SmartHomeConnector.js

Abstract

Ist das Licht aus? Wurde die Jalousie heruntergelassen? Läuft die Klimaanlage auf Hochtouren? Habe ich das Fenster offen gelassen?
Home Automation Systeme sind zu kompliziert und schwer zu integrieren?
Immer mehr Haushalte werden “smart” - wir helfen mit dass Ihre Benutzer ihre Systeme zu Hause auch smart nutzen können! Mit der neuen open source JavaScript Library “Smart Home Connector” können verschiedene Home Automation Systeme angesteuert werden. So lässt sich über unsere API der Status von bestimmten Peripheriegeräten sowohl anfragen als auch ändern.
Wandeln Sie Ihre Web Applikation in ein Home Automation Center in nur wenigen Schritten. Integrieren Sie die Smart Home Connector Library und bieten Sie Ihren Benutzern auf einfache Art und Weise einen großartigen und modernen Mehrwert am Puls der Zeit!
Home Automation Systeme verfolgen zwar sehr ähnliche Ziele - aus der Nähe betrachtet unterscheiden sich jedoch die meisten konzeptionell recht deutlich. Wir ersparen Ihnen, die Konzepte von unterschiedlichen Home Automation Systemen erlernen zu müssen und bieten Ihnen eine einheitliche API zur Steuerung an.
Zum Beispiel können so auf einfache Art und Weise Kamerabilder abgerufen oder Lichtquellen aus bzw. eingeschaltet werden.

Funktionale Anforderungen

1. Abfragen und Hinzufügen von Home Automation System Components
2. Abfragen von angebotenen Component Services
3. Abfragen und Ändern des Status Component Services
4. Anzeigen von Kamerabildern (Component.category = camera; Voraussetzung openHAB: Camera binding)

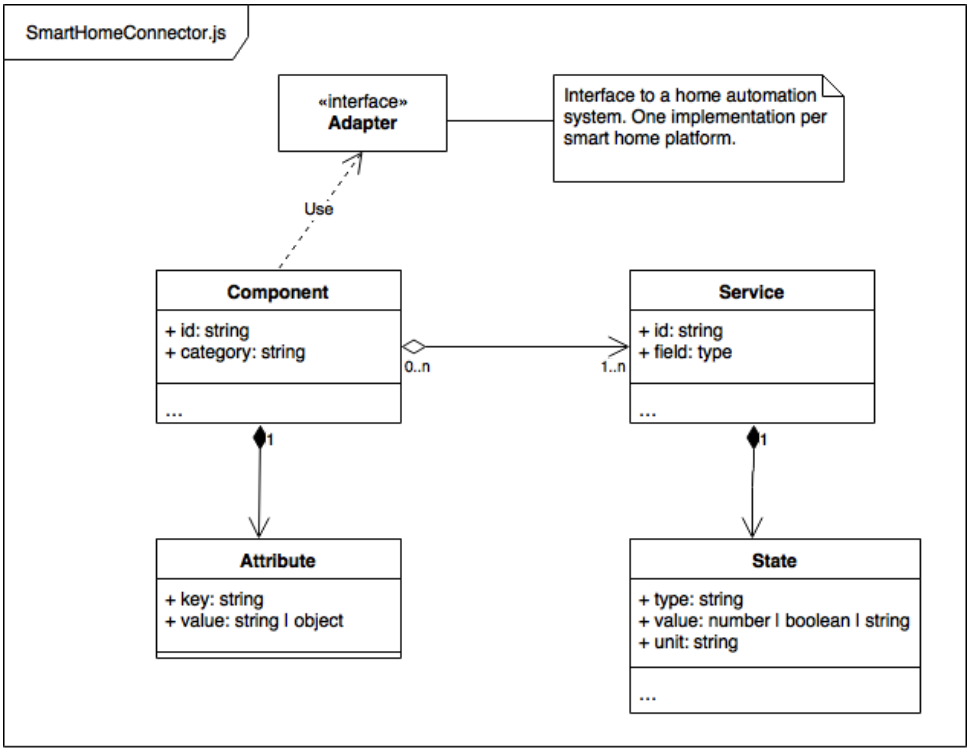
Nicht-Funktionale Anforderungen

1. Einfachheit - die API soll für Entwickler möglichst schnell erschließbar und verwendbar sein
2. Wenige Abhängigkeiten: axios: http request, openHAB Camera binding
3. Erweiterbarkeit: Adapter-Schnittstelle zur Anbindung neuer Home Automation Systeme
4. open source licensing (MIT)
5. Dokumentation (API)

https://www.openhab.org/

Solution Design

Klassendiagramm



Objekte

Adapter	zur Steuerung eines bestimmten Systems (REST API)
Component	entspricht einem Gerät / einer Software / Sensor
Service	entspricht einer Schnittstellenkategorie / zur Änderung des Status; z.B. Lautstärke
Attribute	optional key/value attributes pro Component (können unterschiedlich sein je Component Typ)
State	Objekt mit dem Status einer Schnittstelle; z.B. ON/OFF, 20db/10db; true/false verschiedene State Objekte denkbar (Boolean, Numerical, Percentage, String)

Beispiel:

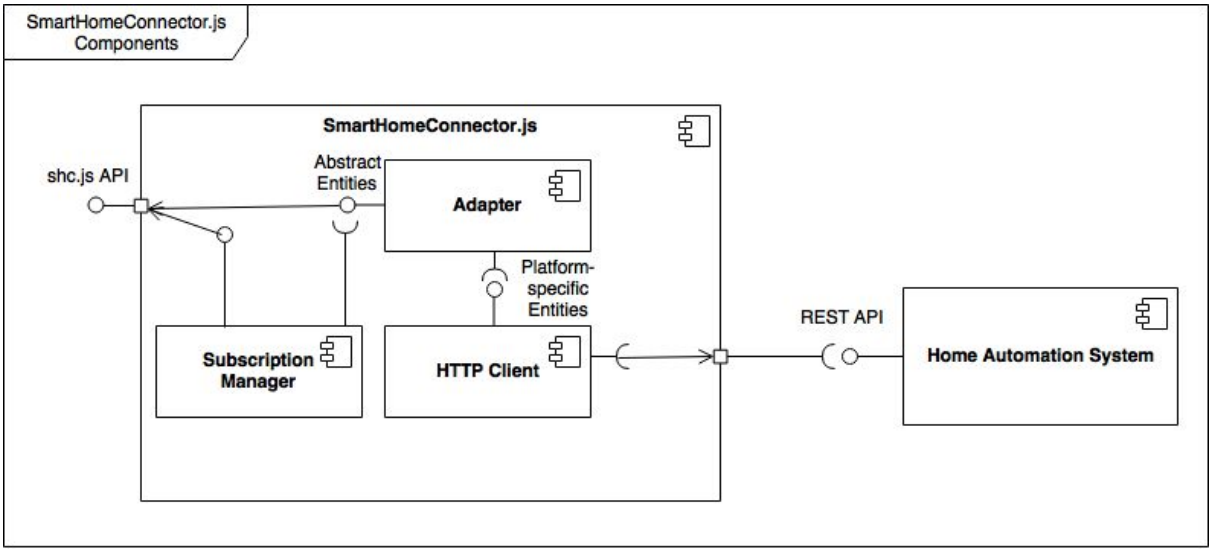
```
Component {
      category: “AV-Receiver”
      services: {
            “id” : “1”
            “name” : “noise_regulation”
            “state”: {
                  “type”: “Numerical”,
```

```
                “value”: 20,
                “unit”: “db”
          }
    },
    attributes: {
          “name”: “Denon AVR-2200”,
          “producer”: “Denon”
    }
}
```

Interaktion – Beispiel

```
const shc = new SmartHomeConnector({
  adapter: new OpenHabAdapter(),    // Select adapter for backend
  url: "my-openhab-controller"
})

// Get available components, add components (Requirement 1)
shc.getComponents().then(/* provides components[] */)
shc.getComponent(id).then(/* provides component (if avaliable) */)
shc.addComponent(…).then(/* succeeded */).catch(/* failed */)

// Get component services (Requirement 2)
shc.getServices().then(/* provides services[] */)
shc.getService(id|name).then(/* provides service (if available) */)

component.getAttributes().then(/* provides attributes[] */)
component.getServices().then(/* provides services[] */)

// Get/Set state (Requirement 3)
service.getState().then(/* provides state[] (all components) */)
service.getState(component).then(/* provides state[] */)
service.updateState({…})

// Monitor state changes
state.on('change', callback)
state.on('change', attributes /* e.g. {min: 10, max: 20} */, callback)

// Get camera image (Requirement 4) → just get the corresponding state
const camera = shc.getComponent(cameraId)
const imageService = shc.getService(‘image’)
imageService.getState(camera).then(/* contains image */)
```

Komponentendiagramm



Adapter	siehe oben
Subscription Manager	für die Überwachung von Zustandsänderungen
HTTP Client	zur Kommunikation mit dem entsprechenden System
Abstract Entities	Objekte von SmartHomeConnector.js, siehe oben
Platform-specific Entities	spezifische Objekte je Smart-Home-System