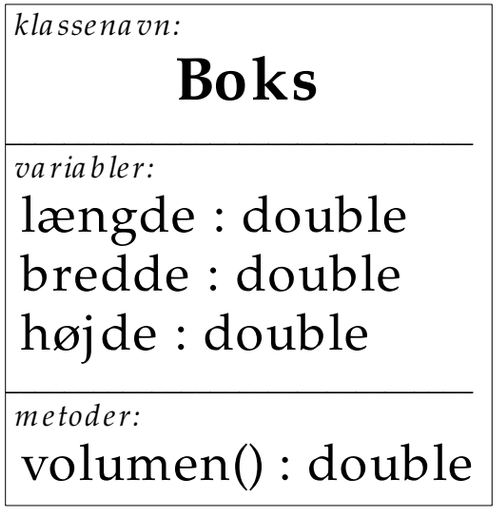
Kursus i OOP og Java

- Kursus i Objektorienteret programmering i Java
-

Åben Dokumentlicens

- Dette foredragsmateriale er under Åben Dokumentlicens (ÅDL)
  - Du har derfor lov til frit at kopiere dette værk
  - Bruger du dele af værket i et nyt værk, skal de dele, der stammer fra dette værk, igen frigives under ÅDL
  - Den fulde licens kan ses på http://www.sslug.dk/linuxbog/licens.html

Objekter og klasser



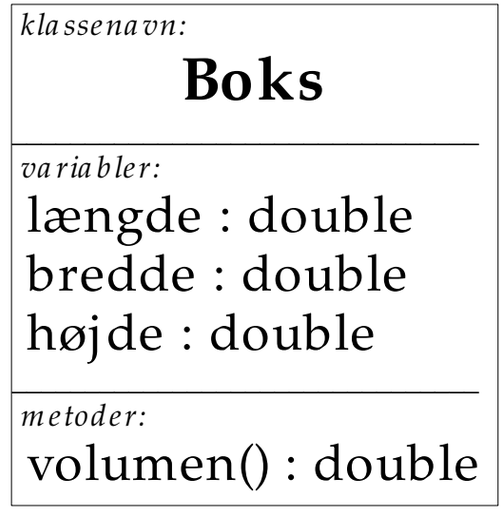

```
public class Boks
{
        double længde;
        double bredde;
        double højde;

        double volumen()
        {
                double vol;
                vol = længde*bredde*højde;
                return vol;
        }
}
```

```
public class BenytBoks
{
        public static void main(String[] arg)
        {
                double rumfang;
                Boks boksobjekt;
                boksobjekt = new Boks();
                boksobjekt.længde= 12.3;
                boksobjekt.bredde= 2.22;
                boksobjekt.højde = 6.18;
                rumfang = boksobjekt.volumen();
                System.out.println("Boksens volume: "+ rumfang);
        }
}
```

```
Boksens volume: 168.75108
```

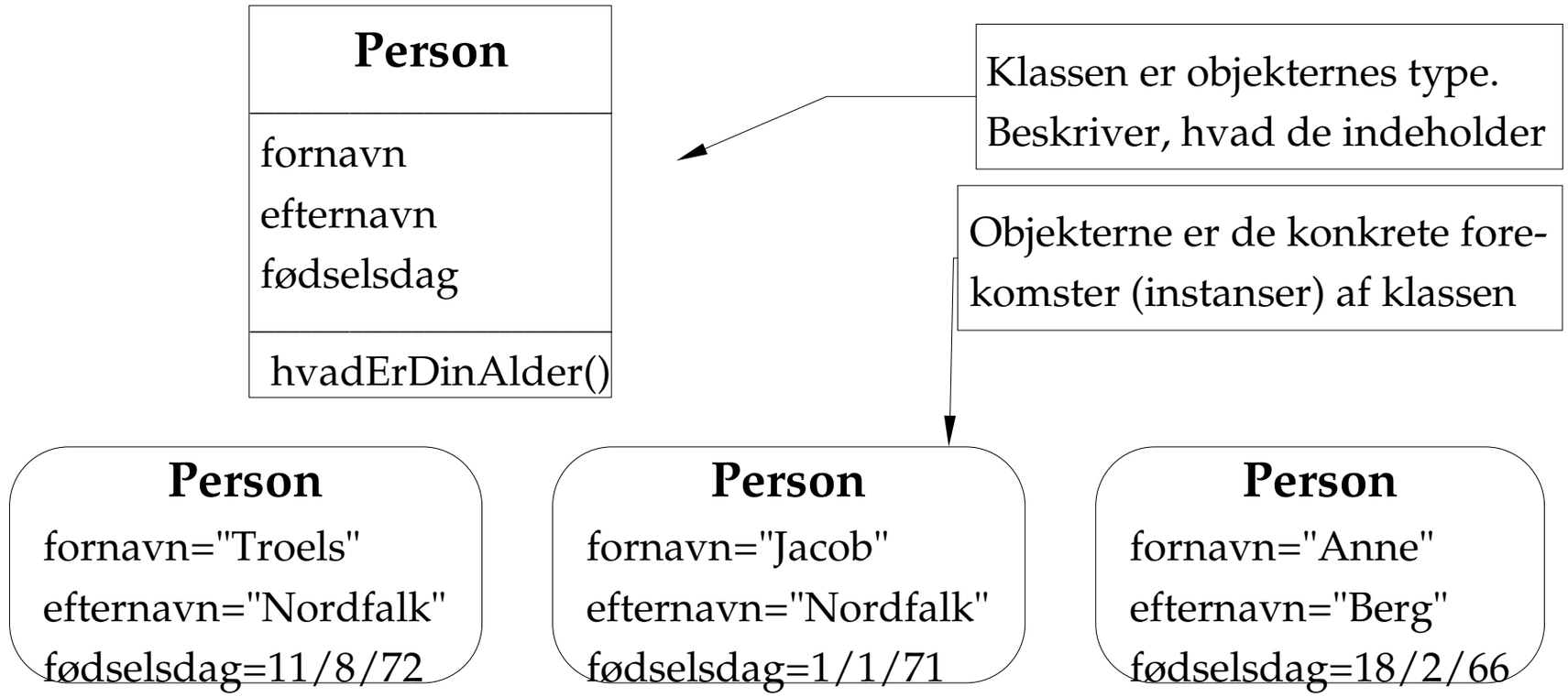
Objekter og klasser




```
public class BenytBokse
{
        public static void main(String[] arg)
        {
                Boks boks1, boks2;
                boks1 = new Boks();
                boks2 = new Boks();
                boks1.længde= 12.3;
                boks1.bredde= 2.22;
                boks1.højde= 6.18;
                boks2.længde= 13.3;
                boks2.bredde= 3.33;
                boks2.højde= 7.18;
                double v1, v2;
                v1 = boks1.volumen();
                v2 = boks2.volumen();
                System.out.println("Volumenforskel: "+ (v2 - v1));
        }
}
```

```
 Volumenforskel: 149.24394
```

Objekter og klasser



- Giv flere (mindst 3) eksempler for Person-klassen på
  - Data
    - foranderlige eller uforanderlige?
  - Metoder
    - kommandoer eller spørgsmål?
- Gør det samme for Punkt (i et x-y-koordinatsystem)

Samlinger af data

- Eksempel på brug af liste
  - `List liste;`
  - `liste = new ArrayList<String>();`
  - `liste.add("æh");`
  - `liste.add("øh",0);`

- Gennemløb v.hj.a. tællevariabel

```
for (int i=0; i<liste.size(); i++) {
        String s = liste.get(i);
        // gør noget med s
        System.out.println(s);
}
```

Objekter og klasser

```
import java.util.*;
public class Eventyr
{
  public static void main(String[] arg)
  {
    ArrayList<String> personer = new ArrayList<String>();
    personer.add("De tre små grise");
    personer.add("Ulven");
    personer.add("Rødhætte");

    ArrayList<String> handlinger = new ArrayList<String>();
    handlinger.add("slikker sig om munden");
    handlinger.add("får en idé!");
    handlinger.add("gemmer sig i skoven");

    for (int i=0; i<5; i++) {
      String person   = personer.get(   (int) (Math.random()*3));
      String handling = handlinger.get( (int) (Math.random()*handlinger.size()));
      System.out.println(person + " " + handling);
    }
  }
}

Ulven får en idé!
Ulven slikker sig om munden
Rødhætte gemmer sig i skoven
De tre små grise slikker sig om munden
Ulven gemmer sig i skoven
```

Test hinanden

- Slå op på afsnit 3.7 (Test dig selv)

Hvis I kan svare på spørgsmålene herunder, kan I være rimelig sikker på, at I har forstået alt det væsentlige i kapitlet.
Sæt jer sammen to og to og hør hinanden i spørgsmålene.
I afsnit 3.8 er nogle vejledende svar

1)
- 1)Hvad er et objekt?
- 2)Hvad er en klasse?
- 4)Hvordan oprettes et objekt?
- 5)Hvordan kan man ændre på et objekt (dets data)?
- 6)Hvad er en metode?
- 7)Hvad er en returtype?
- 8)Hvad er en parameter?

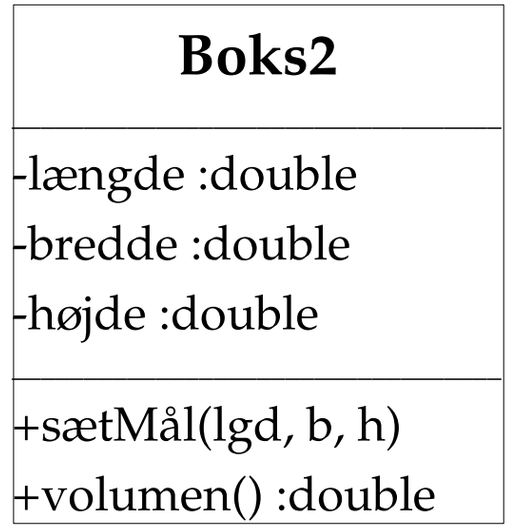
-

Indkapsling

```
public class Boks2
{
        private double længde;
        private double bredde;
        private double højde;

        public void sætMål(double lgd, double b, double h)
        {
                if (lgd<=0 || b<=0 || h<=0)
                {
                        længde = 10.0;
                        bredde = 10.0;
                        højde  = 10.0;
                        System.out.println("Ugyldige mål. Bruger standardmål.");
                } else {
                        længde = lgd;
                        bredde = b;
                        højde  = h;
                }
        }

        public double volumen()
        {
                double vol;
                vol = længde*bredde*højde;
                return vol;
        }
}
```



Konstruktører

Boks3
-længde :double -bredde :double -højde :double
+Boks3() +Boks3(lgd, b, h) +sætMål(lgd, b, h) +volumen() :double

```
public class Boks3
{
    private double længde;
    private double bredde;
    private double højde;

    public Boks3() {
       længde = 10.0;
       bredde = 10.0;
       højde  = 10.0;
    }

    public Boks3(double lgd, double b, double h) {
       sætMål(lgd,b,h);
    }

    public void sætMål(double lgd, double b, double h) {
       if (lgd<=0 || b<=0 || h<=0)
       {
          System.out.println("Ugyldige mål. Bruger standardmål.");
          længde = 10.0;
          bredde = 10.0;
          højde  = 10.0;
       } else {
          længde = lgd;
          bredde = b;
          højde  = h;
       }
    }

    public double volumen() {
       double vol = længde*bredde*højde ;
       return vol;
    }
}
```

```
public class BenytBoks3
{
    public static void main(String[] arg)
    {
       Boks3 enBoks;
       enBoks = new Boks3();              // brug konstruktøren
       System.out.println("Volumen er: "+ enBoks.volumen());

       Boks3 enAndenBoks;
       enAndenBoks = new Boks3(5,5,10);  // brug den anden kons
       System.out.println("Volumen er: "+ enAndenBoks.volumen()
    }
}
```

```
Volumen er: 1000.0
Volumen er: 250.0
```

Konstruktører

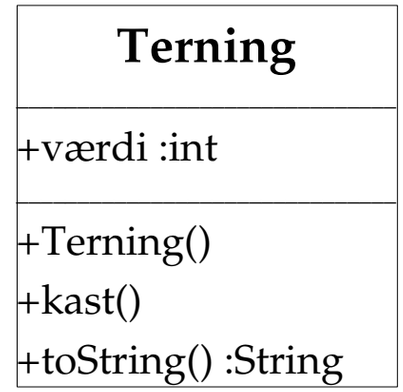
- speciel metode, der har samme navn som klassen
- kaldes automatisk ved oprettelse af et objekt med 'new'
- skal initialisere variablerne i det nye objekt
- Hvad hvis der ikke er nogen konstruktør defineret?
  - Så definerer java en uden parametre
    - Standardkonstruktøren

Objektororienteret modellering

- Hvad er en klasse?
- Hvad giver det mening at oprette et objekt ud fra?
  - Noget hvor hvert objekt har nogle unikke data
  - Noget med metoder (kommandoer/spørgsmål)

- Klasse: Terning
  - Hvilke (relevante) data?
  - Hvilke (relevante) kommandoer?
  - Hvilke (relevante) spørgsmål?
- Dette er Ternings *ansvarsområder* i et program

Terning-klassen




```
public class BenytTerning
{
    public static void main(String[] arg)
    {
       Terning t;
       t = new Terning();   // opret terning

       boolean sekser = false;
       int antalKast = 0;

       while (sekser==false)
       {
          t.kast();
          antalKast = antalKast + 1;
          System.out.println(
             "kast "+antalKast+": "+t.værdi);
          if (t.værdi == 6) sekser = true;
       }

       System.out.println(
          "Vi slog en 6'er efter "
          +antalKast+" slag.");
    }
}
```

```
kast 1: 4
kast 2: 2
kast 3: 6
Vi slog en 6'er efter 3 slag.
```

```
public class Terning
{
    public int værdi;

    public Terning()
    {
       kast(); // kald kast() der sætter værdi til noget fornuftigt
    }

    /** kaster terningen, så den får en anden værdi */
    public void kast()
    {
       // find en tilfældig side
       double tilfældigtTal = Math.random();
       værdi = (int) (tilfældigtTal * 6 + 1);
    }

    /** giver en beskrivelse af terningen som en streng */
    public String toString()
    {
       String svar = ""+værdi;  // værdi som streng, f.eks. "4"
       return svar;
    }
}
```

Relationer mellem objekter

- Ansvarsområder

Raflebæger-klassen

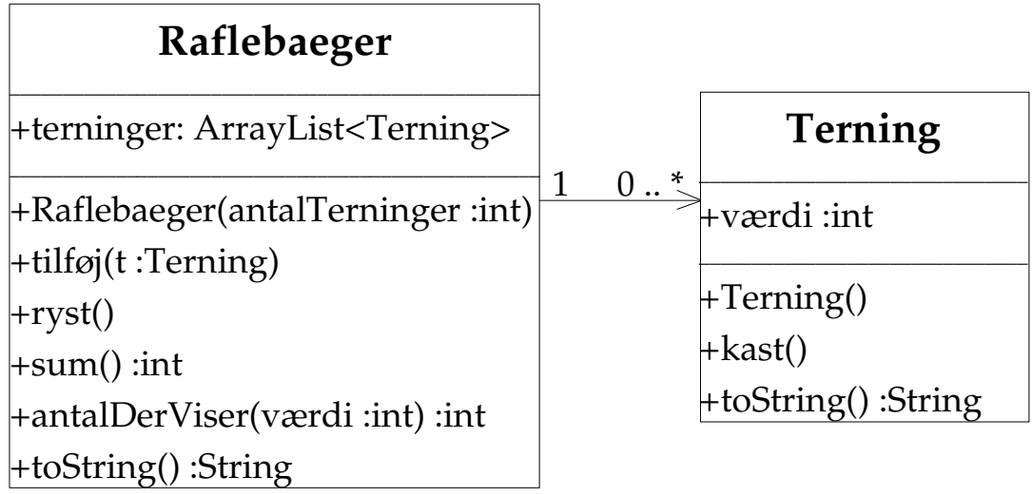
```
public class Raflebaeger
{
    public ArrayList<Terning> terninger;

    public Raflebaeger(int antalTerninger) {
       terninger = new ArrayList<Terning>();
       for (int i=0; i<antalTerninger; i++) {
          Terning t;
          t = new Terning();
          tilføj(t);
       }
    }

    public void tilføj(Terning t) {
       terninger.add(t);
    }

    public void ryst() {
       for (Terning t : terninger)
       {
          t.kast();
       }
    }

    public int sum() {
       int resultat=0;
       for (Terning t : terninger) {
          resultat = resultat + t.værdi;
       }
       return resultat;
    }
```



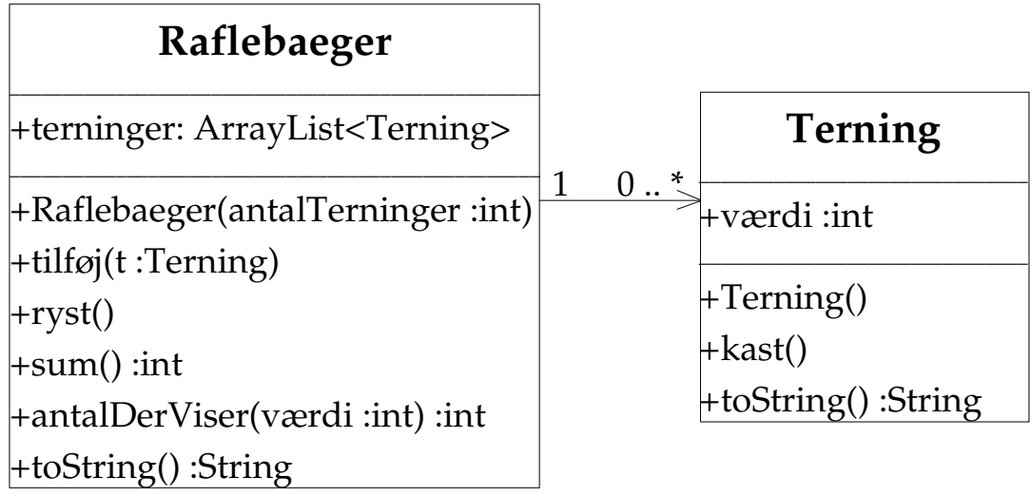

```
    public int antalDerViser(int værdi) {
       int resultat;
       resultat = 0;
       for (Terning t : terninger)
       {
          if (t.værdi==værdi)
          {
             resultat = resultat + 1;
          }
       }
       return resultat;
    }

    public String toString() {
       return terninger.toString();
    }
}
```

Raflebæger-klassen

```
public class BenytRaflebaeger
{
    public static void main(String[] arg)
    {
      Raflebaeger bæger;
      boolean toSeksere;
      int antalForsøg;

      bæger = new Raflebaeger(3);
      toSeksere=false;
      antalForsøg = 0;
      while (toSeksere==false)
      {
        bæger.ryst();               // kast alle terningerne
        System.out.print("Bæger: " + bæger + " sum: " + bæger.sum());
        System.out.println(" Antal 6'ere: "+bæger.antalDerViser(6)
                           + " antal 5'ere: "+bæger.antalDerViser(5));
        if (bæger.antalDerViser(6) == 2) {
          toSeksere = true;
        }
        antalForsøg++;
      }
      System.out.println("Du fik to seksere efter "+ antalForsøg+" forsøg.");
    }
}
```




```
Bæger: [4, 4, 4] sum: 12 Antal 6'ere: 0 antal 5'ere: 0
Bæger: [5, 5, 6] sum: 16 Antal 6'ere: 1 antal 5'ere: 2
Bæger: [2, 5, 6] sum: 13 Antal 6'ere: 1 antal 5'ere: 1
Bæger: [4, 2, 4] sum: 10 Antal 6'ere: 0 antal 5'ere: 0
Bæger: [6, 4, 1] sum: 11 Antal 6'ere: 1 antal 5'ere: 0
Bæger: [6, 6, 4] sum: 16 Antal 6'ere: 2 antal 5'ere: 0
Du fik to seksere efter 6 forsøg.
```

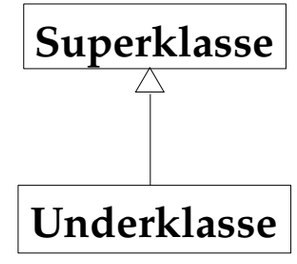
Test hinanden (afsnit 4.8)

- 1)Hvad er en klasse?
- 2)Hvad kan en klasse indeholde?
- 3)Lav en klassedefinition og beskriv syntaksen.
- 4)Hvad er en metode?
- 5)Definér en metode og beskriv syntaksen.
- 6)Hvad betyder det, at en metode returnerer noget?
- 7)Beskriv reglerne omkring return og returværdi.
- 8)Lav en metode, der finder kvadratet af et tal (x*x) og returnerer det.
- 9)Hvad er et objekt ?
- 10)Hvad er en objektvariabel?
- 11)Hvordan oprettes et objekt.
- 12)Hvad sker der, når et objekt oprettes?
- 13)Hvad er en konstruktør? Hvilken kode bør være i konstruktøren?
- 14)Hvad er standardkonstruktøren?

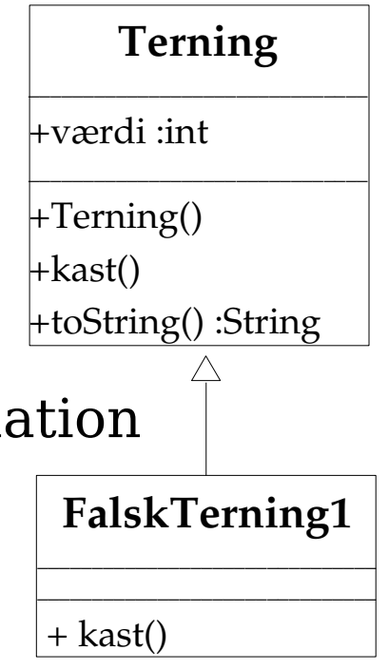
- Vejledende svar kan findes i afsnit 4.9.

Arv fra eksisterende klasser

- Hvad gør man, hvis man ønsker en klasse, der ligner en eksisterende klasse, men alligevel ikke helt er den samme?
  - Kopiere koden?
    - Svært at vedligeholde flere kopier af den samme kode
  - Kalde koden i den anden klasse?
    - Besværligt og svært at gennemskue programmet
- Nedarving
  - underklasse, der arver fra en anden klasse
    - genbruger al koden fra "stamklassen" (superklassen)
    - definerer kun den ekstra kode, der gør underklassen anderledes i forhold til superklasen



Falske terninger




```
/** En klasse der beskriver 6-sidede terninger */
public class Terning
{
    public int værdi;

    public Terning()
    {
       kast();
    }

    public void kast()
    {
       // find en tilfældig side
       double tilfældigtTal = Math.random();
       værdi = (int) (tilfældigtTal * 6 + 1);
    }

    public String toString()
    {
       String svar = ""+værdi;
       return svar;
    }
}
```

```
/** En Terning-klasse for falske terninger. */

public class FalskTerning1 extends Terning
{
    /** tilsidesæt kast med en "bedre" udgave */
    public void kast()
    {
       // udskriv så vi kan se at metoden bliver kaldt
       // System.out.println("[kast() på FalskTerning1] ");

       værdi = (int) (6*Math.random() + 1);

       // er det 1 eller 2? Så lav det om til 6!
       if ( værdi <= 2 ) værdi = 6;
    }
}
```

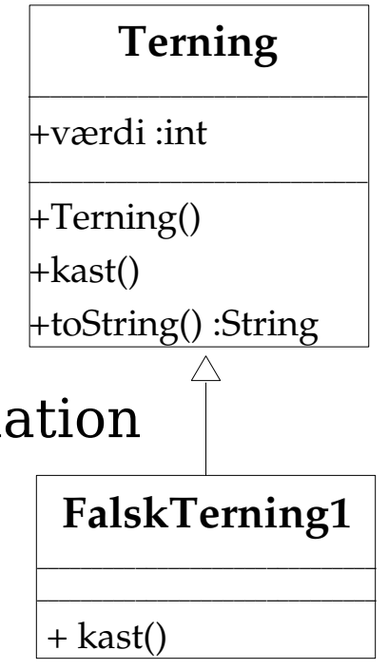
Falske terninger

```
public class Snydespil1
{
    public static void main(String[] arg)
    {
       Terning t1 = new Terning();
       FalskTerning1 t2 = new FalskTerning1();

       System.out.println("t1: "+t1);
       System.out.println("t2: "+t2);

       for (int i=0; i<5; i++)
       {
          t1.kast();
          t2.kast();
          System.out.println("t1=" + t1 + "  t2=" + t2);
          if (t1.værdi == t2.værdi)
             System.out.println("To ens!");
       }
    }
}
```

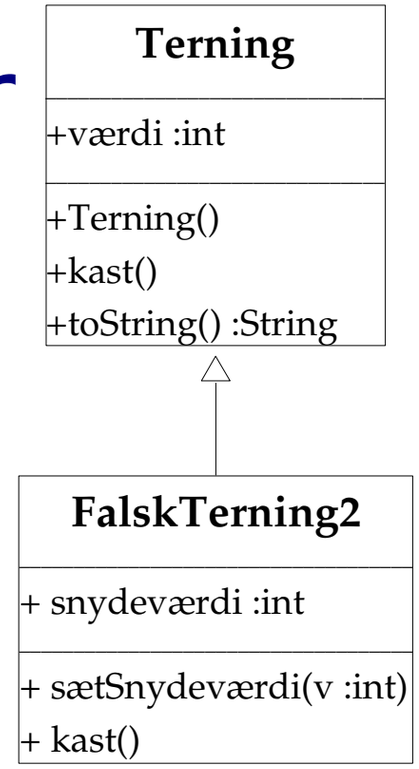
```
t1: 1
t2: 3
t1=1  t2=5
t1=1  t2=6
t1=4  t2=3
t1=6  t2=6
To ens!
t1=2  t2=6}
```



FalskTerning1 **er en** Terning, da den har alle de variabler og metoder, Terning har

FalskTerning1 har omdefineret (tilsidesat) kast()-metoden med en nu udgave

Arv: Ekstra metoder




```
/** En klasse der beskriver 6-sidede terninger */
public class Terning
{
    public int værdi;

    public Terning()
    {
       kast();
    }

    public void kast()
    {
       // find en tilfældig side
       double tilfældigtTal = Math.random();
       værdi = (int) (tilfældigtTal * 6 + 1);
    }

    public String toString()
    {
       String svar = ""+værdi;
       return svar;
    }
}
```

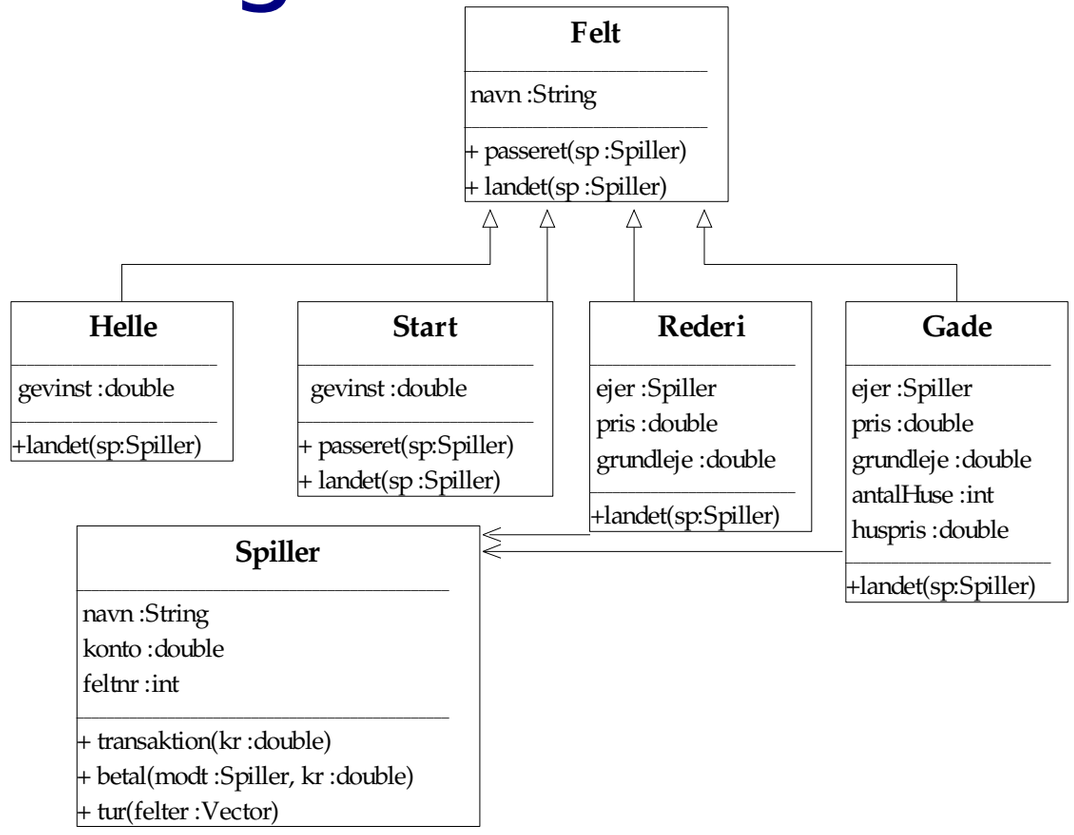
```
public class FalskTerning2 extends Terning
{
    public int snydeværdi;

    public void sætSnydeværdi(int nySnydeværdi)
    {
       snydeværdi = nySnydeværdi;
    }

    public void kast()
    {
       værdi = (int) (6*Math.random() + 1);

       // 1 eller 2? Så lav det om til snydeværdi!
       if ( værdi <= 2 ) værdi = snydeværdi;
    }
}
```

UML og klasserelationer



- Rederi og Gade *har en* Spiller.
- Helle, Start, Rederi og Gade *er-et* Felt.

```
public class Felt
{
    String navn;

    public void passeret(Spiller sp)
    {
       sp.besked("Du passerer "+navn);
    }

    public void landet(Spiller sp)
    {
    }
}
```

```
public class Helle extends Felt
{
    double gevinst;

    public Helle (int gevinst)
    {
       navn="Helle";
       this.gevinst=gevinst;
    }

    public void landet(Spiller sp)
    {
       sp.besked("Du lander på helle "
          +"og får overført "+gevinst);
       sp.transaktion(gevinst);
    }
}
```

```
public class Spiller
{
    String navn;
    double konto;
    int feltnr;

    public Spiller(String navn, double konto) {
       this.navn=navn;
       this.konto=konto;
       feltnr = 0;
    }

    public void besked(String besked) {
       System.out.println(navn+": "+besked);
    }

    public boolean spørgsmål(String spørgsmål)
    {
       String spm = navn+": Vil du "+spørgsmål+"?";
       String svar = javax.swing.JOptionPane.showInputDialog(spm, "ja");
       System.out.println(spm+" "+svar);
       if ("ja".equalsIgnoreCase(svar)) return true;
       else return false;
    }

    public void transaktion(double kr) {
       konto = konto + kr;
       System.out.println(navn+"s konto lyder nu på "
          +konto+" kr.");
    }

    public void betal(Spiller modtager, double kr) {
       System.out.println(navn+" betaler "
          +modtager.navn+": "+kr+" kr.");
       modtager.transaktion(kr);
       transaktion(-kr);
    }
}
```

```
public class Rederi extends Felt
{
    Spiller ejer;
    double pris;
    double grundleje;

    public Rederi(String navn, double pris, double leje) {
       this.navn = navn;
       this.pris = pris;
       this.grundleje = leje;
    }

    public void landet(Spiller sp) {
       sp.besked("Du er landet på "+navn);
       if (sp==ejer)
       {                                          // spiller ejer selv grunden
          sp.besked("Det er din egen grund");
       }
       else if (ejer==null)
       {                                          // ingen ejer grunden, så køb den
          if (sp.konto > pris)
          {
             if (sp.spørgsmål("købe "+navn+" for "+pris))
             {
                sp.transaktion( -pris );
                ejer=sp;
             }
          }
          else sp.besked("Du har ikke penge nok til at købe "+navn);
       }
       else
       {                                          // feltet ejes af anden spiller
          sp.besked("Leje: "+grundleje);
          sp.betal(ejer, grundleje);              // spiller betaler til ejeren
       }
    }
}
```

```
public class Gade extends Felt
{
    Spiller ejer;
    double pris;
    double grundleje;
    int antalHuse = 0;
    double huspris;

    public Gade(String navn, double pris, double leje, double huspris) {
       this.navn=navn;
       this.pris=pris;
       this.grundleje=leje;
       this.huspris=huspris;
    }

    public void landet(Spiller sp) {
       sp.besked("Du er landet på "+navn);

       if (sp==ejer {
          sp.besked("Det er din egen grund");
          if (antalHuse<5 && sp.konto>huspris && sp.spørgsmål("købe hus for "+huspris)) {
             sp.besked("Du bygger hus på "+navn+" for "+huspris);
             ejer.transaktion( -huspris );
             antalHuse = antalHuse + 1;
          }
       }
       else if (ejer==null) {
          if (sp.konto > pris && sp.spørgsmål("købe "+navn+" for "+pris)) {
             sp.transaktion( -pris );
             ejer=sp;
          }
       }
       else {                                    // felt ejes af anden spiller
          double leje = grundleje + antalHuse * huspris;
          sp.besked("Leje: "+leje);
          sp.betal(ejer, leje);                  // spiller betaler til ejeren
       }
    }
```