**Hardware Used**

ARDUINO SIM - MKR GSM 1400 CELLULAR KIT is used for this example. It is a 3G device and can be purchased here and is only $2 more than the cost of the device itself.

<https://store.arduino.cc/usa/sim-bundle>

The SIM card which comes with the bundle allows you to connect only to the Arduino IoT cloud. Because of that, an example of how to get data from the Arduino IoT cloud for use elsewhere is included here.

<https://github.com/Freeno83/Dog-Tracker/blob/master/Arduino_IoT_Cloud_API.ipynb>

This Jupyter notebook can be ran in [Google Colab](https://colab.research.google.com/) without having to install any software on your PC. You will need to provide the 3 ID’s which are blank at the top of the notebook.

Other SIM cards can be used with the MKR GSM 1400 but that is not covered in this example.

A close up of a computer

Description automatically generated

**Arduino Code**

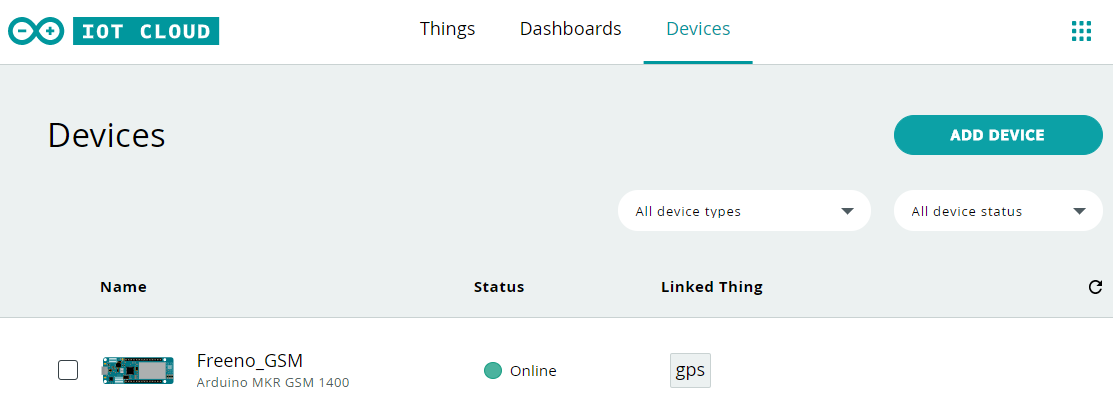
The code for dog tracker is very simple and is based on example sketches from the Arduino IoT Cloud and TinyGPS++ libraries. The following are the main callouts:

* Originally this project was attempted with the Arduino MKR GPS library but when connected to the Arduino IoT Cloud, the GPS data would never update. This issue has been reported here:
  + <https://github.com/arduino-libraries/Arduino_MKRGPS/issues/8>
* The fix for the above issue was to use the “TinyGPS++” Library. In order for it to communicate with the GPS shield, the “Serial1” serial port has to be initialized to 9600 baud.
* The line of code which waits until a serial monitor window is opened before proceeding needs to be commented out for the device to run stand-alone. If it is not commented out, the code will never proceed past that point and no GPS data will ever be received or sent.
* A delay is placed in the main loop to restrict how often data is sent to the IoT Cloud. This depends on the use case and is intended to save available data when the SIM card being used has a limitation on how much data can be sent or when additional charges are incurred based on the amount of data sent.
* The property named in the “ArduinoCloud.addProperty” function call must be the same as the property defined in Arduino IoT Cloud and is case sensitive.

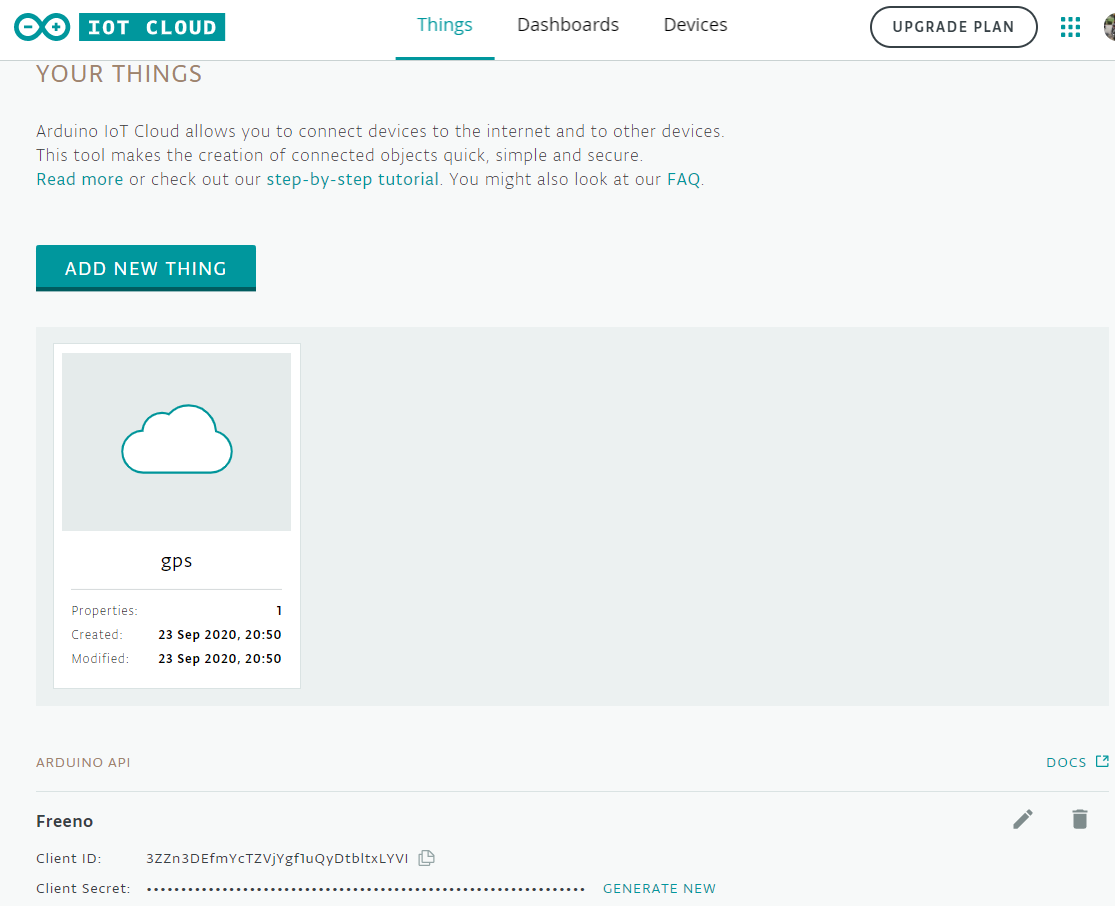
**Arduino IoT Cloud Setup**

Below are the general steps needed to get setup to transmit data to Arduino IoT Cloud:

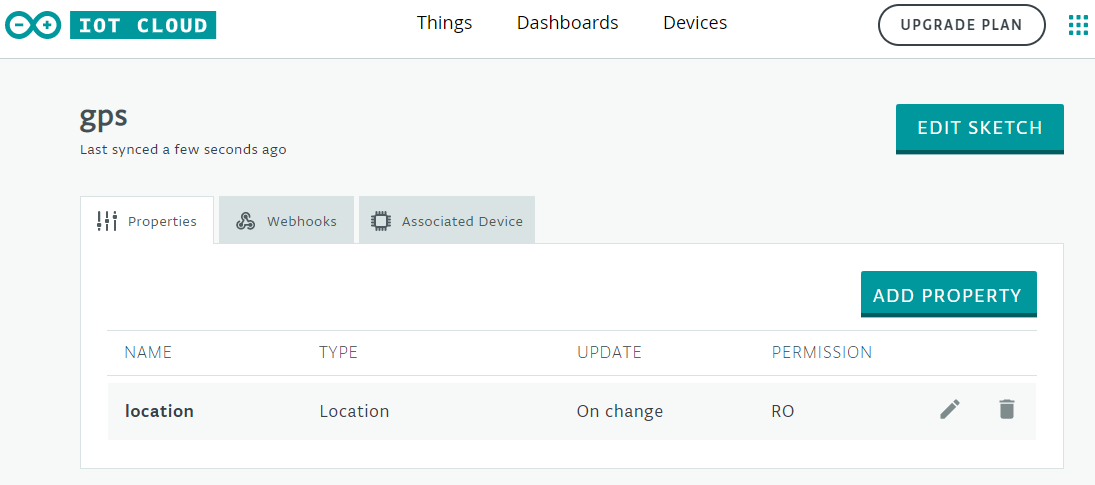
1. Add a device: <https://create.arduino.cc/iot/dashboards/devices>
   1. In this step, some security programming is done to your board, without completing this step, it is not possible to transmit data to Arduino IoT Cloud
   2. The “Device ID” from here is programmed into the device via the sketch as “BOARD\_ID”



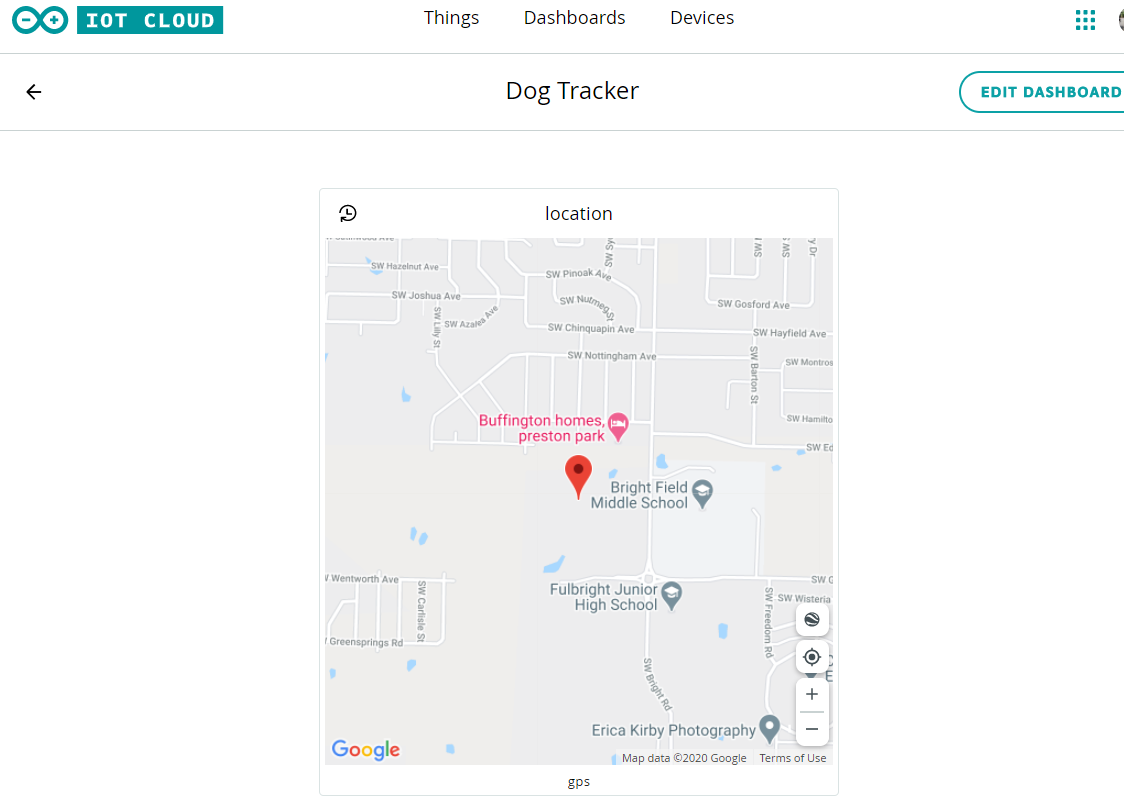
1. Add a thing: <https://create.arduino.cc/iot/things>
   1. In the case of this example, a thing called “gps” has been added.
   2. The “THING\_ID” from here needs to be programmed into the board via the sketch.
   3. This is where you can also choose to create an API to access the data externally



1. Add a property to the thing
   1. Here a read only location property has been added



1. Create a dashboard
   1. Here a single map widget is used and linked to “location”



1. Program your board using the sketch from:
   1. <https://github.com/Freeno83/Dog-Tracker/blob/master/dog_tracker.ino>
   2. All credentials at the top of the sketch must be filled in, 6 separate pieces of information
2. Download the “Arduino IoT Cloud Remote” from the app store
3. Open the app and view your dashboard. Move the device around to make sure the location is changing correctly. You will need a battery such as [this one](https://www.adafruit.com/product/328?_gclid=5b7652375cbb79.35408738-5b7652375cbbf7.83226318&gclid=CjwKCAjw8MD7BRArEiwAGZsrBQIWv4eRSTsKkwSzAhpDjcCspEC_zHW2FP86aeDrvkn-5DBadJ3qXhoChyIQAvD_BwE).

Map

Description automatically generated

**Furthering work**

The next part of this project is to solve the following:

* How to mount to the dog in the smallest possible form factor
* Compute battery life and extend as much as possible
* Make waterproof including an external port to charge the battery
* Have a way to turn the device off/on and to know the state
* Exception handling such as cannot connect to GSM network or Arduino cloud

If you have good ideas, especially about how to mount in the smallest possible form factor while being water proof and acceptable to the dog, please let me know at <freeno83@gmail.com>.