Scripte

Scripte sind eine Auflistung von Kommandos, die automatisch ausgeführt werden können. Es können beliebig viel Scripte erstellt werden. Bei längeren Scripten kann es zur kurzen Ladezeiten kommen. Scripte sind in der Datei script.json organisiert. Diese Datei kann per Konfigurationstool in die App integriert werden oder manuell geladen werden. Falls Fehler in der Datei vorhanden sind, wird die App mit entsprechendem Hinweis beendet. Zum bearbeiten empfiehlt sich eine Editor wie Notepad oder ein JSON online Editor.

Laden von Scripten

Falls das laden von externen Scripten aktiviert ist, können diese über die Benutzeröberfläche geladen werden. Zum öffnen von Scripten ist der Menüpunkt „Load Script“ zu benutzen. Hier kann eine Datei ausgewählt werden die die benötigten Scripte enthält.

script.json - Umgang mit der Datei

Scripte

Scripte sind eine Auflistung von Kommandos, die automatisch ausgeführt werden können. Es können beliebig viel Scripte erstellt werden. Bei längeren Scripten kann es zur kurzen Ladezeiten kommen. Scripte sind in der Datei script.json organisiert. Diese Datei kann per Konfigurationstool in die App integriert werden oder manuell geladen werden. Falls Fehler in der Datei vorhanden sind, wird die App mit entsprechendem Hinweis beendet. Zum bearbeiten empfiehlt sich eine Editor wie Notepad oder ein JSON online Editor.

Laden von Scripten

Falls das laden von externen Scripten aktiviert ist, können diese über die Benutzeröberfläche geladen werden. Zum öffnen von Scripten ist der Menüpunkt „Load Script“ zu benutzen. Hier kann eine Datei ausgewählt werden die die benötigten Scripte enthält.

script.json - Umgang mit der Datei

Block

Einzelne Scripte sind in Blöcken organisiert. Ein block sieht wie folgt aus:

```
{
```

```
  "name": "Name des Scripts",
  "cmd": "kommando1;kommando2;"
}
```

Dieser Script würden den Namen. „Name des Scripts“ tragen und die Kommandos: „kommando1;kommando2;“ enthalten.

Mehrere Scripte

Einzelne Blöcke werden jeweils per Kommata getrennt, eine Ausnahme hierbei stellt der letzte Block dar.

Beispiel:

```
[
{
Block1
},
{
Block2
}
]
```

Als letztes ist es wichtig, alle Blöcke in [ ] zu verschachteln.

Mehrere Scripte

Einzelne Blöcke werden jeweils per Kommata getrennt, eine Ausnahme hierbei stellt der letzte Block dar.

Beispiel:

```
[
{
Block1
},
{
Block2
}
]
```

Als letztes ist es wichtig, alle Blöcke in [ ] zu verschachteln.

Fertige Datei

Als Beispiel für eine syntaktisch korrekte script.json, hier eine Datei die fünf Scripte definiert.

```
[
  {
    "name": "Erster Script",
    "cmd": "scan 3;connect 0;disconnect;wait 5;connect 0;disconnect;"
  },
  {
    "name": "Zweiter script",
    "cmd": "scan 5;"
  },
  {
    "name": "Dritter Script",
    "cmd": "connect XX:XX:XX:XX:XX:XX;"
  },
  {
    "name": "Vierter Script",
    "cmd": "write 2a00 'test';"
  },
  {
    "name": "Fünfter Script",
    "cmd": "disconnect;"
  }
]
```