Projekt IT - Meilenstein 2

Gruppe 2
(Friederike Bartels, Sebastian Brodersen,
Nicolas Marin, Daniel Rickert)

24.05.2017

Java Maven Projekt

Für die Aufgaben wurde das Java Maven Projekt auf github.com erweitert. Dieses ist öffentlich und kann unter der URL `https://github.com/sebr8041/projekt-it` erreicht werden. Mit Hilfe von 'scp' kann dieses auf den Raspberry Pi kopiert werden. Der Code für den Arduino kann mit der Arduino IDE auf diesen übertragen werden.

Aufgabe 2.1

nCoap klonen

Um den Branch `some-fixes` herunterzuladen, zu bauen und ins lokale Maven-Repository zu installieren benötigt man die folgenden Kommandos:

- `git clone https://github.com/projekt-internet-technologien/nCoAP.git ./`
- `git checkout -b some-fixes`
- `mvn clean install`

Coap Client: Copper ausprobieren

In der gegebenen Beispielanwendung[1] wird der itm-Prefix für den SimpleObservableTimeService angepasst:

- @prefix itm: <http://gruppe**02**.pit.itm.uni-luebeck.de/>

Nach dem Übertragen und Starten auf dem Pi, können die Nachrichten mit Copper beobachtet werden. Dazu wird

- coap://141.83.175.235:5683/utc-time

in die Adresszeile des Browsers eingegeben. Durch Drücken des GET-Buttons kann die aktuelle UTC-Zeit eingeholt werden. Durch Drücken des OBSERVE-Buttons erscheint immer der aktuelle Wert. Wie in Abb. 1 zu sehen, wird die Zeit per Default als Plaintext ausgegeben.

In der rechten Seitenleiste kann das Ausgabeformat geändert werden. Abb. 2 zeigt die Ausgabe für XML.

[1] https://github.com/projekt-internet-technologien/ssp-example-application

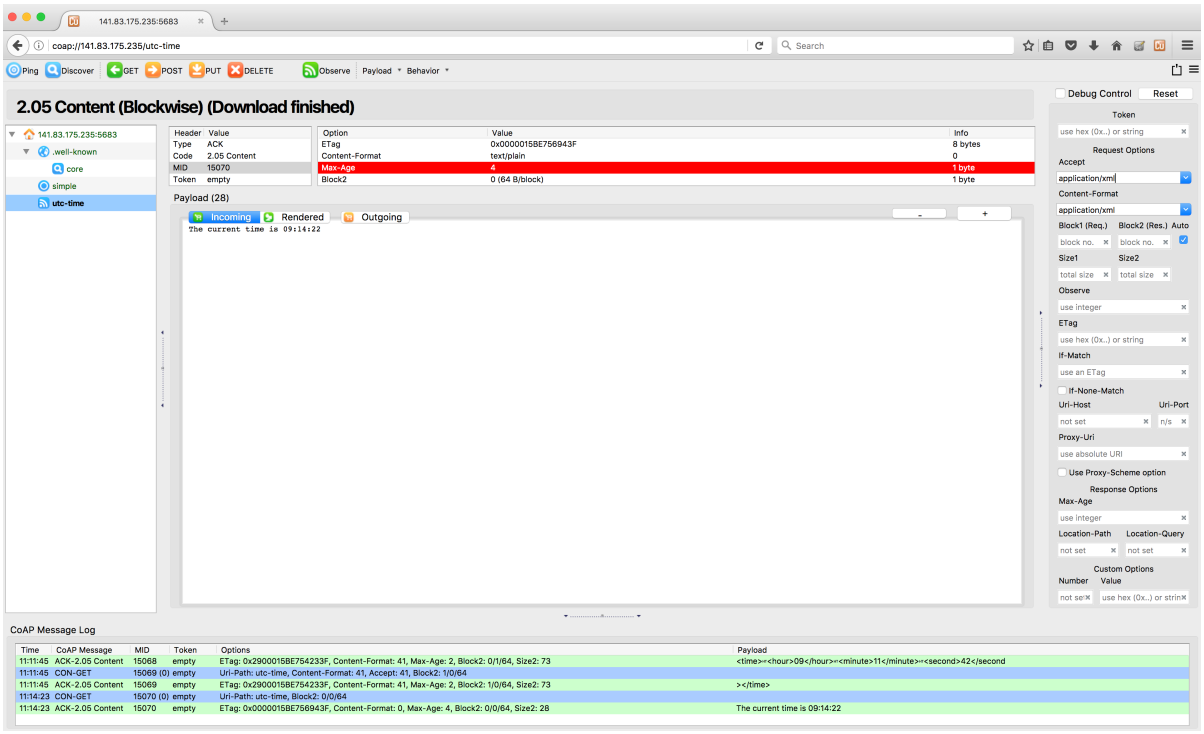

Abbildung 1: UTC in Copper - text/plain

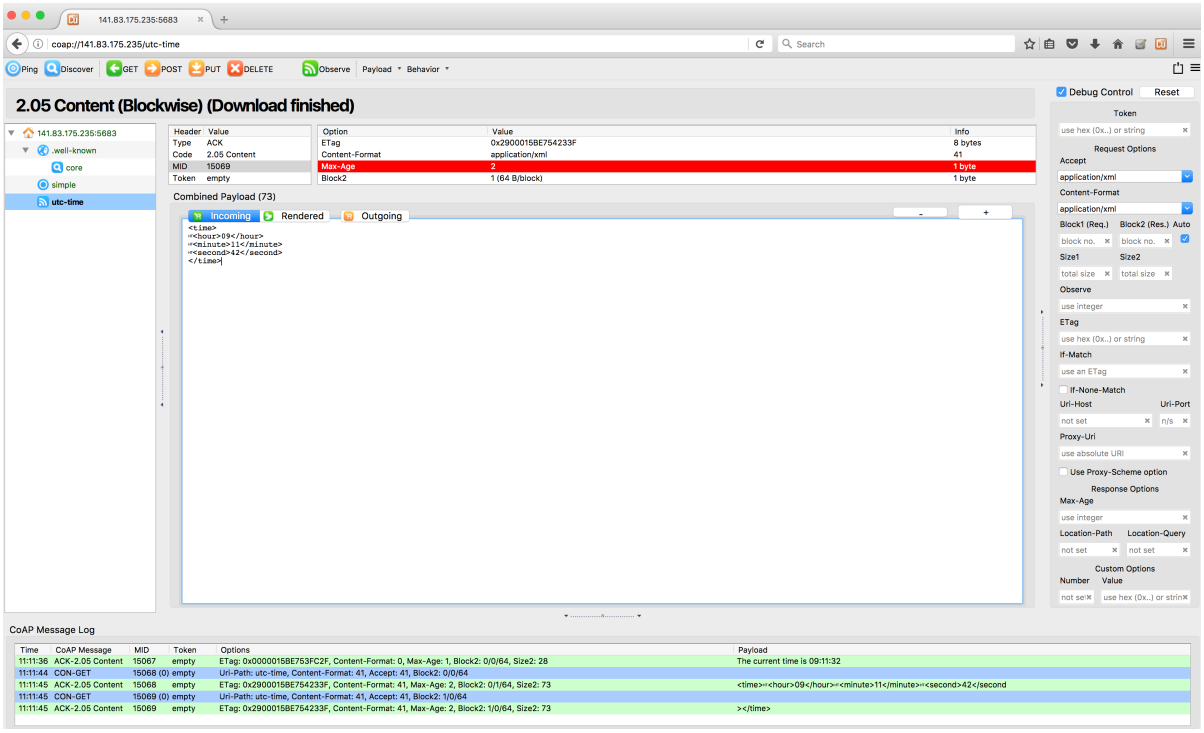

Abbildung 2: UTC in Copper - XML

Suchen Sie mittels SPARQL im SSP nach Ihrer UTC-Zeit

Um den SPARQL-Endpunkt verwenden zu können, muss man sich zunächst im UzL-Konferenz WLAN einloggen. Der Endpunkt kann anschließend unter `http://141.83.151.196:8080/services/sparql-endpoint` abgeru-

fen werden.

Folgender Query wurde verwendet, um unsere UTC-Zeit herauszufinden:

Listing 1: SPARQL-Anfrage zum Auslesen der UTC-Zeit

```
PREFIX itm: <http://gruppe02.pit.itm.uni-luebeck.de/>

SELECT * WHERE {
 itm:time1 ?p ?o.
}
```

Java Programm

Um den Helligkeitswert aus dem letzten Meilenstein observieren zu können, wird in Anlehnung an den SimpleObservableTimeService ein SimpleObservableLightService[2] entwickelt. In diesem werden, wie in Listing 2 zu sehen, die entsprechenden RDF-Daten generiert. Als CURRENT LDR VALUE wird der aktuelle Helligkeitswert eingesetzt. Dieser Dienst kann, wie auch der SimpleObservableTimeService, von dem SSP observiert werden.

Listing 2: RDF-Daten zum Senden an den SSP

```
@prefix gruppe2: <http://gruppe02.pit.itm.uni-luebeck.de/>
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>
@prefix itm: <https://pit.itm.uni-luebeck.de/>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

gruppe2:ldr          rdf:type          itm:Component.
gruppe2:ldr          itm:hasStatus     gruppe2:ldrStatus.
gruppe2:ldrStatus    itm:hasValue      "CURRENT LDR VALUE"^^xsd:string.
gruppe2:ldrStatus    itm:hasScaleUnit  "Lux"^^xsd:string.
gruppe2:ldr          itm:isType        "LDR"^^xsd:string.
gruppe2:myPi         rdf:type          itm:Device.
gruppe2:myPi         itm:hasIP         "141.83.175.235"^^xsd:string.
gruppe2:myPi         itm:hasGroup      "PIT_02-SS17"^^xsd:string.
gruppe2:myPi         itm:hasLabel      "PIET"^^xsd:string."
gruppe2:ldr          itm:hasURL        "coap://141.83.175.235:5683/ldr"^^xsd:anyURI
```

Aufgabe 2.2

Um den durchschnittlichen Wert aller LDR-Sensoren zu ermitteln, wurde die SPARQL-Anfrage in Listing 3 entwickelt. Für jede Komponente vom Typ LDR wird der zugehörige Statuswert ermittelt und mit `AVG` der Durchschnitt dieser Werte berechnet. Mithilfe des REST-Client aus dem Handbuch[3] wird die Anfrage jede Sekunde an den Server geschickt. Die JSON-Rückgabe wird mit Jackson in Java-Objekte geparst, um dann den Durchschnittswert zu übernehmen[4].

[2] https://raw.githubusercontent.com/sebr8041/projekt-it/master/src/main/java/de/uni_luebeck/itm/itp2017/gruppe2/PiLib/tasks/SimpleObservableLightService.java

[3] https://github.com/projekt-internet-technologien/SSP-REST-Util

[4] https://raw.githubusercontent.com/sebr8041/projekt-it/master/src/main/java/de/uni_luebeck/itm/itp2017/gruppe2/PiLib/tasks/Task2.java

Die Klasse zur Steuerung der LED aus Aufgabe 1.3 wurde um die beiden Methoden `setAvgValue` und `getCurrentValue` erweitert. Die erste schaltet abhängig vom Eingabewert die LED ein- oder aus. `getCurrentValue` liefert den aktuellen Zustand des an über den Arduino an den Raspberry Pi angeschlossenen LDR[5].

Listing 3: SPARQL-Query zum Abfragen des Durchschnittswertes am SSP

```
PREFIX itm: <https://pit.itm.uni-luebeck.de/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT ( AVG(xsd:float(?value)) AS ?v ) WHERE {
    ?comp       rdf:type            itm:Component.
    ?comp       itm:isType          "LDR"^^xsd:string.
    ?comp       itm:hasStatus       ?status.
    ?status     itm:hasScaleUnit    "Lux"^^xsd:string.
    ?status     itm:hasValue        ?value.
}
```

[5] `https://raw.githubusercontent.com/sebr8041/projekt-it/master/src/main/java/de/uni_luebeck/itm/itp2017/gruppe2/PiLib/tasks/Task1_3.java`