

Kurz und langform? Video?

Dokumentation zum Erstellen des Dashboards

Was wird benötigt?

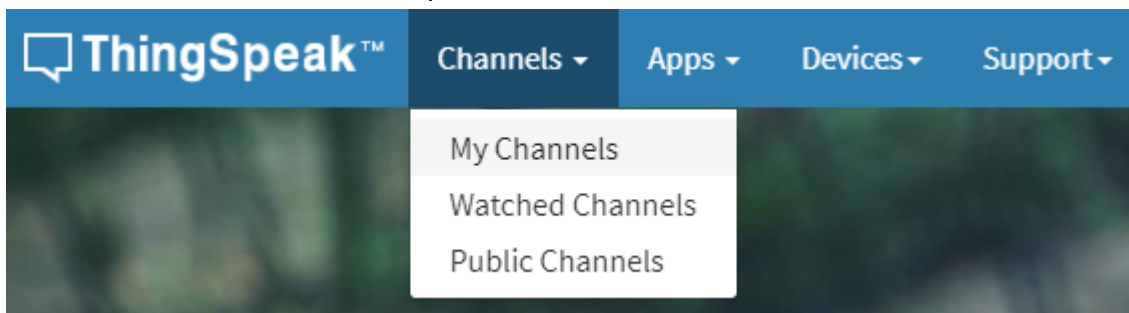
1 X MatLab-Lizenz [Kostenlose Testversion von MATLAB - MATLAB & Simulink \(mathworks.com\)](https://www.mathworks.com/products/matlab/trial.html)

Funktionsweise des Dashboards

Für die Datenspeicherung und -visualisierung bietet sich die cloudbasierte ThingSpeak-Plattform an. Dort können in einem Channel die Sensordaten gespeichert und in einem weiteren näher analysiert werden. Zuvor muss allerdings eine Integration zur Applikation im TheThingsNetwork erstellt werden, die für die Datenbereitstellung sorgt.

Integration der Daten von TheThingsNetwork zu ThingSpeak

1. Registrieren bei [IoT Analytics - ThingSpeak Internet of Things](https://thingspeak.com/)
2. Neuen Channel für das Speichern der Daten erstellen



3. Auf "New Channel" klicken und dann einen Namen eintragen sowie die ersten beiden Felder ausfüllen

New Channel

Name

Description

Field 1



Field 2



Field 3



Field 4



- Öffnen des Tabs API Keys. Die ChannelID (oben) und der Write API Key sind so sichtbar für den nächsten Schritt.

DatenLaermampel

Channel ID: 1488846

Author: mwa0000023508449

Access: Private

Private View

Public View

Channel Settings

Sharing

API Keys

Write API Key

Key

MPC3TVKKGWIKW8V

Generate New Write API Key

Read API Keys

Key

9UDZBOUNECDEI1SP

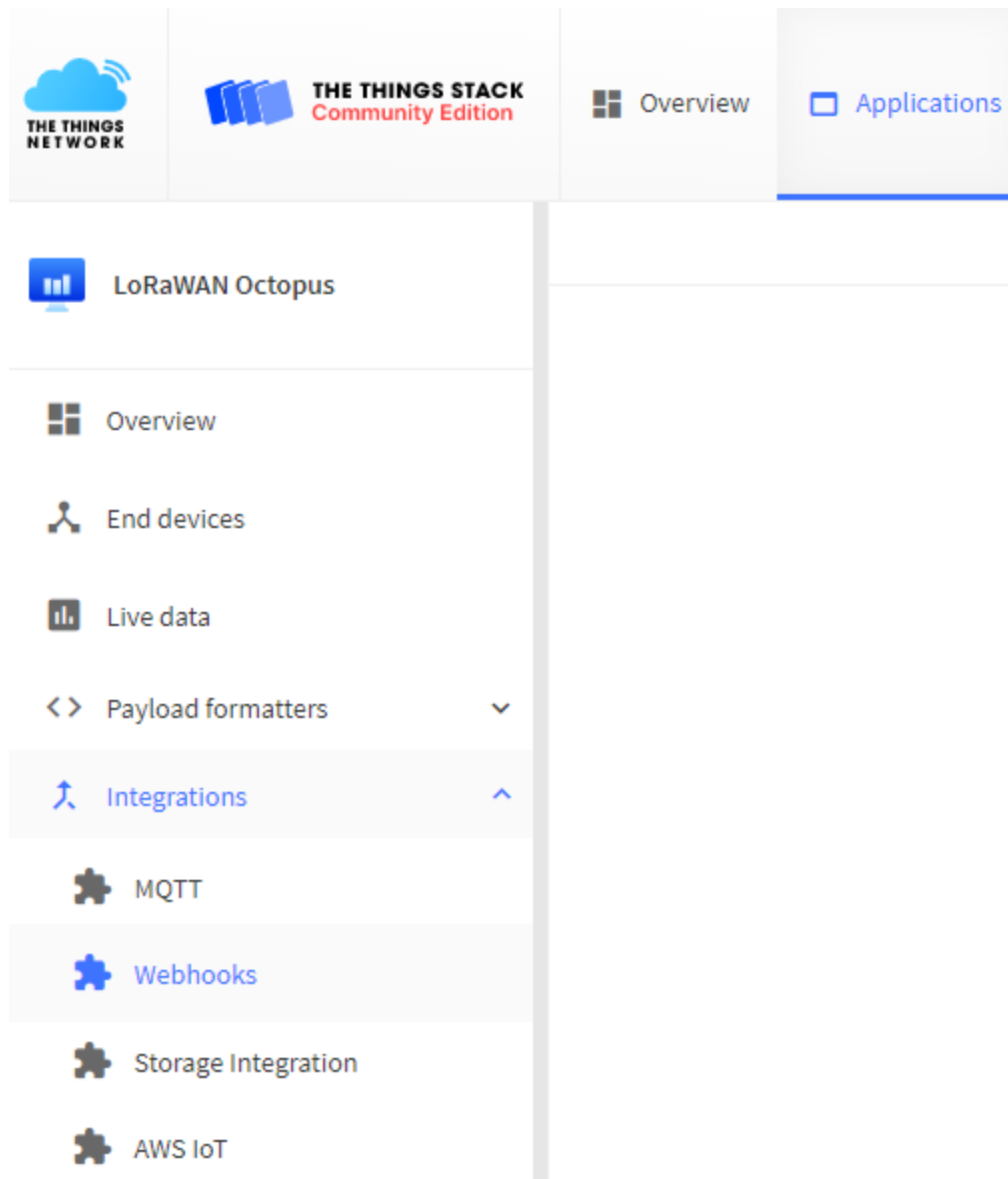
Note

Save Note

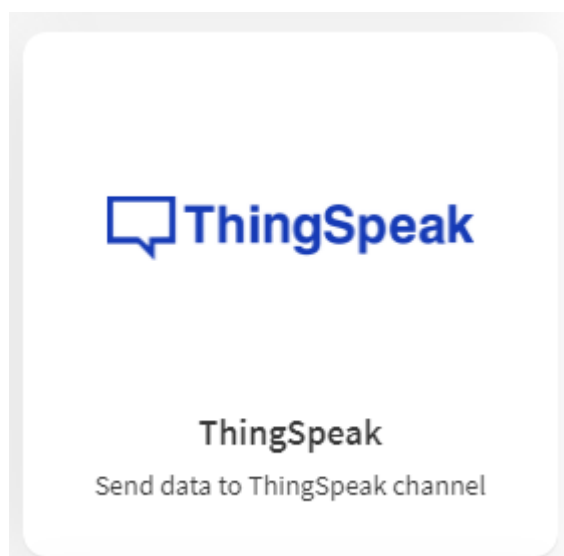
Delete API Key

Add New Read API Key

5. Öffnen der erstellten Applikation in TheThingsNetwork und navigieren zu Integrations - Webhooks (links)



6. "Add new Webhook" anklicken und ThingSpeak auswählen




7. Festlegen einer Webhook ID und übertragen der Channel ID und des API Keys aus der ThingSpeak Übersicht unter 4.

Applications > LoRaWAN Octopus > Webhooks > Add > ThingSpeak

Add custom webhook

Template information

 **ThingSpeak**

ThingSpeak

Send data to ThingSpeak channel

[About ThingSpeak](#) | [Documentation](#)

Template settings

Webhook ID *

Channel ID *



ThingSpeak Channel ID

API Key *

ThingSpeak Write API Key

Create thingspeak webhook

8. Öffnen der erstellten Applikation in TheThingsNetwork und navigieren zu End Devices (links)



Overview

Applications

Gateways

Organizations

LoRaWAN Octopus

Overview

End devices

Live data

Payload formatters

Integrations


Collaborators

Applications > LoRaWAN Octopus > End devices

End devices (3)

ID	Name
eui-70b3d57ed00446bd	ampel
eui-70b3d57ed00446e7	schranke
eui-70b3d57ed0044863	

9. Auswahl des Devices, das die Lautstärke misst und öffnen des Payload formatters Tab

**eui-70b3d57ed00446bd-ampel**
ID: eui-70b3d57ed00446bd-ampel

Last seen 3 days ago

↑ 48 ↓ 22

Overview

Live data

Messaging

Location

Payload formatters

Claiming

Uplink

Downlink

Setup

Formatter type *

JavaScript | ▼

Formatter parameter *

1

2

Save changes

10. Code aus GitHub einfügen und Klick auf Save Changes
Wandelt die gesendeten Daten in eine Zahl um und weist sie dem ersten Feld im
ThingSpeak Channel zu

```
function Decoder(b, port) {  
  
    var decoded = {};  
    decoded.port = port;  
    decoded.var1 = 0;  
  
    if (port === 1 ) {  
        var var1 = b[0];  
    }  
  
    return {
```

```

    field1: var1
  };
}

```

11. Zurückkehren zur Übersicht der End Devices und das Device zum Zählen der Personen öffnen

12. In den Payload Formatter Tab diesen Code einfügen

Wandelt die gesendeten Daten in eine Zahl um und weist sie dem zweiten Feld im ThingSpeak Channel zu

```

function Decoder(b, port) {

  var decoded = {};
  decoded.port = port;
  decoded.var1 = 0;

  if (port === 1 ) {
    var var1 = b[0]; //(b[0]<<16);
  }

  return {
    field2: var1
  };
}

```

13. Wenn die Applikation läuft, erscheinen die Daten im Zahlenformat zum einen in der TheThingsNetwork-Console und zum anderen im erstellten ThingSpeak-Channel.

Visualisierung und Analyse der Daten in ThingSpeak mit MatLap

Für die Gegenüberstellung der Anzahl an Personen in einem Raum und der gemessenen Lautstärke zu einem bestimmten Zeitpunkt bietet sich ein Liniendiagramm mit zwei Y-Achsen an (Lautstärke und Personenzahl).

1. Um das Dashboard MitschülerInnen zugänglich zu machen, sollte das Dashboard unter dem Tab "Public View" erstellt werden. Im ThinkSpeak-Channel zur Analyse der Daten muss dazu über den grünen Button "MATLAP Visualization" ein neues Feld dem Dashboard hinzugefügt werden.

HTW_Ampel_Analyse

Watch Tweet Gefällt mir 0 Weiterempfehlen

Channel ID: 1480622
Author: mwa0000023508449
Access: Public

Private View Public View Channel Settings Sharing API Keys Data Import / Export

+ Add Visualizations + Add Widgets Export recent data

MATLAB Analysis MATLAB Visualization

Channel Stats

Created: 24 days ago
Last entry: 6 days ago
Entries: 27

2. Eine neue Seite öffnet sich. Unter “Templates” soll “Custom (no starter code)” ausgewählt sein, da der nötige Code in GitHub bereitgestellt wird. Anschließend auf den Button “Create” klicken.

Apps / MATLAB Visualizations / New

Templates:

- ☒ Custom (no starter code)
- ☐ Create a filled area 2-D plot
- ☐ Create a 2-D line plot
- ☐ Create 2-D line plots with y-axes on both left and right side
- ☐ Create a correlated data plot
- ☐ Create a discrete sequence data plot

Examples: Sample code to visualize data

- ☐ Use a histogram to understand variation in data
 - ☐ Visualize directional data with compass plot
3. Im nächsten Schritt kann das Diagramm benannt werden als “Lautstärke und Personenzahl” und der Code kann von GitHub xxx LINK in das Feld für den MATLAB Code kopiert werden. Die Channel-IDs und möglicherweise die API Keys müssen wie in den Inline-Kommentaren zu erkennen ist, angepasst werden.

Name

Lautstärke und Personenzahl

MATLAB Code

```
1 % Lautstärke - 1. Channel ID von dem Daten zur Lautstärke gelesen werden
2 readChannelID1 = 1476009;
3 fieldID1 = 1;
4
5 % 1. Channel Read API Key
6 % Falls der Channel private ist, muss der read API Key zwischen '' eingesetzt werden
7 readAPIKey1 = 'SP0E5VY6DR085RSH';
8
9 % Personenzahl - 2. Channel ID von dem Daten zur Personenanzahl gelesen werden
10 readChannelID2 = 1480627;
11 fieldID2 = 1;
12
13 % 2. Channel Read API Key
14 % Falls der Channel private ist, muss der read API Key zwischen '' eingesetzt werden
15 readAPIKey2 = 'VXAUFZIO9MMPTBU0';
16
17 %% Lesen der Daten %%
18 % Lesen der Personenzahl
19 [Lautstaerke, time1] = thingSpeakRead(readChannelID1, 'Field', fieldID1, 'NumPoints', 10, 'Read');
20
21 % Lesen der Lautstärke
22 [Personenzahl, time2] = thingSpeakRead(readChannelID2, 'Field', fieldID2, 'NumPoints', 10, 'Read');
23
24 %% Visualisierung der Daten %%
25 xlabel('Uhrzeit');
26
27 yyaxis left;
28 plot(time1, Lautstaerke)
29 ylabel('Lautstärke');
30
31 yyaxis right;
32 plot(time2, Personenzahl);
33 ylabel('Personenzahl');
```

Save and Run

Save

4. Nach dem Anpassen des Codes kann über den Button “Save and Run” der Code gespeichert und ausgeführt werden. Das Ausgabefeld “MATLAB Plot Output” zeigt bei korrektem Code das gewünschte Diagramm. Falls dies nicht der Fall sein sollte, ist eine Überprüfung der Anpassungen zu empfehlen und die gezeigte Fehlermeldung zu beachten.

Als zusätzliche Analyse der Lautstärke und Personenzahl können die jeweiligen Durchschnitte mittels MATLAB berechnet werden.

1. Über den grünen Button “MATLAB Analysis” kann dies umgesetzt werden.

ThingSpeak™

Channels ▾

Apps ▾

Devices ▾

Support ▾

Commercial Use

How to Buy

MK

HTW_Ampel_Analyse

Watch

Tweet

Gefällt mir

Weiterempfehlen

Channel ID: 1480622

Author: mwa0000023508449

Access: Public

Private View

Public View

Channel Settings

Sharing

API Keys

Data Import / Export

Add Visualizations

Add Widgets

Export recent data

MATLAB Analysis

MATLAB Visualization

Channel Stats

Created: 24 days ago

Last entry: 6 days ago

Entries: 27

2. Eine neue Seite öffnet sich. Unter “Templates” soll “Custom (no starter code)” ausgewählt sein, da der nötige Code in GitHub bereitgestellt wird. Anschließend auf den Button “Create” klicken.

ThingSpeak™

Channels ▾

Apps ▾

Devices ▾

Support ▾

Apps / MATLAB Analysis / New

Templates:

- ☒ Custom (no starter code)
- ☐ Get data from a private channel
- ☐ Get data from a public channel
- ☐ Get data from a webpage

Examples: Sample code to analyze and transform data

- ☐ Calculate and display average humidity
 - ☐ Calculate wind chill and update channel
3. Im folgenden Fenster kann nun ein Name festgelegt und der Code aus GitHub xxxLink in den Bereich für den MATLAB Code eingefügt werden. Die berechneten Durchschnitte werden dabei in den Channel für die Analyse geschrieben. Die Inline-Kommentare innerhalb des Codes beschreiben die einzelnen Abschnitte und geben an, wo möglicherweise Parameter angepasst werden müssen (z.B. der für die Durchschnittsberechnung betrachtete Zeitrahmen).

[Apps](#) / [MATLAB Analysis](#) / [Durchschnitte berechnen](#) / [Edit](#)

Name

Durchschnitte berechnen

MATLAB Code

```
1 % Liest die Personenzahl aus dem ThingSpeak channel und schreibt den Durchschnitt der letzten
2 % 45 Minuten in einen anderen ThingSpeak channel.
3
4 %% Der Read-Channel enthält die gemessene Lautstärke. %%
5 % Channel ID von dem Daten gelesen werden
6 readChannelID1 = 1488846;
7 % Personenanzahl ID
```

4. Mit "Save and Run" wird der Code ausgeführt und im Output-Feld können die Ergebnisse auf ihre Richtigkeit kontrolliert werden. Rote Meldungen deuten dabei auf einen Fehler hin. Die Beschreibungen helfen bei dem Error-Handling. Im unteren Beispiel fehlt eine Variable oder sie ist nicht korrekt.

Save and Run

Save

Output

Durchschnittliche Lautst~~är~~~~k~~e =


NaN

Unrecognized function or variable 'readAPIKey2'.


5. Um diese Analyse nicht nur einmalig durchzuführen und die aktuelle Berechnung für das Dashboard bereitstellen zu können, bietet ThingSpeak Reacts an. Diese können einen MATLAB-Code ausführen wenn ein vordefiniertes Ereignis eintritt. Für die Anwendung ist es sinnvoll, wenn neue Messwerte im Channel für die Daten geladen werden den Durchschnitt neu zu berechnen.
Die Erstellung eines Reacts erfolgt mit einem Klick darauf im unteren Teil der Seite.

Schedule Actions

☒ Notify me via email if this MATLAB Analysis fails when triggered by TimeControl or React.

 TimeControl

Schedule your saved code to run at a specified time

 React

Name	Test Frequency
------	----------------

- Der Name des Reacts kann frei gewählt werden. Der “Condition Type” ist “Numeric”, da es sich um eine Eingabe von Zahlen handelt und die Prüfung muss auf “On Data Insertion” stehen, sodass bei neu in den Channel geschriebenen Daten immer geprüft wird, ob das React ausgeführt werden soll. Der Channel und das zu testende Feld sind je nach Setup einzufügen. Da keine Lautstärke oder Personenzahl in dem betrachteten Szenario von 1000 erreicht werden kann, wird das React bei jeder neuen Eingabe von Daten ausgeführt und ermöglicht die Bereitstellung aktueller Daten. Mit dem Klick auf “Save React” wird alles abgespeichert.

Apps / React / UpdateLautstärke&Personenzahl / Edit

React Name

UpdateLautstärke&Personenzahl

Condition Type

Numeric ▾

Test Frequency

On Data Insertion ▾

Condition

If channel

HTW_Ampel_Lautstaerke (1476009) ▾

field

1 (Lautstaerke) ▾

is not equal to ▾

1000

Action

MATLAB Analysis ▾

Code to execute

Durchschnitte berechnen ▾

Options

- ☐ Run action only the first time the condition is met
☒ Run action each time condition is met

Save React

- Jetzt können die berechneten Durchschnitte über das Feld “Add Widgets” hinzugefügt werden.

HTW_Ampel_Analyse

Channel ID: 1480622

Author: mwa0000023508449

Access: Public

Private View

Public View

Channel Settings

Sharing

API Keys

Data Import / Export

+ Add Visualizations

+ Add Widgets

Export recent data

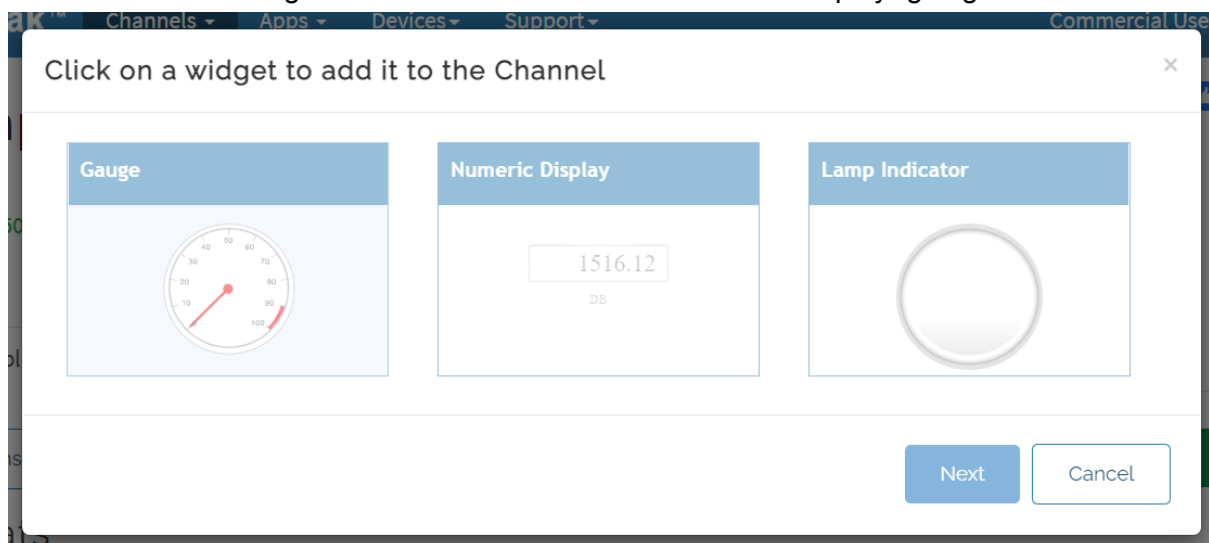
Channel Stats

Created: 26 days ago

Last entry: 33 minutes ago

Entries: 28

8. Zur Darstellung eines Durchschnitts ist das “Numeric Display” geeignet.



9. Im nächsten Fenster wird der Name festgelegt, das Feld je nach Setup, das Intervall für das Update und die Einheit. Das Beispiel ist für Lautstärke ausgefüllt und da nur ganze Zahlen von dem Sensor gesendet werden und keine genaueren Werte nötig sind, ist als Datentyp “Integer” auszuwählen. Mit Klick auf “Create” wird das Widget erstellt und erscheint im Dashboard.

Channels ▾ Apps ▾ Devices ▾ Support ▾ Commercial Use

Configure widget parameters
? x

Name

Field
 ▾

Update Interval
second(s)

Units

Data Type
☒ Integer
☐ Decimal
 ▾ (# of places)

Create
Cancel

10. Für den zweiten Durchschnittswert muss der Prozess ab 7. noch einmal durchlaufen werden.

11. Um die Rohdaten detaillierter auswerten zu können, besteht die Möglichkeit mittels des "Data Import/ Export"-Tabs einen

ThingSpeak™
Channels ▾ Apps ▾ Devices ▾ Support ▾

DatenLaermampel

Channel ID: **1488846**
Author: [mwa0000023508449](#)
Access: Private

Private View
Public View
Channel Settings
Sharing
API Keys
Data Import / Export

+ Add Visualizations

+ Add Widgets

📄 Export recent data

Add on - Temperatur Wetter von anderem Channel: **143789 Field 5**

ToDo:

Excel-Download der Daten -

Git Hub -

Fragen Mevre:

Warum neue Applikation? -> gleich von Jonas mit anderem End device

Wie oft Daten gesendet? Zeitabstände: 15 bis 30 Sekunden

GitHub setup und alle Codes-Schnipsel

Präsentation/ Plakat: Ich öffne und leite durch (Präzi)

0. Hinführung

1. Folie Jonas
2. Mevre
3. Martin Dashboard

Video und live rübersprechen in Präsi

PDF

10 Minuten all data is saved. adjust time in beginning -> Presentation -
schönes Dashboard für Präsentation / Screenshots mit Data-Upload?

Videos vs. Screenshots?

Aufbau

- 1.
- 2.
- 3.

ThingSpeak Account erstellen

Registrieren bei [ThingSpeak](#)

3 Channel aufsetzen

Verbindung von ThingSpeak zu den durch die Devices an das TheThingsNetwork
gesendeten Daten

1. Lautstärke
2. Personenzahl

Analyse

Visualisierung

analysis add to channel

2. Erstellen von 3 Channels im Channels-Tab

- a) Name: Lärmampel_Personenanzahl
Field 1: Personenanzahl
 - speichert die vom TTN gesendete Personenanzahl
- b) Name: Lärmampel_Lautstärke
Field 1: Lautstaerke
 - speichert die vom TTN gesendete Lautstärke
- c) Name: Lärmampel_Analyse
Field 1: DurchschnittlichePersonenzahl
Field 2: DurchschnittlicheLautstaerke
 - speichert die Durchschnittswerte und kommt zur Visualisierung zum Einsatz

3. Set-up TTN/ run TTN - webhooks -> ThingSpeak

API-Keys des Channels

Nach Datensenden/ Check ->

4. Dashboard: Lärmampel_Analyse

- In den Code-Vorlagen müssen die Channel IDs und API-Keys angepasst werden, sodass die Daten aus den Channels getestet werden können

combine both!

- a) Kennzahl: Durchschnittliche Personenanzahl der letzten 45 Minuten
MATLAB Analysis
Name: Durchschnittliche Personenanzahl
git Hub Code - Parameter im Code
save and run
Update: React (unten)
React Name: Personenanzahl_Update
Condition Type: String
Test Frequency: On Data Insertion
Condition: If Channel - Lärmampel_Personenanzahl
field - 1 (Personenanzahl)
is not equal to
100000
Action: MATLAB Analysis
Code to execute - Durchschnittliche Personenanzahl
- b) Kennzahl: Durchschnittliche Lautstärke der letzten 45 Minuten
MATLAB Analysis
Name: Durchschnittliche Lautstärke
git Hub Code
save and run
Update: React (unten)
React Name: lautstaerke_Update
Condition Type: String

Test Frequency: On Data Insertion

Condition: If Channel - Lärmampel_Lautstärke
field - 1 (Lautstaerke)
is not equal to
100000

Action: MATLAB Analysis

Code to execute - Durchschnittliche Lautstärke

- c) Liniendiagramm zur Gegenüberstellung der Personenzahl und Lautstärke
MatLab Visualization
Name: Personenzahl und Lautstärke
code GitHub
Save and Run - check ob läuft

Channels:

HTW_Lernampel - Lautstärke

HTW_Lernampel_Personen - Personenzahl

HTW_Lernampel_Analyse - Dashboard

MatLab Analysis:

Durchschnittliche Personenzahl und Lautstärke + React

MatLab Visualization:

Personenzahl und Lautstärke

Widget:

Durchschnittliche lautstärke und Personenzahl + senden (letzten 45 Minuten)

-> all public view

-> Git Hub?

Video:

1. Channel aus Übersicht
2. Data Channel für Messwerte
3. Analyse Channel (MATLAB Analysis + React)
4. Dashboard Kennzahlen
5. Grafik MATLAB
6. Zusatz aus anderem Channel

