

## function



Function-Nodes können beliebigen Java-Script-Code enthalten und sie dienen zur Filterung und Manipulation von Nachrichten, die zwischen den Knoten ausgetauscht werden.

Node 'function' bearbeiten

Löschen Fertig

Eigenschaften

Name: function 5

Setup Start Funktion Stopp

Ausgänge: 1

Name gibt den Namen der Node an.

Im Reiter Setup wird die Anzahl der Ausgänge festgelegt.

Name: function 5

Setup Start Funktion Stopp

```
1 // Der Code hier wird ausgeführt,  
2 // wenn der Node gestartet wird  
3
```

Setup Start Funktion Stopp

```
1 msg.payload = Math.round(Math.random() * 100);  
2 return msg;
```

Setup Start Funktion Stopp

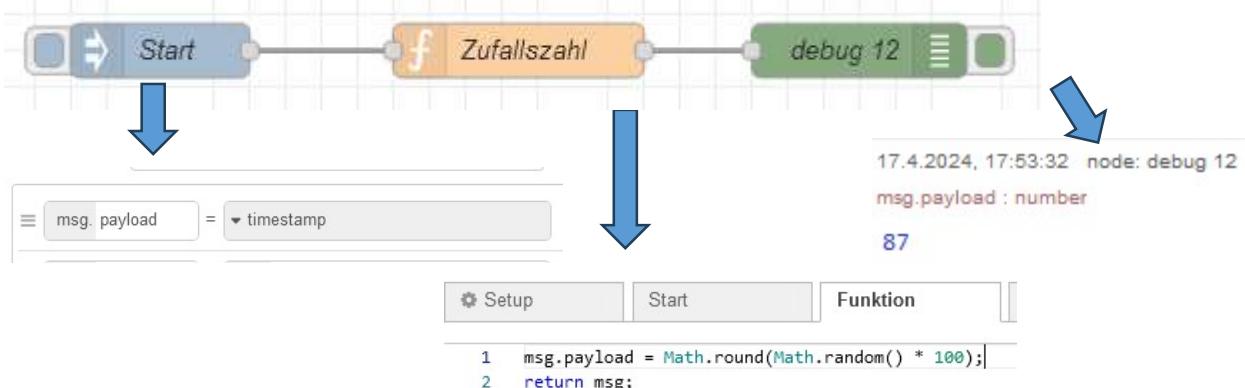
```
1 // Der Code hier wird ausgeführt,  
2 // wenn der Node gestoppt wird  
3
```

Hier liegt der Code, der bei einem Deployment des Node ausgeführt wird.

Im Reiter Funktion wird der Java-Script-Code eingefügt.

Hier steht der Code, der bei einem Stopp von Node-RED ausgeführt wird.

### Beispiel 1: Erzeugung und Ausgabe einer Zufallszahl



### Hinweis: Funktionen speichern

Name: function 5

Setup Start Funktion Stopp

```
1 msg.payload = Math.round(Math.random() * 100);  
2 return msg;
```

Einmal geschriebene Funktionen können in einer Bibliothek gespeichert werden. Dazu wird das Buch angeklickt und Bibliothek speichern ausgewählt.

Im folgenden Fenster kann nun ein Verzeichnis

In Bibliothek speichern ...

Lokal

function

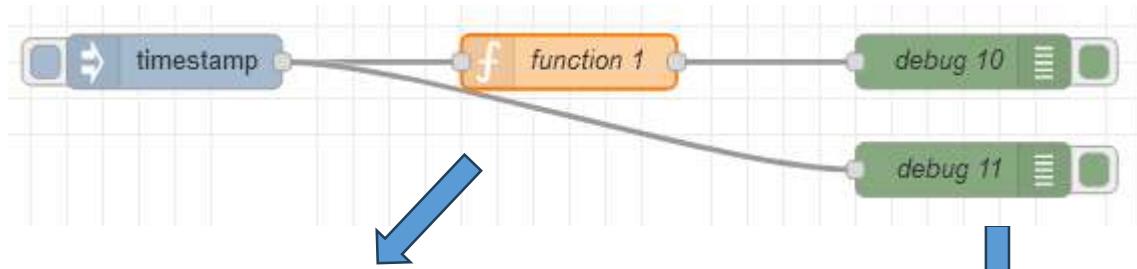
test

Fehlerfunktion.js

und ein Name für die Funktion vergeben werden.

Der Abruf der Funktion erfolgt dann über Bibliothek öffnen...

**Beispiel 2: Ausgabe von Timestamp (Anzahl der Millisekunden seit dem 01.01.1970) und der umgerechneten Echtzeit.**



```

1 const timestamp = new Date(msg.payload);
2 const year = timestamp.getFullYear();
3 let month = timestamp.getMonth();
4 month = (month < 9 ? '0' : '') + (month + 1);
5 let day = timestamp.getDate();
6 day = (day < 10 ? '0' : '') + day;
7
8 let timestring = year + '-' + month + '-' + day + ' ';
9 timestring += timestamp.toLocaleTimeString();
10 msg.payload = timestring;
11 return msg;
  
```

```

17.4.2024, 17:45:40 node: debug 11
msg.payload : number
1713368740008
17.4.2024, 17:45:40 node: debug 10
msg.payload : string[19]
"2024-04-17 17:45:40"
  
```

**Beispiel 3: gesplittete Ausgabe einer msg.payload in mehrere msg.payload's**

Node 'inject' bearbeiten

Löschen Abbrechen

Eigenschaften

Name Name

msg. payload = Zange,Hammer,Säge

Ausgabe:

```

9.9.2024, 11:14:08 node: source
msg.payload : string[17]
"Zange,Hammer,Säge"
9.9.2024, 11:14:08 node: 1
msg.payload : string[5]
"Zange"
9.9.2024, 11:14:08 node: 2
msg.payload : string[6]
"Hammer"
9.9.2024, 11:14:08 node: 3
msg.payload : string[4]
"Säge"
  
```

Node 'function' bearbeiten

Löschen Abbrechen

Eigenschaften

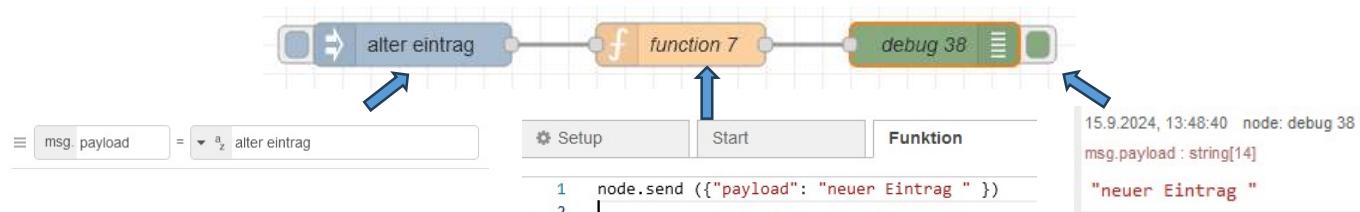
Name Funktion als Splitter

Setup Start Funktion Stopp

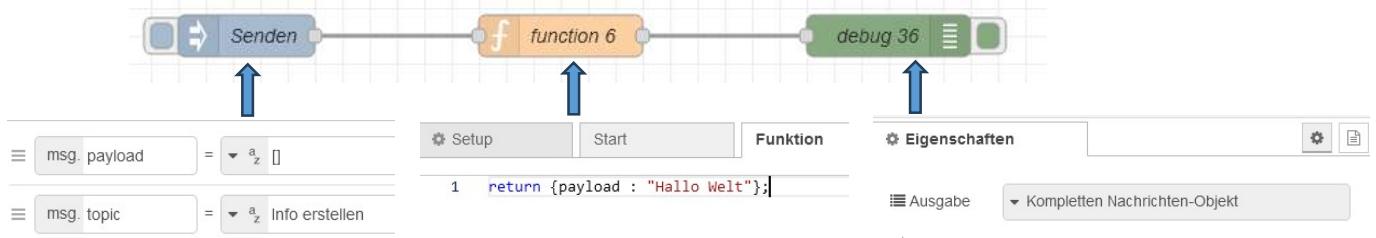
```

1 msg1 = {};
2 msg2 = {};
3 msg3 = {};
4 var array = msg.payload.split(',');
5 a = array[0], b = array[1], c = array[2];
6 msg1.payload = a;
7 msg2.payload = b;
8 msg3.payload = c;
9
10 return [msg1,msg2,msg3];
11
  
```

**Beispiel 4: andere Ausgabe von msg.payload**



### Beispiel 5: Nachrichten erstellen Version 1 (Verlust von msg.topic)

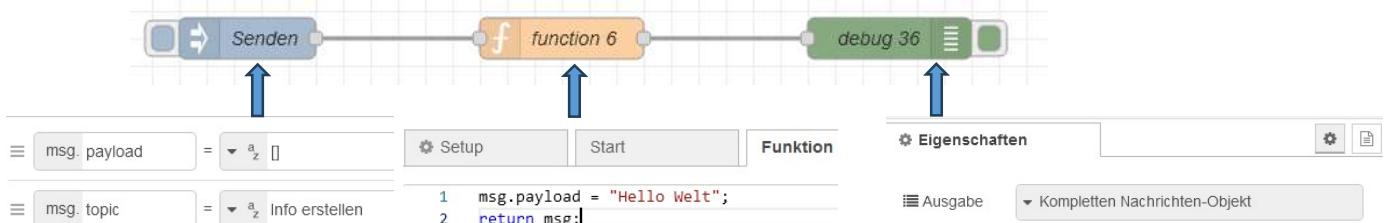


10.9.2024, 08:28:14 node: debug 36

msg : Object  
object  
payload: "Hallo Welt"  
\_msgid: "e3c47fd795ea0969"

Der msg.payload bekommt einen Platzhalter und wird in der function-Node beschrieben.

### Beispiel 6: Nachrichten erstellen Version 2 (msg.topic wird ebenfalls ausgegeben)

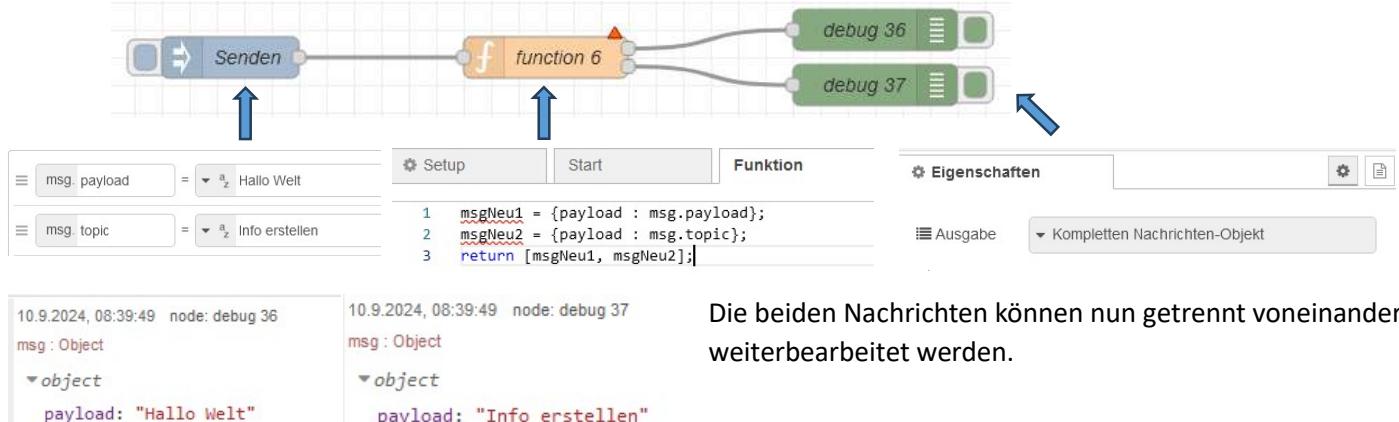


10.9.2024, 08:34:31 node: debug 36

Info erstellen : msg : Object  
object  
\_msgid: "9c29dba38b5301df"  
payload: "Hello Welt"  
topic: "Info erstellen"

Wie Beispiel 5. Allerdings werden alle weiteren Nachrichten ausgegeben.

### Beispiel 7: eingehende Nachricht auf mehrere Ausgangsports



10.9.2024, 08:39:49 node: debug 36

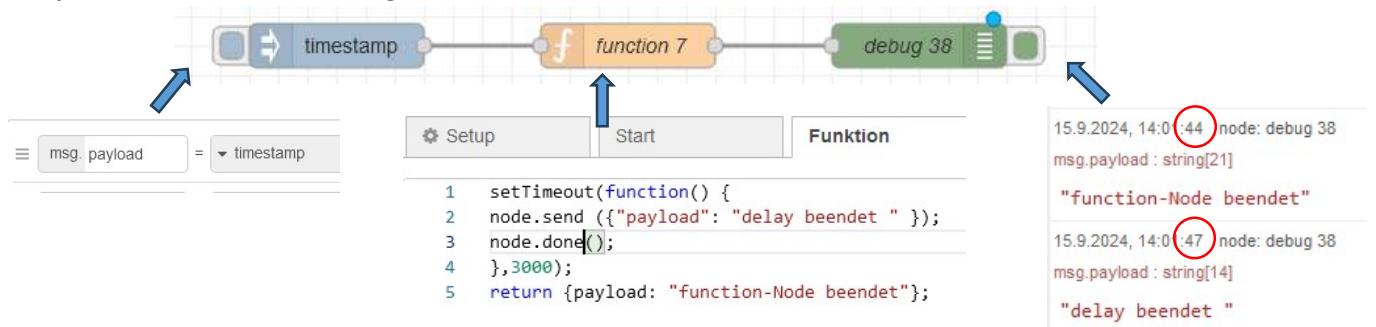
msg : Object  
object  
payload: "Hallo Welt"  
\_msgid: "c09ad8092539c391"

10.9.2024, 08:39:49 node: debug 37

msg : Object  
object  
payload: "Info erstellen"  
\_msgid: "c09ad8092539c391"

Die beiden Nachrichten können nun getrennt voneinander weiterbearbeitet werden.

### Beispiel 8: Eine Nachricht verzögern



setTimeout() bietet eine einstellbare Startverzögerung (3000 ms = 3 Sekunden).

15.9.2024, 14:0:44 node: debug 38

msg.payload : string[21]

"function-Node beendet"

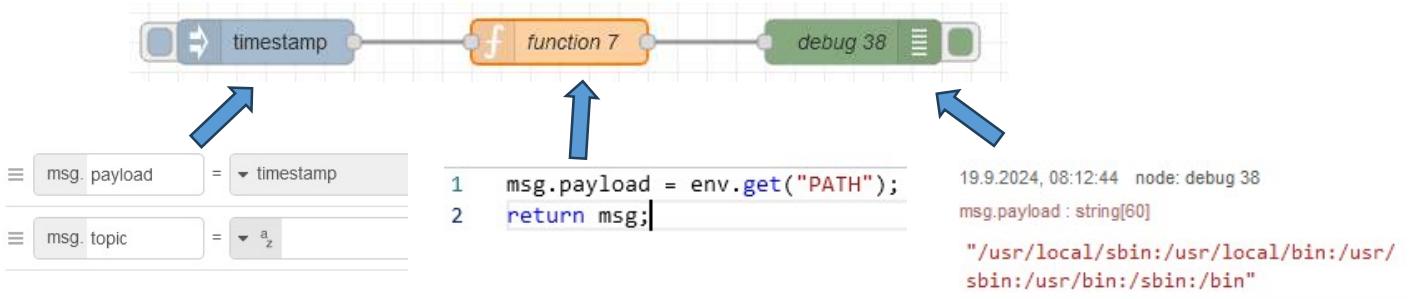
15.9.2024, 14:0:47 node: debug 38

msg.payload : string[14]

"delay beendet "

### Beispiel 9: auf Umgebungsvariablen zugreifen

Mit dem Befehl env können aktuelle Umgebungsvariablen angezeigt werden.



### Beispiel 10 Abfrage für mehrere Ausgänge

