210301401 ADVANCED JAVA

UNIT	MODULES	WEIGHTAGE
1	File Handling	20 %
2	Java Collection Framework	20 %
3	Event Handling, Swing and GUI Components	20 %
4	Swing, GUI Components and Layout Manager	20 %
5	Database Connectivity (JDBC)	20 %

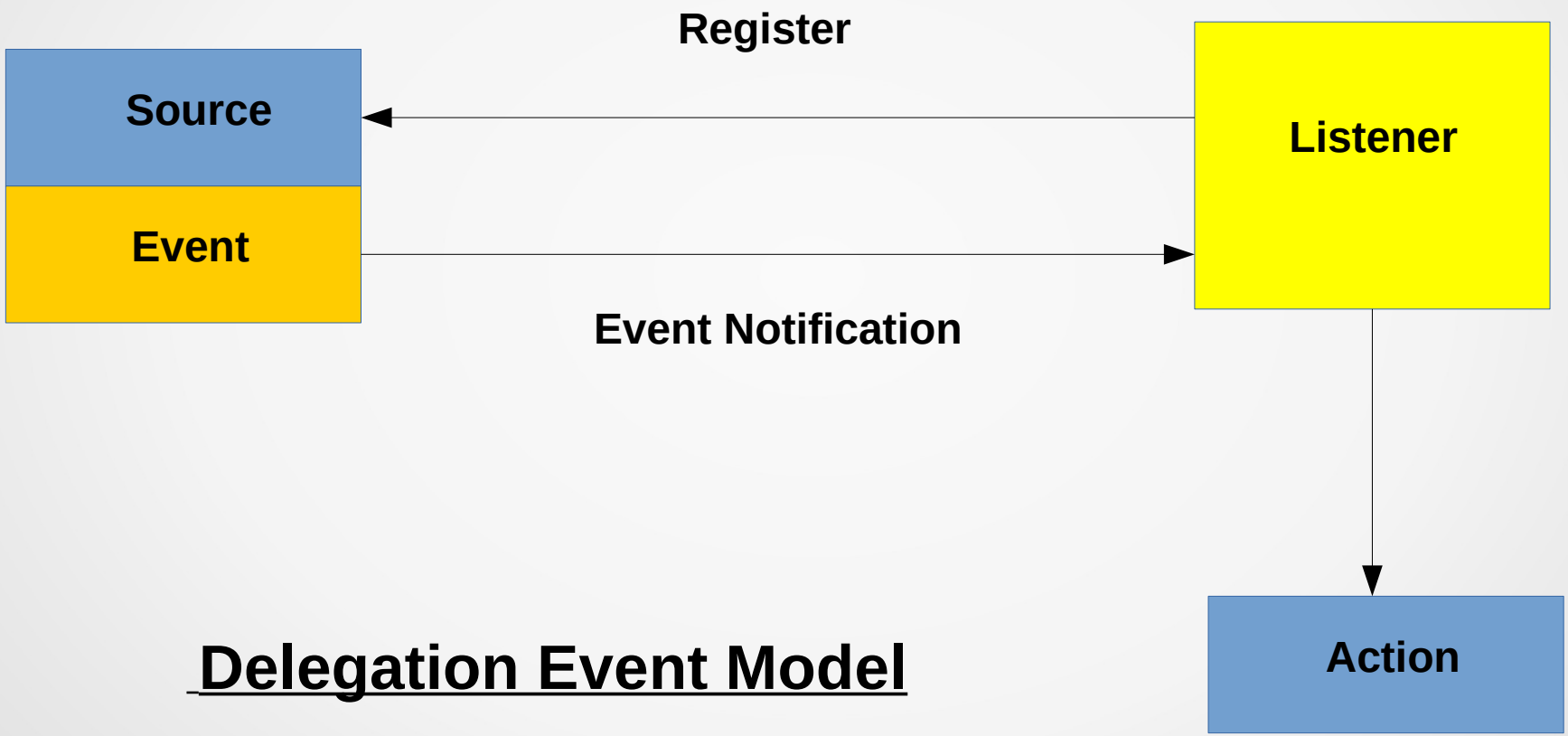
UNIT -3 Event Handling, Swing and GUI Components

- Event Handling
- Delegation Event Model
- Events
- Events Listeners
- Registering Listeners with sources
- Swing GUI Components

UNIT - 3 Delegation Event Model

- In Graphical User Interface **(GUI) environment, actions are initiated by the press of a button**, click of a button, a key press etc.
- There for **appropriate mechanisms are needed to capture such events and to react to the events by executing a piece of code.**
- Event in Java are handled by **Delegation Event Model.**
- In this model, there is a **source, which generates events.** There is a **listener, which can listen** to the happening of an event and initiate an action. JavaProvide such mechanisms.

UNIT - 3 Delegation Event Model



Delegation Event Model

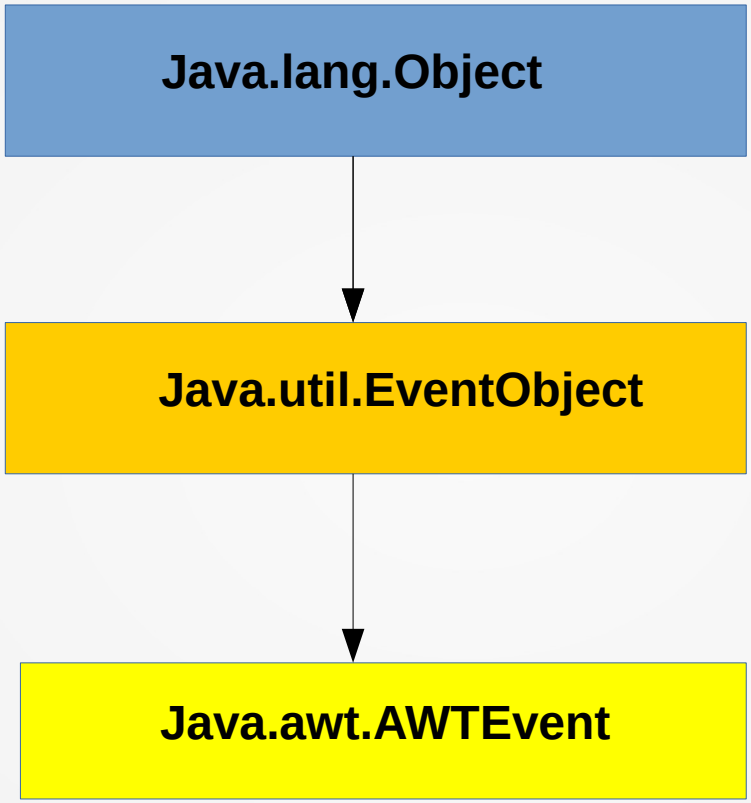
UNIT - 3 Delegation Event Model

- A **listener has to register with a source**. Any number of listeners can register with a source.
- A listener can register with many event sources.
- When an **event takes place, it is notified to the listeners**, which are registered with the source.
- **The listener then initiates an action.**

UNIT - 3 Event Handling

- An **events is an object that describe** the changes of state of a source.
  - i.e – Mouse click is event from the source mouse.
- The superclass of all events is
  - ***java.until.EventObject***
- The superclass of all AWT evetns is
  - ***java.util.AWTEvent***
- ***AWTEvent*** class is an abstract class contain subclasses, which are concrete and are packaged in ***java.awt.event***

UNIT – 3 Event Handling



UNIT -2 classes of java.awt.event Package

Name of Class	Purpose
Action Event	This class deals with **high-level event. The event occurs when the componet-specific action** take place. i.e - Button Press, Menu item selection
Adjustment Event	This class deals with events generated by **the adjustable objects like Scroll bar change**
Component Event	This class deals with the **lower – level events. The event occurs when the component is moved, resized or visibility is changed.** i.e - Button, Checkbox and Scroll bars display on screen
Item Event	This class deals with the events generated when **a Check box or list iteam choice is selcted or deselected**

UNIT -3 classes of java.awt.event Package

Name of Class	Purpose
Key Event	This class deals with the **events generated by key strokes.** i.e - Key is pressed, typed or released
Mouse Event	This class deals with **events generated by mouse licks and movements.** i.e - On mouse action
Text Event	This class deals with **events generated by the change of object's text.** i.e. - A text of an object is changed
Window Event	This class deals with events generated **by the change of window status.** i.e - Indicates the changes in the status of the window

UNIT – 3 Event Handling

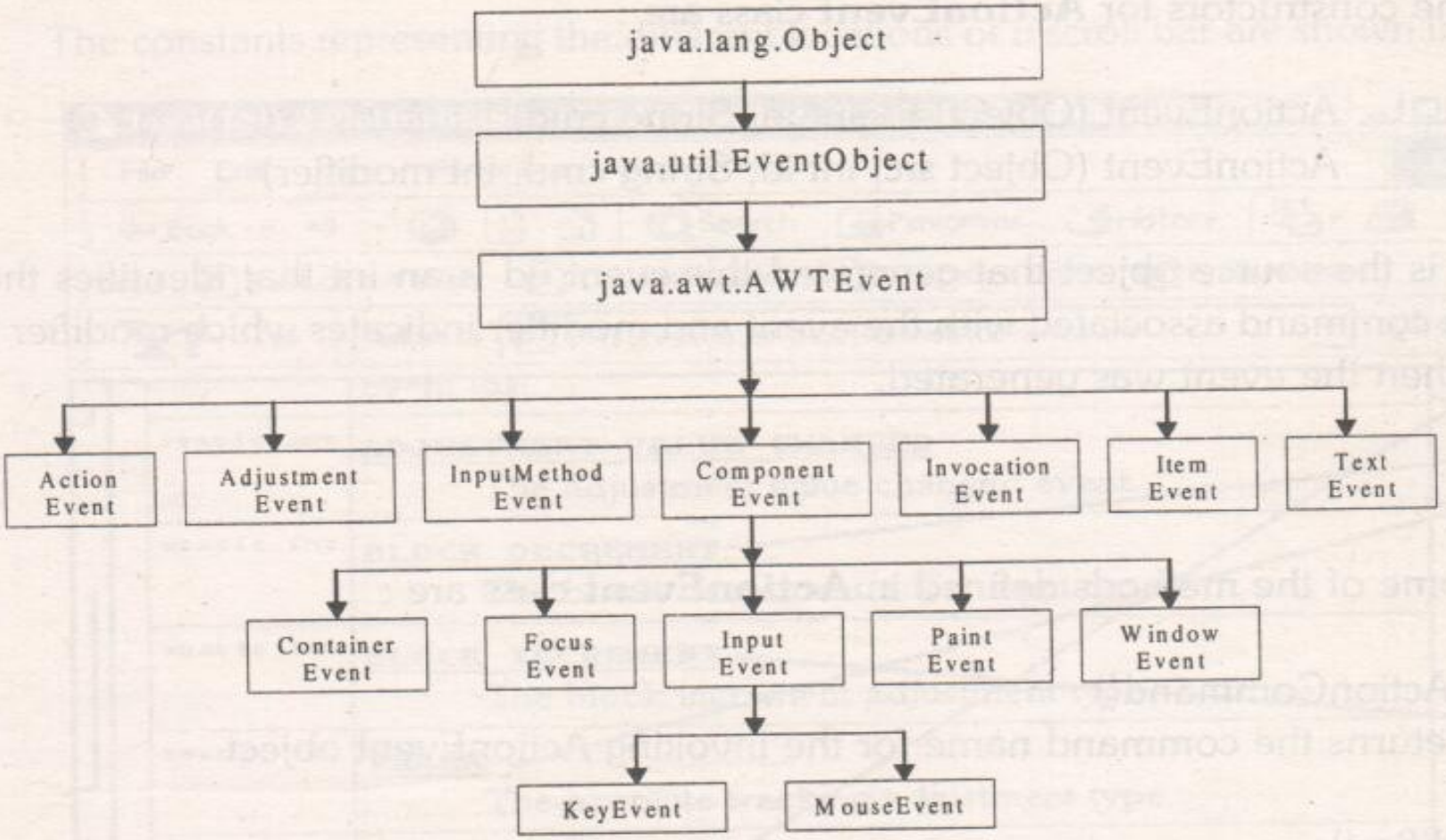


Fig.18.3 Event Class Hierarchy

UNIT - 3 Swing GUI Components

- Initially, Java introduced a package called **Abstract Window Took Kit (AWT).**
- This package contained a large number of classes and interfaces that **supported the creation of GUI.**
- Newer version of this package in Java2 is called **Swing.**
- The Swing classes are a part of the **Java Foundation Classes (JFC).**
- The Swing classes are contained in a Java extension package called **javax.**

UNIT - 3 Swing GUI Components

- The Swing classes are subclasses of **java.awt.Container** & **java.awt.Component**
- The name of the Swing class starts **with the latter J.**
- The **top – level class of swing is Jcomponent** is both a container and a component..
- GUI Components like **button, label, checkbox etc are handled in Jcomponent class.**
- GUI **components can be added on a panel window or a frame window.**
- The **frame in Swing is handled in Jframe class.**

UNIT – 3 Class Hierarchy of AWT, Jcomponent and JFrame

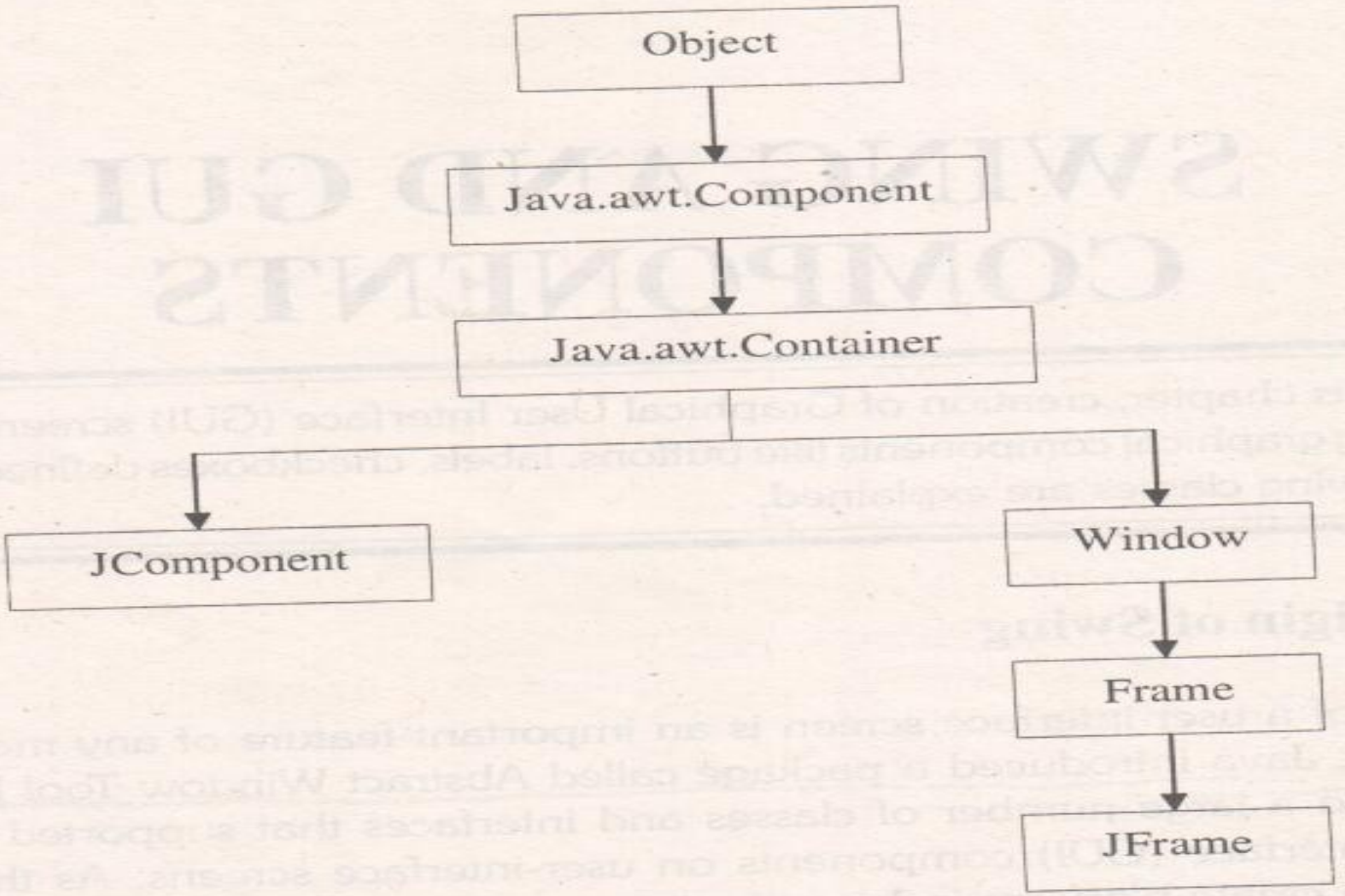
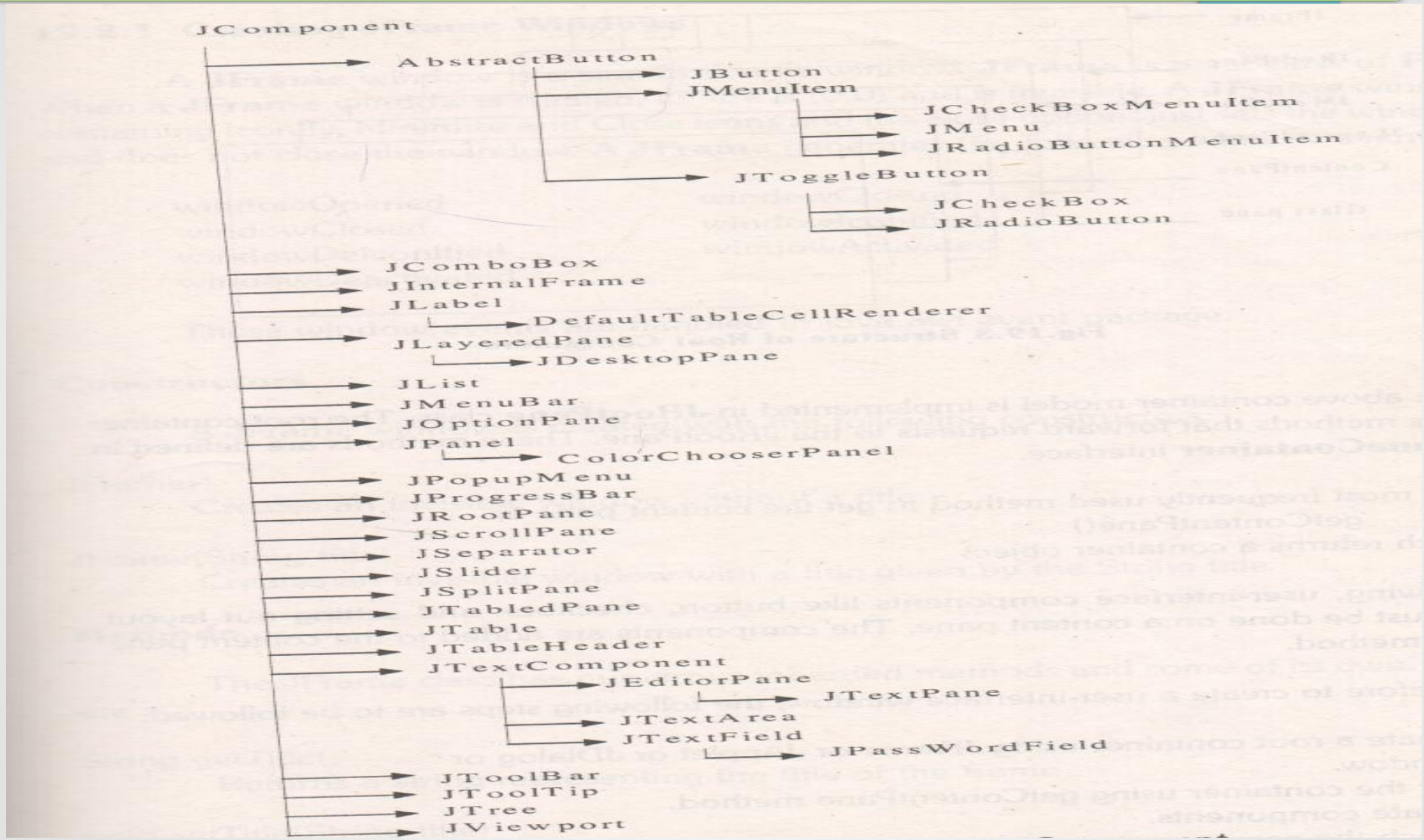


Fig.19.1 Class Hierarchy of AWT, JComponent and JFrame

UNIT – 3 Subclasses of JComponent



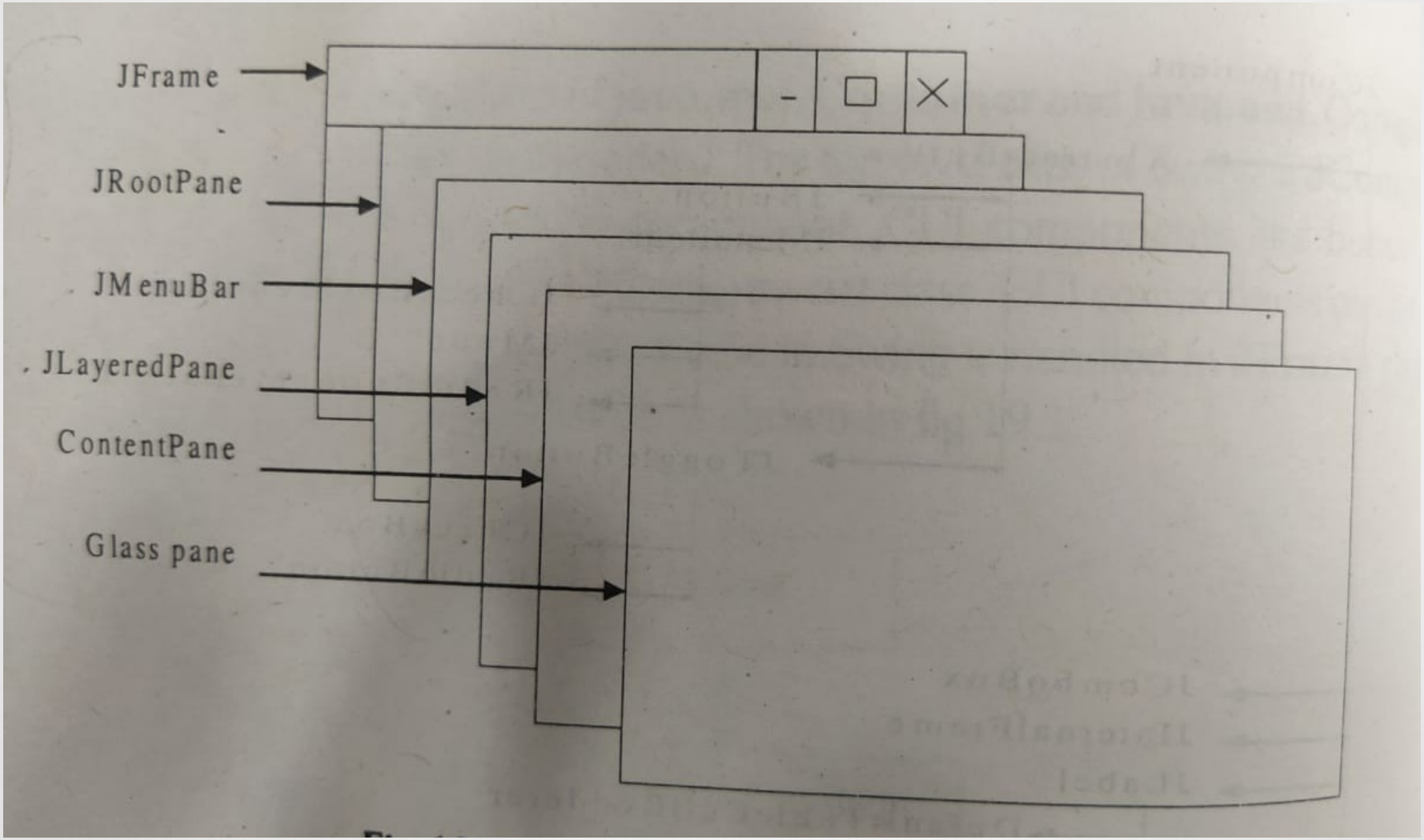
UNIT - 3 Creating Window in Swing

- A Converntional **window is created in Swing in a top-level window**.
- The top level window is supported by following classes:
  - ***JFrame***
  - ***JApplet***
  - ***JDialog***
  - ***JWindow***
- They are called **root container.**

UNIT - 3 Creating Window in Swing

- These root containers have several panes,
  - ***JrootPane***
  - ***JmenuBar***
  - ***JlayeredPane***
  - ***Content pane***
  - ***Glass pane***

UNIT - 3 Creating Window in Swing



UNIT - 3 Creating Window in Swing

- These root containers have several panes,
  - ***JrootPane***
  - ***JmenuBar***
  - ***JlayeredPane***
  - ***Content pane***
  - ***Glass pane***
- The above container model **is implemented in JrootPane class.** The root container classes have methods that forward requests to the JrootPane.
- The most frequently used method to get the content pane is :
  - ***GetContentPane()***
- The components are added to the content pane using ***add() method.***

UNIT - 3 Creating Window in Swing

- To create a user-interface window, **the following steps are to be followed:**
  - **Create a root container** using **JFrame** or **JApplet** or **JDialog** or **Jwindow**.
  - **Get the container** using **getContentPane method.**
  - **Create components**
  - **Attach the components to a container** using **add()** method

UNIT - 3 Swing - JFrame

- A Jframe window is a standard style window.
- **JFrame is a subclass of Frame class.**
- When **JFrame** is created, its size is (0,0) and is **invisible.**

UNIT - 3 Swing - JFrame

- A **JFrame** generates the following window events:
  - windowOpened
  - WindowClosed
  - WindowDeiconified
  - WindowDeactivated
  - WindowClosing
  - WindowIconfied
  - WindowActivated
- **These window events are handled in java.awt.event**

UNIT -3 Events – WindowEvent Class

Event	Purpose
WINDOW_ACTIVATED	This event occurs when the **window becomes the user's active window**
WINDOW_CLOSED	This event occurs after the **window has been closed**
WINDOW_CLOSING	This **event occurs when user attempt to close** the window
WINDOW_DEICONIFIED	This event occurs when the **window has been changed from a msinimized state to a normal state**
WINDOW_ICONIFIED	This event occurs when the **window has been changed from a normal state to a minimized state**
WINDOW_OPENED	This event occurs when the **window is made visible**

UNIT -3 Events Listeners – WindowListener

Interface	Interface Methods
WindowListener	*Void windowActivated(WindowEvent we)*
	Void windowDeactivated(WindowEvent we)
	Void windowClosed(WindowEvent we)
	Void windowClosing(WindowEvent we)
	Void windowDeiconified(WindowEvent we)
	Void windowIconified(WindowEvent we)
	Void windowOpened(WindowEvent we)

UNIT - 3 Swing - JFrame

- A **JFrame** Constructors
  - *JFrame()*
  - *JFrame(String title)*

UNIT - 3 Swing - JFrame

Event	Purpose
String getTitle()	Returns a **string representing the title** of the frame
Void setTitle(String title)	**Sets the title** of the frame to this string
Void setVisible(boolean b)	**Shows or hides this frame window**
Void setSize(int width, int height)	Sets the **size of the window to the pecified** width and height in pixels
Void setLayout(LayoutManager mgr)	**Sets the layout** manager for this container
Int getX()	Returns the X component of the window location

UNIT - 3 Swing - JFrame

Event	Purpose
Int getY()	Returns the Y component fo the window location
Void setLocation(int x, int y)	**Moves the frame window to a new location** on the scree, the top left corner of the window is specified by x and y
Int getHeight()	Returns the **current height** of the window
Int getWidth()	Returns the **current width** of the window

UNIT – 3 Swing - JFrame

- Examples :
  - Jframe1.java
  - Jframe2.java

UNIT – 3 MouseEvent Class

- A Mouse event is generated by **mouse action in a component.**
- **Two types of events**
  - **Mouse Event**
    - This event generated when a mouse button is **pressed, released, clicked, mouse enters a componenet or mouse exits a component.**
  - **Mouse Motion Event**
    - This event generated when the **mouse is moved or dragged.**

UNIT -3 Events – MouseEvent class

Constants	Purpose
MOUSE_CLICKED	This represents the mouse clicked event. This MouseEvent occurs when a **mouse button is pressed and released.**
MOUSE_ENTERED	This represents the mouse entered event. This MouseEvent occurs when a **mouse cursor enters a component's area.**
MOUSE_EXITED	This represents the mouse exited event. This MouseEvent occurs when a **mouse cursor exits a component's area.**
MOUSE_PRESSED	This represents the mouse pressed event. This MouseEvent occurs when a **mouse button is pushed down.**
MOUSE_RELEASED	This represents the mouse released event. This MouseEvent occurs when a **mouse button is released.**

UNIT -3 Events – MouseEvent class

Constants	Purpose
MOUSE_DRAGGED	This represents the mouse dragged event. This **MouseMotionEvent occurs when a mouse is dragged.**
MOUSE_MOVED	This represents the mouse moved event.This **MouseMotionEvent occurs when a mouse is moved.**

- **Constructor**
  - ***MouseEvent(Component src, int id, long when, int modifiers, int x, int y, int clickcount, boolean puptrig)***

UNIT -3 MouseEvent Class -Methods

Method Name	Purpose of Method
Int getX()	*Returns an **integer representing the x position** of the event relative to the component.*
Int getY()	Returns an **integer representing the y position** of the event relative to the component.
Void translatePoint(int x, int y)	Translates the e**vent's co-ordinates to a new position by adding x and y to the x and y of the current position**
Int clickCount()	Returns the **number of clicks associated** with this event.
Boolean isPopupTrigger()	Return **ture, if this event is the popup menu** trigger for this platform
String paramString()	Returns a **string identifying this event.**

UNIT -3 Events Listeners – MouseListener

Interface	Interface Methods
MouseListener	*Void mouseClicked(MouseEvent me)*
	Void mouseEntered(MouseEvent me)
	Void mouseExited(MouseEvent me)
	Void mousePressed(MouseEvent me)
	Void mouseReleased(MouseEvent me)

UNIT -3 Events Listeners – MouseMotionListener

Interface	Interface Methods
MouseMotionListener	*Void mouseDragged(MouseEvent me)*
	Void mouseMoved(MouseEvent me)

- Examples :
  - MouseListenerExample2.java

UNIT - 3 Swing - JButton

- The JButton is a concrete **subclass of abstract Button** which is a sub class of **Jcomponent.**
- When a button is clicked, an **ActionEvent is** created.
- The JButton class is used to **mouse press** and **mouse release events** can be precessed separately.
- **Constructors:**
  - ***JButton()***
  - ***JButton(String label)***
  - ***JButton(Icon i)***
  - ***Jbutton(String label, Icon i)***

UNIT - 3 Swing - Jbutton class hierarchy

- *Java.lang.Object*
  - *Java.awt.Component*
    - *Java.awt.Container*
      - *Javax.swing.Jcomponent*
        - *Javax.swing.AbstractButton*
          - *Javax.swing.JButton*

UNIT - 3 Swing -Jbutton Methods

Method	Description
Void addActionListener(ActionListener al)	Add the specified **action listener to receive action from this button.**
String getActionCommand()	**Returns the command name of the action** event fired by this button.
Void setText(String label)	**Sets the button's label** to the specified string
Void getText(String label)	**Returns the label of the button**

UNIT - 3 Swing -Jbutton Methods

Method	Description
Icon getIcon()	**Returns the icon** of the button
Void setIcon(Icon i)	**Sets the icon** for this button
Void removeActionEvent(ActionEvent ae)	**Remove the actionlistener**
Void processActionEvent(ActionEvent ae)	**Processesthe action events** occurring on this button
Void setRolloverIcon(Icon i)	**Sets the icon i as the rollover icon** for the button

UNIT - 3 Action Event class

- An ActionEvent is **generated when a button is pressed or menu item is selected.**
- **Constructors:**
  - ***ActionEvent(Object src, int id, String cmd)***
  - ***ActionEvent(Object src, int id, String cmd, int modifier)***

UNIT -3 Events – ActionEvent Class

Constants	Purpose
ALT_MASK	The alt modifier. An indicator that the **alt key was held down** during the event.
CTRL_MASK	The control modifier. An indicator that **the control key was held** down during the event.
META_MASK	The meta modifier. An indicator that the **meta key was held down** during the event.
SHIFT_MASK	The shift modifier. An indicator that the **shift key was held down** during the event.

UNIT -3 Action Event Class -Methods

Method Name	Purpose of Method
String getActionCommand()	*Return the command name for the invoking ActionEvent object.*
Int getModifier()	Return an int value that indicates which modifier key was pressed when the event was generated.
String paramString()	Returns a string identifying the event.

UNIT -3 Action Listeners

Interface	Interface Methods
ActionListener	***Void actionPerformed(ActionEvent ae)***

UNIT – 3 Swing -Jbutton

- Examples :
  - Jfrmbut_1.java
  - Jfrmbut_img.java

UNIT - 3 Swing - JLabel

- JLabel is a built in Java Swing class that holds text you can display within an applet.
- Jlabel class is concrete subclass of **Jcomponent.**
- A Jlabel display a **single line of read only text** in a container.
- Java.lang.Object
  - *Java.awt.Component*
    - *Java.awt.Container*
      - *Javax.swing.Jcomponent*
        - *Javax.swing.JLabel*

UNIT - 3 Jlabel Constuctors

- ***JLabel ()***
- ***JLabel (Icon image)***
- ***JLabel (Icon image, int horizontalAlignment)***
- ***JLabel (String text)***
- ***JLabel (String text, Icon icon, int horizontalAlignment)***
- ***JLabel (String text, int horizontalAlignment)***

UNIT - 3 Jlabel Align

- The Jlabel has the follwing int type constants that indicate the alignment of the label content:
  - ***JLabel.CENTER***
  - ***JLabel.LEFT***
  - ***JLabel.RIGHT***
  - ***JLabel.TOP***
  - ***JLabel.BOTTOM***

UNIT - 3 Swing -JLabel Methods

Method	Description
Int getHorizontalAlignment()	Returns the **horizontal alignmen**t for the label's content.
Int getVertical Alignment()	Retruns the **vertical alignment** for the label's content
Icon getIcon()	**Returns the icon** of the label
String getText()	**Returns the text of the label**
void setFont(Font f)	**Sets the font** for the label's text
void setText(String str)	**Sets the specified string** str as the label's content

UNIT - 3 Swing -JLabel Methods

Method	Description
Void setHorizontalAlignment(int alignment)	Sets the horizontal alignment for the label's content
Void setIcon(Icon i)	Sets the specified icon as the label's cotent
Void setVerticalAlignment(int alignment)	Sets the vertical alignment for the label's content

UNIT – 3 Swing -JLabel

- JFrmlbl_1.java

UNIT - 3 Adjustment Event class

- The Adjustment event is **generated by a scroll bar.**
- **Five types of adjustment events** are defined for the adjustment of a scroll bar.
- **Constructors:**
  - ***AdjustmentEvent(Adjustable src, int id, int type, int value)***

UNIT -3 Events – AdjustmentEvent class

constants	Purpose
BLOCK_DECREMENT	The mouse is clicked **inside the scroll bar to decrease** its value
BLOCK_INCREMENT	The mouse is clicked **inside the scroll bar to increase** its value
TRACK	The **slider is dragged**
UNIT_DECREMENT	The **button at the end of the scroll bar is clicked to decrease** its value
UNIT_INCREMENT	The **button at the end of the scroll bar is clicked to increase** it s value

UNIT -3 AdjustmentEvent Class -Methods

- The listener interfaces are defined in **java.awt.event** package

class	Methods
Adjustable getAdjustable()	*Returns the **adjustable object** where this event originated*
Int getAdjustableType()	Returns **the type of adjusstment which** caused the value changed event
Int getValue()	Returns the **current value in the adjustment event**
String paramString()	Returns a **string representing the state of this event.**

UNIT -3 Events Listeners – AdjustmentListener

- The listener interfaces are defined in **java.awt.event** package

Interface	Interface Methods
AdjustmentListener	*Void adjustmentValuechanged(AjustmentEvent ae)*

UNIT - 3 ComponentEvent class

- The Component **is an object having a graphical representation** that can be displayed on the screen and that can interact with user.
- Buttons, Checkboxes and scroll bars are the example of components.
- Component event **is generated when a component is moved, changed in size or changed in visibility.**
- **Constructors:**
  - ***ComponentEvent(Component src, int id)***

UNIT -3 Events – ComponentEvent class

constants	Purpose
COMPONENT_MOVED	This event indicates that the **component position has changed.**
COMPONENT_RESIZED	This event indicates that the **component size has changed**
COMPONENT_SHOWN	This event indicates that the **component was made visible**
COMPONENT_HIDDEN	This event indicates that the **component was made invisible**

UNIT -3 Component Event Class -Methods

- The listener interfaces are defined in **java.awt.event** package

class	Methods
Component getComponent()	*Returns the* ***originator of the event***
String paramString()	Returns the **String identifying the event**

UNIT -3 Events Listeners – ComponentListener

- The listener interfaces are defined in **java.awt.event** package

Interface	Interface Methods
ComponentListener	*Void componentHidden(ComponentEvent ce)*
	Void componentMoved(ComponentEvent ce)
	Void componentResized(ComponentEvent ce)
	Void componentShown(ComponentEvent ce)

UNIT - 3 ItemEvent class

- A semantic event indicates t**hat an item, like check box or choice is selected or deselected.**
- **Constructors:**
  - ***ItemEvent(ItemSelectable src, int id, Object item, int stateChange)***

UNIT -3 Events – ItemEvent class

constants	Purpose
DESELECTED	This state change value **indicates that an item is deselected**
ITEM_STATE_CHANGED	This event **indicates that an item's state has hanged.**
SELECTED	This state change value **indicates that an item is selected.**

UNIT -3 Item Event Class -Methods

class	Methods
ItemSelectable getItemSelectable()	***Returns the ItemSelectable object*** *that originated the event.*
Object getItem()	**Returns the item object** that was affected by the event.
Int getStateChange()	Returns an **integer that indicates whether the item was selected or deselected.**
String paramString()	Returns a string identifying the event.

UNIT -3 Events Listeners – ItemListener

Interface	Interface Methods
ItemListener	*Void itemStateChanged(ItemEvent ie)*

UNIT - 3 KeyEvent class

- The key event is generated when key is pressed, typed or released.
- **Constructors:**
  - ***KeyEvent(Component src, int id, long when, int modifier, int keycode, char keyChar)***
  - ***KeyEvent(Component src, int id, long when, int modifier, int keycode)***

UNIT -3 Events – keyEvent class

constants	Purpose
KEY_TYPED	This event is **generated when a character is entered**
KEY_PRESSED	This event is **generated when a key is pushed down**
KEY_RELEASED	This event is **generated when a key is released**
VK_0 to VK_9	Represents the keys **ASCII 0 to ASCII 9**
VK_A to VK_Z	Represents the keys **ASCII A to ASCII Z**

UNIT -3 KeyEvent Class -Methods

class	*Methods*
Int getKeyCode()	***Returns the integer code for an actual key*** *on the keyboard.*
Void setKeyCode()	***Sets the keyCode value*** *to represent a physical key*
Void setKeyChar(char keyChar)	***Sets the keychar value*** *to represent a logical character*
Char getKeyChar()	***Returns the Unicode character*** *defined for this key event.*

UNIT -3 KeyEvent Class -Methods

class	*Methods*
String getKeyText(int keyCode)	*Returns a* ***string describing the keyCode such as "Home", F1"...***
String getKeyModifierText(int modifiers)	*Returns a* ***String describing the modifier keys such as "Shift" or "Shif" + "ctrl"*** *that were held down during the event.*
Boolean isActionKey()	*Returns* ***true if the key is an action key***
String paramString()	*Returns* ***a parameter string identifying this event.***

UNIT -3 Events Listeners – KeyListener

Interface	Interface Methods
KeyListener	*Void keyPressed(KeyEvent ke)*
	Void keyReleased(KeyEvent ke)
	Void keyTyped(KeyEvent ke)

UNIT - 3 TextEvent class

- A text event is generated when the text of an object is changed.
- **Constructors:**
  - ***TextEvent(Component src, int id)***
- ***Constant:***
  - ***TEXT_VALUE_CHANGED***
    - which indicates that the object's text is changed.

UNIT - 3 TextEvent class

- **Method:**
  - ***String paramString()***
    - Returns a string identifying this text event.
- **TextListener:**
    - ***Void textValueChanged(TextEvent te)***

UNIT 3 COMPLETED