OSTBAYERISCHE TECHNISCHE HOCHSCHULE REGENSBURG

INFORMATIK UND MATHEMATIK

Übungsaufgaben Blatt 11 - Serverseitige Entwicklung mit Node.js und Express

Medieninformatik Webtechnologien - Prof. Dr. Markus Heckner

Aufgabe 1 : Hello Kermit! Hosting einer statischen Website Node.js und Express

Alle bisher entwickelten Anwendungen waren auf einen von Brackets bereitgestellten Webserver angewiesen. D.h. Brackets startet einen eigenen Dienst, der die Website (inkl. JavaScript, CSS, Bildern und weiteren Assets) für den Zugriff durch den Browser (z.B. auf `http://127.0.0.1:55986/index.html`) zur Verfügung stellt.

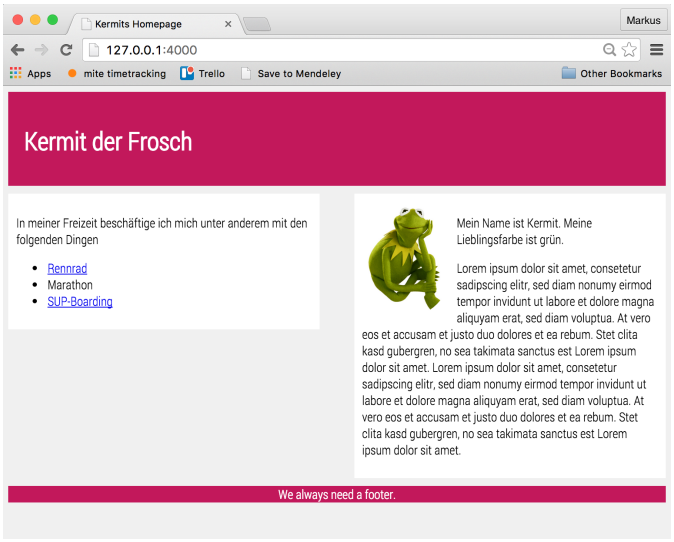
In dieser Aufgabe sollen Sie einen eigenen Webserver mit Node.js und Express entwickeln, der die HTML Seite anstatt des Brackets-Servers an den Client ausliefert.
Für diese Aufgabe steht Ihnen eine fertige kleine Website zum Download in Grips zur Verfügung (Starterprojekt Static Content Express).
Gehen Sie wie folgt vor:

- Erstellen Sie ein neues Verzeichnis für Ihr Projekt und wechseln Sie auf der Kommandozeile auf diesen Ordner.
- Installieren Sie in diesem Ordner mit `npm` das Paket *express*.
- Erstellen Sie das Verzeichnis `public_html` und kopieren Sie den HTML-Code und den JavaScript Code aus dem Starterprojekt in dieses Verzeichnis.
- Legen Sie eine Datei für den Server an `server.js`. Erstellen Sie jetzt den statischen Server mit *express* und testen Sie, ob die HTML Seiten korrekt ausgeliefert werden.

Ihre Projektstruktur sollte wie folgt aufgebaut sein:

```
project directory
├── node_modules
│   └── express
├── public_html
│   ├── css
│   ├── fonts
│   ├── img
│   └── index.html
└── server.js
```

Die fertig gehostete Website sollte wie folgt dargestellt werden:



Finden Sie jetzt wieder Ihre IP heraus und testen, ob Sie mit anderen Clients gegenseitig auf Ihren Webserver zugreifen können.

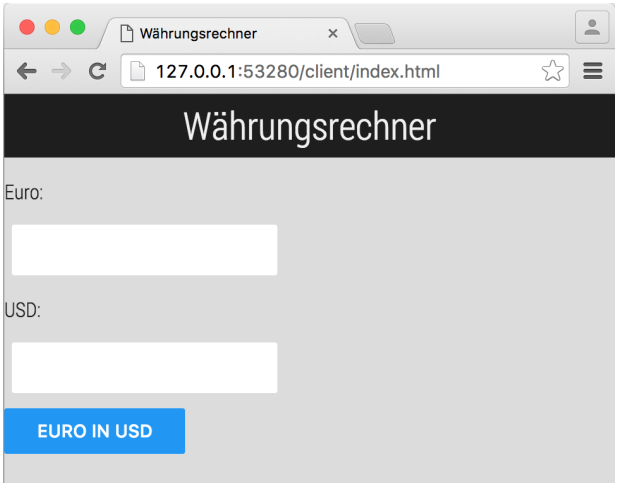
Aufgabe 2 : REST API für einen Währungsrechner mit Node.js und Routing (ohne Express)

Implementieren Sie ein REST-API für einen Währungsrechner. Verwenden Sie für diese Aufgabe das bereitgestellte Starterprojekt aus Grips. Dort finden Sie den vollständigen clientseitigen HTML Code (*Starterprojekt Currency Converter*).

Ihr Server soll Antworten nach dem folgenden Muster beantworten:
`http://127.0.0.1:4000/convertcurrency?originalCurrency=100`
In diesem Beispiel soll eine Währung mit einem Wert von 100 in eine Zielwährung umgerechnet werden.

Achten Sie darauf, dass Ihre Serveranwendung nur Anfragen an den Pfad `convertcurrency` entgegenimmt und beantwortet (Routing). Für Anfragen an andere Pfade soll Ihre Anwendung den Statuscode 404 im *http-header* zurückgeben und so verdeutlichen, dass auf diesem Pfad keine Funktionalität zur Verfügung gestellt wird.

Ihre Seite sollte wie auf der folgenden Abbildung aussehen:



In dieser einfachen Version reicht es, wenn der Server nur eine URL bereitstellt, die einen Euro-Wert in USD umrechnet.
Der aktuelle Wechselkurs ist wie folgt: 1 € = 1,12525 US-Dollar.

Eine Beispielantwort des Servers sollte wie folgt aussehen:

```
{
    "usd" : 112.525
}
```

Erweiterungen (optional)

- Lagern Sie den Code zur Berechnung des Währungskurses in ein eigenes Modul aus und importieren Sie das neu angelegte Modul mit `require`. Orientieren Sie sich an den Beispielen aus der Vorlesung.
- Optional: Erweitern Sie die Aufgabe so, dass man beliebig zwischen Euro und USD umrechnen kann. Hierzu könnten Sie z.B. verschiedene Pfade als Funktionen anbieten (z.B. `eurotousdconversion` und `usdtoeuroconversion`) und die Anfragen entsprechend routen.

Aufgabe 3 : Hosting einer vollständigen Website mit statischem Content und dynamische Anfragen und Antworten mit Node.js
Laden Sie sich die Musterlösung der vorherigen Aufgabe aus Grips
(*Starterprojekt Currency Converter vollständig mit Express*).
Erstellen Sie einen Node.js / Express Server, der die folgenden Anforderungen erfüllt:

- Ausliefern des statischen Contents aus einem Verzeichnis `public_html`. Hier soll die Website inkl. des clientseitigen JavaScript-Codes an den Client ausgeliefert werden.
- Implementieren Sie die folgenden Routen:

- `/` Ausliefern der statischen Inhalte aus `public_html`
- `/convertcurrency` zur Konvertierung der Währung in USD (identische Funktionalität wie in der vorhergehenden Aufgabe)

Gehen Sie bei der Bearbeitung wie folgt vor:

1. Erstellen Sie ein neues Verzeichnis für Ihr Projekt und wechseln Sie auf der Kommandozeile auf diesen Ordner.
2. Installieren Sie in diesem Ordner mit `npm` das Paket *express*.
3. Erstellen Sie das Verzeichnis `public_html` und kopieren Sie den HTML-Code und den JavaScript Code aus dem Starterprojekt in dieses Verzeichnis.
4. Erstellen Sie jetzt den statischen Server mit *express* und testen Sie, ob die HTML Seiten korrekt ausgeliefert werden.
5. Ergänzen Sie erst jetzt das Routing sowie die Verarbeitung und Beantwortung der Anfragen des Clients. Achtung: Das Routing soll jetzt durch die von Express bereitgestellte Funktionalität erfolgen (vgl. Foliensatz aus der Vorlesung).

Ihre Ordnerstruktur sollte so ähnlich wie folgt aussehen:

```
project directory
├── node_modules
│   └── express
├── public_html
│   ├── css
│   ├── js
│   ├── index.html
│   └── ...
├── currencyConverter.js
└── server.js
```