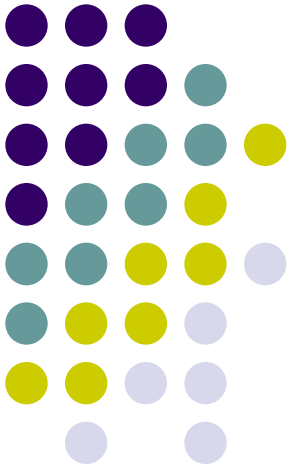
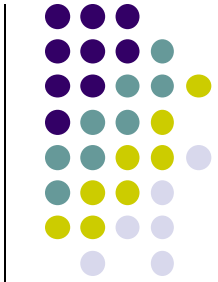
Event Handling

12 Marks



Event Classes



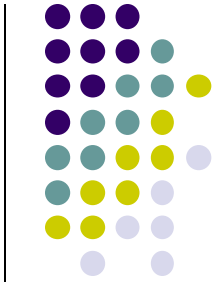
- EventObject is a superclass of all events.
- AWTEvent is a superclass of all AWT events that are handled by the delegation event model.

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.

Table 20-1. *Main Event Classes in* java.awt.event

Event Class	Description
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. (Added by Java 2, version 1.4)
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.

The ActionEvent Class



- The **ActionEvent class defines three type ACTION_FIRST, ACTION_LAST and ACTION_PERFORMED**

- The **ActionEvent class defines four integer constants ALT_MASK, CTRL_MASK, META_MASK, and SHIFT_MASK**

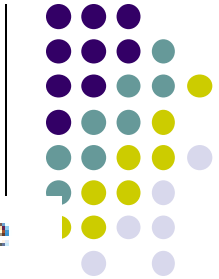
- **ActionEvent has these three constructors:**

  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)*

The Adjustment Event Class



BLOCK_DECREMENT	The user clicked inside the scroll bar to decrease its value.
BLOCK_INCREMENT	The user clicked inside the scroll bar to increase its value.
TRACK	The slider was dragged.
UNIT_DECREMENT	The button at the end of the scroll bar was clicked to decrease its value.
UNIT_INCREMENT	The button at the end of the scroll bar was clicked to increase its value.

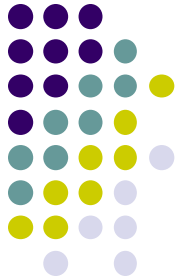
AdjustmentEvent(Adjustable *src, int id, int type, int data)*

The ComponentEvent Class

COMPONENT_HIDDEN	The component was hidden.
COMPONENT_MOVED	The component was moved.
COMPONENT_RESIZED	The component was resized.
COMPONENT_SHOWN	The component became visible.

ComponentEvent(Component *src*, *int type)*

The ContainerEvent Class

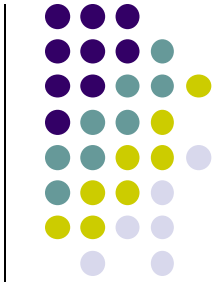


- The **ContainerEvent class defines int constants that can be used to identify them:**

  COMPONENT_ADDED and COMPONENT_REMOVED

ContainerEvent(Component *src, int type, Component comp)

The FocusEvent Class



- These events are identified by the integer constants

  FOCUS_GAINED and FOCUS_LOST

FocusEvent(Component *src, int type)*

FocusEvent(Component *src, int type, boolean temporaryFlag)*

Focus Event(Component *src, int type, boolean temporaryFlag, Component other)*

The InputEvent Class

InputEvent defines several integer constants that represent any modifiers, such as the control key being pressed, that might be associated with the event

ALT_MASK	BUTTON2_MASK	META_MASK
ALT_GRAPH_MASK	BUTTON3_MASK	SHIFT_MASK
BUTTON1_MASK	CTRL_MASK	

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	

- To test if a modifier was pressed at the time an event is generated, use the

isAltDown( ), isAltGraphDown( ), isControlDown( ), isMetaDown( ), and isShiftDown( ) methods.

The ItemEvent Class

- An **ItemEvent is generated when a check box or a list item is clicked or when a checkable** menu item is selected or deselected.

DESELECTED	The user deselected an item.
SELECTED	The user selected an item.

ItemEvent has this constructor:

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

The KeyEvent Class

- There are three types of key events, which are identified by these integer constants:

KEY_PRESSED, KEY_RELEASED, and KEY_TYPED

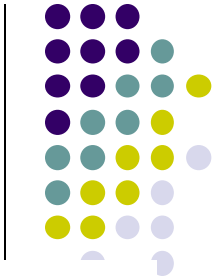
There are many other integer constants that are defined by **KeyEvent.**

VK_0 through VK_9 and VK_A through VK_Z

VK_ENTER	VK_ESCAPE	VK_CANCEL	VK_UP
VK_DOWN	VK_LEFT	VK_RIGHT	VK_PAGE_DOWN
VK_PAGE_UP	VK_SHIFT	VK_ALT	VK_CONTROL

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

The MouseEvent Class

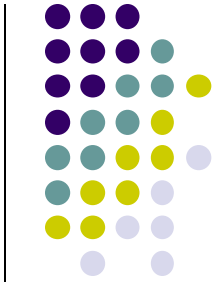


MOUSE_CLICKED	The user clicked the mouse.
MOUSE_DRAGGED	The user dragged the mouse.
MOUSE_ENTERED	The mouse entered a component.
MOUSE_EXITED	The mouse exited from a component.
MOUSE_MOVED	The mouse moved.
MOUSE_PRESSED	The mouse was pressed.
MOUSE_RELEASED	The mouse was released.
MOUSE_WHEEL	The mouse wheel was moved (Java 2, v1.4).

MouseEvent is a subclass of **InputEvent**. Here is one of its constructors.

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

The TextEvent Class



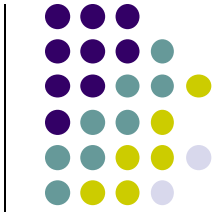
- **TextEvent defines the integer** constant

TEXT_VALUE_CHANGED

The one constructor for this class is shown here:

TextEvent(Object *src, int type)*

The WindowEvent Class



WINDOW_ACTIVATED	The window was activated.
WINDOW_CLOSED	The window has been closed.
WINDOW_CLOSING	The user requested that the window be closed.
WINDOW_DEACTIVATED	The window was deactivated.
WINDOW_DEICONIFIED	The window was deiconified.
WINDOW_GAINED_FOCUS	The window gained input focus.
WINDOW_ICONIFIED	The window was iconified.
WINDOW_LOST_FOCUS	The window lost input focus.
WINDOW_OPENED	The window was opened.
WINDOW_STATE_CHANGED	The state of the window changed. (Added by Java 2, version 1.4.)

WindowEvent is a subclass of **ComponentEvent**. It defines several constructors. The first is

WindowEvent(Window *src*, int *type*)

Delegation Event Model

The ActionListener Interface

This interface defines the **actionPerformed( )** method that is invoked when an action event occurs. Its general form is shown here:

void actionPerformed(ActionEvent *ae*)

The AdjustmentListener Interface

This interface defines the **adjustmentValueChanged( )** method that is invoked when an adjustment event occurs. Its general form is shown here:

void adjustmentValueChanged(AdjustmentEvent *ae*)

The ComponentListener Interface

This interface defines four methods that are invoked when a component is resized, moved, shown, or hidden. Their general forms are shown here:

void componentResized(ComponentEvent *ce*)
void componentMoved(ComponentEvent *ce*)
void componentShown(ComponentEvent *ce*)
void componentHidden(ComponentEvent *ce*)

The ContainerListener Interface

This interface contains two methods. When a component is added to a container, **componentAdded( )** is invoked. When a component is removed from a container, **componentRemoved( )** is invoked. Their general forms are shown here:

void componentAdded(ContainerEvent *ce*)
void componentRemoved(ContainerEvent *ce*)

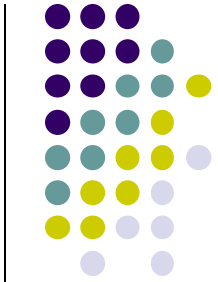
The FocusListener Interface

This interface defines two methods. When a component obtains keyboard focus, **focusGained( )** is invoked. When a component loses keyboard focus, **focusLost( )** is called. Their general forms are shown here:

void focusGained(FocusEvent *fe*)
void focusLost(FocusEvent *fe*)

The ItemListener Interface

void itemStateChanged(ItemEvent *ie)*



The KeyListener Interface

The general forms of these methods are shown here:

void keyPressed(KeyEvent *ke)*

void keyReleased(KeyEvent *ke)*

void keyTyped(KeyEvent *ke)*

The MouseListener Interface

void mouseClicked(MouseEvent *me)*

void mouseEntered(MouseEvent *me)*

void mouseExited(MouseEvent *me)*

void mousePressed(MouseEvent *me)*

void mouseReleased(MouseEvent *me)*

The MouseMotionListener Interface

void mouseDragged(MouseEvent *me)*

void mouseMoved(MouseEvent *me)*

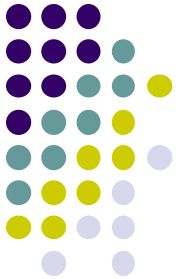
The TextListener Interface

void textChanged(TextEvent *te)*

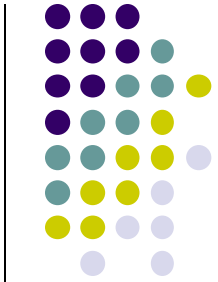
The WindowFocusListener Interface

void windowGainedFocus(WindowEvent *we)*

void windowLostFocus(WindowEvent *we)*



The WindowListener Interface



void windowActivated(WindowEvent *we)*

void windowClosed(WindowEvent *we)*

void windowClosing(WindowEvent *we)*

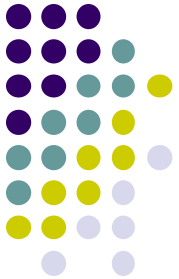
void windowDeactivated(WindowEvent *we)*

void windowDeiconified(WindowEvent *we)*

void windowIconified(WindowEvent *we)*

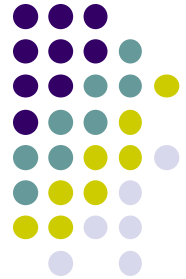
void windowOpened(WindowEvent *we)*

Adapter Classes



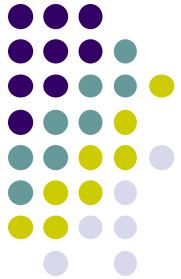
- Java adapter classes *provide the default implementation of listener interfaces*.
- If you inherit the adapter class, you will not be forced to provide the implementation of all the methods of listener interfaces.
- It *saves code*.

Advantages of Adapter classes



- It assists the unrelated classes to work combinedly.
- It provides ways to use classes in different ways.
- It increases the transparency of classes.
- It provides a way to include related patterns in the class.
- It provides a pluggable kit for developing an application.
- It increases the reusability of the class.

Adapter classes



Adapter class	Listener interface
WindowAdapter	WindowListener
KeyAdapter	KeyListener
MouseAdapter	MouseListener
MouseMotionAdapter	MouseMotionListener
FocusAdapter	FocusListener
ComponentAdapter	ComponentListener
ContainerAdapter	ContainerListener
HierarchyBoundsAdapter	HierarchyBoundsListener

java.awt.dnd Adapter classes

Adapter class	Listener interface
DragSourceAdapter	DragSourceListener
DragTargetAdapter	DragTargetListener

javax.swing.event Adapter classes

Adapter class	Listener interface
MouseInputAdapter	MouseInputListener
InternalFrameAdapter	InternalFrameListener

Program (Windows)

Program (Mouse)