Java tutorial

JavaBeans

- Nagyon divatos technológia
- Nem tévesztendő össze az Enterprise Java Beans technológiával
- Programozási konvenciók gyűjteménye
- Komponens-elvű programozás
- Vizuális fejlesztőeszköz segítségével

Komponens-elvű programozás

- A programot komponensekből állítom össze
- Egy komponens egy önálló egység, jól meghatározott interfésszel
  - olyasmi, mint az objektum, csak nagyobb is lehet
- Lehetőleg előregyártott komponensekből építkezem

Off-the-shelf

- Vannak „áruházak”, ahol komponenseket vehetek
- Csak össze kell kapcsolni őket, és kész is van az alkalmazás
- A programozás mérnöki, vagy pláne szerelői munkává válik
- Olcsón megbízható terméket állítunk elő

Összeszerelés

- A beszerzett komponenseket be kell konfigurálni (testreszabás, customization)
- A szabványok garantálják, hogy
  - a komponenseket össze lehet rakni
  - bármikor ki lehet cserélni egy másik, hasonló funkcionalitást nyújtóval
- A szerelés támogatható vizuális fejlesztő eszközzel

Megoldás Java módra

- JavaBean - komponens
- Egy JavaBean attól JavaBean, hogy elkészítésekor bizonyos konvenciókat követünk
- java.beans csomag
  - Önelemzést (reflection) használ: java.lang.reflect
- Eseménykezelés (mint AWT-ben láttuk)
- Perzisztencia
- Vizuális megjelenés
- BeanBox környezet

BeanBox

- A Sun készített egy ingyen letölthető bean-fejlesztő környezetet
- Csak játékra / demonstrációra jó
  - Az elveket ki lehet próbálni rajta
  - Bonyolultabb dolgokat nem lehet vele csinálni
  - Sok olyan dolog, ami szükséges egy program elkészítéséhez, nincs benne
  - A professzionális eszközök működési elve is ugyanaz

Java tutorial

Bean Development Kit

- BDK 1.1 - 1999
- http://java.sun.com/javabeans/
- Benne van a BeanBox, dokumentációk, példaprogramok
- A BDK mellett más, kapcsolódó technológiák:
  - JavaBeans Tools for ActiveX
  - InfoBus
  - JavaBeans Activation Framework (JAF)

A BeanBox futtatása

- beans/beanbox/run.sh
- beans\beanbox\run.bat

- Toolbox, Beanbox, Properties, Method Tracer

Játék

- Bean-ek elhelyezése a BeanBox-ban
- Tulajdonságok módosítása
- Bean-ek összekapcsolása
  - Juggler vezérlése nyomógombokkal
- Mentés és visszatöltés
- Applet-té alakítás
- Feladat: Vízmolekula forgatása

Mi történik?

- A BeanBox segítségével testreszabhatunk, összekapcsolhatunk olyan bean-eket, amelyek nem is léteztek még a BeanBox megírásakor
- A BeanBox futási időben megismerkedik velük, és segítségével manipulálhatjuk őket
- Egy időben fut a BeanBox és a vele tervezett és összeállított alkalmazás

Önelemzés (reflection)

- Egy Java program információt gyűjthet saját magáról, vagy más Java programról - futás közben
- A típusdefiníciókat leíró class fájlok sok információt tartalmaznak
- Egy Java program megvizsgálhatja a virtuális gép által betöltött bájtkódot
- Sőt, a belső reprezentációjához hozzáférve nem csak elemezheti, de akár aktiválhatja is

```
import java.lang.reflect.*;
class Meghiv {
  public static void main(String[] args)
  throws Exception {
      Class c = Class.forName(args[0]);
      Class[] formalArgs = (args.length > 2)
                           ? new Class[] {String.class}
                           : new Class[] {};
      Object o = c.newInstance();
      Method m = c.getMethod(args[1],formalArgs);
      Object[] actualArgs = (args.length > 2)
                           ? new Object[] {args[2]}
                           : new Object[] {};
      m.invoke(o,actualArgs);
  }
}
```

```
public class Alma {
  public void kiir() {
    System.out.println( "alma "+this );
  }
  public void kiir(String s) {
    System.out.println( "alma "+s+" "+this );
  }
  public void print() {
    System.out.println( "apple "+this );
  }
}
```

```
$ java Meghiv Alma kiir szia
alma szia Alma@fee6fc
$
```

JavaBeans konvenciók

- Az önelemzés segítségével egy bean fejlesztőeszköz képes felderíteni és aktiválni metódusokat
- Sőt, tudja őket speciálisan is kezelni, ha követjük az elnevezési konvenciókat
  - tulajdonságok testreszabása
  - eseménykezelés

Java tutorial

Tulajdonságok

- Tulajdonság: amit be lehet állítani (set) és le lehet kérdezni (get)
- Például egy bean-nek lesz egy „méret” nevű „int” típusú tulajdonsága, ha definiálunk benne ilyen metódusokat:

```
public void setMéret( int méret )
public int getMéret()
```

- Vannak még egyéb konvenciók is, pl. tömbök esetén...

Testreszabás

- Tetszőleges bean tulajdonságaihoz a BeanBox futás közben elkészítheti a tulajdonságszerkesztő dialógust (Properties) a futási idejű információk alapján
- Lehetőség van saját tulajdonságszerkesztők elkészítésére, és a tulajdonságszerkesztő dialógusban történő megjelenítésére (PropertyEditor)
- Sőt, akár saját tulajdonságszerkesztő dialógus definiálására is (Customizer, BeanInfo)

Feladat

- Írjunk olyan JavaBean-t, melynek grafikus felülete egy piros pontot tartalmaz. A pont mérete legyen a bean tulajdonsága.

Java tutorial

Perzisztencia, vizuális megjelenés

- A JavaBean-ek két fontos tulajdonsága:
  - el lehessen tárolni őket (és a belőlük összeállított programot)
  - lehessen vizuális eszközzel manipulálni őket
- A BeanBox a Java szerializációt használja
  - megköveteli, hogy a JavaBean-ek megvalósítsák a Serializable interfészt
- Általában a java.awt.Component-ből származtatjuk le a bean-einket...

JAR fájlok

- Az elkészített JavaBean-eket JAR fájlokban szokás tárolni.
- Ez a class fájl(ok)on kívül tartalmazza a szükséges erőforrásokat, és egy manifest.tmp fájlt.
- manifest.tmp:
```
Name: Pirospont.class
Java-Bean: True
```
- **`jar cfm Pirospont.jar manifest.tmp Pirospont.class`**
- A JAR fájl betölthető egy futó BeanBox-ba, vagy, ha bemásoljuk a beans/jars könyvtárba, automatikusan betöltődik induláskor

Eseménykezelés

- Forrás - esemény - figyelő
- Beépített események:
  - tulajdonság változása:
    kötött (bound) tulajdonság, PropertyChangeEvent
  - tulajdonság változásának megkísérlése
    vétózható (vetoable) tulajdonság
    VetoableChangeListener, PropertyVetoException

Saját események esetén

- Definiáljuk az esemény osztályt: java.util.EventObject
- Definiáljuk a figyelők interfészét: java.util.EventListener
- Megvalósítjuk a forrást
  - implementáljuk a figyelők regisztrációját
  - ahol kell, kiváltjuk az eseményt, és értesítjük a regisztrált figyelőket
- Írhatunk speciális figyelő osztályokat is, de (egy BeanBox által generált adapter segítségével) összeköthetjük a forrást más bean-nel is…

Feladat

- Számoló bean: egy Label, melynek felirata egy szám. A szám növelhető a Számoló növel() metódusával. Amikor elérjük a „max” értéket, a számoló 0-ra csökken, és MaxEvent váltódik ki. A max értéke egy tulajdonság.
  Alarm bean: egy nyomógomb, mely figyeli a MaxEvent-et, és piros színre vált, ha az bekövetkezik.

Java tutorial