Projektbeschreibung

API DOC

**Autor:** Jiri Valenta

**Datum:** 20.12.2017

**Ort:** Wien

**Applikationname:** API DOC

**Kurs:**Ausbildung zum Web Developer - JavaScript - Diplom-Lehrgang

**Kurskode:**18294507

**Thema:** Speichern von detaillierten technischen Information für AJAX Kommunikation zwischen Frontend (JavaScript, HTML, CSS, Frameworks) und Backend.

**Ziel:** Verbesserung der Kommunikation zwischen den front- und backend Entwicklern. Da die Parameter einer Netzwerkkommunikation sehr komplex sind, passieren sehr oft Probleme bei der Kommunikation zwischen den Schnittstellen auf beiden Seiten. Die Fehlerbehebung ist oft sehr kompliziert und es ist auch nicht klar, an welcher Seite überhaupt die Fehler passieren. Eine am besten online editierbare Kommunikation-Schnittstelle hilft, ähnlich wie bei Cloud-basierten Software-tools, sofortige Änderungen auf allen Seiten zu beobachten und die Fehlersuche zeitlich zu minimieren.

**Verwendete Technologien:**

**Frontend:**HTML,CSS, JavaScript, JQuery, JQuery UI, JSON, AJAX

**Backend:** Node.js (mit Express, fs und body-parser), JQuery, JSON, AJAX

**Entwicklungsumgebung:** Windows 10, Atom Editor, Firefox, Node.js

**Frontendbeschreibung**: Der Design vom Frontend ist absichtlich schlicht und möglichst übersichtlich gehalten (Datei **indexProjekt.html** in client Ordner). Die Auswahl von Projekten und Connections wird mit Hilfe von JQuery UI platzangebotsmässig minimiert. Die Projekte werden einzeln vom Backend geholt und im Frontend als ein JavaScript Objekt zwischengespeichert. Um die Daten per asynchrone AJAX Kommunikation zu schicken, wir das JavaScript Objekt im Backend zuerst als JSON Objekt stringifiziert und per Ajax geschickt. Frontend entpackt das JSON Objekt und speichert es mit Hilfe von JSON.parse Funktion wieder als JavaScript. Alle Felder, die im Projekt und im Connection-Objekt verwendet werden, sind objektorientiert gespeichert. Nachdem der Benutzer das Eingabefeld verlassen hat, wird mit Hilfe von „on lost focus“ die Speicherung von den geänderten Informationen ausgeführt und per AJAX in der Textdatei gespeichert. Um Bandbreite zu schonen, wird nicht die ganze gespeicherte Information vom Server geholt, sondern nur Projektweise (für Projektcombo wird Array nur mit id, name, und detail geschickt). Diese Liste wird in der Combobox gespeichert und ermöglicht, andere Projekte zu wählen. Erst dann wird nur das aktuelle Projekt vom Server geholt und das Projekt wird dem Benutzer auf der Arbeitsfläche visualisiert. Zuerst wird die Funktion updateProjectList aufgerufen, die zuständig für den Aufbau von der Project-Auswahl zuständig ist. Danach wird die Funktion showProjectDetail aufgerufen, die den Connection-Teil aufbaut. Das ganze Frontend-Objekt wird mit Hilfe von der Funktion saveCurrentProjectToServer gespeichert. Es wird zum speichern, anlegen und löschen Dialog-Technik von JQuery UI verwendet. HTML-Elemente, die von JQuery UI nicht unterstützt werden (z.B. textarea, label und input), sind wegen einheitlichem Design mit CSS-Klassen von JQuery UI verknüpft. Einzelne Projekte werden von der Klasse „Projekt“ instanziert, die wichtigste Struktur wird objektorientiert angelegt.

**Backendbeschreibung:** Als Server für Backend dient Node.js. Server wird mit dem Befehl „node serverstart.js“ gestartet. Danach erfolgt der Zugriff per Browser auf „localhost“ mit dem Port 26893 (<http://localhost:26893/projectsND> ). Detailierte Beschreibung von node.js wird in der Schnittstellendokumentation beschrieben.