

UNIVERZITET U BEOGRADU

MATEMATIČKI FAKULTET

SEMINARSKI RAD

**Primena XML Tehnologija u .NET-u**

Predmet: XML tehnologije

Nastavnik: prof. dr Dušan Tošić

Student: Miloš Stanković, 2014/2013

Beograd, 2013/2014

SADRŽAJ

[MVC obrazac dizajniranja - 2 -](#_Toc389325581)

# MVC obrazac dizajniranja

Osnovna ideja obrazaca dizajniranja (*design patterns*) je da izdvoje nacrt rešenja za situacije (probleme) koje se često javljaju i ponavljaju u različitim sferama tokom razvoja softvera. Obrazac dizajna ne predstavlja gotovo rešenje problema koje se može direktno pretočiti u kod. To je opis rešenja problema koje se može primeniti u različitim situacijama.

Obrasci dizajniranje nisu vezani ni za jedan programski jezik, biblioteku (*framework*) i slično, već predstavljaju rešenja koja su se kroz praksu dokazala kao dobra (najčešće i najbolja). Zbog toga, oni mogu da značajno ubrzaju razvoj softvera. Nema potrebe trošiti vreme na pronalaženje rešenja problema, već se može odmah krenuti sa implementacijom obrasca dizajniranja za konkretni problem. Druga veoma bitna prednost obrazaca dizajniranja je ta što poboljšavaju komunikaciju između programera. Programeri mogu da lakše stvore sliku o nekom softverskom proizvodu ukoliko znaju koji od obrazaca dizajniranja je bio korišćen tokom njegovog razvoja.

MVC (*Model-View-Controller*) predstavlja primer jednog obrasca dizajniranja. On nije povezan konkretno sa AngularJS-om, već je uspešno implementiran u različitim programskim jezicima. MVC predstavlja arhitekturu na kojoj se zasniva najveći broj modernih veb, Android, iOS i desktop aplikacija. Na slici x prikazan je MVC obrazac dizajniranja koji će u nastavku biti detaljno opisan.



MVC se sastoji od tri osnovne logičke komponente: model podataka (*model*), prezentacioni deo ili pogled (view), i kontrolni deo (*controller*). Svaka od navedenih komponenti se sastoji od većeg broja klasa.

Model se sastoji od čistih podataka koji se dobijaju iz baze podataka, veb servisa ili iz nekog drugog izvora, i koje je potrebno prikazati korisniku. Zbog toga su modeli najčešće POJO (*Plain Old Java Object*) objekti [x]. POJO klase su jednostavne klase koje ne nasleđuju neke specijalne Java klase. One se sastoje od skupa (privatnih) polja i odgovarajućih *get* i *set* metoda, a njihova uloga je u prenosu podataka.

Pogled je komponenta koju korisnik direktno vidi. Pogled prikazuje podatke iz modela, pri čemu ne zna ništa o semantici tih podataka. U eri mobilnih uređaja i velikog broja različitih ekrana na kojima se pogled prikazuje, česta je pojava da se za određeni model definiše nekoliko pogleda.

Kontroler predstavlja deo MVC-a koji koordinira između modela i različitih pogleda na aplikaciju. On osluškuje događaje koji se dešavaju na pogledu (npr. klik na dugme) i izvršava odgovarajuće akcije pridružene tim događajima. Neke od tih akcija mogu se odnositi na dohvatanje novih podataka iz baze podataka od kojih se formira novi model, a zatim se taj model prosleđuje nekom drugom pogledu koji korisniku prikazuje zahtevane podatke.

Prva očigledna prednost MVC obrasca jeste razdvajanje prezentacionog dela aplikacije od njenog logičkog dela. Podrška različitim vrstama korisnika koji koriste različite vrste uređaja je standardni zahtev današnjih veb aplikacija. Korisnički interfejs mora biti različiti ako zahtev za aplikacijom stiže od desktop računara ili mobilnog uređaja. U oba slučaja model je isti, tj. predstavlja iste podatke koje je potrebno prikazati. Kontroler ima ključnu ulogu u izboru pogleda koji će prikazati zahtevane podatke. Pored izdvajanja pogleda iz poslovne logike aplikacije, MVC smanjuje kompleksnost velikih aplikacija. Kod je struktuiraniji, pa je samim tim jednostavnije upravljati njime, testirati ga i ponovo koristiti.

# Javascript MVC biblioteke

Moderne veb aplikacije se sastoje od skupa kontrolera, modela i pogleda. Obično postoji jedan osnovni kontroler koji prihvata sve zahteve, a zatim ih prosleđuje odgovarajućim specifičnim kontrolerima. Kontoler najpre poziva servisni kod koji izračunava model, a zatim određuje odgovarajući pogled. Postoje čitave biblioteke koje implemeniraju MVC obrazac i omogućuju kreiranje MVC veb aplikacija. Neke od njih su Spring MVC i ASP.NET MVC. Na primer, kod Spring MVC biblioteke kontroleri su posebne Java klase, modeli su (najčešće) POJO Java klase, dok su pogledi JSP (*Java Server Page*) dokumenti koji se prevode u HTML.

Popularnost Javascript-a je dovela do toga da se sve više poslovne logike veb aplikacija odvija na klijentskoj strani. Takođe, sa razvojem AJAX (*Asynchronous Javascript And XML*) tehnologije nastala je ideja o kreiranju jednostranih veb aplikacija (*single page web applications*). Jednostrane veb aplikacije se sastoje od jedne osnovne strane. Tokom interakcije korisnika sa aplikacijom, umesto da se ponovo učitavaju kompletne stranice, AJAX tehnologijom se delovi početne strane zamenjuju drugim delovima, tako da korisnik ima osećaj da radi sa klasičnom desktop aplikacijom. Zbog svega navedenog, količina Javascript koda kod prosečne veb aplikacije u stalnom je porastu. Zato se javila potreba za boljom organizacijom skript koda, pa su tako nastale čitave biblioteke (frameworks). Najpoznatije biblioteke koje implementiraju MVC obrazac dizajniranja su Backbone.js, Ember.js i AngularJS.

## AngularJS biblioteka

AngularJS je jedna od najpopularnijih Javascript MVC biblioteka. Prvu verziju ove biblioteke objavili su Miško Hevery i Adam Abrons iz kompanije Brat Tech LLC 2009. godine. Ubrzo je Adam Abrons napustio projekat, dok je Miško Hevery nastavio da radi na AngularJS-u u okviru Google-a.