ALPOO – Aplicações de Linguagem de Programação Orientada a Objetos

Prof. Ms. Gustavo Molina

msc.gustavo.unip@gmail.com

Aula 04 – Swing Parte III

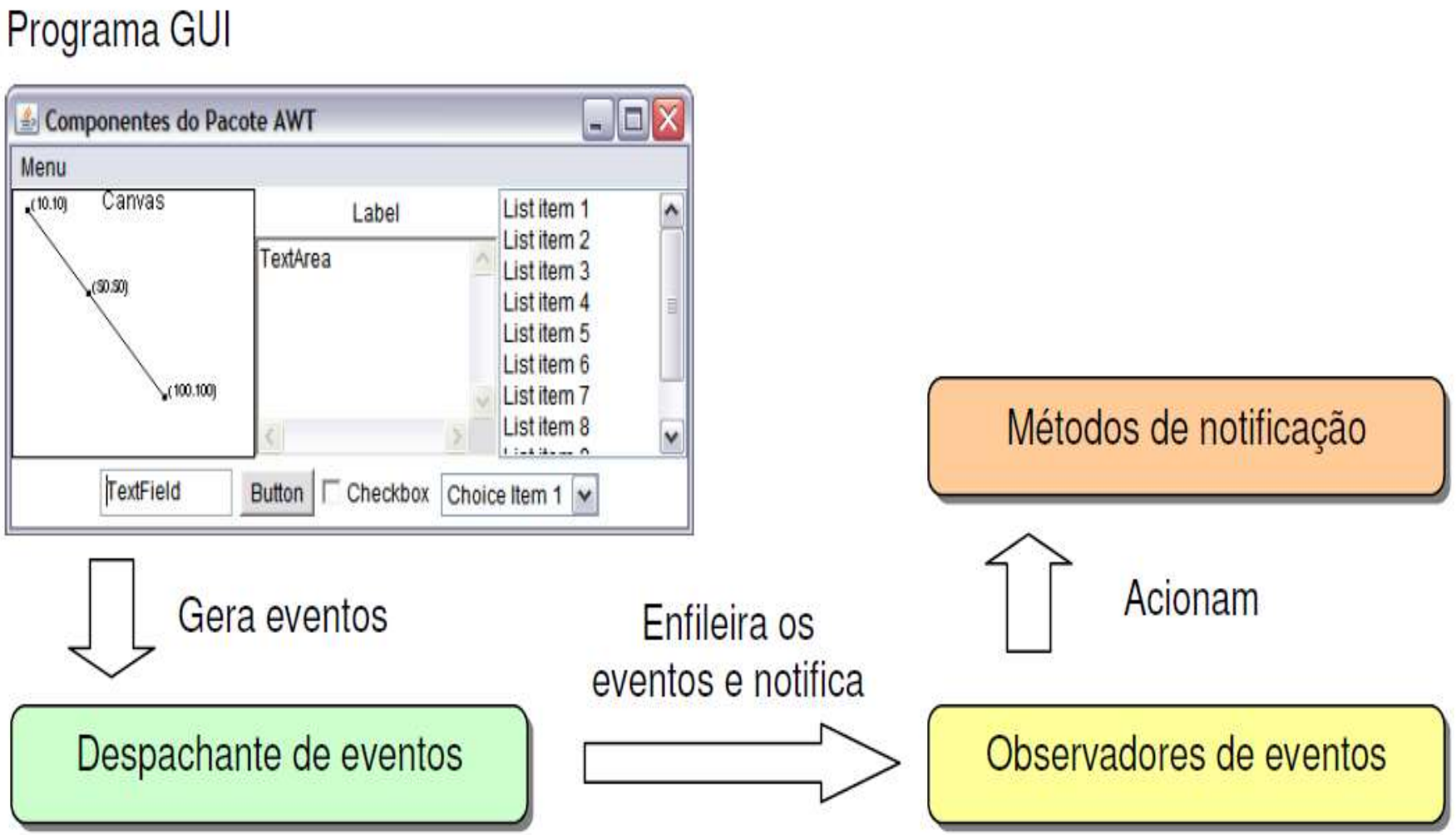
Eventos

- Programas com interface gráfica de usuário (GUI) são controlados por eventos.
- O programa reage as ações do usuário. Estas ações são chamadas de eventos.

Exemplos:

- ✓ Pressionar de uma tecla do teclado
- ✓ Mover o mouse
- ✓ Pressionar um botão
- ✓ Selecionar um item de menu
- ✓ Fechar uma tela

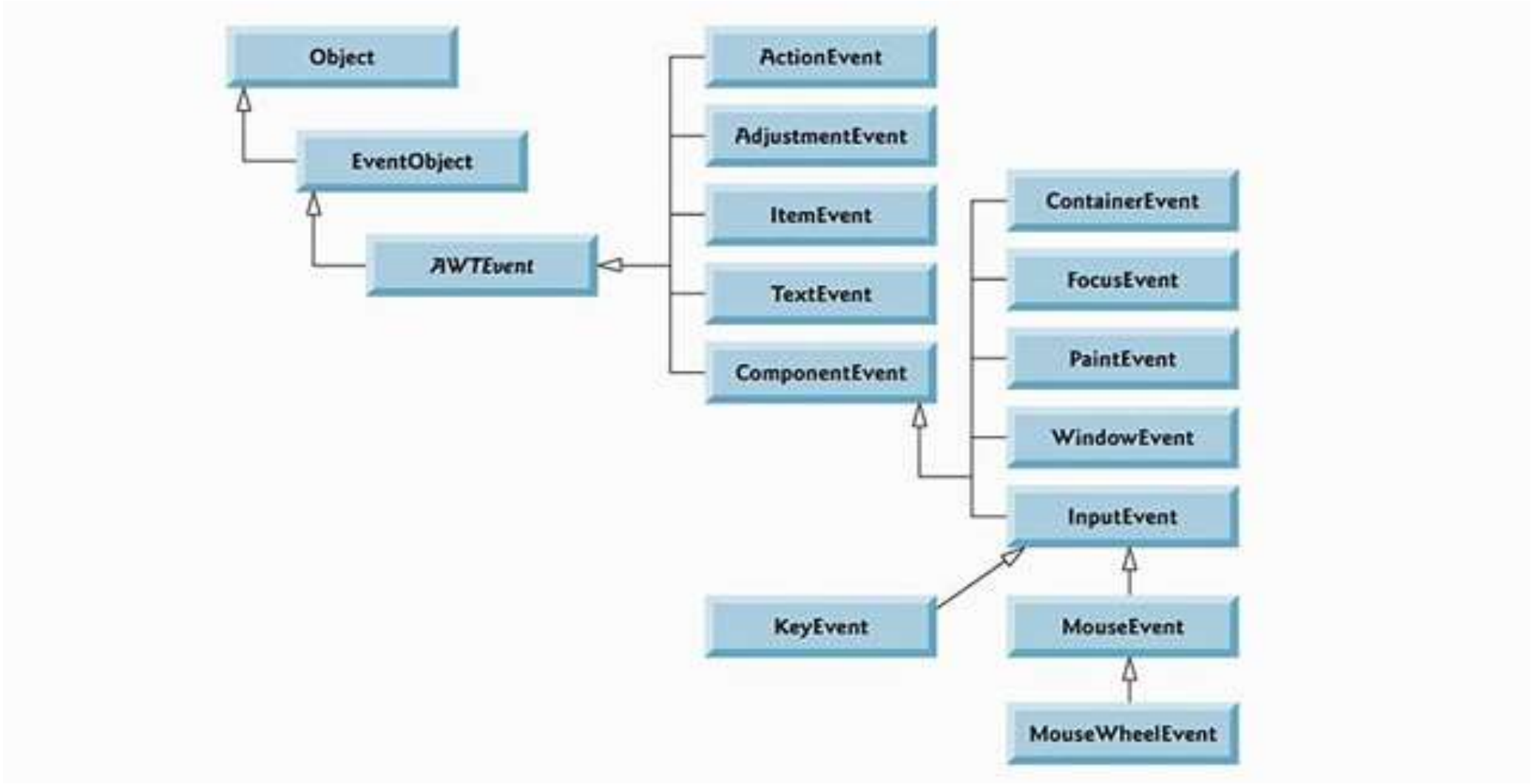
Eventos


Programa GUI
Componentes do Pacote AWT
Menu
Canvas
(10,10)
(50,50)
(100,100)
Label
TextArea
List item 1
List item 2
List item 3
List item 4
List item 5
List item 6
List item 7
List item 8
TextField
Button
Checkbox
Choice Item 1
Gera eventos
Despachante de eventos
Enfileira os eventos e notifica
Observadores de eventos
Acionam
Métodos de notificação

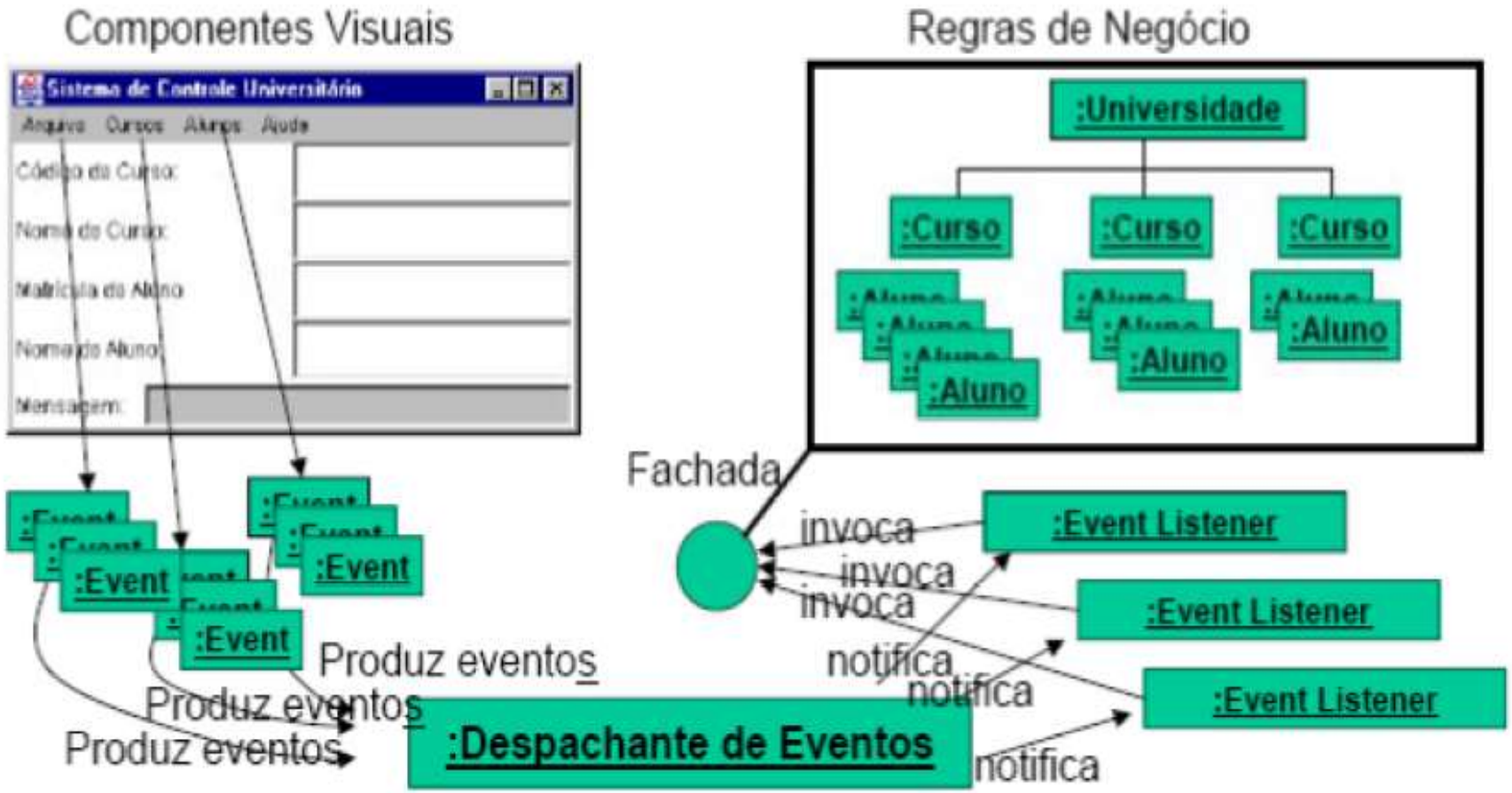
Eventos

- As tarefas de respostas realizadas em um evento são conhecidas como ***handler*** de eventos.
- A resposta aos eventos é conhecida como tratamento de evento.
- Para cada tipo de evento precisa ser implementada uma interface de escuta conhecida como ***listener*** (ouvinte).

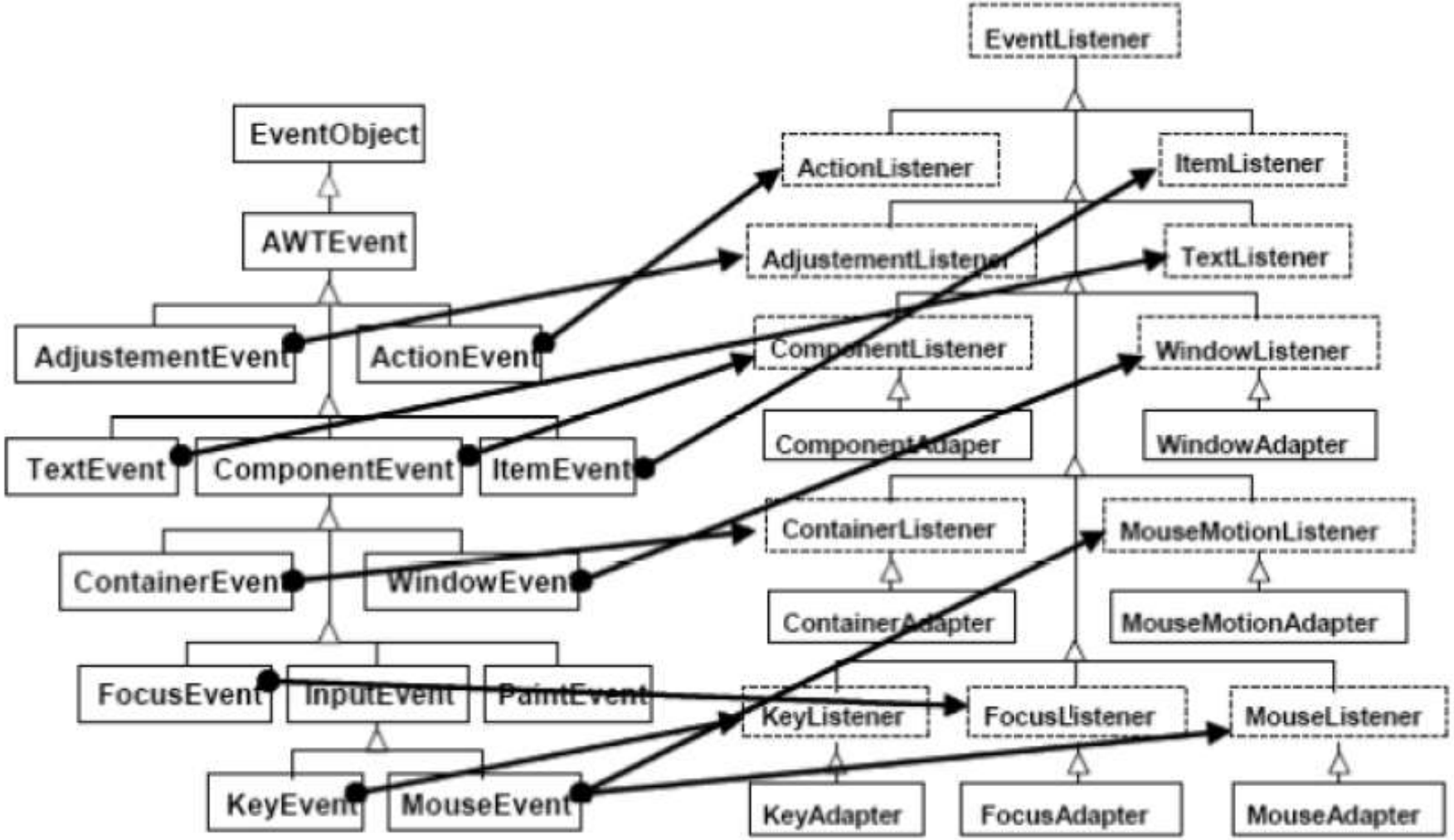
Eventos


Object
EventObject
AWTEvent
ActionEvent
AdjustmentEvent
ItemEvent
TextEvent
ComponentEvent
ContainerEvent
FocusEvent
PaintEvent
WindowEvent
InputEvent
KeyEvent
MouseEvent
MouseWheelEvent

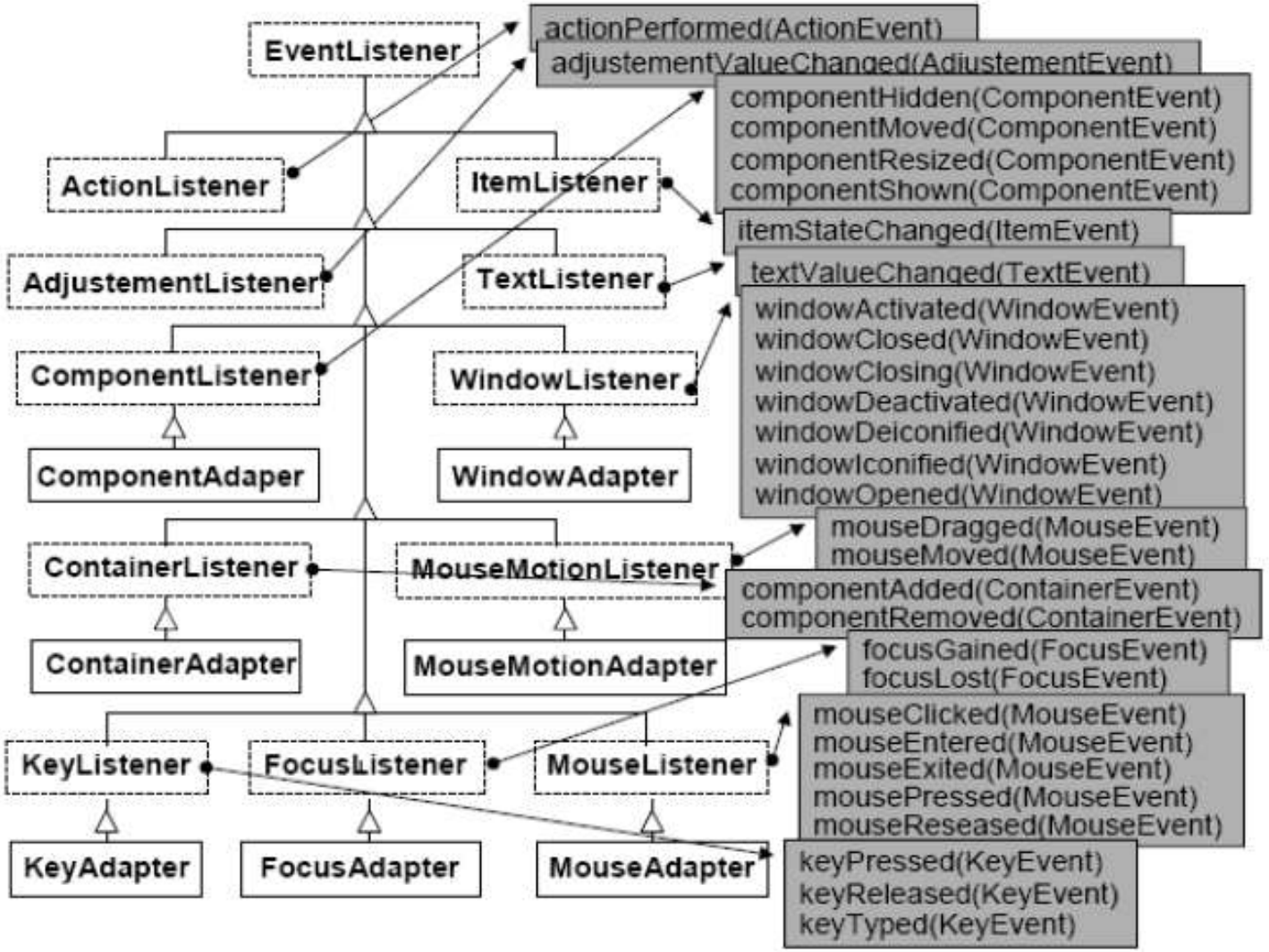
Arquitetura de um programa com interface gráfica



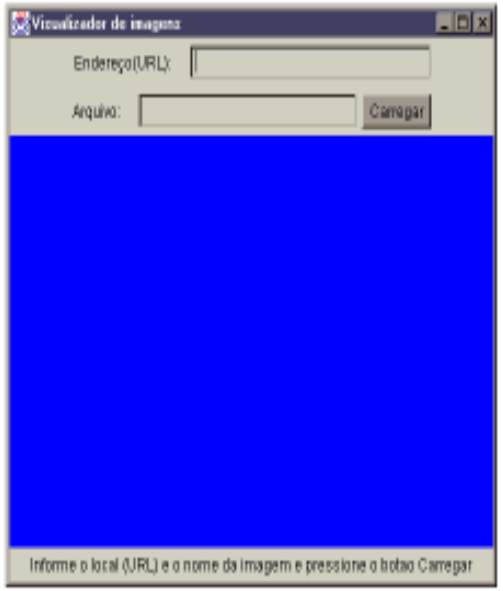
Eventos e seus observadores (listeners)



Observadores e seus métodos de notificação



Principais tipos de eventos



Labels

Panel

TextFields

Frame

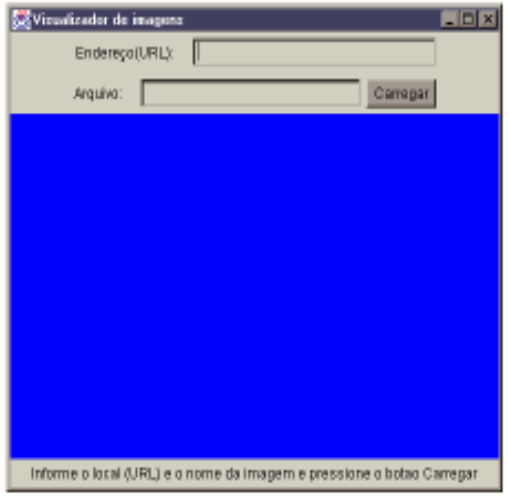
Button

Labels não utilizam muitos eventos.

Um dos únicos eventos que faz sentido deles gerar é o evento que avisa que o mouse está passando por cima deles.

Em Java, cada componente de interface tem seu conjunto específico de eventos.

Principais tipos de eventos



Labels

Panel

TextFields

Frame

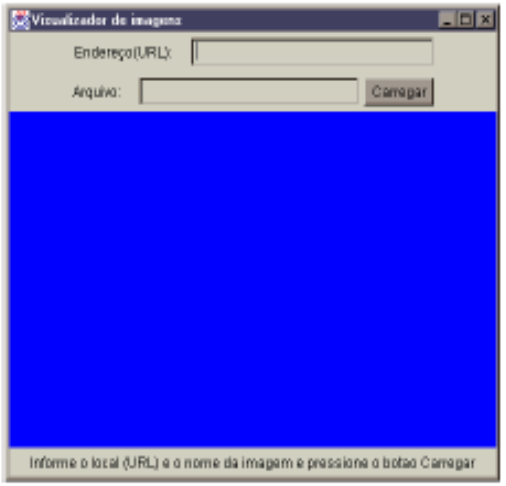
Button

Em Java, cada componente de interface tem seu conjunto específico de eventos.

Com Panels outros eventos mais interessantes podem ser capturados:

- **Movimento do mouse;**
- **Ação de mouse;**
- **Teclado;**
- **Re-pintura/atualização;**

Principais tipos de eventos



Labels

Panel

TextFields

Frame

Button

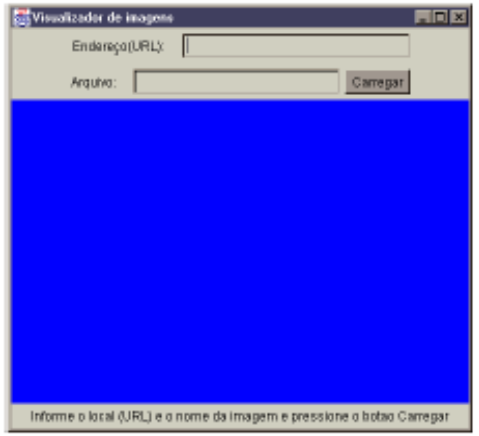
Em Java, cada componente de interface tem seu conjunto específico de eventos.

Como os TextFields são caixas de texto que coletam informações do usuário, faz sentido capturar deles os eventos de:

- **Teclado;**
- **Alteração de seu conteúdo;**

(ou até mesmo os eventos de movimento ou ação do mouse, em alguns casos)

Principais tipos de eventos



Labels

Panel

TextFields

Frame

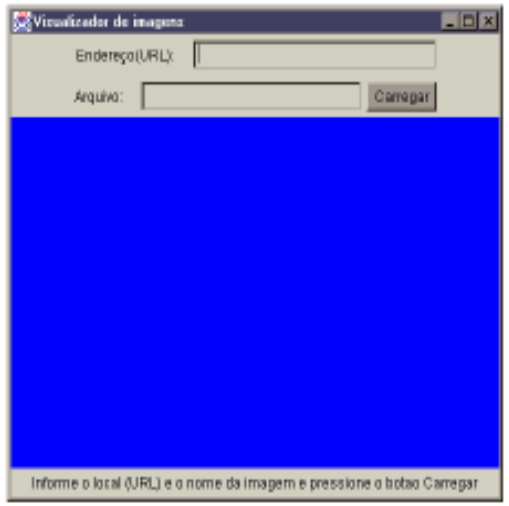
Button

Em Java, cada componente de interface tem seu conjunto específico de eventos.

Os Frames são parecidos com os Painéis. Na verdade eles são painéis com bordas! Logo, além dos eventos de um painel, eles também geram eventos de janelas:

- **Movimento do mouse;**
- **Ação de mouse;**
- **Teclado;**
- **Re-pintura/atualização;**
- **Manipulação de janela;**

Principais tipos de eventos



Labels

Panel

TextFields

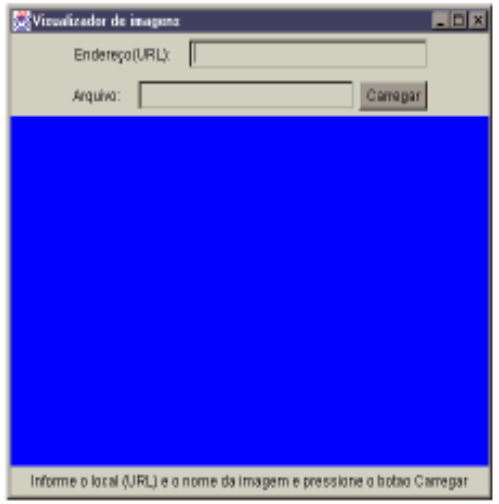
Frame

Button

Em Java, cada componente de interface tem seu conjunto específico de eventos.

Já os Buttons (botões), geram eventos de ação (pressionado, solto...) e de mouse (clicado, passando por cima...)

Principais tipos de eventos



Labels

Panel

TextFields

Frame

Button

Logo, existem diferentes classes de eventos e tratadores de eventos, que devem ser utilizadas de acordo com os componentes e as necessidades que o programador possui.

Em Java, cada componente de interface tem seu conjunto específico de eventos.

Principais tipos de eventos

Eventos de janela (WindowEvents)

Gerados quando uma janela é aberta, fechada, maximizada ou minimizada, entre outros.

Eventos de ações ocorridas em componentes (ActionEvents)

Gerados quando um componente sofre uma ação gerada pelo usuário (seleção de um elemento ou clique do mouse em um botão, por exemplo).

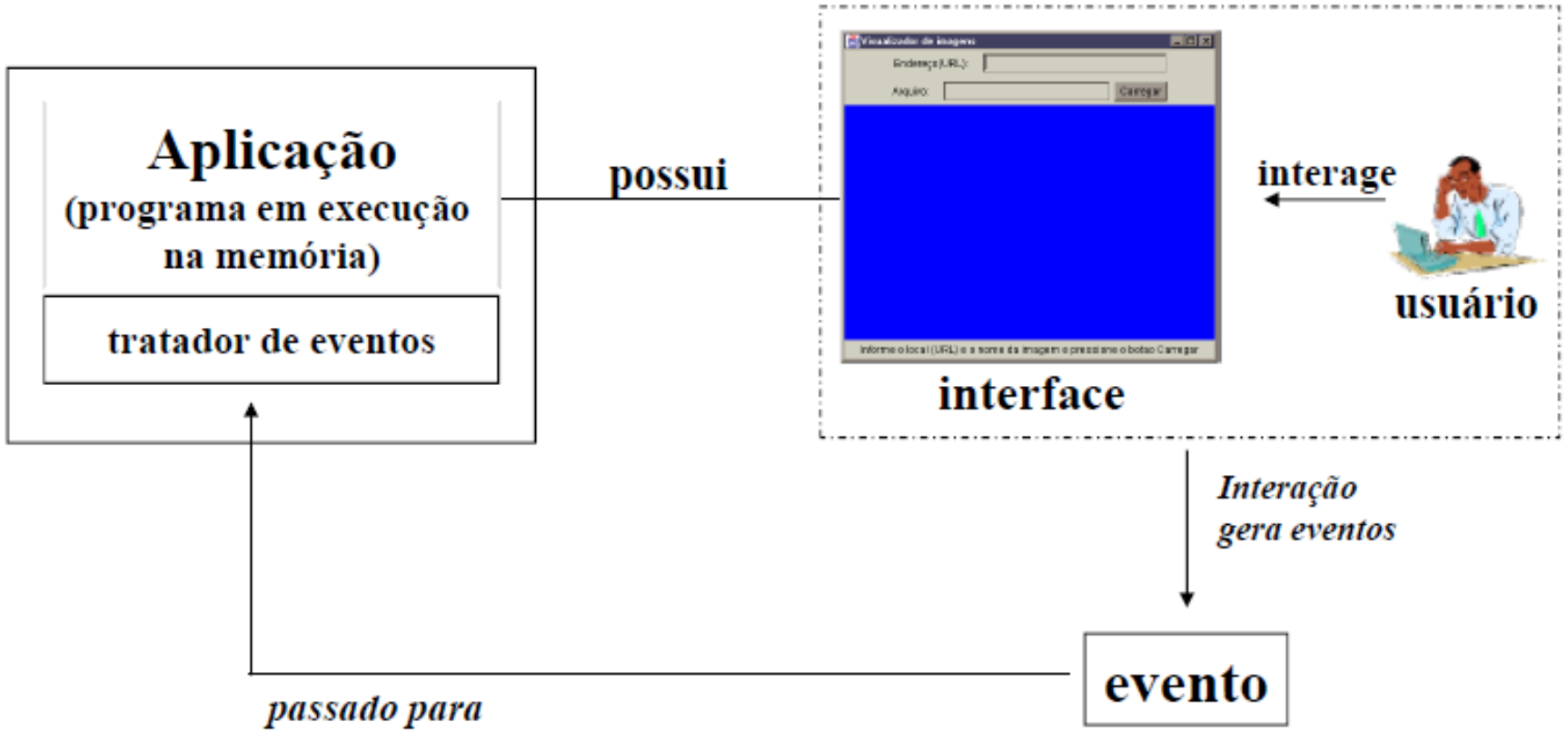
Eventos gerados pelo mouse (MouseEvents)

Pelo movimento do mouse;

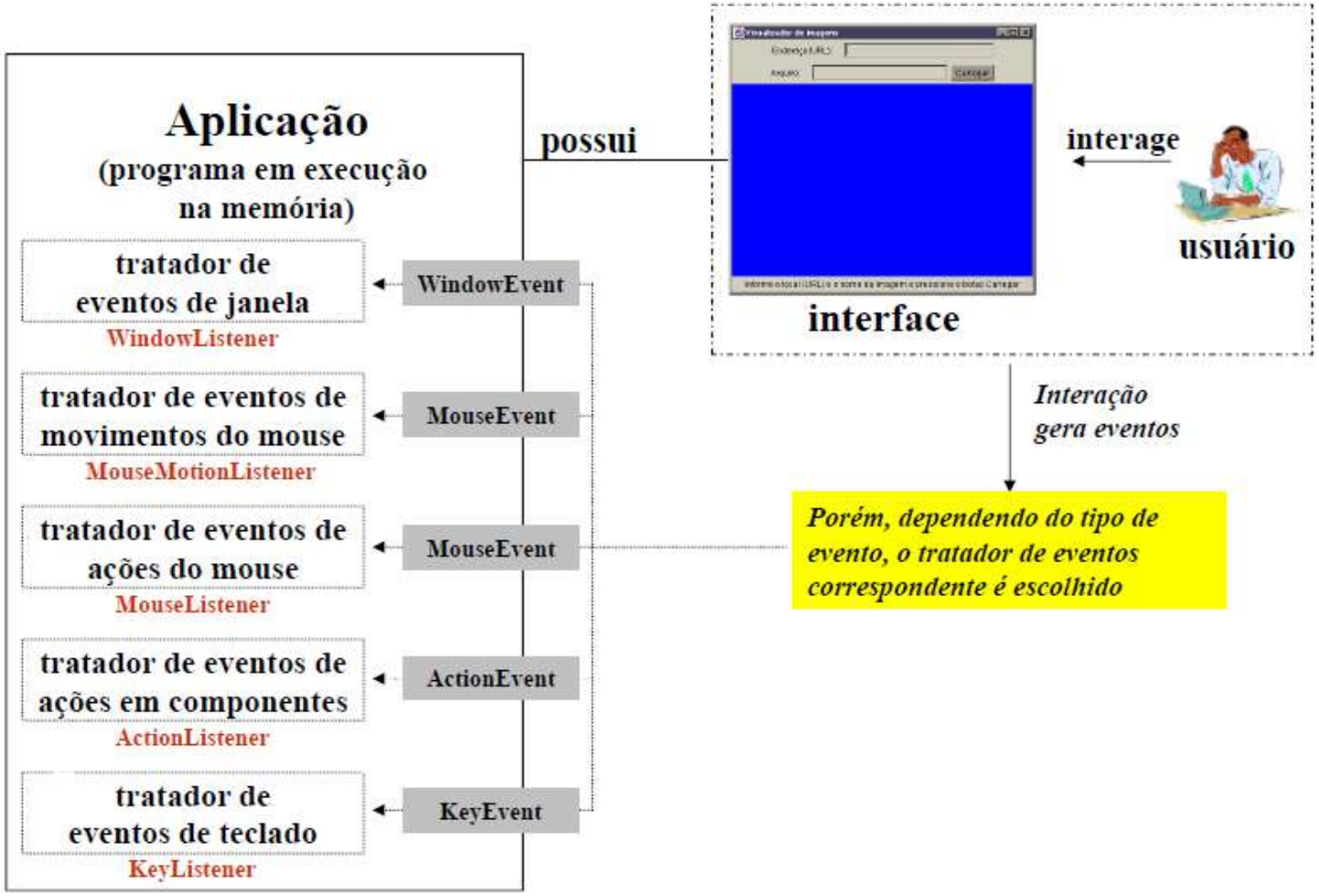
Por uma ação do mouse (botão clicado, pressionado ou solto);

Eventos gerados pelo teclado (KeyEvents).

Interface Gráfica



Interface Gráfica



Tipos de Tratadores de Eventos

- **WindowListener** - eventos de janelas
- **MouseListener** - eventos de mouse (clique)
- **MouseMotionListener** - eventos de movimento de mouse
- **ActionListener** - eventos de ação (geralmente gerados por botões)
- **KeyListener** - eventos gerados pelo teclado

Para cada um desses tipos o Java oferece uma Classe ou Interface que pode ser utilizada em programas.

Cada um deles possui uma série diferente de métodos, que tratam eventos específicos

Tratadores de Eventos

- Os eventos não são tratados automaticamente.
- Para cada componente de interface criado (janela, botão, painel, caixa de texto...), decida quais são os eventos que devem ser tratados (cada componente pode gerar um ou mais tipos de eventos);
- Após, defina uma classe adicional no programa que seja capaz de tratar cada um desses eventos. Essa classe, tratadora de eventos, deve ser uma classe filha de uma das classes tratadoras de eventos vistas anteriormente (WindowListener, MouseListener, MouseMotionListener, ActionListener ou KeyListener);
- Finalmente, crie objetos que sejam do tipo da classe tratadora de eventos que você definiu e depois diga para cada componente, qual é o objeto que trata seus eventos.

Tipos de Tratadores de Eventos

- Para capturar e tratar os eventos devem ser criados objetos de manipulação de eventos. Para cada tipo de evento existe uma classe Java específica para tratá-lo.
- Para os eventos, as classes-pai que podem ser utilizadas são:
- Eventos de janela (WindowEvent)

  WindowListener
- Eventos de Ação (ActionEvent)

  ActionListener
- Eventos de mouse (MouseEvent)

  MouseMotionListener → para movimentos do mouse

  MouseListener → para demais ações do mouse
- Eventos de teclado (KeyEvent) → KeyListener

```
public class Aplicação{
   public static void main(String argumentos[]){
      Janela         jan  = new Janela();
      TratEventosJan trat = new TratEventosJan();
      jan.addWindowListener(TratEventos());
      jan.show();
   }
}
```

A primeira classe é a que define a aplicação

```
class Janela extends Frame{
   public Janela(){
      setBackground(Color.blue);
      add("Center", new Label("Janela da aplicação"));
   }
}
```

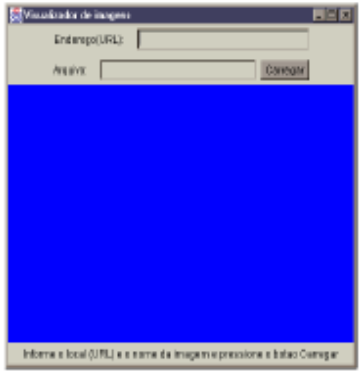
A segunda classe é a que define a interface (o tipo de janela) da aplicação

```
class TratEventosJan extends WindowAdapter{
   public void windowClosing(WindowEvent evento){
      System.exit(0);
   }
}
```

A última classe é a que define o tratador de eventos de janela da aplicação

```
public class Aplicação{
   public static void main(String argumentos[]){
      Janela          jan  = new Janela();
      TratEventosJan trat = new TratEventosJan();
      jan.addWindowListener(trat);
      jan.show();
   }
}
```

A primeira classe é responsável por criar a janela da aplicação.



```
class Janela extends Frame{
   public Janela(){
      setBackground(Color.blue);
      add("Center", new Label("Janela da aplicação"));
   }
}
```

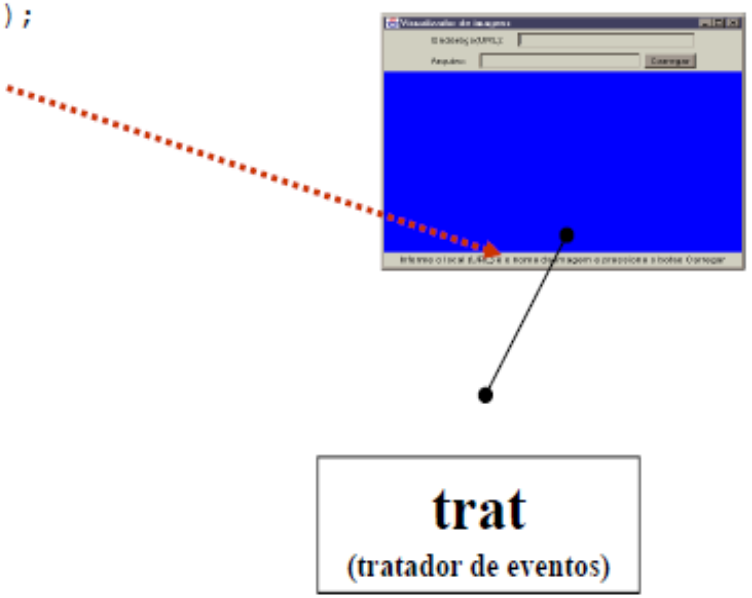
A janela é criada de acordo com a classe de janela definida!

```
class TratEventosJan extends WindowAdapter{
   public void windowClosing(WindowEvent evento){
      System.exit(0);
   }
}
```

```
public class Aplicação{
   public static void main(String argumentos[]){
      Janela          jan  = new Janela();
      TratEventosJan trat = new TratEventosJan();
      jan.addWindowListener(trat);
      jan.show();
   }
}
class Janela extends Frame{
   public Janela(){
      setBackground(Color.blue);
      add("Center", new Label("Janela da
aplicação"));
   }
}

class TratEventosJan extends WindowAdapter{
   public void windowClosing(WindowEvent evento){
      System.exit(0);
   }
}
```

Finalmente, o tratador de eventos é adicionado à janela:



Assim, todo evento de janela gerado nesta janela é capturado e tratado pelo objeto tratador de eventos.

```
public class Aplicação{
   public static void main(String argumentos[]){
      Janela          jan  = new Janela();
      TratEventosJan trat = new TratEventosJan();
      jan.addWindowListener(trat);
      jan.show();
   }
}

class Janela extends Frame{
   public Janela(){
      setBackground(Color.blue);
      add("Center", new Label("Janela da aplicação"));
   }
}
```

```
class TratEventosJan extends WindowAdapter{
   public void windowClosing(WindowEvent evento){
      System.exit(0);
   }
}
```

Em seguida, é criado o objeto tratador de eventos.

trat
(tratador de eventos)

Ele é criado de acordo com a classe definida para tratar eventos!

Eventos gerados pelo mouse

Os eventos gerados por alguma ação do mouse, são tratados pela classe

MouseListener ou MouseAdapter e suas classes-filhas:

mouseClicked(MouseEvent ev) → quando um botão do mouse é clicado;

mouseEntered(MouseEvent ev) → quando o mouse entra em um componente (passa por cima);

mouseExited(MouseEvent ev) → quando o mouse sai de um componente;

mousePressed(MouseEvent ev) → quando um botão do mouse é pressionado;

mouseReleased(MouseEvent ev) → quando um botão do mouse é solto

mouseMoved(MouseEvent ev) → quando o mouse é movido dentro de Eventos gerados pelo mouse um componente ou janela;

mouseDraged(MouseEvent ev) → quando o mouse é movido com um botão pressionado dentro de um componente ou janela (para implementar o recurso arrastar-e-soltar);

Dúvidas?