ioBroker

Hier ist alles zum Thema ioBroker enthalten.

https://www.iobroker.net/

- Installation
  - ioBroker als Dockercontainer
- Adapterentwicklung
  - Adapter Creator
  - Dev-Server
  - Konfigurationsdialog erstellen

Installation

ioBroker als Dockercontainer

YAML-Datei:

```
services:
  iobroker:
    container_name: iobroker
    image: buanet/iobroker:latest-v8
    hostname: iobroker
    network_mode: "host"
    restart: always

    volumes:
      - iobroker_data:/opt/iobroker

    environment:
       AVAHI: true

volumes:
  iobroker_data:
```

Erläuterungen:

Informationen zu dem verwendeten Image:

https://hub.docker.com/r/buanet/iobroker

https://smarthome.buanet.de/

Da einige Adapter in einem "normalen" Dockernetzwerk nicht sauber laufen, wurde *network_mode:* "*host*" gewählt. Dann müssen auch keine Ports freigeschaltet werden.

Die Installation erfolgt dann im Portainer, in dem ein neuer Stack angelegt wird. Der Inhalt der YAML-Datei wird in den Editor kopiert und dann "Deploy the Stack" ausgeführt.

Die Version des Image wurde hier mit "latest-v8" angegeben. Das entspricht "latest" zum aktuellen Zeitpunkt. Da der Container automatisiert aktualisiert werden soll, ist mit "latest-v8" sichergestellt, dass die Updates sich nur innerhalb der Imageversion 8 bewegen. Sollte eine Version 9 erscheinen, muss dieses manuell erfolgen. Dieses ist sinnvoll, da der größere Versionssprung weitere manuelle

Änderungen erfordern könnte.

Updates:

Auf folgender Seite ist beschrieben, wie Updates eingespielt werden können/sollen:

https://smarthome.buanet.de/2020/10/iobroker-docker-container-updates-upgrades/

Adapterentwicklung

Hier sammle ich Informationen zu der Entwicklung von Adaptern für ioBroker.

Adapter Creator

Für den ioBroker wird ein Adpater Creator zur Verfügung gestellt. Mit diesem wird u.a. eine grundlegende Adapterstruktur auf dem Entwicklungsrechner abgelegt.

Der Adapter Creater ist hier zu finden: https://www.iobroker.dev/create-adapter

Es gibt einen Adapter Creator mit einem Wizard in der Web-GUI. Der hat bei mir leider nicht funktioniert. Daher bitte die CLI-Version verwenden.

Bei der Nutzung des Creators werden einige Fragen gestellt, damit der erstellte Adapter gleich die passenden Einstellungen und Features bietet.

Dev-Server

Für die Entwicklung eines Adapters steht ein s.g. Dev-Server zur Verfügung. Das ist ioBroker-Server, mit zusätzlichen Möglichkeiten. In dem Server läuft der Adapter, den man entwickelt. Ändert man eine Datei und speichert wird der Server neu gestartet und man die Änderungen direkt betrachten.

Man sollte den Dev-Server auf einem eigenen System installieren, z.B. in einer virtuellen Maschine. In dieser installiert man dann z.B. Ubuntu und führt anschließend folgende Schritte aus:

```
npm install --global @iobroker/dev-server
```

Dann wechselt man in das Verzeichnis, in dem der der Code des Adapters liegt und führt noch folgendes aus:

```
dev-server setup
```

In die Dateien ".gitignore" und ".npmignore" das Verzeichnis ".dev-server" eintragen.

Nun kann man folgende Befehle ausführen um den Dev-Server zu starten:

dev-server run //HTML Datei in deinem lokalen Entwicklungsverzeichnis ändern und speichern

```
dev-server run //HTML Datei in deinem lokalen Entwicklungsverzeichnis ändern und speichern
```

```
dev-server watch //JavaScript (oder TypeScript) Adapter Code ändern und speichern
```

```
dev-server debug //Wenn du willst, kannst du nun deinen Debugger attachen
```

Weiterführende Links:

https://forum.iobroker.net/topic/42658/entwicklungs-tool-iobroker-dev-server

Konfigurationsdialog erstellen

Der Konfigurationsdialog wird ab Admin 5 in der Datei jsonConfig.json eingetragen, die im Unterordner "admin" angelegt werden muss.

Hier ein Beispiel:

Ein Default für eine Option kann in der Datei io-package.json definiert werden. In dieser Json-Datei kann unter "native" zu jedem Optionswert ein Default hinterlegt werden:

```
 "native": {
        "serverIp": "192.168.0.1"
 },
```

Mit Hilfe des Pakets "@iobroker/adapter-dev" und dem Script ""translate" können die Optionseinträge in divese Sprachen automatisch übersetzt werden. Dazu muss zunächst der Pfad "admin->i18n->en" angelegt werden. Anschließnd darin die Datei "translations.json" darin erstellen. In diese werden dann alle Wörter aus den Optionen, für die eine Übersetzung sinnvoll ist, wie folgt in die Datei eingetragen:

```
{
    "Options": "Options",
    "Server": "Server",
    "IP or hostname of your Gira X1 or HomeServer": "IP or hostname of your Gira X1 or
HomeServer",
    "Port": "Port"
}
```

Da es sich um die Übersetzungsdatei in englische handelt stehen auf beiden Seiten immer der gleichen (englischen) Begriffe. Mit folgendem Befehl können dazu die Übersetzungen in andere Sprachen automatisiert angelegt werden:

```
npm run translate
```