Uvod u Angular programski okvir

Uvod

- Angular je platforma i programski okvir za kreiranje single-page frontend aplikacije
- Koristi HTML i TypeScript jezik
- Temelji se na gradivnim blokovima NgModules koji omogućavaju izgradnju komponenata
- Komponente definiraju view-ove, elemente grafičkog sučelja te koriste servise koji omogućavaju funkcionalnosti aplikacije
- Komponente i servisi su jednostavne klase

Moduli i komponente

- Angular NgModuli deklariraju kontekst prevođenja za komponente koje su vezane uz aplikacijsku domenu ili workflow
- Svaka Angular aplikacija sadrži korijenski modul koji se naziva "AppModule" pomoću kojeg se pokreće aplikacija
- NgModuli importaju funkcionalnosti drugih NgModula
- Svaka Angular aplikacija sadrži barem jednu root komponentu koja tvori hijerarhiju komponenti
- Svaka komponenta definira klasu koja sadržava aplikativnu i podatkovnu logiku te je povezan HTML predloškom koji definira view komponentu

Primjer modula AppModule

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';
import { AppComponent } from './app.component';
import { StudentsComponent } from './students/students.component';
import { StudentDetailComponent } from './student-detail/student-detail.component';
@NgModule({
  declarations: [
   AppComponent,
   StudentsComponent,
    StudentDetailComponent
```

Primjer modula AppModule

```
imports: [
    BrowserModule,
    FormsModule
],
    providers: [],
    bootstrap: [AppComponent]
})
export class AppModule { }
```

Primjer komponente AppComponent

```
import { Component } from '@angular/core';

@Component({
   selector: 'app-root',
   templateUrl: './app.component.html',
   styleUrls: ['./app.component.css']
})
export class AppComponent {
   title = 'Student App';
}
```

Primjer app HTML komponente

```
<h1>{{title}}</h1>
<app-students></app-students>
```

Primjer sučelja i polja studenata

```
export interface Student {
   jmbag: string;
   numberOfECTS: number;
   tuitionShouldBePaid: boolean
}
```

Predlošci direktive i povezivanje podacima

- Predlošci kombiniraju HTML s Angular oznakama koje mogu mijenjati HTML elemente prije nego što se prikazuju
- Direktive sadržavaju programsku logiku
- Oznake za povezivanje služe za definiranje veze podataka sa strukturom dokumenta
- Povezivanje može biti realizirano događajima (engl. event binding) i svojstvima (engl. property binding)
- Povezivanje događajima omogućava odgovaranje na unos korisničkih podataka koji se prenose na aplikacijske podatke
- Povezivanje svojstvima prenosi izračunate podatke do HTML-a

Servisi i injektiranje ovisnosti

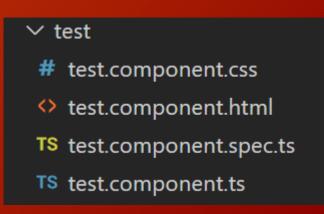
- Za podatke ili logiku koja nije povezana sa specifičnom view komponentom te koja se dijeli među komponentama potrebno je kreirati servisnu klasu
- Definicija servisne klase počinje s dekoratorom @Injectable koji omogućava injektiranje u druge klase

Primjer servisa

```
import { Injectable } from '@angular/core';
import { Student } from './student';
import { STUDENTS } from './mock-students';
import { Observable, of } from "rxjs";
@Injectable({
  providedIn: 'root'
export class StudentService {
  constructor() { }
  getStudents(): Observable<Student[]> {
    return of(STUDENTS);
```

Korištenje Angular sučelja naredbenog retka

- Kako bi se mogao koristiti Angular CLI, potrebno je instalirati Node.js (https://nodejs.org/en/download/) i sam Angular CLI (https://cli.angular.io/)
- Kreiranje nove Angular aplikacije moguće je izvršiti na sljedeći način:
- ng new my-app --no-standalone --routing --ssr=false
- Naredba za kreiranje komponenti može se izvršiti na ovaj način:
- ng generate component test
- Ili:
- ng g c test
- Što generira mapu i sljedeće elemente:
- Na sličan način generiraju se i servisi:
- ng generate service test



Primjer "students" HTML komponente

F	Popis studenata		
	0243235445	120 ECTS	
	0243235337	180 ECTS	
	0243234432	300 ECTS	

Angular direktive

- Direktiva *ngFor služi kao petlja i ispisuje podatke o studentima onoliko često koliko ih je spremljeno u polju "students"
- Naredbom "let student of students" dohvaća se pojedini student i pohranjuje u privremenu varijablu "student"
- Osim toga postoje još direktive poput *nglf koja provjerava logički uvjet

Primjer "students" komponente

```
import { Component, OnInit } from '@angular/core';
import { Student } from '../student';
import { StudentService} from "../student.service";
@Component({
  selector: 'app-students',
  templateUrl: './students.component.html',
  styleUrls: ['./students.component.css']
export class StudentsComponent implements OnInit {
  students: Student[];
  selectedStudent: Student;
  constructor(private studentService: StudentService) { }
```

Primjer "students" komponente

```
ngOnInit(): void {
 this.getStudents();
getStudents(): void {
 this.studentService.getStudents()
    .subscribe(students => this.students = students);
onSelect(student: Student): void {
 this.selectedStudent = student;
```

Primjer "student-detail" HTML komponente

```
<div *ngIf="student">
   <h2>{{student.jmbag}} - detalji studenta</h2>
   <div><span>JMBAG: </span>{{student.jmbag}}</div>
   <div><span>Broj ECTS bodova: </span>{{student.numberOfECTS}}</div>
   <div *ngIf="student.tuitionShouldBePaid"> Student treba platiti školarinu. </div>
</div></div>
```



Primjer "student-detail" komponente

```
import { Component, OnInit, Input } from '@angular/core';
import { Student } from '../student';
@Component({
  selector: 'app-student-detail',
  templateUrl: './student-detail.component.html',
  styleUrls: ['./student-detail.component.css']
export class StudentDetailComponent implements OnInit {
  @Input() student: Student;
  constructor() { }
  ngOnInit(): void {
```

Literatura

 https://www.positronx.io/angular-httpclient-http-tutorial-buildconsume-restful-api/

Pitanja?