Angular dokumentacija

1. Uvod

Angular je jedan od najpopularnijih razvojnih okvira za razvoj frontend-a koji se koristi za izradu web, desktop i mobilnih aplikacija. Razvijen je od strane Google-a. Koristi se za kreiranje **klijentskog dela** web aplikacije (eng. *client-side framework*) i objedinjuje HTML – koji se koristi za kreiranje strukture, CSS – za stilizovanje, i TypeScript – za logiku aplikacije. Objedinjuje sve neophodne funkcionalnosti i alate za kreiranje interaktivnih i dinamičnih aplikacija sa visokim performansama.

U svojoj prvoj verziji, ovaj framework je nazivan AngularJS, gde sufiks „JS“ znači da se radi o JavaScript framework-u za razvoj klijentskih web aplikacija. Izbacivši „JS“ iz naziva, Google ukazuje na to da se uz razvoj web aplikacija omogućava i razvoj mobilnih i desktop aplikacija, a time Angular postaje univerzalni framework. Napisan je u programskom jeziku TypeScript i njegova svrha je bila da nadomesti nedostatke JavaScript-a. TypeScript kod se ne može direktno izvršavati u pretraživaču, pa se on prevodi (eng. *compiling*) u JavaScript kod. Pored toga, jedna od ključnih karakteristika koja je omogućena kroz TypeScript je navođenje tipova (eng. *strong typing*). Ovo je koncept koji se zasniva na definisanju tipova podataka promenljivih, parametara funkcija i povratnih vrednosti funkcija unutar svog koda. Ovaj proces se vrši u fazi pisanja koda, pre nego što se aplikacija pokrene ili izvrši.

Kao platforma, Angular uključuje:

- Komponente za izradu skalabilnih i dinamičkih web aplikacija
- Raznovrsne biblioteke koje pokrivaju mnoge funkcionalnosti, uključujući rutiranje, upravljanje formama, komunikaciju klijent-server i još mnogo toga
- Skup alata za razvoj, izgradnju, testiranje i ažuriranje koda

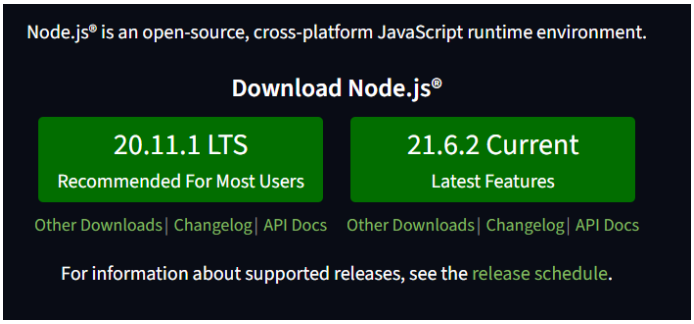
Korišćenje Angulara donosi niz prednosti koje čine ovaj framework popularnim i korisnim za razvoj web aplikacija. Neke od njegovih osnovnih prednosti su:

1. **Kreiran je za velike aplikacije**: Angular je dizajniran s fokusom na skalabilnost, što ga čini idealnim za izgradnju velikih i kompleksnih web aplikacija. Svojom modularnom arhitekturom i alatima za organizaciju koda, Angular olakšava upravljanje složenim projektima.
2. **TypeScript podrška**: Angular je napisan u TypeScriptu, koji donosi statičku tipizaciju i druge napredne funkcije jezika JavaScript, kao što su interfejsi, generici i enums. Ovo poboljšava produktivnost programera, pomažući u detekciji grešaka tokom razvoja i poboljšava održivost koda.

3. **Komponentna arhitektura**: Angular se oslanja na komponentnu arhitekturu, što znači da se korisnički interfejs razbija na male, ponovno upotrebljive komponente. Ovo olakšava organizaciju koda, omogućava lakše testiranje i održavanje aplikacije.
4. **Jednostranične aplikacije (SPA)**: Angular je idealan za izgradnju jednostraničnih aplikacija (SPA), gde se jedna HTML stranica dinamički ažurira kako bi prikazivala različite sadržaje, umesto da se oslanja na tradicionalno učitavanje novih stranica sa servera. Ovo omogućava brže učitavanje aplikacije i bolje korisničko iskustvo.
5. **Raznovrsni alati i biblioteke**: Angular ima bogat set alata i biblioteka koji olakšavaju razvoj aplikacija. To uključuje Angular CLI za brzo generisanje projekata i automatizaciju zadatka, Angular Material za brzu izradu modernog korisničkog interfejsa, RxJS za obradu asinhronih događaja i mnoge druge.
6. **Snažne funkcionalnosti**: Angular sadrži niz funkcionalnosti kao što su rutiranje, validacija forme, HTTP klijent, među komponentna komunikacija, dependency injection, i mnoge druge. Ove funkcionalnosti olakšavaju razvoj i poboljšavaju funkcionalnost aplikacija.

2. Instalacija i podešavanje

1. Potrebno je instalirati **Visual Studio Code** koji se može preuzeti sa sajta: **https://code.visualstudio.com/download**
2. Neophodno je instalirati **Node.js** se može preuzeti na stranici **https://nodejs.org/**en, a na taj način se omogućava izvršavanje JavaScript koda na serverskoj strani.
3. Potrebno je preuzeti i instalirati poslednju verziju **Node.js.**



Slika 1. Preuzimanje Node.js-a

4. Potrebno je kroz terminal **(Command Prompt, PowerShell, Terminal)** instalirati **Angular CLI** globalno koristeći *nmp* (Node Package Manager) pomoću sledeće komande:

npm install -g @angular/cli

Angular CLI (Command Line Interface) je alat komandne linije za kreiranje angular aplikacija koja olakšava razvoj, upravljanje i održavanje Angular aplikacija

Kreiranje nove Angular applikacije

Za kreiranje novog Angular projekta potrebno je odrediti direktorijum/folder u kome će se smestiti projekat i locirati se na njega komandom ***cd***.

cd c:\<naziv-foldera>

Dalje se koristi komanda za kreiranje projekta i navodi se naziv: **ng new <naziv_projekta>**

CLI zatim nudi nekoliko opcija kao što su rutiranje, stilovi (CSS, SCSS, …) koje korisnik bira na osnovu potrebe projekta. Nakon što je kreiranje projekta završeno, iz foldera projekta se dalje izvršavaju potrebne komande.

<naziv_foldera>: cd <naziv_projekta>

Pokretanje aplikacije vrši se u Angular CLI pomoću komande: **ng serve --open**

Komanda će pokrenuti razvojni server, a aplikacija će biti dostupna na **http://localhost:4200/.**

Neke od komandi koje se često koriste pri kreiranju Angular aplikacija su:

ng generate component [ime-komponente]: Generiše novu Angular komponentu sa zadatim imenom. Ova komanda automatski kreira sve potrebne fajlove za komponentu (HTML, CSS, TypeScript) i registruje je u odgovarajućem modulu.

ng generate service [ime-servisa]: Generiše novi Angular servis sa zadatim imenom. Ova komanda kreira TypeScript klasu koja može sadržavati poslovnu logiku ili funkcionalnost koja se može deliti između komponenti.

ng build: Kompajlira Angular aplikaciju za produkciju. Ova komanda generiše optimizovanu verziju aplikacije koja se može distribuirati na serveru.

ng test: Pokreće testove jedinica (unit tests) definisane u Angular projektu. Ova komanda koristi alat kao što su Karma i Jasmine za pokretanje testova i proveru funkcionalnosti aplikacije.

ng lint: Pokreće alat za statičku analizu koda kako bi se pronašle i ispravile potencijalne greške i stilski problemi u kodu. Ova komanda koristi alat kao što je TSLint ili ESLint.

ng update [ime-paketa]: Ažurira zavisnosti projekta na najnovije verzije Angular paketa i njihovih zavisnosti. Ova komanda automatski ažurira **package.json** fajl i izvršava neophodne izmene u projektu.

ng help: Prikazuje listu dostupnih Angular CLI komandi sa kratkim opisom njihove funkcionalnosti.

ng deploy: Distribuira Angular aplikaciju na hosting platformu. Ova komanda se često koristi za automatsko deploy-ovanje aplikacije na platforme kao što su Firebase, GitHub Pages, Netlify, i druge.

ng add [ime-paketa]: Dodaje funkcionalnosti u Angular projekat instaliranjem odgovarajućeg paketa i podešavanjem projekta. Na primer, komanda **ng add @angular/material** dodaje Angular Material biblioteku u projekat.

Struktura Angular aplikacije

Struktura Angular aplikacije može da varira u zavisnosti od potreba projekta, ali postoji nekoliko osnovnih elemenata koji se često koriste.

Svaka Angular aplikacija mora imati folder **node_modules** koji sadrži sve pakete i biblioteke koje projekat koristi. Komandom **npm install** u glavnom direktorijumu projekta, npm će preuzeti sve biblioteke i alate koji postoje definisani u package.json fajlu. Na ovaj način je omogućeno lakše instaliranje i održavanje projekta.

tsconfig.json je konfiguracioni fajl koji se koristi za konfigurisanje TypeScript kompajlera.

package.json je JSON fajl koji se koristi za definisanje metapodataka o projektu, kao i za upravljanje zavisnostima projekta. Sadrži ključne informacije o nazivu, verziji projekta, komande za pokretanje skripti, pakete koje projekat koristi u produkciji (dependencies), pakete koje projekat koristi samo tokom razvoja kao što su alati za testiranje, linting i slično itd.

angular.json je konfiguracioni fajl koji se koristi za konfigurisanje Angular projekta. Sadrži informacije o strukturi projekta, postavkama okruženja i drugim podešavanjima.

README.md je tekstualni fajl koji služi kao dokumentacija projekta i sadrži informacije o projektu kao što su opis, svrha i osnovne karakteristike; uputstva za instalaciju i pokretanje projekta; konfiguraciju okruženja i sve što je potrebno da bi se projekat pokrenuo; informacije koje su korisne za korisnike i ljude koji rade na projektu; važne komande itd.

U okviru **src** foldera se nalazi većina koda aplikacije:

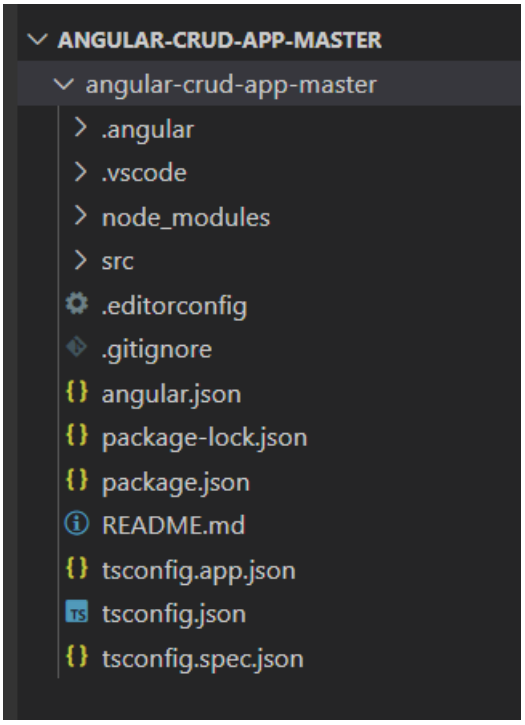
- **app** folder sa kodom aplikacije koji sadrži brojne podfoldere za komponente, servise, direktive itd;
- **assets** folder koji sadrži statičke resurse koji se koriste u aplikaciji;

i druge foldere koji mogu varirati u zavisnosti od vrste projekta.

Pored navedenih, često se u strukturi Angular projekta koriste i alati kojima se napisan kod testira i kojima se proverava ispravnost funkcionalnosti aplikacije. Najčešće se koriste **e2e (End-toEnd)** testovi kojima se simulira stvarno korisničko ponašanje, tako što se aplikacija testita onako kako

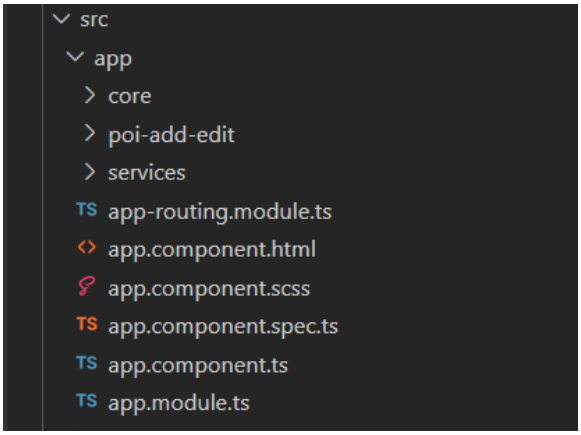
bi to radili stvarni korisnici. Ovim testovima se proverava celokupno ponašanje aplikacije, uključujući i interakciju između različitih komponenti. Druga vrsta testova koja se često sreće su **Unit testovi** koji testiraju ispravnost pojedinačnih delova koda kao što su komponente, direktive, servisi itd. Oni se nalaze u fajlovima koji imaju ekstenziju *spec.ts.* i po tome ih je lako prepoznati u strukturi projekta.

Na Slici 2 je prikazana struktura jednog Angular projekta sa svim prethodno opisanim elementima.



Slika 2. Struktura Angular projekta

U src/app folderu nalazi se kod aplikacije i tu se nalazi glavni modul **app.module.ts**. U njemu se nalaze deklaracije, importi i provajderi. U okviru deklaracija (*declarations*) se navode sve komponente koje pripadaju ovom modulu. To uključuje komponente koje su kreirane u okviru projekta ili koje su uvezene iz drugih modula. Importi (*imports*) služe za navođenje drugih modula koji se koriste u ovom modulu. Oni mogu uključivati Angular module kao što su: HttpClientModule, RouterModule, FormsModule, kao i druge custom kreirane modele u projektu. U okviru provajdera (*providers*) se navode servisi koji su dostupni na nivou modula. Oni se mogu injectovati u komponente i druge servise unutar ovog modula. *Bootstrap* deo sadrži informaciju o glavnoj komponenti aplikacije koja se pokrećre kada se aplikacija učita.



Slika 3. Struktura src/app foldera

3. Osnovni koncepti

Angular je modularni framework za izgradnju aplikacija koji se oslanja na niz osnovnih koncepata kako bi olakšao razvoj modernih aplikacija. Neki od osnovnih koncepata koji se koriste su:

Komponente (Components)

Komponente su osnovni gradnivni blokovi za kreiranje aplikacije u Angular-u, svaka komponenta predstavlja određeni deo korisničkog interfejsa ili funkcionalnost. Angular se oslanja na arhitekturu komponenata kao onsnovnu strukturu organizacije projekta. To znači da se aplikacija razbija na manje delove, manje komponente, koje imaju jasno definisane odgovornosti. Kroz korišćenje komponenti kod postaje održiv i skalabilan. Održivost se postiže jasnom organizacijom koda, dok se skalabilnost postiže dodavanje novih komponenti kako aplikacija raste.

Angular komponentu možemo prepoznati po sufiksu “component” (Primer: new-name.component.ts).

Moduli (Modules)

Moduli su organizacione jedinice u Angular aplikaciji koje grupišu srodne komponente, servise I druge funkcionalnosti. Svaka Angular aplikacija mora imati barem jedan glavni modul, nazvan AppModule, koji definiše glavne komponente i druge konfiguracije za aplikaciju.

Servisi (Sercvices)

Servisi se koriste za deljenje podataka i funkcionalnosti između različitih delova aplikacije poput komponenti, drugih servisa i sl. i koriste se za logiku koja nije direktno povezana sa prikazom, kao što su komunikacija sa serverom, deljenje podataka između komponenti i druge globalne funkcionalnosti.

Dependency Injection (DI)

Ovo je tehnika koja se koristi za „injectovanje“ zavisnosti u komponente i druge klase. Na primer, umesto da komponenta sama kreira instancu servisa koju koristi, ona samo deklariše da želi koristiti taj servis, a Angular će automatski pronaći odgovarajuću instancu servisa i ubrizgati je u komponentu prilikom kreiranja. Prednosti korišćenja Dependency Injection-a uključuju:

1. **Poboljšana modularnost**: Zavisnosti se ubrizgavaju izvana, što olakšava reorganizaciju koda i izmenu zavisnosti bez potrebe za izmenom samih komponenti.
2. **Lakše testiranje**: Zavisnosti se mogu lako zameniti sa lažnim ili mock implementacijama prilikom testiranja, što olakšava pisanje testova i izolaciju jedinica za testiranje.
3. **Smanjena zavisnost**: Komponente ili klase ne moraju da znaju detalje o tome kako se zavisnost kreira ili konfiguriše. Umesto toga, one samo deklarišu svoje potrebe i Angular se brine o ubrizgavanju odgovarajućih zavisnosti.

U Angularu, Dependency Injection se automatski primenjuje kroz Angularov injector. Ovo omogućava efikasno upravljanje zavisnostima i olakšava razvoj aplikacija.

Rutiranje (Routing)

Sistem rutiranja je osnovni deo svake Angular aplikacije. Omogućava navigaciju između različitih pogleda i komponenti, što je ključno za stvaranje glatkog korisničkog iskustva.

Routing omogućava organizaciju i upravljanje navigacijom između različitih delova aplikacije na osnovu URL-ova. Pojam “Routing” odnosi se na definisanje ruta, povezivanje ruta sa odgovarajućim komponentama, i upravljanje navigacijom. Rutiranje omogućava korisnicima da prelaze između različitih "prikaza" (view-ova) u aplikaciji bez osvežavanja stranice.

Rute **-** definišu vezu između URL-a i komponenti koje treba prikazati (mapiranje).