Event Handling

Event Handling

- User actions are called “events”
  - Low-level window events
    - Window changes
  - Low-level component events
    - Mouse events, Keyboard events, Component changes
  - High-level component-specific events
    - Button events, Menu events, List events, etc.
- Events are handled by attaching “event listeners” to windows and components
- Swing defines a “listener interface” for each category of events
  - Examples: WindowListener, MouseListener, MenuListener

Event Handling

- A “listener” is an object that implements a listener interface
- Listeners are attached to windows and components by calling the appropriate “addXXXListener” methods
  - addWindowListener, addMouseListener, addMenuListener, etc.
- The window or component will notify all attached listeners about all relevant events
  - windowActivated, mouseClicked, menuSelected, etc.
- Listener methods accept an EventObject parameter that describes the event that occurred
  - WindowEvent, MouseEvent, MenuEvent, etc.
  - The EventObject.getSource method returns the window or component that generated the event

Event Handling

- Listeners are typically implemented as inner classes
  - Named inner class
    - class MyMouseHandler implements MouseListener { … }
    - this.addMouseListener(new MyMouseHandler());
  - Anonymous inner class
    - this.addMouseListener(new MouseListener() { … });

Event Handling

- Event Adapter classes
  - Frequently, a program only wants to handle a few of the events defined by a listener interface
  - However, implementing the listener interface requires implementations for all event methods
  - Swing provides Adapter superclasses that provide empty implementations of all methods in a listener interface
    - WindowAdapter, MouseAdapter, KeyAdapter, etc.
  - This allows a program to create a listener by subclassing an adapter class, and overriding only the event methods it cares about
  - Some adapter classes implement multiple listener interfaces
    - EXAMPLE: MouseAdapter implements MouseListener, MouseMotionListener, and MouseWheelListener

Mouse Events

- MouseListener interface
- MouseMotionListener interface
- MouseWheelListener interface
- MouseAdapter class
- EXAMPLE: Dragging and scrolling shapes in Drawing

Component Events

- ComponentListener interface
- ComponentAdapter class
- EXAMPLE: Updating width and height in text shapes in Drawing

Window Events

- WindowListener interface
- WindowStateListener interface
- WindowAdapter class
- EXAMPLE: Frame window in Drawing

Keyboard Events

- Keyboard Focus
  - When the user types a key, which component gets the key event?
    - The component with the “keyboard focus”
  - Only one window has the keyboard focus at a time
  - Within that window, only one component has the keyboard focus at a time
  - A component can request the keyboard focus by calling Component.requestFocus or Component.requestFocusInWindow
  - EXAMPLE: A text field requests the keyboard focus when the user clicks on it so it will receive key events

Keyboard Events

- When a window receives or loses the keyboard focus, it generates “gained focus” and “lost focus” events
  - WindowFocusListener interface
- When a component receives or loses the keyboard focus, it generates “gained focus” and “lost focus” events
  - FocusListener interface
- EXAMPLE: In Drawing, when the frame window receives keyboard focus, DrawingFrame calls requestFocusInWindow on the DrawingComponent so it will receive keyboard events

Keyboard Events

- KeyListener interface
- KeyAdapter class
- EXAMPLE: Moving shapes with arrow keys in Drawing