Event Handling

Event and Listener

- Changing the state of an object is known as an event.
- For example, click on button, dragging mouse etc.
- The **java.awt.event** package provides many event classes and Listener interfaces for event handling.

The Delegation Event Model

- defines standard & consistent mechanism to generate and process events.
- A source code generates an event and sends it to one or more listeners.
- The listener simply waits until it receives an event.
- Once an event is received, the listener processes the event and then returns.
- In this model, the listeners must register with a source in order to receive an event notification.

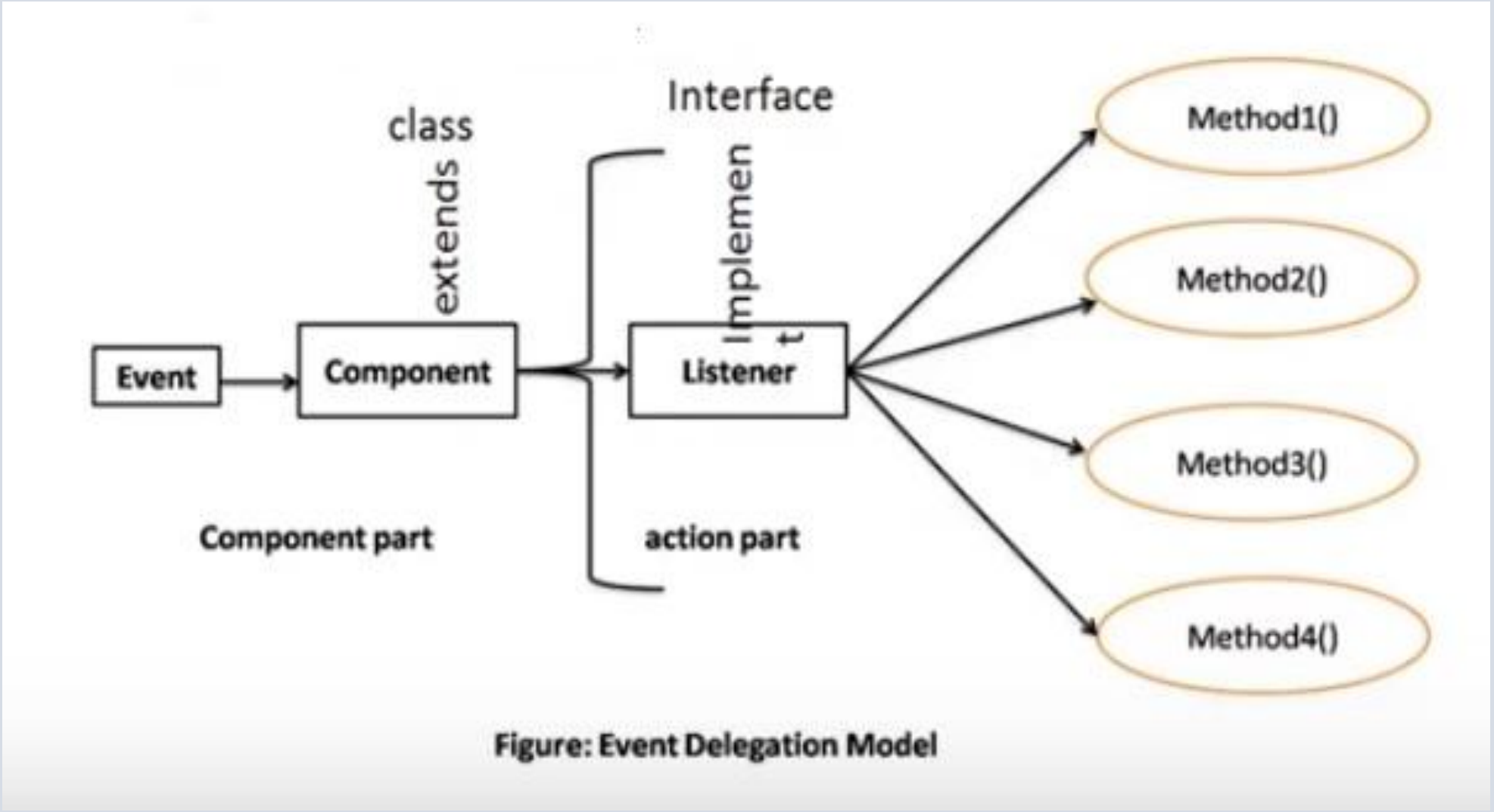

Figure: Event Delegation Model

Java Event classes & Listener Interfaces

Event Classes	Listener Interfaces
ActionEvent	ActionListener
MouseEvent	MouseListener and MouseMotionListener
MouseWheelEvent	MouseWheelListener
KeyEvent	KeyListener
ItemEvent	ItemListener
TextEvent	TextListener
AdjustmentEvent	AdjustmentListener
WindowEvent	WindowListener
ComponentEvent	ComponentListener
ContainerEvent	ContainerListener
FocusEvent	FocusListener

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 superclass 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.
KeyEvent	Generated when input is received from the keyboard.
MouseEvent	Generated when the mouse is dragged, moved, clicked, pressed, or released; also generated when the mouse enters or exits a component.
MouseWheelEvent	Generated when the mouse wheel is moved.
TextEvent	Generated when the value of a text area or text field is changed.
WindowEvent	Generated when a window is activated, closed, deactivated, deiconified, iconified, opened, or quit.

Steps to perform Event Handling

- Following steps are required to perform event handling:

    1. Register the component with the Listener

Registration Methods

- For registering the component with the Listener, many classes provide the registration methods. For example:
- **Button**
    - public void addActionListener(ActionListener a){}
- **MenuItem**
    - public void addActionListener(ActionListener a){}
- **TextField**
    - public void addActionListener(ActionListener a){}
    - public void addTextListener(TextListener a){}

- **TextArea**
  - public void addTextListener(TextListener a){}
- **Checkbox**
  - public void addItemListener(ItemListener a){}
- **Choice**
  - public void addItemListener(ItemListener a){}
- **List**
  - public void addActionListener(ActionListener a){}
  - public void addItemListener(ItemListener a){}

Events

- In the delegation model, an event is an object that describes a state change in a source.
- events generated by interacting directly
  - pressing a button
  - entering a character via the keyboard
  - selecting an item in a list
  - clicking the mouse
- events generated by not directly interacting
  - when a timer expires
  - a counter exceeds a value
  - a hardware or software failure occurs
  - an operation completed

Event Sources

- A source is an object that generates an event.
- Sources 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)

Type -> name of the event

el -> reference to the event listener

eg) addKeyListener( ) -> registers a keyboard event listener

addMouseMotionListener( )

Multicasting

- when an event occurs, all registered listeners are notified and receive a copy of the event object.

Unicasting

- some sources may allow only one listener to register.
- when such event occurs, only the registered listener is notified.

General form

public void addTypeListener(TypeListener el) throws java.util.TooManyListenersException

- A source must also provide a method that allows a listener to unregister an specific type of event

General form

public void removeTypeListener(TypeListener el)

eg) removeKeyListener( ) -> to remove a keyboard listener

Event Listeners

- A listener is an object that is notified when an event occurs.
- It has 2 major requirements
  - it must have been registered with one or more sources to receive notifications about specific types of events.
  - it 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

Event Classes

EventObject class

- super class of all events
- root of the Java event class hierarchy; it is in **java.util** package
- constructor
  - EventObject(Object src)

    src -> object that generates this event
- Event objects contains 2 methods
  - getSource( ) -> returns the source of the event
  - toString( ) -> returns the string equivalent of the event

AWTEvent class

- super class of all AWT based events.
- defined within the java.awt package
- subclass of EventObject
- Methods
  - getID( ) -> used to determine the type of the event

    int getID( )
- The package java.awt.event defines several event classes.

1. ActionEvent
2. AdjustmentEvent
3. ComponentEvent
4. ContainerEvent

5. FocusEvent
6. InputEvent
7. ItemEvent
8. KeyEvent
9. MouseEvent
10. MouseWheelEvent
11. TextEvent
12. WindowEvent

1. The ActionEvent class

- Generated when a button is pressed, a list item is double-clicked or a menu item is selected.
- Defines 4 integer constants
  - ALT_MASK
  - CTRL_MASK
  - META-MASK
  - SHIFT-MASK

  -> To identify any modifiers associated with an actionEvent
- 3 constructors
  - ActionEvent(Object src, int type, String cmd)
  - ActionEvent(Object src, int type, String cmd, int modifiers)
  - ActionEvent(Object src, int type, String cmd, long when, int modifiers))

- src -> reference to the object that generated the event
- type -> type of the event
- cmd -> command string of the event
- modifiers -> indicates which modifier key(ALT, CTRL, META, SHIFT), user pressed when the event was generated.
- when -> specifies when the event occurred.

- Methods
  - getActionCommand( ) -> to obtain the command name of the invoking object

    String getActionCommand( )
  - getModifiers( ) -> returns a value that indicates which modifier keys were pressed.

    int getModifiers( )
  - getWhen( ) -> returns the time at which the event took place.

    -> called event's time stamp

    long getWhen( )

2. The AdjustmentEvent class

- Generated when a scroll bar is manipulated.
- 5types of adjustment events identified by the following constants.

  BLOCK_DECREMENT -> user clicked inside the scroll bar to decrease its value

  BLOCK_INCREMENT

  TRACK -> the slider was dragged

  UNIT_DECREMENT -> button at the end of the scroll bar was clicked to decrease its value

  UNIT_INCREMENT

  ADJUSTMENT_VALUE_CHANGED -> indicates that a change has occurred

- Constructor

  AdjustmentEvent(Adjustable src, int id, int type, int data)

  id -> ADJUSTMENT_VALUE_CHANGED

- Methods
  - getAdjustable( ) -> returns the object that generated the event

    Adjustable getAdjustable( )

  - getAdjustmentType( ) -> returns one of the constants defined by AdjustmentEvent

    int getAdjustmentType( )

  - getValue( ) -> amount of adjustment that can be obtained.

    int getValue( )

3. The ComponentEvent class

- Generated when a component is hidden, moved, resized or becomes visible.
- Integer constants
  - COMPONENT_HIDDEN
  - COMPONENT_MOVED
  - COMPONENT_RESIZED
  - COMPONENT_SHOWN
- Constructor

  ComponentEvent(Component src, int type)
- Method

  getComponent( ) -> returns the component that generated the event.
- superclass of ContainerEvent, FocusEvent, KeyEvent, MouseEvent and WindowEvent.

4. The ContainerEvent class

- Generated when a component is added or removed from a container.
- Integer constants
  - COMPONENT_ADDED
  - COMPONENT_REMOVED
- Constructor

  ContainerEvent(Component src, int type, Component comp)

  comp -> component that has been added or removed from the container
- Methods
  - getContainer( ) -> returns a reference to the container that generated the event

    Container getContainer( )
  - getChild( ) -> returns a reference to the component added or removed

    Component getChild( )

5. The FocusEvent class

- Generated when a component gains or loses keyboard focus.
- Integer constants
  - FOCUS_GAINED
  - FOCUS_LOST
- Constructors
  - FocusEvent(Component src, int type)
  - FocusEvent(Component src, int type, boolean temporaryFlag)
  - FocusEvent(Component src, int type, boolean temporaryFlag, Component other)

    temporaryFlag -> true if focus event is temporary, otherwise false

    other -> the other component involved in focus change, called the opposite component, is passed in other.

    (eg. assume focus is in text field; if the user moves the mouse to adjust scroll bar, focus is temporarily lost)

- Methods
  - getOppositeComponent( )
    Component getOppositeComponent( )
  - isTemporary( )
    - indicates if this focus change is temporary
    - boolean isTemporary( )

6. The InputEvent class

- superclass for KeyEvent and MouseEvent classes
- subclass of ComponentEvent class
- Defines 8 values to represent the modifiers
  - ALT_DOWN_MASK
  - ALT_GRAPH_DOWN_MASK
  - BUTTON1_DOWN_MASK
  - BUTTON2_DOWN_MASK
  - BUTTON3_DOWN_MASK
  - CTRL_DOWN_MASK
  - META_DOWN_MASK
  - SHIFT_DOWN_MASK

- Methods
  - boolean isAltDown( )
  - boolean isAltGraphDown( )
  - boolean isControlDown( )
  - boolean isMetaDown( )
  - boolean isShiftDown( )

    → these methods are used to test if a modifier was pressed at the time an event is generated.

7. The ItemEvent class

- Generated when a check box or list item is checked.
- also occurs when a choice selection is made or a checkable menu item is selected or deselected.
- Integer constants
  - DESELECTED
  - SELECTED
  - ITEM_STATE_CHANGED

- Constructor

  ItemEvent(ItemSelectable src, int type, Object entry, int state)

  ItemSelectable -> interface

  entry -> specific item that generated the item event

  state -> current state of that item

- Methods
  - Object getItem( )
  - ItemSelectable getItemselectable( )
  - int getStateChange( ) -> returns the state change for this event(SELECTED or DESELECTED)

8. The KeyEvent class

- generated when input is received from the keyboard.
- Integer constants
  - KEY_PRESSED
  - KEY_RELEASED
  - KEY_TYPED
- Constructor

  KeyEvent(Component src, int type, long when, int modifiers, int code, char ch)

  code -> the virtual keycode VK_0 through VK_9 and VK_A through VK_Z, VK_UP, VK_DOWN….etc.,

  ch -> the character equivalent(if one exists) is passed in ch. If no valid character exists, then ch contains CHAR_UNDEFINED (pressing shift key will not generate a character)

9. The MouseEvent class

- Generated when the mouse is dragged, moved, clicked, pressed, or released.
- Integer constants
  - MOUSE_CLICKED
  - MOUSE_DRAGGED
  - MOUSE_ENTERED -> mouse entered a component
  - MOUSE_EXITED -> mouse exited from a component
  - MOUSE_MOVED
  - MOUSE_PRESSED
  - MOUSE_RELEASED
  - MOUSE_WHEEL

- Constructor

  MouseEvent(Component src, int type, long when, int modifiers, int x, int y, int clicks, boolean triggersPopup)

  when -> system time at which the mouse event occurred

  x & y -> coordinates of the mouse

  clicks -> the click count

  triggersPopup -> indicates if this event causes a pop-up menu to appear

- Methods
  - int getX( )
  - int getY( ) -> returns the X & Y coordinates of the mouse when the event occurred.
  - Point getPoint( ) -> returns a Point object that contains the X & Y coordinates
  - void translatePoint(int x, int y) -> changes the location of the event

- int getClickCount( ) -> obtain the no.of mouse clicks for this event.
- boolean isPopupTrigger( ) -> tests if this event causes a pop-up menu to appear on this platform
- int getButton( ) -> return value that represents the button that caused the event.

  -> the return values will be one of these constants,

  NOBUTTON, BUTTON1, BUTTON2, BUTTON3

10. The MouseWheelEvent class

- Generated when the mouse wheel is moved.
- subclass of MouseEvent
- Integer constants
  - WHEEL_BLOCK_SCROLL -> page-up or page-down scroll
  - WHEEL_UNIT_SCROLL -> line-up or line-down scroll
- Constructor

  MousewheelEvent(Component src, int type, long when, int modifiers, int x, int y, int clicks, boolean triggersPopup, int scrollHow, int amount, int count)

  ScrollHow -> either WHEEL_UNIT_SCROLL or WHEEL_BLOCK_SCROLL

  amount -> no.of units to scroll

  count -> no.of rotational units that the wheel moved

Methods

- int getWheelRotation( )
  - to obtain the no.of rotational units
  - value -> positive -> wheel moved counter clockwise
    negative -> wheel moved clockwise
- int getScrollType( )
- int getScrollAmount( )

11. WindowEvent class

- Generated when a window is activated, closed, deactivated, deiconified, iconified, opened or quit.
- 10 types of window events
- subclass of ComponentEvent
- constructors

  WindowEvent(Window src, int type)

  WindowEvent(Window src, int type, Window other)

  WindowEvent(Window src, int type, int fromState, int toState)

  WindowEvent(Window src, int type, Window other, int fromState, int toState)

    other -> opposite window when a focus event occurs

    fromState -> prior state of the window

    toState -> new state of the window

Methods

- Window getWindow( )
- Window getOppositeWindow( )
- int getOldState( )
- int getNewState( )

12. TextEvent class

- Generated when the value of a text area or text field is changed.
- Integer constant

  TEXT_VALUE_CHANGED
- Constructor

  TextEvent(Object src, int type)

Sources of Events

- Button
- Checkbox
- Choice
- List
- Menu item
- Scroll bar
- Text components
- Window

EventListener Interfaces

- ActionListener Interface
  - defines one method to receive action events

    void actionPerformed(ActionEvent ae)
- AdjustmentListener Interface
  - defines one method to receive adjustment events

    void adjustmentValueChanged(AdjustmentEvent ae)
- ComponentListener Interface
  - defines 4 methods to recognize when a component is hidden, moved, resized or shown

    void ComponentHidden(ComponentEvent ce)

    void ComponentMoved(ComponentEvent ce)

    void ComponentResized(ComponentEvent ce)

    void ComponentShown(ComponentEvent ce)

- ContainerListener Interface
  - defines 2 methods to recognize when a component is added or removed from a container

    void ComponentAdded(ContainerEvent ce)

    void ComponentRemoved(ContainerEvent ce)
- FocusListener Interface
  - defines 2 methods to recognize when a component gains or loses keyboard focus

    void focusGained(FocusEvent fe)

    void focusLost(FocusEvent fe)
- ItemListener Interface
  - defines one method to recognize when the state of an item changes

    void itemStateChanged(ItemEvent ie)

- KeyListener Interface
  - defines 3 methods to recognize when a key is pressed, released or typed.
    - void keyPressed(KeyEvent ke)
    - void keyReleased(KeyEvent ke)
    - void keyTyped(KeyEvent ke)
- MouseListener Interface
  - defines 5 methods
    - void mouseClicked(MouseEvent me)
    - void mouseEntered(MouseEvent me)
    - void mouseExited(MouseEvent me)
    - void mousePressed(MouseEvent me)
    - void mouseReleased(MouseEvent me)

- MouseWheelListener Interface
  - defines one method to recognize when the mouse wheel is moved

    void mouseWheelMoved(MouseWheelEvent mwe)
- TextListener Interface
  - defines one method to recognize when a text value changes

    void textChanged(TextEvent te)
- WindowFocusListener Interface
  - defines 2 methods to recognize when a window gains or loses input focus

    void windowGainedFocus(WindowEvent we)

    void windowLostFocus(WindowEvent we)

- WindowListener Interface
  - defines 7 methods
    - void windowActivated(WindowEvent we)
    - void windowClosed(WindowEvent we)
    - void windowClosing(WindowEvent we)
    - void windowDeactivated(WindowEvent we)
    - void windowDeiconified(WindowEvent we)
    - void windowIconified(WindowEvent we)
    - void windowOpened(WindowEvent we)

Adapter Classes

- Simplify the creation of event handlers in certain situations.
- useful when you want to receive and process only some of the events that are handled by a particular event listener interface.
- eg. the MouseMotionAdapter class has two methods, mouseDragged( ) and mouseMoved( )
  - signatures same as defined in MouseMotionListener interface
  - if only interested in mouse drag events, simply extend MouseMotionAdapter and implement mouseDragged( )

- The commonly used adapter classes in java.awt.event are,
  - ComponentAdapter
  - ContainerAdapter
  - FocusAdapter
  - KeyAdapter
  - MouseAdapter
  - MouseMotionAdapter
  - WindowAdapter