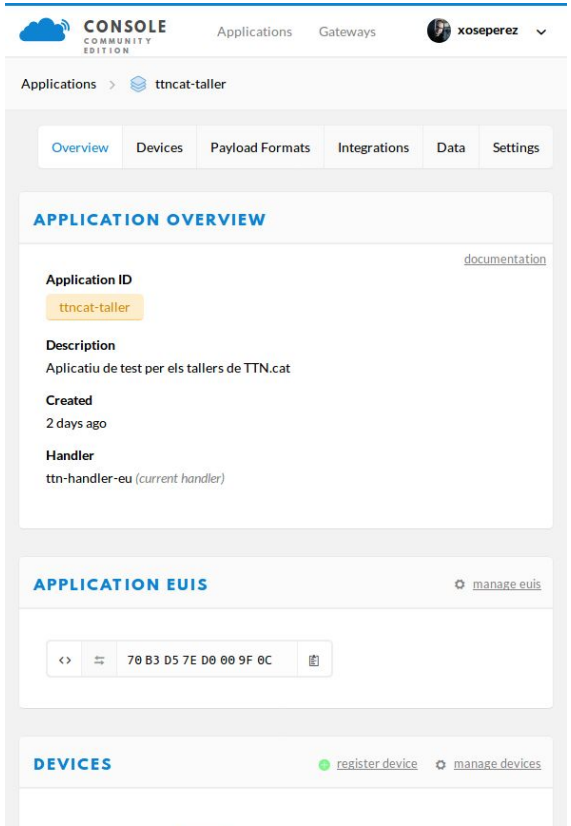
Consola

Consola TTN

Des de la consola de TTN podem **gestionar les passarel·les i aplicacions**. També podem monitoritzar el tràfic que passa o arriba a qualsevol de les dues i veure els continguts si tenim les claus de desxifratge.

Els nodes (*devices*) estan associats a les aplicacions. Podem donar-ne d'alta, configurar el mode en que es connectaran a la xarxa (OTAA o ABP), veure els missatges que envien o enviar nosaltres un missatge a qualsevol dispositiu.

També des de l'aplicació podem fer integracions amb diferents serveis *online* com ara IFTTT.

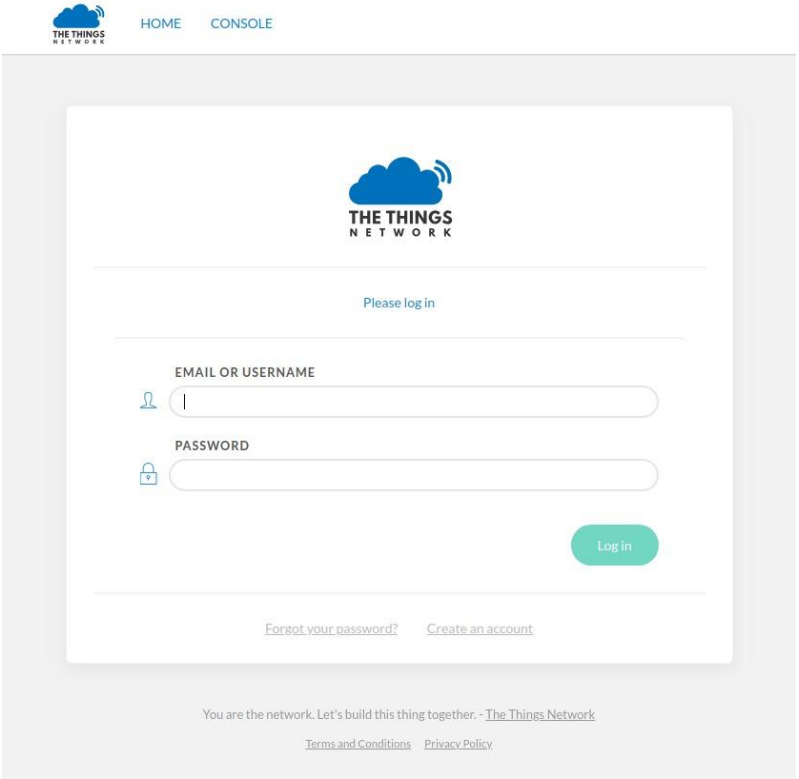


Compte per el taller

URL: console.thethingsnetwork.org

User: ttncat-tallers

Clau: ttncat-tallers

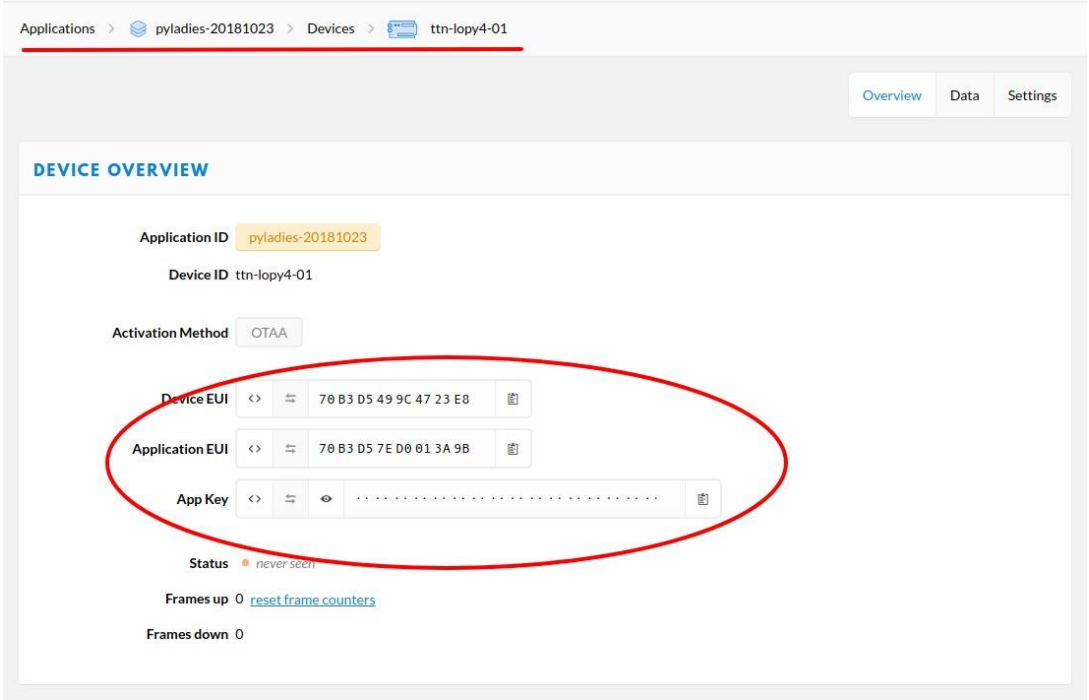


Consola TTN

Cada dispositiu té unes claus per accedir a la xarxa (**network session key**) i connectar-se amb un aplicatiu (**application session key**).

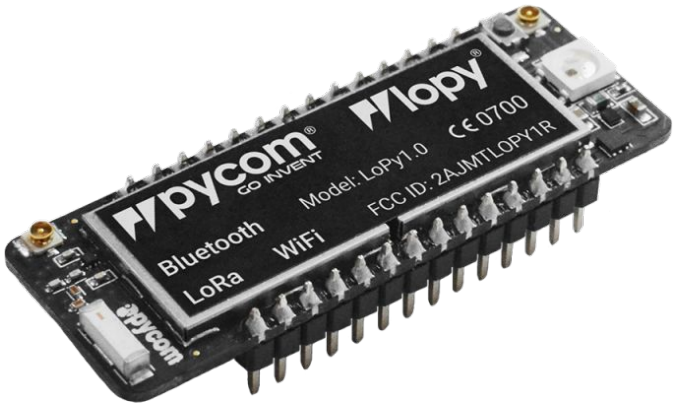
Aquestes claus es poden gravar al firmware (**ABP, Activation by Personalisation**) o negociar-se en el moment de fer el *join* (**OTAA, Over the Air Activation**).

Amb OTAA el dispositiu caldrà estar donat d'alta a l'aplicatiu (**Device EUI**).



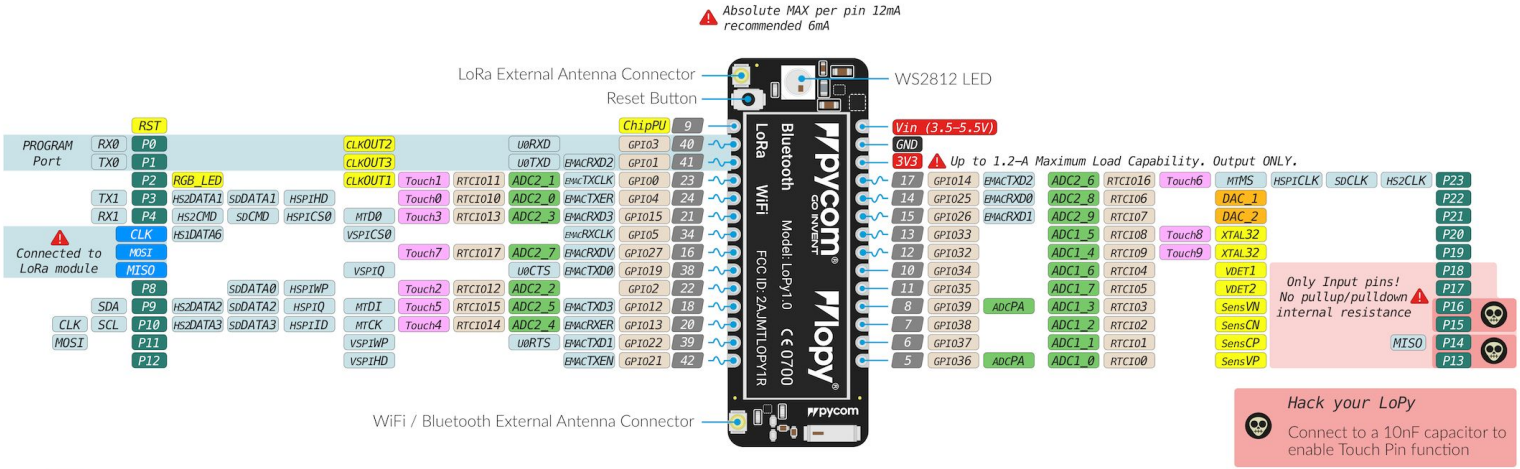
HARDWARE

HARDWARE (LOPY / LOPY4)



- Espressif ESP32. Xtensa dual-core 32-bit LX6 microprocessor, up to 240 MHz
- 512Kb RAM, 4MB external flash
- 802.11 b/g/n 16mbps
- Bluetooh LE and classic
- 868 LoRa +14dBm, LoRaWan stack class A&C
- 2xUART, 2xSPI, I2C, I2S, micro SD card
- 8x 12bits ADCs
- 4x 16bits Timers
- 24x GPIOs
- Ethernet MAC
- SSL/TLS support
- WPA Enterprise security
- FCC & CE certs

LOPY



- Power
- GND
- Serial Pin
- Analog Pin
- Control
- Physical Pin
- Port Pin
- Touch Pin
- DAC Pin
- PWM Pin

Low Level Bootloader

P2 + GND

Boot modes and safe boot

P12 + 3V3

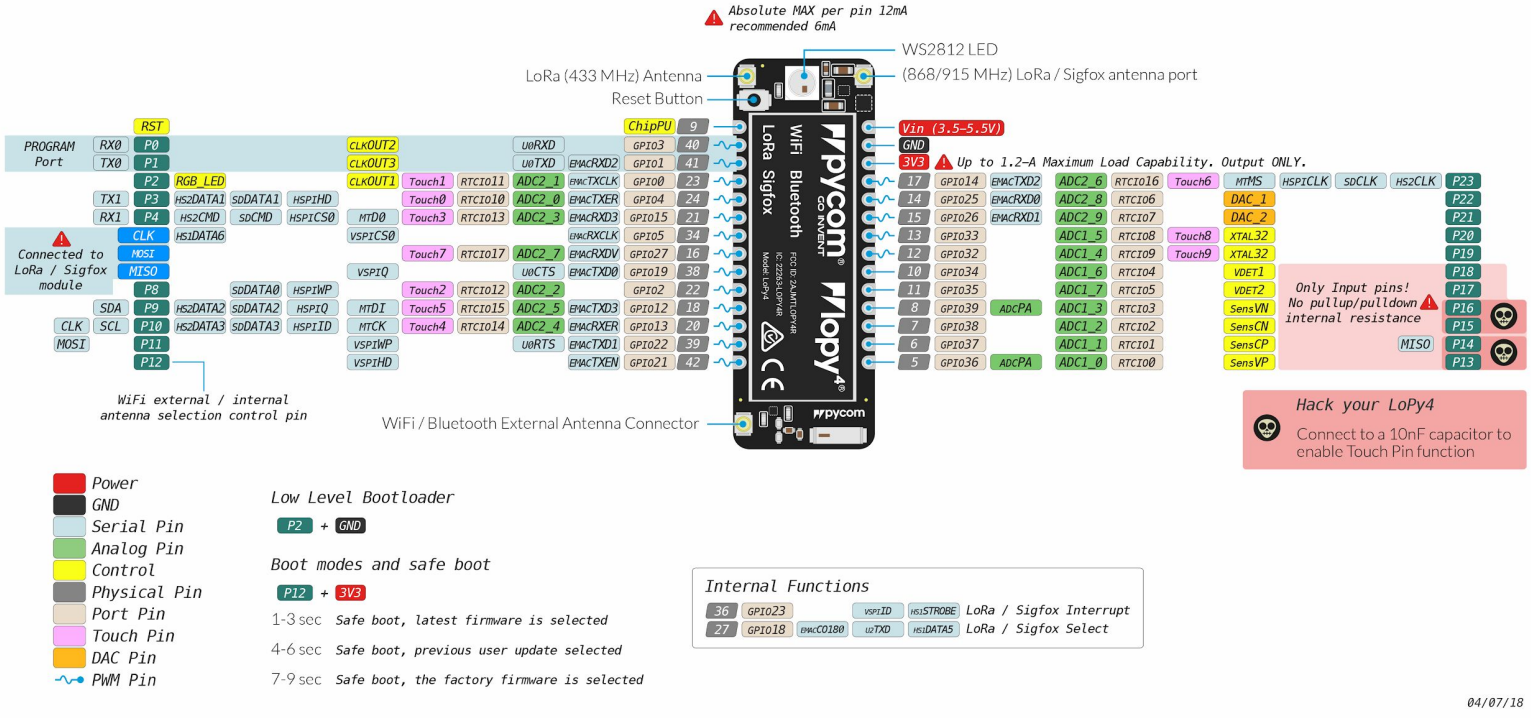
1-3 sec	Safe boot, latest firmware is selected
4-6 sec	Safe boot, previous user update selected
7-9 sec	Safe boot, the factory firmware is selected

Internal Functions

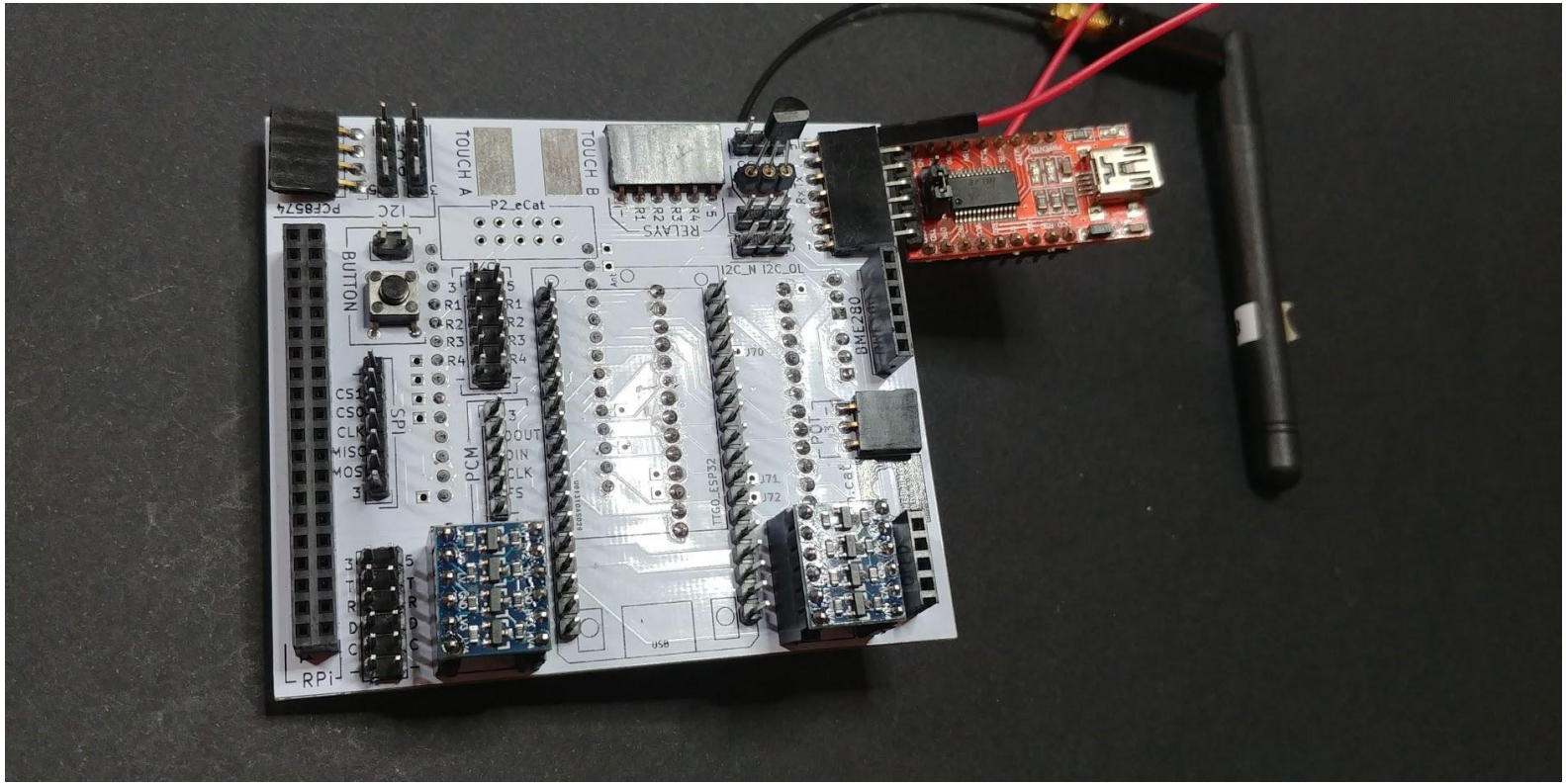
35	GPIO18	SPICLK	HSIDATA7	LoRa Reset
25	GPIO16	EMACCLKOUT	U0RXD	HSIDATA4	External Antenna Switch
36	GPIO23	SPIID	HSISTROBE	LoRa Interrupt
27	GPIO17	EMACCOL00	U2TXD	HSIDATA5	LoRa Select

22/03/18

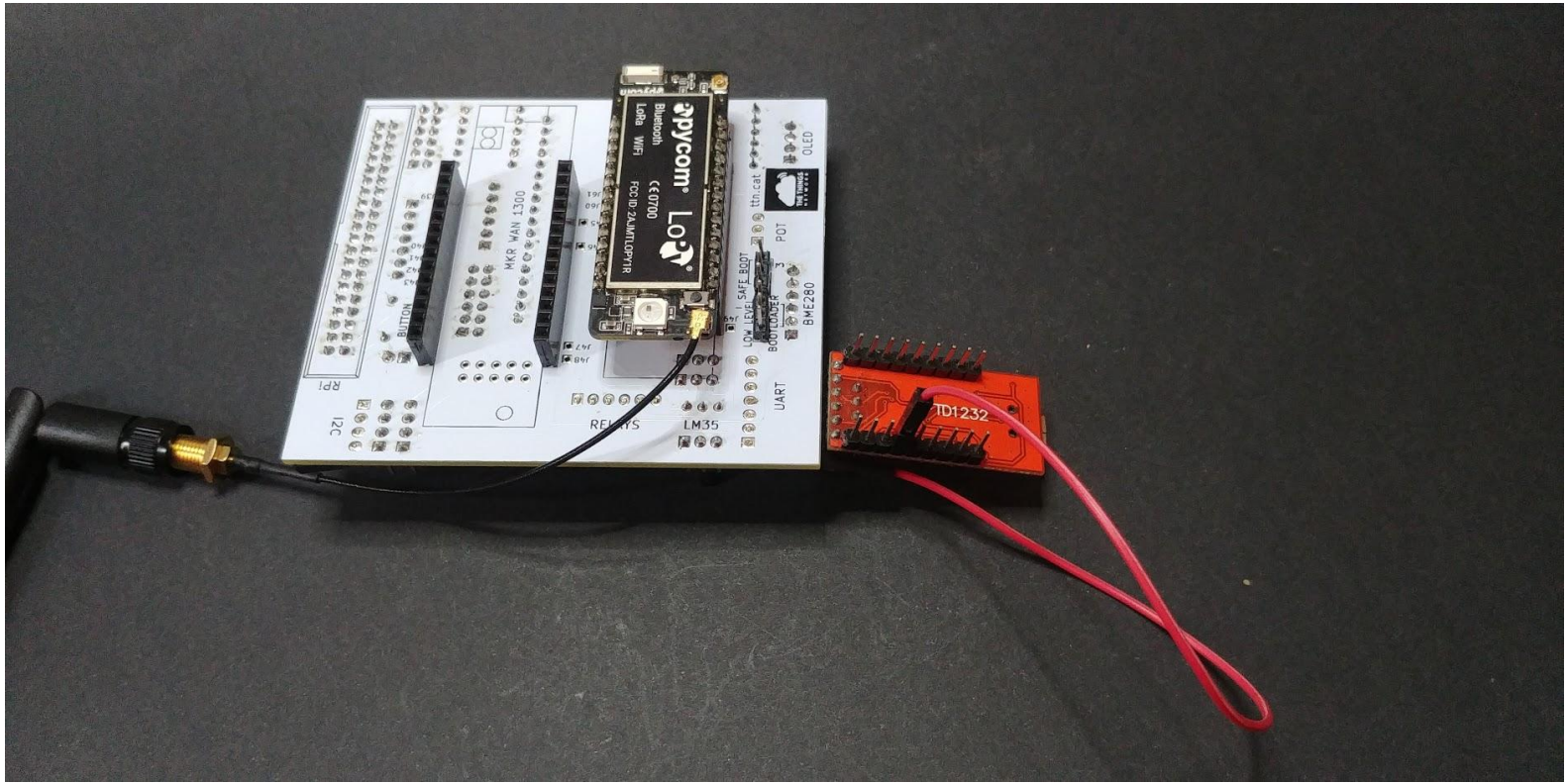
LOPY4


Absolute MAX per pin 12mA
recommended 6mA
WS2812 LED
LoRa (433 MHz) Antenna
(868/915 MHz) LoRa / Sigfox antenna port
Reset Button
WiFi Bluetooth
LoRa Sigfox
pycom
Vin (3.5–5.5V)
GND
3V3
Up to 1.2–A Maximum Load Capability. Output ONLY.
PROGRAM Port
Connected to LoRa / Sigfox module
WiFi external / internal antenna selection control pin
WiFi / Bluetooth External Antenna Connector
Only Input pins!
No pullup/pulldown
internal resistance
Hack your LoPy4
Connect to a 10nF capacitor to enable Touch Pin function
Power
GND
Serial Pin
Analog Pin
Control
Physical Pin
Port Pin
Touch Pin
DAC Pin
PWM Pin
Low Level Bootloader
P2 + GND
Boot modes and safe boot
P12 + 3V3
1-3 sec Safe boot, latest firmware is selected
4-6 sec Safe boot, previous user update selected
7-9 sec Safe boot, the factory firmware is selected
Internal Functions
LoRa / Sigfox Interrupt
LoRa / Sigfox Select
04/07/18

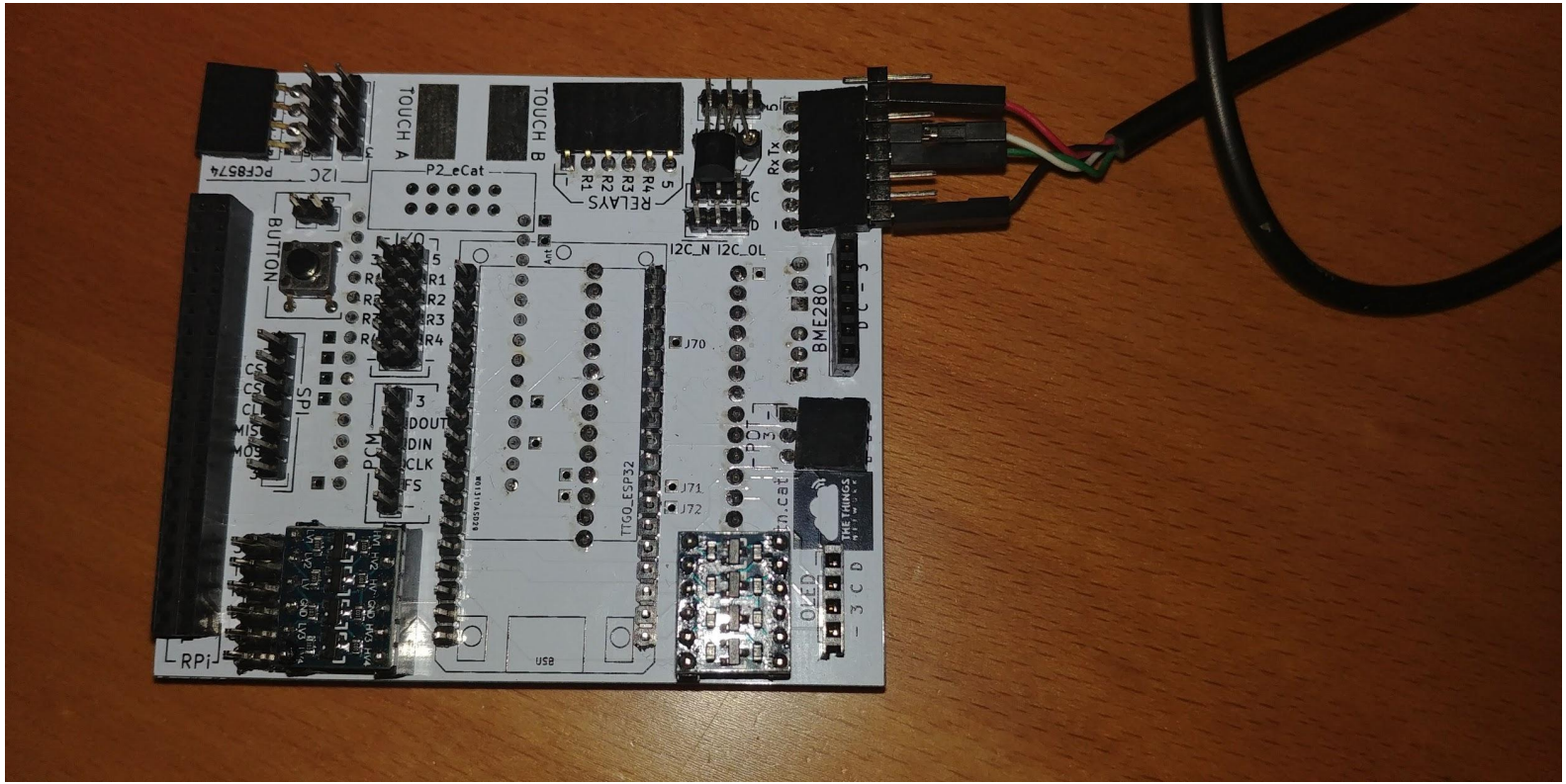
Carrier board



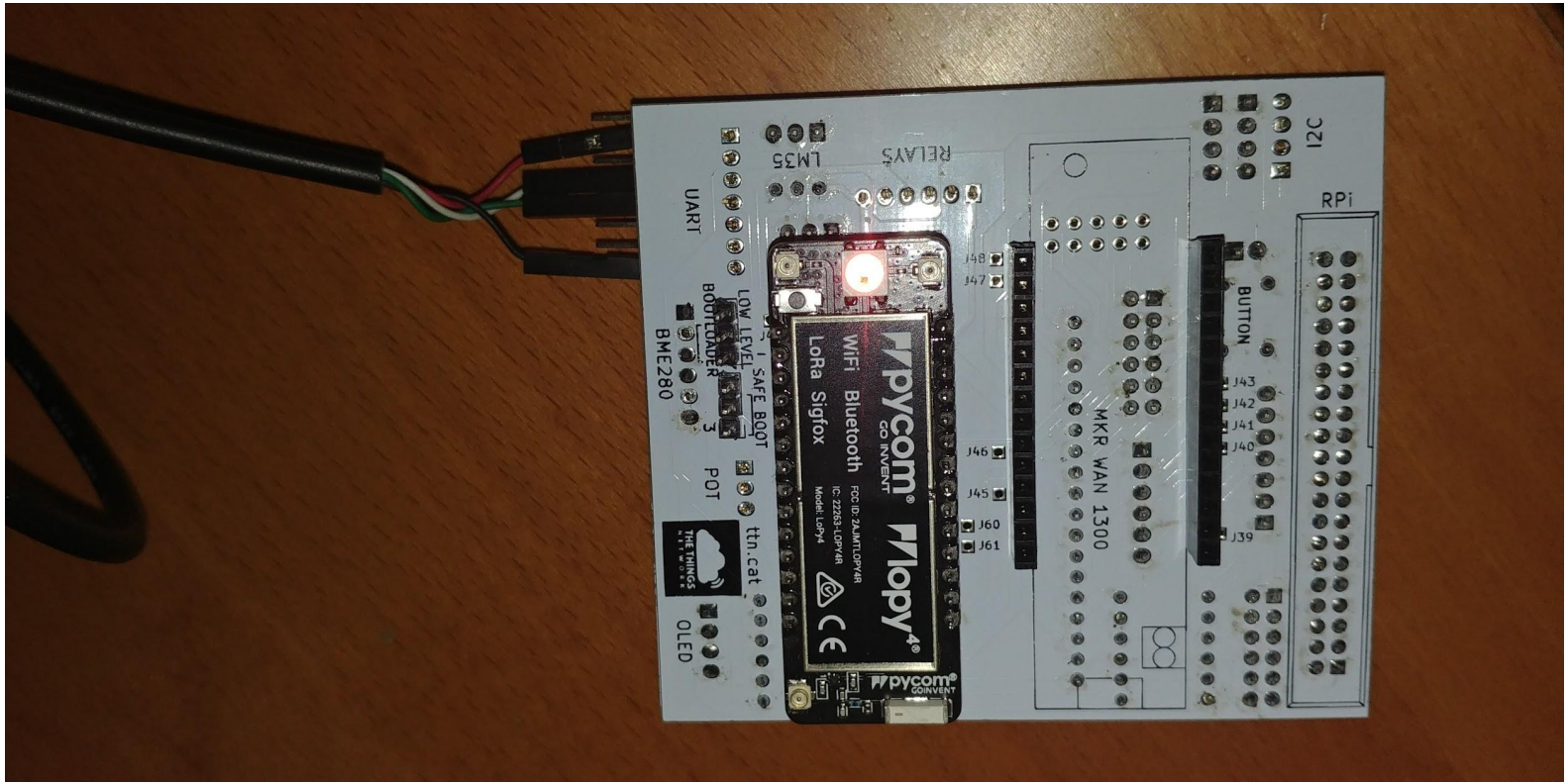
Carrier board



Carrier board



Carrier board



SOFTWARE

MICROPYTHON BY PYCOM

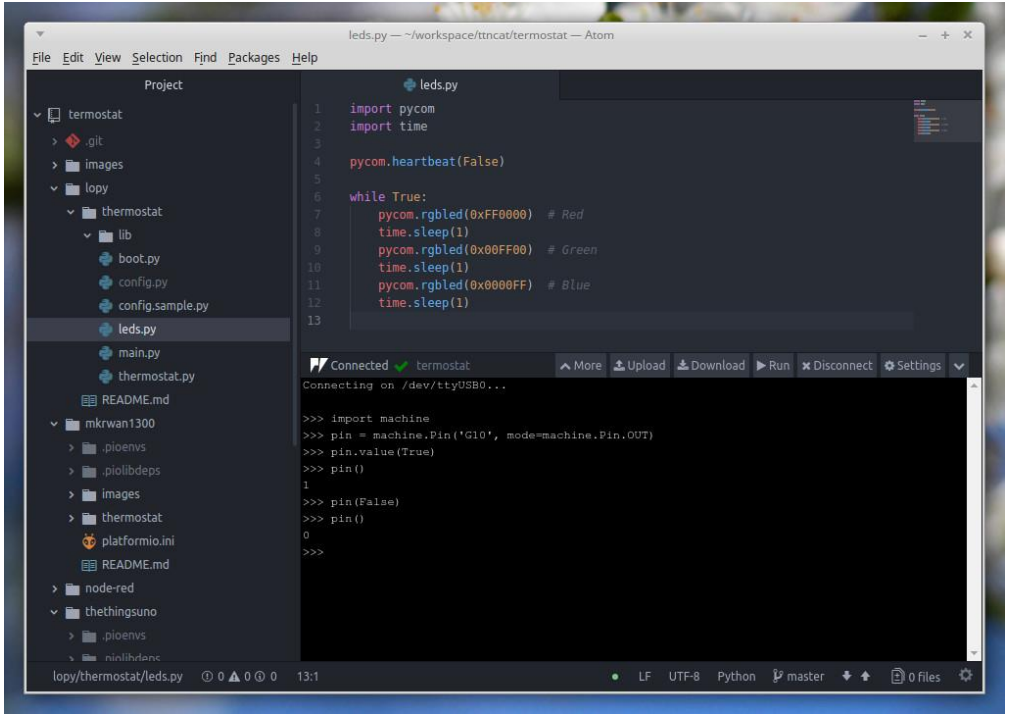
https://micropython.org/

https://github.com/micropython/micropython

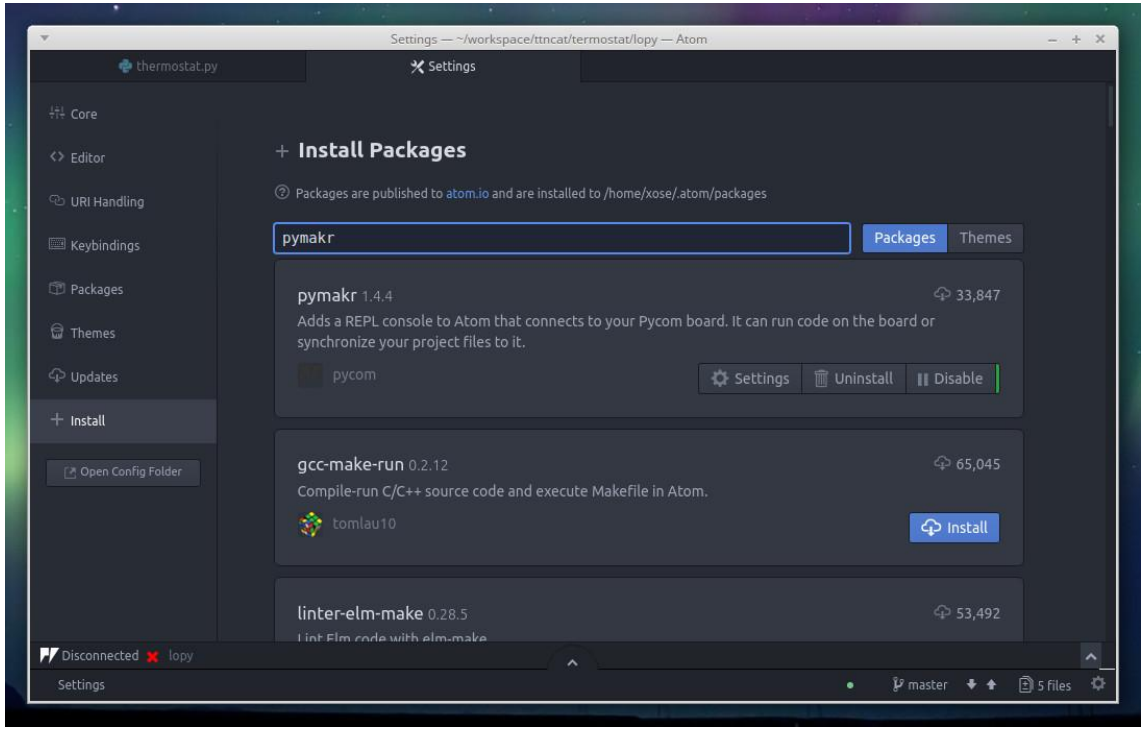
https://docs.pycom.io/

https://github.com/pycom/pycom-micropython-sigfox

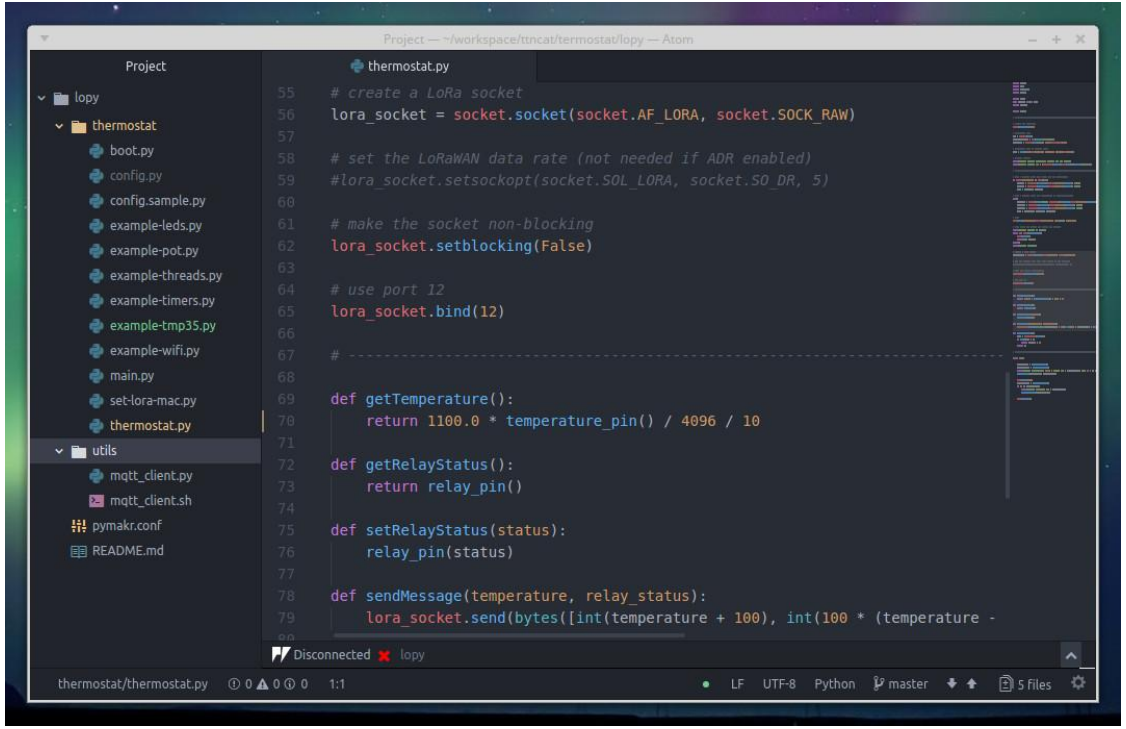
"MicroPython is intended for constrained environments, in particular, microcontrollers, which have orders of magnitude less performance and memory than "desktop" systems on which CPython3 runs."



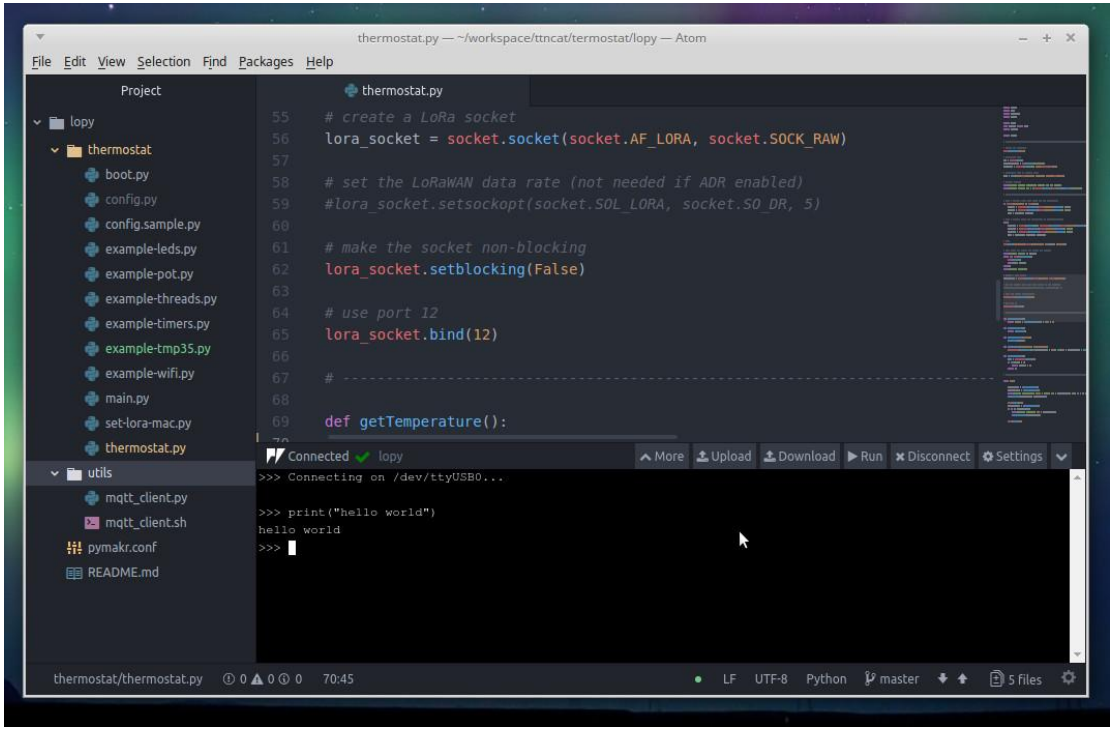
Atom - Instal·lar Pymakr



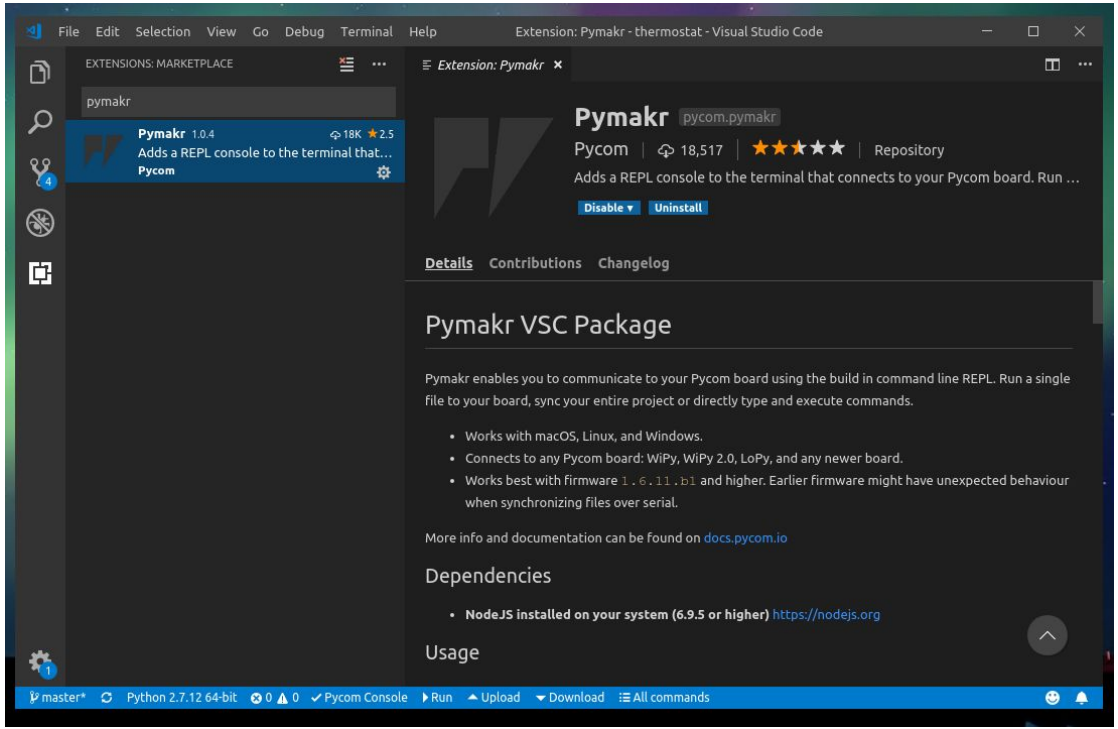
ATOM - PYMAKR


Project — ~/workspace/ttncat/termostat/lopy — Atom
Project
lopy
thermostat
boot.py
config.py
config.sample.py
example-leds.py
example-pot.py
example-threads.py
example-timers.py
example-tmp35.py
example-wifi.py
main.py
set-lora-mac.py
thermostat.py
utils
mqtt_client.py
mqtt_client.sh
pymakr.conf
README.md
thermostat.py
# create a LoRa socket
lora_socket = socket.socket(socket.AF_LORA, socket.SOCK_RAW)
# set the LoRaWAN data rate (not needed if ADR enabled)
#lora_socket.setsockopt(socket.SOL_LORA, socket.SO_DR, 5)
# make the socket non-blocking
lora_socket.setblocking(False)
# use port 12
lora_socket.bind(12)
def getTemperature():
return 1100.0 * temperature_pin() / 4096 / 10
def getRelayStatus():
return relay_pin()
def setRelayStatus(status):
relay_pin(status)
def sendMessage(temperature, relay_status):
lora_socket.send(bytes([int(temperature + 100), int(100 * (temperature -
Disconnected
lopy
thermostat/thermostat.py
1:1
LF
UTF-8
Python
master
5 files

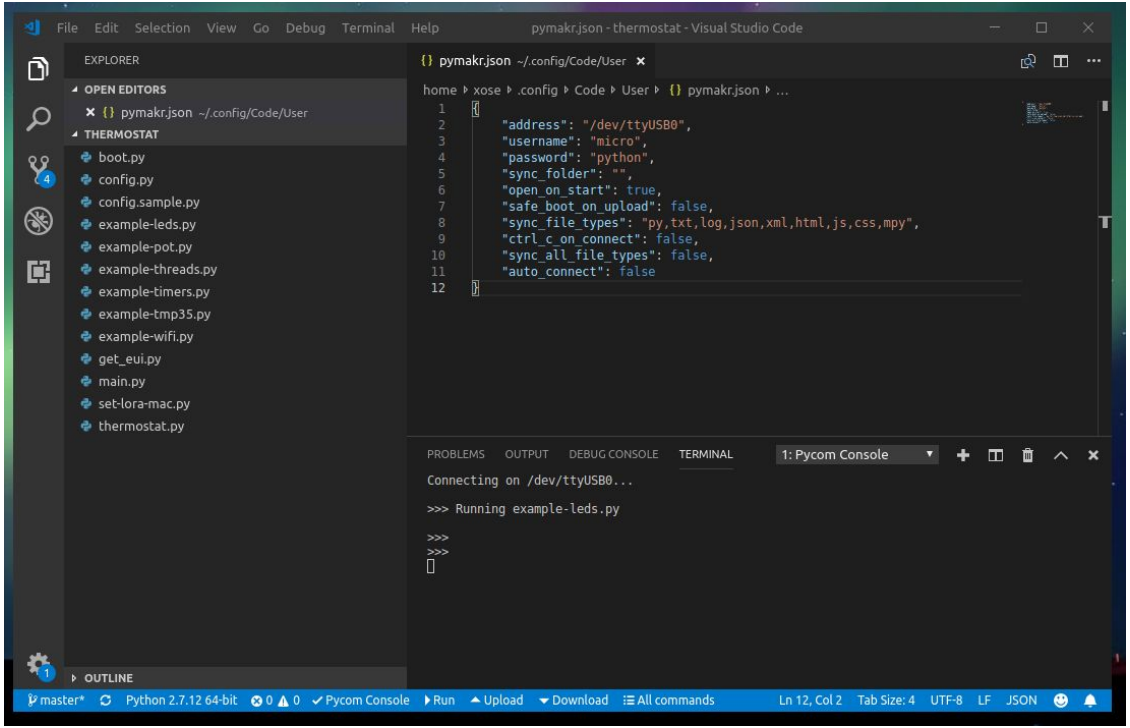
ATOM - PYMAKR REPL



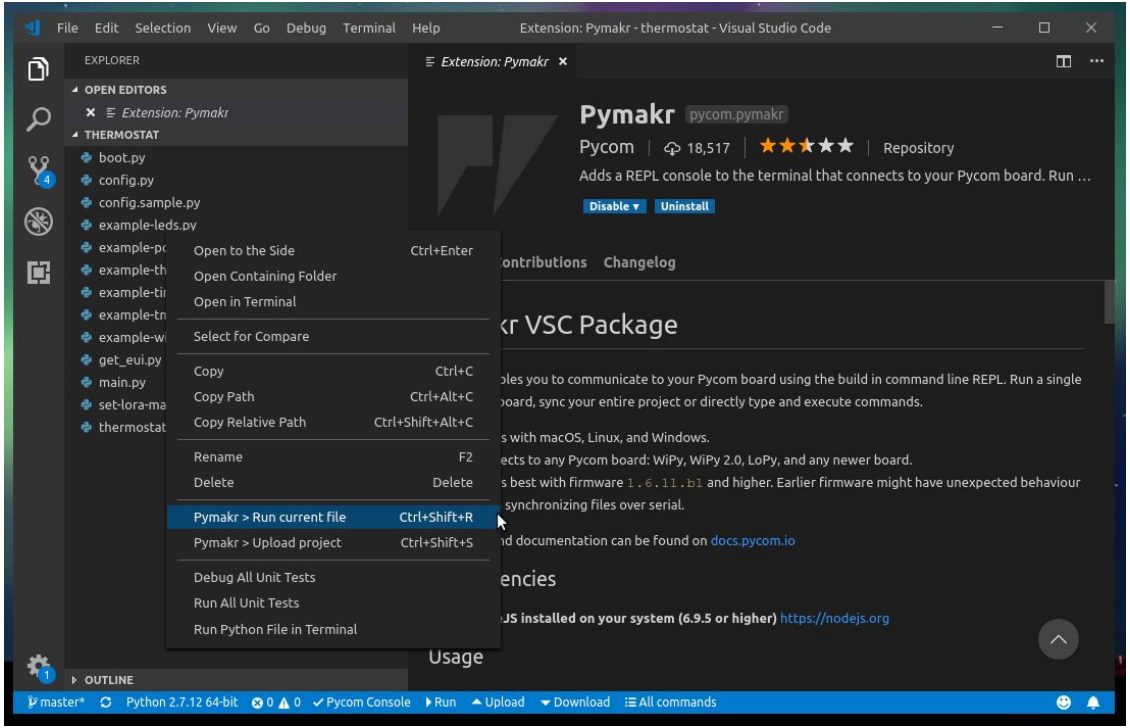
VSCode - Instal·lar Pymakr



VSCode - Pymakr Config

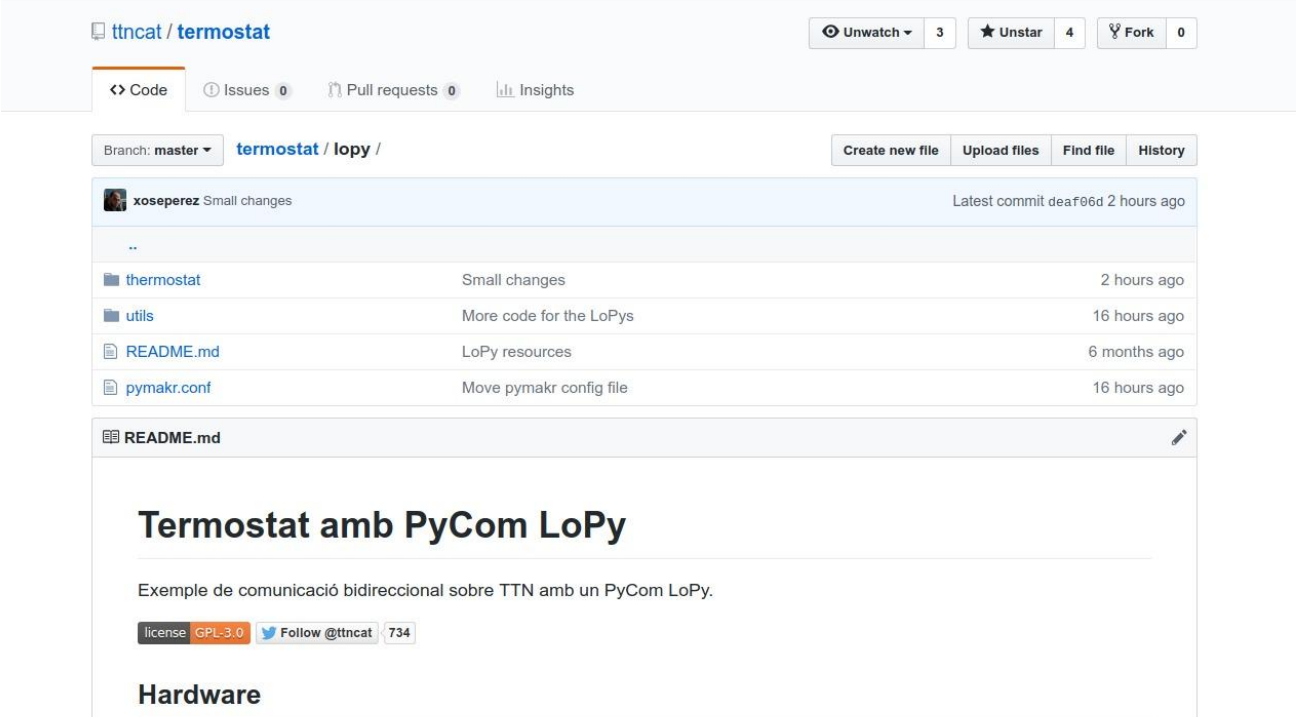


VSCode - PyMAKR Upload

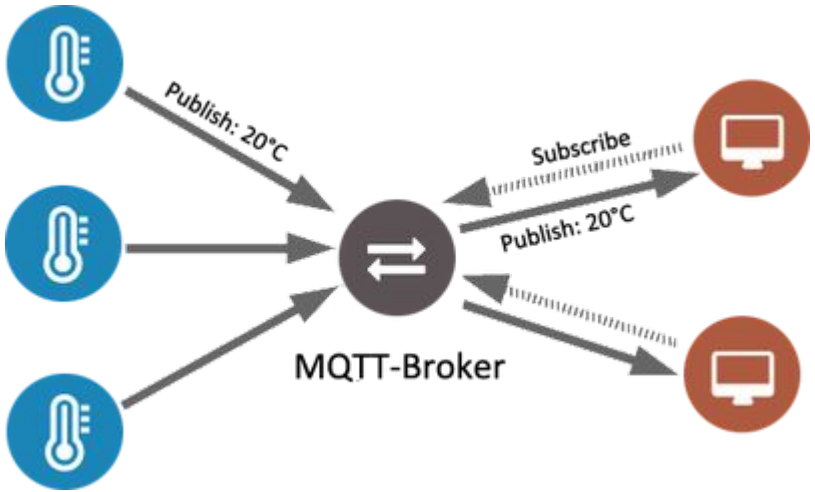


Exemples

https://github.com/ttncat/termostat/tree/master/lopy



MQTT



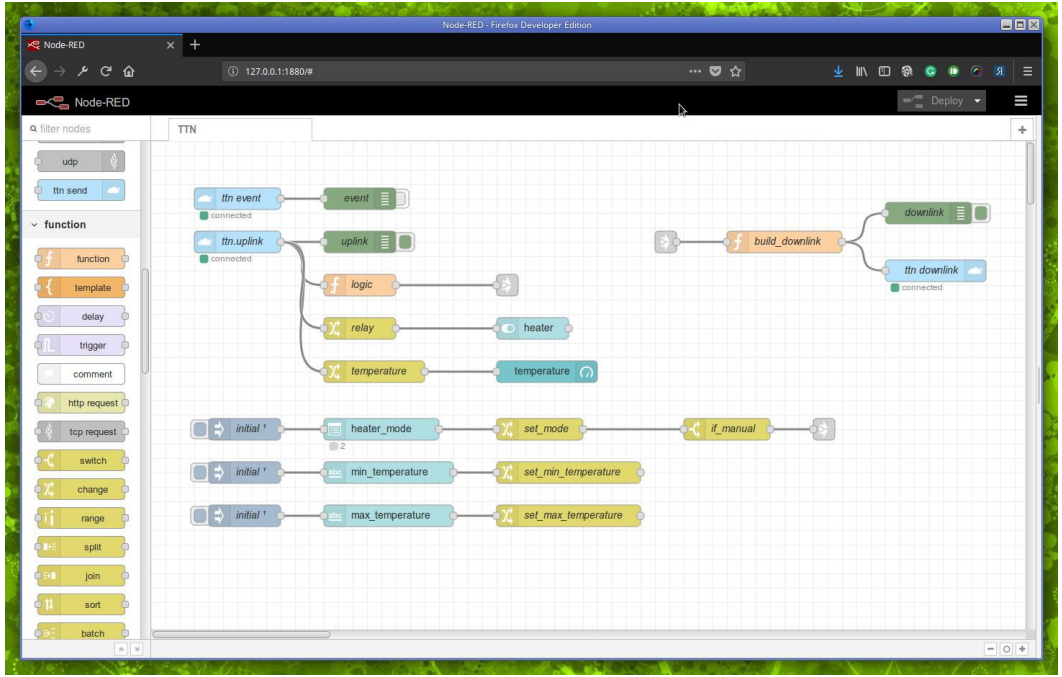

```python
import sys
import json
from pygments import highlight, lexers, formatters
import paho.mqtt.client as mqtt

# configuration
app_id = "test"
access_key = "ttn-account-v2.ALT....."

# callback functions
def on_connect(client, userdata, flags, rc):
    print("Subscribing...")
    # subscribe for all devices of user
    client.subscribe('+/devices/+/up')

def on_subscribe(client, userdata, mid, granted_qos):
    print("Subscribed")

def on_message(client, userdata, msg):
    formatted_json = json.dumps(json.loads(msg.payload), indent=4)
    colorful_json = highlight(unicode(formatted_json, 'UTF-8'), lexers.Jso
    print(colorful_json)

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message
client.on_subscribe = on_subscribe
client.username_pw_set(app_id, access_key)

print("Connecting...")
client.connect("eu.thethings.network", 1883, 60)

# and listen to server
client.loop_forever()
```

Aplicatiu (node-red)

L'aplicatiu de destí pot ser el que volguem. Ens podem connectar a TTN des de qualsevol aplicatiu que "parli" MQTT.

Node-RED és un entorn de treball on podem definir la lògica de l'aplicació a partir de nodes que reben, manipulen i re-envien missatges.

La gent de TTN ha creat un *plugin* específic (node-red-contrib-ttn) que ens permet rebre notificacions i rebre missatges (*uplink*) o enviar-ne (*downlink*).

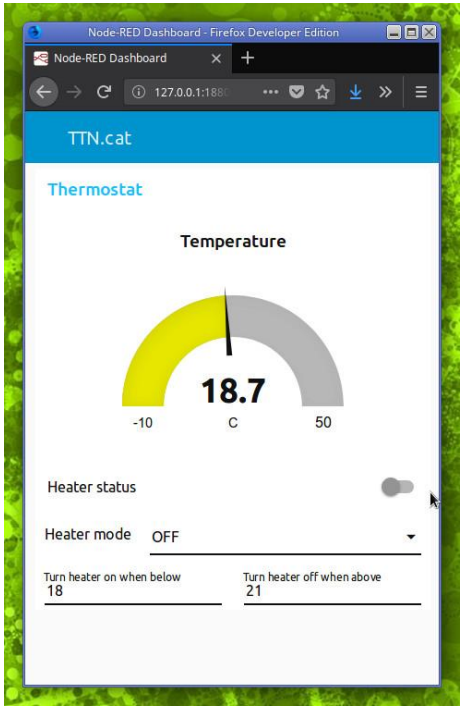


Aplicatiu (node-red)

A més Node-RED disposa d'un senzill entorn gràfic on visualitzar la informació i interactuar amb ella (node-red-dashboard).

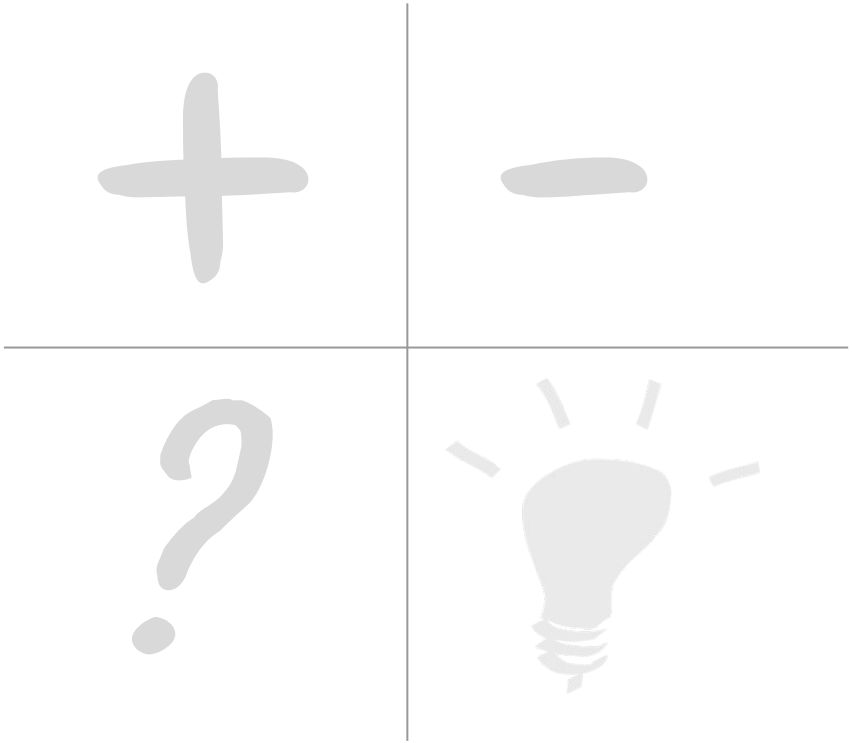
Aquest entorn ens permet fer, per exemple, el control del nostre termòstat i definir si volem que el calefactor estigui obert, tancat o que depengui de la temperatura.

Un aspecte important a tenir en compte és que Node-RED és una aplicació web, de manera que el podem tenir executant-se en un servidor web (que suporti node.js) i accedir-hi des de qualsevol lloc: des de l'ordinador de casa, el mòbil en el tren cap a la feina o des de Fuerteventura.



Què t'ha semblat

GRÀCIES



thethingsnetwork.cat - ttn.cat
@ttncat a twitter