Објектно орјентисано програмирање

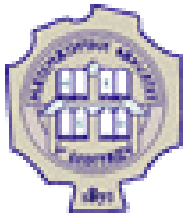
Владимир Филиповић
vladaf@matf.bg.ac.rs
Александар Картељ
kartelj@matf.bg.ac.rs

Унутрашње класе

Владимир Филиповић
vladaf@matf.bg.ac.rs
Александар Картељ
kartelj@matf.bg.ac.rs

Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs



Угнеждене класе

- Јава допушта да се дефинише класа унутар неке друге класе. Таква класа се назива **угнеждена класа**.

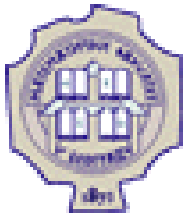
- Постоје два начина креирања угнеждених класа:

```
class OuterClass {
        ...
        static class StaticNestedClass {
                ...
        }
        ...
        class InnerClass {
                ...
        }
}
```

- Прва се назива **статичка угнеждена класа**, а друга само **унутрашња класа**.

Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs



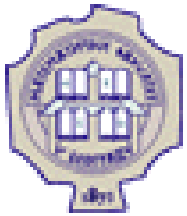
Угнеждене класе (2)

Постоји неколико разлога за коришћење угнеждених класа:

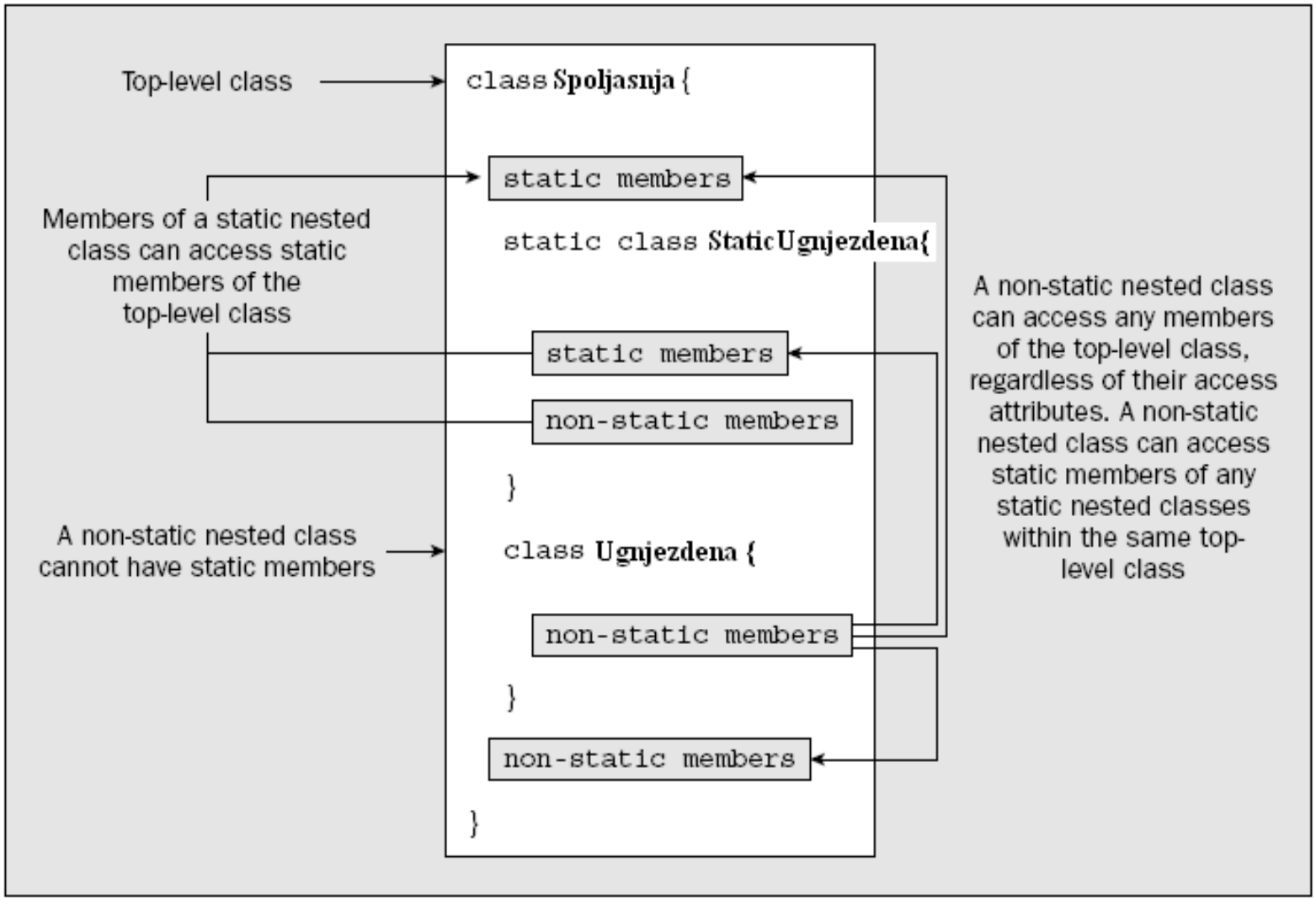
1. То је начин логичког груписања класа које се користе само на једном месту.
2. Тиме се побољшава енкапулација (учаурење).
   Ако се класа B смести унутар класе A, поља класе A могу бити приватна а класа B и даље може да им приступа.
3. Угнеждене класе доводе до читљивијег кода који се лакше одржава.
   Угнеждавање малих класа у велику доводи до тога да код неке операције буде смештен близу места коришћења.

Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs

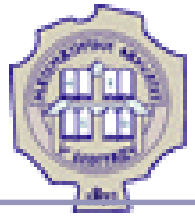


Угнеждене класе (3)



Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs

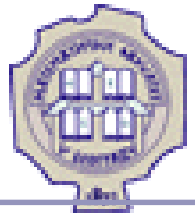


Угнеждене класе (4)

```
package ugnjezdeneklase;

public class Spoljasnja {
        private static int sp;
        private int np;
        private void nm(){}
        private static void sm(){}

        //instanca ugnjezdene staticke klase nije uslovljena postojanjem instance spoljne klase
        static class StatickaUgnjezdena{
                private static int sp;
                private int np;

                private void nm(){
                        //pristup spoljnoj statickoj
                        Spoljasnja.sp=5;
                        //ne mozemo pristupiti nestatickoj promenljivoj iz
                        //staticke klase
                        //Spoljasnja.np = 8;
                        //ne mozemo pristupiti ni nestatickoj metodi
                        //spoljne klase
                        //Spoljasnja.nm();
                        //ali mozemo statickoj metodi spoljne klase
                        Spoljasnja.sm();
                }
                private static void sm(){
                        //slicno kao i u nestatickoj nm() metodi,
                        //mozemo pristupiti sledecim elementima spoljne klase
                        Spoljasnja.sp=5;
                        Spoljasnja.sm();
                }
        }
```

Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs

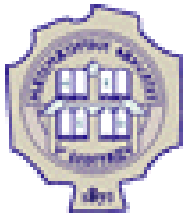


Угнеждене класе (5)

```
class Unutrasnja{
        //unutrasnja klasa ne moze imati staticke promenljive i metode
        //private static int sp;
        private int np;
        private void nm(){
                //pristup spoljnim statickim i nestatickim promenljivama
                Spoljasnja.sp=5;
                //ovo je np iz objekta Spoljasnje klase
                Spoljasnja.this.np=7;
                //a ovo je np iz objekta klase Ugnjezdena
                np=8;
                //slicno i za metode
                Spoljasnja.sm();
                //nestaticka metoda spoljne klase
                Spoljasnja.this.nm();
                //nestaticka metoda ove klase (rekurzija!)
                nm();
                //pristup statickim elementima klase StatickaUgnjezdena
                StatickaUgnjezdena.sp=5;
                StatickaUgnjezdena.sm();
        }
}
public static void main(String[] args) {
        //ne moze ovako
        //Unutrasnja un=new Unutrasnja();
        Spoljasnja sp=new Spoljasnja();
        Unutrasnja un=sp.new Unutrasnja();
        StatickaUgnjezdena su=new StatickaUgnjezdena();
}
}
```

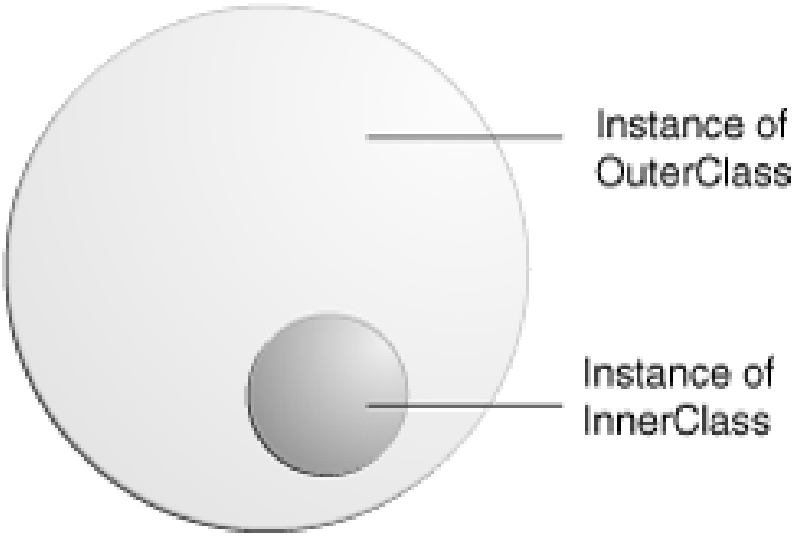
Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs



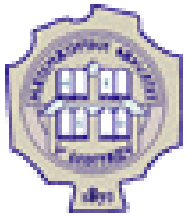
Унутрашње класе

- Унутрашња класа придружује примерку класе која је обухвата, па и она има директан приступ до свих поља и метода објекта који је садржи.
- Како је унутрашња класа придружена примерку тј. објекту, то она сама не може садржати статички члан.
- Објекти примерци унутрашње класе постоје само у оквиру примерка спољашње класе.



Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs

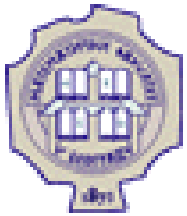


Унутрашње класе (2)

- Када се дефинише унутрашња класа, она је члан спољашње класе на исти начин као и остали чланови (поља и методе).
- Унутрашња класа може имати модификаторе приступа као и остали чланови.
- Угњеждена класа би требало да има специфичну повезаност са спољашњом класом.
  - Произвољно угњеждење једне класе у другу нема много смисла.
- Класа највишег нивоа је класа која садржи угњеждену класу, али сама по себи није угњеждена

Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs



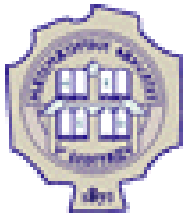
Пример унутрашње класе

- Последом наредбом је креиран објекат типа Spoljasnja.

```
public class Spoljasnja {
	// Unutrasnja
	public class Unutrasnja {
		//Detalji unutrasnje klase ...
	}
	// Ostali clanovi Spoljasnje klase
}

Spoljasnja sp = new Spoljasnja();
```

Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs



Пример унутрашње класе (2)

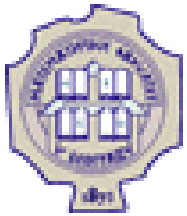
- Последњом наредбом на претходном слајду није креиран објекат класе Unutrasnja.
- Ако је потребно да се креира примерак унутрашње класе, то се постиже тако што се реферише на име унутрашње класе са именом спољашње класе као квалификатором:

```
// definisanje objekta unutrasnje klase
Spoljasnja.Unutrasnja unut = sp.new Unutrasnja();
```

- У горњем примеру је креиран објекат унутрашње класе који је придружен раније креираном објекту sp.

Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs



Пример унутрашње класе (3)

- Унутар нестатичких метода класе Spoljasnja, може се користити име класе Unutrasnja без квалификовања.
- Тако се унутар нестатичког метода класе Spoljasnja може креирати нови примерак класе Unutrasnja са:

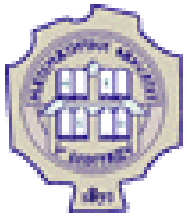
```
Unutrasnja unut = new Unutrasnja();
```

што је еквивалнтно са:

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

Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs



Локалне унутрашње класе

- Класа се може дефинисати унутар метода, и то је тзв. локална унутрашња класа (енг. local inner class).
- Примерци тј. објекти локалне унутрашње класе могу се креирати само локално, тј. унутар метода у коме је и дефиниција класе.
- То је корисно када израчунавање у методу захтева употребу специјализоване класе.
  - Добар пример су ослушкивачи догађаја (енг. event listener).
- Локална унутрашња класа може референцирати променљиве декларисане у методу у чијој дефиницији се појављује, али само ако су те променљиве финалне.

Математички факултет

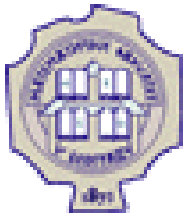
{vladaf, kartelj}@matf.bg.ac.rs

Локалне унутрашње класе (2)

```
public class PrimerLokalneKlase {
        static String reg= "[^0-9]";
        public static void validirajBrojTelefona(String broj) {
                //mora da bude finalna kako bi se videla unutar lokalne metode
                final int duzinaBroja = 10;
                //lokalna klasa jer je koristimo samo za potrebe ove metode
                class BrojTelefona {
                        String formatiraniBroj = null;
                        BrojTelefona(String broj) {
                                String trenutniBroj =broj.replaceAll(reg, "");
                                if (trenutniBroj.length() == duzinaBroja)
                                        formatiraniBroj = trenutniBroj;
                                else
                                        formatiraniBroj = null;
                        }
                        public String vratiBroj() {
                                return formatiraniBroj;
                        }
                }
                BrojTelefona mojBroj = new BrojTelefona(broj);
                if (mojBroj.vratiBroj() == null)
                        System.out.println("Broj nije dobar");
        }

        public static void main(String... args) {
                validirajBrojTelefona("123-456-7890");
        }
}
```

Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs

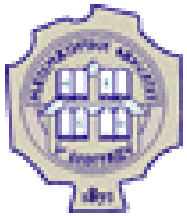


Анонимне класе

- Анонимне класе омогућују програмеру да пише концизан код.
- Оне допуштају да се истовремено декларише класа и креира њен примерак, при чему та класа нема име.
- Другим речима, оне су као локална класа, само што немају име. Анонимне класе се користе када се локална класа користи само једном.

Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs

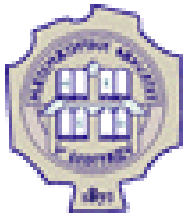


Анонимне класе (2)

```
public class Student {
        private int godinaRodjenja;
        private String ime, prezime;

        public Student(int godinaRodjenja, String ime, String prezime) {
        this.godinaRodjenja = godinaRodjenja;
                this.ime = ime;
                this.prezime = prezime;
        }

        public static void main(String[] args) {
                Student[] nizStudenata = new Student[3];
                nizStudenata[0] = new Student(1992, "Ana", "Pesic");
                nizStudenata[1] = new Student(1990, "Mirko", "Petrovic");
                nizStudenata[2] = new Student(1985, "Drgana", "Djordjevic");
                /*Koristimo anonimnu klasu koja nasledjuje Comparator jer nam treba specifican
                nacin sortiranja niza i to samo jedanput. Neefikasna alternativa bi bila da smo
                nasledili klasu komparator nasom klasom npr. MojKomparator, potom redefinisali
                njen compare metod, ali bi to bilo previse koda za jednokratnu upotrebu*/
                Arrays.sort(nizStudenata, new Comparator() {
                        @Override public int compare(Object o1, Object o2) {
                                Student s1 = (Student) o1; Student s2 = (Student) o2;
                                return s1.godinaRodjenja - s2.godinaRodjenja;
                        }
                });
        }
}
```

Математички факултет

{vladaf, kartelj}@matf.bg.ac.rs



Захвалница

Велики део материјала који је укључен у ову презентацију је преузет из презентације коју је раније (у време када је он држао курс Објектно орјентисано програмирање) направио проф. др Душан Тошић.

Хвала проф. Тошићу што се сагласио са укључивањем тог материјала у садашњу презентацију, као и на помоћи коју ми је пружио током конципцирања и реализације курса.

Надаље, један део материјала је преузет од колегинице Марије Милановић.

Хвала Марији Милановић на помоћи у реализацији ове презентације.