Kapitel 6

MQTT und Node-Red mit Raspberry Pi

In diesem Kapitel wird der Raspberry Pi als Schaltzentrale für das IoT-System oder Smarthome aufgebaut.

Die Schaltzentrale wird Daten von verschiedenen Quellen empfangen und verarbeiten. Die zentralen Komponenten dieser Schaltzentrale sind der MQTT-Broker und die grafische Oberfläche Node-Red.

6.1 Raspberry Pi als Schaltzentrale

Der Raspberry Pi als Schaltzentrale für unser System wurde bereits in Kapitel 1 kurz vorgestellt.

In Abbildung 6.1 ist nochmals das Prinzip der Infrastruktur abgebildet.

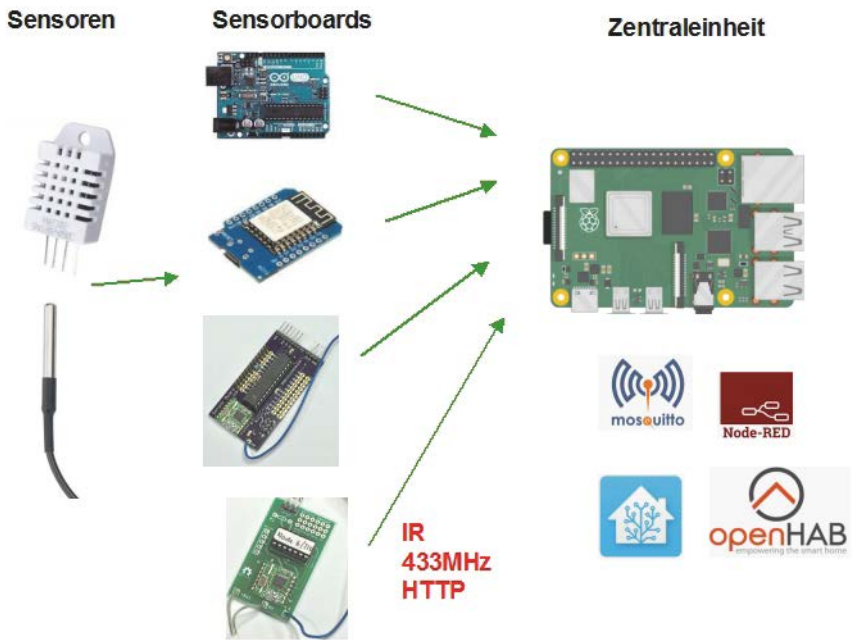

Abb. 6.1: Raspberry Pi als Schaltzentrale

Der Raspberry Pi ist die Zentraleinheit und verarbeitet über seine Schnittstellen die empfangenen Daten und löst Aktionen aus.

Der in Abschnitt 1.2 installierte Raspberry Pi mit dem grafischen Betriebssystem und dem in Kapitel 4 installierten MQTT-Broker ist eine gute Basis für das zukünftige System.

Was bisher noch nicht erwähnt wurde, ist die grafische Entwicklungsumgebung Node-Red. Diese erkläre ich später noch im Detail. Node-Red ist auch bereits installiert und kann über das Startmenü des Raspberry Pi OS aufgerufen werden.

Über die USB-Schnittstellen können externe Arduino-Module direkt serielle Daten liefern. In einem Praxisbeispiel in diesem Kapitel zeige ich diese Lösung.

Mittels MQTT-Schnittstelle können Sensor-Boards, basierend auf dem ESP8266, drahtlos Daten und Informationen an die Zentraleinheit liefern.

Das Raspberry-Board ist ein richtiger Computer und kann somit direkt mit einem Bildschirm zur Anzeige von Daten und zur Dateneingabe eingesetzt werden.

6.2 Mosquitto als MQTT-Broker

Der Mosquitto-MQTT-Broker ist eine kompakte Serveranwendung und wurde bereits in Kapitel 4 installiert und überprüft.

Ebenfalls in Kapitel 4 wurden die ersten Topics über die Terminal-Konsole abonniert und mit Daten befüllt.

Für den Betrieb des MQTT-Brokers ist grundsätzlich kein User-Interface nötig.

Informationen zum Broker können Sie über das Terminal abfragen.

Mit dem Befehl

```
mosquitto
```

erhalten Sie die Version des installierten MQTT-Brokers (Abbildung 6.2).

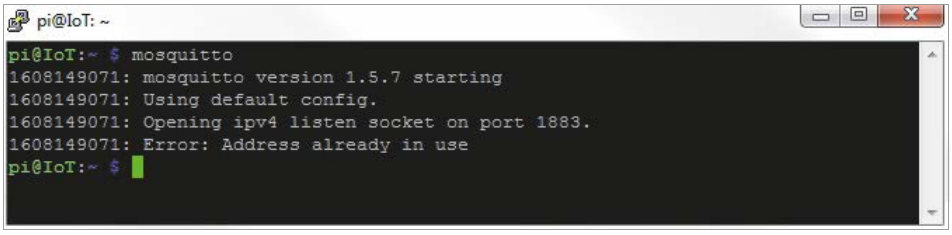

Abb. 6.2: Installierter Mosquitto-Broker

Da der MQTT-Broker auf dem Raspberry Pi läuft, ist entsprechend auch die zugehörige IP-Adresse bekannt. Sein Standard-Port ist 1883. Beide Werte, IP und Port, werden später für die Konfiguration des MQTT-Brokers in Node-Red noch benötigt.

6.3 Node-Red

Node-Red ist ein grafisches Werkzeug, um sogenannte Flows zu erstellen. Durch das Zusammensetzen von einzelnen Elementen, als Nodes bezeichnet, kann man grafische Informationsflüsse realisieren.

In der aktuellen Version des Raspberry Pi OS ist Node-Red bereits installiert.

Start von Node-Red

Das Frontend von Node-Red wird über den Browser aufgerufen. Vor dem Start im Browser muss der Node-Red-Server über ein Terminalfenster auf dem Raspberry Pi mit folgendem Befehl gestartet werden.

```
node-red start
```

Nach Start im Terminal meldet sich das System (Abbildung 6.3).

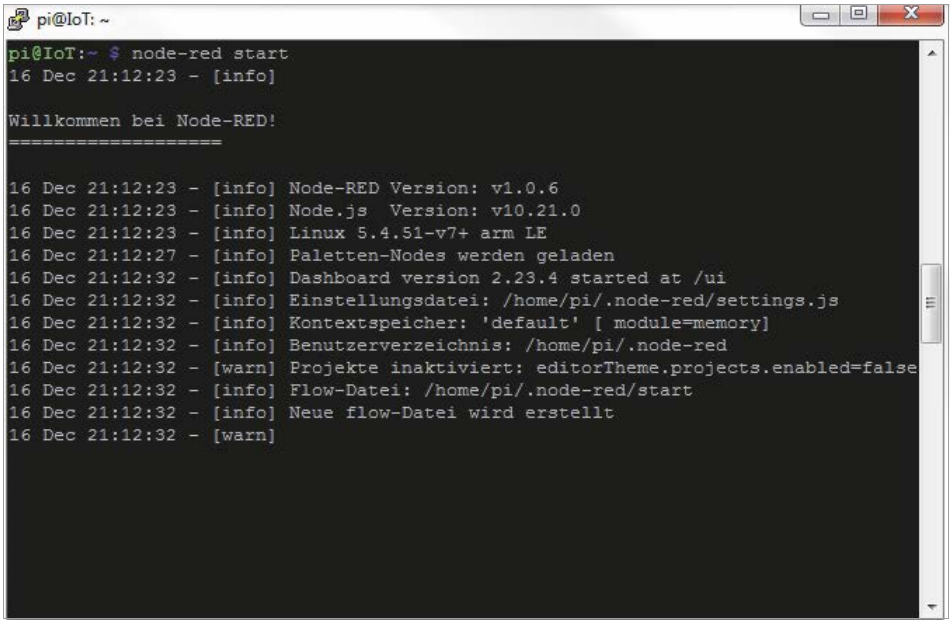

Abb. 6.3: Node-Red-Start

Nach dem Start-Befehl läuft der Node-Red-Server, bis das System gestoppt oder der Raspberry Pi neu gestartet wird.

Folgende Betriebsarten stehen für den Betrieb von Node-Red zur Verfügung.

Befehl	Beschreibung
node-red-start	Starten von Node-Red
node-red-stop	Beenden von Node-Red
node-red-log	Anzeigen der Log-Dateien
sudo systemctl enable nodered.service	Aktivieren des automatischen Starts bei jedem Neustart
sudo systemctl disable nodered.service	Deaktivieren des automatischen Starts bei jedem Neustart

Tabelle 6.1: Node-Red-Betriebsarten

Der automatische Start von Node-Red erfolgt also gemäß Tabelle 6.1 mit dem Befehl.

```
sudo systemctl enable nodered.service
```

Damit ist der Start von Node-Red bei einem Neustart des Systems aktiviert (Abbildung 6.4).

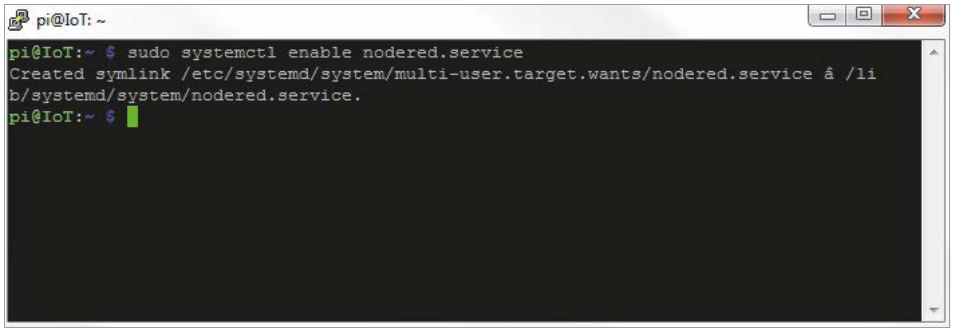

Abb. 6.4: Node-Red für Autostart einrichten

Nach Start des Node-Red-Servers kann das Node-Red-Frontend über den Browser aufgerufen werden. Dieser Service läuft standardmäßig über den Port 1880.

```
http://IP-des-Raspberry:1880
```

Im Browser erscheint die Oberfläche der Anwendung (Abbildung 6.5).

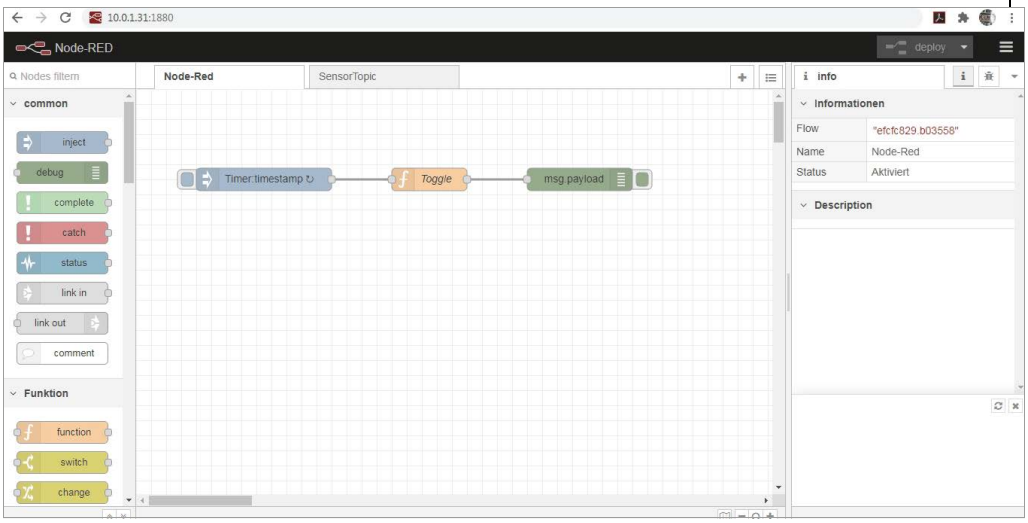

Abb. 6.5: Node-Red-Frontend

Oberfläche

In Abbildung 6.6 ist die Oberfläche von Node-Red dargestellt.

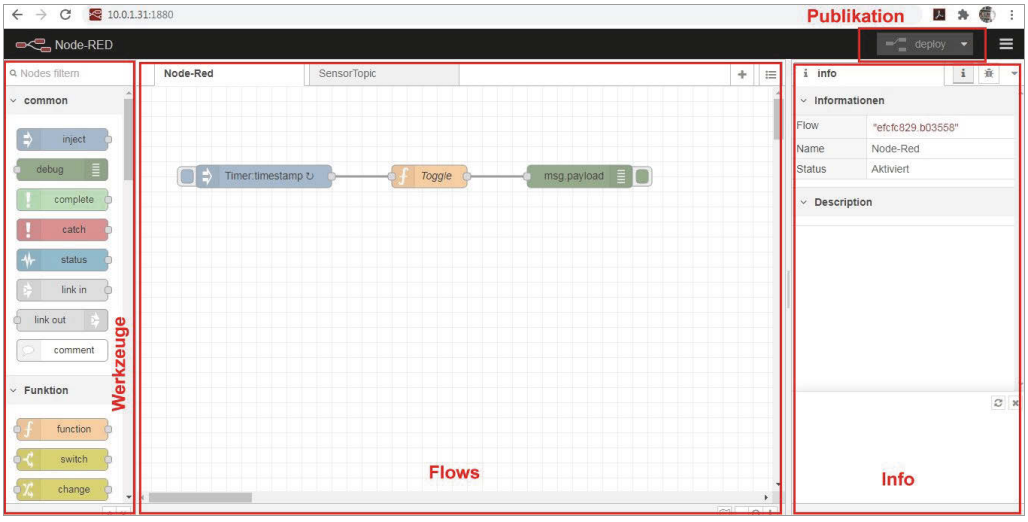

Abb. 6.6: Node-Red – Oberfläche

Das zentrale Element ist der mittlere Bereich mit den unterschiedlichen Flows. Über einzelne Tabs kann man die einzelnen Flows strukturieren.

In der linken Spalte ist die Werkzeugpalette angeordnet.

Die rechte Spalte ist für Informationen und das Debbuging vorbereitet.

Oberhalb der rechten Spalte ist ein Knopf platziert, der zum Speichern der Änderungen verwendet wird. Sobald eine Änderung im Flow erfolgt, wird dieser Knopf aktiviert und mit roter Farbe markiert. Mit Klick auf diesen Knopf mit der Beschriftung DEPLOY werden die Änderungen gespeichert und aktiviert (Abbildung 6.7).

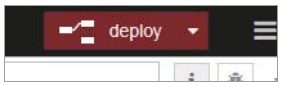

Abb. 6.7: Node-Red: Änderungen speichern mit »deploy«

In Abbildung 6.6 sehen Sie den ersten Flow-Tab, der mit NODE-RED betitelt ist.

Im Flow selbst sind drei Nodes sichtbar, die miteinander verknüpft sind.

Werkzeugleiste

Die Werkzeugliste oder Node-Palette listet die einzelnen verfügbaren Nodes in Kategorien sortiert auf.

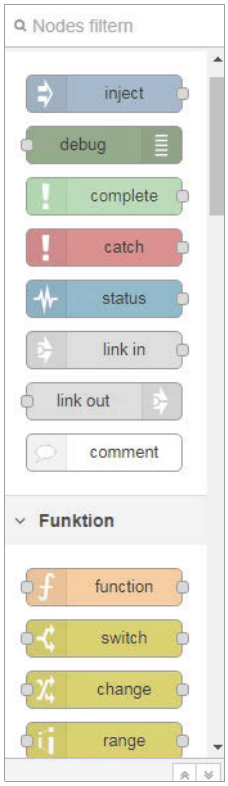

Abb. 6.8: Node-Red: Werkzeugpalette

Mit der Standard-Installation von Node-Red wird ein Standard-Set von Nodes mitgeliefert. Weitere Nodes können aus dem Internet geladen werden.

Über das Menü oben rechts kann die Menüstruktur der Node-Red-Verwaltung aufgerufen werden (Abbildung 6.9).

Unter dem Menüpunkt PALETTE VERWALTEN gelangen Sie in die Node-Verwaltung. Im Register INSTALLIEREN können Sie nach Nodes suchen. Im Beispiel suchen wir nach dem Dashboard-Node (Abbildung 6.10). Das Dashboard werden wir uns später noch im Detail anschauen.

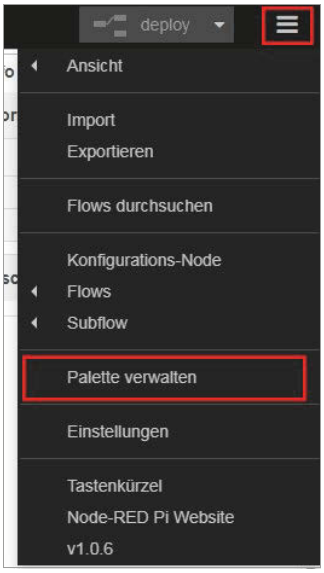

Abb. 6.9: Node-Red: Node-Palette verwalten

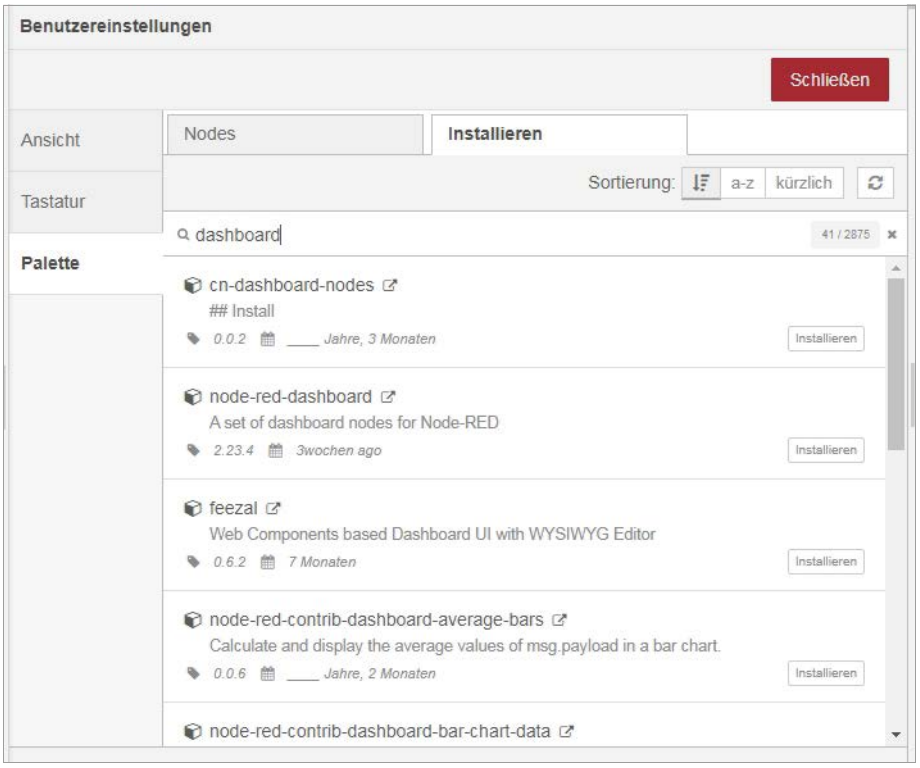

Abb. 6.10: Node-Red: verfügbare Nodes zum Installieren

Den gesuchten Node-Eintrag können Sie durch Klick auf INSTALLIEREN zur Installation wählen.

Nach der Bestätigung der Installation wird der gewählte Node installiert und in der Werkzeugpalette verfügbar gemacht (Abbildung 6.11).

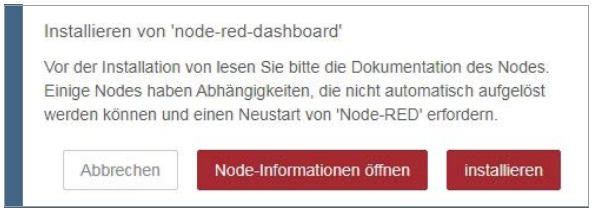

Abb. 6.11: Node-Red: Installation von Node bestätigen

Debug-Fenster

Das Debug-Fenster in der rechten Spalte beinhaltet sowohl ein Informationsregister als auch ein Debug-Register.

Im Debugging können Ausgaben, die via Debug-Node ausgegeben werden, dargestellt werden.

In Abbildung 6.12 ist die Debugging-Ausgabe für den ersten Flow dargestellt. Hier wird im Sekundentakt eine Ausgabe gemacht.

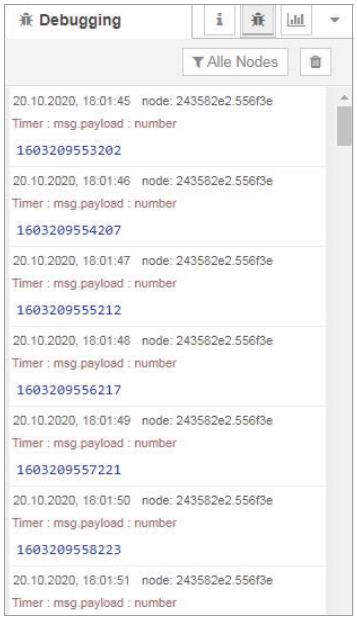

Abb. 6.12: Node-Red: Debugging-Ausgabe

Diese Debugging-Ausgabe ist sehr nützlich bei der Flow-Erstellung und zum Überwachen des Datenflusses.

6.4 Flows mit Node-Red

Ein Flow in Node-Red besteht aus mehreren Nodes, die miteinander verbunden sind (Abbildung 6.13).

Abb. 6.13: Node-Red: Flow mit Nodes

Hinter jedem Node versteckt sich eine Funktion beziehungsweise Aktion. Durch das Zusammenschalten der einzelnen Nodes kann man einfache oder komplexe Informationsflüsse realisieren.

Einen einzelnen Node können Sie mittels Drag&Drop aus der Werkzeugpalette auf den mittleren Flow-Bereich ziehen.

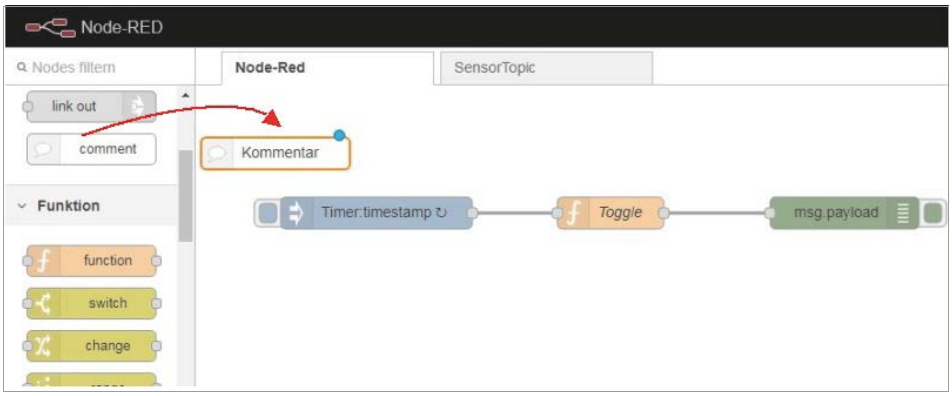

Abb. 6.14: Node-Red: Node wählen mit Drag&Drop

Im Flow-Bereich können Sie dann den Node nach Wunsch platzieren.

Der Beispiel-Node in Abbildung 6.14 ist ein Kommentar. Ein Kommentar ist nie mit einem anderen Node verbunden.

Platziert man nun zwei einzelne Nodes, die dann im Flow zusammengehören, muss anschließend eine Verbindung geschaffen werden.

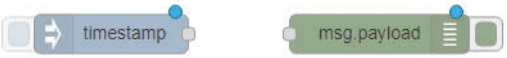

Abb. 6.15: Node-Red: Nodes ohne Verbindung

Dazu müssen die in Abbildung 6.16 rot markierten Verbindungsstellen miteinander verbunden werden.

Abb. 6.16: Node-Red: Nodes verbinden

Mit gedrückter Maustaste (meist die Taste mit dem Zeigefinger) klicken Sie auf den ersten Verbindungspunkt. Wenn Sie nun die Taste gedrückt lassen und die Maus verschieben, sehen Sie den orange Verbindungsfaden. Führen Sie den Mauszeiger nun auf den zweiten Verbindungspunkt, bis dieser orange markiert ist.

Abb. 6.17: Node-Red: Nodes verbinden

Nun können Sie die Maustaste loslassen und die Verbindung zwischen den beiden Nodes ist erstellt (Abbildung 6.18).

Abb. 6.18: Node-Red: Nodes sind verbunden.

Hinweis

Beim Erstellen von Flows sind auf den einzelnen Nodes teilweise blaue Punkte sichtbar. Die blauen Punkte geben an, dass diese Nodes noch nicht gespeichert wurden. Sobald man mittels `deploy` den Flow speichert, verschwinden die blauen Punkte.

Eigenschaften der Nodes

Mit dem Platzieren einzelner Nodes und dem Verbinden mit anderen Objekten ist es meist nicht getan. Hinter den einzelnen Nodes aus der Werkzeugpalette verstecken sich oft umfangreiche Funktionen, die über die Eigenschaften des jeweiligen Nodes konfiguriert werden können.

Mit Klick auf einen Node, im Beispiel der `timestamp`-Node, öffnet sich das Eigenschaften-Fenster und verschiedene Konfigurationsmöglichkeiten werden aufgelistet (Abbildung 6.19).

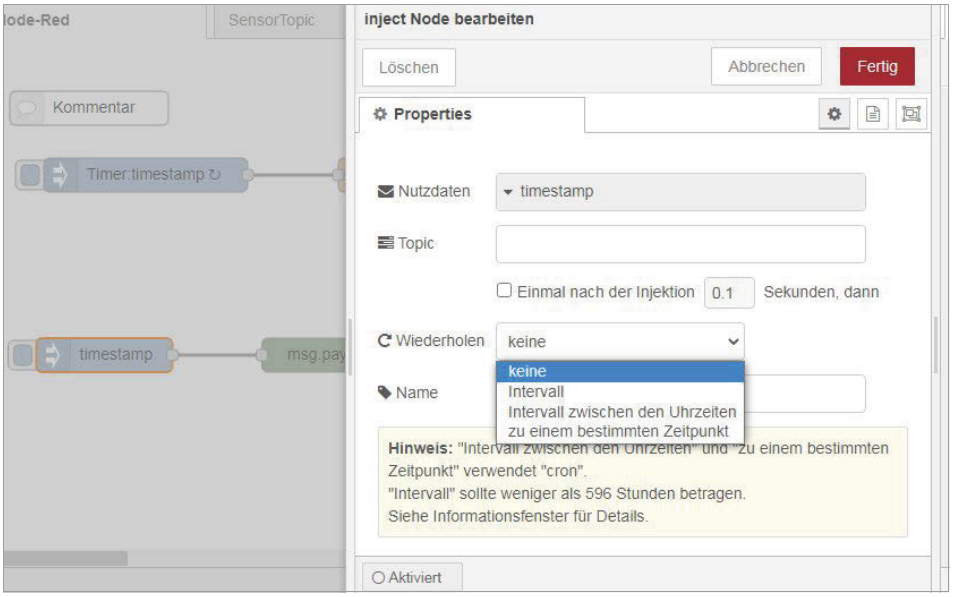

Abb. 6.19: Node-Red: Eigenschaften des Nodes

Der Node `timestamp` kann beispielsweise als Taktgeber (Intervall) mit einstellbarer Zeit verwendet werden (Abbildung 6.20).

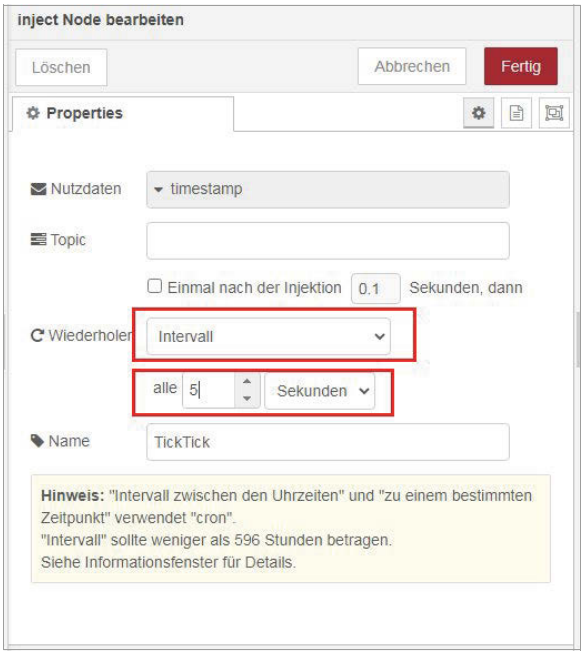

Abb. 6.20: Node-Red: Node-Eigenschaften für Intervall-Timer

Jeder Node besitzt unterschiedliche Funktionen und Eigenschaften. In der rechten Info-Spalte werden beim Klicken auf einen einzelnen Node zusätzliche Informationen dargestellt.

Mehrere Flows

Ein Flow einer Anwendung kann schnell aus vielen Nodes und entsprechend vielen Verbindungen zwischen ihnen bestehen.

Hat man verschiedene Datenflüsse, lohnt es sich, diese etwas zu strukturieren.

Ein hilfreiches Mittel ist dabei, die einzelnen Flows in einzelnen Tabs zu speichern.

Mit dem Pluszeichen am rechten Rand des Flow-Bereichs kann ein neuer Tab oder Reiter erstellt werden (Abbildung 6.21). Idealerweise gibt man dem Reiter einen sinnvollen Namen.

Abb. 6.21: Node-Red: neuen Tab oder Reiter erstellen

Flows mit JavaScript erweitern

Viele Funktionen können in Node-Red durch Zusammenstellen realisiert werden. Dabei kann der Anwender die vielen Konfigurationsmöglichkeiten der einzelnen Nodes nutzen. Informationen fließen dabei von Node zu Node. Viele Nodes besitzen dazu Ein- und Ausgänge.

Node-Red ist bekanntlich eine webbasierte Anwendung. Die Darstellung und die Funktion der einzelnen Nodes erfolgen mittels der bekannten Webtechnologien HTML und JavaScript. JavaScript als Skript-Sprache wird im Browser des Anwenders ausgeführt und ein Entwickler kann eigenen JavaScript-Code in seine Webseite einbauen.

Mit eigenem JavaScript-Code können Sie somit die Funktionalität Ihrer Flows erweitern oder anpassen. Der für solche Funktionserweiterungen vorgesehene Node ist in der Werkzeugleiste mit FUNCTION bezeichnet (Abbildung 6.22).

Abb. 6.22: Node-Red: Funktions-Node

Dieser Funktions-Node hat standardmäßig einen Eingang und einen Ausgang.

In Abbildung 6.23 ist der ganze Flow zum Toggeln dargestellt.

Abb. 6.23: Node-Red: Toggle-Flow

Nachdem Sie den Funktions-Node auf die Arbeitsfläche gezogen haben, können Sie ihn mit einem Doppelklick bearbeiten. Es öffnet sich ein Eigenschaften-Fenster, über das Sie einen Titel und Funktionscode eingeben können (Abbildung 6.24). Am unteren Ende des Fensters kann zusätzlich die Anzahl der Ausgänge angegeben werden.

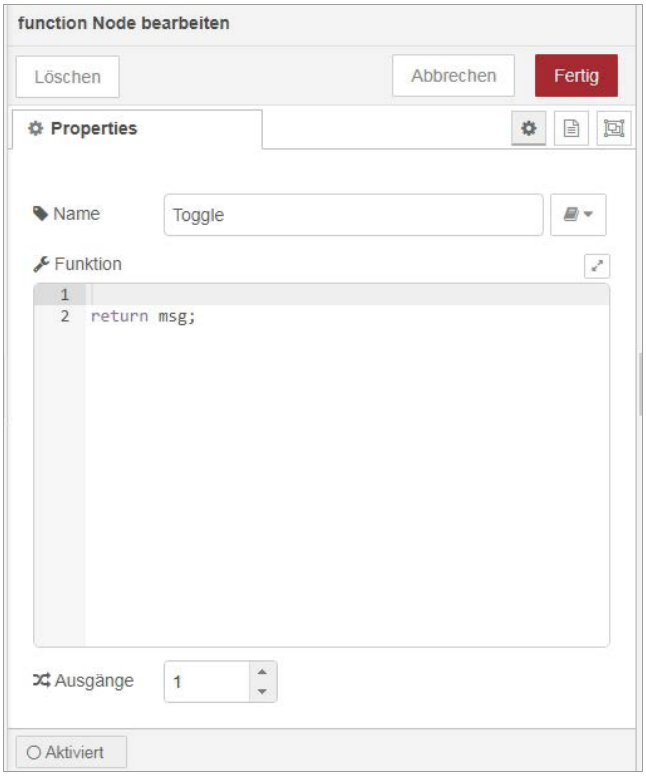

Abb. 6.24: Node-Red: Funktions-Node bearbeiten

Der Funktionsblock in diesem Beispiel wird mit TOGGLE betitelt und soll einen Input vom Inject-Node, der als Intervall läuft, auswerten und jeweils den Ausgang von 1 auf 0 und zurück schalten (Vorgang wird auch als Toggeln bezeichnet).

Der lauffähige Code ist im Codebereich eingetragen und gespeichert. (Abbildung 6.25).

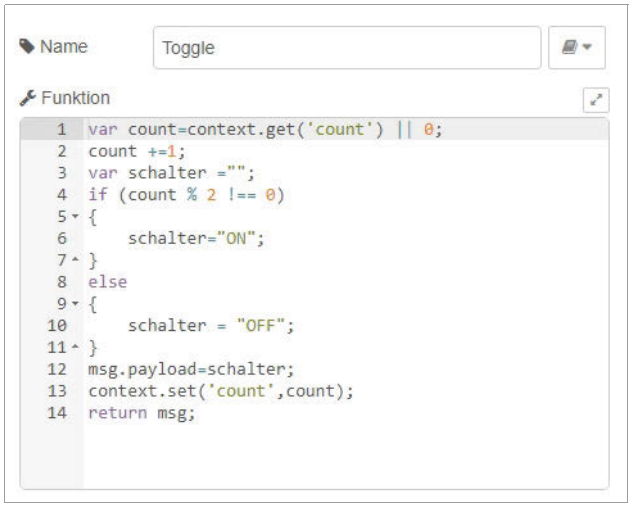

Abb. 6.25: Node-Red: JavaScript-Code für Toggle

Über den Debug-Node kann nun die Ausgabe im Debug-Fenster überprüft werden. Die Ausgabe toggelt zwischen den Werten ON und OFF (Abbildung 6.26).

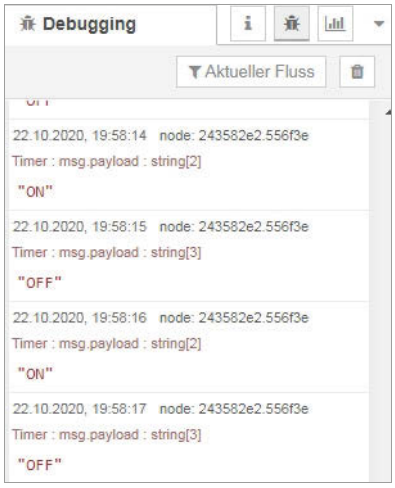

Abb. 6.26: Node-Red: Debug-Ausgabe

In diesem JavaScript-Code wird zuerst eine Variable `count` deklariert, falls diese noch nicht existiert, und ein Wert 0 zugewiesen (`smarthome_kap6_funktion_toogle.txt`):

```
var count=context.get('count') || 0;
```

Nun wird der Wert von `count` um 1 erhöht:

```
count +=1;
```

In der Variablen wird ein leerer String gespeichert:

```
var schalter ="";
```

Jetzt wird mit der Modulo-Funktion geprüft, ob die Zahl gerade ist. Dies ist der Fall beim Resultat 0.

Wenn das Resultat 0 ist, wird in der Variablen `schalter` ein String mit `ON` gespeichert, andernfalls ist der String-Wert `OFF`:

```
if (count % 2 !== 0)
{
    schalter="ON";
}
else
{
    schalter = "OFF";
}
```

Die Payload, die sogenannte Nutzlast, beinhaltet den String mit der zu übertragenden Nachricht. In diese Payload wird nun der Zählerstand gespeichert:

```
msg.payload=count;
```

Der Zählerstand wird im Context-Store der Variablen `count` zugewiesen:

```
context.set('count',count);
```

Zum Schluss wird die Nachricht `msg` zurückgegeben und gelangt zum nächsten Node. In diesem Fall zum Ausgabe-Node:

```
return msg;
```

Das Eingabefeld für den Code ist nicht nur eine Texteingabe, sondern beinhaltet eine Code-Validierung.

Falls ein Fehler in der Syntax vorhanden ist, wird dies mit verschiedenen Zeichen ausgegeben. In Abbildung 6.27 ist die Modulo-Funktion fehlerhaft. Es fehlt die Zahl 0 für den korrekten Vergleich.

Abb. 6.27: Node-Red: JavaScript-Code mit Fehler

6.5 MQTT mit Node-Red

Das Einlesen und Ausgeben von MQTT-Topics ist recht simpel und kann mit den beiden Nodes `mqtt in` und `mqtt out` realisiert werden (Abbildung 6.28).

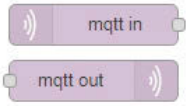

Abb. 6.28: Node-Red: MQTT-Nodes

Das Abonnieren und Ausgeben eines Topics, in unserem Fall des bereits getesteten Topics `SensorTopic`, ist in Abbildung 6.29 abgebildet.

Abb. 6.29: Node-Red: MQTT-Flow

Mit Klick auf den MQTT-Node können die Eigenschaften eingetragen werden. In diesem Fall sind das der MQTT-Server und der Name des Topics (Abbildung 6.30).

Die weiteren Einstellungen in diesem Fenster können mit den Standard-Einstellungen übernommen werden.

Das Feld QoS bezeichnet das MQTT-Feature `Quality of Service`. Damit wird die Zuverlässigkeit der Nachrichtenzustellung definiert. MQTT kennt drei Stufen, die mit 0, 1 oder 2 definiert werden. Genau diese drei Werte können im Eigenschaften-Fenster des MQTT-Servers eingetragen werden.

Die drei Stufen des QoS sind in Tabelle 6.2 beschrieben:

Quality-of-Service-Stufe	Beschreibung
0	Nachrichtenversand einmalig Keine Prüfung, ob erhalten
1	Nachricht wird versendet Empfänger bestätigt dem Sender Nachrichtenempfang
2	Nachricht wird versendet Nachricht kommt garantiert an und garantiert nur einmal

Tabelle 6.2: MQTT – Quality of Service (QoS)

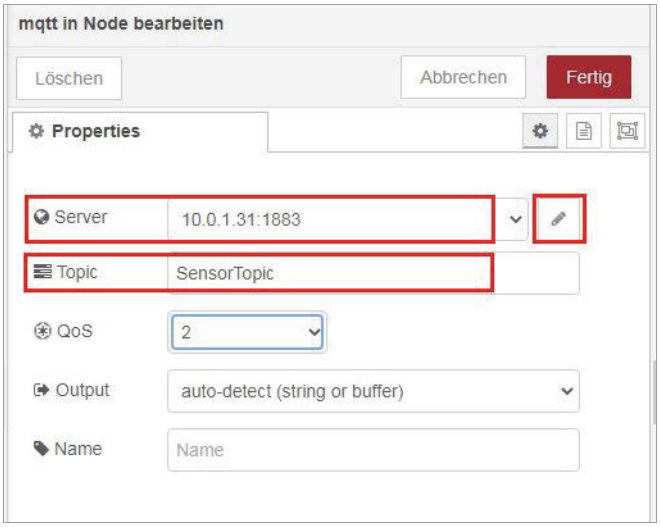

Abb. 6.30: Node-Red: MQTT-Topic

Über das Auswahlmenü kann ein bestehender Server ausgewählt oder ein neuer Server erfasst werden (Abbildung 6.31).

Abb. 6.31: Node-Red: MQTT-Server

Ein bestehender Server kann mittels der Editier-Funktion mit dem Stift bearbeitet werden (Abbildung 6.32).

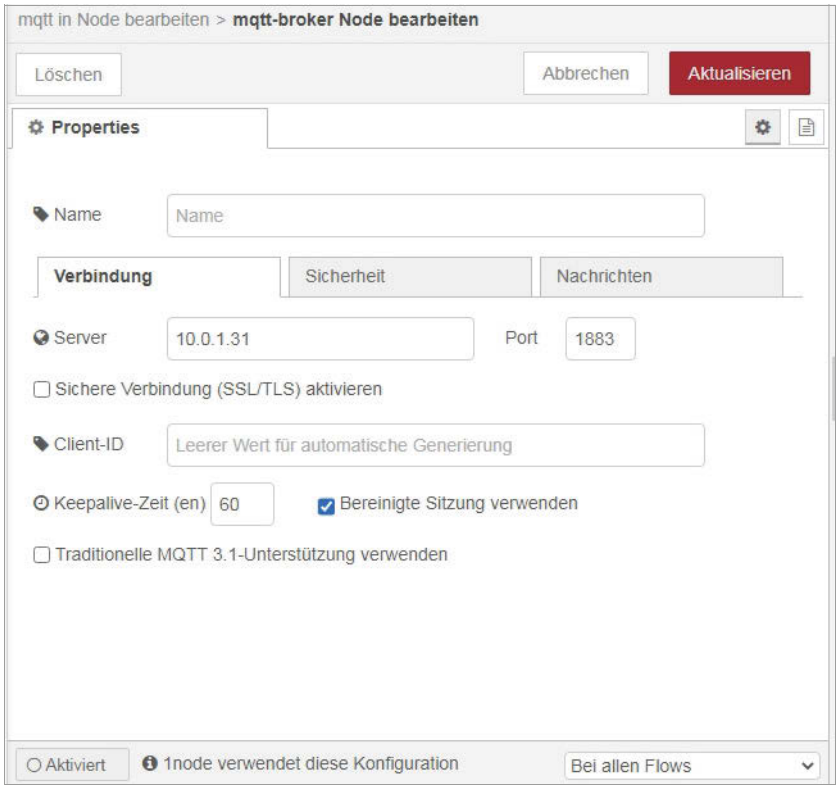

Abb. 6.32: Node-Red: MQTT-Server bearbeiten

Wird nun über das Terminal ein Wert auf den Topic `SensorTopic` publiziert (Abbildung 6.33), wird dieser dann umgehend im Debug-Fenster von Node-Red ausgegeben (Abbildung 6.34).

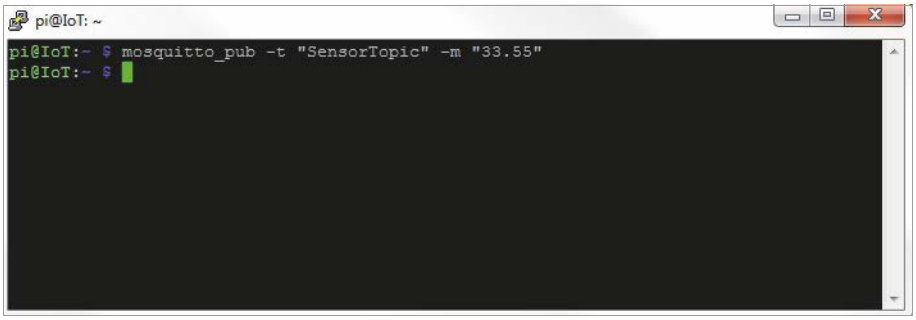

Abb. 6.33: Node-Red: Publizieren von Topic-Daten

Abb. 6.34: Node-Red: Abonnierter Topic im Debug-Fenster

In der Praxis wird eine von einem Topic empfangene Nachricht im Flow weiterverarbeitet oder an eine Ausgabefunktion weitergegeben.

Im nachfolgenden Abschnitt wird das Node-Red-Dashboard in Betrieb genommen. Über das Dashboard können Sensordaten übersichtlich dargestellt werden.

6.6 Node-Red-Dashboard

Das Node-Red-Dashboard ist eine nützliche Erweiterung, die in keiner Node-Red-Installation fehlen darf.

Nach der Installation der Dashboard-Palette, wir haben diese bereits bei der Einführung von Node-Red (Abschnitt 6.3) installiert, stehen Ihnen in der Werkzeugpalette Nodes zur Erstellung von Bedienelementen und Anzeigen zur Verfügung (Abbildung 6.35).

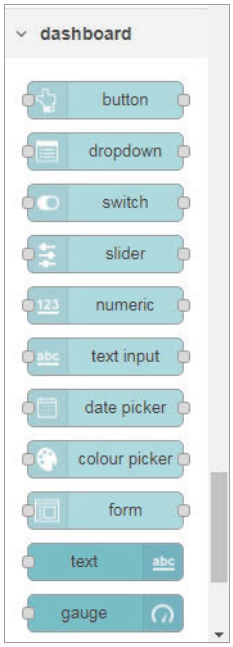

Abb. 6.35: Node-Red: Werkzeugpalette des Dashboards

Diese einzelnen Nodes für Eingabe oder Ausgabe auf Ihrem Informationsbildschirm können wie alle Nodes in den eigenen Flow integriert werden.

In Abbildung 6.36 ist ein Flow dargestellt, der einen Schiebeschalter (Sonoff) als Eingabeelement und zwei Anzeigeelemente in Form eines Zeigerinstruments (Gauge) und eines Grafik-Diagramms (On/Off) enthält.

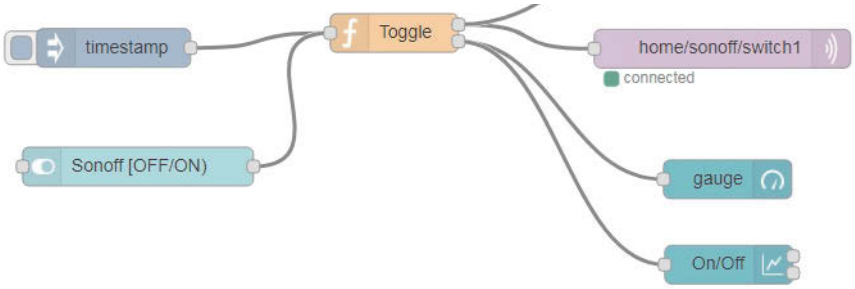

Abb. 6.36: Node-Red: Flow mit Dashboard-Nodes

Im Dashboard schlussendlich sehen die Ein- und Ausgabe-Nodes gemäß Abbildung 6.37 aus. Mit dem Schiebeschalter kann ein Sonoff-Schaltelement ein- und ausgeschaltet werden.

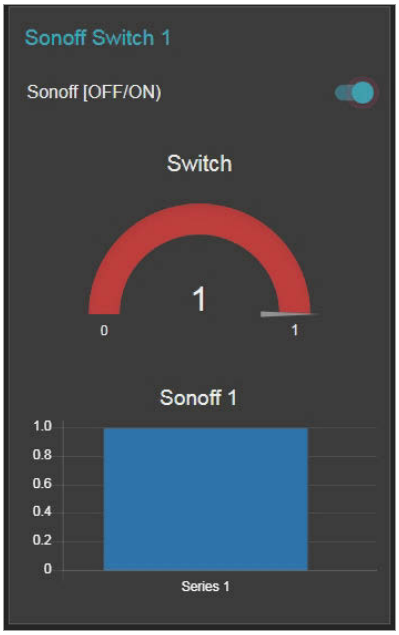

Abb. 6.37: Node-Red: Dashboard

Aufruf Dashboard

Das Dashboard wird über die folgende Adresse aufgerufen:

`http://IP-des-Raspberry:1880/ui/`

Beim Erstaufruf ohne konfiguriertes Dashboard erscheint ein Informationsbildschirm (Abbildung 6.38).

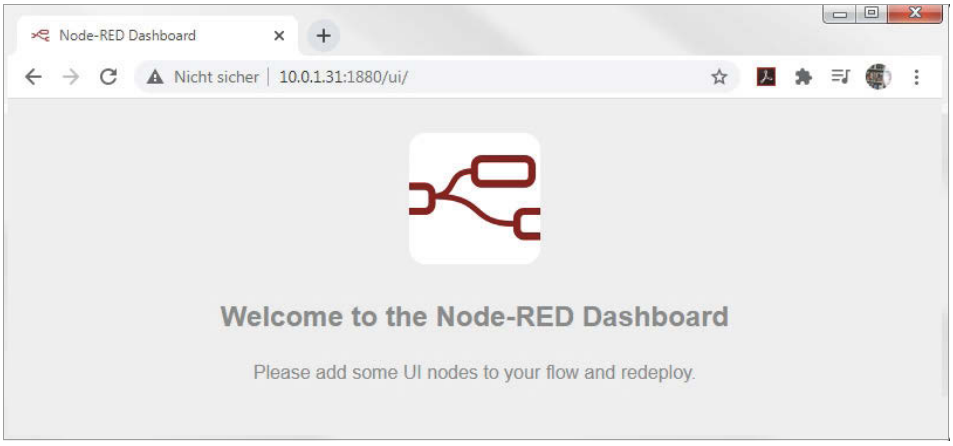

Abb. 6.38: Node-Red: Dashboard-Startseite

Dashboard erstellen

Bei der Erstellung eines Dashboards wird im ersten Schritt immer ein Flow mit den nötigen Eingabe- und Ausgabe-Nodes erstellt.

Im ersten Beispiel wird ein Schalter auf dem Dashboard platziert. Der Status (Ein/Aus) wird auf einem Zeigerinstrument angezeigt. Für den Flow benötigt man einen Switch-Node und einen Gauge-Node (Abbildung 6.39).

Abb. 6.39: Node-Red: Dashboard-Flow

Mit Doppelklick auf den Schalter-Node können Sie die Eigenschaften bearbeiten.

Der Node bekommt die Bezeichnung `Ein/Aus`, die über das Label erfasst werden. Für die Anzeige auf dem Dashboard muss das Element einer `ui_group` (Group) zugewiesen werden. Dies ist die Zuordnung zum jeweiligen Dashboard (Abbildung 6.40).

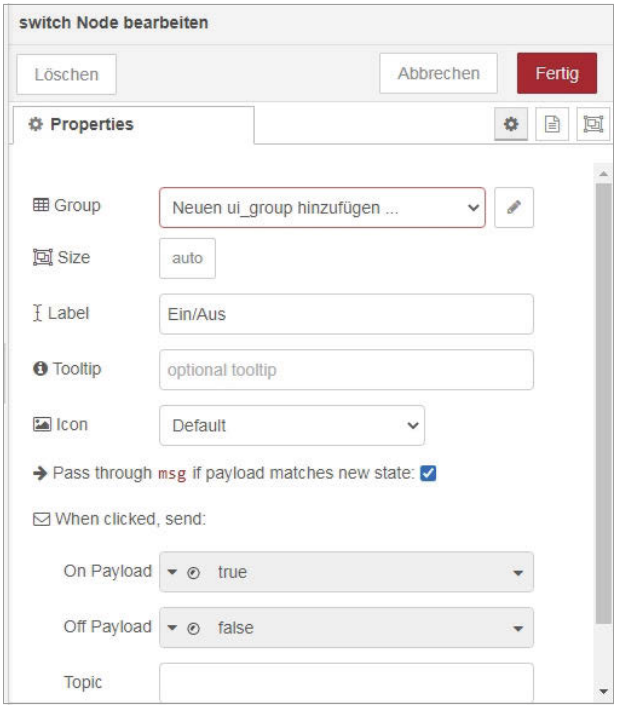

Abb. 6.40: Node-Red: Eigenschaften-Switch

Darum muss zuerst mit dem Stiftzeichen eine neue Gruppe definiert werden. Wir nennen sie `Smarthome`.

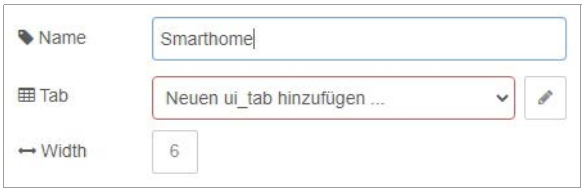

Abb. 6.41: Node-Red: Gruppe

Der nächste Schritt ist die Erstellung eines Tabs, also eines Menüpunkts in der Navigation, über den das Dashboard mit dem Schalter erreicht werden kann.

Mit dem Stiftzeichen kann ein neues `ui_tab` erstellt werden.

Wir nennen es `Home`.

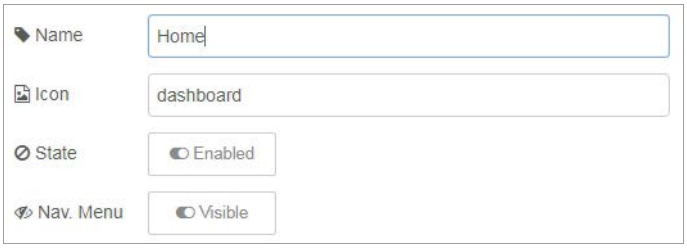

Abb. 6.42: Node-Red: UI-Tab erstellen

Nach Hinzufügen und FERTIG ist die Gruppe und der Navigationspunkt erstellt.

Nun werden noch die Werte in der Payload definiert. Dazu verwenden wir Zahlenwerte (Number). Bei `On Payload` wird eine 1, bei `Off Payload` eine 0 geliefert (Abbildung 6.44).

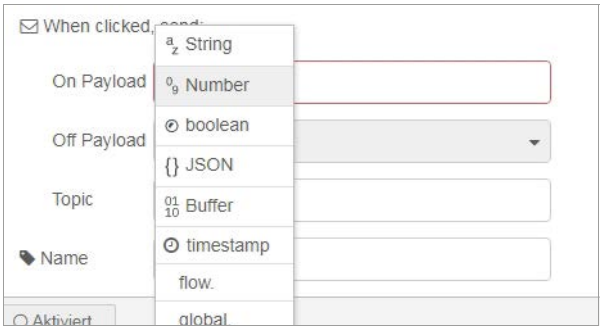

Abb. 6.43: Node-Red: Payload-Werte I

Abb. 6.44: Node-Red: Payload-Werte II

Nun kann der Flow mit DEPLOY gespeichert werden.

In der DEBUG-Spalte unter DASHBOARD findet man den Tab HOME mit der Node-Gruppe SMARTHOME (Abbildung 6.45).

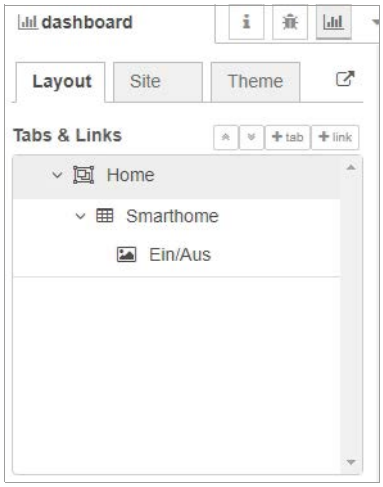

Abb. 6.45: Node-Red: Dashboard-Struktur

Nun wird noch der Gauge-Node mit einem Doppelklick bearbeitet.

Der Node bekommt das Label SCHALTER und der Wertebereich (RANGE) wird von 0 bis 1 gesetzt.

Die Gruppenzuordnung zu SMARTHOME belassen wir. So erscheint das Zeigerinstrument im gleichen Dashboard und in der gleichen Gruppe wie der Schalter.

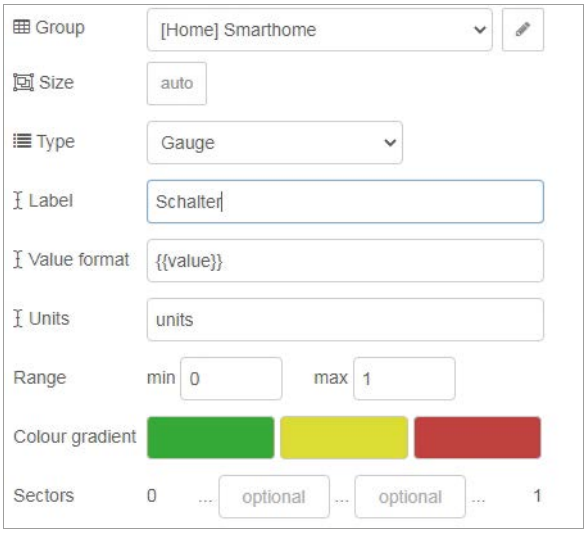

Abb. 6.46: Node-Red: Gauge-Node, Eigenschaften setzen

Nach dem Speichern kann das Dashboard aufgerufen werden. Dazu geben Sie die Adresse im Browser ein oder verwenden aus der Dashboard-Struktur-Anzeige den Link mit dem Pfeil (Abbildung 6.47).

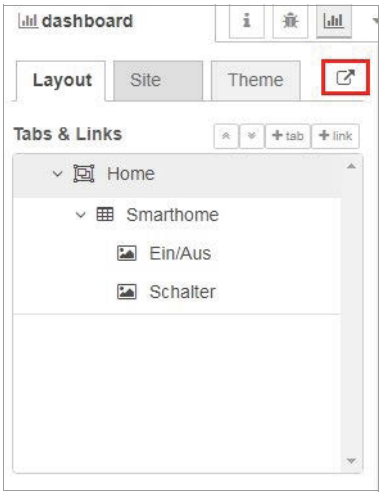

Abb. 6.47: Node-Red: Aufruf Dashboard

Jetzt öffnet sich ein neuer Browser-Reiter und das Dashboard wird angezeigt (Abbildung 6.48).

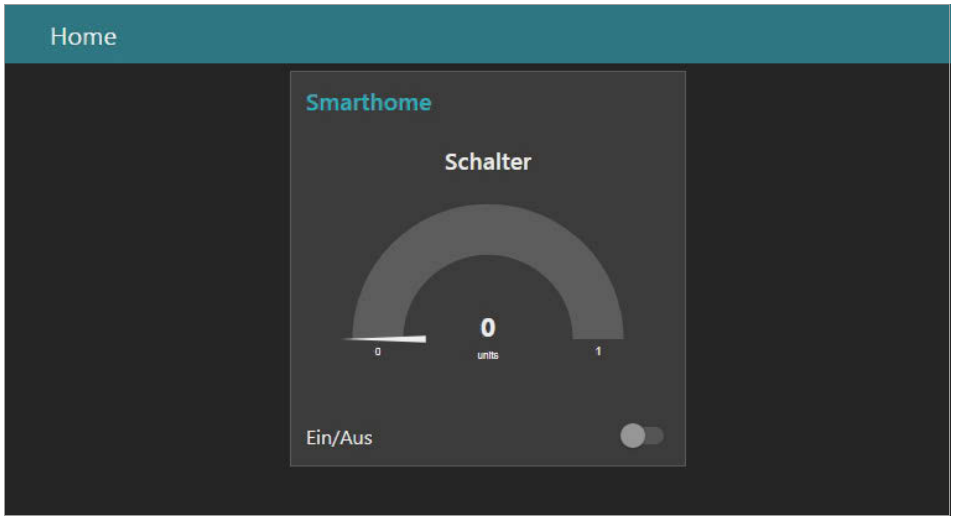

Abb. 6.48: Node-Red: Dashboard mit Schalter (AUS)

Mit der Betätigung des Schiebeschalters kann der Schalter eingeschaltet werden (Abbildung 6.49).

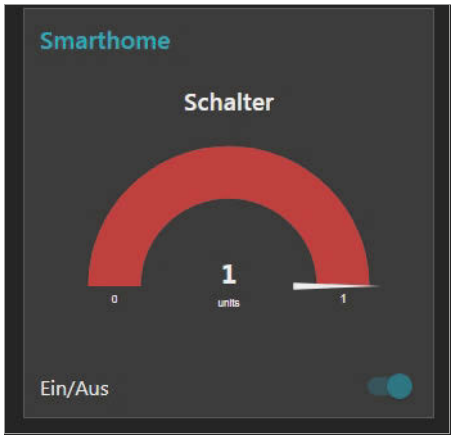

Abb. 6.49: Dashboard mit Schalter (EIN)

Damit ist das erste Dashboard erstellt.

Die Schalterbetätigung hat in diesem Flow noch keine Funktion. Hierzu muss der Fluss mit einem Ausgang eines Boards oder einem Topic (`mqtt out`) verbunden werden.