Module - V

Events & GUI Application

→ Evnt handling :

* An evnt is an obj that describes a State change in a service
(ഒരു idle state -ൽ നിന്ന് active state -ലേക്ക് മാറുന്നത്)
evnt handle ചെയ്യാൻ ഉള്ള ഒരു model → delegation evnt model.

$$\begin{tikzcd} & \text{delegation evnt model} \arrow[dl] \arrow[dr] & \\ \text{Source} & & \text{Listeners} \end{tikzcd}$$

Source

* evnt generates
* eg: txt box, btn
* എവിടെ നിന്നാണ് evnt start ചെയ്യുന്നത് അത്.

Listeners

* gnrate ചെയ്ത evnt നെ handle ചെയ്യുന്നത്
* It is a intrface.

* It is mechanism that controls an evnt & deside wht should hpn if the evnt occurs.
* This mechanism hue the code which is known as evnt handler that is excecuted when an evnt occurs.

* JAVA ~~uses~~ uses delegation evnt model to handle the evnts.
* This model defines the standard mechanism to generate & handle the evnts.

* source $\xrightarrow{\text{evnt}}$ listner.
(ഉണ്ടാക്കുന്ന evnt)
* 2 types of evnt →
① foreground (e): mouse click ചെയ്യുമ്പോ evnt ...
(systemത്തിന് [illegible] [illegible])
② background (e): comp-ന്റെ ഭാഗത്ത് നടക്കുന്ന evnt.

→ Evnt Listners =

* A listner is an obj that is notified when an evnt occurs.
* It has 2 major req.
* It must have been registered with 1/more sources to receive notification about specific type of evnts.
* It must implimnt methods to receive & process this notifications.

source-ന് listner-ലേക്ക് connection കൊടുക്കുന്നു.
(addTypeListener() → stx. for this)

* ~~addTypeListeners():~~

* Evnts:
1) key evnt → addkeyevent.
2) mouse (e) → addMouseevent.
3) Text (e) "
4) Compound (e) "
5) Input (e) "

I Key evnt:
An evnt which indicates that a key stroke occurred in a component.

II Mouse evnt:
An evnt which indicates that a mouse action occured in a component.

III Text (e):
A semantic evnt which indicates that an objects txt change.

IV Component (e):
A low level (e) indicates that a component move, change size / change visibility.

V Input (e):

The root (e) cls for all component level i/p events.

~~Delegation [illegible]~~

→ Handling Keyevnt → key press, release / type ചെയ്യുമ്പോൾ.

events

key evnt cls. — key listnr interface.

method → getkeychar().

key press / " release / " type.

(evnts implement ചെയ്യാൻ) — evnts-നെ implement ചെയ്യുന്ന cls. use ചെയ്യും.

~~key handling [illegible]~~

Key handling:

1) import package
2) Applet code
3) Extnd Applet
4) Implemnt interface
5) init method
6) Set FG, BG color.
7) register intrface with source.

Prgm

```
import java.awt.*;
  "      "   .applet.*;
/* <applet code = key width=600 height = 500>
   </applet> */
public class Key extends Applet
                 implement KeyListner() {
    int x = 40, y=40;
    msg = key events;
  public void init() {
       set background (Color.black);
       Set foreground (color.yellow);
       add Keylistener (this);
  }
 public void keypressed (key evnt ke){
        msg= "key pressed";
        SetBackGround (Color.pink);
        showstatus ("key pressed");
  }
 public keyReleased (key event ke) {
       msg = "key released";
       setBackground (color.red);
       showstatus ("key Released");
 }
 public void keyTyped (key evnt ke){
        msg = "key Typed";
        msg+=ke.getKeyChar();
        setBackground(color.black);
        showStatus("key typed");
    }
    public void paint(Graphics g) {
        g.drawString(msg, x,y);
    }
}
```

* Awt (Abstract window toolkit):
  Applet prgm ഉപയോഗിച്ച് ചെയ്യുന്ന web pg. ഉണ്ടാക്കാൻ ഒരു tool kit
* GUI → user തമ്മിൽ interact ചെയ്യുന്ന വരുന്ന screen:
- Terminologies of GUI:
  1) component → btns, txtbox, etc.
  2) window → popup ആയി വരുന്ന window. (search ചെയ്യുന്ന pg)
  3) frame → overall screen.
  4) panel → sub ÷ the pg → (nav, section)
  5) container → set of component
  6) canvas → ഒരു screen-ൽ നമ്മൾ എന്ത് വേണമെങ്കിൽ ചെയ്യാം.
     → plain screen.

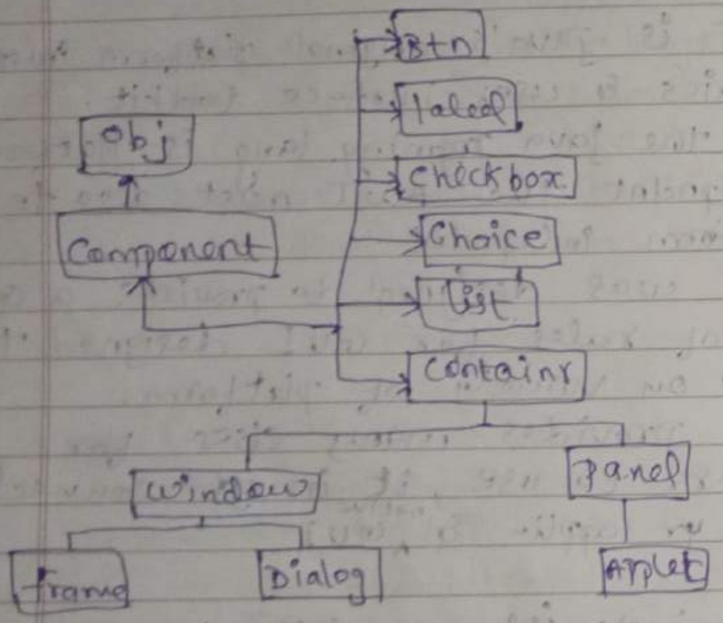
⇒ AWT [Abstract window Tool kit]:

* AWT is Java's original platform independent, graphics & user interface toolkit.
* BCz the java prgming lang is platform independent the AWT must also be platform independent.
* AWT was designed to provide a common set of rules for GUI designed that works on variety of platform.
* AWT provides many clses for prgmrs to use, it is yr connection b/w yr appli- & native GUI

→ Terminologies of GUI:

1) Component → Btn, txtbox
2) Container → set of component
3) window → pop up window
4) frame → entire pg
5) panel → page division
6) Canvas → plain screen.

* Hierarchy of AWT clses :



AWT control : (3 കാര്യങ്ങൾ ശ്രദ്ധിക്കണം for AWT prgm)

UI element	Layout	Behaviour.
web pg-ൽ കാണുന്ന ഉള്ള components	ഒരു comp എവിടെ arrange ചെയ്യണം എന്ന്.	evnt perform ചെയ്യുന്നത്.

we can add —
① Label , ② btns, choice, dialog, checkbox , list , txt field, img , scroll bar.

① Label : written txt → Label. → constructor obj
Label l1 = new label()
Syx→ Label l1 = new label ("paramtr")
↓2 methods.
get txt () → to retrieve values
set txt () → o/p set ചെയ്യാൻ ഉപയോഗിക്കുന്നു.

② If it is btn , give btn instead of label.
for btn we need evnt handler — action listener.

→ AWT Controls =

1) UI elements
2) Layout
3) Behaviour
— types of controls.

other controls [Label, btn, choice, dialog, checkbox, list, txt field, img, scroll bar

* for ~~Btn~~ label creation —
~~Button B = new~~
Label l1 = new label ();
Label l1 = new label ("label");

2 methods → getText() → To retrieve
→ SetText() → To Set the label.

* prgm Steps —
1) import packages → awt
→ applet
→ evnt.
2) Applet code.
3) class extnds Applet implemnt interface
4) init()
5) paint()
6) method of interface.

* prgm — (for Label)

```
import java.awt.*;
import java.awt.event;
import java.applet.*;
/* < applet code = LabelDemo width = 600
   height = 500>
   </applet> */
public class LabelDemo extnds Applet{
      public void init(){
           setBackground (Color.black);
           Set foreground (Color.yellow);
              Label l1= new label ("Branch");
              Label l2 = new Label ("college");
              add (l1);
              add (l2);
      }
      public void paint (Graphics g){
                g.drawString ("LabelDemo", 50,50)
      },
}
```

11* prgm — (for btn)

```
import java.awt.*;
import java.applet.*;
  "      "  .awt.event.*;
/* < applet code = Buttondemo width = 600
   height = 500> </applet> */
Public cls Buttondemo extends
   Applet implemnts Actionlistener{
   public void init () {
        Button b1 = new Button ("Red");
          "    b2 =     "     ("blue");
          "    b3 =     "     ("Green");
container{ add(b1);
           add(b2);
           add(b3);
           b1.addActionlistener(this);
           b2.     "            " ;
           b3.     "
}
```

```
public void paint (Graphics g) {
        g.drawString ("Buttondemo", 100,100)
}
public void actionperformed (ActionEvent ae) {
    String str= ae.getActionCommand();
    if (str.equals ("Red"))
    set Background (color.~~R~~ed);
    else if (str.equals ("blue");
    Set Background (color.blue);
    elseif (str.equals ("green");
    set Background (color.green);
}
```

* Prgm ⟶ (for checkbox)

method —

1) import package → awt
                  ↳ applet
2) applet code
3) Class name extends Applet
4) init method
5) Constructor → components
6) Add components to Container
7) paint method {
        g. drawString ( );
   }

Prgm —

```
import java.awt.*;
import java.applet.*;
/* < applet code = ~~[illegible]~~ checkboxDemo
   width=600 height = 590 > </applet>*/
public class checkboxDemo ~~[illegible]~~
        extends Applet {
    public class init () {
        set Background (color.black);
        set foreground (color.blue);
        Checkbox c1 = new Checkbox
                        ("football")
        Checkbox c2 =   ''     ("cricket")
           ''    c3 =   ''     ("hockey");
        add (c1);
        add (c2);
        add (c3);
    public class paint ~~[illegible]~~
        (Graphics g) ~~[illegible]~~ {
            g.drawString ("checkboxDem
                    150,150);
    }
```

12 * Prgm— (for choice)

```
import java.awt.*;
public class choiceExample {
        choice Example() {
        frame f= new frame ();
        choice c = new choice();
        c. setBound (100,100, 75,75);
        c.add ("item1");
          "        item2    ;
          "        item3    ;
          "        item4    ;
          "        item5    ;
    f.add(c);
    f. setSize (400,400);
    f.setlayout(null);
    F. set visible (true);
  }
  public static voidmain (Str args[]){
            new ChoiceExample ();
  }
```

* Choice cls used to create a popup list of items from which the user may choose.
* choice control is a form of menu. when a user clicks on it, the whole list of choices pop up & a new selection can be made.
* To add a selection to the list we hue to call <u>add ().</u> method.
* To determine which item is currently selected you may call getSelected Item or getselected Index.

→ <u>AWT color</u> =

1) java. awt package
2) java .awt .color can be declared a public cls color extends obj implemnt paint, serializable
3) 2 format →

a) <u>RGB model</u>:
* s/x —
color c= new color(int red, int green, int blue);
color c = new color(150, 250, 50);

(b) <u>HSB model</u>:
* Hue Saturation & brightness.
* s/x —
Color c= new Color (float red, float blue, float green)

→ AWT Font =

1) visible to end users
2) java.awt.font can be declared as public class font extends object implements Serializable
3) font → font name, font style, font size

eg →

```
public void paint (Graphics g) {
    font plainfont = new font ("serif",
                    font plain, 24);
    g.set font (plain.font);
    g.set color (color.red);
    g.drawString ("welcome", 50, 70);
}
```

→ working with Graphics :

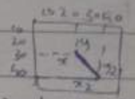
* It is an abstract cls by java AWT to draw / paint.
* It is an abstract cls, so cannot be initialized directly

↓ By using methods –

```
void paint (Graphics g)
 "   update (   "    )
```

* Str → drawline (int x, int y, int x_2, int y_2)

eg →



→ for a line draw

drawRect (int x, int y, int width, int Height)

* Prgm →

```
import Java.awt.*;
  "     "   " .event.windowAdapter;
  "     "   "     "   Event;

public myframe () {
    set visible (true);
    set size (300,200);
    addwindowListener (new windowAdapter()
    {
        p.v. window closing (window Event e)
        {
            System.exit(0)
        }
    }
    p.v. paint (Graphics g) {
        g.drawRect (100, 50, 100, 50);
    }
    public static void main (Str[] arg) {
        new. myframe ();
    }
}
```