Ugnježdene klase

Staša Vujičić Stanković

Ugnježdene klase

- Definicija jedne klase može se naći unutar druge klase.
- Unutrašnja klasa se tada naziva **ugnježdena klasa**. Naravno, ugneždena klasa je u nekakvoj vezi sa spoljašnjom klasom.
- Ugnježdena klasa je članica spoljašnje klase i na isti način joj se dodeljuju prava pristupa.

```
public class Spoljasnja {
    public class Unutrasnja {
    }
}
```

Ugnježdene klase

- U prethodnom primeru, klasa **Unutrašnja** deklarisana je kao public članica,
  tako da je vidljiva izvan klase **Spoljašnja**.
- Ovakva ugnježdena klasa ima važnost samo u kontekstu objekta spoljašnje klase,
  iz razloga što nije deklarisana kao statička članica.
- Dok se ne kreira objekat tipa **Spoljašnja**,
  ne može se kreirati objekat tipa **Unutrašnja**.

Ugnježdene klase

- Deklaracija objekta tipa **Spoljašnja**,
  ne povlači nužno kreiranje objekta tipa **Unutrašnja**,
  osim ako, naravno, to ne radi konstruktor.

```
Spoljasnja sp = new Spoljasnja();
 /* kreira se objekat klase Spoljasnja,
ali ne i objekat klase Unutrasnja */
Spoljasnja.Unutrasnja un = sp.new Unutrasnja();
/* kreira se objekat tipa Unutrasnja.
Tip ugneždene klase kvalifikovan je tipom spoljašnje klase. */
```

- Ovim smo kreirali objekat ugnježdenog klasnog tipa koji je povezan sa objektom **sp** koji je ranije kreiran.

Ugnježdene klase

- Unutar nestatičkih metoda klase **Spoljasnja**, ime klase **Unutrasnja** može se koristiti bez kvalifikovanja, jer će automatski biti kvalifikovano promenljivom **this**.

- Stoga je moguće unutar nestatičkog metoda klase **Spoljasnja** kreirati objekat klase **Unutrasnja** na sledeći način:

```
Unutrasnja un = new Unutrasnja();
```

To se tumači kao:

```
this.Unutrasnja un = this.new Unutrasnja();
```

Ugnježdene klase

- Statički metod spoljašnje klase ne može da kreira objekte nestatičke ugnježdene klase.
- Pošto je u gornjem primeru klasa Unutrasnja nestatička članica klase Spoljasnja, ukoliko statički metod klase Spoljasnja pokuša da kreira objekat klase Unutrasnja direktno, bez prethodnog uključenja objekta klase Spoljasnja, to bi značilo pokušaj kreiranja objekta van opsega objekta klase Spoljasnja.
- Sa druge strane, pošto klasa Unutrasnja nije statička članica klase Spoljasnja, ne može sama po sebi da sadrži nijednu statičku članicu. Dakle, ne može da se ponaša kao samostalna klasa sa statičkim članicama.

Ugnježdene klase

- Da bi kreirali objekte ugnježdene klase nezavisno od objekata spoljašnje klase, deklarišemo ugnježdenu klasu kao **static**.

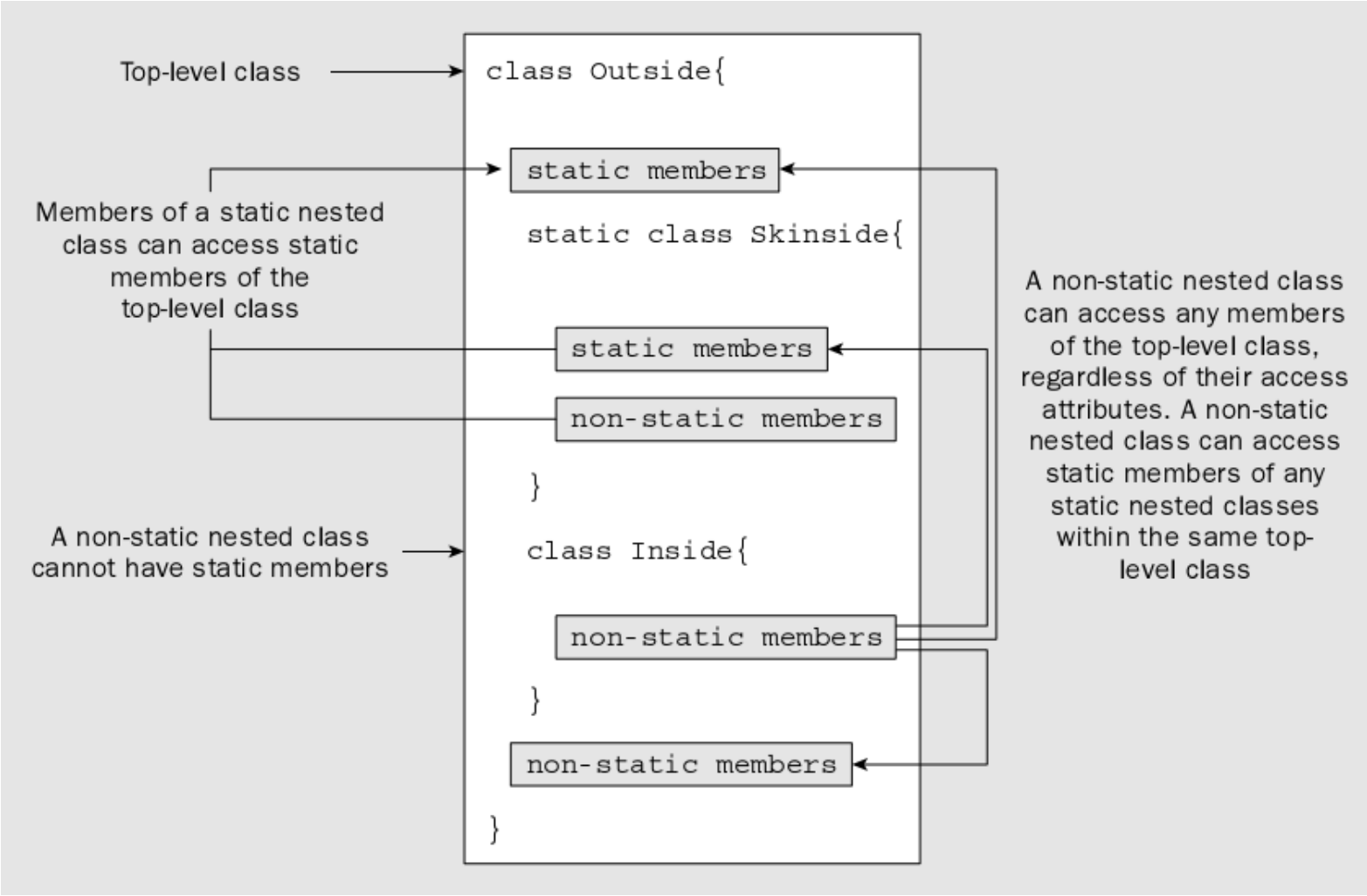
```
public class Spoljasnja {
        public static class StaticUn {
        }
}
```

- Sada možemo da deklarišemo objekat klase StaticUn nezavisno od objekta tipa Spoljasnja i to bez obzira da li je kreiran i jedan objekat klase Spoljasnja.
- Moramo da koristimo ime ugnježdene klase kvalifikovano imenom spoljašnje klase pri kreiranju objekta.

Ugnježdene klase

- Statičke ugnježdene klase mogu imati statičke članice, dok nestatičke ne mogu.
- Članice statičke ugnježdene klase mogu da pristupe statičkim članicama spoljašnje klase.
- Nestatička ugnježdena klasa može da pristupi bilo kojoj članici spoljašnje klase, bez obzira na njihove atribute pristupa.
- Takođe, ona može da pristupi i statičkim članicama bilo koje statičke ugnježdene klase unutar iste spoljašnje klase.

Ugnježdene klase



Primeri – Zec iz sesira

- Ukoliko stavimo da je klasa Zec nestatička članica klase MagicniSesir, program neće moći da se prevede.
- Problem su statičke članice imenaZeceva i brojImenaZeceva, jer nestatička ugnježdena klasa ne sme da sadrži statičke članice.
- Ukoliko se ove članice postave da budu nestatičke, svaki objekat klase Zec imaće svoju kopiju ovih nizova, što nije poželjno. Ozbiljniji problem je što imenovanje zečeva neće biti izvršeno korektno, tj. neće se generisati jedinstvena imena.

- Rešenje je da se ova dva niza zadrže kao static, ali da se prebace u spoljašnju klasu.
- Efekat je sličan prethodnom slučaju kada je klasa Zec bila static.
- Iako se dobija sličan efekat rada, ono što se događa u međusobnom odnosu objekata je dosta drugačije. Objekti klase Zec kreirani pozivom konstruktora klase MagicniSesir su sada vezani za konkretan objekat MagicniSesir koji je kreiran. Kada pozovemo konstruktor Zec(), to je kao da smo pozvali this.Zec().