

Java AWT and Java Swing

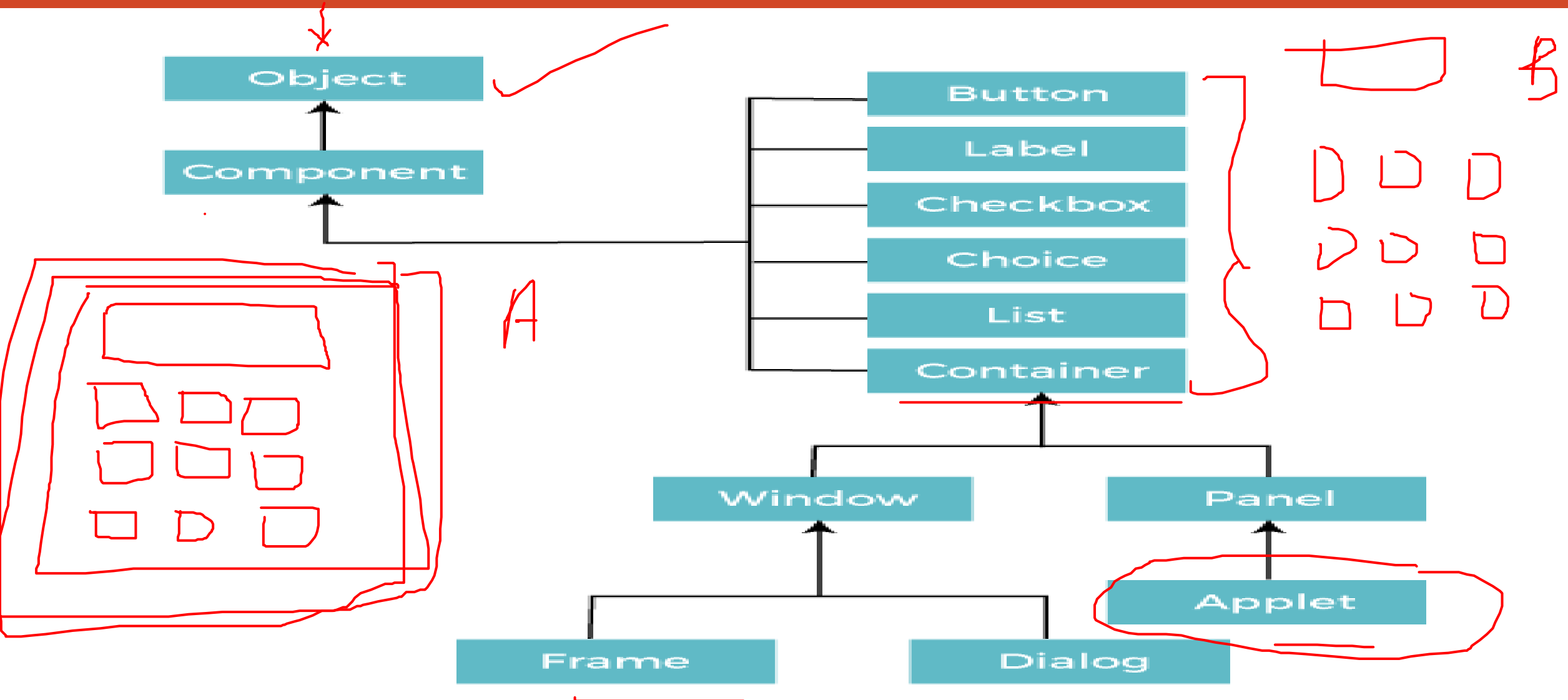
Java AWT

- Java **AWT (Abstract Window Toolkit)** is an **API to develop Graphical User Interface (GUI) or windows-based applications** in Java.
- Java AWT components are **platform-dependent** i.e. components are displayed according to the view of operating system.
- AWT is **heavy weight** i.e. its components are using the **resources of underlying operating system (OS)**.
- The **java.awt** package provides classes for **AWT API** such as **TextField, Label, TextArea, RadioButton, CheckBox, Choice, List** etc.
- Java programmer should be able to write program that **move and size windows, put components, display texts, colours and fonts** etc.

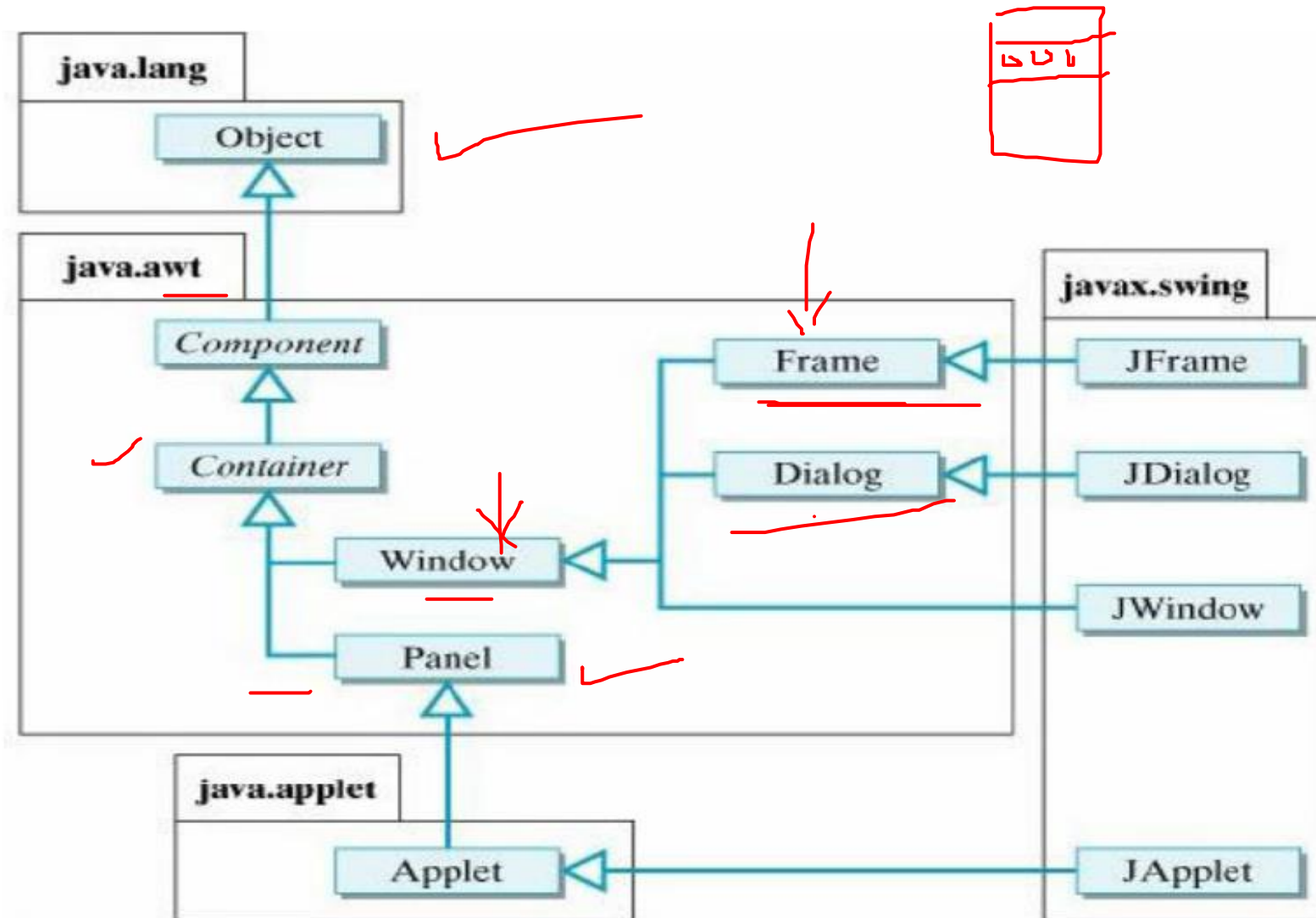
Java AWT

- **Java AWT Library**
- AWT is general purpose , multi-platform windowing library.
- AWT is a standard part of java environment
- AWT provides all basic functionality that may be needed for use in developing GUI application
- AWT library provides classes that encapsulates many useful GUI components


Java AWT Hierarchy

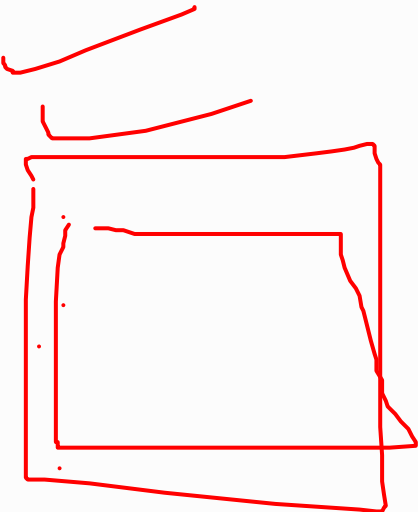


Java AWT Hierarchy

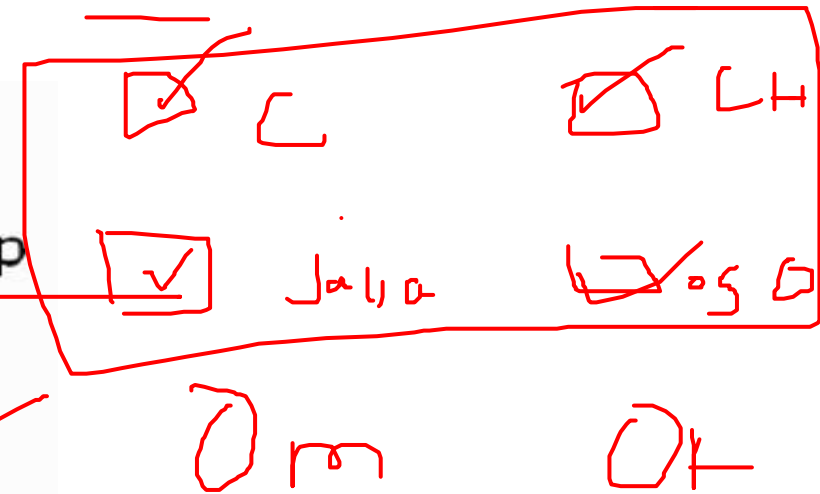


Java AWT

- Components 
- All the elements like the button, text fields, scroll bars, etc. are called components.
- In Java AWT, there are classes for each component as shown in above diagram.
- In order to place every component in a particular position on a screen, we need to add them to a container.

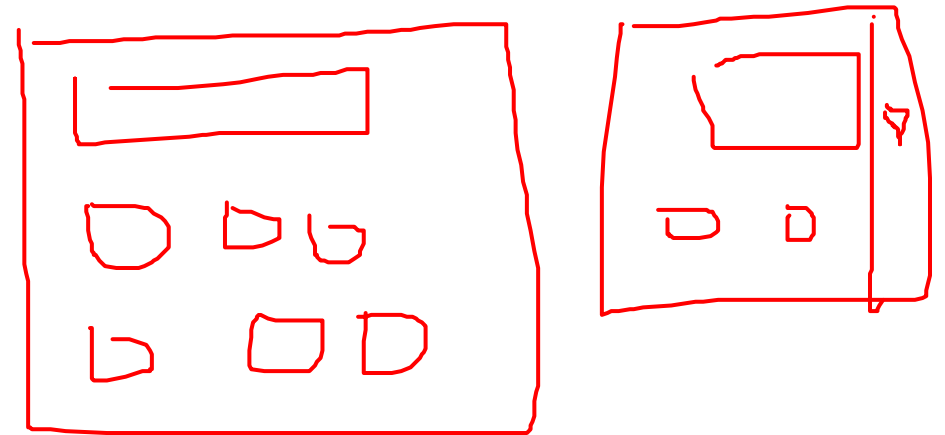
- Button
 - Checkbox
 - Choice
 - Menu
 - TextField
 - Scrollbar
- 

- Canvas ✓
- CheckboxGroup ✓
- List ✓
- Label ✓
- TextArea ✓
- ScrollPane ✓



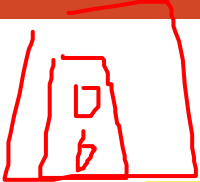
Java AWT

- Container
- The Container is a component in AWT that can contain another components like buttons, textfields, labels etc.
- The classes that extends Container class are known as container such as Frame, Panel.
- It is basically a screen where the components are placed at their specific locations. Thus it contains and controls the layout of components.
- Types of containers:
- There are three types of containers in Java AWT:
 1. Window
 2. Panel
 3. Frame



Java AWT

Window



The window is the container that have no borders and menu bars. You must use frame, dialog or another window for creating a window. We need to create an instance of Window class to create this container.

Panel

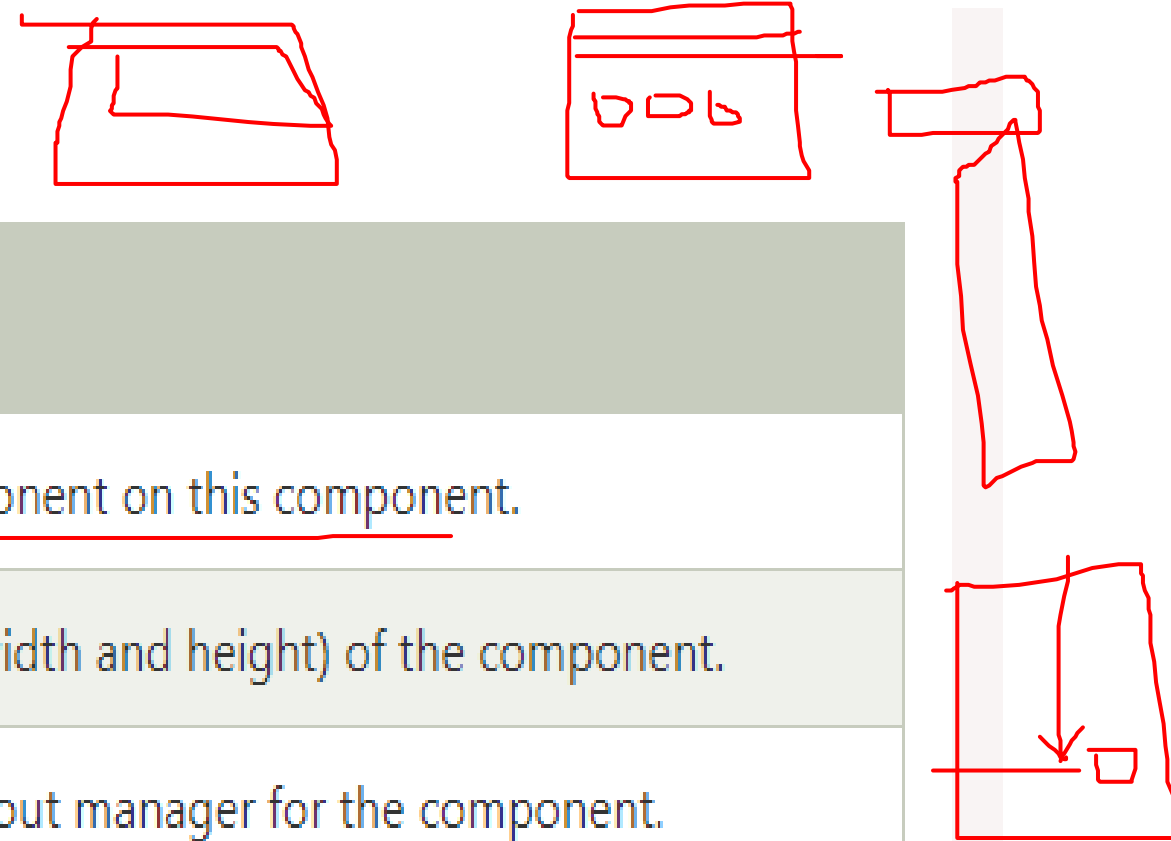
The Panel is the container that doesn't contain title bar, border or menu bar. It is generic container for holding the components. It can have other components like button, text field etc. An instance of Panel class creates a container, in which we can add components.

Frame

The Frame is the container that contain title bar and border and can have menu bars. It can have other components like button, text field, scrollbar etc. Frame is most widely used container while developing an AWT application.

Java AWT

Useful Methods of Component Class



Method	Description
<u>public void add(Component c)</u>	Inserts a <u>component on this component</u> .
<u>public void setSize(int width,int height)</u>	Sets the size (width and height) of the component.
<u>public void setLayout(LayoutManager m)</u>	Defines the layout manager for the component.
<u>public void setVisible(boolean status)</u>	Changes the visibility of the component, by default false.

Java AWT

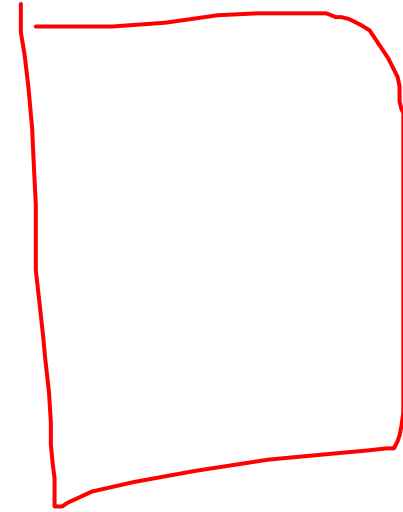
- Java AWT Example
- To create simple AWT example, you need a frame.
- There are two ways to create a GUI using Frame in AWT.

1. By extending Frame class (inheritance)

For Example: [AWTExample1.java](#)

2. By creating the object of Frame class (association)

For Example: [AWTExample2.java](#)



Java Swing

- Java Swing is a part of Java Foundation Classes (JFC) that is used to create window-based applications.
- It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.
- Unlike AWT, Java Swing provides platform-independent and lightweight components.
- The javax.swing package provides classes for java swing API such as
- JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.

Java Swing

Difference between AWT and Swing

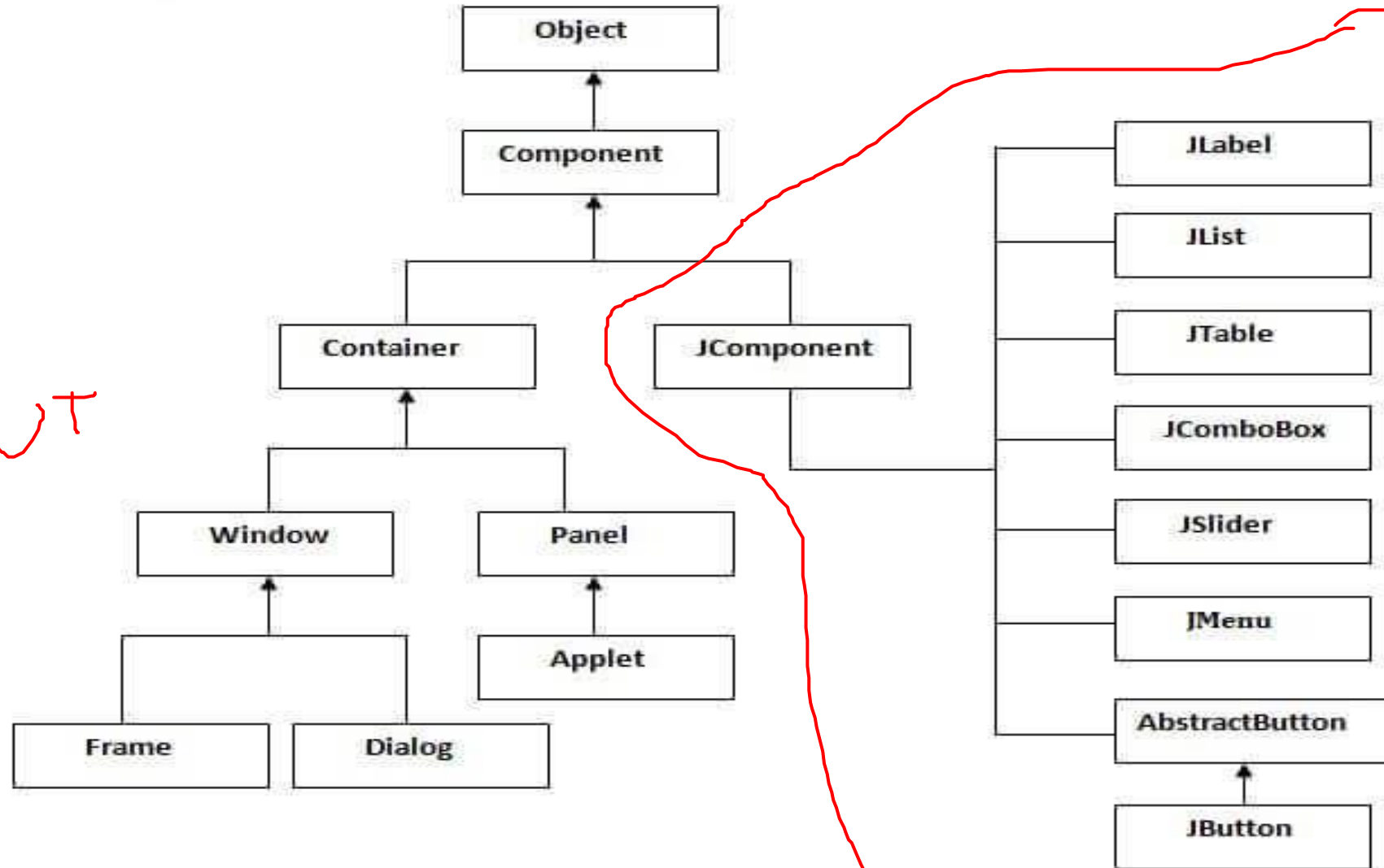
There are many differences between java awt and swing that are given below.

No.	Java AWT	Java Swing
1)	AWT components are platform-dependent .	Java swing components are platform-independent .
2)	AWT components are heavyweight .	Swing components are lightweight .
3)	AWT doesn't support pluggable look and feel .	Swing supports pluggable look and feel .
4)	AWT provides less components than Swing.	Swing provides more powerful components such as tables, lists, scrollpanes, colorchooser, tabbedpane etc.
5)	AWT doesn't follows MVC (Model View Controller) where model represents data, view represents presentation and controller acts as an interface between model and view.	Swing follows MVC .

Java Swing

- What is JFC
- The Java Foundation Classes (JFC) are a set of GUI components which simplify the development of desktop applications.

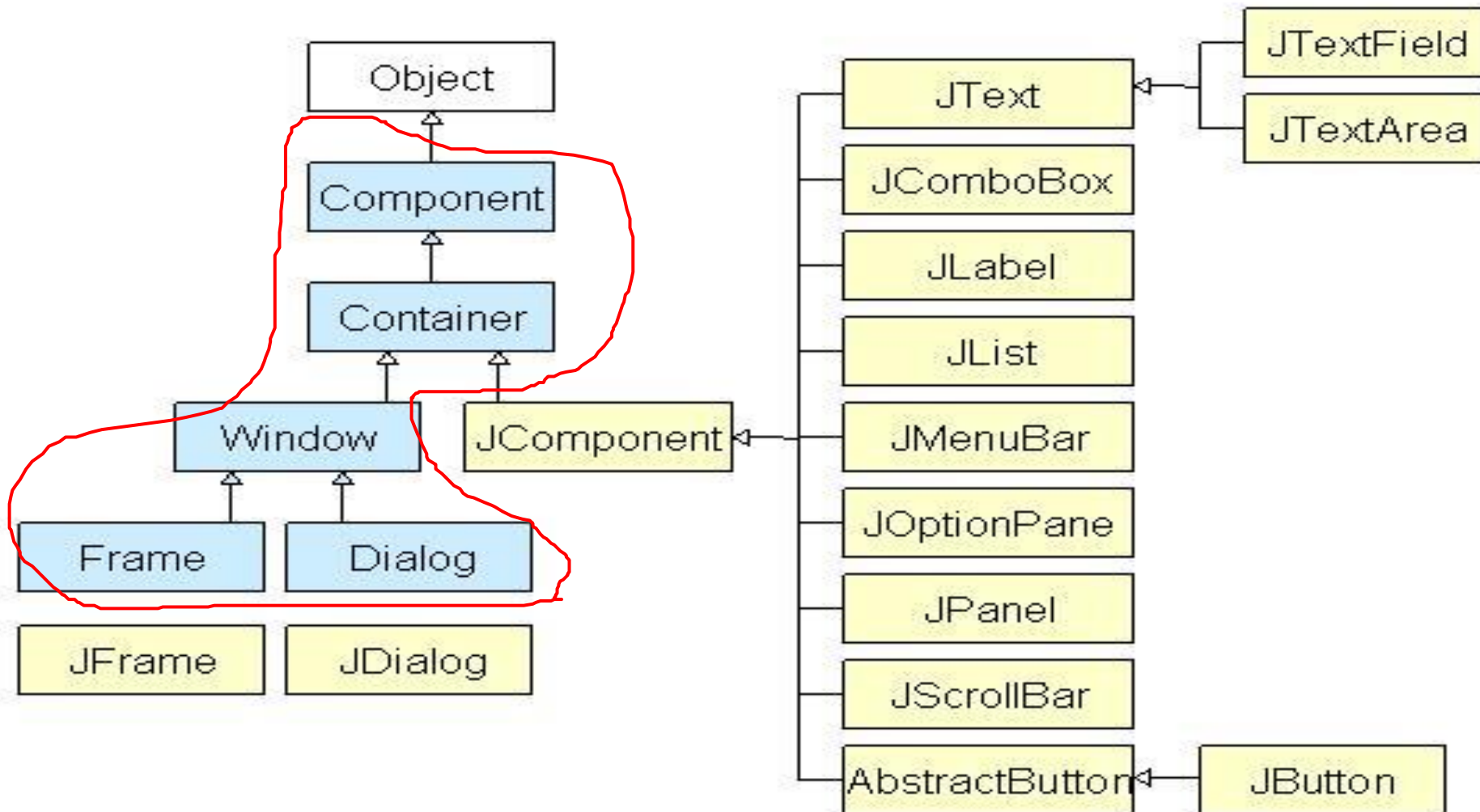
Java Swing



Swing

AwT

Java Swing



Java Swing

- Java Swing Examples
- There are two ways to create a frame:
 1. By creating the object of Frame class (association)
[Simple.java](#)
 2. By extending Frame class (inheritance)
[Simple2.java](#)
- We can write the code of swing inside the main(), constructor or any other method.

[FirstSwingExample.java](#)

Java Event Handling

- Changing the state of an object is known as an event.
- For example, click on button, dragging mouse etc.
- GUI applications are event-driven applications .
- GUI applications generate events when the user of the program interacts with the GUI.
- Underlying OS is constantly monitoring these events.
- When event occurs OS report these events to the programs that are running.
- The application will handle event using a appropriate **event handler**.
- The `java.awt.event` package provides many event classes and Listener interfaces for event handling.

Java Event Handling

- Event:
 - It is an object that describes a state change in a source.
- Event Source:
 - It is an object that generates an event.
 - A source must register listeners to receive notifications about a specific type of event.
- Event Listener :
 - It is an object that is notified when an event occurs.
 - It must be registered with a source.
 - It must implement methods to receive and process these notifications.

Java Event Handling

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

Java Event Handling

- Steps to perform Event Handling
- Following steps are required to perform event handling:
 1. Implement the Listener interface and override its methods
 2. Register the component with the Listener.
 3. Put the event handling code into one of the following places:
 - I. Within class
 - II. Other class

Java Event Handling

1. 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){}`

Java Event Handling

•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 Event Handling

Interface	Methods	Parameter	Events Generated by
ActionListner	actionPerformed	ActionEvent getActionCommand getmodifiers	Button List MenuItem TextField
Adjustment Listner	adjustmentValue Changed	AdjustmentEvent getAdjustable getAdjustmentType getvalue	Scrollbar
ItemListner	itemStateChanged	ItemEvent getItem getItemSelectable getstateChange	Checkbox CheckboxMenu Item Choice List
TextListner	textValue Changed	TextEvent	TextComponent

Java Event Handling

Interface	Methods	Parameter	Events Generated by
Component Listener	componentMoved componentHidden componentResized	ComponentEvent getComponent	Component
Container Listener	componentAdded componentRemoved	ContainerEvent getChild getContainer	Container
FocusListener	focusGained focusLost	focusEvent IsTemporary	Component
KeyListener	keyPressed keyReleased keyTyped	KeyEvent getKeyChar getKeyCode getKeyModifiersText isActionKey	Component

Java Event Handling

Interface	Methods	Parameter	Events Generated by
MouseListener	mousePressed mouseReleased mouseEntered mouseExited mouseClicked	MouseEvent getClickCount getX getY getPoint translatePoint isPopupTrigger	Component
MouseMotionListener	mouseDragged mouseMoved		Component
WindowListener	windowClosing windowOpened windowIconed windowDeiconed windowClosed windowActivated windowDeactivated	WindowEvent getWindow	Window

Java Event Handling

1. Java event handling within class (by implementing ActionListener)

- [AEvent.java](#)
- `public void setBounds(int xaxis, int yaxis, int width, int height);`
- It is used in the above example that sets the position of the component it may be button, textfield etc.

2. Java event handling by outer class

- [AEvent2.java](#)
- Swing Example
- [SwingButtonExample.java](#)