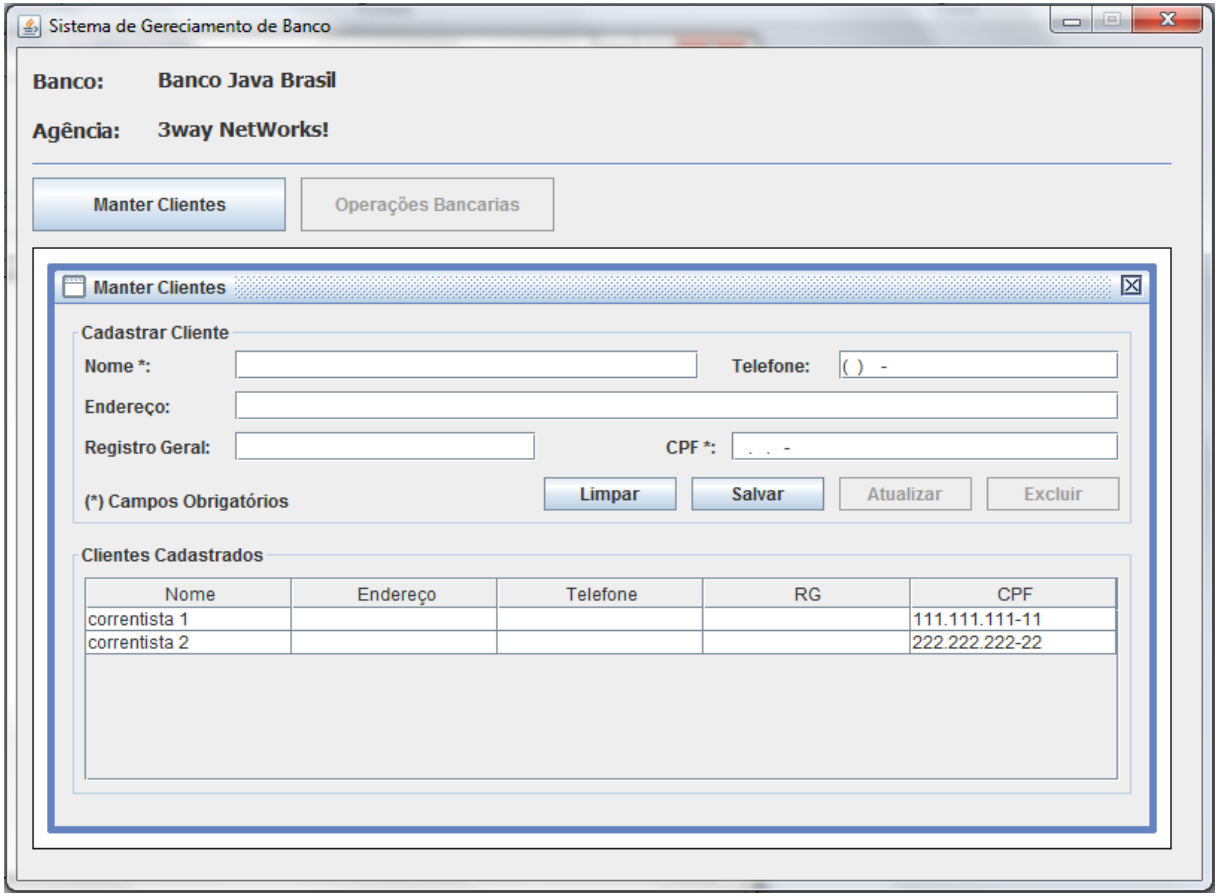
Java Orientado a Objetos
Construindo interfaces gráficas com Swing



AWT versus Swing

AWT

Código nativo

São dependente de plataforma

Assegura que a aparência de uma aplicação executada em diferentes máquinas seja comparável, mutável baseado no SO

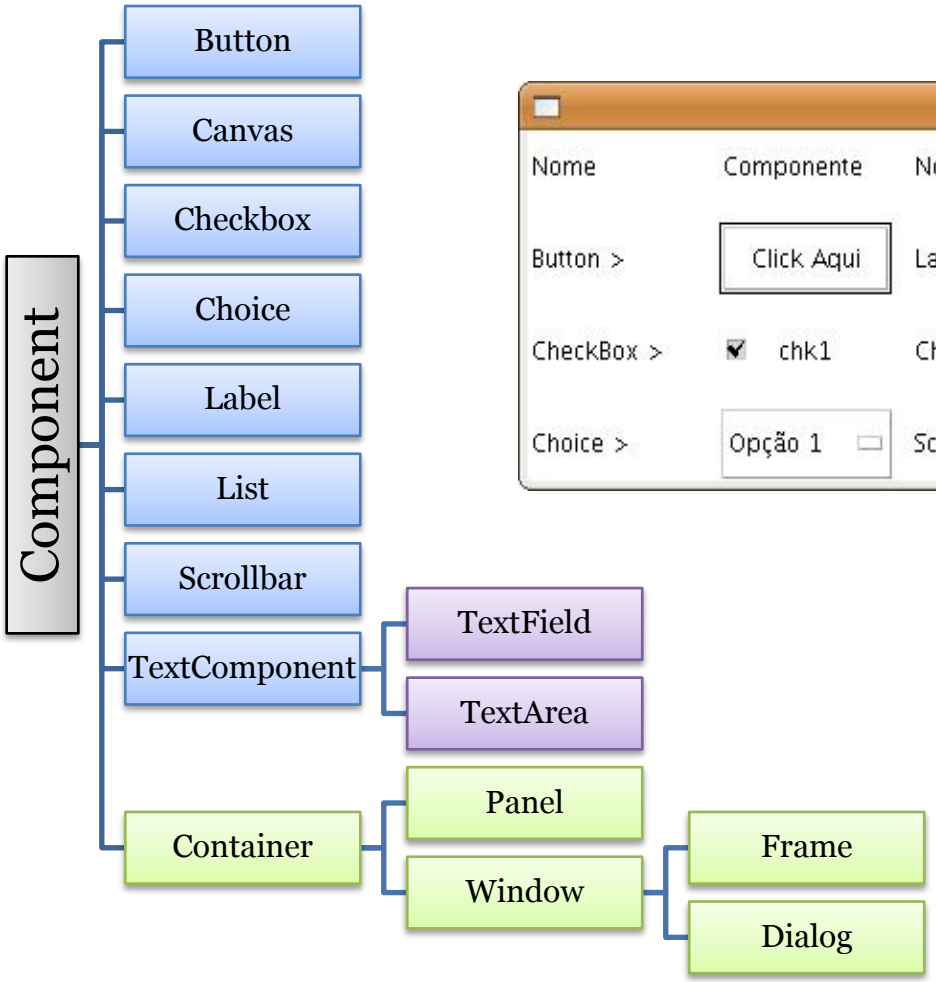
Swing

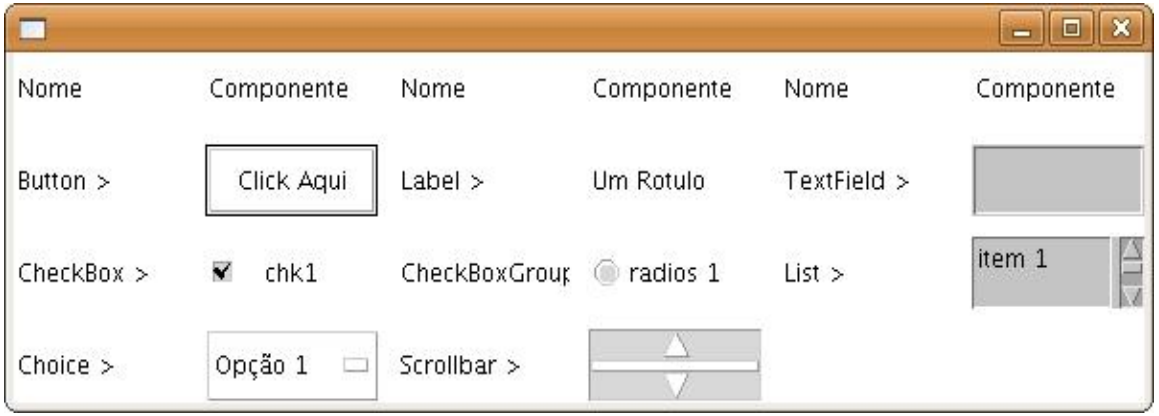
Escrito em Java

Independente de plataforma

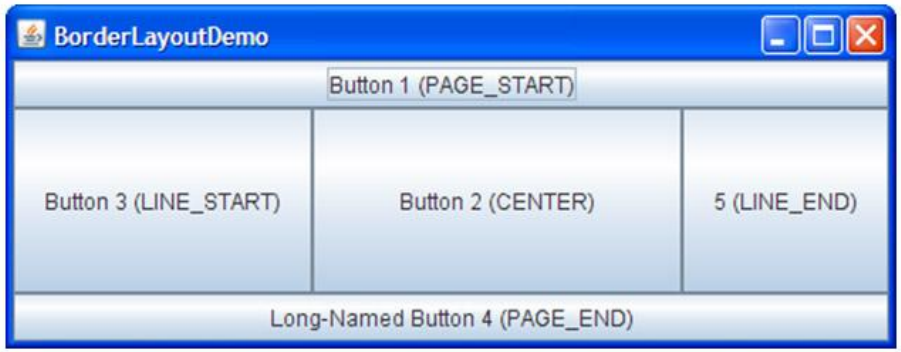
Mesma aparência em plataformas diferentes

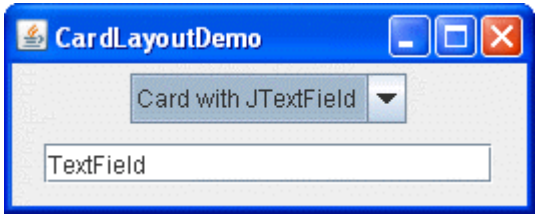
Componentes AWT

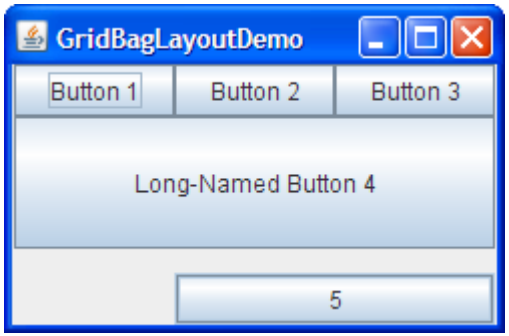


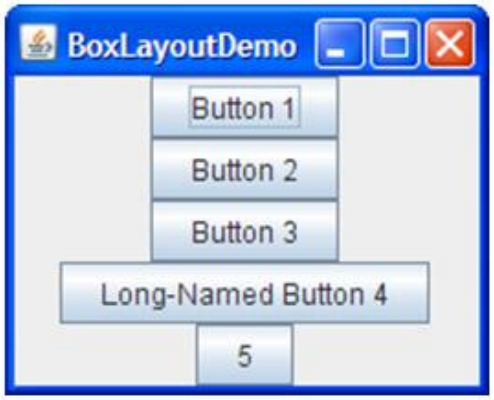


Gerenciadores de layout

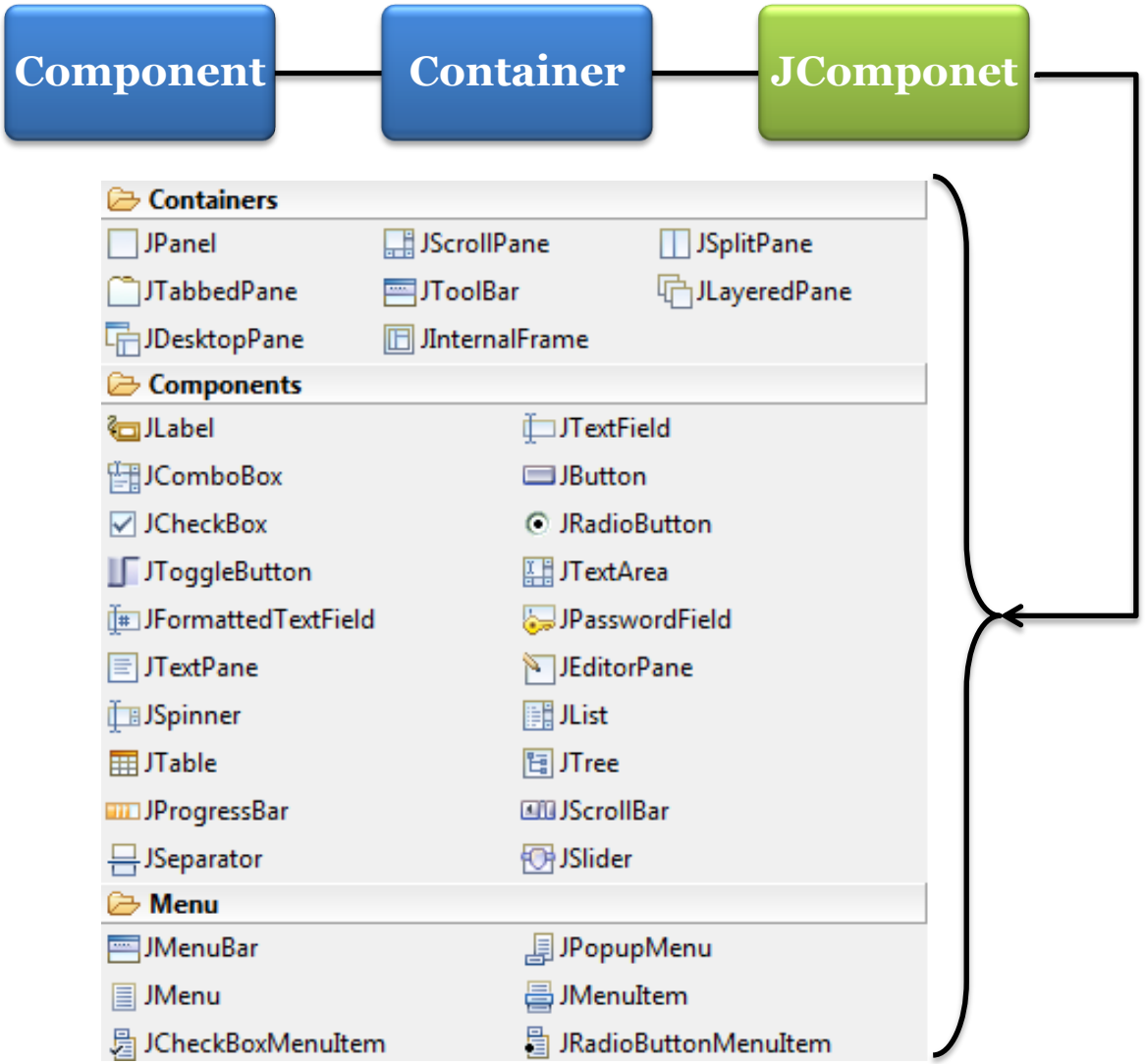








Componentes Swing



Containers JFrame

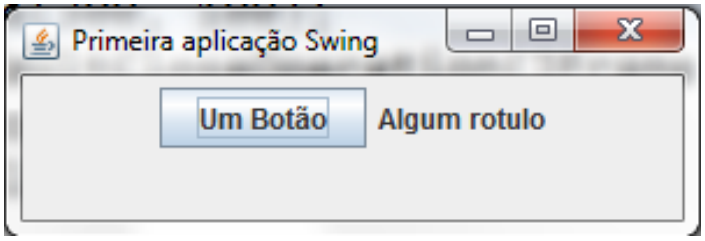
```
public class AppSwing extends JFrame {
    JButton botao;
    JLabel label;
    public AppSwing() {
        super("Primeira aplicação Swing");
        setSize(300, 100);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new FlowLayout());
        initialize();
    }
    private void initialize() {
        botao = new JButton("Um Botão");
        label = new JLabel("Algum rotulo");
        getContentPane().add(botao);
        getContentPane().add(label);
    }
    public static void main(String[] args) {
        AppSwing app = new AppSwing();
        app.setVisible(Boolean.TRUE);
    }
}
```

Um container é um agrupamento ou uma coleção de JComponents

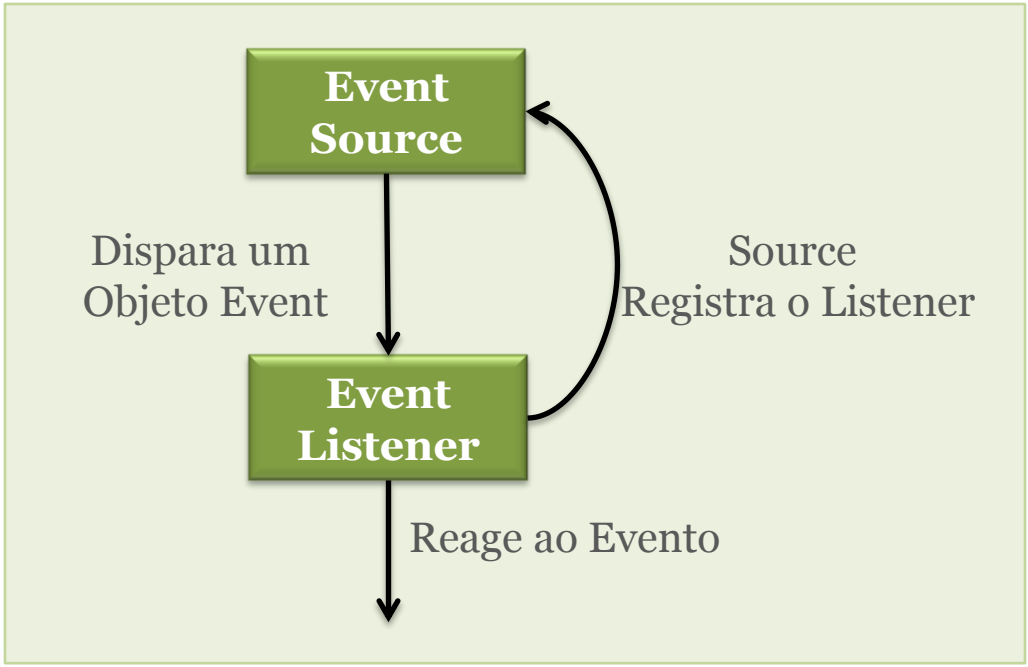
Construtor

Container que recebe os componentes

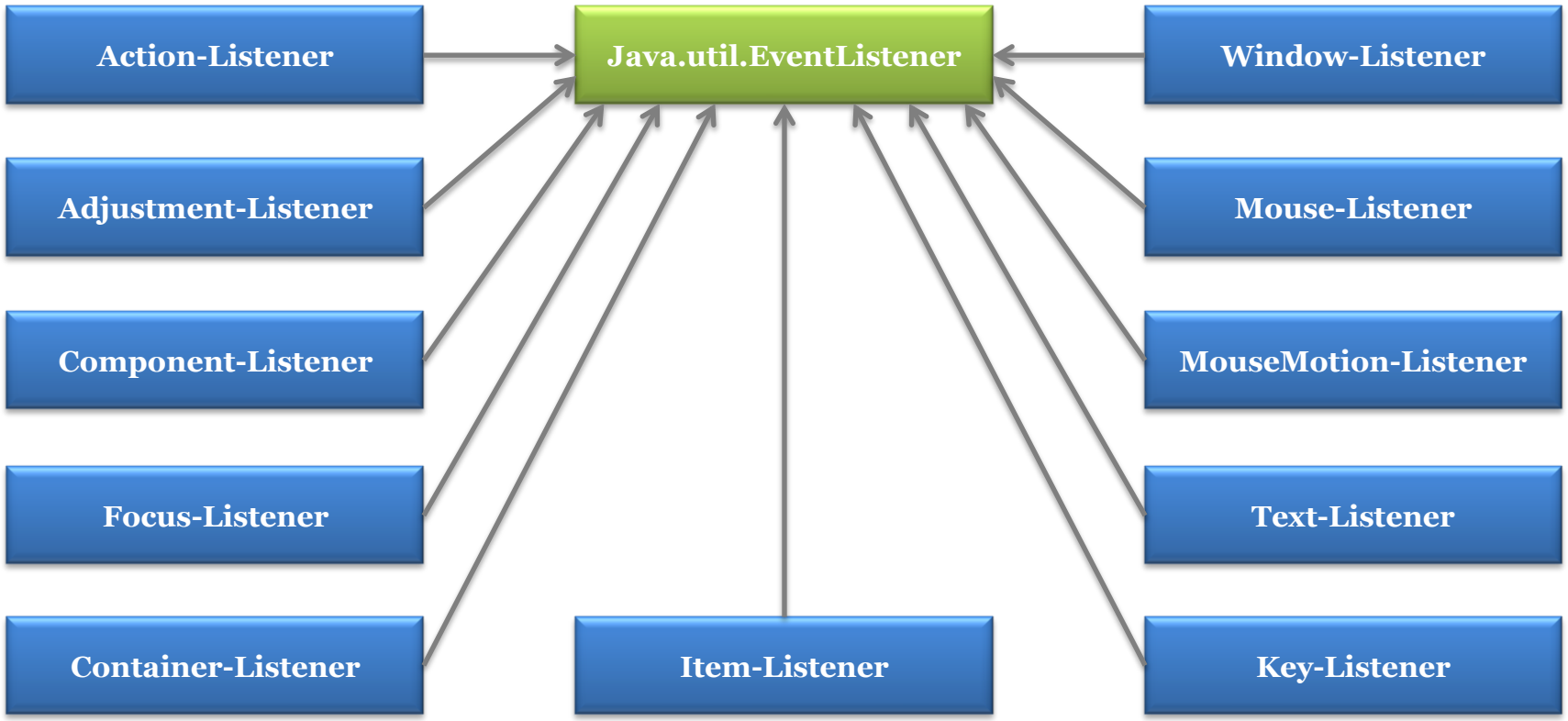
Inicia o frame e o deixa visível



Manipulação de Evento



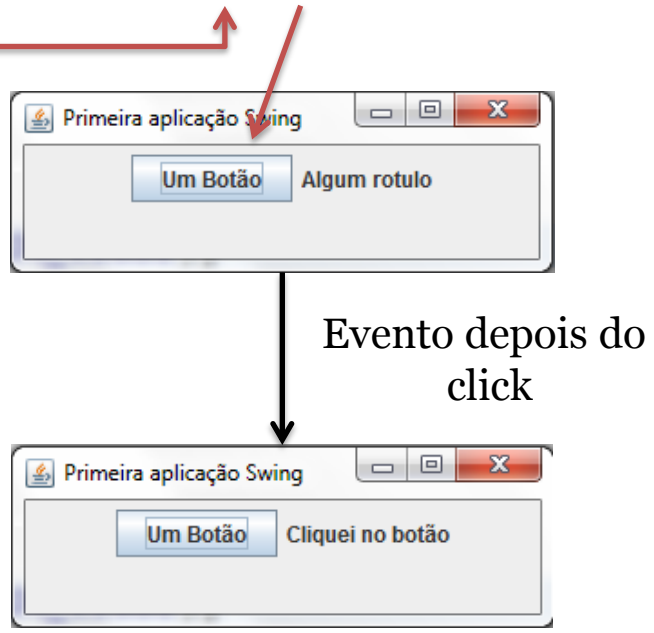
Classes de Evento



Criação de aplicações gráficas com Eventos

```
public class AppSwing extends JFrame {
    JButton botao;
    JLabel label;
    public AppSwing() {
        super("Primeira aplicação Swing");
        setSize(300, 100);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new FlowLayout());
        initialize();
    }
    private void initialize() {
        botao = new JButton("Um Botão");
        botao.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                label.setText("Cliquei no botão");
            }
        });
        label = new JLabel("Algum rotulo");
        getContentPane().add(botao);
        getContentPane().add(label);
    }
    public static void main(String[] args) {
        AppSwing app = new AppSwing();
        app.setVisible(Boolean.TRUE);
    }
}
```

Declarando ação do botão



Classes Adaptadoras

Com a utilização de Classes Adaptadoras a classe que implementa o manipulador de um evento apenas herda da classe adaptadora e sobrescreve os métodos que precisar.

```
public class AcaoTecla extends KeyAdapter {

    public void keyPressed(KeyEvent e) {

        System.out.println("Tecla pressionada");
    }

    // Não é necessário declarar os outros métodos da
    // interface KeyListener (keyReleased e keyTyped)
}
```