Enkel lesing og skriving i Java

Dag Langmyhr
dag@ifi.uio.no

26. januar 2023

Innhold

Eksempler

1 Skriving

Det er enkelt å skrive ut i Java: Man får gjort det meste av det man ønsker av utskrift i terminalvinduet med metodene

```
System.out.println(...);
System.out.println();
```

som skriver ut en linje med parameteren på skjermen, slik det er vist i demo 1. Parameteren vil oftest være en String, men kan også være en int, en double eller en annen verdi. Hvis man dropper parameteren, får man en blank linje.

```
1 class Hallo {
2     public static void main(String[] args) {
3         System.out.println("Hallo, alle sammen!");
4     }
5 }
```

```
Hallo, alle sammen!
```

Demo 1: Enkel utskrift til skjermen

Noen ganger ønsker man kanskje ikke å avslutte linjen med en gang, og da kan man benytte

```
System.out.print(...);
```

i stedet. Da kan man fortsette på samme linje til man omsider avslutter den med å kalle på System.out.println(); se eksempelet i demo 2.

```java
class Hallo2 {
    public static void main(String[] args) {
        System.out.print("Hallo");
        System.out.print(',');
        System.out.print(' ');
        String navn = "Anne";
        System.out.print(navn);
        System.out.print('!');
        System.out.println();
    }
}
```

```
Hallo, Anne!
```

Demo 2: Oppdelt utskrift til skjermen

Hvis det er flere verdier som skal skrives ut på samme linje, er det lurt å slå dem sammen til én String før man kaller på System.out.println. Dette gjøres med operatoren **+** som automatisk vil konvertere ulike datatyper til String og så skjøte dem sammen.[1] Demo 3 viser et eksempel på det.

```java
class Areal {
    public static void main(String[] args) {
        double r = 2.5;
        double areal = Math.PI * r * r;  // Formel for areal av sirkel: πr²
        System.out.println("En sirkel med radius " + r +
                           " har areal " + areal + '.');
    }
}
```

```
En sirkel med radius 2.5 har areal 19.634954084936208.
```

Demo 3: Sammenskjøting av flere verdier før utskrift

[1] Operatoren **+** kan brukes til flere forskjellige ting:

- Når + står mellom to tallverdier, vil den fungere som *pluss*, dvs at den vil legge sammen de to tallene:

  $$1 + 2 \Rightarrow 3$$

- Når + står mellom to String-er, vil den skjøte sammen de to tekstene:

  $$\text{"Hokus"} + \text{"Pokus"} \Rightarrow \text{"HokusPokus"}$$

- Når + står mellom en String og en annen verdi, vil den konvertere verdien til en String før den skjøter de to sammen:

  $$\text{"Tallet er "} + 42 \Rightarrow \text{"Tallet er 42"}$$

1.1 Skriving til fil

Det er nesten like enkelt å skrive noe til en fil som på skjermen; vi må bare åpne filen først. Til det trenger vi klassen PrintWriter fra Java-biblioteket:

```
import java.io.PrintWriter;
```

Så kan vi deklarere filen og opprette den, men dette skjer på en litt komplisert måte fordi det kan hende vi ikke har lov til å lage filen:

```
PrintWriter f = null;
try {
  f = new PrintWriter("...");
} catch (Exception e) {
  System.out.println("...feilmelding...");
  System.exit(1);
}
```

Parameteren til PrintWriter er navnet på filen vi ønsker å lage. Om dette ikke er mulig, må vi skrive ut en passende feilmelding og avslutte programmet.

Når filen er ferdig skrevet, må vi lukke den:

```
f.close();
```

Eksempel Demo 4 viser et program som oppretter filen tall-til-10.txt og skriver tallene 1, 2, . . . , 10; resultatfilen ser slik ut:

tall-til-10.txt

```
1 2 3 4 5 6 7 8 9 10
```

```
1  import java.io.PrintWriter;
2
3  class LagFil {
4      public static void main(String[] args) {
5          PrintWriter f = null;
6          try {
7              f = new PrintWriter("tall-til-10.txt");
8          } catch (Exception e) {
9              System.out.println("Kan ikke lage filen tall-til-10.txt.");
10             System.exit(1);
11         }
12
13         // Skriv tallene fra 1 til 10:
14         for (int i = 1;  i <= 10;  i++) {
15             f.print(" " + i);
16         }
17         f.println();  f.close();
18     }
19 }
```

Demo 4: Skriving til fil

2 Lesing

Når man ønsker å lese data brukeren skriver inn på tastaturet, er det enklest å bruke Scanner-klassen i Java. Den må importeres fra Java-biblioteket:

```
import java.util.Scanner;
```

Så må vi opprette et Scanner-objekt som kan lese fra tastaturet:

```
Scanner tastatur = new Scanner(System.in);
```

I Scanner-klassen finnes metoder for å lese ulike typer verdier, for eksempel heltall (dvs en int), flyt-tall (dvs en double) eller ord[2] (dvs en String). De viktigste metodene er vist i tabell 1.

nextInt()	Les et heltall (f eks –17)
nextDouble()	Les et flyt-tall (f eks 3.14)
next()	Les et ord (f eks Donald)
nextLine()	Les resten av linjen

Tabell 1: Noen lesemetoder i Scanner

Eksempel Demo 5 viser et program som ber brukeren om fire heltall og bruker nextInt() til å lese dem. Til slutt skriver programmet ut summen av de fire tallene.

```
1  import java.util.Scanner;
2
3  class LeggSammenTall {
4      public static void main(String[] args) {
5          Scanner tastatur = new Scanner(System.in);
6          int sum = 0;
7
8          for (int i = 1;  i <= 4;  i++) {
9              System.out.print("Skriv tall nr " + i + ": ");
10             int v = tastatur.nextInt();
11             sum += v;
12         }
13         System.out.println();  // Skriv en blank linje
14         System.out.println("Summen er " + sum);
15     }
16 }
```

```
$ java LeggSammenTall
Skriv tall nr 1: 4
Skriv tall nr 2: 0
Skriv tall nr 3: -2
Skriv tall nr 4: 12

Summen er 14
```

Demo 5: Innlesing av heltall

[2] Et *ord* i denne sammenhengen betyr en samling ikke-blanke tegn; her er fem eksempler:

A Irene 455 x2yz98-q ++->#

2.1 Lesing fra fil

Det er like enkelt å lese fra en fil som fra tastaturet. Til det trenger vi klassen File i tillegg til Scanner:

```
import java.io.File;
import java.util.Scanner;
```

Vi må åpne filen før vi kan lese fra den. Dette skjer på omtrent samme måten som vist for skriving til fil, for det er ikke sikkert at filen vi oppgir finnes, eller at vi har lov til å lese den.

```
Scanner fil = null;
try {
  fil = new Scanner(new File("..."));
} catch (Exception e) {
  System.out.println("...feilmelding...");
  System.exit(1);
}
```

Selve lesingen skjer med nextInt() og de andre next...-metodene i Scanner (vist i tabell 1 på forrige side), men i tillegg finnes noen metoder som kan fortelle om det finnes flere verdier av den typen vi ønsker; disse er vist i tabell 2.

hasNextInt()	Er neste ord i filen et lovlig heltall?
hasNextDouble()	Er neste ord i filen et lovlig flyt-tall?
hasNext()	Finnes det flere ord i filen?
hasNextLine()	Finnes det flere linjer i filen?

Tabell 2: Noen sjekkemetoder i Scanner

Eksempel Demo 6 på neste side gjør tre ting som alle viser ulike former for lesing og skriving:

1. Programmet ber brukeren om navnet på en fil med navn og alder på diverse personer; formatet på filen er slik det er vist i eksempelfilen alder.txt:

   alder.txt
   ```
   62 Anne Berit
   27 Per Oddvar
    4 Inger
   50 Elisabeth
   ```

2. Programmet vil så lese alle linjene med navn[3] og alder fra filen og merke seg den eldste.

3. Programmet skriver ut navn og alder på den eldste.

[3] Legg merke til at vi bruker nextLine() til å lese navnet, og ikke next(). Det er fordi et navn kan bestå av flere ord.

```java
import java.io.File;
import java.util.Scanner;

class FinnEldst {
    public static void main(String[] args) {
        // Be brukeren om navn på fil med navn og alder:
        Scanner tastatur = new Scanner(System.in);
        System.out.print("Hva heter filen med navn og alder? ");
        String filNavn = tastatur.next();

        // Åpne datafilen:
        Scanner fil = null;
        try {
            fil = new Scanner(new File(filNavn));
        } catch (Exception e) {
            System.out.println("Kan ikke lese " + filNavn + "!");
            System.exit(1);
        }

        // Les datafilen og ta vare på navn og alder til den eldste hittil:
        String eldst = "?";
        int hoyestAlder = 0;
        while (fil.hasNextInt()) {
            int alder = fil.nextInt();
            String navn = fil.nextLine().trim();
            if (alder >= hoyestAlder) {
                eldst = navn;  hoyestAlder = alder;
            }
        }
        fil.close();

        // Skriv ut svaret:
        System.out.println(eldst + " er eldst med " + hoyestAlder + " år.");
    }
}
```

```
$ java FinnEldst
Hva heter filen med navn og høyde? alder.txt
Anne Berit er eldst med 62 år.
```

Demo 6: Lesing fra fil