Google Summer of Code 2023

Final Evaluation for "Development of a DIY IoT Physics lab for educators**"**

General Information:

Organization's Name: Open Technologies Alliance - GFOSS

Mentor Names: Hariton Polatoglou, Panagiotis Koustoumpardis

Contributor's Name: Thomas Kampouris

Country: Greece

A brief description of the goals of the project

The point of this project is to provide the code and sencors for five different physics experiments that will be controlled by multiple Node MCU ESP8266 microcontrollers, so that they can be conducted wirelessly through a server from anywhere, at any time. The concept of an IoT physics lab is exciting and using this project as an example, anyone can build and modify it to his/her needs. The code provided is simple to read, has plenty of instructional comments and can also be used directly with very minor changes.

Experiments, Sensors and Improvements

Five different experiments have been implemented during this project, each needing different sensors and interfaces for the people to use them. In this section all the required sensors and their libraries are presented. The description of all the experiments can be found in **GitHub**. Note: For the code it is important to install the specific versions for each library, so that it can run smoothly.

Mass and spring vertical oscillator:

In this experiment the motor used to lower the weight to the desired place will be a DC geared motor with extended shaft. This motor is widely used in follow the line Arduino robots and is quite easy to use and very low cost. This motor needs a dedicated driver board to provide the desired voltage and current and not damage the controller board. This driver is the L298N. The electromagnet actuator used to hook the weight can be the WF-P30/22 which is more than capable of holding a weight up to 100N while needing 12V DC. As for the distance measurement the widely used ultrasonic sensor HC-SR04 is suitable for this experiment. In the ThingsBoard Dashboard the user will be able to command the microcontroller to start the motor in order to hook the weight and bring it to the starting point. When the weight is at the desired location the user will commence the release of the weight and the distance measurements will be displayed. The required libraries for the code are ThingsBoard (Version 0.6.0), Pubsubclient (Version 2.8.0), ArduinoJson (Version 6.19.4), ESP8266WiFi (Latest version), Wire (latest version) and AccelStepper (Latest version).

Refraction and Reflection of light

In this experiment there are only three things needed. A step motor like 28BYJ-48 with its control board ULN2003 and a camera. The user will use a controll knob in the ThingsBoard Dashboard in order to control the angle of the laser. The user will be able to see the refracted or reflected beam of the laser with the camera and measure the degrees with the help of the background goniometer. The live feed from the camera unfortunately cannot be streamed in the ThingsBoard server, so the user can use any streaming platform of his/her own choosing. The required libraries for the code are ThingsBoard (Version 0.6.0), Pubsubclient (Version 2.8.0), ArduinoJson (Version 6.19.4), ESP8266WiFi (Latest version) and Servo (Latest version).

Simple pendulum

For this experiment the sensors and actuators are the following. The electromagnet actuator used to hook the weight is the WF-P30/22, which is the same as for the first experiment. For measuring the angle over time, the MPU6050 Gyro and Accelerometer can be fixed on the rotation axis of the pendulum. By measuring the angle with the gyro instead of a direct on axis sensor, friction is reduced drastically and the pendulum can continue to oscillate for longer periods of time. The step motor for pulling the weight to the electromagnet can be the same 28BYJ-48 with its control board ULN2003 as used in the previous example. In the ThingsBoard Dashboard the user can command the microcontroller to pull pendulum of balance and hold it there with the electromagnet. Then the pendulum is released and the experiment starts. The values of the angle vs time will also be displayed in this Dashboard. The required libraries for the code are ThingsBoard (Version 0.6.0), Pubsubclient (Version 2.8.0), ArduinoJson (Version 6.19.4), ESP8266WiFi (Latest version), Wire (latest version) and AccelStepper (Latest version).

Energy Boxes

For the implementation of this experiment the most suitable sensor is the DHT11 humidity and temperature sensor. It is a low cost and easy to use solution that can get accurate results. Multiple sensors are used for each enclosure and an additional one is used to measure the environment outside the boxes in order to have a base temperature to compare to. The ThingsBoard Dashboard will provide the corresponding data from all the different boxes and graphs showing the data. The required libraries for the code are ThingsBoard (Version 0.6.0), Pubsubclient (Version 2.8.0), ArduinoJson (Version 6.19.4), ESP8266WiFi (Latest version) and DHT (Lates version).

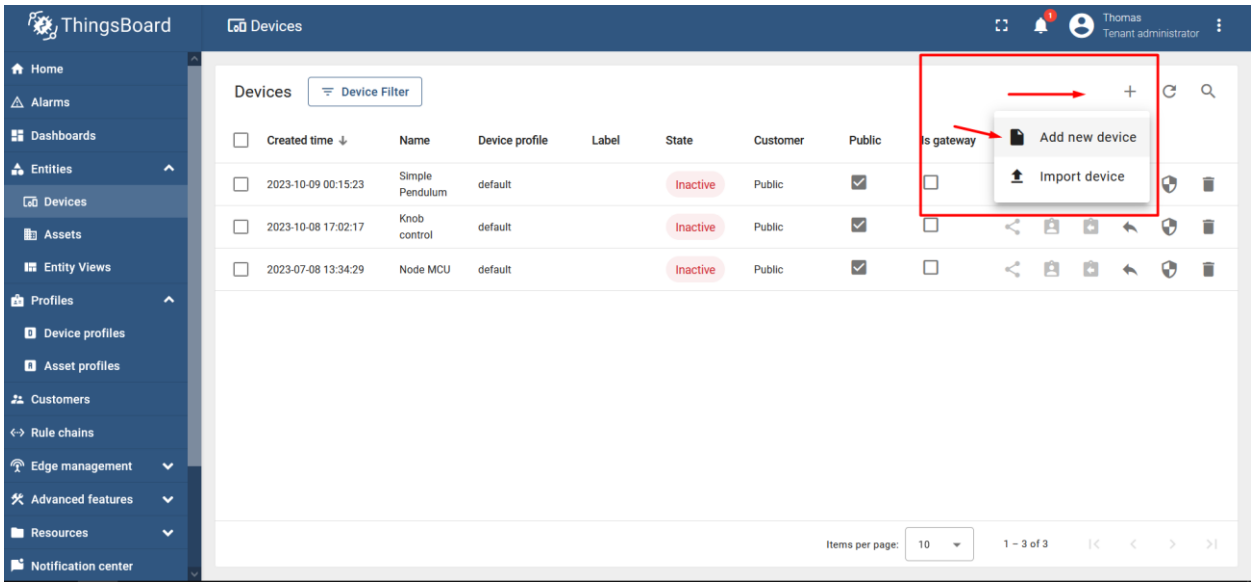
Robotic arm

For the fifth experiment the robotic arm will be controlled with the help of 5 servo motors. For small loads, the classic servo 9g SG90 is a very low-cost solution, but for the supporting and base servos a MG995 or similar servo is better suited. A camera is needed so that the user will be able to see the moving object and the robotic arm. In the ThingsBoard Dashboard the user will be able to control the robotic arm using the control knobs. The live feed from the camera unfortunately cannot be streamed in the ThingsBoard server, so the user can use any streaming platform of his/her own choosing. The required libraries for the code are ThingsBoard (Version 0.6.0), Pubsubclient (Version 2.8.0), ArduinoJson (Version 6.19.4), ESP8266WiFi (Latest version) and Servo (Latest version).

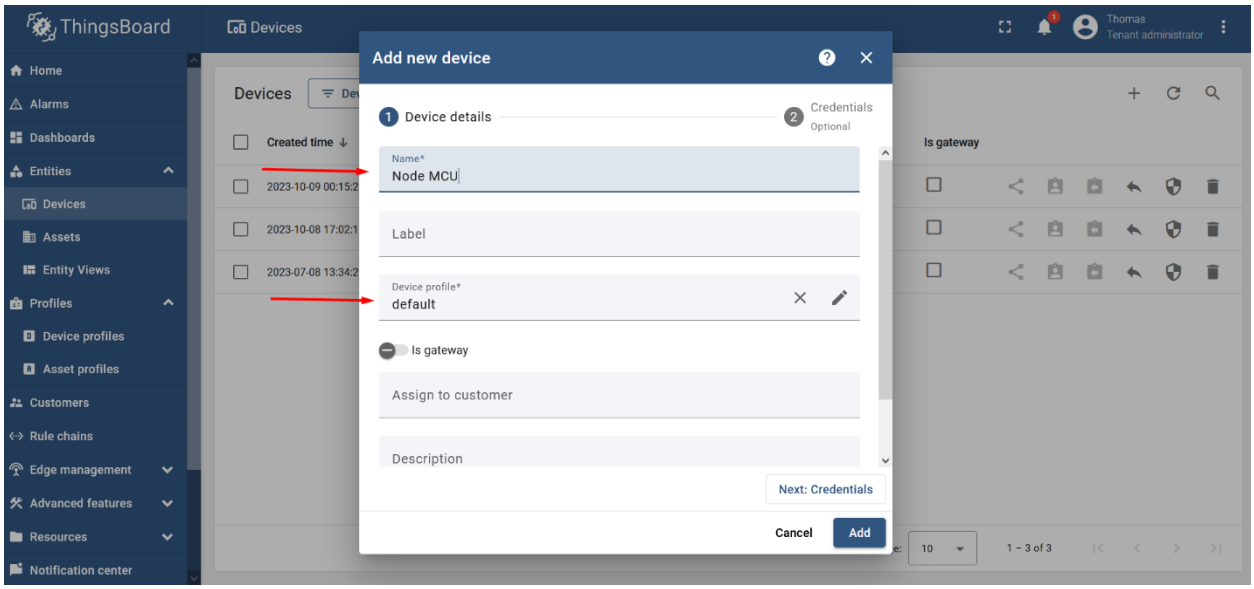
Make new device in ThingsBoard

Here are the steps you need to take to create an new device in ThingsBoard.

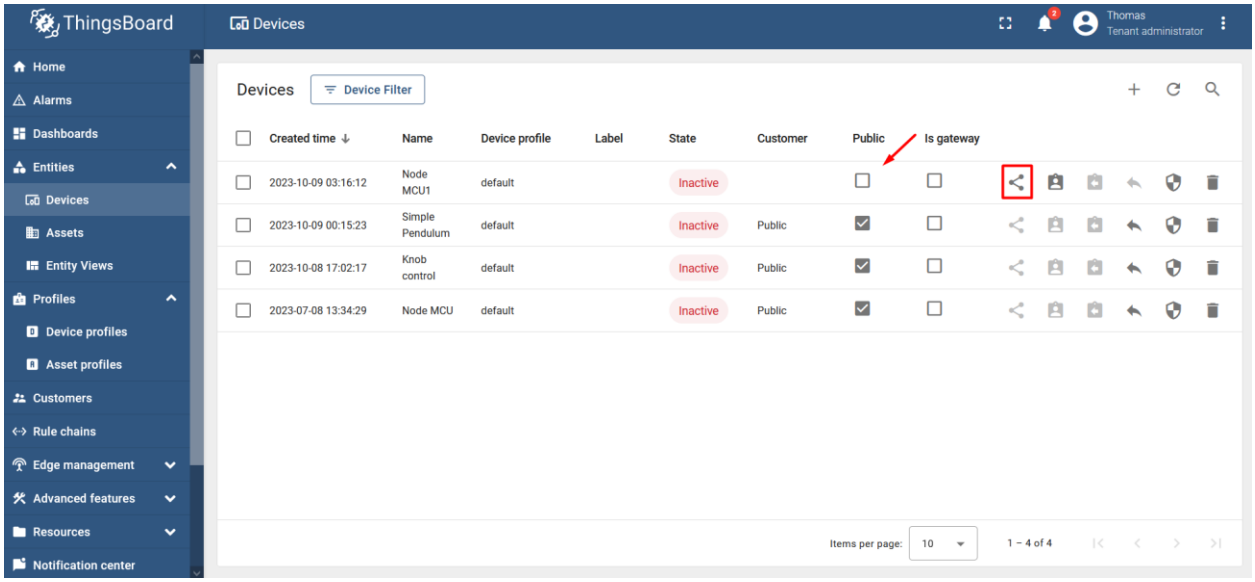
1. First you go to the Devices page and then you click on the + icon -> Add new device



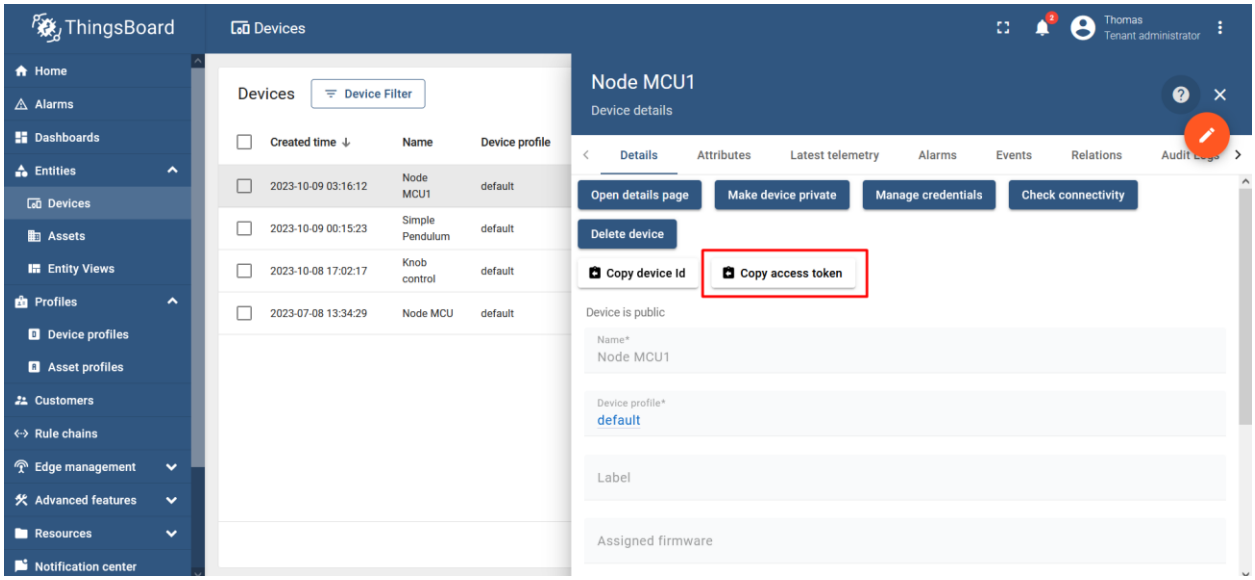
2. You name the device and you set the Device profile to Default and press add



3. You make the device Public by pressing the icon with the three dots

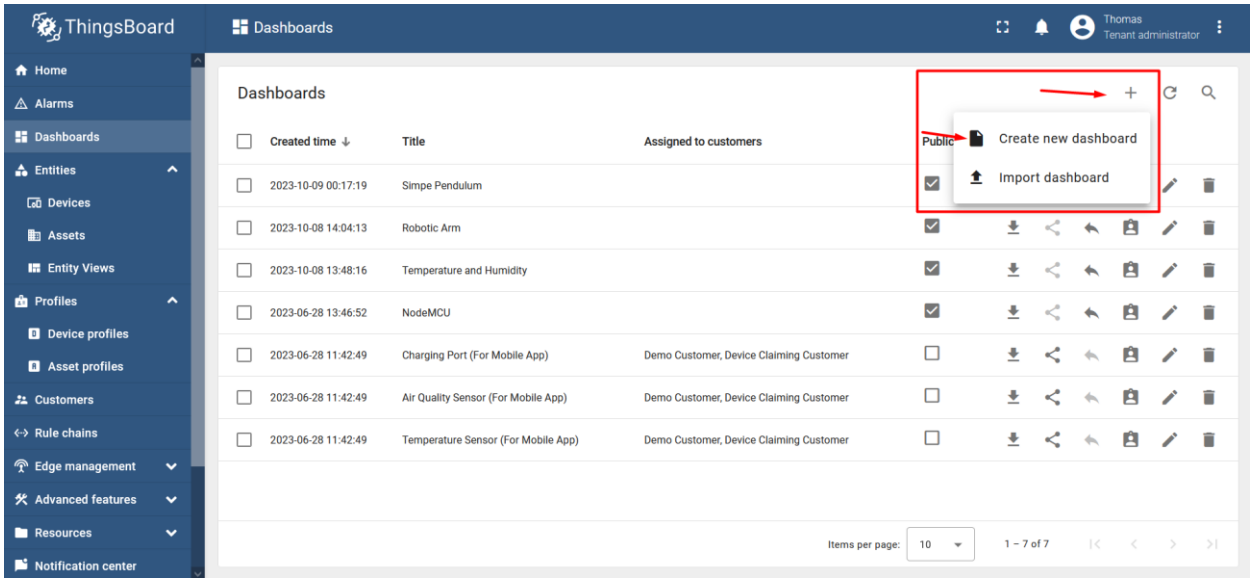


4. When you click on your new device you can get the Access token from here. It is used in the code.

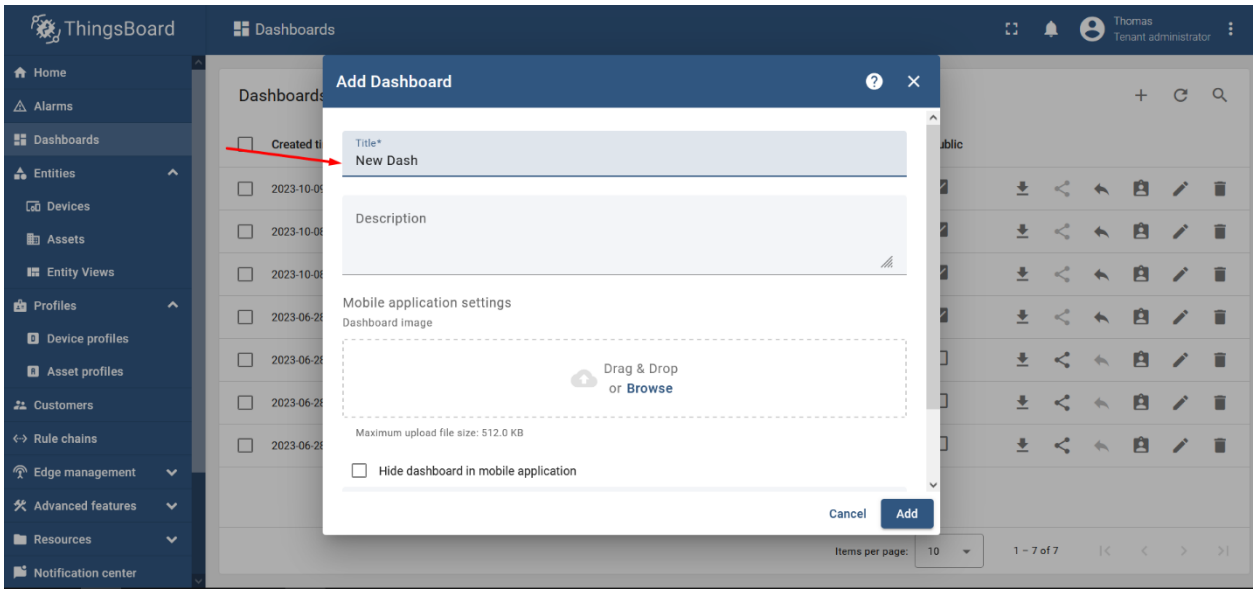


Make new Dashboard in ThingsBoard

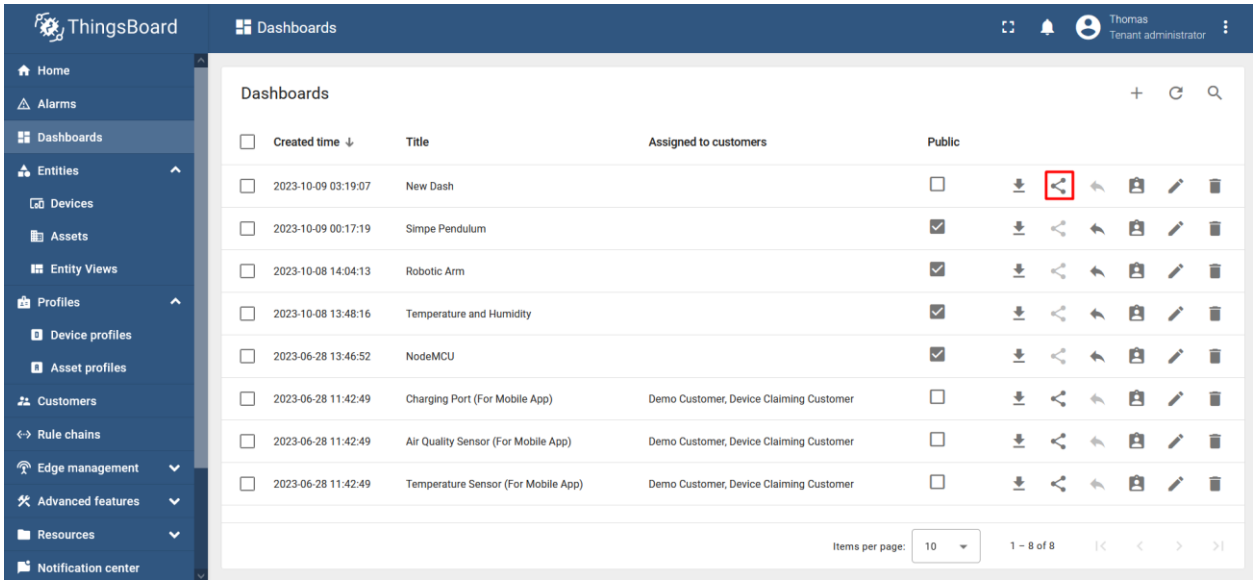
1. Go to you Dashboard page and click on the + icon -> Create new dashboard



2. Give your dashboard a name and click Add

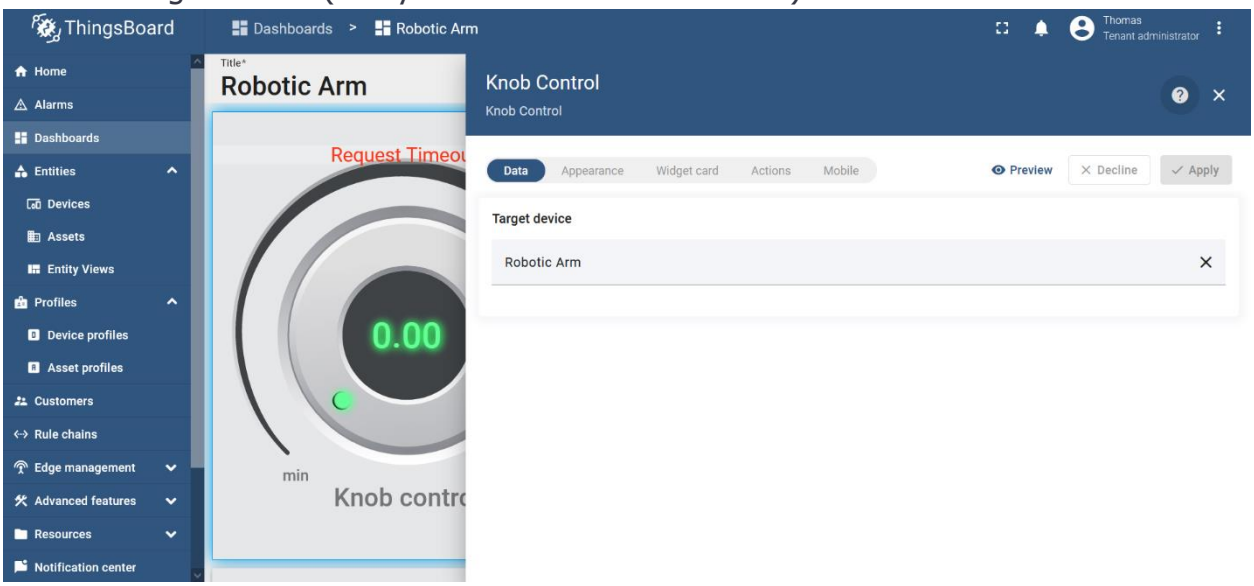


3. (optional) You can make your dashboard public, but all the devices you use in it must be public as well.

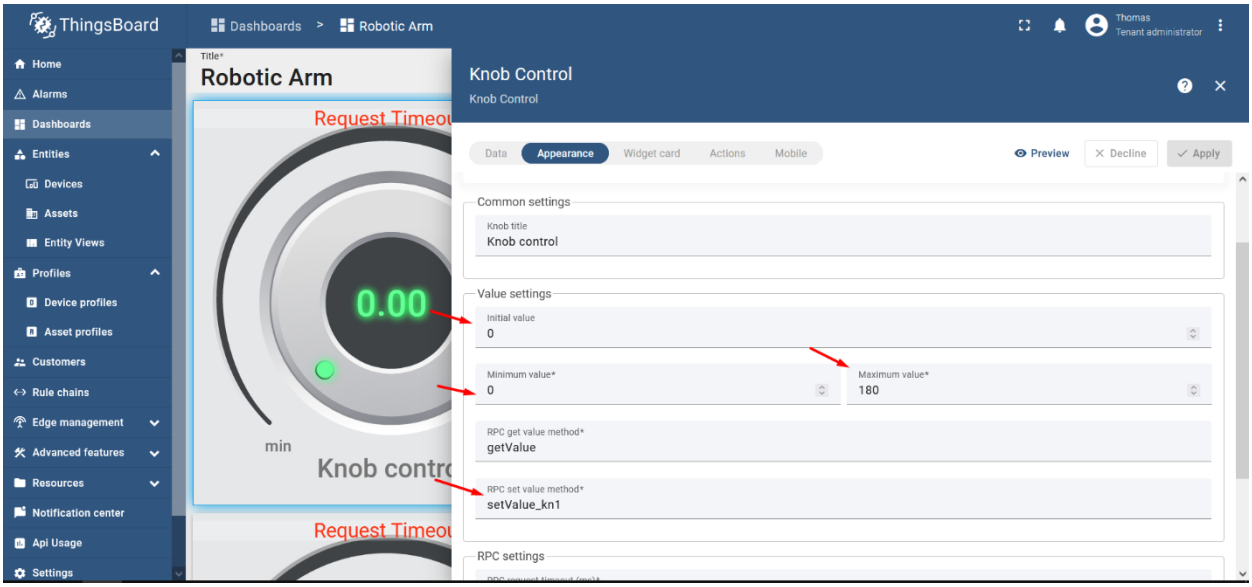


Set up the Control knob

You can find the knob control widget in the edit mode in any dashboard under the control widgets bundle. Then by clicking on the pencil icon in the Data page you must add the Target device (in my case is called Robotic Arm).

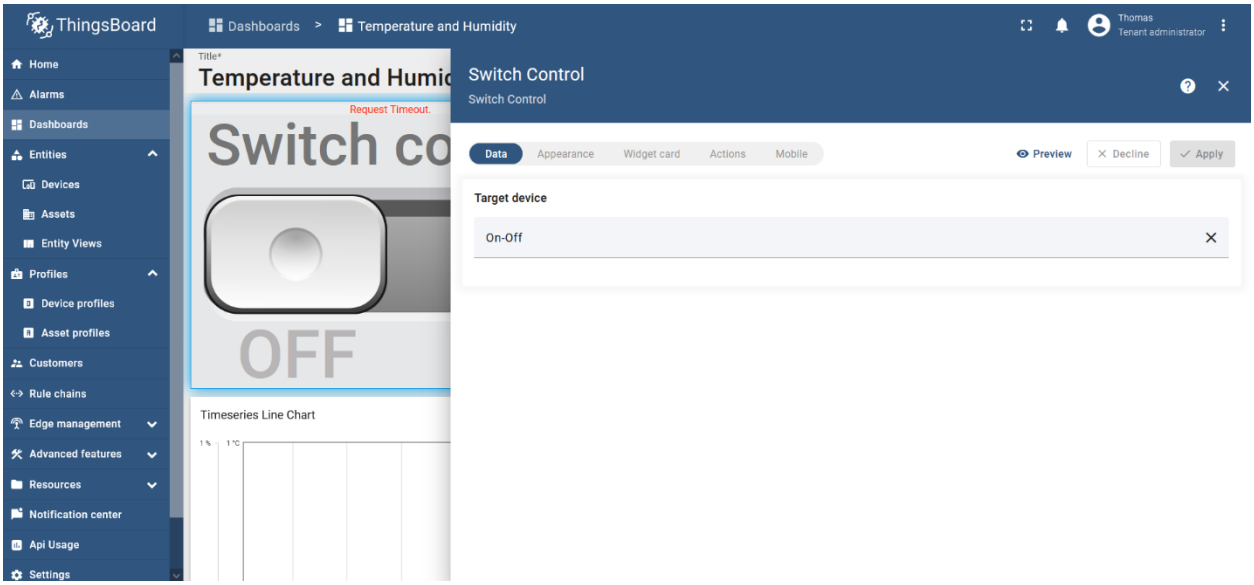


In the Appearance page you must change the Minimum, Maximum and Initial values to your needs. Important: if you are going to use multiple servos for your example you also must change the RPC set value method. Each knob must have different RPC set values so that you can distinguish the method during the coding phase. (see example code for robotic arm)

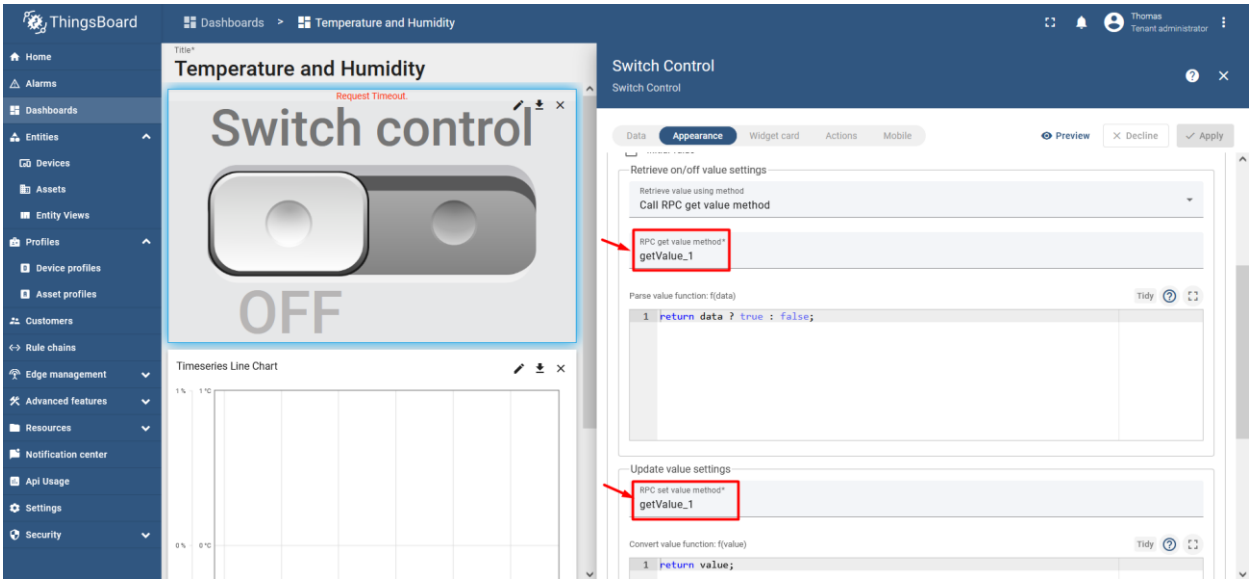


Set up the Switch Control

You can find the switch control widget in the edit mode in any dashboard under the control widgets bundle. Then by clicking on the pencil icon in the Data page you must add the Target device (in my case is called On-Off).

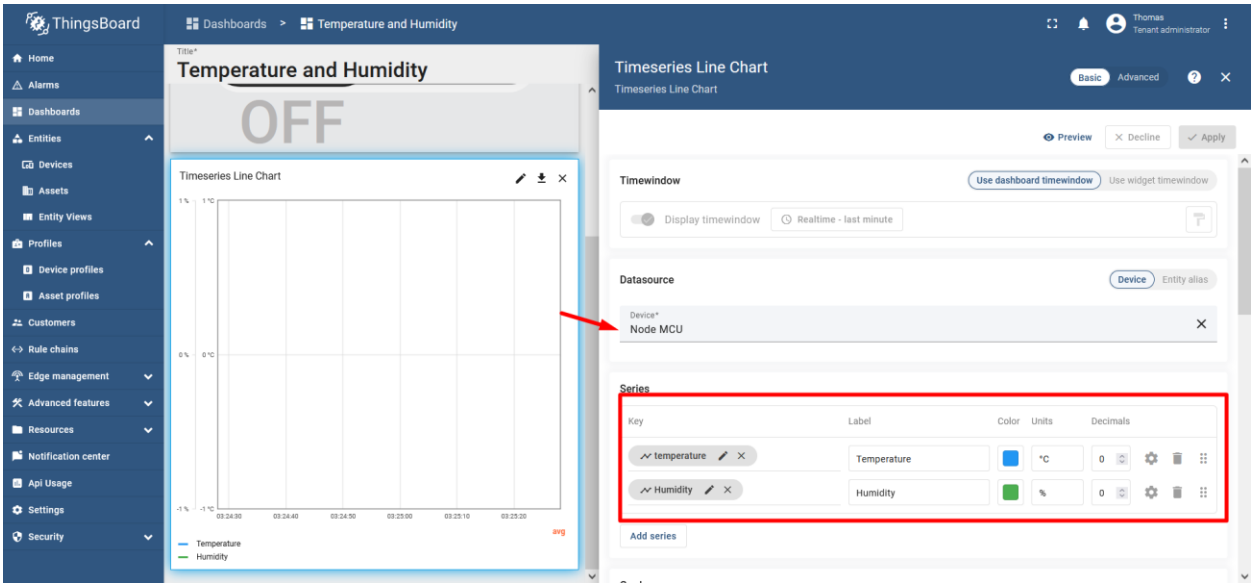


In the Appearance page change the RPC get and set value method so that they are the same command (in my case getValue_1). That way you can identify with the same RPC method if the toggle switch is on or off.



Set up the Timeless Line Chart

You can find the Timeless Line Chart widget in the edit mode in any dashboard under the Charts widgets bundle. Then by clicking on the pencil icon you change the Device (in my case NodeMCU) and then in the series section you change the Keys to your telemetry parameters. **NOTE:** In order to get the correct telemetry keys you first have to upload the code to your ESP8266 and start sending few measurements. Only then the keys can recognize the correct parameters.



Adoption from the Community

Another very crucial element is the adoption of the IoT Physics Lab from the educators' community. This will be achieved with a friendly and easy to read step by step instructions manual for each experiment with details and photos on how to set up the sensors and the digital interfaces. IoT physics labs are cost-effective and scalable. Traditional physics labs require expensive equipment, which can be a barrier to entry for many educational institutions. However, with IoT devices and sensors, schools can set up labs that are affordable and can be easily scaled to accommodate different class sizes and levels of complexity.

What is left to do?

The code can always be improved to perform faster and/or with lower power consumption. This way an experiment could become portable and easier to transfer on multiple locations if needed.

As an expansion to this project a development board could be made to connect one ESP8266 module and all the different sensors used for all the experiments. This board could be used during the testing phase of every experiment so that minor changes to the circuits could easily be made if needed.

Experience gained

During this project I learned many things about coding and about the limitations of the microcontroller. In particular:

- Furthered my skills coding in C++ and learned how to receive, create and manage text and data in JSON format, which is mandatory when anyone want to send or receive data from and to a server.
- Learned to use the ThingsBoard platform, where I created new devices and Dashboards that are used for controlling each experiment.
- Learned how the subscribe and publish commands are executed in the MQTT protocol.
- Learned how to use different sensors and find the relevant drivers and libraries
- Learned to provide detailed documentation of the produced design for the IOT sensors and actuators so that others either implement the design and produce the units, or further develop them.