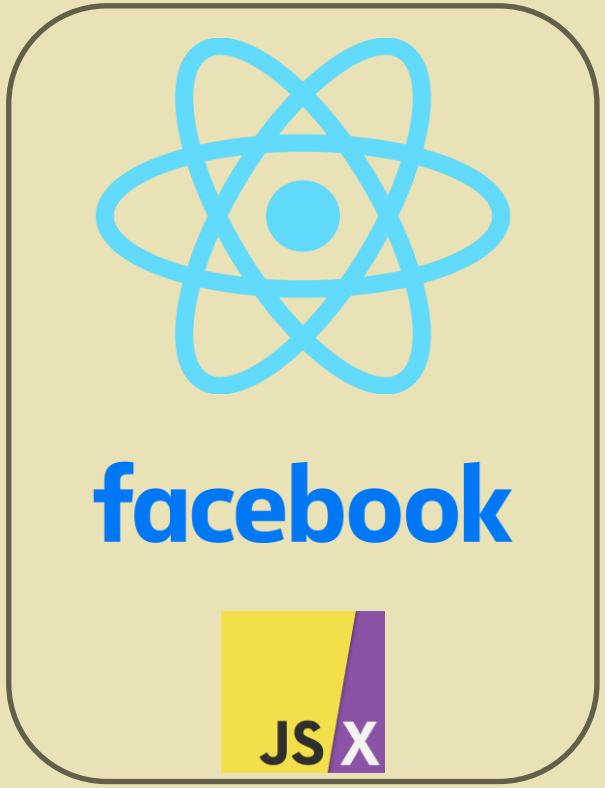
Programiranje internet aplikacija

Elektrotehnički fakultet, Univerzitet u Beogradu

2025/2026

Uvod

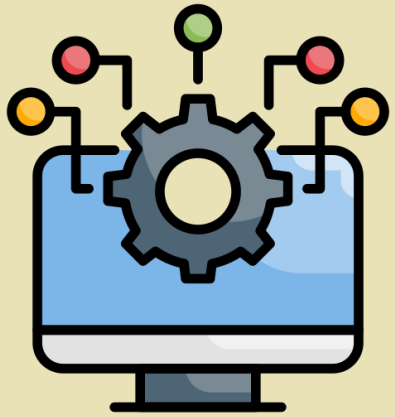
- Front-end JavaScript framework
- Najpoznatiji:



Angular

- TypeScript – proširuje JavaScript tipovima
- Glavni koncepti
  - NgModules
  - Komponente
  - Two-way binding
  - Servisi
  - Dependency injection
  - Rutiranje
- Single page aplikacija

Instalacija, kreiranje i pokretanje aplikacije



- Node.js (22.21.0)
  - https://nodejs.org/dist/v22.21.0/
  - Node package manager

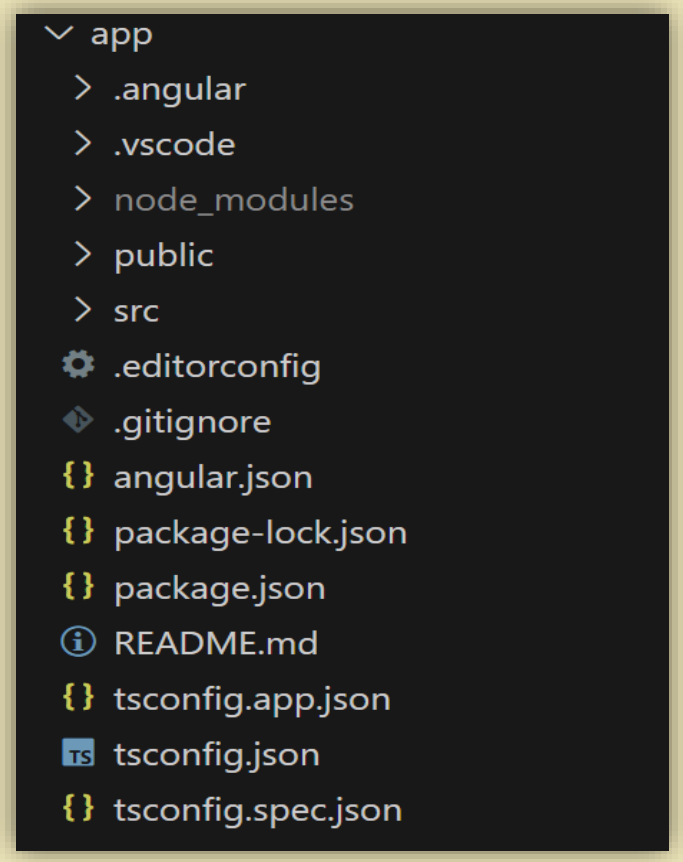
```
npm install –g @angular/cli@20.3.6
```

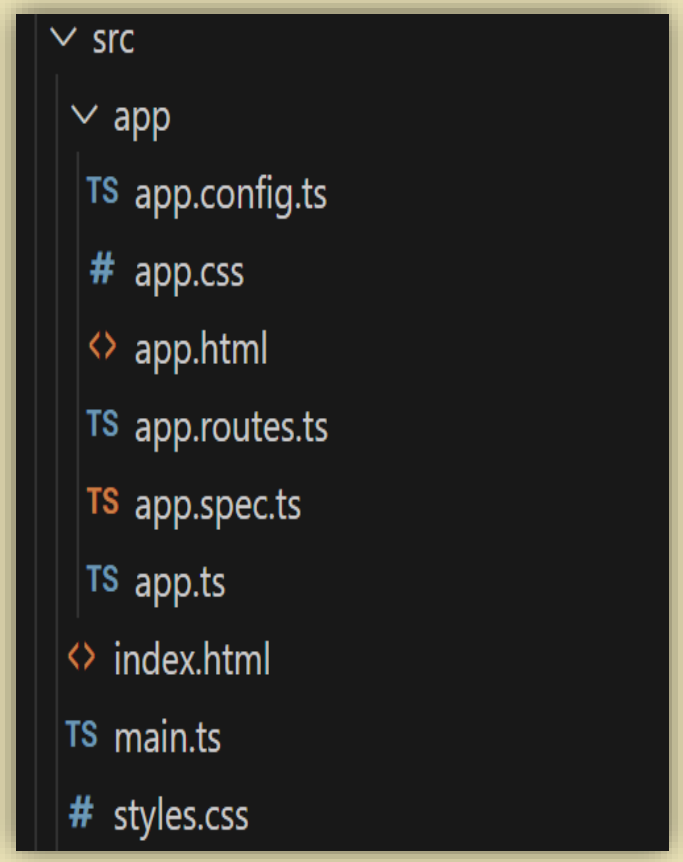
```
ng new naziv_aplikacije
```

- style – CSS
- Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? No
- Do you want to create a 'zoneless' application without zone.js? No
- AI tools? None 🤖

```
cd naziv_aplikacije; npm start
```

Struktura aplikacije





Interpolacija

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

@Component({
  selector: 'app-root',
  imports: [],
  templateUrl: './app.html',
  styleUrl: './app.css'
})
export class App {
  protected readonly title = signal('app');
}
```

Signali

```ts
<h1>Hello, {{ title() }}</h1>
```


```ts
export class App {
  title = 'app'
}
```

```ts
<h1>Hello, {{ title }}</h1>
```

Kreiranje komponenti i rutiranje

- Kreiranje komponenti

ng generate component component_name
(ng g c component_name)

- Rutiranje

```
import { Routes } from '@angular/router';
import { About } from './about/about';
import { Books } from './books/books';
import { Writers } from './writers/writers';

export const routes: Routes = [
    {path: 'about', component: About},
    {path: 'books', component: Books},
    {path: 'writers', component: Writers}
];
```

```
import { Component } from '@angular/core';
import { RouterOutlet, RouterLinkWithHref } from '@angular/router';

@Component({
  selector: 'app-root',
  imports: [RouterOutlet, RouterLinkWithHref],
  templateUrl: './app.html',
  styleUrl: './app.css'
})
export class App {
  title = 'app'
}
```

```
<h3>Welcome {{title}}</h3>
<a routerLink="/about">About</a> |
<a routerLink="/books">Books</a> |
<a routerLink="/writers">Writers</a>
<router-outlet></router-outlet>
```

Servisi

Kreiranje servisa

ng generate service service_name
(ng g s service_name)

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

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

}
```

Korišćenje servisa u komponenti

```
import { Component, inject } from '@angular/core';
import { WritersService } from '../writers-service';

@Component({
  selector: 'app-writers',
  imports: [],
  templateUrl: './writers.html',
  styleUrl: './writers.css',
})
export class Writers {
  private writersService = inject(WritersService)
}
```

Direktive

Klase koje definišu strukturu i ponašanje

- **Direktiva komponenta** – direktive sa šablonom
- **Strukturalne direktive** – direktive koje manipulišu elementima u DOM-u
  - *ngFor, *ngIf, *ngSwitch
  - @for, @if, @else, @switch
  - Jedna strukturalna direktiva po elementu
- **Atributske direktive** – direktive koje manipulišu izgledom elemenata u DOM-u

Filteri (*pipes*)

Specijalni operatori koji omogućavaju transformaciju podataka

Ugrađeni:

- CurrencyPipe
- DatePipe
- LowerCasePipe
- ...

```
<p>
        The event will occur on
        {{ scheduledOn | date | uppercase }}.
</p>
```

```
<p>
        The event will occur at
        {{ scheduledOn | date:’shortDate' }}.
</p>
```

Mogu se kreirati i filteri za specifične potrebe

One-way binding

- **Interpolacija**
- **Property binding**

  <img alt="item" [src]="itemImageUrl"> HTML

  itemImageUrl = "img/newImage.jpg"; TS
- **Attribute binding**

  <button type="button" [attr.aria-label]="actionName">

  {{actionName}} with Aria

  </button>
- **Class and style binding**

  [class.sale]="onSale" true, false

from data source to view target

- **Event binding**

  <button (click)="onSave()">Save</button>

from view target to data source

Prosleđivanje podataka u komponentu (1)

```
@Component({...})
export class CustomSlider {
  @Input() value = 0;
}
```

```
<custom-slider [value]="50" />
```

Prosleđivanje podataka u komponentu (2)

```
@Component({...})
export class CustomSlider {
  @Input({required: true}) value = 0;
}
```

```
@Component({...})
export class CustomSlider {
  @Input({alias: 'sliderValue'}) value = 0;
}
```

```
<custom-slider [sliderValue]="50" />
```

Forme

- reactive vs **template-driven**

```
import { FormsModule }
from '@angular/forms'
```

```
<form>
    <input type="text" name="param" [(ngModel)]="searchParam">
    <button (click)="search()">Search</button>
</form>
```

```
<input type="text" class="form-control" id="param"
       required
       [(ngModel)]="searchParam" name="param" #param="ngModel">

  @if(param.invalid){
      Name is required
  }
```

Kontrolisanje pristupa komponentama

```
export const authenticationGuard : CanActivateFn = (route, state) => {
        const oauthService: AuthService = inject(AuthService);

        if (oauthService.hasAccess() ) {
                return true;
        }

        return false;
        };
}
```

ng generate guard guard_name
(ng g g guard_name)

```
const routes: Route[] = [
  {
        path: 'home',
        component: HomeComponent,
        canActivate: [authenticationGuard]
  }
]
```

Povezivanje sa backend-om

- **ProvideHttpClient**

```
import { provideHttpClient } from '@angular/common/http';

export const appConfig: ApplicationConfig = {
  providers: [ provideBrowserGlobalErrorListeners(),
    provideZoneChangeDetection({ eventCoalescing: true }),
    provideRouter(routes), provideHttpClient()]
};
```

- **HttpClient**

```
import { HttpClient } from '@angular/common/http'
private http = inject(HttpClient)
```

- Uz pomoć HttpClient-a šaljemo HTTP zahteve ka backend-u, a kao odgovor dobijamo Observable

- Observable se izvršava tek kada se na njega subscribe-ujemo

HVALA NA PAŽNJI!