Projektovanje softvera

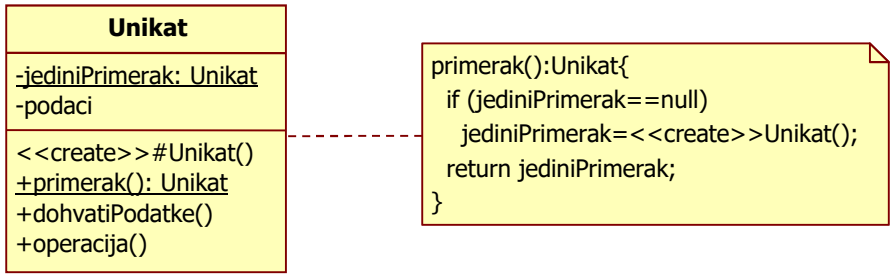
Unikat

Unikat (1)

- Ime i klasifikacija:
  - Unikat (*Singleton*) – objektni uzorak stvaranja
- Namena:
  - obezbeđuje da klasa ima samo jedan objekat i daje globalni pristup tom objektu
- Motivacija:
  - za neke klase je važno obezbediti da imaju samo po jedan objekat
  - na primer, u sistemu treba da postoji samo jedan dispečer štampe (*spooler*)
  - globalna promenljiva obezbeđuje globalan pristup, ali ne brani više objekata
  - rešenje je da klasa sama bude odgovorna za jedinstvenost njenog objekta
- Primenljivost: uzorak treba koristiti kada
  - mora postojati tačno jedan objekat klase
  - on mora biti pristupačan klijentima preko svima poznate tačke pristupa
  - klasa treba da bude proširiva izvođenjem

Unikat (2)

- Struktura:



- Učesnici:
  - `Unikat`
    - definiše operaciju `primerak()` kao operaciju klase (statički metod)
      - operacija daje klijentima pristup do jedinstvenog objekta
      - operacija je odgovorna za kreiranje jedinog objekta
    - zaštićeni konstruktor sprečava da klijenti kreiraju objekte klase
- Saradnje:
  - klijenti dohvataju objekat klase `Unikat` isključivo kroz operaciju `primerak()`

Unikat (3)

- Posledice: dobre strane
  - kontrolisani pristup do jedinog objekta
    - pošto klasa kapsulira jedini objekat, ona može imati striktnu kontrolu nad tim kako i kada klijenti pristupaju objektu
  - rasterećenje prostora imena
    - uzorak *Unikat* je bolji koncept od globalne promenljive jer izbegava opterećivanje prostora imena globalnom promenljivom koja čuva jedini objekat
  - dozvoljeno doterivanje operacija
    - iz klase `Unikat` se može izvoditi
    - lako je objekat izvedene klase zameniti, čak i u vreme izvršenja
  - moguće kontrolisano povećanje broja objekata
    - projektant može da po maloj ceni poveća broj dozvoljenih objekata
  - fleksibilnost u odnosu na uslužnu (*utility*) klasu
    - sa uslužnom klasom je praktično nemoguće promeniti broj objekata
    - statičke funkcije nisu virtuelne, potklase ne mogu da ih polimorfno redefinišu

Unikat (4)

- Implementacija C++:

```
class Unikat {
public:
      static Unikat* primerak();
      virtual void operacija();
 protected:
      Unikat();
 private:
      static Unikat* jediniPrimerak;
}

// implementacija:
Unikat* Unikat::jediniPrimerak=nullptr;

Unikat* Unikat::primerak(){
      if (jediniPrimerak == nullptr) jediniPrimerak = new Unikat;
      return jediniPrimerak;
}

// koriscenje:
Unikat::primerak()->operacija();
```

Unikat (5)

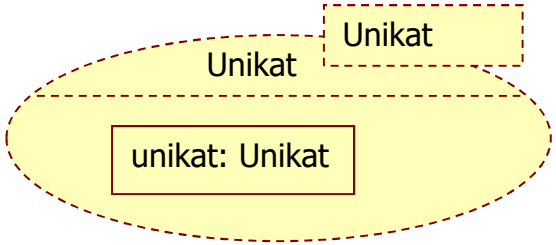
- Implementacija Java:

```
class Unikat{
      public static Unikat primerak(){
               if (jediniPrimerak == null)
                        jediniPrimerak = new Unikat();
               return jediniPrimerak;
      }
      public void operacija(){/*...*/};
      protected Unikat(){}
      private static Unikat jediniPrimerak=null;
};
// koriscenje:
Unikat.primerak().operacija();
```

- Poznate primene:
  - u aplikacijama sa jednim dokumentom (*single-document*) klasa `Dokument` je unikat
  - u *Smalltalk* jeziku svaka metaklasa (klasa čiji su primerci klase) ima samo jedan objekat
- Povezani uzorci:
  - mnogi uzorci koriste *Unikat* (*Apstraktna fabrika*, *Graditelj*, *Fasada*)

Unikat (6)

- UML notacija
  - definicija



  - primena