Projektni uzorci - strukturni (Structural design patterns-DP)

Adapter DP - strukturni objektni i klasni DP

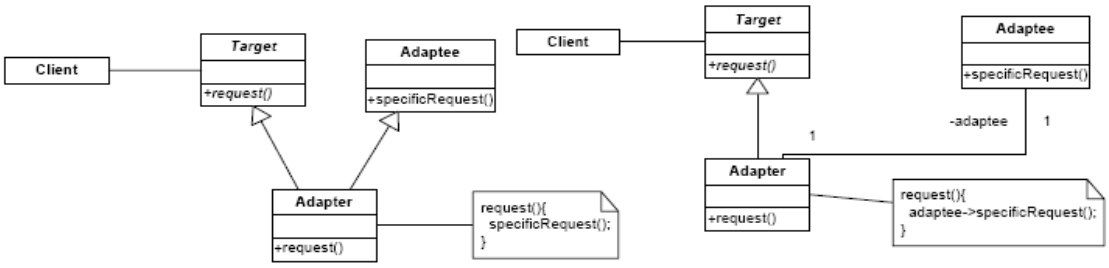
Drugo ime

Omotač(engl. Wrapper–dvoznačno, koristi se i za Dekorater), Pregovarač

Namena

Konvertuje interfejs jedne klase u interfejs kakav očekuje druga klasa (klijent). Adapter DP omogućava da funkcionišu zajedno klase koje inače to ne bi mogle, zbog različitog interfejsa.

- klasni adapter: objektni adapter:



Osnovni učesnici su:

Target klasa

Implementira *Adapter* interfejs i vrši njegovu implementaciju upotrebom instance *Adaptee* klase. Kažemo da se klasa *Adaptee* na ovaj način prilagođava klijentu.

Client klasa

korisnik funkcionalnosti klase koja se adaptira posredno preko Adaptera, sarađuje sa objektima koji poštuju interfejs Target

Adaptee klasa

klasa koja se adaptira, pretpostavka je da se ova klasa ne može menjati jer nemamo pristup i prava da menjamo izvorni kod ili bi direktne izmene nad izvornim kodom bile previše kompleksne.

Adapter klasa

interfejs (može biti i klasa) koji koristi Klijent. Klasa koja vrši prilagođavanje. Nadležnost ove klase su konverzije tipova, parametara i sl. Najčešće kao mehanizam prilagođavanja koristi delegacija poziva metoda, ređe se koristi nasleđivanje.

Saradnja:

Klijenti pozivaju operacije objekta Adapter, dok adapter poziva operacije adaptiranog podobjekta (nasleđivanjem ili agragacijom) da izvrše zahtev.

Klasni adapter koristi višestruko nasleđivanje za međusobno prilagođavanje interfejsa.

Objektno adapter se oslanja na sastavljanje objekta.

Prednost

Prilagođavamo komponentu čija nam je funkcionalnost potrebna ali njen interfejs ne odgovara našim zahtevima.

Primer: prilagođavanje COTS komponenti našem sistemu ili prilagođavanje starih komponenti zahtevima novog sistema.

Srodni uzorci:

1.Decorator poboljšava drugi objekat (dodavanjem funkcionalnosti) ne menjajući mu

interfejs, te je transparentniji od adaptera. Decorator podržava rekurzivne kompozicije, koje nisu moguće sa samim adapterima.
2. Bridge ima sličnu strukturu kao Adapter, ali drukčiju namenu.

PRIMERI: Adapter

Ilustracija.java
izlaz
Pozvan Zahtev()
Pozvan KonkretanZahtev()

RealanSlucaj.java
Jedinjenje: ------ Voda
Formula: H20
Masa : 18.015
Tacka topljenja: 0.0
Tacka kljucanja: 100.0

Jedinjenje: ------ Benzen
Formula: C6H6
Masa : 78.1134
Tacka topljenja: 5.5
Tacka kljucanja: 80.1

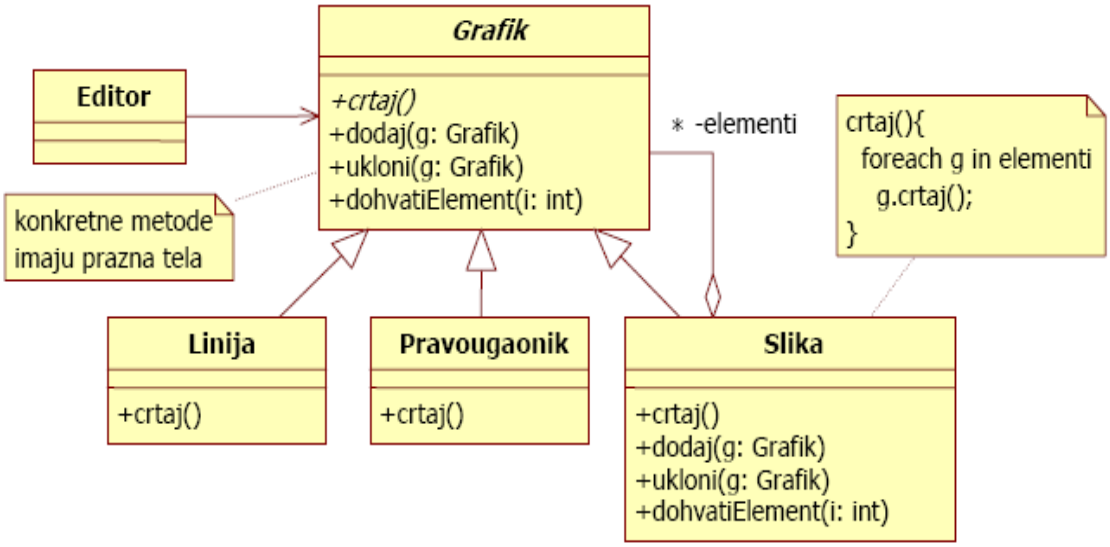
Jedinjenje: ------ Alkohol
Formula: C2H6O2
Masa : 46.0688
Tacka topljenja: -114.1
Tacka kljucanja: 78.3

Kompozicija (engl. *Composite*) - strukturni objektni DP

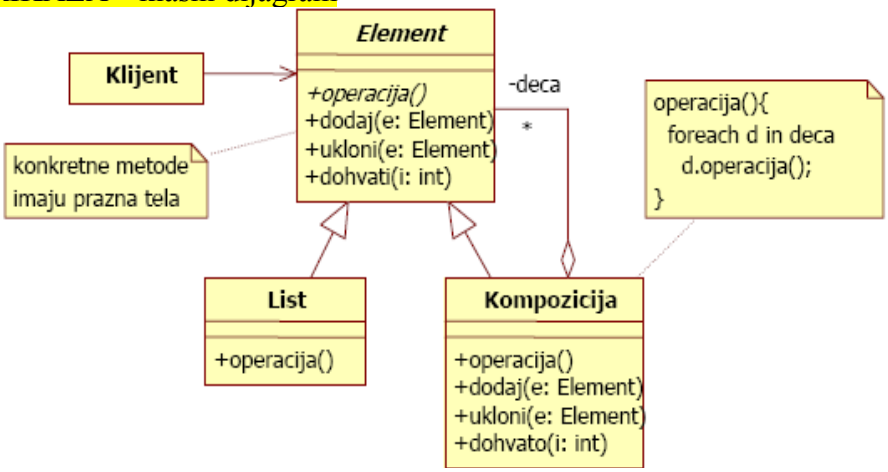
Namena:
–komponuje objekte u strukturu stabla (hijerarhija celina-deo)
–omogućava klijentima da uniformno tretiraju
- individualne objekte
- njihove kompozicije

Primenljivost
Ovaj DP treba koristiti kada se želi da:
–postoje hijerarhije objekata celina-deo takve da su celina i deo iste vrste
–klijenti mogu da ignorišu razlike između kompozicija i pojedinih objekata

Kratak opis problema
–grafičke aplikacije kao što su editori crteža i sistemi za unos šema:
- omogućavaju crtanje kompleksnih slika (dijagrama) sastavljenih od jednostavnih grafičkih elemenata (komponenata, primitiva)
- elementi se mogu grupisati da se formiraju složeni elementi (kompozicije)
- složeni element je potrebno tretirati i kao vrstu grafičkog elementa
- ključno: apstraktna klasa koja predstavlja i proste i složene elemente



DIJAGRAM PRIKAZA – klasni dijagram



Učesnici:

Element(Component) –klasa(Grafik)

- deklariše zajednički interfejs za sve objekte u kompoziciji
- implementira podrazumevano ponašanje zajedničko za sve klase
- deklariše interfejs za pristupanje i upravljanje decom
- implementira prazne metode za pristup i upravljanje decom (zbog listova)
- opciono deklariše i implementira interfejs za pristup roditelju

List(Leaf) -klasa (Linija, Pravougaonik)

- reprezentuje individualne objekte –listove u stablu
- definiše ponašanje za primitivne objekte u kompoziciji, PRIMITIVE

Kompozicija(Composite) –klasa(Slika)

- definiše ponašanje za objekte koji imaju decu
- sadrži komponente decu
- implementira operacije za pristup i upravljanje decom

Klijent(Client) –klasa (Editor)

- manipuliše objektima u kompoziciji kroz interfejs Component klase

Saradnja:

–klijenti koriste interfejs apstraktne klase Element da interaguju sa objektima složene strukture

- ako je primalac zahteva List, zahtev se neposredno izvršava
- ako je primalac zahteva Kompozicija, obično se zahtev prosleđuje komponentama deci

Posledice:
–čini jedostavnim klijente (oni tretiraju na jedinstven način sve objekte u hijerarhiji)
–čini jednostavnim dodavanje nove vrste elemenata
–nije jednostavno ograničiti vrste elemenata koje neke kompozicije sadrže

Bliski DP:
–Iterator se koristi često za obilazak strukture Kompozicije
– Visitor lokalizuje operacije i ponašanje koje bi inače bilo distribuirano između klasa Kompozicija i List
- Dekorater ima sličnu strukturu klasa i često se koristi sa Kompozicijom (kada se koriste zajedno obično imaju zajedničku natklasu)

PRIMERI: Kompozicija
Ilustracija.java
-root
-List A
-List B
-Kompozicija X
-List XA
-List XB
-List C
RealanSlucaj.java
- nacrtaj Slika
- nacrtaj Crvena Linija
- nacrtaj Plavi Krug
- nacrtaj Dvostruki Krug
- nacrtaj Crni Krug
- nacrtaj Beli Krug
- nacrtaj Zeleni Pravougaonik
RealanSlucaj2.java
test.html