

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

SEMINAR

Izrada grafičkog sučelja pomoću Swing skupa alata

Ivan Belfinger

Voditelj: *Mr. sc. Marko Čupić*

Zagreb, travanj, 2011.

Sadržaj

1. Uvod.....	4
2. Izvorišni razred tehnologije Swing – razred JComponent	1
3. Vršni kontejneri	2
3.1 Razred JFrame.....	3
3.2 Razredi JDialog i JOptionPane	5
3.3 Razred JWindow	7
4. Kontejneri posebne namjene.....	8
4.1 Razredi JDesktopPane i JInternalFrame	8
4.2 Ostali kontejneri posebne namjene.....	8
5. Svojstvo layout i upravitelji razmještajem.....	9
5.1 Razred FlowLayout	9
5.2 Razred BorderLayout.....	10
5.3 Razred GridLayout	10
6. Jednostavne Swing komponente	11
6.1 Razred JLabel.....	11
6.2 Razred JButton	12
6.3 Razredi JToggleButton, JCheckBox i JRadioButton.....	12
6.4 Razredi JTextField i JPasswordField	13
6.5 Razred JToolTipText.....	14
6.6 Razred JProgressBar.....	14
6.7 Razred JSlider	14
7. Naprednije Swing komponente	15
7.1 Razred JList.....	15
7.2 Razred JComboBox	16
7.3 Izbornici.....	17
8. Napredne tekstualne komponente	19
9. Kratki pregled ostalih komponenti tehnologije Swing	20
10. Uređivanje okvira komponente.....	22

11. Zaključak	25
12. Literatura	26
13. Sažetak	27

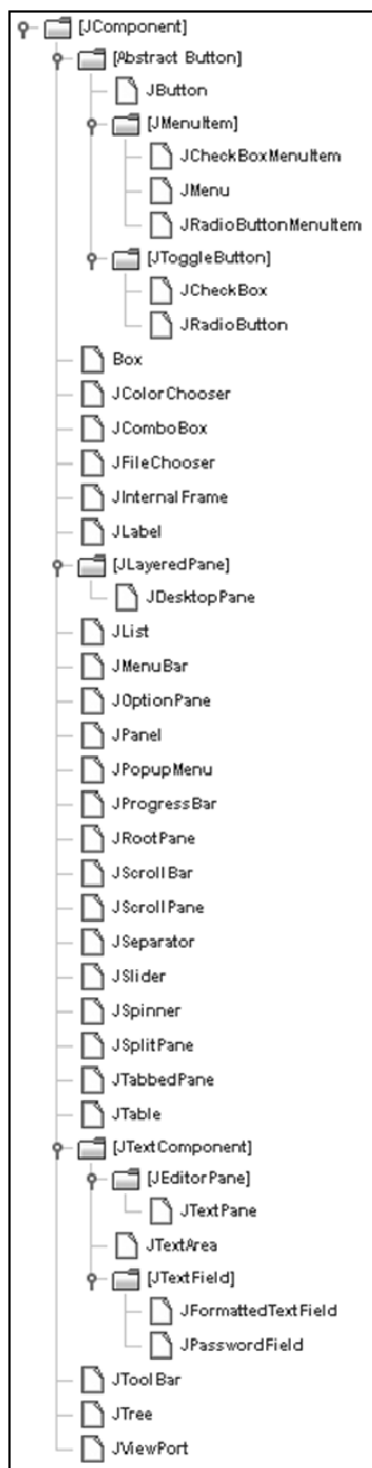
1. Uvod

Tehnologija Swing je skup alata elemenata grafičkog sučelja (engl. *widget toolkit*) za izgradnju grafičkih sučelja u programskom jeziku Java. Predstavljena sredinom 90-ih godina prošlog stoljeća, tehnologija Swing je nastala kao odgovor na probleme uočene u tehnologiji AWT (engl. *Abstract Window Toolkit*), prvotnom grafičkom skupu alata. Tehnologija AWT je bila spora i nepouzdana, i je prikaz komponenata ovisio je operacijskom sustavu u kojem je aplikacija pokrenuta. Time je programeru onemogućeno precizno upravljanje izgledom same aplikacije na različitim platformama (zbog toga se AWT komponente često naziva teškim (engl. *heavyweight*) komponentama).

Tehnologija Swing se za razliku od tehnologije AWT, u potpunosti oslanja na kod napisan u programskom jeziku Java (Swing komponente se često naziva lakim (engl. *lightweight*) komponentama). Zbog toga omogućuje puno više fleksibilnosti i kontrole prilikom dizajniranja korisničkog sučelja. Primjerice, programer je u mogućnosti mjenjati boju, pozadinu, prozirnost, i mnoge druge osobine gotovo svih Swing komponenti neovisno o platformi na kojoj je aplikacija pokrenuta. Također, sve se te osobine mogu mjenjati tokom samog izvođenja aplikacije.

U ovom seminaru obradit ćemo neke vršne (engl. *top level*) kontejnere (kao što su `JFrame` i `JDialog`), neke upravitelje razmještajem (engl. *layout manager*). Analizirati ćemo neke jednostavne elemente grafičkog sučelja (kao što su `JButton`, `JCheckBox`, `JComboBox`, `JLabel`) te neke kompliciranije (kao što su `JList`, `JColorChooser`, `JFileChooser`). Ukratko ćemo spomenuti i postupak uređivanja okvira komponente (engl. *border*), te pozadinu iza čitave logike grafičkih sučelja – upravljanje događajima (engl. *event handling*).

2. Izvorišni razred tehnologije Swing – razred JComponent



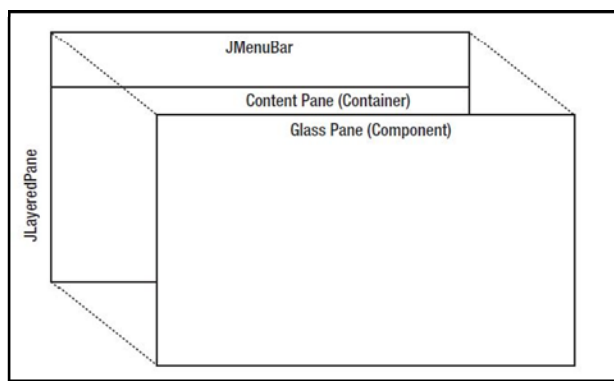
Slika 1. Stablo nasljeđivanja razreda JComponent. Preuzeto iz [2].

Razred JComponent je korjenski apstraktni razred iz kojega potiče većina Swing komponenti. Razred JComponent ima 42 podrazreda, od kojih svaki nasljeđuje većinu funkcionalnosti razreda JComponent, kako se može vidjeti na slici 1. Spomenuti razred nasljeđuje od razreda Container, sadržanog u tehnologiji AWT. Razred Container nasljeđuje od razreda Component, također sadržanog u tehnologiji AWT. Prema tome, razred JComponent mnoge aspekte dijeli s dva prethodno spomenuta razreda. Iako je razred JComponent ishodišni svim Swing komponentama, ne nasljeđuju svi razredi koje koristimo prilikom izgradnje grafičkog sučelja iz razreda JComponent. Primjeri razreda koji ne nasljeđuju su razredi vršnih kontejnera poput JFrame i JInternalFrame, a među njima su razredi i sučelja zadužena za obradu događaja.

Svaka komponenta sadrži u sebi podlogu za obradu događaja. Svaka interakcija korisnika s grafičkim sučeljem proizvodi događaj (*engl.* event). Događaj može biti unos teksta na tipkovnici, klik ili potez mišem, promjena fokusa s jedne komponente na drugu i slično. Svaki primjerak razreda JComponent u sebi sadrži primjerak razreda EventListenerList naziva listenerList. Primjerak listenerList u sebi sadrži sve događaje na koje je komponenta dužna reagirati, a koje definira sam korisnik. Kada komponenta ustanovi da se dogodio takav događaj, ona jednostavno poziva odgovarajuću metodu za obradu događaja. Više o samoj strukturi razreda JComponent može se naći u [1] i [6]. Opširnije o samim detaljima obrade događaja biti će u nastavku seminara.

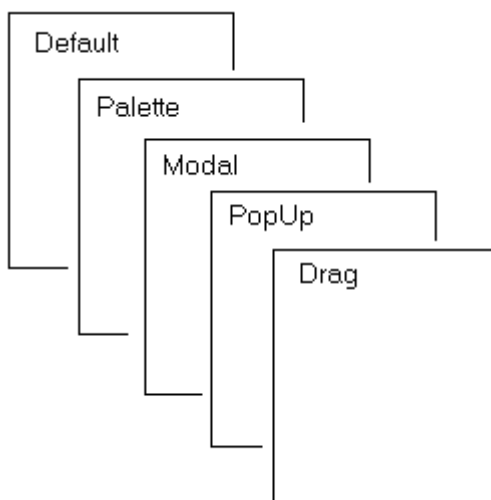
3. Vršni kontejneri

Vršni kontejneri su osnovni gradivni elementi svakog grafičkog sučelja u tehnologiji Swing. Sve komponente dodaju se u neki od vršnih kontejnera. Komponente se ne dodaju direktno u kontejner (za razliku od tehnologije AWT), nego se oslanjaju na razred `JRootPane`. On komponente organizira u slojeve, kako bi određene komponente uvijek bile ispred ostalih (npr. tooltip tekst). Razred `JRootPane` se sastoji od dva elementa: primjerka razreda `JLayeredPane` naziva `layeredPane` i primjerka razreda `Component` naziva `glassPane` (slika 2).



Slika 2. Struktura razreda `JRootPane`

Primjerak `glassPane` je proziran i nalazi se ispred ostalih komponenti. Primjerak `JLayeredPane` organizira komponente u slojeve, kako bi osigurao ispravan prikaz svih elemenata grafičkog sučelja. Ti slojevi su i pripadajuće konstante su prikazani na slici 3 i u tablici 1.



Slika 3. Slojevi razreda `JLayeredPane`.
Preuzeto sa [7].

Tablica 1. Nazivi, konstante i opisi slojeva razreda JLayeredPane

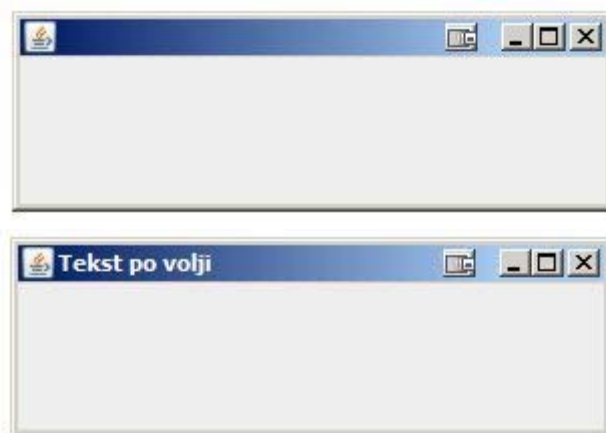
Naziv sloja	Konstanta sloja	Opis sloja
DEFAULT_LAYER	0	Standardni sloj u koji ide većina komponenti.
PALETTE_LAYER	100	Nalazi se iznad sloja DEFAULT_LAYER. Koristi se za toolbar i slično.
MODAL_LAYER	200	Korišten za prozore s obavijestima.
POPUP_LAYER	300	Korišten za iskočne menije i tooltip.
DRAG_LAYER	400	Kada se komponenta povlači, prebacuje se u DRAG_LAYER kako bi se osiguralo da bude iznad svih ostalih komponenata. Kada se povlačenje završi, komponenta se vraća u svoj sloj.

3.1 Razred JFrame

Razred JFrame je razred najjednostavnijeg i najkorištenijeg vršnog kontejnera od svih sadržanih u tehnologiji Swing. Razred JFrame ima dva glavna konstruktora, jedan za stvaranje prozora bez teksta, jedan za stvaranje prozora sa tekstom (tablica 2). Dva sporedna, sa specijaliziranim grafičkim konfiguracijama nisu nam zanimljiva.

Tablica 2. Konstruktori razreda JFrame

Konstruktor	<code>public JFrame()</code>	<code>public JFrame(String title)</code>
Primjerak konstruktora	<code>JFrame frame = new JFrame();</code>	<code>JFrame frame = new JFrame(„Tekst po volji”);</code>
Slika	Slika 4, gornji prozor	Slika 4, donji prozor



Slika 4. Primjerci razreda `JFrame` inicijaliziranih različitim konstruktorima.

Dva specifična i nama zanimljiva svojstva (engl. *property*) razreda `JFrame` koja nisu posljedica implementacije sučelja `RootPaneContainer` su svojstvo `defaultCloseOperation` i svojstvo `layout`. Svojstvo `DefaultCloseOperation` uvjetuje kako će prozor reagirati kada kliknemo na njegov gumb za zatvaranje. Začudo, početna vrijednost tog svojstva nije izlazak, nego sakrivanje prozora, zbog čega aplikacija može ostati pokrenuta iako su svi prozori zatvoreni. Obično se vrijednost tog svojstva mijenja metodom `frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)`. Konstante koje svojstvo `defaultCloseOperation` može poprimiti i njihov opis su navedeni u sljedećoj tablici.

Tablica 3. Vrijednosti koje može poprimiti svojstvo `defaultCloseOperation` i njihov opis.

Konstanta	Opis
<code>EXIT_ON_CLOSE</code>	Zatvara sve prozore i samu aplikaciju, tj. zove metodu <code>System.exit(0)</code> .
<code>DISPOSE_ON_CLOSE</code>	Zove <code>dispose()</code> na prozoru. Prozor se zatvara, ostali prozori ostaju upaljeni.
<code>DO_NOTHING_ON_CLOSE</code>	Ignorira zahtjev.
<code>HIDE_ON_CLOSE</code>	Čini prozor nevidljivim (zove na prozoru metodu <code>setVisible(false)</code>). Početna vrijednost.

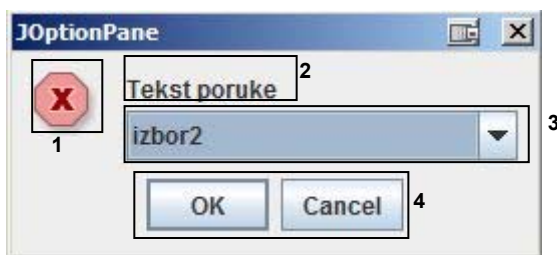
Sljedeće zanimljivo svojstvo jest `layout`. Svojstvo `layout` određuje na koji se način komponente sučelja razmještaju u kontejner. Više o svojstvu `layout` i upraviteljima razmještaja biti će u nastavku seminara.

Iako se komponente ne dodaju direktno u primjerak razreda `JFrame` (nego u njegov primjerak razreda `JRootPane` naziva `RootPane`, koji se onda brine u kojim slojevima su komponente), od platforme J2SE 5.0 (Java 2 Platform Standard Edition) komponente se mogu dodavati metodom, kao da se dodaju direktno u primjerak razreda `JFrame` (ne mora se dohvaćati primjerak `RootPane`). Ta metoda je `Frame.add(Component);`

3.2 Razredi `JDialog` i `JOptionPane`

Razred `JDialog` nam omogućuje prikaz podprozora koji u sebi nose informaciju koja je vezana uz samu aplikaciju. On u sebi može zadržavati bilo koju kombinaciju komponenata, no nama su najzanimljiviji dijaloški prozori proizvedeni pomoću razreda `JOptionPane`.

Razred `JOptionPane` je razred vršnih kontejnera koji prikazuje standardni iskočni dijaloški podprozor. Koristeći razred `JOptionPane` možemo brzo oblikovati razne vrste podprozora, i on svoj sadržaj predstavlja u četiri područja koja su prikazana na slici 5 i opisana u tablici 4.



Slika 5. Vizualna područja razreda `JOptionPane`

Niti jedno od ovih polja nije obavezno, premda je poželjno da ih bude što više, kako bi informacija upućena korisniku bila što potpunija. Standardni konstruktor je razreda `JOptionPane` je

```
JOptionPane(Object message, int messageType, int optionType,  
Icon icon, Object options[], Object initialValue).
```

Najjednostavnije modalne dijaloške podprozore se stvara i prikazuje jednom od `showXxxDialog` metoda (modalni podprozori su oni čijim prikazivanjem onemogućujemo interakciju s glavnim prozorom. Nakon što se podprozor zatvori, interakcija je ponovno omogućena.). U tom slučaju će primjerak razreda `JOptionPane` biti umetnut u primjerak razreda `JDialog`. Ako ga želimo umetnuti u primjerak razreda `JInternalFrame`, koristimo jednu od `showInternalXxxDialog` metoda.

Tablica 4. Nazivi i opisi vizualnih područja razreda *JOptionPane*

Naziv područja	Broj (slika 5)	Opis područja
Ikona	1	Ikona sugerira tip poruke koja se prikazuje. Obično se od trenutnog „look and feel“-a očekuje da osigura ikonu, premda programer može odabrati proizvoljnu.
Poruka	2	Smisao ovoga područja je da prikaže informaciju u obliku teksta.
Unos	3	Ovo područje omogućava reakciju korisnika na poruku. Polje unosa može biti različitoga oblika.
Gumbi	4	Ovo područje također očekuje unos od korisnika. Klik na neki od gumba obično znači i kraj rada sa primjerkom razreda <i>JOptionPane</i> . Broj i naziv gumba može biti predefiniran, ali i proizvoljan.

Postoji pet vrsta poruka, i četiri vrste predefiniranih gumba koje *JOptionPane* može prikazati. Konstante koje možemo upisati u polje `messageType` konstruktora, koje uvjetuju tip prikazanog podprozora, su:

- `ERROR_MESSAGE`,
- `INFORMATION_MESSAGE`,
- `QUESTION_MESSAGE`,
- `WARNING_MESSAGE` te
- `PLAIN_MESSAGE`.

Konstante koje možemo upisati u polje `optionType` konstruktora, koje uvjetuju tip i broj gumba prikazanih u dijaloškom okviru su:

- `DEFAULT_OPTION`,
- `OK_CANCEL_OPTION`,
- `YES_NO_CANCEL_OPTION` te
- `YES_NO_OPTION`.

Tablica 5. Vrste *showXxxDialog* metode

Metoda	Opis informacije koju prikazuje nastali podprozor
<code>showConfirmDialog</code>	Podprozor postavlja pitanje koje zahtjeva odgovor oblika Yes/No/Cancel.
<code>showInputDialog</code>	Podprozor zahtijeva unos korisnika.
<code>showMessageDialog</code>	Podprozor prikazuje jednostavnu obavjest.
<code>showOptionDialog</code>	Prozor koji objedinjuje gornja tri prozora.

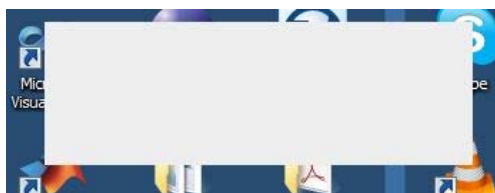
Izbor korisnika (stisnutu tipku) možemo dobiti preko metode razreda `JOptionPane` `getValue()`. Konstante koje ta metoda može vratiti su prikazane i opisane u sljedećoj tablici.

Tablica 6. Konstante koje vraća *getValue()* metoda

Konstanta	Pritisnuti gumb u dijaloškom podprozoru
<code>CANCEL_OPTION</code>	Cancel
<code>CLOSED_OPTION</code>	Prozor je zatvoren bez da je stisnuta ijedna tipka
<code>NO_OPTION</code>	No
<code>OK_OPTION</code>	Ok
<code>YES_OPTION</code>	Yes

3.3 Razred `JWindow`

Još jedan razred vršnih kontejnera jest razred `JWindow`. On je izgledom indentičan razredu `JFrame` bez okvira s nazivom i tipkama za minimizaciju, maksimizaciju i izlazak.



Slika 6. Primjerci i usporedba razreda `JWindow` (lijevo) i razreda `JFrame` (desno).

Kontejneri posebne namjene

4.1 Razredi JDesktopPane i JInternalFrame

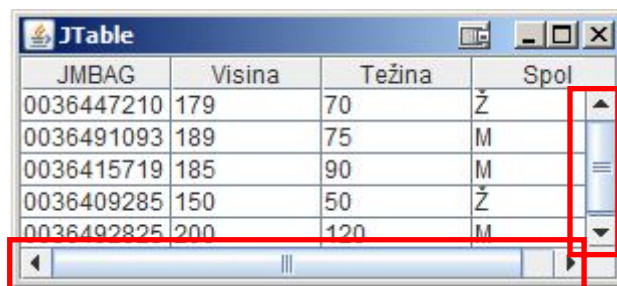
Razred JDesktopPane je razred kontejnera posebne namjene koji se koristi za simuliranje radne površine. U tu radnu površinu (tj. primjerak razreda JDesktopPane) se dodaju primjerci razreda JInternalFrame, koji su ekvivalentne prozorima na toj radnoj površini.



Slika 7. Primjerci razreda JDesktopPane i razreda JInternalFrame

4.2 Ostali kontejneri posebne namjene

Spomenuti ćemo još dva razreda kontejnerskih komponenti: razred JPanel i razred JScrollPane. Razred JPanel nam omogućava da prozor vizualno i logički podijelimo na više dijelova, i te dijelove zatim uređujemo odvojeno, kao zasebne komponente. Razred JScrollPane omogućava prikaz velike komponente unutar manjeg prozora, i to pomoću jednog ili više scrollbar-a.



Slika 8. Primjerak razreda JScrollPane sa crveno uokvirenim scrollbar-ima

5. Svojstvo layout i upravitelji razmještajem

Svaka kontejnerska komponenta ima svojstvo `layout` u kojemu čuva referencu na pridruženi upravitelj razmještajem. Upravitelj razmještajem svake komponente se brine o razmještaju elemenata unutar nje, bez potrebe da se položaj i veličina svakoga elementa ručno izračunava. Također, veličine određenih elemenata ovise o platformi na kojoj je aplikacija pokrenuta, te postoji mogućnost da korisnik promjeni veličinu prozora. U tim situacijama proces računanja položaja komponenti i njihovih veličina može biti iznimno mukotrpan posao. Tada nam u pomoć priskaču upravitelji razmještajem.

Kako bi saznao koliko je mjesta komponenti potrebno, upravitelju razmještajem su na raspolaganju tri metode: `getMinimumSize()`, `getPreferredSize()` i `getMaximumSize()`. Pomoću njih može saznati minimalnu, poželjnu i maksimalnu veličinu potrebnu za ispravno prikazivanje određene komponente. Također je u određenim situacijama u mogućnosti ignorirati određene komponente (kontejner čije svojstvo `layout` u sebi čuva referencu na razred `BorderLayout` može u sebi imati desetak komponenata; ali će biti prikazano njih samo pet), pa čak i sve zahtjeve komponente za veličinom (primjerice, poželjnu veličinu nije niti problem ignorirati).

5.1 Razred `FlowLayout`

Razred `FlowLayout` je jedan od najjednostavnijih upravitelja razmještajem, te je ujedno i podrazumjevani upravitelj razmještajem razreda `JFrame`. Razred `FlowLayout` slaže komponente u redove. Kada u trenutnom redu više nema mjesta, slijedeća komponenta se stavlja red niže. U slučaju da za neku komponentu nema dovoljno mjesta, ostale se neće smanjiti.



Slika 9. Primjerci razreda `FlowLayout` sa dovoljno mjesta za sve komponente (lijevo) i nedovoljno mjesta za sve komponente (desno)

Podrazumjevani razmak između komponenti u razredu `FlowLayout` je pet piksela.

5.2 Razred BorderLayout

Razred BorderLayout je upravitelj razmještajem u kojem se elementi raspoređuju u ekvivalentno stranama svijeta. U slučaju da komponenta na jednoj strani nije prikazana, centar će se raširiti na njeno mjesto. Elementi se dodaju na ogovarajuća mjesta pomoću konstanti:

- `BorderLayout.NORTH`,
- `BorderLayout.SOUTH`,
- `BorderLayout.WEST`,
- `BorderLayout.EAST` te
- `BorderLayout.CENTER`.



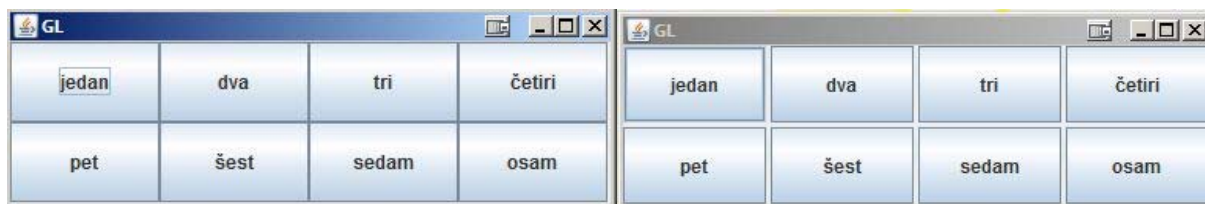
Slika 10. Raspored komponenti u razredu BorderLayout

5.3 Razred GridLayout

Razred GridLayout je upravitelj razmještajem koji organizira komponente u mrežu. Konstruktori ovog razreda su:

- `public GridLayout();`,
- `public GridLayout(int rows, int columns);` te
- `public GridLayout(int rows, int columns, int hgap, int vgap);`.

Stvarni broj redova i stupaca ovisi o broju komponenata u kontejneru. Razred GridLayout prvo promotri broj zatraženih redova. U slučaju da broj redova nije nula, broj stupaca se izračunava po formuli $(\text{broj komp.} + \text{redovi} - 1) / \text{redovi}$. Ako se zatraži nula redova, broj redova jest $(\text{broj komp.} + \text{stupci} - 1) / \text{stupci}$.



*Slika 11. Razred GridLayout inicijaliziran konstruktorom
GridLayout(2, 4) (lijevo) i konstruktorom GridLayout(2, 4, 3, 3)
(desno)*

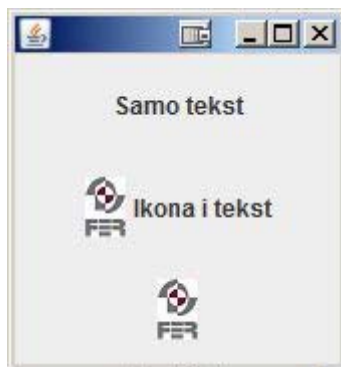
Spomenut ćemo još neke bitnije upravitelje razmjestajem, premda ih nećemo detaljno objašnjavati. Ti su GridBagLayout, CardLayout, BoxLayout i drugi.

6. Jednostavne Swing komponente

6.1 Razred JLabel

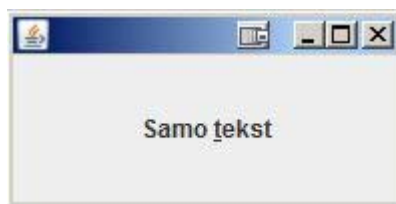
Razred JLabel je jednostavna komponenta koja je u mogućnosti prikazivati tekst, sliku ili oboje. Razred JLabel ima šest konstruktora, nama je najzanimljivij onaj koji nam omogućuje i najviše opcija:

`JLabel(String text, Icon icon, int hAlignment)`



*Slika 12. Primjerci razreda
JLabel*

Ovaj konstruktor nam omogućuje definiciju prikazanog teksta, slike i horizontalne orijentacije komponente. Ovaj razred je u mogućnosti prikazivati mnemonike pomoću metode `displayMnemonic()`.



*Slika 13. Primjerak razreda JLabel s postavljenim
mnemonikom metodom
setDisplayMnemonic(KeyEvent.VK_T)*

6.2 Razred JButton

Ovaj razred nam omogućuje stvaranje komponente u obliku gumba koji se može stisnuti, a u sebi može sadržavati sliku, tekst ili oboje te tekst u HTML formatu. Razred JButton ima pet konstruktora, a nama najzanimljiviji je JButton(String text, Icon icon), koji nam omogućuje da definiramo tekst i sliku koja će se prikazivati unutar gumba.



Slika 14. Primjerci razreda JButton inicijalizirani sa različitim konstruktorima

Obrada događaja vezanih uz razred JButton se vrši pomoću sučelja ActionListener. Kada se primjerak razreda JButton pritisne, svi registrirani primjerci razreda koji implementiraju sučelje ActionListener su obavješteni, i primjerak razreda(ActionEvent) je proslijeđen svakom slušatelju (engl. *event listener*). Ovaj događaj u sebi sadrži svojstvo `actionCommand` pomoću koje se može saznati točno koja tipka je stisnuta. Pomoću metode `addActionListener` se na komponentu registrira primjerak razreda ActionListener.

6.3 Razredi JToggleButton, JCheckBox i JRadioButton

Ova tri razreda tehnologije Swing obraditi ćemo skupa, pošto imaju određena slična svojstva. Prvo od njih je da svaka od ovih komponenata može biti u dva stanja: označeno ili neoznačeno. Svaka od njih se također može ubacivati u razred ButtonGroup, i na taj način samo jedna komponenta unutar grupe može biti označena (iako je ovo pomalo neobično ponašanje za razred JCheckBox).

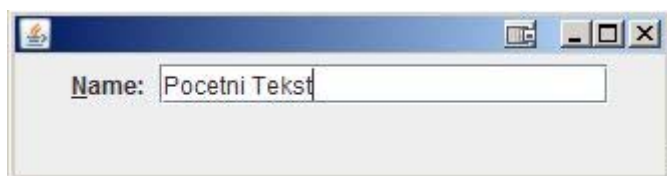


Slika 15. Primjerci razreda JToggleButton, JCheckBox i JRadioButton ubačenih u razred ButtonGroup

Sva tri razreda podržavaju prikazivanje mnemonika. Osluškivanje događaja je kod ovih razreda najbolje vršiti pomoću sučelja `ItemListener`. Razred `ItemEvent` koji nastaje kod takvih događaja u sebi nosi i komponentu i njeno stanje, i te se informacije mogu saznati pomoću metoda `getSource()`; i `getStateChange()`;

6.4 Razredi `JTextField` i `JPasswordField`

Razred `JTextField` je razred tekstualne komponente koja služi za unos jedne linije teksta. Najčešće korišteni konstruktor je `public JTextField(String pocetniTekst, int Sirina)` pomoću kojega možemo ubaciti početni tekst i namjestiti početnu širinu polja. Najčešće se koristi u kombinaciji sa primjerkom razreda `JLabel`, pri čemu se često mnemonik pridružen primjerku razreda `JLabel` koristi kako bi se fokus pomaknuo na primjerak razreda `JTextField` (slika 16.).



Slika 16. Primjerci razreda `JLabel` i `JTextField`

Pritiskom tipke enter dok je fokus unutar primjerka razreda `JTextField`, stvara se događaj i obavještavaju se svi registrirani primjerci razreda koji implementiraju sučelje `ActionListener`. Razred `JTextField` također omogućava dohvaćanje teksta iz datoteke, spremanje teksta u datoteku, kao i dohvat i spremanje teksta u clipboard.

Razred `JPasswordField` je iznimno sličan prethodnom razredu, samo što je u njemu sav sadržaj (tekst) zamjenjen maskom. Iz ove komponente se tekst ne može izrezivati niti kopirati. Konstruktor ovog razreda je sličan prethodnom; `public JPasswordField(String pocetniTekst, int Sirina)`. Tekst iz primjerka ovog razreda može se dohvatiti metodom `getPassword()`.

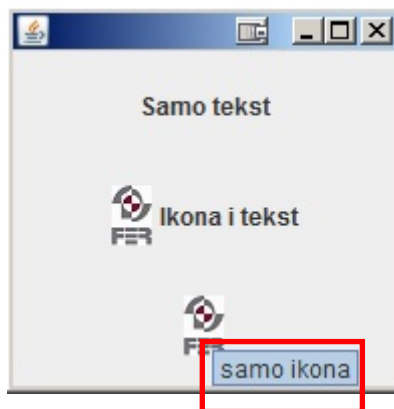
6.5



Slika 17. Primjerak razreda `JPasswordField`

Razred JToolTipText

Još jedna od jednostavnih komponenti je razred JToolTip koji služi sa prikazivanje iskočnoga teksta kada je pokazivač miša iznad određene komponente. Tooltip se u komponentu dodaje pomoću jednostavne metode `setToolTipText(String)`.



Slika 18. Primjerak razreda JToolTip

6.6 Razred JProgressBar

Razred JProgressBar je razred komponente pomoću koje možemo pratiti koliki postotak nekog posla je završen. Konstruktor je `public JProgressBar(int minimum, int maximum)`, u kojem definiramo broj piksela koji traka sadrži. Vrijednost u traci se mijenja metodom `setValue(int)`.



Slika 19. Primjerci razreda JProgressBar

6.7 Razred JSlider

Razred JSlider služi nam za prikaz kliznog gumba. Konstruktor je `JSlider(int orientation, int min, int max, int value)`. Razmak između velikih crtica se postavlja metodom `setMajorTickSpacing(int)`, a između malih `setMinorTickSpacing(int)`. Promjene se osluškuju razredom koji implementira sučelje `ChangeListener`.

7. Naprednije Swing komponente

7.1 Razred JList

Razred JList je razred Swing komponente koja nam omogućuje odabir jedne od predefiniраниh vrijednosti koje su sačuvane unutar strukture podataka samog razreda. Tri osnovna elementa koja definiraju strukturu samog razreda JList su:

- *model podataka* u kojem se nalaze svi podatci unutar razreda JList,
- *crtič ćelije* koji je zadužen za iscrtavanje svake ćelije razreda JList te
- *model selekcije* koji definira na koji način možemo označavati vrijednosti unutar modela podataka razreda JList.

Razred JList ima četiri konstruktora, a tri nama najzanimljivija su:

- `public JList(),`
- `public JList(Object listData[])` i
- `public JList(Vector listData).`

U slučaju da koristimo konstruktor bez argumenata, primjerak razreda JList podacima punimo naknadno. Ako kontejner u koji stavljamo primjerak razreda JList nije dovoljno velik da drži sve vrijednosti, on se mora smjestiti u JScrollPane, inače sve vrijednosti neće biti vidljive.



Slika 20. Primjerci razreda JList

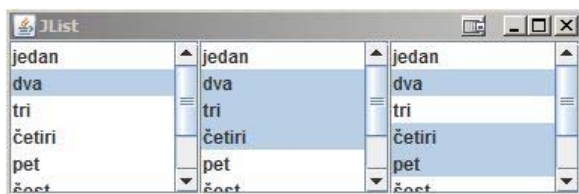
Razred DefaultDataModel nam osigurava model podataka pri kojem podatke čuvamo u strukturi Vector. To je najjednostavniji način čuvanja podataka, i jedini koji ćemo spomenuti u ovom seminaru. Razred DefaultDataModel nam omogućava da bez puno razmišljanja dodajemo podatke, a neke od metoda za dodavanje, mijenjanje i uklanjanje elemenata su:

- `public void add(int index, Object element)` (metoda za dodavanje elemenata u listu),
- `public void setElementAt(Object element, int index)` (metoda za mijenjanje elementa liste), te
- `public void removeElementAt(int index)` (metoda za uklanjanje elementa liste).

Podrazumjevani crtač ćelije kod razreda `JList` je razred `DefaultListCellRenderer`. Razred `DefaultListCellRenderer` je podrazred razreda `JLabel` razreda, stoga se svaka ćelija smatra jednim primjerkom razreda `JLabel`, i prema tome može prikazivati tekst i sliku. Tri modela odabira vrijednosti kod razreda `JList` su

- `SINGLE_SELECTION` u kojem odjednom može biti odabrana samo jedna vrijednost,
- `SINGLE_INTERVAL_SELECTION` u kojem odjednom može biti odabran samo interval vrijednosti te
- `MULTIPLE_INTERVAL_SELECTION` u kojem nema ograničenja odabira vrijednosti.

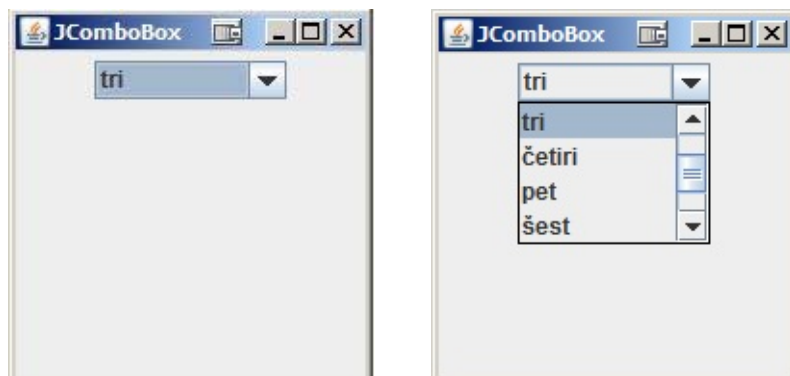
Događaje povezane sa razredom `JList` se sluša pomoću razreda koji implementira sučelje `ListSelectionListener`. Implementirajući metodu `valueChanged()` i pomoću metode `selectedIndex()` možemo saznati točno koja vrijednost je označena (u slučaju da je model odabira vrijednosti liste `SINGLE_SELECTION`)



Slika 21. Razred `JList` sa sva tri modela odabira vrijednosti: `SINGLE_SELECTION` (lijevo), `SINGLE_INTERVAL_SELECTION` (sredina) te `MULTIPLE_INTERVAL_SELECTION` (desno).

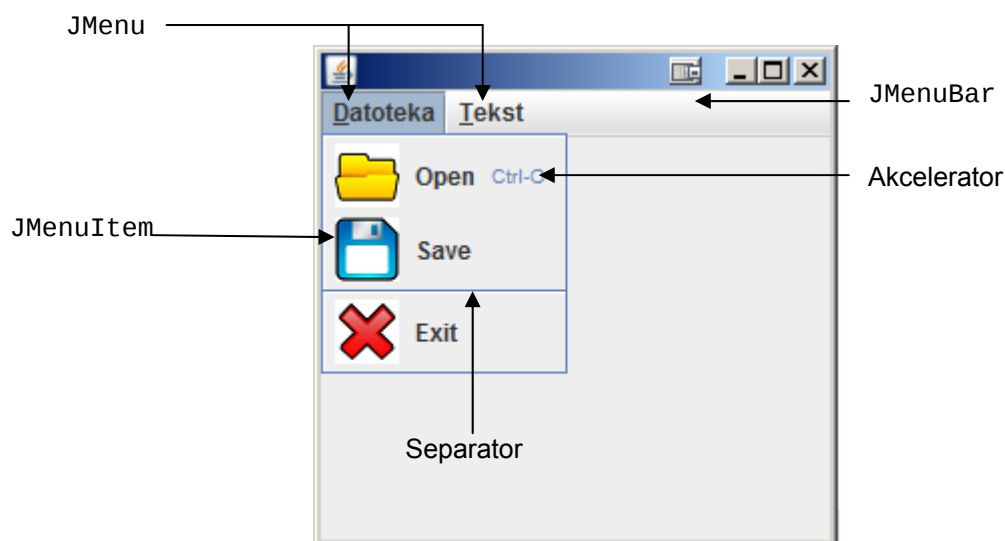
7.2 Razred `JComboBox`

Razred `JComboBox` je razred Swing komponente koja nam omogućuje odabir jednog od predefiniranih vrijednosti iz padajućeg izbornika. Kada je zatvoren, primjerak razreda `JComboBox` je ustvari primjerak razreda `JLabel` u kojega je integriran padajući izbornik koji je ustvari primjerak razreda `JList`. U slučaju da želimo dodati vlastitu vrijednost, razred `JComboBox` nas opskrbljuje primjerkom razreda `JTextField`. Razredi `JList` i `JComboBox` su veoma slične komponente. Neke od sličnosti su model podataka (stoga se elementi dodaju na isti način), crtač ćelije i drugi. Bitna razlika je implementacija sučelja `KeySelectionManager`, koji omogućava korisniku da utipkavanjem dođe do vrijednosti unutar primjerka razreda `JComboBox` koji se poklapa sa nizom utipkanim u tipkovnicu.



Slika 22. Primjerak razreda `JComboBox` zatvoren (lijevo) i otvoren (desno)

7.3 Izbornici

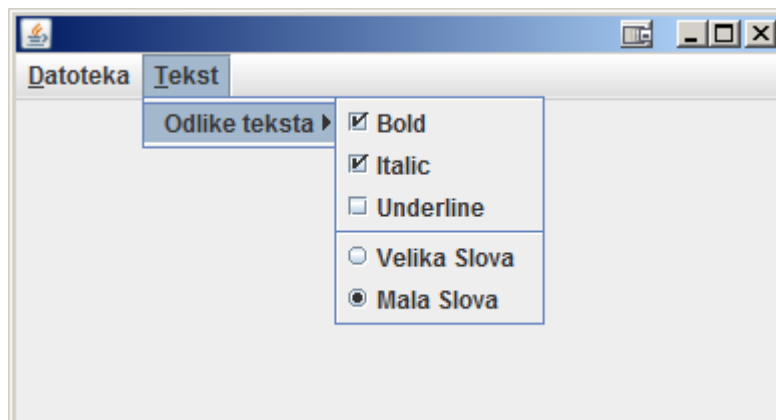


Slika 23. Primjer klasičnog izbornika u grafičkom sučelju

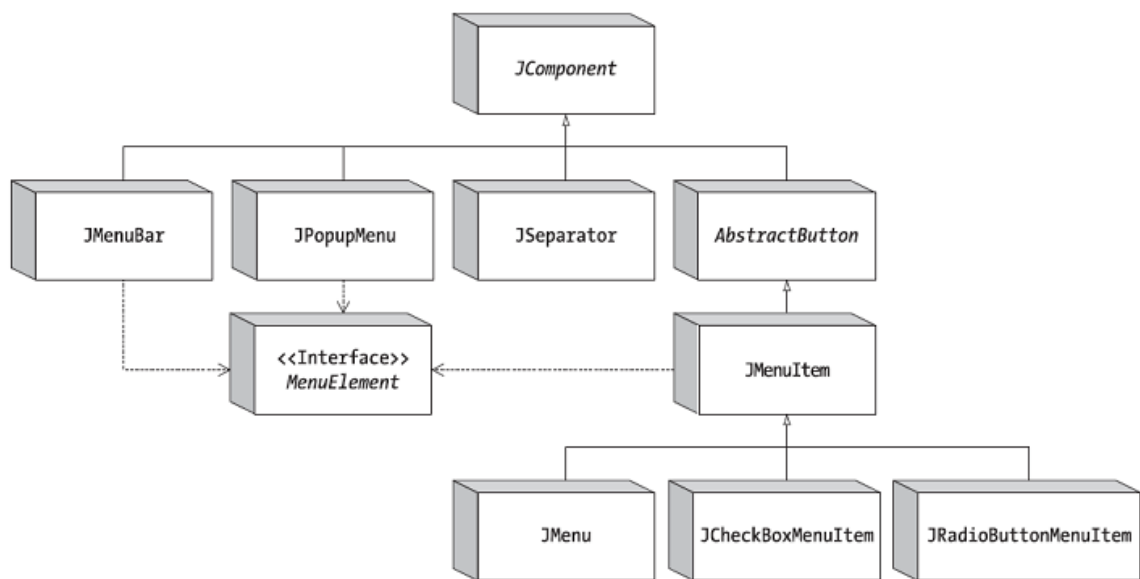
Procedura stvaranja izbornika je navedena u sljedećoj listi.

1. Za padajući izbornik, stvaramo primjerak razreda `JMenu` i dodajemo ga u primjerak razreda `JMenuBar`.
2. Za elemente izbornika, stvaramo primjerke razreda `JMenuItem` i dodajemo ih u primjerak razreda `JMenu`.
3. Za podizbornike, primjerak razreda `JMenu` dodajemo u primjerak razreda `JMenu`, te u njega ubacujemo primjerak razreda `JMenuItem`.
4. Kada kod se primjerak razreda `JMenu` odabere (obično klikom mišem), komponente se prikazuju u primjerku razreda `JPopupMenu`.
5. Separatore se dodaje `addSeparator()` metodom.

U izbornike također možemo dodavati primjerke razreda `JMenuCheckboxItem` i `JMenuRadioButton` po istim pravilima koja vrijede za razrede `JCheckBox` i



Slika 24. Primjerci razreda `JMenuCheckboxItem` i `JMenuRadioButton`

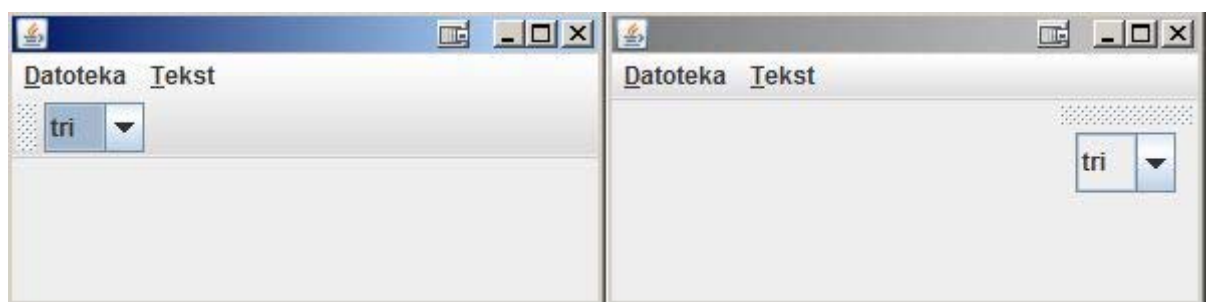


JRadioButton (organizacija u ButtonGroup).

Slika 25. Stablo nasljeđivanja razreda pomoću kojih se grade izbornici. Preuzeto iz [1].

Još jedna veoma korisna i jednostavna komponenta za prikaz izbornika je razred JToolBar. Razred JToolBar je osnovni dio svakog modernog grafičkog sučelja. Ova komponenta može u sebi sadržavati bilo koju drugu komponentu, te biti horizontalno ili vertikalno orijentirana. Četiri konstruktora razreda JToolBar konstruktora su:

- JToolBar(),
- JToolBar(int orientation),
- JToolBar(String name)te



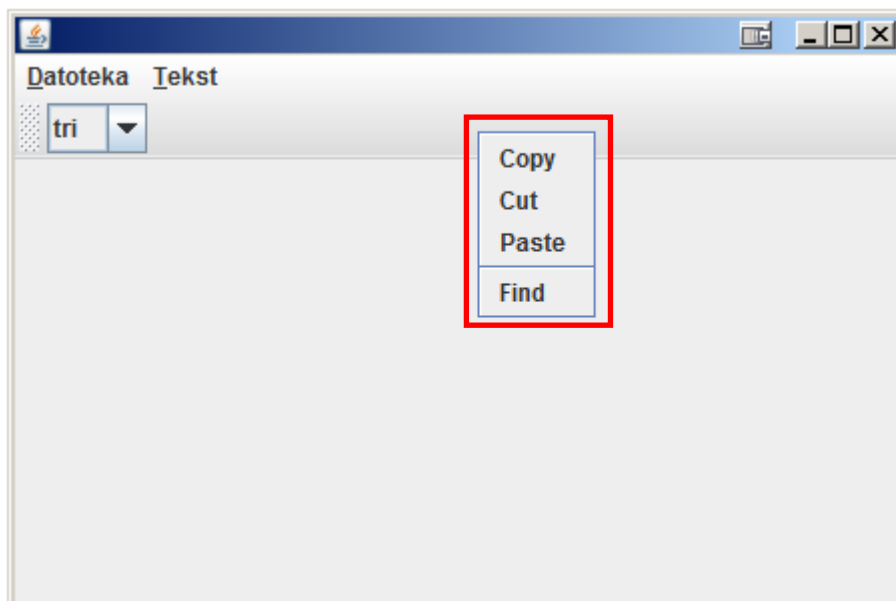
- JToolBar(Sting name, int orientation).

Razred JToolBar također posjeduje već prije upoznatu metodu za dodavanje separatora addSeparator.

Zadnji razred koji ćemo spomenuti u okviru ovog podpoglavlja je razred JPopupMenu. Ova komponenta služi kao kontejner padajućem izborniku kojem se pristupa pritiskom na desnu tipku miša. Primjerak razred JPopupMenu se koristi na sličan način kao primjerak razreda JMenu. Razlika je u tome što primjerak razreda

Slika 26. Primjerak razreda JToolBar u horizontalnoj (lijevo) i vertikalnoj orijentaciji (desno)

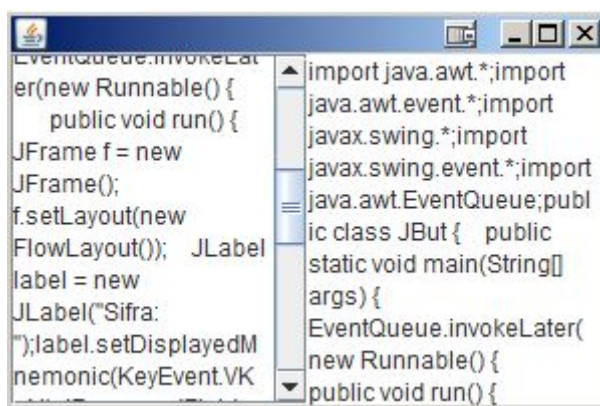
JPopupMenu nije dovoljno konstruirati, nego ga je potrebno i povezati sa komponentom iznad koje će se prikazati metodom `setComponentPopupMenu()`.



Slika 27. Primjerak razreda JPopupMenu

8. Napredne tekstualne komponente

Razred `JTextArea` je razred tekstualne komponente koja služi za prikaz višerednog teksta. Poput razreda `JList`, razred `JTextArea` ne podržava scrollanje, te ako želimo tu funkcionalnost, moramo ga dodati u primjerak razreda `JScrollPane`. Za ovu komponentu postoji šest konstruktora, a nama najzanimljiviji je `public JTextArea(Document document, String text, int rows, int columns)`. Sav tekst unutar primjerka razreda `JTextArea` je istog stila (font, veličina i slično).



Slika 28. Primjerak razreda JTextArea

Spomenut ćemo i razrede `JEditorPane` i `JTextPane`. To su razredi naprednih tekstualnih komponenti, koje omogućavaju da tekst unutar njih bude različitih stilova.

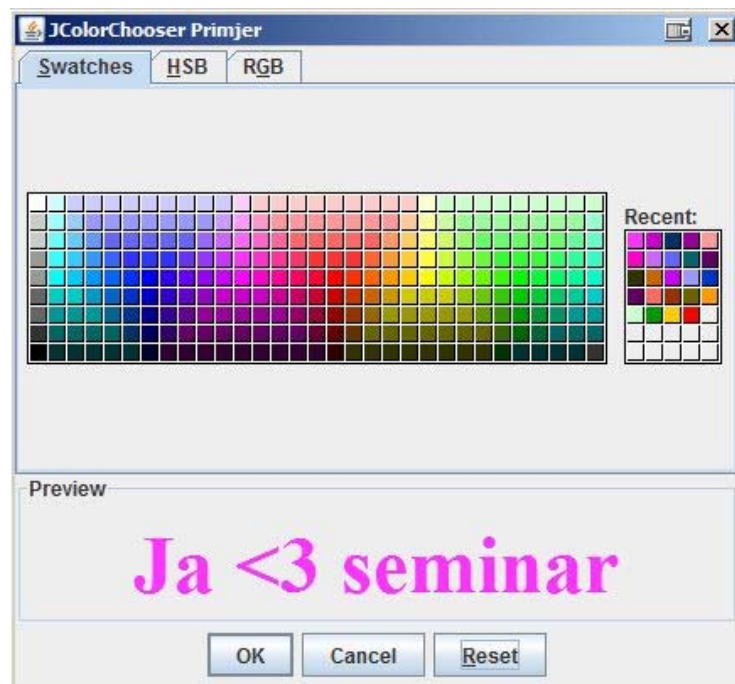
9. Kratki pregled ostalih komponenti tehnologije Swing

Razred `JColorChooser` (slika 29) je razred komponente koja od korisnika zahtjeva da odabere boju. Komponenta se sastoji od tri panela za odabir boje (Swatches, HSB, RGB). Pri dnu komponente se također nalazi okvir sa primjerkom odabrane boje (slika 29.), čiji sadržaj se može uređivati. Panel se najjednostavnije prikazuje metodom `showDialog()`, koja ujedno vraća vrijednost izabrane boje.

Razred `JTable` (slika 30) je razred komponente pomoću koje možemo prikazivati podatke u tabličnom obliku. Nazive stupaca sadrži druga komponenta, primjerak razreda `JTableHeader`, i ona se automatski prikazuje ako se primjerak razreda `JTable` umetne u primjerak razreda `JScrollPane`.

Još jedna napredna komponenta je razred `JFileChooser` (slika 32), koja nam služi za odabir datoteke unutar diska.

Razred `JTree` (slika 31) je razred komponente koja podržava prikaz hijerarhijski organiziranih podataka. Sami podatci u njemu su oblika `Vector`, polja ili `Hashtable`.

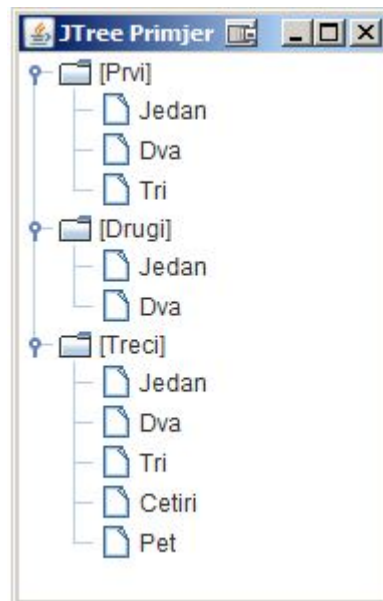


Slika 29. Primjerak razreda `JColorChooser`

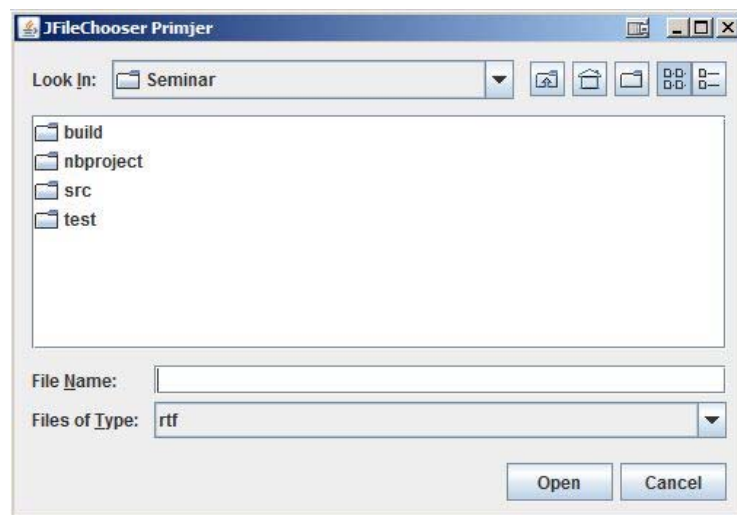


JMBAG	Visina	Težina	Spol
0036447210	179	70	Ž
0036491093	189	75	M
0036415719	185	90	M
0036409285	150	50	Ž
0036492825	200	120	M

Slika 30. Primjerak razreda `JTable`



Slika 31. Primjerak razreda `JTree`



Slika 32. Primjerak razreda `JFileChooser`

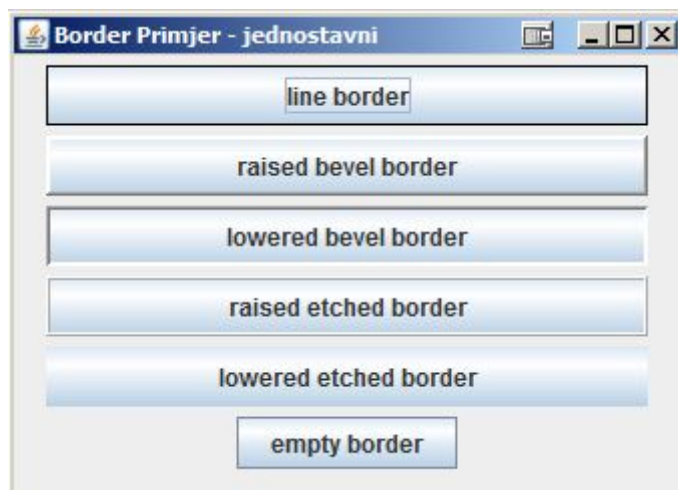
10. Uređivanje okvira komponente

Swing komponente nude opciju prilagođavanja izgleda okvira komponente, koje je sadržano u svojstvu `border` komponente. Svojstvo `border` čuva referencu na jedan od razreda okvira, i svaka komponenta ima `setBorder()` i `getBorder()` metodu. Kada je stvorena, komponenta nema nikakav specifičan okvir, nego izgled okvira ovisi o njenom trenutnom stanju. Primjerice, okvir primjerka razreda `JButton` ovisi dali o tome dali je tipka neoznačena, označena ili stisnuta. Ako nismo postavili okvir komponente, a pozovemo metodu `getBorder()`, dobiti ćemo povratnu vrijednost `null`. Okvir komponente možemo vratiti na podrazumijevanu vrijednost pozivanjem metode `setBorder(null)`.

Najjednostavniji način stvaranja i korištenja okvira jest pomoću razreda `BorderFactory`. Ovaj razred nam omogućava stvaranje okvira pomoću jednostavnih metoda, i on pamti neke češće stvarane okvire. Time se izbjegava ponovno stvaranje često korištenih primjeraka razreda koji se koriste kao okviri komponente. Razred `BorderFactory` omogućava stvaranje devet različitih vrsta bordera. Metode za stvaranje jednostavnih okvira i njihov izgled su prikazani na slici 33 i u tablici 7. Više o samim okvirima komponente može se naći u [3].

Tablica 7. Metode za stvaranje jednostavnih okvira i njihov prikaz

Gumb na slici 33.	Metoda
Prvi	<code>BorderFactory.createLineBorder(Color.BLACK)</code>
Drugi	<code>BorderFactory.createBevelBorder(BevelBorder.RAISED)</code>
Treći	<code>BorderFactory.createBevelBorder(BevelBorder.LOWERED)</code>
Četvrti	<code>BorderFactory.createEtchedBorder(EtchedBorder.RAISED)</code>
Peti	<code>BorderFactory.createEtchedBorder(EtchedBorder.LOWERED)</code>
Šesti	<code>BorderFactory.createEmptyBorder()</code>

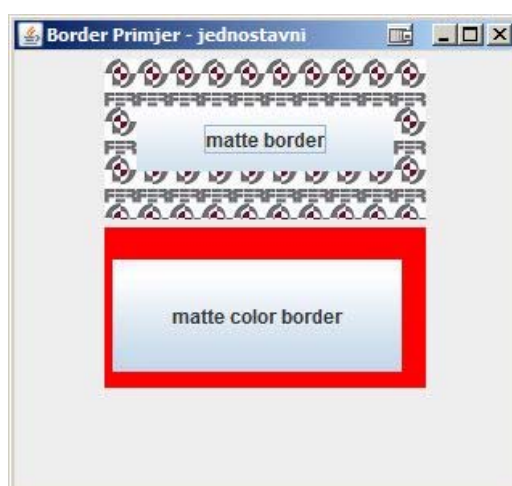


Slika 33. Primjerci različitih okvira komponente

Razred `MatteBorder` iscrtava boju ili ikonu u onoliko piksela koliko specificiramo.

Tablica 8. Razred `MatteBorder`

Gumb na slici 34.	Metoda
Prvi	<code>BorderFactory.createMatteBorder(-1, -1, -1, -1, ikona)</code>
Drugi	<code>BorderFactory.createMatteBorder(20, 5, 10, 15, Color.red)</code>



Slika 34. Primjerci razreda `MatteBorder`

Razred `TitledBorder` nam omogućava umetanje teksta u okvir komponente, kao što je vidljivo na slici 35.



Slika 35. Primjerci razreda `TitledBorder`

Zadnji okvir koji ćemo spomenuti jest predstavljen razredom `CompoundBorder`, koji je ustvari kombinacija dva okvira, unutarnjeg i vanjskog. Svaki od ta dva također može biti primjerak razreda `CompoundBorder`, što na praktički omogućuje kombiniranje bezbroj okvira.

U slučaju da u okvir želimo staviti više komponenti, najjednostavniji način je da komponente umetnemo u jedan primjerak razreda `JPanel`, te onda uređujemo okvir tog primjerka, kako je prikazano na slici 36.



Slika 36. Uređivanje okvira primjerka razreda `JPanel`

11. Zaključak

Prema istraživanjima, 2006. godine 47% aplikacija koje su proizvedene u Sjevernoj Americi imale su grafička sučelja izgrađena pomoću tehnologije Swing. Premda je informatička tehnologija područje stalnog napredka, ovaj udio je i danas zasigurno značajan. Koristeći tehnologiju Swing, prosječan programer je u mogućnosti značajno smanjiti opseg posla potreban za modeliranje grafičkog sučelja. Nastao iz tehnologije AWT, Swing u sebi sadrži mnoge često korištene komponente koje su prilagođene i optimirane za korištenje na raznim platformama. Swing podilazi i neiskusnim i dugogodišnjim programerima: koristeći podrazumjevanje modele, programer je u mogućnosti inicijalizirati i koristiti komponentu uz par linija koda, dok je iskusnijim programerima omogućeno pisanje vlastitih modela, uz koje se svaka komponenta pretvara u moćnu grafičku alatku.

12. Literatura

1. Deitel H. M., Deitel P.J. *Java: How to program*. Third Edition. Upper Saddle River, New Jersey: Prentice Hall, 1999.
2. Zukowski J. *The Definite Guide to Java Swing*. Third Edition. New York: Apress, 2005.
3. Fischer P. *An Introduction to Graphical User Interfaces with Java Swing*. Harlow: Pearson Education, 2005.
4. Oracle. *Trail: Creating a GUI With JFC/ Swing (The Java Tutorials)*, <http://download.oracle.com/javase/tutorial/uiswing/>, datum pristupa: 7.4.2011.
5. Oracle. *A Visual Guides to Swing Components (Java Look and Feel)*, <http://download.oracle.com/javase/tutorial/ui/features/components.html>, datum pristupa: 7.4.2011.
6. Oracle. *JComponent (Java 2 Platform SE v1.4.2)*, <http://download.oracle.com/javase/tutorial/ui/features/components.html>, datum pristupa: 7.4.2011.
7. Oracle. *How to Use Root Panes*, <http://download.oracle.com/javase/tutorial/uiswing/components/rootpane.html>, datum pristupa: 7.4.2011.

13. Sažetak

Tehnologija Swing nastala je iz tehnologije AWT. Ispravljajući većinu problema prethodne tehnologije, Swing omogućava isti izgled aplikacije na svim platformama. Temelj svake Swing komponente je razred `JComponent`. On je izgrađen tako da omogući podršku za obradu događaja. Iako je razred `JComponent` osnovni razred tehnologije Swing, ne nasljeđuju svi razredi koje koristimo u Swing projektima iz tog razreda. Neki primjeri su vršni kontejneri, te razredi i sučelja zadužena za obradu događaja.

Kontejneri su komponente koje su zadužene za držanje drugih komponenti. Neke od njih su vršni kontejneri, i oni su temelj svakog grafičkog sučelja. Neki razredi vršnih kontejnera su `JFrame`, `JDialog` i `JApplet`. Također, postoje i kontejneri opće namjene. Neki od njih su predstavljeni razredima `JPanel`, `JToolBar` i `JScrollPane`. Valja spomenuti i kontejnere opće namjene, koji imaju specifične zadatke u grafičkom sučelju, a predstavljeni su razredima `JInternalFrame`, `RootPane` i drugi.

Razred `JLabel` je komponenta koja prikazuje jednostavno područje sa tekstom i slikom. Razred `JButton` predstavlja tipku koja se može stisnuti a u sebi također može sadržavati tekst i sliku. Razredi `JCheckBox`, `JRadioButton` i `JToggleButton` su komponente koje mogu biti trajno označene, te se često povezuju u `ButtonGroup`, kako bi osigurali da samo jedna može biti označena istovremeno. Razredi `JTextField` i `JPasswordField` su komponente za unos teksta, razred `JComboBox` za odabir jednog elementa iz padajućeg izbornika, razred `JToolTipText` za prikaz iskočnog teksta kada je miš postavljen iznad određene komponente, razred `JProgressBar` za prikaz postotka završenosti neke zadatke, razred `JSlider` za odabir vrijednosti unutar nekog intervala pomoću slidera.

Određene komponente su naprednije, ali i ujedno kompliciranije za korištenje. Razred `JList` prikazuje listu iz koje odabiremo jednu ili više opcija, ovisno o modelu odabira. Razred `JTable` prikazuje podatke u obliku tablice, razred `JTree` prikazuje hijerarhijski organizirane podatke u obliku stabla, Razredi `JTextArea`, `JEditorPane` i `JTextPane` omogućuju prikaz teksta u raznim oblicima (čak i u html formatu), dok razredi `JMenu` i `JMenuItem` omogućavaju implementaciju i prikaz izbornika i izborničkih komponenti.

Neke od visoko formatiranih komponenti koje smo spomenuli u ovom seminaru su predstavljene razredima `JColorChooser` i `JFileChooser` koji nam služe, kao što im i samo ime govori, za odabir datoteke sa diska ili boje.

Zadnja stvar koju smo spomenuli su okviri komponente te razred `BorderFactory` i njegove metode, pomoću kojih možemo uređivati okvir komponente.