Ohjelmointi 1 / syksy 2007
Ensimmäinen luento

Paavo Nieminen

nieminen@jyu.fi

Tietotekniikan laitos

Informaatioteknologian tiedekunta

Jyväskylän yliopisto

Jatkuva tiedotus WWW:n kautta

Syksyn 2007 ajan:

`http://www.cc.jyu.fi/~nieminen/ohj1/`

Tärkeä huomautus

- Luentokalvot on kirjoitettu Java-ohjelmointikielellä.
- Kieltä ei vielä ole tarkoituskaan osata! Lähtötaso-oletus tällä kurssilla on nolla eli ei mitään tietoa mistään.
- Tarkoitus on demonstroida, että
  - Ohjelmointikieli ei ole pelottavaa.
  - Se on yksi tapa kuvailla maailmaa.
  - Ohjelmasta voi hahmottaa *ainakin jotakin*, jopa paljonkin, vaikkei vielä osaisi kieltä.

Luennon suorittava ”Java-ohjelma”

```
/**
 * Leikkimielinen ``ohjelma'', joka toimii ``luentokalvoina''
 * Ohjelmointi 1 -kurssin ensimmäisellä luennolla.
 * Liian yksinkertainen ollakseen ns. oikea ohjelma!
 *
 * @author Paavo Nieminen <nieminen@jyu.fi>
 * @version 0.1
 */
public class EkaLuentoOhjelma{
  public static void main(String[] args){
    YliopistoKurssi ohj1 = new OhjelmointiYkkonen();
    ohj1.toteutaLuento(1);
    // ohj1.toteutaLuento(2); // FIXME toteuttaa vasta ekan.
  }
}
```

Tietty kurssi ”Javalla mallinnettuna”

```
/**
 * Tämä on johdantokurssi ohjelmoinnista.
 */
public class OhjelmointiYkkonen extends YliopistoKurssi
        implements Verkkokurssi, LukioonSoveltuva {

  /* Oletuksena muodostetaan syksyn 2007 kurssin kokoonpano */
  public OhjelmointiYkkonen(){...}
  public void toteutaLuento(int luennonNumero){...}
  public void taukoMinuutteja(double m){...}
  public void arvostele(){...}
  public int opintopistemaara(){return 6;}
  ...
}
```

HUOM: Kolme pistettä ”...” ohjelmalistauksessa tarkoittaa usein, että ”tässä kohtaa on oikeasti paljon muutakin, mutta juuri nyt ei olla tarkemmin kiinnostuneita juuri siitä”.

Abstrakti yliopistokurssi (”yläkäsite” Ohjelmointi 1:lle)

```
/** Kaikille yliopistokursseille yhteiset piirteet. */
public abstract class YliopistoKurssi{
  public static final int MAX_ARVOSANA = 5;
  public static final int MIN_ARVOSANA = 1;

  public abstract int opintopistemaara();

  private Henkilo   luennoitsija;
  private Henkilo[] tuntiopettajat;
  private Henkilo[] opiskelijat;
  private double[]  opiskelijoidenTenttipisteet;

  public abstract void toteutaLuento(int monesko);
  ...
}
```

HUOM: Entä SUOR/HYL -arvostelu? Monta luennoitsijaa?

Suorittaminen alkakoon!

```
1  /**
2   * Leikkimielinen ``ohjelma'', joka toimii ``luentokalvoina''
3   * Ohjelmointi 1 -kurssin ensimmäisellä luennolla.
4   * Liian yksinkertainen ollakseen ns. oikea ohjelma!
5   *
6   * @author Paavo Nieminen <nieminen@jyu.fi>
7   * @version 0.1
8   */
9  public class EkaLuentoOhjelma{
10   public static void main(String[] args){
11     YliopistoKurssi ohj1 = new OhjelmointiYkkonen();
12     ohj1.toteutaLuento(1);
13   }
14 }
```

HUOM: Rivinumerot eivät kuulu ohjelmaan, mutta ne helpottavat selittämistä.

Kurssiyksilö muodostuu suoritta-malla tarvittava alustus!

```
1  public class OhjelmointiYkkonen extends YliopistoKurssi
2          implements Verkkokurssi, LukioonSoveltuva {
3
4    /* Oletuksena muodostetaan syksyn 2007 kurssin kokoonpano */
5    public OhjelmointiYkkonen(){
6      asetaLuennoitsijaksi(
7              Jyu.henkkuntaNimella("Paavo Nieminen"));
8      Henkilo[] tuntiopet = new Henkilo[4];
9      tuntiopet[0] = Jyu.henkkuntaNimella("Jarmo Ernvall");
10     tuntiopet[1] = Jyu.opiskelijaNimella("Joel Lehtonen");
11     tuntiopet[2] = Jyu.opiskelijaNimella("Juho Tamminen");
12     tuntiopet[3] = Jyu.opiskelijaNimella("Irene Venäläinen");
13     asetaOpettajiksi(tuntiopet);
14     asetaOpiskelijoiksi(
15             Jyu.ketkaKurssilla("Ohjelmointi 1 syksy 2007"));
16   }
```

Suorittaminen jatkuu: Tapahtuu luento!

Suoritusjärjestys on: ensin piti muodostaa kurssi (rivi # 11) Sen jälkeen voi pyytää kurssiyksilön tarjoamaa palvelua, luennon numero 1 toteuttamista:

```
11          YliopistoKurssi ohj1 = new OhjelmointiYkkonen();
12          ohj1.toteutaLuento(1);
```

. . . ja OhjelmointiYkkonen -tyyppinen kurssiyksilö sitten toteuttaa luennon . . .

(jatkoa OhjelmointiYkkosen toteutuksesta.)

```
17    /** Toteuttaa luennon, jonka järjestysnumero annetaan. */
18    public void toteutaLuento(int luennonNumero){
19      if (luennonNumero == 1){
20        Henkilo luennoitsija = this.annaLuennoitsija();
21        Henkilo[] opiskelijat = this.annaOpiskelijat();
22        Henkilo[] tuntiopettajat = this.annaTuntiopet();
23
24        /* Henkilöiden esittelyt */
25        luennoitsija.esitteleItsesi();
26
27        for (int i=0; i<tuntiopettajat.length; i++){
28          Henkilo tuntiope = tuntiopettajat[i];
29          tuntiope.esitteleItsesi();
30        }
```

(jatkoa edellisestä.)

```
31          /* Ensimmäisen luennon asiat */
32          luennoitsija.otaOpetettavaksi(opiskelijat);
33          luennoitsija.virittaydyTunnelmaan("Ohjelmointi 1, s07");
34
35          luennoitsija.kerroAiheesta("Kurssin kuvaus");
36          luennoitsija.kerroAiheesta("Suoritusvaatimukset");
37          luennoitsija.kerroAiheesta("Arvostelu");
38          luennoitsija.kerroAiheesta("Aikataulu");
39          luennoitsija.kerroAiheesta("Demot");
40          luennoitsija.kerroAiheesta("Harjoitustyö");
41          luennoitsija.kerroAiheesta("Oikotiet / tenttiminen");
42
43          luennoitsija.vastaaKysymyksiin();
44          taukoMinuutteja(15.0);
```

(jatkoa edellisistä.)

```
45          luennoitsija.kerroAiheesta("Mitä on ohjelmistotyö");
46          luennoitsija.kerroAiheesta("Mitä on ohjelmointi");
47          luennoitsija.kerroAiheesta("Tämän kurssin rooli");
48          luennoitsija.kerroAiheesta("Opittavat käsitteet");
49          luennoitsija.kerroAiheesta("Yleiskuva kurssin luennoista");
50          luennoitsija.kerroAiheesta("Toimintaympäristömme");
51          luennoitsija.kerroAiheesta("Tämän viikon kotitehtävät");
52          luennoitsija.vastaaKysymyksiin();
53          luennoitsija.kiitaJaPoistu();
54          return;
55        }
56      }
```

HUOM: Vaikka ”Java-ohjelmana” ensimmäinen luento päättyisi näin, todellisessa maail-

massa jatketaan vielä yhteenvedolla, jos suinkin jää aikaa.

Ohjelma on päättynyt.

”Sovellusohjelmassa” oli kaksi riviä, jotka on nyt suoritettu:

```
/**
 * Leikkimielinen ``ohjelma'', joka toimii ``luentokalvoina''
 * Ohjelmointi 1 -kurssin ensimmäisellä luennolla.
 *
 * @author Paavo Nieminen <nieminen@jyu.fi>
 * @version 0.1
 */
public class EkaLuentoOhjelma{
  public static void main(String[] args){
    YliopistoKurssi ohj1 = new OhjelmointiYkkonen();
    ohj1.toteutaLuento(1);
  }
}
```

Vaikka sovellus oli lyhyt, sen käyttämät oliot suorittivat paljon toimenpiteitä.

Tehdään yhteenveto ja huomioita.

- Vaikka tavoite oli puhtaasti olla kalvosarja, kaikki Java-ohjelmointikielellä kirjoitettu toimii ohjelmana, joka voidaan suorittaa tietokoneella. ($\rightarrow$ näytetäänpä esimerkki videoscreenillä)
- Ohjelma on yksi tapa ilmaista asioita täsmällisesti.
- Eritoten ohjelmalla voi kuvata toimenpiteiden suunnitelmallista toteuttamista jossain järjestyksessä joidenkin tapahtumien (tai kuten meilläpäin sanotaan, *syötteiden*) ilmetessä.

Näkökulma ohjelmointiin: mallintaminen

- Ohjelma *voi olla malli reaalimaailmasta* eli ns. jonkinlainen *simulaattori*. On vaikea edes keksiä ohjelmaa, joka ei olisi malli jostakin joko todellisesta tai epätodellisesta sydeemistä, esim:
  - pelit ovat malleja virtuaalimaailmoista
  - pankkijärjestelmät ovat malleja tileistä, asiakkaista, rahavirroista ym.
  - musiikkisoftat ovat malleja nuoteista, instrumenteista, reaalimaailman ääni-ilmiöistä ym.
- Silloin ohjelma on useimmiten epätäydellinen ja/tai epätarkka verrattuna todellisuuteen.
- *Mallit useimmiten ovat! Siltä ei voi välttyä!*

Ohjelmointikielen rooli

- Ohjelmointikielellä ilmaiseminen on vain yksi tapa esittää malli.
- Se on *tarkin esitysmuoto ohjelmalle* — ja siinäpä sen yksi alkuvaikeus onkin . . . Jotta ohjelmakoodi on mahdollista kirjoittaa tai ymmärtää, täytyy ihmisen osata pohtia sen merkitystä jollain *abstraktimmilla tavoilla*! Suoraan ohjelmointikieleen on vaikea hypätä (paitsi triviaaleissa ohjelmissa, joista ei ole kunnon hyötyä kenellekään. . .)!
- Ohjelmointikieli on välttämätön, koska tieto(kone)järjestelmä ymmärtää vain sellaista.
- Silti ohjelmointi on paljon enemmän kuin kieli!

Malli on ”näkymä” ohjelmaan.

Ohjelmointikieli on siis tarkin mahdollinen malli. Mitä yksinkertaisempaa/yleispätevämpää on? Vain pari esimerkkiä:

- Algoritmimallit (esitys esim. ”pseudokoodina”)
- Oliomallit (luokittelu, luodut yksilöt)
- Tapahtumien järjestyksen ja osallistujien kuvaukset
- Käyttöliittymäkuvaukset (esim. ”screenshotit”)
- Järjestelmämallit (mukana ihmiset, laitteet, tietovarastot, kommunikaatio, . . .)

HUOM: Olennaista on osata mallin asteittainen tarkentaminen: Ensin karkea malli, sitten

hienojakoisempi ja hienojakoisempi . . . lopulta ohjelmointikieltä . . . mutta vasta lopulta!

Näyttäisinkö esimerkkejä mallinnuksesta?

- Vaikeus: Pitäisi kuulemma välttää ”perinteistä yliopistovirhettä”, että asioiden esitys menee yli hilseen . . . Tarkoitus on oppia ”vain ohjelmoimaan”, joten kannattaako edes näyttää mitään muuta?
- Luennoitsijan näkemys 1: abstraktit esitysmuodot on nimenomaan kehitetty yksinkertaistamaan mm. ohjelmien ymmärtämistä! Niitä katsomalla pitäisi kenen tahansa vastaantulijan tajuta. . . ainakin jotain . . .
  Mm. siksi ne ovat paljolti kuvallisia ja jättävät kertomatta paljon yksityiskohtia.

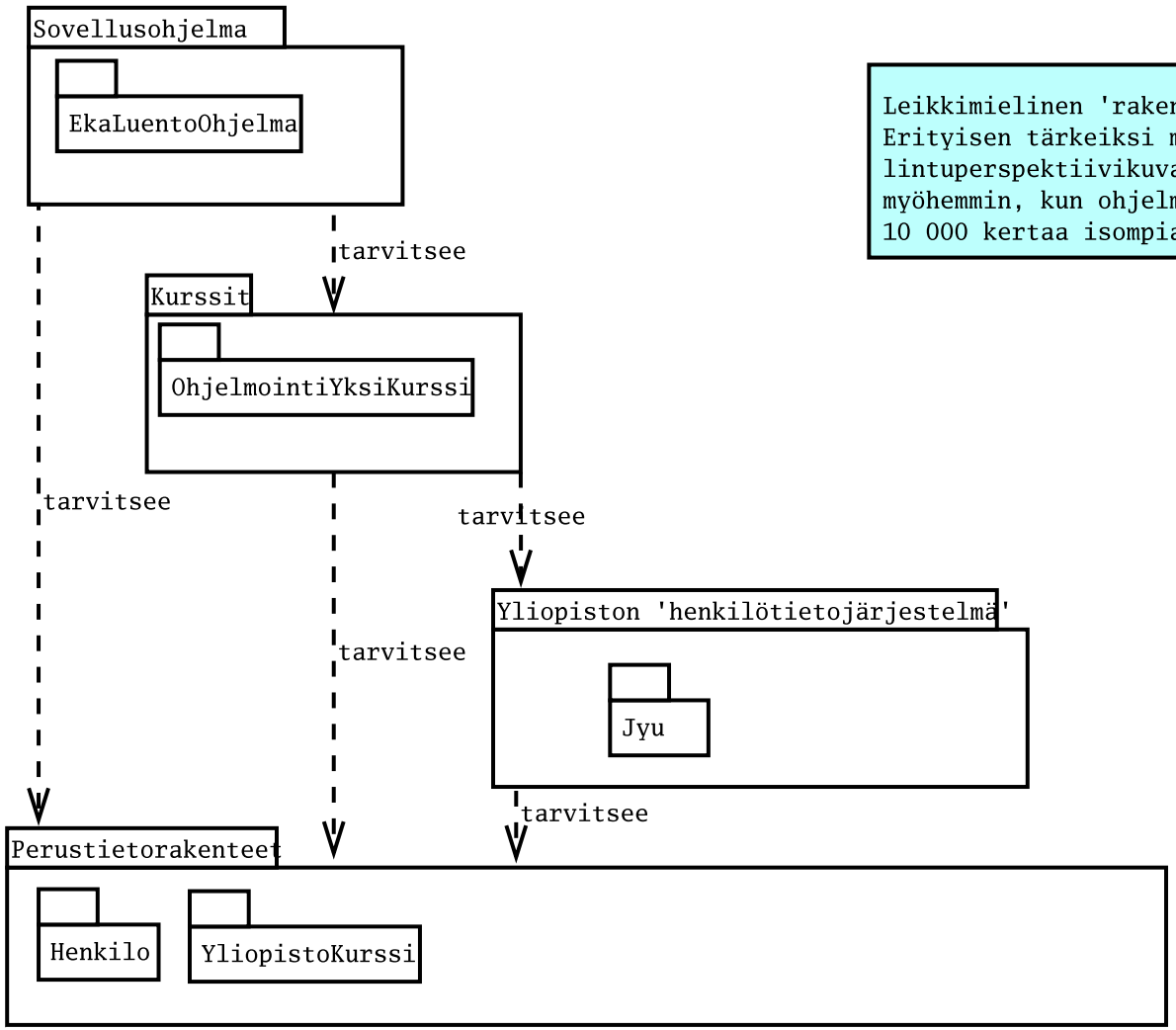
Näyttäisinkö sittenkään esimerkkejä?

- Luennoitsijan näkemys 2: Ohjelmointitaito on välttämätön näiden kuvien *piirtämiselle* ja ns. *”täydelliselle” ymmärtämiselle*, mutta ei niiden *katsomiselle* ja niiden kautta oppimiselle.
- Luennoitsijan oletus: Näiden katselu tukee ja helpottaa (mutta ei yksinään tee mahdolliseksi!) ohjelmoinnin oppimista
- Fakta: Monien eli ohjelmamallien “syvällinen” merkitys ja se, miten näitä kannattaa tehdä, on ehdottomasti jatkokurssien asia (mm. Oliokeskeinen tietojärjestelmien kehittäminen, Johdatus ohjelmistotekniikkaan, . . .).

Taidanpa siis näyttää vähän kuvia...

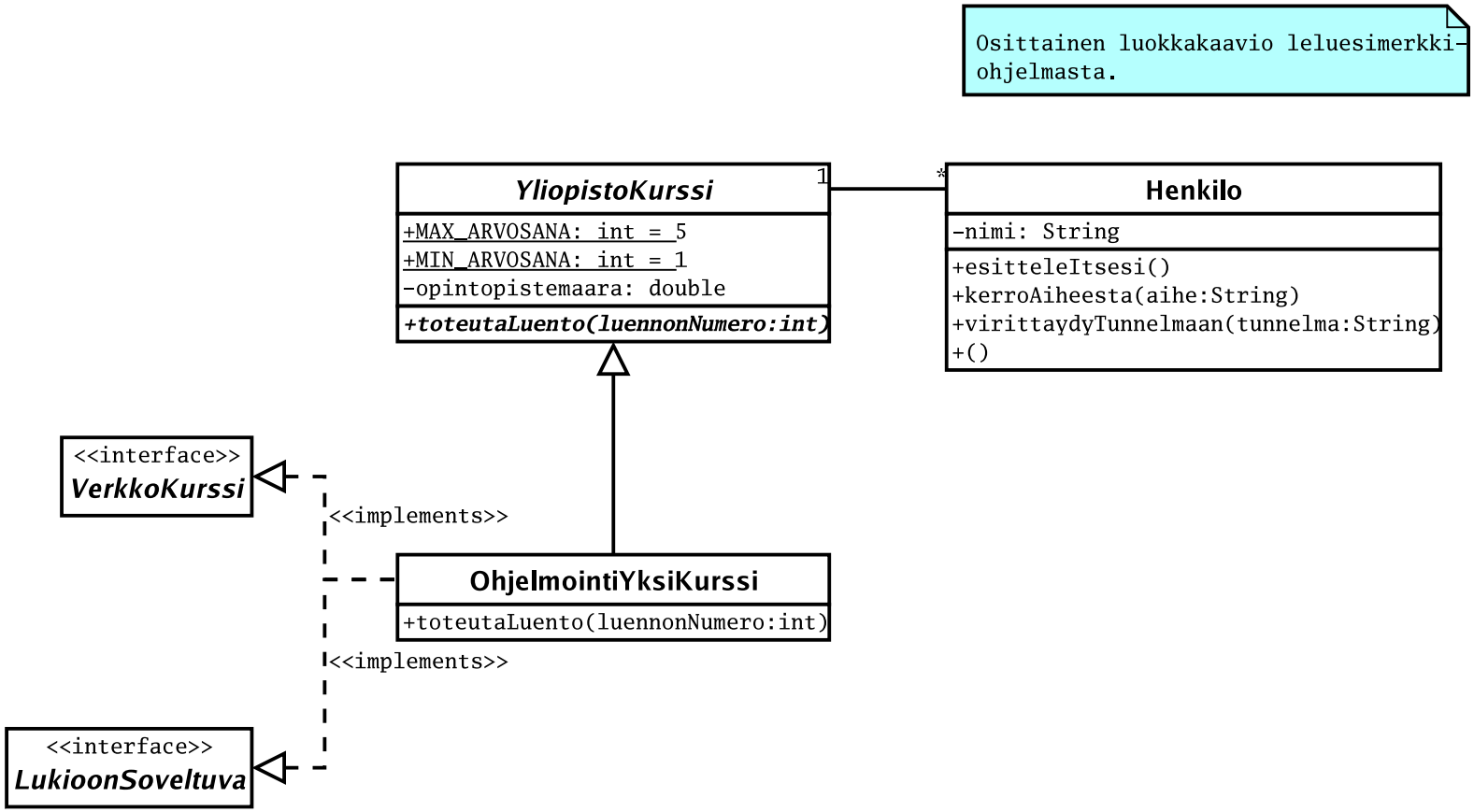
- Aika paljon IT-opinnoista tulee pyörimään näiden parissa, joten mielelläni otan ne mukaan heti aluksi!

Esim: Moduulijakokaavio

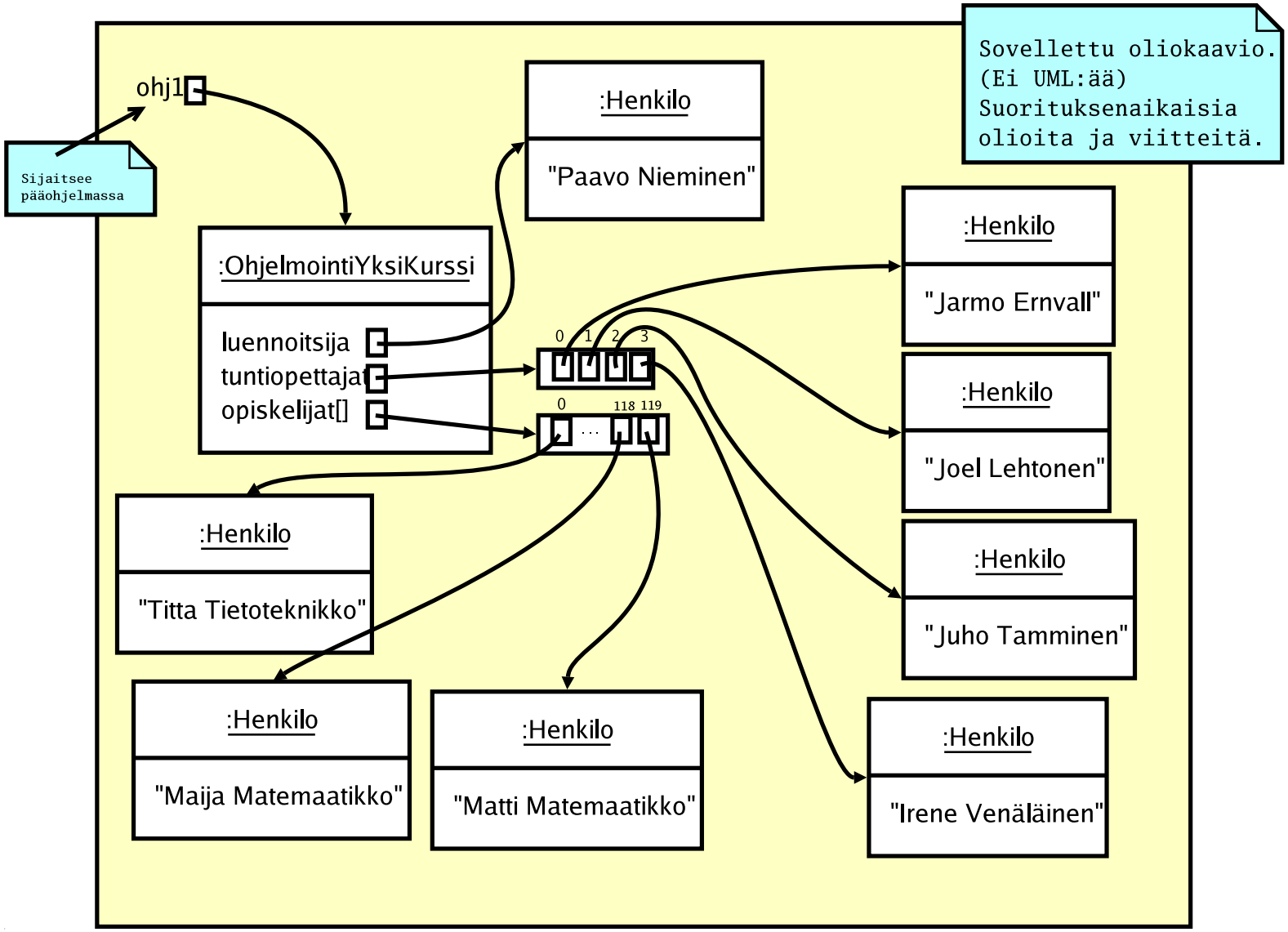


Leikkimielinen 'rakennekuva'.
Erityisen tärkeiksi mm. tällaiset
lintuperspektiivikuvat tulevat
myöhemmin, kun ohjelmistot ovat esim.
10 000 kertaa isompia kuin luentoesimerkki.

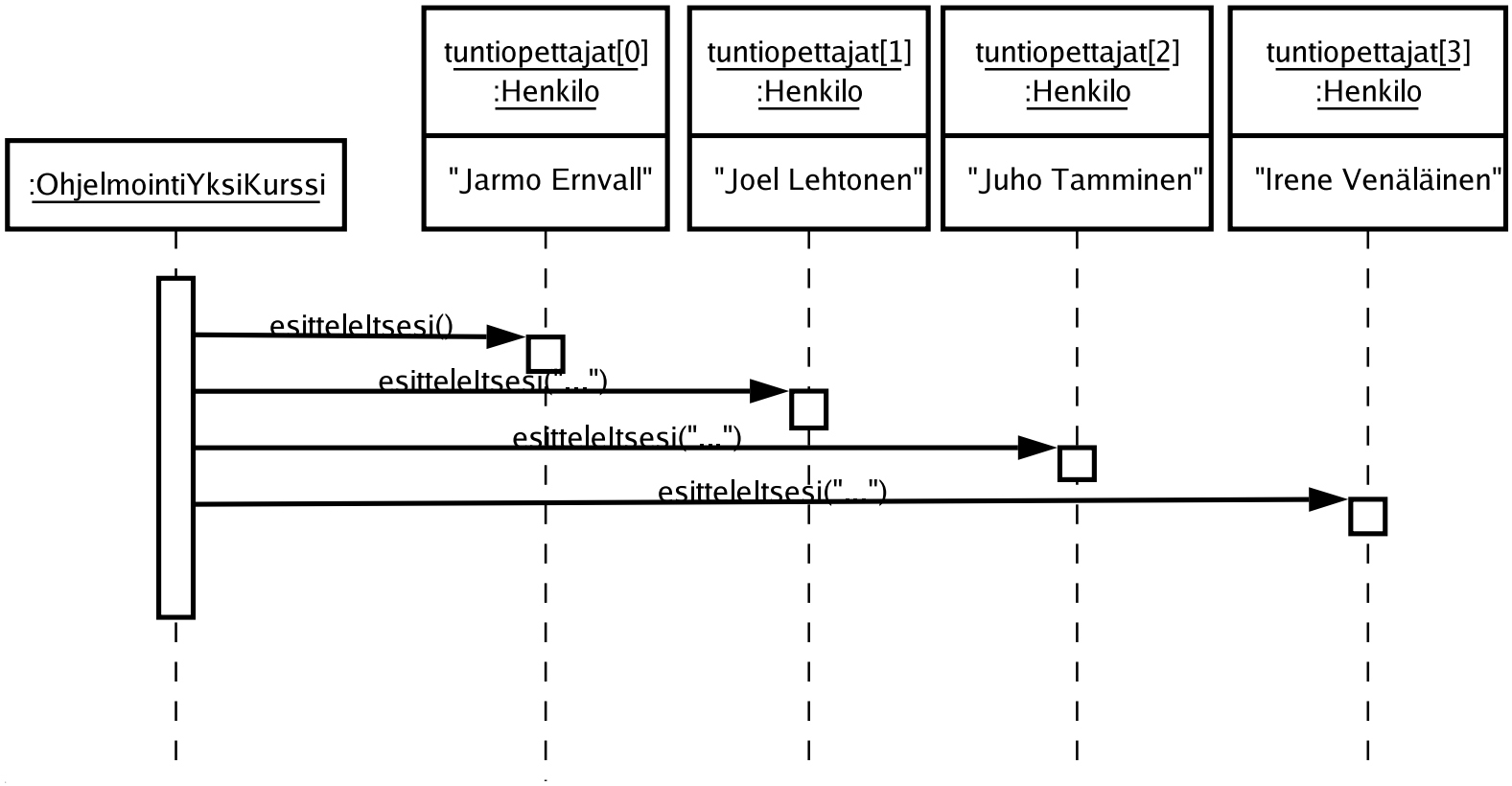
Esim: Luokkakaavio



Esim: Oliokaavio

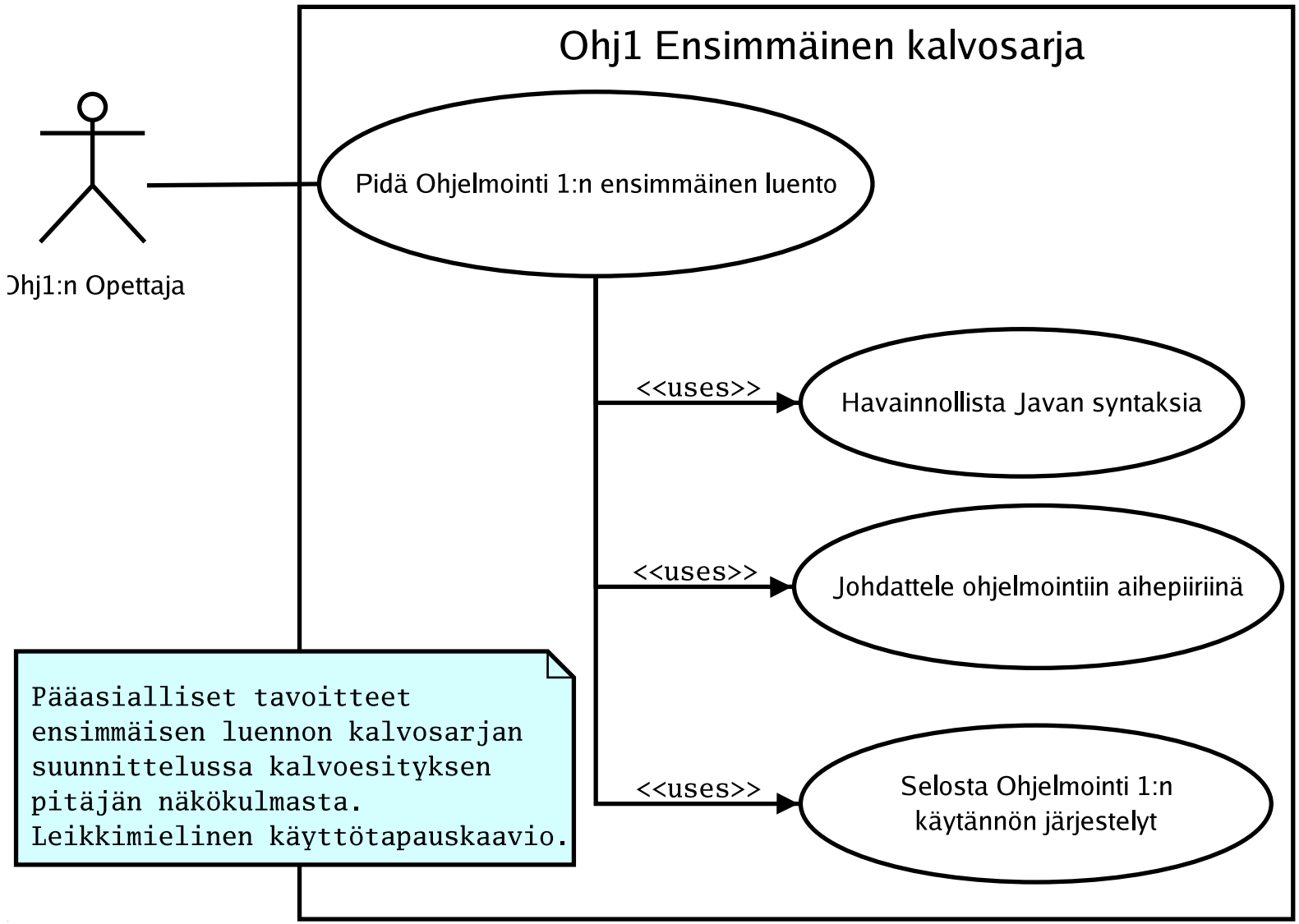


Esim: Sekvenssikaavio



Tämä on varmaan viimeinen sekvenssikaavio Ohjelmointi 1 -kurssilla, koska emme pääse olioiden välisten toimenpiteiden suunnitteluun vielä juurikaan käsiksi . . .

Esim: Käyttötapauskaavio



Esim: Pseudokoodi – mitä se on

- Tarkoituksella ei puhtaasti ohjelmointikieltä
- Kuitenkin rakenteeltaan vastaavaa + riittävän täsmällistä
- Voi kirjoittaa lähes siten kuin jotakin tuntemaansa ohjelmointikieltä
- Tavoite 1: kuka tahansa ohjelmointitaitoinen ymmärtää, vaikkei osaisikaan nimenomaan samoja ohjelmointikieliä kuin kirjoittaja
- Tavoite 2: ohjelmointitaitoinen osaa toteuttaa itse tuntemallaan kielellä

Esim: Pseudokoodi – mihin soveltuu

- Pseudokoodi on hyvä kommunikointikeino ohjelmoinnillisten ideoiden siirtoon. Se on myös suhteellisen täsmällinen tapa kirjoittaa algoritmi.
- Suositeltava vaihe algoritmin suunnittelussa ennen ohjelmointia
- Esimerkki: *Kirjoita pseudokoodina, miten koordinoidaan, että jokainen opettaja vastaa kaikkien opiskelijoiden kaikkiin kysymyksiin.*

Esim: Pseudokoodi, lähes suomea

```
Toimenpide vastausKaikkiinKysymyksiin:
    /* Oletetaan, että on henkilöitä sisältävät rakenteet:
       'opiskelijat' ja 'opettajat' */

   jokaiselle opiskelijalle tee seuraavaa:
     purista kysymykset irti opiskelijasta
     jokaiselle kysymykselle tee seuraavaa:
       jokaiselle opettajalle tee seuraavaa:
         Kommunikoi opiskelijalle opettajan vastaus kysymykseen

/* On jätetty vielä tarkentamatta, miten kysymys puristetaan
irti opiskelijasta. Vaikeampia osatehtäviä kannattaakin miettiä
erillisinä, aina pienemmiksi osatehtäviksi pilkkoen. Samoin on
jätetty tarkentamatta, miten vastaus kommunikoidaan opiskelijalle.
Osatehtävät on kuitenkin pseudokoodissa jo oikeilla paikoillaan!*/
```

Esim: Algoritmi suomen kielellä

Pseudokoodi on jo lähellä tietokonetoteutusta. Ennen pseudokoodia, on varmasti aina hyvä miettiä tehtävää parhaiten tuntemallaan kielellä, yleensä meilläpäin suomeksi. Esim:

Tiskaaminen tarkemmin ajatellen on sitä, että 1) jos on vielä likaisia astioita, niin otan niistä seuraavan käteeni 2) harjaan sitä kunnes se ei ole likainen 3) laitan astian kuivumaan 4) siirryn seuraavaan likaiseen astiaan eli palaan kohtaan 1.

Esim: Raapustukset

Kynä ja paperi näppiksen viereen! Jos voit piirtää, kirjoittaa tai tuhrata ruutupaperiin ihan mitä tahansa, joka auttaa sinua ajattelemaan ohjelmointiongelmaasi, TEE SE! Älä tuijota kuvaruutua vaan ajattele aivoilla, käsillä, jaloilla, millä se helpoiten vaan käy! Tartu näppikseen vasta kun on jotain kirjoitettavaa ohjelmointikielellä. Silloin sinulla on jo tärkein eli ajatus.

Missä määrin Ohj1:llä mallinnetaan

- Algoritmit, algoritmien tarkentaminen asteittain ja niiden ajatteleminen pseudokoodina on kurssin tärkein (oikeastaan AINOA) asia!
- Java on työkalumme; sen osaaminen tulee pakosti ”kaupan päälle”. Java perustuu täysin olioiden hyödyntämiseen, joten olioita on ymmärrettävä.
- *Ymmärrämme* oliot *alustavasti*, mutta *emme suunnittele* niitä vielä! $\rightarrow$ Ohj2, GKO, projektityöt
- Emme ”suunnittele järjestelmiä” $\rightarrow$ kurssit mallintamisesta ja järjestelmäsuunnittelusta (Oliokeskeinen tietojärjestelmien kehittäminen jne.)
- Kuitenkin ymmärrämme pienenkin ohjelman olevan tietynlainen (yksinkertainen) järjestelmä

Säännöt on tehty rikottaviksi?

- Edellä sanottiin, että ensin pitää mallintaa ja sitten tehdä ohjelma . . . Siihen onkin syytä palata *aina, kun jonkun ohjelman tekeminen tuntuu vaikealta!*
- Mutta yksi ohjelmoinnin ilo on, että voit vaan kirjoitella ohjelmointikieltä ja kokeilla mitä kivaa mikäkin tekee. Voit muokkailla ohjelmaa mielin määrin, kunnes se on mallina ihan eri kuin aluksi . . .
- Kunhan *JATKUVASTI YMMÄRRÄT*, *mitä* ohjelmasi tekee, *miksi* se tekee juuri niin, ja *onko se riittävän hyvä tapa* kulloiseenkin tarkoitukseesi!!
- Hallitusti tehtynä tämä on ihan oikea suunnittelumenetelmä.

Yhteenveto

- Tämä oli ”shokkialoitus” ohjelmointiin
- Hämmennys ja ihmettely on toivottu olotila ;)
- Ensi viikolla otetaan rauhallisempi lähtöruutu
- Herännee kysymys: mikä tässä kaikessa oli olennaista?

Olennaista oli

- Ohjelmointikieli on ainoa mitä tietokone ymmärtää
- Ohjelmointikieli on vain kieli kielien joukossa; joku hullu voi kuvata sillä vaikka luennon etenemisen aiheesta toiseen!
- Ohjelmointikieli on vain yksi monista perspektiiveistä nähdä tietokoneohjelma
- Jostain päästä aina aloitetaan: Ohjelmointi 1 keskittyy algoritmien tekemiseen, joka on kaiken ohjelmoinnin peruslähtökohta
- Suuri osa äsken nähdyistä asioista jätetään hautumaan tulevia kursseja varten.

Ensi viikolla nähdään!

- Tehkää ahkerasti demoja ja muistakaa myös ajatella.
- Seuraava luento on maanantaina 17.9. klo 12.
- Luvassa esimerkkiohjelmia, lofi-actionia (?) ja kielioppien ymmärtämisen alkeita.