Viladecans

XOIC xarxa oberta i comunitària d'internet de les coses

THE THINGS NETWORK CATALUNYA

Dispositius LoRaWAN

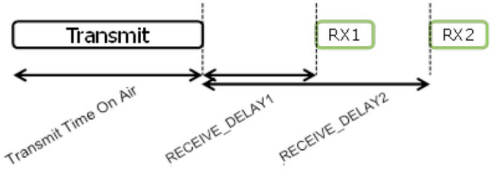
Teoria

Característiques d’un dispositiu LoRaWAN

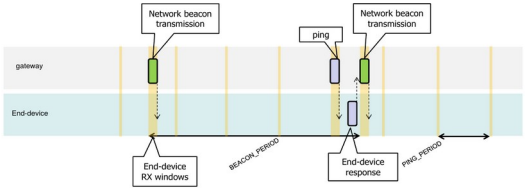
Tipus de dispositius

Tres tipus de dispositius (*device classes*):

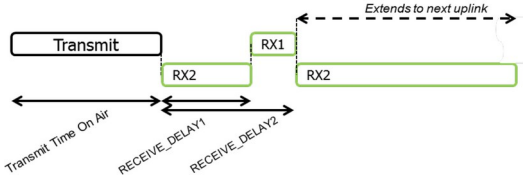
- **Classe A**: Tenen una comunicació bidireccional parcial, donat que només poden rebre dades de la Gateway quan han enviat prèviament un paquet. Aquesta classe és la que menys energia necessita, els dispositius estan normalment dormint. No són temps-real.



- **Classe B**: Aquesta classe de dispositius estan **sincronitzats** amb la Gateway corresponent de manera que poden rebre paquets de dades des de la Gateway a certs intervals pre-negociats (*beacons*) sense la necessitat d'haver enviat un paquet prèviament. No són temps-real, però són previsibles.



- **Classe C**: Els dispositius d'aquesta classe estan permanentment en disposició de rebre paquets des de la Gateway (sempre que no estigui enviant). Aquesta classe és la que més energia consumeix. Són temps-real.



Velocitat adaptativa (ADR)

Hi ha dues maneres d'arribar més lluny: cridar més o parlar més a poc a poc. Ambdues, però, consumeixen més energia. La xarxa s'autogestiona per optimitzar consum i congestió:

- Si ADR està activat, la xarxa ajusta el **SF** i **potència TX** del dispositiu:
  - Si bona cobertura → Disminueix SF (més ràpid, menys consum, menys rang)
  - Si mala cobertura → Augmenta SF (més lent, més consum, més rang)
- Millora el funcionament de la xarxa
  - Reduint el temps en aire → **menys col·lisions**
  - Reduint els dispositius que una passarel·la ha de gestionar → **més capacitat**

Pot haver-hi **situacions on no es recomana ADR**:

- Dispositius mòbils
- Dispositius amb entorn molt variable

Cicle de treball

El cicle de treball (***duty cycle***) màxim està regulat per l'ETSI (*European Telecomunications Standards Institute*). Aquest defineix per la banda 868 MHz:

- Ocupació de l'1% del temps
- 1% de 3600 segons → 36 segons per hora
- Mòduls ràdio fan càlcul automàtic i no permeten sobre-passar-lo

A sobre d'aquest, diferents xarxes poden imposar cicles més restrictius. TTN defineix una «política de joc net» (***Fair Access Policy***) que imposa:

- Enviar: 30 segons cada 24 hores (*uplink*)
- Rebre: 10 missatges cada 24 hores (*downlink*)

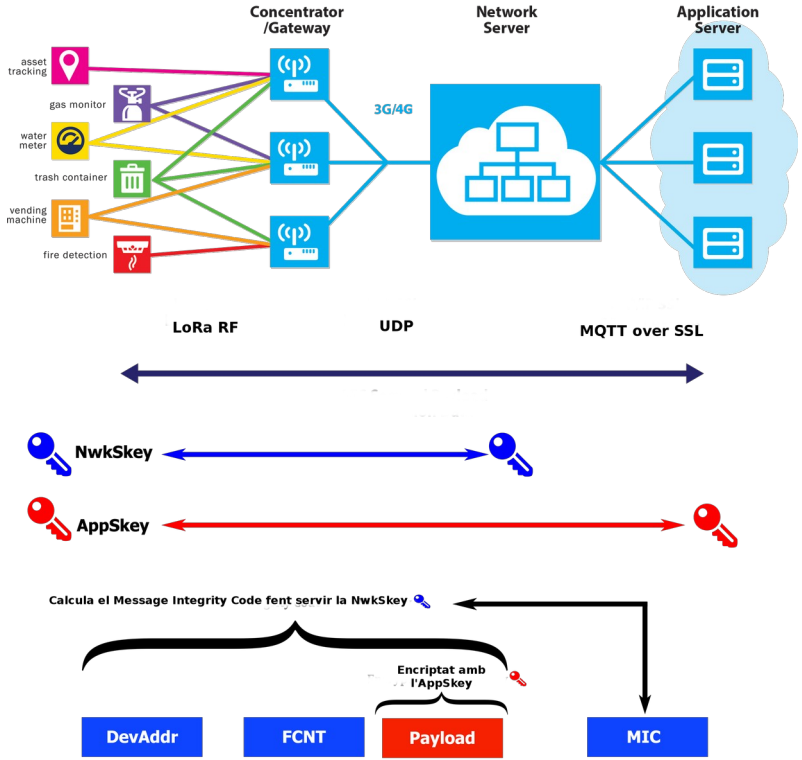
Seguretat

Les dades del sensor (***payload***) estan encriptades amb l'AppSkey (AES128).

El missatge està signat amb el MIC (**codi d'integritat del missatge**), que es calcula amb el *payload*, el *devaddr*, el *fcnt* i fent servir la NwkSkey.

La xarxa "no pot saber" què s'està enviant, només l'aplicació.

TTN permet descodificar el missatge en el *backend*, per tant es recomana fer servir un *handler* segur per connectar-se (MQTT sobre SSL).



Activació

Un dispossitiu connectat a una xarxa LoRaWAN ha de emmagatzemar els següents valors:

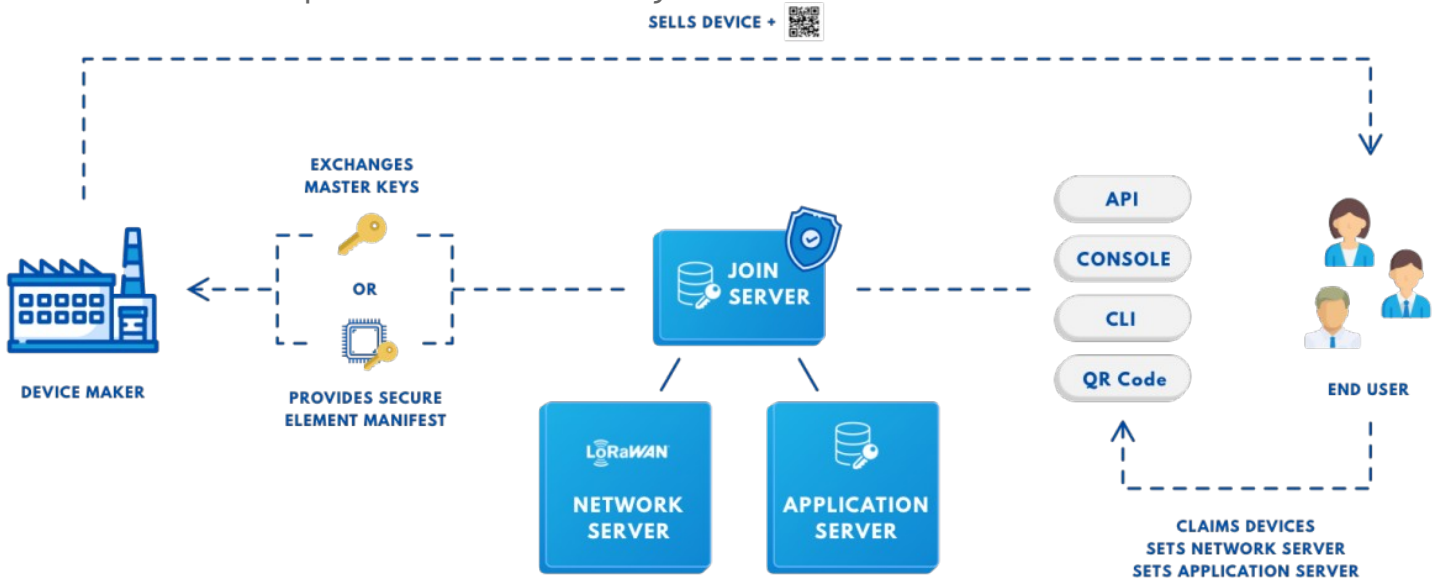
- DevAddr (una adreça)
- NwkSKey (clau de xarxa per la sessió)
- AppSKey (clau d'aplicació per la sessió)

LoRaWAN defineix dos mètodes per disposar d'aquestes dades:

- **ABP** (*Activation By Personalization*): Cada dispositiu porta les aquest valors (identificador I claus) pre-programades. En general és més insegur (les claus estan al dispositiu) però té l'avantatge que no cal negociació prèvia.
- **OTAA** (*Over The Air Activation*): Cal una negociació prèvia per cada sessió de connexió. És més segur i és el que normalment fan servir els dispositius comercials. Per realitzar aquesta negociació el dispositiu necessita:
  - DevEUI (identificador únic del dispositiu)
  - AppEUI o JoinEUI (identificador de l'aplicatiu en el qual el dispositiu està registrat)
  - AppKey (clau única de registre)

Activació (manifest file)

Alguns fabricants (com Microchip) comencen a proporconar *manifest files* per poder registrar automàticament els teus dispositius a través d'un *Join Server*.



Components

Circuits integrats

Semtech fabrica *transceivers* (ràdios) per nodes: SX1276, SX1277, SX1278 i SX1279...

Aquestes ràdios estan preparades per diferents freqüències i *spreading factors* i orientades a diferents mercats.

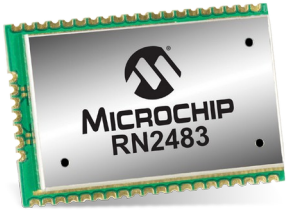
TTN a Europa és comparible amb dispositius basats en SX1276 (137-1020Mhz i SF6-12, tot i que l'ús de SF6 està limitat).

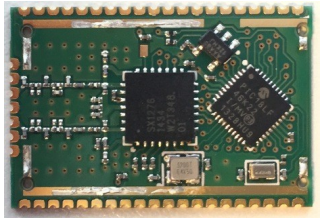
Semtech llicencia la fabricació a altres fabricants. De moment només HopeRF, Microchip i ST fabriquen xips.

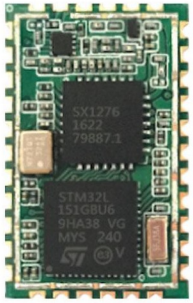
Cada cop hi ha més i més xips i mòduls disponibles i és més i més fàcil integrar LoRa en un projecte.

Mòduls

Una aproximació habitual a l'hora d'integrar LoRaWAN en un projecte és fer servir mòduls. Aquest mòduls inclouen un microcontrolador encarregat de la pila LoRaWAN i un xip LoRa per la transmissió i recepció. Habitualment implementen un protocol sèrie per interaccionar amb el controlador principal del producte.







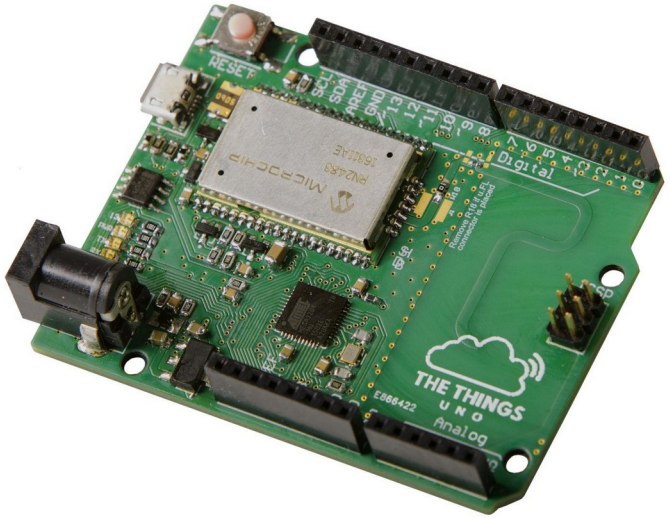
Dispositius de desenvolupament

Plaques de desenvolupament

Més enllà de les plaques d'avaluació i desenvolupament dels fabricants (cares i orietades a un mercat molt professional) hi ha moltes opcions de plaques de desenvolupament pensades per proves de concepte i, algunes, per desplegaments petits.

The Things Uno

- Arduino Leonardo compatible board
- Microchip Atmel ATMEGA32U4
  - 8-bit AVR RISC-based
  - 32KB flash
  - 2.5KB SRAM
  - 1KB EEPROM
- Microchip RN2483 LoRaWAN
  - PIC-based
  - UART interface
- C-programable
- Arduino IDE compatible
- ~48€



Arduino MKRWAN 1300

- Arduino MKR family
- Microchip Atmel SAMD21G
  - Cortex M0+ 32bits
  - 48MHz
  - 256Kb flash
  - 32Kb SRAM
- Murata CMWX1ZZABZ LoRaWAN module
  - STM32-based
  - UART interface
- C -programable
- Arduino IDE compatible
- ~35€

BastWAN

- RAK4260
  - Microchip SAMR34 (ARM Cortex M0-
  - 48 MHz
  - 256Kb Flash
  - 32Kb RAM
- LoRa radio in module based on SX127X
- ATECC608A crypto chip
- C
- Arduino IDE compatible
- Designed by Electronic Cats a OSHW
- Manufactured by RAKwireless
- ~12€

WisBlock

- Modular prototyping, PoC, production ready platform
- LPWAN Module
  - Nordic nRF52840 (ARM Cortex M4F)
  - 64 MHz
  - 1MB flash
  - 256Kb RAM
- Several radios
  - Bluetooth LE
  - LoRa
- SX1272
  - SPI interface
- C
- Arduino IDE compatible
- ~30€ (base + LPWAN module)

CubeCell

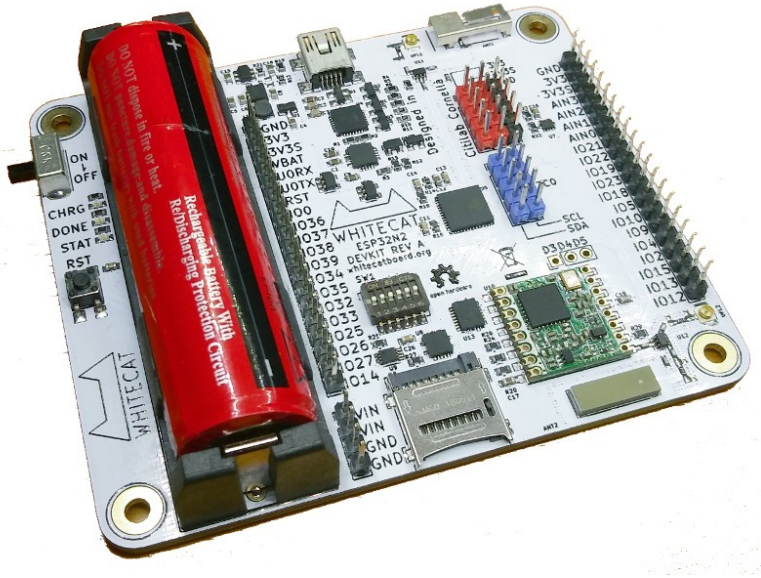
- ASR6052
  - ARM Cotex M0+
  - 48 MHz
  - 128Kb flash
  - 16Kb SRAM
- SX1276
  - SPI interface
- C
- Arduino IDE compatible
- ~12€

PyCom LoPy4

- Espressif ESP32 based
  - Xtensa dual-core 32-bit LX6
  - 240 MHz
  - 4MB external flash
  - 512Kb RAM
- Several radios
  - WiFi
  - Bluetooth LE
  - LoRa
  - Sigfox
- SX1276
  - SPI interface
- MicroPython programable
- ~35€

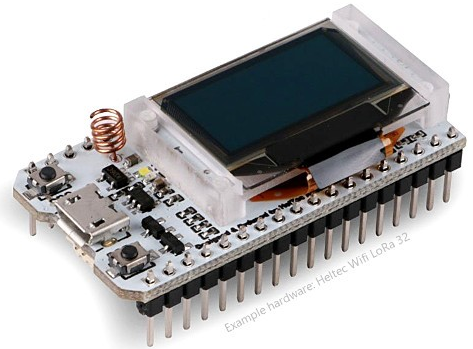
WhiteCatBoard N2

- Espressif ESP32 based
  - Xtensa dual-core 32-bit LX6
  - 240 MHz
  - 4MB external flash
  - 512Kb RAM
- Several radios
  - WiFi
  - Bluetooth LE
  - LoRa
- Integrates a HopeRF95 (SX1276)
  - SPI interface
- KM0 (designed at the Citilab Cornellà)
- Lua o Blockly
- ~40€



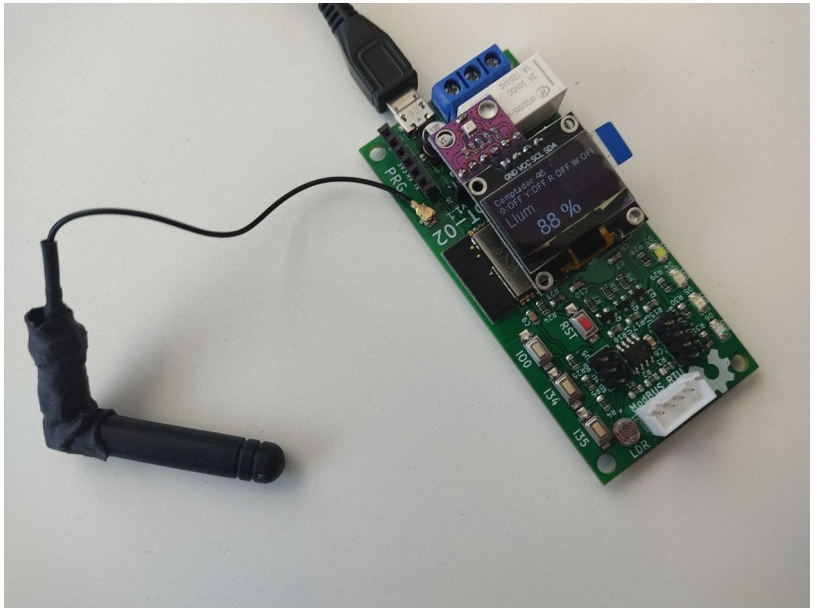
TTGO / HELTEC LoRa32

- Espressif ESP32 based
  - Xtensa dual-core 32-bit LX6
  - 240 MHz
  - 4MB external flash
  - 512Kb RAM
- Several radios
  - WiFi
  - Bluetooth LE
  - LoRa
- SX1276
  - SPI interface
- C
- Arduino IDE compatible
- ~12€



Kit IoT-02

- Espressif ESP32 based
  - Xtensa dual-core 32-bit LX6
  - 240 MHz
  - 4MB external flash
  - 512Kb RAM
- Several radios
  - WiFi (embedded antenna)
  - Bluetooth LE (embedded antenna)
  - LoRa + antenna
- SX1276
  - SPI interface
- 3 user buttons, 4 leds, LDR
- Latching relay
- 3V3 UART board + mini USB cable
- I2C OLED
- I2C BME280
- MODBUS RTU + USB / RS485 +cable
- C
- Arduino IDE compatible
- Micropython
- ~58€

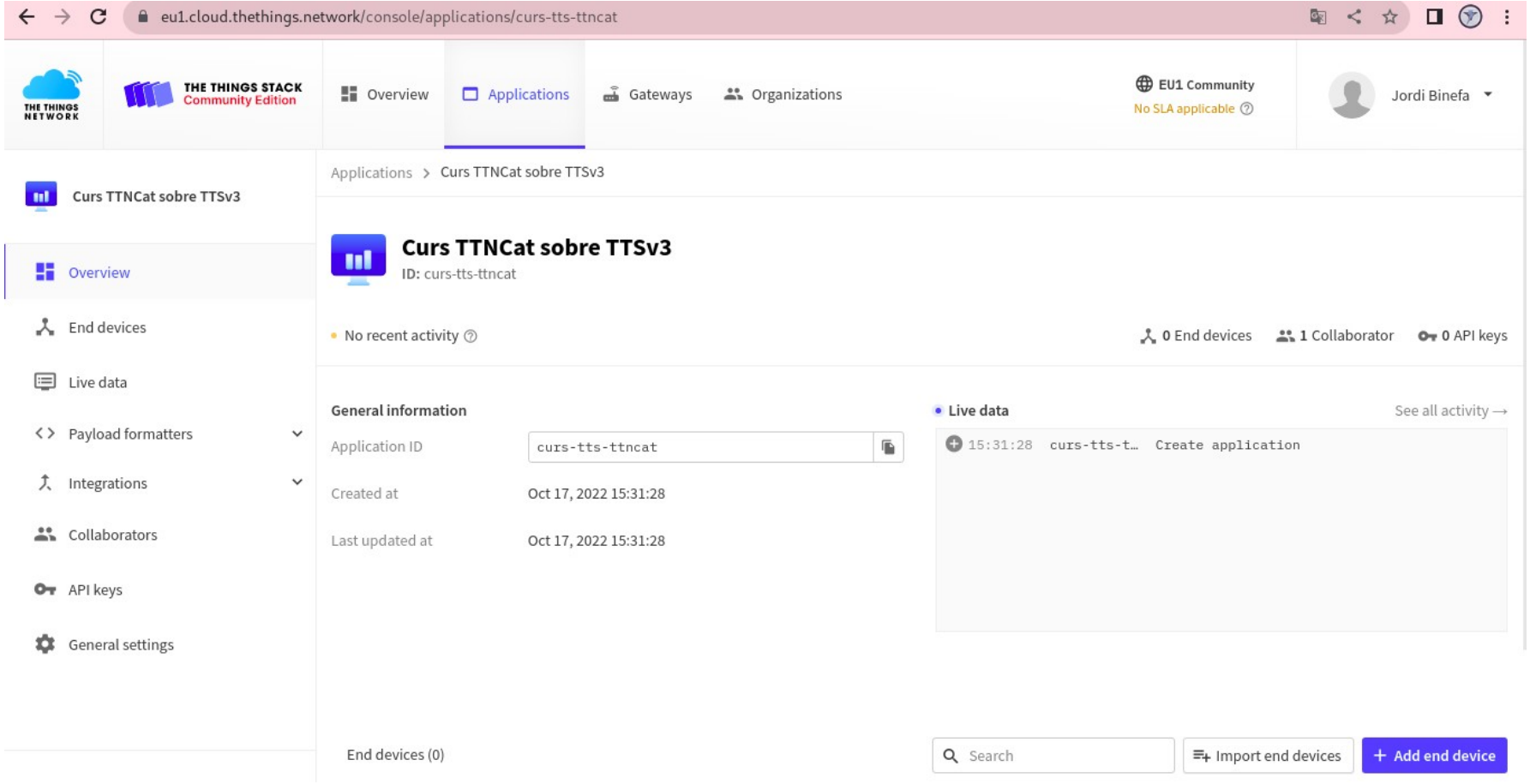


Pràctica

Alta de dispositius OTAA

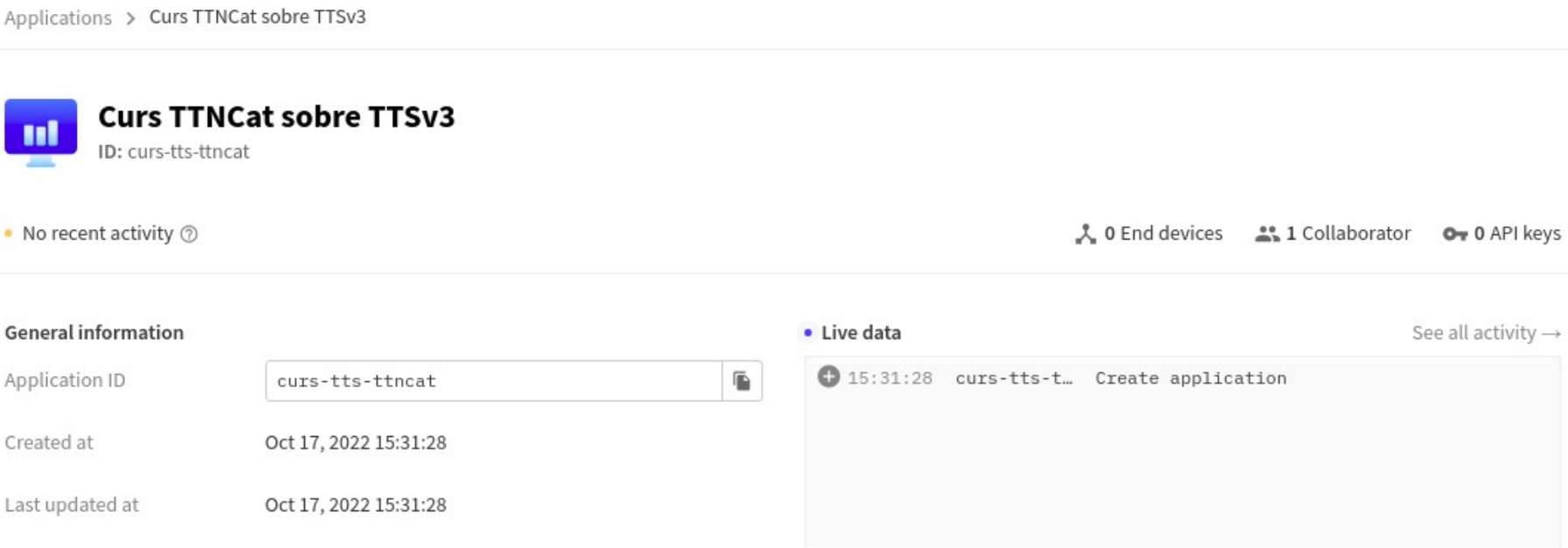
Alta OTAA (1)

Viladecans



Alta OTAA (2)

Viladecans



Alta OTAA (3)

Viladecans

Applications > Curs TTNCat sobre TTSv3 > End devices

End devices (0)

Search | Import end devices | + Add end device

ID | Name | DevEUI | JoinEUI | Last activity

No items found

Applications > Curs TTNCat sobre TTSv3 > End devices

Register end device

Does your end device have a QR code? Scan it to speed up onboarding.

Scan end device QR code | Learn more

End device type

Input Method

- (•) Select the end device in the LoRaWAN Device Repository
- ( ) Enter end device specifics manually

End device brand *

Type to search…

Cannot find your exact end device? Get help here and try **enter end device specifics manually** option above.

Applications > Curs TTNCat sobre TTSv3 > End devices

Register end device

Does your end device have a QR code? Scan it to speed up onboarding.

Scan end device QR code | Learn more

End device type

Input Method

- ( ) Select the end device in the LoRaWAN Device Repository
- (•) Enter end device specifics manually

Frequency plan *

Select...

LoRaWAN version *

Select...

Regional Parameters version *

Select...

To continue, please enter versions and frequency plan information

Alta OTAA (4)

Viladecans

Register end device

Does your end device have a QR code? Scan it to speed up onboarding.

Scan end device QR code Learn more

End device type

Input Method

- ○ Select the end device in the LoRaWAN Device Repository
- ◉ Enter end device specifics manually

Frequency plan *
Europe 863-870 MHz (SF9 for RX2 - recommended)

LoRaWAN version *
LoRaWAN Specification 1.0.2

Regional Parameters version *
RP001 Regional Parameters 1.0.2

Show advanced activation, LoRaWAN class and cluster settings

◉ Enter end device specifics manually

Frequency plan *
Europe 863-870 MHz (SF9 for RX2 - recommended)

LoRaWAN version *
LoRaWAN Specification 1.0.2

Regional Parameters version *
RP001 Regional Parameters 1.0.2

Show advanced activation, LoRaWAN class and cluster settings

Activation mode *

- ◉ Over the air activation (OTAA)
- ○ Activation by personalization (ABP)
- ○ Define multicast group (ABP & Multicast)

Additional LoRaWAN class capabilities
None (class A only)

Network defaults

- ☑ Use network's default MAC settings

Cluster settings

- ☐ Skip registration on Join Server

Alta OTAA (5)

Viladecans

Show advanced activation, LoRaWAN class and cluster settings

Activation mode *

- (•) Over the air activation (OTAA)
- ( ) Activation by personalization (ABP)
- ( ) Define multicast group (ABP & Multicast)

Additional LoRaWAN class capabilities

None (class A only)

Network defaults

[x] Use network's default MAC settings

Cluster settings

[ ] Skip registration on Join Server

Provisioning information

JoinEUI *

81 1A DE CA 75 00 00 01 | Confirm

To continue, please enter the JoinEUI of the end device so we can determine onboarding options

Provisioning information

JoinEUI *

81 1A DE CA 75 00 00 01 | Reset

This end device can be registered on the network

DevEUI *

70 B3 D5 7E D0 05 68 B7 | Generate | 1/50 used

AppKey *

56 3B EC E8 F1 94 65 9D 9F 94 7C 4E 40 69 FF 3E | Generate

End device ID *

ttncat-otaa-01

This value is automatically prefilled using the DevEUI

After registration

- (•) View registered end device
- ( ) Register another end device of this type

Register end device

Alta OTAA (6)

Viladecans

Applications > Curs TTNCat sobre TTSv3 > End devices > ttncat-otaa-01

ttncat-otaa-01
ID: ttncat-otaa-01

↑ n/a ↓ n/a • No activity yet

Overview | Live data | Messaging | Location | Payload formatters | Claiming | General settings

General information

End device ID	ttncat-otaa-01
Frequency plan	Europe 863-870 MHz (SF9 for RX2 - recom…
LoRaWAN version	LoRaWAN Specification 1.0.2
Regional Parameters version	RP001 Regional Parameters 1.0.2
Created at	Oct 17, 2022 15:40:18

Activation information

AppEUI	81 1A DE CA 75 00 00 01
DevEUI	70 B3 D5 7E D0 05 68 B7
AppKey	•• •• •• •• •• •• •• •• •• •• •• •• •• •• •• ••

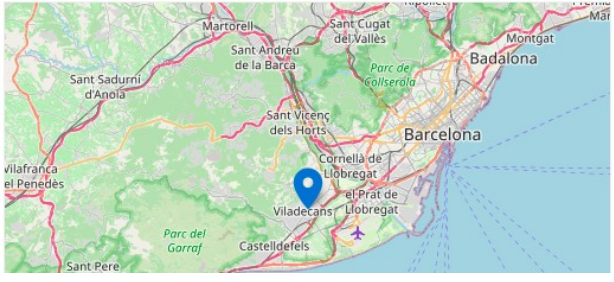
Session information

This device has not joined the network yet

Live data See all activity →

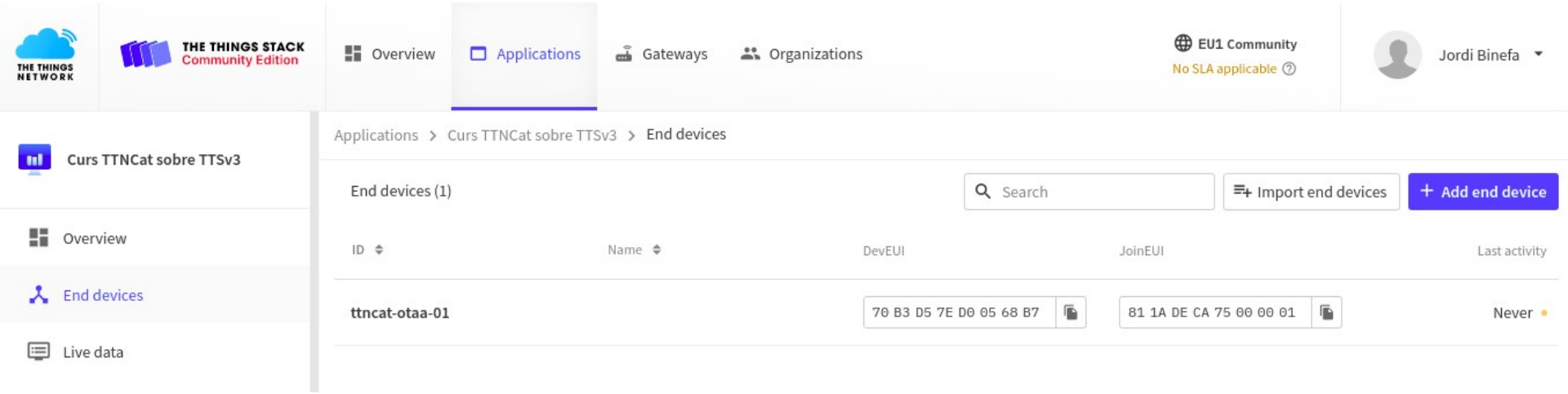
15:41:33 Update end device ["locations"]
15:40:18 Create end device

Location Change location settings →



Alta OTAA (7)

Viladecans



Alta OTAA (8)

Viladecans

ttncat-otaa-01

ID: ttncat-otaa-01

↑ n/a ↓ n/a • No activity yet

Overview | Live data | Messaging | Location | Payload formatters | Claiming | Ge

General information

End device ID	ttncat-otaa-01
Frequency plan	Europe 863-870 MHz (SF9 for RX2 - recommen…
LoRaWAN version	LoRaWAN Specification 1.0.2
Regional Parameters version	RP001 Regional Parameters 1.0.2
Created at	Oct 17, 2022 15:40:18

Activation information

AppEUI	0x01, 0x00, 0x00, 0x75, 0xCA, 0x… lsb
DevEUI	0xB7, 0x68, 0x05, 0xD0, 0x7E, 0x… lsb
AppKey	0x56, 0x3B, 0xEC, 0xE8, 0xF… msb

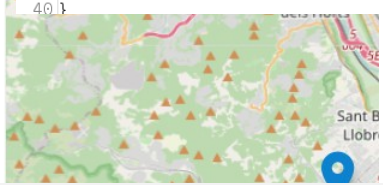
Session information

IoT-02-23_ttn-otaa_3_int_bme280_01 | Arduino 1.8.15

Fitxer Edita Esbós Eines Ajuda

IoT-02-23_ttn-otaa_3_int_bme280_01 | IoT-02_bme280.cpp | IoT-02_bme280.h | IoT-02_common.cpp | IoT-02_common.h

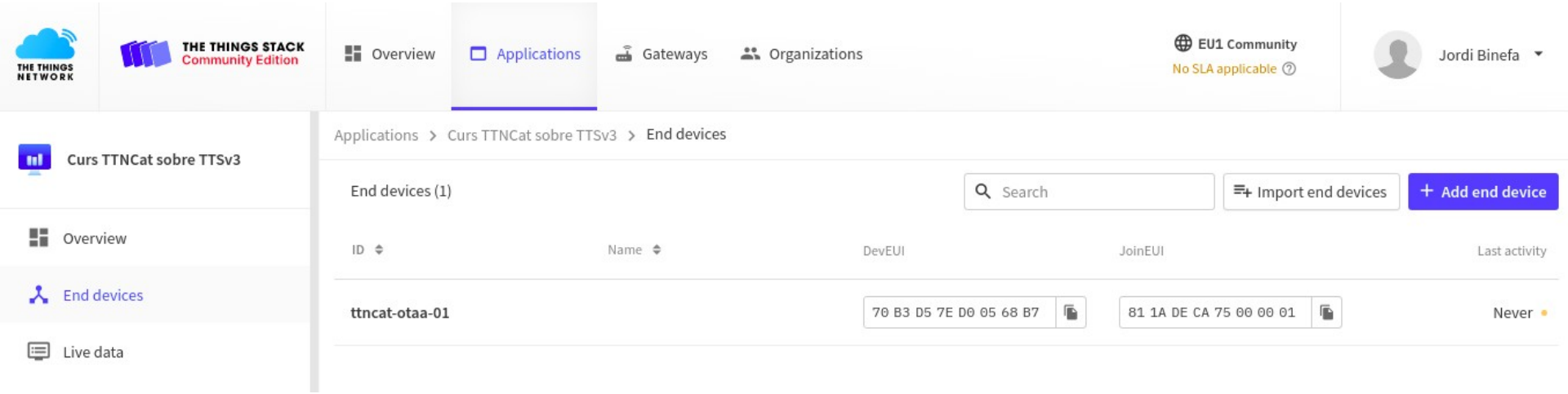
```
SSD1306  display(0x3c, I2C_SDA, I2C_SCL);

float fT,fP,fRH;

// This EUI must be in little-endian format, so least-significant-byte
// first. When copying an EUI from ttnctl output, this means to reverse
// the bytes. For TTN issued EUIs the last bytes should be 0xD5, 0xB3,
// 0x70.
static const u1_t PROGMEM APPEUI[8] = { 0x01, 0x00, 0x00, 0x75, 0xCA, 0xDE, 0x1A, 0x81 };
void os_getArtEui (u1_t* buf) {
  memcpy_P(buf, APPEUI, 8);
}

// This should also be in little endian format, see above.
static const u1_t PROGMEM DEVEUI[8] = { 0xB7, 0x68, 0x05, 0xD0, 0x7E, 0xD5, 0xB3, 0x70 };
void os_getDevEui (u1_t* buf) {
  memcpy_P(buf, DEVEUI, 8);
}

// This key should be in big endian format (or, since it is not really a
// number but a block of memory, endianness does not really apply). In
// practice, a key taken from ttnctl can be copied as-is.
// The key shown here is the semtech default key.
static const u1_t PROGMEM APPKEY[16] = { 0x56, 0x3B, 0xEC, 0xE8, 0xF1, 0x94, 0x65, 0x9D, 0x9F, 0x94,
void os_getDevKey (u1_t* buf) {
  memcpy_P(buf, APPKEY, 16);
}
```



↑ 00:30:23 Forward uplink data message DevAddr: 26 0B 4E A8 Payload: { humitatRelativa: 53.67, pressio: 1024.97, temperatura: 24.95 } 00 00 09 BF 00 01 90 61 … FPort: 1 Data

Alta de dispositius ABP

Alta ABP (1)


THE THINGS NETWORK
THE THINGS STACK
Community Edition
Overview
Applications
Gateways
Organizations
EU1 Community
No SLA applicable
Jordi Binefa
Curs TTNCat sobre TTSv3
Overview
End devices
Live data
Applications > Curs TTNCat sobre TTSv3 > End devices
End devices (1)
Search
Import end devices
Add end device
ID
Name
DevEUI
JoinEUI
Last activity
ttncat-otaa-01
70 B3 D5 7E D0 05 68 B7
81 1A DE CA 75 00 00 01
Never

Alta ABP (2)

Viladecans

Select the end device in the LoRaWAN Device Repository

Enter end device specifics manually

Frequency plan *

Europe 863-870 MHz (SF9 for RX2 - recommended)

LoRaWAN version *

LoRaWAN Specification 1.0.2

Regional Parameters version *

RP001 Regional Parameters 1.0.2

Show advanced activation, LoRaWAN class and cluster settings

Activation mode *

Over the air activation (OTAA)

Activation by personalization (ABP)

Define multicast group (ABP & Multicast)

Additional LoRaWAN class capabilities

None (class A only)

Network defaults

Use network's default MAC settings

Cluster settings

Activation by personalization (ABP)

Define multicast group (ABP & Multicast)

Additional LoRaWAN class capabilities

None (class A only)

Network defaults

Use network's default MAC settings

Cluster settings

Skip registration on Join Server

Provisioning information

JoinEUI *

`81 1A DE CA 75 00 00 01` Reset

This end device can be registered on the network

DevEUI

`70 B3 D5 7E D0 05 68 BA` Generate 2/50 used

Device address *

`26 0B 36 23` Generate

AppSKey *

`CF 51 A9 83 50 E0 95 9D 9A 1F 48 93 58 81 56 5B` Generate

Alta ABP (3)

Viladecans

Provisioning information

JoinEUI *

81 1A DE CA 75 00 00 01 Reset

This end device can be registered on the network

DevEUI

70 B3 D5 7E D0 05 68 BA Generate 2/50 used

Device address *

26 0B 36 23 Generate

AppSKey *

CF 51 A9 83 50 E0 95 9D 9A 1F 48 93 58 81 56 5B Generate

NwkSKey *

55 8B 45 49 12 BA AA 15 BA 21 57 4A 3C E4 B3 74 Generate

End device ID *

ttncat-abp-01

This value is automatically prefilled using the DevEUI

After registration

- (•) View registered end device
- ( ) Register another end device of this type

Alta ABP (4)

Viladecans

Applications > Curs TTNCat sobre TTSv3 > End devices > ttncat-abp-01

ttncat-abp-01
ID: ttncat-abp-01

↑ n/a ↓ n/a • No activity yet

Overview | Live data | Messaging | Location | Payload formatters | General settings

General information

End device ID	ttncat-abp-01
Frequency plan	Europe 863-870 MHz (SF9 for RX2 - recom…
LoRaWAN version	LoRaWAN Specification 1.0.2
Regional Parameters version	RP001 Regional Parameters 1.0.2
Created at	Oct 17, 2022 15:46:59

Activation information

AppEUI	81 1A DE CA 75 00 00 01
DevEUI	70 B3 D5 7E D0 05 68 BA

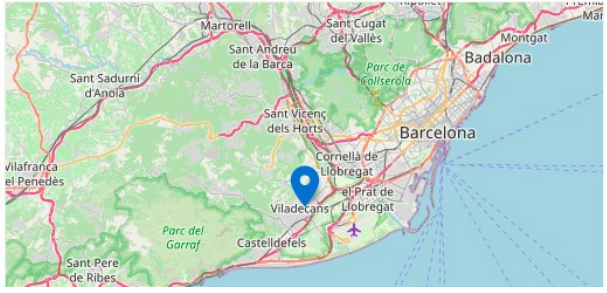
Session information

Session start	Oct 17, 2022 15:46:59
Device address	26 0B 36 23
NwkSKey	•• •• •• •• •• •• •• •• •• •• •• •• •• •• •• ••

Live data See all activity →

15:47:23 Update end device ["locations"]
15:46:59 Create end device DevAddr: 26 0B 36 23

Location Change location settings →



Alta ABP (5)

Viladecans

Applications > Curs TTNCat sobre TTSv3 > End devices > ttncat-abp-01 > General settings

ttncat-abp-01
ID: ttncat-abp-01

↑ n/a ↓ n/a • No activity yet

Overview | Live data | Messaging | Location | Payload formatters | General settings

Basic

Collapse

Description, cluster information and metadata

End device ID *
ttncat-abp-01

AppEUI *
81 1A DE CA 75 00 00 01

DevEUI *
70 B3 D5 7E D0 05 68 BA

End device name
My new end device

End device description

Optional end device description; can also be used to save notes about the end device

organization

Save changes | Delete end device

Network layer

Collapse

LoRaWAN network-layer settings, behavior and session

Frequency plan *
Europe 863-870 MHz (SF9 for RX2 - recommended)

LoRaWAN version *
LoRaWAN Specification 1.0.2

Regional Parameters version *
RP001 Regional Parameters 1.0.2

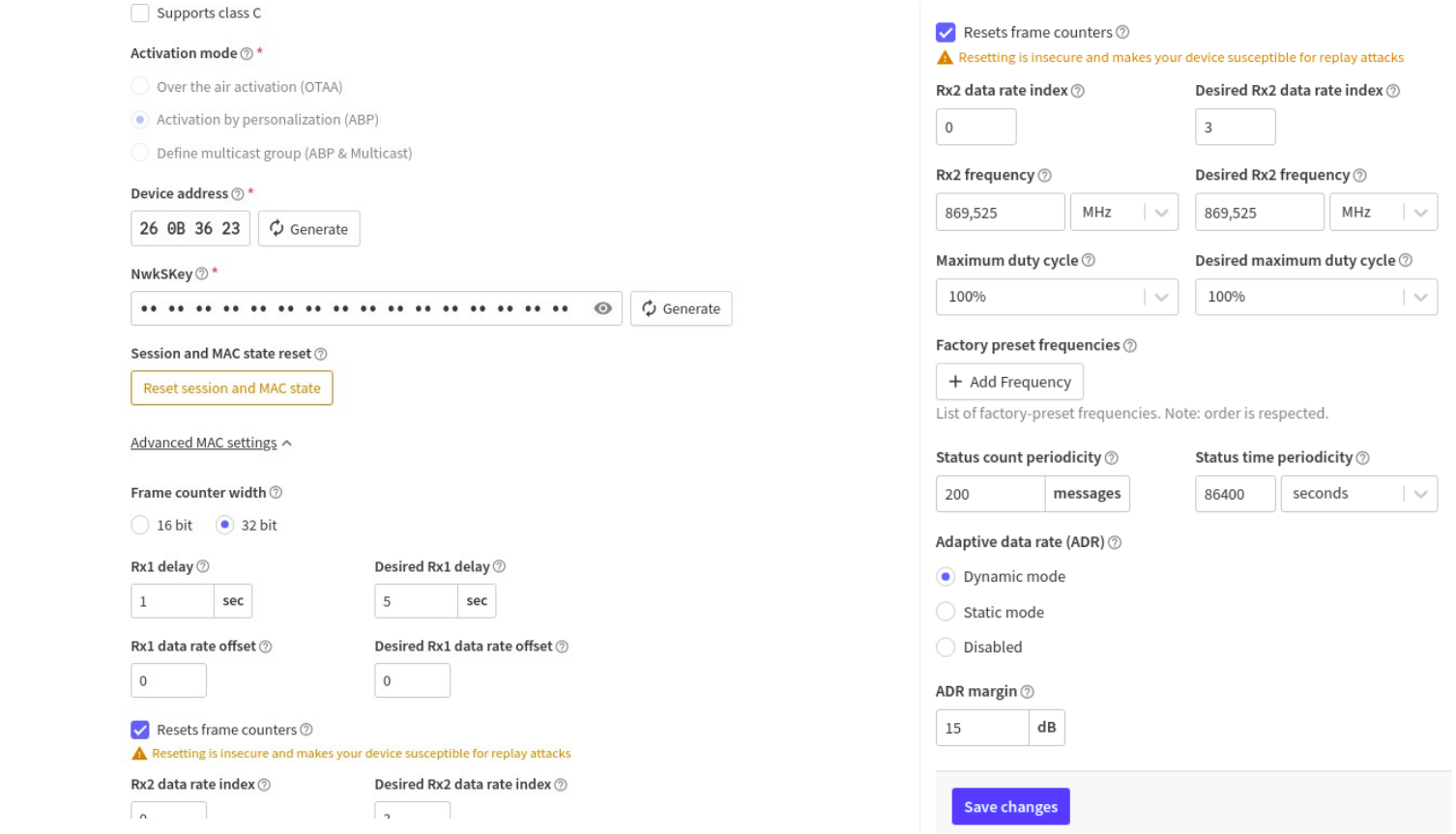
LoRaWAN class capabilities
- [ ] Supports class B
- [ ] Supports class C

Activation mode *
- ( ) Over the air activation (OTAA)
- (•) Activation by personalization (ABP)
- ( ) Define multicast group (ABP & Multicast)

Device address *
26 0B 36 23 | Generate

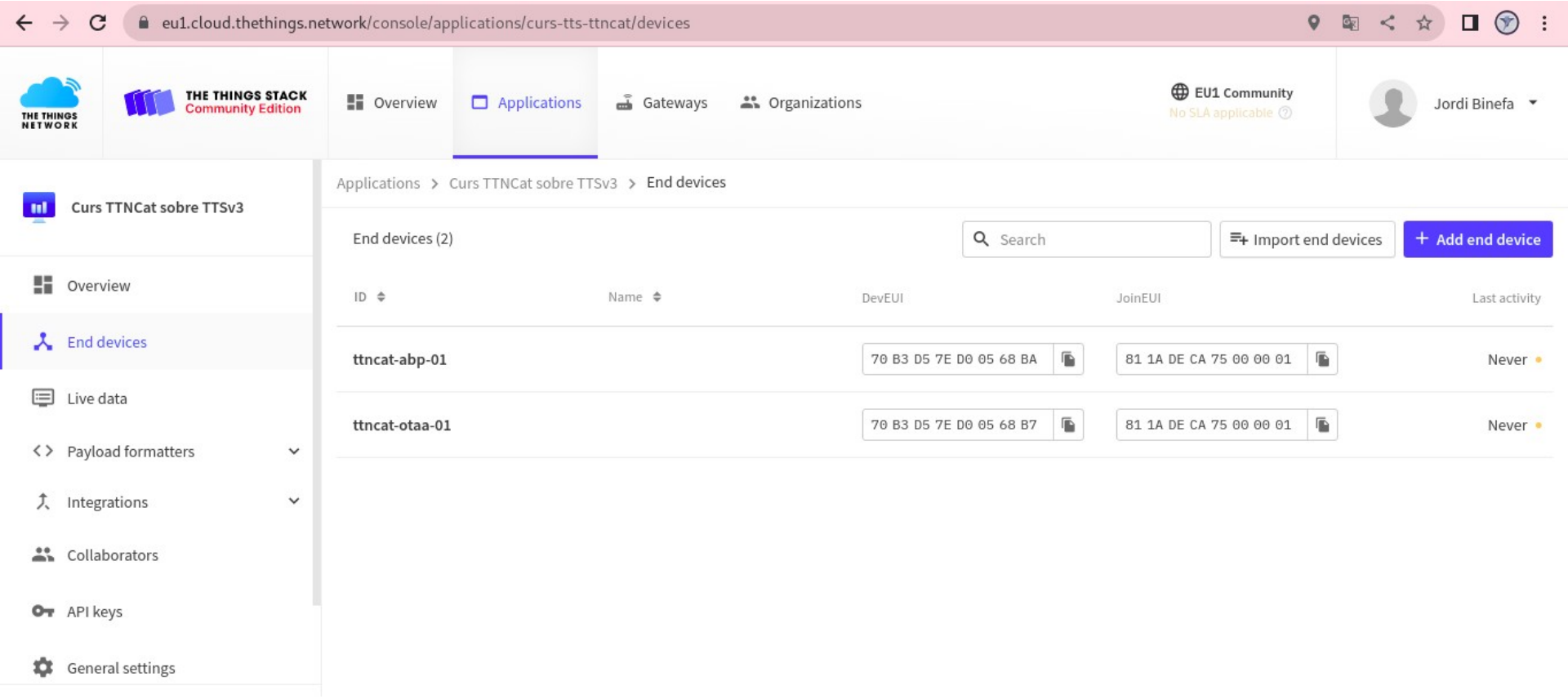
Alta ABP (6)

Viladecans



Alta ABP (7)

Viladecans



Alta ABP (Arduino IDE)

Viladecans

/curs-tts-ttncat/devices/ttncat-abp-01

Last activity 1 minute ago

Overview | Live data | Messaging | Location | Payload formatters | General setti

General information

End device ID	ttncat-abp-01
Frequency plan	Europe 863-870 MHz (SF9 for RX2 - recommen…
LoRaWAN version	LoRaWAN Specification 1.0.2
Regional Parameters version	RP001 Regional Parameters 1.0.2
Created at	Oct 17, 2022 15:46:59

Activation information

AppEUI	81 1A DE CA 75 00 00 01
DevEUI	70 B3 D5 7E D0 05 68 BA

Session information

Session start	Oct 17, 2022 15:46:59
Device address	26 0B 36 23
NwkSKey	0x55, 0x8B, 0x45, 0x49, 0x1… msb
SNwkSIntKey	•• •• •• •• •• •• •• •• •• •• •• •• •• •• •• ••
NwkSEncKey	•• •• •• •• •• •• •• •• •• •• •• •• •• •• •• ••
AppSKey	0xCF, 0x51, 0xA9, 0x83, 0x5… msb

MAC data

Fitxer Edita Esbós Eines Ajuda

IoT-02-25_ttn-abp_3_float_bme280_02

```
23   To use this sketch, first register your application and device wi
24   the things network, to set or generate a DevAddr, NwkSKey and
25   AppSKey. Each device should have their own unique values for thes
26   fields.
27
28   Do not forget to define the radio type correctly in config.h.
29
30 *******************************************************************
31
32 #include <lmic.h>
33 #include <hal/hal.h>
34 #include <SPI.h>
35
36 // LoRaWAN NwkSKey, network session key
37 // This is the default Semtech key, which is used by the early protc
38 // network.
39 static const PROGMEM u1_t NWKSKEY[16] = { 0x55, 0x8B, 0x45, 0x49, 0x
40
41 // LoRaWAN AppSKey, application session key
42 // This is the default Semtech key, which is used by the early protc
43 // network.
44 static const u1_t PROGMEM APPSKEY[16] = { 0xCF, 0x51, 0xA9, 0x83, 0x
45
46 // LoRaWAN end-device address (DevAddr)
47 static const u4_t DEVADDR = 0x260B3623 ; // <-- Change this address
48
49 // These callbacks are only used in over-the-air activation, so they
50 // left empty here (we cannot leave them out completely unless
51 // DISABLE_JOIN is set in config.h, otherwise the linker will compla
52 void os_getArtEui (u1_t* buf) { }
53 void os_getDevEui (u1_t* buf) { }
54 void os_getDevKey (u1_t* buf) { }
55
56 static uint8_t mydata[] = "Bon dia, Viladecans!";
57 static osjob_t sendjob;
```

Pujada enllestida.

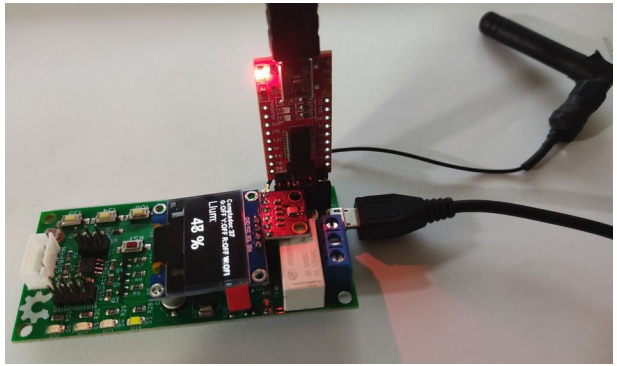
Seqüència per passar a mode programació a la placa IoT-02:

* Premeu el botó vermell (***RST***)

* Premeu el botó blanc ***IO0***

* Deixeu de prémer el botó vermell (***RST***)

* Deixeu de prémer el botó blanc ***IO0***



Formatador de dades rebudes 1

Viladecans

LoRa OTAA
ID: eui-70b3d57ed004bdd3

↑1 ↓n/a • Last activity 10 minutes ago

Overview | Live data | Messaging | Location | Payload formatters | Claiming | General settings

Verbose stream | Export as JSON | Pause | Clear

Time	Type	Data preview
12:47:40	Forward uplink data message	DevAddr: 26 0B 94 71 Payload: { humitatRelativa: 53.23, pressio: 1027.78, temperatura: 22.75 } 00 00 08 E3 00 01 91 7A … FPort: 1 Data r
12:47:40	Successfully processed data messa…	DevAddr: 26 0B 94 71
12:47:37	Forward join-accept message	DevAddr: 26 0B 94 71
12:47:35	Accept join-request	DevAddr: 26 0B 94 71

/dev/ttyUSB0

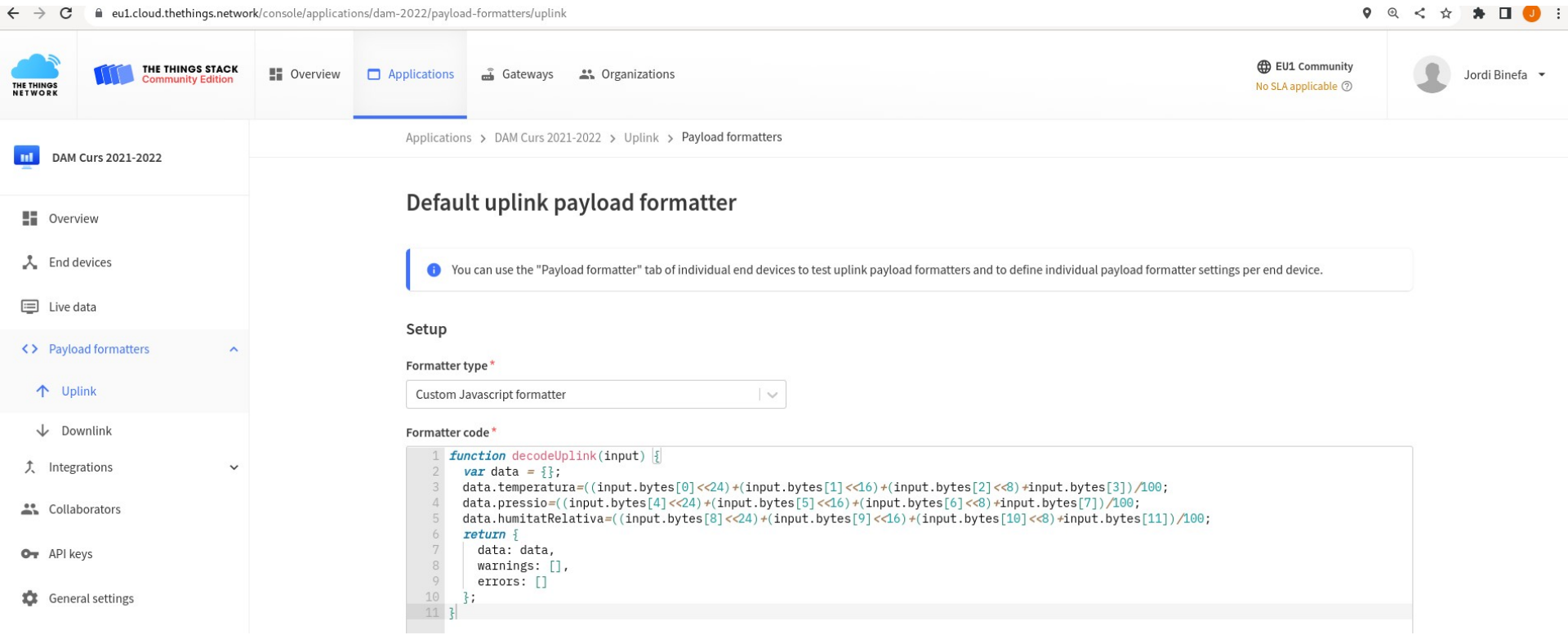
Envia

```
/home/jordi/Documents/electronics.cat/formacio/TTN_20221020/IoT-02-22_ttn-otaa_dam_07_bme280/IoT-02-22_ttn-otaa_dam_07_bme280.ino
Starting
Boot number: 1
Wakeup was not caused by deep sleep
Setup ESP32 to sleep for every 600 Seconds
Vector de bytes: 0 0 8 E3
Vector de bytes: 0 1 91 7A
Vector de bytes: 0 0 14 CB
Packet queued
T:  22.75ºC
P:  1027.78 hPa
HR: 53.23 %
13795: EV_JOINING
146947: Unknown event
467998: EV_JOINED
468025: Unknown event
848662: EV_TXCOMPLETE (includes waiting for RX windows)
Save LMIC to RTC ...
Going to sleep now
```

Desplaçament automàtic | Mostra la marca de temps | Retorn de carro | 115200 baud | Neteja la sortida

Formatador de dades rebudes 2

Viladecans



Formatador de dades rebudes 3

Viladecans

Formatter type *

Custom Javascript formatter

Formatter code *

```
function decodeUplink(input) {
  var data = {};
  data.temperatura=((input.bytes[0]<<24)+(input.bytes[1]<<16)+(input.bytes[2]<<8)+input.bytes[3])/100;
  data.pressio=((input.bytes[4]<<24)+(input.bytes[5]<<16)+(input.bytes[6]<<8)+input.bytes[7])/100;
  data.humitatRelativa=((input.bytes[8]<<24)+(input.bytes[9]<<16)+(input.bytes[10]<<8)+input.bytes[11])/100;
  return {
    data: data,
    warnings: [],
    errors: []
  };
}
```

Formatador de dades rebudes 4

Viladecans

arjanvanb Arjan 3 arjanvanb Jan '17

When sending 3 bytes, we're basically not sending the 4th byte of each integer, which should be `0xFF` for negative numbers. Like for New York (40.712784, -74.005941, which would be sent as integers 407127 and -740059) we would send:

- For LSB: `0x573606 25B5F4` instead of `0x57360600 25B5F4FF` (Least Significant Bit/Byte first 16, to match @JohanAdriaans' encoding example above).
- For MSB: `0x063657 F4B525` instead of `0x00063657 FFF4B525`.

When decoding these 6 bytes back into two 32 bits signed integers, we need to compute the missing 4th and 8th bytes ourselves, to make JavaScript properly convert the negative values for us. Those bytes should be `0xFF` if the most significant bytes have their "high bit" set, which is called "sign extending 13":

```
// LSB, Least Significant Bit/Byte first
// Sign-extend the 3rd and 6th bytes into a 4th and 8th byte:
lat = (b[0] | b[1]<<8 | b[2]<<16 | (b[2] & 0x80 ? 0xFF<<24 : 0)) / 10000;
lng = (b[3] | b[4]<<8 | b[5]<<16 | (b[5] & 0x80 ? 0xFF<<24 : 0)) / 10000;
```

```
// MSB, Most Significant Bit/Byte first
// Sign-extend the 1st and 4th bytes into leading bytes:
lat = ((b[0] & 0x80 ? 0xFF<<24 : 0) | b[0]<<16 | b[1]<<8 | b[2]) / 10000;
lng = ((b[3] & 0x80 ? 0xFF<<24 : 0) | b[3]<<16 | b[4]<<8 | b[5]) / 10000;
```

Alternatively, shift the most significant byte 8 bits too far to the left, and then shift it back, which will do the sign extension on the fly, as the bitwise operator `>>` is the sign-propagating right shift 9:

```
lat = (b[0]<<24>>8 | b[1]<<8 | b[2]) / 10000;
lng = (b[3]<<24>>8 | b[4]<<8 | b[5]) / 10000;
```

Meanwhile, for the new production environment, payload functions should also include the function name, `Decoder`. So, to support 6 byte coordinates with possible negative values:

```
function Decoder(b, port) {

  // Amsterdam: 52.3731, 4.8924 = MSB 07FDD3 00BF1C, LSB D3FD07 1CBF00
  // La Paz: -16.4896, -68.1192 = MSB FD7BE0 F59B18, LSB E07BFD 189BF5
  // New York: 40.7127, -74.0059 = MSB 063657 F4B525, LSB 573606 25B5F4
  // Sidney: -33.8688, 151.2092 = MSB FAD500 17129C, LSB 00D5FA 9C1217

  // LSB, Least Significant Bit/Byte first! Your node likely sends MSB inste

  // Sign-extend the 3rd and 6th bytes into a 4th and 8th byte:
  var lat = (b[0] | b[1]<<8 | b[2]<<16 | (b[2] & 0x80 ? 0xFF<<24 : 0)) / 100
  var lng = (b[3] | b[4]<<8 | b[5]<<16 | (b[5] & 0x80 ? 0xFF<<24 : 0)) / 100

  return {
    location: {
      lat: lat,
      lng: lng
    },
    love: "TTN payload functions"
  };
}
```

Alternatively, as coordinates in decimal degrees are -90…+90 for latitude, and -180…+180 for longitude: one could add 90 to the latitude and 180 to the longitude before sending, then send the positive values, and reverse that in the payload function.

And life can be made easy using libraries such as https://github.com/thesolarnomad/lora-serialization 161 (though that one does not support 3 byte coordinates).

Alta ABP (Thonny)

Viladecans

:/dam-2022/devices/eui-70b3d57ed004c502

Overview | Live data | Messaging | Location | Payload formatters | Claiming | General settings

General information

End device ID	eui-70b3d57ed004c502
Description	Activation By Personalization
Frequency plan	Europe 863-870 MHz (SF9 for RX2 - recommen…
LoRaWAN version	LoRaWAN Specification 1.0.2
Regional Parameters version	RP001 Regional Parameters 1.0.2
Created at	Feb 7, 2022 10:53:45

Activation information

AppEUI	n/a
DevEUI	70 B3 D5 7E D0 04 C5 02

Session information

Session start	Sep 12, 2022 00:08:16
Device address	26 0B 71 90
NwkSKey	0x27, 0x92, 0xAA, 0x1E, 0x8… msb
SNwkSIntKey	•• •• •• •• •• •• •• •• •• •• •• •• •• •• •• ••
NwkSEncKey	•• •• •• •• •• •• •• •• •• •• •• •• •• •• •• ••
AppSKey	0xE7, 0x76, 0xF4, 0x8E, 0xB… msb

Thonny - MicroPython device :: /lora.py @

File Edit View Run Tools Help

lora.py | [lora.py]

```
35 LORA_RST = const(9)
36 '''
37 #Set the pinout for the IoT-02
38 LORA_SCK = const(5)
39 LORA_MOSI = const(25)
40 LORA_MISO = const(15)
41 LORA_CS = const(18)
42 LORA_IRQ = const(26) # <---- IRQ -> DIO0
43 LORA_RST = const(14)
44
45 #LoRa configuration
46 LORA_DATARATE = "SF9BW125"  # Choose from several available
47
48 #Enter the Data for the TTN access here
49 DEVADDR = bytearray([0x26, 0x0B, 0x71, 0x90])
50 # DEVADDR = bytearray([INSERT DATA HERE])
51 # static const u4_t DEVADDR = 0x260B7190 ;
52 NWKEY = bytearray([0x27, 0x92, 0xAA, 0x1E, 0x8A, 0xC5, 0x8D, 0x7F, 0x47, 0
53 # NWKEY = bytearray([INSERT DATA HERE])
54 # static const PROGMEM u1_t NWKSKEY[16] = { 0x27, 0x92, 0xAA, 0x1E, 0x8A,
55 APP = bytearray([0xE7, 0x76, 0xF4, 0x8E, 0xBB, 0xA1, 0xE3, 0xD7, 0xBF, 0xB
56 # APP = bytearray([INSERT DATA HERE])
57 # static const u1_t PROGMEM APPSKEY[16] = { 0xE7, 0x76, 0xF4, 0x8E, 0xBB,
58 #Configure your contry accodtingly for Europe you may use ="EU"
59 TTN_CONFIG = TTN(DEVADDR, NWKEY, APP, country="EU")
60 FPORT = 1
61
```

Shell

Instal·lant MicroPython a la placa IoT-02

Viladecans

Cal tenir instal·lat el Python al vostre sistema

```
pip install esptool
```

```
esptool.py --port /dev/ttyUSB0 erase_flash
```

Poseu la placa IoT-02 en mode de programació

*Premeu el botó vermell (***RST***)
* Premeu el botó blanc ***IO0***
* Deixeu de prémer el botó vermell (***RST***)
* Deixeu de prémer el botó blanc ***IO0***

```
esptool.py --chip esp32 --port /dev/ttyUSB0 write_flash -z 0x1000 esp32-ota-20220618-v1.19.1.bin
```

Explicació a la pàgina oficial de MicroPython

Gestió de fitxers a un dispositiu amb MicroPython

Viladecans

```
pip install adafruit-ampy

ampy --help

ampy --port /serial/port run test.py

ampy --port /serial/port put test.py

ampy --port /serial/port put /directori/altre_test.py

ampy --port /serial/port get main.py

ampy --port /serial/port get boot.py placa_boot.py

ampy --port /serial/port mkdir nom_directori

ampy --port /serial/port mkdir /nom_directori/subdirectori

ampy --port /serial/port ls

ampy --port /serial/port rm fitxer.py

ampy --port /serial/port rmdir /ruta/absoluta
```

MicroPython

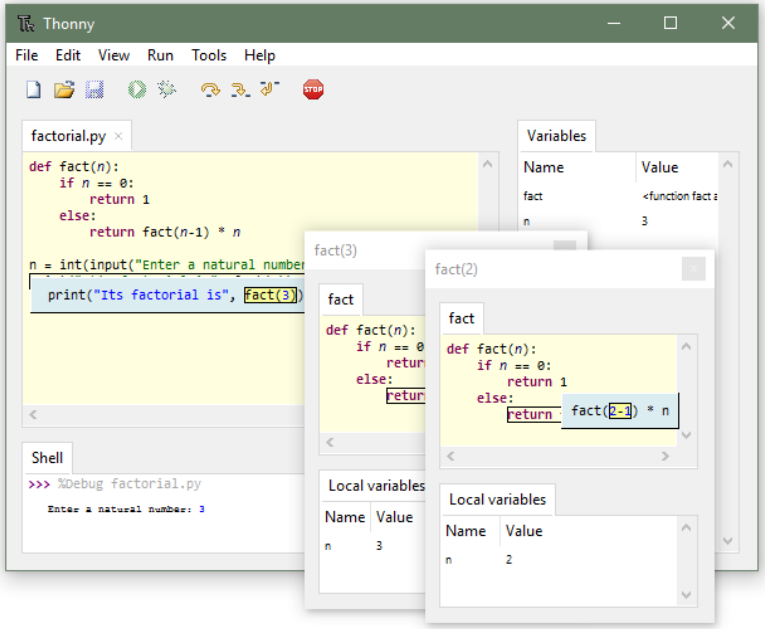
MicroPython Basics: Load Files & Run Code
feta per Adafruit Industries

Gestió gràfica de fitxers a un dispositiu amb MicroPython

Viladecans

Thonny
Python IDE for beginners

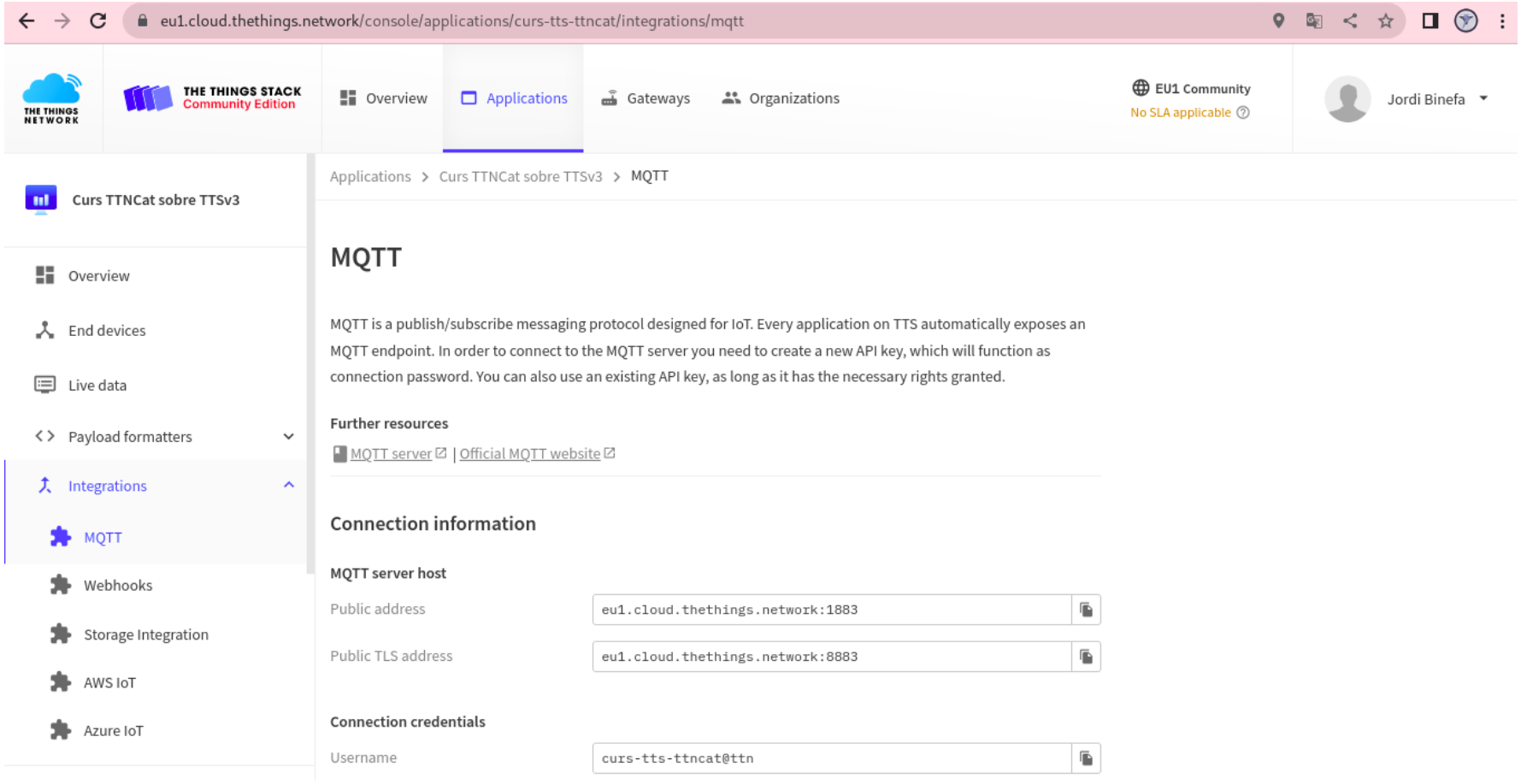
Download version 4.0.1 for Windows • Mac • Linux



Connectivitat MQTT

Connectivitat MQTT (1)

Viladecans



Connectivitat MQTT (2)

Viladecans

Applications > Curs TTNCat sobre TTSv3 > MQTT

MQTT

MQTT is a publish/subscribe messaging protocol designed for IoT. Every application on TTS automatically exposes an MQTT endpoint. In order to connect to the MQTT server you need to create a new API key, which will function as connection password. You can also use an existing API key, as long as it has the necessary rights granted.

Further resources

MQTT server | Official MQTT website

Connection information

MQTT server host

Public address	`eu1.cloud.thethings.network:1883`
Public TLS address	`eu1.cloud.thethings.network:8883`

Connection credentials

Username	`curs-tts-ttncat@ttn`
Password	Generate new API key Go to API keys