Denne guide er oprindeligt
udgivet på Eksperten.dk

COMPUTERWORLD

Hvad er Objekter - Programmering

En rigtig god gennemgang af hvad objekter er! Hvordan de oprettes og anvendes! Det er helt klart til nybegyndere, som ikke helt har forstået hvad objekter går ud på:)

Skrevet den **05. Feb 2009** af **kalp** I kategorien **Programmering / Java** | ★★★☆☆

ARTIKEL ER HERMED GRATIS:) 27/5-05

På grund af personlige erfaringer med personer, som ikke har forstået begrebet "objekter" har jeg ladet mig inspirere til, at skrive denne artikel. Jeg mener selvfølgelig det er ret vigtigt, at have en forståelse for hvad objekter egentlig er, hvordan de andvendes og hvordan man laver objekter. Eksemplerne vil foregå i programmeringssproget Java, men er gældende for alle Objekt orienterede programmeringssprog vil jeg mene.

Hvad er objekter så? I den fysiske verden er det storset alt omkring dig. Din computerskærm, bord, seng, bil, hus og hvad ved jeg er alle objekter! Alle disse genstande, ting eller som vi nu kalder det objekter kan hver især beskrives(defineres) ud fra nogen værdier. Det kan være navn, størrelse, vægt, hastighed alt afhængig hvad der er tale om. Spørgsmålet er mere bare hvordan man definere et fysisk objekt i sit programmeringssprog og hvordan man kan bruge det. For at tage udgangspunk i noget lad os prøve, at lave objekter af mennesker eller med andre ord personer.

Jeg går ud fra du har forstået der er noget som hedder metoder, attributter og klasser hvor det sidste hænger godt sammen med objekter.

Derfor opret en tom klasse og kald den for Person. Skriv følgende ind i klassen

```
public class Person
{
  public Person()
  {
  }
}
```

public Person()

er ikke en metode, men en konstruktør. Konstruktøren bliver kun kaldt en gang og det er når objektet oprettes. Det mellem {} til konstruktøren er helt normalt kode, men typisk anvendes dette område til at initialisere de øverste attributter. Det kommer vi til.

Først lad og definere hvad et menneske/person er. Vi kunne selvfølgelig gå helt ned i detaljer hvor vi nævner hvor mange øjne, arme og ben et menneske har, men vi skal holde det simpelt så vi tager kun de vigtigste egenskaber med.

Dette er min beskrivelse af et menneske/person

En person har

Et navn
En alder
Han/hun køn
Har en højde
Har en vægt

Skal man overføre dette til computersprog må navn være en String, alder en int, han/hun køn en String, højde en int og vægt ligeledes en int.

Indsæt disse i vores Person klasse så resultatet ville ligne dette.

```
public class Person
{

String navn;
int alder;
String koen;
int hoejde;
int vaegt;

  public Person()
  {
  }
}
```

Indtil videre er vores Person klasse ikke specielt brugbar derfor må vi finde ud af hvilke værdier vores attributter skal have når man vil lave et person objekt.
Eftersom alle som bliver født har en vægt, et køn og en højde men ikke nødvendigvis er blevet navngivet endnu vil vi vente med navngivning. Angående alderen sætter vi den automatisk til at være 0 og kan selvfølgelig ændres senere.
For at intitialisere vores attributer skal vi have dem sat ind i vores konstruktør hvilket jeg forventer du godt ved hvordan gøres.
Samlede resultat vil være dette.

```
public class Person
{

String navn;
int alder;
String koen;
int hoejde;
int vaegt;

  public Person(String nytnavn, String nytkoen, int nyhoejde, int nyvaegt)
  {
  navn = nytnavn;
  koen = nytkoen;
  hoejde = nyhoejde;
  vaegt = nyvaegt;
  }
```

```
}
```

husk på at variablerne i vores konstruktør ikke behøver være sigende.. Den fortæller bare hvilke type parametre den forventer og i hvilken rækkefølge.

public Person(String nytnavn, String nytkoen, int nyhoejde, int nyvaegt)

Vores attributter er ikke erklæret private som man normalt ville gøre, men det for at gøre det mere overskueligt jeg med vilje har undladt dette så egentlig kan vi godt tilgå attributterne ret let, men lad os nu forestille os det ikke er muligt og at det derfor er nødvendigt med metodekald. Vi starter med 2 metode kald.. et til at hente navn og et til at hente køn.

Resultatet ville se nogenlunde sådan ud

```
public class Person
{

String navn;
int alder;
String koen;
int hoejde;
int vaegt;

  public Person(String nytnavn, String nytkoen, int nyhoejde, int nyvaegt)
  {
  navn = nytnavn;
  koen = nytkoen;
  hoejde = nyhoejde;
  vaegt = nyvaegt;
  }

  public String hentNavn()
  {
    return navn;
  }

  public String hentKoen()
  {
    return koen;
  }
}
```

lad os da prøve at lave et objekt ud af vores klasse før vi arbejder videre!

Opret en ny klasse... kald den evt for Main. Klassen kunne se sådan ud

```
public class Main
{
  public Main()
  {
  }

  public static void main(String[] args)
```

```
  {
    Main main = new Main();
  }
}
```

Nu gælder det lidt om at holde hovedet koldt og tænke tilbage på vores klasse, vores model, vores beskrivelse af hvad en person "er".

Vi prøver først at oprette en enkelt person. Alt efter tegnene "//" i kode eksemplet herunder er kommentare fra mig til dig og har intet med koden at gøre!

```
public class Main
{
  public Main()
  {

  Person person = new Person("Hans", "Mand", 170, 70); //kig i vores person klasse.. i konstruktøren og lig mærke til rækkefølgen af vores parametre her

  }

  public static void main(String[] args)
  {
    Main main = new Main();
  }
}
```

linjen her skal forstås på følgende måde
Person person = new Person("Hans", "Mand", 170, 70);

Person -> Den klasse vi vil lave et objekt af hvilket i dette tilfælde er den klasse vi selv har lavet:)

person -> lig mærke til p'et er skrevet med småt.. (java skelner mellem store og små bogstaver) person står for reference.. det kan med andre ord forklares som at være en kopi af vores klasse Person. Vi går altså aldrig ind i vores Person klasse og piller ved attributterne der.. men vi laver en kopi af klassen og det er i KOPIEN vi ændre på attributterne.. Vores Person klasse er vores skabelon eller model, som definere hvad et Person objekt er.

new Person("Hans", "Mand", 170, 70); -> new anvendes til at oprette et objekt og efter new har vi hvad der skal oprettes et nyt af lig mærke til vi har fat i konstruktøren som bliver fyldt op med de parametre vi har valgt skal fyldes ud ved dets oprettelse.

Før vi går videre må vi lige sikre os du helt har forstået dette!!

Et andet eksempel du sikkert selv har anvendt tit er

String tekst;

her fortæller vi at vi vil lave et kopir af String klassen og vores kopi skal hedde tekst. Det korrekte begreb er reference, men det lettere at sætte sig ind i når man ser på det som kopirer:)
String klassen er opbygget på samme måde som vores Person klasse.. der er intet hokus pokus over den.

men lad os da gå videre! Vi har nu oprettet en enkelt person.. han hedder Hans og er en mand. Hvis vi vil have fat i Hans ved vi at alle informationer af ham ligger gemt i person.
For at gøre det tydeligt at hans ikke er gemt i vores Person klasse men i vores kopi person opretter vi lige endnu et objekt.. altså en kopir af vores Person klasse.

```
public class Main
{
  public Main()
  {

  Person person = new Person("Hans", "Mand", 170, 70);
  Person person2 = new Person("Tanja", "Kvinde", 166, 49);
  }

  public static void main(String[] args)
  {
    Main main = new Main();
  }
}
```

Nu har vi oprettet endnu et objekt! Denne gang hedder vores kopi person2 og i denne kopi ligger alt information om Tanja gemt.

Det er kun fordi vi laver kopirer at Tanja ikke overskriver Hans'es værdier!
For objektet person og person2 er kopirer af Person klassen og har derfor hver især deres egne

navn = nytnavn;
koen = nytkoen;
hoejde = nyhoejde;
vaegt = nyvaegt;

Vores personer mangler dog stadig en del egenskaber.. de mangler for eksempel at have en alder! så lad os hoppe tilbage til vores Person klasse og sørge for man kan give dem en alder. Til dette skal vi lave en metode.

```
public class Person
{

String navn;
int alder;
String koen;
int hoejde;
int vaegt;

  public Person(String nytnavn, String nytkoen, int nyhoejde, int nyvaegt)
  {
  navn = nytnavn;
  koen = nytkoen;
  hoejde = nyhoejde;
  vaegt = nyvaegt;
  }
```

```
  public String hentNavn()
  {
    return navn;
  }

  public String hentKoen()
  {
    return koen;
  }

  public void nyAlder(int al)
  {
    alder = al;
  }

  public void fodselsdag()
  {
    alder++;
  }
}
```

Du har sikkert lagt mærke til der er kommet 2 metoder på... denne metode

```
  public void fodselsdag()
  {
    alder++;
  }
```

den er meget praktisk hvis en person fylder år.. Istedet for at man skal ind og sætte en ny alder på en person ved at metoden nyAlder(int al) som er lidt besværlig for den kræver man ved hvor gammel personen er og husker at forøge alderen med et år. Det lettere med fodselsdag() som bare går ind og forøger alderen med 1.

Vi tester vores nye metoder.

```
public class Main
{
  public Main()
  {

  Person person = new Person("Hans", "Mand", 170, 70);
  Person person2 = new Person("Tanja", "Kvinde", 166, 49);

  person.nyAlder(24);
  person2.nyAlder(21);

  // Tanja har fødselsdag i dag

  person2.fodselsdag();

  }
```

```
  public static void main(String[] args)
  {
    Main main = new Main();
  }
}
```

Vi giver først begge personer en alder... efterfølgende har person2 fødselsdag så vi gå lige ind og sætter alderen op der.
Prøv selv at lave det sidste med højde og det hvad der ellers er tilbage:)

Det gælder altså om at forestille sig at man har en masse skabeloner til rådighed! Det er op til dig at finde de interessante skabeloner, som du kan lave kopirer af så du kan manipulere med dem.

ArrayList nyliste = new ArrayList();

nyliste er nu en kopir af ArrayList.. du kan slå ArrayList op som også er bare er en klasse og se hvad den kan.. hvilke metoder den har osv.

JFrame frame;

En reference til et JFrame objekt.. igen JFrame er også bare en skabelon! Vores kopi af denne skabelon er frame..

Håber det gav lidt mening alt det her:) Hvis du bare blev endnu mere tosset oven i hovedet så fratager jeg mig alt ansvar;o) ej.. i/du er velkommen til at spørge løs...
Til de mere garvede brugere som evt har læst dette.. husk på min mening med det her er bare at forklare folk hvad objekter er så det bare står lidt klarere for dem:)
Jeg VED at nybegyndere ikke fatter hat af hvad objekter er og det er lidt af et handikap!

Ps. Hvis du kunne lide og fik noget ud af denne forklaring på objekter og kunne tænke dig en lignende forklaring på noget andet smid da en kommentar så skal jeg kigge på det.

Kommentar af milo d. 28. Jan 2005 | 1

Lækkert

Kommentar af tbirch d. 09. Jun 2007 | 2

Meget informativt og klart.
Vh Thomas Birch

Kommentar af riversen d. 23. Jan 2005 | 3

Fint og udemærket initiativ, som jeg dog synes har et par fejl og mangler, som jeg synes man bør nævne for at sikre forståelsen. 1. nyliste og frame er ikke kopier men objekter der bygger på klasserne efter "new". 2. Eller rettere: nyliste og frame er jo faktisk referencer til objekter af de 2 typer. Denne detalje har jo en del at sige ved sammenligning af objekter og er en del af den rigtige forståelse.

Kommentar af konder d. 27. Jan 2005 | 4

Du skal ha mange tak for forklaringen.

Kommentar af avj d. 16. Feb 2005 | 5

Kommentar af cronck d. 24. Jan 2005 | 6

Kommentar af webcreator d. 23. Jan 2005 | 7

En god forklaring på hvad objekter er og hvordan de bruges

Kommentar af gladmhensk d. 30. Jan 2005 | 8

Smukt forklaret, og så er det rart du holder det på et simpelt plan. Bedste forklaring af objekter jeg nogensinde har læst (har læst mange ;))

Kommentar af htmlkongen d. 31. Jan 2005 | 9

Tak for ny viden :) Til andre ---> Fint sted at starte - Jeg fik noget ud af det trods jeg vidste intet om det i forvejen... :) /Htmlkongen

Kommentar af visualdeveloper d. 01. Nov 2005 | 10

noget om nedarving ville være godt synes jeg ;)

Kommentar af andr3as d. 23. Jan 2005 | 11

:) meget godt, beskrivelserne og alt

Kommentar af alister_crowley d. 23. Jan 2005 | 12

Og som nybegynder vil jeg sige at den giver en rigtig god gennemgang, specielt mht. de advancerede ord (objeter,konstruktor,osv) som bliver brugt i svar forslag når man stiller spørgsmål. Et absolut must for enhver begynder. Men der er også lidt forslag til en forbedring, da det ville være relavant at indrage interfaces i denne artikel vil jeg mene.

Kommentar af schwarz84 d. 22. Jun 2005 | 13

nyliste er jo netop _ikke_ en klasse, men et objekt (en instans af klassen ArrayList). Når din artikel hedder "Hvad er Ojekter" så skal den være lidt skarpere her.
Det hjælper nok ikke en begynders forståelse at kalde klassen Person og objektet person. Hvorfor ikke kalde objektet "peter" eller "poul" eller noget andet så det er tydeligt at der er tale om en konkret person (objekt) og ikke begrebet person (klasse). Du gør det samme senere ved at du opretter et object main ud fra klassen Main i metoden main.
Du bytter også rundt på variabler og klasser. "String tekst;" giver dig en variabel af typen String, som ikke er initialiseret (det peger altså _ikke_ på et objekt). Der er altså endnu ikke oprettet en reference til et objekt og det er direkte forkert at tale om at en uinitialiseret variabel er en kopi af klassen.
Generelt er det instanser, du lavet, når du bruger "new", det er ikke kopier.
Din kode ville være mere læsbar hvis den var indenteret.