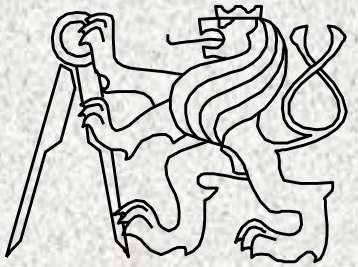
Grafické uživatelské rozhraní v Javě



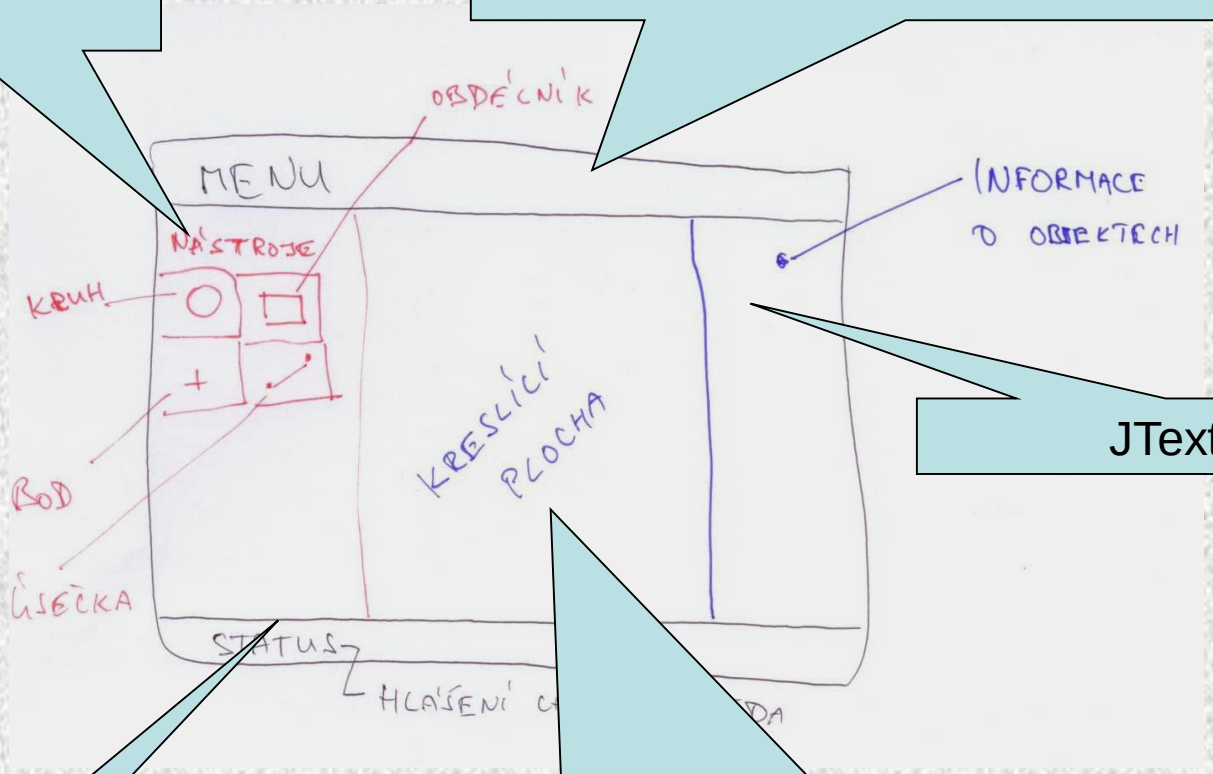
BD6B36PJV 6

Fakulta elektrotechnická
České vysoké učení technické

Grafický návrh - příklad

Menu
soubor – nový, otevřít, uložit, exportovat, konec ...

ToolBar

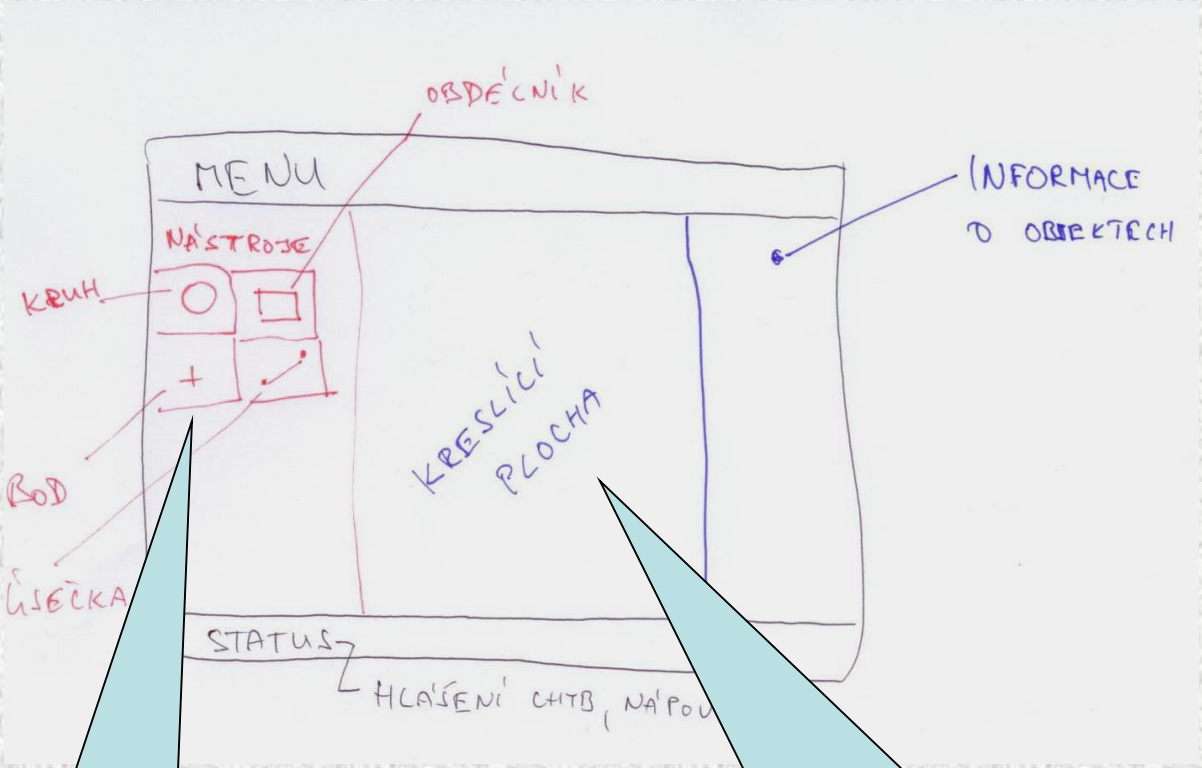


JTextArea

JLabel

Vlastní komponenta, vlastní chování na kliknutí, vlastní vykreslení, ...

Grafický návrh - příklad



FlowLayout či
GridLayout

BorderLayout
zajistí vykreslení všech částí, největší zbytek bude pro střed

Zásady návrhu GUI - 1

- Kvalita GUI podstatně ovlivňuje efektivitu práce uživatele (i negativně) .
- Uživatel podle GUI posuzuje kvalitu aplikace
- Usilujte o jednoduše elegantní návrh s intuitivní a konzistentní funkcionalitou.
- Rozumně s rozměry, barvami a kontrasty - mají asociované významy.
- Respektujte styl a zvyklosti uživatele.
  - Poznejte zkušenosti a prostředí uživatelů (laik vs. expert).
  - Uvažte jak eventuálně hladce dále GUI rozšiřovat.
  - Jednoduchost bývá lepší než složitost - nepřeplácat komponentami.

Zásady návrhu GUI - 2

- Uživatel se nesmí ztratit – vyznačujte stopu jak se tam dostal.
- Nezahltit informacemi a vizuálními podněty – usability testy prototypů.
- Udržovat konzistenci použití komponent.
  - Konzistence mezi aplikacemi – look and feel.
  - Vnitřní konzistence aplikace.
- Komponenty mají váhu – navozují závažnost (velikost, font, barva).
- Uvažte standardy a zvyklosti platforem.
- Uvažte i18n

Zásady návrhu GUI - 2

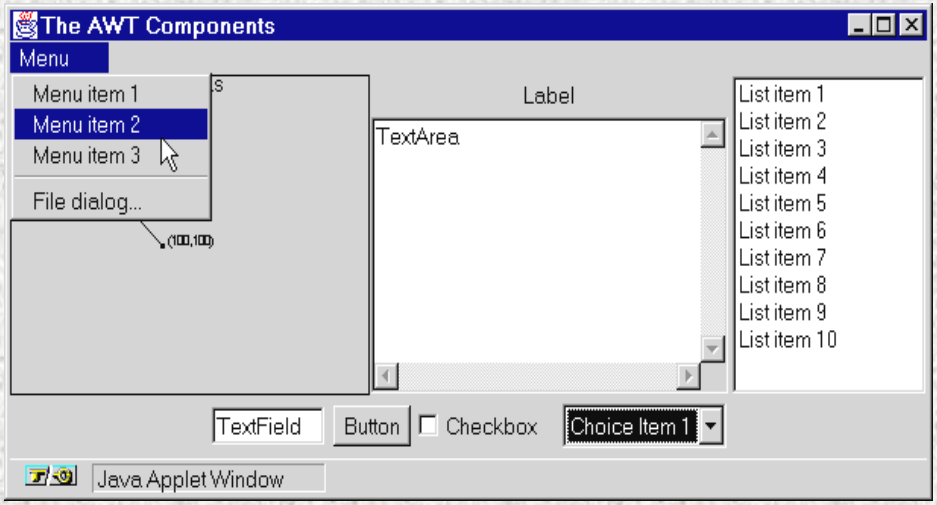
- Uživatel se nesmí ztratit – vyznačujte stopu jak se tam dostal.
- Nezahltit informacemi a vizuálními podněty – usability testy prototypů.
- Udržovat konzistenci použití komponent.
  - Konzistence mezi aplikacemi – look and feel.
  - Vnitřní konzistence aplikace.
- Komponenty mají váhu – navozují závažnost (velikost, font, barva).
- Uvažte standardy a zvyklosti platforem.
- Uvažte i18n (i-nternationalizatio-n)

GUI (Graphical User Interface)

Vizuální a interaktivní komunikaci počítač-člověk podporují balíčky:

- **`java.awt`** - obsahuje:
  - komponenty: tlačítka, textová pole, menu, posuvníky, grafiku ....
  - kontejnery: tj. komponenty, do kterých lze vkládat komponenty,
  - layout managery: rozmisťují komponenty v ploše kontejneru.
- **`java.awt.event`** - události a jejich zachytávání.
- **`javax.swing`** - podstatně vylepšuje GUI, nahrazuje plně java.awt.

Ukázka v awt :



Knihovny pro GUI

- AWT - Abstract Window ToolKit
  - první, těžké(heavyweight),
  - vykreslení zajišťuje platforma – rychlejší, ne vždy vše funguje vše stejně
- Swing
  - doporučené, nové komponenty (tree-view, list box,...),
  - robustní
  - Look and Feel - na **platformě** nezávislý a vypadá stejně na všech platformách a přitom respektuje i18n
  - důsledné oddělení modelu od pohledu
- SWT-Standard Widget Toolkit, Eclipse IBM,
  - podobné AWT (platformově závislé vykreslení)
  - mnoho rozšiřujících vlastností

Knihovny pro GUI

Základní součásti GUI

1. **Komponenty** (dialogové prvky) - v knihovně *javax.swing*

- tlačítka, seznamy, jezdci, textová pole, zatrhávací tlačítka, rádio tlačítka, …
- společné metody pro velikost, barvu, umístění textu, …

2. **Kontejnery** (v oknech) - v knihovně *javax.swing*

- Kontejnery se vkládají do oken
- komponenty musí být umístěny v kontejnerech
- dva základní typy kontejnerů
  - **`JPanel`**– nejjednodušší, přidělí se komponenty (také **`JApplet`**)
  - **`JFrame`** – složitější, ale více možností

3. **Správce rozmístění** (Layout Manager) - v knihovně *javax.swing a java.awt*

- definuje pozici komponent v kontejneru
- postupně, pevná pozice, podle mřížky, sdružování, ..
- vzhled a chování celé aplikace,

4. **Obsluha událostí** (events) - v knihovně *java.awt.event*

Přehled základních prvků GUI

Komponenty

- JButton Tlačítko, událostí je kliknutí na tlačítko
- JCheckBox Zaškrtávací políčko, prvek je/není vybrán,událost po uzavření okna
- JComboBox Rozevírací seznam položek, klepnutím na položku se generuje událost
- JLabel Zobrazení popisku, bez generování události
- JPasswordField Zobrazení hesla, místo vložených znaků se zobrazí hvězdičky
- JRadioButton Přepínač, množina tlačítek, jen jedno lze zvolit, událost po uzavření okna
- JTextField Zadávání textu, událost se generuje po uzavření okna

Kontejnery

- JFrame Kontejner s ohraničením a záhlavím
- JPanel Kontejner bez ohraničení, implicitně rozmístění FlowLayout, jednodušší

Rozvržení, správci - Layout Manager

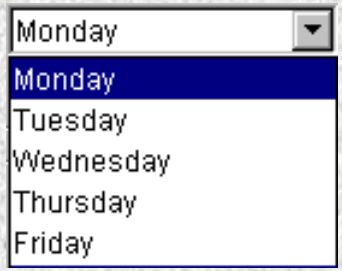
- BorderLayout Rozmístění podle světových stran
- BoxLayout Rozmístění do podkontejnerů, sdružování komponent
- FlowLayout Rozmístění zleva doprava a shora dolů, nejjednodušší, implicitní
- GridLayout Rozmístění do pevné mřížky

Řídící komponenty

Tlačítka:

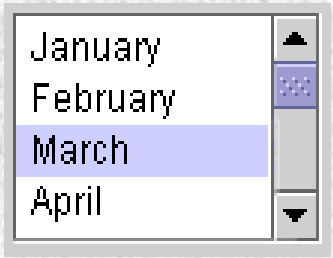
- zvonková **`JButton`**,
- přepínací **`TogleButton`**
- zaškrtávací **`JCheckBox`**
- radio **`JRadioButton`**

- pro spojení do sady se použije **`ButtonGroup`**



`JComboBox`

- seznam, rozbalí se po kliknutí



`JList` – seznam, rozbalený

Řídící komponenty

JTextField vstupní pole pro data
lze nastavit, zda má být editovatelné

JMenuBar, JMenu, JMenuItem

JSlider

Vytvoření jednoduchého okna - Swing

```
package zaklady;
import javax.swing.JButton;
import javax.swing.JFrame;
public class PrvniOkno0 extends JFrame{
  JButton tlacitko = new JButton("Konec");
  public PrvniOkno0(){
     this.getContentPane().add(tlacitko);
  }
  public static void main(String[] args) {
     PrvniOkno0 po = new PrvniOkno0();
     po.pack();
     po.setVisible(true);
} }
```

Swing

Okno je potomkem JFrame

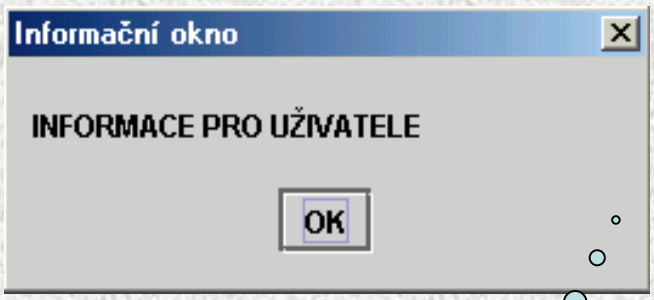
Vložení tlačítka do okna

Zabal okno, vytvoř dostatek místa pro všechny komponenty

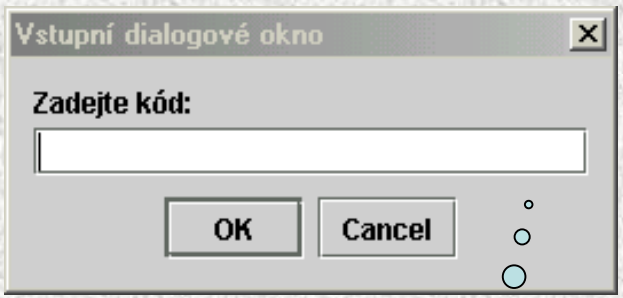
zobraz okno

Nejjednodušší GUI - JOptionPane

```
import javax.swing.*;
class Demo1 {
public static void main(String[] args) {
String str;
JOptionPane.showMessageDialog(null,"INFORMACE PROUŽIVATELE",
            "Informační okno", JOptionPane.PLAIN_MESSAGE);
str = JOptionPane.showInputDialog(null, "Zadejte kód:   ",
    "Vstupní dialogové okno", JOptionPane.QUESTION_MESSAGE);

System.out.println("Kód je:  " + str);
System.exit(0);
  }
}
```



informace



dialog

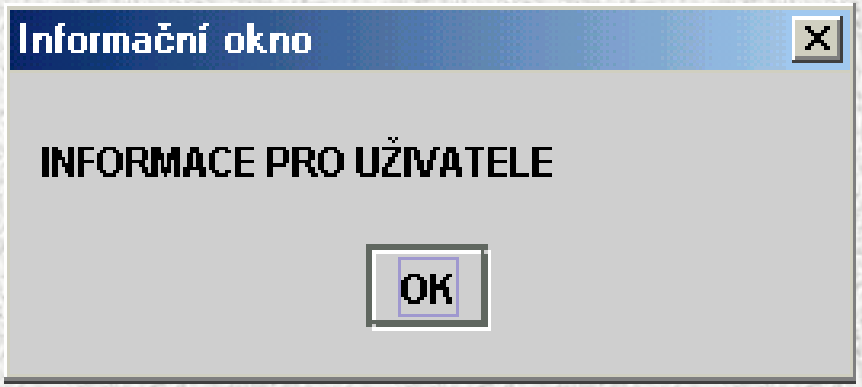
Nejjednodušší GUI – zobrazeni informace

```
JOptionPane.showMessageDialog(null,"INFORMACE PROUŽIVATELE",
                "Informační okno", JOptionPane.PLAIN_MESSAGE);
```

- metoda třídy `JOptionPane`, knihovny `javax.swing`:

```
+--java.awt.Component
        +--java.awt.Container
                +--javax.swing.JComponent
                        +--javax.swing.JOptionPane
```

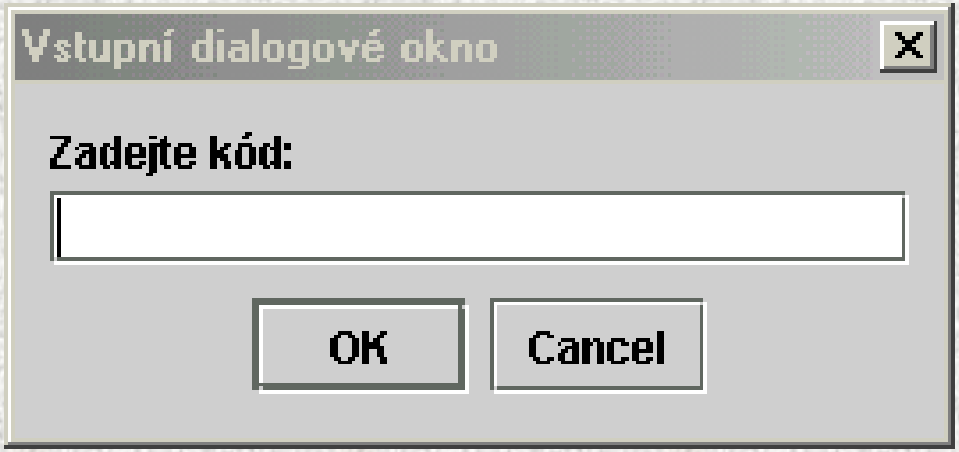
- jiné možné konstanty:
  - `ERROR_MESSAGE`
  - `INFORMATION_MESSAGE`
  - `WARNING_MESSAGE`
  - `QUESTION_MESSAGE`
  - `PLAIN_MESSAGE`



Nejjednodušší GUI– zobrazeni dialogu

```
str = JOptionPane.showInputDialog(null,
  "Zadejte kód:   ",
     "Vstupní dialogové okno",
        JOptionPane.QUESTION_MESSAGE);
```

- hodnota ze vstupního pole je hodnotou funkce
- nekontroluje povolené hodnoty
- kontrolní výstup



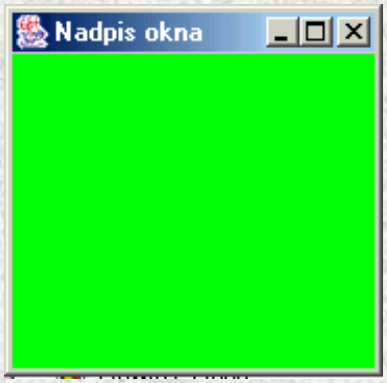
Zobrazení okna

```
class Demo2 {
  public static void main(String[] args) {
    Okno okno = new Okno();
  }
}
class Okno extends JFrame{
public Okno (){
super("Nadpis okna"); // konstruktor JFrame
setSize (300,100);    // nastavení velikosti
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                      // také DO_NOTHING_ON_CLOSE ☺
setVisible(true);} // zobrazení
}
```

Vytvoření kontejneru

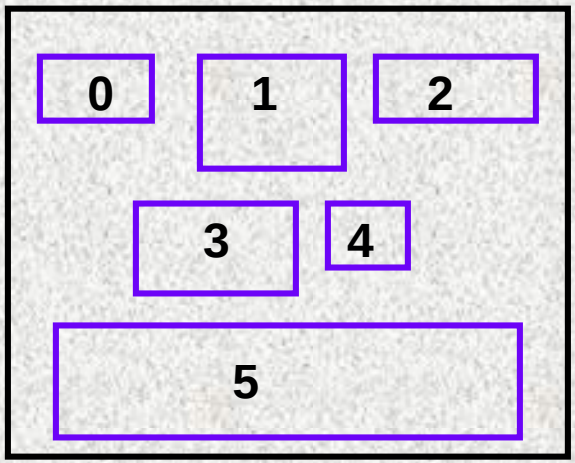
Kontejner

- se vkládá do okna
- obsahuje dialogové prvky
- obsah se zobrazuje v okně

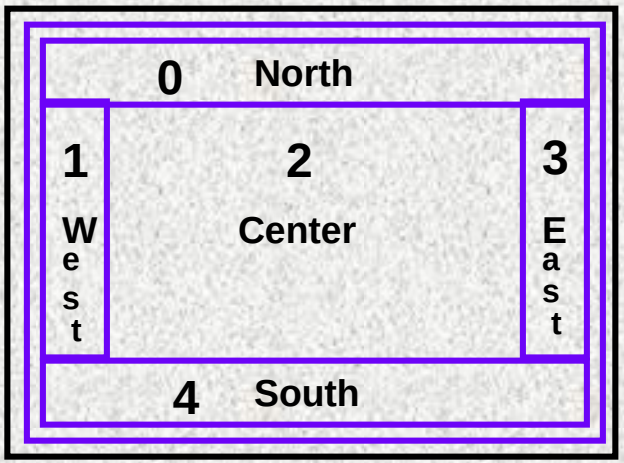



```
import java.awt.*;
import javax.swing.*;

class Okno3 extends JFrame{
public Okno3 (){
super ("Nadpis okna");
setSize (100,100);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setVisible(true);
Container kon = getContentPane(); // vrací kontejner
    kon.setBackground(Color.green);
  }
}
```

Správci rozmístění komponent – Layout manager

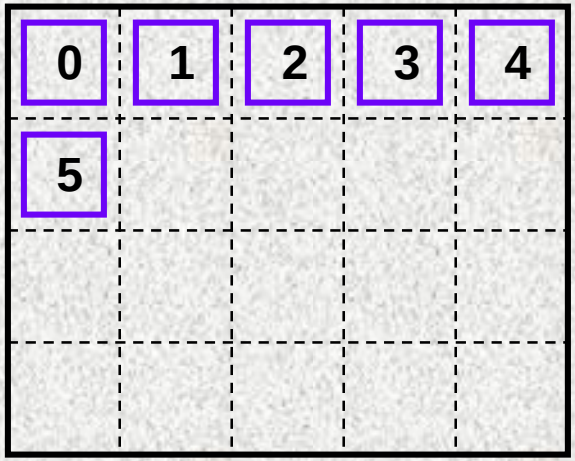


FlowLayout

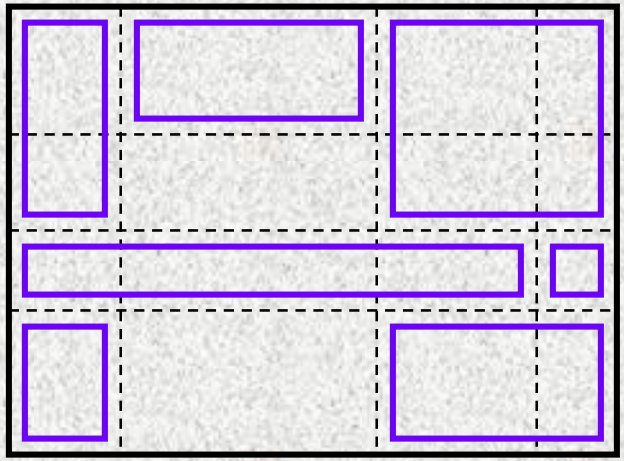


BorderLayout

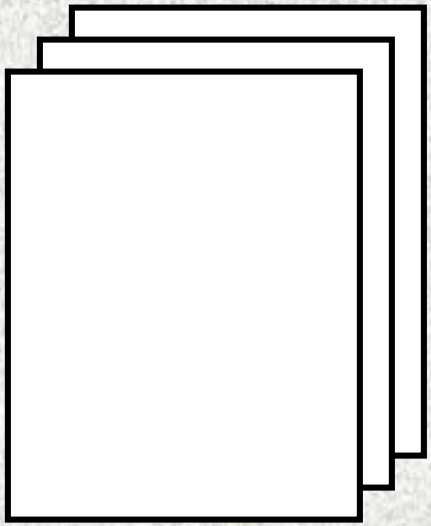
- Pro každý kontejner
- java.awt.*;



GridLayout



GridBagLayout

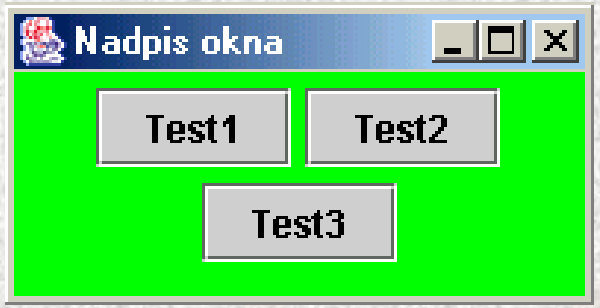


CardLayout

FlowLayout

- Nejjednodušší
  - rozmisťuje zprava doleva a shora dolu a doprostřed

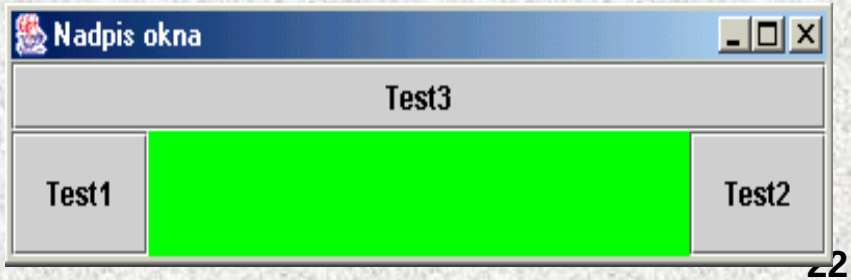
```
class Okno4 extends JFrame{
…
Container kon = getContentPane();
     kon.setBackground(Color.green);
     FlowLayout srb = new FlowLayout();
     kon.setLayout(srb);
     JButton tl1 = new JButton("Test1");
     kon.add(tl1);
     JButton tl2 = new JButton("Test2");
     kon.add(tl2);
     JButton tl3 = new JButton("Test3");
     kon.add(tl3);
     setContentPane(kon);
   }
}
```



BorderLayout

- Rozmisťuje do pěti oblastí podle „světových stran“ **BorderLayout**

```
class Okno4_1 extends JFrame{
public Okno4_1 (){
…
Container kon = getContentPane();
     kon.setBackground(Color.green);
     BorderLayout srb = new BorderLayout();
     kon.setLayout(srb);
     JButton tl1 = new JButton("Test1");
     kon.add(tl1,srb.WEST);
     JButton tl2 = new JButton("Test2");
     kon.add(tl2,srb.EAST);
     JButton tl3 = new JButton("Test3");
     kon.add(tl3,srb.NORTH);
     setContentPane(kon);
   }
}
```



GridLayout

- Rozložení v pravidelné mřížce,
  - rafinovanější je **`GridBagLayout`**, mřížku možno určit

```
class Okno5 extends JFrame{
[…].
Container kon = getContentPane();
     kon.setBackground(Color.green);
     GridLayout srg = new GridLayout(3,3);
     kon.setLayout(srg);
     JButton tl1 = new JButton("Test1");
     kon.add(tl1);
     JButton tl2 = new JButton("Test2");
     kon.add(tl2);
     JButton tl3 = new JButton("Test3");
     kon.add(tl3);
     JButton tl4 = new JButton("Test4");
     kon.add(tl4);
     JButton tl5 = new JButton("Test5");
     kon.add(tl5);
     setContentPane(kon);
   }
}
```

Komponenty

- Funkčnost je řízena událostmi
- Speciální lekce

Tlačítka

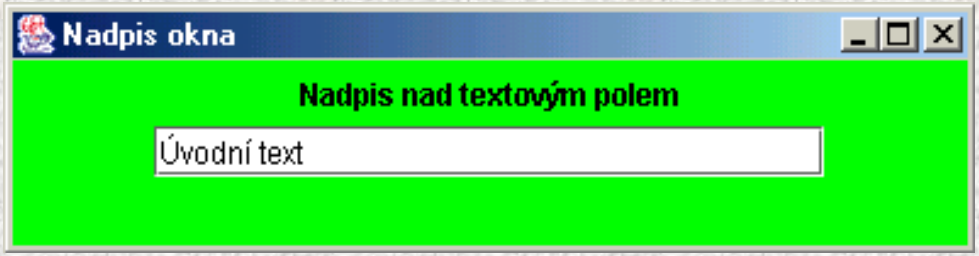
- Komunikační komponenty jsou tlačítka - **`JButton`** – už jsme poznali

```
class Okno6 extends JFrame{
…
setVisible(true);

Container kon = getContentPane();
kon.setBackground(Color.green);
FlowLayout srf = new FlowLayout();
kon.setLayout(srf);
 JButton tl1 = new JButton("Start");
    kon.add(tl1);
    JButton tl2 = new JButton("Stop");
    kon.add(tl2);
    setContentPane(kon);
  }
}
```

Textová pole, popisky

- Textová pole (aktivní) - **`JTextField`** a popisky (pasivní) – **`JLabel`**

```
class Okno7 extends JFrame{
...
setVisible(true);
```

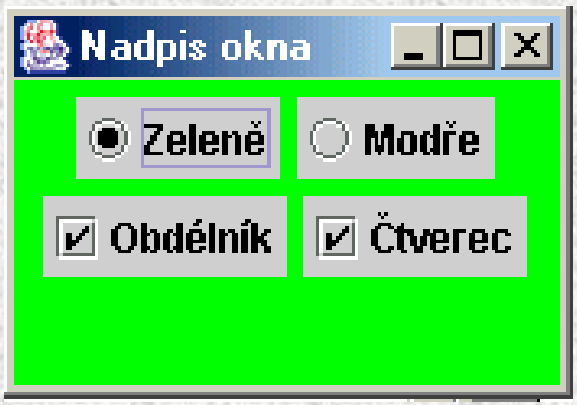



```
Container kon = getContentPane();
kon.setBackground(Color.green);
FlowLayout srf = new FlowLayout();
kon.setLayout(srf);
JLabel popisek = new JLabel("Nadpis nad textovým polem");
kon.add(popisek);
JTextField text = new JTextField("Úvodní text", 25);
kon.add(text);
setContentPane(kon);
  }
}
```

Přepínače (radio) a zaškrtávací tlačítka

- Přepínač umožní výběr jedné možnosti z více - **`JRadioButton`**
- Zaškrtávací tlačítka umožní zadávání dvouhodnotových parametrů **`JCheckBox`**

```
class Okno8 extends JFrame{
[…]
kon.setLayout(srf);
JCheckBox zt1 = new JCheckBox ("Obdélník");
JCheckBox zt2 = new JCheckBox ("Čtverec");
ButtonGroup vyhovelNevyhovel = new ButtonGroup();
JRadioButton rt1 = new JRadioButton ("Zeleně");
JRadioButton rt2 = new JRadioButton ("Modře");
vyhovelNevyhovel.add(rt1);
vyhovelNevyhovel.add(rt2);
kon.add(rt1);
kon.add(rt2);
kon.add(zt1);
kon.add(zt2);
setContentPane(kon);
  }
}
```



Seznamy

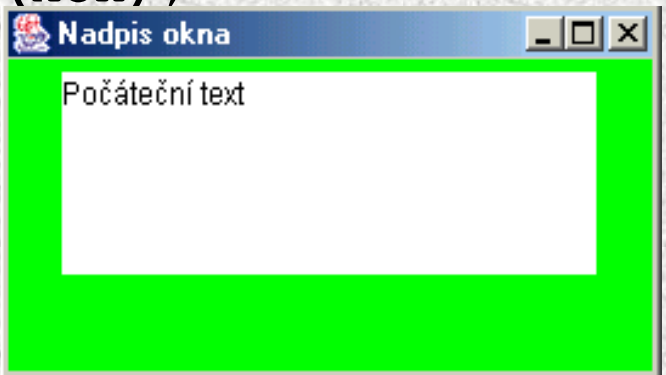
- Výběr z více možností - **JComboBox**

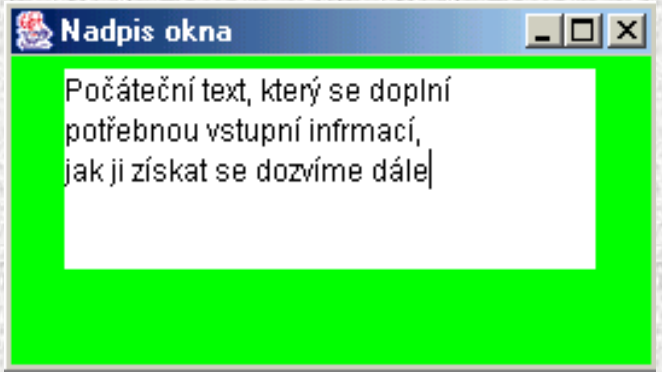
```
class Okno9 extends JFrame{
…
Container kon = getContentPane();
    kon.setBackground(Color.green);
    FlowLayout srf = new FlowLayout();
    kon.setLayout(srf);
    JComboBox rseznam1 = new JComboBox();
    rseznam1.addItem("První");
    rseznam1.addItem("Druhý");
    rseznam1.addItem("Třetí");
    kon.add(rseznam1);
    setContentPane(kon);
  }
}
```

Textová oblast + víceřádková

- Slouží ke vstupu textu s počáteční nápovědou – **`JTextArea`**
- pro větší texty slouží **`JSrollPane,`** který umožní rolování textu

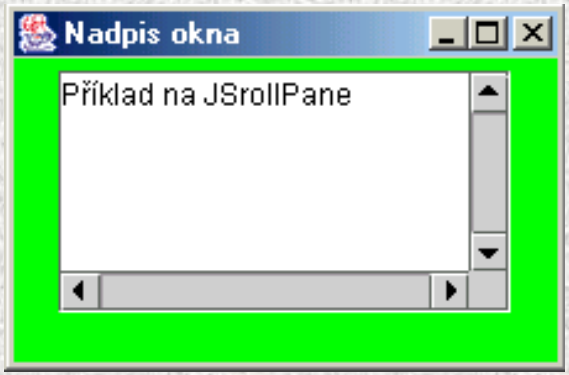
```
class Okno10 extends JFrame{
…
 Container kon = getContentPane();
 kon.setBackground(Color.green);
 FlowLayout srf = new FlowLayout();
 kon.setLayout(srf);
 JTextArea to = new JTextArea("Počáteční text", 5, 20);
 kon.add(to);
 setContentPane(kon);
  }

}
```



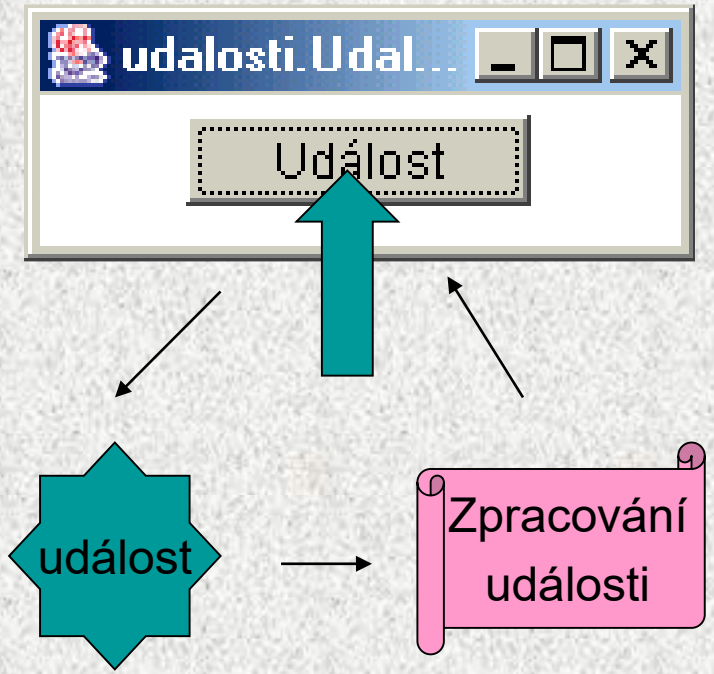


Textová oblast + víceřádková

```
class Okno11 extends JFrame{
…

Container kon = getContentPane();
kon.setBackground(Color.green);
FlowLayout srf = new FlowLayout();
kon.setLayout(srf);
JTextArea to = new JTextArea("Příklad na JSrollPane", 5, 15);

JScrollPane rp = new JScrollPane (to,
            JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
                 JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);

kon.add(rp)
setContentPane(kon);
}
```



Zpracování událostí



Obsluha událostí

Mechanismus reakce na akci uživatele

- stisk tlačítka, zadání textu, stisk tlačítka myši, …

1. Pro každou komponentu je třeba:
   1. deklarovat typ zachycované události, kterou je zájem zpracovat
   2. určit „posluchače“, který má událost obsloužit
2. Akcí uživatele vznikne událost
   - událost je objektem Javy!
3. Události jsou zachyceny
   - události jsou zpracovány (obslouženy) **„posluchači“** (listener)
     - třídami s **uživatelskými metodami pro reakci na událost**
   - **„posluchači“** jsou třídy, které implementují rozhraní (**`interface`**) naslouchání
     - musejí mít schopnost „naslouchání“

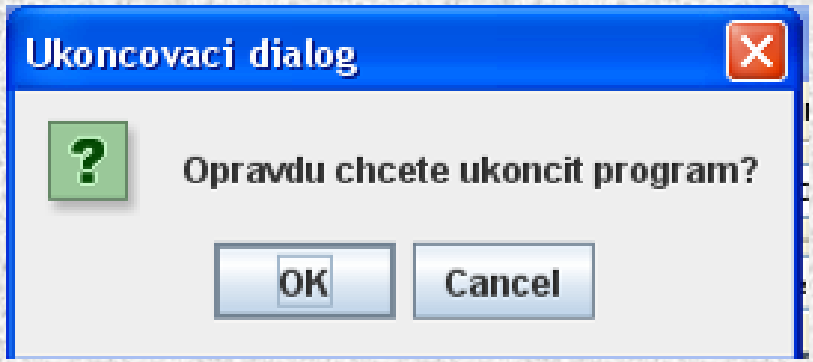
Pozn.: O obsluze událostí bude speciální přednáška

Jednoduché zpracování události od tlačítka

```
public class PrvniOkno extends JFrame implements
                                              ActionListener{
   JButton tlacitko = new JButton("Konec");
   public PrvniOkno(){
       this.getContentPane().add(tlacitko);
       tlacitko.addActionListener(this);
   }
   public static void main(String[] args) {
       PrvniOkno po = new PrvniOkno();
       po.pack();
       po.setVisible(true);
   }
   public void actionPerformed(ActionEvent e) {
//       obsluha události, viz další slide
 }
```

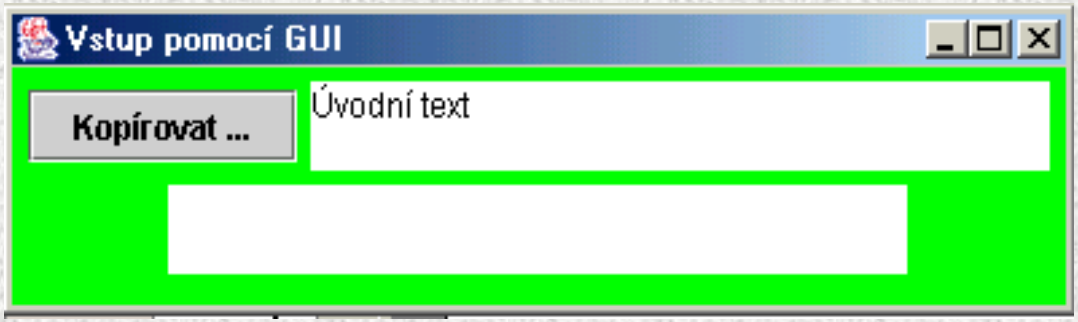
Obsluha dialogu (podrobně v následující přdnášce)

```
public void actionPerformed(ActionEvent e) {
switch (JOptionPane.showConfirmDialog(this,
 "Opravdu chcete ukoncit program?", "Ukoncovaci dialog",
                                JOptionPane.WARNING_MESSAGE)){
case JOptionPane.OK_OPTION: //ukonci program
   System.exit(0);break;
case JOptionPane.CANCEL_OPTION: //rozmyslel si to,nedelej nic.
   }
}
```

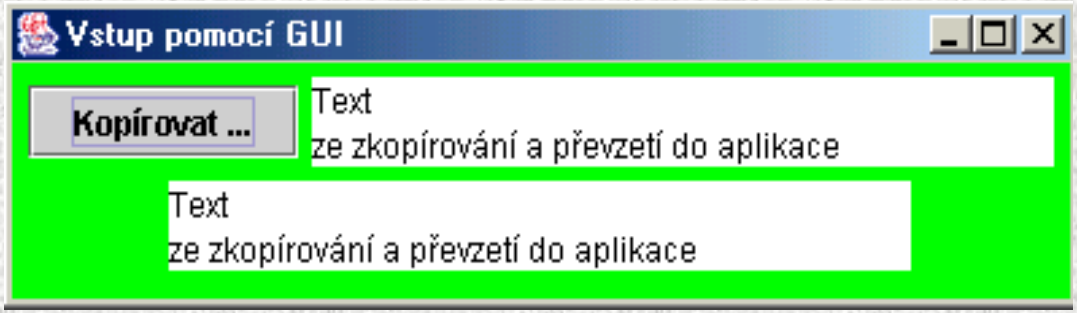


Obsluha události Kopírování textu

- **Počáteční stav**



- **Stav po stisku tlačítka**



Obsluha události Kopírování textu

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.lang.*;
class Demo121 {
  public static void main(String[] args) {
    Okno121 okno = new Okno121();
}
}
class Okno121 extends JFrame implements ActionListener{
  JTextArea to1 = new JTextArea ("Úvodní text", 2, 25);
  JTextArea to2 = new JTextArea (2, 25);
  JButton tl1 = new JButton ("Kopírovat ...");
```

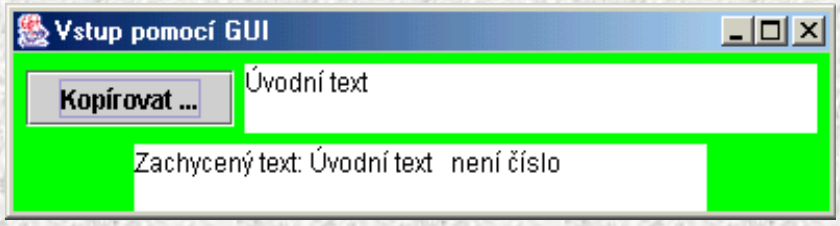
Obsluha události Kopírování textu

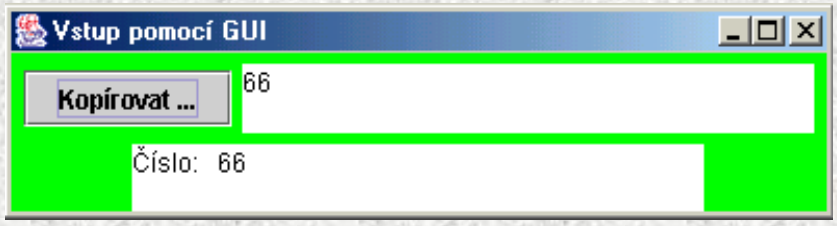
```
public Okno121(){
    super ("Vstup pomocí GUI");
    setSize(400, 100);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setVisible(true);
    Container kon = getContentPane();
    kon.setBackground(Color.green);
    FlowLayout srf = new FlowLayout();
    kon.setLayout(srf);
    tl1.addActionListener(this);
    kon.add(tl1);
    kon.add(to1);
    kon.add(to2);
    setContentPane(kon);
      }
 public void actionPerformed (ActionEvent event){
 String t = to1.getText();
 to2.setText(t);
  }
}
```

Obsluha události Kopírování čísla

- U vstupních dat je třeba testování – např. pomocí výjimek, viz speciální přednáška

```
public void actionPerformed (ActionEvent event){
String t = to1.getText();
 try {
 int i = Integer.valueOf(t).intValue();
  to2.setText("Číslo:   " + i);
}
  catch (NumberFormatException e){
 to2.setText("Zachycený text:"+" " + t + " "+ "  není
  číslo!");
}
}
```





Grafika v Javě

- Základní třída **`java.awt.Graphics, java.awt.Graphics2D (JDK1.2)`**
- Základní možnosti třídy **`Graphics`**:
  - kreslení základní 2D objektů, grafických primitiv
  - vykreslování textu a obrázků
  - nastavování a testování barev, fontu, ořezání, ploch
- Okamžik zobrazení není časově determinován
- Kreslit lze v komponentách **`JPanel,JFrame (JApplet),`**
- Vykreslování probíhá v tzv. „grafickém kontextu“, tvořeného třídou **`Graphics`**
  - Grafický kontext je parametrem zděděné metody **`Container.paint(Graphics g)`**, kde je popsáno vlastní kreslení
    - parametr **`g`** je abstraktní, formální
    - "automatický" objekt, o který se nestaráme, třída **`Graphics`** je abstraktní, nelze vytvořit její objekt
    - definuje počáteční vykreslení, nevolá se přímo!
  - Překreslování **`repaint, update`**

Grafika v Javě

java.lang.Object

java.awt.Component

java.awt.Container (`paint`)

java.awt.Window

java.awt.Frame

javax.swing.JFrame

- Obsah **`paint()`** je poprvé vykreslen někdy po dokončení generování objektu Trida
- Je vždy někdy překreslen pro ikonizaci/a deiokonizaci, či posunu okna apod

Trida extends JFrame
Trida()
void paint(Graphics g)

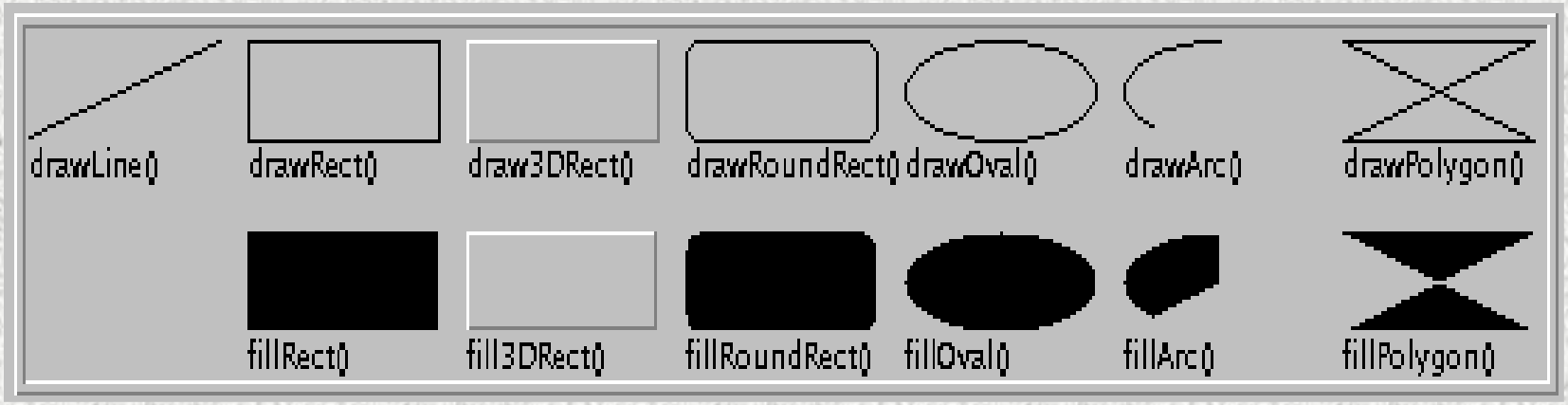
Grafická primitiva

Grafická primitiva:

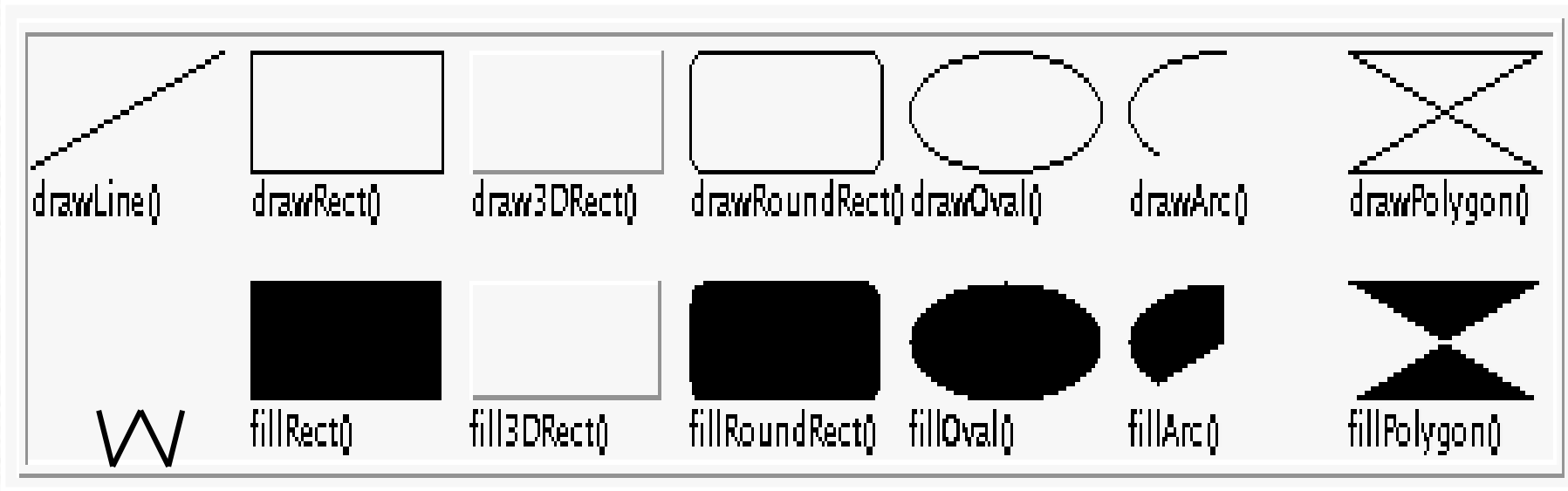
- kreslení tvaru, obrysu – **`drawXXX()`**
- vyplnění tvaru – **`fillXXX()`**

`XXX:`

- **`Line (jen draw)`**
- **`Rect, 3Drect, RoundRect`**
- **`Oval`**
- **`Arc`**
- **`Polyline`**



Graphics

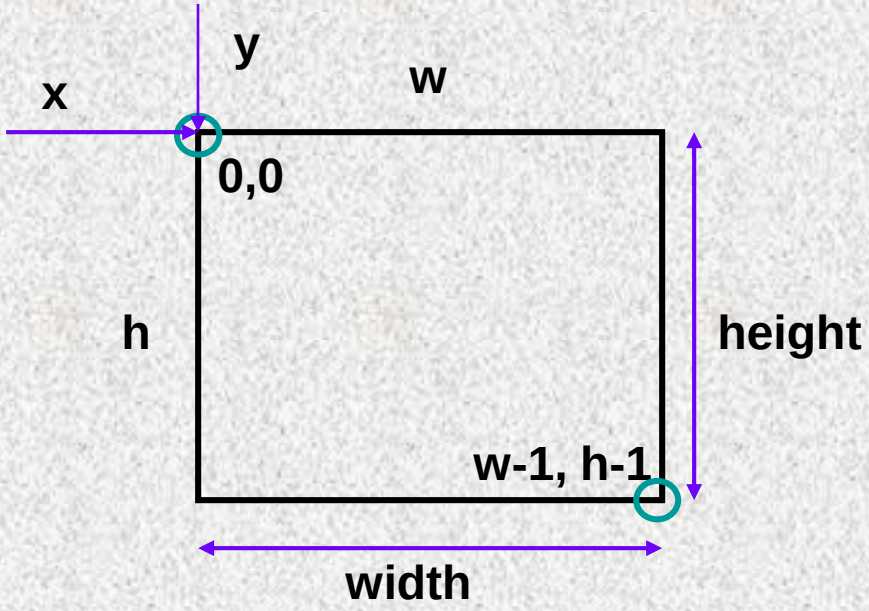


Další metody:

- **`clearRect( … )`** - přemalování na barvu pozadí
- **`clipRect()`,`getClip()`,`setClip()`**- vystřihovánky a nalepovánky
- **`copyArea( … )`** - kopírování plošky
- **`setFont( … ), getFont( )`** - práce s fonty
- **`getFontMetrics( )`** - měření nápisů

Metrika

Vizuální komponenty a displej se rozměřují v pixelech takto:



Grafika v Javě, příklad


```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
public class GrafikaSimple extends JFrame {
  public static void main(String[] args) {
    JFrame f = new GrafikaSimple();
  }
GrafikaSimple() {
  super ("Jednoduchá grafika");
  setSize(300, 200);
  setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  setVisible(true);
  }
public void paint(Graphics g) {
    g.drawString("Hello, World",  175, 100);
    g.drawOval(70,100,140,70);
  }
}
```

Umí nakreslit okno

Vykresli okno

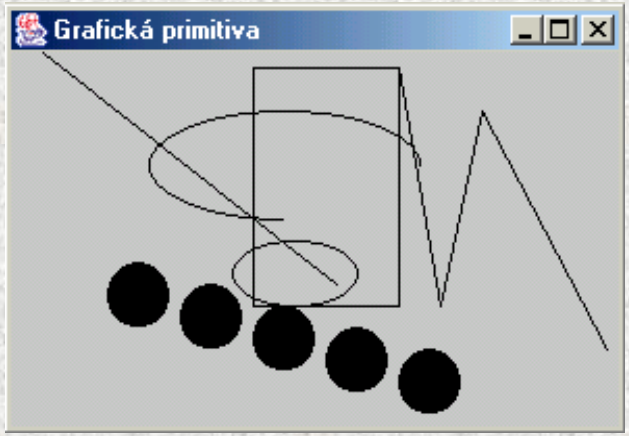
Vykresli okno

Obsahuje obsah okna

Vykreslí se kdy bude chtít

Příklad na grafiku

```
PrimitivaLine() {
super ("Jednoduchá grafika");
setSize(300, 200);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setVisible(true);
}
public void paint(Graphics g) {
g.drawLine(15, 20, 160, 130);
g.drawRect(120,30, 70,110);
g.drawOval(110,110,60,30);
g.drawArc(70,50,130,50,0,270);
  int[] xp = {190, 210, 230, 290};
  int[] yp = {30, 140, 50, 160};
g.drawPolyline(xp,yp,4);
 for (int i = 0; i<5 ; i++) {
g.fillArc(50+i*35,120+i*10,30,30,0,360);
}
```

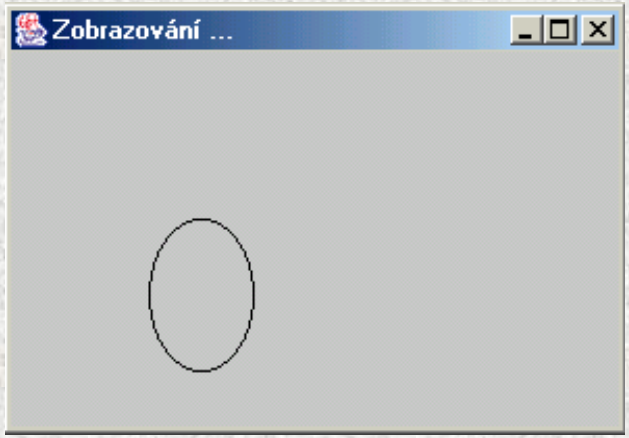


Vložení obrázku

```
public class Obr extends JFrame {
……..
public void paint(Graphics g) {
g.fillRect(0, 0, 600, 600);
g.setColor(new Color(200, 200, 200));
g.setColor(Color.YELLOW);
g.drawString("Hello, World", 220, 100);
Toolkit t = Toolkit.getDefaultToolkit();
Image img = t.getImage("kocka.jpg");
g.drawImage(img, 100, 100, this);
}
```

Zobrazování, `paint, repaint, update`

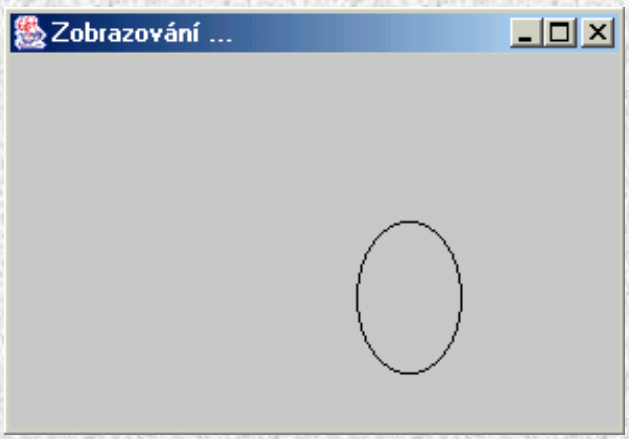
```
public class GrafikaSimplePokus extends JFrame {
  int x= 70; int y = 100; int w = 50; int h = 70;static int
   r=0;
  public static void main(String[] args) {
   GrafikaSimplePokus fr=new GrafikaSimplePokus();
   try { Thread.sleep(2000);}
   catch (InterruptedException ex) {}
   fr.x+=100;
   fr.repaint();
 }
  public GrafikaSimplePokus() {
    super ("Zobrazování ...");
    setSize(300, 200);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setVisible(true);
    }
```



Pro zájemce

Zobrazování, `paint, repaint, update`

```
//public void update(Graphics g){
 // paint(g); }

  public void paint(Graphics g) {
// if (r>0)g.clearRect(0, 0,300, 200);
  g.drawOval(x,y,w,h);
    System.out.println("paint" + r++);
  }}
```



Pro zájemce

Zobrazování, `paint, repaint, update`

- Metoda `paint` je volána automaticky při “změně” okna a vykreslí grafický kontext (Graphics) definovany obsahem `paint`
- Metoda `repaint`
  - volá `paint`
  - „přikreslí” přes stávající okno aktuální grafický kontext
  - volá `update`
- Je-li metoda `update` přetížena, pak smaže celé okno při manipulaci s oknem a volá `paint`

Pro zájemce