UNITED
GROUP OF INSTITUTIONS

Web Technology (KCS-602) Unit 1

Prepared By

Abhishek Kesharwani

Assistant Professor,UCER Naini,Allahabad

Event and Listener (Java Event Handling)

- 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.

Event classes and 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

Steps to perform Event Handling

Following steps are required to perform event handling:

- Implement the Listener interface and overrides its methods
- Register the component with the Listener

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){}

Java ActionListener Interface

The Java ActionListener is notified whenever you click on the button or menu item. It is notified against ActionEvent. The ActionListener interface is found in java.awt.event package.

It has only one method: actionPerformed().

- actionPerformed() method
- The actionPerformed() method is invoked automatically whenever you click on the registered component.

public abstract void actionPerformed(ActionEvent e);

Java MouseListener Interface

The Java MouseListener is notified whenever you change the state of mouse. It is notified against MouseEvent. The MouseListener interface is found in java.awt.event package. It has five methods.

Methods of MouseListener interface

The signature of 5 methods found in MouseListener interface are given below:

public abstract void mouseClicked(MouseEvent e);

public abstract void mouseEntered(MouseEvent e);

public abstract void mouseExited(MouseEvent e);

public abstract void mousePressed(MouseEvent e);

public abstract void mouseReleased(MouseEvent e);

MouseMotionListener Interface

- The Java MouseMotionListener is notified whenever you move or drag mouse. It is notified against MouseEvent. The MouseMotionListener interface is found in java.awt.event package. It has two methods.

Methods of MouseMotionListener interface

The signature of 2 methods found in MouseMotionListener interface are given below:

public abstract void mouseDragged(MouseEvent e);

public abstract void mouseMoved(MouseEvent e);

ItemListener Interface

The Java ItemListener is notified whenever you click on the checkbox. It is notified against ItemEvent. The ItemListener interface is found in java.awt.event package. It has only one method: itemStateChanged().

- The itemStateChanged() method is invoked automatically whenever you click or unclick on the registered checkbox component.

public abstract void itemStateChanged(ItemEvent e);

KeyListener Interface

The Java KeyListener is notified whenever you change the state of key. It is notified against KeyEvent. The KeyListener interface is found in java.awt.event package. It has three methods.

Methods of KeyListener interface

The signature of 3 methods found in KeyListener interface are given below:

- **public abstract void** keyPressed(KeyEvent e);
- **public abstract void** keyReleased(KeyEvent e);
- **public abstract void** keyTyped(KeyEvent e);

WindowListener Interface

The Java WindowListener is notified whenever you change the state of window. It is notified against WindowEvent. The WindowListener interface is found in java.awt.event package. It has three methods.

Methods of WindowListener interface

The signature of 7 methods found in WindowListener interface are given below:

- **public abstract void** windowActivated(WindowEvent e);
- **public abstract void** windowClosed(WindowEvent e);
- **public abstract void** windowClosing(WindowEvent e);
- **public abstract void** windowDeactivated(WindowEvent e);
- **public abstract void** windowDeiconified(WindowEvent e);
- **public abstract void** windowIconified(WindowEvent e);
- **public abstract void** windowOpened(WindowEvent e);

LayoutManagers

The LayoutManagers are used to arrange components in a particular manner. LayoutManager is an interface that is implemented by all the classes of layout managers.

There are following classes that represents the layout managers:

- java.awt.BorderLayout
- java.awt.FlowLayout
- java.awt.GridLayout
- java.awt.CardLayout
- java.awt.GridBagLayout

BorderLayout

- The BorderLayout is used to arrange the components in five regions: north, south, east, west and center. Each region (area) may contain one component only. It is the default layout of frame or window. The BorderLayout provides five constants for each region:
- **public static final int NORTH**
- **public static final int SOUTH**
- **public static final int EAST**
- **public static final int WEST**
- **public static final int CENTER**

Constructors of BorderLayout class:

- **BorderLayout():** creates a border layout but with no gaps between the components.
- **JBorderLayout(int hgap, int vgap):** creates a border layout with the given horizontal and vertical gaps between the components.

GridLayout

The GridLayout is used to arrange the components in rectangular grid. One component is displayed in each rectangle.

Constructors of GridLayout class

- **GridLayout():** creates a grid layout with one column per component in a row.
- **GridLayout(int rows, int columns):** creates a grid layout with the given rows and columns but no gaps between the components.
- **GridLayout(int rows, int columns, int hgap, int vgap):** creates a grid layout with the given rows and columns alongwith given horizontal and vertical gaps.

FlowLayout

The FlowLayout is used to arrange the components in a line, one after another (in a flow). It is the default layout of applet or panel.

Fields of FlowLayout class

- **public static final int LEFT**
- **public static final int RIGHT**
- **public static final int CENTER**
- **public static final int LEADING**
- **public static final int TRAILING**