PRG1000B Grunnleggende programmering

Leksjon 4

Metoder

Del 1

Roy M. Istad, 2017

Program, klasse og metode

Kode.java

```
import static java. ... // ulike klasser
public class Kode {

    main-metode

}
```

Våre program så langt:

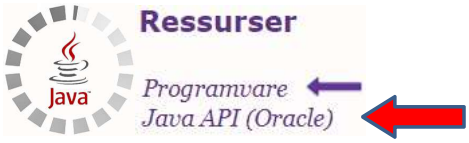
- Én fil
- Én klasse (`import`?)
- Én main-metode

Større program: Flere filer, flere klasser og flere metoder.

Hvorfor? Kode med bedre oversikt, gir gjenbruk og arbeidsdeling

Oppdeling av program i metoder

```
public class A {
    metode 1
    metode 2
    metode 3
}
```



Store program "krever" oppdeling i mindre biter.

- Metode – navnsatt enhet som inneholder én eller flere setninger.
- Metode – et «delprogram» som utfører en avgrenset oppgave.
- Java-biblioteket (**API**) har (veldig) mange ferdiglagede metoder

Eksempler fra API:

- `parseInt()`
- `round()`
- `min(), max()`
- `showMessageDialog()`

Eksempel: MetodeTest.java

```
import static java.lang.System.*;

public class MetodeTest {
  public static void main(String[] args) {
    skrivStjerner();
    out.println("Her er en setning.");
    skrivStjerner();
    out.println("Ferdig !");
  }
} private static void skrivStjerner() {
    out.println("********************");
  }
}
```

Skopet
(mellom krøllparenteser)

Programmet gir denne utskriften:

```
********************
Her er en setning.
********************
Ferdig!
```

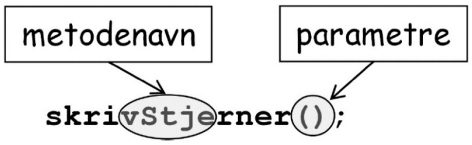
NB! Metoden *skrivStjerner* finnes ikke i noe bibliotek, vi må lage den selv . . .

`void`?
"Tom", metoden sender ikke ut noen verdi (i en gitt datatype)

`private`?
Hjelpemetode, deklarert inne i klassen som skal bruke den.

Metodekall og parameterliste

- Setningene i en metode blir utført ved *kall* på metoden
- Et *metodekall* består i det enkleste tilfellet av navnet på metoden, etterfulgt av **( )** – altså en *tom parameterliste*.
- Kall på en `void`-metode står som en selvstendig setning i programkoden.
- Metode med *returverdi* sender fra seg en verdi, og må som andre verdier stå i en tilordning, som del av et utrykk eller parameter i et nytt metodekall.



Eksempel:

```
int tall = min(3, 8);
double meterLengde = round(yard*CM_I_YARD)/100;
int allerStørst = max( max(2,tall), max(3,8) );
```


Formelle og aktuelle parametre

```
public static void main(String[] args) {
  skrivStjerner(2);
  skrivStjerner(4);
  skrivStjerner(8);
}
```

aktuell parameter

Metodekall:

```
skrivStjerner( 2 );
               4   8
```

```
private static void skrivStjerner(int antall) {
  for (int i=1; i<=antall; i++)
    out.print("*");
  out.println();
}
```

formell parameter

Konsoll

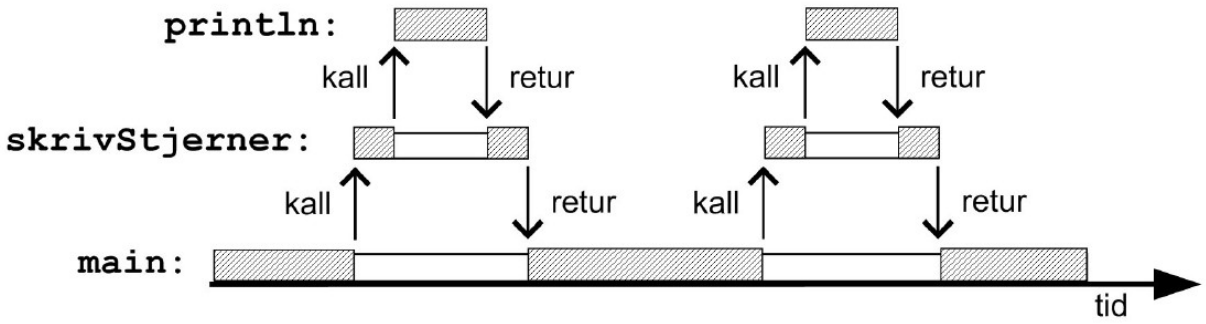
```
**********
**  Lengde: 2
****  Lengde: 4
********  Lengde: 8
```

Altså:

skrivStjerner-lengde:2
skrivStjerner-lengde:4
skrivStjerner-lengde:8

Utførelse av metodekall

`skrivStjerner` blir kalt fra `main`-metoden, og `println` fra `skrivStjerner`:



Ved metodekall «hopper» programutførelsen ut av en metode og inn i en annen metode.
Den hopper tilbake igjen når alle setningene i den aktuelle metoden er utført.

Parameterliste

Metodekall: `skrivTegn('+', 7);` Metodekall: `skrivTegn('*', 7);`

```
private static void skrivTegn(char t, int antall) {
  for (int i=1; i<=antall; i++)
    out.print(t);
  out.println();
}
```

Konsoll

```
+++++++
```

Samme metode, men andre aktuelle parametre i kallet:

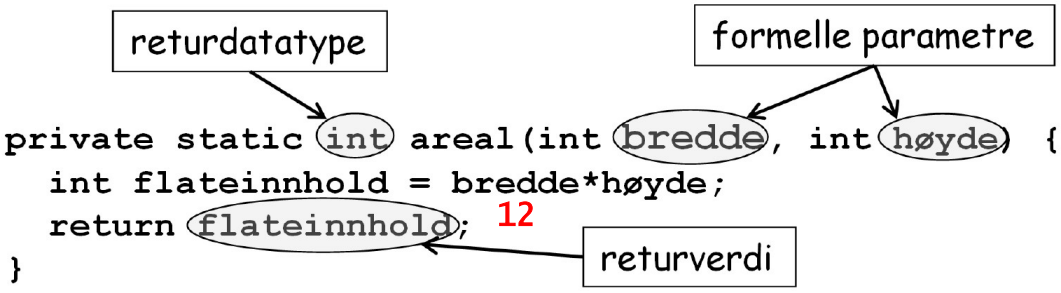
```
skrivTegn('P', 12);
```

Konsoll

```
PPPPPPPPPPPP
```

Metode med returverdi

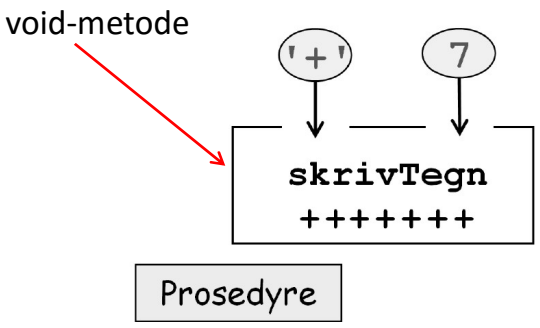
```
public static void main(String[] args) {
  int svar = areal(3, 4);   // Som om det sto:  int svar = 12;
  out.println("Rektangelareal: " + svar);
}
```

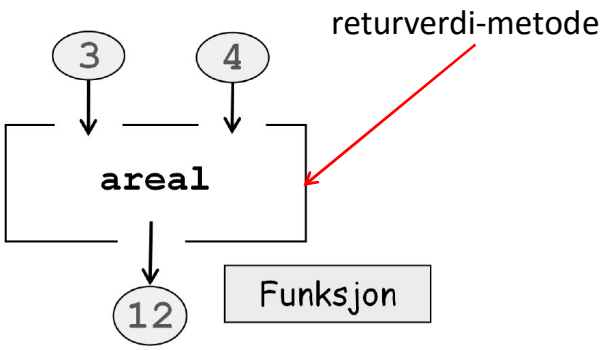


Metoder med *returverdi* kan brukes som uttrykk. Her kan metodekallet like gjerne stå i utskriften:

```
out.println("Rektangelareal: " + areal(3, 4) );
```


«Prosedyre» kontra «funksjon»





Metoden står som egen setning

Slå plena! Metoden utfører et oppdrag, men gir ingen kvittering tilbake

Metoden står som del av et utrykk

Hva er klokka? Metoden "beregner" ett resultat (en verdi) som kvitteres til oppdragsgiver

Sammensatte metodekall

```
int allerMinst = minst(2, minst(3, 4));
```

2

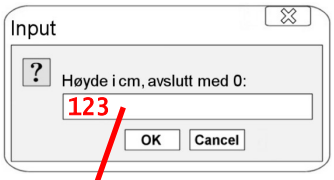
3 4

2 minst 3

minst

2

Oppgave:
Skriv en metode som kan finne det minste av tre heltall.



Innlesing av heltall via dialogvindu:

123

```
int høyde = parseInt( showInputDialog(ledetekst) );
```


Uttrykk som parametre

```
private static void skrivTegn(char t, int antall) {
  for (int i=1; i<=antall; i++)
    out.print(t);
}
```

Konsoll

```
+
```

Mulige metodekall nå:

```
  :
for (int i=1; i<=5; i++) {
  skrivTegn(' ', 5-i );
  skrivTegn('+', 2*i-1 );
  out.println();
}
  :
```

uttrykk

+	4
t	antall

i

Konsoll

```
1 XXXXX
2  XXXXX
3   XXXXX
str=4    XXXXX
```

```
int str = // lest linjetall
for (int i=1; i<=str; i++) {
  skrivTegn(' ', i-1 );
  skrivTegn('X', str+1 );
  out.println();
}
```