Laborator 12 - Interfețe Grafice Java

Librării standard pentru interfețe grafice în Java:

- **AWT** - vine de la **Abstract Window Toolkit**, și este un API pentru dezvoltarea interfețelor grafice.

- **Swing** - este varianta care a succedat AWT, cu avantajul ca oferă alternativa creării unor interfețe grafice complet portabile pe orice platforma (spre deosebire de AWT)

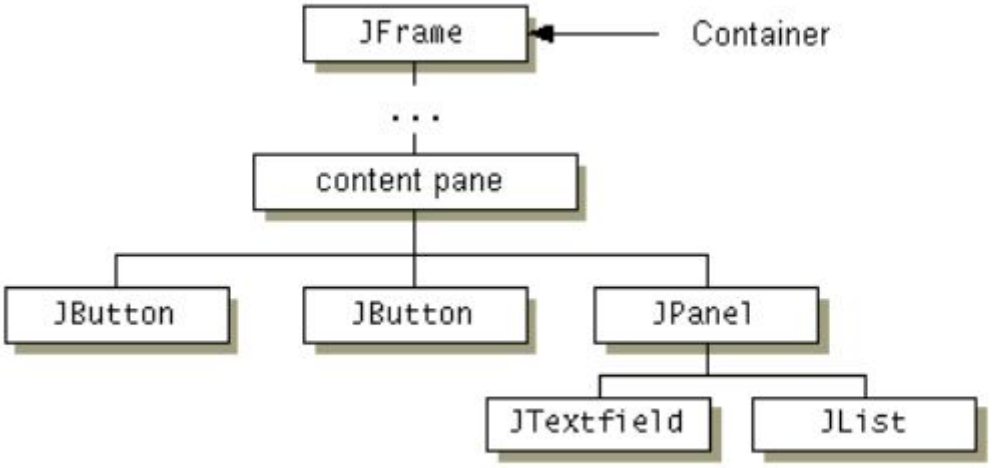
- **JavaFX** - cel mai nou “membru” al librăriilor folosite pentru crearea interfețelor grafice pentru utilizator. Cateva dintre caracteristicile acestora fiind:
    - functionalitati de grafica 3D
    - limbajul de adnotare FXML şi utilitarul SceneBuilder
    - interoperabilitate cu tehnologia Swing.

Java Swing

- pachetul care conține majoritatea claselor pentru aplicațiile Swing este **java.swing**
- permite crearea unor interfeţe utilizator grafice complet portabile pe orice platformă
- clasa principală pentru aplicații Java Swing este clasa **javax.swing.JFrame**
- este posibil să se mixeze componentele AWT cu cele Swing, într-o aplicaţie Java Swing (dar în general se recomandă utilizarea exclusivă a componentelor Swing)

Pachetul swing contine doua tipuri de componente:

- containere (*JFrame, JApplet, JWindow, JDialog*)



- componente "lightweight" (*JButton*, *JPanel*, *JList etc.*)

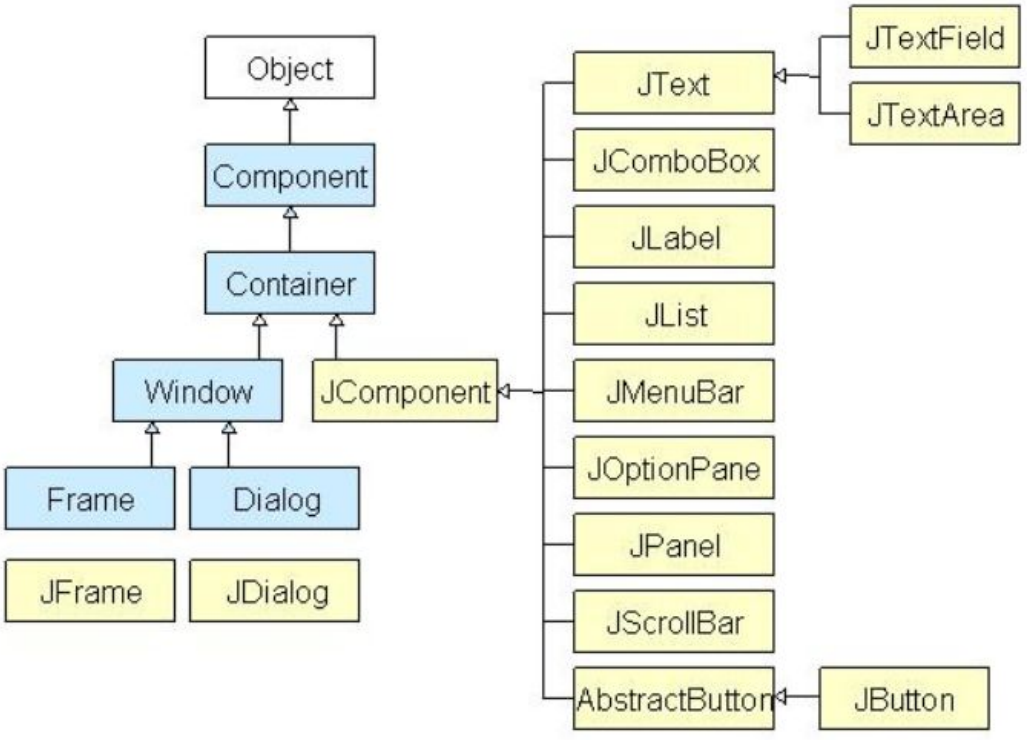


Fig 1 - Ierarhia componentelor Java Swing

Toate obiectele Swing, cu excepția clasei JFrame, moștenesc clasa **javax.swing.JComponent**, care la rândul ei moștenește (indirect) clasa **java.awt.Container**.

- **JPanel** - este un container generic care poate conține alte elemente
- **JWindow** - este similară cu JFrame exceptând faptul că nu are no “title bar”, nu este redimensionabilă, minimizabilă, maximizabilă, şi nu se poate închide.
- **JLabel** - este o componenta utilizata pentru a afișa text sau imagini într-un container.
- **JTextField** - reprezintă o zonă în care se poate introduce text scurt, de o singură linie.
- **JTextArea** - reprezintă o zonă în care se poate introduce text de mai multe linii.
- **JButton** - De cele mai multe ori este utilizat asociindu-i-se un *Event Handler* de tip *ActionListener*

Metode de precizare (setare) sau determinare a dimensiunilor şi a coordonatelor unei componente:

- **public void setSize(int width, int height)** - setează dimensiunile (lăţime, înălţime);
- **public void setSize(Dimension d)** - setează dimensiunile;
- · **public void setBounds(int x, int y, int width, int height)** - setează coordonatele x,y ale originii componentei (colţul din stânga-sus) şi dimensiunile acesteia (lăţime, înălţime);
- **public void setBounds(Rectangle r)** - setează dimensiunile şi coordonatele componentei; · public int getX() - determină coordonata x a originii componentei;
- **public int getY()** - determină coordonata y a originii componentei;
- **public int getWidth** - determină lăţimea componentei;
- **public int getHeight** - determină înălţimea componentei;

JFrame

- este o versiune extinsă a clasei Frame care adaugă suport pentru un comportament de desenare special
- permite componentelor Swing MenuBars să fie ataşate nu numai în partea de sus a ferestrei dar oriunde în fereastră
- obiectele asociate unui JFrame sunt manipulate de o instanţă a clasei JRootPane.
- JRootPane este un container simplu pentru alte câteva componente

```
public class TestFrame extends JFrame {

    public TestFrame() {
        setTitle("Test Application");
        setSize(200, 200);
        setBackground(Color.gray);
        Panel topPanel = new Panel();
        topPanel.setLayout(new BorderLayout());
        getContentPane().add(topPanel);
        Label labelHello = new Label("Hello World!");
        topPanel.add(labelHello, BorderLayout.NORTH);
        JButton jButton1 = new JButton("APASA BUTTONU");
        jButton1.setSize(50, 50);
        topPanel.add(jButton1);
    }

    public static void main(String args[]) {
        TestFrame mainFrame = new TestFrame();
        mainFrame.setVisible(true);
    }

}
```

Ierarhia de obiecte într-o instanță JFrame

JFrame

JRootPane

GlassPane

JLayeredPane

ContentPane

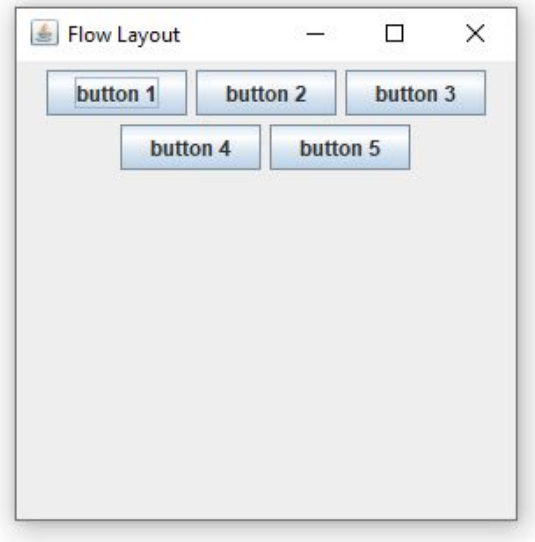
JMenuBar

Layouts

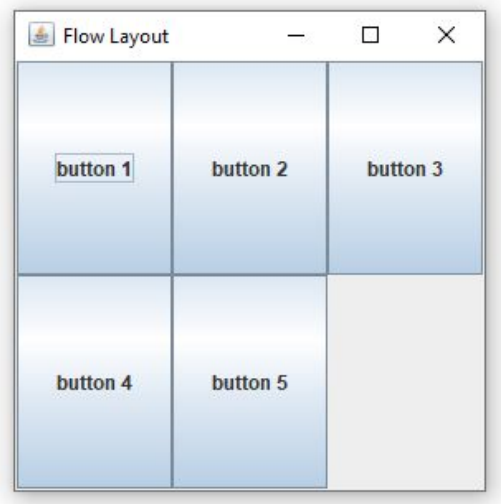
- setul de clase care extind interfața java.awt.LayoutManager
- descriu modul în care elementele se așează într-un **Container**

Cele mai importante sunt:

- **java.awt.FlowLayout -** este utilizat pentru a așeza elementele pe orizontală. Acesta este cel mai simplu tip de layout.

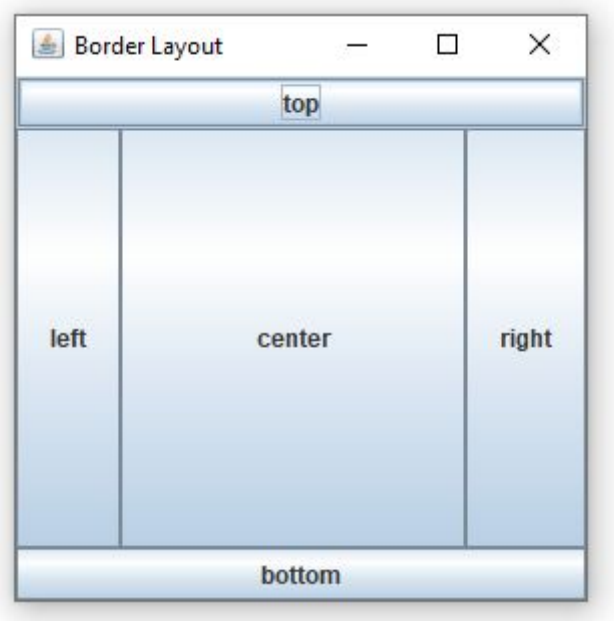


- **java.awt.GridLayout -** este utilizat pentru a așeza elementele într-o matrice cu număr configurabil de linii și coloane.



- **java.awt.BorderLayout** - Este layout-ul default pentru JFrame și este unul dintre cele mai folosite layout-uri.

  Poate sa fie utilizat pentru a așeza elementele de-a lungul celor patru margini și în centru, adică NORTH, SOUTH, EAST, WEST și CENTER. Un singur element poate fi plasat în fiecare din aceste poziții și tot spațiul liber este alocat în centru.



```
BorderLayoutsExample() {
    JFrame frame = new JFrame("Border Layout");
    JButton button, button1, button2, button3, button4;
    button = new JButton("left");
    button1 = new JButton("right");
    button2 = new JButton("top");
    button3 = new JButton("bottom");
    button4 = new JButton("center");
    frame.add(button, BorderLayout.WEST);
    frame.add(button1, BorderLayout.EAST);
    frame.add(button2, BorderLayout.NORTH);
    frame.add(button3, BorderLayout.SOUTH);
    frame.add(button4, BorderLayout.CENTER);

    frame.setSize(300, 300);
    frame.setVisible(true);
}

public static void main(String[] args) {
    new BorderLayoutsExample();
}
```

Poziționarea componentelor în cadrul unui container se folosește un *gestionar de poziționare (Layout Managers)* , care implementează interfata LayoutManager.
Prin intermediul Layout Managerului default poziționarea componentelor se face în mod automat.

Event Handlers

- în Swing, fiecare componenta suportă o listă de evenimente la care e sensibil si fiecarui eveniment i se

poate asociază o acțiune care se execută când acel eveniment se declanșează.
- aceste *handlere* sunt, de fapt, niște metode, definite în anumite interfețe
- metodele se execută în paralel cu programul principal (ca și thread-uri) în momentul în care evenimentul se declanșează
- același handler poate fi asociat mai multor elemente.

Exemple de de listener:
- ActionListener
- Component Listener
- MouseListener
- WindowListener

Exista doua moduri de a adăuga de exemplu un Actionlistener pentru o componenta:

1. Folosește clase anonime și iti definesti acțiunea per button.

```
public class MyGuiForm extends JFrame {

    private JButton button1;
    private javax.swing.JLabel JLabel;
    private JPanel rootPanel;

    public MyGuiForm(){
        add(rootPanel);
        setTitle("This is my title");
        setSize(400, 500);

        button1.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JOptionPane.showMessageDialog(rootPanel, "You've pushed the button...");
            }
        });
    }


}
```

2. Implementează interfața ActionListener și apelează metoda de .addActionListener(this) pe componenta pentru care definesti acțiunea. Ulterior definesteti metoda **public void actionPerformed(ActionEvent e)**

```
public class TestJButton extends JFrame implements ActionListener
{
    private int iCounter = 0; // Keep track of button presses
    private JButton button = null;
    public TestJButton()
    {
        setTitle( "ActionListener Application" );
        setBackground( Color.gray );
        JPanel topPanel = new JPanel();
        topPanel.setLayout( new FlowLayout() );
        topPanel.setPreferredSize( new Dimension( 300, 200 ) );

        getContentPane().add( topPanel );
        button = new JButton( "Press Me" );
        topPanel.add( button );
        button.addActionListener( this );
    }
    public void actionPerformed( ActionEvent event )
    {
        if( event.getSource() == button )
        {
            iCounter++;
            button.setText( "Pressed " + iCounter + " times" );
            System.out.println( "Click" );
            pack();
        }
    }
    public static void main( String args[] )
    {
        TestJButton mainFrame = new TestJButton();
        mainFrame.pack();
        mainFrame.setVisible( true );
    }
}
```