Objektno-orijentisano programiranje

Generički metodi i tipovi, ugnježđene klase

Nevena Ćirić
www.matf.bg.ac.rs/~nevena_ciric

Generički metodi

- Mehanizmom generičkih metoda i tipova omogućava se opisivanje algoritama koji se mogu primeniti na više tipova.
- Generički (parametarski) metodi su metodi koji za tipove argumenata, povratne vrednosti ili lokalnih promenljivih umesto konkretnih tipova imaju parametar.
- Parametri tipova se ponašaju slično kao formalni parametri pri definisanju metoda, tj. predstavljaju još jedan dodatni parametar koji se treba konkretizovati pri pozivu metoda.
- Metod može imati 1 ili više parametarskih tipova. Lista parametarskih tipova se navodi između zagrada < i > neposredno pre tipa povratne vrednosti u deklaraciji metoda.

PRIMER:
```
public static <T> void ispisiNiz(T[] niz) {
        for (int i = 0; i<niz.length; i++)
                System.out.print(niz[i].toString() + " ");
}
```

Generički metodi

- Preporuka je da imena parametara tipova budu što je moguće kraća radi lakšeg razlikovanja parametarskog tipa od imena konkretnih tipova.
- Za konkretnetizaciju parametarskih tipova mogue se koristiti samo klasni tipovi. Primitivni tipovi ne mogu da se koriste. Umesto njih se mogu koristiti njima odgovarajuće omotač klase.
- Parametar tipova generičkog metoda se konkretizuju pri pozivanju metoda na jedan od sledeća dva načina:
  - eksplicitno navođenjem imena konkretnih tipova u zagradama < i > u okviru poziva metoda
  - implicitno na osnovu tipa objekta koji je prosleđen kao odgovarajući argument sa parametrizovanim tipov
- Generički metod može biti definisan unutar obične ili generičke klase.

Generički metodi

- Ponekada je potrebno da generički metod postavi ograničenje na tipove koji se mogu naći na mestu parametarskog tipa.
- Notacija <T extends TipKojiOgranicava> označava se da parametarski tip T treba da bude podtip tipa koji ograničava, pri čemu TipKojiOgranicava može biti i klasa i interfejs.
- Jedan parametarski tip može imati više od jednog ograničenja. Tada se tipovi koji ograničavaju navode razdvojeni znakom &.
- Ograničenja za parametarske tipove se navode u okviru definicije liste parametarskih tipova pri deklaraciji metoda.

PRIMER:
```
public static <T extends Comparable> void sortirajNiz (T[] niz) {
        . . . niz[i].compareTo(niz[j]) . . .
}
```

PRIMER:
```
public static <T extends Obim & Povrsina> {
        . . . niz[i].obim() . . .
        . . . niz[i].povrsina() . . .
}
```

Generički tipovi

- Generički tip je klasni ili interfejsni tip čija definicija (deklaracije atributa i/ili definicije metoda) zavisi od jednog ili više parametara tipa.
- Generički tipovi definišu skup tipova dobijenih za različite konkretizacije parametarskih tipova.
- Lista parametarskih tipova se navodi u deklaraciji klase/interfejsa između zagrada < i > neposredno iza imena klase/interfejsa.
  PRIMER: public class Kutija<T>{
  …
  }
- Kreiranje konkretne klase/interfejsa na osnovu generičkog tipa vrši se zadavanjem konkretnog tipa za svaki od parametara generičkog tipa. Sva pojavljivanja nekog parametra T u definiciji generičkog tipa biće zamenjena datim konkretnim tipom.

Generički tipovi

- Parametari tipa se upotrebljavaju u deklaracijama atributa i definicijama metoda gde postoji zavisnost od odgovarajućeg parametarskog tipa.
- Oblast važenja parametarskog tipa je cela definicija generičke klase, izuzev statičkih atributa i metoda.
- To povlači da se unutar definicije generičkog tipa ne može deklarisati statički atribut parametarskog tipa.
- Međutim, to ne znači da generička klasa ne može imati generičke statičke metode. Ovde je reč samo o parametrima tipa koji odgovaraju parametrima generičkog tipa. Statički metodi mogu da imaju sopstveni skup parametara tipa nezavisan od parametara generičkog tipa i oni se definišu na isti način kao kod generičkih metodi koji se ne nalaze u okviru definicije generičkog tipa (u okviru zagrada < i > neposredno pre tipa povratne vrednosti u deklaraciji metoda).

Generički tipovi

- Sintaksa koja se koristi za deklarisanje reference tipa klase/interfejsa definisanog na osnovu nekog generičkog tipa je ista kao za obične klase/interfejse, s tim što je ime generičkog tipa praćeno listom konkretizacije njegovih parametara tipa u okviru zagrada < i >.
  PRIMER: Kutija<Integer> kutijaInteger;
- Pozivanje konstruktora generičkog klase se vrši na sličan način, navođenjem liste konretizacija parametara generičkog tipa u okviru zagrada < i > između imena generičkog tipa i zagrada (i) u kojima se navode argumenti konostruktora.
  PRIMER: Kutija<Integer> kutijaInteger = new Kutija<Integer>();
- Dopušteno je da se prilikom poziva konstruktora izostavi navođenje liste parametara ukoliko kompajler može na osnovu konteksta da odredi koji konkretni tipovi odgovaraju parametrima generičkog tipa.
  PRIMER: Kutija<Integer> kutijaInteger = new Kutija<>();
  Kutija<Integer> kutijaInteger = new Kutija();

Ugnježdene klase

- Definicija jedne klase može da se nađe unutar druge klase. Unutrašnja klasa se tada naziva ugnježdena klasa.

```
public class Spoljasnja{
        private class Unutrasnja{
                ...
        }
}
```

- Ugnježdena kasa je u nekakvoj vezi sa spoljašnjom klasom – kao što su to atributi i metodi, i ugnježdena klasa je član spoljašnje klase.*
- Ugnježdene klase se koriste za grupisanje klasa koje se koriste samo na jednom mestu (kada je neka klasa od koristi samo jednoj klasi).
- Kako je ugnježdena klasa član spoljašnje klase, u nekim aspektima ima drugačije ponašanje u odnosu na samostalne klase.

* zato joj se može dodeliti private modifikator pristupa!

Ugnježdene klase

- Ugnježdena klasa može biti statička ili nestatička.
- Nestatička ugnježdena klasa ima važnost samo u kontekstu obejkata spoljašnje klase. Dok se ne kreira objekat tipa spoljašnje klase ne može se kreirati objekat tipa ugnježdene klase.*
- Objekti statičke ugnježdene klase nisu zavisni od objekata spoljašnje klase, mogu se kreirati nezavisno od objekata odgovarajuće spoljašnje klase.
- Statičke ugnježdene klase mogu imati statičke atribute i metode, dok nestatičke ne mogu.
- Metodi statičke ugnježdene klase mogu da pristupe samo statičkim članovima (atributima i metodima) spoljašnje klase, za razliku od metoda nestatičke ugnježdene klase koji mogu da pristupe svim članovima spoljašnje klase.

* kreiranje objekta spoljašnje klase ne povlači nužno kreiranje objekta unutrašnje klase, osim ako to ne radi konstruktor spoljašnje klase!