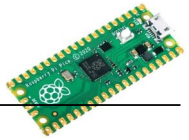


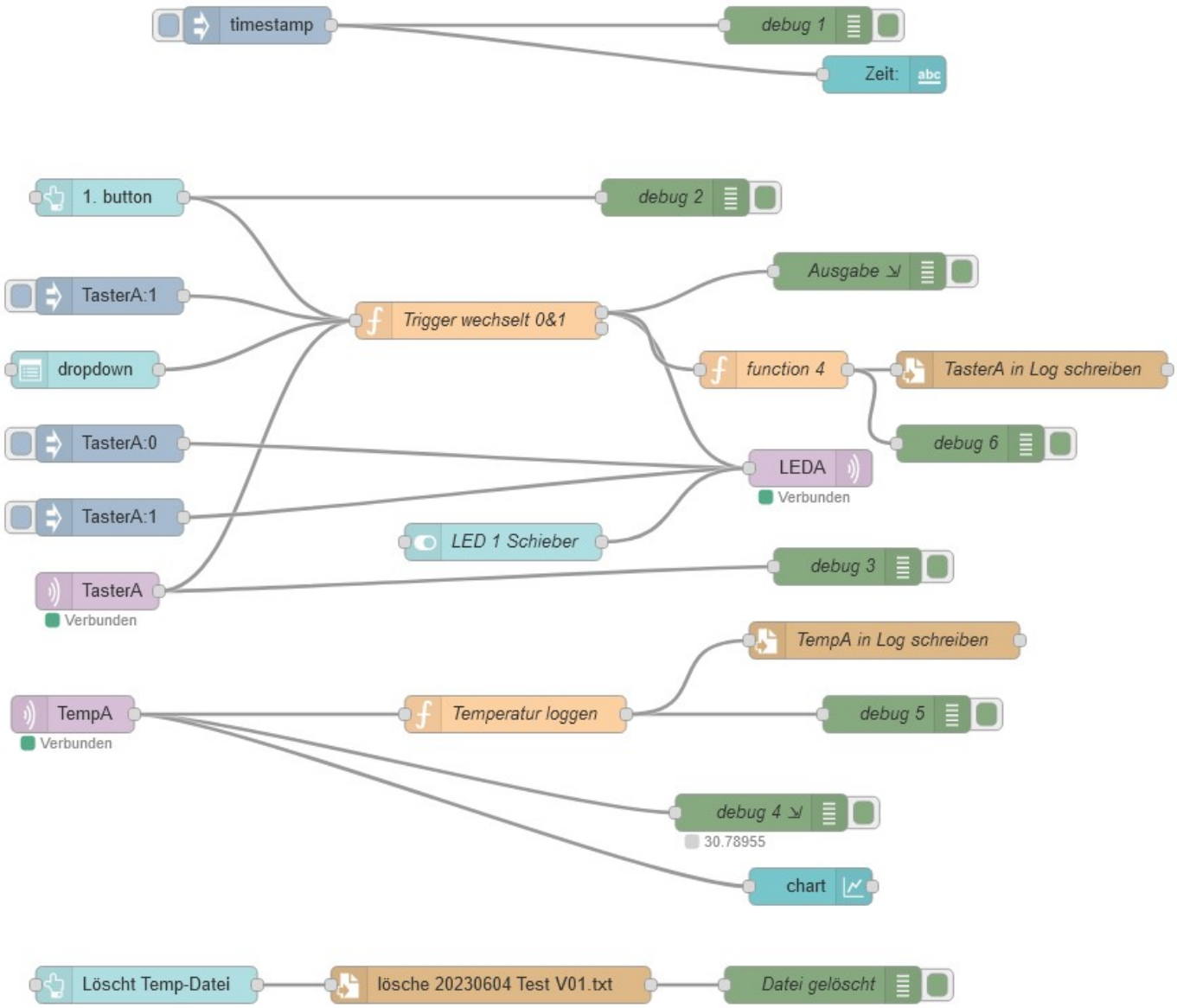
SchrittWeise* PicoBello-08

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

Inhaltsverzeichnis

(* 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.

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.

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

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

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

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.

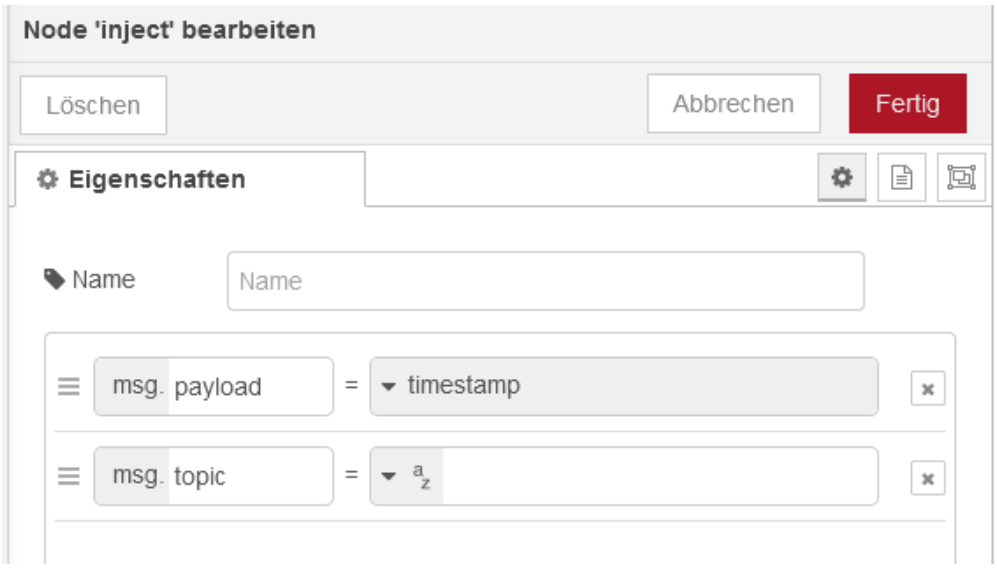
The value of a property can be any valid JavaScript type, such as:

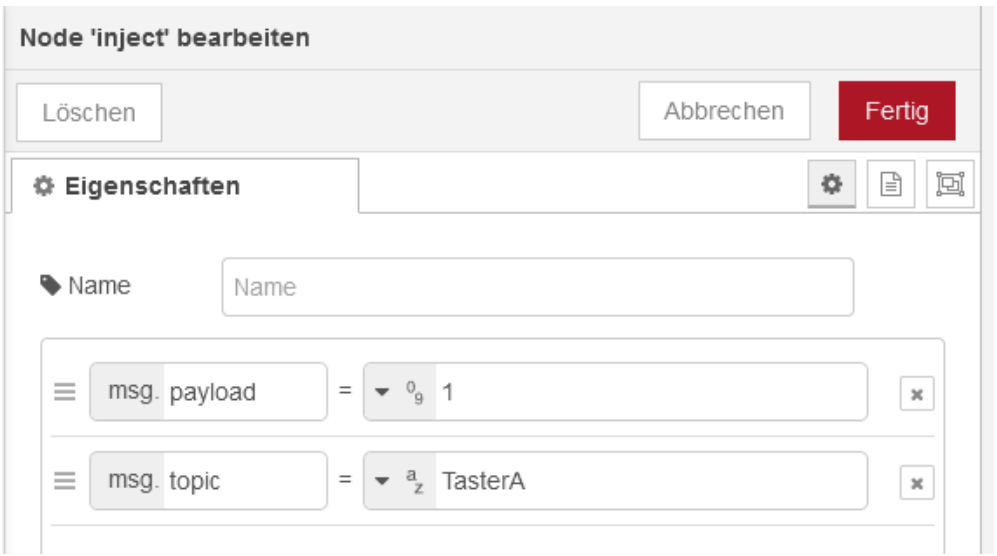
- Boolean - `true`, `false`
- Number - eg `0`, `123.4`
- String - `"hello"`
- Array - `[1,2,3,4]`
- Object - `{ "a": 1, "b": 2}`
- Null

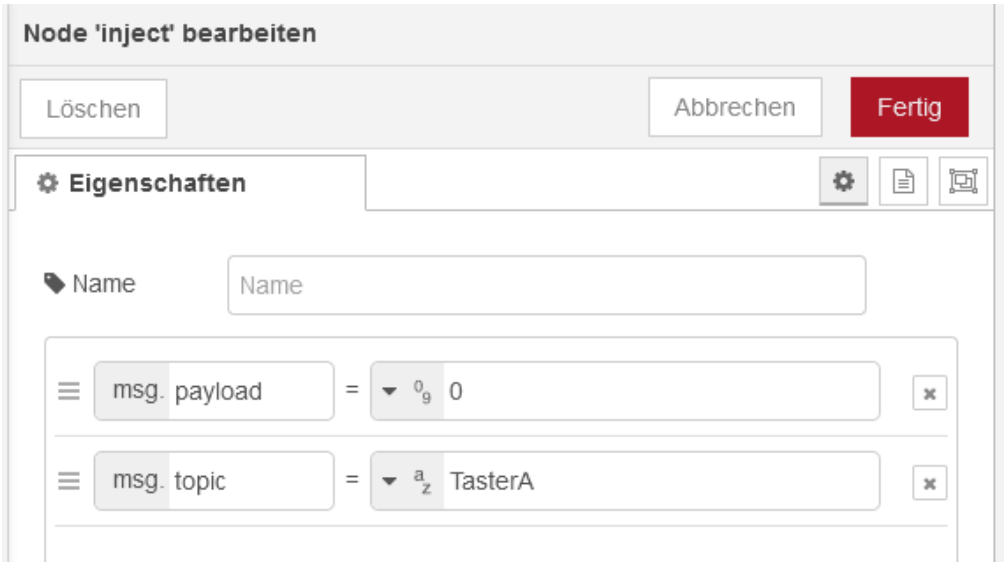
Der Wert einer Eigenschaft kann jeden gültigen JavaScript-Typ haben, wie zum Beispiel:

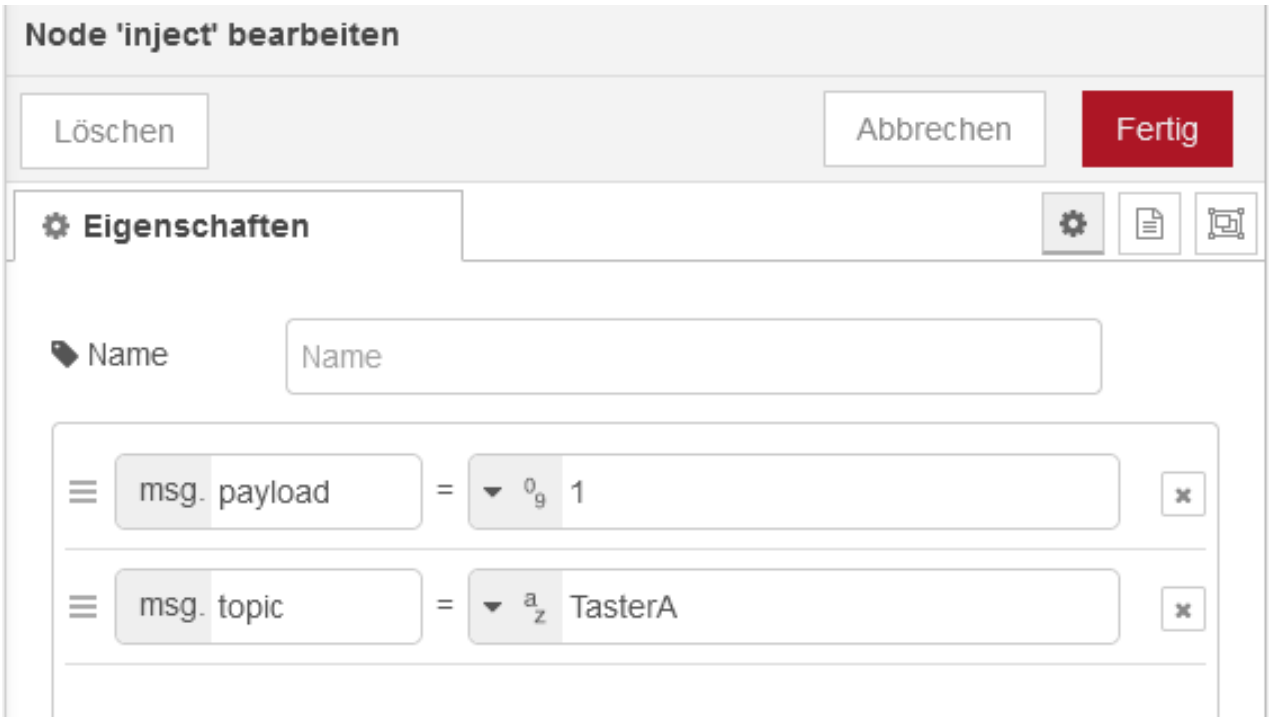
- Boolean - `true`, `false`
- Number - eg `0`, `123.4`
- String - `"hello"`
- Array - `[1,2,3,4]`
- Object - `{ "a": 1, "b": 2}`
- Null

3) Inject


Node 'inject' bearbeiten
Löschen
Abbrechen
Fertig
Eigenschaften
Name
Name
msg. payload = timestamp
msg. topic =

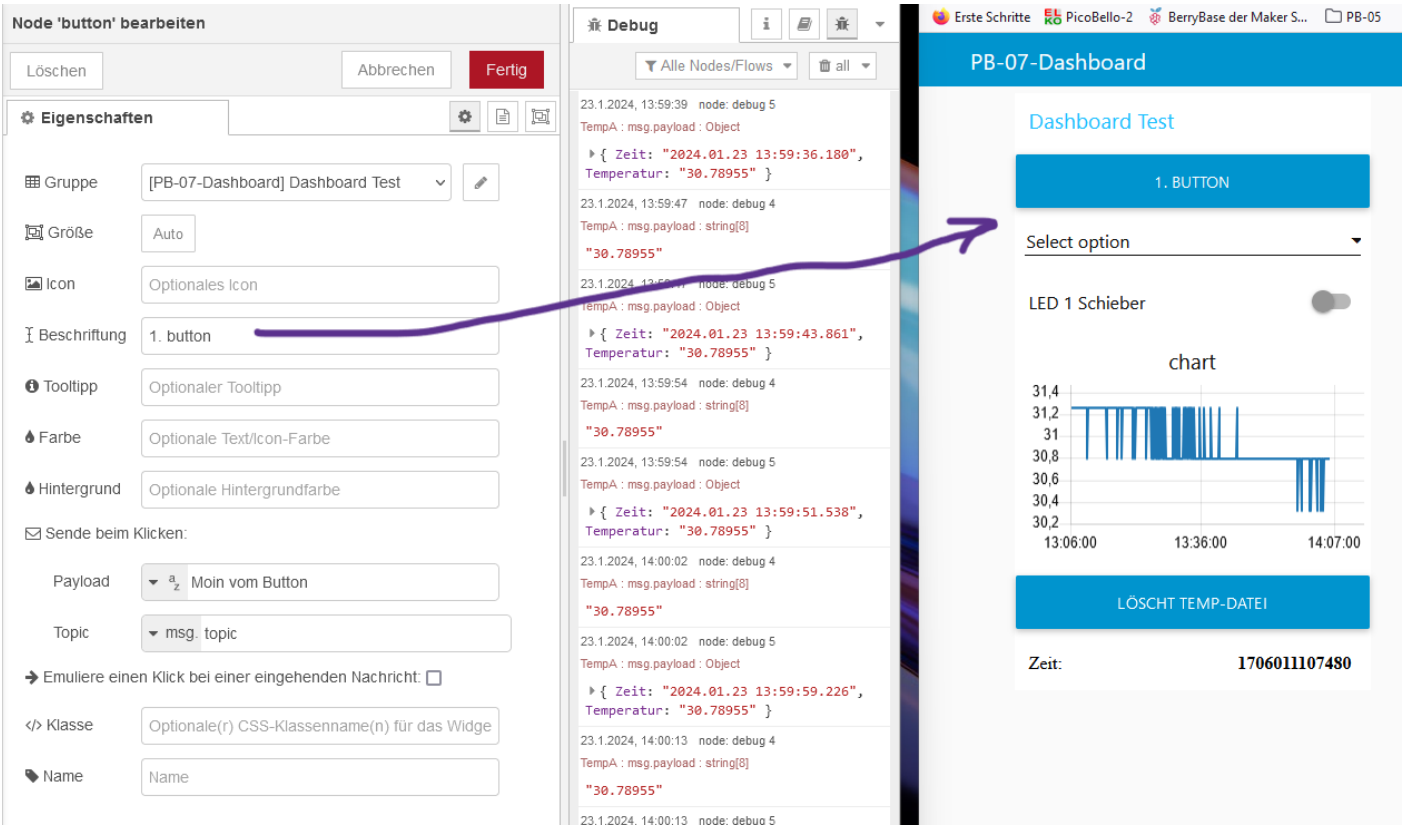

Node 'inject' bearbeiten
Löschen
Abbrechen
Fertig
Eigenschaften
Name
Name
msg. payload = 1
msg. topic = TasterA


Node 'inject' bearbeiten
Löschen
Abbrechen
Fertig
Eigenschaften
Name
Name
msg. payload = 0
msg. topic = TasterA

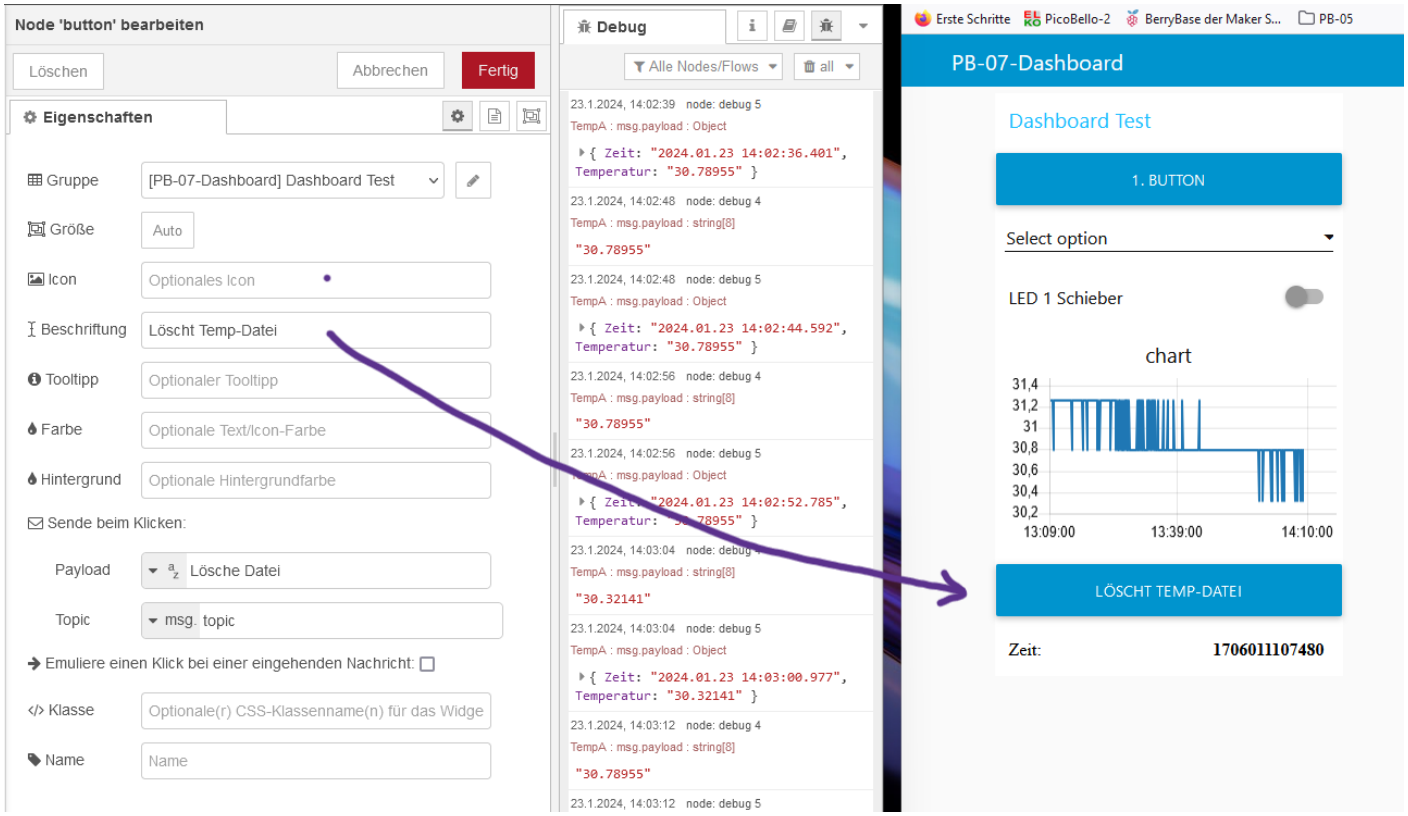


4) button (Dashboard)

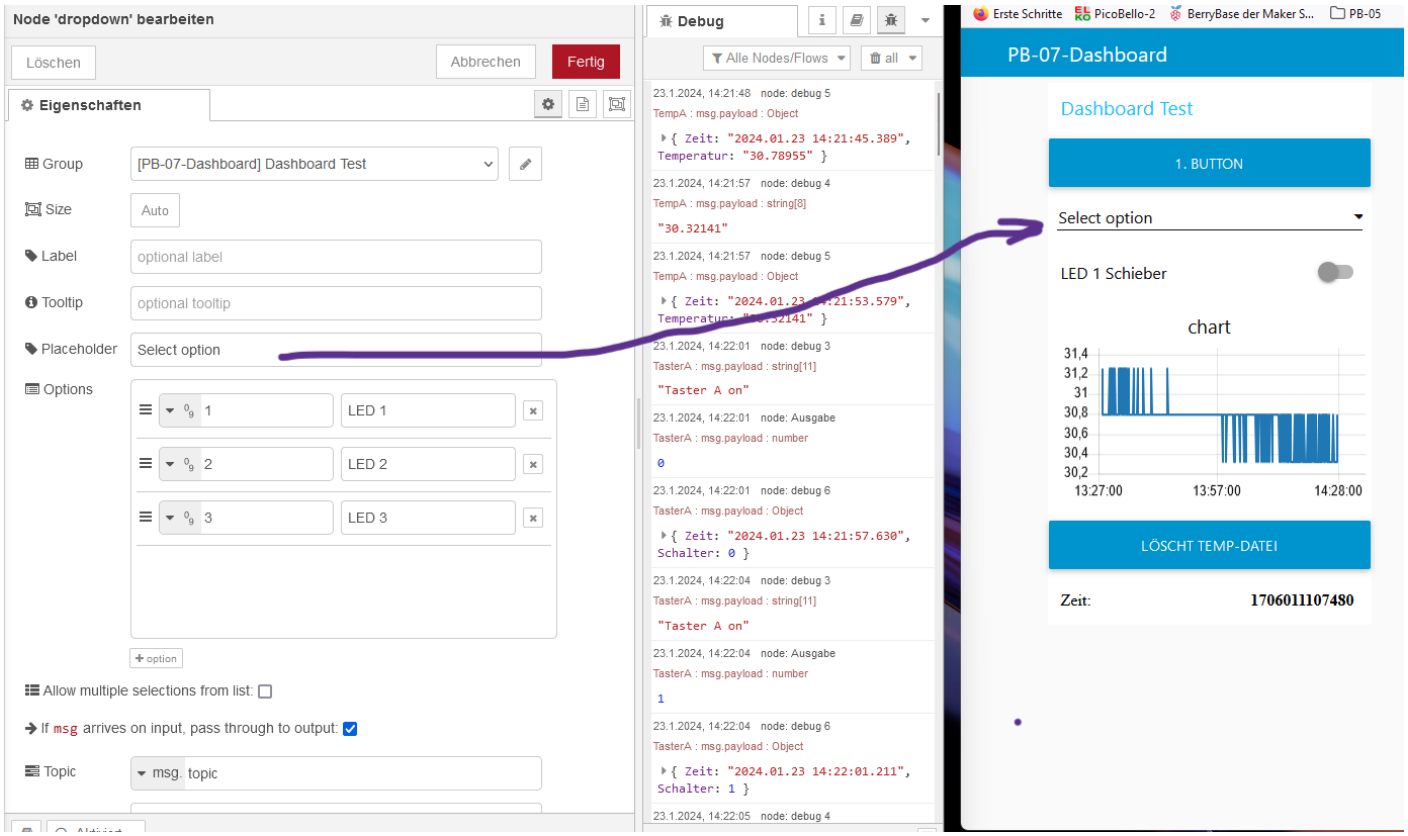
4.1 1. button



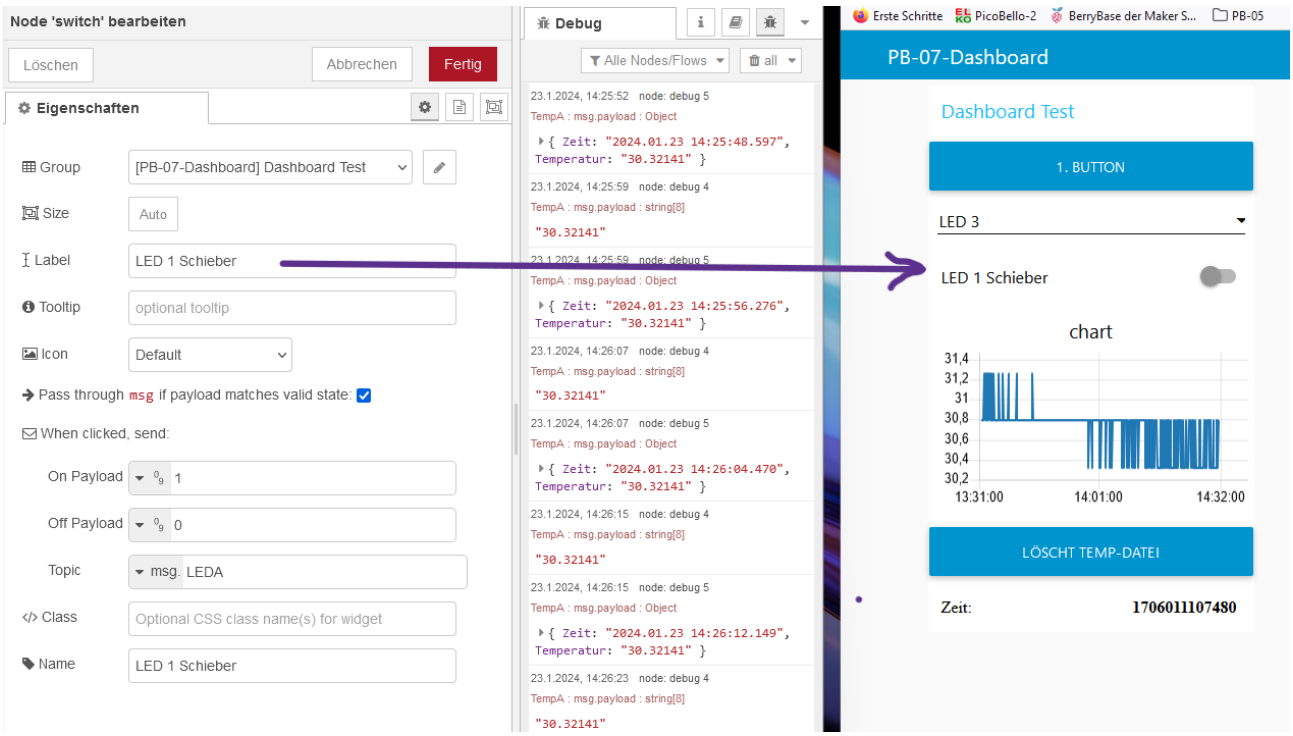
4.2 Löscht Temp-Datei



5) dropdown (Dashboard)

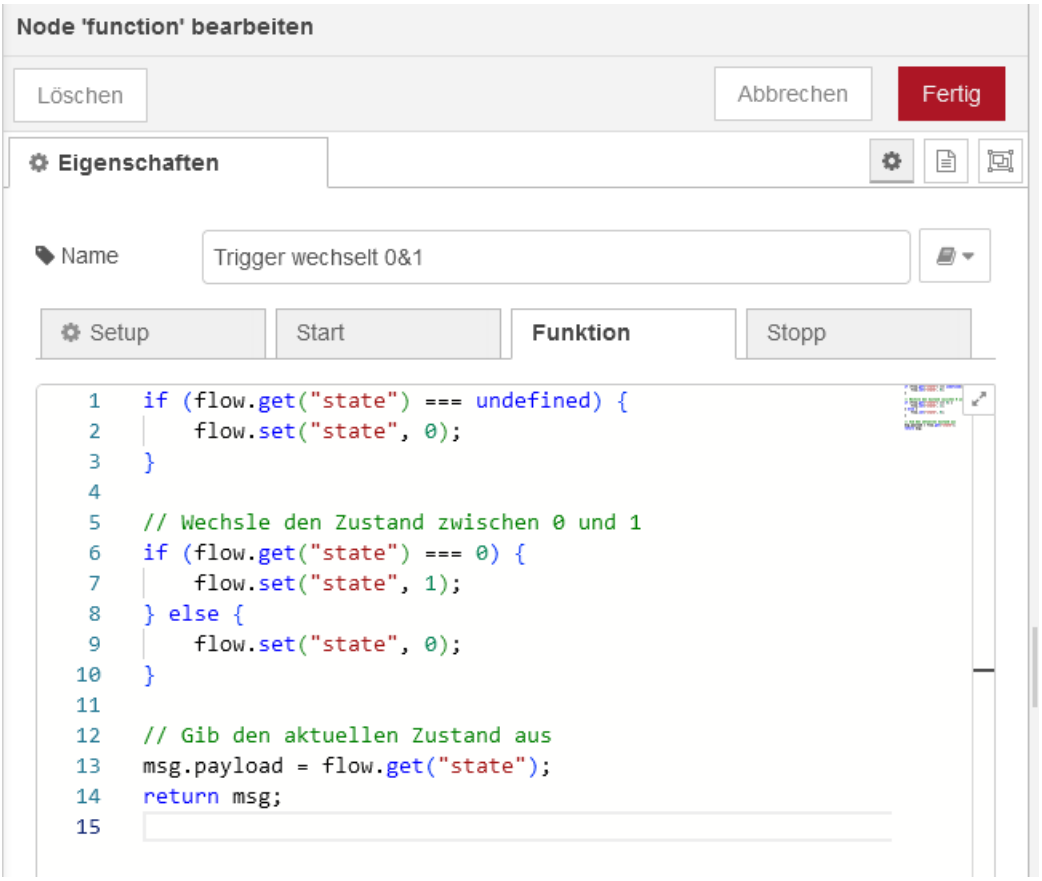


6) switch (Dashboard)



7) function

7.1 Trigger wechselt 0&1

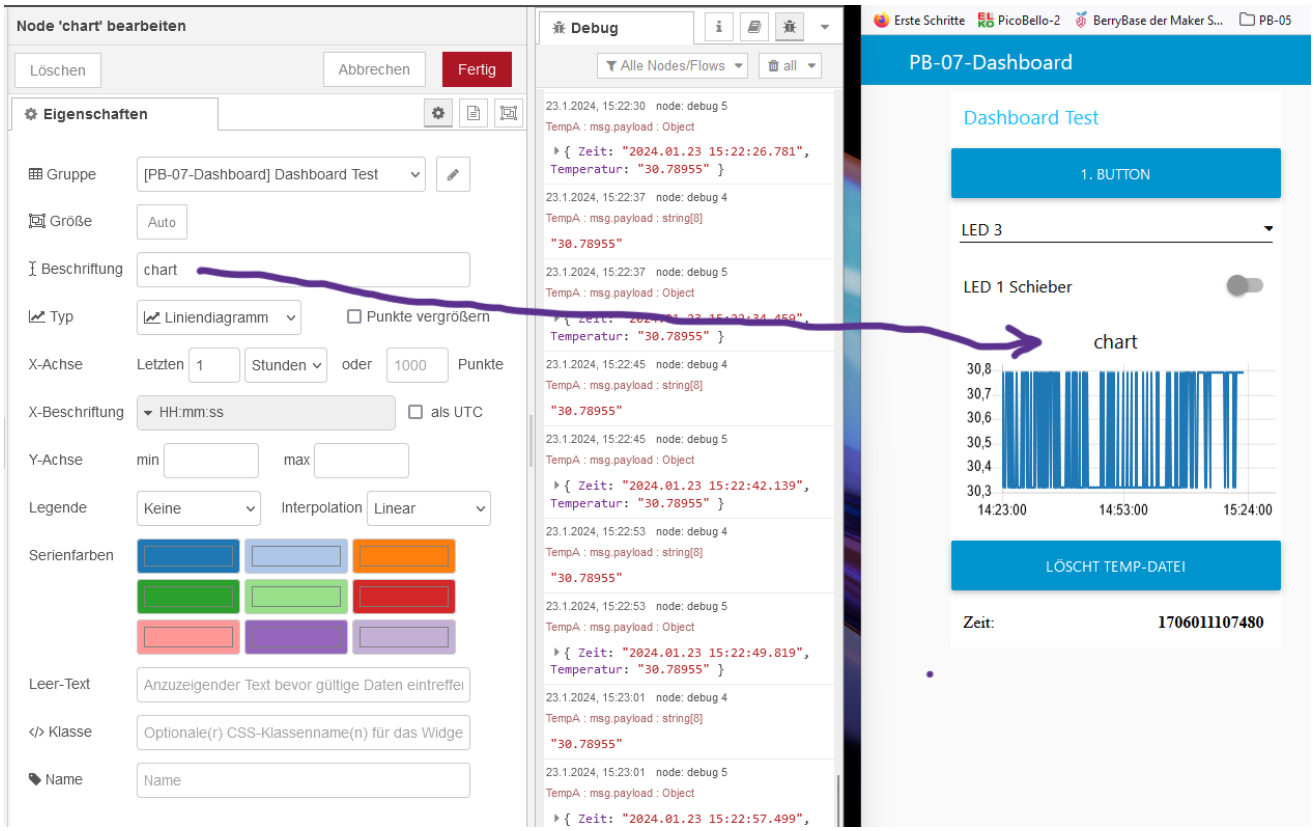


Dieser Code ist wahrscheinlich Teil eines Fluss (Flow) in Node-RED oder einer ähnlichen Flussprogrammierungsumgebung. Lassen Sie uns den Code Schritt für Schritt durchgehen:

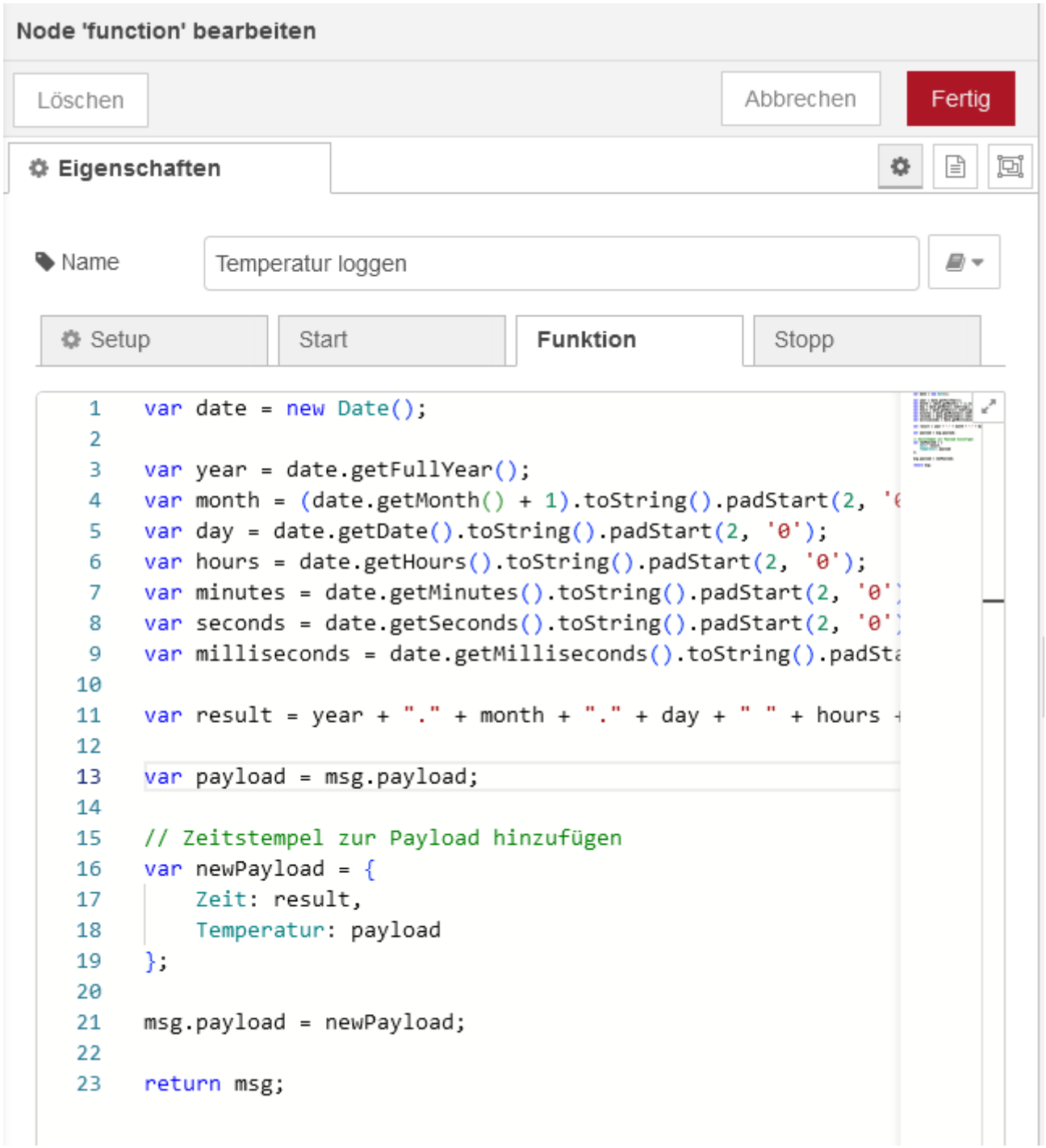
1. `if (flow.get("state") === undefined) { flow.set("state", 0); }`: Überprüft, ob der Fluss (Flow) bereits eine Variable mit dem Namen "state" hat. Wenn nicht (also wenn der Wert `undefined` ist), wird die Variable mit dem Wert 0 initialisiert.
    - `flow.get("state")`: Holt den Wert der Variable "state" 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)



8.1 Temperatur loggen



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

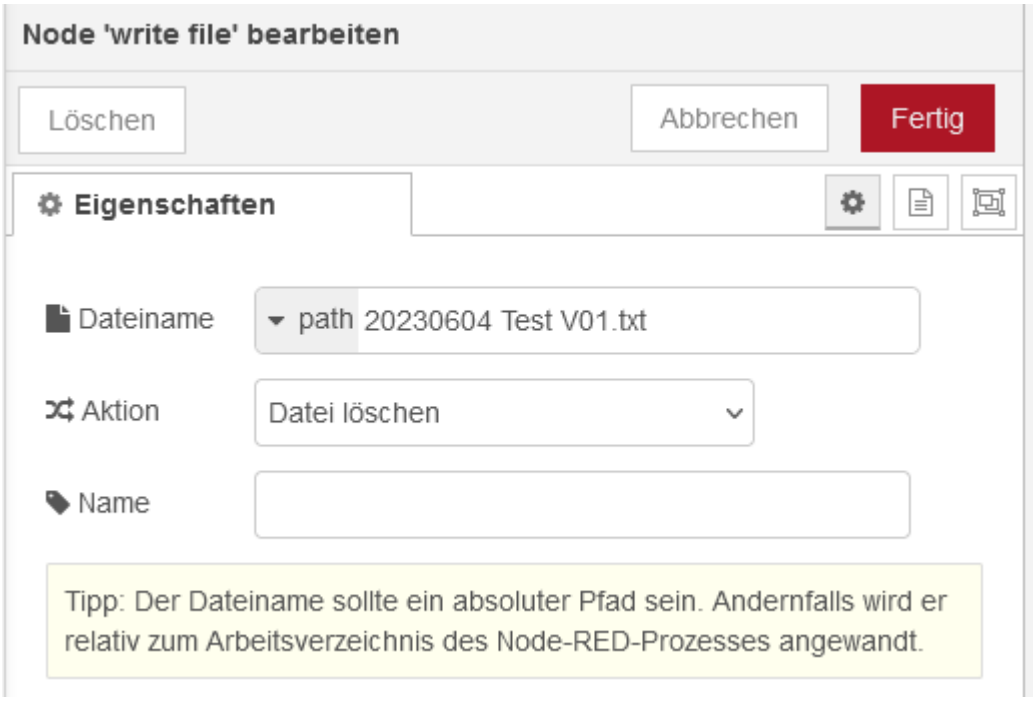
1. `var date = new Date();`: Hier wird ein neues Date-Objekt erstellt, das den aktuellen Zeitpunkt repräsentiert.
2. Die nächsten Zeilen extrahieren verschiedene Zeitkomponenten aus dem Date-Objekt:
   - `var year = date.getFullYear();`: Holt das aktuelle Jahr.
   - `var month = (date.getMonth() + 1).toString().padStart(2, '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.

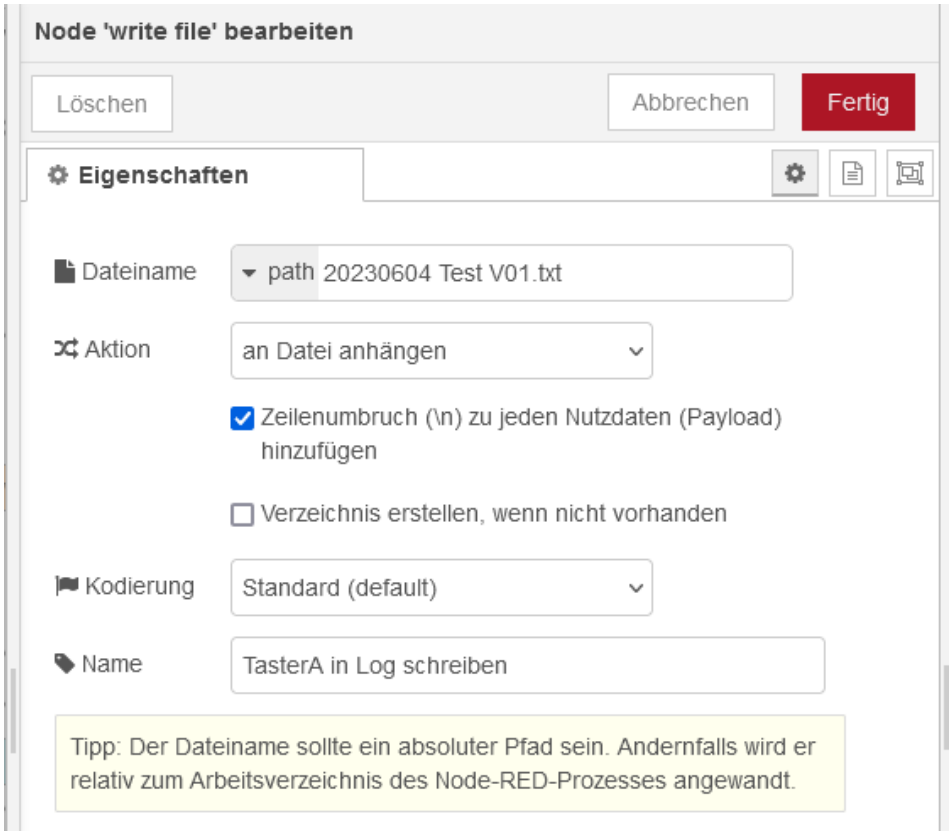
3. `var result = year + "." + month + "." + day + " " + hours + ":" + minutes + ":" + seconds + "." + milliseconds;`: Hier wird der Zeitstempel erstellt, indem die zuvor extrahierten Komponenten zu einem String zusammengesetzt werden. Der Zeitstempel hat das Format "YYYY.MM.DD HH:MM:SS.SSS".
4. `var 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

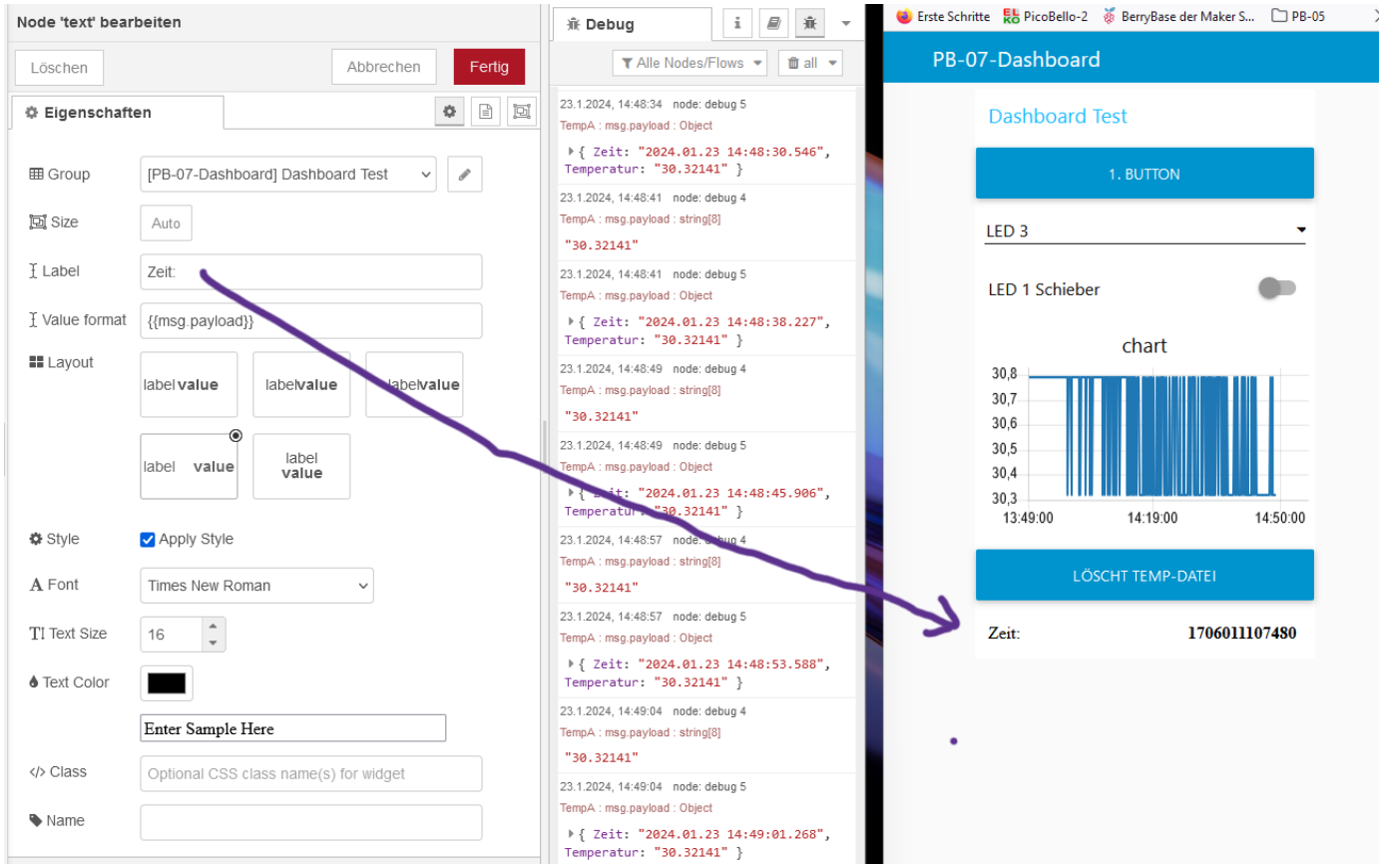
9.1 Datei löschen



9.2 an Datei anhängen

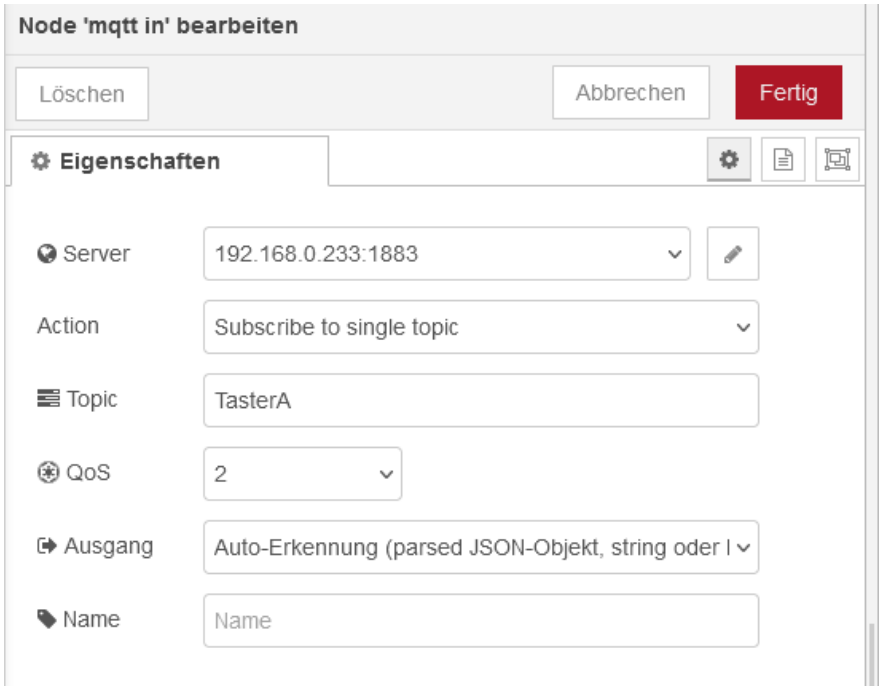


10)text (Dashboard)

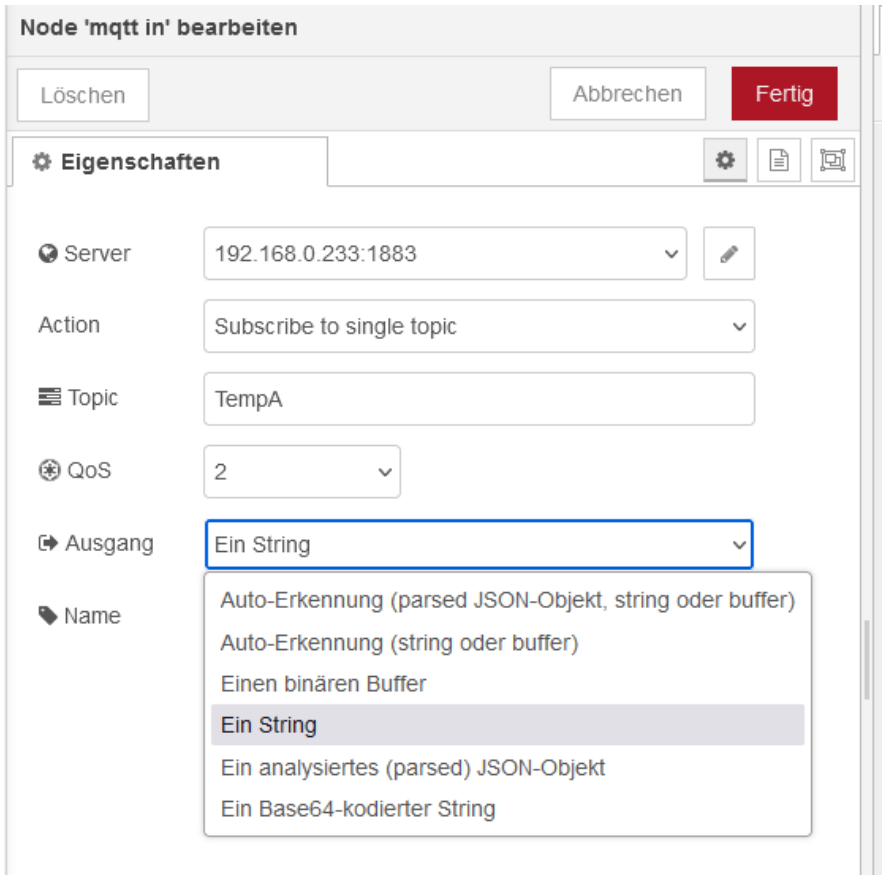


11)mqtt in

11.1 TasterA



11.2 TempA



12)mqtt out

12.1 LEDA