ABSTRACT WINDOW TOOLKIT [AWT]

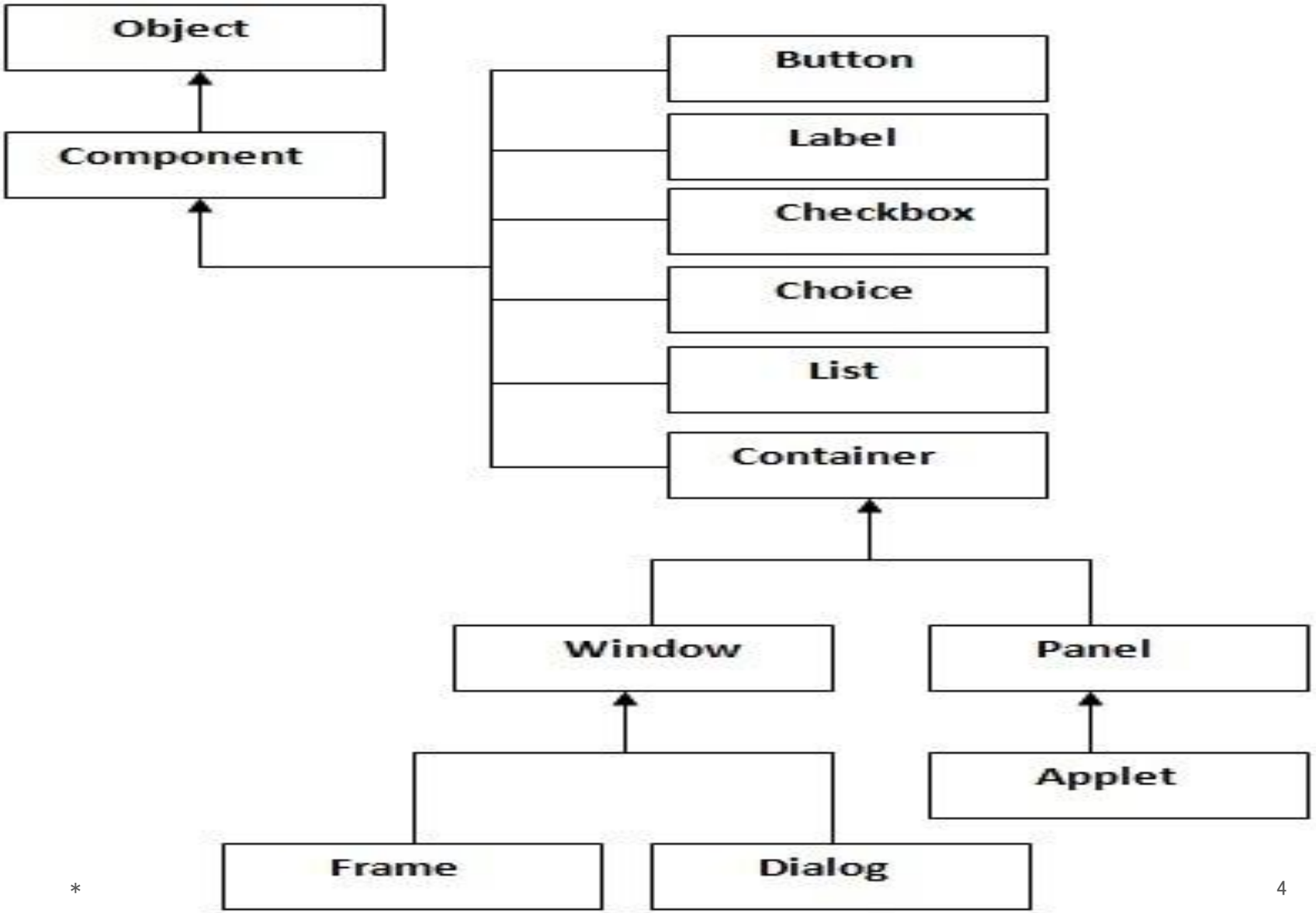
CONTENT

- AWT
- AWT Hierarchy
- Component Class
- Useful Methods in Component Class
- Subclasses of Component Class
- Types of Components
- Event Handling
- Event Handlers and their Interfaces

AWT

- Stands for Abstract Window Toolkit
- AWT is part of the Java Foundation Classes (JFC)
- Set of Application Program Interfaces(APIs)
- Used to create Graphical User Interface(GUI) or window-based applications in Java
- Platform-dependent
- Heavyweight
- import java.awt.*;
  import java.awt.event.*;

AWT HIERARCHY



COMPONENT CLASS

- Superclass of most of the displayable classes defined within the AWT.
- All user interface elements that are displayed on the screen and that interact with the user are subclasses of Components.
- Button, Checkbox, Choice, Lists, Label are the examples of Component.
- Container is the subclass of Component that holds all other Components.

USEFUL METHODS OF COMPONENT CLASS

Methods	Description
public void add(Component c)	Inserts the Component
public void setSize(int width, int height)	Sets the size of the Component
Public void setLayout(LayoutManager m)	Defines the Layout Manager
public void setVisible(boolean status)	Defines visibility of Component

SUBCLASSES OF COMPONENT CLASS

- **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, Dialog and Panel.

- **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.

- **Panel:** The Panel is the container that doesn't contain title bar and menu bars. It can have other components like button, textfield etc.

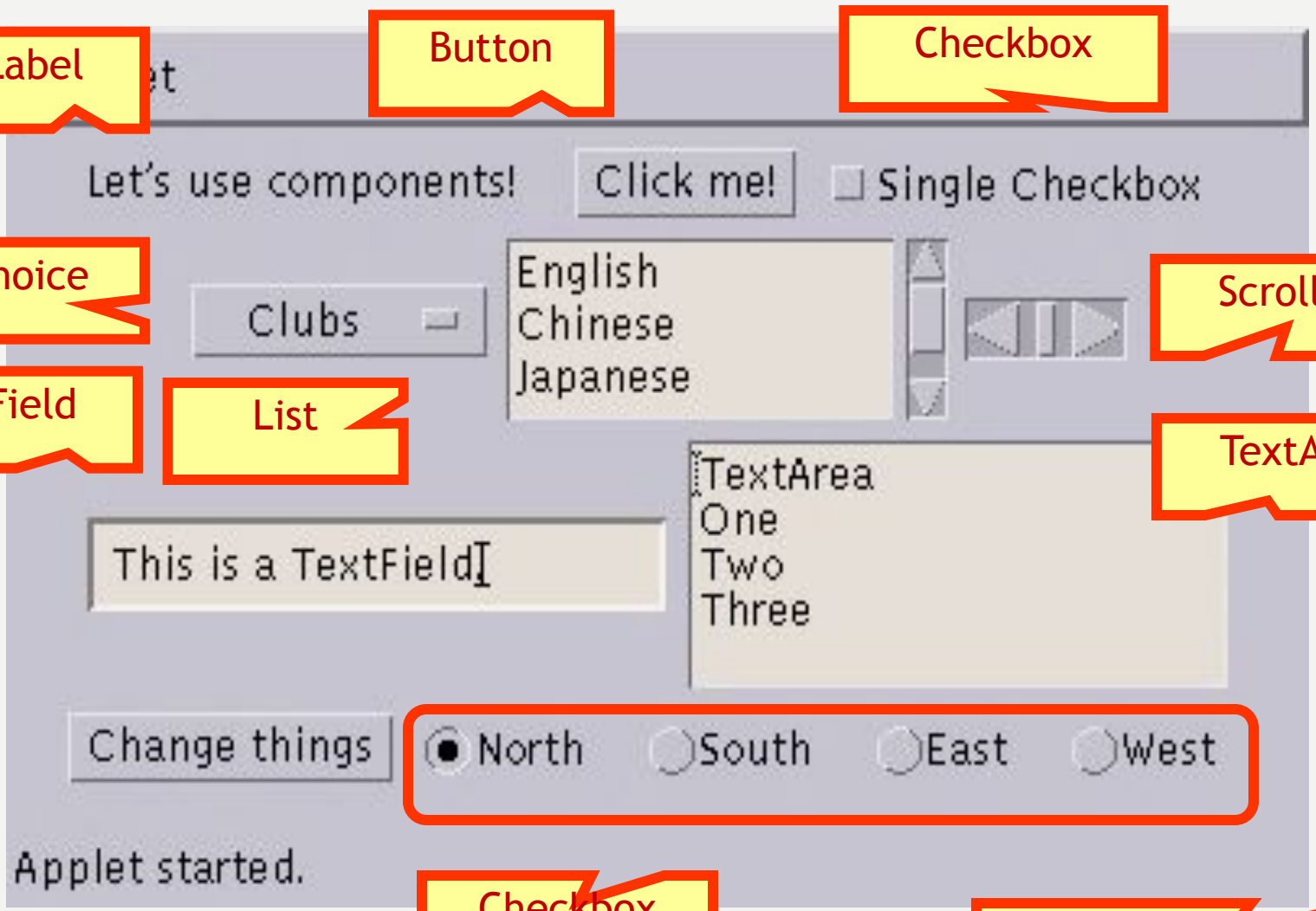
- **Frame:** The Frame is the container that contain title bar and can have menu bars. It can have other components like button, textfield etc.

JAVA LAYOUT MANAGERS

- 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:

1. java.awt.BorderLayout(by default)
2. java.awt.FlowLayout
3. java.awt.GridLayout
4. java.awt.CardLayout
5. java.awt.GridBagLayout

SOME TYPES OF COMPONENTS



DECLARATION AND INITIALIZATION OF COMPONENTS

- To declare and initialize the component, first the object for that particular component must be created.
- Syntax:

  Component_name Object = new Component_name();

- Example:

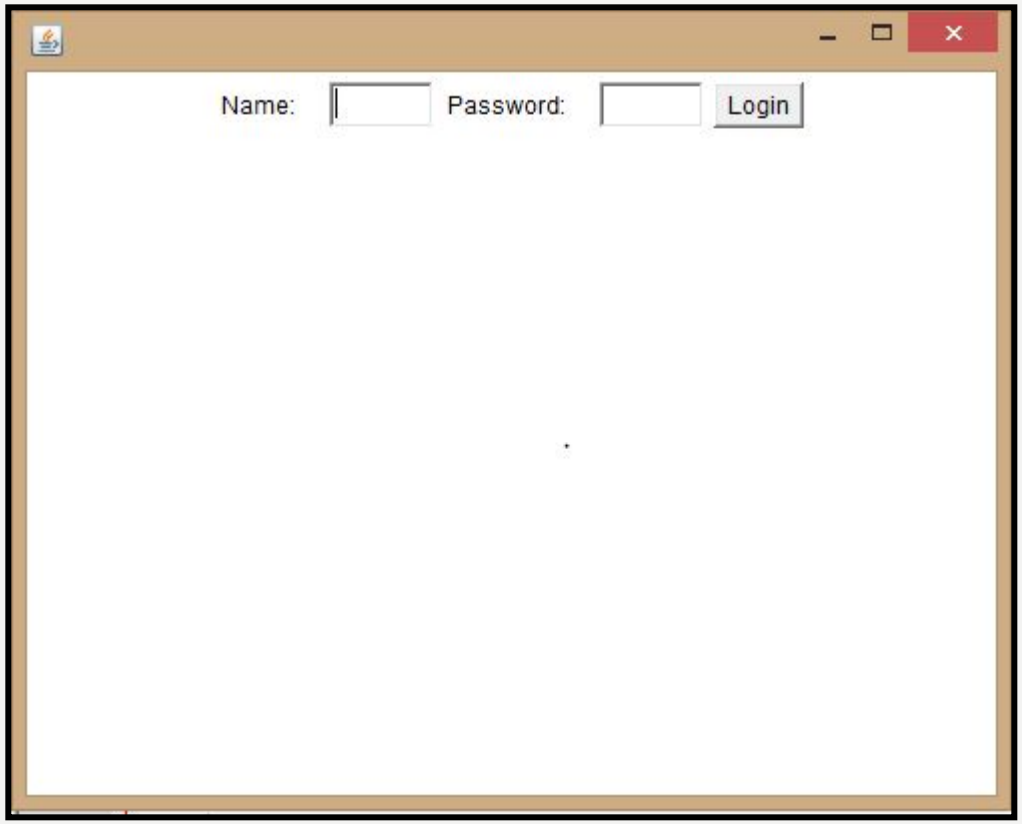
  1. Button b1 = new Button(“Click”);
  2. TextField tf1 = new TextField(20);

FRAME EXAMPLE

```
import java.awt.*;
public class Awt extends Frame
{
    Awt()
    {   FlowLayout f1=new FlowLayout();
        setLayout(f1);
        Label a1=new Label("Name:");
        TextField t1=new TextField(4);
        Label a2=new Label("Password:");
        TextField t2=new TextField(4);
        Button b=new Button("Login");
```

```
            add(a1);
        add(t1);
        add(a2);
        add(t2);
        add(b);
        setVisible(true);
        setSize(500,400);
    }
    public static void main(String[] args)
    {
        Awt f1=new Awt();
    }
}
```

Output:



OTHER COMPONENTS

Checkbox

```
Checkbox checkbox1 = new Checkbox("C++");
Checkbox checkbox1 = new Checkbox("C");
```

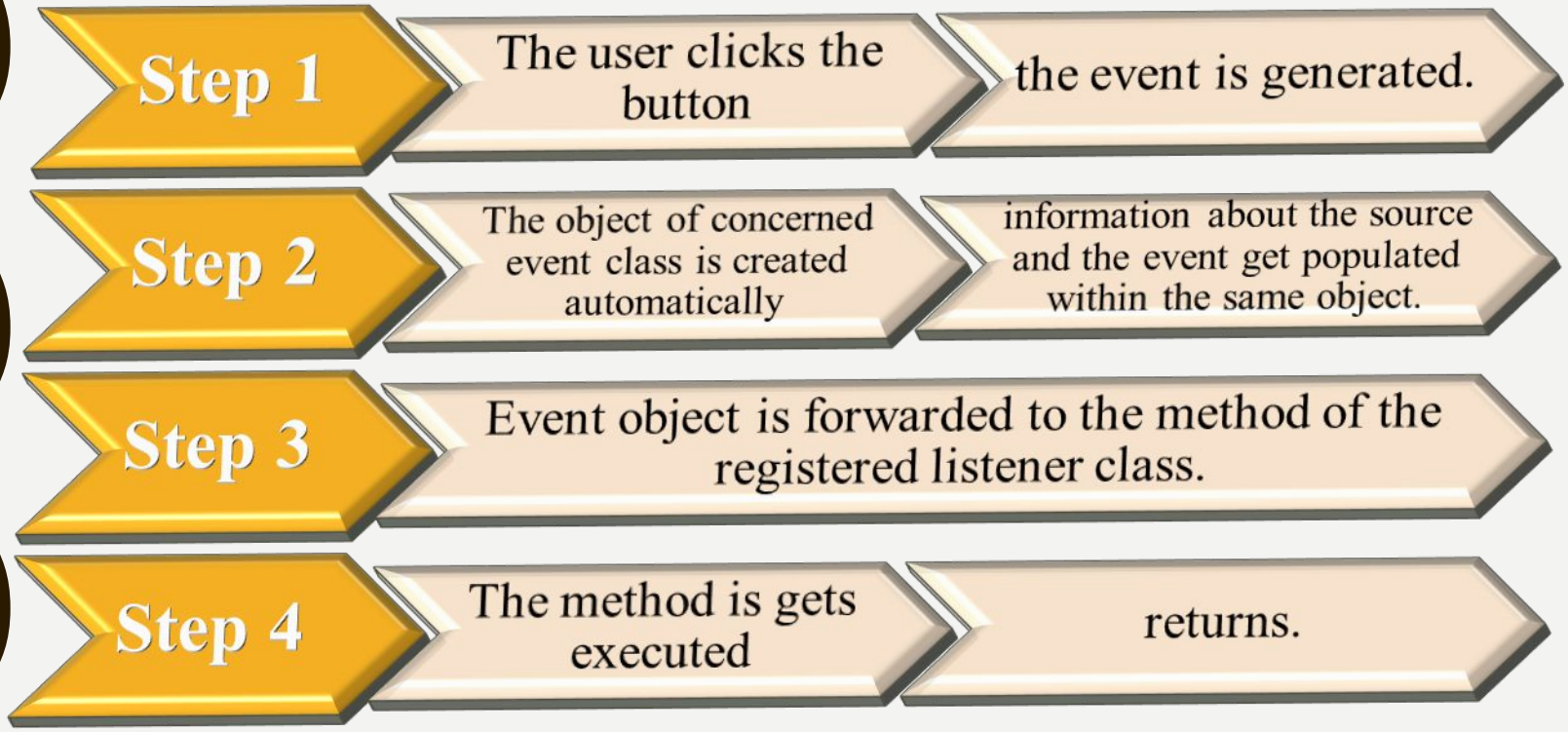
Radio button:

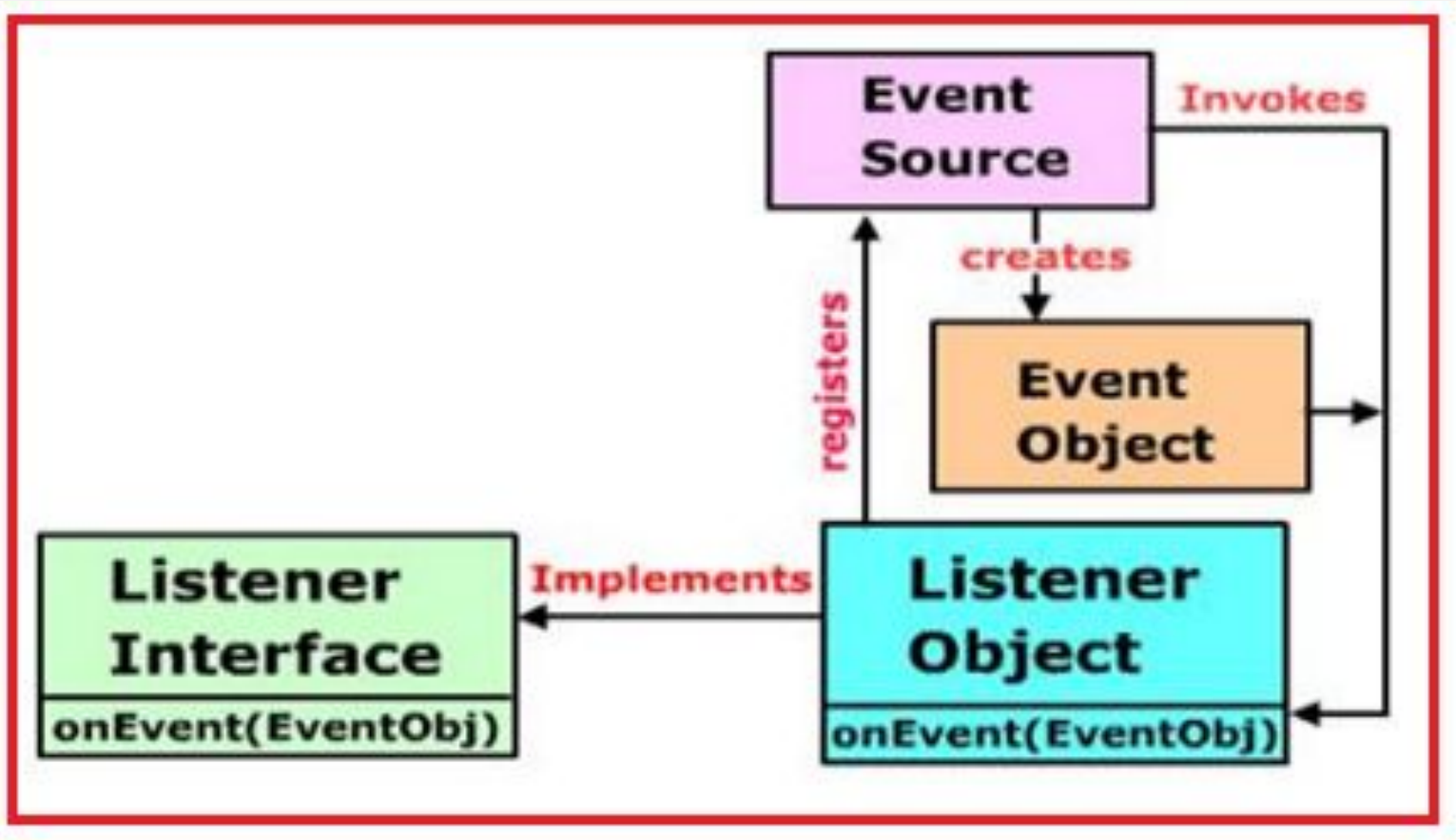
```
CheckboxGroup cbg = new CheckboxGroup();
Checkbox checkBox1 = new Checkbox("C++", cbg, false);
Checkbox checkBox2 = new Checkbox("Java", cbg,  false);
```

EVENT HANDLING

- An event is generated whenever a user clicks a mouse, presses or releases a key
- Handlers are automatically called when event takes place
- Event handlers called ‘Listeners’ are created with objects whose events need to be captured
- Event Listeners are implemented as Interfaces in Java

STEPS INVOLVED IN EVENT HANDLING:-




Event Source
Invokes
creates
registers
Event Object
Listener Interface
Implements
Listener Object
onEvent(EventObj)
onEvent(EventObj)

EVENT CLASSES AND THEIR INTERFACES

Event Class	Description	Interface
ActionEvent	Button is pressed or list item is double clicked	ActionListener
ItemEvent	When an item is selected or deselected or when the checkbox or list item is clicked	ItemListener
MouseEvent	Mouse is clicked. Pressed or released	MouseListener
KeyEvent	Input is received fro, the keyboard	KeyListener

*

EXAMPLES OF EVENTS

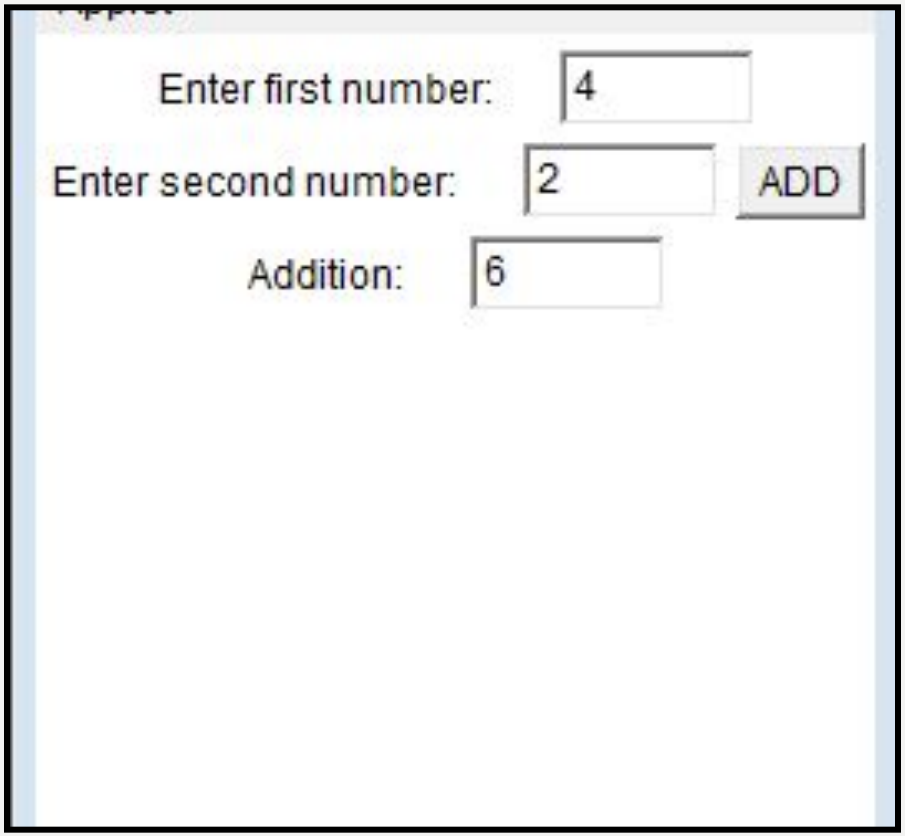
1.ActionEvent:

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
public class Event_add extends Frame implements ActionListener
{
    Label l1,l2,l3;
    TextField tf1,tf2,tf3;
    Button b1;
     Event_add()
     {
          l1=new Label("Enter first number: ");
```

```
l2=new Label("Enter second number: ");
l3=new Label("Addition: ");
tf1=new TextField(5);
tf2=new TextField(5);
tf3=new TextField(5);
b1=new Button("ADD");
add(l1);
add(tf1);
add(l2);
add(tf2);
add(b1);
add(l3);
add(tf3);
b1.addActionListener(this);
```

```
    }
     public void actionPerformed(ActionEvent ae)
     {
          Int a,b,c;
           a=Integer.parseInt(tf1.getText());
           b=Integer.parseInt(tf2.getText());
           c=a+b;
           tf3.setText(""+c);
      }
public static void main(String[] args)
   {
       Event_add e1=new Event_add ();
   }
}
```

OUTPUT:



ITEM EVENT AND ITEM LISTENER

- An event of type **ItemEvent** is generated when a source such as a checkbox is clicked to check/uncheck it or when a list item is clicked.
- A class to listen and respond to an event of type,
- **ItemEvent**, must implement an interface, **ItemListener**.

·It should implement an interface, **ItemListener** , by providing an implementation of its method:

Method	Description
public void itemStateChanged(ItemEvent e)	Invoked when an item has been selected or deselected by the user.

2.ItemEvent:

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

public class Radio extends Frame implements ItemListener
{
    Label l1;
    CheckboxGroup cg;
    Checkbox c1,c2,c3;
    TextField tf1;
```

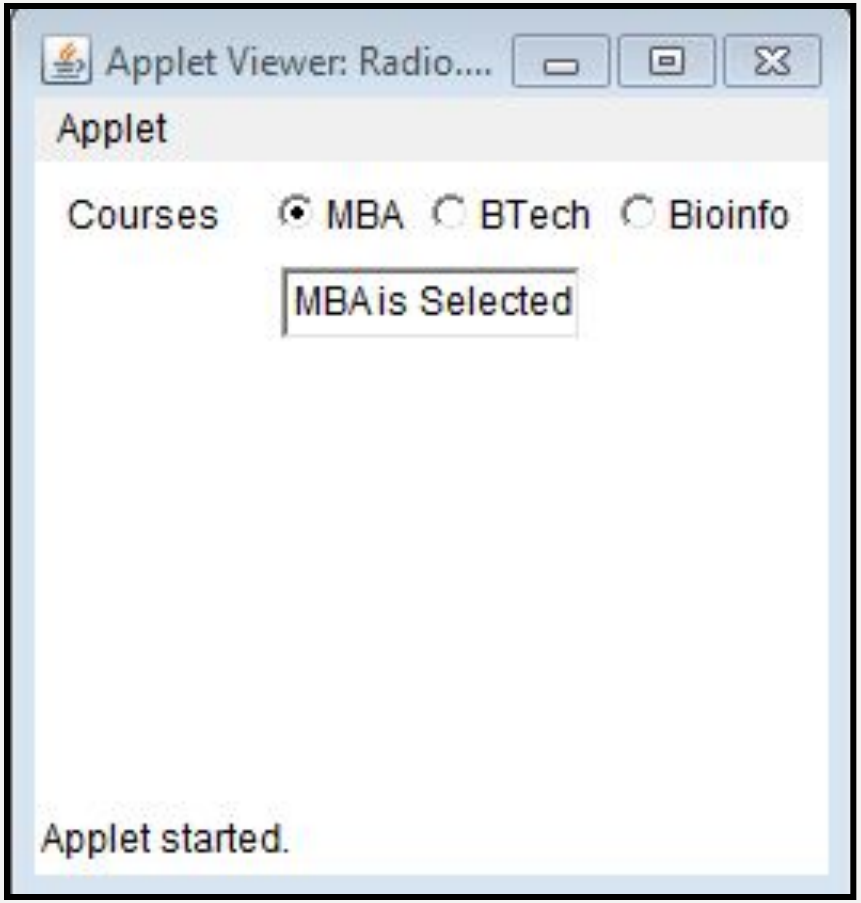
```
Radio()
{
    l1=new Label("Courses");
    cg=new CheckboxGroup();
    c1=new Checkbox("MBA",cg,false);
    c2=new Checkbox("BTech",cg,false);
    c3=new Checkbox("Bioinfo",cg,false);
    tf1=new TextField(10);
    add(l1);
    add(c1);
    add(c2);
    add(c3);
    add(tf1);
```

```
    c1.addItemListener(this);
    c2.addItemListener(this);
    c3.addItemListener(this);
}
public void itemStateChanged(ItemEvent ie)
{
    if(c1.getState()==true)
    {
        tf1.setText("MBA is Selected");
    }
    else if(c2.getState()==true)
    {
        tf1.setText("BTech is Selected");
    }
```

```
        else
        {
            tf1.setText("Bioinfo is Selected");
        }
public static void main(String[] args)
   {
       Radio r1=new Radio();
   }

    }
}
```

OUTPUT:


Applet Viewer: Radio....
Applet
Courses
MBA
BTech
Bioinfo
MBA is Selected
Applet started.

*

MOUSE EVENT AND MOUSE LISTENER

- An event of type **MouseEvent** is generated in a situations when - A mouse cursor enters a window area.
- A mouse cursor exists a window area.
- A mouse button is being pressed.
- A mouse button is released.

Method	Description
public void mouseEntered(MouseEvent me)	This method is called when a mouse cursor enters a window listening for **MouseEvent**
public void mousePressed(MouseEvent me)	This method is called when a mouse button is being pressed.
public void mouseClicked(MouseEvent me)	This method is called when a mouse button was clicked.
public void mouseReleased(MouseEvent me)	This method is called when a mouse button is released.
public void mouseExited(MouseEvent me)	This method is called when a mouse cursor exists the window.

*

KEY EVENT AND KEY LISTENER

- An event of type **KeyEvent** class is generated when a *source* such as, a **key** on the keyboard is pressed in a **textfield** or in a **textarea**.

Method	Description
public void keyPressed(KeyEvent e)	This method is called when a key is pressed on the keyboard.
public void keyReleased(KeyEvent ke)	This method is called when a key is released on the keyboard.
public void keyTyped(KeyEvent ke)	This method is called when pressing a key on they keyboard has resulted in a character.

ONLINE COMPILER FOR AWT AND SWING

https://replit.com/

thank you!