6. Olion perustoiminnoista

1. Alustus
2. Pinta- ja syväoperaatioista
3. `equals`
4. *`clone`*
5. *`hashCode`*
6. *`toString`*
7. *`Comparable` ja `Comparator`*

Javan konstruktori 1(2)

- Nimi tarkalleen sama kuin luokan nimi
- Ei palautusarvoa (ei edes `void`)
- Kutsu vain `new`-operaattorilla
- Ei periydy (!)
  - perivän luokan täytyy määritellä konstruktori(t) uudestaan
- Voi kutsua yliluokan konstruktoria
  - `super(`*parametrit*`);`
  - täytyy olla konstruktorin ensimmäinen operaatio

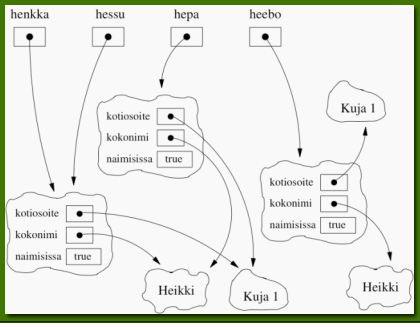
Javan konstruktori 2(2)

- Oletuksena sama suojausmääre kuin luokalla
- Voidaan ylikuormittaa
  - voi kutsua toista konstruktoria: `this(`*parametrit*`);`
- Mikäli käyttäjä *ei* ole määritellyt yhtään konstuktoria, systeemi luo oletuskonstruktorin
  - ei parametreja
  - alustaa jäsenmuuttujat oletusarvoihin

Olioiden samuus

- Identtisyys
  - ehto `a == b` on tosi
- Pintasamuus
  - olioiden a ja b jäsenmuuttujat sisältävät identtiset arvot
- Syväsamuus
  - oliot a ja b ovat pintasamat
  - kaikki olioiden a ja b viittaustyyppisistä vastinjäsenmuuttujista on rekursiivisesti pintasamat

Samanlaisuuden asteet



`equals`-operaation toteutusvaatimukset

i. Refleksiivisyys: `x.equals(x)`
ii. Symmetrisyys: `x.equals(y) <==> y.equals(x)`
iii. Transitiivisuus: `x.equals(y) & y.equals(z) ==> x.equals(z)`
iv. Konsistenssi: Jos pintasamuuteen liittyviä tietoja ei muuteta, vertailun on palautettava johdonmukaisesti tosi tai epätosi
v. Null-epäsamuus: `x.equals(null) == false`
vi. Rutiinin on toimittava olioiden tyypistä huolimatta.

Esimerkki: Henkilö

```
public boolean equals(Object toinen) {
    if (toinen == null) return false;
    if (toinen == this) return true;
    if (!(toinen instanceof Henkilö)) return false;
    Henkilö kaveri = (Henkilö)toinen;
    return nimi.equals(kaveri.nimi) &
           osoite.equals(kaveri.osoite) &
           syntymävuosi == kaveri.syntymävuosi;
}
```

Comparable-rajapinta

```
public class Henkilö implements Comparable<Henkilö> {
    /* ... */
    public int compareTo(Henkilö toinen) {
        int tulos = this.nimi.compareTo(toinen.nimi);
        if (tulos != 0) return tulos;
        if (this.syntymävuosi != toinen.syntymävuosi)
            return (this.syntymävuosi <
                    toinen.syntymävuosi) ? -1 : 1;
        else return
            this.osoite.compareTo(toinen.osoite);
    }
}
```

Perusoperaatioiden väliset riippuvuudet