Sample Applications and Applets in the Java Programming Language

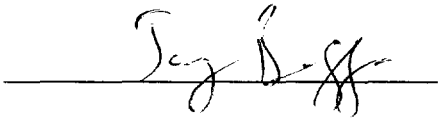
Honors Thesis (CS 499)

by

Vijay Pandurangadu

Thesis Advisor

Dr. Jay Bagga



Ball State University

Muncie, Indiana

June, 1997

Expected date of graduation: July, 1997

Abstract

The following programs are examples of the Java programming language. The first one labeled myFrame.java creates different geometric shapes. It demonstrates the use of built in graphics capabilities of Java. The second program named myCalc.java is an Java calculator applet. It contains all the basic functionality of a calculator as well as scientific functions. It shows how applets are created and how actions are handled in them. The third program named TowerHanoi.java graphically displays the solution for the Towers of Hanoi. This is also a demonstration of the built in graphic capabilities. It provides a graphical interface that allows users to choose the number of disks, select the speed at which the disks move, and allow them to pause the movement and to reset it.

The first and third programs are application in the Java language. They will run on a machine that has a Java compiler and is capable of supporting graphics. The second program is an applet. Applets are different from applications in that they are developed to be used on the Internet. They can be run on any browser that supports Java.

Program 1

Graphics application

```
/************************************************
 myFrame.java

Extends Thread and creates objects of Tower that
draw the towers on the canvas

************************************************/

import java.awt.*;
import java.lang.Math;

class myFrame extends Frame
{
        public int[] Xarray = new int[10];
         public int[] Yarray = new int[10];
         public int[] CalArray = new int[10];
         public int i;
         public int Xdis = 0;
         public int displacement;
         public int select;
         public int ArcAngle;

         mysort mySort = new mysort();

         public static void main(String[] args)
         {       Frame f = new myFrame();
                 f.resize(600,400);
                 f.setBackground(Color.darkGray);
                 f.setLayout(new FlowLayout());
                 Button button1 = new Button("Arc");
                 Button button2 = new Button("Rectangle");
                 Button button3 = new Button("Circle");
                 Button button4 = new Button("Exit");
                 f.add(button1);
                 f.add(button2);
                 f.add(button3);
                 f.add(button4);
                 f.show();
         }

         public boolean handleEvent(Event e)
         {
                 if(e.id == Event.WINDOW_DESTROY)
                 System.exit(0);
                 return super.handleEvent(e);
         }

         public void paint(Graphics g)
         {
                 if(select == 1)
                 {
                  displacement = 50;
                  for(i=0; i<4; i++)
                 {
```

```
                int ArcAngle = (int) (java.lang.Math.random()*200);
                int Xdis = 100 ;
                CalArray[i] = ArcAngle;
                System.out.println(" The angle of arc "+ i + "is " + CalArray[i]);
        }

        mySort.sort(Xarray , Yarray , CalArray );
        for(i=0; i<4; i++)
        {
                int rval = (int) (java.lang.Math.random()*100);
                int gval = (int) (java.lang.Math.random()*100);
                int bval  = (int) (java.lang.Math.random()*100);
                displacement += 100;
                g.setColor(new Color(rval,gval,rval * 2));
                g.fillArc(Xdis+displacement,250,100,100 ,0,CalArray[i]);
         }
         }

         if(select == 2)
         {
            displacement = 20;
            for(i=0;i<4;i++)
            {

                int length = (int) (java.lang.Math.random()*100);
                int height = (int) (java.lang.Math.random()*100);
                int Rarea = length * height;
                Xarray[i] = length;
                Yarray[i] = height;
                CalArray[i] = Rarea;
                System.out.println("The area of rectangle "+ i + "is " + CalArray[i]);
            }
            mySort.sort(Xarray , Yarray , CalArray);

             for(i=0; i<4; i++)
             {
                displacement += 100;
                g.setColor(new Color(CalArray[i] * 1, CalArray[i] * 2, CalArray[i + 1] * 3));
               g.fillRect(150 + displacement,100 + displacement,Xarray[i],Yarray[i]);
             }
}

if(select == 3)
{
        for(i=0;i<4;i++)
        {
                int height = (int) (java.lang.Math.random()*100);
                int Carea = (int) 3.1415 * height * height;
                Yarray[i] = height;
                CalArray[i] = Carea;
                System.out.println("The area of circle "+ i + "is " + CalArray[i]);
        }
        mySort.sort(Xarray , Yarray , CalArray);
        for(i=0; i<4; i++)
```

```
                {
                  g.setColor(new Color( CalArray[i] * 1, CalArray[i] * 2, CalArray[i + 1] * 3));
                 g.fillOval(200 - Yarray[i] / 2,200 - Yarray[i] / 2,Yarray[i],Yarray[i]);
                }
        }

    }


    public boolean action(Event e, Object arg)
    {
        if(arg.equals("Arc"))
            {
                    setBackground(Color.darkGray);
                    select = 1;
                    repaint();
            }
     if(arg.equals("Rectangle"))
            {
                    setBackground(Color.darkGray);
                    select = 2;
                    repaint();
            }
            if(arg.equals("Circle"))
            {
                    setBackground(Color.darkGray);
                    select = 3;
                    repaint();
            }
            if(arg.equals("Exit"))
                    System.exit(0);

            return true;
    }

}
```

Program 2

Calculator applet

```
/************************************************
Mysort.java

This class sorts the array according to the CalArray

************************************************/

import java.awt.*;
import java.lang.Math;

public class mysort {
        public void sort(int[] Xarray , int[] Yarray , int[] CalArray)
        {
                int i ,j;
                int temp;
                for(i=0 ;i<4 ;i++)
                {
                        for(j=i+1 ; j<4 ;j++)
                        {
                                if(CalArray[i] > CalArray[j])
                                {
                                        temp = CalArray[j];
                                        CalArray[j] = CalArray[i];
                                        CalArray[i] = temp;
                                        temp=Xarray[j];
                                        Xarray[j] = Xarray[i];
                                        Xarray[i] = temp;
                                        temp=Yarray[j];
                                        Yarray[j] = Yarray[i];
                                        Yarray[i] = temp;

                                }
                        }
                }
        }

}
```

```
/*************************************************
MyCalc.java

This program creates an calculator applet.

*************************************************/

import java.awt.*;
import java.applet.*;
import java.lang.*;

public class MyCalc extends Applet
{    public void init()
     {    setLayout(new BorderLayout());

          display = new TextField("0");
          display.setEditable(false);
                    display.setForeground(Color.red);
                    display1 = new TextField("0", 10);
                    display1.setEditable(false);
                    display1.setForeground(Color.red);
          add("North", display);
                    add("South", display1);

          Panel p = new Panel();
          p.setLayout(new GridLayout(7, 4));
          p.add(new Button("Clr"));
              p.add(new Button("M+"));
                    p.add(new Button("M-"));
                    p.add(new Button("RM"));
                    p.add(new Button("CM"));
          p.add(new Button("INV"));
          p.add(new Button("SIN"));
          p.add(new Button("COS"));
          p.add(new Button("TAN"));
          p.add(new Button("Ln"));
                    for(int i = 9; i>= 0; i--)
                    p.add(new Button(String.valueOf(i)));
          p.add(new Button("."));
          p.add(new Button("+"));
          p.add(new Button("-"));
          p.add(new Button("*"));
          p.add(new Button("/"));
                    p.add(new Button("+/-"));
          p.add(new Button("="));
                    add("Center", p);
     }

     public boolean handleEvent (Event evt)
     {    if (evt.id == Event.WINDOW_DESTROY) System.exit(0);
          return super.handleEvent(evt);
     }

     public boolean action(Event evt, Object arg)
```

```
{  if (arg instanceof String)
   {  String s = (String) arg;
      char ch = s.charAt(0);
      //Double dd = new java.lang.Double(display.getText());
      String x = display.getText();

      if(s.equals("Clr"))
         display.setText("0");

      else if(s.equals("M+"))
      {   Double dd = new java.lang.Double(x);
         mem += dd.doubleValue();
         display1.setText(" " + mem);
         start = false;
                  function = true;
      }

      else if(s.equals("M-"))
      {   Double dd = new java.lang.Double(x);
         mem -= dd.doubleValue();
         display1.setText(" "+mem);
         start = false;
                  function = true;
      }

      else if(s.equals("RM"))
      {    display1.setText(" " + mem);
                  display.setText(" " + mem);
          start = false;
                  function = true;
      }

      else if(s.equals("CM"))
      {   mem = 0.0;
         display1.setText(" 0");
         start = false;
                  function = true;
      }

      else if('0' <= ch && ch <= '9' || (ch == '.'&& (!periodUsed)))
      {
        if (ch == '.') periodUsed = true;
                  if(function)
                  {  start = true; function = false; }

        if(start) display.setText(s);
        else display.setText(display.getText() + s);
        start = false;
      }

          else if(s.equals("INV"))
                           inverse = true;

          else if(s.equals("SIN") ||
```

```
                    s.equals("TAN") ||
                    s.equals("COS") ||
                        s.equals("Ln"))
        {  Double d = new java.lang.Double(display.getText());

              if(inverse)
              {    op = "I" + s;
             inverse = false;
              }
              else
                 op = s;

              calculateTrig(d.doubleValue());
     op = "=";
              start = false;
              function = true;
         }

    else if(s.equals("+/-"))
        {  String MyString = display.getText();
              Double D = new Double(MyString);
              double d = D.doubleValue() * -1;
              display.setText(" " + d);
     op = "=";
              start = false;
    }

    else
    {  if(start)
        op =s;
              else
              {   Double d = new java.lang.Double(display.getText());
                 calculate(d.doubleValue());
                 op = s;
            start = true;
              }

         }

       }
   else return super.action(evt, arg);
   return true;
}

public void calculate(double n)
{    if (op.equals("+")) arg += n;
    else if(op.equals("-")) arg -= n;
    else if(op.equals("*")) arg *= n;
    else if(op.equals("/")) arg /= n;
    else if(op.equals("%")) arg %= n;
    else if(op.equals("=")) arg = n;
    display.setText(" " + arg);
}
```

```
        public void calculateTrig(double n)
        {   if(op.equals("SIN")) arg = java.lang.Math.sin(n);
            else if(op.equals("TAN")) arg = java.lang.Math.tan(n);
            else if(op.equals("COS")) arg = java.lang.Math.cos(n);
            else if(op.equals("Ln")) arg = java.lang.Math.log(n);
            else if(op.equals("ISIN")) arg = 1/java.lang.Math.sin(n);
            else if(op.equals("ITAN")) arg = 1/java.lang.Math.tan(n);
            else if(op.equals("ICOS")) arg = 1/java.lang.Math.cos(n);
        display.setText(" " + arg);
        }

        boolean periodUsed = false;
        boolean inverse = false;
        boolean function = false;
    private double mem = 0;
        private TextField display1;
    private TextField display;
    private double arg = 0;
    private String op = "=";
    private boolean start = true;
}
```

Program 3

Towers of Hanoi application

```
/*************************************************************
TowerHanoi.java

Implements the Towers of Hanoi with multiple threads in Java.
This class has buttons and scroll bar for control.

*************************************************************/

import java.awt.*;
import RRTower; // RRTower extends Thread
import Tower; //Tower is the calculation

class TowerHanoi extends Frame {

        RRTower rRTower;

        //Different Flags
        //private boolean inAnApplet = false;
        private boolean running = false; //flag to check if running
        private boolean paused = true; //flag to check if pused
        //private boolean autoPlayMode = true;
        private boolean ssstop = false; //flag to check if stopped

        public int noOfDisc = 1;  //defalut is 1
        public int noOfMoves = 0;
        public int sleepness = 5; //default sleep time

        public Choice c;
        public Scrollbar sbar;          //speed control
        public Canvas canvas;           //contains the image of towers
        public static TextField display2; //contains the counter

        //Different Buttons
        private Button Start ;
        private Button Stop ;
        private Button Pause ;
        private   Button Quit ;

        //Constructors
        TowerHanoi() {
                super("Tower of Hanoi");
                this.noOfDisc = 1;
                setFundamental();
        }

        TowerHanoi (int arg) {
                super("Tower of Hanoi");
                this.noOfDisc = arg;
                setFundamental();
        }

        //Creates panels containg functionality associated with
        //the towers.
        public void setFundamental() {
```

```
Panel canPanel = new Panel();
Panel p = new Panel();
Panel p1 = new Panel();
{
        setLayout(new BorderLayout());
        p1.add(new Label("# of  Disc:",Label.RIGHT));
        c = new Choice();
                c.addItem("1");
           c.addItem("2");
           c.addItem("3");
           c.addItem("4");
           c.addItem("5");
           c.addItem("6");
           c.addItem("7");
           c.addItem("8");
           c.addItem("9");
           c.addItem("10");
           c.addItem("11");
           c.addItem("12");
           c.addItem("13");
           c.addItem("14");
           c.addItem("15");
        p1.add(c);

        Start = new Button("Start");
        Start.setForeground(Color.blue);
        p1.add(Start);

        Stop = new Button("Stop");
        Stop.setForeground(Color.blue);
        p1.add(Stop);

        Pause = new Button("Pause");
        Pause.setForeground(Color.blue);
        p1.add(Pause);

        Quit = new Button("Quit");
        Quit.setForeground(Color.red);
        p1.add(Quit);

        p1.add(new Label("Num of Moves so far:",Label.RIGHT));
        display2 = new TextField("0",4);
        display2.setEditable(false);
        p1.add(display2);
}
Panel p2 = new Panel();
{
        setLayout(new BorderLayout());
        p2.add(new Button("Manual"));
        p2.add(new Label("fast",Label.RIGHT));
        sbar = new Scrollbar(Scrollbar.HORIZONTAL,
                Thread.NORM_PRIORITY,5,
                Thread.MIN_PRIORITY,Thread.MAX_PRIORITY);
```

```
                p2.add(sbar);
                p2.add(new Label("slow",Label.LEFT));
                p2.add(new Button("Auto"));
        }
        p.setLayout(new GridLayout(2,1,0,0));
        p.add(p1);
        p.add(p2);
        add("South", p);

        canvas = new Canvas();
        canvas.setBackground(Color.black);
        add("Center", canvas);
        rRTower = new RRTower(canvas, noOfDisc, sleepness, display2);
        //rRTower.setPriority(Thread.NORM_PRIORITY - 2);
}

//Handels events
public boolean handleEvent (Event event){
        if (event.id == Event.WINDOW_DESTROY) {
                //if (inAnApplet)
            //  dispose();
                System.exit(0);
        }
        if (event.target instanceof Scrollbar) {
                if (running) suspend();
                //stop();
                sleepness = sbar.getValue();
                System.out.println("you scrolled to:"+sleepness);
                rRTower.setSleepness(sleepness);
                //rRTower.setPriority(Thread.NORM_PRIORITY);
                resume();
        }
        if (ssstop) {
                stop();
                System.out.println("you tried to stop");
        }
        return super.handleEvent(event);
}

// Handels actions on the buttons
public boolean action(Event event, Object arg) {

        // The number of rings
        if (event.target instanceof Choice ) {
                if (running) stop();
                noOfDisc = c.getSelectedIndex()+1;
                rRTower.setMaxHeight(noOfDisc);
                System.out.println("you chose:"+noOfDisc);
                //rRTower.setPriority(Thread.NORM_PRIORITY);
                start();
                running = true;
        }

        // The start button
```

```
        else if (event.target instanceof Button &&
                                ((String)arg).equals("Start")) {
                System.out.println("you tried to start");
                if (running) stop();
                start();
                running = true;
        }

        // The equals button
        else if (event.target instanceof Button &&
                                ((String)arg).equals("Stop")) {
                System.out.println("you tried to stop");
                //rRTower.setPriority(Thread.MIN_PRIORITY);
                ssstop = true;
                if (running) stop();
                ssstop = false;
        }

        // The pause button
        else if (event.target instanceof Button &&
                        ((String)arg).equals("Pause")) {
                System.out.println("you tried to pause");
                if (running && !paused) suspend();
        }

        // The resume button
        else if (event.target instanceof Button &&
                        ((String)arg).equals("Resume")) {
                System.out.println("you tried to Resume");
                if (paused) resume();
        }

        // The quit button
        else if (event.target instanceof Button &&
                        (arg.equals("Quit"))) {
                //if (inAnApplet)
                //      dispose();
                System.exit(0);
        }
        else {
                return super.action(event, arg);
        }
        return true;
}

//Functions associated with the buttons

public void start(){
        Pause.setLabel("Pause");
        paused = false;
        running = true;
        display2.setText("0");
        rRTower.start();
}
```

```
	public void stop(){
		Pause.setLabel("||>>");
		paused = false;
		running = false;
		//rRTower.setPriority(Thread.NORM_PRIORITY + 3);
		rRTower.stopThread();
	}

	public void suspend(){
		Pause.setLabel("Resume");
		//paused = false;
		paused = true;
		//running = true;
		running = false;
		//rRTower.setPriority(Thread.NORM_PRIORITY);
		rRTower.suspendThread();
	}

	public void resume(){
		Pause.setLabel("Pause");
		running = true;
		paused = false;
		//rRTower.setPriority(Thread.NORM_PRIORITY);
		rRTower.resumeThread();
	}

	public static void main(String[] args) {
			Frame frame = new TowerHanoi();
			frame.resize(555,400);
			frame.show();
	}
}
```

```
/************************************************
 Tower.java

************************************************/

import java.awt.*;

class Tower{
        private int x;          //x cord of vertical line
        private int y;          //y cord of the horizontal line
        private int h;          //(length||height) of the tower
        private static int W = 5; //width of each ring
        private int[] ary;
        private int height;
        public static Canvas can;

        //Constructor
        Tower(Canvas can, int lengthOfTower, int xval, int yval) {
                this.can = can;
                x = xval;
                y = yval - 2 * W;
                h = lengthOfTower;
                ary = new int[h];
                height = -1;
        }

        Tower(Canvas can, int lengthOfTower, int xval,
                                int yval, int widthOfRing) {
                this.can = can;
                h = lengthOfTower;
                ary = new int[h];
                W = widthOfRing;
                x = xval;
                y = yval - 2 * W;
                height = -1;
        }

        public void adjustHeightInAdd(int size) {
                height++;
                ary[height] = size;
        }

        public int adjustHeightInRemoval() {
                int size = ary[height];
                ary[height] = 0;
                height--;
                return size;
        }

        // Paints the graphics
        public void paint(Graphics g){
                System.out.println("Inside paint() Tower.java\n");
                g.setColor(Color.red);
                g.drawLine(x-(h +1)*W, y, x+(h+1)*W, y);//horizontal line
```

```
                g.drawLine(x, y, x,y-2*(h+1)*W);      //vertical line
                g.setColor(Color.green);
                for(int i = 0; i<=height; i++)
                        g.fillRect(x-ary[i]*W, y-(i+1)*2*W, ary[i]*2*W, 2*W-1);
        }

        public static void main (String[] args) {
                Canvas cann = new Canvas();
                Tower tower = new Tower(cann, 125, 250,250);
        }
}
```

```
/*************************************************
RRTower.java

Extends Thread and creates objects of Tower that
draw the towers on the canvas

*************************************************/

import java.awt.*;
import Tower;

class RRTower extends Thread {
   static public Canvas can;
   static final int HEIGHT = 200;
   private int sleepness;  //sleep time
   private Thread thread;
   private int maxheight;

   public int numToDisplay;
   public static TextField display2 ;

   //declaration of towers
   Tower tower[] = new Tower [3];

   //Constructor
   RRTower (Canvas can, int noOfDisc, int sleepness, TextField p) {
                this.sleepness = sleepness;
                this.can = can;
                maxheight = noOfDisc;
                numToDisplay = 0;
                this.display2 = p;
   }

   //Excessor functions

   public void setSleepness(int sleepness) {
                this.sleepness = sleepness;
        }

   public void setMaxHeight(int no) {
                this.maxheight = no;
        }

   public int getMaxHeight() {
                return maxheight;
        }

   public int getSleepness() {
                return sleepness;
        }

   // This function is called first and creats and initalizes the
   // tower objects
```

```
public void start () {
                tower[0] = new Tower (can, maxheight, 90, HEIGHT);
                for (int i = maxheight; i > 0 ; i--)
                        tower[0].adjustHeightInAdd(i);
                tower[1] = new Tower (can, maxheight, 445, HEIGHT);
                tower[2] = new Tower (can, maxheight, 267, HEIGHT);
                //System.out.println("Initializing Towers\n");
                thread = new Thread(this);
                thread.setPriority(Thread.NORM_PRIORITY-1);
                thread.start();
}

public void run () {

        Graphics g = can.getGraphics();
   try {
    while (true) {
                System.out.println("Sleepness:"+sleepness);
                System.out.println("noOfDisc:"+maxheight);
                paint();
                g.dispose();
       Thread.sleep (sleepness*100);
                        numToDisplay =0;
       moveTower (maxheight, 0, 1, 2);

     /**
      Thread.sleep (sleepness*100);
      moveTower (maxheight, 1, 2, 0);
      Thread.sleep (sleepness*100);
      moveTower (maxheight, 2, 0, 1);
     */
     }
   } catch (Exception e) { }
}

public void paint() {
                //System.out.println("Inside RRTower paint()\n");
                Graphics g = can.getGraphics();
                g.setColor(Color.black);
                g.fillRect(0,0,550,400);
                for (int i = 0; i < 3; i++) {
                        tower[i].paint(g);
                }
        }


public void suspendThread () {
  thread.suspend();
}

public void resumeThread () {
  thread.resume();
}
```

```
public void stopThread () {
 thread.stop();
}

void moveTower (int numDisc, int source, int dest, int via) {
   if (numDisc == 1)
      moveDisc (numDisc, source, dest);
   else {
      moveTower (numDisc-1, source, via, dest);
      moveDisc (numDisc, source, dest);
      moveTower (numDisc-1, via, dest, source);
   }

}

void moveDisc (int numDics, int source, int dest) {
 int size = tower[source].adjustHeightInRemoval();
 tower[dest].adjustHeightInAdd(size);
 try {
       paint();
  Thread.sleep (sleepness*100);
 } catch (InterruptedException e) { }
                numToDisplay++;
                display2.setText(""+numToDisplay);
                System.out.println("Num of Move:"+numToDisplay);
}

public static void main(String[] arg) {
//              RRTower f = new RRTower(can, 3, 5);
        }
}
```