Veranstaltungseinheit Promises

Promises verwenden

Ziel der Übung

- Sie kennen den Unterschied von Promises und Callbacks
- Sie verstehen das grundlegende Konzept der Javascript Promises
- Sie können Funktionen mit Callbacks zu Promises adaptieren
- Sie können den Aufruf von Promises orchestrieren

Aufgabe(n)

Für die Aufgabe werden asynchrone Request mit der setTimeout Methode simuliert. Damit das debugging einfacher wird, sind die Timeouts hart-codiert. Die Simulation entspricht der folgenden Geschäftslogik:

1. Es sollen zwei Files von einem Server herunter geladen werden
2. Ein bestimmtes Stück Information wird aus jedem der beiden herunter geladenen Files extrahiert
3. Diese Information wird kombiniert und ergibt das dritte File das herunter geladen werden soll
4. Die Daten aus dem dritten File sollen in der Developer console ausgegeben werden

In der Simulation geben die beiden ersten Funktionen einen Promise zurück, die Funktion welche das dritte File runterlädt akzeptiert aber nur einen callback.

Die Funktionen im Programm sind (alle Datei-Transfers sind simuliert):

- `initialRequestA()` holt eine Datei vom Server und gibt einen Promise zu einem a Objekt zurück
- `initialRequestB()` holt eine Datei vom Server und gibt einen Promise zu einem b Objekt zurück
- `getOptionsFromInitialData(a, b)` gibt das für `finalRequest` benötigte `options` Argument zurück
- `finalRequest(options, callback)` holt das dritte File vom Server und ruft `callback(error, data)` auf, wenn die Operation beendet ist. Bei einem Fehler ist `data` *undefined* und `error` ist *undefined* wenn kein Fehler vorliegt.[1]

In dem File `PromisingRequests.js` sollen die vorgegebenen Funktionen nicht verändert werden.

[1] In Node.js ist dies ein gängiges Pattern.

Erwartete Resultate und Abgabe

- Adaptierung der Funktion `finalRequest` zu einem Promise `finalRequestPromise`.
- Aufruf der Funktionen in der richtigen Reihenfolge mit Hilfe von Promises.
- Bei Erfüllung analoger Kriterien kann das Resultat auch in ihr Projekt aus der Projektschiene integriert werden.
- Abgabe während dem Labor.

Tasklisten Applikation (freiwillig)

Ziel der Übung

- Sie können bestehende Applikationen mit Promises optimieren
- Sie können mit den Jquery Deferred Objekten umgehen

Aufgabe(n)

Analysieren Sie den Source Code ihrer Tasklisten Applikation aus den vorhergehenden Übungen. Wo bietet sich der Gebrauch von Promises anstatt Callbacks an? Finden Sie solche Bereiche im Code und schreiben Sie diese zu Promises um.

Bitte beachten Sie dass viele Jquery Funktionen schon Promises zurückgeben. Dies sind jedoch keine ECMA 6 Promises sondern Jquery Deferred Objekte die leicht anders funktionieren: http://api.jquery.com/category/deferred-object/

Erwartete Resultate und Abgabe

- Adaptierung der Applikation, wenn möglich inklusive Testsuite.
- Abgabe während dem Labor.