Repetisjonstime

GRUNNLEGGENDE JAVA

11. mars 2025

Klasse og objekt

- Når vi programmerer så skriver vi en forenklet modell av virkeligheten.
  - Har lyst til å **løse et problem** elller **oppfylle et behov**
  - Bestemmer selv hva **slags egenskaper** denne modellen skal ha.

- **Klasse** definerer hvilken egenskaper, struktur og hvilke handlinger objekter av klassen skal ha.
- **Objekt** er en instans av en klasse, som inneholder data, variabler, metoder.

Java syntaks

- For å kunne kjøre kode må koden ligge i en klasse
- Koden begynner å kjøre i main(), deretter kaller den på andre metoder.
- God praksis å ha samme navn for filnavn og klasse
- Kodelinje slutter med ;
- Angir innholdet {}



Variabler

- Brukes for å lagre verdier.
- Java er statisk typing, variabler må bli deklarert en datatype, kan bare få verdier av denne typen.
  - **primitiver** : int, double, boolean, char etc
  - **referansetyper**: de inneholder referanse til objekter, f.eks String

```java
// primitiver
int tall1 = 3;
boolean erSant = false;
char char1 = 'A';

// referansetype
String tekststreng = "Hei";
```

Arrays

- “Sammenhengende variabler”
- Kan lagre flere verdier i en variabel
- Fast størrelse.
- index starter på 0.
    - gyldig index blir derfor: antall elementer -1

```java
int[] liste = new int[5];

liste[0] = 1;
liste[1] = 0;
liste[4] = 50;

for (int x : liste){
    System.out.println(x);
}
```

For-løkker

- Når vi skal iterere over samme linje med kode
- Mest brukte metoder er for loop, for each loop

```
for (initierging; if-sjekk; oppdtatering){
    ...
}

for (int i = 0; i < 5; i++){
    System.out.println(i);
}
```

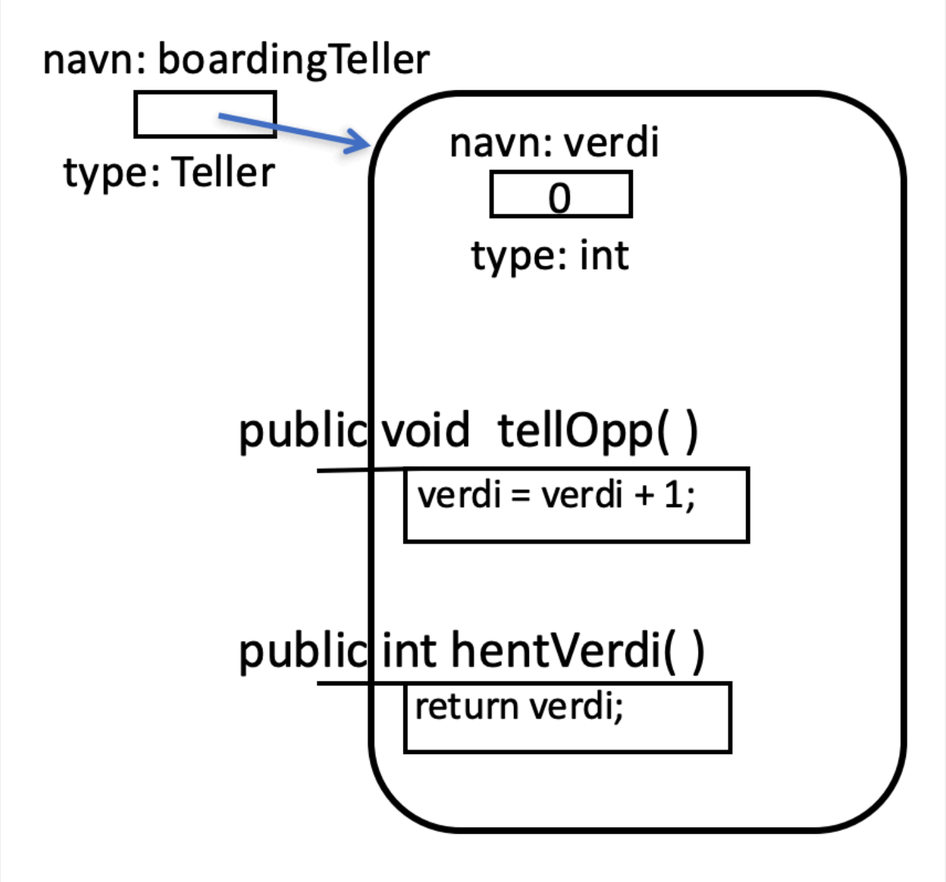
```
for (datatype x : beholder){
    ...

int[] liste = new int[5];

for (int x : liste){
    System.out.println(x);
}
```

Datastrukturtegning

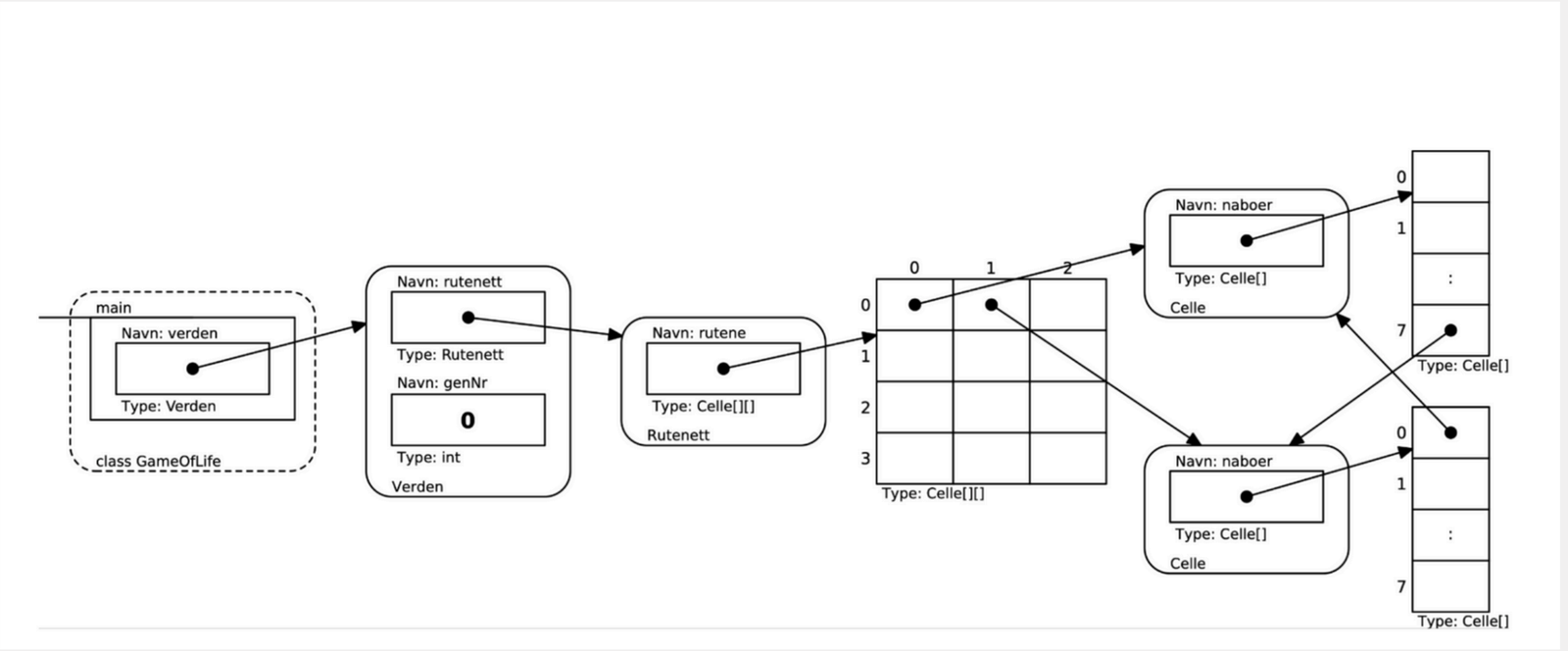
```
class Teller {
    private int verdi;
    Teller() {
        verdi = 0;
    }
    public void tellOpp( ) {
        verdi = verdi + 1;
    }
    public int hentVerdi( ) {
        return verdi;
    }
}
```



- Brukes for å visualisere metoder og variabler til en klasse.
- Hjelpe med forståelsen og se sammenhengen mellom større systemer

Datastrukturtegning

- Større system med flere klasser og metoder
- Fra innlevering1 Game of life



Synlighet for klasser

- **public** - betyr at andre klasser har tilgang til klassen
- **private** - Bare indre klasse kan vøre private. Bare den ytre klassen har tilgang til den private klasssen.

- **abstract** - kan ikke lage objekter av klasse, brukers for å forenkle detaljer. Kan brukes som blueprint av andre klasser (arv og polymorfi).

Synlighet for metoder og variabler

- **final** - kan ikke blir overskrevet etter tilordning
- **static** - tilhører klassen, ikke objektet
- **abstrakt** - ingen implementasjon, abstrakte metoder tilhører abstrakte klasser.
- **protected** - bare subklasser av klassen har tilgang til variabler, metoder

Metodesignatur

- Navnet, antall og rekkefølgen på parametere identifiserer en metode
- Kan finnes flere metoder med samme navn men ulike parametre

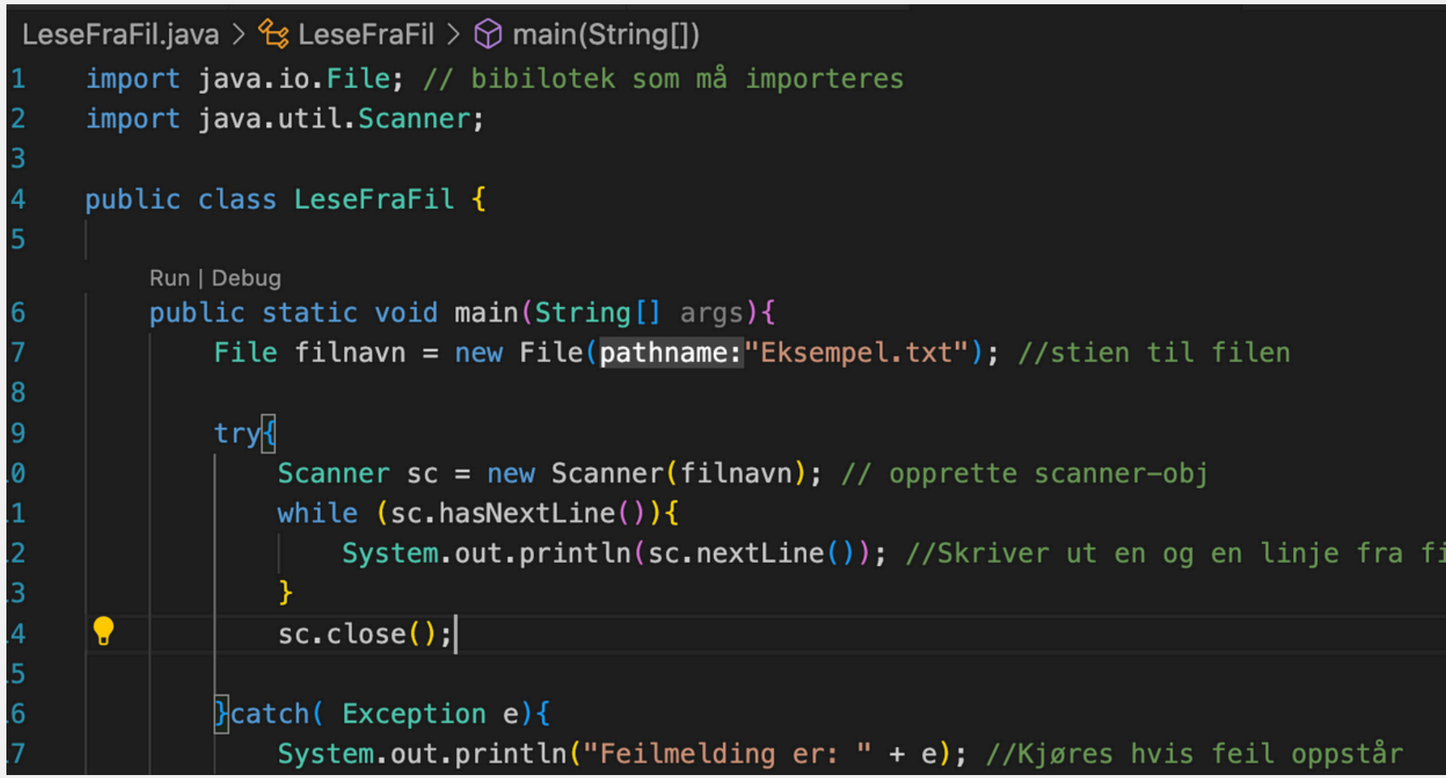
```
void exampleMethod(int a) {
    // Method with one int parameter
}

void exampleMethod(String b) {
    // Method with one String parameter
}

void exampleMethod(int a, String b) {
    // Method with one int and one String parameter
}
```

Lesing fra fil

- Trenger **Scanner, File klassen**. Må importeres.
- Må **sjekke unntak.** Dersom det oppstår feil under lesing av filen vil vi kunne håndtere feilen
  - try -> Koden som skal testes
  - catch -> Kjøres bare om feil ble oppdaget.
- **hasNextLine**() -> gir oss true/false
- **nextLine**() -> leser neste linje



Sammenligne string

- Siden strenger er objekter og ikke er primitiver så kan man ikke bruke ==
- s1 == s2 sjekker om variablene **peker** på det samme objektet.
- Må bruke **.equals().** Den sjekker om **innholdet** er det samme
- String literal lagrer streng-verdien i spesielt området i minne slik at det finnes bare en kopi.

```
String s1 = new String(original:"Hei");
String s2 = new String(original:"Hei");

System.out.println(s1==s2); // skrive ut false
```

```
String s3 = "et annet eksempel";
String s4 = "et annet eksempel";

System.out.println(s3==s4); // skrive ut true
```