Java for deg som kan Python

del 1

Dag Langmyhr
(dag@ifi.uio.no)

Hvorfor lære et nytt programmeringsspråk

Selv om nesten alle problemer lar seg kode i nesten alle programmeringsspråk, er det noen språk som egner seg mye bedre enn andre.

- Programmere en «dings» (f eks en værsensor; «IoT») med lite minne og batteri — Velegnet: **C**
- Hente informasjon fra en database — Velegnet: **SQL**
- Introduksjonskurs i programmering — Velegnet: **Python**
- Skrive et stort og komplisert driftssystem med høye krav til sikkerhet — Velegnet: **Java**

Det er også flere grunner til at dere skal lære nye programmeringsspråk:

- Dere skal gjennom studiet og senere i arbeidslivet måtte løse mange ulike oppgaver.
- Arbeidsgiver/prosjektleder har ofte krav til hvilket språk man skal bruke.
- Programmere er forskjellige – de foretrekker ulike språk.

Men, heldigvis:

- Etter hvert som man lærer flere programmeringsspråk, blir det enklere å tilegne seg nye.
  - Man får en større forståelse for hva et programmeringsspråk er og hva det ikke er.
  - Man kjenner igjen språkmekanismer man har møtt før.

Hvordan best lære et nytt språk?

Dette avhenger av hvor erfaren man er, men jeg vil anbefale for dere:

1. Få en kort oversikt over de viktigste egenskapene ved språket (for eksempel Wikipedia eller forelesningene de første par ukene).
2. Prøv å skrive noen veldig enkle programmer i det nye språket (for eksempel oppgaver fra IN1000).
3. Les en litt lengre bok eller artikkel om språket for å oppdage de ulike finessene man møter der (for eksempel læreboken Big Java).
4. Hvis du lurer på «Er dette lov?» eller «Hva skjer om jeg skriver dette?», skriv små testprogrammer.
5. Når du møter problemer du ikke finner ut av, sjekk språkets dokumentasjon først og deretter søk på nettet.

Et svært enkelt program

positiv.py

```
v = 42
if v < 0:
    print(v, "er negativ")
elif v == 0:
    print(v, "er null")
else:
    print(v, "er positiv")
```

Positiv.java

```
class Positiv {
    public static void main (String[] args) {
        int v = 42;
        if (v < 0) {
            System.out.println(v + " er negativ");
        } else if (v == 0) {
            System.out.println(v + " er null");
        } else {
            System.out.println(v + " er positiv");
        }
    }
}
```

I Java ligger *alt* i klasser

```
class Positiv {
    public static void main (String[] args) {
        int v = 42;
        if (v < 0) {
            System.out.println(v + " er negativ");
        } else if (v == 0) {
            System.out.println(v + " er null");
        } else {
            System.out.println(v + " er positiv");
        }
    }
}
```

Krøllparenteser angir innholdet av klassen

```
class Positiv {
    public static void main (String[] args) {
        int v = 42;
        if (v < 0) {
            System.out.println(v + " er negativ");
        } else if (v == 0) {
            System.out.println(v + " er null");
        } else {
            System.out.println(v + " er positiv");
        }
    }
}
```

Linjeskift og innrykk betyr *ingenting* i Java

```
class Positiv {
    public static void main (String[] args) {
        int v = 42;
        if (v < 0) {
            System.out.println(v + " er negativ");
        } else if (v == 0) {
            System.out.println(v + " er null");
        } else {
            System.out.println(v + " er positiv");
        }
    }
}
```

er (sett fra Javas side) like bra som

```
class Positiv {
    public static void main (String[] args) {
        int v = 42;
        if (v < 0) {System.out.println(v + " er negativ");}
        else if (v == 0) {System.out.println(v + " er null");}
        else {System.out.println(v + " er positiv");
}}}
```

Hovedprogrammet i Java er metoden main

```
class Positiv {
    public static void main (String[] args) {
        int v = 42;
        if (v < 0) {
            System.out.println(v + " er negativ");
        } else if (v == 0) {
            System.out.println(v + " er null");
        } else {
            System.out.println(v + " er positiv");
        }
    }
}
```

Også her brukes krøllparenteser til å angi innholdet.

Alle setninger (som ikke slutter med en }) slutter med ;

```
class Positiv {
    public static void main (String[] args) {
        int v = 42;
        if (v < 0) {
            System.out.println(v + " er negativ");
        } else if (v == 0) {
            System.out.println(v + " er null");
        } else {
            System.out.println(v + " er positiv");
        }
    }
}
```

Deklarasjon av variabler

En fundamental forskjell på Python og Java er at i Java må alle variabler deklareres med en angitt type.

```
class Positiv {
    public static void main (String[] args) {
        int v = 42;
        if (v < 0) {
            System.out.println(v + " er negativ");
        } else if (v == 0) {
            System.out.println(v + " er null");
        } else {
            System.out.println(v + " er positiv");
        }
    }
}
```

I Python benyttes **dynamisk typing**. Da er typeinformasjonen knyttet til *verdiene* og sjekkes først under kjøring.

I Java brukes **statisk typing**. Da har alle variabler en angitt type og kan bare få verdier av den typen; dette sjekkes av kompilatoren.

Python

```
x = 42

x = "ukjent"
```

Java

```
        int x = 42;

        x = "ukjent";
```

```
Dynamisk.java:5: error: incompatible types:
        String cannot be converted to int
        x = "ukjent";
            ^
1 error
```

Hva er fordelene med de to måtene å programmere på?

Dynamisk typing (Python)

- Mindre å skrive
- Mer fleksibelt (som listen `[ 2, "?", True ]`)

Statisk typing

- Programmereren kan kode inn hva han/hun tenker.
- Kompilatoren vil oppdage mange flere feil.
- Kjøringen vil oftest gå raskere.

Kjøretid

Noen ganger er kjøretid viktig – andre ganger ikke.

Python

```
def sorter():
    ok = False
    while not ok:
        ok = True
        for i in range(1,n):
            if a[i] > a[i+1]:
                (a[i],a[i+1]) = (a[i+1],a[i])
                ok = False
```

56,0 s

Java

```
static void sorter() {
    boolean ok = false;
    while (! ok) {
        ok = true;
        for (int i = 1;  i < n;  ++i) {
            if (a[i] > a[i+1]) {
                int temp = a[i];
                a[i] = a[i+1];
                a[i+1] = temp;
                ok = false;
            }
        }
    }
}
```

2,0 s

Typer i Java

De viktigste datatypene i Java er disse:

- `int` for heltall (0, 1, 2 til rundt 1 milliard og -1, -2 osv))
- `double` for flyt-tall (`2.5`, `3.14159265`, `...`)
- `boolean` for logiske verdier (**`true`** og **`false`**)
- `char` for enkelttegn (som `'$'`)

Dessuten har vi `String` for å lagre tekster, men det er en *klasse* og ikke en *verditype*.

If- og While-setninger

Disse ligner veldig på de tilsvarende i Python:

```
class Positiv {
    public static void main (String[] args) {
        int v = 42;
        if (v < 0) {
            System.out.println(v + " er negativ");
        } else if (v == 0) {
            System.out.println(v + " er null");
        } else {
            System.out.println(v + " er positiv");
        }
    }
}
```

Den største forskjellen er at testen *må* være en verdi av typen **`boolean`**.

Python

```
a = 4
s = ""

if (a):
    print("Ja 1")
if (s):
    print("Ja 2")
```

Java

```
int a = 4;
String s = "";

if (a) {
    System.out.println("Ja 1");
}
if (s) {
    System.out.println("Ja 2");
}
```

```
IfFeil.java:6: error: incompatible types: int cannot be converted to boolean
        if (a) {
            ^
IfFeil.java:9: error: incompatible types: String cannot be converted to boolean
        if (s) {
            ^
2 errors
```

Slik må det skrives i Java:

Python

```
a = 4
s = ""

if (a):
    print("Ja 1")
if (s):
    print("Ja 2")
```

Java

```
int a = 4;
String s = "";

if (a != 0) {
    System.out.println("Ja 1");
}
if (s.length() > 0) {
    System.out.println("Ja 2");
}
```

Utskrift

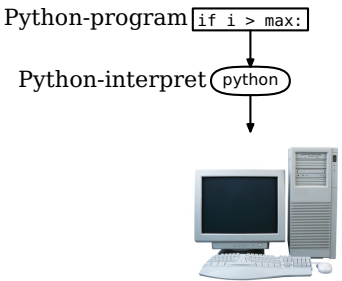
I Java brukes metoden `System.out.println(...)` til å skrive ut på skjermen.

```
class Positiv {
    public static void main (String[] args) {
        int v = 42;
        if (v < 0) {
            System.out.println(v + " er negativ");
        } else if (v == 0) {
            System.out.println(v + " er null");
        } else {
            System.out.println(v + " er positiv");
        }
    }
}
```

Den har bare én parameter, men vi kan skjøte verdier av ulike typer med +.

Å kjøre et Python-program

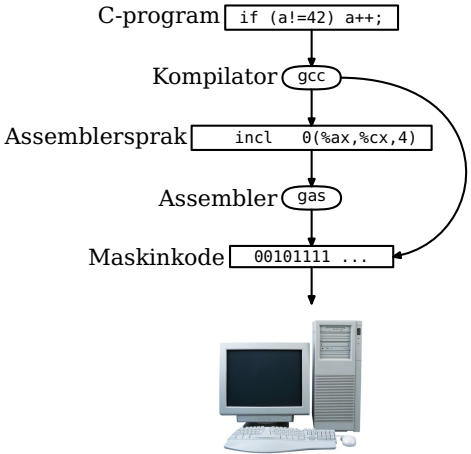
En Python-interpret leser kildekoden og utfører den.



Kompilering

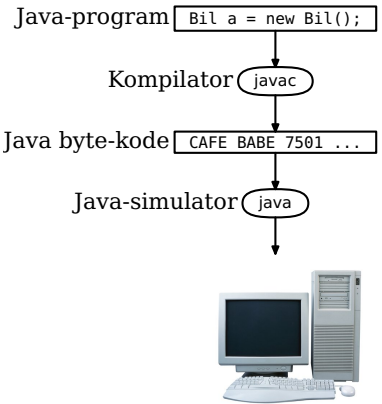
En **kompilator** oversetter et program til en annen kode, oftest **maskinkode**.

Ekempler: C, C++



Kompilering og kjøring av Java-programmer

Java benytter en mellomting: Java-koden kompileres til `.class`-filer som så kan interpreteres.



Oppsummering

- En god del skrives omtrent likt i Python og Java:
  - uttrykk
  - tilordningssetninger
- Mye er egentlig ganske likt men skrives på en annen måte:
  - innrykk og linjeskift / krøllparenteser
  - if- og while-tester
  - utskrift (`print(...)` / `System.out.println(...)`)
- Noe er fundamentalt forskjellig:
  - alt må ligge i klasser i Java
  - statisk og dynamisk typing