Java

Curs 9

Danciu Gabriel
Decembrie 2018

Swing

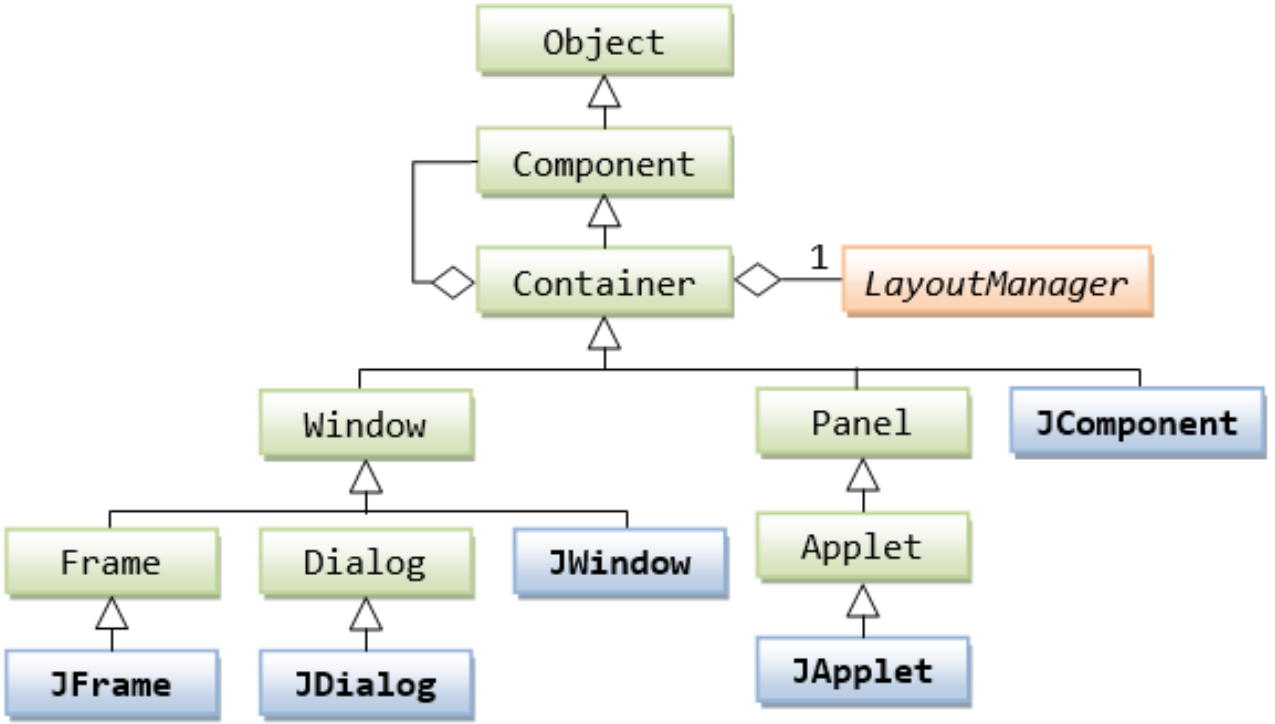
- Deoarece componentele AWT folosesc direct resurse native, ele se numesc *heavyweight*.
  - De aceea aplicațiile scrise cu AWT arată și se comportă diferit pe platforme diferite -> folozofia Java (write once, run anywhere) nu este respectată
  - “Look and feel” variază pe platforme diferite
  - Apar restricții pe unele platforme (exemplu: opacitatea unor componente)
- Swing se bazează pe AWT, nu-l înlocuiește
- Componentele Swing sunt *lightweight*: nu se mulează direct pe componentele native
- Swing suportă PLAF (Pluggable Look And Feel): metal sau Windows “look and feel”

MVC - Swing

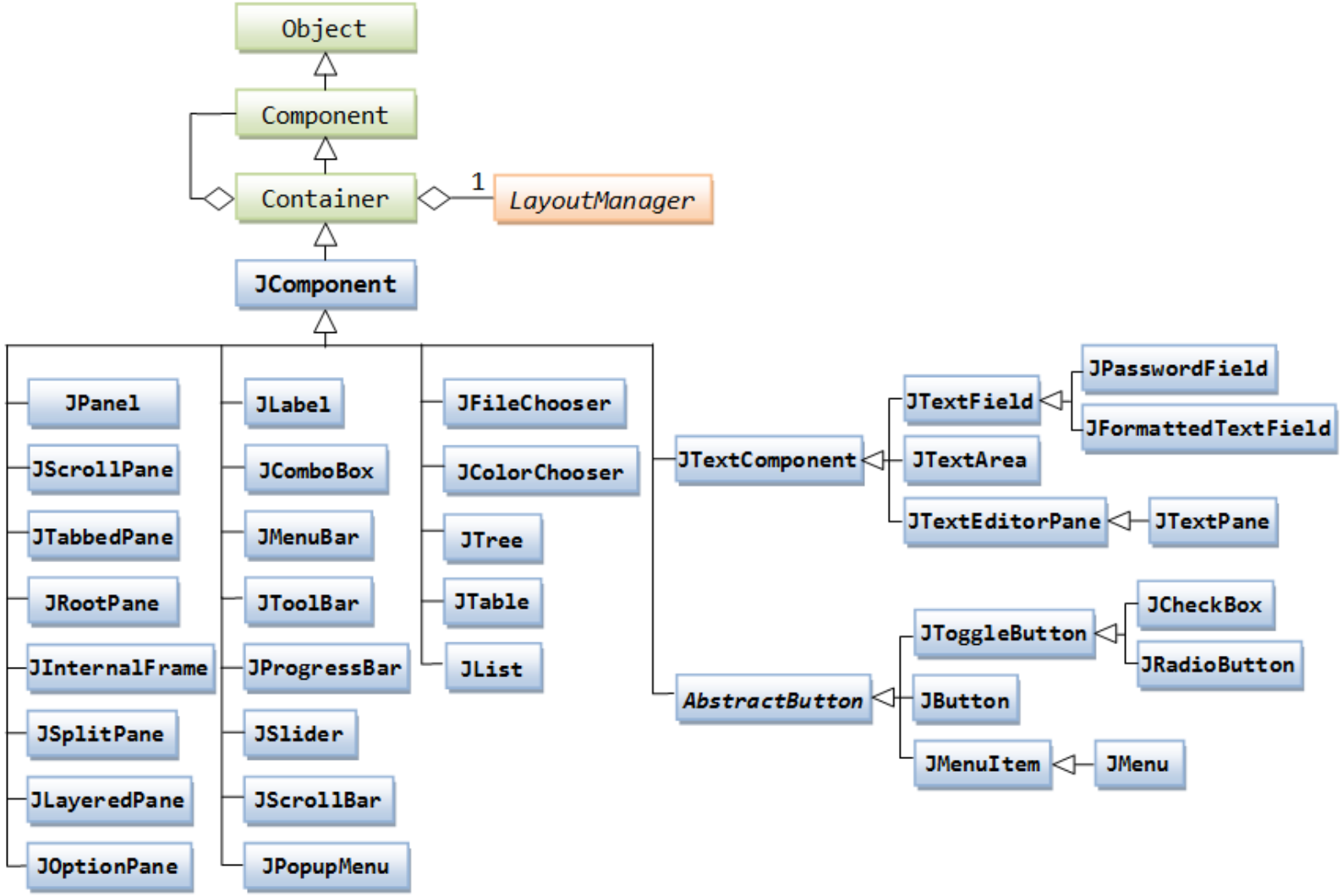
- Orice componentă vizuală este alcătuită din:
  - Modul în care componenta arată pe ecran
  - Modul în care componenta reacționează la acțiunile utilizatorului
  - Starea informației asociată componentei
- MVC – Model – View – Controller
- Model – starea informației asociată componentei
  - Exemplul unui checkbox: un membru boolean încapsulează starea de checked/unchecked a componentei
- View – cum arată componenta pe ecran
  - Exemplu: atunci când se redimensionează fereastra un buton se poate redimensiona sau nu. Totodată își poate schimba locația sau nu.
- Controller – cum reacționează componenta la diferite acțiuni
  - Exemplu: Utilizatorul bifează un checkbox controllerul va modifica starea membrului boolean in *true* și updatează interfața: componenta este bifată.

Component și Container

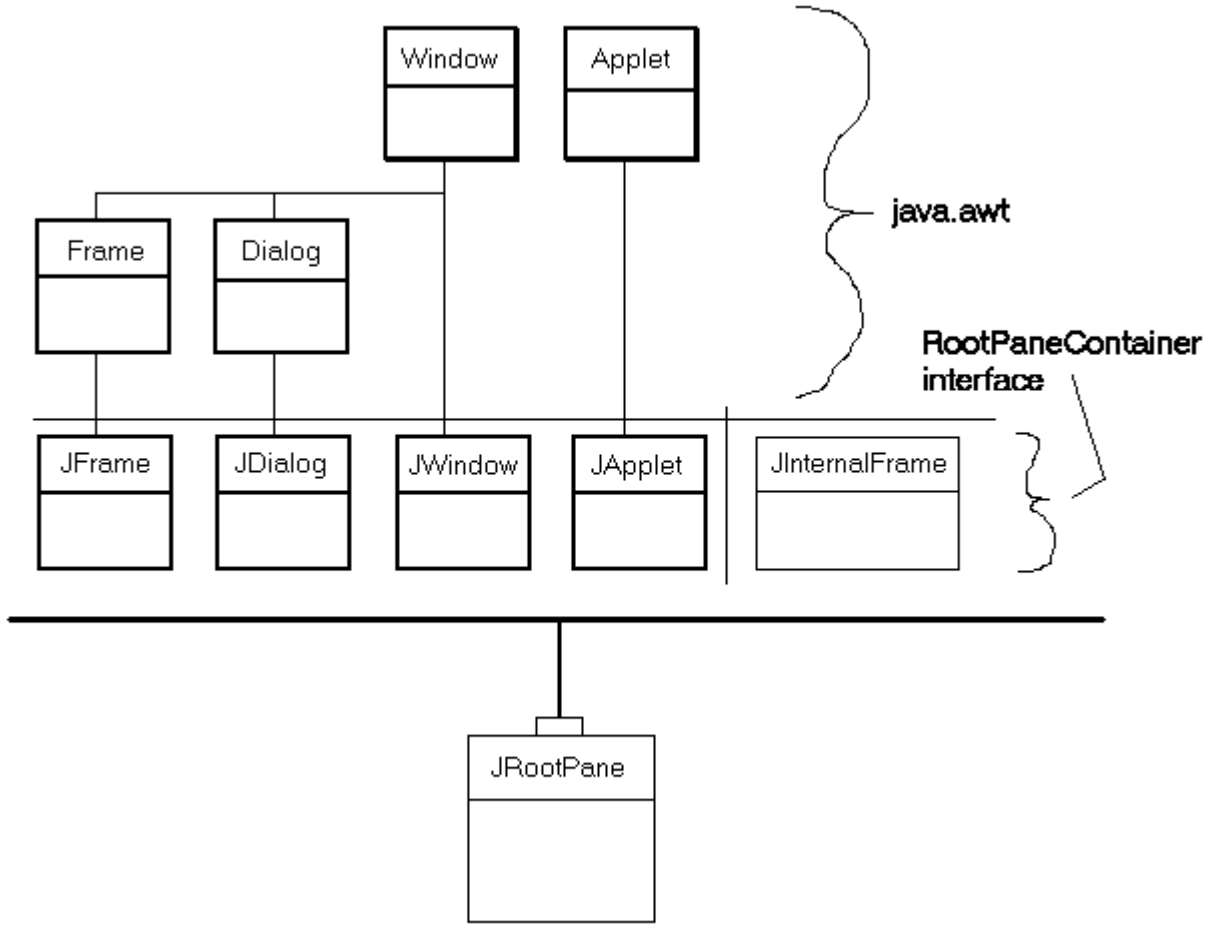
- *Component* derivă din *JComponent*
- Un *Component* este un control independent
- Un *Container* conține un grup de *Component*

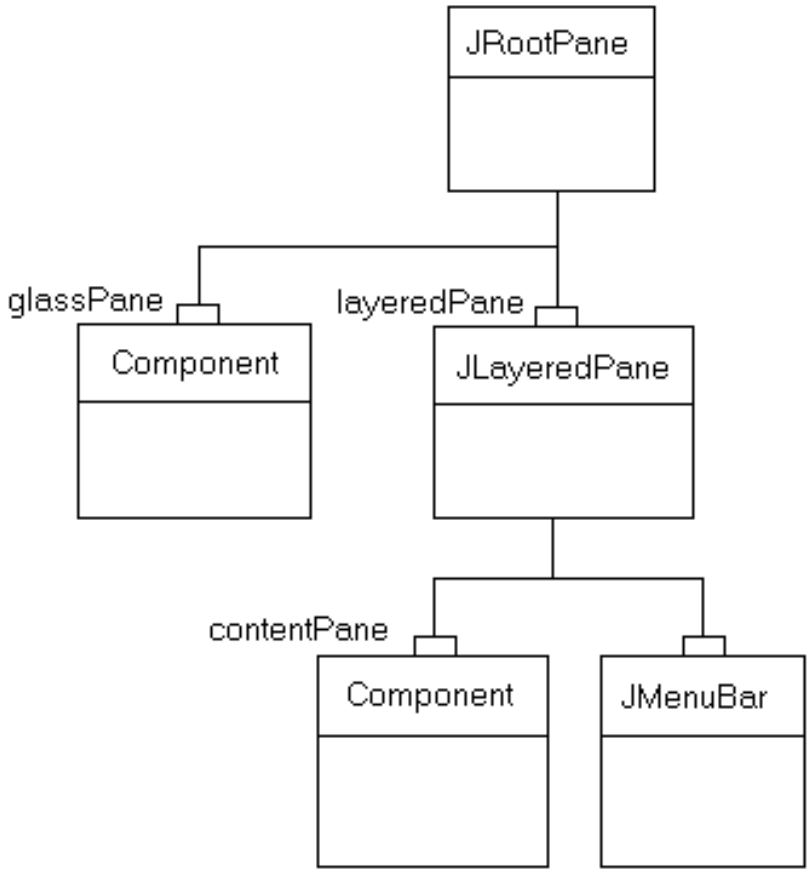


JComponent


Object
Component
Container
1
LayoutManager
JComponent
JPanel
JScrollPane
JTabbedPane
JRootPane
JInternalFrame
JSplitPane
JLayeredPane
JOptionPane
JLabel
JComboBox
JMenuBar
JToolBar
JProgressBar
JSlider
JScrollBar
JPopupMenu
JFileChooser
JColorChooser
JTree
JTable
JList
JTextComponent
JTextField
JPasswordField
JFormattedTextField
JTextArea
JTextEditorPane
JTextPane
AbstractButton
JToggleButton
JCheckBox
JRadioButton
JButton
JMenuItem
JMenu

JRootPane extends JComponent





Un prim exemplu

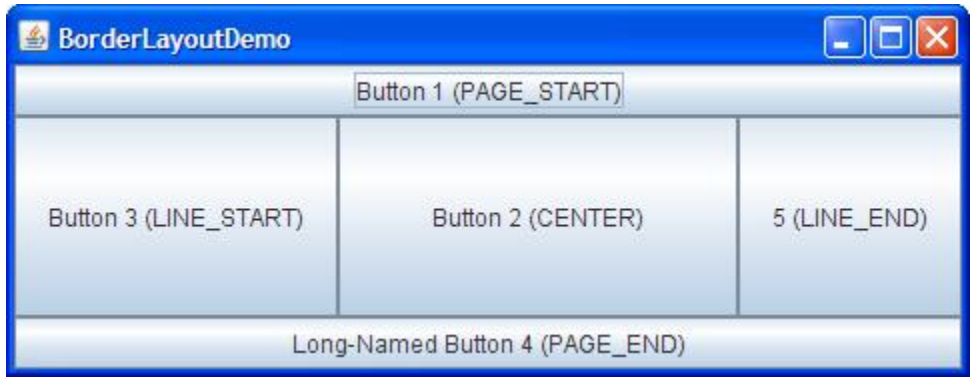
```
package demo;
import javax.swing.*;

// A simple Swing application.
class SwingDemo {

    SwingDemo() {
// Create a new JFrame container.
        JFrame jfrm = new JFrame("A Simple Swing Application");
// Give the frame an initial size.
        jfrm.setSize(400, 300);
// Terminate the program when the user closes the application.

jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
// Create a text-based label.
        JLabel jlab = new JLabel(" Swing means powerful GUIs.");
// Add the label to the content pane.
        jfrm.add(jlab);
// Display the frame.
        jfrm.setVisible(true);
    }

    public static void main(String args[]) {
// Create the frame on the event dispatching thread.
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new SwingDemo();
            }
        });
    }
}
```

Evenimente Swing

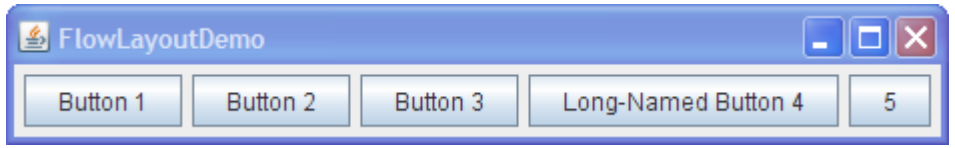
```
package demo;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
class EventDemo {
    JLabel jlab;
    EventDemo() {
// Create a new JFrame container.
        JFrame jfrm = new JFrame("An Event Example");
// Specify FlowLayout for the layout manager.
        jfrm.setLayout(new FlowLayout());
// Give the frame an initial size.
        jfrm.setSize(220, 90);
// Terminate the program when the user closes the application.
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
// Make two buttons.
        JButton jbtnAlpha = new JButton("Alpha");
        JButton jbtnBeta = new JButton("Beta");
// Add action listener for Alpha.
        jbtnAlpha.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                jlab.setText("Alpha was pressed.");
            }
        });
// Add action listener for Beta.
        jbtnBeta.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                jlab.setText("Beta was pressed.");
            }
        });
// Add the buttons to the content pane.
        jfrm.add(jbtnAlpha);
        jfrm.add(jbtnBeta);
// Create a text-based label.
        jlab = new JLabel("Press a button.");
// Add the label to the content pane.
        jfrm.add(jlab);
// Display the frame.
        jfrm.setVisible(true);
    }
    public static void main(String args[]) {
// Create the frame on the event dispatching thread.
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new EventDemo();
            }
        });
    }
}
```

Layouts

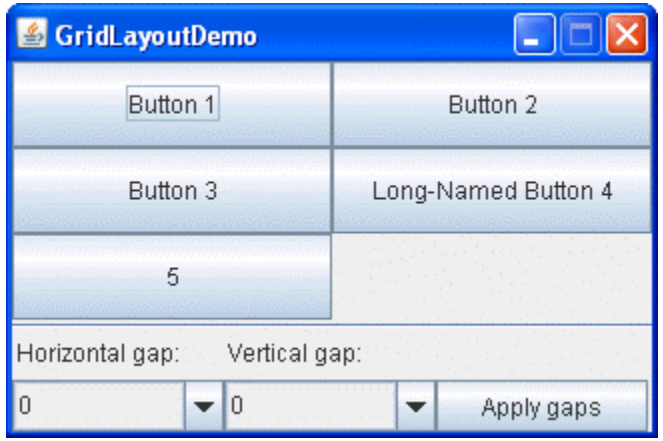
- BorderLayout



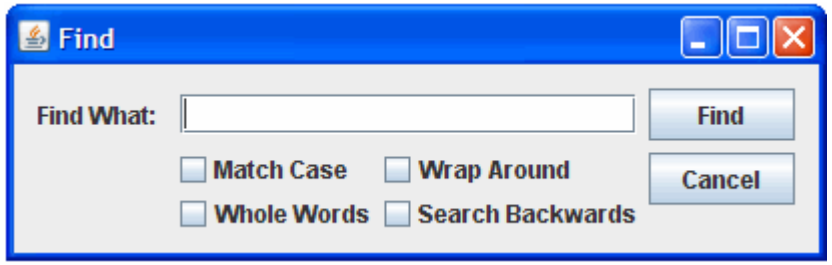
- FlowLayout



- GridLayout



- GroupLayout



Desenarea in Swing

- Mecanism bazat pe AWT *paint()*
- Jcomponent moștenește Component deci și metoda *paint()*
- Metoda paint() este separată în Swing în 3 componente
  - paintComponent()
  - paintBorder()
  - paintChildren()
- Desenarea pe o Componentă se va face prin suprascrierea *paintComponent()*. În metoda suprascrisă prima linie trebuie să fie *super.paintComponent()*.
- Pentru a calcula aria desenabilă se folosește metoda:
  - Insets getInsets()

Componente Swing

- JLabel
- JTextField
- JButton
- JScrollPane
- JList
- JComboBox
- JTree
- JTable

JLabel

```
import java.awt.*;
import javax.swing.*;
public class Demo extends JApplet {
    public void init() {
        try {
            SwingUtilities.invokeAndWait(
                    new Runnable() {
                public void run() {
                    makeGUI();
                }
            }
            );
        } catch (Exception exc) {
            System.out.println("Can't create because of " + exc);
        }
    }
    private void makeGUI() {
// Create an icon.
        ImageIcon ii = new ImageIcon("1.png");
// Create a label.
        JLabel jl = new JLabel("Java", ii, JLabel.CENTER);
// Add the label to the content pane.
        add(jl);
    }
```

JTextField

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class Demo extends JApplet {
    JTextField jtf;
    public void init() {
        try {
            SwingUtilities.invokeAndWait(
                    new Runnable() {
                public void run() {
                    makeGUI();
                }
            }
            );
        } catch (Exception exc) {
            System.out.println("Can't create
because of " + exc);
        }
    }
    private void makeGUI() {
// Change to flow layout.
        setLayout(new FlowLayout());
// Add text field to content pane.
        jtf = new JTextField(15);
        add(jtf);
        jtf.addActionListener(new ActionListener()
{
            public void
actionPerformed(ActionEvent ae) {
// Show text when user presses ENTER.
                showStatus(jtf.getText());
            }
        });
    }
}
```

JButton

```
package demo;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class Demo extends JApplet
        implements ActionListener {
    JLabel jlab;
    public void init() {
        try {
            SwingUtilities.invokeAndWait(
                    new Runnable() {
                public void run() {
                    makeGUI();
                }
            }
            );
        } catch (Exception exc) {
            System.out.println("Can't create because of " + exc);
        }
    }
    private void makeGUI() {
// Change to flow layout.
        setLayout(new FlowLayout());
// Add buttons to content pane.
        ImageIcon hourglass = new ImageIcon("hourglass.jpg");
        JButton jb = new JButton(hourglass);
        jb.setActionCommand("Hourglass");
        jb.addActionListener(this);
        add(jb);
        ImageIcon analog = new ImageIcon("clock.jpg");
        jb = new JButton(analog);
        jb.setActionCommand("Analog Clock");
        jb.addActionListener(this);
        add(jb);
        ImageIcon digital = new ImageIcon("time.jpg");
        jb = new JButton(digital);
        jb.setActionCommand("Digital Clock");
        jb.addActionListener(this);
        add(jb);
        ImageIcon stopwatch = new ImageIcon("timer.jpg");
        jb = new JButton(stopwatch);
        jb.setActionCommand("Timer");
        jb.addActionListener(this);
        add(jb);
    // Create and add the label to content pane.
        jlab = new JLabel("Choose a Timepiece");
        add(jlab);
    }
    // Handle button events.
    public void actionPerformed(ActionEvent ae) {
        jlab.setText("You selected " + ae.getActionCommand());
    }
}
```

CheckBox

```
package demo;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class Demo extends JApplet  implements ItemListener {

    JLabel jlab;
    public void init() {
        try {
            SwingUtilities.invokeAndWait(
                    new Runnable() {
                public void run() {
                    makeGUI();
                }
            }
            );
        } catch (Exception exc) {
            System.out.println("Can't create because of " + exc);
        }
    }
    private void makeGUI() {
// Change to flow layout.
        setLayout(new FlowLayout());
// Add check boxes to the content pane.
        JCheckBox cb = new JCheckBox("C");
        cb.addItemListener(this);
        add(cb);
        cb = new JCheckBox("C++");
        cb.addItemListener(this);
        add(cb);
        cb = new JCheckBox("Java");
        cb.addItemListener(this);
        add(cb);
        cb = new JCheckBox("Perl");
        cb.addItemListener(this);
        add(cb);
// Create the label and add it to the content pane.
        jlab = new JLabel("Select languages");
        add(jlab);
    }
// Handle item events for the check boxes.

    public void itemStateChanged(ItemEvent ie) {
        JCheckBox cb = (JCheckBox) ie.getItem();
        if (cb.isSelected()) {
            jlab.setText(cb.getText() + " is selected");
        } else {
            jlab.setText(cb.getText() + " is cleared");
        }
    }
}
```

JScrollPane

```
// Demonstrate JScrollPane.

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

public class Demo extends JApplet {

    public void init() {
        try {
            SwingUtilities.invokeAndWait(
                    new Runnable() {
                public void run() {

                    makeGUI();
                }
            }
            );
        } catch (Exception exc) {
            System.out.println("Can't create because of " }
+ exc);
        }
    }

    private void makeGUI() {
// Add 400 buttons to a panel.
        JPanel jp = new JPanel();
        jp.setLayout(new GridLayout(20, 20));
        int b = 0;
        for (int i = 0; i < 20; i++) {
            for (int j = 0; j < 20; j++) {
                jp.add(new JButton("Button " + b));
                ++b;
            }
        }
// Create the scroll pane.
        JScrollPane jsp = new JScrollPane(jp);
// Add the scroll pane to the content pane.
// Because the default border layout is used,
// the scroll pane will be added to the center.
        add(jsp, BorderLayout.CENTER);
    }
```

JList

```
package demo;
import javax.swing.*;import javax.swing.event.*;import java.awt.*;import java.awt.event.*;
public class Demo extends JApplet {
    JList<String> jlst;
    JLabel jlab;
    JScrollPane jscrlp;
// Create an array of cities.
    String Cities[] = {"New York", "Chicago", "Houston",
        "Denver", "Los Angeles", "Seattle",
        "London", "Paris", "New Delhi",
        "Hong Kong", "Tokyo", "Sydney"};
    public void init() {
        try {
            SwingUtilities.invokeAndWait(
                    new Runnable() {
                public void run() {
                    makeGUI();
                }
            }
            );
        } catch (Exception exc) {
            System.out.println("Can't create because of " + exc);
        }
    }
    private void makeGUI() {
// Change to flow layout.
        setLayout(new FlowLayout());
// Create a JList.
        jlst = new JList<String>(Cities);
// Set the list selection mode to single selection.
        jlst.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
// Add the list to a scroll pane.
        jscrlp = new JScrollPane(jlst);
// Set the preferred size of the scroll pane.
        jscrlp.setPreferredSize(new Dimension(200, 90));
// Make a label that displays the selection.
        jlab = new JLabel("Choose a City");
// Add selection listener for the list.
        jlst.addListSelectionListener(new ListSelectionListener() {
            public void valueChanged(ListSelectionEvent le) {
// Get the index of the changed item.
                int idx = jlst.getSelectedIndex();
// Display selection, if item was selected.
                if (idx != -1) {
                    jlab.setText("Current selection: " + Cities[idx]);
                } else // Otherwise, reprompt.
                {
                    jlab.setText("Choose a City");
                }
            }
        });
// Add the list and label to the content pane.
        add(jscrlp);
        add(jlab);
    }
}
```

JComboBox

```
package demo;
// Demonstrate JList.

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

public class Demo extends JApplet {

    JLabel jlab;
    ImageIcon hourglass, analog, digital, stopwatch;
    JComboBox<String> jcb;
    String timepieces[] = {"Hourglass", "clock", "time",
"timer"};

    public void init() {
        try {
            SwingUtilities.invokeAndWait(
                    new Runnable() {
                public void run() {
                    makeGUI();
                }
            }
            );
        } catch (Exception exc) {
            System.out.println("Can't create because of " + exc);
        }
    }

    private void makeGUI() {
// Change to flow layout.
        setLayout(new FlowLayout());
// Instantiate a combo box and add it to the content pane.
        jcb = new JComboBox<String>(timepieces);
        add(jcb);
// Handle selections.
        jcb.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                String s = (String) jcb.getSelectedItem();
                jlab.setIcon(new ImageIcon(s + ".jpg"));
            }
        });
// Create a label and add it to the content pane.
        jlab = new JLabel(new ImageIcon("hourglass.jpg"));
        add(jlab);
    }
}
```

JTree

```
import javax.swing.*;
import java.awt.*;
import javax.swing.event.*;
import javax.swing.tree.*;
public class Demo extends JApplet {
    JTree tree;
    JLabel jlab;
    public void init() {
        try {
            SwingUtilities.invokeAndWait(
                    new Runnable() {
                public void run() {
                    makeGUI();
                }
            }
            );
        } catch (Exception exc) {
            System.out.println("Can't create because of " + exc);
        }
    }
    private void makeGUI() {
// Create top node of tree.
        DefaultMutableTreeNode top = new DefaultMutableTreeNode("Options");
// Create subtree of "A".
        DefaultMutableTreeNode a = new DefaultMutableTreeNode("A");
        top.add(a);
        DefaultMutableTreeNode a1 = new DefaultMutableTreeNode("A1");
        a.add(a1);
        DefaultMutableTreeNode a2 = new DefaultMutableTreeNode("A2");
        a.add(a2);
// Create subtree of "B"
        DefaultMutableTreeNode b = new DefaultMutableTreeNode("B");
        top.add(b);
        DefaultMutableTreeNode b1 = new DefaultMutableTreeNode("B1");
        b.add(b1);
        DefaultMutableTreeNode b2 = new DefaultMutableTreeNode("B2");
        b.add(b2);
        DefaultMutableTreeNode b3 = new DefaultMutableTreeNode("B3");
        b.add(b3);
// Create the tree.
        tree = new JTree(top);
// Add the tree to a scroll pane.
        JScrollPane jsp = new JScrollPane(tree);
// Add the scroll pane to the content pane.
        add(jsp);
// Add the label to the content pane.
        jlab = new JLabel();
        add(jlab, BorderLayout.SOUTH);
// Handle tree selection events.
        tree.addTreeSelectionListener(new TreeSelectionListener() {
            public void valueChanged(TreeSelectionEvent tse) {
                jlab.setText("Selection is " + tse.getPath());
            }
        });
    }
}
```

JTable

```
package demo;
// Demonstrate JList.
import javax.swing.*;
import java.awt.*;
import javax.swing.event.*;
public class Demo extends JApplet {
    public void init() {
        try {
            SwingUtilities.invokeAndWait(
                    new Runnable() {
                public void run() {
                    makeGUI();
                }
            }
            );
        } catch (Exception exc) {
            System.out.println("Can't create because of " + exc);
        }
    }
    private void makeGUI() {
// Initialize column headings.
        String[] colHeads = {"Name", "Extension", "ID#"};
// Initialize data.
        Object[][] data = {
            {"Gail", "4567", "865"},
            {"Ken", "7566", "555"},
            {"Viviane", "5634", "587"},
            {"Melanie", "7345", "922"},
            {"Anne", "1237", "333"},
            {"John", "5656", "314"},
            {"Matt", "5672", "217"},
            {"Claire", "6741", "444"},
            {"Erwin", "9023", "519"},
            {"Ellen", "1134", "532"},
            {"Jennifer", "5689", "112"},
            {"Ed", "9030", "133"},
            {"Helen", "6751", "145"}
        };
// Create the table.
        JTable table = new JTable(data, colHeads);
// Add the table to a scroll pane.
        JScrollPane jsp = new JScrollPane(table);
// Add the scroll pane to the content pane.
        add(jsp);
    }
}
```

Exerciții

- Repetați exercițiile de la cursul anterior folosind Swing
- Creați o forma de autentificare:
  - Un textbox pentru nume, un texbox pentru parola
  - Un buton Login, un buton Cancel (închiderea aplicației)
  - Evenimentul atașat *Login* va citi dintr-un fișier *parole.csv* toate id-urile, numele și parolele salvate. Dacă numele și parola sunt găsite în fișier atunci se va afișa un mesaj de succes, altfel se va afișa un mesaj de eroare
  - După 3 încercări nereușite, aplicația se închide
- În cazul în care s-a reușit autentificarea, dintr-un fișier *detalii.csv* se citesc informațiile utilizatorului (id, nume, prenume, vârstă, adresa, telefon) și se încarcă într-o listă (JList) ce va fi afișată într-o nouă formă. Legătura dintre *parole.csv* și *detalii.csv* se face prin *id*.
- Noua formă (cea cu JList) conține și un buton *LogOut* care va închide forma actuală și va reactiva forma de autentificare.