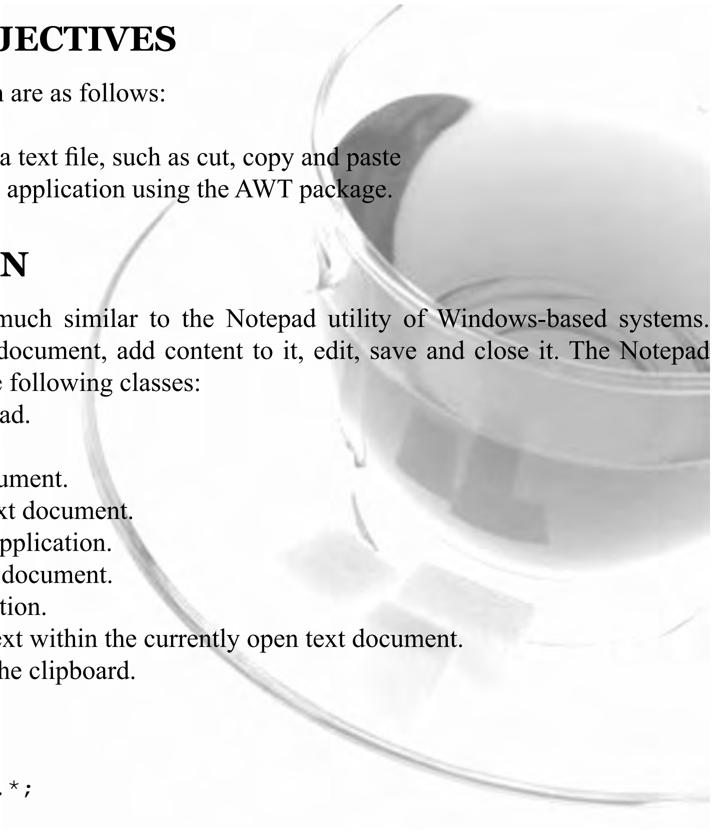
APPENDIX O

Projects

I. NOTEPAD APPLICATION



O.01 LEARNING OBJECTIVES

The objectives of the Notepad application are as follows:

- To create a new text file
- To perform common operations on a text file, such as cut, copy and paste
- To demonstrate how to build a GUI application using the AWT package.

O.02 INTRODUCTION

The Notepad application works pretty much similar to the Notepad utility of Windows-based systems. It allows the users to create a new text document, add content to it, edit, save and close it. The Notepad application program given below uses the following classes:

- **Notepad:** Realizes the actual notepad.
- **gaListener:** Exits the applications.
- **Ne_option:** Creates a new text document.
- **Ope_option:** Opens the existing text document.
- **Clos_option:** Closes the Notepad application.
- **Sav_option:** Saves the current text document.
- **Cu_option:** Performs the cut operation.
- **Cop_option:** Copies the selected text within the currently open text document.
- **Past_option:** Pastes the text from the clipboard.

Notepad Application Program

```
import java.io.*;
import java.awt.datatransfer.*;
import java.awt.event.*;
import java.awt.*;
```

```
public class Notepad extends Frame
  {

  Clipboard cBoard = getToolkit().getSystemClipboard();
  TextArea tArea;
  String fName;

Notepad()
    {

    gaListener gListen = new gaListener();
    addWindowListener(gListen);

    tArea = new TextArea();
    add(tArea);

    MenuBar mBar = new MenuBar();
    Menu fileMenu = new Menu("File");

    MenuItem nOption = new MenuItem("New");
    MenuItem oOption = new MenuItem("Open");
    MenuItem sOption = new MenuItem("Save");
    MenuItem cOption = new MenuItem("Close");

    nOption.addActionListener(new Ne_option());
    fileMenu.add(nOption);

    oOption.addActionListener(new Ope_option());
    fileMenu.add(oOption);

    sOption.addActionListener(new Sav_option());
    fileMenu.add(sOption);

    cOption.addActionListener(new Clos_option());
    fileMenu.add(cOption);

    mBar.add(fileMenu);

    Menu editMenu = new Menu("Edit");
    MenuItem cutOption = new MenuItem("Cut");
    MenuItem copyOption = new MenuItem("Copy");
    MenuItem pasteOption = new MenuItem("Paste");

    cutOption.addActionListener(new Cu_option());
    editMenu.add(cutOption);

    copyOption.addActionListener(new Cop_option());
    editMenu.add(copyOption);

    pasteOption.addActionListener(new Past_option());
    editMenu.add(pasteOption);

    mBar.add(editMenu);
    setMenuBar(mBar);

setTitle("Notepad in Java");
}

  class gaListener extends WindowAdapter
```

```
      {
      public void windowClosing(WindowEvent closeNotepad)
        {
        System.exit(0);
      }
    }

    class Ne_option implements ActionListener
      {
      public void actionPerformed(ActionEvent ne)
        {
        tArea.setText(" ");

      }
    }

    class Ope_option implements ActionListener
      {
      public void actionPerformed(ActionEvent ope)
        {
        FileDialog fDialog = new FileDialog(Notepad.this,
        "Select a text file",FileDialog.LOAD);

        fDialog.show();

        if (fDialog.getFile()!=null)
        {
        fName = fDialog.getDirectory() + fDialog.getFile();
        setTitle(fName);
        ReadFile();
        }
        tArea.requestFocus();
      }
    }

  class Clos_option implements ActionListener
      {
      public void actionPerformed(ActionEvent close_o)
        {
        System.exit(0);
      }
    }

  class Sav_option implements ActionListener
      {
      public void actionPerformed(ActionEvent sav_o)
        {
        FileDialog fDialog = new FileDialog(Notepad.this,"Save the text file
        with .txt extension",FileDialog.SAVE);

        fDialog.show();

        if (fDialog.getFile()!=null)
          {
          fName = fDialog.getDirectory() + fDialog.getFile();
          setTitle(fName);
          try
          {
      DataOutputStream dOutStream = new DataOutputStream(new FileOutputStream(f
      Name));
        String oLine = tArea.getText();
        BufferedReader bReader = new BufferedReader(new StringReader(oLine));
while((oLine = bReader.readLine())!=null)
              {
              dOutStream.writeBytes(oLine + "\r\n");
              dOutStream.close();
            }
          }
          catch(Exception ex)
            {
            System.out.print("Required file not found");
          }
          tArea.requestFocus();
        }
      }
    }
    void ReadFile()
      {
      BufferedReader br;
      StringBuffer sBuffer = new StringBuffer();
      try
        {
          br = new BufferedReader(new FileReader(fName));
          String oLine;
        while((oLine=br.readLine())!=null)
          sBuffer.append(oLine + "\n");
          tArea.setText(sBuffer.toString());
          br.close();
      }
      catch(FileNotFoundException fe) { System.out.print("Required file not
      found");}
      catch(IOException ioe){}
    }
    class Cu_option implements ActionListener
      {
      public void actionPerformed(ActionEvent cut_o)
        {
        String sText = tArea.getSelectedText();
        StringSelection sSelection = new StringSelection(sText);
        cBoard.setContents(sSelection,sSelection);
        tArea.replaceRange
        (",tArea.getSelectionStart(),tArea.getSelectionEnd());
      }
    }

    class Cop_option implements ActionListener
      {
      public void actionPerformed(ActionEvent copy_o)
        {
        String sText = tArea.getSelectedText();
        StringSelection cString = new StringSelection(sText);
        cBoard.setContents(cString,cString);
      }
    }

    class Past_option implements ActionListener
      {
      public void actionPerformed(ActionEvent paste_o)
        {
        Transferable ctransfer = cBoard.getContents(Notepad.this);
        try
          {
          String sText = (String)ctransfer.getTransferData(DataFlavor.stringFlavor);
  tArea.replaceRange(sText,tArea.getSelectionStart(),tArea.getSelectionEnd());
        }
        catch(Exception exc)
          {
          System.out.println("Not a string flavor");
        }
      }
    }

    public static void main(String args[])
      {
      Frame nFrame = new Notepad();
      nFrame.setSize(600,600);
      nFrame.setVisible(true);

    }
  }
```

O.03 RUNNING THE APPLICATION

To run the Notepad application, we need to perform the following steps:

1. Run the following command at the command prompt:

```
javac Notepad.java
```

2. After successful compilation, run the following command:

```
java Notepad
```

3. The Notepad application gets launched, as shown in Fig. 0.1:

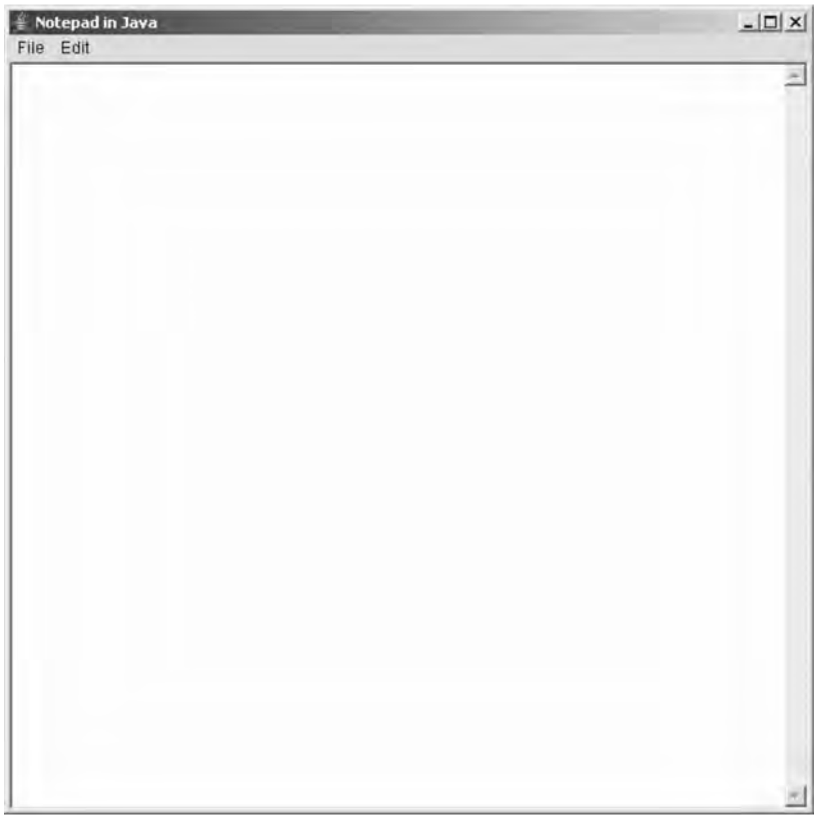


Fig. O.1 *The output of Notepad application*

Now, have can perform text-based tasks just like any other text-editor.

II. SKETCHPAD APPLICATION

O.04 LEARNING OBJECTIVES

The objectives of the Sketchpad application are as follows:

- To draw various geometric shapes (square, rectangle, circle)
- To fill different colours in geometric shapes
- To demonstrate how to build a GUI application using the AWT package

O.05 INTRODUCTION

The sketchpad application allows the users to create common geometric shapes using mouse. The Sketchpad application program given below implements methods of the following interfaces to perform sketching operations:

- ActionListener
- WindowListener
- MouseListener
- MouseMotionListener
- ItemListener

Sketchpad Application Program

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

class sketch_pad extends Frame implements ActionListener,WindowListener,Mouse
Listener,MouseMotionListener,
ItemListener
{

  String selected_shape = new String("Square");

  String selected_color = new String("Blue");

  boolean Eraser=false;

  int up_L_X, up_L_Y, W, H, sel_x1,sel_y1,sel_x2,sel_y2;

  String[] extras_list = {"Clear Canvas","Eraser"};

  String[] color_list = {"Black","Cyan","Green","Yellow","Magenta","Red","Blue"};

  String[] shape_list = {"Line","Rectangle","Square","Circle"};

  public void windowClosing(WindowEvent eve){ System.exit(0);

  }
  public void windowActivated(WindowEvent eve){}
  public void windowOpened(WindowEvent eve){}
  public void windowIconified(WindowEvent eve){}
  public void windowClosed(WindowEvent eve){}
  public void windowDeactivated(WindowEvent eve){}
  public void windowDeiconified(WindowEvent eve){}
  public void mouseMoved(MouseEvent mouse_mov_eve){}
  public void mouseClicked(MouseEvent mouse_clicked_eve){}
  public void mouseExited(MouseEvent mouse_exited_eve){}
  public void mouseEntered(MouseEvent mouse_entered_eve){}
  public void itemStateChanged(ItemEvent item_state_chng_eve){}

  public sketch_pad(String str)
  {
    super(str);
    addMouseMotionListener(this);
    addWindowListener(this);
      addMouseListener(this);
      setLayout(null);
    set_menu_items();
    setBackground(Color.white);
  }
  public void actionPerformed(ActionEvent action_performed_eve)
  {
    Graphics ga = getGraphics();
    Object s = action_performed_eve.getActionCommand();
    for (int i=0; i != color_list.length; i++)
      if (s.equals(color_list[i]))
      {
        selected_color = color_list[i];
        return;
      }
    for (int i=0; i != shape_list.length; i++)
        if (s.equals(shape_list[i]))
        {
          selected_shape = shape_list[i];
          return;
        }
    if (s.equals("Eraser"))
    {
        Eraser = true;
        return;
    }
    else if (s.equals("Clear Canvas"))
    {
        ga.clearRect(0,0,700,700);
        return;
    }
  }
  void choose_color(Graphics ga)
  {
    for (int i=0; i!= color_list.length; i++)
    {
      if (selected_color.equals(color_list[i]))
      {
        switch (i)
        {
case 0: ga.setColor(Color.black);break;
case 1: ga.setColor(Color.cyan);break;
case 2: ga.setColor(Color.green);break;
case 3: ga.setColor(Color.yellow);break;
case 4: ga.setColor(Color.magenta);break;
case 5: ga.setColor(Color.red);break;
case 6: ga.setColor(Color.blue);
        }
      }
    }
  }
  public void mouseReleased(MouseEvent mouse_reles_eve)
  {
    Graphics ga = getGraphics();
  if(Eraser)
  {
    Eraser = false;
    return;
  }
  choose_color(ga);
  sel_x2=mouse_reles_eve.getX();
  sel_y2=mouse_reles_eve.getY();

  if (selected_shape.equals("Line"))
  {
    ga.drawLine(sel_x1,sel_y1,sel_x2,sel_y2);
  }
  else if (selected_shape.equals("Circle"))
  {
    draw_selected_shape(ga,"Circle");
  }

  else if (selected_shape.equals("Square"))
  {
    draw_selected_shape(ga,"Square");
  }
  else if (selected_shape.equals("Rectangle"))
  {
    draw_selected_shape(ga,"Rectangle");
  }
  ga.setColor(Color.yellow);
  ga.drawString(".",sel_x1,sel_y1);
  ga.setColor(Color.black);
}

void draw_selected_shape(Graphics ga,String sel_shape)
{
  up_L_X = Math.min(sel_x1,sel_x2);
  up_L_Y = Math.min(sel_y1,sel_y2);
  W = Math.abs(sel_x1-sel_x2);
  H = Math.abs(sel_y1-sel_y2);
  if (sel_shape.equals("Square") )
    ga.fillRect(up_L_X,up_L_Y,W,W);
    else if (sel_shape.equals("Rectangle"))
    ga.fillRect(up_L_X,up_L_Y,W,H);
  else if (sel_shape.equals("Circle") )
    ga.fillOval(up_L_X,up_L_Y,W,W);
}

public void mouseDragged(MouseEvent mouse_drag_eve)
{
  Graphics ga = getGraphics();
  sel_x2=mouse_drag_eve.getX();
  sel_y2=mouse_drag_eve.getY();
  if (Eraser)
  {
    ga.setColor(Color.white);
```

```
      ga.fillRect(sel_x2,sel_y2,10,10);
    }
  }

  public void mousePressed(MouseEvent mouse_press_eve)
  {
    if (Eraser) return;
    up_L_X=0; up_L_Y=0; W=0; H=0;
    sel_x1=mouse_press_eve.getX();
    sel_y1=mouse_press_eve.getY();
    Graphics ga = getGraphics();
    ga.drawString(".",sel_x1,sel_y1);
  }

void set_menu_items()
  {
    MenuBar mBar = new MenuBar();

      Menu menu_sh = new Menu("Shapes");
    for (int i=0; i != shape_list.length; i++)
      menu_sh.add(shape_list[i]);
    mBar.add(menu_sh);
    menu_sh.addActionListener(this);

      Menu menu_col = new Menu("Colors");
    for (int i=0; i != color_list.length; i++)
      menu_col.add(color_list[i]);
    mBar.add(menu_col);
    menu_col.addActionListener(this);

    Menu Ex = new Menu("Extras");
    for (int i=0; i != extras_list.length; i++)
      Ex.add(extras_list[i]);
    mBar.add(Ex);
    Ex.addActionListener(this);
  setMenuBar(mBar);
  }
}

class Sk_pad
{
  public static void main(String[] args)
  {
      sketch_pad draw_win = new sketch_pad("Sketchpad in Java");
      draw_win.setSize(700,700);
      draw_win.setVisible(true);
  }
}
```

O.06 RUNNING THE APPLICATION

To run the Sketchpad application, we need to perform the following steps:

1. Run the following command at the command prompt:

```
javac Sk_pad.java
```

2. After successful compilation, run the following command:

```
java Sk_pad
```

3. The sketchpad application gets launched, as shown in Figure O.2:

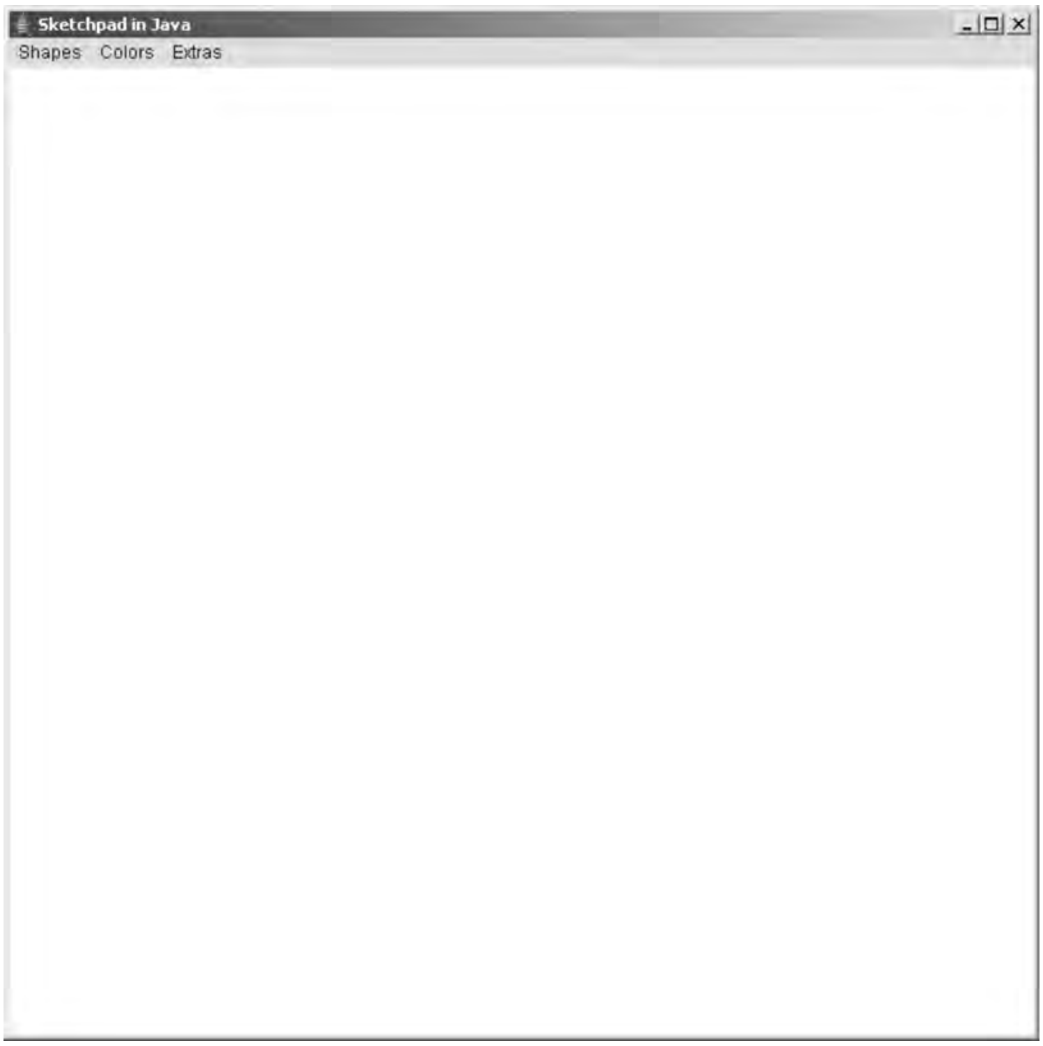


Fig. O.2 *The output of Sketchpad application*

Now, we can work on the Sketchpad application by using mouse to sketch different geometric shapes, as shown in Figure O.3:

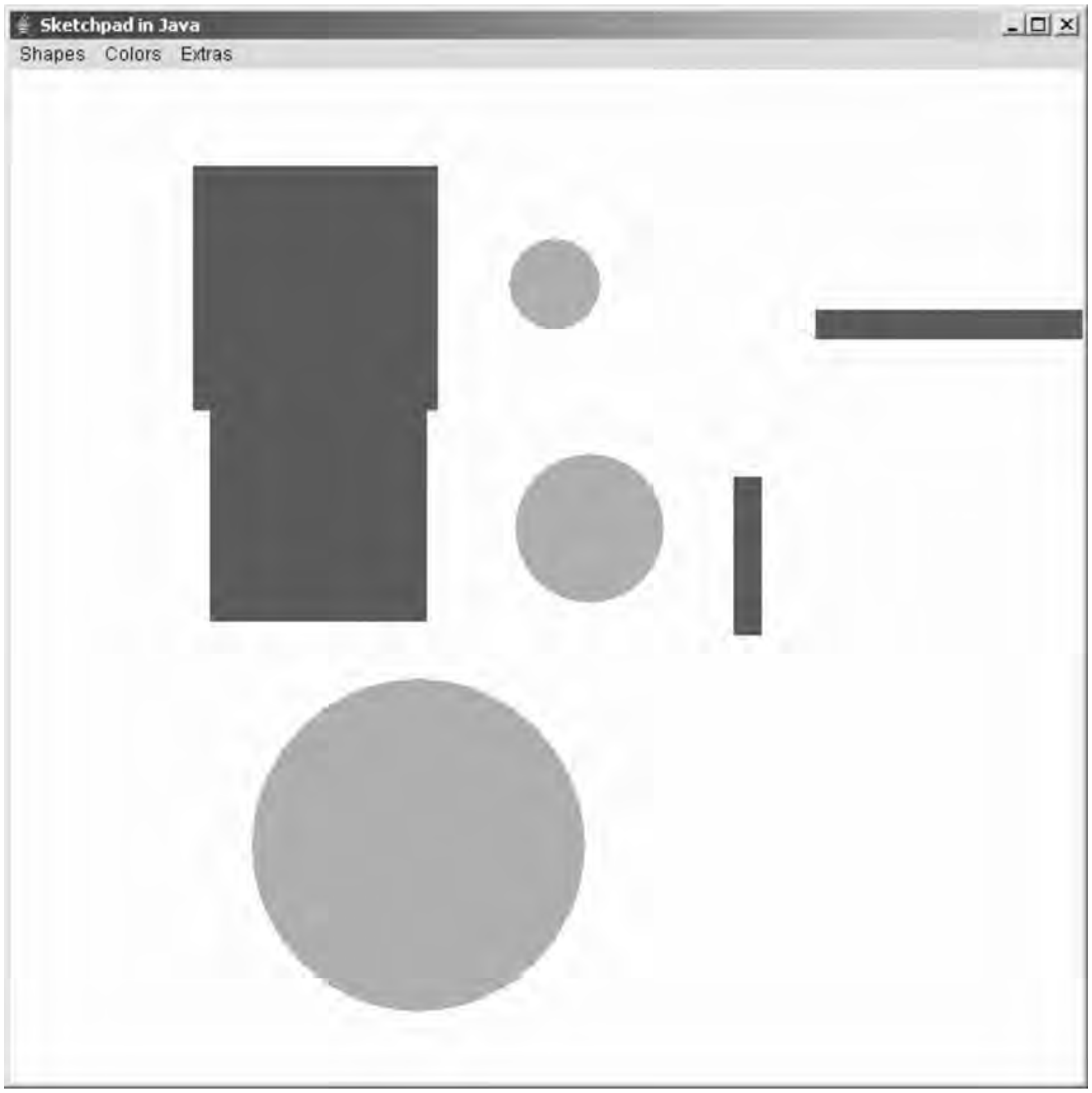


Fig. O.3 *Drawing shapes in Sketchpad*