
A “Year of IoT” with MenterMon and NorthWalesTech

IoT Sensors Workshop

The guide to learning how to build IoT Sensors as part of a workshop with MenterMon and NorthWalesTech. **#yoiot**



Ymddiriedolaeth
Elusennol Ynys Môn
*Isle of Anglesey
Charitable Trust*



This work is licensed under a

[Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-nc-sa/4.0/).

OVERVIEW

The Workshop

In this workshop you will first register a sensor device on *TheThingsNetwork*. This is a public network for receiving wireless sensor data using LoRaWAN and sending over the Internet. You will then create the device using a LoRaWAN-enabled development board and attach various sensors to it. You will be able to see the sensor data, transmitted over LoRaWAN, and viewed on the online portal.

The Equipment

There are two different types of LoRaWAN enabled device to use in this workshop. The process and outcome is pretty much the same for both, but one has easier code to read, and the other a bit more tricky.

If you are a beginner to programming for devices such as *Arduino* or indeed a beginner to programming at all, then it is recommended to use the **TTN UNO** kit (green box) and instructions in section 2.

Otherwise, if you are proficient at programming and have some experience with, for example, Arduino boards, then opt for the **FeatherMO** kit (red box) and instructions in section 3.

We'll be working in groups. Please invite a beginner to work with you and help them get to grips with the process.

There may be the opportunity to try out both kits during this session, but there will be other sessions in the future to spend more time with either.

1. The Things Network (TTN)

Summary

TheThingsNetwork (TTN) is a public, open, community built network of gateways that receive wireless sensor data from LoRaWAN enabled devices. Anyone can register with a device, make it transmit, and retrieve the sensor data. That's exactly what we'll do now.

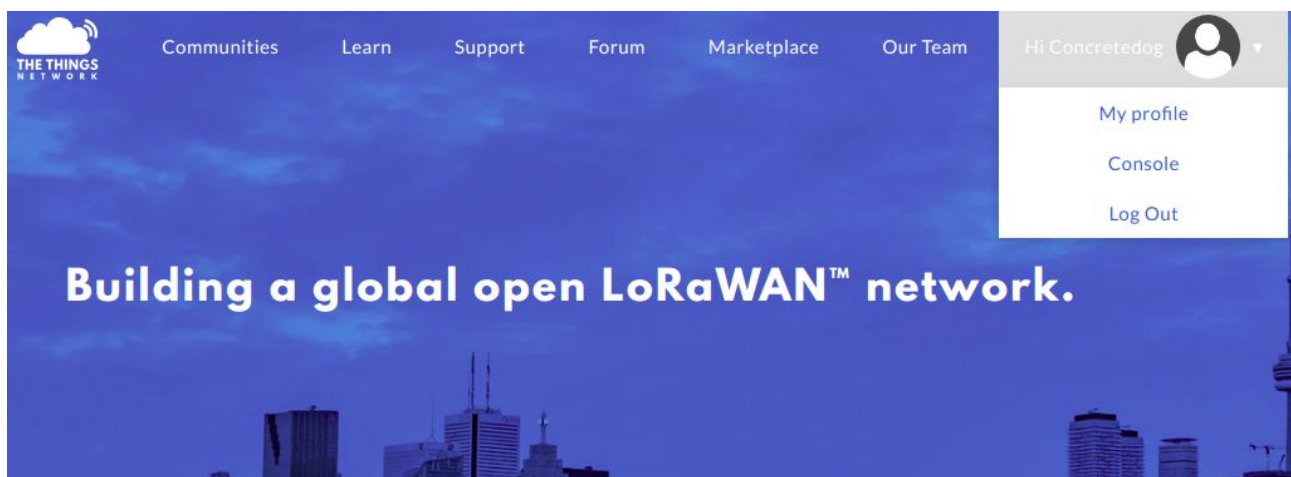
Everyone should create an account and register their own device. Please take turns to do this in your groups.

We will each:

- Register for an account
- Create a new application
- Register a device.

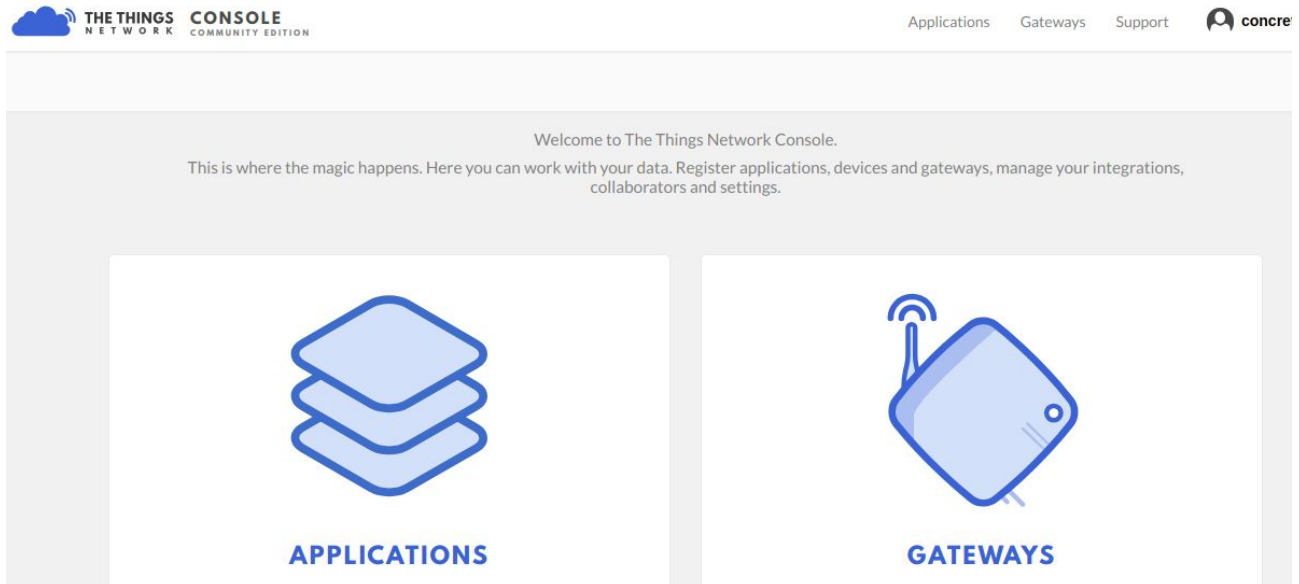
1-A Register an Account

- Go to the things network and register an account <https://www.thethingsnetwork.org>
- Once registered, on your dashboard click the “**console**” link by clicking the drop down menu next to your username on the top right hand side.

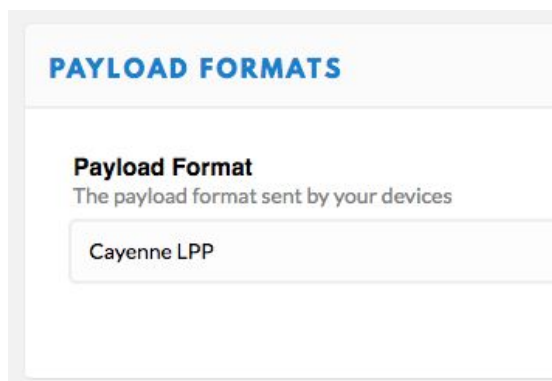


1-B Create an Application

An application is just a group of devices that will do a similar thing.



- In the console click the **applications** icon on the left
- Click the “**add application**” link and give your application an ID and a description. Note application ID’s have to be unique on the things network so if you try “test123” for example you will probably discover it is unavailable!
- Select the application to display the **Application Overview** page,
- Then choose the **Payload Formats** Tab and select **CayenneLPP** from the dropdown. You’ll learn what this means later.
- Click **Save**



1-C Register a Device

Each device must first be registered before the data can be received. The process for both device types is the same.

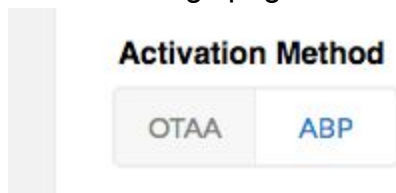
- Select the application to display the **Application Overview** page, then choose the **Devices** Tab and click the **register device** link. (or **get started by registering one**)
- Give the device a unique **Device ID**, it *doesn't matter what at this stage*.
- Click the button on the left of the Device EUI input to automatically generate this.



- Click the green register button at the bottom of the page.

In the resulting device overview you will notice that the activation method is set to **OTAA**, for these workshops it's important that we change this and use the **ABP** method.

- Click the settings tab in the upper right hand side
- On the settings page scroll down and click the **ABP** button next to the OTAA button



- Then at the bottom of the page un-tick the "**frame counter checks**" box. Note that this will bring up a yellow error message warning this is less secure and for development use only. Ignore this warning and click save.



- Click **Save**

2. Build An IoT Sensor with The Things Uno

The **TTN Uno** is an Arduino compatible development board, designed by the creators of TheThingsNetwork to make it really easy to get started with LoRaWAN and IoT. We've coupled it with sensors from the **Grove** family which makes it easy to plug and play!

Summary

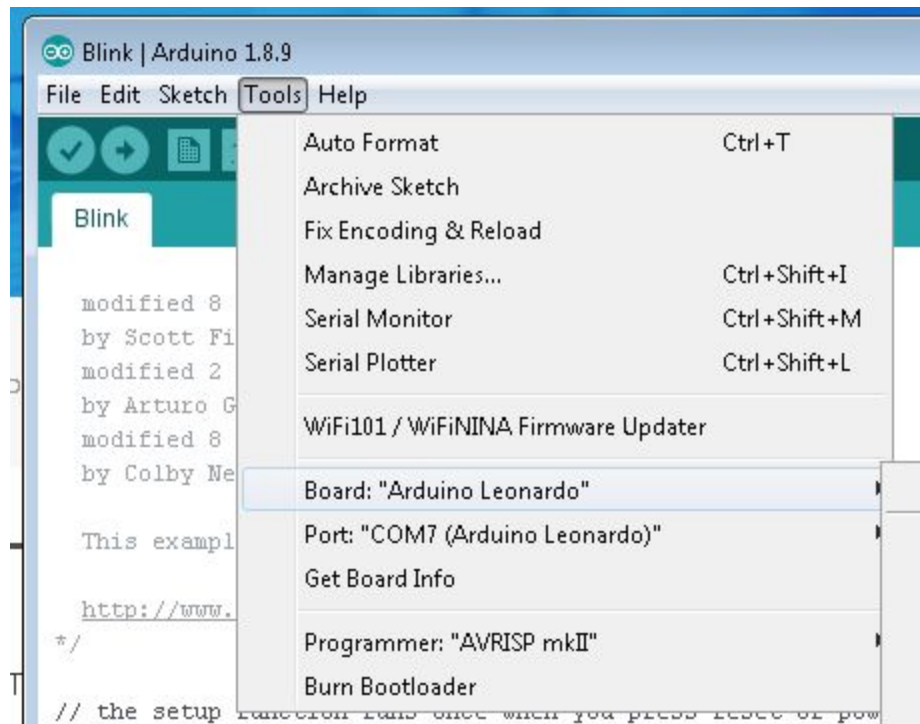
In this section you will:

- Setup the Arduino Developer Environment
- Get The Things Uno connected to your computer and make a basic program
- Connect a sensor and read some data.
- Transmit the data over the LoRaWAN TTN network and read the data.

2-A Setup the Arduino Environment and Connect The Things Uno

- Download and install the latest arduino IDE from <https://www.arduino.cc/en/Main/Software#download>
- If you are on windows, don't forget to install the device drivers included in the download.
- Plug in the device with the Micro USB cable.
- *Windows users:* If your device does not show up as a "Leonardo" when you plug it in, then please follow these steps:
 - a. Download drivers from here: <https://www.ftdichip.com/Drivers/VCP.htm>
 - b. Extract the ZIP somewhere you can find it.
 - c. Refer to the manual driver installation guide here: <https://www.arduino.cc/en/Guide/DriverInstallation>

- In the Arduino IDE click **Tools > Board** and then check it is set to **“Arduino Leonardo”**
- Click **Tools > Port** and select the one that points to a Leonardo.



- Next open a basic example as follows: **File > Examples > 01.Basics > Blink**



- Now click the compile button



- Upload this sketch/program to the Things Uno by clicking the upload button:
- You should see that a green LED labelled “13” near the usb socket on the Things Uno board is now flashing regularly once a second. Let’s change the rate at which it flashes to ensure we can edit, then upload new versions.
- Back in the blink sketch scroll down and change the delay values that are 1000 to 200

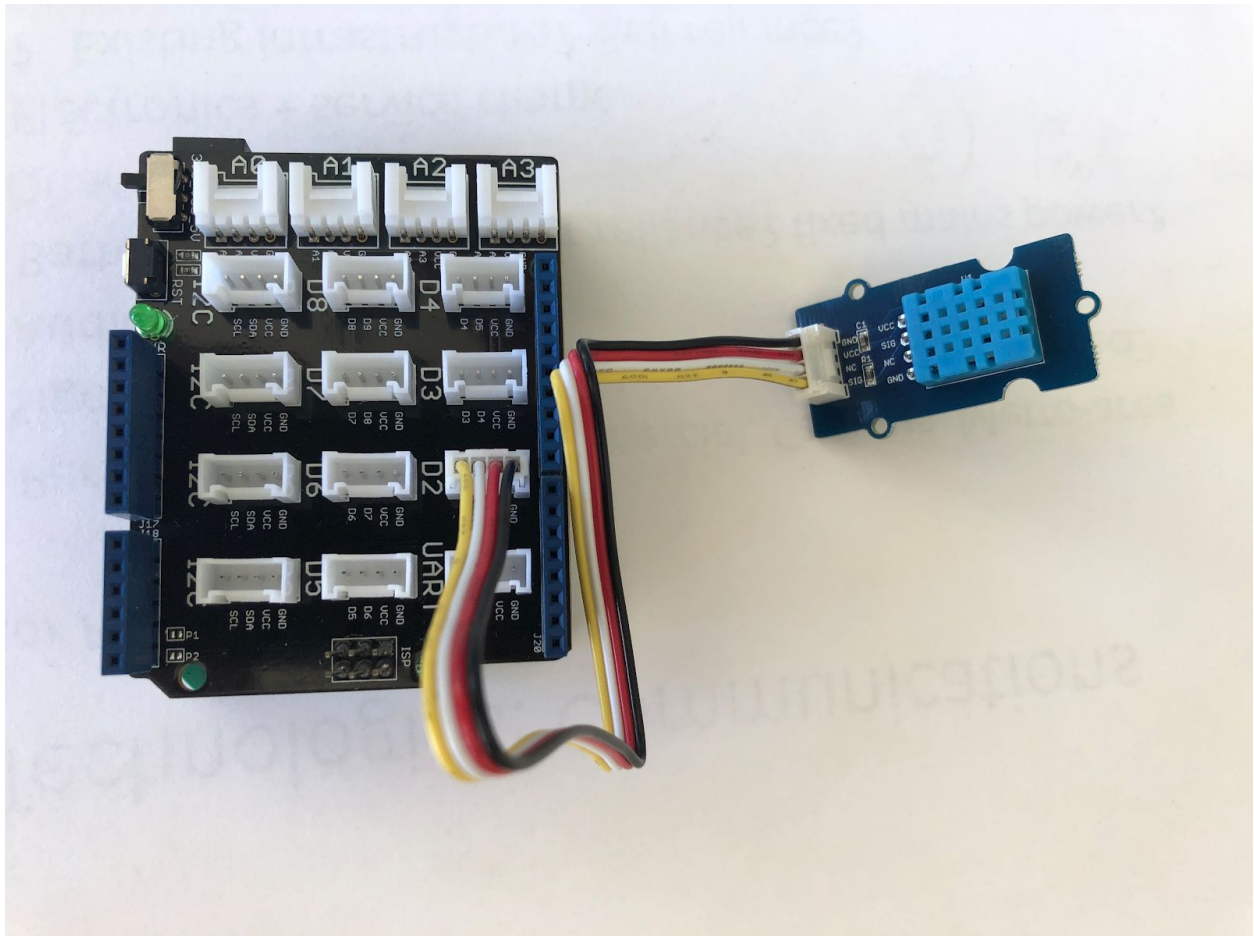
```
// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(200); // wait for a second
  digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
  delay(200); // wait for a second
}
```

- Then repeat the **Compile** and **Upload** steps above to verify and upload the sketch. You should now see the LED is flashing much more quickly!

2-B Connecting a Sensor, Read the Environment

Connect the sensor

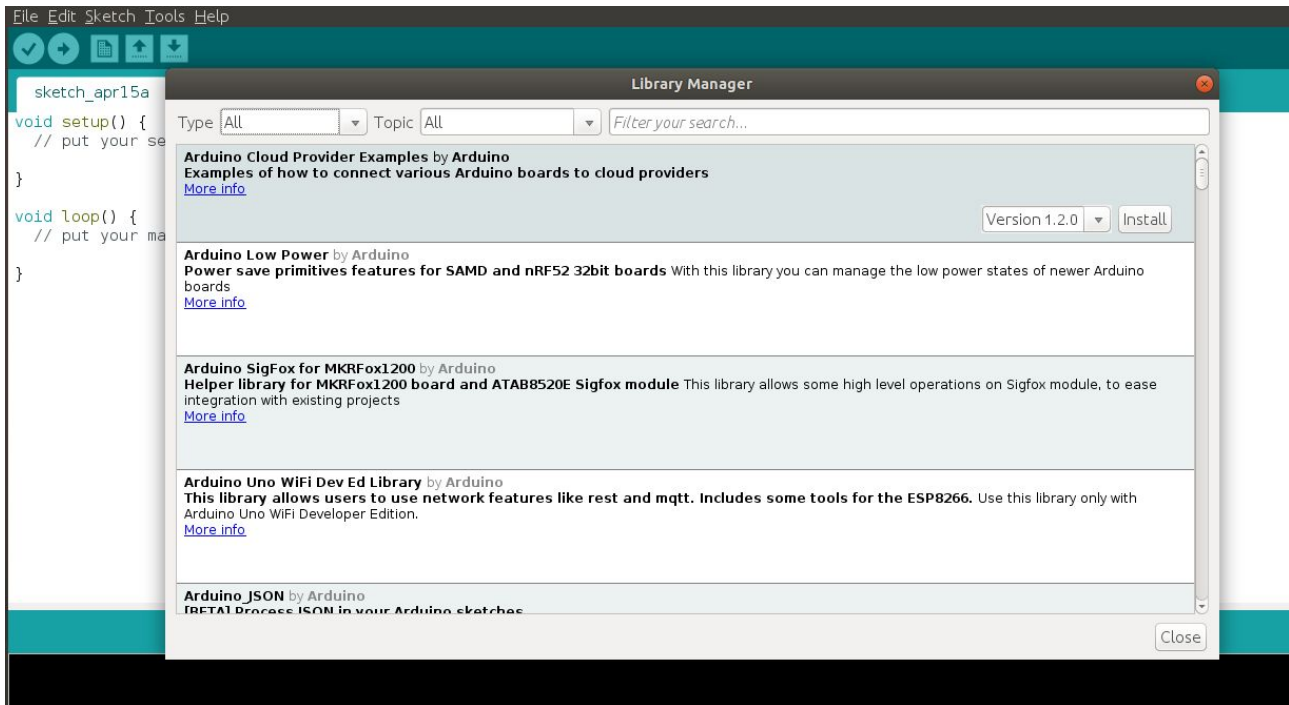
- Find the Grove Temperature and Humidity Sensor in the Kit.
- Use a connector wire and plug it into the Adapter using the port marked D2



Load the library.

Most of the sensors have a pre-built “library” of code which takes the hard work out of getting sensor data from the electronics. We need to download the library and add it to the Arduino IDE so we can use it in our code. We will use the **Arduino Library Manager**

- Open the Arduino IDE and then click **Tools > Manage Libraries**



- Using the search box, search for **Grove DHT11**





- Then click **Install** to download and install the library into the Arduino IDE
- Click **close** to close the library manager.

Load the Program

Our first program can be found here, browse here to have a look:

https://github.com/NorthWalesTech/yoiot-ThingsUno/blob/master/1_TempHum/1_TempHum.ino

- In the Arduino IDE, choose **File > New**
- Delete all the new code in the editor window *just get rid of the empty setup() and loop()*
- Copy and Paste the program code from the link above
- Save it with project name **TempHum**
- Click the Arrow button  to compile and upload the code the TTN UNO
- Click the Magnifying glass  to open the serial monitor and inspect the sensor data.
- **#winner** points awarded for the maximum temperature and/or humidity you can observe. (no blow-torches!) Change the code to record the max and min for each.

3-C Transmit the Data

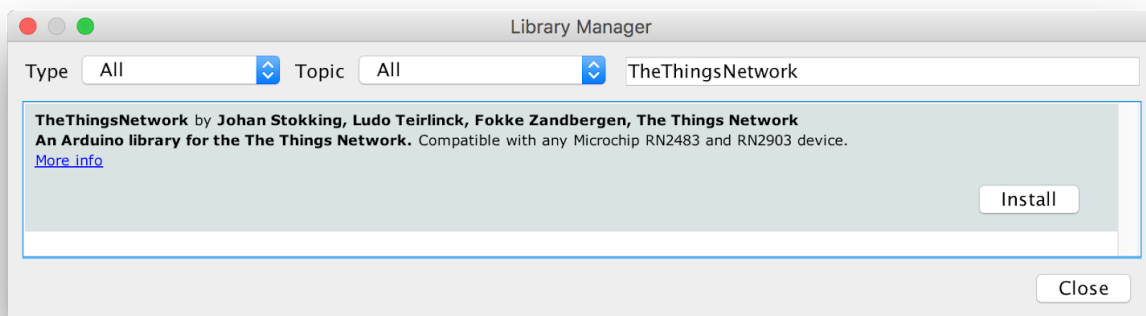
Summary of steps coming up.....

- Load more libraries
- Load the new program
- Edit it for your TTN device keys
- Upload and run it!

Load more libraries

We are now going to add another library to enable our code to use the LoRaWAN transmitter on the ThingsUno device. Again using the **Arduino Library Manager**.

- Open the Arduino IDE and then click **Tools > Manage Libraries**
- Using the search box, search for **TheThingsNetwork**



- Then click **Install** to download and install the library into the Arduino IDE

Load the Program

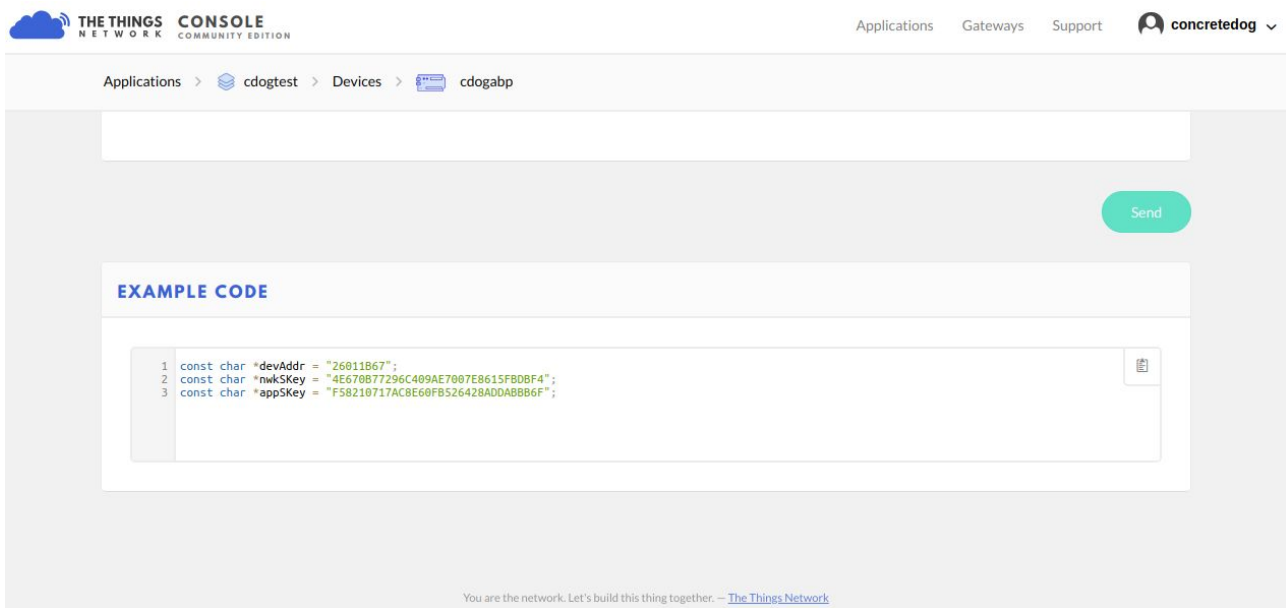
Our transmitter program can be found here, browse here to have a look:

https://github.com/NorthWalesTech/yoiot-ThingsUno/blob/master/2_TempHum_TTN_ABP/2_TempHum_TTN_ABP.ino

- In the Arduino IDE, choose File > New
- Delete all the new code in the editor window *just get rid of the empty setup() and loop()*
- Copy and Paste the program code from the link above
- Save it with project name **TempHumLoRaWAN**

We now need to copy some settings from the TTN console to this code.

- Head to the TTN console and find the device settings page.
- Scroll to the bottom and locate the **Example Code** section.



- Copy these three lines
- Back in the Arduino Code editor, replace the the corresponding lines near the top of the file to replace the ones with all '0's with your own specific settings.

So replace these;



```
File Edit Sketch Tools Help
[Icons: Checkmark, Arrow, File, Upload, Download]

SendABP

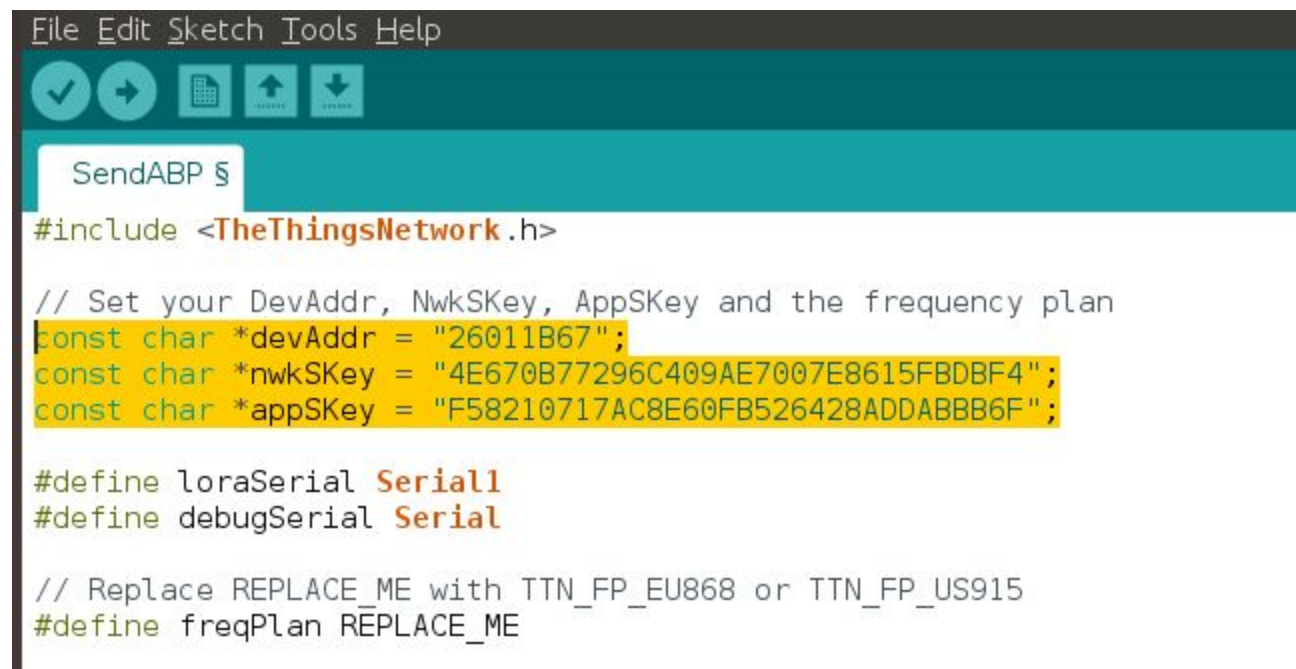
#include <TheThingsNetwork.h>

// Set your DevAddr, NwkSKey, AppSKey and the frequency plan
const char *devAddr = "00000000";
const char *nwkSKey = "00000000000000000000000000000000";
const char *appSKey = "00000000000000000000000000000000";

#define loraSerial Serial1
#define debugSerial Serial

// Replace REPLACE ME with TTN_FP_EU868 or TTN_FP_US915
```

with the ones you copied



```
File Edit Sketch Tools Help
[Icons: Checkmark, Arrow, File, Upload, Download]

SendABP §



#include <TheThingsNetwork.h>

// Set your DevAddr, NwkSKey, AppSKey and the frequency plan
const char *devAddr = "26011B67";
const char *nwkSKey = "4E670B77296C409AE7007E8615FBDBF4";
const char *appSKey = "F58210717AC8E60FB526428ADDABBB6F";

#define loraSerial Serial1
#define debugSerial Serial

// Replace REPLACE_ME with TTN_FP_EU868 or TTN_FP_US915
#define freqPlan REPLACE_ME
```

Compile and Run FTW!

- Click the Arrow button  to compile and upload the code the Feather
- Click the Magnifying glass  to open the serial monitor and inspect the sensor data!!
- Take a look at the **Data** tab on TheThingsNetwork console for your device. You should see transmissions being received. You should also see the data fields behind decoded `temperature_1: 20.0` etc.

References and Links

1. <https://github.com/NorthWalesTech/yoiot-ThingsUno>
2. <https://www.thethingsnetwork.org/docs/devices/uno/quick-start.html>
3. <https://www.arduino.cc/en/Guide/ArduinoLeonardoMicro#toc2>
4. <https://www.arduino.cc/en/Guide/Troubleshooting#toc16>





4. Libraries and Sensors

The Kits come with various sensors to experiment with. Here are the links to the information about how to use them, and how to find any libraries that are needed, using the Arduino Library Manager

Sensors Examples

Most of the Grove Libraries comes with some examples for reading data. These can be found in the **File > Examples > ...your sensor...** menu.

Sensors

<p>Grove Sensor - Magnetic Switch http://wiki.seeedstudio.com/Grove-Magnetic_Switch/</p> <pre>int switched digitalRead(...);</pre>	
<p>Grove Button http://wiki.seeedstudio.com/Grove-Button/</p> <pre>int pressed digitalRead(...);</pre>	
<p>Grove Sensor - Temp and Humidity http://wiki.seeedstudio.com/Grove-TemperatureAndHumidity_Sensor/</p> <div><p>Grove Temperature And Humidity Sensor by Seeed Studio Version 1.0.0 INSTALLED Arduino library to control Grove Temperature And Humidity Sensor, it contains chip DHT11 AM2302. This temperature & humidity sensor provides a pre-calibrated digital output. A unique capacitive sensor element measures relative humidity and the temperature is measured by a negative temperature coefficient (NTC) thermistor. It has excellent reliability and long term stability. More info</p></div>	
<p>Grove Buzzer http://wiki.seeedstudio.com/Grove-Buzzer/</p> <pre>pinMode(PIN, OUTPUT); digitalWrite(PIN, HIGH/LOW); // buzz on/off</pre>	

<p>Grove Rotary Angle Sensor</p> <p>http://wiki.seeedstudio.com/Grove-Rotary_Angle_Sensor/</p> <pre>int angle_val analogRead(...);</pre>	
<p>Grove Light Sensor</p> <p>http://wiki.seeedstudio.com/Grove-Light_Sensor/</p> <pre>int light_val analogRead(...);</pre>	
<p>Grove Water Sensor</p> <p>http://wiki.seeedstudio.com/Grove-Water_Sensor/</p> <pre>int water_present digitalRead(...);</pre>	
<p>Grove Ultrasonic Distance Sensor</p> <p>http://wiki.seeedstudio.com/Grove-Ultrasonic_Ranger/</p> <div> <p>Grove Ultrasonic Ranger by Seeed Studio</p> <p>Arduino library for controlling Grove Ultrasonic Ranger, using gennal I/O communication. More info</p> <p>Version 1.0.1 <input type="button" value="Install"/></p> </div>	
<p>Grove 4-digit Display</p> <p>http://wiki.seeedstudio.com/Grove-4-Digit_Display/</p> <div> <p>Grove 4-Digit Display by Seeed Studio Version 1.0.0 INSTALLED</p> <p>Arduino library to control Grove_4Digital_Display TM1637. 4 digit display module is usually a 12 pin module. In this Grove gadget, we utilize a TM1637 to scale down the controlling pins into 2 Grove pins. It only takes 2 digital pins of Arduino or Seedeuino to control the content, even the luminance of this display. For projects that require of alpha-numeric display, this can be a nice choice. More info</p> </div>	