Hochschule RheinMain
Fachbereich DCSM - Informatik
Dr. Oliver Hahm

Verteilte Systeme
SS 2020
LV 4132

Übungsblatt 9
Praktische Übungen
Bearbeitungszeit: 2 Wochen
Abgabe: 05.07.2021, 04:00 Uhr MESZ

Aufgabe 9.1 (Projekt „Hamsterasyl als RESTFul-Webservice“):

Nachdem das aufgekaufte westhessische Hamsterverwahrungsunternehmen die IoT-Lösung erfolgreich entwickelt hat, ist dem CTO auf dem Golfplatz erklärt worden, das IoT ohne REST nicht geht. Alles muss heutzutage REST sein!

Dem CTO wurde auch gleich ein indischer Software-Dienstleister empfohlen, der sich hervorragendst mit REST auskennt. Selbstverständlich wurde direkt noch auf dem Golfplatz, per Blackberry (ja, da ist er etwas altmodisch!) der Auftrag vergeben.

Alles muss schnell gehen heutzutage. Das heißt, die Mitarbeiter der IT-Abteilung haben nun den Auftrag, den Client der Hamsterverwaltungs-Software auf REST umzustellen, während die Inder den Server entwickeln.

Die Support-E-Mail-Adresse der Inder ist: andreas.werner@hs-rm.de

Bei Support-Anfragen müssen Sie darauf achten, dass Sie den Fehler gut und nachvollziehbar beschreiben und alle relevanten Informationen beifügen. Mit der Beschwerde „Irgendwas geht nicht“ können die Inder nichts anfangen!

In Ihrem SVN-Repository finden Sie den neuen Ordner „9“ und darin die Ordner „src“, „include“, „lib“ und „Aufgabe“. Im Ordner „include“ können Sie diesmal **keine** Änderungen vornehmen!

Zur Unterscheidung werden alle Daten mit einer Versionsnummer gekennzeichnet. Diese hat das Format: (Datum nach ISO-Format)-(Zähler für Tag). Also bspw. `2018-06-21-1`.

Aufgabe 9.1.1 Informieren über REST

REST (*Representational State Transfer*) ist ein Programmierparadigma, keine spezielle Technologie. Wie RPC wird auch REST im Kontext von Client-Server Anwendungen verwendet, wobei der Server zustandslos ist. Als unterlagertes Protokoll wird HTTP verwendet. Ein REST-Server bietet Zugriff auf *Ressourcen*, die durch URLs beschrieben werden. Zum Zugriff können die HTTP-Methoden (GET, POST, PUT, DELETE, etc.) verwendet werden, um den Inhalt der Ressource zu lesen oder zu ändern. Durch die Verwendung von HTTP ist REST grundsätzlich für alle Plattformen und Programmiersprachen nutzbar.

Der Client wird in C geschrieben, daher verwenden Sie REST mit C. Sie können, wenn Sie wollen alles von Hand bauen, oder Sie können eine Library verwenden, die Ihnen Arbeit abnimmt. Als

Vorschlag wird hier libcurl bereitgestellt (`https://curl.haxx.se/libcurl/`). Informieren Sie sich über die API-Nutzung.

Curl ist auch als Kommandozeilen-Programm auf Unix-Rechnern verfügbar. Damit können Sie direkt von der Shell REST-Dienste ansprechen. Nutzen Sie diese Möglichkeit, um den REST-Service zu erkunden und Dinge zu testen.

Wenn Sie eine andere C-basierte Library verwenden wollen, schicken Sie einen Request an die „Inder“ (aka. Swami Kaiser). Ggf. wird die Library dann dem Projekt (und damit allen) zur Verfügung gestellt.

Aufgabe 9.1.2 RESTFul-Hamster-Service

Aufgabe:

- Passen Sie den aus Aufgabe 5 bekannten CLI-Client für die Nutzung mit REST an.
- Damit Sie nicht direkt am Menüprogramm `hamster_cli.c` arbeiten müssen, hat ein chinesischer Subcontractor eine passende C-API entwickelt und eingebunden. Die Implementierung müssen Sie in der Datei `hamster_client.c` erledigen, das Hauptprogramm `hamster_cli.c` braucht nicht verändert zu werden. Die Parameter der API sind selbsterklärend (vom Doku schreiben stand nichts im Vertrag des chinesischen Subcontractors)

Die Dokumentation des REST-Service wird, wie es sich für REST gehört, als Web-Seite zur Verfügung gestellt.

Die URL ist: `http://hamsteriot.vs.cs.hs-rm.de:8080/HamsterREST/doc`

Dort finden Sie eine Auflistung der verfügbaren Ressourcen und für jede Ressource eine Beschreibung, welche HTTP-Methoden auf sie anwendbar sind, und was diese jeweils bewirken.

Der REST-Service wird unter `http://hamsteriot.vs.cs.hs-rm.de:8080/HamsterREST/rest` bereit gestellt.

Die für Sie am Anfang wichtigste REST-Funktion wird die Abfrage der aktuellen Versionsnummer sein. Die Resource-URI dafür ist `/version`.

Testen Sie das bspw. mit `curl`: `curl -v -H "Accept:text/plain" http://hamsteriot.vs.cs.hs-rm.de:8080/HamsterREST/rest/version`[1]

Vorgehen:

- Testen Sie die REST-Schnittstelle zuerst mit `curl`.
- Sehen Sie immer Debug-Code vor, der Ihnen ihren Request und die Antwort des Servers als Text ausgibt.
- Die REST-Schnittstelle unterscheidet sich von der gewohnten C-Schnittstelle (ist jetzt ja auch REST!). Sie müssen daher die richtigen Optionen auswählen. Ggf. benötigen Sie nicht alles was die REST-Schnittstelle Ihnen bietet!
- Implementieren Sie den Client zunächst ohne Authentifizierung.

[1] Per default macht das Programm `curl` HTTP GET Requests. Andere HTTP-Methoden können über die Option -X <Methode> ausgewählt werden