Programozási technológia

Grafikus felhasználói felület

Dr. Szendrei Rudolf
ELTE Informatikai Kar
2020.

Grafikus felhasználói felület

- Két csomag elemeiből lehet a felületet elkészíteni
  - `awt` ,,heavy weight" komponensek
  - `swing` ,,light weight" komponensek

- Mi hogyan használjuk?
  - `awt` eseménykezelés, speciális elemek
  - `swing` ,,látható" komponensek

GUI – Swing komponensek

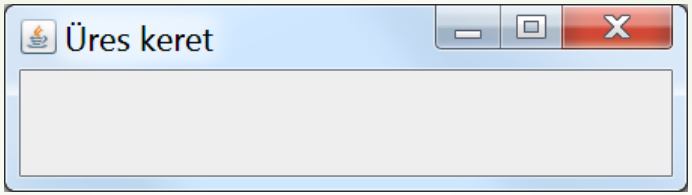
- Közös ősosztály: `JComponent`
- Származatott osztályok:
  - `JFrame` keret
  - `JLabel` címke
  - `JButton` gomb
  - `JPanel` panel
  - `JComboBox` legördülő menü
  - `JList` lista
  - `JSlider` csúszka
  - `JSpinner` spin vezérlő
  - `JTable` táblázat

GUI – ablak létrehozása

```
import javax.swing.JFrame;
public class MyFrame extends JFrame {
  public MyFrame() {
    setTitle("Üres keret");
    setSize(200, 60);
    setVisible(true);
  }
  public static void main(String[] args) { new MyFrame(); }
}
```

Program futtatása

A program futtatásakor a következő ablak jelenik meg:



- A bezárás gombra kattintva a program ablaka bezárul, de a program nem fejeződik be.
  - parancssorban nem kapjuk vissza a promptot
  - a futtatási konzol ablak mutatja, hogy a program fut
- A bezárás gomb megnyomásával csak a keret tűnik el
- Program leállítása:
  - parancssorból `Ctrl+C` lenyomásával
  - NetBeans környezetben a `Run` menü `Stop Build/Run` pontjával

GUI – ablak bezárása

- Az alapértelmezett megvalósítás szerint bezáráskor csak eltűnik a keret.

- Lehetséges megoldások:
  - A bezáráskori viselkedés megadása
  - A bezárás eseményének figyelése eseménykezelő hozzárendelésével

GUI – ablak bezárása

A keret automatikus bezáró műveletének előírása:

- `setDefaultCloseOperation` művelet segítségével
- Paramétere: `WindowConstants`-beli konstans

```
import javax.swing.JFrame;
public class MyFrame extends JFrame {
  public MyFrame() {
    setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    setTitle("Üres keret");
    setSize(200, 60);
    setVisible(true);
  }
  public static void main(String[] args){ new MyFrame(); }
}
```

GUI – ablak bezárása

Program leállítása a keret bezárásakor – eseménykezelővel

- A kezelőt a `WindowAdapter` osztályból kell származtatni, és az `addWindowListener` művelettel kell a kerethez rendelni.
- A kezelő egyetlen feladata az ablak bezárásának kezelése, amit a `windowClosing` művelet átdefiniálásával tehetünk meg.
- A művelet paramétere a bezárást okozó `WindowEvent` típusú esemény, amit mi nem használunk.
- A műveletben ki kell lépni a programból.
- Ha kilépéskor egyéb tevékenységet is akarunk végezni, akkor mindenképpen ezt az utat kell követni.

GUI – ablak bezárása

Program leállítása a keret bezárásakor – eseménykezelővel

```
import javax.swing.*;
import java.awt.event.*;

public class MyFrame extends JFrame {
  public MyFrame() {
    addWindowListener(new WindowAdapter() {
      @Override
      public void windowClosing(WindowEvent e) { System.exit(0); }
    });
    setTitle("Üres keret");
    setSize(200, 60);
    setVisible(true);
  }
}
```

GUI – ablak bezárása

- Az ,,implicit" objektum helyett az objektumot létrehozhatjuk az osztályon belül. Az alábbi megoldással egy névtelen osztályt hozunk létre és példányosítunk.

```
public class MyFrame extends JFrame {
  private WindowAdapter exit = new WindowAdapter() {
    @Override
    public void windowClosing(WindowEvent e){ System.exit(0);}
  });
  public MyFrame() {
    addWindowListener(exit);
    ...
  }
}
```

GUI – ablak bezárása

- Az eseménykezelő lehet külön osztály is, így azt több keret esetében újra felhasználhatjuk majd

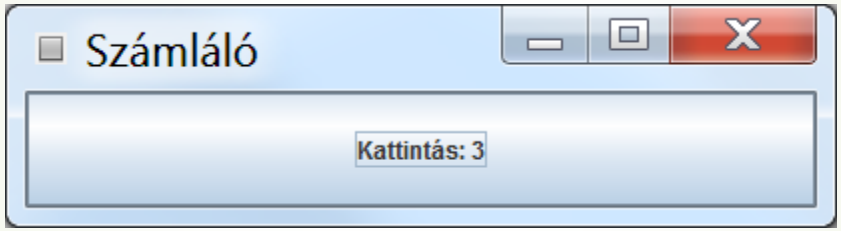
```
public class ExitClass extends WindowAdapter() {
  @Override
  public void windowClosing(WindowEvent e){ System.exit(0);}
}

public class MyFrame extends JFrame {
  public MyFrame() {
    addWindowListener(new ExitClass());
    ...
  }
}
```

GUI események kezelése

- A felhasználói akciók a felületi elemeken eseményt váltanak ki: `ActionEvent`
- Ezeket kell kezelni az elemekkel úgy, hogy eseményfigyelőt rendelünk hozzájuk (`addActionListener` vagy `Action` objektum beágyazása)
- Események figyelése: `ActionListener` interfész
- Egyetlen művelet: `actionPerformed`, aminek paramétere az esemény

Példa – Kattintások számlálása

- Elhelyezünk egy gombot a keretben, amely figyeli a kattintásokat, és megjeleníti a kattintások számát.
- Ezen kívül a keret ikonját is megváltoztatjuk.



Példa – Kattintások számlálása

- A számláló gombot a keret „adatterületére” helyezzük el , amit a `getContentPane()` művelet ad meg
- Ehhez az `add` művelettel vehetünk hozzá elemet.
- A gombot a `CounterButton` osztály segítségével valósítjuk meg
- Ezt a `JButton` osztályból kell származtatni, és ki kell egészíteni eseménykezeléssel is.
- Az osztálynak meg kell valósítania ezért az `ActionListener` interfészt.

Példa – Kattintások számlálása

- Az ikont a `setIconImage` művelettel adhatjuk meg, aminek a paramétere az ikon elérési útja, amit a megfelelő eszközön (`Toolkit`) keresztül rendelünk a kerethez.

```
setIconImage(Toolkit.getDefaultTookit().getImage("x.png"));
```

- Így a `jar` állományból nem töltődik be az ikon. Ehhez a fájl nevét URL-ként kell megadni. A PATH a projekt gyökeréből indul (Pl.: icons/main.png)

```
import java.net.URL;
URL url = getClass().getClassLoader().getResource("PATH");
setIconImage(Toolkit.getDefaultTookit().getImage(url));
```

Példa – Kattintások számlálása

Főprogram megvalósítása

```
import java.awt.*;
import javax.swing.*;
import java.net.URL;
public class MyFrame extends JFrame {
  public MyFrame() {
    addWindowListener(new ExitClass());
    setTitle("Számláló");
    setSize(400, 100);
    getContentPane().setLayout(new FlowLayout());
    getContentPane().add(new CounterButton());
    getContentPane().add(new ExitButton()); // később hozzuk létre
    URL url = getClass().getClassLoader().getResource("icon.png");
    setIconImage(Toolkit.getDefaultToolkit().getImage(url));
    setVisible(true);
  }
}
```

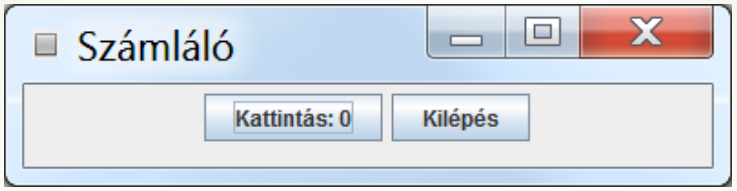
Példa – Kattintások számlálása

Kattintás számláló gomb osztályának megvalósítása

```
import java.awt.event.*;
import javax.swing.*;
public class CounterButton extends JButton
                           implements ActionListener {
  private int numClicks = 0;
  public CounterButton(){
    setText("Kattintás: 0");
    addActionListener(this);
  }
  @Override
  public void actionPerformed(ActionEvent e){
    setText("Kattintás: " + ++numClicks);
  }
}
```

Példa – Kattintások számlálása

- Egészítsük ki az előző programot úgy, hogy a keretben elhelyezünk egy kilépésre lehetőséget adó gombot is



- El kell készítenünk az előzőekhez hasonlóan a kilépésre szolgáló gombot, ahol az esemény kezelése a program befejezését jelenti
- A kereten most nem egy, hanem két gombot kell elhelyeznünk, megadva azok elhelyezkedési módját (`setLayout`)
- A legegyszerűbb lehetőség a `FlowLayout`, amelybe folytonosan helyezhetünk el elemeket

Kilépés egységes kezelése

- A programból való kilépéskor ugyanaz következzen be, bárhogy is lépünk ki a programból (még a mostani egyszerű esetben is)
- Célszerű az ablak bezárásához és a kilépés gomb lenyomásához ugyanazt az eljárást rendelnünk (`exit`)
- A bezárásához eseménykezelőt rendelünk, amelyben ezt hívjuk meg, és a gomb megnyomásakor is ezt hívjuk meg
- A kilépés gombot most másként hozzuk létre
- Egy gomb konstruktorának paramétere lehet egy `Action` típusú objektum
- Ebben meg lehet adni az esemény kezelését, illetve a gomb további tulajdonságait
- Az `Action` interfészt megvalósító `AbstractAction` osztály, alapértelmezésben üres megvalósításokkal látja el (csak a számunkra ,,érdekeseket" kell megvalósítanunk)

Példa – Kattintások számlálása

Kilépés gomb osztályának megvalósítása

```
import java.awt.event.*;
import javax.swing.*;
public class ExitButton extends JButton{
  public ExitButton() {
    super(new AbstractAction("Kilépés"){
      @Override
      public void actionPerformed(ActionEvent e){
        new ExitClass().windowClosing(null);
      }
    });
  }
}
```

GUI elemek elhelyezési módjai

- `FlowLayout` — az elemeket folyamatosan helyezi el egy sorban
- `BorderLayout` — az elemeket 5 helyre tudja elhelyezni: CENTER / SOUTH / NORTH / WEST / EAST
- `GridLayout` — az elemek folyamatos elhelyezése megadott méretű rácsban, ugyanolyan méretben
- `GridBagLayout` — az elemek elhelyezése rácsban, az elemek kiterjedése több egység is lehet
- `BoxLayout` — elemek elhelyezése adott méretben, adott távolságra vízszintes vagy függőleges irányban
- `CardLayout` — kártyapakli elhelyezés (csak egy elem látszik, ami változtatható)
- `OverlayLayout` — elemek egymásra helyezése

https://docs.oracle.com/javase/tutorial/uiswing/layout/visual.html