**Änderungen Version 0.2**

* **Switches werden als eigene Kategorie angelegt**
* **„buttonevent“ in Switches generiert 4-stellige id bei Drücken eines Buttons auf der Fernbedienung (kann dann in IPS für eigene Aktivitäten ausgewertet werden, da von Deconz nicht unterstützt zur Zeit**
* **Timer kann gesetzt werden:**
  + **RB\_set\_timer($bridge,"",$instance,$timer,$timer\_on\_off)**
  + **$bridge=Instanz der Bridge**
  + **$instance=Instanz des plugs, lights oder der group**
  + **$timer=format ist “00:00:00“**
  + **$timer\_on\_off=**
    - **“on“ Timer schaltet das Gerät nach Ablauf der Zeit ein**
    - **„off“ Timer schaltet das Gerät nach Ablauf der Zeit aus. Gerät muss vorher eingeschaltet werden ☺**
* **Kompatibel zu IPS 5.1**
* **Deconz Api sollte >=** [2.05.66](https://github.com/dresden-elektronik/deconz-rest-plugin/commit/6683c360fca0884412320c2573ce62c0c2f73ac7)  **sein**
* **Am besten das Modul vor Einspielung des neuen sichern**

**Modul RaspBee**

Das Modul verbindet das RaspBee Zigbee Aufsatzmodul (bzw. den Stick) mit IP Symcon.

Die Updates des aktuellen Status der Geräte erfolgt über den Websocket automatisch. Es gibt hier keinen Timer / Pull. Das Gateway schickt automatisch diese Änderungen.

Es ist noch nicht fertig oder optimiert. Ich würde hier gerne Input/Feedback zu **allen** Themen haben.

Es ist nichts in Stein gemeisselt und alles kann (muss vielleicht sogar) angepasst werden.

Am schwersten war die Integration der Szenen. Bin mir nicht sicher ob das so optimal ist oder sein muss oder vielleicht anders geht…

Der Code selbst ist nicht optimiert. Wahrscheinlich kann ich den um 50% reduzieren. Fehler wird´s auch noch geben.

**Insgesamt würde ich mich über viele Anregungen, Feedback, auch Kritik etc. freuen.**

**Da noch nicht alles fehlerfrei läuft bitte vorab:**

**Kategorien** für **lights, plugs, groups, sensors, scenes und switches** anlegen

**Ip Adresse** des gateways griffbereit

**Port** des gateways griffbereit

**Api Key** griffbereit

**Websocket port** griffbereit

**Websocket Modul** von Nall-Chan installieren (Anleitung unten)

Es gibt zwar bei Anlage der RaspBee Bridge „Gateway suchen“ und „Api Key generieren“ um die diversen Parameter anzuzeigen, ich bin mir aber nicht sicher ob das so automatisch funktioniert. Passieren kann nichts, aber die Anzeige könnte leer sein, je nach Einrichtung Deines Netzwerkes, Firewall etc.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Installation \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

File entzippen und unter „.../symcon/modules entpacken. Nachher sollte dort ein Modul Ordner „RaspBee“ enthalten sein.

Zuerst das Websocket Modul von NALL – Chan über das Modul Control hinzufügen

git://github.com/Nall-chan/IPSNetwork.git

Die „RaspBee Bridge“ Modul ist im Dialog 'Instanz hinzufügen' anzulegen.

Mit diesem Schritt wird auch direkt die Websocket Instanz und der zugehörige Splitter angelegt.



Client Socket für Websocket Client

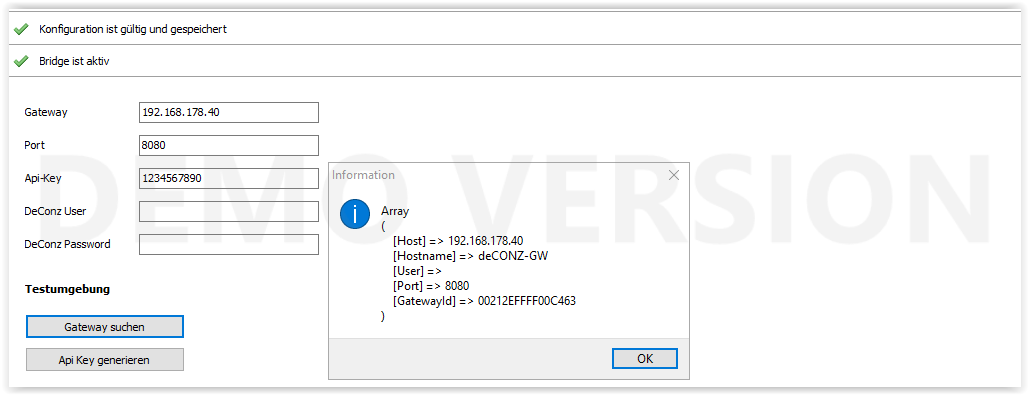
Websocket Client

RaspBee Bridge

Konfiguration RaspBee Bridge:

* IP des RaspBee eingeben
* Port eingeben
* Api-Key eingeben

Falls Port und IP nicht bekannt „Gateway suchen“ Button drücken



Funktioniert noch nicht:

Falls der Api-Key nicht bekannt oder nicht vorhanden „Api Key generieren“ Button drücken.

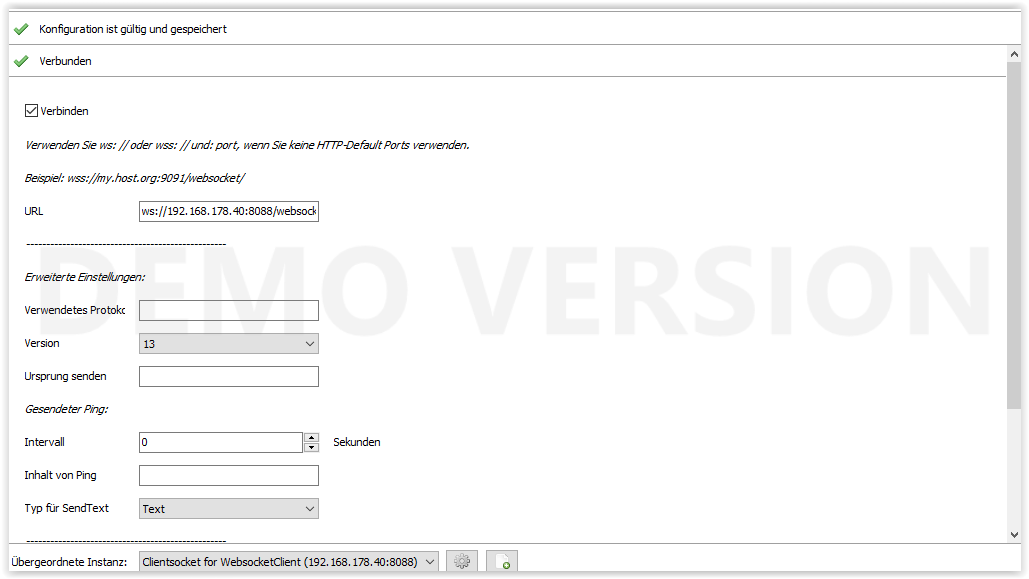
Diese beiden Funktionen konnte ich nur in meiner Installation testen…am besten wäre natürlich der Port und der Api-Key sind bekannt. Falls das nicht funktioniert muss man sich die DeConz Seite ansehen. Dort ist alles genau beschrieben.

<http://dresden-elektronik.github.io/deconz-rest-doc/>

Nun muss noch der **Websocket Client** konfiguriert werden. Einfach Doppelklick auf „Clientsocket für WebsocketClient“

Dort „ws://127.0.0.1:8088/websocket/“ als URL eingeben und „verbinden“ ankreuzen. IP (hier 127.0.0.1) natürlich ersetzen mit der, auf der der RaspBee installiert ist. Ebenfalls den Websocket Port (hier 8088).

Den Websocket Port kann man auch über die „http://xxx.xxx.xxx.xxx:xxxx/api/<apikey>/config“ im Browser herausfinden. Das Feld in der Ausgabe ist : "websocketport":8088

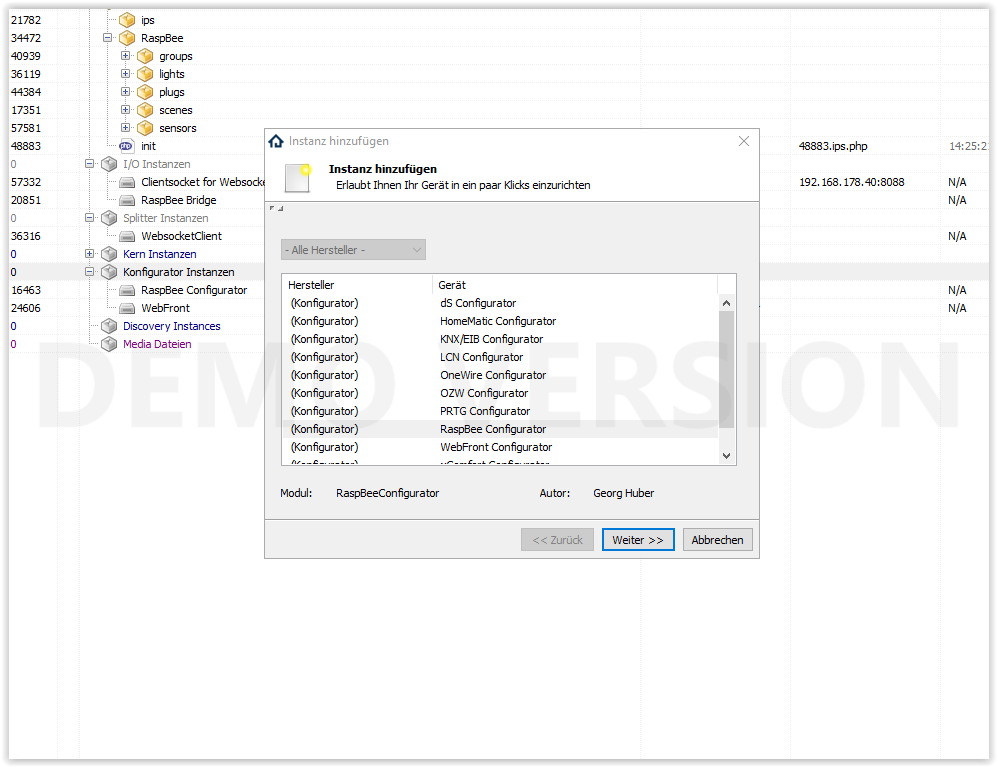


verbinden

Websocket URL

Im Augenblick gibt der Websocket Client von Nall-Chan einige Fehlermeldungen aus, die aber ignoriert werden können. Nach dem Verbinden ist auch der „Client Socket für Websocket Client„ automatisch gefüllt und verbunden

Nun kann über „Konfigurator Instanzen“ der „**RaspBee Konfigurator**“ angelegt werden.

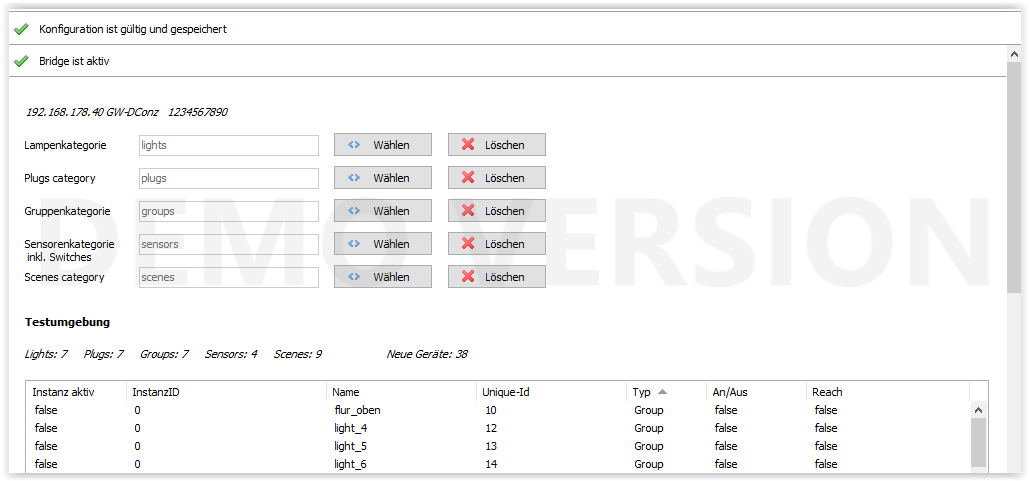


RaspBee Kategorien

Nun sollte alles soweit eingerichtet sein und die Devices können angelegt werden.

Vorher sollte man einige Kategorien anlegen für Lampen, Schalter, Sensoren, Gruppen und Szenen. Bewährt hat sich eine „Ober“ – Kategorie (hier: Raspbee“) und darunter die Sub Kategorien

Nach der Anlage kann man den Raspbee Konfigurator starten (=Doppelklick auf den Konfigurator)



Liste Geräte

# gefundene Geräte

Falls vorher alles richtig angelegt wurde, kann man hier schon sehen wie viele und welche Geräte gefunden wurden.

Jetzt müssen die jeweiligen Kategorien ausgewählt werden in die nachher die Geräte eingefügt werden.

Nachdem das passiert ist kann man weiter nach unten scrollen. (Grün sind Geräte markiert, die bereits angelegt wurden. Beim ersten mal sollten alle nicht grün sein.

**Buttons:**

**Markiertes Device anlegen** : Zeile mit Gerät markieren (es geht immer nur eine Zeile) und dann mit dem Button dieses Gerät anlegen

**Markiertes Device löschen** : Zeile mit Gerät markieren (es geht immer nur eine Zeile) und dann mit dem Button dieses Gerät löschen

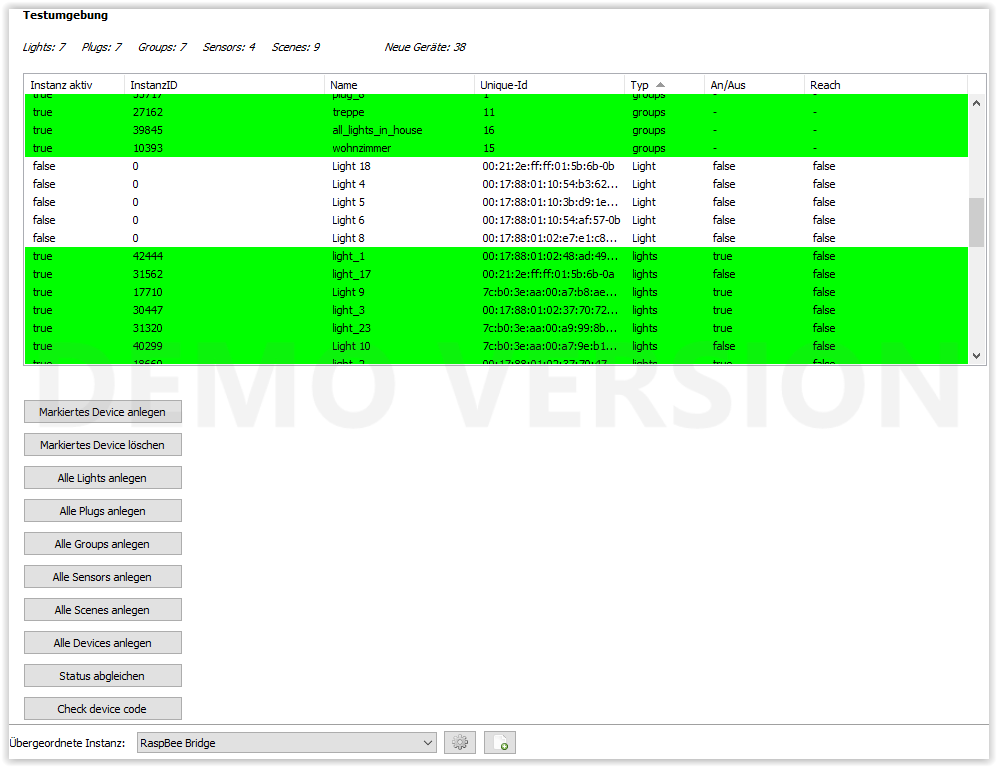
**Alle Lights anlegen** : Alle Lights werden direkt angelegt

Das gleiche mit Plugs, Groups, Scenes und Sensors

**Alle Devices anlegen** : **Alle Geräte** anlegen

**Status abgleichen** : **Geräte Status ermitteln/abgleichen**

**Check Device Code** : Gibt das Json dieses Gerätes (Zeile muss markiert sein) aus



Noch einige Screenshots und Erklärungen…..

Alle Geräte bzw. Einzelinstanzen haben eine solche „Konfigurationsseite“ mit den wichtigsten Daten.

Hier kann im Moment nichts geändert werden, ausser der Markierung bzw. De-Markierung einzelner Attribute.

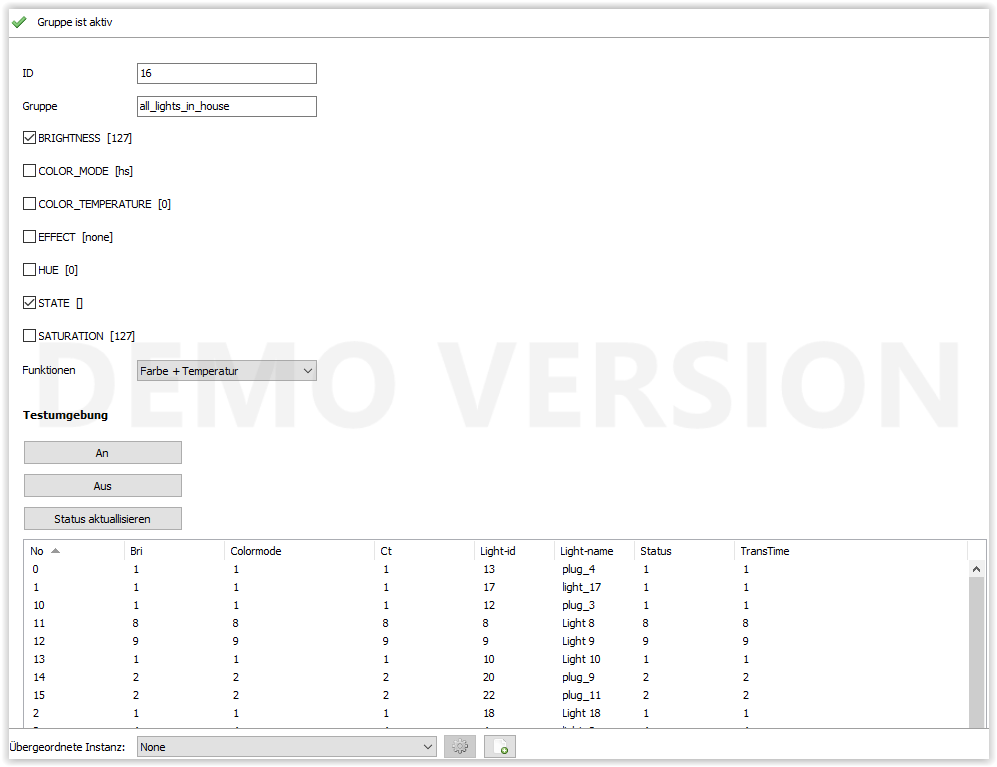
Allgemein werden nur die wichtigsten Attribute (An/Aus, Helligkeit etc.) generell angelegt.

Je Gerätegruppe und verfügbaren Attributen des jeweiligen Gerätes, ist diese Liste länger oder kürzer aus. (zb. bei einer Lampe macht ein „Bewegung erkannt“ keinen Sinn.

Bei einer Änderung muss der Button „Status aktualisieren gedrückt werden. Dabei werden die zusätzlichen Attribute angelegt oder bei De-Markierung auch gelöscht.

Bei Groups oder Szenen sieht man auch eine Liste der enthaltenen Lampen, Plugs etc. zur Info.

Bei einigen Geräten kann man mit „An/Aus“ auch die Funktion testen.



Es gibt auch **Befehle**, die man direkt in seinen Scripts benutzen kann. **Diese ganze Geschichte ist aber noch sehr im Test und lange nicht vollständig oder fehlerfrei !**

**Aber…man kann nichts kaputt machen. Im schlimmsten Fall funktioniert es halt einfach nicht.**

**Funktionen**

Anpassung eines Geräteparameter (siehe SetValues)

RB\_SetValue($lightId, $key, $value);

Anpassung mehrere Lampenparameter

array('KEY1' => 'VALUE1', 'KEY2' => 'VALUE2'...)

Mögliche Keys:

STATE -> true oder false für an/aus

COLOR\_TEMPERATURE -> Farbtemperatur (153 bis 500)

SATURATION -> Sättigung (0 bis 254)

BRIGHTNESS -> Helligkeit in (0 bis 254)

COLOR -> Farbe als integer

ALERT -> Wird durchgereicht

EFFECT -> Wird durchgereicht

TRANSITIONTIME -> Wird durchgereicht

RB\_SetValues($lightId, $list);

Liefert einen Lampenparameter (siehe RB\_SetValue)

RB\_GetValue($lightId, $key);

Weitere Helperfunktionen für Direktverknüpfungen

RB\_SetState($lightId, $value)

RB\_GetState($lightId)

RB\_SetColor($lightId, $value)

RB\_GetColor($lightId)

RB\_SetBrightness($lightId, $value)

RB\_GetBrightness($lightId)

Insgesamt ist das natürlich, je nach Bedarf ausbaubar. Es gibt noch einige „public“ – Funktionen die in „Befehle“ angezeigt werden, die sind aber nicht wirklich „public“ und aufrufbar. Muss die noch rausnehmen.

Noch einige Screenshots:

