

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

[1. Istorijski uvod u XML - 2 -](#_Toc378537485)

[2. Osnovi XML-a - 3 -](#_Toc378537486)

[2.1. Sintaksa i dobro formatirani XML dokument - 4 -](#_Toc378537487)

[2.2. XML prostori naziva (namespaces) - 6 -](#_Toc378537488)

[2.3. Definicija tipa dokumenta - 7 -](#_Toc378537489)

[2.4. XML Šema - 10 -](#_Toc378537490)

[2.5. Primene XML-a - 12 -](#_Toc378537491)

[3. Primena SAML-a u .NET biblioteci - 13 -](#_Toc378537492)

[3.1 Nastanak i opravdanost SAML-a - 13 -](#_Toc378537493)

[3.2 Privatnost i sigurnost kod SAML-a - 15 -](#_Toc378537494)

[3.3 Arhitektura SAML-a - 15 -](#_Toc378537495)

[3.4 Opis SAML komponenti i veza između njih - 16 -](#_Toc378537496)

[3.5 Profil jedinstvenog prijavljivanja internet pretraživača - 18 -](#_Toc378537497)

[3.6 Implementacija SAML SSO modula u .NET biblioteci - 23 -](#_Toc378537498)

[4. Zaključak - 28 -](#_Toc378537499)

[5. Literatura - 28 -](#_Toc378537500)

# 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 Angular-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. 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 podledu (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.