UFRPE

Eventos

Richarlyson A. D'Emery
site: https://sites.google.com/site/profricodemery/mpoo
grupo: http://groups.google.com/group/**mpoo_uast**
email grupo: **mpoo_uast**@googlegroups.com
contato: rico_demery@yahoo.com.br

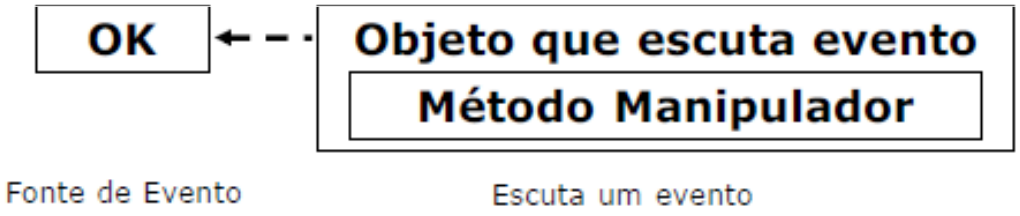
Sumário

- Eventos
- Listener

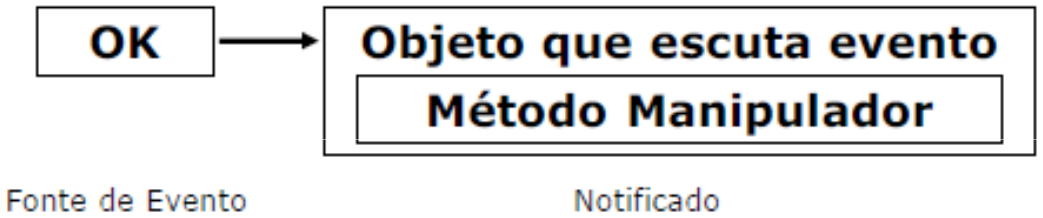
Eventos em Java

UFRPE

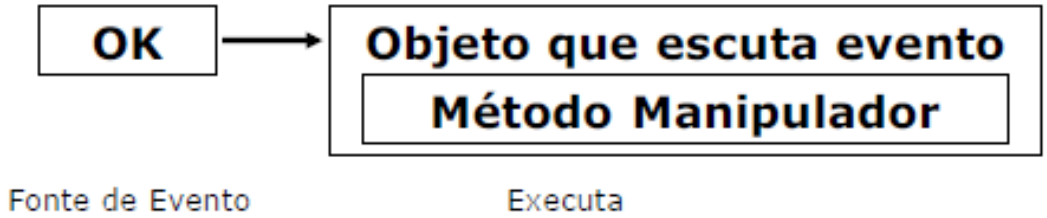
- 1º) Registra-se um *listener*



- 2º) Um evento acontece



- 3º) O método manipulador é executado



Eventos em Java

UFRPE

- Cada tipo de evento tem uma interface correspondente:

```
public interface ActionListener{
    void actionPerformed(ActionEvent e){
      //ação a ser executada
    }
}
```

- *listeners* devem implementar sua(s) interface(s):

```
public class ButtonHandler implements ActionListener{
    void actionPerformed(ActionEvent e){...}
}
```

Eventos em Java

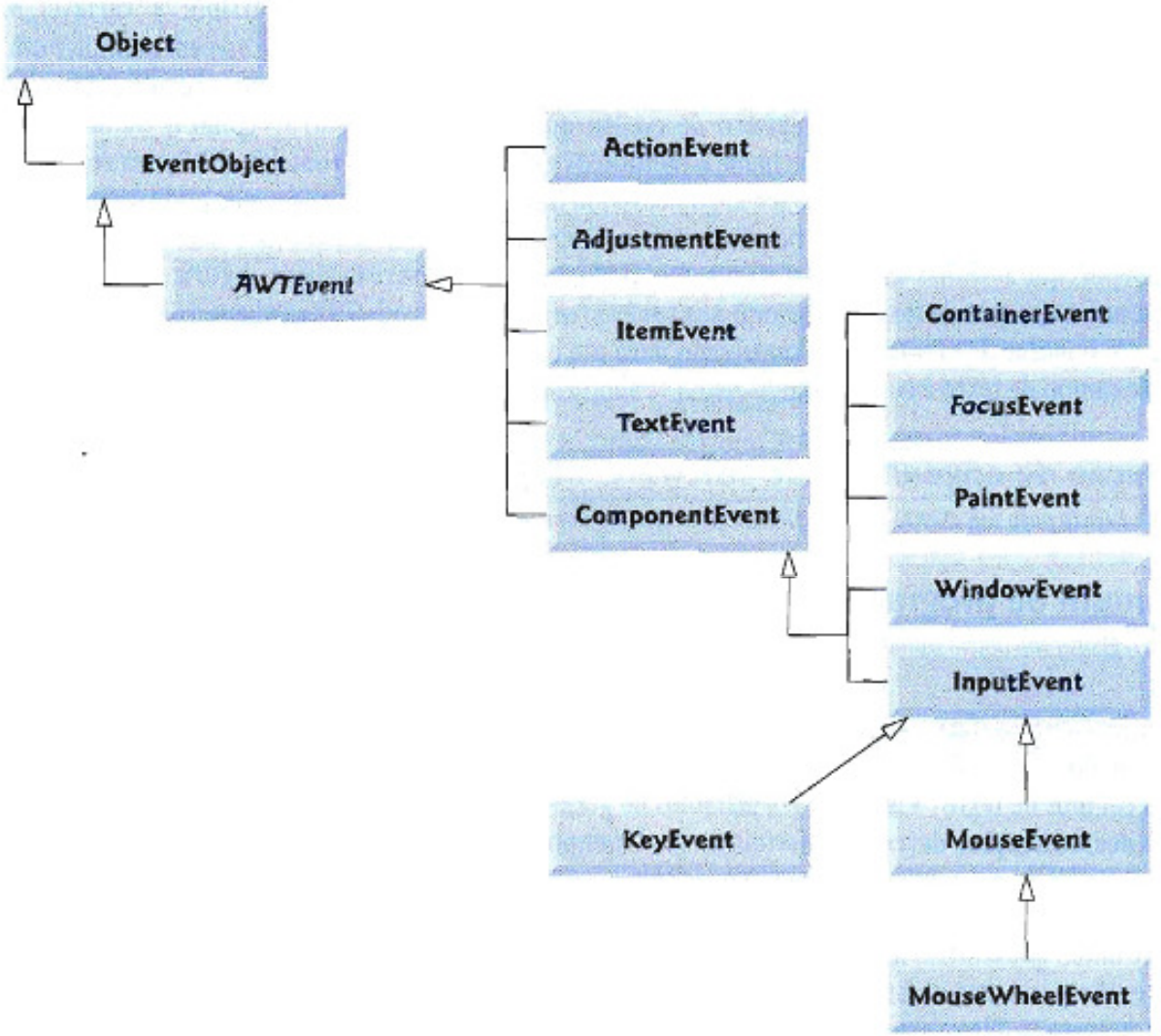
UFRPE

- Pacote `java.awt.event`
  - O Java Abstract Window Toolkit Event Package contém classes e interfaces que permitem o tratamento de eventos para componentes GUI tanto nos pacotes `java.awt` como `javax.swing`
- Pacote `javax.swing.event`
  - O Java Swing Event Package contém classes e interfaces que permitem o tratamento de eventos (por exemplo, responder a cliques de botão) para componentes GUI do pacote `javax.swing`

Eventos em Java

UFRPE

- Algumas Classe de `java.awt.event`



Exemplo: ActionListener e ActionEvent – Classe Interna


```
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JButton;
import javax.swing.JFrame;

public class ExemploEventoJButton extends JFrame {
    JButton botao;
    public ExemploEventoJButton(){
            super (Evento);
            botao = new JButton("botão");
            add(botao, BorderLayout.CENTER);

            //Registra o listener - caso alguma ação seja realizada
            botao.addActionListener(new ButtonHandler());

            setSize(200,200);
            setVisible(true);
            setLocationRelativeTo(null);
            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

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

    public class ButtonHandler implements ActionListener{ //classe interna
            public void actionPerformed(ActionEvent e) {
                        System.out.println("Botão Pressionado!");
            }
    }
}
```

Classe Interna Anônima

UFRPE

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class ExemploClasseInternaAnonima extends JFrame{
     JButton botao1, botao2;

     public ExemploClasseInternaAnonima() {
             super ("Evento");
             setLayout (new FlowLayout());

             botao1 = new JButton("botão1");
             botao1.addActionListener(
                  new ActionListener(){ //classe interna anônima
                         public void actionPerformed(ActionEvent e) {
                              System.out.println("Botão Pressionado!");
                         }
                  });
             botao2 = new JButton("botão2");
             botao2.addActionListener(
                   new ActionListener(){ //classe interna anônima
                         public void actionPerformed(ActionEvent e) {
                               JOptionPane.showMessageDialog(null, "Botão2 pressionado");
                         }
                   });

             add(botao1); add(botao2);
             setSize(100,120); setVisible(true); setLocationRelativeTo(null);
             setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     }

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

Métodos Manipuladores na Classe


```
import java.awt.*;
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class ExemploMetodoManipuladornaClasse extends JFrame implements ActionListener{
     JButton botao1, botao2;

     public ExemploMetodoManipuladornaClasse() {
          super ("Evento");
          setLayout (new FlowLayout());

          botao1 = new JButton("botão1");
          botao1.addActionListener(this);
          botao2 = new JButton("botão2");
          botao2.addActionListener(this);

          add(botao1);add(botao2);

          setSize(100,120);
          setVisible(true);
          setLocationRelativeTo(null);
          setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     }

     public void actionPerformed(ActionEvent evento) {//Método manipulador pertence a classe
          if (evento.getSource()==botao1)
            System.out.println("Botão1 Pressionado!");
          if (evento.getSource()==botao2)
            JOptionPane.showMessageDialog(null, "Botão2 pressionado");
     }

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

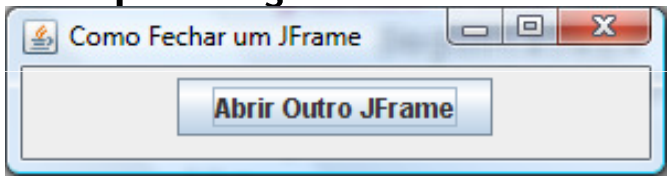
Exercícios

1) Implemente uma calculadora que possua, pelo menos, as quatro operações.

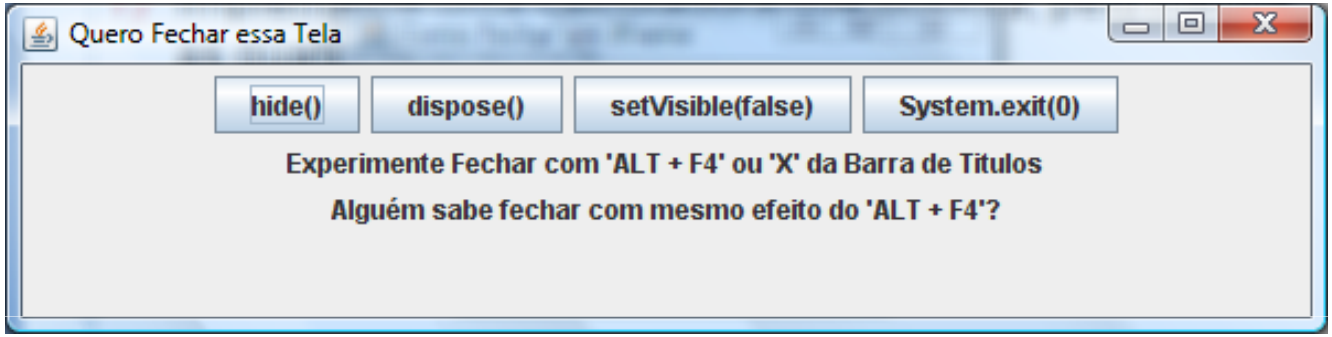
Exercícios

UFRPE

2) Implemente a aplicação Java:



Ao clicar no botão “Abrir Outro JFrame”, irá carregar outro JFrame:



Os botões:

- hide() - dispara o método hide() fechando o segundo JFrame
- dispose () – dispara o método dispose() fechando o segundo JFrame
- setVisible(false) – dispara o método setVisible() fechando o segundo JFrame
- System.exit(0) – encerra a aplicação

Exercícios

3) Qual a diferença entre os métodos:

hide( ), dispose ( ), setVisible( ) e System.exit( )

UFRPE

FIM

Prof. Richarlyson D'Emery

site: https://sites.google.com/site/profricodemery/mpoo

grupo: http://groups.google.com/group/**mpoo_uast**

email grupo: **mpoo_uast**@googlegroups.com

contato: rico_demery@yahoo.com.br