

H2S-Dev board Tutorials

LoRaWAN Helium network sensor development board

V1.2

Initial programming environment setup

1. Abstract

H2S-Dev board is a ready-to-use, Open source hardware, Arduino compatible, LoRaWAN Helium network sensor development board. It is capable of implementing a number of sensor and positioning applications.

This tutorial will guide you through the initial setup of the programming environment to run an example code and start developing your own application with LoRaWAN Helium network sensor development board - the H2S-Dev board.

2. Install Arduino IDE

Install or update your Arduino IDE to the most recent version.

<https://www.arduino.cc/en/software/>

The screenshot shows the Arduino IDE 2.3.6 download page. On the left, there's a large image of the Arduino IDE interface with two windows open. The top window shows a sketch named "weatherstation.ino" with the code:void setup() {
} // put your setup code here, to run once:
void loop() {
} // put your main code here, to run repeatedly:

```
On the right, there's a section titled "Arduino IDE 2.3.6 Release notes" which includes a brief summary of the new features and a link to the documentation. Below that is a "DOWNLOAD" button, which is highlighted with a red rectangle. Further down, there's a "Nightly Builds" section with a link to download a preview of the incoming release.
```

Arduino IDE 2.3.6
[Release notes](#)

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger. For more details, check the [Arduino IDE 2.0 documentation](#).

Linux AppImage (64-bit X86-64) [DOWNLOAD](#)

Nightly Builds
Download a preview of the incoming release with the most updated features and bugfixes.

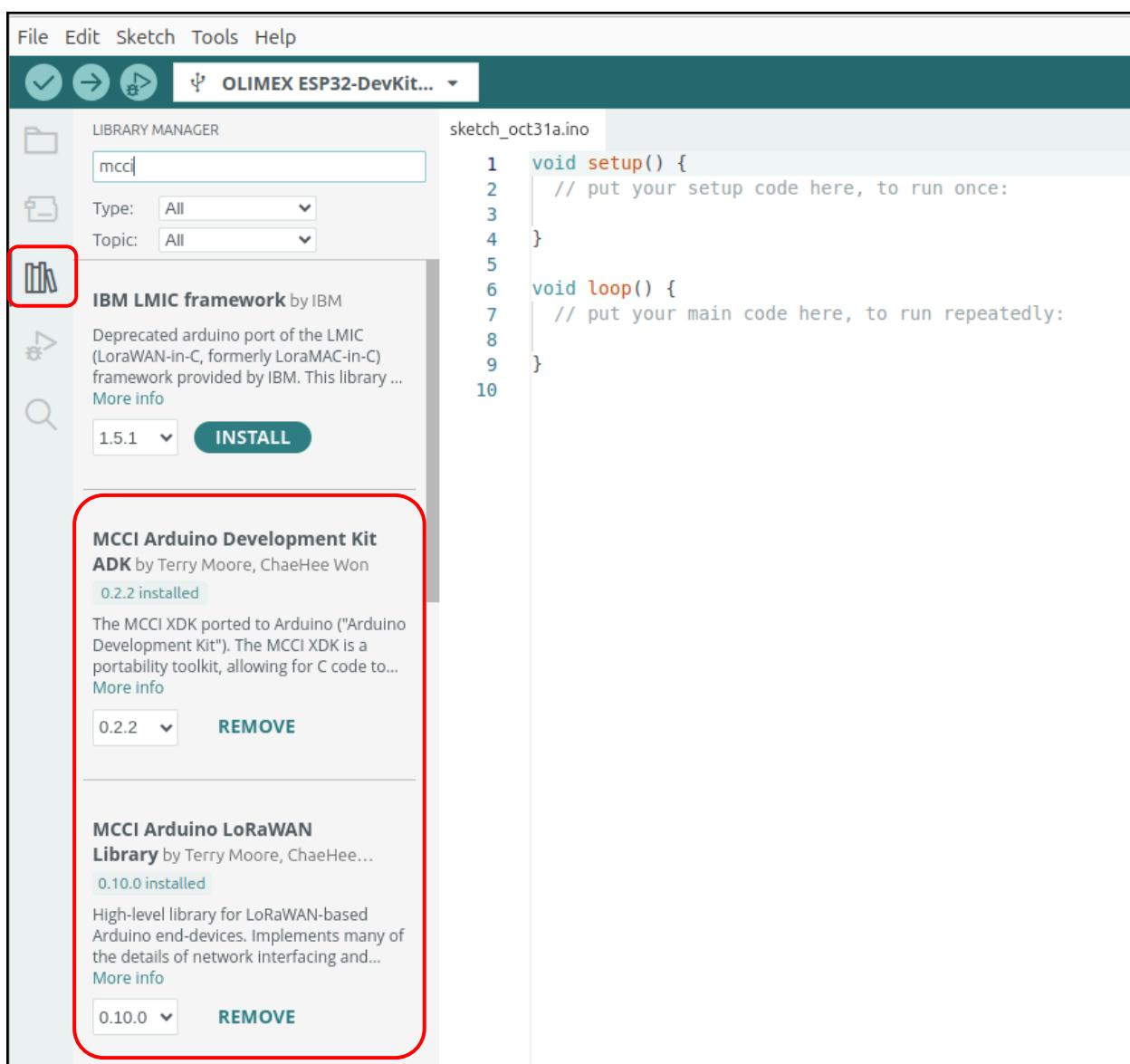
The Arduino IDE 2.0 is open source and its source code is hosted on [GitHub](#).

3. Install libraries required

A few libraries are required for the H2S-Dev board to start functioning.

3.1 Install LoRaWAN libraries for Arduino

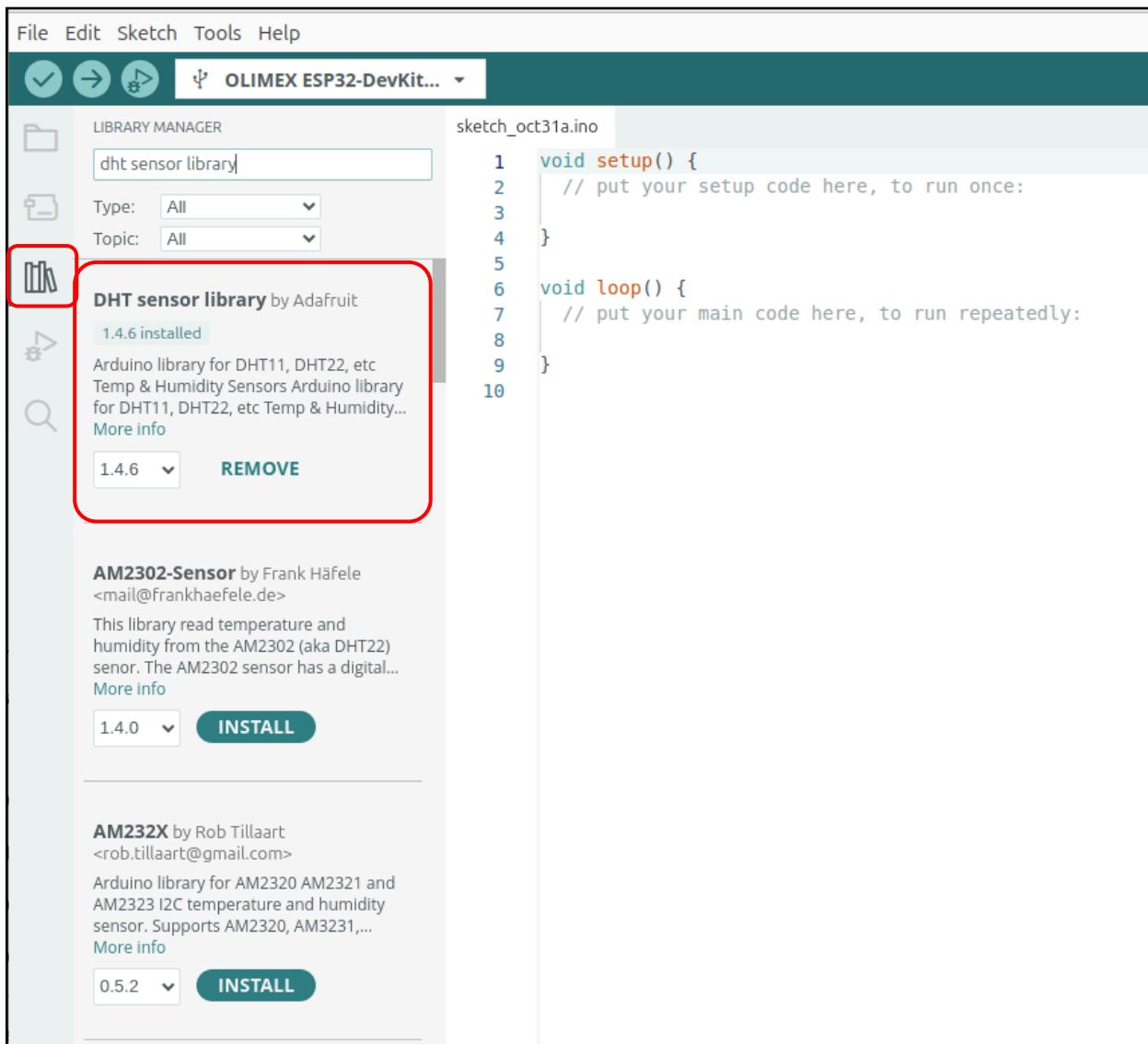
As per our experience so far, we recommend the following libraries to be installed together with all the dependencies required.



More information: <https://github.com/hobbyiot/HELIUM-SENSORS>

3.2 Install DHTxx library

For a specific application example - reading temperature and humidity from a DHT11 sensor, you will also need to install the corresponding library support. We recommend the THT sensor library from Adafruit. Please select the most recent version and install it.



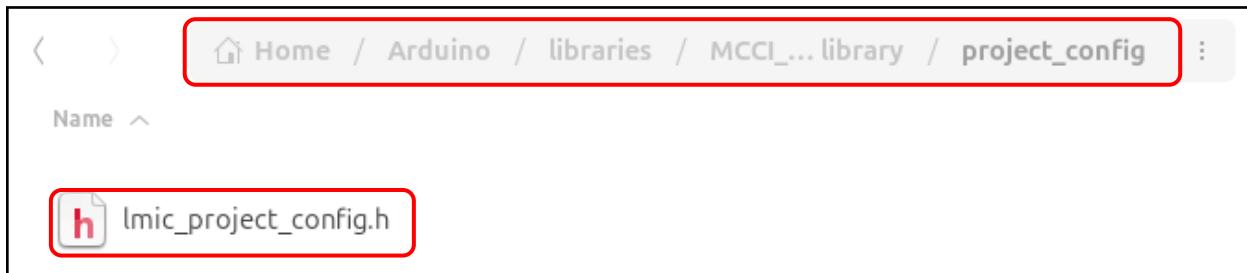
More information: <https://github.com/hobbyiot/HELIUM-SENSORS>

4. Configure lmic_project_config.h

An important thing to set up before starting to use the H2S-Dev board is to correctly configure the project parameters for LoRa communication. Please find the *project_config* folder within the Arduino libraries installed as shown in the picture below and open the *lmic_project_config.h*.

For a typical Linux installation the path to that file could be as follows:

/home/user/Arduino/libraries/MCCI_LoRaWAN_LMIC_library/project_config



Open the file *lmic_project_config.h* and set by comment/uncomment using '#':

- **Frequency** set according to your country region;
- **Chipset** select *CFG_sx1276_radio*;
- **Interrupts** allow using interrupts by *LMIC_USE_INTERRUPTS*

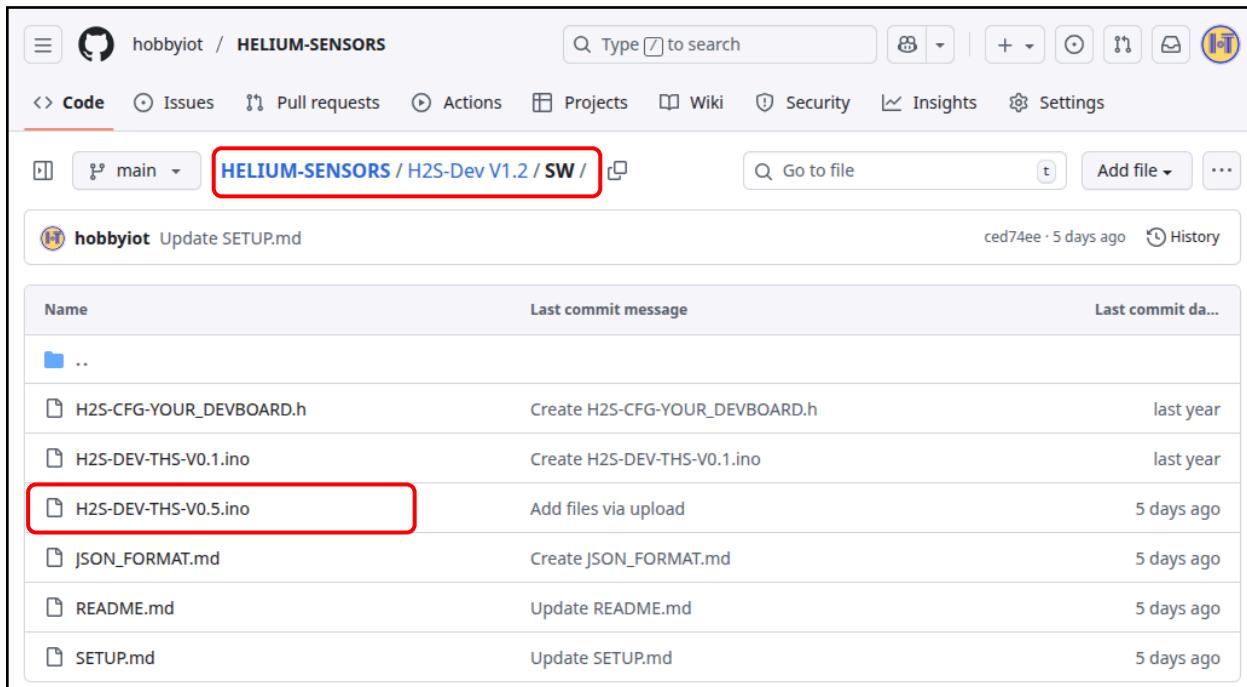
A screenshot of a code editor window showing the content of 'lmic_project_config.h'. The code includes several preprocessor directives. Some lines are highlighted with red boxes:

- #define CFG_eu868 1
- #define CFG_us915 1
- #define CFG_au915 1
- #define CFG_as923 1
- #define LMIC_COUNTRY_CODE LMIC_COUNTRY_CODE_JP /* for as923-JP; also define CFG_as923 */
- #define CFG_kr920 1
- #define CFG_in866 1
- #define CFG_sx1276_radio 1
- #define CFG_sx1261_radio 1
- #define CFG_sx1262_radio 1
- #define ARDUINO heltec_wifi_lora_32_V3
- #define LMIC_USE_INTERRUPTS

5. Download the example code from the project Github repository

<https://github.com/hobbyiot/HELIUM-SENSORS/blob/main/H2S-Dev%20V1.2/SW/H2S-DEV-THS-V0.5.ino>

The example code provided uses a DHT11 temperature and humidity sensor and sends data over the Helium network in a specific format according to the Helium network requirements. DHT11 is programmed to be connected to GPIO13 in the example but you can change the configuration as per your needs.

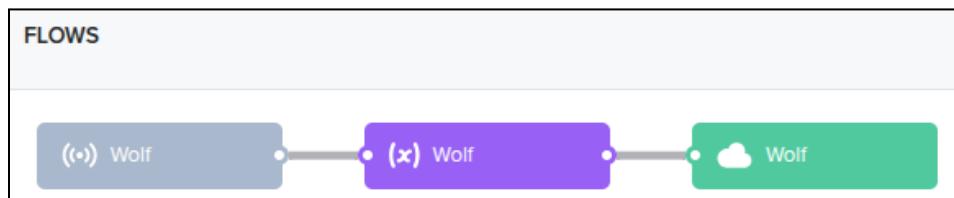


The screenshot shows a GitHub repository named 'hobbyiot / HELIUM-SENSORS'. The 'Code' tab is selected. A red box highlights the folder 'main' and the file 'H2S-DEV-THS-V0.5.ino' within it. The file was updated by 'hobbyiot' on 'ced74ee · 5 days ago'. The repository contains several other files: 'H2S-CFG-YOUR_DEVBOARD.h', 'H2S-DEV-THS-V0.1.ino', 'JSON_FORMAT.md', 'README.md', and 'SETUP.md'. All these files were last modified 5 days ago.

The algorithm periodically reads the data from the connected DHT11 sensor. Temperature and humidity values are then combined into a specific JSON structure and sent over the Helium network.

The specific JSON format and various ways of receiving and using the data are all described in separate documents.

Generally, the data path is constructed as a set of logical rules (flows) within the Helium console environment and could be directed to a shared spreadsheet, MQTT broker, etc.

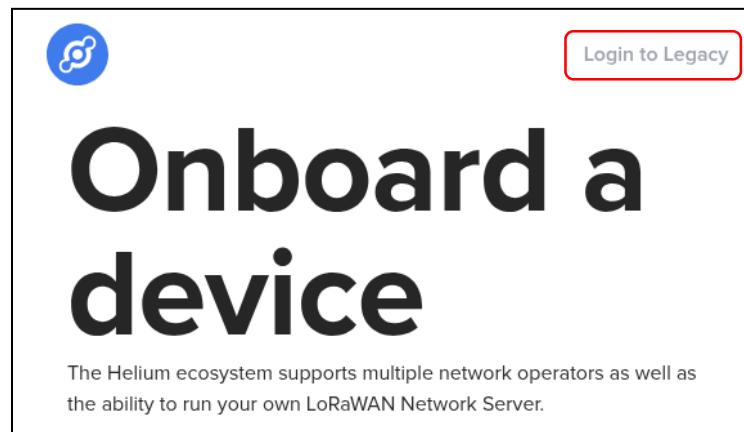


More information: <https://github.com/hobbyiot/HELIUM-SENSORS>

6. Set up a Helium network connection

To connect your H2S-Dev board to the Helium network, you need device credentials to be generated. Currently the system has been tested with the legacy console interface. Please use the following link to connect:

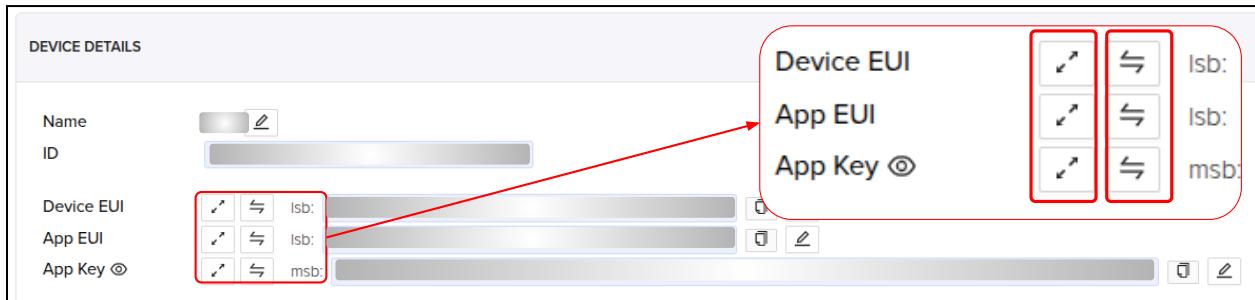
<https://console.helium.com/>



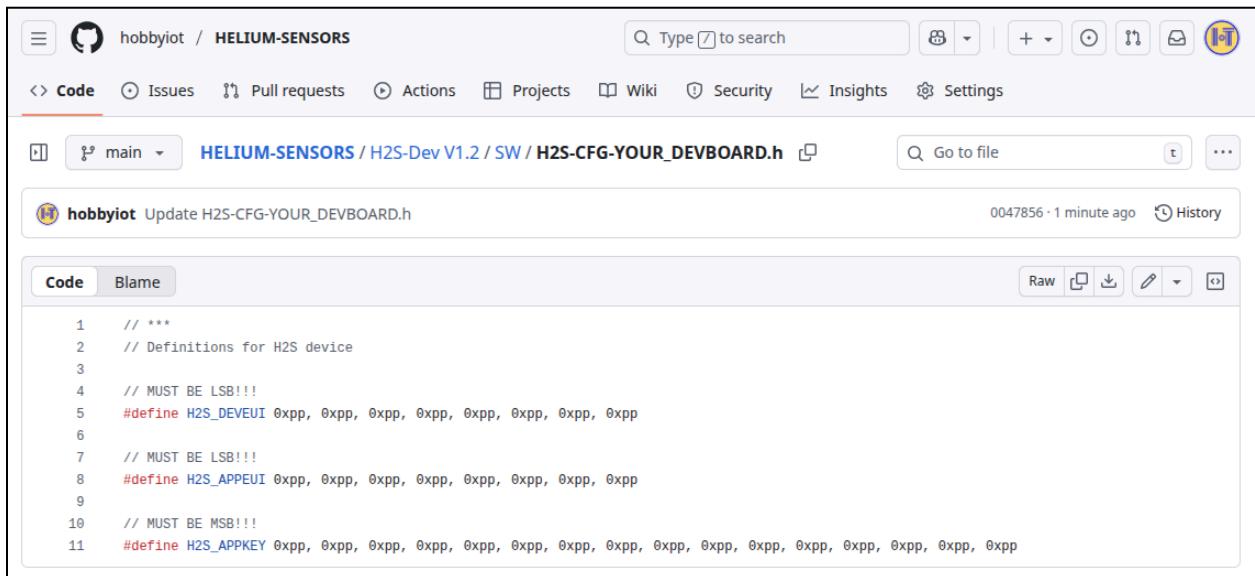
The screenshot shows the 'DEVICE DETAILS' configuration page. It includes fields for 'Name' (with a pencil icon), 'ID' (a long gray input field), and three credential fields: 'Device EUI', 'App EUI', and 'App Key'. The 'App Key' field contains a placeholder string of asterisks. The 'Activation Method' is set to 'OTAA', and the 'Profile' is set to 'None'. The 'Device EUI', 'App EUI', and 'App Key' fields are highlighted with a red rounded rectangle.

More information: <https://github.com/hobbyiot/HELIUM-SENSORS>

Use the “expand” key  and select the proper order  (MSB/LSB)



Copy and paste these values into the configuration file ([H2S-CFG-YOUR_DEVBOARD.h](#))



```
// ***
// Definitions for H2S device
//
// MUST BE LSB!!!
#define H2S_DEV_EUI 0xpp, 0xpp, 0xpp, 0xpp, 0xpp, 0xpp, 0xpp, 0xpp
//
// MUST BE LSB!!!
#define H2S_APP_EUI 0xpp, 0xpp, 0xpp, 0xpp, 0xpp, 0xpp, 0xpp, 0xpp
//
// MUST BE MSB!!!
#define H2S_APPKEY 0xpp, 0xpp
```

7. Compile the project for ESP32_DevKit_LiPo

In case of correct settings you will see the new device join process within the Helium console.

Event Log		<input type="checkbox"/> Expand All	<input type="checkbox"/> Filter Events w/ Commands	Show Dropped Uplinks:	<input type="checkbox"/> Late	<input type="checkbox"/> Inactive Device
Event	Type	No. of Hotspots				
+  0	Uplink	1				
+  0	Join Accept	1				
+  0	Join Request	21				

Your H2S-Dev board is now connected to the Helium network! Good luck!

More information: <https://github.com/hobbyiot/HELIUM-SENSORS>