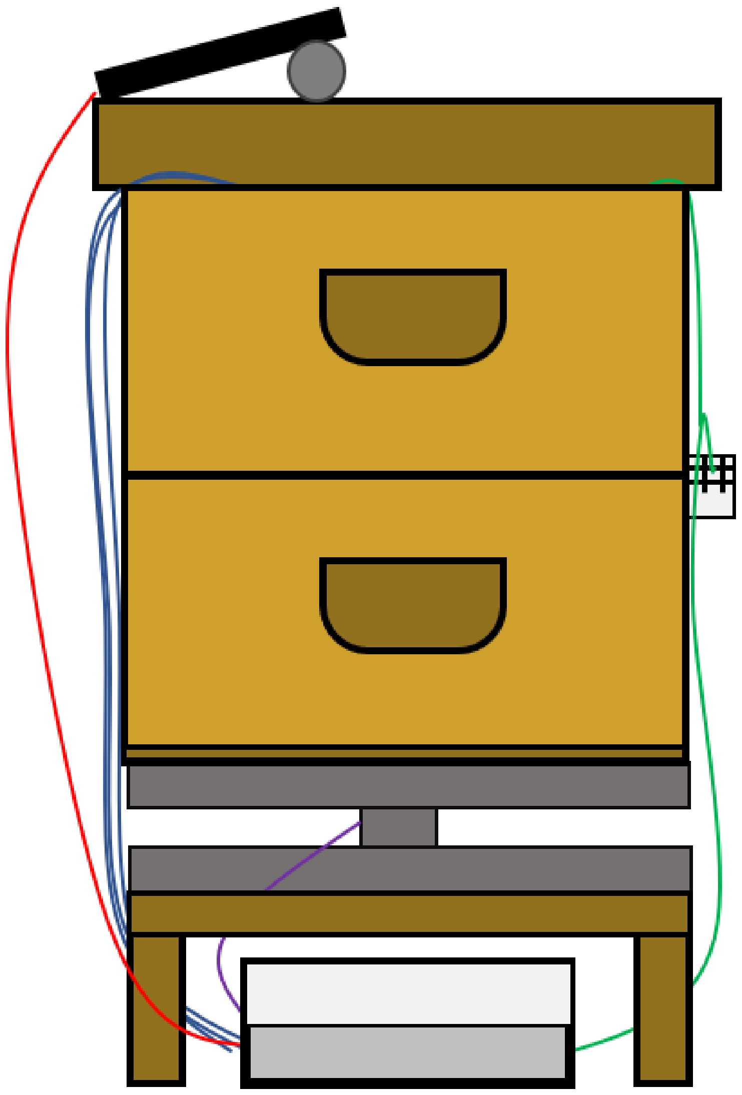


User guide

CONNECTED BEEHIVE

DELAS Léo | DOS SANTOS Damien | DURAND DE GEVIGNEY Basile | TAN Caroline
EI2I 4 II – 19/01/2022

We are very proud to present you our user manual to implement our project of connected beehive.

You will be able to find all documents, sources codes and other useful files of the project on the following GitHub:

https://github.com/Ithoh/OpenRucheDLCB

THANKS, AND ENJOY !!!!

Summary

I – Required components

For this project, you need to have these components:

Solar cell of 5,5 V/360 mA and dimensions of 180 x 80 mm	
Power adapter card (LiPo Rider Pro)	
Arduino MKR FOX 1200 board ABX00014 with SigFox interface	
Li-Ion battery 3,7V 1050 mAh	
Multiple temperature sensors	
Weight sensor: strain gauge and HX711	
Temperature and humidity sensor DHT22	

II – PCB and connecting everything

For our project we have decided to design a PCB to connect all our components, for this we used the software EasyEDA, which is a free software that allows you to do some PCB design and also some electrical schematics and others functions that we haven't used in this project.

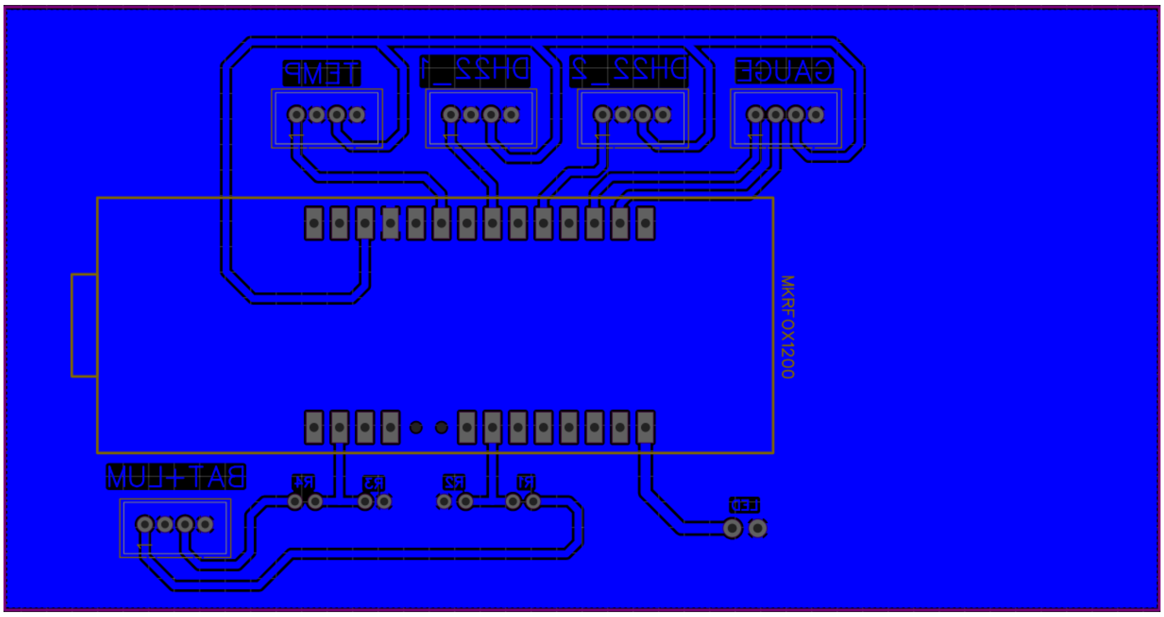
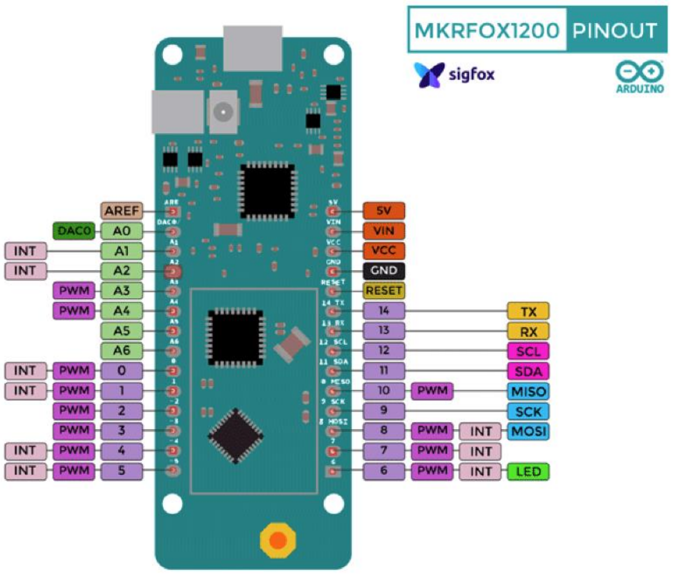
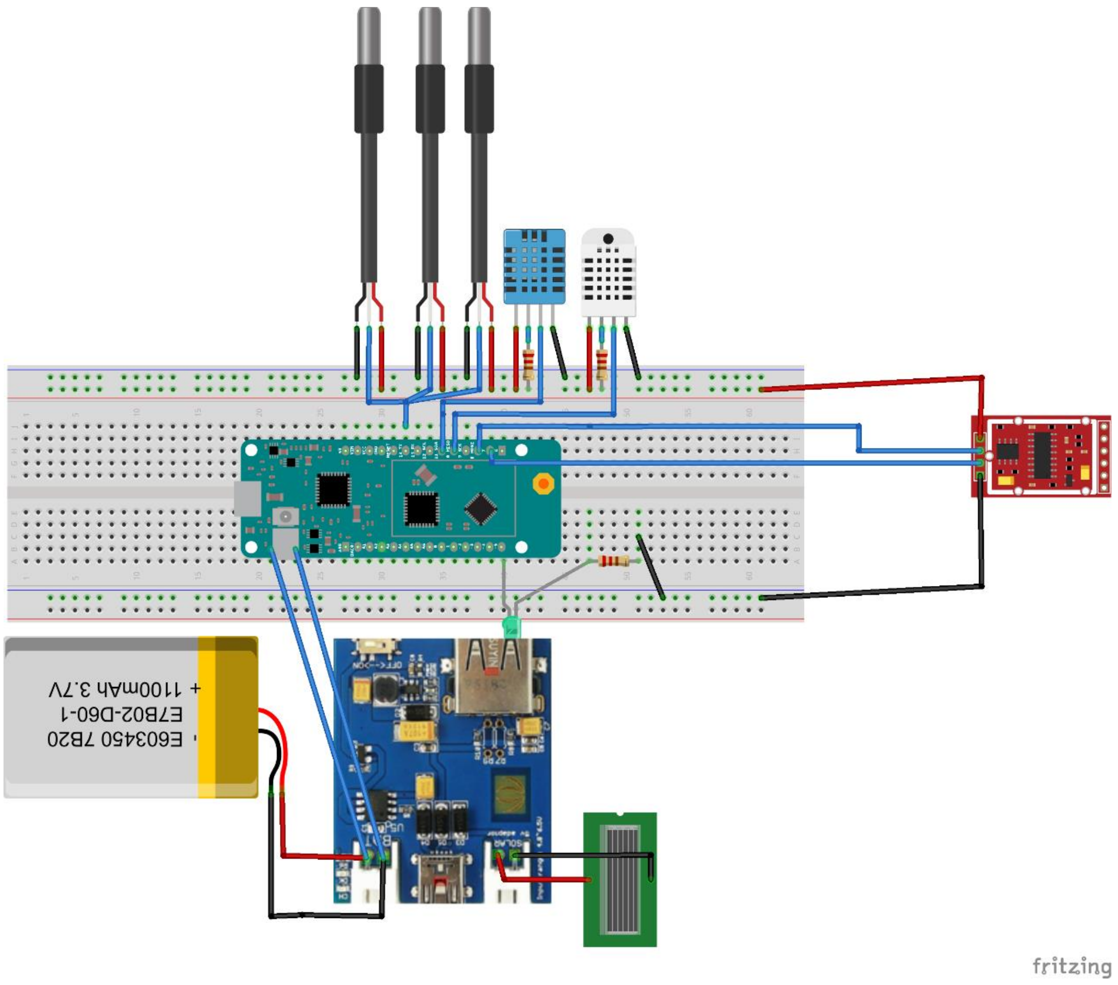


Figure 1 - PCB of the project

Everything is written on the PCB on how to link everything. It is purposely reversed so the print is not reversed and can be easily read. The way we put our Arduino board here, is by using some female pin header to easily swap out the board if needed while maintaining some robustness. The pinout of the maker fox is as follow:



Here is a schematic on how our project was implemented on a breadboard for tests:



You can easily find our PCB files here:
https://github.com/Ithoh/OpenRucheDLCB/tree/main/Ressources/PCB_Schematic

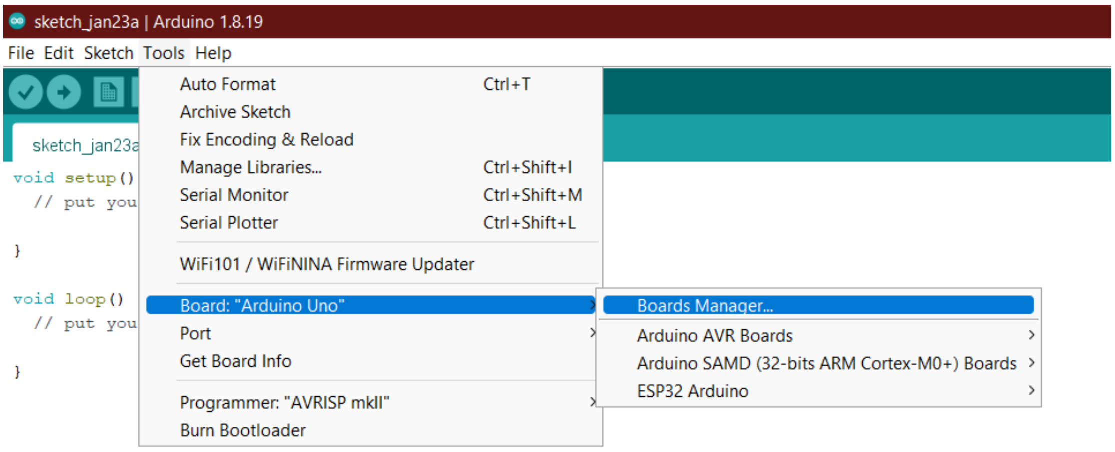
III – Code

In this project we will use Arduino IDE to program the MKRFOX board.

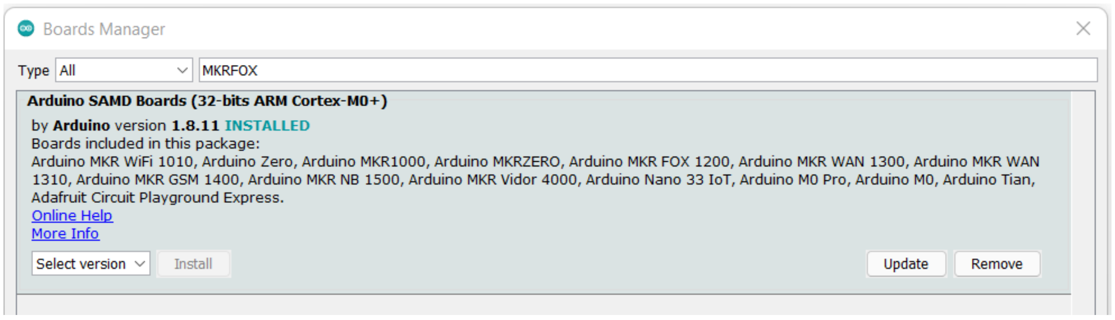
You need to install the Arduino IDE and download the codes from our GitHub: https://github.com/Ithoh/OpenRucheDLCB/tree/main/Codes

Before uploading our code, you need to prepare your IDE installing all needed dependencies.

First of all you need to access to the boards manager via: Tools->Board: "..." ->Boards Manager.


sketch_jan23a | Arduino 1.8.19
File Edit Sketch Tools Help
sketch_jan23a
void setup()
// put you
}
void loop()
// put you
}
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Uno"
Port
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Boards Manager...
Arduino AVR Boards
Arduino SAMD (32-bits ARM Cortex-M0+) Boards
ESP32 Arduino

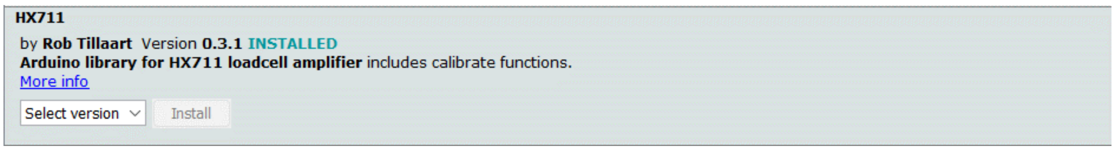
And searching "MKRFOX", you will be able to install the following add-on:

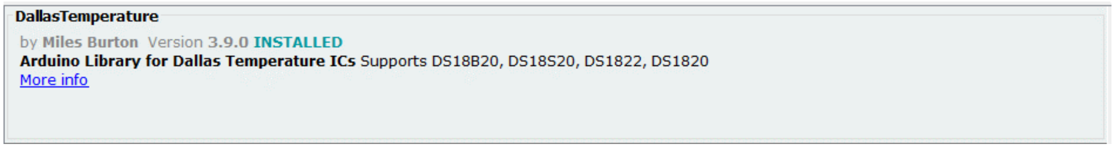


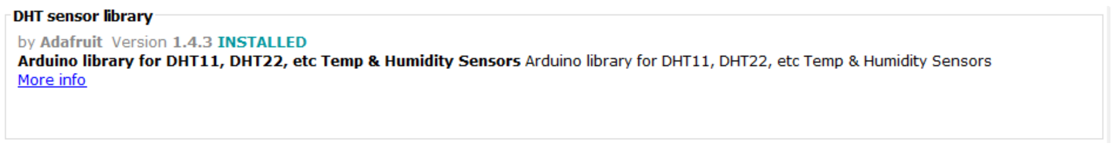
Arduino SAMD Boards (32-bits ARM Cortex-M0+)
by **Arduino** version **1.8.11** **INSTALLED**
Boards included in this package:
Arduino MKR WiFi 1010, Arduino Zero, Arduino MKR1000, Arduino MKRZERO, Arduino MKR FOX 1200, Arduino MKR WAN 1300, Arduino MKR WAN 1310, Arduino MKR GSM 1400, Arduino MKR NB 1500, Arduino MKR Vidor 4000, Arduino Nano 33 IoT, Arduino M0 Pro, Arduino M0, Arduino Tian, Adafruit Circuit Playground Express.
Online Help
More Info

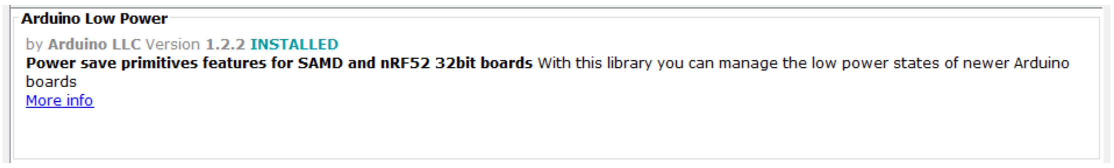
Be sure to install "1.8.11" Version, we encountered some troubles using later version.

After that you need to install all following libraries used for sensors:


HX711
by **Rob Tillaart** Version **0.3.1** **INSTALLED**
Arduino library for HX711 loadcell amplifier includes calibrate functions.
More info
Select version | Install

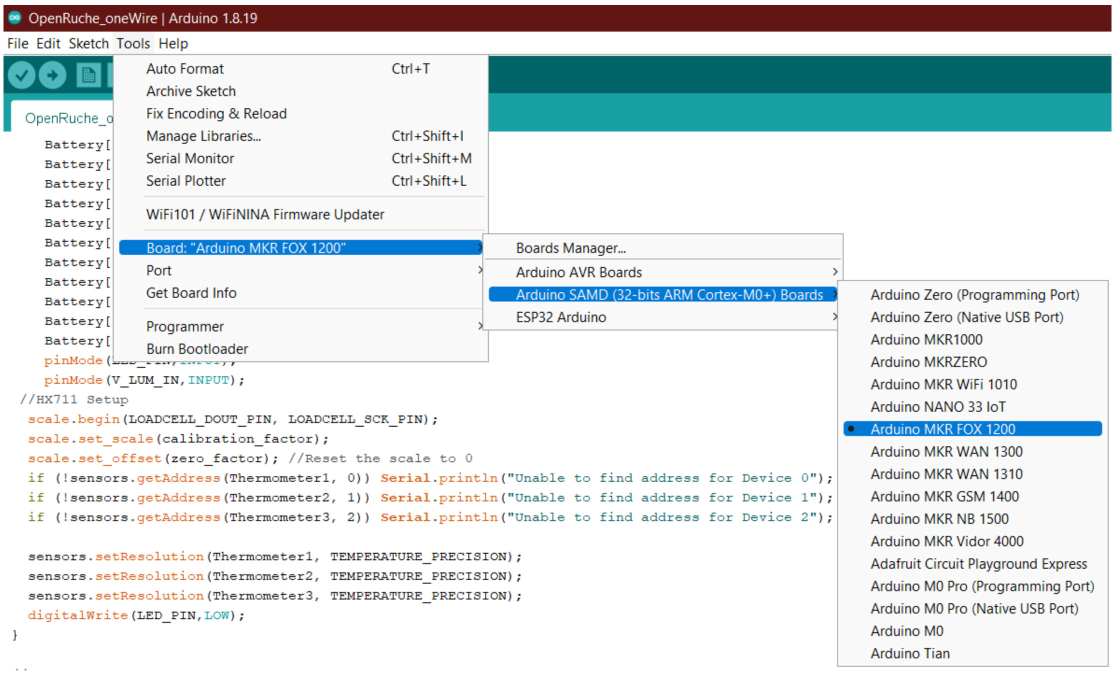

DallasTemperature
by **Miles Burton** Version **3.9.0** **INSTALLED**
Arduino Library for Dallas Temperature ICs Supports DS18B20, DS18S20, DS1822, DS1820
More info


DHT sensor library
by **Adafruit** Version **1.4.3** **INSTALLED**
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors
More info


Arduino Low Power
by **Arduino LLC** Version **1.2.2** **INSTALLED**
Power save primitives features for SAMD and nRF52 32bit boards With this library you can manage the low power states of newer Arduino boards
More info

If the software asks to install dependencies, do it, it is needed to the project.

Now, you will be able to compile the project without issues, to do it choose the right board in Tools->Board button as showed in the picture bellow:


OpenRuche_oneWire | Arduino 1.8.19

File Edit Sketch Tools Help

Tools menu:
- Auto Format — Ctrl+T
- Archive Sketch
- Fix Encoding & Reload
- Manage Libraries... — Ctrl+Shift+I
- Serial Monitor — Ctrl+Shift+M
- Serial Plotter — Ctrl+Shift+L
- WiFi101 / WiFiNINA Firmware Updater
- Board: "Arduino MKR FOX 1200"
- Port
- Get Board Info
- Programmer
- Burn Bootloader

Board submenu:
- Boards Manager...
- Arduino AVR Boards
- Arduino SAMD (32-bits ARM Cortex-M0+) Boards
- ESP32 Arduino

Arduino SAMD boards:
- Arduino Zero (Programming Port)
- Arduino Zero (Native USB Port)
- Arduino MKR1000
- Arduino MKRZERO
- Arduino MKR WiFi 1010
- Arduino NANO 33 IoT
- ● Arduino MKR FOX 1200
- Arduino MKR WAN 1300
- Arduino MKR WAN 1310
- Arduino MKR GSM 1400
- Arduino MKR NB 1500
- Arduino MKR Vidor 4000
- Adafruit Circuit Playground Express
- Arduino M0 Pro (Programming Port)
- Arduino M0 Pro (Native USB Port)
- Arduino M0
- Arduino Tian

```
    Battery[
    Battery[
    Battery[
    Battery[
    Battery[
    Battery[
    Battery[
    Battery[
    Battery[
    Battery[
    Battery[
    pinMode(
    pinMode(V_LUM_IN,INPUT);
 //HX711 Setup
  scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
  scale.set_scale(calibration_factor);
  scale.set_offset(zero_factor); //Reset the scale to 0
  if (!sensors.getAddress(Thermometer1, 0)) Serial.println("Unable to find address for Device 0");
  if (!sensors.getAddress(Thermometer2, 1)) Serial.println("Unable to find address for Device 1");
  if (!sensors.getAddress(Thermometer3, 2)) Serial.println("Unable to find address for Device 2");

  sensors.setResolution(Thermometer1, TEMPERATURE_PRECISION);
  sensors.setResolution(Thermometer2, TEMPERATURE_PRECISION);
  sensors.setResolution(Thermometer3, TEMPERATURE_PRECISION);
  digitalWrite(LED_PIN,LOW);
}
```

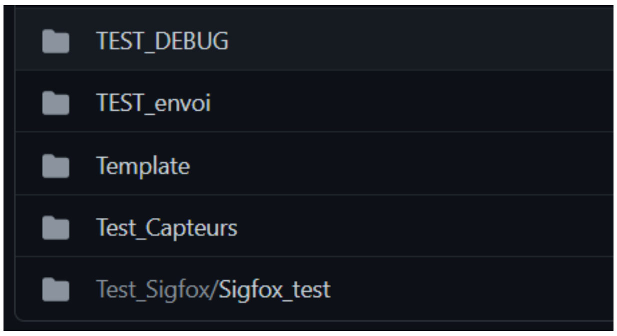
And then select the port (the name of the board should be shown beside "COMx"

Our code:

Our codes are divided between the main code that are needed for the project:

https://github.com/Ithoh/OpenRucheDLCB/tree/main/Codes/Programmes_principaux/OpenRuche_oneWire

And the test codes to check whether the sensors are working or not :


TEST_DEBUG
TEST_envoi
Template
Test_Capteurs
Test_Sigfox/Sigfox_test

At first, you should test the sensors through compilation and then upload the code on the Arduino. Do not forget to select the right Arduino and the right port.

After checking that everything is working correctly, you can compile and upload the main code on the Arduino.

You can find our project on our github, the only things you could have to do it to change port of your sensors as you plug yours on the board.

https://github.com/Ithoh/OpenRucheDLCB/tree/main/Codes

IV – Sigfox connection

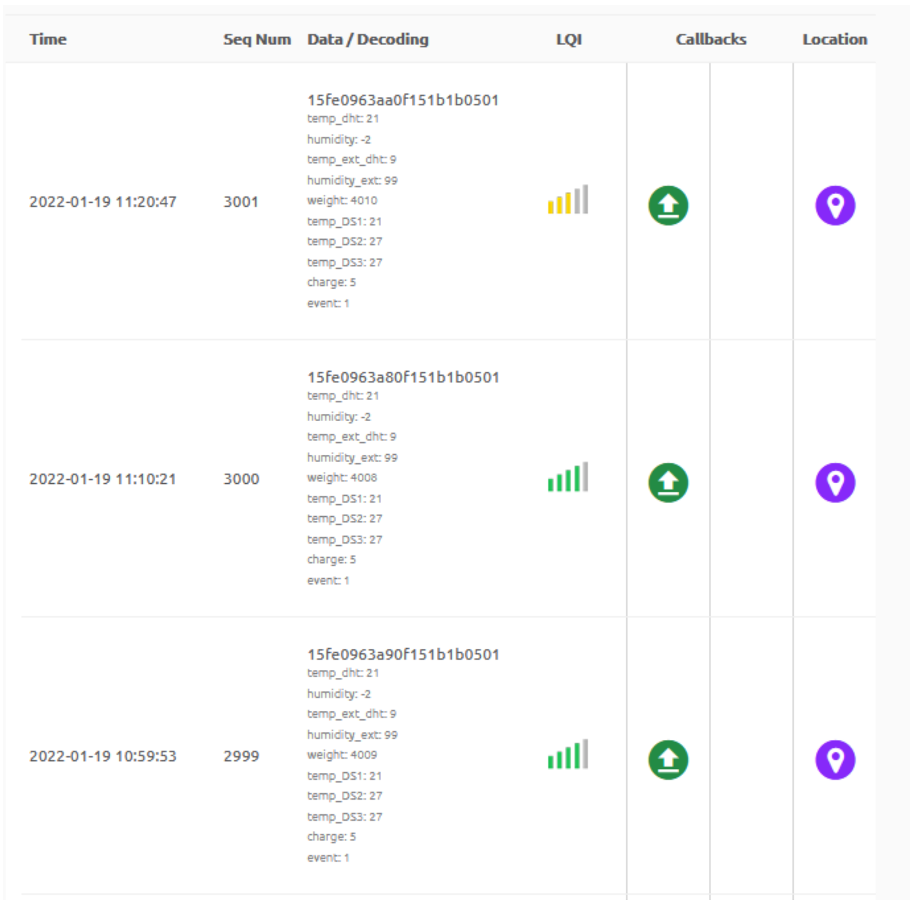
To access the data that is recovered, you need to log in your account Backend Sigfox with your login. We recommend using a Yopmail account if the use of the beehive is for a school project, otherwise your regular mail is totally fine.

If it is for school purposes, the name of the mail should be the one on the Arduino and the password should be something easy to remember like below :

Mail	001d80XX@yopmail.com
Password	Sigfox21#

On the website, you can check the messages that are sent every ten minutes by clicking on: Device, then Messages.

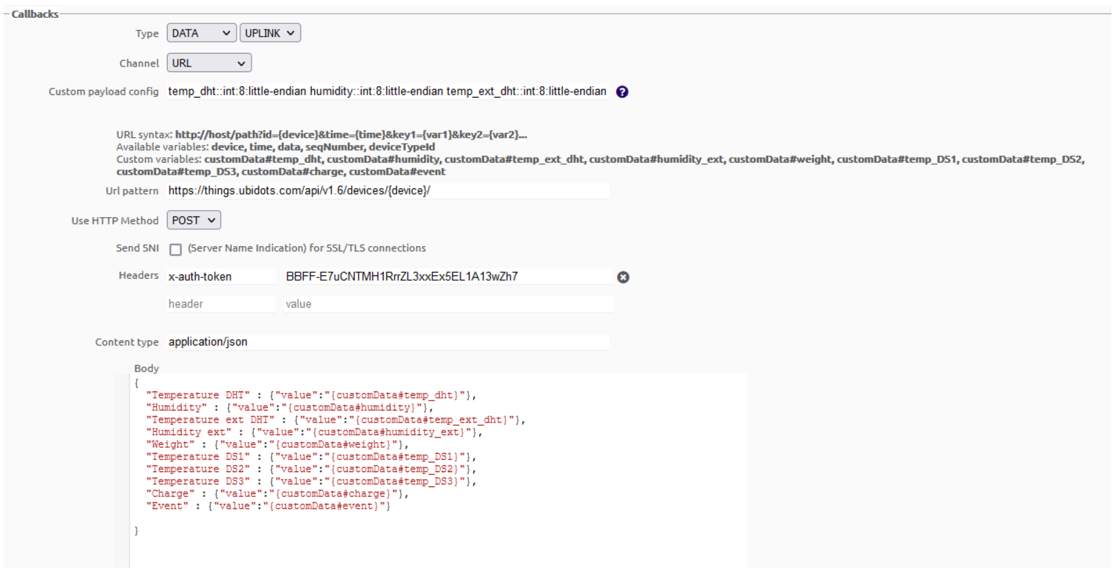
You should be able to see this after following these steps:



Time	Seq Num	Data / Decoding	LQI	Callbacks		Location
2022-01-19 11:20:47	3001	15fe0963aa0f151b1b0501 temp_dht: 21 humidity: -2 temp_ext_dht: 9 humidity_ext: 99 weight: 4010 temp_DS1: 21 temp_DS2: 27 temp_DS3: 27 charge: 5 event: 1	▂▄▆	⬆		📍
2022-01-19 11:10:21	3000	15fe0963a80f151b1b0501 temp_dht: 21 humidity: -2 temp_ext_dht: 9 humidity_ext: 99 weight: 4008 temp_DS1: 21 temp_DS2: 27 temp_DS3: 27 charge: 5 event: 1	▂▄▆█	⬆		📍
2022-01-19 10:59:53	2999	15fe0963a90f151b1b0501 temp_dht: 21 humidity: -2 temp_ext_dht: 9 humidity_ext: 99 weight: 4009 temp_DS1: 21 temp_DS2: 27 temp_DS3: 27 charge: 5 event: 1	▂▄▆█	⬆		📍

Figure 2 - Sigfox messages

If you want to know more about how Sigfox can read these messages, it is thanks to the parameters of the callback which you can access via Device Type, Callbacks and editing the only one that is there:



Callbacks

Type: DATA, UPLINK

Channel: URL

Custom payload config: temp_dht::int:8:little-endian humidity::int:8:little-endian temp_ext_dht::int:8:little-endian

URL syntax: **http://host/path?id={device}&time={time}&key1={var1}&key2={var2}...**
Available variables: **device, time, data, seqNumber, deviceTypeId**
Custom variables: **customData#temp_dht, customData#humidity, customData#temp_ext_dht, customData#humidity_ext, customData#weight, customData#temp_DS1, customData#temp_DS2, customData#temp_DS3, customData#charge, customData#event**

Url pattern: https://things.ubidots.com/api/v1.6/devices/{device}/

Use HTTP Method: POST

Send SNI: ☐ (Server Name Indication) for SSL/TLS connections

Headers: x-auth-token BBFF-E7uCNTMH1RrrZL3xxEx5EL1A13wZh7

header value

Content type: application/json

Body

```
{
  "Temperature DHT" : {"value":"{customData#temp_dht}"},
  "Humidity" : {"value":"{customData#humidity}"},
  "Temperature ext DHT" : {"value":"{customData#temp_ext_dht}"},
  "Humidity ext" : {"value":"{customData#humidity_ext}"},
  "Weight" : {"value":"{customData#weight}"},
  "Temperature DS1" : {"value":"{customData#temp_DS1}"},
  "Temperature DS2" : {"value":"{customData#temp_DS2}"},
  "Temperature DS3" : {"value":"{customData#temp_DS3}"},
  "Charge" : {"value":"{customData#charge}"},
  "Event" : {"value":"{customData#event}"}

}
```

Figure 3 - Screenshot of the callbacks

V – Ubidots

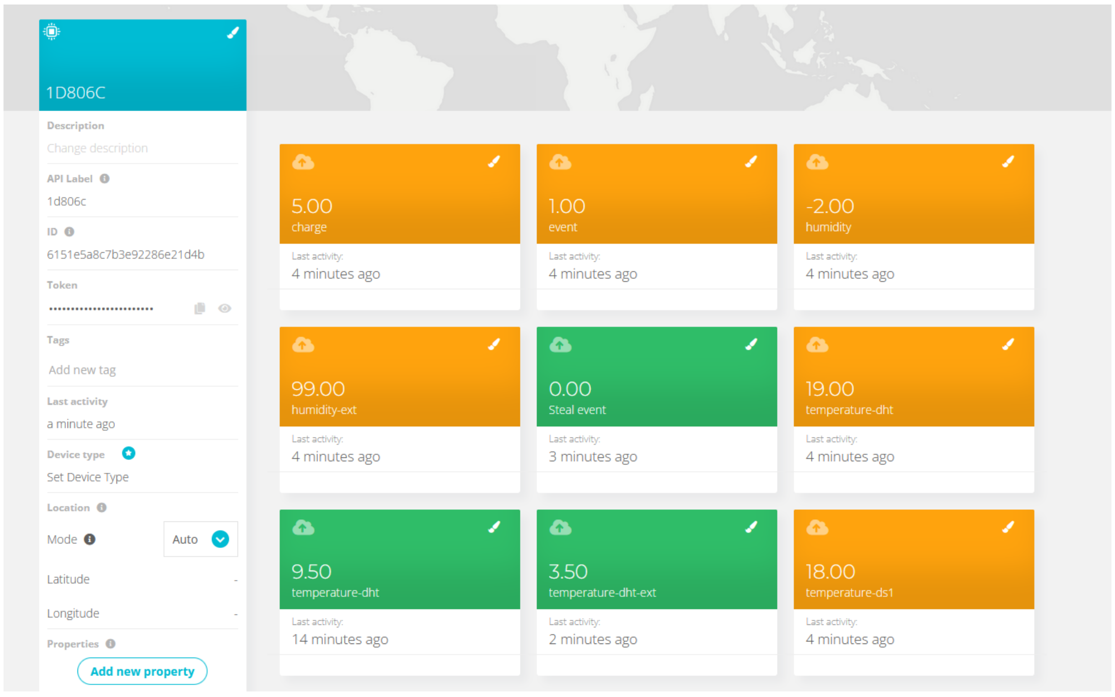
First, you need to create an account on Ubidots, we recommend to use the same name on that website to not lose the login. For this project, we have decided to use the same password on Sigfox and Ubidots.

To connect to this website, the logins are :

Login	001d80XX
Password	Sigfox21#

After that, you need to add your device, which should be named like the login if it is your school project. In our case, we selected "Blank device" on Ubidots.

When this is done, you can access your device and its token. This is very important because it will be the link between Sigfox and Ubidots so it is unique and should be kept secret.


1D806C

Description
Change description

API Label
1d806c

ID
6151e5a8c7b3e92286e21d4b

Token
••••••••••••••••••••••••••

Tags
Add new tag

Last activity
a minute ago

Device type
Set Device Type

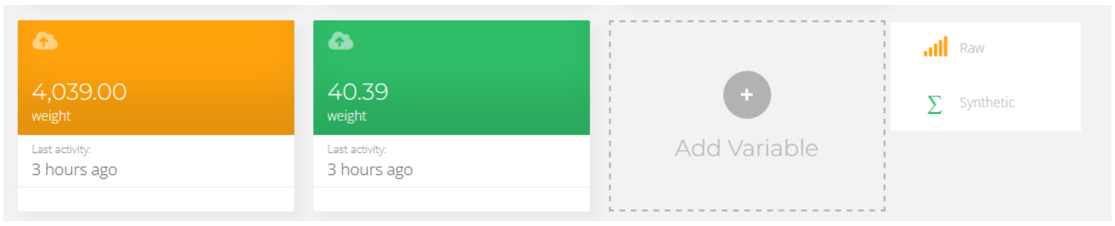
Location
Mode: Auto
Latitude -
Longitude -

Properties
Add new property

5.00 charge — Last activity: 4 minutes ago	1.00 event — Last activity: 4 minutes ago	-2.00 humidity — Last activity: 4 minutes ago
99.00 humidity-ext — Last activity: 4 minutes ago	0.00 Steal event — Last activity: 3 minutes ago	19.00 temperature-dht — Last activity: 4 minutes ago
9.50 temperature-dht — Last activity: 14 minutes ago	3.50 temperature-dht-ext — Last activity: 2 minutes ago	18.00 temperature-ds1 — Last activity: 4 minutes ago

Figure 4 - Exemple of our device on Ubidots

On this screenshot, you can see the different variables created in figure 3. You need to add them by clicking on "Add variable" and then on "Raw" :


4,039.00 weight — Last activity: 3 hours ago

40.39 weight — Last activity: 3 hours ago

Figure 5 - Screenshot of Ubidots

After that, you need to change the name for it to correspond to the ones gave in the callbacks made in Sigfox.

And after configurating all variables, you can see the inputs sent from the Arduino to Sigfox and then to Ubidots.

Thanks to this website, you can see charts of the temperature and humidity, the battery level and the weight of the beehive:

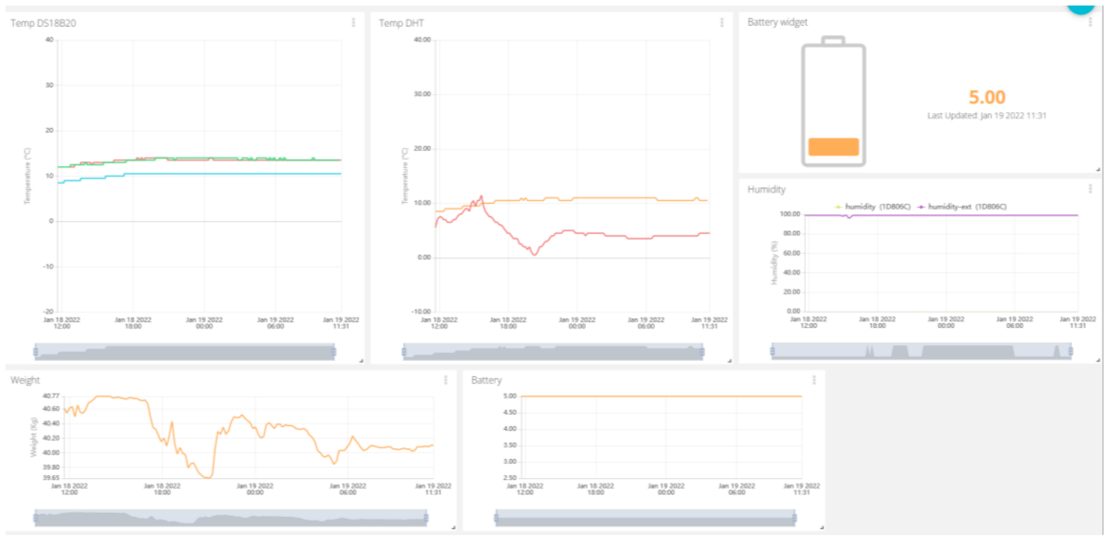


Figure 6 - Information of the beehive through Ubidots

With this website, it will send you alerts via mail and text message if your battery level is too low or if your beehive has been stolen.

Thanks to this interface, you can monitor the condition of your beehive and if everything is functioning properly.

VI – Installing the system on the beehive

The system on the beehive is set up like this :

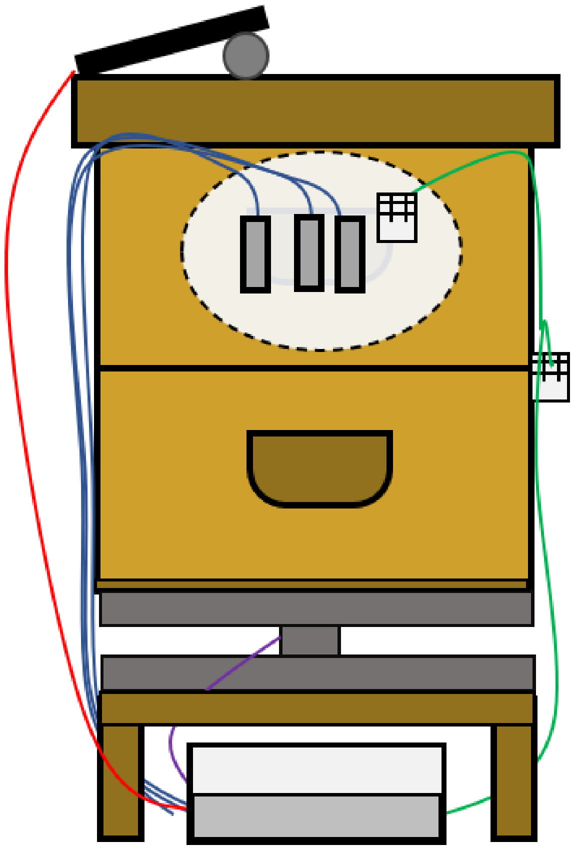


Figure 7 - Schematic of the beehive with the sensors

There are three temperature sensors and one DHT22 in the inside of the beehive and one DHT22 on the oustide. On the top of the beehive you have the solar cell. And it is on the weight gauge :

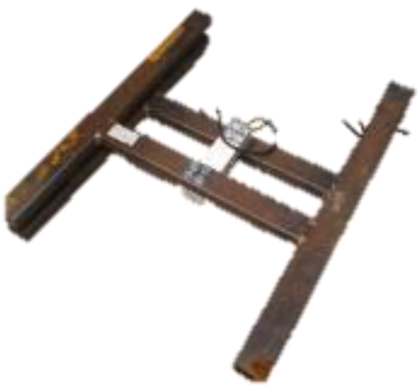


Figure 8 - Weight gauge

More precisely, the sensors are placed like this :

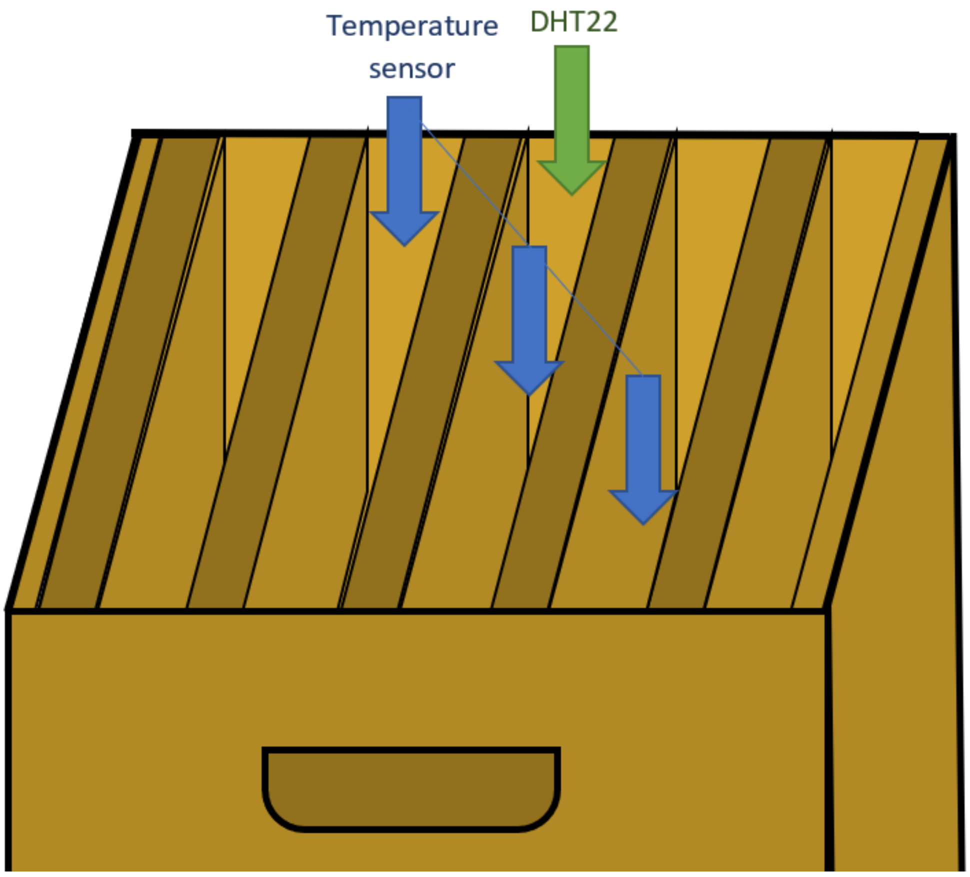


Figure 9 - Detailed position of the sensors

Conclusion

With this project, you should be able to monitor the status of your beehive and take action if needed for the well-being of the bees !

Thank you for reading this user guide and hope that everything is clear !