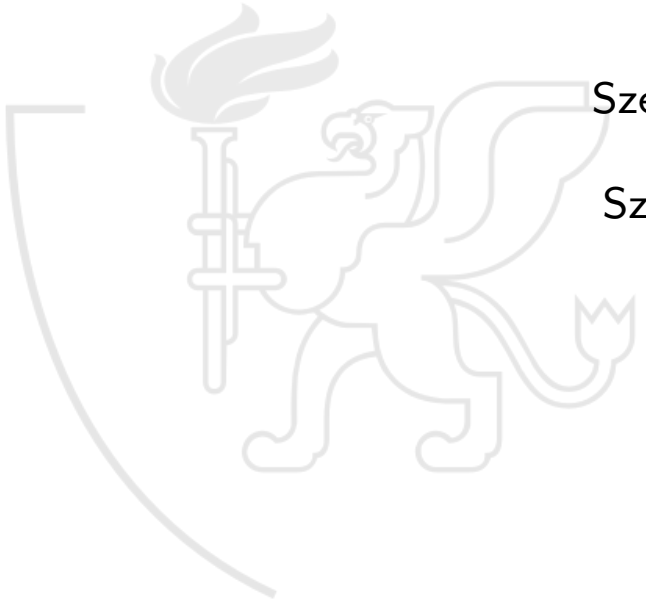
Programozási Ismeretek

Rendezések

Dr. Gergely Tamás
Dr. Ferenc Rudolf, Dr. Jász Judit, Dr. Kiss Ákos

Szegedi Tudományegyetem
Informatikai Intézet
Szoftverfejlesztés Tanszék



2024

(v0214)

Tartalom

Tartalom

Grafikus programok

- Teljesen más gondolkodásmódot igényel, mint a „hagyományos” imperatív programozás:
  - a programnak nem egy fix belépési pontja van, ami után az objektumok megadott sorrendben üzengetnek egymásnak, hanem
  - eseményvezérelten működik, az egyes részfeladatok a felhasználói tevékenység függvényében „bármikor” elindíthatóak
  - és virtuálisan akár egyszerre hajtódhatnak végre.
- Jól illeszkedik az objektumorientált gondolkodásmódhoz:
  - az egyes grafikus entitások objektumoknak feleltethetőek meg, amelyek
  - az adott eseményeket a megfelelő metódusokkal „önállóan” képesek lekezelni, miközben
  - más objektumok felé újabb üzeneteket küldhetnek és
  - a saját állapotukat is megváltoztathatják.

AWT

Abstract Window Toolkit

- Eredeti célkitűzés (Java 1.0)
  - olyan GUI könyvtár készítése, amely egyformán jól néz ki minden platformon
- Eredmény
  - AWT – Abstract Window Toolkit
  - olyan GUI könyvtár, amely egyformán rosszul néz ki minden platformon
  - ráadásul korlátozott
    - csak 4 font érhető el
    - nem érhetők el a speciálisabb GUI elemek
    - a programozási modellje is esetlen és nem objektum orientált
    - a rossz tervezés iskolapéldája (állítólag 1 hónap alatt készült el)
- Java 1.1
  - továbbfejlesztett AWT
  - objektum orientált programozási modell bevezetése
  - még mindig nem az igazi

JFC – Swing

Java Foundation Classes – Swing

- Java 1.2
  - JFC – Java Foundation Classes
  - Swing – a JFC GUI-val foglalkozó része
  - szépen megtervezett könyvtár
    - kár, hogy továbbra is függ az AWT-től

JFX

JavaFX

- Eredetileg a Swing helyére szánták
- Az 1-es verziókban (2008-tól) Java-hoz hasonló JavaFX szkriptnyelven lehetett progamozni (ami azután Java nyelvre fordult le)
- Java 2.0-tól (2011) a JavaFX egy külön osztálycsomag, közvetlenül Java-ból programozható
- Java 8-tól (2014) JavaFX 8 néven az Oracle Java SE része (a Java verzióval azonos számozással)

Tartalom

JFC komponensek

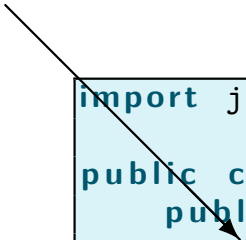
- Swing GUI komponensek
  - minden a nyomógomboktól a táblázatokig
- Külalak plug-in-ek (look-and-feel)
  - paraméterezhető külalak lehetősége
    - pl. az aktuális operációs rendszer ablakozójához igazodhat, de lehet akár saját egyedi is
  - Elérhetőség API
    - fogyatékkal élő felhasználók támogatása
- Java 2D API
  - 2D grafika, szöveg és képek kezelése
  - nyomtatás támogatása
- „Húzd és ejtsd” támogatása (Drag-and-drop)
  - adatok mozgatása „húzd és ejtsd” módon Java és natív alkalmazások között
- Internacionalizálás
  - különleges karakterek támogatása, pl. japán, kínai, koreai

Swing API

- Szükséges csomagok (általában)
  - `javax.swing.*`
  - `javax.swing.event.*`
  - `java.awt.*`
  - `java.awt.event.*`
- Minden Swing-es programnak kell legyen legalább egy legfelső szintű tárolója az alábbiak közül
  - `JFrame` – fő ablak
  - `JDialog` – másodlagos ablak (függ más ablaktól)
  - `JApplet` – applet esetén a böngészőn belüli terület
- A legfelső szintű tárolóknak van egy ún. tároló mezője
  - content pane: `java.awt.Container` típusú
  - összefogja tárolt komponenseket (kivéve a menüket)
  - A Swing-es komponensek őse a `JComponent` (kivéve a legfelső szintű tárolókét)

'HelloWorld' Swing módra

Létrehozzuk az ablakot



```
import javax.swing.*;

public class HelloWorldSwing {
    public static void main(String[] args) {
        JFrame frame = new JFrame("HelloWorldSwing");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel label = new JLabel("Hello␣World");
        frame.add(label);
        frame.pack();
        frame.setVisible(true);
    }
}
```

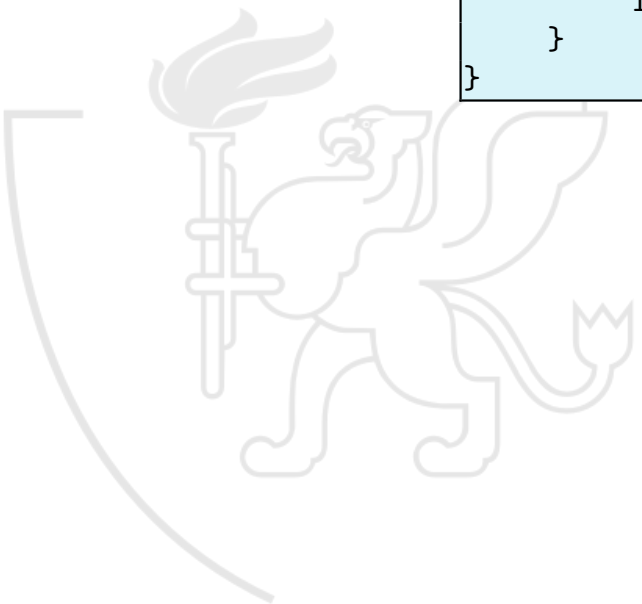
'HelloWorld' Swing módra

Létrehozzuk az ablakot

és megmondjuk neki, hogy ha bezárjuk, akkor záródjon be.

```
import javax.swing.*;

public class HelloWorldSwing {
    public static void main(String[] args) {
        JFrame frame = new JFrame("HelloWorldSwing");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel label = new JLabel("Hello World");
        frame.add(label);
        frame.pack();
        frame.setVisible(true);
    }
}
```



'HelloWorld' Swing módra

Létrehozzuk az ablakot

és megmondjuk neki, hogy ha bezárjuk, akkor záródjon be.

Csinálunk egy új „cimkét" a megadott felirattal

```
import javax.swing.*;

public class HelloWorldSwing {
    public static void main(String[] args) {
        JFrame frame = new JFrame("HelloWorldSwing");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel label = new JLabel("Hello World");
        frame.add(label);
        frame.pack();
        frame.setVisible(true);
    }
}
```

'HelloWorld' Swing módra

Létrehozzuk az ablakot

és megmondjuk neki, hogy ha bezárjuk, akkor záródjon be.

Csinálunk egy új „cimkét" a megadott felirattal

és rárakjuk az ablakra.

```
import javax.swing.*;

public class HelloWorldSwing {
    public static void main(String[] args) {
        JFrame frame = new JFrame("HelloWorldSwing");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel label = new JLabel("Hello World");
        frame.add(label);
        frame.pack();
        frame.setVisible(true);
    }
}
```

'HelloWorld' Swing módra

Létrehozzuk az ablakot

és megmondjuk neki, hogy ha bezárjuk, akkor záródjon be.

Csinálunk egy új „cimkét" a megadott felirattal

és rárakjuk az ablakra.

Megkérjük az ablakot, hogy rendezze el magát

```
import javax.swing.*;

public class HelloWorldSwing {
    public static void main(String[] args) {
        JFrame frame = new JFrame("HelloWorldSwing");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel label = new JLabel("Hello World");
        frame.add(label);
        frame.pack();
        frame.setVisible(true);
    }
}
```

'HelloWorld' Swing módra

Létrehozzuk az ablakot

és megmondjuk neki, hogy ha bezárjuk, akkor záródjon be.

Csinálunk egy új „cimkét" a megadott felirattal

és rárakjuk az ablakra.

Megkérjük az ablakot, hogy rendezze el magát

és jelenjen meg.

```
import javax.swing.*;

public class HelloWorldSwing {
    public static void main(String[] args) {
        JFrame frame = new JFrame("HelloWorldSwing");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel label = new JLabel("Hello World");
        frame.add(label);
        frame.pack();
        frame.setVisible(true);
    }
}
```

Eseménykezelés

- A komponensek tudnak generálni eseményeket
  - „tüzelés”
  - minden egyes esemény külön osztállyal van ábrázolva
- Egy esemény egy vagy több „hallgatózóhoz” (listener) érkezik meg, amelyek reagál(hat)nak rá
  - olyan osztály, amely implementál valamilyen listener interfészt
- A komponens objektumban regisztrálni kell a listener-t
  - `komp.addXYZListener(listenerObj);`
  - `XYZ` == az esemény típusa
    - `Action` – tipikusan klikk v. Enter a komponensen
    - `Window` – ablak bezárása
    - `MouseMotion` – egér mozdul a komponens felett
    - `Component` – láthatóvá válik a komponens
    - `Focus` – a komponens megkapja a fókuszt
    - stb.

Eseménykezelés

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

public class SwingApplication implements ActionListener {

    private static String labelPrefix = "Számláló: ";
    private int numClicks = 0;
    final JLabel label = new JLabel(labelPrefix + numClicks);

    public void createComponents(JFrame frame) {
        JButton button = new JButton("Swing nyomógomb");
        button.setMnemonic(KeyEvent.VK_W);
        button.addActionListener(this);
        JPanel pane = new JPanel(new GridLayout(0,1));
        pane.add(button);
        pane.add(label);
        pane.setBorder(BorderFactory.createEmptyBorder(30,30,10,30));
                                ...
    public static void main(String[] args) {
        JFrame frame = new JFrame("SwingApplication");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        SwingApplication app = new SwingApplication();
        app.createComponents(frame);
        frame.pack();
        frame.setVisible(true);
    }
}
```

Létrehozunk egy ablakot és egy „*app*”-ot. Az „*app*” objektum az adatok tárolására szolgál.

Eseménykezelés

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

public class SwingApplication implements ActionListener {

    private static String labelPrefix = "Számláló: ";
    private int numClicks = 0;
    final JLabel label = new JLabel(labelPrefix + numClicks);

    public void createComponents(JFrame frame) {
        JButton button = new JButton("Swing nyomógomb");
        button.setMnemonic(KeyEvent.VK_W);
        button.addActionListener(this);
        JPanel pane = new JPanel(new GridLayout(0,1));
        pane.add(button);
        pane.add(label);
        pane.setBorder(BorderFactory.createEmptyBorder(30,30,10,30));
                                . . .
    public static void main(String[] args) {
        JFrame frame = new JFrame("SwingApplication");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        SwingApplication app = new SwingApplication();
        app.createComponents(frame);
        frame.pack();
        frame.setVisible(true);
    }
}
```

Létrehozunk egy ablakot és egy „*app*”-ot. Az „*app*” objektum az adatok tárolására szolgál.

Az ablakot feltöltjük az „*app*” elemeivel, és megjelenítjük.

Eseménykezelés

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

public class SwingApplication implements ActionListener {

    private static String labelPrefix = "Számláló:␣";
    private int numClicks = 0;
    final JLabel label = new JLabel(labelPrefix + numClicks);

    public void createComponents(JFrame frame) {
        JButton button = new JButton("Swing␣nyomógomb");
        button.setMnemonic(KeyEvent.VK_W);
        button.addActionListener(this);
        JPanel pane = new JPanel(new GridLayout(0,1));
        pane.add(button);
        pane.add(label);
        pane.setBorder(BorderFactory.createEmptyBorder(30,30,10,30));
        frame.add(pane, BorderLayout.CENTER);
    }
```

...

```
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        SwingApplication app = new SwingApplication();
        app.createComponents(frame);
        frame.pack();
        frame.setVisible(true);
    }
}
```

Az „*app*” objektum egy számlálót (`numClicks`) és egy cimkét (`label`) tartalmaz.

Eseménykezelés

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

public class SwingApplication implements ActionListener {

    private static String labelPrefix = "Számláló:␣";
    private int numClicks = 0;
    final JLabel label = new JLabel(labelPrefix + numClicks);

    public void createComponents(JFrame frame) {
        JButton button = new JButton("Swing␣nyomógomb");
        button.setMnemonic(KeyEvent.VK_W);
        button.addActionListener(this);
        JPanel pane = new JPanel(new GridLayout(0,1));
        pane.add(button);
        pane.add(label);
        pane.setBorder(BorderFactory.createEmptyBorder(30,30,10,30));
        frame.add(pane, BorderLayout.CENTER);
    }

    public void actionPerformed(ActionEvent e) {
```

...

```
        app.createComponents(frame);
        frame.pack();
        frame.setVisible(true);
    }
}
```

Az „*app*” az ablakot a következőképpen tölti fel:

Eseménykezelés

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

public class SwingApplication implements ActionListener {

    private static String labelPrefix = "Számláló:␣";
    private int numClicks = 0;
    final JLabel label = new JLabel(labelPrefix + numClicks);

    public void createComponents(JFrame frame) {
        JButton button = new JButton("Swing␣nyomógomb");
        button.setMnemonic(KeyEvent.VK_W);
        button.addActionListener(this);
        JPanel pane = new JPanel(new GridLayout(0,1));
        pane.add(button);
        pane.add(label);
        pane.setBorder(BorderFactory.createEmptyBorder(30,30,10,30));
        frame.add(pane, BorderLayout.CENTER);
    }

    public void actionPerformed(ActionEvent e) {
        numClicks++;
        label.setText(labelPrefix + numClicks);
```

...

```
        frame.setVisible(true);
    }
}
```

Az „*app*” az ablakot a következőképpen tölti fel:

Először készít egy nyomógombot amit a 'W' gombbal is lehet majd aktiválni.

Eseménykezelés

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

public class SwingApplication implements ActionListener {

    private static String labelPrefix = "Számláló:␣";
    private int numClicks = 0;
    final JLabel label = new JLabel(labelPrefix + numClicks);

    public void createComponents(JFrame frame) {
        JButton button = new JButton("Swing␣nyomógomb");
        button.setMnemonic(KeyEvent.VK_W);
        button.addActionListener(this);
        JPanel pane = new JPanel(new GridLayout(0,1));
        pane.add(button);
        pane.add(label);
        pane.setBorder(BorderFactory.createEmptyBorder(30,30,10,30));
        frame.add(pane, BorderLayout.CENTER);
    }

    public void actionPerformed(ActionEvent e) {
        numClicks++;
        label.setText(labelPrefix + numClicks);
    }

                              ...

    }
}
```

Az osztály objektumai le tudják kezelni az „Action” eseményt: számolják hányszor következtetik be és módosítják a `label` feliratát.

Eseménykezelés

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

public class SwingApplication implements ActionListener {

    private static String labelPrefix = "Számláló:␣";
    private int numClicks = 0;
    final JLabel label = new JLabel(labelPrefix + numClicks);

    public void createComponents(JFrame frame) {
        JButton button = new JButton("Swing␣nyomógomb");
        button.setMnemonic(KeyEvent.VK_W);
        button.addActionListener(this);
        JPanel pane = new JPanel(new GridLayout(0,1));
        pane.add(button);
        pane.add(label);
        pane.setBorder(BorderFactory.createEmptyBorder(30,30,10,30));
        frame.add(pane, BorderLayout.CENTER);
    }

    public void actionPerformed(ActionEvent e) {
        numClicks++;
        label.setText(labelPrefix + numClicks);
    }

    public static void main(String[] args) {
                                ...
```

Az „*app*”-ot így be tudjuk állítani, mint a nyomógomb kezelőjét.

Eseménykezelés

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

public class SwingApplication implements ActionListener {

    private static String labelPrefix = "Számláló:␣";
    private int numClicks = 0;
    final JLabel label = new JLabel(labelPrefix + numClicks);

    public void createComponents(JFrame frame) {
        JButton button = new JButton("Swing␣nyomógomb");
        button.setMnemonic(KeyEvent.VK_W);
        button.addActionListener(this);
        JPanel pane = new JPanel(new GridLayout(0,1));
        pane.add(button);
        pane.add(label);
        pane.setBorder(BorderFactory.createEmptyBorder(30,30,10,30));
        frame.add(pane, BorderLayout.CENTER);
    }

    public void actionPerformed(ActionEvent e) {
        numClicks++;
        label.setText(labelPrefix + numClicks);
    }

    public static void main(String[] args) {
                                ...
```

Egy panelra felpakoljuk a gombot és a cimkét,

Eseménykezelés

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

public class SwingApplication implements ActionListener {

    private static String labelPrefix = "Számláló:␣";
    private int numClicks = 0;
    final JLabel label = new JLabel(labelPrefix + numClicks);

    public void createComponents(JFrame frame) {
        JButton button = new JButton("Swing␣nyomógomb");
        button.setMnemonic(KeyEvent.VK_W);
        button.addActionListener(this);
        JPanel pane = new JPanel(new GridLayout(0,1));
        pane.add(button);
        pane.add(label);
        pane.setBorder(BorderFactory.createEmptyBorder(30,30,10,30));
        frame.add(pane, BorderLayout.CENTER);
    }

    public void actionPerformed(ActionEvent e) {
        numClicks++;
        label.setText(labelPrefix + numClicks);
    }

    public static void main(String[] args) {
                                ...
```

Egy panelra felpakoljuk a gombot és a cimkét, majd a panelt az ablak közepére rakjuk.

Tartalom

'HelloWorld' JavaFX módra

```java
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class HelloWorldJavaFX extends Application {
    public static void main(String[] args) {
        Application.launch(HelloWorldJavaFX.class, args);
    }

    public void start(Stage primaryStage) {
        primaryStage.setTitle("Hello World");
        Button button = new Button();
        button.setText("Hello!");
        button.setOnAction(new EventHandler<ActionEvent>() {
            public void handle(ActionEvent event) {
                System.out.println("Hello World!");
            }
        });

        StackPane panel = new StackPane();
        panel.getChildren().add(button);
        primaryStage.setScene(new Scene(panel, 300, 250));
        primaryStage.show();
    }
}
```

Minden JavaFX program az `Application`-ből származik, és a `launch()` statikus metódussal indítható.

'HelloWorld' JavaFX módra

```java
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class HelloWorldJavaFX extends Application {
    public static void main(String[] args) {
        Application.launch(HelloWorldJavaFX.class, args);
    }

    public void start(Stage primaryStage) {
        primaryStage.setTitle("Hello World");
        Button button = new Button();
        button.setText("Hello!");
        button.setOnAction(new EventHandler<ActionEvent>() {
            public void handle(ActionEvent event) {
                System.out.println("Hello World!");
            }
        });

        StackPane panel = new StackPane();
        panel.getChildren().add(button);
        primaryStage.setScene(new Scene(panel, 300, 250));
        primaryStage.show();
    }
}
```

Minden JavaFX program az `Application`-ből származik, és a `launch()` statikus metódussal indítható.

Indításkor az `init()` metódus után a `start()` hívódik meg (ez a JavaFX alkalmazás „main”-je).

Leállításkor/kilépéskor a `stop()` hívódik.

'HelloWorld' JavaFX módra

```java
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class HelloWorldJavaFX extends Application {
    public static void main(String[] args) {
        Application.launch(HelloWorldJavaFX.class, args);
    }

    public void start(Stage primaryStage) {
        primaryStage.setTitle("Hello World");
        Button button = new Button();
        button.setText("Hello!");
        button.setOnAction(new EventHandler<ActionEvent>() {
            public void handle(ActionEvent event) {
                System.out.println("Hello World!");
            }
        });

        StackPane panel = new StackPane();
        panel.getChildren().add(button);
        primaryStage.setScene(new Scene(panel, 300, 250));
        primaryStage.show();
    }
}
```

A fő ablak `Stage` típusú, erre `Scene`-t, arra `Pane`-t, amire pedig egy `Button`-t rakunk (fordított sorrendben felépítve).

A gomb kezelését egy „névtelen” `EventHandler` osztállyal oldjuk meg: lenyomásakor kiírjuk, hogy „*Hello World!*”.