Klase i Objekti

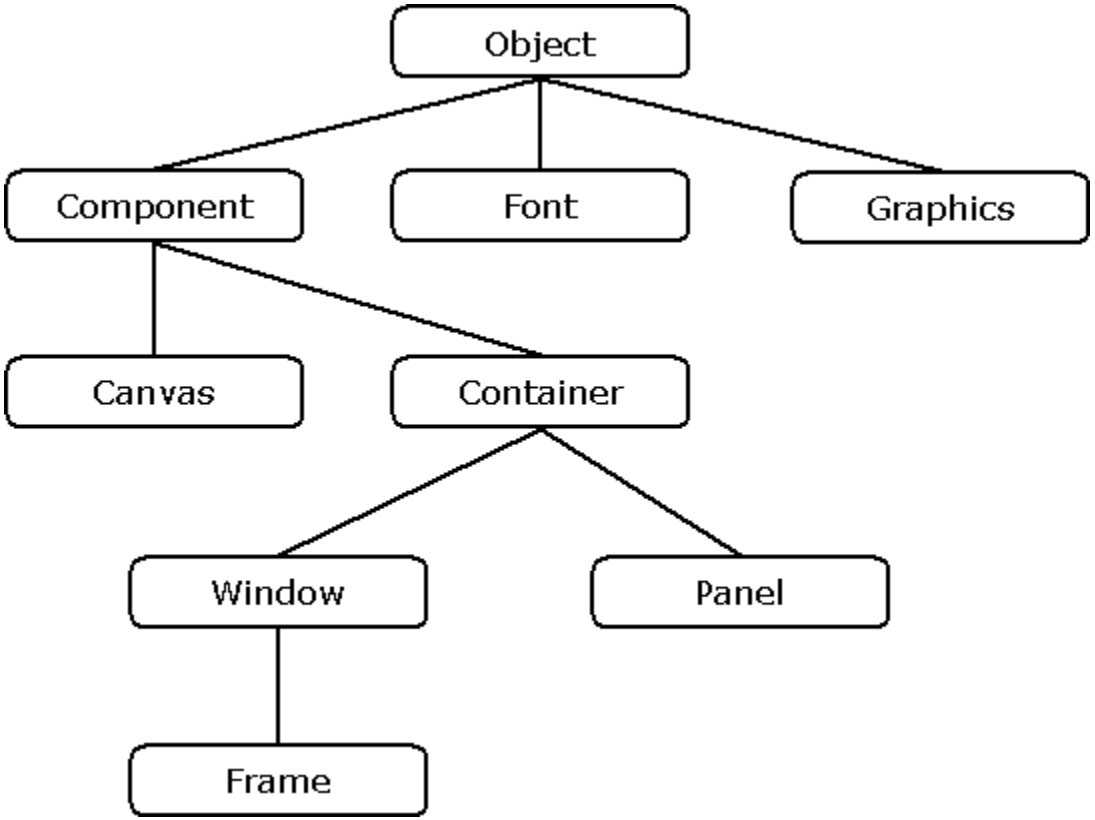
Primitivni tipovi

- Primitivni tipovi nisu objekti
- Elementi proceduralnog programiranja

Naziv	**Podrazumevana vrednost**
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d
char	'\u0000'
boolean	false

Objekti

- Svaka klasa podrazumevano nasleđuje klasu Object



Klasa Object

Metode	Namena
protected Object clone()	klonira objekat
boolean equals(Object o)	proverava da li je ovaj objekat isti kao prosleđeni
int hashCode()	vraća specijalni celobrojni identifikator objekta
String toString()	prikazuje objekat u formatu teksta
…	

- Sve klase imaju ove metode. Zašto?

Konstruktor klase

- Metod koji služi za kreiranje objekta
- Može biti više različitih konstruktora

```
public class Bicikl{
        …
        public Bicikl(int pocetnaBrzina, int pocetniPogon){
                this.brzina=pocetnaBrzina;
                this.pocetniPogon=pocetniPogon;
        }
        //this je referenca na trenutni objekat

        @Override
        String public toString(){
                return pocetnaBrzina+” ”+pocetniPogon;
        }
}
```

Konstruktor klase (nasleđeni)

- Nasleđena klasa može koristiti nasleđeni konstruktor

```
public class BrdskiBicikl extends Bicikl{
        int visinaSedista;
        //ostala polja su nasleđena

        public BrdskiBicikl(int pocetnaBrzina, int pocetniPogon, int visinaSedista){
                super(pocetnaBrzina, pocetniPogon);
                this.visinaSedista=visinaSedista;
        }
        //super je referenca na nadklasu
}
```

Modifikatori pristupa

Modifikator	Efekat nad poljem i metodom
default (bez modifikatora)	vidljivo u paketu
private	vidljivo samo u klasi
public	vidljivo svuda
protected	vidljivo samo u nasleđenim klasama

Objektne i statičke metode

- Operacija koja se izvršava nad kreiranim objektom
- Razlika u odnosu na statičku operaciju tj. funkciju

```
class Bicikl{
        public Bicikl(){}
        void vozi(){…}
        static Bicikl popravi(Bicikl b){…}

        public static void main(String args[]){
                Bicikl b1 = new Bicikl();
                b.vozi();
                Bicikl b2 = Bicikl.popravi(b1);
        }
}
```

Preopterećivanje metoda

- Mogućnost da se jedna metoda koristi za različite ulazne argumente
- Isto važi i za konstruktore

```
class Platno{
        void crtaj(String tekst);
        void crtaj(Point tacka);
        void crtaj(int broj1, int broj2);
}
```

Prosleđivanje argumenata metodi ili konstruktoru

- Primitivni tip se ne može menjati u metodi
- Objekat može

```
class Test{
        void f(int x, Integer y){
                x+=3;
                y+=4;
        }
        static void main(String args[]){
                int x = 5; Integer y=10;
                f(x,y);
                System.out.println(x+” ”+y);
        }
}
```

Zadatak 1

- Definisati klase za figure: Tacka, Duz, Krug, Pravougaonik
- Za svaki od njih dati odgovarajuću reprezentaciju (sva polja postaviti da budu vidljiva u okviru paketa)
- Napisati konstruktore za sve klase
  - za pravougaonik, krug i duž dati po 2 konstruktora, npr. pravougaonik se može iscrtati korišćenjem informacija gornjem levom ćošku, dužini i širini, ili npr. na osnovu gornjeg levog ćoška i donjeg desnog

Zadatak 1 - nastavak

- Napisati drugu klasu Platno koja u konstruktoru prihvata širinu i visinu
- Ove dve vrednosti se koriste da bi se kreirala matrica boolean tipa date širine i visine (inicijalno su sve vrednosti false)
- Napraviti preopterećene metode pod nazivom “iscrtaj” za iscrtavanje svih ranije uvedenih figura
  - Svaka od metoda iscrtava figuru tako što postavlja true na adekvatnu poziciju u matrici

Zadatak 1 - nastavak

- U klasi platno napraviti i metod **prikazi** koji prikazuje platno nakon iscrtavanja nekoliko objekata.
  - Polja true se prikazuju kao zvezdica, dok se false prikazuju kao prazna mesta
- Kreirati klasu TestPlatno i u njoj testirati program
  - Napraviti objekat Platno dimenzija 50 x 50
  - Na njemu iscrtati po jedan objekat svake figure
  - Prikazati sve ovo na gore opisani način u konzoli