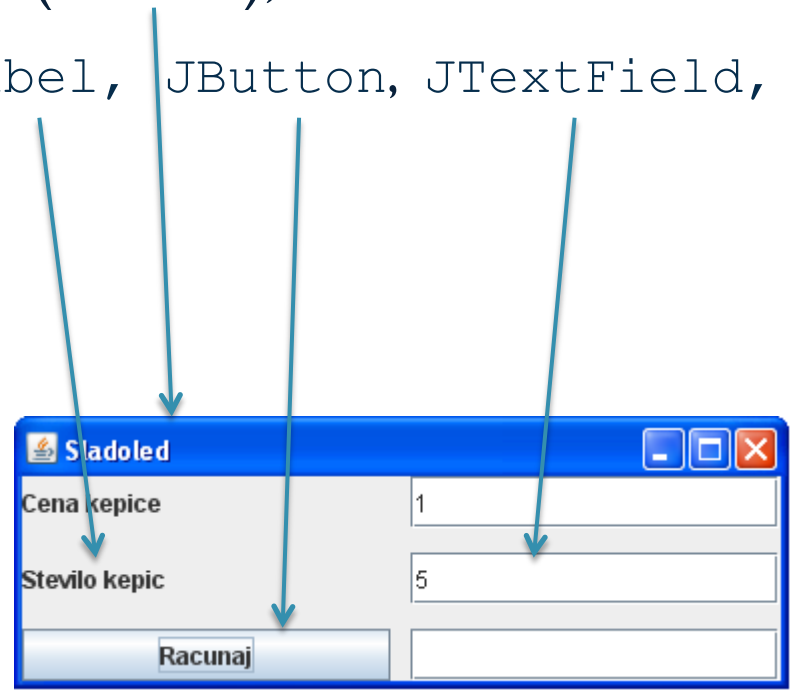
Uporabniški grafični vmesnik (GUI)

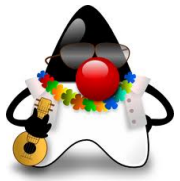
Programiranje 2, Tomaž Dobravec

Okna v Javi

Kaj bi se radi naučili?

- kako izdelati osnovno okno (`JFrame`),
- dodajanje komponent (`JLabel`, `JButton`, `JTextField`, ...),
- razporejanje komponent,
- akcije (pritisk na gumb, ...),
- izdelava celotne aplikacije.





Okna v Javi

- Za delo z okni Java ponuja dve knjižnici:

  `java.awt` (od verzije Java 1.0)

  `javax.swing` (od verzije Java 1.2)

- AWT = Abstract Windowing Toolkit

- Swing (del Java Fundation Classes)

- JavaFX

Swing in AWT

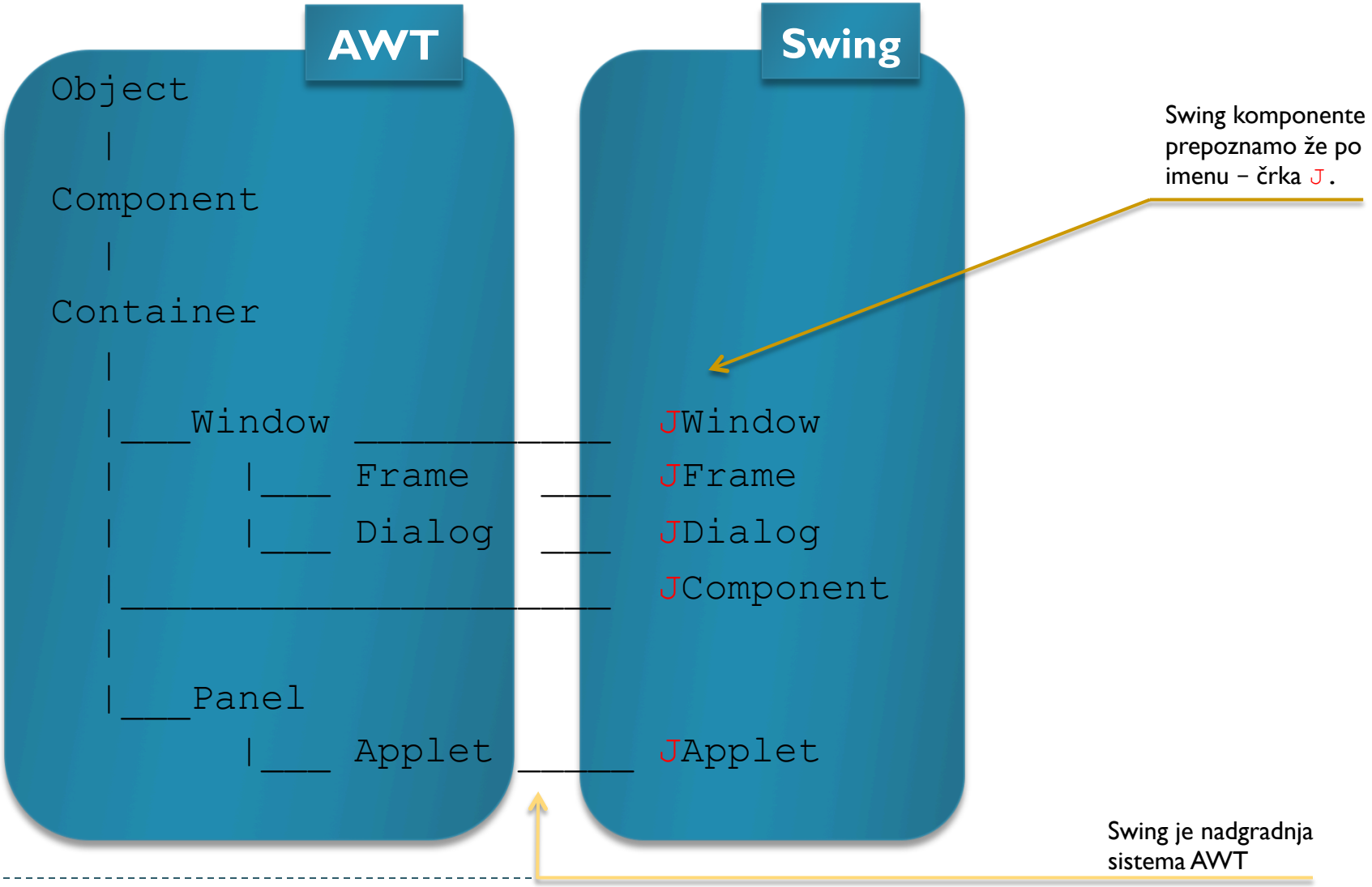
- Prednosti Swing:
  - Swing ponuja več komponent.
  - Swing ponuja lepše komponente.
  - Swing komponente imajo lahko različne (prilagojene) izglede.

- Najpomembnejša razlika med AWT in Swing:

  AWT risanje vseh kontrol prepusti sistemu,
  Swing večino kontrol izriše sam

- AWT je močno odvisen od sistema, Swing le delno. Zato: program, ki uporablja Swing, na vseh sistemih izgleda približno enako.

Swing in AWT


AWT
Swing
Object
Component
Container
Window
Frame
Dialog
Panel
Applet
JWindow
JFrame
JDialog
JComponent
JApplet
Swing komponente prepoznamo že po imenu – črka J.
Swing je nadgradnja sistema AWT

Swing in AWT

- `JWindow`, `JFrame`, `JDialog` so izpeljani iz `Window`, `Frame`, `Dialog`, zato **niso** neodvisni od sistema.
- Vse ostale kontrole knjižnice Swing pa **so** neodvisne od sistema (za njihovo risanje poskrbi java)!

- AWT še vedno obstaja, Swing je odvisen od njega.
- Priporočilo: Pri pisanju grafičnih programov uporabi Swing.
- V programu nikoli ne uporabljaj hkrati AWT in Swing komponent.
- AWT uporabi le, če Swing željene funkcionalnosti nima.

Osnovni vsebniki

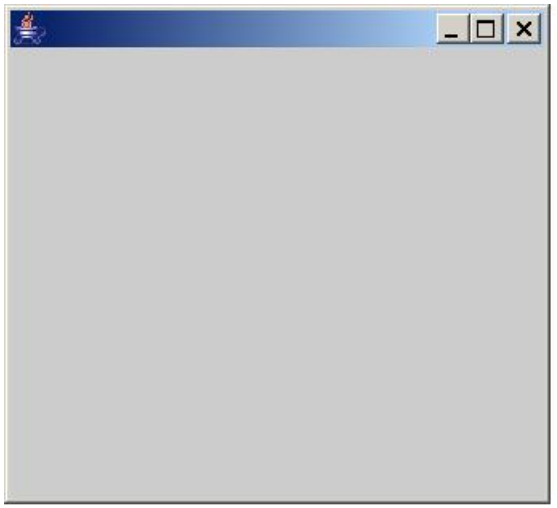
- `JFrame, JDialog in JWindow` **so osnovni vsebniki (top-level containers).**

`JWindow`

- Okno brez glave in obrobe
- Uporablja se redko

Razred `JFrame`

`JFrame`



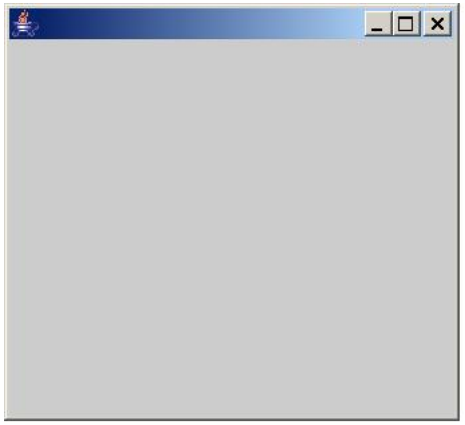
- `JFrame` je glavno okno programa.

- Vsak Swing program naj bi imel vsaj eno `JFrame` okno.

```
import javax.swing.*;

JFrame okno = new JFrame("Test");
okno.setVisible(true);
```

Razred JDialog

`JDialog`



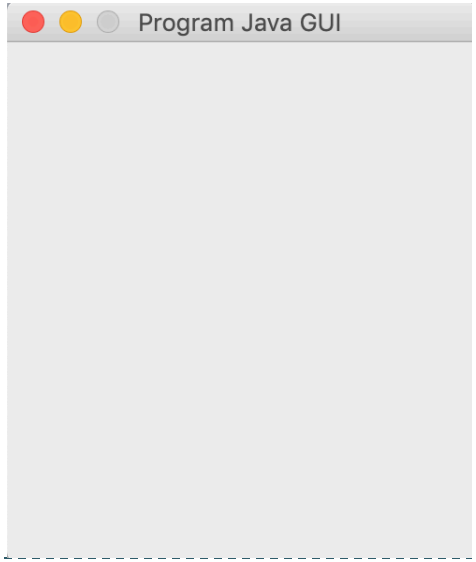
- `JDialog` je okno, ki se odpre med izvajanjem programa.
- Objekt tipa `JDialog` ni samostojen, podrejen je glavnemu `JFrame` objektu.
- Objekt tipa `JDialog` se lahko prikaže tudi **modalno** (dokler se dialog ne zapre, glavnega programa ne moremo uporabljati).

Metoda razreda `JFrame`

Nekatere metode razreda `JFrame`:

```
public void setTitle(String title)
public void setSize(int width, int height)
public void setLocation(int x, int y)
public void setResizable(boolean resizable)
public boolean isResizable()
```

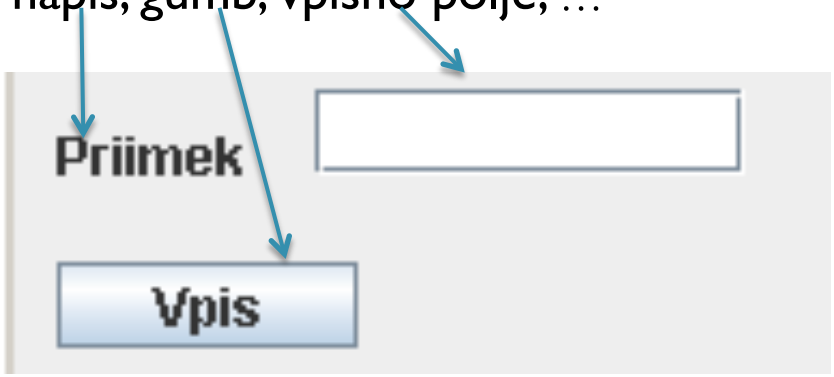
```
JFrame okno = new JFrame();
okno.setTitle("Program Java GUI");
okno.setSize(250,300);
okno.setLocation(300,300);
okno.setResizable(false);
okno.setVisible(true);
```



Swing komponente

- Primer komponent:

  napis, gumb, vpisno polje, …



- Komponente prikažemo v oknu (npr. v `JFrame`).
- Swing komponente so naslednice razreda `JComponent`.

```
JComponent
   |___ JLabel
   |___ JList
   |___ JPanel
   |___ JMenuBar
   |___ JPopupMenu
   |___ JScrollPane
   |___ JAbstractButton
   |          |___ JToggleButton
   |          |       |___JCheckBox
   |          |       |___JRadioButton
   |          |___ JButton
   |          |___ JMenuItem ___ JMenu
   |___ JTextComponent
          |___ JTextArea
          |___ JTextField ___ JPasswordField
```

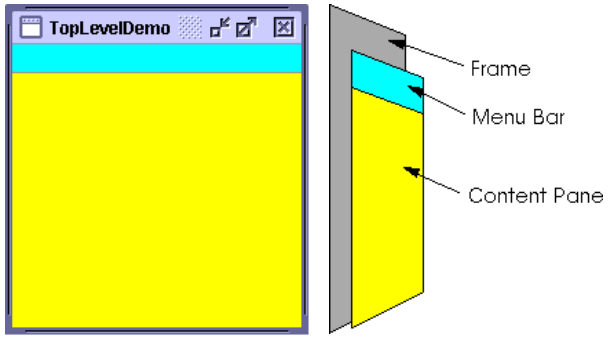
Za grafičen prikaz Swing komponen glej: A Visual Guide to Swing Components (https://cutt.ly/ChRO7L6)

Nekatere metode razreda `JComponent`

```
public void setSize(int width, int height)

public void setLocation(Point p)

public void setBounds(int x, int y, int width,  int height)

public void setEnabled(boolean enabled)

public void setForeground(Color fg)

public void setFont(Font font)
```

Dodajanje komponent na vsebnik

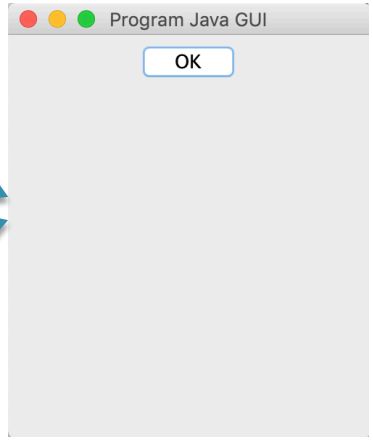
- Vsebniki (*angl. containers*) so odlagalne površine, na katere lahko odlagamo komponente;
- osnovni vsebnik na oknu se imenuje "*Content Pane*"; vse, kar želimo prikazati v oknu, moramo odložiti na ta vsebnik;
- glavni vsebnik dobimo z metodo `getContentPane();`
- za odlaganje komponent uporabimo metodo `add()`.




```
JButton okGumb = new JButton("OK");
Container vsebnik = okno.getContentPane();
vsebnik.add(okGumb);
```

(bolj preprosto, učinek je enak!)

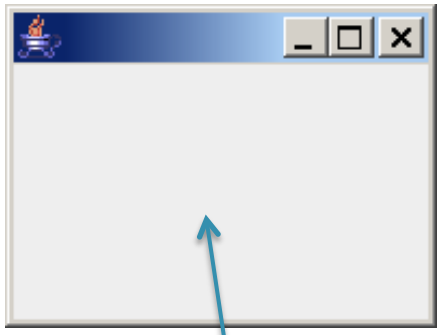
```
JButton okGumb = new JButton("OK");
okno.add(okGumb);
```



Komponenta `JPanel`

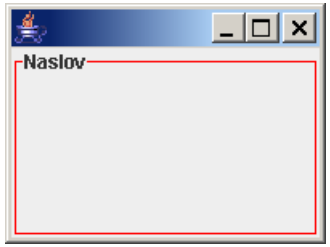
- Na osnovni vsebnik običajno odložimo enega ali več pomožnih vsebnikov (komponente tipa `JPanel`).

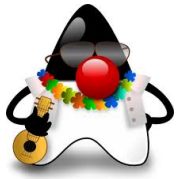
- `JPanel` je običajno neviden; uporabljamo ga, da nanj odlagamo druge komponente.



Koliko panelov je na osnovnem oknu?

- `JPanel` ima lahko okvir (border), okvir ima lahko naslov (title).
- Za izdelavo okvirov uporabim `BorderFactory`.





Razporejanje komponent na vsebniku

Kako določimo položaj in velikost komponente, ki jo odložimo na vsebnik?

Uporabimo lahko eno od dveh možnosti:

- absolutno pozicioniranje,

Za vsako komponento posebej poveš, kje na formi se bo nahajala in kako velika bo.

- pozicioniranje s pomočjo razporejevalnika.

Komponento dodaš na formo in pustiš da mesto zanjo določi izbrani razporejevalnik.

Absolutno pozicioniranje

- Javi povemo, da želimo komponente razporejati sami:

```
Container vsebnik = mojeOkno.getContentPane();
vsebnik.setLayout(null);
```

- vsaki komponenti posebej določim položaj in velikost

```
mojGumb.setBounds(10,20,50,30); //  polozaj:  x=10, y=20
                //  velikost: širina=50, višina=30
```

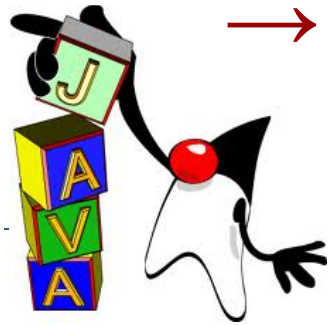
- komponento postavimo na okno

```
vsebnik.add(mojGumb);
```

Naloga

Absolutno pozicioniranje

okna/AbsPos.java



Napiši program, ki z absolutnim pozicioniranjem komponent izriše okno, kot prikazuje spodnja slika

Razporejevalniki

- Razporejevalnik pomaga (sodeluje) pri razporejanju komponent
- Komponente razporeja glede na
  - način razporejanja in
  - vrstni red dodajanja komponent.
- Velikost in lego komponent določi razporejevalnik.
- Uporabnik lahko določi le priporočeno velikost (*angl. preferred size*) posamezne komponente.

Razporejevalniki

- FlowLayout ... razporejanje v vrsto
- BorderLayout ... razporejanje v pet con (gor, levo, center, desno, dol)
- GridLayout ... razporejanje v mrežo, (ena komponenta, eno polje)
- GridBagLayout ... razporejanje v mrežo (komponenta lahko v več poljih)
- CardLayout ... razporejanje na eno mesto

Razporejevalnik `FlowLayout`

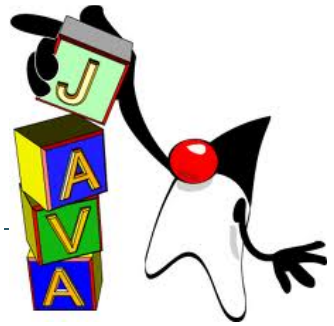
- Komponente se dodajajo ena za drugo v vrsto.

- Če je vrsta prekratka za naslednjo komponento, se ta doda v novo vrsto.
- Če je vrsta predolga, se komponente poravnajo na sredino.
- V konstruktorju za `FlowLayout` lahko določimo poravnavo (privzeto: center) in prostor okoli komponent
- `FlowLayout` je privzet razporejevalnik za `JPanel`

Naloga

Primer `FlowLayout`

`okna/FlowLayout.java`

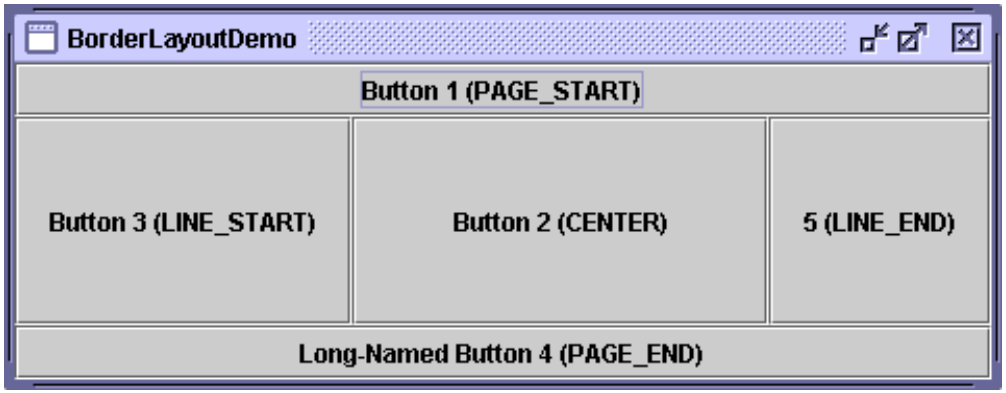


Napiši program, ki na okno postavi pet gumbov, kot prikazuje spodnja slika

Razporejevalnik `BorderLayout`

- BorderLayout predivdeva pet con:

  `PAGE_START`, `PAGE_END`, `LINE_START`, `LINE_END` in `CENTER`



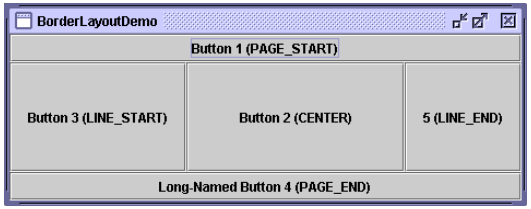
- dodajanje komponente:

```
add(gumb, BorderLayout.PAGE_END);
```

Razporejevalnik `BorderLayout`

- komponente lahko dodamo tudi na manj kot

  5 con (na primer: samo v center in bottom)



- CENTER je glavni:

  - srednja cona zavzame ves prazen prostor;

  - komponenta, za katero eksplicitno ne navedemo mesta:

    ```
    add(gumb);
    ```

    se doda v center;

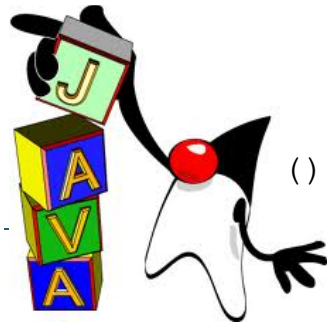
  - če dam več komponent v isto cono, se prikaže le zadnja dodana.

- BorderLayout je privzet razporejevalnik za `JFrame, JDialog` in `JApplet`

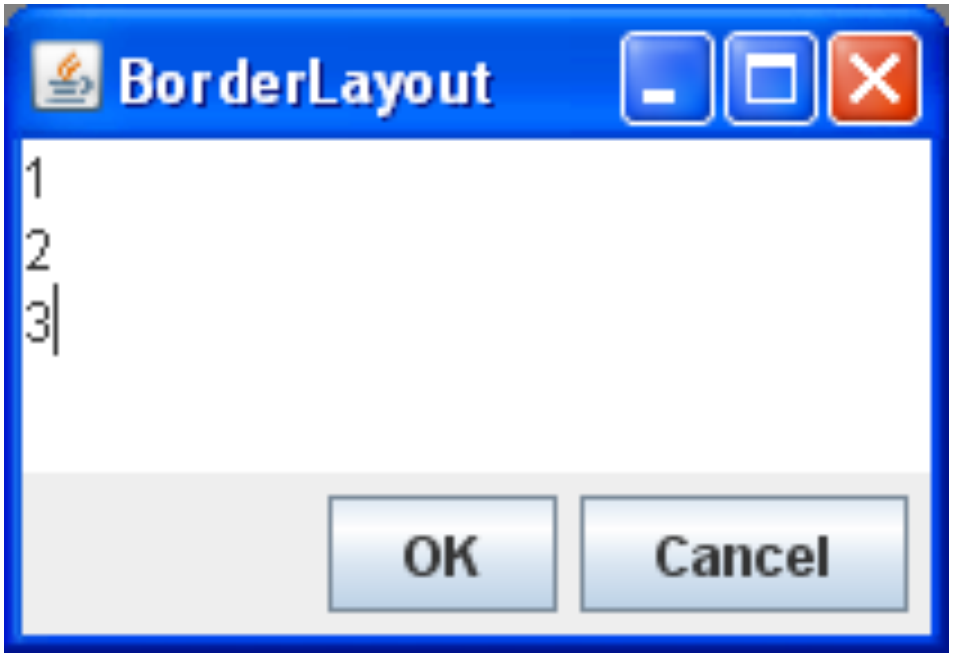
Naloga

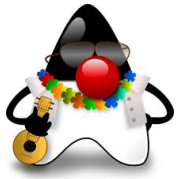
Primer `BorderLayout`

`okna/BorderLayout.java`



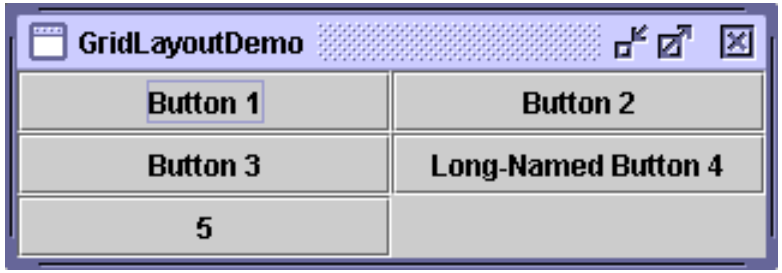
Napiši program, ki nariše okno kot prikazuje spodnja slika (namig: za gumbe uporabi dodatni `JPanel`).





Razporejevalnik `GridLayout`

- Mreža enakih celic: m stolpcev in n vrstic.
- Komponente se razporejajo v celice: vsaka komponenta v eno celico.

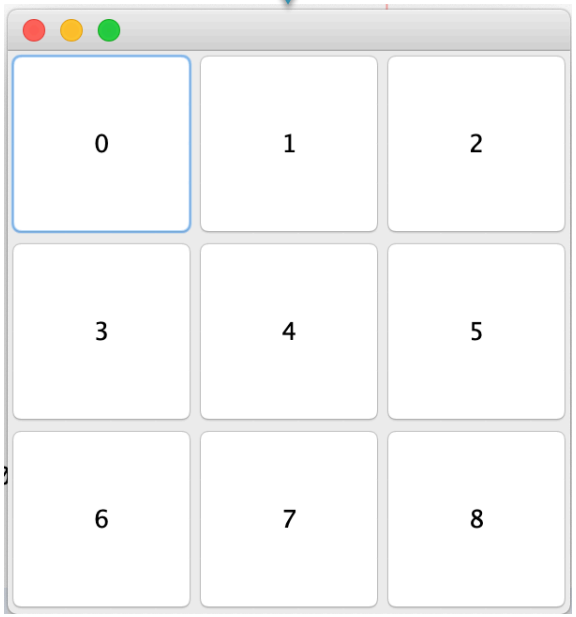


- Komponenta zasede ves prostor v celici.
- Vse celice so enako velike; velikost se določi tako, da ustreza največji komponenti v mreži.

Razporejevalnik `GridLayout`

```
okno.setLayout(new GridLayout(3, 3));
for (int i = 0; i < 9; i++) {
  JButton b = new JButton(Integer.toString(i));
  b.setPreferredSize(new Dimension(70,70));
  okno.add(b);
}
```

- Ukaz `add()` komponento doda v naslednjo prosto celico.
- Polnjenje celic poteka od leve proti desni, od zgoraj navzdol.
- Preskakovanje celic ni možno!
- Neposreden vpis v celico ni možen.



Razporejevalnik `GridBagLayout`

- Razporejanje v mrežo (podobno kot `GridLayout`)
- Ena komponenta lahko zasede več celic mreže.
- Natančno lahko določimo, v kateri celici bo posamezna komponenta.
- Vrstni red dodajanja ni pomemben.
- Pri dodajanju komponent podamo tudi omejitve (*angl. constraints*).

- Omejitve so objekt razreda `GridBagConstraints`

```
vsebnik.setLayout(new GridBagLayout());
GridBagConstraints omejitve = new GridBagConstraints();

// nastavim omejitve
omejtive.gridx=0;
...

vsebnik.add(komponenta, omejitve);
```

Razporejevalnik `GridBagLayout`

- Pomen nekaterih atributov v `GridBagConstraints`

`gridx`, `gridy`	položaj (x in y koordinata celice)
`gridwidth`, `gridheight`	število celic (po širini in višini), ki jih komponenta zaseda
`fill`	raztezanje komponent (H/V/B/N)
`insets`	prazen prostor okoli komponente
`anchor`	poravnava v celici (N/W/E/S/NW,...)
`weightx`, `weighty`	utež, ki določa, kako naj se razdeli prostor med posamezne celice mreže

Razporejevalnik `GridBagLayout`

- Fill – raztezanje (HORIZONTAL, VERTICAL, BOTH, NONE)

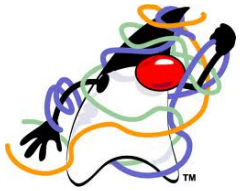
```
omejitve.fill = GridBagConstraints.BOTH;
```

- Insets - prazen prostor (zgoraj, levo, spodaj, desno)

```
omejitve.insets = new Insets(10,0,0,0);
```

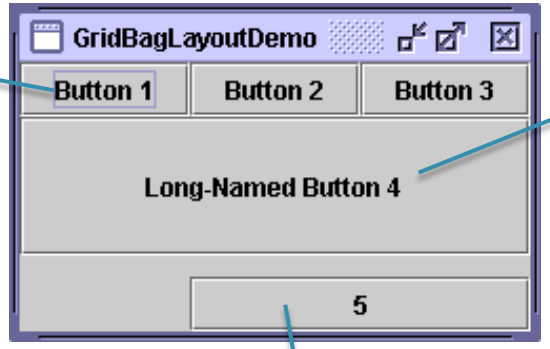
- Anchor (North, West, South, East, NorthWest, ...)

```
omejitve.anchor = GridBagConstraints.NORTH;
```



Razporejevalnik GridBagLayout

```
gridx=0
gridy=0
gridwidth=1
gridheight=1
```



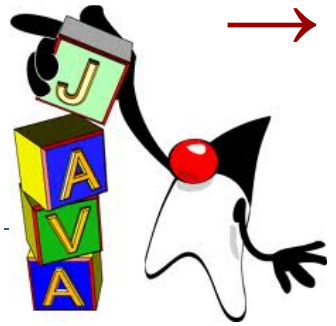

```
gridx=0
gridy=1
gridwidth=3
gridheight=1
fill=HORIZONTAL
```

```
gridx=1                 gridy=2
gridwidth=2             gridheight=1
fill=HORIZONTAL         weighty=1
insets=(10,0,0,0)       anchor=SOUTH
```

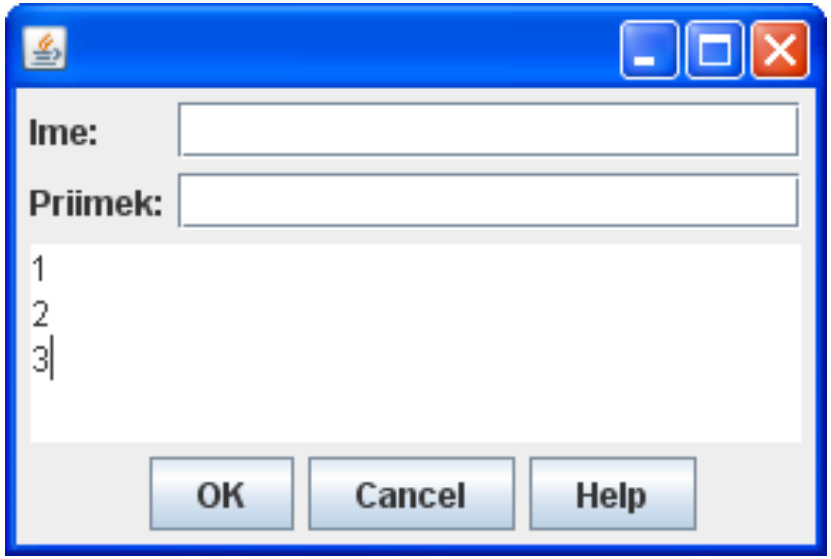
Naloga

Primer `GridBagLayout`

`okna/GridBagLayout.java`



Napiši program, ki nariše okno, kot prikazuje spodnja slika.



Razred `java.awt.Toolkit`

- Objekt razreda Toolkit dobimo s klicem statične metode

```
Toolkit tk = Toolkit.getDefaultToolkit();
```

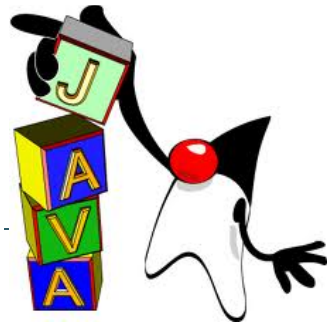
- Nekatere metode:

```
public abstract void beep()

public boolean getLockingKeyState(int keyCode)
public void setLockingKeyState(int keyCode, boolean on)
public abstract Dimension getScreenSize()
```

Naloga

Okno čez ves zaslon

`okna/GridBagLayout.java`

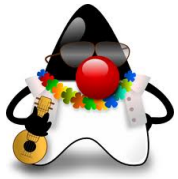


Popravi program `okna/GridBagLayout.java` tako, da se bo okno ob zagonu programa raztezalo čez ves zaslon.

Akcija ob pritisku na gumb

Če želimo, da se ob pritisku na gumb sproži neka akcija, moramo napisati in registrirati ustreznega poslušalca:

```
JButton gumb = new JButton("Klikni me");
gumb.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent ae) {
     // ... akcija, ki se zgodi ob kliku na gumb
  }
});
```



Zapiranje okna

- Okno (`JFrame`) se ob pritisku na križec samo skrije.
- Program se ob tem ne konča (isto okno lahko kasneje s `setVisible()` ponovno prikažemo).
- Rešitev: uporabimo metodo `setDefaultCloseOperation()`

```
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

Okna razreda `JOptionPane`

- Razred `JOptionPane` uporabimo za prikaz standardnih dialogov za prikaz obvestil, branje niza, izbiro opcij, …

- Nekatere `JOptionPane` metode:

Ime metode	Opis
`showConfirmDialog`	Potrditev (Da/Ne, ...)
`showInputDialog`	Vpis besedila
`showMessageDialog`	Okno s sporočilom
`showOptionDialog`	Splošen dialog (veliko opcij)

Vsebina vpisnih polj

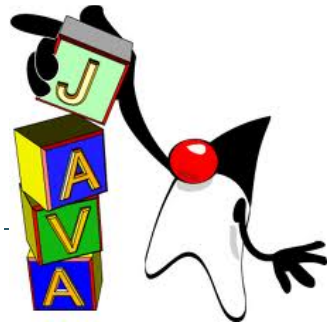
Vsebino vpisnega polja (`JTextField`) preberemo z metodo `getText()`, nastavimo pa z metodo `setText()`.

```
JTextField polje = new JTextField();
polje.setText("Besedilo v polju");
```

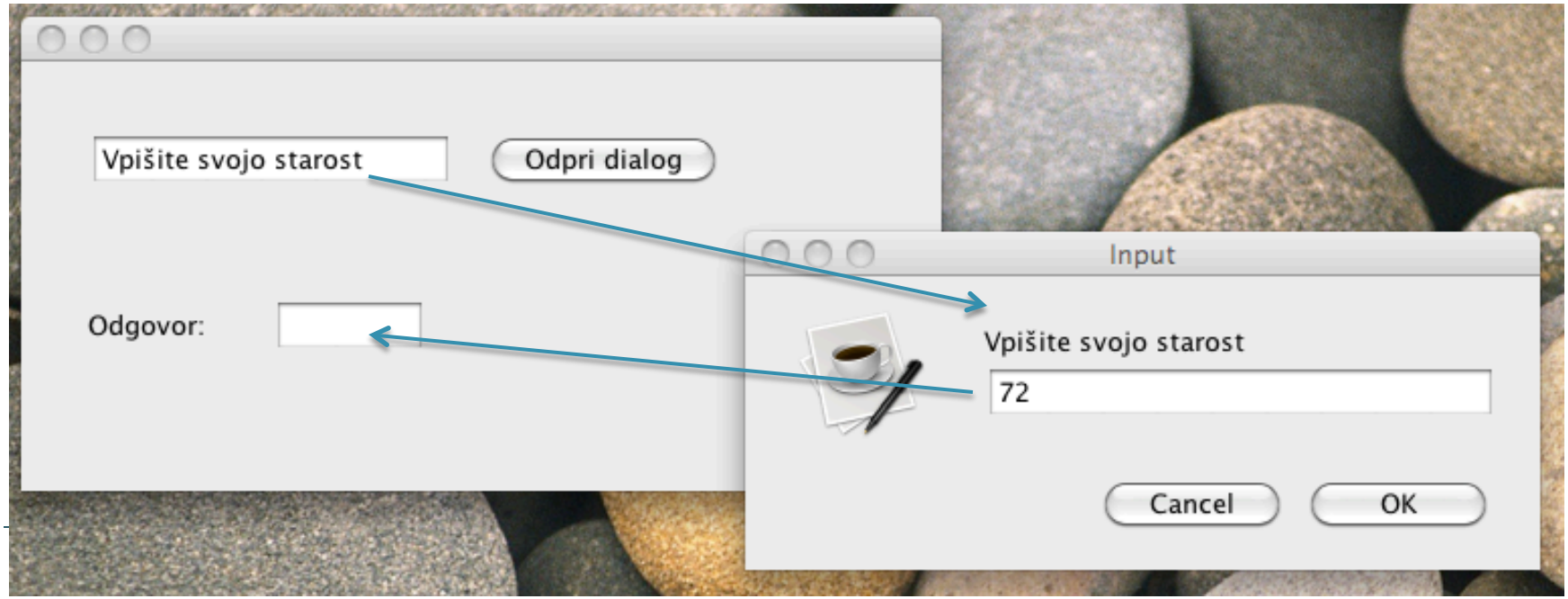
Naloga

Branje in pisanje v vpisna polja

okna/SetGetText.java



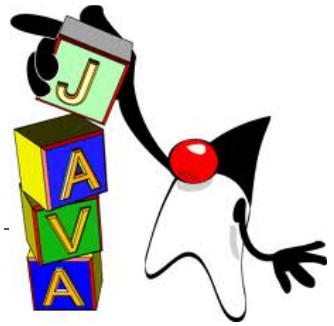
Napišite program, ki ob kliku na gumb "Odpri dialog" odpre dialog za branje niza. Vprašanje v dialogu naj bo enako vsebini prvega vpisnega polja. Ob zaprtju dialoga, naj se odgovor vpiše v drugo vpisno polje.



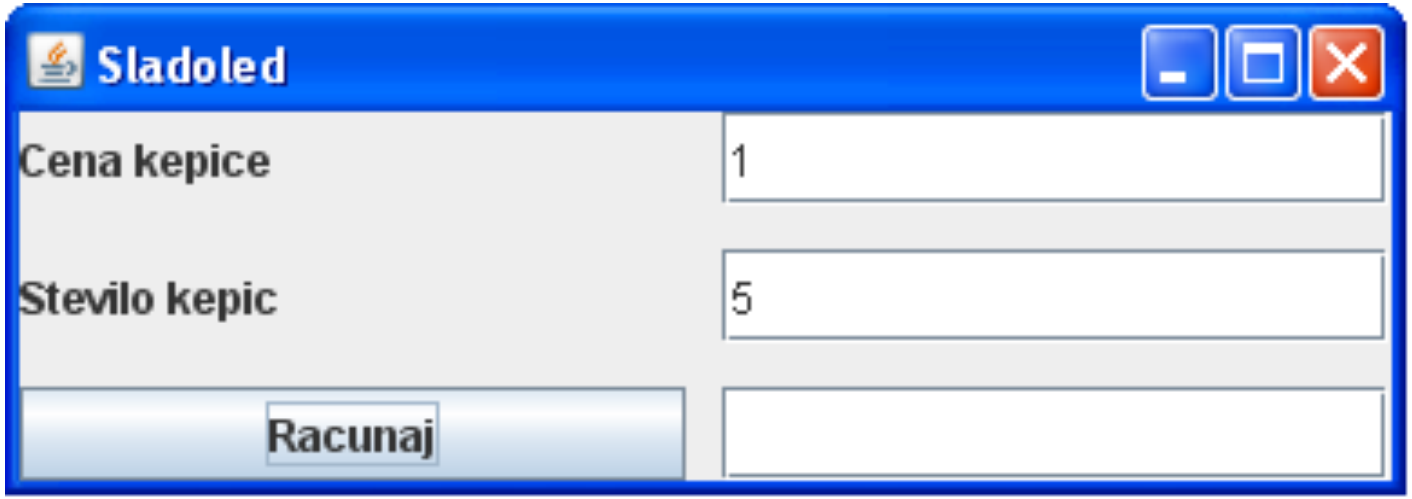
Naloga

Program za izračun cene sladoleda

okna/Sladoled.java



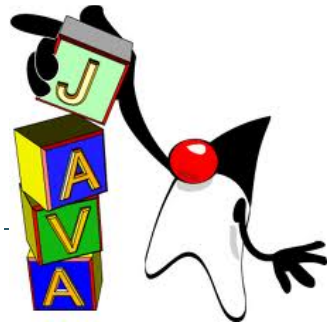
Napiši program za računanje cene sladoleda.



Naloga

Kalkulator

okna/Racunalnik.java



Napiši preprost kalkulator, kot prikazuje spodnja slika; kalkulator naj omogoča operacije +, -, *, /.