Grafikus felhasználói felületek

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Grafikus felhasználói felületek

- A felhasználó a programmal úgy tartja a kapcsolatot, hogy
  - egeret húzogat
  - menüpontokból válogat
  - gombokat nyomogat
  - listákban kijelölget
  - ...

Java Foundation Classes - JFC

- Abstract Windowing Toolkit - AWT
  - ablakozó rendszer
- Swing
- Accessability
- Drag-and-drop
- Java2D
- Pluggable Look-And-Feel

Abstract Windowing Toolkit - AWT

- Ablakozó rendszer
- Egységes lehetőségek minden platformon
  - Motif, Windows, MacOS, ...
  - A különféle rendszerek lehetőségeinek metszete
  - Picit fapados...(?)
- java.awt csomag, valamint alcsomagok
  - java.awt.color, java.awt.event, java.awt.font, ...
- Ugyanaz a program minden platformon fut

Első program

```
import java.awt.Frame;
class Ablak {
  public static void main(String args[]){
    Frame f = new Frame();
    f.setSize(100,200);
    f.setVisible(true);
  }
}
```


PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

Frame

- Egy keret, fő program-képernyő
- Alapvető funkcionalitás
  - mozgatható
  - átméretezhető (a sarkánál fogva)
  - ikonizálható, visszaállítható
  - opcionális címsor
- Sok minden hiányzik még
  - például bezárás

Hogyan írjunk programot a tanfolyamon

```
import java.awt.*;
class Hello extends Frame {
  public static void main(String args[]){
    Hello hello = new Hello();
    hello.setSize(100,200);
    hello.setVisible(true);
  }
}
```

Hogyan írjunk programot a tanfolyamon

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
  }
  public static void main(String args[]){
    Hello hello = new Hello();
    hello.setSize(100,200);
    hello.setVisible(true);
  }
}
```

A grafikus felület elemei

- Komponensek (Component)
- Szokás widget-nek is nevezni
- Címkék, nyomógombok, listák, legördülő listák, kiválasztó dobozok, "rádió" dobozok, szöveg beviteli mezők, ...
- Label, Button, List, Choice, Checkbox, TextField, TextArea, ...
- A menük egy picit más témakörbe tartoznak

A komponensek kipróbálása

- Végigpróbáljuk a komponenseket
- Nagyjából ugyanúgy néznek ki és működnek minden platformon
- Az eddigi programban a konstruktor törzsébe tesszük a komponensek létrehozását
- Piszkálni már lehet a komponenseket, de a programunkat még nem készítjük fel arra, hogy csináljanak valamit is

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

A komponensek: Label

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     add(new Label("Hello"));
  }
  public static void main(String args[]){
    Hello hello = new Hello();
    hello.setSize(100,200);
    hello.setVisible(true);
  }
}
```

A komponensek: Label középre igazított szöveggel

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     add(new Label("Hello", Label.CENTER));
  }
  public static void main(String args[]){
    Hello hello = new Hello();
    hello.setSize(100,200);
    hello.setVisible(true);
  }
}
```

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

A komponensek: Button

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     add(new Button("Hello"));
  }
  public static void main(String args[]){
    Hello hello = new Hello();
    hello.setSize(100,200);
    hello.setVisible(true);
  }
}
```

A komponensek: Checkbox

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     add(new Checkbox("Hello"));
  }
  public static void main(String args[]){
    Hello hello = new Hello();
    hello.setSize(100,200);
    hello.setVisible(true);
  }
}
```

A komponensek: TextField

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     add(new TextField("Hello"));
  }
  public static void main(String args[]){
    Hello hello = new Hello();
    hello.setSize(100,200);
    hello.setVisible(true);
  }
}
```


PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

A komponensek: TextArea

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     add(new TextArea("Hello"));
  }
  public static void main(String args[]){
    Hello hello = new Hello();
    hello.setSize(100,200);
    hello.setVisible(true);
  }
}
```

A komponensek: List

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     List list = new List(10);
     list.add("Szia");
     list.add("Hello");
     list.add("Salut");
     add(list);
  }
  public static void main(String args[]){...}
}
```

A komponensek: List (multi)

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     List list = new List(10, true);
     list.add("Szia");
     list.add("Hello");
     list.add("Salut");
     add(list);
  }
  public static void main(String args[]){...}
}
```

A komponensek: Choice

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     Choice choice = new Choice();
     choice.add("Szia");
     choice.add("Hello");
     choice.add("Salut");
     add(choice);
  }
  public static void main(String args[]){...}
}
```

Java tutorial

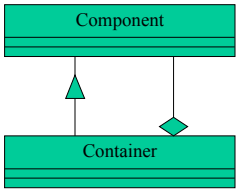
Copyright © 2000-2002, Kozsik Tamás

Konténer (Container)

- Komponenseket tartalmazhat
- A konténerek is komponensek
- A konténerek további konténereket tartalmazhatnak: egymásba ágyazás
- A Frame is egy konténer
- Egy másik gyakran használt: Panel
  - Egyebek: Window, Dialog, ScrollPane, Applet
- Együvé tartozó képernyőelemek összefogására

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

A konténer tervezési minta (Container design pattern)



Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Panel

- Egy konténer (egyben komponens)
- Igazándiból nem is látszik, hogy ott van

Panel, mint komponens

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     add(new Panel());
  }
  public static void main(String args[]){
    Hello hello = new Hello();
    hello.setSize(100,200);
    hello.setVisible(true);
  }
}
```

Panel, mint konténer

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     Panel panel = new Panel();
     panel.add(new Button("Szia"));
     panel.add(new Button("Hello"));
     panel.add(new Button("Salut"));
     add(panel);
  }
  public static void main(String args[]){...}
}
```

Komponensek elhelyezése

- Eddig csak létrehoztuk a komponenseket és belepakoltuk egy konténerbe
- Nem mondtuk meg, hogy mekkora legyen, és hova kerüljön
- Átméretezésnél automatikusan változtak/mozogtak a komponensek

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

Explicit elhelyezés

- Egy felhasználói felület elkészítésénél az egyik legnehezebb feladat az elhelyezés...
- ... Aminek nagy része az átméretezés követése
- Explicit programozással ugyan megoldható, de az nagyon macerás, sokat kell számolgatni
- Megoldás Java-ban: Layout Manager

Layout Manager

- A programozónak nem kell foglalkozni az elhelyezés részleteivel
- Magas szinten (kellően absztrakt nyelven) fogalmazhatja meg az igényeit
  - mi hova kerüljön
  - mi mekkora legyen
- Nem kötelező Layout Manager-t használni, de tényleg könnyebb azzal...

Layout manager - elhelyezési stratégia

- Egy konténeren belül hogyan kell a komponenseket elhelyezni
- Több predefinit LM használható
  - BorderLayout
  - FlowLayout
  - GridLayout
  - CardLayout
  - GridBagLayout (ehhez már gyakorlat kell)
  - egyéb (lásd Swing)
- Írhatunk sajátot is (ehhez nagy gyakorlat kell)

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

FlowLayout

- A komponenseket **folyamatosan** kell elhelyezni
- Meg kell kérdezni a komponensektől, hogy mekkorák „szeretnének lenni”
- Amíg férnek egymás mellé...
- Ha nem, akkor következő sorban folytatni
  - alapértelmezésben középre igazítva a sorban
- Ami sehogy sem fér, az nem jelenik meg rendesen

FlowLayout - Panel

- A Panel konténerekben az alapértelmezett Layout Manager a FlowLayout
- A példánk is ilyet mutatott be
- Figyeljük meg, hogyan viselkedik átméretezésnél

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

Frame - FlowLayout?

- A Frame konténereknél az alapértelmezett **nem** a FlowLayout
- De ott is be lehet állítani, hogy egy bizonyos Frame objektumnak az legyen a Layout Manager-e

```
setLayout(new FlowLayout());
```

Frame, de nem FlowLayout

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     add(new Button("Szia"));
     add(new Button("Hello"));
     add(new Button("Salut"));
  }
  public static void main(String args[]){...}
}
```

Frame + FlowLayout

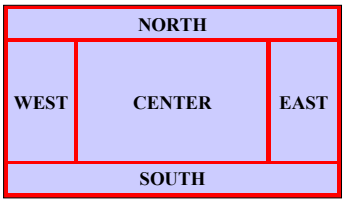
```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     setLayout(new FlowLayout());
     add(new Button("Szia"));
     add(new Button("Hello"));
     add(new Button("Salut"));
  }
  public static void main(String args[]){...}
}
```

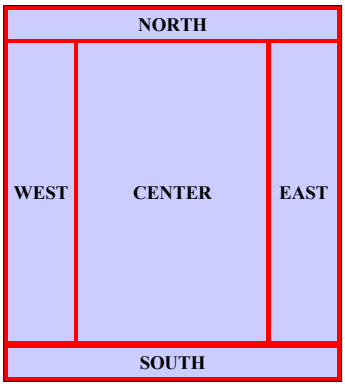
Java tutorial

Copyright © 2000-2002, Kozsik Tamás

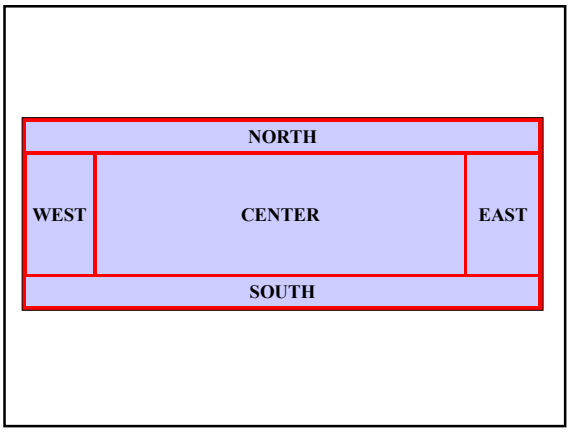
Border Layout

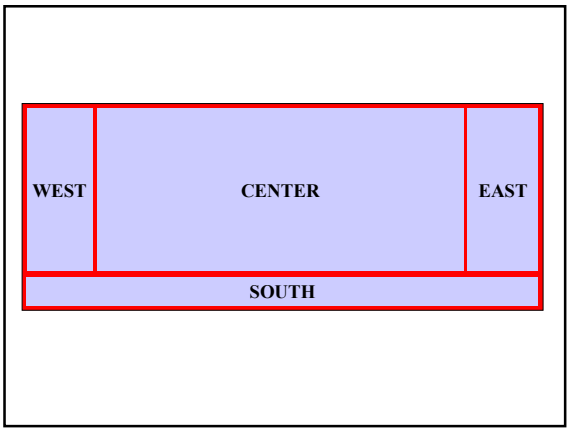
- A konténer széléhez történő igazítás
- Elhelyezés az égtájaknak megfelelően
- Legfeljebb öt komponens helyezhető el

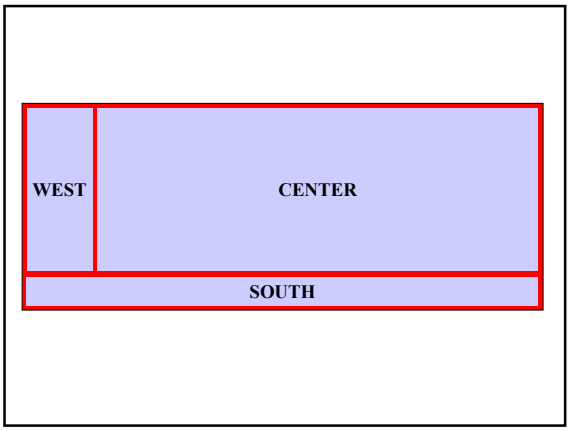


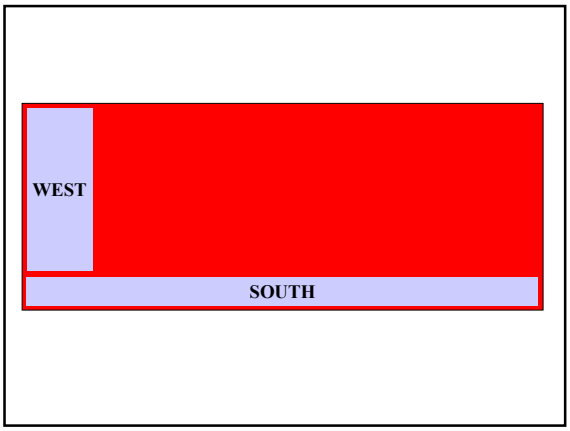


PDF created with FinePrint pdfFactory trial version http://www.fineprint.com









Java tutorial

Copyright © 2000-2002, Kozsik Tamás

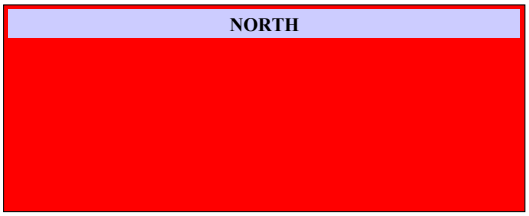
Például öt nyomógomb

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
    super("Hello");
    add(new Button("Fent"),     BorderLayout.NORTH);
    add(new Button("Lent"),     BorderLayout.SOUTH);
    add(new Button("Balra"),    BorderLayout.WEST);
    add(new Button("Jobbra"),   BorderLayout.EAST);
    add(new Button("Középen"), BorderLayout.CENTER);
  }
  public static void main(String args[]){...}
}
```


PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

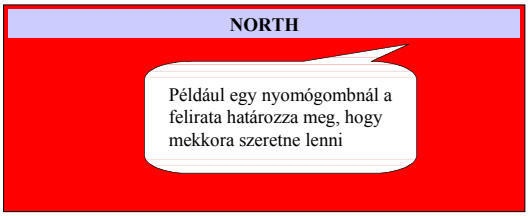
Az elhelyezés sorrendje és módja

- Ha van, akit északra tettek, a BorderLayout elhelyező objektum lekérdezi, milyen magas szeretne lenni, és olyan magas lesz, északon
- Széltében meg akkora, hogy kitöltse a konténert



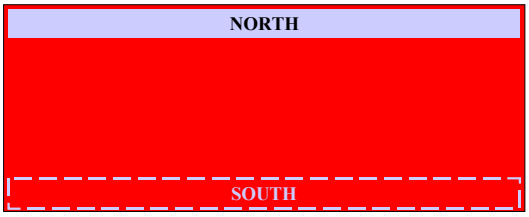
Az elhelyezés sorrendje és módja

- Ha van, akit északra tettek, a BorderLayout elhelyező objektum lekérdezi, milyen magas szeretne lenni, és olyan magas lesz, északon
- Széltében meg akkora, hogy kitöltse a konténert



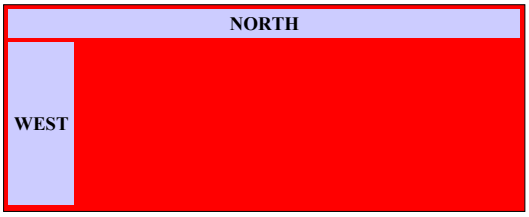
Az elhelyezés sorrendje és módja

- Ha van, akit délre tettek, ugyanezen stratégiával lesz elhelyezve, csak délen
- Tegyük fel, hogy most nincs senki délen:



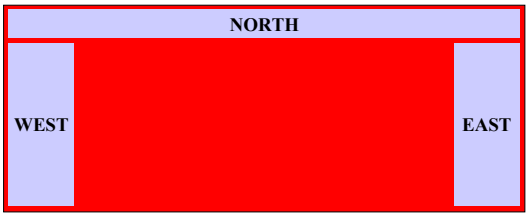
Az elhelyezés sorrendje és módja

- Ha van, akit nyugatra tettek, lekérdezzük, milyen széles szeretne lenni, és olyan lesz
- A magassága: amilyen magas csak lehet az észak és dél figyelembe vételével



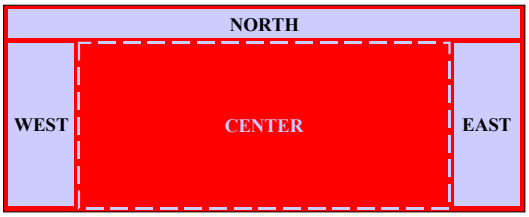
Az elhelyezés sorrendje és módja

- Ha van, akit keletre tettek, ugyanezzel a stratégiával lesz elhelyezve, csak keletre



Az elhelyezés sorrendje és módja

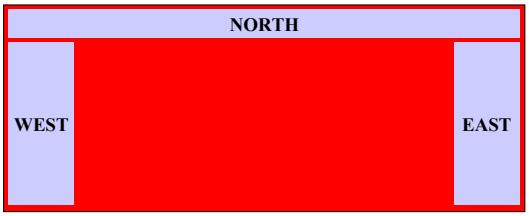
- Végül a középre tett komponens, ha van olyan, akkor kitölti a rendelkezésre álló maradék területet.



PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

Az elhelyezés sorrendje és módja

- Végül a középre tett komponens, ha van olyan, akkor kitölti a rendelkezésre álló maradék területet.
- A példánkban most ne legyen center komponens



BorderLayout beállítása

- A Frame-nél a BorderLayout az alapértelmezett
- Más konténernél a setLayout metódus meghívásával kell beállítanunk
- Például egy panelnél:

```
Panel panel = new Panel();
panel.setLayout( new BorderLayout() );
panel.add(   new Button("Bal"),
             BorderLayout.WEST    );
...
```

Visszatérve egy korábbi programra

```
import java.awt.*;
class Hello extends Frame {
  public Hello(){
     super("Hello");
     add(new Button("Szia"));
     add(new Button("Hello"));
     add(new Button("Salut"));
  }
  public static void main(String args[]){...}
}
```

- Az egy paraméteres **`add()`** a **`CENTER`**-be tesz

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Grid Layout Manager

- Ha ugyanakkora komponenseket akarok elhelyezni egy franciakockás rácsban

```
setLayout( new GridLayout(3,0) );
```

- Meg kell adni, hogy a rácsnak hány sora **vagy** hány oszlopa legyen
- Utána az egy paraméteres add() használható
- A rács méretét a másik dimenzióban a komponensek száma határozza meg
- Például a fent létrehozott elrendező objektum három sort alakít majd ki...

GridLayout konstruktora

- Két paramétert vár
- Ha az első paraméternek pozitív számot adunk meg, akkor annyi sort alakít majd ki. Ilyenkor a második paraméter indifferens.
- Ha az első paraméter nulla, akkor a másodiknak kell pozitívnak lennie, és ez a kialakítandó oszlopok számát adja meg

```
setLayout( new GridLayout(3,0) );
setLayout( new GridLayout(3,4) );
setLayout( new GridLayout(3,1000) );
```


PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

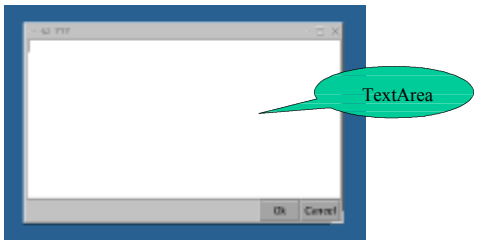
Java tutorial

Copyright © 2000-2002, Kozsik Tamás

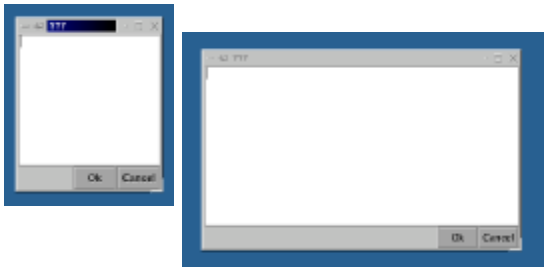
Feladat

- Készítsünk nyomógombokból egy telefonbillentyűzetet, amin 12 gomb található: a számok 1-től 9-ig, valamint a
  *
  0
  #
  karakterek. Minden gomb legyen ugyanakkora, és legyenek egy 4 soros és három oszlopos rácsban!

Bonyolultabb képernyőkép



Probléma



Megoldás: Konténerek és Layout Manager-ek egymásba ágyazása

Frame
BorderLayout

Panel p1
BorderLayout

Panel p2
GridLayout

A megoldó programkód

```
import java.awt.*;
class OC extends Frame {
  public OC(){
    add(new TextArea(), BorderLayout.CENTER);
    Panel p1 = new Panel();
    add(p1,BorderLayout.SOUTH);
    p1.setLayout(new BorderLayout());
    Panel p2 = new Panel();
    p1.add(p2,BorderLayout.EAST);
    p2.setLayout(new GridLayout(1,0));
    p2.add(new Button("Ok"));
    p2.add(new Button("Cancel"));
  }
  public static void main( String[] args ){
    OC oc = new OC();
    oc.pack();
    oc.setVisible(true);
  }
}
```


PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

Feladat

- Próbáljuk ki az előbbi programot, de kis módosításokkal: írjunk egy egyszerű szövegszerkesztő programot
  - A két nyomógomb: Save és Load legyen
  - Legyen egy TextField is, amibe a szerkesztendő fájlnevet kell megadni. Ezt egy címke jelezze előtte.

Feladat

- Tervezzünk meg egy Chat programot, melyben egymás alatt két, egyforma méretű TextArea található, valamint alattuk egy, az ablak szélességét felvevő, „Send” feliratú nyomógomb.

Házi feladat

- A telefonos programunkat alakítsuk úgy át, hogy a Frame-en belül a telefonos panel mindig a bal felső sarokban helyezkedjen el, és mindig ugyanakkora legyen. (Amekkora szeretne lenni...)
- Figyeljük meg, hogy hogyan viselkedik a telefonos panel, ha egész kicsire vesszük le az ablakot!

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Csináljon végre valamit a program...

- Rendeljünk funkcionalitást a felületelemekhez
- Specifikáljuk, hogy mi történjen, amikor valaki piszkál egy felületelemet
- Esemény-orientált szemlélet
- Más nyelvekben / ablakozó rendszerekben gyakran: call-back eljárások

Mi történjen, ha...

- Hagyományos eseménymodell
  - A komponens eldönti, hogy mit akar csinálni, ha őt megpiszkálták
  - Ha mégsem, akkor értesíti az őt tartalmazó konténert
  - És így tovább...
  - Java 1.0
- **Új, fejlettebb eseménymodell: Java 1.1-től**

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

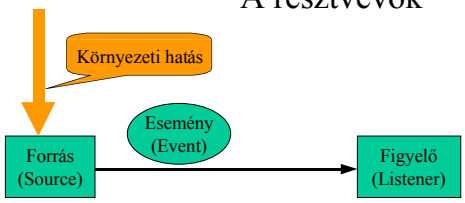
Eseménymodell

- Esemény forrása: amivel történhet valami
  - például egy felületelem, például egy nyomógomb
- Esemény: ami történt
  - például lenyomták
- Figyelő: ami reagálni tud az eseményre
  - tartalmazza a kódot, amit végre kell hajtani

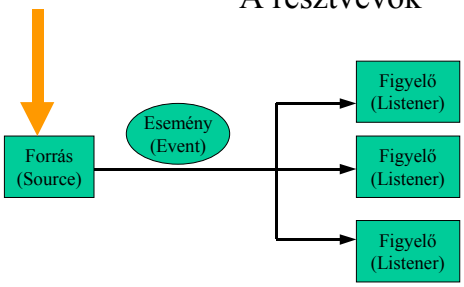
Java tutorial

Copyright © 2000-2002, Kozsik Tamás

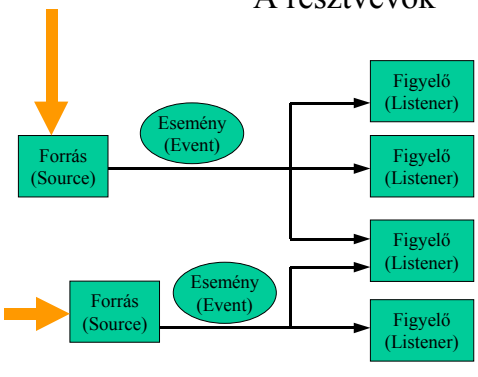
A résztvevők



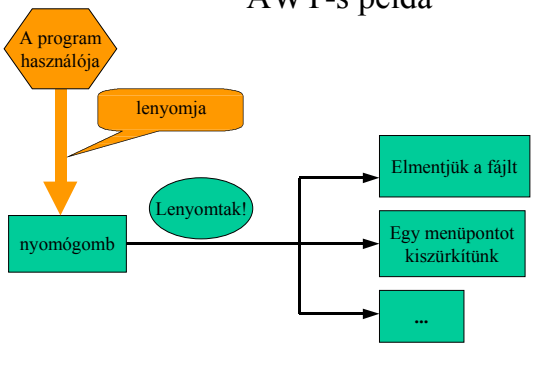
A résztvevők



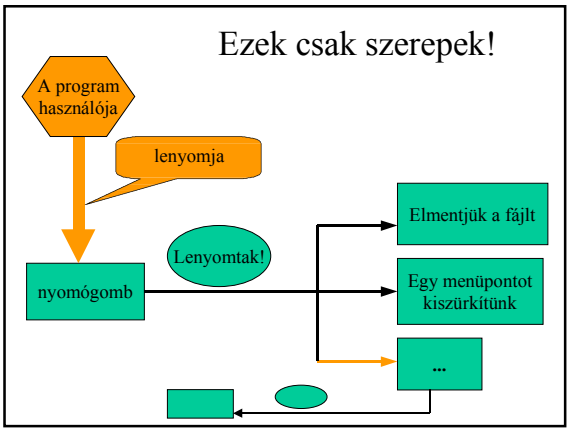
A résztvevők

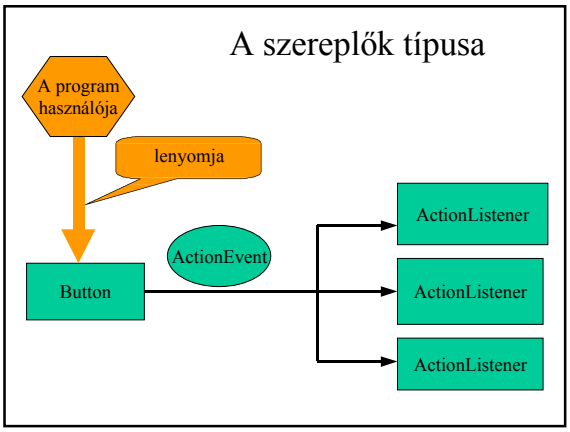


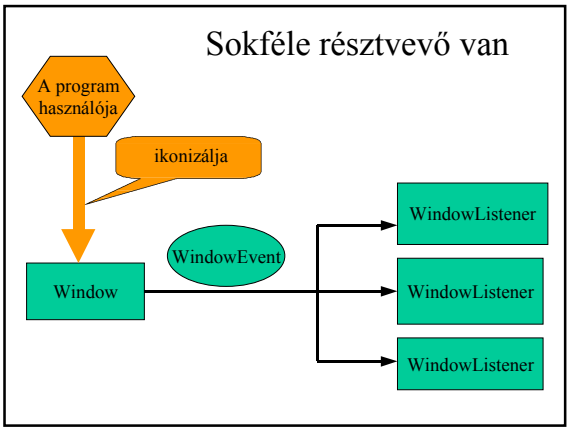
AWT-s példa

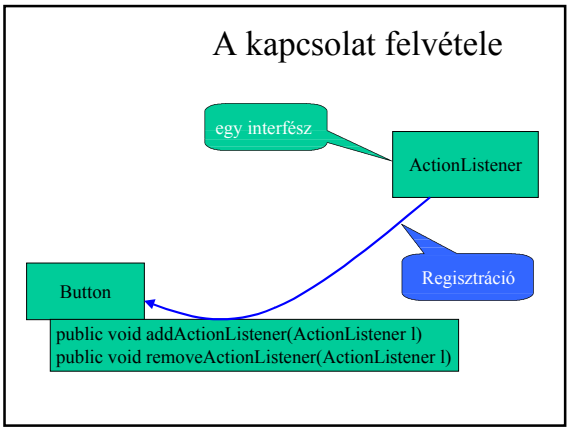


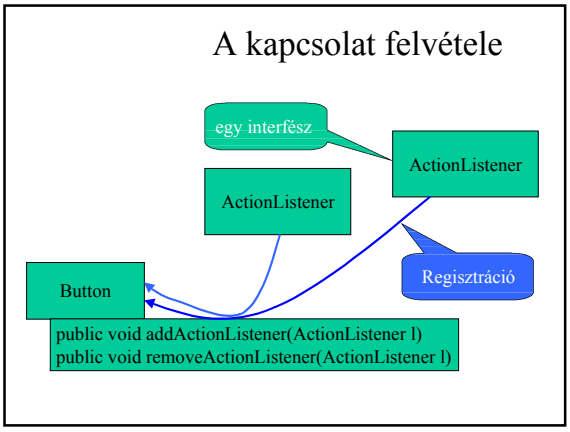
PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

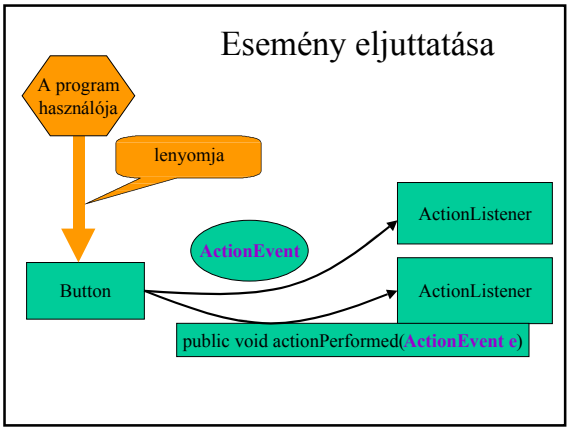






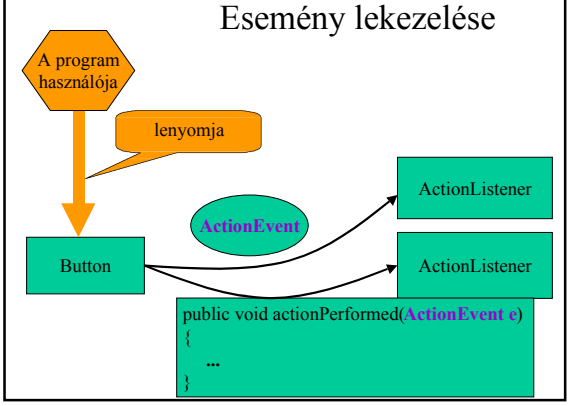






PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

Esemény lekezelése



Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Feladat

- Írjunk eseménykezelő osztályt, és egy példányát kapcsoljuk hozzá Chat program Send nyomógombjához. A figyelő reakciója az legyen, hogy kiírja a szabványos kimenetre, hogy lenyomták az általa figyelt gombot.
- Használd a **java.awt.event** csomagot!
- Próbálj meg létrehozni és a nyomógombhoz kapcsolni két példányt a figyelő osztályból

Megoldás

```
import java.awt.event.*;
public class Figyelo implements ActionListener {
  public void actionPerformed( ActionEvent e ){
    System.out.println("Lenyomták!");
  }
}
```

```
Button b = new Button("Send");
add(b, BorderLayout.SOUTH);
b.addActionListener( new Figyelo() );
b.addActionListener( new Figyelo() );
```

Feladat

- Lehet maga a Chat program is egy figyelő. Csinálja azt, hogy az alsó TextArea tartalmát hozzáfűzi a felsőéhez, és le is törli az alsót.
  - getText, setText, append
  - a TextArea-kból attribútumot kell csinálni, hogy a referenciák a metódushívások közben is megmaradjanak (hogy minden metódusból használhassuk őket)
- Aztán idővel a két másik figyelőt ki is lehet ám venni... :-)

Különálló osztály

- Sok esetben kényelmes, ha az eseménykezelőt nem különálló osztályként valósítjuk meg
- Direkt hozzáférés szükséges a komponensekhez
- Van, amikor az sem kényelmes, ha a Frame-be tesszük az eseménykezelést
- Beágyazott osztályok segítenek majd...

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

Feladat

- Írjuk meg a Chat programot úgy, hogy az eseménykezelés külön osztályban legyen.
- A szükséges információkat el kell juttatni az eseménykezelést végző objektumhoz...

Beágyazott osztályok

- A beágyazott osztályok segítségével kényelmesebb
- Névtelen osztályok
- Az információ megfelelő helyre történő eljuttatásával nem kell foglalkozni

Feladat

- Írd meg a Chat programot úgy, hogy névtelen osztályt használsz az eseménykezeléshez!

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Más események

- WindowEvent, KeyEvent, ItemEvent, MouseEvent, MouseMotionEvent, ComponentEvent, ContainerEvent, stb.
- Minden komponensnél kitalálható, hogy milyen eseményt tud kiváltani
  - add*Listener (addWindowListener, addKeyListener)
- Ablak becsukása: WindowEvent

WindowEvent, WindowListener

- WindowEvent: az ablakokkal kapcsolatos események összefoglaló neve
- WindowListener
  - windowActivated(WindowEvent e)
  - windowDeactivated(WindowEvent e)
  - windowOpened(WindowEvent e)
  - windowClosed(WindowEvent e)
  - **windowClosing(WindowEvent e)**
  - windowIconified(WindowEvent e)
  - windowDeiconified(WindowEvent e)

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

(Window)Adapter

- Ha egy WindowListener-t írunk, akkor meg kell valósítani 7 metódust
- Fölöslegesen macerás, mert sokszor csak egy metódus érdekel minket, a többi nem
  - a többinek úgyis üres törzset írnánk
- Használjunk helyette WindowAdapter-t
- Ez egy osztály, ami üres törzzsel valósítja meg a 7 metódust a WindowListener-ből
- Leszármaztatunk belőle, és ami kell, felüldefiniáljuk, a többit örököljük

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Feladat

- A Chat programot egészítsük ki úgy, hogy ki lehessen belőle lépni.
- System.exit(0);
- Figyelem! Hibaforrás: ha rosszul írjuk be a windowClosing nevét vagy paraméterezését, a fordító nem fog szólni, mert öröklődik az üres törzsű metódus...

Feladat

- Csináljuk azt is meg a Chat programban, hogy a felső TextArea-t ne lehessen szerkeszteni, és hogy egyből az alsó TextArea kapja meg az input fókuszt, amikor elindul a program, meg miután lenyomtuk a Send nyomógombot.

Feladat

- Fejezzük be a szövegszerkesztő programot! A Load és Save nyomógombhoz rendeljünk értelemszerű eseménykezelőket. A fájlnevet vegyük a TextField-ből. A kilépésre is adjunk lehetőséget.

Lokális változók elérése beágyazott osztályokból

- Az előző feladat során a komponenseket adattagként vettük fel
- Elég, ha lokális változók
  - az eseménykezelésért felelős névtelen osztályokból így is elérhetők
- Ilyenkor final módosítószóval kell deklarálni őket
- Próbáld ki!

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

```
...
public class SzovegSzerkeszto extends Frame {
  TextArea ta = new TextArea();
  TextField tf = new TextField();
  Button load = new Button("Load"),
         save = new Button("Save");
  public SzovegSzerkeszto(){
     ...
     save.addActionListener( new ActionListener(){
         public void actionPerformed( ActionEvent e ){
           try {
               PrintWriter w = new PrintWriter(
                           new FileWriter(tf.getText()) );
               w.print(ta.getText());
               w.close();
           } catch( IOException ioe ){
              System.err.println("Sikertelen mentés!");
           }
         }
       }
     );
  }
  ...
}
```

```
...
public class SzovegSzerkeszto extends Frame {
  public SzovegSzerkeszto(){
     final TextArea ta = new TextArea();
     final TextField tf = new TextField();
     Button load = new Button("Load"),
            save = new Button("Save");
     ...
     save.addActionListener( new ActionListener(){
         public void actionPerformed( ActionEvent e ){
           try {
               PrintWriter w = new PrintWriter(
                           new FileWriter(tf.getText()) );
               w.print(ta.getText());
               w.close();
           } catch( IOException ioe ){
              System.err.println("Sikertelen mentés!");
           }
         }
       }
     );
  }
  ...
}
```

Ugyanaz a figyelő több forráshoz

- Ugyanaz a figyelő több forráshoz is hozzárendelhető
- Bármelyik forrásban fellép az esemény, a figyelő megfelelő metódusa végrehajtódik
- Az átadott esemény objektum információt hordoz
- Például azt, hogy melyik forrásból származik: **`getSource()`** metódus

```
public void actionPerformed(ActionEvent e){
  if( e.getSource() == ... ) ...
```

Feladat

- A szövegszerkesztő programban a nyomógombok lekezelését ne névtelen osztállyal valósítsuk meg, hanem a Frame kapja meg az eseményeket. Döntsön a getSource() alapján arról, hogy mit kell csinálnia.

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Menük

- Minden Frame objektumhoz hozzákapcsolható (legfeljebb egy) menüsor: **MenuBar**
- A menüsorba menüket lehet felvenni: **Menu**
- A menük menüpontokat tartalmazhatnak: **MenuItem**
- A menüpontok között lehetnek speciálisak
  - elválasztó jelek (Separator)
  - almenük
  - ki/bekacsolhatós menüpontok (CheckboxMenuItem)

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

Menüsor

- Csak Frame-hez rendelhető hozzá
- Applethez például nem!
  - Pop-up menük használhatók appleteknél is...
  - Swing-ben már lehet appletekhez is menüt rendelni, ott sokkal több lehetőség van...

```
MenuBar mb = new MenuBar();
setMenuBar(mb);
```

Menü

- A menübárhoz lehet hozzávenni

```
Menu m = new Menu("File");
mb.add(m);
m = new Menu("Edit");
mb.add(m);
```

- A help menü speciálisan kezelhető
  - ez bizonyos rendszereken a menüsor jobb szélén jelenik majd meg (Windows-on nem)

```
m = new Menu("Help");
mb.setHelpMenu(m);
```

Menüpont

- A menükhöz lehet hozzávenni

```
MenuItem mi = new MenuItem("Load");
m.add(mi);
mi = new MenuItem("Save");
m.add(mi)
m.addSeparator();
```

- A menüpontok szintén ActionEvent-eket generálnak, amikor kiválasztják őket

```
mi.addActionListener( ... )
```

Feladat

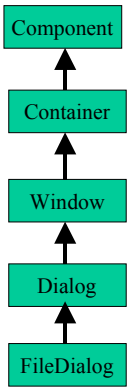
- Írjuk újra a szövegszerkesztő programot úgy, hogy ne nyomógombokkal, hanem menükkel lehessen vezérelni.
- A menüsor alatt csak egy TextArea legyen
- Legyen File menü (Load, Save, elválasztó jel, Exit), legyen Edit menü (üres) és legyen Help menü (szintén üres)
- Egyelőre a Load és a Save ne csináljon semmit, de az Exit már működjön...

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Dialógus ablak

- Konténer (Container)
- Külön ablakként jelenik meg (Window)
- Speciális, előre elkészített dialógusablak a FileDialog
- Modális vagy sem



PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

FileDialog használata

- A konstruktorral létrehozzuk - paraméterként átadjuk a Frame-et, amihez képest modális
- Kirakás előtt beállítjuk, hogy Save vagy Load
- A **`show()`** metódussal kitesszük
- Amikor Ok-t vagy Cancel-t nyomtak, eltűnik, és megint a Frame-mel dolgozhatunk
- A **`getFile()`** és **`getDirectory()`** metódussal lekérdezhető a kiválasztott fájl és útvonal
  - Cancel esetén a **`getFile()`** eredménye **`null`**

Feladat

- A szövegszerkesztő programban az elmenteni, illetve betölteni való fájl nevét egy FileDialog dialógusablakból szerezzük meg!
- A FileDialog objektumot csak egyszer hozzuk létre, eltároljuk a szövegszerkesztő egy adattagjában, és amikor kell, kitesszük
- A fájl megnyitásához egy java.io.File objektumot használhatunk, ami egy fájlnevet reprezentál
- A menüpontok közötti választás az ActionEvent **`getActionCommand()`** metódusával mehet

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Almenük

- A menük használhatók menüpontként is
- Így lehet almenüt készíteni

```
public class Menu extends MenuItem
```

```
Menu m = new Menu("Edit");
Menu am = new Menu("Settings");
m.add(am);
```

CheckboxMenuItem

- Kijelölős menü
- ItemEvent generálódik, ha kijelölik
- Nem lehet CheckboxGroup-ba szervezni
  - "radio" jelleg nem lehet, mint a Checkbox-nál
- Le lehet kérdezni, hogy bekapcsolt állapotban van-e, vagy sem

```
Menu am = new Menu("Settings");
CheckboxMenuItem mi =
  new CheckboxMenuItem("Monospaced font");
mi.addItemListener(...);
am.add(mi);
```

Feladat

- A szövegszerkesztő program Edit menüjében helyezzünk el egy Settings almenüt, amelyben - egyelőre - csak egy menüpont legyen, amivel ki/be kapcsolhatjuk, hogy monospaced fonttal jelenjen-e meg a TextArea tartalma
- Használd a java.awt.Font osztályt és a TextArea **`setFont()`** metódusát
- Az ItemEvent **`getStateChange()`** metódusa hasznos lehet
- Ne kérdezzük le túl korán a TextArea fontját!

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Clipboard használata

- Saját is definiálható, de elérhető a programok közötti, közös vágólap is

```
getToolkit().getSystemClipboard();
```

- A TextArea automatikusan kezeli
  Pl. Windows alatt Ctrl-X, Ctrl-C, Ctrl-V
- Stringek átvitelére jól használható:

```
String str = (String)clipboard.getContents(null).
        getTransferData(DataFlavor.stringFlavor);

StringSelection ss = new StringSelection(str);
clipboard.setContents(ss,ss);
```

- java.awt.datatransfer, Transferable, Clipboard

Feladat

- Egészítsük ki a szövegszerkesztő programot Cut, Copy és Paste menüpontokkal, melyek a programok közötti vágólapot kezelik!
- Vigyázat: valamilyen misztikus ok miatt az első nem, csak a második működik majd jól

```
//String str = ta.getSelectedText();
String str =
  new String(ta.getSelectedText());
StringSelection ss = new StringSelection(str);
```

További lehetőségekre példa

- Kurzor beállítása
  - Component osztályban `setCursor(Cursor)`
  - `new Cursor( Cursor.HAND_CURSOR )`
- Ikon beállítása
  - Frame osztályban `setIconImage(Image)`
  - `toolkit.createImage(fájlnév)`
- Szín beállítása
  - Component osztályban `setForeground(Color)` és `setBackground(Color)`
  - `Color.red` `new Color(140,180,200)`

Feladat

- A Chat programban az alsó TextArea háttérszíne legyen fekete, a betűk színe legyen sárga.
- A Send nyomógomb felett a kurzor legyen kéz alakú.
- Használjuk a chat.gif fájlt a Chat program ikonjaként.

Rajzolás

- A Component osztályban `paint()` metódus

```
public void paint( Graphics g )
```

- Megadja, hogy hogyan rajzolódjon ki a komponens
- Ezt felüldefiniálva rajzolhatunk
- A Graphics objektumon keresztül
- Jellemző példa: Applet vagy Canvas

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

Példa

```
import java.awt.*;
class Vonal extends Frame {
  public void paint( Graphics g ){
    g.drawLine(20,30,40,50);
  }
  public static void main(String args[]){
    Frame f = new Vonal();
    f.setSize(100,200);
    f.setVisible(true);
  }
}
```

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Graphics

- A komponens látható részét reprezentálja
- Képernyőpontok mátrixa
- A komponens bal felső sarka a 0,0 koordinátájú pont, ehhez képest lehet pozícionálni

Vonalak, téglalapok, ellipszisek

- A Graphics osztályban definiált műveletek

```
void drawLine(int x1, int y1, int x2, int y2)
void drawRect(int x, int y, int width, int height)
void drawOval(int x, int y, int width, int height)
```

- Ugyanezek kitöltéssel

```
void fillRect(int x, int y, int width, int height)
void fillOval(int x, int y, int width, int height)
```

Trükkösebb dolgok

- Bonyolultabb alakzatok (draw és fill)

```
void drawPolygon(int[] xPoints, int[] yPoints,
                 int nPoints)
void drawRoundRect(int x, int y, int width, int height,
                   int arcWidth, int arcHeight)
void draw3DRect(int x, int y, int width, int height,
                boolean raised)
void drawArc(int x, int y, int width, int height,
             int startAngle, int arcAngle)
```

- Képek, szövegek

```
void drawString(String str, int x, int y)
void drawImage(Image i, int x, int y, ImageObserver o)
```

Beállítások

- A rajzolás előtt beállítható
  - a rajzolás színe

    `graphics.setColor(Color)`
  - a használt font (drawString)

    `graphics.setFont(Font)`

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

Téglalapok, ellipszisek mérete

- Ha egy 10 x 10 -es négyzetet akarunk rajzolni, akkor

```
g.drawRect(0,0,9,9)
```

  – mert a 9 azt jelenti, hogy 9-szer jobbra/lefelé kell menni
- Ha viszont kitöltött alakzatot csinálunk, akkor nem kell egyet levonni

Feladat

- Rajzolj pálcikaemberkét.

- Definiáld felül a Frame paint metódusát

Mire rajzoljunk?

- Jellemző, hogy egy Applet egy rajzolt komponens
- Ha egy alkalmazásban/appletben csak egy komponens az, ami rajzolt, akkor azt a komponenst válasszuk Canvas-nek

Feladat

- Az emberkés programot alakítsuk úgy ki, hogy alul legyen egy nyomógomb.
- A rajzot egy Canvas objektumon helyezzük el...

Java tutorial

Copyright © 2000-2002, Kozsik Tamás

Ha változtatni akarunk a rajzon

- A komponens automatikusan újrarajzolódik, ha elveszünk előle valamit, ami eddig egy részét eltakarta
  – vagy pl. ha ikonizált állapotból visszahozzuk
- Ilyenkor a paint() magától meghívódik
- Ha mi akarunk valamit változtatni, és ezért újra akarjuk rajzolni a képet, ne a paint() metódust hívjuk, hanem a repaint()-et

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com

repaint()

```
public void repaint()
```

- Ez letörli a komponensünk képét, majd meghívja a paint()-et
- Egész pontosan a repaint() az update()-et hívja, az törli le a komponenst és hívja meg a paint()-et
  - ha nem akarunk törölni, vagy offscreen rajzolást akarunk csinálni, akkor itt lehet közbeavatkozni

Feladat

- Az emberkés programban a Fel/Le nyomógomb egy logikai változót billentgessen. A Canvas objektumunkban a paint() ettől a logikai változótól függően az emberke kezét felfelé vagy lefelé rajzolja meg
- Tehát a nyomógomb hatására az emberke mozgatja majd a kezét

Graphics2D

- Amit a paint megkap, az nem egyszerűen egy Graphics, hanem egy Graphics2D
- Ez a Graphics leszármazottja
- Sokkal többet tud
  - forgatás, eltolás, stb...
- Nyugodtan konvertáljuk...

```
public void paint( Graphics g ){
  ((Graphics2D)g).rotate(0.42);
  ...
```

Feladat

- Az emberke elforgatva és eltolva jelenjen meg...

Platform-függő információk

- A Toolkit objektumon keresztül
  `getToolkit().getFontList()`
  `getToolkit().getScreenSize()`
- Nem csak a komponenseken keresztül juthatunk hozzá Toolkit objektumhoz
  `Toolkit.getDefaultToolkit()`

Feladat

- A szövegszerkesztő program kezdetben foglalja el az egész képernyőt.

PDF created with FinePrint pdfFactory trial version http://www.fineprint.com