IN1010

Fra Python til Java

En introduksjon til programmeringsspråkenes verden

Dag Langmyhr
dag@ifi.uio.no

Oversikt

- Introduksjon
- Python ⇒ Java
  - Noe er likt
  - Noe bare ser anderledes ut
  - Noe er helt forskjellig
- Et par eksempler
- Klasser (i morgen ved *Stein Gjessing*)

10 000-er av programmeringsspråk

Hvorfor finnes det så mange?

- Det finnes mange ulike behov; f eks

  Python bash Excel SQL

- Nye behov oppstår
- I dag kan vi lage bedre programmeringsspråk enn for 50 år siden

Ikke veldig vanskelig

Om dere programmerer en del og tar emnene

- INF2100 Prosjektoppgave i programmering
- INF3110 Programmeringsspråk
- INF5110 Kompilatorteknikk

kan dere lage deres eget programmeringsspråk.

Mitt forsøk: **Knod**

```
® Syntax_expression: {|
   ® parts: §[1];

   ® width: {|
      ® w: 0.max(| @ #.parts: £.width() |);
      ¿ #.parts.n().>(1): w.©+(¶.h_sep.×(2)) ?;
      ¥ w;
   |};

   ® height: {|
      ¥ #.parts[1].height()
   |};

   ® depth: {|
      ¥ ¿ #.parts.n().=(1):
            #.parts[1].depth()
        | True:
            #.parts[1].depth().
               +(| @ #.parts[2..]: £.height().+(£.depth()) |).
               +(#.parts.n().--().×(2,¶.v_sep))
        ?
   |};
```

Hvorfor finnes Python?

Python ble opprinnelig laget i 1989 som et hobbyprosjekt av *Guido van Rossum* med denne filosofien:

- Beautiful is better than ugly
- Explicit is better than implicit
- Simple is better than complex
- Complex is better than complicated
- Readability counts

Hvorfor ble Java laget?

Java ble laget i 1996 av *James Gosling* og andre i SUN Microsystems (nå kjøpt av Oracle) med følgende mål:

- It must be “simple, object-oriented, and familiar”.
- It must be “robust and secure”.
- It must be “architecture-neutral and portable”.
- It must execute with “high performance”.
- It must be “interpreted, threaded, and dynamic”.

Min personlige mening

Python er best til

- nybegynneropplæring
- korte, enkle programmer jeg trenger *nå*
- kopling mot nyttige programpakker i kjemi, fysikk, matematikk, . . .

Java er best til

- objektorientert programmering
- store programmer
- trygge programmer
- raske programmer

Noe er helt likt

Uttrykk

Uttrykk er svært like i de to språkene:

Python	Java
`2 + v*(t+1)` `2 * 3.1416 * r`	`2 + v*(t+1)` `2 * 3.1416 * r`

Tilordning

Tilordning (dvs gi en variabel en verdi) er også likt:

Python	Java
`n = 2 + v*(t+1)` `omkr = 2 * 3.1416 * r`	`n = 2 + v*(t+1)` `omkr = 2 * 3.1416 * r`

Noe ser anderledes ut men er egentlig likt

- I Java brukes *krøllparenteser* (**{}**) i stedet for innrykk.
- Alle Java-setninger avsluttes med *semikolon* (**;**).

if-tester

Python	**Java**

`if v < 0:`
`  v = -v`
	`if (v < 0) {`
	`  v = -v;`
	`}`

Linjeskift og innrykk betyr altså ikke noe i et Java-program:

```
if (v < 0) { v = -v; }
```

```
if (
   v<  0) { v = -
v;}
```

If-tester kan ha flere alternativer:

Python

```
if  x < 0:
    f = -1
elif x == 0:
    f = 0
else:
    f = 1
```

Java

```
if (x < 0) {
    f = -1;
} else if (x == 0) {
    f = 0;
} else {
    f = 1;
}
```

Legg merke til:

- Java har parenteser rundt if-testen (i stedet for kolon etter)
- Java har ingen **elif** men i stedet er det mulig å skrive **else if**

Krøllparenteser brukes alle steder der Python krever innrykk, også i while- og for-løkker:

Python

```
while x <= 255:
    total = total + x
    x = 2 * x

for i in range(1,10):
    sum = sum + i
```

Java

```
while (x <= 255) {
    total = total + x;
    x = 2 * x;
}

for (int i = 1;  i < 10;  i++) {
    sum = sum + i;
}
```

Legg også merke til:

- For-løkkene skrives ganske anderledes i Java enn i Python

Utskrift

I java finnes to metoder for utskrift:

System.out.print skriver ut parameteren; hvis det er flere elementer, må de skjøtes sammen med +.

System.out.println gjør det samme, men setter på et linjeskift.

Python

```
r = 2.0
v = 3.1416 * r * r
print("r = "+str(r),end="")
print(" gir areal "+str(v))
```

Java

```
double r = 2.0;
double v = 3.1416 * r * r;
System.out.print("r = " + r);
System.out.println(" gir areal " + v);
```

Innpakning

I Java må *alt* ligge i klasser

```
class MinKlasse {
        :
}
```

Metoden **main** starter det hele, og den må alltid deklareres som

public static void main(String[] arg)

```
class Hei {
    public static void main(String[] arg) {
        System.out.println("Hei!");
    }
}
```

Kjøring

I Java brukes to programmer:

javac oversetter **.java**-filen til én eller flere **.class**-filer.

java utfører .class-filen(e)

```
$ javac Hei.java
$ java Hei
Hei!
$
```

```
class Hei {
    public static void main(String[] arg) {
        System.out.println("Hei!");
    }
}
```

Angivelse av variabeltype

- Python har **dynamisk typing** der *verdiene* har en gitt type; derfor kan variblene tilordnes verdier av ulike typer.
- Java har **statisk typing** der både *variablene* og *verdiene* har type; variabler kan bare få verdier av den riktige typen.

Derfor: I Java må *alle* variabler deklareres.

Python

```
i = 5
v = 2.5
v = v + i
s = "Svaret er "
print(s+str(v))
```

Java

```
int i = 5;
double v = 2.5;
v = v + i;
String s = "Svaret er";
System.out.println(s + " " + v);
```

Javas typer

De viktigste typene i Java er:

int	heltall	(dvs 0, 1, 2, -3, ...)
double	flyt-tall	(dvs 0.0, 1.5, -22.7, 3.14, ...)
boolean	logisk verdi	(dvs **true** og **false**)
char	enkelttegn	(dvs **'a'**, **'b'**, **'?'**, **'5'**, ...)

I tillegg finnes klassen

String	tekst	(dvs **"x"**, **"abcd"**, **"Hei!"**, ...)

Finn medianen (dvs det midterste) av tre tall

```
def finnMedian(a):
  return sorted(a)[1]

data = [0] * 3
leser = open("tall.data")
for i in range(3):
    data[i] = int(leser.readline())
print("Medianen er " + str(finnMedian(data)))
```

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

class Median {
    private static int finnMedian(int[] a) {
        Arrays.sort(a);
        return a[1];
    }

    public static void main(String[] arg) throws Exception {
        int[] data = new int[3];
        Scanner leser = new Scanner(new File("tall.data"));

        for (int i = 0;  i < 3;  i++) {
            data[i] = Integer.parseInt(leser.nextLine());
        }
        System.out.println("Medianen er " + finnMedian(data));
    }
}
```

Konklusjon

- Det finnes ikke noe beste språk for alle formål!
- Det er en styrke å kunne flere språk
- Alle informatikere må regne med å lære flere språk
- Det er egentlig bare én måte å lære et nytt programmeringsspråk:
  1. les dokumentasjonen
  2. prøv selv med mange eksempler
  3. om ikke alt fungerer som forventet, gjenta.