Message- Grundlage von Node-RED

Node-RED ist eine auf Flow-basierte Programmierung, d.h.eine strukturierte Kommunikation zwischen Datenblöcken (Messages). Diese Nachrichten laufen von Node zu Node, wobei jede Nodes diese Daten verändern, eränzen oder reagieren. Dazu braucht es einen einheitlichen Ansatz, eine übergreifende Struktur der Nachrichten.

JSON

In Node-RED werden dazu die JavaScript-Objekte (JSON) verwendet. Das JSON-Format speichert Daten strukturiert und textbasiert und hat Ansätze der objektorientierten Programmierung.

Dabei gibt 2 Strukturen:

Name-Wert-Paare:
{"Marke": "Citroen","Baujahr": 2022}

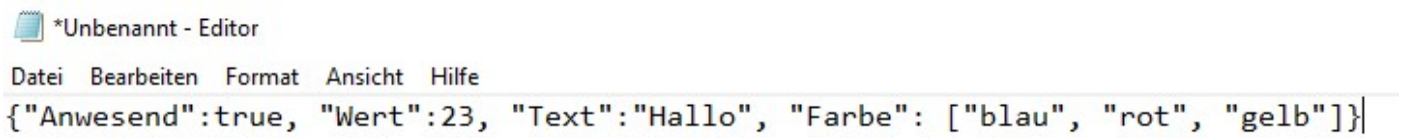
geordnete Liste (Tabellen) von Werten
{"Farbe": ["braun", "lila", "weiß"]}

Bei JSON gibt es folgende Datentypen:

Boolcher Wert:	{Antwort“:true}	(true / false)
Zahl:	{„Temp“:23.1}	(keine führende Null, Dezimalpunkt)
Zeichenkette:	{„Name“:“Axel“}	(hochgestellte Anführungszeichen)
Tabelle:	{"Farbe": ["braun", "lila", "weiß"]}	(beginnen mit [und enden mit])

Art der Darstellung:

Texteditor:



Browser:

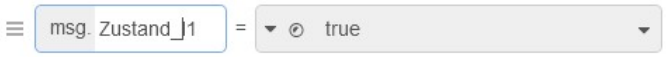
JSON-Formatter:

Unter https://jsonformatter.org kann die Datei eingelesen werden und wird dann übersichtlich dargestellt.
Hier können ebenfalls JSON-Dateien auf ihre Gültigkeit überprüft werden.

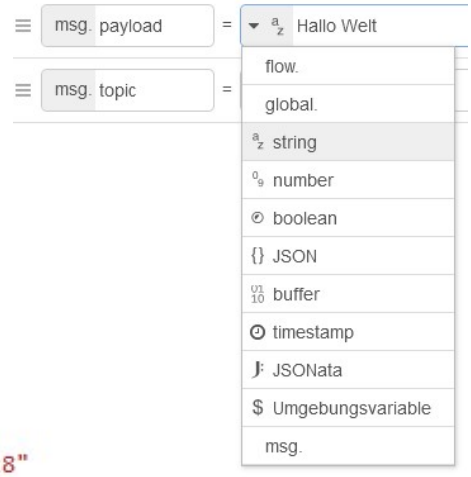
Message bei Node-RED

Das JSON-Objekt für eine Node-RED-Message besteht im einfachsten Fall aus einer *_msgid* mit einer eindeutigen Nummer, die die Nachricht kennzeichnet. Im Beispiel gibt es noch *payload* mit der Nachricht „Hello World“. Der Eintrag topic ist leer, denn er wurde zwar deklariert aber nicht mit Daten gefüllt.

Eine Message kann aber noch weitere Eigenschaften haben, die in ausgewählten Nodes (inject, change…) festgelegt werden können.



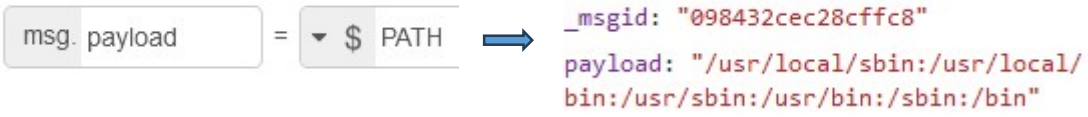
Ein Wertepaar kann z.B. die Auswertung eines Eingangs vom TXT 4.0 sein. Hier ist der Wert ein boolscher Wert.



Möglichkeiten wären:

flow/global: Kontextvariablen
string: Zeichenkette
number: Zahl; Deimaltrennzeichen als Punkt
boolean: wahr oder falsch
JSON: ein JSON-Objekt; (Eingabe über die drei Punkte am Ende)
buffer: es werden Daten Byte für Byte in den Datenpuffe gelesen.
timestamp: Sekunden seit dem 01 Januar 1970
JSONNata: Abfrage und Transformationssprache für JSON-Dateien
Umgebungsvariable: konfigurierbare Variablen in Betriebssystemen

Beispiel für PATH:



Hinweise:

weitere Message erstellen mit: 

Message löschen: