ISCTE IUL
Instituto Universitário de Lisboa

Graphical User Interfaces in Java SWING

Programação Concorrente e Distribuída
Parallel and Distributed Programming

2012-2013

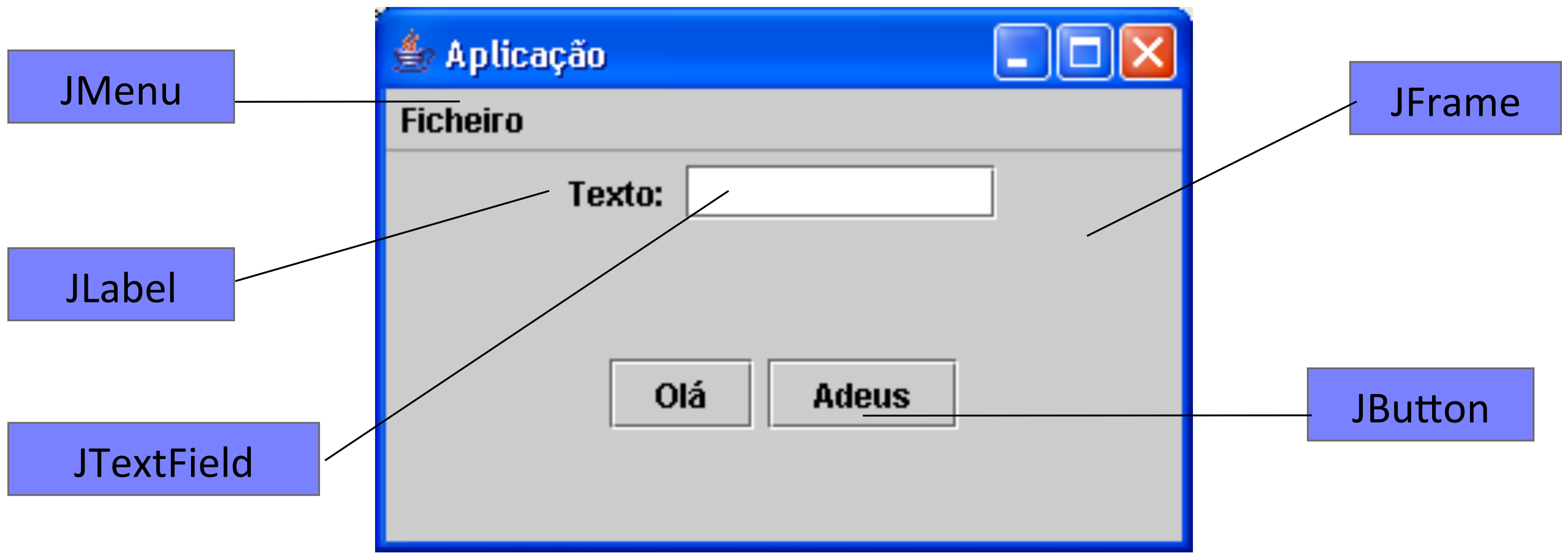
Graphical User Interfaces (GUI)

Each graphical component that the user can see on the screen corresponds to an object of a class

- Component:
  - Window
  - Button
  - Menu
  - ...

- Class:
  - JFrame
  - JButton
  - JMenu
  - ...

Example


JMenu
JLabel
JTextField
Aplicação
Ficheiro
Texto:
Olá
Adeus
JFrame
JButton

Graphical User Interfaces in Java (I)

There are more than 250 classes for graphical user interface components and logic

- They can be used to create intuitive and usable GUIs.
- Programming these interfaces may not always be intuitive and the code can sometimes be difficult to read.
- Some IDEs include tools for graphically designing GUIs and automatic code generation **(we will not use those in the course)**.

Graphical User Interfaces in Java (II)

- The AWT library
  - The first library for implementing GUIs in Java.
  - For performance reasons, the AWT components used the underlying components on the execution platform (Solaris, Windows, Linux, …):

- Version J2SE 1.2 and onwards include the Swing framework.

Swing’s Architecture

- Swing is a framework for implementation of GUIs that allows for separation of interface and data.

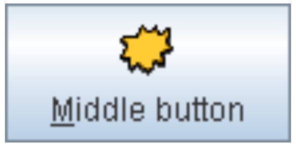
- Swing uses one thread (*event-dispatch thread*).

- Swing allows for the implementation of platform independent GUIs.

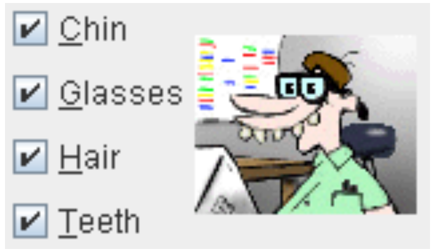
A Visual Index to the Swing Components (I)

http://docs.oracle.com/javase/tutorial/ui/features/components.html

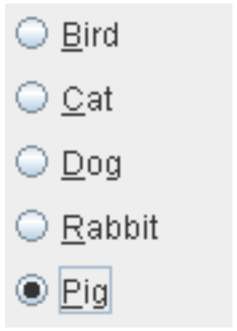
JButton



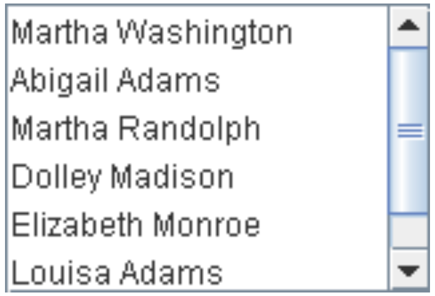
JCheckBox



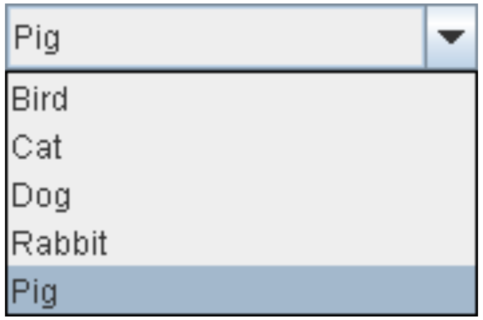
JRadioButton



JList



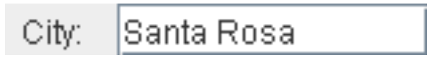
JComboBox



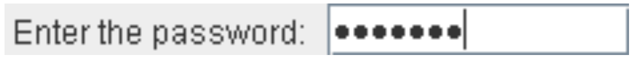
A Visual Index to the Swing Components (II)

http://docs.oracle.com/javase/tutorial/ui/features/components.html

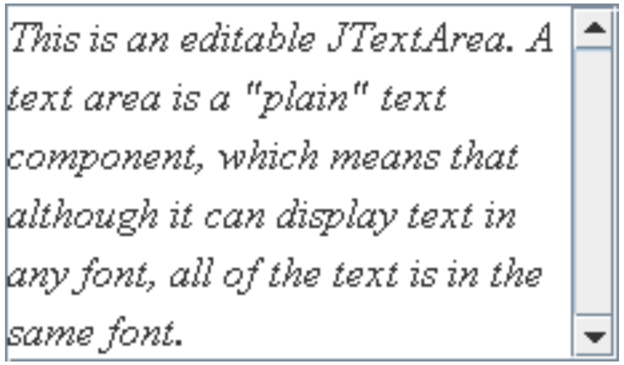
JTextField


City: Santa Rosa

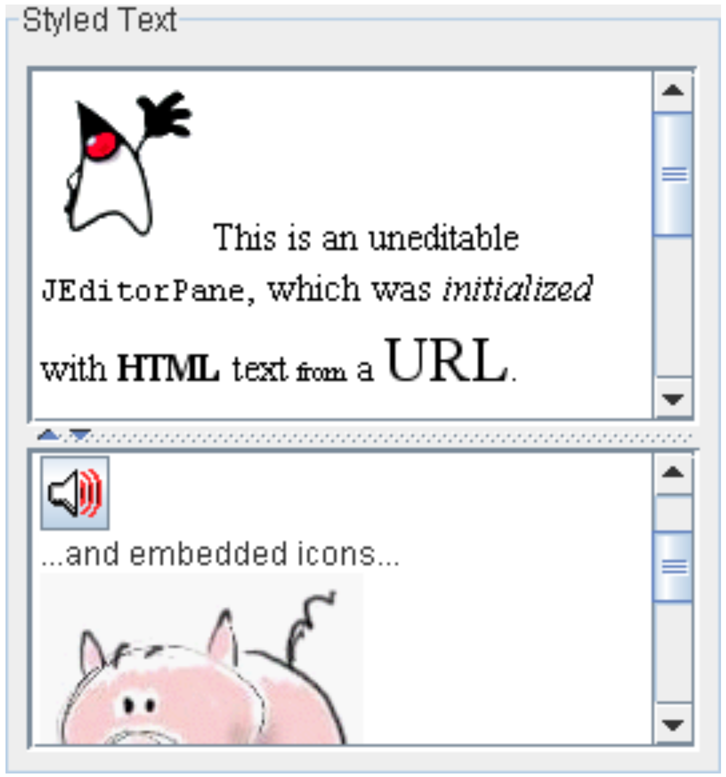
JPasswordField


Enter the password: ●●●●●●●●

JTextArea


This is an editable JTextArea. A text area is a "plain" text component, which means that although it can display text in any font, all of the text is in the same font.

JEditorPane


Styled Text

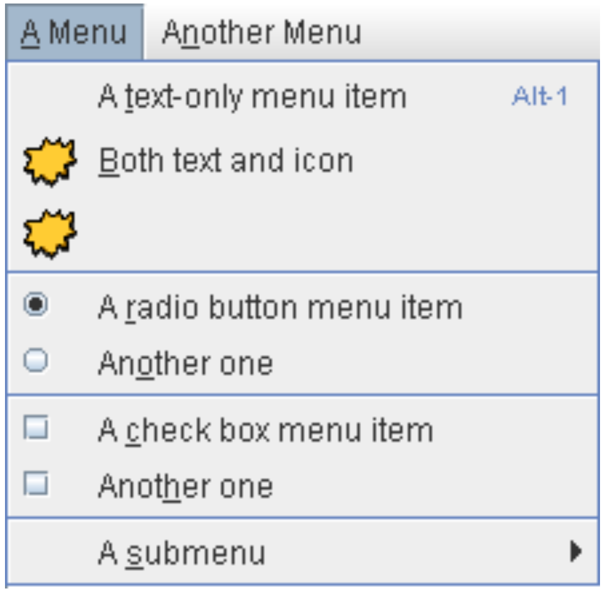
This is an uneditable `JEditorPane`, which was *initialized* with **HTML** text from a URL.

...and embedded icons...

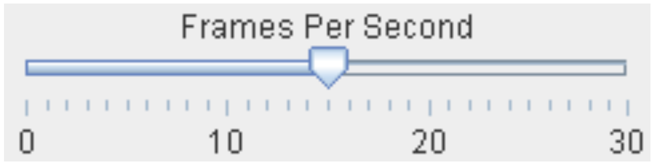
A Visual Index to the Swing Components (III)

http://docs.oracle.com/javase/tutorial/ui/features/components.html

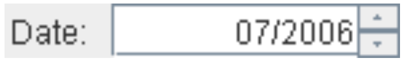
JMenu



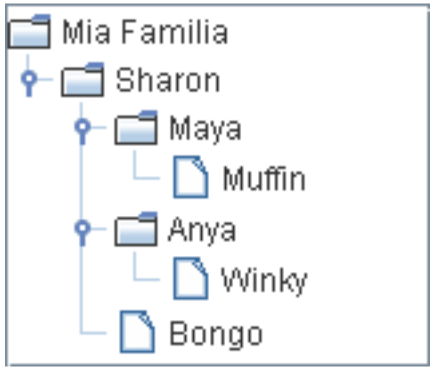
JSlider



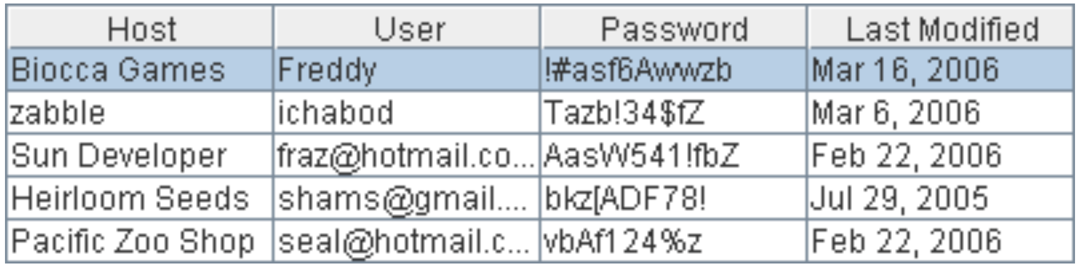
JSpinner



JTree



JTable

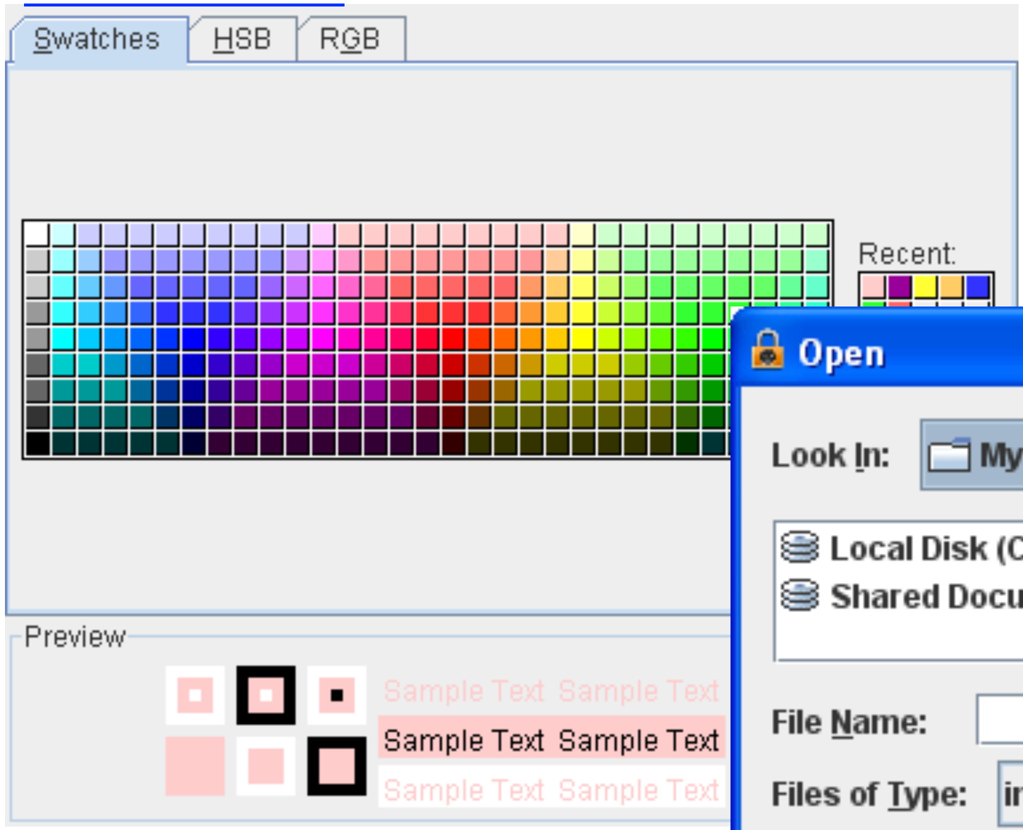


Host	User	Password	Last Modified
Biocca Games	Freddy	!#asf6Awwzb	Mar 16, 2006
zabble	ichabod	Tazb!34$fZ	Mar 6, 2006
Sun Developer	fraz@hotmail.co...	AasW541!fbZ	Feb 22, 2006
Heirloom Seeds	shams@gmail....	bkz[ADF78!	Jul 29, 2005
Pacific Zoo Shop	seal@hotmail.c...	vbAf124%z	Feb 22, 2006

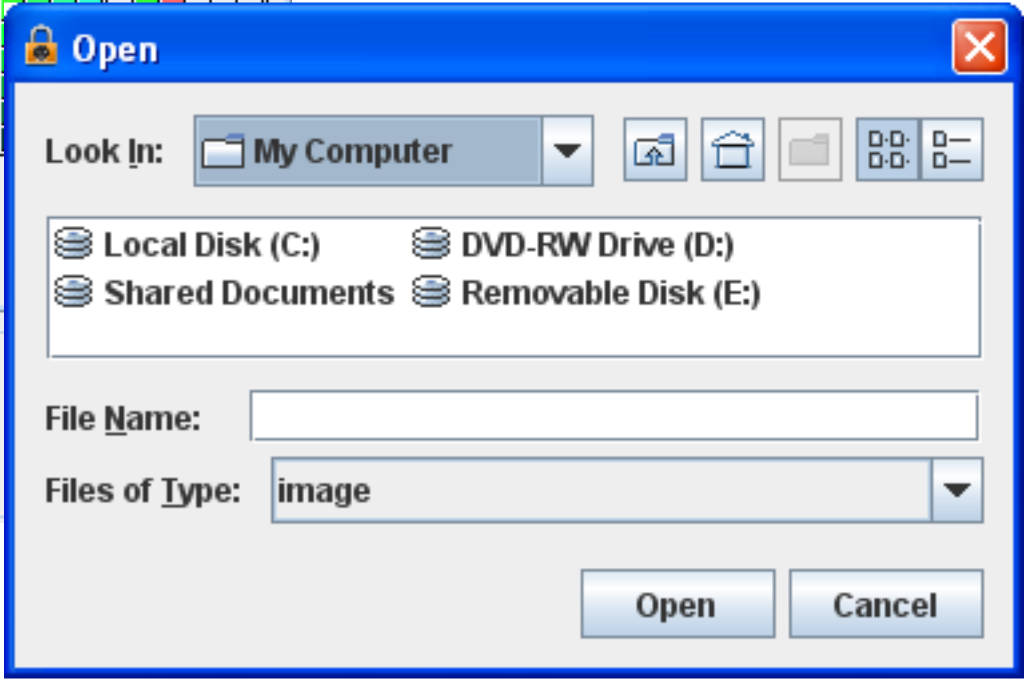
A Visual Index to the Swing Components (IV)

http://docs.oracle.com/javase/tutorial/ui/features/components.html

JColorChooser



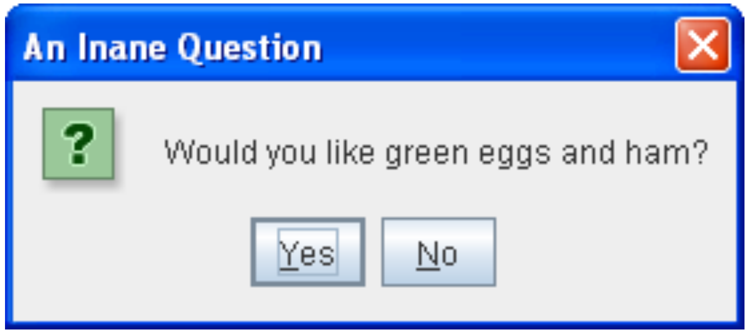
JFileChooser



A Visual Index to the Swing Components (V)

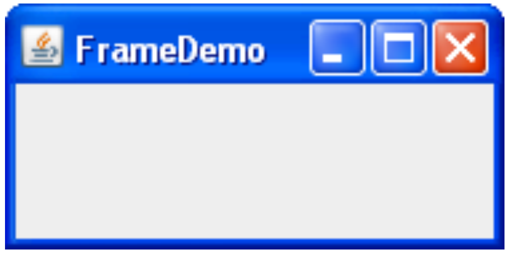
http://docs.oracle.com/javase/tutorial/ui/features/components.html

JDialog



Would you like green eggs and ham?

JFrame

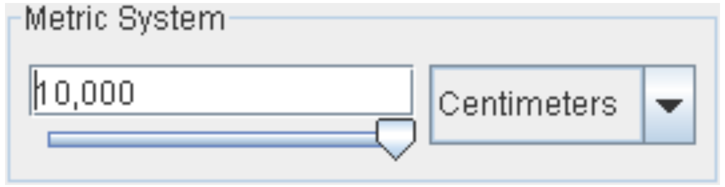


JApplet

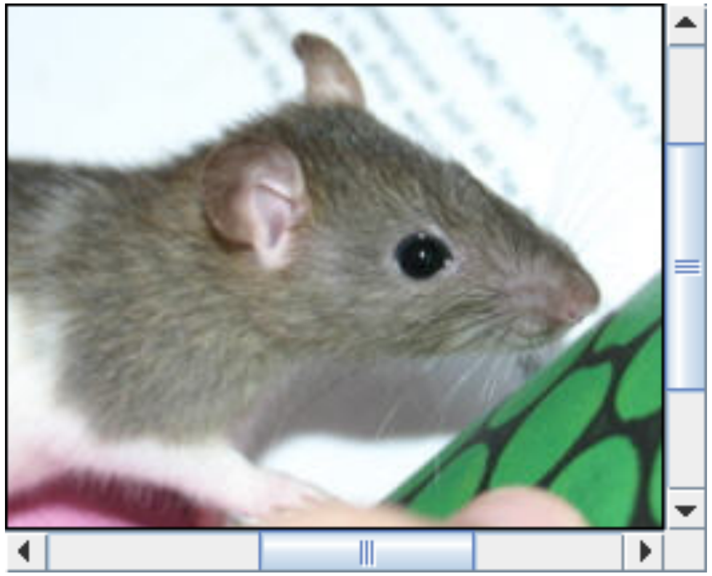
A Visual Index to the Swing Components (VI)

http://docs.oracle.com/javase/tutorial/ui/features/components.html

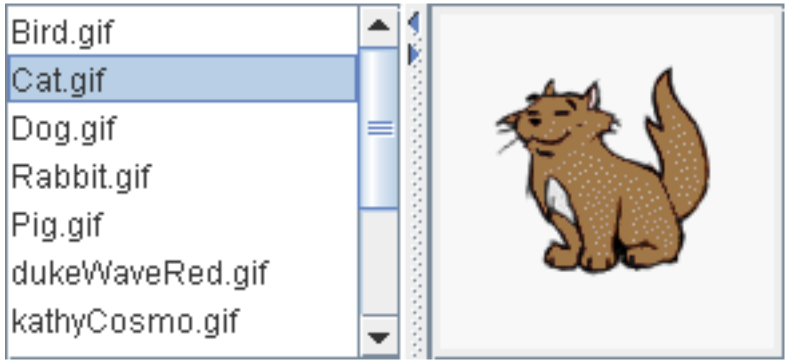
JPanel



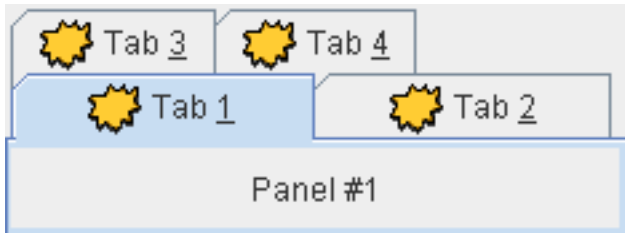
JScrollPane



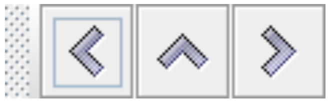
JSplitPane



JTabbedPane



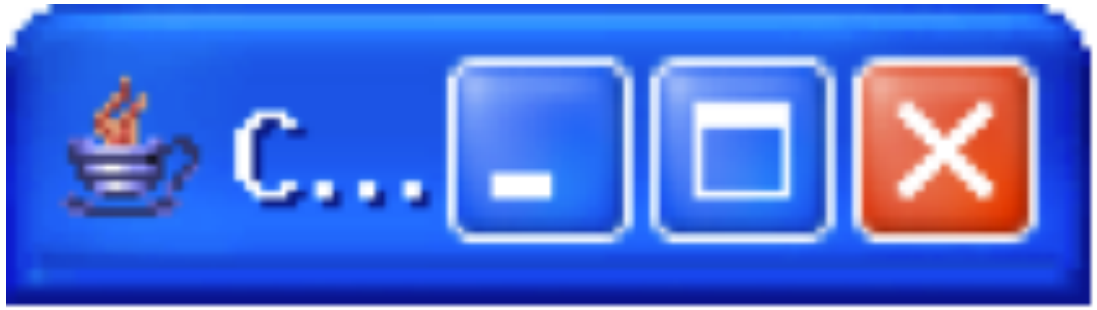
JToolBar



Window based GUIs

Creating and displaying a window:

```
import javax.swing.JFrame;

public class Calculator{
    public static void main(String[] args) {
      JFrame frame= new JFrame(”Calculator");
      frame.setVisible(true);
    }
}
```

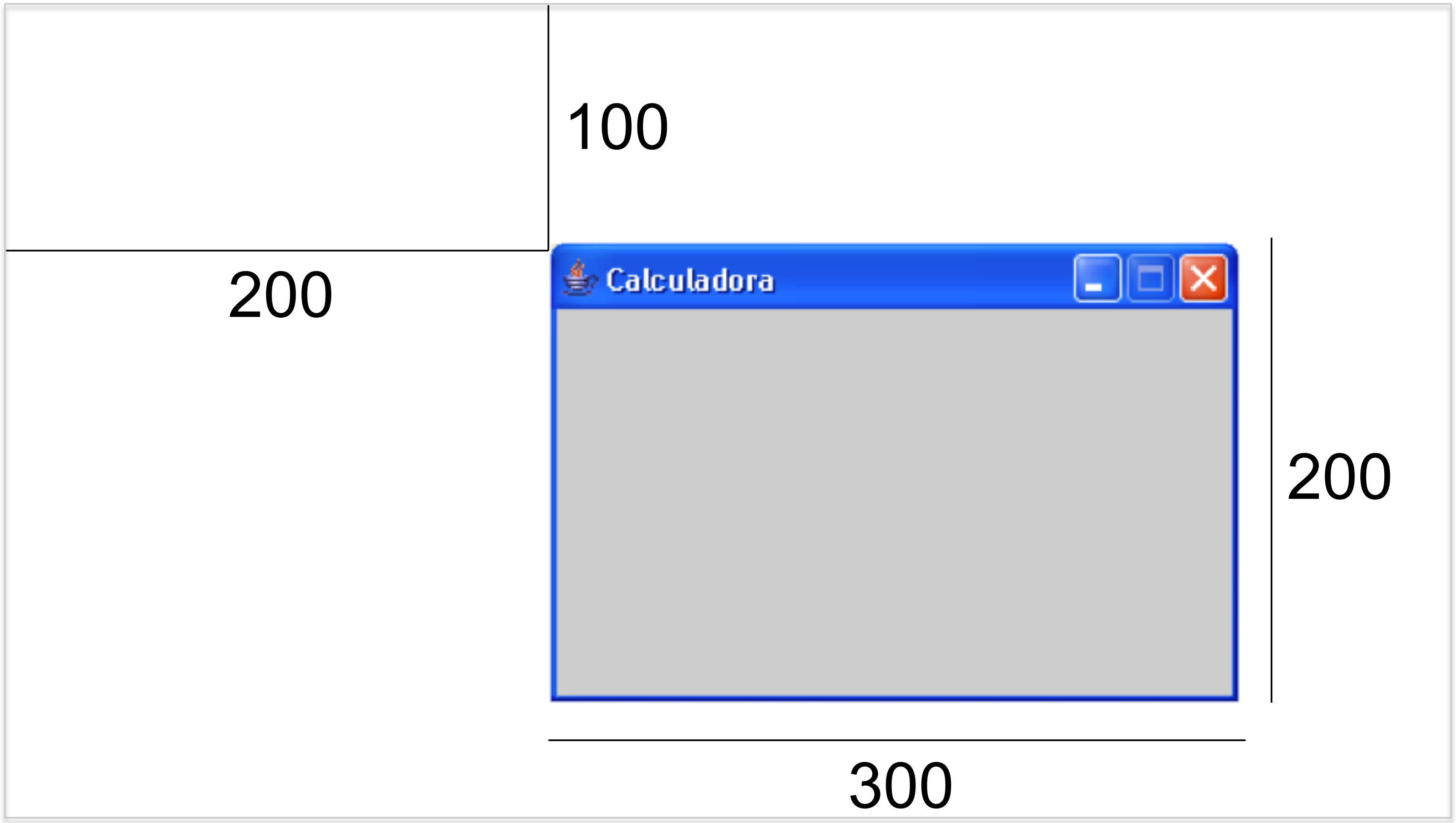


JFrame:

```
import javax.swing.Jframe;

public class Calculator{
  public static void main(String[] args) {
        Jframe frame= new JFrame("Calculadora");

        // Estabelece o tamanho da janela em pixeis (largura, altura)
        frame.setSize(300, 200);

        // Estabelece a localização da janela: distância do canto superior
        // esquerdo da janela ao canto superior esquerdo do ecrã
        frame.setLocation(200, 100);

        // A janela não é redimensionável
        frame.setResizable(false);

        // Quando a janela é fechada a aplicaçãotermina
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        frame.setVisible(true);
  }
}
```

Very important!

A `JFrame` on the desktop

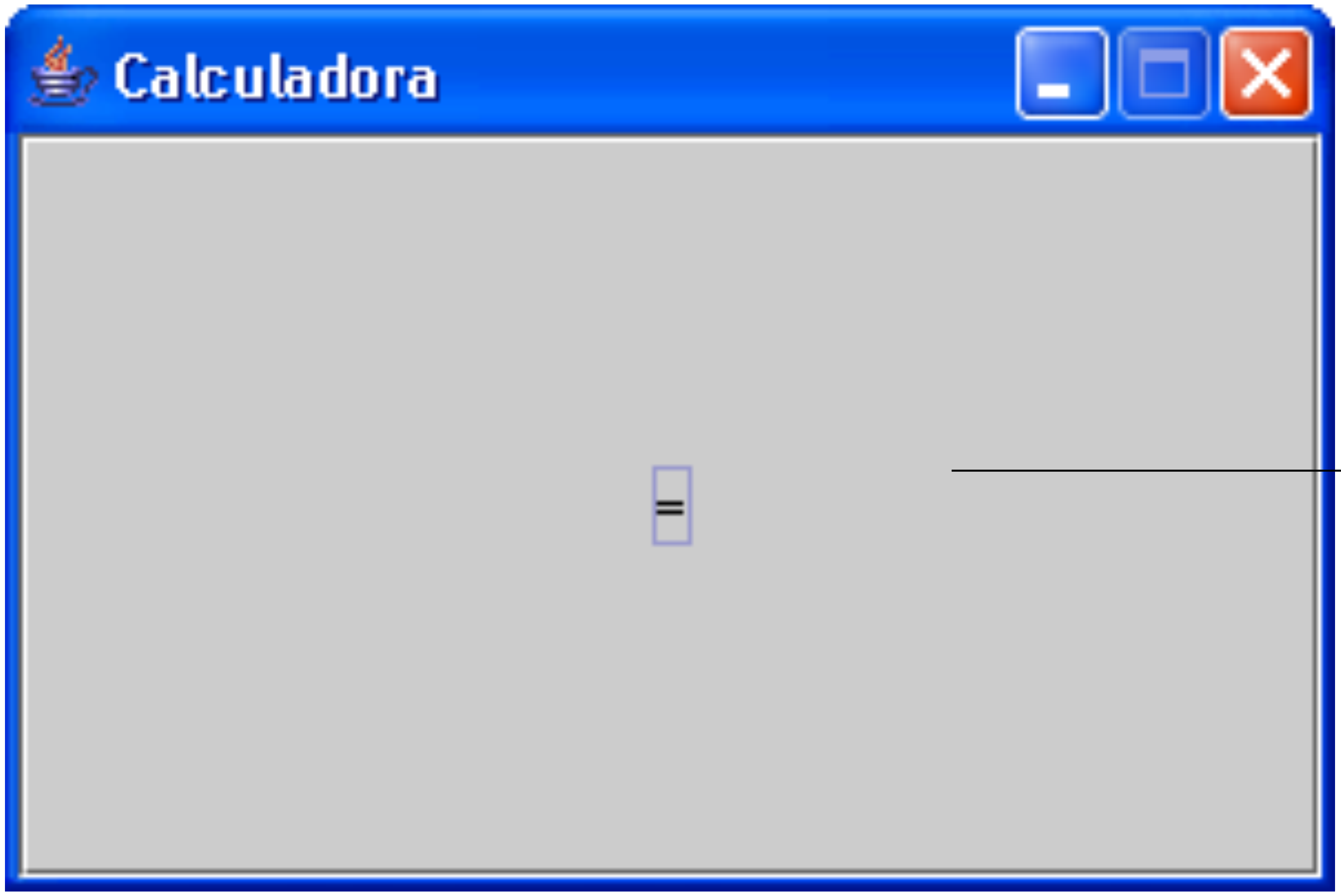


Adding components to the window

Add a button:

```
import javax.swing.JFrame;
import javax.swing.JButton;

public class Calculator{
    public static void main(String[] args) {
         JFrame frame= new JFrame("Calculadora");

         frame.add(new JButton("="));

         frame.setSize(300, 200);
         frame.setLocation(200, 100);
         frame.setResizable(false);
         frame.setDefaultCloseOperation(
                             JFrame.EXIT_ON_CLOSE);
         frame.setVisible(true);
    }
}
```

A window with a button ("=")



The button occupies the entire window

Layout of components

- There are different layout managers in Swing:
  - `FlowLayout`
  - `GridLayout`
  - `BorderLayout`
  - …
- The components added to another component (such as a container) will be laid out according to the layout manager set on that component.
- `import java.awt.`*`LayoutEspecífico`*`;`

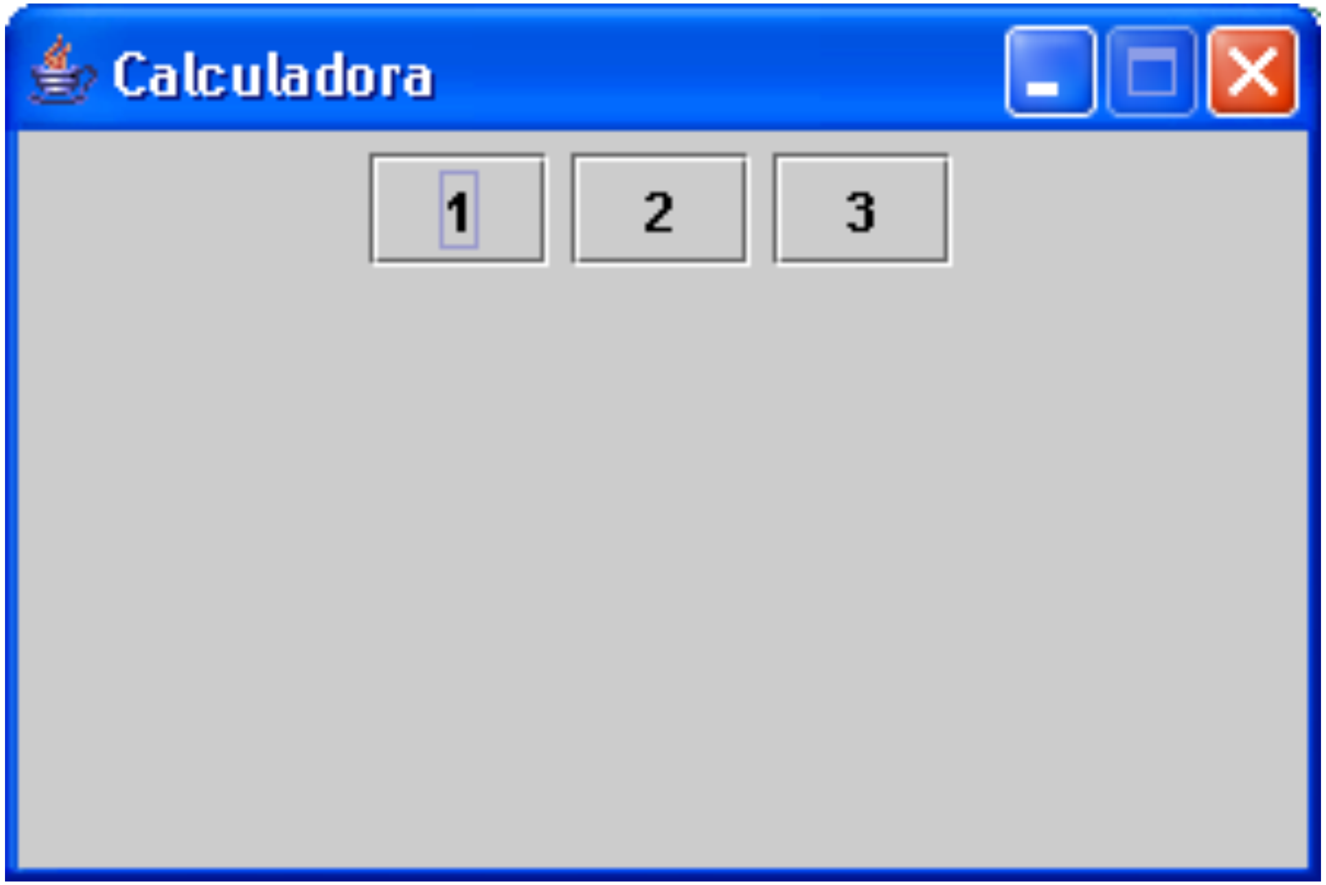
Layout Managers

- LayoutManagers do the following:
  - Compute the minimum, preferred and maximum size of a container.
  - Lays out the components added to the container (its children)
- The layout is done using the characteristics of the manager and the minimum, preferred, and maximum size of its children.

FlowLayout

```
import java.awt.Container;
import java.awt.FlowLayout;
import javax.swing.JFrame;
import javax.swing.JButton;

public class Calculator{
    public static void main(String[] args) {
        JFrame frame= new JFrame("Calculadora");

        frame.setLayout(new FlowLayout());
        frame.add(new JButton("1"));
        frame.add(new JButton("2"));
        frame.add(new JButton("3"));

        frame.setSize(300, 200);
        frame.setLocation(200, 100);
        frame.setResizable(false);
        frame.setDefaultCloseOperation(
                                         JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

FlowLayout

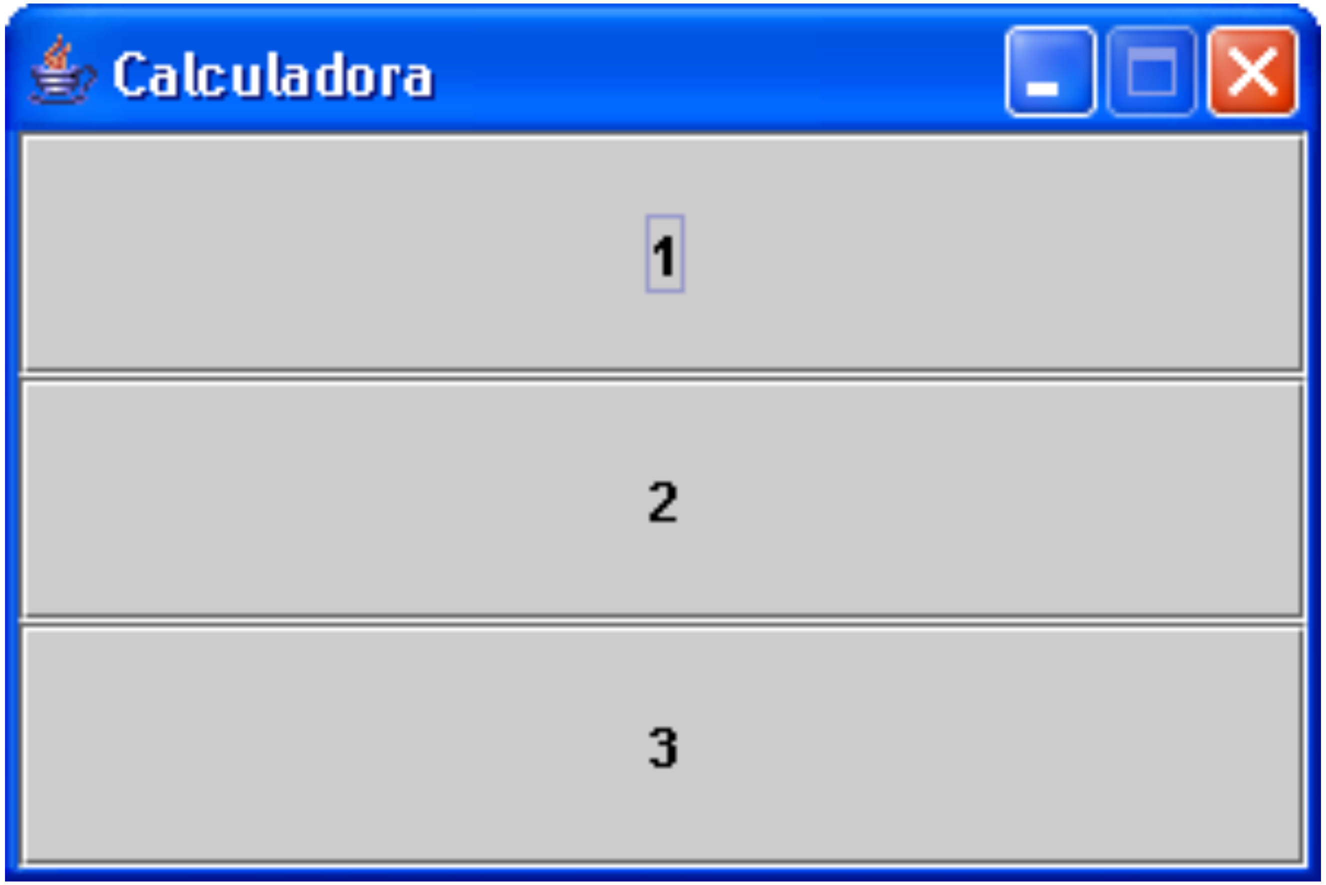


GridLayout

```
import java.awt.Container;
import java.awt.GridLayout;
import javax.swing.JFrame;
import javax.swing.JButton;

public class Calculator{
    public static void main(String[] args) {
        JFrame frame= new JFrame("Calculadora");

        frame.setLayout(new GridLayout(3,1));
        frame.add(new JButton("1"));
        frame.add(new JButton("2"));
        frame.add(new JButton("3"));

        frame.setSize(300, 200);
        frame.setLocation(200, 100);
        frame.setResizable(false);
        frame.setDefaultCloseOperation(
                                 JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
     }
}
```

3 rows and
1 column

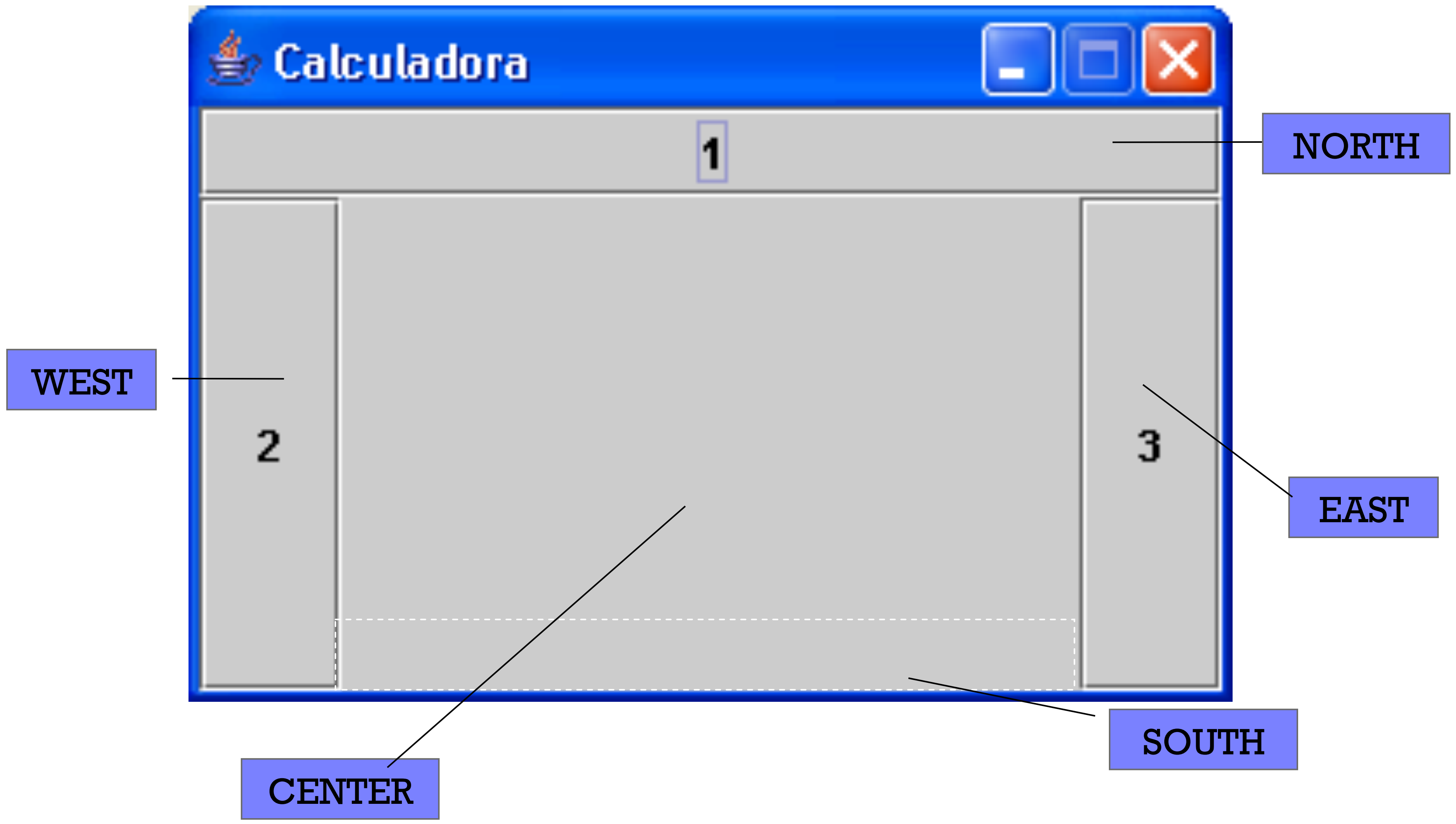
GridLayout



BorderLayout

```
import java.awt.Container;
import java.awt.BorderLayout;
import javax.swing.JFrame;
import javax.swing.JButton;

public class Calculator{
    public static void main(String[] args) {
        JFrame frame= new JFrame("Calculadora");

        frame.setLayout(new BorderLayout());
        frame.add(new JButton("1"), BorderLayout.NORTH);
        frame.add(new JButton("2"), BorderLayout.WEST);
        frame.add(new JButton("3"), BorderLayout.EAST);

        frame.setSize(300, 200);
        frame.setLocation(200, 100);
        frame.setResizable(false);
        frame.setDefaultCloseOperation(
                                         JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

BorderLayout

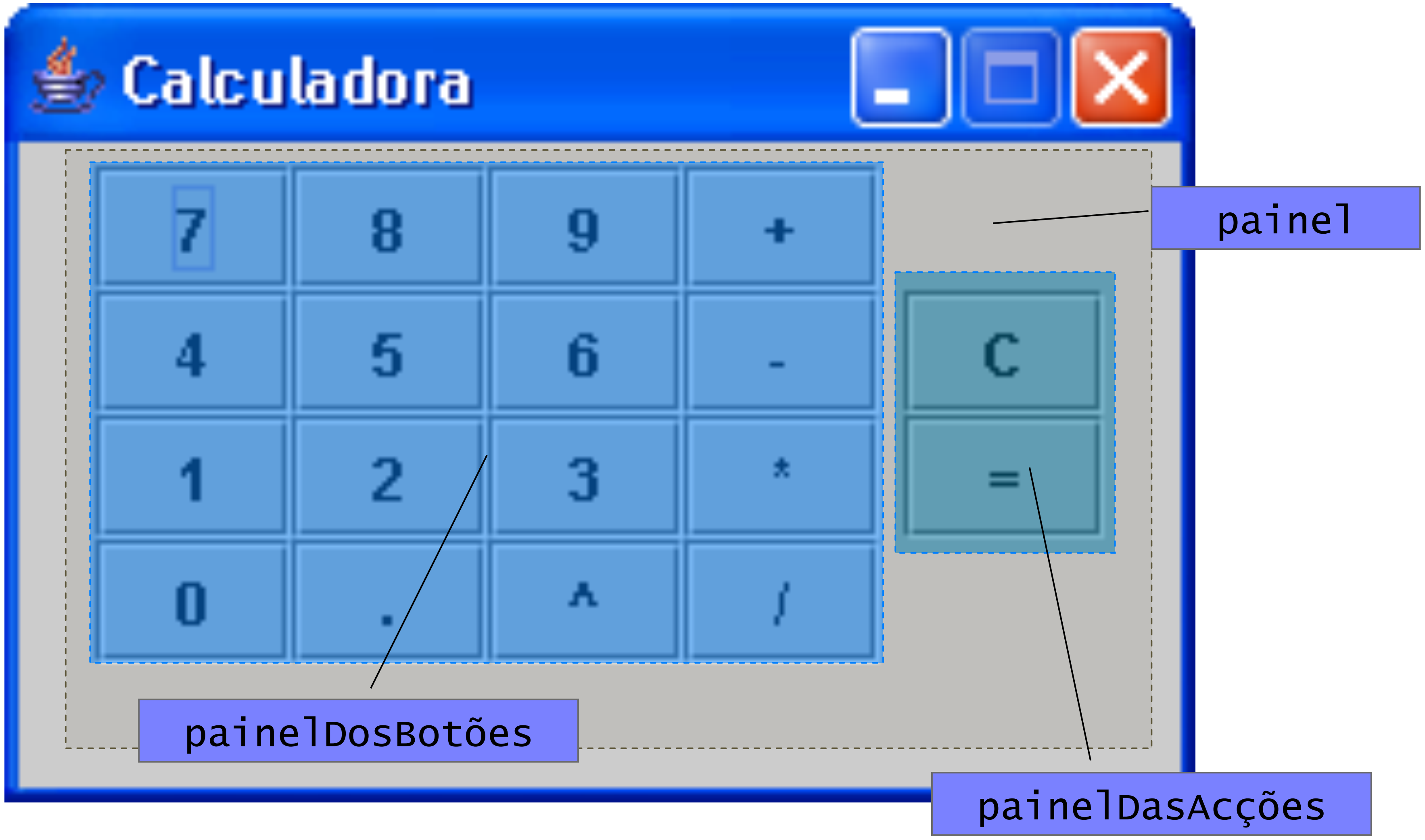


Other Layout Managers

There are other LayoutManagers:

- BoxLayout like FlowLayout, but with more options.
- CardLayout allows for several components to occupy the same space. Allows for creation of taps and tabbed panes.
- GridBagLayout is very flexible (and complex). Defines a grid, but allows fine control over how components are placed in the grid .
- GroupLayout was developed for IDE frameworks for GUI builders but can be used by hand as well. Uses two independent layouts (vertical and horizontal).
- SpringLayout is very flexible and very low-level. Only for GUI builders.
- **And you can create your own …**

JPanels



Calculadora
7
8
9
+
4
5
6
-
1
2
3
*
0
.
^
/
C
=
painel
painelDosBotões
painelDasAcções

Combining different layouts using `JPanel`

```
frame.setLayout(new BorderLayout());

JPanel painel = new JPanel();

final String caracteresDosBotões= "789+456-123*0.^/";
JPanel painelDosBotões= new JPanel();
painelDosBotões.setLayout(new GridLayout(4, 4));

for(int i = 0; i != caracteresDosBotões.length(); i++)
       painelDosBotões.add(new JButton("" +
              caracteresDosBotões.charAt(i)));

painel.add(painelDosBotões);

JPanel painelDasAcções= new JPanel();
painelDasAcções.setLayout(new GridLayout(2, 1));
painelDasAcções.add(new JButton("C"));
painelDasAcções.add(new JButton("="));

painel.add(painelDasAcções);

frame.add(painel, BorderLayout.CENTER);
```

If no layout is given, **`FlowLayout`** is used in `JPanel`s

JTextField

Create and add to a `Container` or to a `JPanel`:

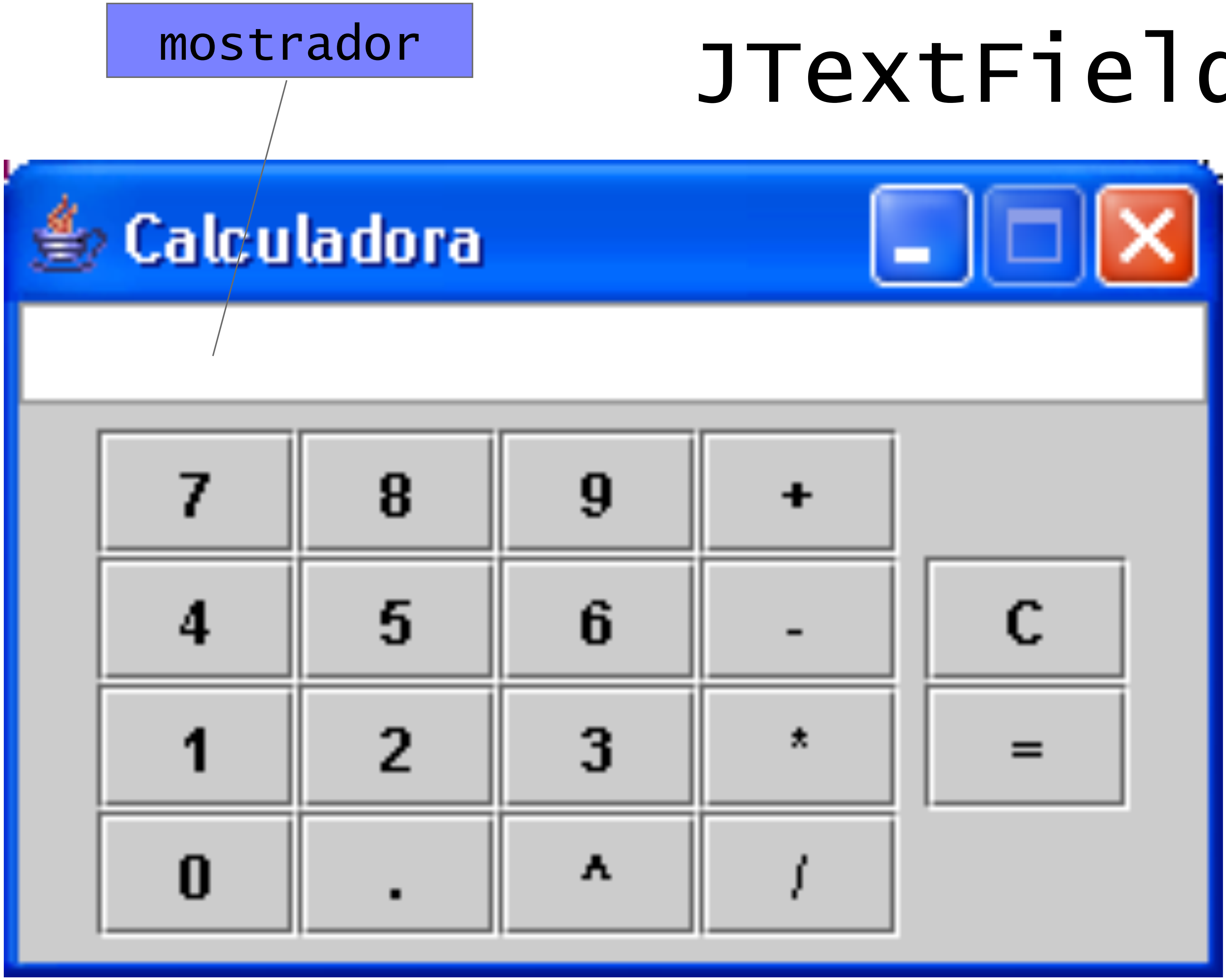
```
JTextField mostrador = new JTextField();
frame.add(mostrador, BorderLayout.NORTH);
```

Some operations:

In order to use `Font`, do
`import java.awt.Font;`

```
// Make the background white
mostrador.setBackground(Color.WHITE);
// Use a specific font "Arial, regular, tamanho 14"
mostrador.setFont(new Font("Arial", Font.PLAIN, 14));
// Right-align text
mostrador.setHorizontalAlignment(JTextField.RIGHT);
// Disable editing
mostrador.setEditable(false);
```

JTextField



JTextField

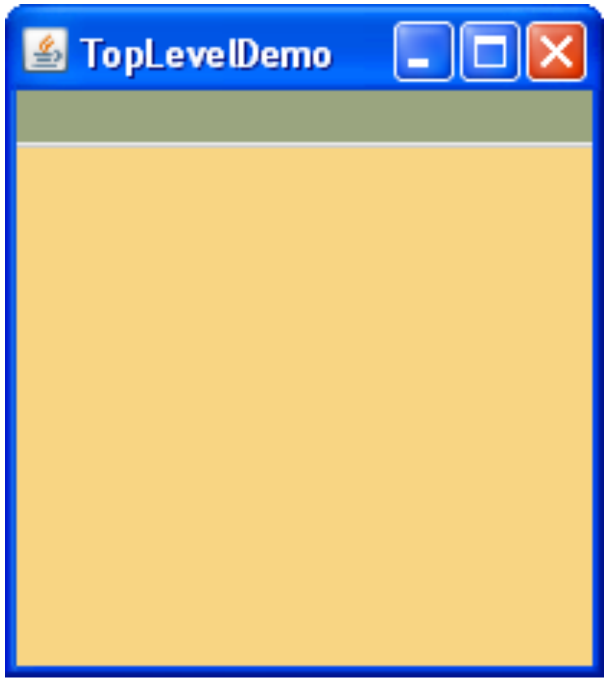
Getting the text in a text field:

```
String texto = mostrador.getText();
```

Setting the text in a text field

```
mostrador.setText(“1 + 3 * 4”);
```

Top level Containers



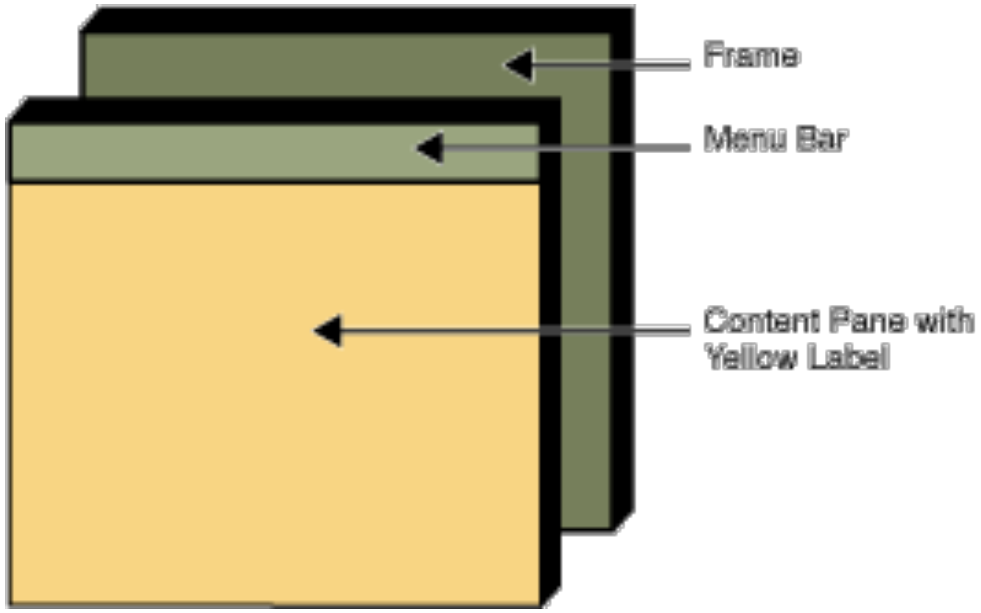


Image source: http://docs.oracle.com/javase/tutorial/uiswing/components/toplevel.html

Container: Access content in the window

```
JFrame frame = new JFrame("Calculadora");
Container container = frame.getContentPane();
container.setBackground(Color.WHITE)
container.add(new JButton("="));
```

```
To use Containerand Color:

import java.awt.Container;
import java.awt.Color;
```

Set the background color of the window

How do we make an application react when a button is pressed?

- Event-based programming
  - Whenever an event occurs, a specific piece of code is execute

- Events:
  - Mouse button pressed
  - Mouse is moved
  - Key is pressed
  - Timers
  - …

Events

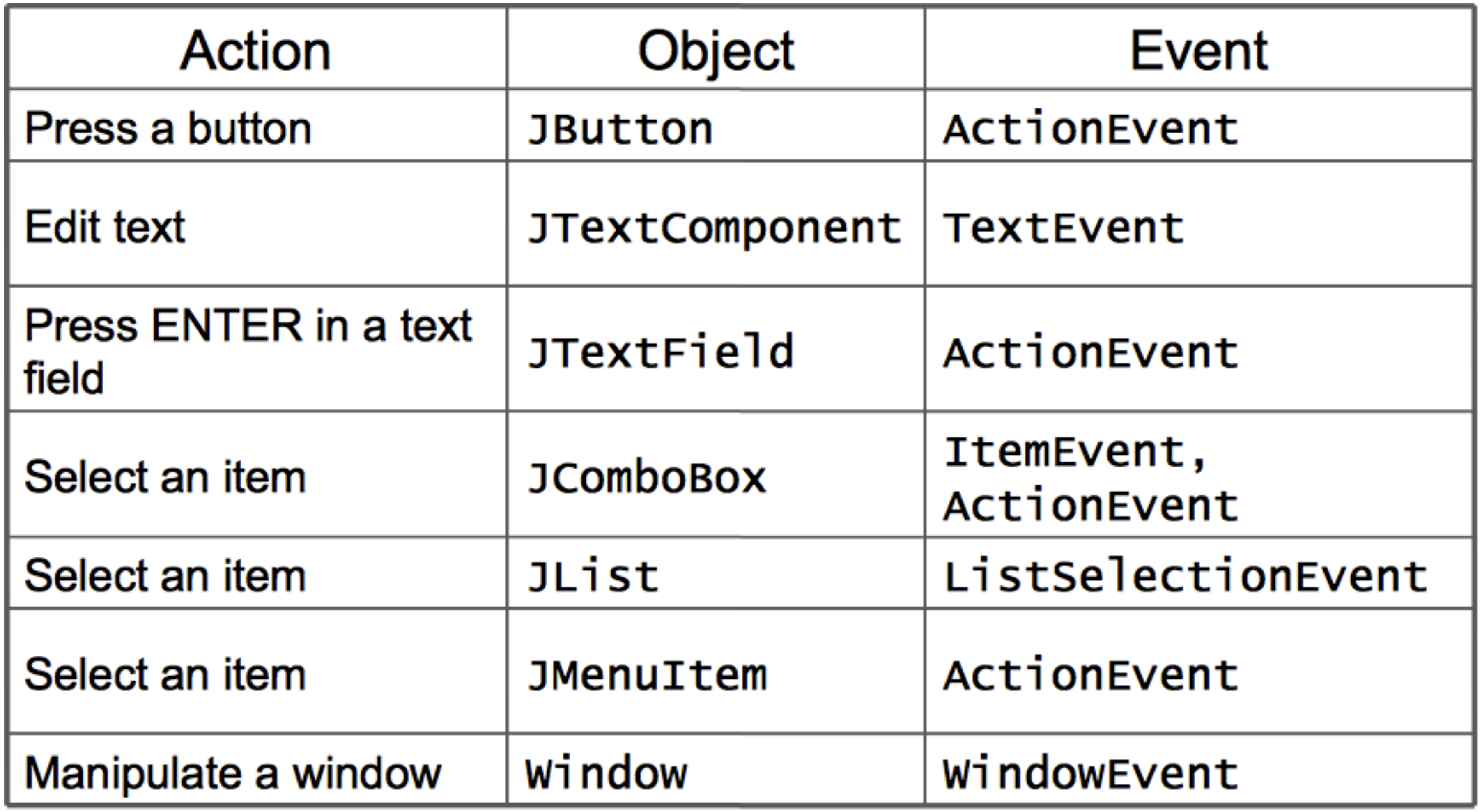
- A signal sent to a program
  - A program can choose to react or ignore
- All events inherit from

```
      EventObject
```

- Events have a source – the object that generated the event. The source can be determined by calling the following method on the event:

```
      Object getSource()
```

Actions, objects and events…

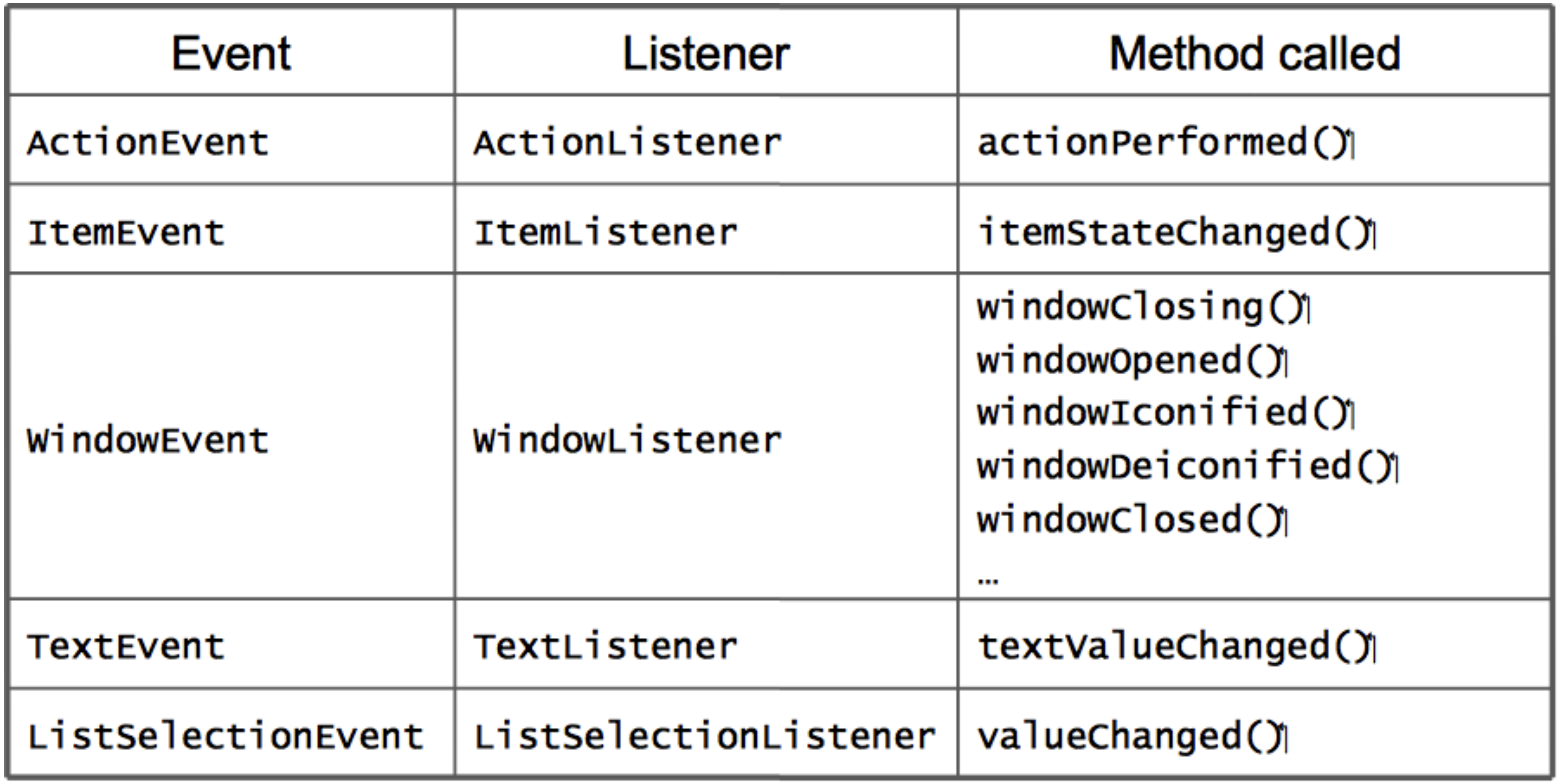


Action	Object	Event
Press a button	`JButton`	`ActionEvent`
Edit text	`JTextComponent`	`TextEvent`
Press ENTER in a text field	`JTextField`	`ActionEvent`
Select an item	`JComboBox`	`ItemEvent`, `ActionEvent`
Select an item	`JList`	`ListSelectionEvent`
Select an item	`JMenuItem`	`ActionEvent`
Manipulate a window	`Window`	`WindowEvent`

Subscription, listening and handling events

- An external action causes an event to be generated
  - Example: a button is pressed
- Objects that have subscribed to the event receive the event
  - We say that objects are *listening* for events
- Objects that can generate events maintain a list of event subscribers
  - Whenever an event occurs, the event generating object goes through the list of subscribers and calls a method on each subscriber that allows the subscriber to handle the event.

Events and listeners



Event	Listener	Method called
`ActionEvent`	`ActionListener`	`actionPerformed()`
`ItemEvent`	`ItemListener`	`itemStateChanged()`

`WindowEvent`	`WindowListener`	`windowClosing()`
		`windowOpened()`
		`windowIconified()`
		`windowDeiconified()`
		`windowClosed()`
		…

`TextEvent`	`TextListener`	`textValueChanged()`
`ListSelectionEvent`	`ListSelectionListener`	`valueChanged()`

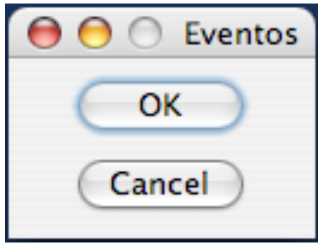
Libraries

- Events and Listeners are defined in the AWT library in the `event` package:
  - `import java.awt.event.ActionEvent;`
  - `import java.awt.event.ActionListener;`
  - ...
- Except:
  - `javax.swing.event.ListSelectionEvent`
  - `javax.swing.event.ListSelectionListener`

Defining a listener

```
public class ListnerForButtons implements ActionListener {

    private JButton okButton, cancelButton;
    public ListnerForButtons (JButton okButton,
                                JButton cancelButton){…}

    public void actionPerformed(ActionEvent e) {
        if(e.getSource() == okButton) {
            System.out.println(“Carregou no botão OK.”);
        }
        else if(e.getSource() == cancelButton) {
            System.out.println(“Carregou no botão Cancel.”);
        }
    }
}
```

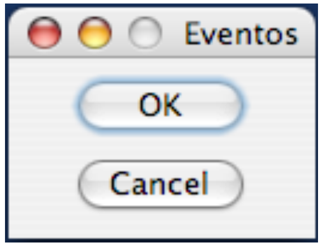
getSource() returns a reference to the object that sent the event.



Example

```
public class MyButton{

    private JFrame frame          = new JFrame("Eventos");
    private JButton okButton      = new JButton("OK");
    private Jbutton cancelButton = new JButton("Cancel");

    private ListnerForButtons listener =
         new ListnerForButtons(okButton, cancelButton);

    public void execute() {
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        MyButton b = new MyButton ();
        b.execute();
    }
```

(continues)



Subscribing to events

…

```
    public MyButton() {
        frame.setLayout(new FlowLayout());

        frame.add(botãoOK);
        frame.add(botãoCancel);

        // Regista sentinelas:
        okButton.addActionListener(listner);
        cancelButton.addActionListener(listner);

        frame.setSize(100, 100);
        frame.setLocation(200, 100);
        frame.setDefaultCloseOperation(
                    JFrame.EXIT_ON_CLOSE);
    }
}
```

JComponent

- The base class for all Swing components except top-level containers.
- Provides:
  - Tool tips
  - Painting and borders
  - Application-wide pluggable look and feel
  - Custom properties
  - Support for layout
  - Support for accessibility
  - Support for drag and drop
  - Key bindings

JComponent

- Painting
  - `public void repaint()`
  - `public void paintComponent(Graphics g)`

- Changing the component's structure
  - `public void revalidate()`

JComponent

```
public class MyLabel extends JComponent {
    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        g.drawString("Hello", 50, 50);
        g.drawLine(0, 0, getWidth(), getHeight());
    }
}

public class TesteLabel {
    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.add(new MyLabel());
        frame.setSize(200, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

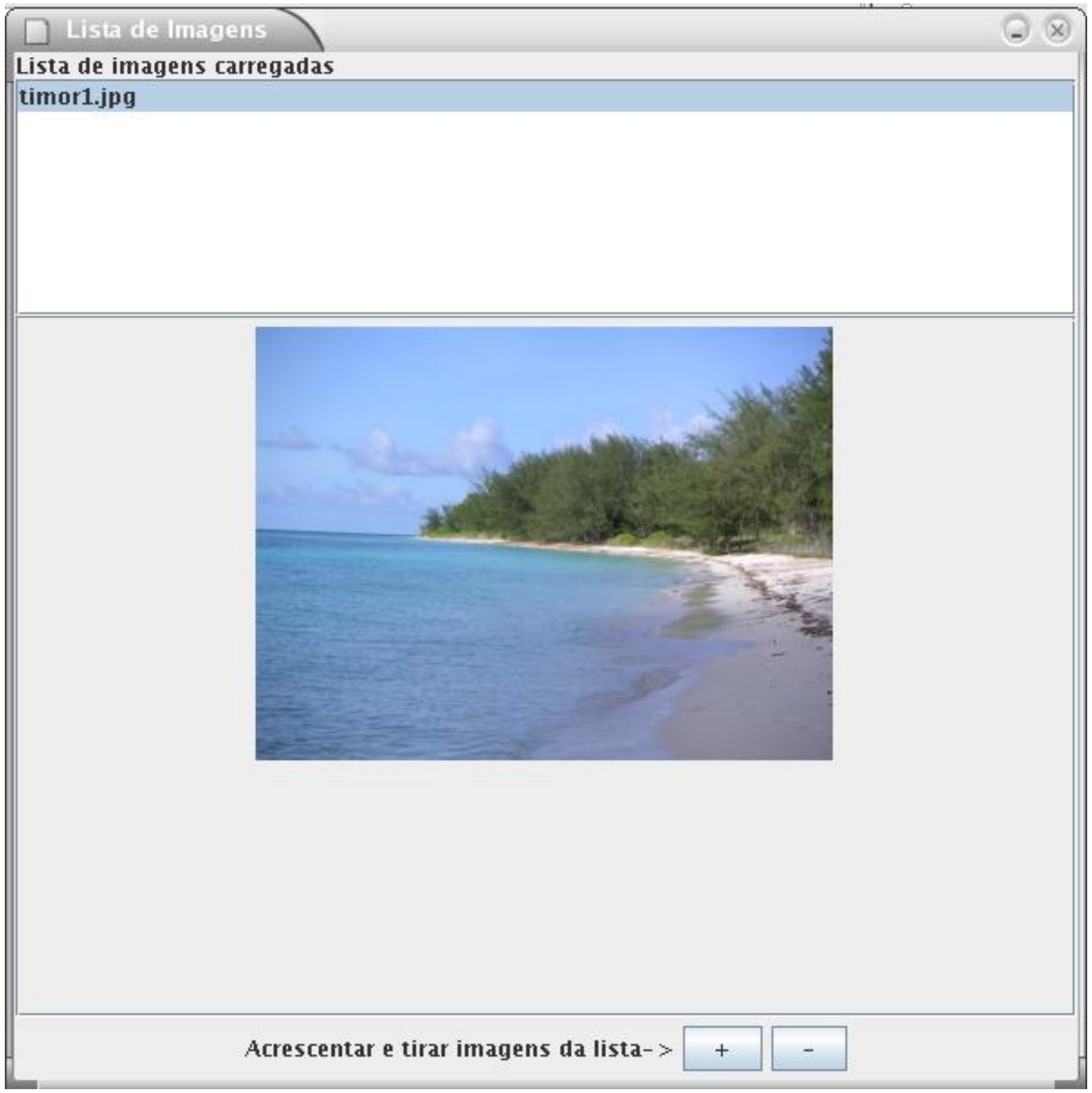
Notes

- For more on creating user interfaces with Swing, see:
  - **http://docs.oracle.com/javase/tutorial/uiswing/index.html**
- There are more than 250 classes…:
  - It is impossible to remember all of them, so it is essential that you understand the basic principles and then learn to find the specific information you need in order to create the GUI.

Summary

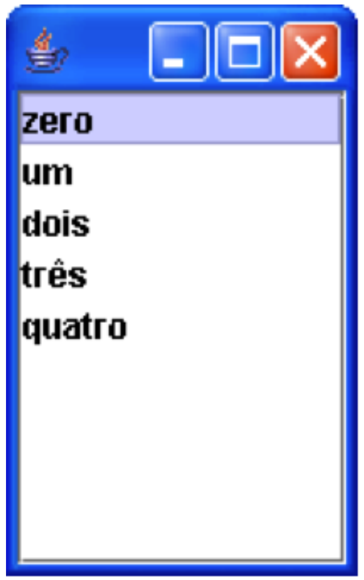
- GUIs
- Event-based programming
    - Events
    - Actions
    - Source objects
    - Different types of events
    - Subscription, listening and handling events
        - Listeners
        - Methods for handing events
        - Examples

Exercício - Album de Fotografias


Lista de Imagens
Lista de imagens carregadas
timor1.jpg
Acrescentar e tirar imagens da lista- >
+
-

An example with lists...

- `javax.swing.JList`
- `javax.swing.JScrollPane`
- `javax.swing.ListSelectionModel`
- `javax.swing.event.ListSelectionEvent`
- `javax.swing.event.ListSelectionListener`



List, items and the listener

```
public class Listador {

      private static final String[] nomesDoItens = {
            "zero",
            "um",
            "dois",
            "três",
            "quatro"
      };


      private int índiceDoItemSeleccionado = 0;
      private JFrame janela = new JFrame("Listas");

      private JList lista = new JList(nomesDoItens);
      private SentilenaParaALista sentinela =
                 new SentilenaParaALista();
```

(continues)

Adding the list and setting up the window

```
public Listador() {
        janela.getContentPane().add(new JScrollPane(lista));
        lista.setSelectionMode(
                ListSelectionModel.SINGLE_SELECTION);
        lista.setSelectedIndex(0);
        lista.addListSelectionListener(sentinela);


  janela.setSize(100, 200);
  janela.setLocation(200, 100);
  janela.setDefaultCloseOperation(
                              JFrame.EXIT_ON_CLOSE);
}


public void executa() {
        janela.setVisible(true);
}
```

(continues)

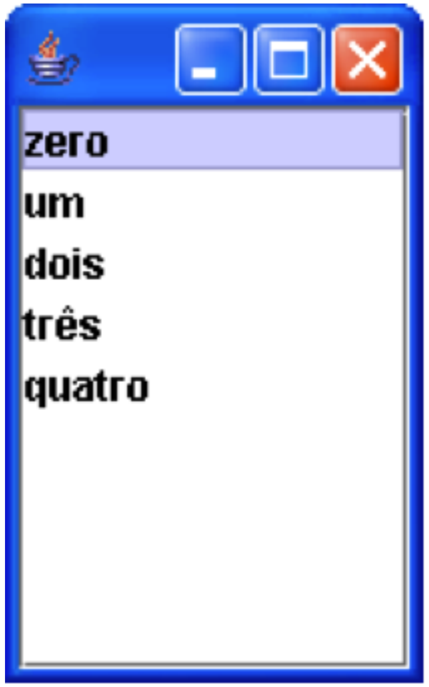
Handling events

(continued)

```
    private class SentilenaParaALista implements
                                    ListSelectionListener {

        public void valueChanged(ListSelectionEvent e) {
            if(índiceDoItemSeleccionado !=
                              lista.getSelectedIndex()) {

                System.out.println(lista.getSelectedIndex()
                                     + " --> "
                                     + lista.getSelectedValue());

                índiceDoItemSeleccionado =
                                   lista.getSelectedIndex();
            }
        }
    }
```

(continues)

```
(continuation)

    public static void main(String[] args) {
        Listador l = new Listador();
        l.executa();
    }
}
```



```
zero
um
dois
três
quatro
```