Veranstaltungseinheit Ajax Grundlagen

TaskList JSON Repräsentation

Ziel der Übung

Sie können ein Javascript Objekt zur Übertragung in das JSON Format serialisieren.

Aufgabe(n)

Implementieren Sie die Methode `toJSON` in der Datei `src/TaskList.js`. Die Methode soll einen String entsprechend dem aktuellen Status des Objekte zurückgeben. Der String soll verwertbares JSON beinhalten und von derselben Form wie die Daten der REST Schnittstelle sein.

Ein Beispiel String:

```
{"id":"demo", "tasks":
[{"title":"Buy milk","done":false},
{"title":"Invite friends","done":false},
{"title":"Call Bill","done":false},
{"title":"Write Task List","done":true}]}
```

In der Datei `spec/TaskListSpec.js` sind bereits Tests vorbereitet, welche die Funktion überprüfen. Beachten Sie, dass die Funktion einen String zurückgeben soll, und nicht ein Object von einem andern Typ. Um ein beliebiges Objekt in einen String im JSON Format umzuwandeln bietet Javascript die Funktion `JSON.stringify()`.

Erwartete Resultate und Abgabe

- Um die genannte Funktionalität erweitertes Programm.
- Bei Erfüllung analoger Kriterien kann das Resultat auch in ihr Projekt aus der Projektschiene integriert werden.
- Die nächste Übung baut auf diesem Programm auf.

Pendenzenlisten über REST persistieren

Ziel der Übung

Sie können eine Taskliste über ein REST API persistieren.

Aufgaben

Implementieren Sie die Methode `save` in der Datei `src/TaskList.js`. Die Methode soll die JSON Repräsentation des aktuellen Objektes an den Server übertragen.

Tasklisten sollen unabhängig davon ob sie bereits einmal gespeichert wurden oder nicht mit einem HTTP POST übertragen werden. Neue Tasklisten sollen an die Addresse http://zhaw.herokuapp.com/task_lists/ übertragen werden. Bereits einmal gesicherte Tasklisten sollen an den Endpunkt http://zhaw.herokuapp.com/task_lists/:id übertragen werden, wobei :id eine eindeutige Identifizierung der Taskliste darstellt.

Der Server antwortet auf jeden gültigen POST Request mit einer JSON Repräsentation der Taskliste. Diese Antwort ist im Falle von neu erstellten Tasklisten zu verwerten. Dann wird nämlich auch die serverseitig erzeugte Identifizierung der Taskliste zurückgegeben. Stellen Sie sicher, dass diese Id für neu erstellte Tasklisten auf dem Objekt abgelegt wird und bei folgenden POST Requests verwendet wird. Ansonsten wird jeder Ihrer POST Requests eine neue Taskliste erstellen. In der Datei `spec/TaskListSpec.js` ist bereits ein Tests vorbereitet, der dieses Verhalten überprüft.

In der Browser Ansicht der Website befindet sich ein Button mit der Aufschrift `Save to server`. Die diesem Button hinterlegte Funktionalität benutzt die zu erstellende Funktion `save`.

Erwartete Resultate und Abgabe

- Javascript Projekt welches Tasklisten zum Server übermitteln und speichern kann.
- Bei Erfüllung analoger Kriterien kann das Resultat auch in ihr Projekt aus der Projektschiene integriert werden.
- Abgabe während dem Labor.

Pendenzenlisten Bookmark - freiwillig

Ziel der Übung

Sie können bestehende Tasklisten über eine URI im Browser laden.

Aufgaben

Viele Javascript Applikationen brauchen den Hash Teil der URL zum Speichern von Status-Informationen. Bei einer Veränderung des Status der Seite (z.B. Wechsel zu einer anderen Taskliste) wird ein anderer Hash gesetzt. Im Gegensatz

zum ändern des Pfades in der URL wird bei einer ausschliesslichen Änderung des Hashes die Seite nie neu geladen.

Stellen Sie sicher, dass nach dem Speichern der Taskliste die Identifizierung in der URL gesetzt wird. Benutzen Sie dazu `window.location.hash`. Nach dem speichern sollte in Ihrem Location Bar ein Wert wie folgt stehen:

```
file:///web2/10_rest/labor/index.html#1376066459108558
```

Stellen Sie ausserdem sicher, dass in `src/Demo.js` auf den Hash Teil der URL reagiert wird. Ist ein Hash vorhanden, soll via `TaskList#load` die entsprechende Pendenzenliste geladen werden. So sind Bookmarks auf Tasklisten und Browser Reloads möglich.

Erwartete Resultate und Abgabe

Freiwillige Übung