Angular Random Service Blog

Übersicht

Mit dem Blog Angular Hello World haben wir eine erste einfache Angualar Anwendung erstellt. Nun zeigen wir auf, wie eine Angular Anwendung einen Golang JSON Service verwendet.Wir erstellen hierzu eine neue Angular Applikation "randomserviceapp", als Service steht uns der Golang Random Service unter dem folgenden Link zur Verfügung:http://www.simtech-ag.ch/std-ajax/randomservice?min=0&max=1000Der Service erstellt eine Random Number im Bereich min und max und gibt das folgende JSON Format zurück:{
  "time": "2021-07-09T16:53:06",
  "value": 9.382119599871013
}Unsere Angular Anwendung soll den Simtech Random Service mit den Parametern in einem Formular erfasst verwenden und das Resultat anzeigen.Für die lokale Entwicklung arbeiten wir mit der folgenden Architektur:Die Angular Applikation wird über den Angular Proxy konfiguriert durch die Datei proxy.conf.json auf den Simtech Random Service umgeleitet.Dieser Blog zeigt den kompletten Weg zum gesuchten Ziel inkl. Demo und Lösung.

NodeJS

Für die Entwicklung von Angular Anwendungen benötigen wir eine NodeJS Installation .Die exakte Anleitung hierzu findet man direkt unter dem Link https://nodejs.orgAm besten installieren Sie die LTS Version. Verifizeren Sie nach der Installation die NodeJS Version wie folgt über die Command Line Konsole:node --versionDie Anfrage sollte die installierte NodeJS Version anzeigen, z.B:v14.17.1

Angular CLI

Das Angular-Team hat ein Command Line Tool (CLI) entwickelt, um das Bootstrap und die Entwicklung Ihrer Angular-Anwendungen zu vereinfachen. Da es den Entwicklungsprozess erheblich erleichtert, empfehlen wir dringend, es für die ersten Angular-Projekte zu verwenden. Die Angular CLI Installation erfolgt mit dem folgenden Command Line Befehl:npm install -g @angular/cliWir verifzieren die Angular CLI Installation mit dem folgenden Befehl:ng --versionDie Ausgabe zeigt detailliert die Angular Packages an:Angular CLI: 12.1.0
Node: 14.17.1
Package Manager: npm 6.14.13
OS: win32 x64

Angular:
...

Package Version

@angular-devkit/architect 0.1201.0 (cli-only)
@angular-devkit/core 12.1.0 (cli-only)
@angular-devkit/schematics 12.1.0 (cli-only)
@schematics/angular 12.1.0 (cli-only)

Text Editor

Angular Anwendungen werden mit einem Text Editor programmiert. Es gibt verschiedene Tools, die für Angular-Entwickler entwickelt wurden, aber nur sehr wenige Tools helfen Entwicklern, Hier sind die Top 8 der Angular-Entwicklungs-IDE, mit der Sie herausragende Webprojekte erstellen können:Angular IDEWebstormVisual Studio CodeBracketsAtomSublime TextALM IDEAptana StudioWir empfehlen den Einsatz von Webstorm oder Visual Studio Code. Für diesen Blog verwenden wir Visual Studio Code.

Random Service Application erstellen

Erstellen Sie einen Ordner für Ihre Anwendung am gewünschten Ort auf Ihrem System und öffnen Sie ihn in VSCode. Wir verwenden unter Windows den Pfad c:\angularblog. Öffnen Sie ein neues Terminal und wechseln Sie in den zuvor erstellten Pfad. Geben Sie den folgenden Befehl Angular CLI ein, um Ihren App-Ordner zu erstellen:ng new randomserviceappVerwenden Sie das Angular Routing (y) und das Stylesheet Format SCSS (Achtung, nicht die default Wahl).Die Angular Applikation wird erstellt und die Packages geladen. Am Schluss sollte die folgenden Dateistruktur enthalten sein:Der Folder enthält zugleich auch die Daten für das Git Repository, mit dem Parameter --skip-git wird solches nicht erstellt.Mehr Infos zum Befehl ng new finden Sie unter dem Link https://angular.io/cli/new.

Random Service Application ausführen

Um die Anwendung auszuführen, ändern Sie das Verzeichnis in den erstellten Ordner und verwenden Sie den folgenden Befehl:ng serveÖffnen Sie nach der Ausführung Ihren Browser und navigieren Sie zu localhost:4200. Wenn eine andere Anwendung auf dieser Adresse läuft, können Sie den Befehl einfach ausführen:ng serve --portEs wird einen Port generieren, zu dem Sie navigieren können. Normalerweise sieht der Browser etwa so aus:

Angular Proxy Configuration

Für de Angular Proxy definieren wir im Root Projektverzeichnis die Datei proxy.conf.json mit dem folgenden Inhalt:{
  "/std-ajax": {
    "target": "http://www.simtech-ag.ch",
    "secure": false,
 "headers": { "host": "www.simtech-ag.ch" }
  }
}
In der Datei angular.json tragen wir die proxy.conf.json Datei ein:"serve": {
 "builder": "@angular-devkit/build-angular:dev-server",
 "configurations": {
    "production": {
      "browserTarget": "randomserviceapp:build:production"
    },
    "development": {
  "browserTarget": "randomserviceapp:build:development",
 "proxyConfig": "proxy.conf.json"
    }
  },
 "defaultConfiguration": "development",
},Wir definieren die Proxy Config nur im Development Mode, also nur für die lokale Entwicklung.In der package.json Datei definiert wird das Start Script wie folgt:"start": "ng serve --proxy-config proxy.conf.json",Wir starten die Applikation neu via dem folgenden Befehl:npm startDie Anwendung sollte nun unter Berücksichtigung der Proxy Configuration aufstarten. Damit ist der Proxy aktiv und definiert.

Angular Random Service

Für die Anbindung der Angular Applikation an den Random Backend Service erstellen wir den Random Angular Service. Hierzu öffnen wir ein Terminal (Konsole) und wechseln im randomserviceapp Folder in den Pfad src/app und erstellen das Verzeichnis services.Wir wechseln in das services-Verzeichnis und erstellen den Angular Random Service mit dem folgenden Befehl:ng generate service randomDer Befehl erstellt die Dateien random.service.ts und random.service.spec.ts. Wir erhalten die folgende Struktur:Wir öffnen die Datei random.service.ts und definieren den Service gemäss dem folgenden Listing:import { Injectable } from '@angular/core';
import { HttpClient } from "@angular/common/http";
import { Observable } from "rxjs";

export class RandomDTO {
  time: string;
  value: string;

  constructor() {
    this.time = "";
    this.value = "";
  }
}

@Injectable({
  providedIn: 'root'
})
export class RandomService {

  constructor(private httpClient: HttpClient) { }

  public getRandom(min: number, max: number): Observable<RandomDTO> {
    return this.httpClient.get<RandomDTO>('/std-ajax/randomservice?min=' + min + '&max=' + max,);
  }

}Die Klasse RandomDTO bildet die JSON Antwort des Backend Services ab. Der Zugriff auf das Backend erfolgt via HttpClient Klasse, solche gibt die Antwort asynchron via Observable zurück. Den Einsatz des RandomService sehen wir im nächsten Step und der Random Component.

Angular Random Component

Angular verwendet Komponenten für die HTML Darstellung. Unsere Random Site soll über die RandomComponent abgebildet werden. Hierzu öffnen wir ein Terminal (Konsole) und wechseln im randomserviceapp Folder in den Pfad src/app und erstellen das Verzeichnis components.Wir wechseln in das components-Verzeichnis und erstellen die Angular Random Component mit dem folgenden Befehl:ng generate component randomDer Befehl erstellt die Dateien random.component.ts, random.component.scss, random.component.html und random.component.spec.ts. Wir erhalten die folgende Struktur:Wir öffnen die Datei random.component.ts und definieren den Service gemäss dem folgenden Listing:import { Component, OnInit } from '@angular/core';
import {FormControl, FormGroup} from "@angular/forms";
import {RandomDTO, RandomService} from "../../services/random.service";

@Component({
  selector:

'random-component',
  templateUrl: './random.component.html',
 styleUrls: ['./random.component.scss']
})
export class RandomComponent implements OnInit {

  constructor(private randomService: RandomService) { }

  randomDTO: RandomDTO;
  randomForm: FormGroup;

  ngOnInit(): void {
 this.randomDTO = new RandomDTO();
    this.randomForm = new FormGroup({
      min: new FormControl('0'),
      max: new FormControl('1'),
    });
 }

  randomClick() {
    const min = this.randomForm.get('min')!.value;
    const max = this.randomForm.get('max')!.value;
    this.randomService.getRandom(min, max).subscribe(result => this.randomDTO = result);
  }
}Der Compiler zeigt einen Initialisierungfehler, den beheben wir indem wir die Datei tsconfig.json öffnen und wie folgt ergänzen:"strictPropertyInitialization": false,Diese Option setzen wir unter die "compilerOptions".Jetzt programmieren wir den HTML Content der RandomComponent in der Datei random.component.html wie folgt:<div class="center">
 <h2>Random Service</h2>
  <form [formGroup]="randomForm">
    <section>
 <div>
      <mat-form-field>
 <mat-label>Min</mat-label>
        <input matInput type="number" placeholder="Min" formControlName="min" min="0" max="100000">
 </mat-form-field>
      <mat-form-field>
 <mat-label>Max</mat-label>
        <input matInput type="number" placeholder="Max" formControlName="max" min="1" max="100000">
 </mat-form-field>

      <button mat-raised-button color="primary" (click)="randomClick()">Get Random from Service</button>
    </div>
 </section>
  <section>
    <div>
 <h3>Random Result</h3>
      <h3>Time : {{randomDTO.time}}</h3>
 <h3>Value : {{randomDTO.value}}</h3>
 </div>
  </section>
 </form>
</div>Die RandomComponent ist bereit für den Einsatz, siehe Next Step.

AppComponent und Routing

Unsere Anwendung wird durch die RandomComponent bestimmt, die AppComponent soll nur das Routing auf die RandomComponent übernehmen. Wir öffnen die Datei app.component.html und reduzieren die HTML Datei auf das Router Outlet wie folgt:<router-outlet></router-outlet>In der Datei app-routing.module definieren wir das Routing auf die RandomService Component wie folgt:import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import {RandomComponent} from "./components/random/random.component";

const routes: Routes = [
  { path: '', component: RandomComponent }
];

@NgModule({
 imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }Jetzt fehlt uns noch die Definition der verwendeten Module und insbesondere Angular Material. Zuerst installieren wir das Angular Material wie folgt in unsere Umgebung: ng add @angular/materialVerwenden Sie das Default Theme Indigo/Pink und die Browser Animations.Öffnen Sie die Datei app.module.ts und definieren Sie den Content wie folgt:import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { MatButtonModule } from '@angular/material/button';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { HttpClientModule } from "@angular/common/http";
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
import {MatSliderModule} from "@angular/material/slider";
import {MAT_FORM_FIELD_DEFAULT_OPTIONS, MatFormFieldModule} from "@angular/material/form-field";
import

{ReactiveFormsModule} from "@angular/forms";
import {MatInputModule} from "@angular/material/input";
import {RandomComponent} from "./components/random/random.component";

@NgModule({
 declarations: [
 AppComponent,
 RandomComponent
 ],
 imports: [
 HttpClientModule,
 BrowserModule,
 MatButtonModule,
 AppRoutingModule,
 BrowserAnimationsModule,
 MatSliderModule,
 MatFormFieldModule,
 ReactiveFormsModule,
 MatInputModule
 ],
 providers: [
 {provide: MAT_FORM_FIELD_DEFAULT_OPTIONS, useValue: {floatLabel: 'always'}}
 ],
 bootstrap: [AppComponent]
})
export class AppModule { }Jetzt fehlen noch die Styles. Hierzu öffnen wir die Datei styles.scss und definieren die folgenden Styles:

html, body {
 height: 100%;
}

body {
 margin: 0;
 font-family: Roboto, "Helvetica Neue", sans-serif;
 background-color: #669acc;
 color: white;
}

mat-form-field {
 padding: 1em;
}

mat-form-field label {

 color: white !important;
 font-size: large !important;
}

.center {
 text-align: center;
}

Application Start

Die Random Service App ist ready und startbereit. Wir starten die Anwendung wie folgt:npm startVia Browser und der URL http://localhost:4200 sollte die Random Service Component verwendbar sein:

Demo

Die laufende Demo dieser im Blog erstellten Anwendung finden Sie unter dem Link randomserviceapp.Die Applikation wurde für die Distribution wie folgt gebuildet:ng buildDer Build erstellt die Anwendung im dist Verzeichnis, solches kann auf dem Web Server installiert und ausgeführt werden. Für das effektive Deployment der Demo haben wir die base URL in der Datei index.html wie folgt angepasst:<base href="./randomserviceapp">

Lösung

Die Angular Ramdom Service Application finden Sie unter dem Link randomserviceapp.zip. Wenn Sie die Lösung lokal entpacken müssen Sie die node_modules via Command Line wie folgt installieren:npm installDas Script muss im Folder der randomserviceapp ausgeführt werden. Danach können Sie die Anwendung wieder starten:npm start

Feedback

War dieser Blog für Sie wertvoll. Wir danken für jede Anregung und Feedback

Kontakt

Simtech AG
Finkenweg 23
3110 Münsingen
Schweiz

Impressum

Das Copyright für sämtliche Inhalte dieser Website liegt bei Simtech AG, Schweiz.
Beachten Sie auch unsere Hinweise zum Urheberrecht, Datenschutz und Haftungsauschluss.
Jeder Hinweis auf Fehler nehmen wir gerne entgegen.

Copyright

2024 Simtech AG, All rights reserved, Powered by stack.ch written in Golang by Daniel Schmutz

https://www.simtech-ag.ch/getrandom