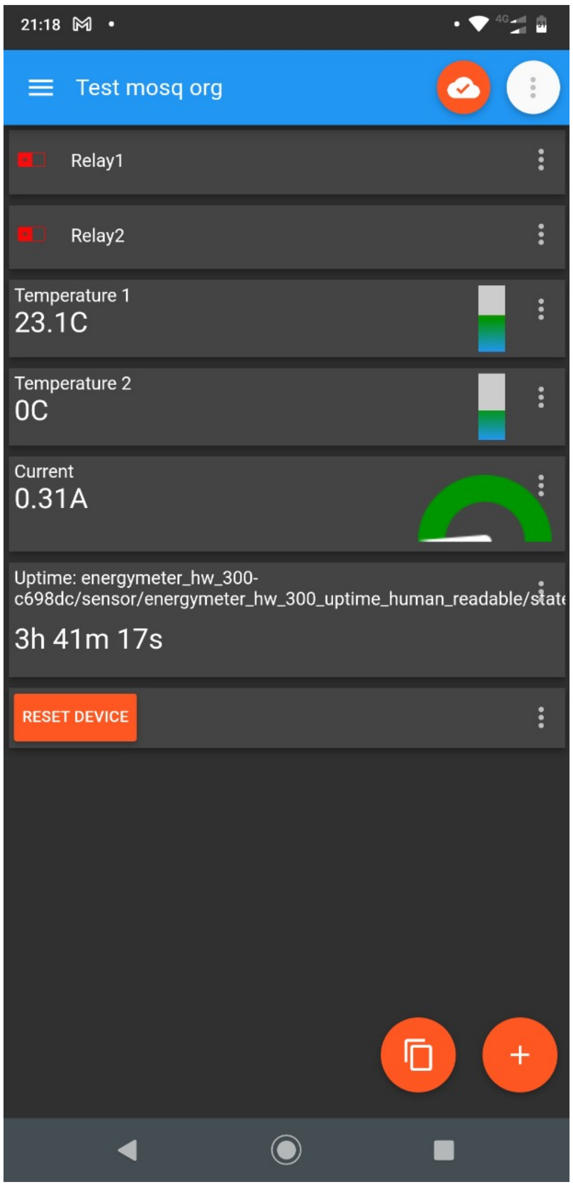
# IoT MQTT Panel for Android HOWTO

This article describes configuring Sowilo Box to be used with the IoT MQTT Panel application with pre-built firmware over a public MQTT server **test.mosquitto.org**.

**MQTT** is a standard IoT protocol for data exchange between devices. It is very popular, lightweight, and supports publish/subscribe messaging model. It is supported in almost every morden IoT product. You can find more information on [mosquitto.org](mosquitto.org), [hivemq.com](hivemq.com).



21:18

Test mosq org

Relay1

Relay2

Temperature 1
23.1C

Temperature 2
0C

Current
0.31A

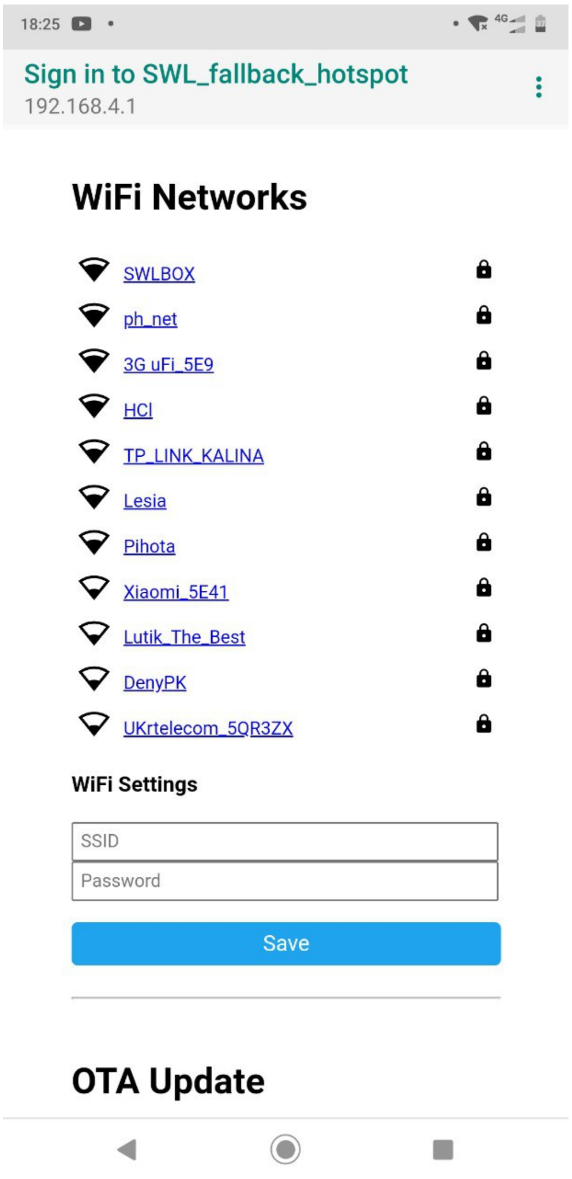
Uptime: energymeter_hw_300-c698dc/sensor/energymeter_hw_300_uptime_human_readable/state
3h 41m 17s

RESET DEVICE

Sowillo IoT board comes with a pre-flashed firmware that will connect to a public MQTT server (default: **test.mosquitto.org**). If you have different firmware, please download and flash the binary from <link> and flash as described in <link>. We provide several variants of pre-build firmware for common public MQTT servers. After getting some experience, you are advised to build your custom firmware for usage with your custom server. Please see <...> when you’re ready to dive in.

First, you need to connect your Sowillo IoT board to WiFi. This can be done via the captive portal in a few steps:

- Wait 1 minute after board reset and find `SWL_fallback_hotspot` WiFi network
- Connect to `SWL_fallback_hotspot` with password `development`
- Open SignIn page from Android messages and enter Your `WiFi_SSID_Name` and `Password`, then save settings.



WiFi Networks

SWLBOX
ph_net
3G uFi_5E9
HCl
TP_LINK_KALINA
Lesia
Pihota
Xiaomi_5E41
Lutik_The_Best
DenyPK
UKrtelecom_5QR3ZX

WiFi Settings

SSID
Password
Save

OTA Update

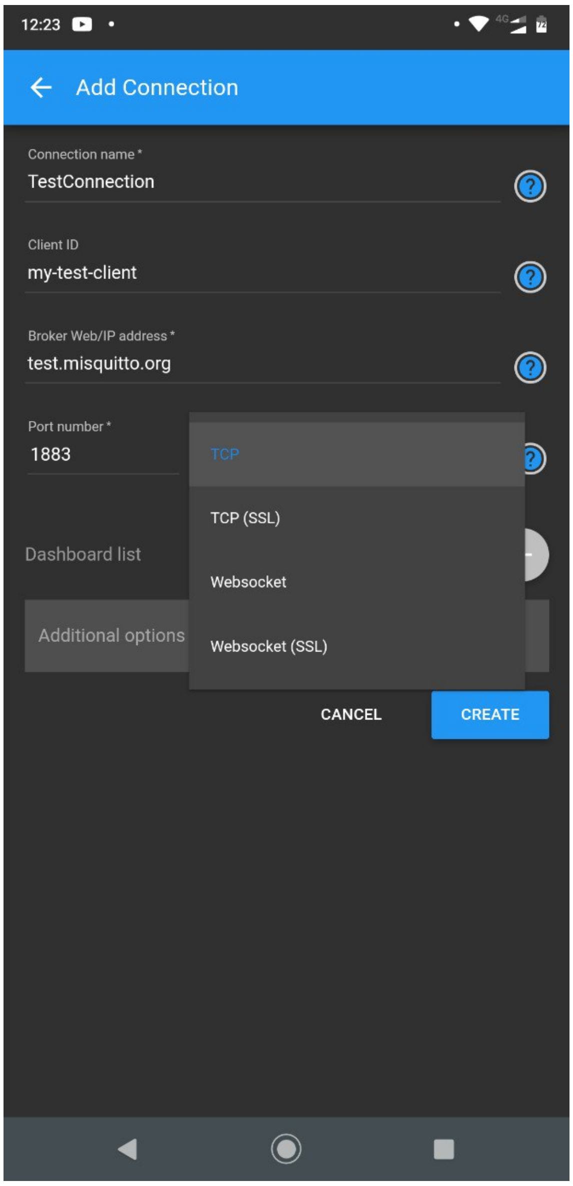
> **Pro hint:** You may want to connect to the MQTT server test.mosquitto.org and verify that the board sends data by monitoring it with an Android app.

Begin by installing the *IoT MQTT Panel* application to your Android phone:

We used this app as an example, you can use any MQTT tool you want, just search MQTT in Play Market/App Store.

Create a connection to public broker test.mosquitto.org and fill in the following data:

- Connection name: `MyTestConnection`
- Client ID: `MyTestClient`
- Broker Web/IP address: `test.mosquitto.org`
- Port number: `1883`
- Network protocol: `TCP`


12:23

Add Connection

Connection name *
TestConnection

Client ID
my-test-client

Broker Web/IP address *
test.misquitto.org

Port number *
1883

Add a new dashboard to the Dashboard list and name it TestBoard. Then press the `Create` button.

Each example FW uses its own unique six char prefix for state and command topics based on the last 3 bytes of the individual MAC address ESP32 board.

For example, a unique MAC address `00:11:22:AA:BB:CC` generates MQTT topics that look like:

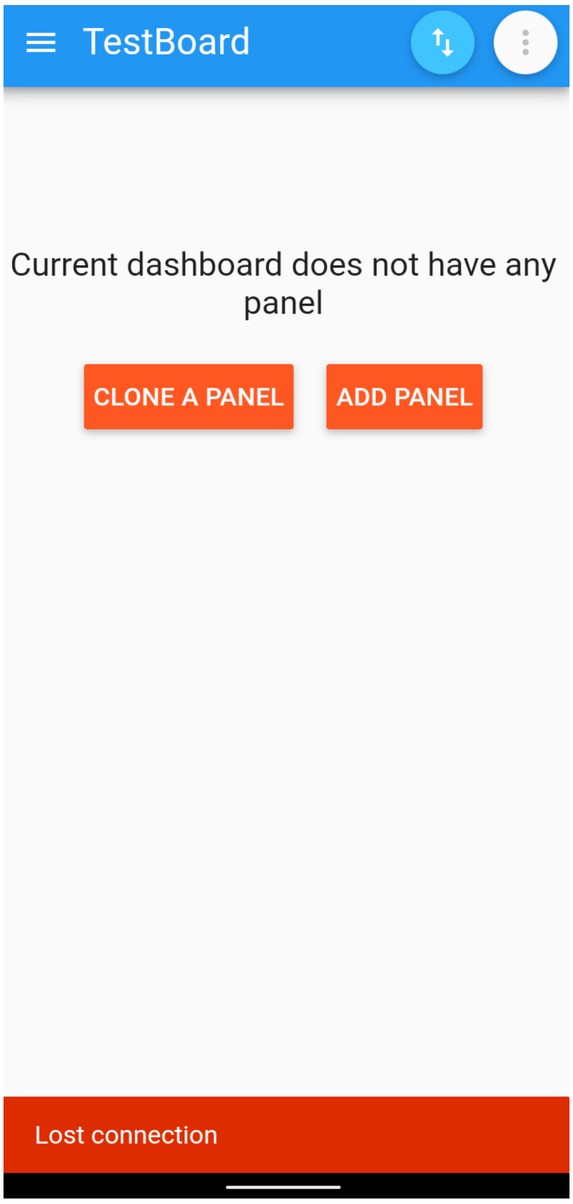
```
energymeter_hw_300-aabbcc/switch/energymeter_hw_300_reboot/
state
energymeter_hw_300-aabbcc/switch/energymeter_hw_300_reboot/
command
energymeter_hw_300-aabbcc/switch/energymeter_hw_300_relay1/
state
energymeter_hw_300-aabbcc/switch/energymeter_hw_300_relay1/
command
energymeter_hw_300-aabbcc/switch/energymeter_hw_300_relay2/
state
energymeter_hw_300-aabbcc/switch/energymeter_hw_300_relay2/
```

```
command
energymeter_hw_300-aabbcc/sensor/
energymeter_hw_300_temperature_1/state
energymeter_hw_300-aabbcc/sensor/
energymeter_hw_300_temperature_2/state
energymeter_hw_300-aabbcc/sensor/
energymeter_hw_300_uptime_human_readable/state
energymeter_hw_300-aabbcc/sensor/energymeter_hw_300_current/
state
```

> **Note:** You can find this information in the Captive Portal.

You may want to add panels with sensor metrics, and relay controls to the dashboard just created. You can do it as follows:

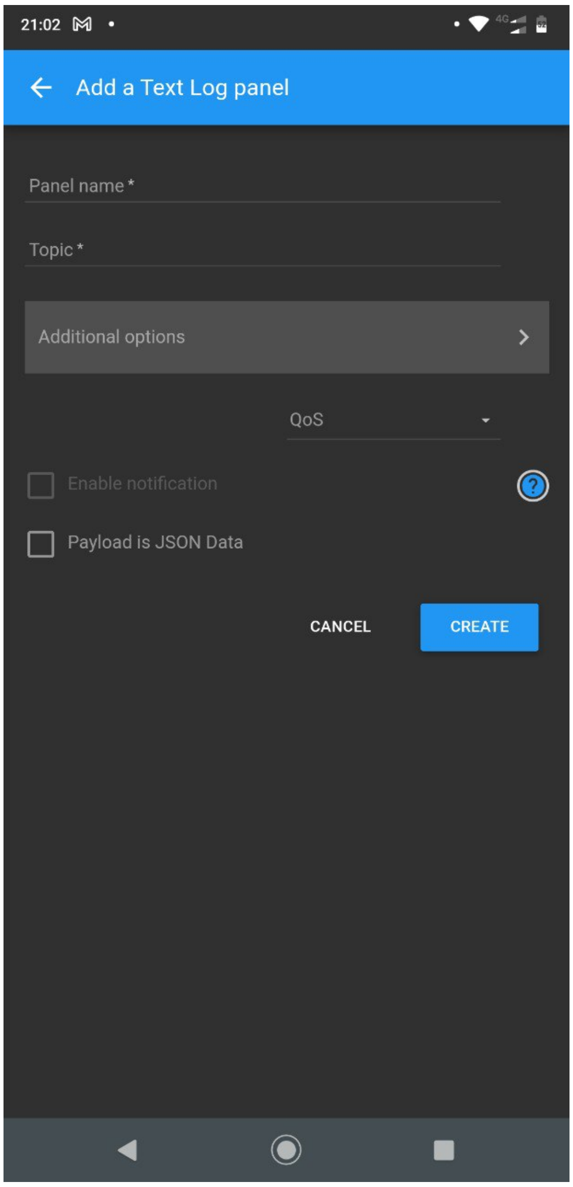
Open TestBoard and press the *AddPanel* button.



Current dashboard does not have any panel

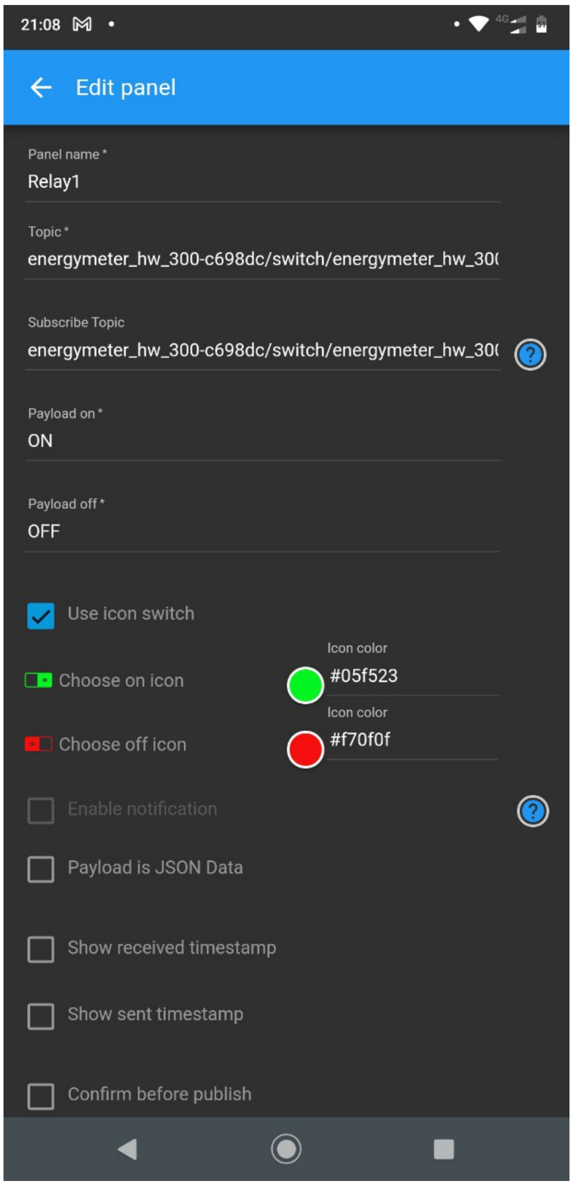
Lost connection

Select *Text Log* and fill the following fields: - Panel Name: `Uptime` - Topic: `energymeter_hw_300-aabbcc/sensor/energymeter_hw_300_uptime_human_readable/state`



In addition to the *TextLog*, there are *Switch, Vertical Meter, Gauge*, and many other panes for commonly used sensor and switch types.

Other panels could be added in a similar way:


21:08

Edit panel

Panel name *
Relay1

Topic *
energymeter_hw_300-c698dc/switch/energymeter_hw_30(

Subscribe Topic
energymeter_hw_300-c698dc/switch/energymeter_hw_30(

Payload on *
ON

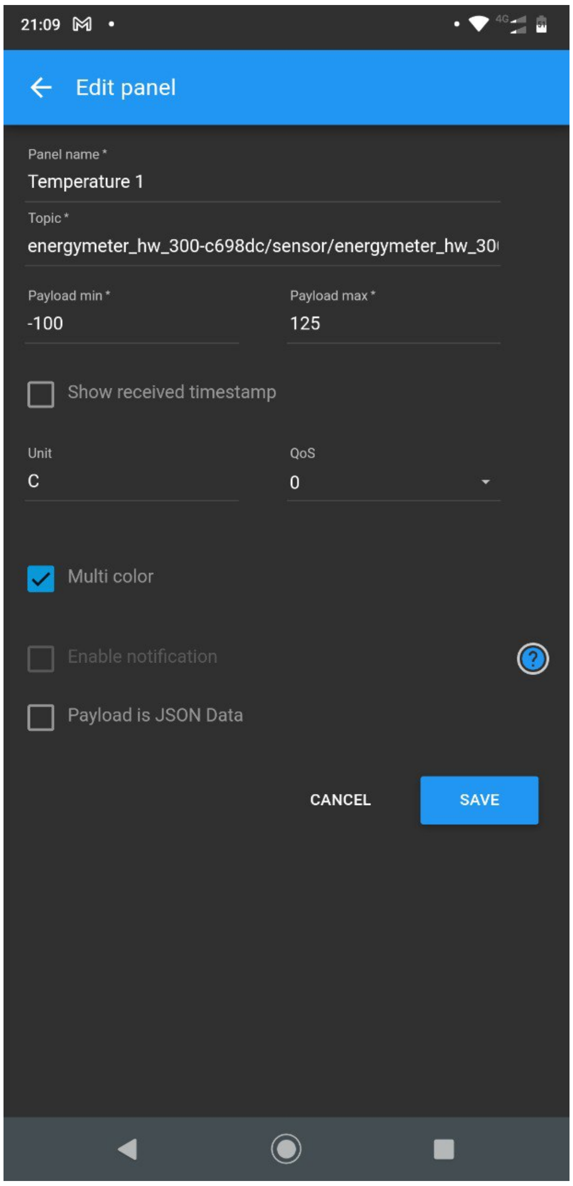
Payload off *
OFF

- [x] Use icon switch

Choose on icon — Icon color #05f523

Choose off icon — Icon color #f70f0f

- [ ] Enable notification
- [ ] Payload is JSON Data
- [ ] Show received timestamp
- [ ] Show sent timestamp
- [ ] Confirm before publish


21:09

Edit panel

Panel name *
Temperature 1

Topic *
energymeter_hw_300-c698dc/sensor/energymeter_hw_30(

Payload min *
-100

Payload max *
125

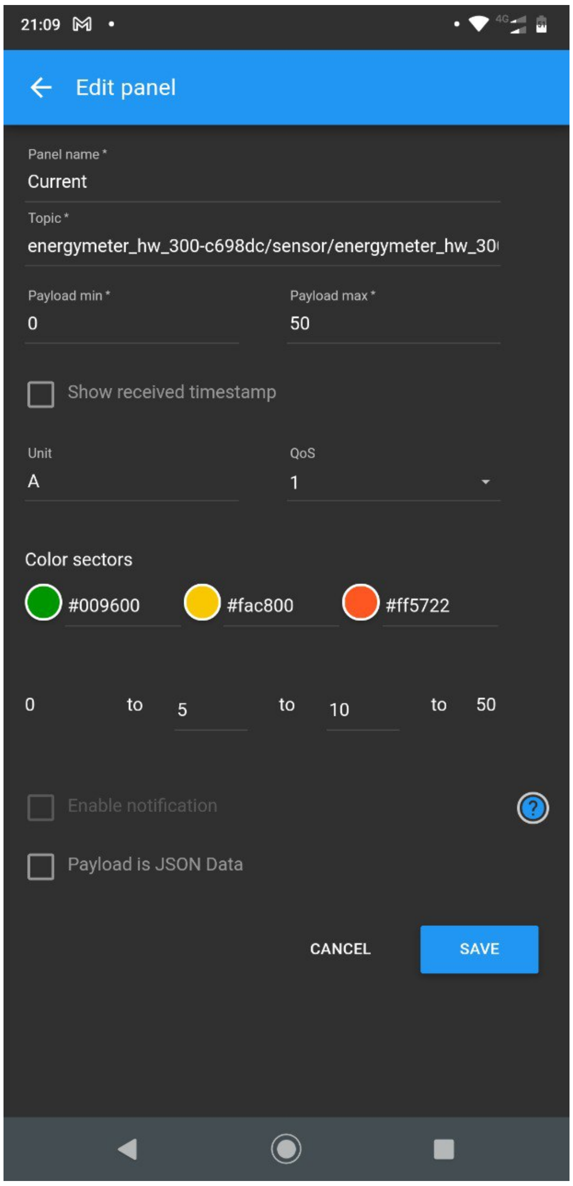
- [ ] Show received timestamp

Unit
C

QoS
0

- [x] Multi color
- [ ] Enable notification
- [ ] Payload is JSON Data

CANCEL SAVE


21:09

Edit panel

Panel name *
Current

Topic *
energymeter_hw_300-c698dc/sensor/energymeter_hw_30

Payload min *
0

Payload max *
50

Show received timestamp

Unit
A

QoS
1

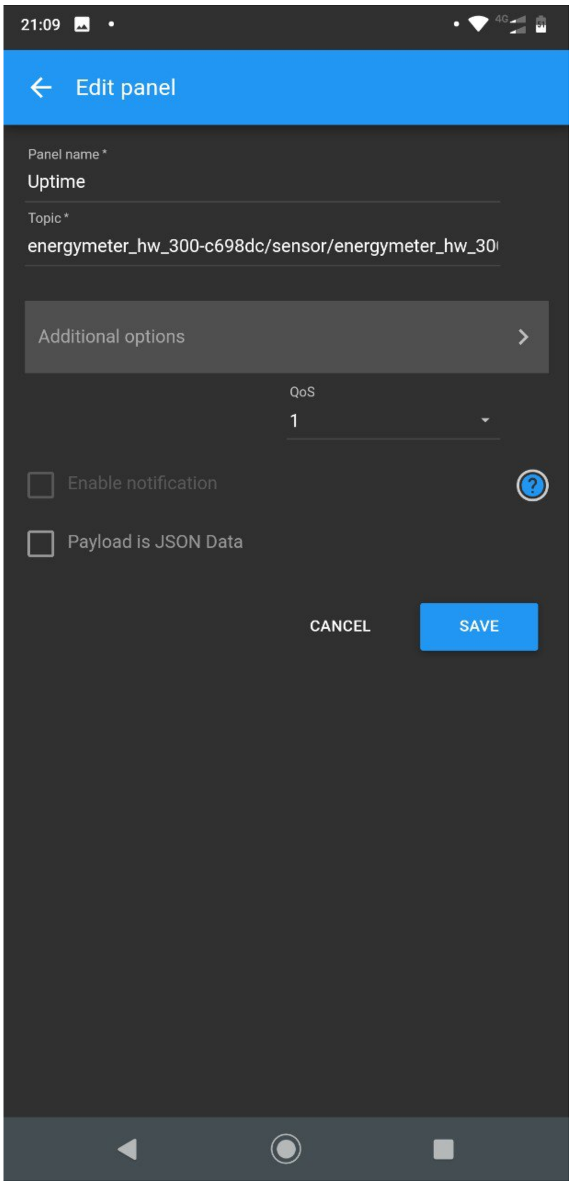
Color sectors

#009600 #fac800 #ff5722

0 to 5 to 10 to 50

Enable notification

Payload is JSON Data


21:09

Edit panel

Panel name *
Uptime

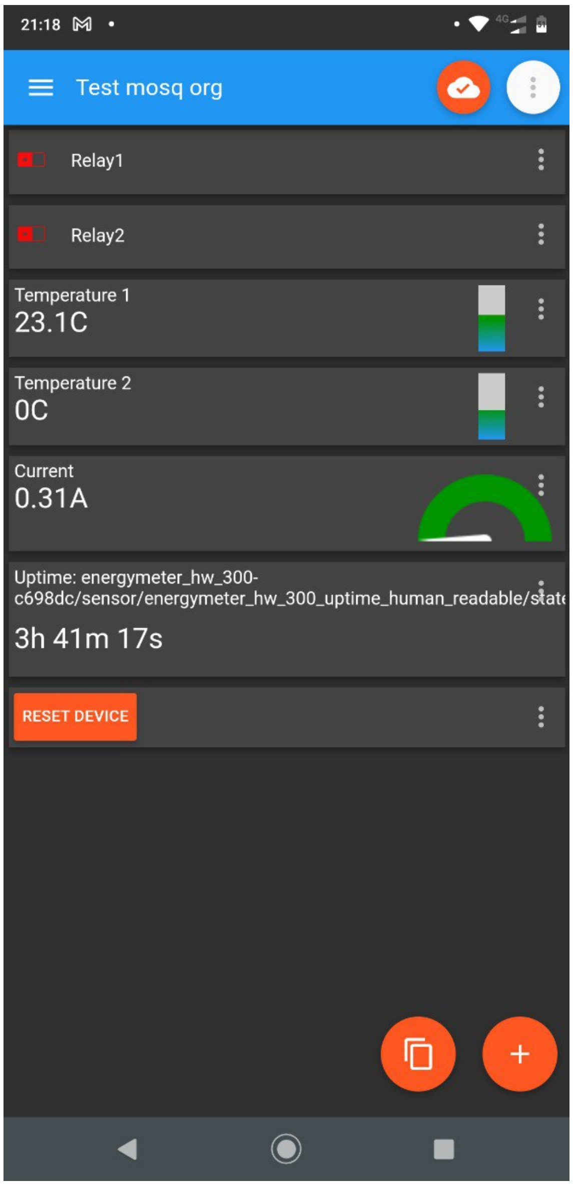
Topic *
energymeter_hw_300-c698dc/sensor/energymeter_hw_30

QoS
1

Enable notification

Payload is JSON Data

The final dashboard has the following look:



21:18

Test mosq org

Relay1

Relay2

Temperature 1
23.1C

Temperature 2
0C

Current
0.31A

Uptime: energymeter_hw_300-c698dc/sensor/energymeter_hw_300_uptime_human_readable/state
3h 41m 17s

RESET DEVICE