03.12.2021.

# Objektno-orijentisano programiranje 2

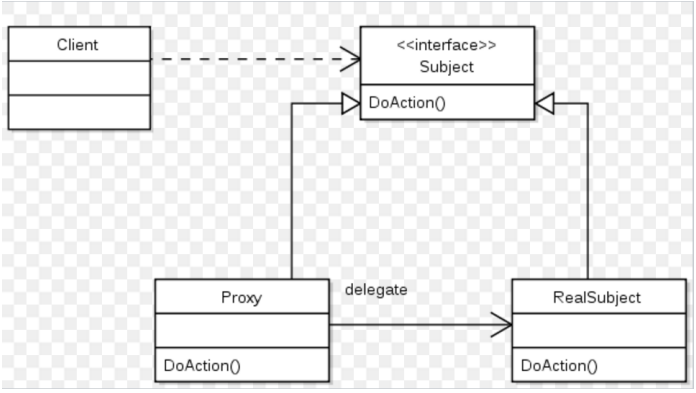
## Zastupnik (engl. proxy)

Motivacija i osnovna ideja:

Po strukturi, projektni uzorak Zastupnik sličan je dekoratoru ili objektnom adapteu. Naime, postoji klasa Zastupnik koja agregira objekat neke klase i metode izvršava tako što (uz možda dodatnu logiku) poziva metode agregiranog objekta. Da bi se objekat za koji želimo zastupnika i zastupnik ponašali na isti način, mogu biti izvedeni iz iste klase. U pogledu structure, ovde se vidi razlika u odnosu na adapter kod koga adaptirani objekat ima drugačiji interfejs od našeg željenog interfejsa. Namena zastupnika je jasno različita i od dekoratora i od adaptera. Adapter prilagođava objekat željenom interfejsu, dekorater dodaje i obogaćuje funkcionalnosti, dok zastupnik ima za cilj da korisniku pruži utisak rada sa zastupljenim objektom iako korisnik zaista komunicira sa zastupnikom. Motivacija za tako nešto može biti raznovrsna.

- Zaštitni zastupnik: želimo da zastupnikom kontrolišemo pristup nekim metodama ili onemogućimo funkcionisanje metoda za određene argumente.
- Zastupnik udaljenog objekta: potrebno je komunicirati sa objektom na drugom računaru za čije je korišćenje potrebno otvoriti soket i razmenjivati poruke. Mi bismo želeli da u kodu komuniciramo sa objektom kao sa “običnim” objektom, a zatupnik bi mogao da obavlja komunikaciju preko mreže.
- Virtuelni zastupnik: kreiramo zastupnika za objekat čije je kreiranje skupa operacija, a koji se možda neće ni koristiti. Kreiranje “skupog” objekta odlažemo dok god ne budemo zahtevali zaista neku operaciju nad njim.
- Odloženo izračunavanje: evaluaciju nekih izraza odlažemo dok god ne izgradimo kompletan izraz (koji je zastupnik rezultata), a izračunavanje nad celim izrazom je računski i memorijski efikasnije od postepenog evaluacija delova izraza.

UML dijagram projektnog uzorka Zastupnik dat je na slici 1.




*Slika 1 Projektni uzorak Zastupnik*

03.12.2021.

Zadaci:

1. Implementirati opšti oblik projektnog uzorka Zastupnik u skladu sa UML dijagramom na slici 1.
2. Napisati apstraktnu klasu InternetService koja ima čistu virtuelnu metodu getPage(std::string url) i konkretnu klasu RealInternetService koja preklapa datu metodu. Implementirati klasu SecuredInternetService koja je zastupnik za RealInternetService i ima listu zabranjenih domena. Klasa treba da onemogući učitavanje stranica sa zabranjenih domena (za dozvoljene stranice prosleđuje poziv objektu klase RealInternetService.
3. Napisati klasu Image sa atributima path, width, height i metodama getWidth(), getHeight(), display() i zastupnika ImageProxy kojim se kreiranje slike odlaže dok god se ne pozove neka od datih metoda.
4. Napisati klasu Vektor i koristeći odloženo izračunavanje omogućiti efikasno izračunavanje (bez međurezultata i samo jednom for petljom) linije poput:
   Vector rezultat;
   rezultat = `v + Sqrt(v2 * v) - Sin(Sqrt(v2))`,
   gde su v i v2 vektori istih veličina, a operacije +, -, /, * kao i date unarne funkcije treba da se izvršavaju pokomponentno.