Angular Cheatsheet

Important Comands

Neues Angular Projekt

```
ng new
```

Fehler beim Angular Projekt erstellen

```
npm install --save-dev @angular-devkit/build-angular
```

Backend starten mit Docker

```
docker build -t <irgendein name> .
```

```
docker run -p 80:80 <irgendein name>
```

Angular Material

Material Design components for Angular

Neues Component

```
ng generate component
```

Neue Klasse

```
ng generate class
```

Routing

in app-routing-module

```
const routes: Routes = [
  { path: 'first-component', component: FirstComponent },
  { path: 'second-component', component: SecondComponent },
];
```

```
<router-outlet></router-outlet>
```

Angular Routing Site

Neue Klasse in Model

```
ng generate class
```

Boilerplate Code

Mat-Table

```
<table mat-table  class="mat-elevation-z8" [dataSource]="this.dataService.dataSource">

  <ng-container matColumnDef="title">
    <th mat-header-cell *matHeaderCellDef> Title </th>
    <td mat-cell *matCellDef="let element">
      {{element.title}}
    </td>
  </ng-container>

  <ng-container matColumnDef="assignedTo">
    <th mat-header-cell *matHeaderCellDef> Assigned To </th>
    <td mat-cell *matCellDef="let element">
      {{element.assignedTo}}
    </td>
  </ng-container>

  <ng-container matColumnDef="dueDate">
    <th mat-header-cell *matHeaderCellDef> Due </th>
    <td mat-cell *matCellDef="let element">
      {{element.dueDate}}
    </td>
  </ng-container>


  <ng-container matColumnDef="description">
    <th mat-header-cell *matHeaderCellDef> Description </th>
    <td mat-cell *matCellDef="let element">
    {{element.description}}
    </td>
  </ng-container>


  <ng-container matColumnDef="done">
    <th mat-header-cell *matHeaderCellDef> Done </th>
    <td mat-cell *matCellDef="let element" >
    <mat-checkbox  [disabled]="true" [checked]="element.status"></mat-checkbox>
    </td>
  </ng-container>

  <ng-container matColumnDef="delete">
    <th mat-header-cell *matHeaderCellDef> Delete </th>
    <td mat-cell *matCellDef="let element" >
     <button mat-raised-button (click)="deleteTodo(element)">Delete</button>
    </td>
  </ng-container>

  <tr mat-header-row *matHeaderRowDef="displayedColumns"></tr>
  <tr mat-row *matRowDef="let row; columns: displayedColumns;"></tr>
</table>
```

displayedColumns fur Mat-Table in TodoListComponent.ts

```
displayedColumns: string[] = ['title', 'assignedTo', 'description', 'dueDate', 'done',
'delete'];
```

headers in DataService.ts

```
const headers = { 'content-type': 'application/json'};
      const temp: any = await this.httpClient.post(this.hostURL + '/todos',
JSON.stringify(td), {headers} ).toPromise();
```

get Request

```
this.persons = await this.httpClient.get<Person[]>(this.hostURL +
'/people').toPromise();
```

Ganze Klassen

TodoListComponent.ts

```
import { Component, OnInit } from '@angular/core';
import {MatTableDataSource} from '@angular/material/table';
import {Todo} from '../../model/todo';
import {DataService} from '../../service/data.service';
import {HttpClient} from '@angular/common/http';

@Component({
  selector: 'app-todo-list',
  templateUrl: './todo-list.component.html',
  styleUrls: ['./todo-list.component.scss']
})
export class TodoListComponent implements OnInit {

  constructor(public dataService: DataService) {
  }

  displayedColumns: string[] = ['title', 'assignedTo', 'description', 'dueDate',
'done', 'delete'];

  async ngOnInit(): Promise<void> {
    await this.dataService.loadTodos();
    await this.dataService.loadPersons();
  }

  async deleteTodo(element: Todo): Promise<void>{
    if (confirm(`Are you sure to delete ${element.title}?`)){
      await this.dataService.deleteTodo(element);
    }
  }

}
```

DataService.ts

```
import { Injectable } from '@angular/core';
import {Todo} from '../model/todo';
import {MatTableDataSource} from '@angular/material/table';
import {HttpClient} from '@angular/common/http';
import {Person} from '../model/person';

@Injectable({
  providedIn: 'root'
})
export class DataService {
  public headers = { 'content-type': 'application/json'};
  public todos: Todo[];
  public persons: Person[];
  public baseURL = 'http://localhost:80/api';
```

```
  public dataSource = new MatTableDataSource<Todo>();
  constructor(public httpClient: HttpClient) {
    this.todos = [];
    this.persons = [];
  }
  async loadTodos(): Promise<void>{
    this.todos = await this.httpClient.get<Todo[]>(this.baseURL +
'/todos').toPromise();
    this.dataSource.data = this.todos;
    console.log(this.todos);
  }
  async deleteTodo(td: Todo): Promise<void>{
    await this.httpClient.delete(this.baseURL + `/todos/${td.id}`).toPromise();
    this.todos.splice(this.todos.indexOf(td), 1);
    this.dataSource.data = this.todos;
  }

  async loadPersons(): Promise<void>{
    this.persons = await this.httpClient.get<Person[]>(this.baseURL +
'/people').toPromise();
  }

  async addTodo(td: Todo): Promise<void>{
    const headers = { 'content-type': 'application/json'};
    const temp: any = await this.httpClient.post(this.baseURL + '/todos',
JSON.stringify(td), {headers} )
      .toPromise()
      .catch(err => {
        alert(err.error.description);
        console.log(err.error.description);
        return;
      });

    td.id = temp.id;
    this.todos.push(td);
    this.dataSource.data = this.todos;
  }
}
```

Todo.ts

```
export class Todo {
  public id: number;
  constructor(public title: string, public assignedTo: string, public description:
string, public dueDate: Date, public status: boolean ) {
    this.id = -1;
  }
}
```

Dockerfile

```
FROM node:lts-alpine as builder
WORKDIR /app
COPY . .
RUN npm install
RUN npm run build
EXPOSE 8080
CMD ["node", "dist/server.js"]
```