Events Handling

"Understanding Events, Handling events and GUI"

Advanced in Programming

Shakirullah Waseeb
shakir.waseeb@gmail.com

Nangarhar University

April 24, 2018

Agenda

Agenda

GUI Overview

- GUI programming is event-driven
- Hence, event handling is at the core of successful GUI programming
- Most events to which GUI components responds are generated by the user
- Events can be passed into GUI components in a variety of ways, depending upon the actual event
- Most common handled events are generated by:
    - Mouse
    - Keyboard
    - Various controls such as button, text box, drop down, check box etc.
- Events are supported by java.awt.event package

Agenda

Event Handling Approach

- Modern approach to event handling is based on **delegation event model**
- This approach defines standards and consistent mechanism to handle and process events
- In this model *source* generates an event and sends it to one or more *listeners*
    - The listener simply waits until it receives an event
    - Once received, the listener processes the event and then returns
    - Listeners must register with a source in order to receive an event notification

Agenda

Events

- In **delegation-model**, an event is an object that describes a *state change* in a source
- It can be generated as a result of users interacting with the elements in a graphical user interface
- Some of the activities that cause events to be generated are
    - pressing a button
    - entering a character via the keyboard
    - selecting an item in a list
    - clicking the mouse
- Events may also occur that are not directly caused by interactions with a user interface
    - an event may be generated when a timer expires
    - a counter exceeds a value
    - a software or hardware failure occurs
    - an operation is completed

Event Sources

- A **source** is an object that generates an event
- Occurs when the internal state of that object changes in some way
- May generate more than one type of event
- A source must register listeners in order for the listeners to receive notifications about a specific type of event
- Each type of event has its own registration method
  - general form

    public void *addTypeListener*(TypeListener el)
  - registering a keyboard listener

    public void *addKeyListener*(KeyEvent el)

Event Listeners

- A **listener** is an object that is notified when an event occurs
- Has two major requirements
  - must have been registered with one or more sources to receive notifications about specific types of events
  - must implement methods to receive and process these notifications
- The methods that receive and process events are defined in a set of interfaces found in java.awt.event
- MouseMotionListener interface defines two methods to receive notifications when the mouse is dragged or moved

Event-Listener Interfaces

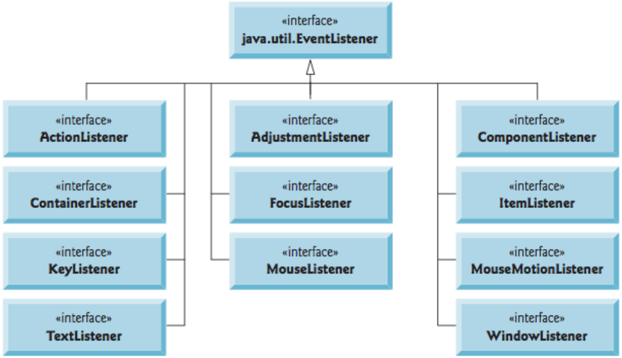


Figure: Some event-listener interfaces of package java.awt.event [1, Page 493]

Event Classes

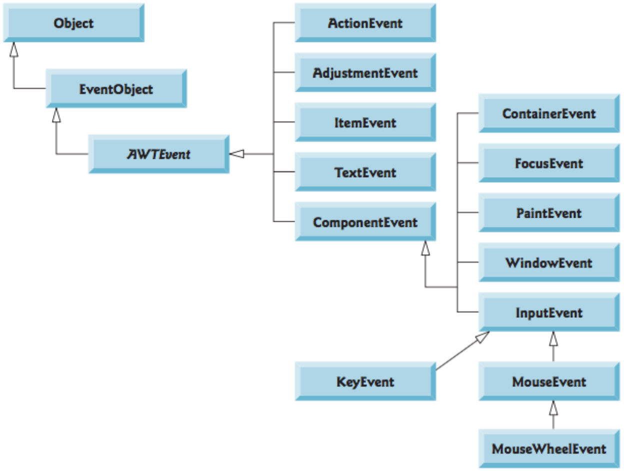
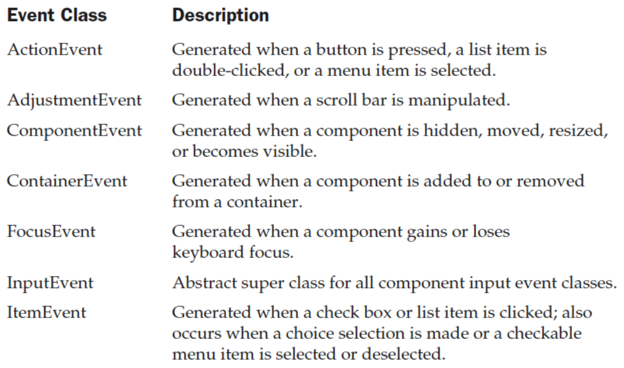


Figure: Some event classes of package java.awt.event [1, Page 492]

Event Classes

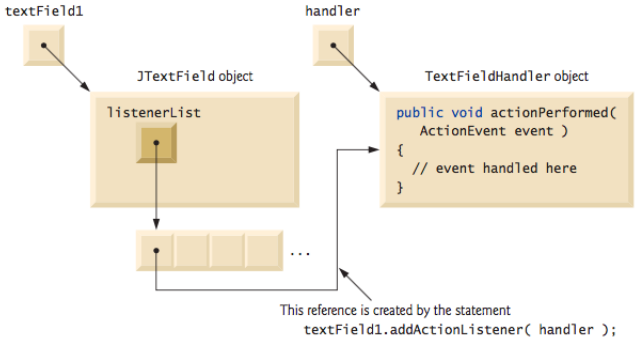
- The classes that represent events are at the core of Java's event handling mechanism
- Some of these classes are listed below



Event Class	Description
ActionEvent	Generated when a button is pressed, a list item is double-clicked, or a menu item is selected.
AdjustmentEvent	Generated when a scroll bar is manipulated.
ComponentEvent	Generated when a component is hidden, moved, resized, or becomes visible.
ContainerEvent	Generated when a component is added to or removed from a container.
FocusEvent	Generated when a component gains or loses keyboard focus.
InputEvent	Abstract super class for all component input event classes.
ItemEvent	Generated when a check box or list item is clicked; also occurs when a choice selection is made or a checkable menu item is selected or deselected.

Figure: Some Event Classes [3, Page 657]

Example


textField1
handler
JTextField object
TextFieldHandler object
listenerList
...
This reference is created by the statement
textField1.addActionListener(handler);

```
public void actionPerformed(
   ActionEvent event )
{
  // event handled here
}
```

Figure: Event registration for JTestField textField1 [1, Page 494]

Your Turn: Time to hear from you!

[1]

[1] https://fensafitters.files.wordpress.com/2013/07/3d095.jpg

References

- P.J. Deitel, H.M. Deitel
  Java How to program, 10th Edition .
  Prentice Hall, 2015.
- P.J. Deitel, H.M. Deitel
  Java How to program, 9th Edition .
  Prentice Hall, 2012.
- Herbert Schildt
  The complete reference Java2, 5th Edition .
  McGraw-Hill/Osborne, 2002.