

Java tečaj

7. dio
Swing

Uvod

- ◆ Kada pričamo o Swingu, pričamo o izradi korisničkog sučelja
- ◆ Svaka aplikacija treba/ima korisničko sučelje
- ◆ Korisničko sučelje može biti:
 - Grafičko
 - Komandno linijsko
 - ...

Uvod

- ◆ Swing == grafičko korisničko sučelje
- ◆ Jezgra temeljena na starijoj implementaciji korisničkog sučelja: AWT (Abstract Windows Toolkit)
(vidljivo npr. Po stablu nasljeđivanja)
- ◆ Dodan niz mogućnosti i unapređenja
- ◆ Postoji još i JavaFX – nećemo ga razmatrati u okviru ove vještine

Vršni elementi

- ◆ Swing definira tri vršna elementa (engl. Top level containers):
 - **JFrame** – standardni prozor
 - **JDialog** – dijaloška kutija
 - **JApplet** – aplet koji se prikazuje u web pregledniku

Vršni elementi: JFrame

- ◆ Loš primjer otvaranja prozora
(zašto loš – vidi slide 16)

OtvaranjaProzora.java

Vršni elementi : JFrame

```
public class OtvaranjeProzora {  
  
    public static void main(String[] args) {  
        JFrame frame = new JFrame();  
  
        frame.setLocation(20, 20);  
        frame.setSize(500, 200);  
  
        frame.setVisible(true);  
    }  
}
```

Vršni elementi : JFrame

- ◆ Prikazani prozor prilikom “ubijanja” zapravo ne nestaje – samo se skriva
- ◆ Posljedica je da aplikacija (proces) i dalje radi, iako korisnik više ništa ne može napraviti

Vršni elementi : JFrame

- ◆ AWT Frame po defaultu ignorira korisnikov pokušaj zatvaranja prozora
- ◆ Swing uvodi metodu

`setDefaultCloseOperation(int)`

```
setDefaultCloseOperation(  
    WindowConstants.DO_NOTHING_ON_CLOSE  
)
```


Vršni elementi : JFrame

```
public class OtvaranjeProzora {  
  
    public static void main(String[] args) {  
        JFrame frame = new JFrame();  
  
        frame.setLocation(20, 20);  
        frame.setSize(500, 200);  
        frame.setDefaultCloseOperation(  
            WindowConstants.DISPOSE_ON_CLOSE);  
  
        frame.setVisible(true);  
    }  
}
```

Vršni elementi : JFrame

◆ WindowConstants navodi nekoliko vrijednosti:

- `DO_NOTHING_ON_CLOSE` – ignorira zahtjev za zatvaranjem
- `HIDE_ON_CLOSE` – samo sakrij prozor
- `DISPOSE_ON_CLOSE` – sakrij prozor pa oslobodi sve njegove resurse
- `EXIT_ON_CLOSE` – terminiraj samu aplikaciju (može izazvati iznimku!)

Vršni elementi : JFrame

◆ Preporučam uporabo:

`DISPOSE_ON_CLOSE`

ili

`EXIT_ON_CLOSE`

(ako je to stvarno nužno)

Vršni elementi : JFrame

◆ Pokretanjem korisničkog sučelja aplikacija automatski dobiva nove dretve:

- Java2D Disposer
- AWT-Shutdown
- AWT-Windows
- AWT-EventQueue-0

Vršni elementi : JFrame

- ◆ Obzirom na višedretvenost, SVE poslove oko rada s korisničkim sučeljem trebala bi obavljati samo jedna dretva:
event dispatching thread
- ◆ Potporu za ovo nudi razred `SwingUtilities`, u okviru metoda `invokeLater` te `invokeAndWait`

Vršni elementi : JFrame

- ◆ Umjesto da netko “izvana” mijenja sadržaj JFrame-a, obično se prozor izvodi kao novi razred koji nasljeđuje JFrame

Primjer: Prozor1.java

Vršni elementi : JFrame

```
public class Prozor1 extends JFrame {  
  
    public Prozor1() {  
        ...  
    }  
  
    protected void initGUI() {  
        ...  
    }  
  
    public static void main(String[] args) {  
        ...  
    }  
}
```

Vršni elementi : JFrame

```
public static void main(String[] args) {  
    try {  
        SwingUtilities.invokeAndWait(  
            new Runnable() {  
                public void run() {  
                    Prozor1 prozor = new Prozor1();  
                }  
            }  
        );  
    } catch (InterruptedException ex) {  
    } catch (InvocationTargetException ex) {  
    }  
}
```


Vršni elementi : JFrame

```
public Prozor1() {  
    super();  
    initGUI();  
}
```

Vršni elementi : JFrame

```
protected void initGUI() {  
    setLocation(20, 20);  
    setSize(500, 200);  
    setDefaultCloseOperation(  
        WindowConstants.DISPOSE_ON_CLOSE);  
    setTitle("Prozor1");  
    setVisible(true);  
}
```

Elementi korisničkog sučelja

- ◆ Swing nudi niz gotovih elemenata (komponenti, kontrola) koje se mogu koristiti (i prilagođavati!)

<http://docs.oracle.com/javase/tutorial/ui/features/components.html>

Elementi korisničkog sučelja

- ◆ Dodajmo u prozor jednu labelu i jedan gumb

Primjer: Prozor2.java


Elementi korisničkog sučelja

- ◆ Ograničenje ovakvog pristupa:
 - Pozicioniranje je apsolutno, širina prozora obično nije → izgleda loše pri promjeni veličine prozora!
- ◆ Kako bi riješili problem, koriste se komponente čija je zadaća automatsko pozicioniranje komponenti → **Layout Manager**

Layout manager

- ◆ Komponenta koja sama prema određenim pravilima razmješta komponente po raspoloživom prostoru
- ◆ Java nudi nekoliko gotovih Layout Managera

Layout manager

- ◆ Flow Layout
 - ◆ Box Layout
 - ◆ Card Layout
 - ◆ Flow Layout
 - ◆ GridBag Layout
 - ◆ Grid Layout
 - ◆ Spring Layout
- 
- A stylized, dark teal silhouette of a mountain range is positioned in the bottom right corner of the slide, extending from the right edge towards the center.

Elementi korisničkog sučelja

- ◆ Pozicioniranje labele i gumba uporabom `BorderLayout-a`

Primjer: `Prozor3.java`

Elementi korisničkog sučelja

- ◆ Složenije pozicioniranje – koristiti `JPanel` kao “container” drugih komponenti koji ima svoj vlastiti Layout Manager
- ◆ Dodajmo još tri gumba!

Primjer: `Prozor4.java`

Elementi korisničkog sučelja

- ◆ Kada se pritisne gumb, generira se event koji opisuje što se točno dogodilo
- ◆ I niz drugih situacija generira evente:
 - Pomak miša, pritisak/otpuštanje tipke miša, klik miša, ...
 - Pritisak/otpuštanje tipke na tipkovnici, klik neke tipke, ...

Elementi korisničkog sučelja

- ◆ Komponente nude sučelje za registraciju zainteresiranih slušača na različite vrste događaja
→ oblikovni obrazac **Observer**
- ◆ Primjerice:
 - `addActionListener`,
 - `addMouseListener`,
 - `addMouseMotionListener`,
 - `addKeyListener`, ...

Elementi korisničkog sučelja

- ◆ "Slušać" za svaku vrstu događaja modeliran je zasebnim sučeljem:
- ◆ Primjerice:
 - `ActionListener`,
 - `MouseListener`,
 - `MouseMotionListener`,
 - `KeyListener`, ...

Elementi korisničkog sučelja

- ◆ Npr. `ActionListener`

```
interface ActionListener {  
    void actionPerformed(ActionEvent e);  
}
```

- ◆ `ActionEvent` objekt enkapsulira različite informacije o događaju: izvor događaja, stanje tipki CTRL/ALT/SHIFT, trenutak kada je događaj izazvan, ...

Elementi korisničkog sučelja

- ◆ Kako postići da kada korisnik klikne gumb, mi nešto napravimo?
 - Trebamo se registrirati kod gumba i javiti da nas zanima taj događaj
→ *event listener*

Primjer: Prozor5.java

Elementi korisničkog sučelja

- ◆ Napravimo prozor s labelom u kojoj se ciklički svake dvije sekunde ispiše jedna od pripremljenih poruka
 - Swing + višedretvenost!
 - WindowListener

Primjer: Prozor6.java

Elementi korisničkog sučelja

- ◆ Napraviti prozor koji ima `JTextField`, lijevo od njega "Unesi broj:", desno od njega gumb "Izračunaj" a iznad njega labelu. Kada korisnik pritisne gumb, u labeli treba ispisati kvadrat broja koji je korisnik upisao u `JTextField`! Ako nastupi problem, dojaviti to korisniku (`JOptionPane`)

Prikaz podataka i modeli

- ◆ Swing obilato koristi MVC paradigmu
- ◆ Komponente koje prikazuju podatke (poput lista, tablica, stabala) su *pogledi*, i one su razdvojene od samih podataka i upravljanja podacima (koji su enkapsulirani u *model*)
- ◆ Tako ista komponenta može prikazivati podatke iz datoteke, baze, memorije, ...

Prikaz podataka i modeli

- ◆ Kako bi prikazi postali svjesni da su se podatci promijenili, oni se registriraju kao slušači nad model (i opet oblikovni obrazac Observer)

Prikaz podataka i modeli

- ◆ To ćemo pogledati na primjeru komponente JList
- ◆ Važni razredi/sučelja:
 - JList (pogled)
 - ListModel (sučelje koje opisuje model podataka)
 - ListDataListener (sučelje slušača)
 - ListDataEvent (opis događaja)

Elementi korisničkog sučelja

- ◆ Za vježbu:
TextReader
- ◆ Program ima dva gumba i jedan tekstualni editor (`JTextArea`)
- ◆ Gumbi su učitaj (učitava neku preddefiniranu datoteku) i sortiraj (čijim se aktiviranjem retci prikazani u editoru sortiraju)
- ◆ Proširiti s dijalogom za odabir datoteke (`JFileChooser#showOpenDialog`)