

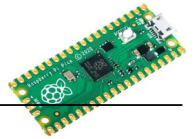
SchrittWeise* PicoBello-08

PB-08 SchrittWeise Node-RED Nodes auf RasPi V02.odt

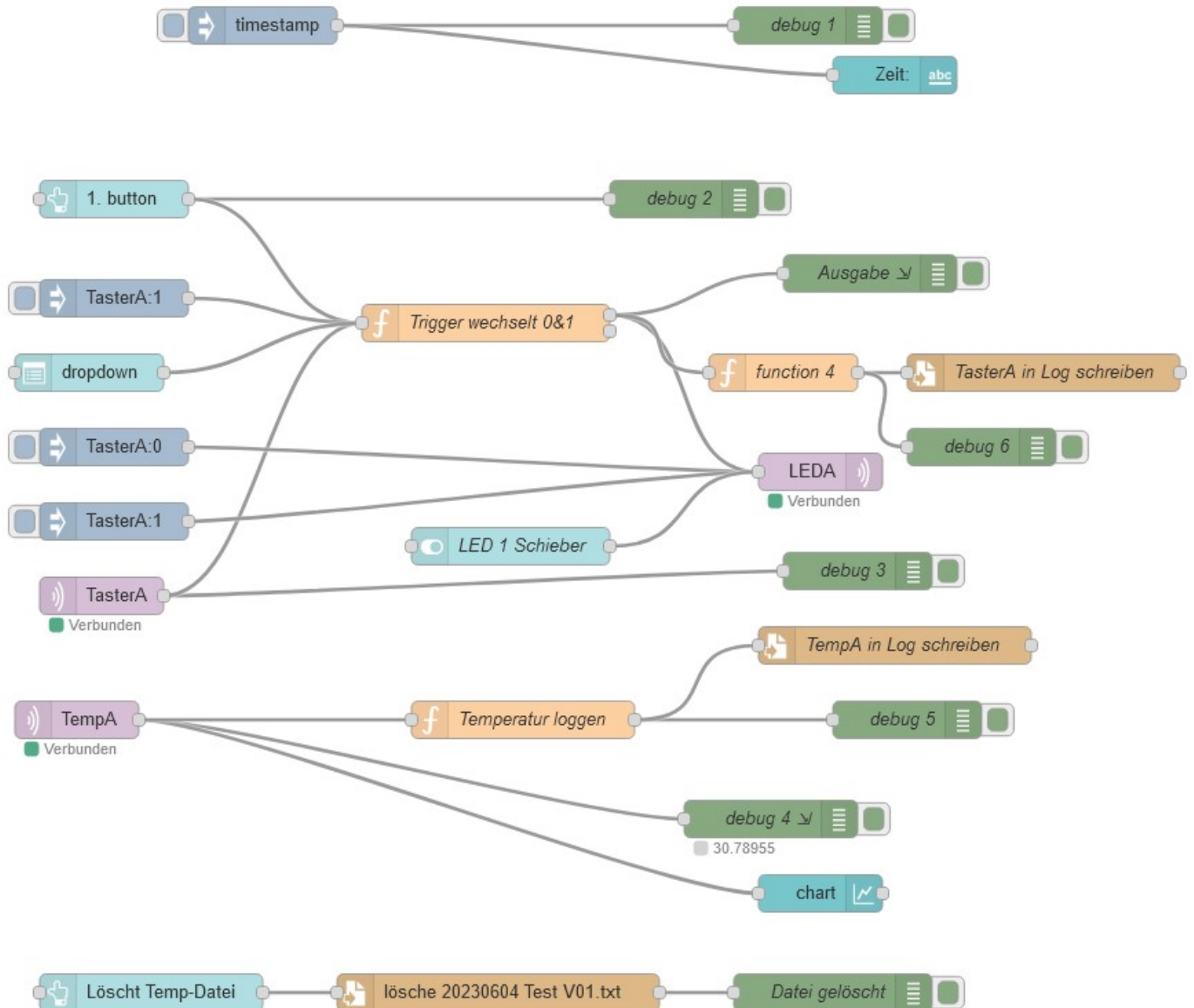
Inhaltsverzeichnis

1)	Gesamtstruktur (aus PicoBello-7).....	2
2)	Working with messages.....	2
3)	Inject.....	3
4)	button (Dashboard).....	5
4.1	1. button.....	5
4.2	Löscht Temp-Datei.....	6
5)	dropdown (Dashboard).....	6
6)	switch (Dashboard).....	7
7)	function.....	7
7.1	Trigger wechselt 0&1.....	7
8)	chart (Dashboard).....	8
8.1	Temperatur loggen.....	9
9)	Write file.....	10
9.1	Datei löschen.....	10
9.2	an Datei anhängen.....	11
10)	text (Dashboard).....	11
11)	mqtt in.....	12
11.1	TasterA.....	12
11.2	TempA.....	12
12)	mqtt out.....	13
12.1	LEDA.....	13

(* Schritt für Schritt ein wenig weiser ...)



1) Gesamtstruktur (aus PicoBello-7)



<https://nodered.org/docs/user-guide/messages>

2) Working with messages

A Node-RED flow works by passing messages between nodes. The messages are simple JavaScript objects that can have any set of properties.

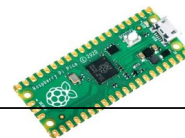
Messages usually have a `payload` property - this is the default property that most nodes will work with.

Node-RED also adds a property called `_msgid` - this is an identifier for the message which can be used to trace its

Ein Node-RED-Fluss funktioniert durch das Weiterleiten von Nachrichten zwischen Knoten. Die Nachrichten sind einfache JavaScript-Objekte, die eine beliebige Anzahl von Eigenschaften haben können.

Nachrichten haben normalerweise eine `Payload`-Eigenschaft - dies ist die Standard-Eigenschaft, mit der die meisten Knoten arbeiten werden.

Node-RED fügt außerdem eine Eigenschaft namens `_msgid` hinzu - dies ist eine Kennung für die Nachricht, die dazu



progress through a flow.

verwendet werden kann, ihren Fortschritt durch einen Fluss nachzuvollziehen.

The value of a property can be any valid JavaScript type, such as: Der Wert einer Eigenschaft kann jeden gültigen JavaScript-Typ haben, wie zum Beispiel:

- Boolean - true, false
- Number - eg 0, 123.4
- String - "hello"
- Array - [1, 2, 3, 4]
- Object - { "a": 1, "b": 2 }
- Null

3) Inject

Node 'inject' bearbeiten

Löschen
Abbrechen
Fertig

Eigenschaften

Name
Name

msg.payload
=

timestamp

msg.topic
=

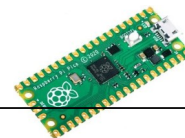
flow.
global.
string
number
boolean
JSON
buffer
timestamp
JSONata
Umgebungsvariable
msg.

+ hinzufügen
inject now

☐ Einmal injizieren nach 0.1 Sekunden, danach

Wiederholung
Keine

Aktiviert



Node 'inject' bearbeiten

Löschen Abbrechen **Fertig**

Eigenschaften

Name

msg. payload = timestamp

msg. topic = a_z

Node 'inject' bearbeiten

Löschen Abbrechen **Fertig**

Eigenschaften

Name

msg. payload = 0₉ 1

msg. topic = a_z TasterA

Node 'inject' bearbeiten

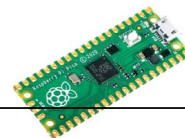
Löschen Abbrechen **Fertig**

Eigenschaften

Name

msg. payload = 0₉ 0

msg. topic = a_z TasterA



Node 'inject' bearbeiten

Löschen
Abbrechen
Fertig

Eigenschaften

Name

msg. payload

=

×

msg. topic

=

×

4) button (Dashboard)

4.1 1. button

Node 'button' bearbeiten

Löschen
Abbrechen
Fertig

Eigenschaften

Gruppe

[PB-07-Dashboard] Dashboard Test

Größe

Auto

Icon

Optionales Icon

Beschriftung

1. button

Tooltipp

Optionaler Tooltipp

Farbe

Optionale Text/Icon-Farbe

Hintergrund

Optionale Hintergrundfarbe

Sende beim Klicken:

Payload

Topic

Emuliere einen Klick bei einer eingehenden Nachricht:

☐

Klasse

Optionale(r) CSS-Klassenname(n) für das Widge

Name

Debug

Alle Nodes/Flows
all

23.1.2024, 13:59:39 node: debug 5
TempA : msg.payload : Object
{ Zeit: "2024.01.23 13:59:36.180",
Temperatur: "30.78955" }
23.1.2024, 13:59:47 node: debug 4
TempA : msg.payload : string[8]
"30.78955"
23.1.2024, 13:59:54 node: debug 5
TempA : msg.payload : Object
{ Zeit: "2024.01.23 13:59:43.861",
Temperatur: "30.78955" }
23.1.2024, 13:59:54 node: debug 4
TempA : msg.payload : string[8]
"30.78955"
23.1.2024, 13:59:54 node: debug 5
TempA : msg.payload : Object
{ Zeit: "2024.01.23 13:59:51.538",
Temperatur: "30.78955" }
23.1.2024, 14:00:02 node: debug 4
TempA : msg.payload : string[8]
"30.78955"
23.1.2024, 14:00:02 node: debug 5
TempA : msg.payload : Object
{ Zeit: "2024.01.23 13:59:59.226",
Temperatur: "30.78955" }
23.1.2024, 14:00:13 node: debug 4
TempA : msg.payload : string[8]
"30.78955"
23.1.2024, 14:00:13 node: debug 5

PB-07-Dashboard

Dashboard Test

1. BUTTON

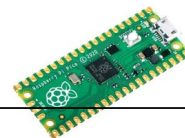
Select option

LED 1 Schieber

chart

LÖSCHT TEMP-DATEI

Zeit: 1706011107480



4.2 Löscht Temp-Datei

The screenshot shows the Node-RED interface with the 'button' node configuration panel on the left, the debug console in the center, and the 'PB-07-Dashboard' on the right.

Node 'button' bearbeiten:

- Buttons: Löschen, Abbrechen, Fertig
- Eigenschaften:
 - Gruppe: [PB-07-Dashboard] Dashboard Test
 - Größe: Auto
 - Icon: Optionales Icon
 - Beschriftung: Löscht Temp-Datei
 - Tooltip: Optionaler Tooltip
 - Farbe: Optionale Text/Icon-Farbe
 - Hintergrund: Optionale Hintergrundfarbe
 - Sende beim Klicken: ☒
 - Payload: a_2 Lösche Datei
 - Topic: msg.topic
 - Emuliere einen Klick bei einer eingehenden Nachricht: ☐
 - Klasse: Optionale(r) CSS-Klassenname(n) für das Widge
 - Name: Name

Debug Console:

```

23.1.2024, 14:02:39 node: debug 5
TempA : msg.payload : Object
  { Zeit: "2024.01.23 14:02:36.401",
    Temperatur: "30.78955" }

23.1.2024, 14:02:48 node: debug 4
TempA : msg.payload : string[8]
"30.78955"

23.1.2024, 14:02:48 node: debug 5
TempA : msg.payload : Object
  { Zeit: "2024.01.23 14:02:44.592",
    Temperatur: "30.78955" }

23.1.2024, 14:02:56 node: debug 4
TempA : msg.payload : string[8]
"30.78955"

23.1.2024, 14:02:56 node: debug 5
TempA : msg.payload : Object
  { Zeit: "2024.01.23 14:02:52.785",
    Temperatur: "30.78955" }

23.1.2024, 14:03:04 node: debug 4
TempA : msg.payload : string[8]
"30.32141"

23.1.2024, 14:03:04 node: debug 5
TempA : msg.payload : Object
  { Zeit: "2024.01.23 14:03:00.977",
    Temperatur: "30.32141" }

23.1.2024, 14:03:12 node: debug 4
TempA : msg.payload : string[8]
"30.78955"

23.1.2024, 14:03:12 node: debug 5

```

PB-07-Dashboard:

- Dashboard Test
- 1. BUTTON
- Select option
- LED 1 Schieber
- chart
- LÖSCHT TEMP-DATEI
- Zeit: 1706011107480

5) dropdown (Dashboard)

The screenshot shows the Node-RED interface with the 'dropdown' node configuration panel on the left, the debug console in the center, and the 'PB-07-Dashboard' on the right.

Node 'dropdown' bearbeiten:

- Buttons: Löschen, Abbrechen, Fertig
- Eigenschaften:
 - Group: [PB-07-Dashboard] Dashboard Test
 - Size: Auto
 - Label: optional label
 - Tooltip: optional tooltip
 - Placeholder: Select option
 - Options:
 - 1 LED 1
 - 2 LED 2
 - 3 LED 3
 - Allow multiple selections from list: ☐
 - If msg arrives on input, pass through to output: ☒
 - Topic: msg.topic

Debug Console:

```

23.1.2024, 14:21:48 node: debug 5
TempA : msg.payload : Object
  { Zeit: "2024.01.23 14:21:45.389",
    Temperatur: "30.78955" }

23.1.2024, 14:21:57 node: debug 4
TempA : msg.payload : string[8]
"30.78955"

23.1.2024, 14:21:57 node: debug 5
TempA : msg.payload : Object
  { Zeit: "2024.01.23 14:21:53.579",
    Temperatur: "30.32141" }

23.1.2024, 14:22:01 node: debug 3
TasterA : msg.payload : string[11]
"Taster A on"

23.1.2024, 14:22:01 node: Ausgabe
TasterA : msg.payload : number
0

23.1.2024, 14:22:01 node: debug 6
TasterA : msg.payload : Object
  { Zeit: "2024.01.23 14:21:57.630",
    Schalter: 0 }

23.1.2024, 14:22:04 node: debug 3
TasterA : msg.payload : string[11]
"Taster A on"

23.1.2024, 14:22:04 node: Ausgabe
TasterA : msg.payload : number
1

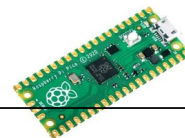
23.1.2024, 14:22:04 node: debug 6
TasterA : msg.payload : Object
  { Zeit: "2024.01.23 14:22:01.211",
    Schalter: 1 }

23.1.2024, 14:22:05 node: debug 4

```

PB-07-Dashboard:

- Dashboard Test
- 1. BUTTON
- Select option
- LED 1 Schieber
- chart
- LÖSCHT TEMP-DATEI
- Zeit: 1706011107480



6) switch (Dashboard)

The screenshot shows the Node-RED 'switch' node configuration panel on the left. The 'Eigenschaften' (Properties) tab is active. The 'Group' is set to '[PB-07-Dashboard] Dashboard Test'. The 'Label' is 'LED 1 Schieber'. The 'Tooltip' is 'optional tooltip'. The 'Icon' is 'Default'. The 'Pass through msg if payload matches valid state' checkbox is checked. The 'When clicked, send:' section shows 'On Payload' as '1' and 'Off Payload' as '0'. The 'Topic' is 'msg.LEDA'. The 'Class' is 'Optional CSS class name(s) for widget'. The 'Name' is 'LED 1 Schieber'. A purple arrow points from the 'Label' field to the 'LED 1 Schieber' widget in the dashboard preview on the right.

The dashboard preview on the right shows a 'Dashboard Test' with a '1. BUTTON' and a 'LED 3' dropdown menu. Below the dropdown is the 'LED 1 Schieber' widget, which is a slider. Below the slider is a 'chart' showing a bar chart with values ranging from 30.2 to 31.4. At the bottom of the dashboard is a 'LÖSCHT TEMP-DATE!' button and a 'Zeit:' label with the value '1706011107480'.

7) function

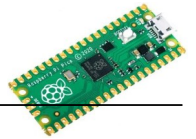
7.1 Trigger wechselt 0&1

The screenshot shows the Node-RED 'function' node configuration panel. The 'Eigenschaften' (Properties) tab is active. The 'Name' is 'Trigger wechselt 0&1'. The 'Funktion' (Function) tab is selected, showing the following JavaScript code:

```

1  if (flow.get("state") === undefined) {
2    flow.set("state", 0);
3  }
4
5  // Wechsle den Zustand zwischen 0 und 1
6  if (flow.get("state") === 0) {
7    flow.set("state", 1);
8  } else {
9    flow.set("state", 0);
10 }
11
12 // Gib den aktuellen Zustand aus
13 msg.payload = flow.get("state");
14 return msg;
15

```



Dieser Code ist wahrscheinlich Teil eines Fluss (Flow) in Node-RED oder einer ähnlichen Flussprogrammierungsumgebung. Lassen Sie uns den Code Schritt für Schritt durchgehen:

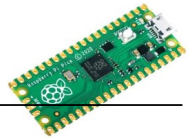
1. `if (flow.get("state") === undefined) { flow.set("state", 0); };` Überprüft, ob der Fluss (Flow) bereits eine Variable mit dem Namen "state" hat. Wenn nicht (also wenn der Wert `undefined` ist), wird die Variable mit dem Wert 0 initialisiert.
 - `flow.get("state")`: Holt den Wert der Variable "state" aus dem Fluss.
 - `=== undefined`: Überprüft, ob der Wert `undefined` ist.
 - `flow.set("state", 0)`: Setzt den Wert der Variable "state" auf 0, wenn sie zuvor nicht existiert hat.
2. `if (flow.get("state") === 0) { flow.set("state", 1); } else { flow.set("state", 0); };` Ändert den Wert der Variable "state" zwischen 0 und 1. Wenn der aktuelle Wert gleich 0 ist, wird er auf 1 gesetzt, und wenn er nicht gleich 0 ist, wird er auf 0 gesetzt.
 - `flow.get("state")`: Holt den aktuellen Wert der Variable "state".
 - `=== 0`: Überprüft, ob der Wert gleich 0 ist.
 - `flow.set("state", 1)`: Setzt den Wert auf 1, wenn der aktuelle Wert gleich 0 ist.
 - `flow.set("state", 0)`: Setzt den Wert auf 0, wenn der aktuelle Wert nicht gleich 0 ist.
3. `msg.payload = flow.get("state");` Setzt den Payload-Wert des Nachrichtenobjekts (msg) auf den aktuellen Wert der Variable "state".
4. `return msg;` Gibt das aktualisierte Nachrichtenobjekt zurück. Dieser aktualisierte Payload kann dann in anderen Teilen des Flusses weiterverarbeitet oder angezeigt werden.

Insgesamt implementiert dieser Code eine einfache Zustandsmaschine mit einem binären Zustand (0 oder 1) und aktualisiert diesen Zustand bei jedem Durchlauf des Flusses. Der Zustand wird in der Variable "state" im Fluss gespeichert, und der aktuelle Zustand wird dem Payload der Nachricht hinzugefügt.

8) chart (Dashboard)

The screenshot shows the Node-RED interface with the 'chart' node configuration on the left and a live dashboard on the right. The configuration panel for the 'chart' node is titled 'Node 'chart' bearbeiten' and includes options for 'Gruppe', 'Größe', 'Beschriftung', 'Typ', 'X-Achse', 'X-Beschriftung', 'Y-Achse', 'Legende', 'Serienfarben', 'Leer-Text', '</> Klasse', and 'Name'. The 'Beschriftung' field is set to 'chart'. The 'Typ' is set to 'Liniendiagramm'. The 'X-Achse' is set to 'Letzten 1 Stunden' and 'Punkte vergrößern' is checked. The 'X-Beschriftung' is set to 'HH:mm:ss' and 'als UTC' is checked. The 'Y-Achse' has 'min' and 'max' fields. The 'Legende' is set to 'Keine' and 'Interpolation' is set to 'Linear'. The 'Serienfarben' are set to a default color palette. The 'Leer-Text' is set to 'Anzuzeigender Text bevor gültige Daten eintreffen'. The '</> Klasse' is set to 'Optionale(r) CSS-Klassenname(n) für das Widge'. The 'Name' is set to 'Name'.

The live dashboard on the right is titled 'PB-07-Dashboard' and contains a 'Dashboard Test' section. It features a '1. BUTTON' and a 'LED 3' dropdown menu. Below these is a 'LED 1 Schieber' slider. The main part of the dashboard is a 'chart' node displaying a line graph with temperature data over time. The x-axis is labeled 'Zeit' and the y-axis is labeled 'Temperatur'. The data points are shown as a series of vertical bars. Below the chart is a 'LÖSCHT TEMP-DATEI' button. At the bottom, there is a 'Zeit:' label and a value '1706011107480'.



8.1 Temperatur loggen

Node 'function' bearbeiten

Löschen Abbrechen Fertig

Eigenschaften

Name Temperatur loggen

Setup Start Funktion Stopp

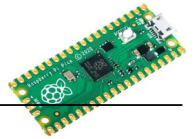
```

1  var date = new Date();
2
3  var year = date.getFullYear();
4  var month = (date.getMonth() + 1).toString().padStart(2, '0');
5  var day = date.getDate().toString().padStart(2, '0');
6  var hours = date.getHours().toString().padStart(2, '0');
7  var minutes = date.getMinutes().toString().padStart(2, '0');
8  var seconds = date.getSeconds().toString().padStart(2, '0');
9  var milliseconds = date.getMilliseconds().toString().padStart(2, '0');
10
11 var result = year + "." + month + "." + day + " " + hours + ":" + minutes + ":" + seconds + "." + milliseconds;
12
13 var payload = msg.payload;
14
15 // Zeitstempel zur Payload hinzufügen
16 var newPayload = {
17     Zeit: result,
18     Temperatur: payload
19 };
20
21 msg.payload = newPayload;
22
23 return msg;

```

Dieser Code erstellt einen Zeitstempel und fügt ihn einer Payload in einem JavaScript-Programm hinzu. Hier ist eine detaillierte Erklärung:

1. `var date = new Date();`; Hier wird ein neues Date-Objekt erstellt, das den aktuellen Zeitpunkt repräsentiert.
2. Die nächsten Zeilen extrahieren verschiedene Zeitkomponenten aus dem Date-Objekt:
 - `var year = date.getFullYear();`; Holt das aktuelle Jahr.
 - `var month = (date.getMonth() + 1).toString().padStart(2, '0');`; Holt den aktuellen Monat. Beachte, dass `getMonth()` mit Werten von 0 bis 11 arbeitet, daher wird 1 addiert. Die `padStart`-Funktion wird verwendet, um sicherzustellen, dass der Monat immer zwei Ziffern hat.



- `var day = date.getDate().toString().padStart(2, '0');` Holt den Tag des Monats und stellt sicher, dass er immer zwei Ziffern hat.
 - `var hours = date.getHours().toString().padStart(2, '0');` Holt die Stunden und stellt sicher, dass sie immer zwei Ziffern haben.
 - `var minutes = date.getMinutes().toString().padStart(2, '0');` Holt die Minuten und stellt sicher, dass sie immer zwei Ziffern haben.
 - `var seconds = date.getSeconds().toString().padStart(2, '0');` Holt die Sekunden und stellt sicher, dass sie immer zwei Ziffern haben.
 - `var milliseconds = date.getMilliseconds().toString().padStart(2, '0');` Holt die Millisekunden und stellt sicher, dass sie immer zwei Ziffern haben.
3. `var result = year + "." + month + "." + day + " " + hours + ":" + minutes + ":" + seconds + "." + milliseconds;` Hier wird der Zeitstempel erstellt, indem die zuvor extrahierten Komponenten zu einem String zusammengesetzt werden. Der Zeitstempel hat das Format "YYYY.MM.DD HH:MM:SS.SSS".
 4. `var payload = msg.payload;` Hier wird der ursprüngliche Payload-Wert von msg in einer Variable payload gespeichert.
 5. `var newPayload = { Zeit: result, Temperatur: payload };` Ein neues Objekt newPayload wird erstellt, das zwei Eigenschaften enthält: "Zeit" mit dem zuvor erstellten Zeitstempel und "Temperatur" mit dem ursprünglichen Payload-Wert.
 6. `msg.payload = newPayload;` Der ursprüngliche Payload-Wert von msg wird durch das neue newPayload-Objekt ersetzt.
 7. `return msg;` Die veränderte msg wird zurückgegeben. Dies könnte Teil eines größeren Programms oder Flusses sein, der diese veränderte msg weiterverarbeitet.

9) Write file

9.1 Datei löschen

Node 'write file' bearbeiten

Löschen
Abbrechen
Fertig

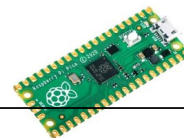
Eigenschaften

Dateiname
path 20230604 Test V01.txt

Aktion
Datei löschen

Name

Tipp: Der Dateiname sollte ein absoluter Pfad sein. Andernfalls wird er relativ zum Arbeitsverzeichnis des Node-RED-Prozesses angewandt.



9.2 an Datei anhängen

Node 'write file' bearbeiten

Löschen Abbrechen Fertig

Eigenschaften

📁 Dateiname

⚙️ Aktion

☒ Zeilenumbruch (\n) zu jeden Nutzdaten (Payload) hinzufügen

☐ Verzeichnis erstellen, wenn nicht vorhanden

🚩 Kodierung

🔑 Name

Tipp: Der Dateiname sollte ein absoluter Pfad sein. Andernfalls wird er relativ zum Arbeitsverzeichnis des Node-RED-Prozesses angewandt.

10)text (Dashboard)

Node 'text' bearbeiten

Löschen Abbrechen Fertig

Eigenschaften

📁 Group

📏 Size

🏷️ Label

📄 Value format

📐 Layout

label value labelvalue labelvalue

label value label value

⚙️ Style ☒ Apply Style

🔤 Font

📏 Text Size

🎨 Text Color

📄 Class

🔑 Name

Debug

23.1.2024, 14:48:34 node: debug 5
TempA : msg.payload : Object
▶ { Zeit: "2024.01.23 14:48:30.546", Temperatur: "30.32141" }

23.1.2024, 14:48:41 node: debug 4
TempA : msg.payload : string[8]
"30.32141"

23.1.2024, 14:48:41 node: debug 5
TempA : msg.payload : Object
▶ { Zeit: "2024.01.23 14:48:38.227", Temperatur: "30.32141" }

23.1.2024, 14:48:49 node: debug 4
TempA : msg.payload : string[8]
"30.32141"

23.1.2024, 14:48:49 node: debug 5
TempA : msg.payload : Object
▶ { Zeit: "2024.01.23 14:48:45.906", Temperatur: "30.32141" }

23.1.2024, 14:48:57 node: debug 4
TempA : msg.payload : string[8]
"30.32141"

23.1.2024, 14:48:57 node: debug 5
TempA : msg.payload : Object
▶ { Zeit: "2024.01.23 14:48:53.588", Temperatur: "30.32141" }

23.1.2024, 14:49:04 node: debug 4
TempA : msg.payload : string[8]
"30.32141"

23.1.2024, 14:49:04 node: debug 5
TempA : msg.payload : Object
▶ { Zeit: "2024.01.23 14:49:01.268", Temperatur: "30.32141" }

PB-07-Dashboard

Dashboard Test

1. BUTTON

LED 3

LED 1 Schieber

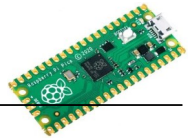
chart

30,8
30,7
30,6
30,5
30,4
30,3

13:49:00 14:19:00 14:50:00

LÖSCHT TEMP-DATEI

Zeit: 1706011107480



11)mqtt in

11.1 TasterA

Node 'mqtt in' bearbeiten

Löschen Abbrechen Fertig

Eigenschaften

Server 192.168.0.233:1883

Action Subscribe to single topic

Topic TasterA

QoS 2

Ausgang Auto-Erkennung (parsed JSON-Objekt, string oder I

Name Name

11.2 TempA

Node 'mqtt in' bearbeiten

Löschen Abbrechen Fertig

Eigenschaften

Server 192.168.0.233:1883

Action Subscribe to single topic

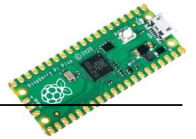
Topic TempA

QoS 2

Ausgang Ein String

Name

- Auto-Erkennung (parsed JSON-Objekt, string oder buffer)
- Auto-Erkennung (string oder buffer)
- Einen binären Buffer
- Ein String**
- Ein analysiertes (parsed) JSON-Objekt
- Ein Base64-kodierter String



12)mqtt out

12.1 LEDA

Node 'mqtt out' bearbeiten

Löschen

Abbrechen

Fertig

Eigenschaften

Server

192.168.0.233:1883

▼

Topic

LEDA

QoS

▼

Retain

▼

Name

Name

Tipp: Topic, QoS oder Retain leer lassen, um diese über die msg-Eigenschaften festzulegen