Programski jezik JAVA
PREDAVANJE 6
2018

www.etf.ac.me

Grafičko programiranje: Interfejs I - Događaji

- Programiranje grafičkog interfejsa - određeno događajima (*events*) izvan njega. Pri tome *događaji* dolaze od tastature (pritisci na tipke) i miša (kretanje miša i *klik* određenom tipkom miša).
- Svaki GUI (*graphical user interface*) program strukturiran je kao jedna beskonačna petlja u kojoj se:
  - skupljaju informacije o događajima koji su se desili
  - obavještavaju svi zainetresovani objekti.
- Objekt koji je zainteresovan za neki tip događaja reaguje na njega izvršavanjem određenog dijela koda (neke svoje metode).
- U Javinom programu koji implementira grafičko okruženje imaćemo tri vrste objekata: *izvore* događaja, o*sluškivače* i same *događaje*:
- Izvori događaja su komponente grafičkog interfejsa (npr. paneli, dugmad, klizačii, itd.).
- Osluškivači (*listeners*) događaja su objekti koji reaguju na neku vrstu događaja.
- Sami događaji (*events*) su instance određenih klasa.

Osluškivači – Registracija i Implementacija

- Komponenta koja je izvor događaja nekog tipa ima metodu kojom registruje osluškivače.
- Na taj način se ostvaruje veza između izvora i osluškivača i samo će osluškivači koji su registrovani biti obavešteni o događaju. Opšti oblik metode ovog tipa je:

```
izvorDogađaja.addDogađajListener(objektSlušač)
```

- Klasa koja modeluje osluškivač nekog događaja mora da implementira interfejs listener odgovarajućeg tipa.
- Interfejs deklariše metodu koja će automatski biti pozvana kad se događaj dogodi.
- Pošto je generisanje događaja u potpunosti pod kontrolom korisnika programa ne možemo znati kada će tačno metoda koju objekt-osluškivač definiše biti pozvana.
- Ono što znamo je da će automatski biti pozvane metode svih registrovanih osluškivača.

Primjer 1: JButton

- Klasa `javax.swing.JButton` modeluje grafičku komponentu koja predstavlja dugme.
- Ta komponenta generiše `ActionEvent` (klasa `java.awt.event.ActionEvent`) kojim se definiše *akcija* koju komponenta realizuje.
- U slučaju `JButton` komponente `ActionEvent` će biti generisan svaki put kada kliknemo na dugme.
- Osluškivač za `ActionEvent` mora da implementira interfejs `ActionListener` (u paketu `java.awt.event`) koje definiše smo jednu metodu: `actionPerformed(ActionEvent e)`.
- Ta će metoda automatski biti pozvana kad se klikne na dugme.
- Registracija osluškivača:

```
MyListener listener=....;       //osluškivč ActionEvent-a
JButton button = new JButton("OK"); // izvor ActionEvent-a
button.addMyListener(listener); // registracija osluškivača
```

Nastavak...

- Implementacija osluškivača

```
class MyListener implements ActionListener
{
    public void actionPerformed(ActionEvent e)
    {
        // reakcija na klik na dugme ide ovdje
    }
}
```

- Sada će svaki puta kad korisnik klikne na dugme sa labelom "OK" biti pozvana metoda `listener.actionPerformed`.
- Metoda kao argument automatski dobija objekt klase `ActionEvent` koji reprezentuje događaj.
- Programer samo kreira i registruje osluškivač, dok se poziv odgovarajućoj metodi (`actionPerformed`) dešava automatski.

Kompletan Primjer

- Napravimo sada kompletan primjer: program će otvoriti prozor u kome se nalaze tri dugmeta, označena jednom bojom. Klikom na dugme mijenja se boja pozadine prozora.
- Konstruktor za `JButton` uzima kao argument labelu u obliku stringa ili ikonu ili oboje. Na primjer:

```
JButton yellow = new JButton("Yellow");
```

- Nakon toga button se mora smjestiti na panel pomoću metode add koju JPanel nasljeđuje iz klase java.awt.Container. Na primjer:

```
class ButtonPanel extends JPanel
{
        public ButtonPanel()
        {
            JButton yellow = new JButton("Yellow");
            add(yellow);
        }
 }
```

Nastavak...

- Nakon što smo button stavili na panel moramo registrovati njegov osluškivač (`ActionListener`).
- Taj će osluškivač promijeniti boju panela za šta treba imati pristup metodi `setBackground` iz `JPanel` klase.
- Prema tome, treba staviti klasu koja implementira `ActionListener` interfejs unutar klase `ButtonPanel`, jer će ona tada moći dohvatiti sve njene metode.
- Kako klasu nećemo koristiti izvan `ButtonPanel` klase možemo je deklarisati kao privatnu.
- Konačno, registracija osluškivača ima ovaj oblik:

```
// ColorAction implementira ActionListener
 ColorAction yellowAction = new ColorAction(Color.YELLOW);
 yellow.addActionListener(yellowAction);
```

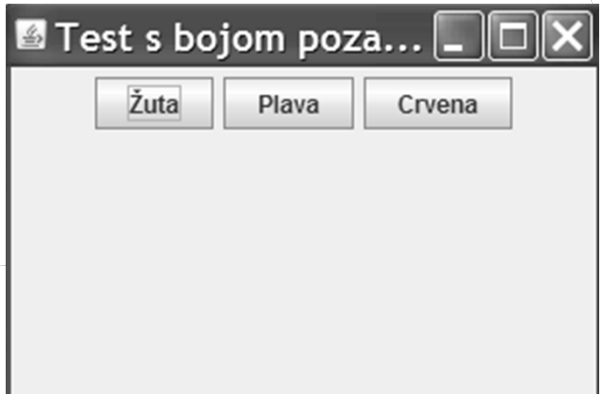
- Ovdje smo iskoristili klasu `java.awt.Color` koja implementira neke temeljne boje u obliku konstanti.

Program

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

 public class Dogadjaj1 {
     public static void main(String[] args){
         ButtonFrame frame = new ButtonFrame();
         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}

class ButtonFrame extends JFrame   {
    public ButtonFrame()       {
        setTitle("Test sa bojom pozadine");
        setSize(300,200);
        Container cp = getContentPane();
        ButtonPanel panel = new ButtonPanel();
        cp.add(panel);
    }
}

class ButtonPanel extends JPanel   {
    public ButtonPanel()      {
        // Tri buttona
        JButton yellow = new JButton("Zuta");
        JButton blue   = new JButton("Plava");
        JButton red    = new JButton("Crvena");

        // Dodajemo ih na panel
        add(yellow);
        add(blue);
        add(red);
```

Primjer..

```
// Kreiramo osluškivače ...
        ColorAction yellowAction = new ColorAction(Color.YELLOW);
        ColorAction blueAction   = new ColorAction(Color.BLUE);
        ColorAction redAction    = new ColorAction(Color.RED);

        // registrujemo osluškivače
        yellow.addActionListener(yellowAction);
        blue.addActionListener(blueAction);
        red.addActionListener(redAction);
    }

    // Privatna unutrašnja klasa
    // Na taj način ColorAction() konstruktor ne mora dobiti
    // referencu na ButtonPanel koja bi joj trebala da dohvati
    // ButtonPanel.setBackground(backgroundColor)

    private class ColorAction implements ActionListener
    {
        public ColorAction(Color c) { backgroundColor=c; }

        public void actionPerformed(ActionEvent e) {
            // metoda iz JComponent klase
            setBackground(backgroundColor);
        }

        private Color backgroundColor;
    }
 }
```



Nastavak...

- Prethodni kod možemo pojednostaviti, jer se kreiranje svakog dugmeta sastoji od četiri akcije:
  - Instanciranje JButtona;
  - Dodavanje na panel (add);
  - Konstrukcija ActionListener objekta;
  - Registracija ActionListener objekta.
- Sve to možemo obaviti u jednoj metodi. Nova metoda neka se zove `makeButton`:

```
void makeButton(String labela, final Color bojaPozadine)
    {
        JButton baton = new JButton(labela);
        add(baton);
        baton.addActionListener(new ActionListener()
                {
                     public void actionPerformed(ActionEvent e) {
                     // metoda iz JComponent klase
                     setBackground(bojaPozadine);
                }
                });
        // Referenca baton sada nestaje, ali to nije bitno jer je čuva ButtonPanel
    }
```

Nastavak...

- Konstruktor klase `ButtonPanel` sada je vrlo jednostavan:

```
public ButtonPanel()
    {
        // Tri buttona
        makeButton("Yellow", Color.YELLOW);
        makeButton("Blue", Color.BLUE);
        makeButton("Red", Color.RED);
    }
```

JButton sa ikonicama

- Swing omogućava korišćcenje grafičkih ImageIcon objekata na komponentama. Ikona je mala slika koja se postavlja na komponentu - u ovom slučaju 24x24 piksela.
- ImageIcon objekat se može kreirati specificiiranjem naziva datoteke ili grafičkog elementa u konstruktoru klase.
- Sljedeći primjer pokazuje učitavanje ikone iz grafičke datoteke subscribe.gif i kreiranje JButton tastera.

```
ImageIcon load = new ImageIcon(“load.gif”);
ImageIcon save = new ImageIcon(“save.gif”);
JButton button = new JButton(save);
JPanel pane = new JPanel();
pane.add(button);
add(pane);
```

JButton sa ikonicama - primjer

```
import javax.swing.*;
public class IconFrame extends JFrame {
          JButton load, save, subscribe, unsubscribe;
    public IconFrame() {
        super("Icon Frame");
         JPanel panel = new JPanel();
                ImageIcon loadIcon = new ImageIcon("load.gif");
                ImageIcon saveIcon = new ImageIcon("save.gif");
                ImageIcon subscribeIcon = new ImageIcon("subscribe.gif");
                ImageIcon unsubscribeIcon = new ImageIcon("unsubscribe.gif");
        load = new JButton("Ucitaj", loadIcon);
        save = new JButton("Sacuvaj", saveIcon);
        subscribe = new JButton("Prihvati", subscribeIcon);
        unsubscribe = new JButton("Ponisti", unsubscribeIcon);
       panel.add(load);     panel.add(save);      panel.add(subscribe);      panel.add(unsubscribe);
       add(panel);             pack();
}
    public static void main(String[] arguments) {
        IconFrame ike = new IconFrame();
        Ike. setVisible(true);
       Ike. setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

Labele i tekstualna polja

- Labela sadrži tekst, iconu ili oboje.
- Kreiraju se na osnovu JLabel klase, a konstruktori su:
  JLabel(String), JLabel(String, int) i JLabel(String, Icon, int).
  Komponenta int je uređenje, a definiše način na koji se tekst ili ikona postavljaju u odnosu na oblast koju zauzima prostor pomoću tri statičke promjenljive: LEFT, CENTER, RIGHT klase SwingConstants.
- Sadržaj Labele se može postaviti metodama
  setText(String) ili setIcon(Icon).
- Očitavanje se vrši pomoću funkcija getText() i getIcon().
- Primjer kreiranja Labela:
  JLabel ime = new JLabel(”Ime”, SwingConstants.LEFT);
  JLabel urlLabel = new JLabel(“URL:”., SwingConstants.CENTER);
  JLabel dateLabel = new JLabel(“Date:”., SwingConstants.RIGHT);

Labele i tekstualna polja

- Tekstualno polje je lokacija na interfejsu na kojoj korisnik može da unosi i ažurira tekst korišćcenjem tastature.
- Ova polja se predstavljaju JTextField klasom i može da sadrži SAMO jednu liniju za unos.
- Postoje sljedeći konstruktori za tekstualna polja: JTextField(), JTextField(int)(određene širine), JTextField(String, int).
- Primjer: kreiranje praznog tekstualnog polja koje može da sadrži 60 karaktera i tekstualnog polja iste veličine ali sa inicijalnim tekstom oblika "Ovdje unesite URL ":

  JTextField rssUrl = new JTextField(60);

  JTextField rssUrl2 = new JTextField("Ovdje unesite URL", 60);
- Metod setText(String) - mijenja tekst, getText()-vraća tekst,

Labele i tekstualna polja

- Polja koja sadrže šifru skrivaju tekst koji korisnik unosi. Definisana su korišćenjem JPasswordField klase. Ona se izvodi iz JTextField klase.
- Sljedećom naredbom kreiraju se polja koja sadrže šifru, pri čemu se kao echo karakter koristi "#"
  JPasswordField polje = new JPasswordField(20);
  polje.setEchoChar(.#.);
- **Regioni koji sadrže tekst:**
- Polja čiji se sadržaj može mijenjati.
- Sadrže više od jedne linije teksta.
- Implementiraju se korišćenjem klase JTextArea.
- Klasa JTextArea sadrži sljedeće konstruktore:
  JTextArea(int, int)-(tačno određeni broj linija i kolona teksta),
  JTextArea(String, int, int)-(odgovarajući tekst i broj linija i kolona).
- Metode: getText(), getSelectedText() i setText().

Labele i tekstualna polja

- Regioni koji sadrže tekst u Swingu ne podrazumijevaju korišćenje vertikalnih i horizontalnih klizača.
- Swing podržava klizače pomoću novog kontejnera JscrollPane i koristi se za čuvanje bilo koje komponente koja se može pomjerati po ekranu.
- Skrolujući panel se pridružuje komponenti u konstruktoru panela:
  JscrollPane(Component)- kreira se skrolujući panel sa datom komponentom
  JscrollPane(Component, int, int)- sadrži komponentu, vertikalni i horzontalni klizač.
- Klizaći se definišu sa statićkim promjenljivim klase definisanim na osnovu interfejsa ScrollPaneConstants.
  VERTICAL_SCROLLBAR_ALWAYS,
  VERTICAL_SCROLLBAR_AS_NEEDED,
  VERTICAL_SCROLLBAR_NEVER
- Isto i za horizontalne kilzače.

Labele i tekstualna polja - Primjer

```
import javax.swing.*;
public class Authenticator2 extends JFrame {
          JTextField username = new JTextField(15);
   JPasswordField password = new JPasswordField(15);
   JTextArea comments = new JTextArea(4, 15);
   JButton ok = new JButton("OK");
   JButton cancel = new JButton("Cancel");

   public Authenticator2() {
      super("Account Information");
      setSize(300, 220);
      JPanel pane = new JPanel();
      JLabel usernameLabel = new JLabel("Username: ");
      JLabel passwordLabel = new JLabel("Password: ");
      JLabel commentsLabel = new JLabel("Comments: ");
      comments.setLineWrap(true);
      comments.setWrapStyleWord(true);
      pane.add(usernameLabel);          pane.add(username);
      pane.add(passwordLabel);          pane.add(password);
      pane.add(commentsLabel);
```

Labele i tekstualna polja - Primjer

```
JScrollPane scroll = new JScrollPane(comments,
        ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,
        ScrollPaneConstants.HORIZONTAL_SCROLLBAR_NEVER);
      pane.add(scroll);
      pane.add(ok);
      pane.add(cancel);
      add(pane);

   }



     public static void main(String[] args) {
      Authenticator2 auth = new Authenticator2();
      auth.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      auth.setVisible(true);
   }
}
```