WebApp mit Node und Express

1 Installation von Node.js

Download

nodejs.org

Test

`$ node -v` v8.9.1
`$ npm -v` 5.5.1

2 Projekt erstellen

`$ npm init` erstellt im aktuellen Verzeichnis die Datei *package.json*

3 Hallo Welt in Konsole

index.js

```
console.log('Hallo Welt!');
```

Script starten

```
$ node index.js
```

4 Webserver nur mit Node

index.js

```
const http = require("http");

http.createServer(function(request, response) {
   response.writeHead(200, {'Content-Type': 'text/plain'});
   response.end('Hallo Welt!');
}).listen(3000);

console.log('Server läuft auf Port 3000');
```

Script starten

```
$ node index.js
```

Mit Browser auf Webserver zugreifen

```
http:\\localhost:3000
```

5 Webanwendung mit express

express installieren

`$ npm install express --save` Installation im aktuellen Verzeichnis, Dependency in package.json

index.js

```
const express = require('express');
const app = express();

app.get('/', function(req, res) {
  res.send('Hallo Welt!');
});

app.listen(3000, function() {
  console.log('Server läuft auf Port 3000');
});
```

Webserver starten

```
$ node index.js
```

Mit Browser auf Webserver zugreifen

`http:\\localhost:3000`

6 Webanwendung mit express und index.html

index.html

```
<!doctype html>
<html>
...
```

index.js

```
...
  res.sendFile(__dirname + '/index.html');
...
```

Webserver neu starten

STRG + C

```
$ node index.js
```

7 Webanwendung mit express und ejs

ejs installieren

`$ npm install ejs --save` Installation im aktuellen Verzeichnis, Dependency in package.json

Template: views/index.ejs

```
<html>
....
<p>Guten Tag, <%= person.vorname %> <%= person.nachname %>!</p>
```

index.js

```
const express = require('express');
const app = express();

var mensch = { vorname: 'Franz', nachname: 'Kohnle' };

app.set('view engine', 'ejs');

app.get('/', function(req, res){
    res.render('index', { person: mensch });
});

app.listen(3000, function(){
    console.log('Server läuft auf Port 3000');
});
```

Webserver starten

```
$ node index.js
```

Mit Browser auf Webserver zugreifen

```
http:\\localhost:3000
```

8 Statische Dateien ausliefern

Verzeichnisstruktur

```
projekt/
   |- index.js
   |- package.json
   |
   |- views/
   |    |- index.ejs
   |
   |- public/
        |- layout.css
```

index.js

```
...
app.use(express.static(__dirname + '/public'));
...
```

index.ejs

```
...
<link rel="stylesheet" type="text/css" href="./layout.css" />
...
```

layout.css

```
...
```

9 Datenbank mit MongoDB

Server starten

```
$ mongod --dbpath projekt/data/
```

Client starten

```
$ mongo
```

Datenbank erstellen

`> use todolist`	Datenbank
`> db.createCollection('aufgaben')`	Collection

Datensätze erstellen

```
db.aufgaben.insert( text: 'Kuchen backen' )
db.aufgaben.insert( text: 'Brotzeit machen' )
db.aufgaben.insert( text: 'Blöd schauen' )
```

10 In Webanwendung auf MongoDB zugreifen

mongodb installieren

`$ npm install mongodb --save` Installation im aktuellen Verzeichnis, Dependency in package.json

Template: views/index.ejs

```
<h1>Todoliste</h1>
<ul>
    <% liste.forEach(function(aufgabe) { %>
        <li><%= aufgabe.text %></li>
    <% }); %>
</ul>
```

index.js

```
const express = require('express');
const app = express();
const MongoClient = require('mongodb').MongoClient;

app.use(express.static(__dirname + '/public'));
app.set('view engine', 'ejs')

app.get('/', function(req, res){
    MongoClient.connect('mongodb://localhost:27017/todolist', function (err, db) {
        if (err) throw err
        db.collection('aufgaben').find().toArray(function (err, result) {
            if (err) throw err
            res.render('index', {liste: result} );
        })
   })
});

app.listen(3000, function(){
    console.log('Server läuft auf Port 3000');
});
```

Webserver starten

```
$ node index.js
```

Mit Browser auf Webserver zugreifen

```
http:\\localhost:3000
```

11 Webserver nur starten, wenn Verbindung zu MongoDB steht

index.js

```
const express = require('express');
const app = express();

app.use(express.static(__dirname + '/public'));
app.set('view engine', 'ejs');

const MongoClient = require('mongodb').MongoClient;

// Verbindung mit Datenbank aufbauen
var db;

MongoClient.connect('mongodb://localhost:27017/todolist', function (err, database) {
    if (err) throw err;
    db = database;
    // Webserver startet nur, wenn Datenbankverbindung ok
    app.listen(3000, function(){
        console.log('Höre auf Port 3000');
    });
})

app.get('/', function(req, res){
    db.collection('aufgaben').find().toArray(function (err, result) {
        if (err) throw err
        res.render('index', {liste: result} );
    });
});
```

12 Neue Datensätze mit Formular übergeben

body-parser installieren

`$ npm install body-parser --save` Zum Lesen des HTTP-Bodys

index.ejs

```
<h2>Neuen Eintrag anlegen</h2>
<form action = "/" method="POST">
    <input type="text" name = "text" placeholder="Bitte neue Aufgabe eingeben" />
    <input type="submit" value="Neu" />
</form>
```

index.js

```
const bodyParser= require('body-parser');
app.use(bodyParser.urlencoded({extended: true}));
...
app.post('/', function(req, res){
    console.log(req.body);
});
```

13 Neue Datensätze in MongoDB speichern

index.js

```
...
app.post('/', function(req, res){
    db.collection('aufgaben').save(req.body, function(err, result){
        if (err) return console.log(err);
        res.redirect('/');
    });
});
```

14 Löschen-Button im Formular

index.ejs

```
<h1>Todoliste</h1>
<ul>
    <% liste.forEach(function(aufgabe) { %>
        <li><%= aufgabe.text %>
            <form action="/delete" method="POST">
                <input type="hidden" name="id" value="<%= aufgabe._id %>">
                <input type="submit" value="Löschen" />
            </form>
        </li>
    <% }); %>
</ul>
...
```

15 Datensatz in MongoDB löschen

index.js

```
var ObjectId = require('mongodb').ObjectID;
...
app.post('/delete', function(req, res){
    var id = req.body.id;
    var aufgabe = { _id: ObjectId(id)};
    db.collection('aufgaben').remove(aufgabe, function(err, result){
        if (err) return console.log(err);
        res.redirect('/');
    });
});
```

Quellen

- MDN: Express/Node introduction
- Building a Simple CRUD Application with Express and MongoDB