

# GUI v Javě

Jiří Vokřínek

Katedra počítačů

Fakulta elektrotechnická

České vysoké učení technické v Praze

Přednáška 5

**B0B36PJV – Programování v JAVA**

# GUI v Javě

GUI v Javě

GUI komponenty a kontejnery

Dialogová okna

Události a obsluha událostí z GUI

# Část I

## GUI v Javě

# Grafické uživatelské rozhraní

- GUI – Graphical User Interface
- Zásadním způsobem ovlivňuje použitelnost, přívětivost aplikace a také produktivitu *User experience*
- Elegantní návrh s intuitivní a **konzistentní** funkcionalitou
- Respektujte styl a **zvyklosti** uživatele *Cílová skupina laik vs expert*
- Jednoduchost bývá zpravidla lepší než složité komponenty  
*Vytvořit jednoduché a dobře použitelné rozhraní je zpravidla výrazně časově náročnější než se na první pohled zdá.*
- Klíčová je zpětná vazba od uživatelů a testování *„Testováno na lidech!“*
- Návrh **dobrého** rozhraní je o rozložení grafických prvků, volbě barev a tvarů, vizualizačních efektech, písmu, . . .  
*V rámci PJV si ukážeme jaké základní prvky (objekty) se pro tvorbu GUI používají.*

# Grafické uživatelské rozhraní

- GUI – Graphical User Interface
- Zásadním způsobem ovlivňuje použitelnost, přívětivost aplikace a také produktivitu *User experience*
- Elegantní návrh s intuitivní a **konzistentní** funkcionalitou
- Respektujte styl a **zvyklosti** uživatele *Cílová skupina laik vs expert*
- Jednoduchost bývá zpravidla lepší než složité komponenty  
*Vytvořit jednoduché a dobře použitelné rozhraní je zpravidla výrazně časově náročnější než se na první pohled zdá.*
- Klíčová je zpětná vazba od uživatelů a testování *„Testováno na lidech!“*
- Návrh **dobrého** rozhraní je o rozložení grafických prvků, volbě barev a tvarů, vizualizačních efektech, písmu, ...

*V rámci PJV si ukážeme jaké základní prvky (objekty) se pro tvorbu GUI používají.*

## Grafické uživatelské rozhraní

- GUI – Graphical User Interface
- Zásadním způsobem ovlivňuje použitelnost, přívětivost aplikace a také produktivitu *User experience*
- Elegantní návrh s intuitivní a **konzistentní** funkcionalitou
- Respektujte styl a **zvyklosti** uživatele *Cílová skupina laik vs expert*
- Jednoduchost bývá zpravidla lepší než složité komponenty  
*Vytvořit jednoduché a dobře použitelné rozhraní je zpravidla výrazně časově náročnější než se na první pohled zdá.*
- Klíčová je zpětná vazba od uživatelů a testování *„Testováno na lidech!“*
- Návrh **dobrého** rozhraní je o rozložení grafických prvků, volbě barev a tvarů, vizualizačních efektech, písmu, ...  
*V rámci PJV si ukážeme jaké základní prvky (objekty) se pro tvorbu GUI používají.*

## Programování a tvorba grafického rozhraní

- Z programátorského hlediska se však vždy v podstatě jedná o zadání vstupu a prezentaci výstupu
- Pro interakci s uživatelem lze využít sadu základních grafických komponent tzv. **Widgets**
- Softwarová knihovna pro tvorbu rozhraní se nazývá **Widget toolkit** nebo grafický **toolkit**
- Klíčem k jednoduchosti, použitelnosti a také přenositelnosti mezi platformami je unifikace grafických prvků

*Různé systémy, různé grafické reprezentace.*

- Velkou výhodou Javy je, že knihovny pro grafické prvky jsou součástí standardního JDK

## Grafické knihovny v Javě

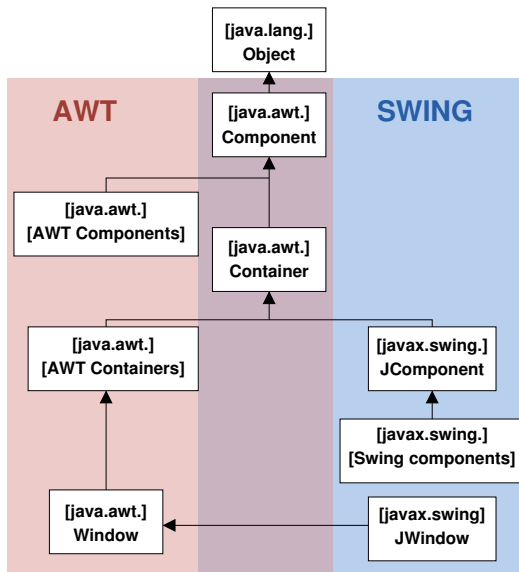
- AWT – Abstract Window ToolKit (První gui v Javě – *heavyweight*)
  - Vykreslování zajišťuje hostitelská platforma, na které běží JVM  
*Vykreslování je tak rychlejší, ale vše nemusí fungovat identicky na jiných platformách*
- Swing – Výrazné rozšíření (a zlepšení) GUI (oproti AWT)
  - Doporučené standardní GUI v Javě
  - *Look&Feel* je platformově nezávislý a respektuje **i18n**  
*i18n – i-internationalizatio-n*
  - Důsledné oddělení modelu od pohledu  
<http://docs.oracle.com/javase/tutorial/uiswing>
- JavaFX – nový GUI ToolKit (následovník Swing)
  - Styl vzhledu přes CSS  
*HMTL rendering engine (WebKit)*  
<http://docs.oracle.com/javase/8/javase-clienttechnologies.htm>
- SWT – Standard Widget Toolkit (Eclipse)
  - Platformově závislý, ale unifikuje vzhled  
<https://www.eclipse.org/swt>



# Základní prvky grafického rozhraní

- **Komponenty** – tlačítka, textová pole, menu, posuvníky, . . .
- **Kontejnery** – komponenty, do kterých lze vkládat komponenty  
*Například pro rozdělení plochy a volbu rozmístění*
- **Správce rozvržení** (*Layout manager*) – rozmísťuje komponenty v ploše kontejneru
- Interakce s uživatelem dále zpravidla vyžaduje mechanismus událostí a jejich zachytávání

# AWT a Swing



# Základní součásti GUI

## ■ **Komponenty** a dialogové prvky

[javax.swing](#)

- Tlačítka, text, textová pole, seznamy, přepínače
- Společné metody pro velikost, barvu, umístění textu, ...

## ■ **Kontejnery** (v oknech, která zpravidla řeší prostředí OS)

[javax.swing](#)

- Kontejnery obsahují komponenty

*Komponenty musí být umístěny v kontejneru*

- Kontejnery se vkládají do oken
- **JFrame** – obecný kontejner
- **JPanel** – kontejner pro jednoduché komponenty

## ■ **Layout Manager** – Správce rozmístění

[javax.swing](#) a [java.awt](#)

- Definuje pozici komponent v kontejneru
- Relativní k okrajům, pevná pozice, v mřížce, ...
- Určuje vzhled a chování aplikace

## ■ **Events** – Obsluha událostí ([java.awt.event](#))

# Přehled základních grafických komponent

## Komponenty

- `JLabel` – Zobrazení popisku, bez generování události
- `JButton` – Tlačítko s událostí kliknutí na tlačítko
- `JTextField` – Zadání textu
- `JPasswordField` – Zadání textu (hesla), vložené znaky se zobrazují jako hvězdičky
- `JList` – Seznam položek, možnost vybrat jednu nebo více položek
- `JComboBox` – Rozevírací seznam položek, klepnutím na položku se generuje událost
- `JCheckBox` – Zaškrtačací políčko, prvek je/není vybrán
- `JRadioButton` – Přepínač, výběr z možností

# Kontejnery a správce rozvržení

## ■ **JFrame** – Kontejner s ohraničením a záhlavím

<http://docs.oracle.com/javase/tutorial/uiswing/components/frame.html>

## ■ **JPanel** – Kontejner bez ohraničení, implicitně rozmístění **FlowLayout**

*Může být jednodušší na použití*

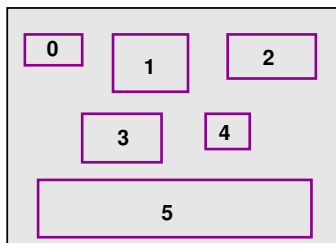
<http://docs.oracle.com/javase/tutorial/uiswing/components/panel.html>

## ■ **Layout Manager** (správce rozvržení)

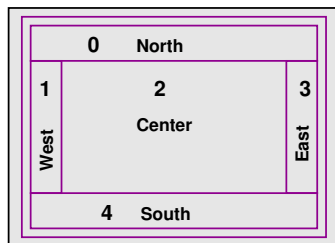
- **BorderLayout** – Rozmístění podle okrajů okna (panelu/kontejneru)
- **BoxLayout** – Rozmístění do podkontejnerů, sdružování komponent
- **FlowLayout** – Rozmístění zleva doprava a shora dolů
- **GridLayout** – Rozmístění do pevné mřížky

<http://docs.oracle.com/javase/tutorial/uiswing/layout/visual.html>

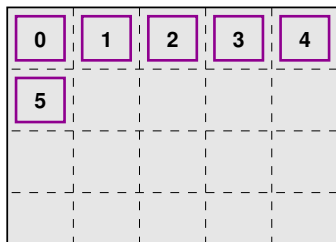
# Správci rozmístění komponent – Layout Manager



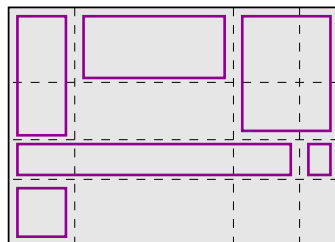
FlowLayout



BorderLayout

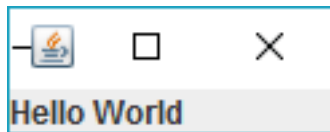


GridLayout



GridBagLayout

## Příklad okna a vložení komponenty (JLabel)



### ■ JFrame Swing – „Hello World”

```
// okno a jeho titulek
JFrame frame = new JFrame("HelloWorldSwing");

frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

JLabel label = new JLabel("Hello World");
frame.getContentPane().add(label, BorderLayout.NORTH);

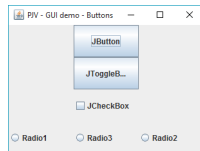
frame.pack(); //nastav velikost okna
frame.setVisible(true); //zobrazí okno
```

Metoda `demo` v `DemoGuiComponents`

# Řídicí komponenty 1/2

## ■ Tlačítka

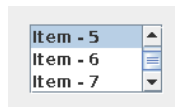
- `JButton` – „zvonková“
- `JToggleButton` – přepínací
- `JCheckBox` – zaškrtačiví
- `JRadioButton` a `ButtonGroup`



<http://docs.oracle.com/javase/tutorial/uiswing/components/button.html>

## ■ `JList` – seznam

`SINGLE_SELECTION`,  
`SINGLE_INTERVAL_SELECTION`,  
`MULTIPLE_INTERVAL_SELECTION`



<http://docs.oracle.com/javase/tutorial/uiswing/components/list.html>

## ■ `JComboBox` – seznam rozbalovací



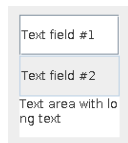
<http://docs.oracle.com/javase/tutorial/uiswing/components/combobox.html>

DemoGuiComponents



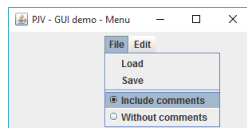
## Řídicí komponenty 2/2

- `JTextField` – vstupní pole pro data (editovatelné nebo needitovatelné)
  - `TextArea`



<http://docs.oracle.com/javase/tutorial/uiswing/components/textfield.html>

- `JMenuBar`, `JMenu`, `JMenuItem`
  - `JRadioButtonMenuItem`,  
`ButtonGroup`



<http://docs.oracle.com/javase/tutorial/uiswing/components/menu.html>

- `JSlider`



<http://docs.oracle.com/javase/tutorial/uiswing/components/slider.html>

DemoGuiComponents

## Vlastní grafika v Javě – Plátno (Canvas)

- Základní třídy `java.awt.Graphics`, `java.awt.Graphics2D`
- Základní možnosti třídy `Graphics`: *Již od JDK ver. 1.2*
  - Kreslení základních 2D objektů (grafických primitiv)
  - Vykreslování textu a obrázků
  - Nastavování a testování barev, fontů, ořezání, ...
- Okamžik zobrazení „není“ časově určen
- Kreslit lze v komponentách `JPanel` a `JFrame`
- Vykreslování probíhá v grafickém kontextu tvořeného třídou `Graphics`
  - Grafický kontext je parametrem (zděděné) metody `Container.paint(Graphics g)`, ve které probíhá vlastní kreslení do kontextu („plátna“)
  - Definuje počáteční vykreslení, nevolá se přímo  
*Třída `Graphics` je abstraktní, předávaný objekt `g` je „automatický“ objekt, o který se nestaráme.*
- Překreslování je realizováno metodami `repaint` a `update`  
<http://docs.oracle.com/javase/tutorial/uiswing/painting>

## Příklad vykreslení grafických primitiv 1/3

```
public class Canvas extends JFrame {  
    public Canvas() {  
        setTitle("PJV Demo Canvas");  
        setSize(640, 480);  
        setVisible(true);  
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    }  
  
    @Override  
    public void paint(Graphics g) { ... }  
}
```

Canvas

## Příklad vykreslení grafických primitiv 2/3

```
@Override
public void paint(Graphics g) {
    Graphics2D g2d = (Graphics2D) g;
    g2d.setColor(Color.RED);
    g2d.fillOval(110, 210, 30, 30);
    g2d.drawOval(360, 320, 30, 30);

    g2d.setColor(Color.BLUE);
    g2d.fillRect(150, 50, 30, 30);

    g2d.drawPolygon(
        new int[]{200, 250, 300, 290, 180},
        new int[]{150, 200, 180, 210, 240},
        5);
    g2d.draw(new Ellipse2D.Double(320, 240, 30, 30));
}
```

Canvas

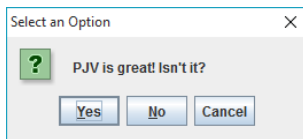
## Příklad vykreslení grafických primitiv 3/3

```
public class DemoGuiCanvas {  
  
    public void demo() {  
        Canvas canvas = new Canvas();  
    }  
  
    public static void main(String[] args) {  
        DemoGuiCanvas gui = new DemoGuiCanvas();  
        gui.demo();  
    }  
  
}
```

DemoGuiCanvas

## Dialogové okno

- Dialogové okno je dočasné „nezávislé“ okno zpravidla vyžadující interakci uživatele
- Slouží pro informování uživatele nebo pro získání uživatelské vstupu



<http://docs.oracle.com/javase/tutorial/uiswing/components/dialog.html>

- Je možné je vyvolat metodami třídy **JOptionPane**, např.
  - `showMessageDialog`
  - `showConfirmDialog`
  - `showInputDialog`

## Příklad dialogového okna

```
import javax.swing.JOptionPane;  
  
JOptionPane.showMessageDialog(null, "Message");  
  
int answer = //0 - Yes, 1 - No, 2 - Cancel  
            JOptionPane.showConfirmDialog(null, "Config?");  
  
String str =  
            JOptionPane.showInputDialog(null, "Entry");  
                                                DemoDialog
```

## Dialog zobrazení informace

- Zobrazení informace můžeme anotovat podle významu
  - `ERROR_MESSAGE`
  - `INFORMATION_MESSAGE`
  - `WARNING_MESSAGE`
  - `QUESTION_MESSAGE`
  - `PLAIN_MESSAGE`

```
int response = JOptionPane.showConfirmDialog(null, "PJV is great
! Isn't it?");
switch (response) {
    case 0:
        JOptionPane.showMessageDialog(null, "You are right!",
            "Confirm", JOptionPane.PLAIN_MESSAGE);
        break;
    case 1:
        JOptionPane.showMessageDialog(null, "You are wrong!",
            "Error", JOptionPane.ERROR_MESSAGE);
        break;
    case 2:
        JOptionPane.showMessageDialog(null, "You should know!",
            "Warn", JOptionPane.WARNING_MESSAGE);
        break;
}
```

DemoDialog



## Modalita dialogových oken

- Modalita dialogu určuje, zdali dialogové okno blokuje ostatní okna
- **Dialog.ModalityType**
  - APPLICATION\_MODAL
  - DOCUMENT\_MODAL
  - MODELESS
  - TOOLKIT\_MODAL
- Volíme dle typu aplikace a dialogu např.:
  - Jedno hlavní okno, ostatní dialogová okna slouží pro zadání vstupu nebo informování uživatele (např. výběr souboru), po uzavření přecházíme do hlavního okna
  - Více „hlavních“ oken, kterými procházíme a vždy pracujeme pouze s jedním oknem
  - Více „plovoucích“ nezávislých oken

*Modeless*

<http://docs.oracle.com/javase/tutorial/uiswing/misc/modality.html>

## Příklad modálního a nemoďálního okna

```
final JFrame parent = new JFrame("Parent Frame");
parent.setLayout(new FlowLayout());
parent.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
parent.setBounds(100, 100, 300, 200);
parent.add(new JButton("Button"));

parent.setVisible(true);

JDialog dialog1 = new JDialog(parent, "Modeless Dialog");
dialog1.setBounds(200, 200, 300, 200);
dialog1.setVisible(true);

JDialog dialog2 = new JDialog(parent,
    "Document-Modal Dialog",
    Dialog.ModalityType.DOCUMENT_MODAL);
dialog2.setBounds(300, 300, 300, 200);
dialog2.setVisible(true);
```

DemoModality

## Zpracování událostí

- Interakce uživatele s rozhraním vyvolává události, na které je potřeba reagovat
- Dialogová okna (modální) představují synchronní mechanismus, kdy je běh aplikace „pozastaven“ a aplikace čeká na uživatelský vstup
- Zpravidla, chceme uživatelům umožnit vyšší interaktivitu a s tím související „nezávislé“ generování událostí
  
- Generované události je však nutné zpracovávat

# Obsluha událostí

- Mechanismus reakce na akci uživatele, např.
  - Stisk tlačítka, zadání textu, pohyb kurzoru

*Množinu možných typů událostí definuje Toolkit a souvisí s rozhraním (uživatelským) počítače*
- Pro každou komponentu je nutné
  1. Deklarovat typ zachytávané událost, kterou chceme zpracovávat
  2. Určit „posluchače“, který má událost obsloužit
- Akcí uživatele vznikne **událost**, která je objektem Javy
- Zachycené události
  - jsou zpracovány (obslouženy) „posluchači“ (**listener**)
    - Třídami s **uživatelskými metodami pro reakci na událost**
  - „**posluchači**“, které implementují rozhraní „naslouchání“

*Tj. musejí mít schopnost naslouchat dané události*

Obsluha souvisí s tzv. **Event-driven programováním**, které je náplní 4. přednášky

## Příklad zpracování stisku tlačítka

```
JFrame frame = new JFrame("PJV - GUI button click demo");  
Container pane = frame.getContentPane();
```

```
JButton printButton = new JButton("Print");  
printButton.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        System.out.println("User click to print");  
    }  
});
```

```
JButton exitButton = new JButton("Quit");  
exitButton.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        System.out.println("User click to exit");  
        System.exit(0);  
    }  
});
```

```
pane.add(printButton);  
pane.add(exitButton);
```

DemoButtonEvent

# Shrnutí přednášky

# Diskutovaná témata

- GUI v Javě
  - Komponenty a kontejnery
  - Dialogová okna (modalita)
  - Události a obsluha událostí (nástin)
  
- Příště: GUI v Javě a událostmi řízené programování (Even-Driven Programming)

# Diskutovaná témata

- GUI v Javě
  - Komponenty a kontejnery
  - Dialogová okna (modalita)
  - Události a obsluha událostí (nástin)
  
- Příště: GUI v Javě a událostmi řízené programování (Even-Driven Programming)