Matematički fakultet

Univerzitet u Beogradu

MASTER RAD

Elektronske lekcije o tabelama, događajima, formama i DOM elementima u radnom okviru AngularJS

Student: Aleksandar Promicać 1136/2016

Mentor: prof. dr Miroslav Marić

Beograd, 2020.

Sadržaj

1 Uvod

Veb je pojam koji se danas koristi kao sinonim za globalnu svetsku mrežu (eng. World Wide Web) i predstavlja skup međusobno povezanih hipertekstualnih dokumenata kojima se pristupa putem interneta. Nastao je u naučnoj laboratoriji CERN u Švajcarskoj. Tim Berns Li je 1989. godine napisao prvi klijentski i serverski kod, a prva veb stranica je objavljena 1991. godine.

Prvi sadržaji na vebu bile su statičke tekstualne strane. Razvojem i unapređivanjem tehnologija menjale su se potrebe i zahtevi korisnika. Pojavom skript jezika JavaScript veb dobija na dinamičnosti. JavaScript (prvobitno izdat kao LiveScript) razvijen je u kompaniji NetScape. On je pružio mogućnost dinamičkog modifikovanja HTML dokumenta programskim putem. Danas na vebu najzastupljeniji su dinamički sadržaji, poznatiji kao veb aplikacije. Ovim aplikacijama se pristupa putem mreže korišćenjem pretraživača. Veb je u procesu stalnog širenja, a veb tehnologije u stalnom razvoju.

Napretkom interneta veb programiranje postaje jedna od najtraženijih profesija danas. Dobar programer, pored jezika JavaScript, mora da poznaje i niz drugih tehnologija koje mu svakodnevni rad pojednostavljuju, a podižu i njegov kvalitet. Izrada većih aplikacija u skript jeziku JavaScript je prilično teška. Koristeći samo JavaScript mnogo je teže dodavati nove funkcionalnosti i ispravljati programske greške. Poslednjih godina primetan je trend pojave radnih okvira (eng. framework) koji značajno olakšavaju kreiranje veb aplikacija, uz primetno manje linija koda. Radni okvir je koncept koji olakšava struktuiranje koda aplikacije, gde je veći akcenat na arhitekturi nego na sintaksi jezika i sadrži razne alate koji pomažu pri rešavanju problema.

AngularJS je radni okvir za skript jezik JavaScript koji se koristi za kreiranje dinamičkih veb aplikacija. Razvili su ga Miško Heveri (Miško Heavery) i Adam Abrons (Adam Abrons) 2009. godine [1], zaposleni u kompaniji Google. AngularJS je u početku bio projekat koji su razvijali u slobodno vreme. Jedan od bitnih trenutaka je bio kada su aplikaciju Google podrška (eng. Google Feedback) uspeli da sa 17000 linija koda svedu na samo 1500 koristeći AngularJS. Prva verzija objavljena je 2012. godine. Ideja se pokazala vrlo dobro, pa je projekat sada i zvanično podržan od strane kompanije Google. Radni okvir AngularJS danas koristi značajan broj programera za izradu veb aplikacija i spada u najpopularnije radne okvire za skript jezik JavaScript.

U računarstvu jedna od osnovnih podela programskih paradigmi je na imperativnu i deklarativnu. Programi napisani imperativnom programskom paradigmom se mogu posmatrati kao niz

naredbi koje računar mora izvršiti u datom redosledu. Kod deklarativne paradigme programer navodi kako bi željeni rezultat trebao izgledati ali ne i kako ga dobiti korak po korak. AngularJS koristi deklarativnu paradigmu za kreiranje korisničkog interfejsa i povezivanje softverskih komponenata, dok je imperativni kôd odličan za izražavanje poslovne logike aplikacije.

AngularJS prilagođava i proširuje tradicionalni HTML vokabular kako bi predstavio dinamički sadržaj kroz dvosmerno vezivanje podataka.

1.1 Arhitekturalni šablon Model-Pogled-Kontroler

Prilikom kreiranja kvalitetnog softvera treba zadovoljiti neke od kriterijuma. Dizajn aplikacije treba biti takav da privuče korisnike. Logika aplikacije treba da omogući praktičnu implementaciju složenih zahteva i da pruži mogućnost jednostavnog unapređivanja i ispravljanja grešaka. Softver takođe treba biti dizajniran tako da odgovara na veliki broj zahteva, treba biti lak za održavanje, brz i efikasan. U nestruktuiranim aplikacijama navedeni zadaci nisu jasno i dobro podeljeni, pa promena na jednom delu aplikacije može znatno uticati na ostale delove. Da bi se sprečili navedeni problemi, za dizajn se koriste arhitekturalni šabloni koji predstavljaju skup ispravnih rešenja za česte probleme koji se pojavljuju tokom razvoja aplikacija. Arhitekturalni šabloni ne predstavljaju gotov deo programskog koda, već opisuju na koji način rešiti problem.

U osnovi radnog okvira AngularJS nalazi se arhitekturalni šablon Model-Pogled-Kontroler (eng. Model-View-Controller, skraćeno MVC). Ovaj arhitekturalni šablon predstavio je norveški informatičar Trigvi Riensku (Trygve Reenskaug) tokom posete američkoj kompaniji Xerox sedamdesetih godina. Kompanija Xerox je prva kompanija koja je primenila MVC u biblioteci klasa jezika Smalltalk-80. U to vreme koncept veb aplikacija nije postojao, pa je prvobitno zamišljen kao koncept za desktop aplikacije. MVC koji se danas koristi u veb programiranju kao adaptacija originalnog arhitekturalnog šablona i danas predstavlja jedan od najčešće korišćenih šablona.

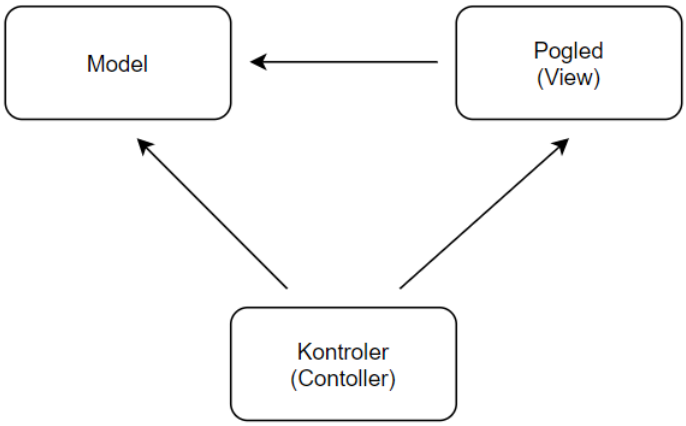
Ovaj arhitekturalni šablon opisuje način struktuiranja aplikacije i u njegovoj osnovi nalaze se dve centralne ideje.

- Korišćenje već postojećeg koda, odnosno mogućnost da se jednom napisan kôd bez izmena ili uz minimalne izmene može ponovo koristiti.

- Jasna raspodela zaduženja među različitim delovima sistema, što podrazumeva podelu sistema na više različitih, međusobno nezavisnih celina od kojih svaka ima svoj zadatak i

koje se pojedinačno mogu modifikovati.

Kao što je već rečeno, MVC se koristi za odvajanje pojedinih delova i komponenti aplikacije u zavisnosti od njihove namene. Svaki deo aplikacije ima svoju odgovornost i pri tome se nalazi u direktnoj ili indirektnoj interakciji sa ostala dva dela sistema. Na ovaj način rešen je problem organizacije koda, olakšan je nezavisan razvoj delova, testiranje i održavanje aplikacije. Na opisan način olakšan je posao programerima jer menjanje jednog sloja aplikacije ne mora nužno promeniti i ostala dva sloja. Jedan od prikaza MVC arhitekturalnog šablona se nalazi na slici 1.



Slika 1: MVC arhitekturalni šablon

1.1.1 Komponente

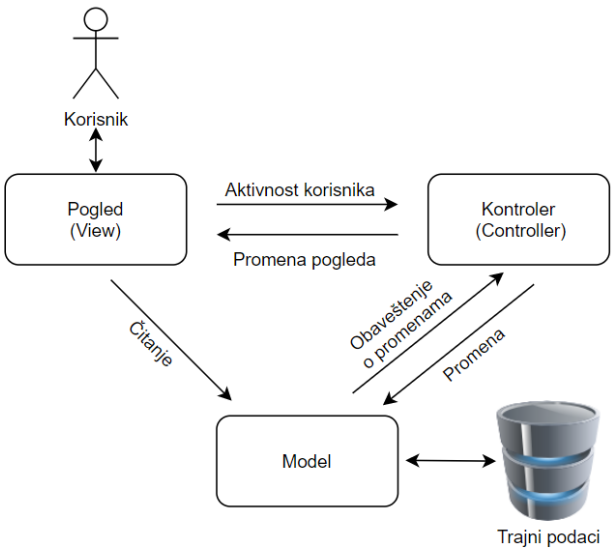
Suština MVC arhitekturalnog šablona je razdvajanje prezentacionog sloja aplikacje od njene logike. Ovaj arhitekturalni šablon deli aplikaciju na tri razdvojena, modularna dela.

- Model (eng. Model) sadrži glavne programske podatke kao što su informacije o objektima iz baze podataka. Sastoji se od skupa klasa koje modeliraju i podržavaju rešavanje problema kojim se aplikacija bavi pa je taj deo obično stabilan i trajan koliko i sam problem. Sva poslovna logika aplikacije se nalazi u modelu. Svi podaci se dobijaju od modela ali se on ne može direktno pozvati, već je kontroler taj koji od modela zahteva određene podatke. Model zatim obrađuje zahteve i vraća kontroleru zahtevane podatke, on ih dalje prosleđuje pogled komponenti koja ih interpretira krajnjem korisniku. Model predstavlja jednu ili više klasa sa svojim stanjima koja se mogu prikazati na zahtev pogleda, a kontroler ih može menjati.

- Kontroler (eng. Controller) sadrži glavne mehanizme za kontrolu programa i odgovoran je za njegov tok, tj. definiše ponašanje same aplikacije. U veb aplikacijama on predstavlja prvi sloj koji se poziva kada pretraživač pozove URL adresu. Takođe, kontroler upravlja korisničkim zahtevima. Obično, kontroler poziva određeni model za zadatak, a zatim bira odgovarajući pogled. Dakle, kontroler predstavlja posrednika između model i pogled komponente u MVC arhitekturalnom šablonu. Kontroler interpretira ulazne podatke korisnika i prosleđuje ih do modela ili pogleda. Za razliku od pogleda, kontroler sadrži deo aplikacijske logike i ima sposobnost da utiče na stanje modela, tačnije, određuje kako model treba da se promeni kao rezultat korisničkih zahteva i koji pogled treba da se koristi.

- Pogled (eng. View) je poslednji sloj MVC arhitekture koji sadrži okruženje aplikacije, odnosno obezbeđuje različite načine za prezentovanje podataka koje dobija od modela preko kontrolera. Korisnik može videti sadržaj koji pogled prikazuje, dok su model i kontroler skriveni od korisnika. Najvažnija osobina pogleda je njegova jednostavnost, tačnije pogled predstavlja vizuelni prikaz modela koji sadrži metode za prikazivanje i omogućava korisniku da menja podatke. Sam pogled nikad ne obrađuje podatke. Njegova zaduženja se završavaju kada su podaci prikazani. Pogled ima informaciju o postojanju modela i preko metoda modela dobija podatke koje prikazuje.

Na slici 2 je predstavljen detaljniji prikaz arhitekture MVC aplikacije.



Slika 2: MVC šema

Bitno je obratiti pažnju da pogled i kontroler zavise od modela, dok je model nezavisan od drugih komponenata. Ova nezavisnost model komponente je veoma bitna jer omogućava da komponenta bude razvijena i testirana nezavisno od prezentacijske logike (nezavisno od pogleda).

MVC arhitekturalni šablon ima i neke svoje nedostatke. Jedan od nedostataka je to što je previše kompleksan za razvoj manjih aplikacija. Tako, jednostavne aplikacije nije potrebno dodatno struktuirati na manje delove jer je tako teže razumeti kôd. Još jedan od nedostataka je i taj što usled previše čestih promena modela pogled može biti preopterećen zahtevima za izmenu, pa previše česti zahtevi mogu dovesti do kašnjenja.

1.2 Veb aplikacije u jednoj strani

Izgled i način korišćenja veba drastično su se promenili od njegovog nastanka do danas. U početnoj fazi veb je korišćen za prezentovanje informacija i nije postojala nikakva interakcija sa korisnikom. Od tada veb neprestano evoluira i korisničko iskustvo se u velikoj meri promenilo. Sada korisnik aktivno interaguje sa sadržajem koji mu je prezentovan.

Prilikom interakcije korisnika sa nekom veb stranicom često je potrebno obezbediti neku povratnu informaciju ili sadržaj koji se nalazi na serveru. Kod tradicionalnog pristupa razvoju veb aplikacija, nakon svake akcije korisnika šalje se novi zahtev serveru gde će se na osnovu te akcije dinamički izgenerisati nova stranica. Ta stranica će na kraju biti poslata nazad i kada pregledač primi tu stranicu od servera on će je učitati i tako prikazati korisniku sadržaj. Ovakav pristup kao posledicu ima celokupno ponovno učitavanje stranice, što nije najbolje korisničko iskustvo jer je korisnik često u situaciji da čeka promenu stranice i prikaz nove.

Jedan od načina da se dodatno poboljša korisničko iskustvo na vebu je da se vreme čekanja nakon akcije korisnika svede na minimum. Upravo to je problem koji veb aplikacije u okviru jedne stranice (eng. Single Page Applications, skraćeno SPA) pokušavaju da reše.

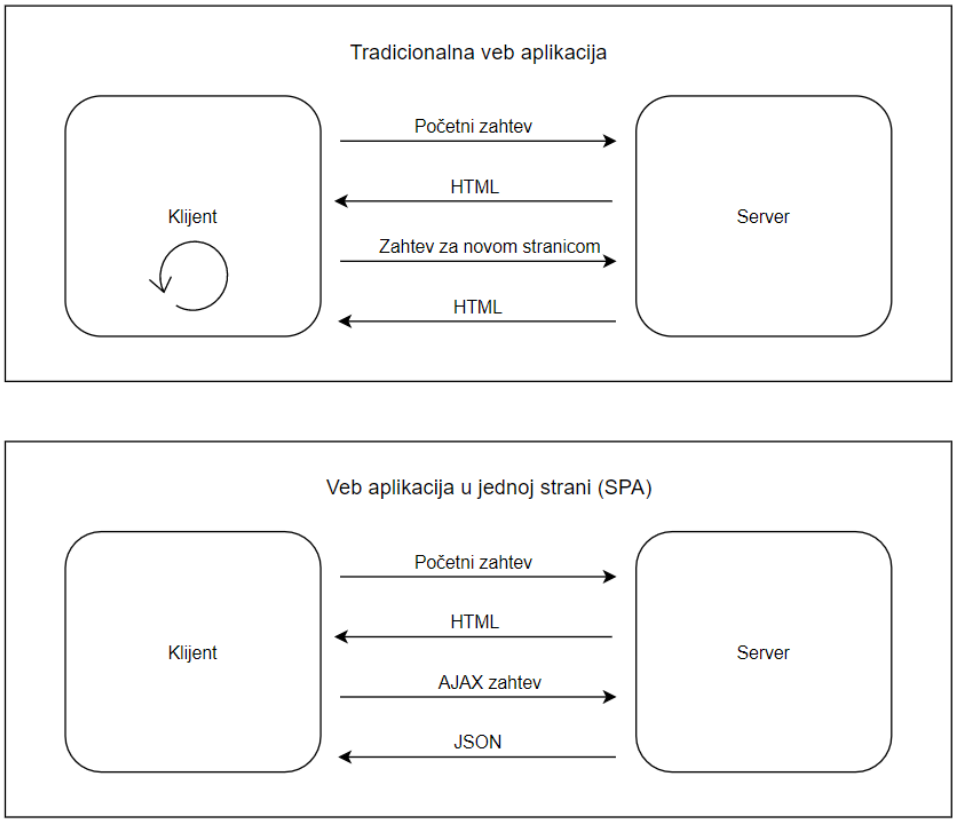
SPA je veb aplikacija koja se jedanput učitava sa servera, a odgovarajući resursi se učitavaju dinamički i po potrebi dodaju stranici, obično kao posledica neke akcije od strane korisnika [2]. Kada klijent pravi dalje zahteve, server mu šalje samo tražene parcijalne podatke, a ne cele stranice kao u tradicionalnom načinu. Ovim pristupom se izbegavaju slabosti koje nastaju učitavanjem uzastopnih stranica, čineći da se veb aplikacija ponaša više kao klasična desktop aplikacija.

Kod SPA nakon što se aplikacija prvi put učita, sva se komunikacija odvija kroz AJAX

(engl. Asynchronous JavaScript and XML) pozive, na koje server odgovara uglavnom u JSON (engl. JavaScript Object Notation) ili XML (engl. Extensible Markup Language) formatu. Nakon toga aplikacija koristi dobijene podatke za dinamičko ažuriranje stranice bez njenog ponovnog učitavanja.

Nedostaci SPA se ogledaju u dužem inicijalnom učitavanju početne stranice aplikacije kada je reč o većoj aplikaciji.

SPA ne treba posmatrati kao ultimativno rešenje koje treba koristiti za sve veb aplikacije, već više kao još jedan od koraka u evoluciji veba i pokušaj da se korisniku pruži što bolje iskustvo pri korišćenju same aplikacije. Na slici 3 predstavljena je razlika između tradicionalne i SPA veb aplikacije.



Slika 3: Tradicionalne i veb aplikacije u jednoj strani

Radni okviri kao što su AngularJS, Ember.js, ExtJS, KnockoutJS, React i VueJS su usvojili SPA principe.

1.3 Osnove radnog okvira AngularJS

AngularJS je MVC radni okvir koji omogućava kreiranje dobro struktuirane, lake za testiranje i održavanje veb aplikacije.

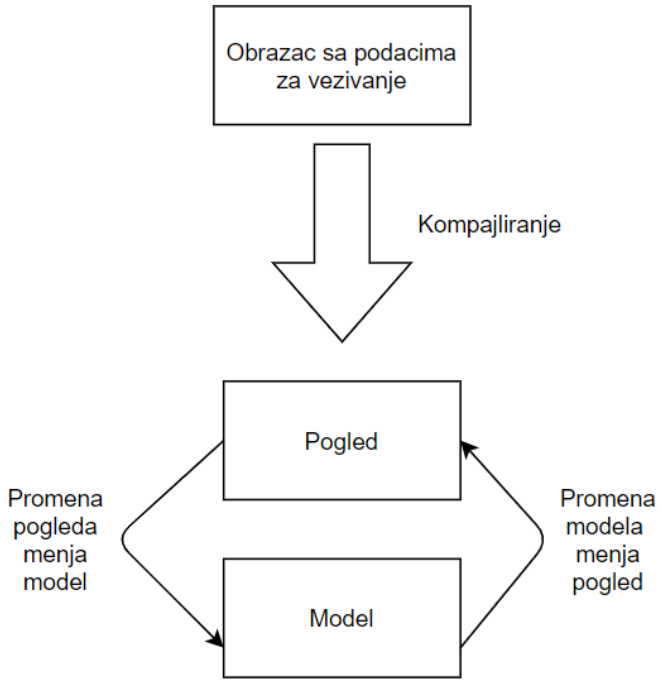
Pomoću radnog okvira AngularJS proširuje se HTML dokument novim atributima koji se nazivaju direktive (eng. Directive). Direktive omogućavaju manipulaciju nad objektnim modelom dokumenta (eng. Document Object Model, skraćeno DOM). DOM predstavlja strukturnu prezentaciju HTML ili XML dokumenata čiji su elementi organizovani u stablo u kome svaki čvor predstavlja jedan element. Radni okvir AngularJS dodatno olakšava manipulaciju DOM stabla dodavanjem direktiva u HTML oznake. AngularJS direktive se lako prepoznaju po prefiksu ng. Direktiva ng-app je početna tačka AngularJS aplikacije koja inicijalizuje radni okvir. Direktiva ng-init inicijalizuje aplikacione podatke, a ng-model direktiva dodeljuje vrednost HTML kontrolama za aplikacione podatke.

AngularJS kontroleri (eng. Controllers) se koriste za upravljane tokom podataka aplikacija. Kontroleri služe da organizuju funkcionalnosti na nivou grupe DOM elemenata. To su JavaScript objekti koji imaju svoja svojstva i funkcije. Direktiva ng-controller definiše kontroler aplikacije.

Koncept modula (eng. Module) zadužen je za grupisanje koda u odgovarajuće logičke celine. AngularJS aplikacija predstavlja skup međusobno povezanih modula.

AngularJS implementira dvosmerno vezivanje podataka (eng. two-way data binding), vezujući HTML (pogled) sa javascript objektima (model) na veoma jednostavan način. Dvosmerno vezivanje podataka je automatska sinhronizacija podataka između model i pogled komponenti. Pogled je projekcija modela u svakom trenutku. Svaka promena modela se odražava na pogled i obrnuto. Ovo se postiže koristeći izraze (eng. expressions) koji vezuju vrednosti promenljivih iz modela sa pogled komponetom. Na ovakav način se zaobilazi potreba da se aktivno manipuliše DOM elementima. AngularJS detektuje promene u modelu poredeći trenutne vrednosti sa vrednostima koje su sačuvane kroz raniji proces provere, za razliku od Ember.js i Backbone.js radnih okvira koji se aktiviraju kada nastanu promene na modelu.

Prikaz dvosmernog vezivanja podataka predstavljen je na slici 4.



Slika 4: Dvosmerno vezivanje podataka

1.3.1 Direktive

Direktive predstavljaju markere na DOM elementima koji ukazuju radnom okviru AngularJS da veže odgovarajuće ponašanje za taj DOM element ili ga u potpunosti promeni. Pored ugrađenih direktiva koje počinju prefiksom ng, postoji mogućnost dodavanja sopstvenih direktiva.

U primeru 1 prikazano je dodavanje osnovne ng-app direktive.

Primer 1. Dodavanje direktive u HTML oznaku

```
<body ng-app="mojaAplikacija">
```

Dodavanjem ng-app direktive u <body> element HTML dokumenta označava se oblast važenja AngularJS aplikacije.

Neke od najkorišćenijih ugrađenih direktiva opisane su u tabeli 1.

Tabela 1: Spisak AngularJS direktiva

AngularJS direktiva	Opis
ng-app	Osnovna, inicijalna direktiva pomoću koje se pokreće AngularJS aplikacija.
ng-controller	Direktiva koja određuje JavaScript klasu za neku datu akciju. Kontroleri se obično nalaze u nekom eksternom .js fajlu.
ng-model	Direktiva koja omogućava vezu između šablona i podataka.
ng-repeat	Direktiva koja vrši prolaz kroz petlju.
ng-init	Direktiva za inicijalizovanje tj. postavljanje stanja.
ng-class, ng-class-even, ng-class-odd, ng-show	Direktive za kontrolu CSS svojstava.
ng-click, ng-mouseup, ng-change, ng-focus, ng-blur, ng-keyup	Direktive koje odgovaraju JS događajima.
ng-required, ng-change, ng-maxlength, ng-form, ng-submit	Direktive koje se koriste u radu sa formama.

1.3.2 Moduli

Moduli u radnom okviru AngularJS su uvedeni da bi se realizovala modularnost koda. Funkcionišu nalik imenskim prostorima i omogućavaju grupisanje različitih delova AngularJS aplikacije, na primer kontrolera, servisa, direktiva, filtera itd. Moduli su logički entiteti na koje se jedna aplikacija može podeliti. Aplikacija se može sastojati od nekoliko modula. Kreiranje modula u radnom okviru AngularJS je vrlo jednostavno.

Primer 2. Kreiranje modula

```
var app = angular.module('mojModul', []);
```

U primeru 2 predstavljeno je kreiranje osnovnog modula. To je glavni deo AngularJS aplikacije. Može se smatrati kao ekvivalent Main metoda u nekim drugim programskim jezicima, odnosno ključna tačka aplikacije. Pri definisanju modula prosleđuju se dva parametra, naziv i prazan JavaScript niz. Taj niz predstavlja niz drugih modula, od kojih kreiran modul zavisi.

1.3.3 Kontroleri

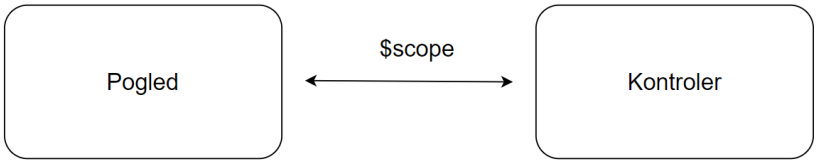
Kontroleri, kao što je već pomenuto, su zaduženi za upravljanje određenim delom DOM stabla. Osnovni zadatak kontrolera je da veže određene promenjive za pogled kojim upravlja. Takođe, vrši pozive prema servisima (eng. services, ugrađene JavaScript funkcije) koji pristupaju modelu, rade sa podacima, vraćaju ih nazad kontroleru koji potom ažurira pogled komponentu.

Primer 3. Dodavanje kontrolera u AngularJS aplikaciju

```
app.controller("mojKontroler", function ($scope) {...}
```

1.3.4 Opseg

Radni okvir AngularJS koristi termin opseg (eng. scope) malo drugačije nego što se taj termin koristi u računarstvu. U programiranju, pod ovim terminom se podrazumeva oblast važenja nekog povezivanja, na primer naziv i vrednost promenjive.
Scope u radnom okviru AngularJS predstavlja ugrađeni JavaScript objekat čija je osnovna uloga povezivanje kontrolera sa pogledom. U kontrolerima, podacima modela se pristupa preko scope objekta. Kontroler kao parametar prima objekat scope kojim se dalje manipuliše.



Slika 5: AngularJS scope objekat

1.3.5 Izrazi

Izrazi u radnom okviru AngularJS imaju ulogu da povežu podatke aplikacije sa HTML kodom. AngularJS određuje i vraća vrednost izraza tačno gde se taj izraz koristi. Izrazi se navode u dvostrukim vitičastim zagradama i vrlo su slični izrazima u skript jeziku JavaScript.

Primer 4. Sintaksa izraza u radnom okviru AngularJS

```
{{"Zdravo" + " " + "Svete"}}
```

1.3.6 Filteri

U radnom okviru AngularJS filteri se koriste za formatiranje podataka prilikom prikazivanja. Filteri imaju sintaksu koja je data u primeru 5.

Primer 5. Sintaksa filtera u radnom okviru AngularJS

```
{{izraz | filter}}
```

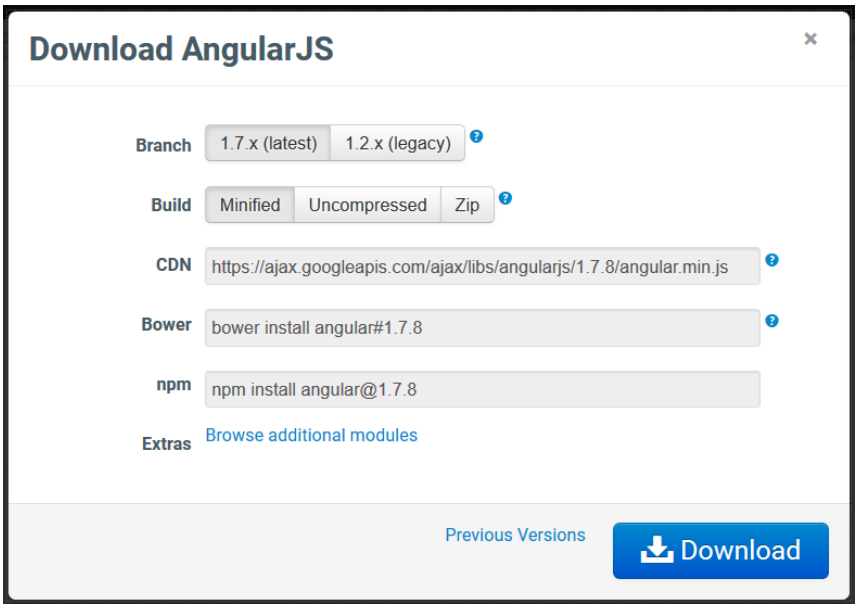
U tabeli 2 dat je spisak filtera koji su dostupni u radnom okviru AngularJS.

Tabela 2: Spisak AngularJS filtera

AngularJS filter	Opis
filter	Bira podskup elemenata nekog niza i vraća nov niz.
lowercase	Konvertuje tekst izraza u mala slova.
uppercase	Konvertuje tekst izraza u velika slova.
orderby	Omogućava da se uredi niz koristeći navedeni izraz.
currency	Konvertuje broj u izrazu u novčanu jedinicu.
date	Koristi se za prikazivanje datuma u odgovarajućem formatu.
json	Koristi se za konvertovanje JavaScript objekata u JSON string.

1.4 Osnovni AngularJS primeri

AngularJS se distribuira preko JavaScript fajla koji se može pronaći na zvaničnoj stranici radnog okvira[6]. Potrebno je uključiti preuzeti fajl u HTML dokument koristeći <script> oznaku.



Slika 6: Preuzimanje AngularJS fajla sa zvanične stranice

Druga mogućnost, za koju nije potrebno preuzimanje JavaScript fajla, je jednostavno dodavanje linka koji je dat na slici 7, u HTML dokument.

```
<script src=
"https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js
   ">
</script>
```

Slika 7: Dodavanje linka ka AngularJS radnom okviru

U primeru 6 predstavljena je osnovna AngularJS aplikacija.

Primer 6. AngularJS aplikacija koja prikazuje pozdravnu poruku

```
<!DOCTYPE html>
<script src=
"https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/
angular.min.js">
</script>
<body>
<div ng-app="">
 <p>Ime : <input type="text" ng-model="ime"></p>
 <h1>Zdravo {{ime}}</h1>
</div>
</body>
</html>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 6 prikazan je na slici 8.

Zdravo Petar

Slika 8: Polje za unos i pozdravna poruka

Aplikacija data u primeru 6 za uneseno ime prikazuje pozdravnu poruku. Direktiva ng-app označava početak AngularJS aplikacije, a ng-model direktiva kreira promenjivu modela pod nazivom „ime” koja se može koristiti u HTML oznaci koja sadrži ng-app direktivu. Zatim se dvostrukim vitičastim zagradama povezuje vrednost promenjive „ime” i prikazuje u pozdravnoj poruci. Ovde se može koristiti i HTML oznaka koja je ekvivalentna vitičastim zagradama. Zatvaranjem oznake zatvara se i AngularJS aplikacija.

Primer 7. AngularJS aplikacija koja sabira dva broja

```
<!DOCTYPE html>
<html>
<script src= "http://ajax.googleapis.com/ajax/libs/
angularjs/1.6.9/angular.min.js">
</script>
<body>
<div ng-app="">
<p>Pet plus pet je: {{ 5 + 5 }}</p>
</div>
</body>
</html>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 7 prikazan je na slici 9.

Pet plus pet je: 10

Slika 9: Rezultat sabiranja

Kao i u primeru 6, ng-app direktiva označava početak AnuglarJS aplikacije. AngularJS zatim prikazuje poruku i izračunava vrednost 5+5 koja se nalazi u okviru vitičastih zagrada, što ukazuje radnom okviru AngularJS da se radi o izrazu. Kroz ovaj primer se može videti primena izraza u radnom okviru AngularJS.

Primer 8. AngularJS aplikacija koja broji zaposlene

```
<html>
    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/
        angular.min.js"></script>
    <script>
        var app = angular.module("mojModul", []);
        app.controller("mojKontroler", function ($scope) {
            var radnici = ['Pera Perić', 'Jovan Jovanović', 'Mika Mikić'];
            $scope.radnici = radnici;});
    </script>
<body ng-app="mojModul" ng-controller="mojKontroler">
    <h2>Broj radnika: {{ radnici.length}}</h2>
</body>
</html>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 8 prikazan je na slici 10.

Broj radnika: 3

Slika 10: Prikaz broja zaposlenih

U primeru 8 na jednostavan način je predstavljena primena MVC arhitekturalnog šablona, koja je u radnom okviru AngularJS vrlo intuitivna i laka.

Deo koda prikazan na slici 11 predstavlja model komponentu. Promenjiva „radnici" predstavlja JavaScript niz koji sadrži imena radnika. U realnoj situaciji podaci modela bi se, na primer, preuzimali iz SQL baze podataka.

```
var radnici = ['Pera Perić', 'Jovan Jovanović', 'Mika
    Mikić'];
```

Slika 11: Deo koda koji predstavlja model

Na slici 12 prikazan je deo koda koji predstavlja pogled komponentu. Sadrži deo HTML koda i izraz predstavljen u vitičastim zagradama za koji se vezuje dužina niza „radnici" iz model komponente. Radni okvir AngularJS izvršava izraz i prikazuje vrednost.

```
<h2>Broj radnika: {{ radnici.length}}</h2>
```

Slika 12: Deo koda koji predstavlja pogled

Na slici 13 prikazan je deo koda koji predstavlja kontroler komponentu u MVC aplikaciji. Najbitniji zadatak kontrolera je da kreira svojstvo „scope" JavaScript objekta pod nazivom „radnici" i prosledi mu vrednost promenjive „radnici ". Na ovaj način se povezuje model i pogled komponenta.

```
var app = angular.module("mojModul", []);
app.controller("mojKontroler", function ($scope) {

    var radnici = ['Pera Perić', 'Jovan Jovanović', 'Mika
        Mikić'];
    $scope.radnici = radnici;
});
```

Slika 13: Deo koda koji predstavlja kontroler

1.5 Prednosti i nedostaci radnog okvira AngularJS

AngularJS spada u najkorišćenije tehnologije koje se primenjuju za razvoj veb aplikacija pomoću script jezika JavaScript. Trenutno je održavan od strane kompanije Google što znači da iza njega stoji iskusan tim programera. Kao što je pomenuto, AngularJS koristi MVC arhitekturalni šablon koji u velikoj meri olakšava podelu aplikacije na logičke celine. Za razliku od drugih radnih okvira i biblioteka, nije potrebno koristiti posebne dodatke. Radni okvir AngularJS je odličan za aplikacije koje se izvršavaju u okviru jedne stranice. Potpuno je prilagodljiv i radi dobro sa drugim alatima i JavaScript bibiliotekama. Testiranje je dosta olakšano zbog korišćenja JavaScript koda. AngularJS koristi dvosmerno povezivanje podataka koje pruža trenutnu povezanost model i pogled komponenti bez dodatnih napora od strane programera. Početi sa Angularom je neverovatno lako. Uz pomoć par atributa dodatih postojećem HTML dokumentu, može se kreitati AngularJS aplikacija kroz samo nekoliko koraka. AnagularJS koriste sajtovi kao što su Netflix, Vemo, Weather kao i mnogi drugi.

Kao i svaka tehnologija i radni okvir AngularJS ima određene mane i nedostatke. AngularJS se u potpunosti oslanja na JavaScript. Drugim rečima ako korisnik onemogući JavaScript, AngularJS takođe neće raditi. Još jedan od nedostataka radnog okvira AngularJS tiče se bezbednosti, tačnije nije bezbedno autentifikovati korisnika koristeći samo AngularJS.

2 Elektronske lekcije o tabelama, događajima, formama i DOM elementima u radnom okviru AngularJS

Postoji mnogo materijala i interaktivnih sadržaja koji mogu pomoći u savladavanju radnog okvira AngularJS. Većina dostupnih sadržaja je na engleskom jeziku, što predstavlja dodatnu prepreku. Elektronske lekcije o tabelama, događajima, formama i DOM elementima u radnom okviru AngularJS kreirane su kako bi korisnik na jednom mestu, na našem jeziku, imao i neophodnu teoriju i mogućnost da stečeno znanje testira kroz primere.

Internet verzija elektronskih lekcija o tabelama, događajima, formama i DOM elementima u radnom okviru AngularJS kreiranih za potrebe ovog master rada nalazi se na sledećoj adresi:

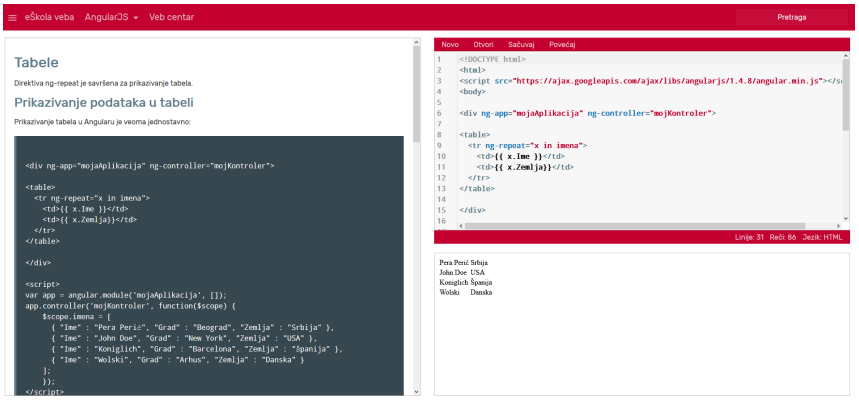
http://edusoft.matf.bg.ac.rs/eskola_veba/#/course-details/ang

Na slici 14 prikazana je početna strana elektronskih lekcija o radnom okviru AngularJS koje su

deo projekta *e-Škola veba* koji se razvija na Matematičkom fakultetu u Beogradu[3].

Slika 14: Elektronske lekcije o radnom okviru AngularJS

Svaka internet stranica u okviru elektronskih lekcija sastavljena je iz tri dela, kao što je prikazano na slici 15. Na levoj strani nalaze se elektronske lekcije, dok se na desnoj strani nalaze dva polja. Sa desne strane korisnik može videti i proveriti naučeno kroz odgovarajuće primere.



Slika 15: Izgled lekcije *Tabele u radnom okviru AngularJS*

2.1 Ukratko o *e-Školi veba*

Platforma *e-Škola veba* predstavlja interaktivne sadržaje za učenje trenutno najpopularnijih tehnologija koje se koriste u veb programiranju. Na jednom mestu korisnik može upotpuniti i unaprediti svoje znanje. Značaj elektronske platforme za učenje veb programiranja je veliki, imajući u vidu nagli razvoj interneta i popularnost alatki i servisa koje nudi, kao i sve veća popularnost jezika JavaScript i radnih okvira koji rad čine efikasnijim.

3 Tabele u radnom okviru AngularJS

Stanje aplikacije se često čuva u bazi podataka, pa je potrebno korisniku prikazati velike količine podataka. Prikazivanje podataka putem tabela je izuzetno praktično zbog svoje preglednosti. U tome mogu pomoći ugrađene direktive radnog okvira AngularJS.

3.1 Prikazivanje podataka u tabeli

Podaci koje je potrebno prikazati u tabeli su često slični, pa je potrebno proći kroz sve podatke, tako da je direktiva ng-repeat savršena za prikazivanje tabela. Ova direktiva omogućava da se dosta jednostavnije i sa manje linija koda prikažu podaci u tabeli za razliku od korišćenja samo HTML i skript jezika JavaScript. Da bi se prikazala tabela koriste se poznate HTML oznake.

Primer 9. AngularJS aplikacija kojom se prikazuju podaci u tabeli

```
<div ng-app="mojaAplikacija" ng-controller="mojKontroler">
<table>
  <tr ng-repeat="x in imena">
    <td>{{ x.Ime }}</td>
    <td>{{ x.Zemlja}}</td>
  </tr>
</table>
</div>
<script>
var app = angular.module('mojaAplikacija', []);
app.controller('mojKontroler', function($scope) {
    $scope.imena = [
      { "Ime" : "Ivan Nikolić", "Grad" : "Beograd",
       "Zemlja" : "Srbija" },
      { "Ime" : "John Simpson", "Grad" : "New York",
       "Zemlja" : "USA" },
      { "Ime" : "Emanuel Lopez", "Grad" : "Barcelona",
      "Zemlja" : "Španija" },
      { "Ime" : "Luis Garnier", "Grad" : "Pariz",
       "Zemlja" : "Francuska" }
    ];
    });
</script>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 9 prikazan je na slici 16.

Slika 16: Tabela sa imenima

Na slici 17 prikazan je modul kojim se inicijalizuje AngularJS aplikacija.

```
var app = angular.module('mojaAplikacija', []);
```

Slika 17: Modul kreiran u primeru 9

Dodavanjem kontrolera u kreiran modul omogućava se pristup imeninima i gradovima datih kao svojstvo „imena" preko objekta „scope". U samom HTML dokumentu se koristi ng-repeat direktiva kojom se može lako pristupiti svim elementima kolekcije, nalik na „foreach" petlju u jezicima kao što su Java i C#.

Primer 10. AngularJS aplikacija koja prikazuje ocene učenika u tabeli

```
<body>
    <h2>AngularJS Ocene</h2>
    <div ng-app = "mojModul" ng-controller = "mojKontroler">
       <table border = "0">
          <tr>
             <td>Ime: </td>
             <td>{{ucenik.punoIme()}}</td>
          </tr>
          <tr>
             <td>Predmeti:</td>
             <td>
                <table>
                   <tr>
                      <th>Naziv predmeta</th>
                      <th>Ocene</th>
                   </tr>
                   <tr ng-repeat = "predmet in ucenik.predmeti">
                      <td>{{ predmet.naziv }}</td>
```

```
                    <td>{{ predmet.ocena }}</td>
                </tr>
            </table>
        </td>
    </tr>
</table> </div>
<script>
    var app = angular.module("mojModul", []);
    app.controller('mojKontroler', function($scope) {
        $scope.ucenik = {
            ime: "Jovan",
            prezime: "Nikolić",
            predmeti:[
                {naziv:'Fizika',ocena:5},
                {naziv:'Matematika',ocena:5},
                {naziv:'Srpski',ocena:5},
                {naziv:'Engleski',ocena:5}
             ],
            punoIme: function() {
                var ucenikObjekat;
                ucenikObjekat = $scope.ucenik;
                return ucenikObjekat.ime + " " + ucenikObjekat.prezime;
            }
        };
    });</script>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 10 prikazan je na slici 18.

AngularJS Ocene

Ime:	Jovan Nikolić	
Predmeti:	**Naziv predmeta**	**Ocene**
	Fizika	5
	Matematika	5
	Srpski	5
	Engleski	5

Slika 18: Tabela sa predmetima i ocenama

Kao i u primeru 9, dodaje se osnovni modul u aplikaciju. U kontroleru preko objekta „scope" vezuju se podaci o učeniku koji se sastoje od imena, prezimena i podataka o predmetima. Podaci o predmetima sadrže naziv i ocenu iz datog predmeta. Takođe je kreirana pomoćna funkcija „punoIme" koja vraća puno ime i prezime radi lakšeg ispisivanja potrebnih podataka. Da bi se prikazali svi predmeti i odgovarajuće ocene koristi se ng-repeat direktiva, kao i pomoćna funkcija „punoIme" za ispisivanje podataka učenika.

3.2 Uređivanje podataka u tabeli korišćenjem filtera

Kao što je navedeno u uvodu, u radnom okviru AngularJS mogu se koristi filteri kako bi se uredili dati podaci. Filteri se dodaju u direktive, kao što je ng-repeat, uz karakter | , nakon čega sledi sam filter.

Primer 11. AngularJS aplikacija koja koristi filter za sortiranje podataka

```
<body ng-app="mojModul">
    <div ng-controller="mojKontroler">
        <h2>Primer sortiranja AngularJS tabele</h2>
        <table>
            <tr>
                <td>Ime</td>
                <td>Godina</td>
                <td>Lokacija</td>
            </tr>
            <tr ng-repeat="korisnik in korisnici | orderBy : 'godina'">
                <td>{{korisnik.ime}}</td>
                <td>{{korisnik.godina}}</td>
                <td>{{korisnik.lokacija}}</td>
            </tr>
        </table>
    </div>
</body>
<script> var app = angular.module("mojModul", []);
        app.controller("mojKontroler", function ($scope) {
            $scope.korisnici = [{
                ime: "Mika Mikić",
                godina: 10,
                lokacija: 'Beograd'
```

```
        }, {
            ime: "Jovan Jovanović",
            godina: 30,
            lokacija: 'Kraljevo'
        }, {
            ime: "Marko Nikolić",
            godina: 29,
            lokacija: 'Kragujevac'
        }, {
            ime: "Ivan Ilić",
            godina: 24,
            lokacija: 'Subotica'
        }];
    });</script>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 11 prikazan je na slici 19.

Primer sortiranja AngularJS tabele

Ime	Godina	Lokacija
Mika Mikić	10	Beograd
Ivan Ilić	24	Subotica
Marko Nikolić	29	Kragujevac
Jovan Jovanović	30	Kraljevo

Slika 19: Primer sortirane tabele

Podaci u tabeli su sortirani u rastućem poretku, koristeći „orderBy" filter. Kao i u primerima 9 i 10, koristi se ng-repeat direktiva.

4 DOM elementi u radnom okviru AngularJS

Kada internet pregledač učita HTML stranicu, kreira se DOM stablo koje predstavlja strukturu sa svim elementima HTML dokumenta. AngularJS poseduje ugrađene direktive koje omogućavaju manipulaciju DOM elemetima, što predstavlja jednu od bitnijih funkcionalnosti.

Opis direktiva koje AngularJS koristi za povezivanje podataka aplikacije u atribute HTML

DOM elemenata dat je u tabeli 3.

Tabela 3: Spisak AngularJS direktiva za povezivanje DOM elemenata

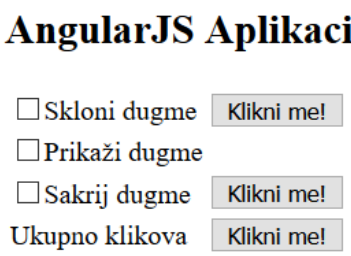
AngularJS direktiva	Opis
ng-disabled	Onemogućava datu kontrolu.
ng-show	Prikazuje datu kontrolu.
ng-hide	Sakriva datu kontrolu.
ng-click	Predstavlja AngularJS događaj klika mišem.

Primer 12. AngularJS aplikacija koja koristi direktive za rad sa DOM elementima

```
<html>
   <head>
      <title>AngularJS HTML DOM</title>
   </head>
   <body>
      <div ng-app = "">
         <table border = "0">
            <tr>
               <td><input type = "checkbox" ng-model = "prikaziSkloniDugme">
                  Skloni dugme</td>
               <td><button ng-disabled = "prikaziSkloniDugme">Klikni me!</
                  button></td>
            </tr>
            <tr>
               <td><input type = "checkbox" ng-model = "skloniPrikazi1">
                  Prikazi dugme</td>
               <td><button ng-show = "skloniPrikazi1">Klikni me!</button></
                  td>
            </tr>
            <tr>
               <td><input type = "checkbox" ng-model = "skloniPrikazi2">
                  Sakrij dugme</td>
               <td><button ng-hide = "skloniPrikazi2">Klikni me!</button></
                  td>
            </tr>
            <tr>
               <td><p>Ukupno klikova {{ brojKlikova }}</p></td>
               <td><button ng-click = "brojKlikova = brojKlikova + 1">Klikni
                   me!</button></td>
```

```
            </tr>
          </table>
      </div>
      <script src = "https://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/
          angular.min.js"></script>
   </body>
</html>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 12 prikazan je na slici 20.



Slika 20: DOM elementi

U primeru 12 prikazana je upotreba sve četiri direktive za rad sa DOM elementima. Ove direktive su izuzetno korisne ako je potrebno sakriti ili prikazati neki HTML element na neku određenu akciju korisnika. Aplikacija u primeru 12 sakriva i prikazuje dugme. Takođe, koristi se direktiva ng-click da bi se izbrojao broj klikova na dugme.

5 Događaji u radnom okviru AngularJS

Radni okvir AngularJS poseduje direktive koje mogu da definišu specifična ponašanja koja se dešavaju prilikom različitih događaja u okviru DOM stabla. Neki od tih događaja su klik mišem, pomeranje kursora, pritiskanje odgovarajućeg tastera itd.

U radnom okviru AngularJS postoji mogućnost dodavanja sopstvenih osluškivača događaja u HTLM elemente koristeći jednu ili više sledećih direktiva:

- ng-blur,
- ng-change,
- ng-click,

- ng-copy,
- ng-cut,
- ng-dblclick,
- ng-focus,
- ng-keydown,
- ng-keypress,
- ng-keyup,
- ng-mousedown,
- ng-mouseenter,
- ng-mouseleave,
- ng-mousemove,
- ng-mouseover,
- ng-mouseup,
- ng-paste.

Direktive događaja dozvoljavaju pokretanje AngularJS funkcija kod određenih korisničkih akcija. AngularJS događaj neće ukloniti postojeći HTML događaj, već će oba biti izvršena.

5.1 Događaji miša

Svaki put kada korisnik pomeri kursor ili pritisne taster miša sistem to beleži u obliku događaja (eng. Mouse Events).
Događaji kretanja miša se dešavaju kada se kursor pomeri preko nekog elementa i tu spadaju:

- ng-mouseenter(trenutak ulaska kursora u oblast elementa),
- ng-mouseover(dok je kursor iznad elementa),
- ng-mousemove(na pokret kursora),

- ng-mouseleave(na napuštanje oblasti elementa).

Prilikom klika na odgovarajući element izvršavaju se naredni događaji:

- ng-mousedown(trenutak pritiskanja dugmeta miša),
- ng-mouseup(trenutak puštanja dugmeta miša)
- ng-click(trenutak potpunog klika miša, pritiskanja i otpuštanja).

Događaj miša se može dodati u bilo koji HTML element kao direktiva.

Primer 13. AngularJS aplikacija koja broji prelazak kursora preko teksta

```
<div ng-app="mojaAplikacija" ng-controller="mojKontroler">
<h1 ng-mousemove="zbir=zbir+1">Kursor je preko!</h1>
<h2>{{ zbir }}</h2>
</div>
<script>
var app = angular.module('mojaAplikacija', []);
app.controller('mojKontroler', function($scope) {
$scope.zbir = 0;
});
</script>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 13 prikazan je na slici 21.

Kursor je preko!

2

Slika 21: Korišćenje događaja miša

U primeru 13 koristi se direktiva ng-mousemove koja detektuje broj prelazaka kursora preko teksta. Svojstvo objekta „scope" se u kontroleru postavja na nulu i ono se uvećava prilikom svakog prelaska. Razultat izraza koji predstavlja broj prelazaka preko teksta prikazuje se u okviru HTML elementa.

5.2 Direktiva ng-click

Direktiva ng-click definiše AngularJS kôd koji će biti izvršen prilikom klika na određen element. Kada direktiva detektuje klik na HTML element, poziva se kôd u kontroleru.

Primer 14. AngularJS aplikacija koja koristi ng-clik direktivu

```
<body ng-app="mojModul">
 <div ng-controller="mojKontroler">
  <table>
   <thead><tr><th>Ime tehnologije</th>
        <th>Svidja mi se</th>
        <th>Ne svidja mi se</th>
        <th>Svidja mi se/Ne svidja mi se</th></tr>
   </thead>
   <tbody>
    <tr ng-repeat="tehnologija in tehnologije">
     <td> {{ tehnologija.ime }} </td>
     <td style="text-align:center"> {{ tehnologija.svidjaMiSe }} </td>
     <td style="text-align:center"> {{ tehnologija.neSvidjaMiSe }} </td>
     <td> <input type="button" ng-click="povecajSvidjaMiSe(tehnologija)"
         value="Svidja mi se" />
          <input type="button" ng-click="povecajNeSvidjaMiSe(tehnologija)"
              value="Ne svidja mi se" /></td>
    </tr>
   </tbody>
  </table>
 </div>
</body>
<script>
var app = angular.module("mojModul", [])
           .controller("mojKontroler", function ($scope) {
        var tehnologije = [
            { ime: "C#", svidjaMiSe: 0, neSvidjaMiSe: 0 },
            { ime: "ASP.NET", svidjaMiSe: 0, neSvidjaMiSe: 0 },
            { ime: "SQL", svidjaMiSe: 0, neSvidjaMiSe: 0 },
            { ime: "AngularJS", svidjaMiSe: 0, neSvidjaMiSe: 0 }
        ];
        $scope.tehnologije = tehnologije;
        $scope.povecajSvidjaMiSe = function (tehnologija) {
```

```
        tehnologija.svidjaMiSe++;
    };
    $scope.povecajNeSvidjaMiSe = function (tehnologija) {
        tehnologija.neSvidjaMiSe++;
    }; }); </script>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 14 prikazan je na slici 22.

Ime tehnologije	Sviđa mi se	Ne sviđa mi se	Sviđa mi se/Ne sviđa mi se
C#	4	1	Sviđa mi se Ne sviđa mi se
ASP.NET	4	2	Sviđa mi se Ne sviđa mi se
SQL	6	6	Sviđa mi se Ne sviđa mi se
AngularJS	10	0	Sviđa mi se Ne sviđa mi se

Slika 22: Korišćenje ng-click direktive

Aplikacija predstavljena u primeru 14 broji klikove na odgovarajuće dugme koristeći direktivu ng-click koja se dodaje u HTML oznaku kao na slici 23.

```
<td> <input type="button" ng-click="povecajSvidjaMiSe(
   tehnologija)" value="Svidja mi se" />
     <input type="button" ng-click="povecajNeSvidjaMiSe(
        tehnologija)" value="Ne svidja mi se" /></td>
```

Slika 23: Dodavanje ng-click direktive u HTML oznaku

Upotrebom pomoćnih funkcija povećava se vrednost u okviru odgovarajućeg HTML elementa. Ovo se postiže dodavanjem direktive ng-clik u HTML oznaku i vezivanjem za funkciju u objektu „scope" koja povećava vrednost klikom na dugme. Deo koda u kontroleru koji je bitan za aplikaciju prikazan je na slici 24.

```
$scope.tehnologije = tehnologije;
$scope.povecajSvidjaMiSe = function (tehnologija) {
    tehnologija.svidjaMiSe++;
};
$scope.povecajNeSvidjaMiSe = function (tehnologija) {
    tehnologija.neSvidjaMiSe++;
```

Slika 24: Deo koda u kontroleru

5.3 Dugme tačno/netačno

Kada je potrebno prikazati deo HTML strane klikom na dugme i sakriti ponovnim klikom (eng. toggle true/false), koristi se direktiva ng-click, kao i direktiva ng-show koja je ranije pomenuta.

Primer 15. AngularJS aplikacija koja prikazuje korišćenje dugmeta tačno/netačno

```
<div ng-app="mojaAplikacija" ng-controller="mojKontroler">
<button ng-click="mojaFunkcija()">Kliknuti!</button>
<div ng-show="pokaziMe">
    <h1>Menu:</h1>
    <div>Pizza</div>
    <div>Pasta</div>
    <div>Pesto</div>
</div>
</div>
<script>
var app = angular.module('mojaAplikacija', []);
    app.controller('mojKontroler', function($scope) {
                $scope.pokaziMe = false;
                $scope.mojaFunkcija = function() {
                            $scope.pokaziMe = !$scope.pokaziMe;}
});</script>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 15 prikazan je na slici 25.

Slika 25: Prikazivanje i sakrivanje elemenata koristeći ng-click direktivu

Kao što je navedeno, koristi se direktiva ng-click koja je vezana za dugme. Takođe se koristi i direktiva ng-show koja je povezana sa svojstvom objekta „scope" u kontroleru pod nazivom

„pokaziMe”, koje je „boolean” tipa i ima početnu vrednost „false”. U objectu „scope” koristi se funkcija koja na klik menja svojstvo u suprotnu vrednost, koristeći operator negacije (operator !). Klikom na dugme prikazuju se elementi menija, a ponovnim klikom se sakrivaju.

6 Forme u radnom okviru AngularJS

Forme u radnom okviru AngularJS omogućavaju povezivanje podataka i validaciju kontrole ulaza. Predstavljaju HTML formu, na kojoj korisnik može da unosi podatke. U najkorišćenije forme spadaju:

- elementi ulaza,
- elementi selektovanja,
- elementi tipa dugme,
- elementi sa tekstom.

6.1 Povezivanje podataka

Za povezivanje podataka koristi se direktiva ng-model koja je već ranije pomenuta. Koristeći ovu direktivu omogućeno je povezivanje podataka i njihovo korišćenje u formama.

Primer 16. AngularJS aplikacija koja pokazuje primenu ng-modela za povezivanje podataka sa formom

```
<!DOCTYPE html>
<html lang="en">
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.
    min.js">
</script>
<body>
<div ng-app="mojModul" ng-controller="mojKontroler">
  <form>
    Ime: <input type="text" ng-model="ime">
  </form>
  <h1>Uneli ste: {{ime}}</h1>
</div>
```

```
<script>
var app = angular.module('mojModul', []);
app.controller('mojKontroler', function($scope) {
    $scope.ime = "Aleksandar";
});
</script>
</body>
</html>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 16 prikazan je na slici 26.

Slika 26: Unos i promena podataka korišćenjem ng-model direktive

Direktiva ng-model povezuje ulazni kontroler sa ostatkom AngularJS aplikacije. Svojstvo „Ime" ima određenu početnu vrednost koja se menja unosom teksta u polje.

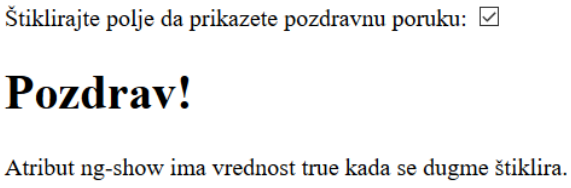
6.2 Polje za čekiranje

Polje za čekiranje (eng. Checkbox) može imati dve vrednosti, „true " ili „false ". Za povezivanje podataka sa formom se koristi direktiva ng-model.

Primer 17. AngularJS aplikacija koja prikazuje pozdravnu poruku ako je polje čekirano

```
<body>
    <div ng-app="">
        <form>
            Štiklirajte dugme da prikažete pozdravnu poruku:
            <input type="checkbox" ng-model="mojaPromenljiva">
        </form>
        <h1 ng-show="mojaPromenljiva">Pozdrav!</h1>
    </div>
    <p>Atribut  ng-show ima vrednost true kada se dugme stiklira. </p>
</body>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 17 prikazan je na slici 27.



Slika 27: Korišćenje polja za štikliranje

Direktivom ng-model vezuje se vrednost „mojaPromenjiva" i koristeći direktivu ng-show prikazuje se pozdravna poruka prilikom čekiranja polja.

6.3 Radio dugme

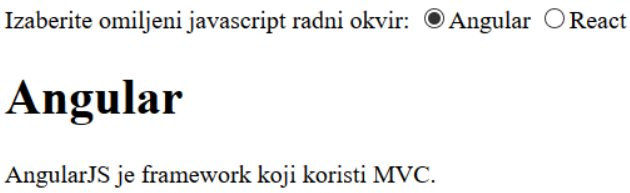
Radio dugme (eng. radio button) predstavlja jedan od standardnih tipova za izbor u HTML dokumentima. Moguće je koristiti istu ng-model direktivu za nekoliko radio dugmadi i oni mogu imati različite vrednosti ali će biti korišćena samo vrednost izabranog dugmeta, kao što je prikazano u primeru 18.

Primer 18. AngularJS aplikacija koja prikazuje rad sa radio dugmetom

```
<body ng-app="">
    <form>
        Izaberite omiljeni javascript radni okvir:
        <input type="radio" ng-model="mojaPromenjiva" value="angular">
            Angular
        <input type="radio" ng-model="mojaPromenjiva" value="react">React
    </form>
    <div ng-switch="mojaPromenljiva">
        <div ng-switch-when="angular">
            <h1>Angular</h1>
            <p> AngularJS je framework koji koristi MVC.</p>
        </div>
        <div ng-switch-when="react">
            <h1>React</h1>
            <p>Razvijen u facebooku.</p>
        </div>
    </div>
</body>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 18 prikazan je na slici

28.



Slika 28: Korišćenje radio dugmeta

Koristeći direktivu ng-model dodeljuju se dve vrednosti, „angular" i „react" koje su vezane za dva radio dugmeta. Koristeći ng-switch direktivu menja se tekst u okviru HTML elemenata u zavisnosti od izbora.

6.4 Polje za izbor

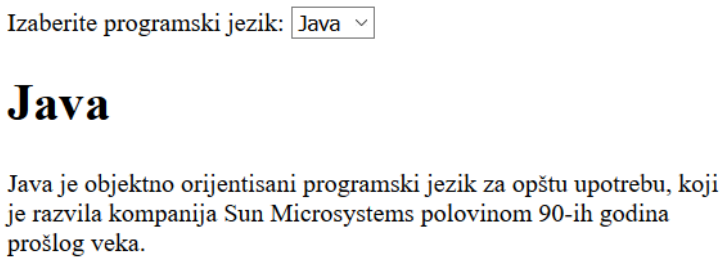
U radnom okviru AngularJS koristeći polje za izbor može se postići vezivanje podataka sa više izbora koristeći direktive ng-repeat ili ng-options. Koristeći ng-model direktivu može se preuzeti trenutno izabrana vrednost. Ovim se postiže i povezanost sa „scope" objektom.

Primer 19. AngularJS aplikacija koja prikazuje rad sa poljem za izbor (eng. selectbox)

```
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.
    min.js">
</script>
<body ng-app="">
<form>
  Izaberite programski jezik:
  <select ng-model="mojaPromenljiva">
    <option value="">
    <option value="c">C
    <option value="java">Java
    <option value="csharp">C#
  </select>
</form>
<div ng-switch="mojaPromenljiva">
  <div ng-switch-when="C">
     <h1>C</h1>
```

```
    <p>C je programski jezik opste namene. Razvio ga je Dennis Ritchie
        sedamdestih godina prošlog veka.</p>
  </div>
  <div ng-switch-when="java">
    <h1>Java</h1>
    <p>Java je objektno orijentisani programski jezik za opštu upotrebu,
        koji je razvila kompanija Sun Microsystems  polovinom 90-ih godina
        prošlog veka.</p>
  </div>
  <div ng-switch-when="csharp">
    <h1>C#</h1>
    <p>Jezik opšte namene za izradu aplikacija .NET Framework platforme. </
        p>
  </div>
</div>
</body>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 19 prikazan je na slici 29.



Slika 29: Korišćenje polja za izbor

Koristeći direktivu ng-model HTML elementu „select” dodeljuje se vrednost promenljive „mojaPromenljiva”. Direktiva ng-switch pokazuje i sakriva delove HTML koda u zavistnosti od izbora.

7 Primer aplikacije

U ovom poglavlju dat je primer AngularJS aplikacije koja povezuje prethodne teme (prikaz podataka u tabeli, rad sa formama i događaje).

Primer 20. AngularJS aplikacija za prikaz, unos, izmenu i brisanje podataka radnika

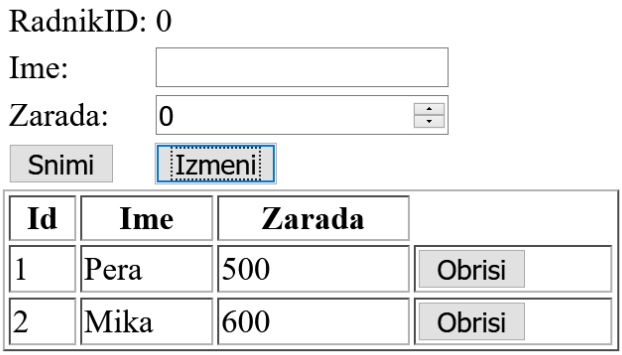
```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>
    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.5/
        angular.min.js"></script>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/
        jquery.min.js"></script>
</head>
<body>

    <div ng-app="mainApp" data-ng-controller="CRUDController">

        <table>
            <tr>
                <td>RadnikID: </td>
                <td>
                    <span>{{ RadnikModel.Id }}</span></td>
            </tr>
            <tr>
                <td>Ime:</td>
                <td>
                    <input type="text" data-ng-model="RadnikModel.Ime" /
                        ></td>
            </tr>
            <tr>
                <td>Zarada:</td>
                <td>
                    <input type="number" data-ng-model="RadnikModel.
                        Zarada" /></td>
            </tr>
            <tr>
```

```
                <td>
                    <input type="button" value="Snimi" data-ng-click="
                        AddData()" /></td>
                <td>
                    <input type="button" value="Izmeni" data-ng-click="
                        UpdateData()" /></td>
            </tr>
        </table>

        <table border="1" style="width: 300px">
            <thead>
                <th>Id</th>
                <th>Ime</th>
                <th>Zarada</th>
            </thead>
            <tr data-ng-repeat="Radnik in RadnikLista" data-ng-click="
                BindSelectedData(Radnik)">
                <td>{{ Radnik.Id }}</td>
                <td>{{ Radnik.Ime }}</td>
                <td>{{ Radnik.Zarada }}</td>
                <td>
                    <input type="button" value="Obrisi" data-ng-click="
                        DeleteData(Radnik)" /></td>
            </tr>
        </table>
    </div>

    <script type="text/javascript">
        var app = angular.module("mainApp", []);
        app.controller('CRUDController', function ($scope) {

            $scope.RadnikModel = {
                Id: 0,
                Zarada: 0,
                Ime: '',
            };

            $scope.RadnikLista = [];
```

```
            $scope.AddData = function () {
                var _Radnik = {
                    Id: $scope.RadnikLista.length + 1,
                    Ime: $scope.RadnikModel.Ime,
                    Zarada: $scope.RadnikModel.Zarada
                };
                $scope.RadnikLista.push(_Radnik);
                ClearModel();
            }

            $scope.DeleteData = function (Radnik) {
                var _index = $scope.RadnikLista.indexOf(Radnik);
                $scope.RadnikLista.splice(_index, 1);
            }

            $scope.BindSelectedData = function (Radnik) {
                $scope.RadnikModel.Id = Radnik.Id;
                $scope.RadnikModel.Ime = Radnik.Ime;
                $scope.RadnikModel.Zarada = Radnik.Zarada;
            }

            $scope.UpdateData = function () {
                $.grep($scope.RadnikLista, function (e) {
                    if (e.Id == $scope.RadnikModel.Id) {
                        e.Ime = $scope.RadnikModel.Ime;
                        e.Zarada = $scope.RadnikModel.Zarada;
                    }
                });
                ClearModel();
            }

            function ClearModel() {
                $scope.RadnikModel.Id = 0;
                $scope.RadnikModel.Ime = '';
                $scope.RadnikModel.Zarada = 0;
            }
        });
    </script>
```

```
    </body>
    </html>
```

Sadržaj internet stranice nakon otvaranja HTML dokumenta iz primera 20 prikazan je na slici 30.



Slika 30: Aplikacija za prikaz, unos, izmenu i brisanje podataka radnika

U prikazanom primeru za rad sa podacima koristi se JavaScript niz. U kontroleru se definišu odgovarajuće funkcije koje se pozivaju klikom na dugme, koristeći ng-clik direktivu. Podaci o radnicima prikazuju se u tabeli, pomoću ng-repeat direktive. Korisnik može uneti podatke za novog radnika, izmeniti ili obrisati postojeće podatke.

8 Zaključak

AngularJS je jedan od prvih deklarativnih JavaScript radnih okvira koji je doneo dosta promena u razvoju veb aplikacija, na način koji je ljudima razumljiviji, istovremeno poboljšavajući kvalitet koda čineći ga lakšim za održavanje. Upotrebom MVC arhiteturalnog šablona postiže se bolja organizacija i čitljivost koda, kao i upotreba koda na više mesta.

Učenici na časovima informatike i računarstva, u osnovnoj i srednjoj školi, upoznaju se sa raznim programskim jezicima. Jezik JavaScript poslednjih godina postaje sve aktuelniji kao i radni okviri koji olakšavaju rad, čineći prve korake u veb programiranju znatno lakšim.

Elektronske lekcije o tabelima, događajima, formama i DOM elementima u radnom okviru AngularJS, koje su deo platforme *e-Škola veba*, imaju veliku prednost u odnosu na štampanu literaturu. Interaktivnost i mogućnost pokretanja koda direktno iz lekcija, treba da pomogne korisnicima da lakše savladaju gradivo, a zadacima da provere naučeno. Primena ovih alata je usmerena na postizanje kvaliteta znanja i razvijanje sposobnosti disciplinovanja i dosledne prakse koja vodi ka zavidnoj efikasnosti u izvršavanju zadataka.

Literatura

[1] S. Seshadri, B. Green, "AngularJS Up and Running", O'Reilly, 2014.

[2] A. Lerner, "ng-book - The Complete Book on AngularJS", Fullstack io, 2013.

[3] N. Jurić, M. Marić, "e-Škola veba", VII Simpozijum Matematika i primene, Matematički fakultet, Beograd, 5. novembar 2016.

[4] D. Flanagan, "JavaScript - The Definitive Guide", O'Reilly, 2006.

[5] B. Green, S. Seshadri, "AngularJS", O'Reilly, 2013.

[6] Sajt radnog okvira AngularJS: https://angularjs.org/,
pristupano avgust 2020. godine

[7] Sajt W3 Schools: https://www.w3schools.com/angular/,
pristupano jul 2020. godine

[8] Sajt Tutorials point: www.tutorialspoint.com/angularjs/,
pristupano jul 2020. godine