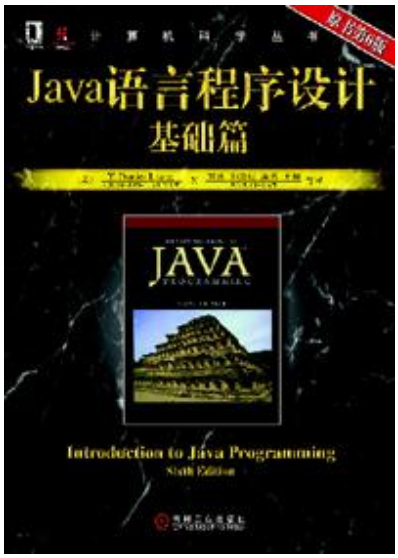
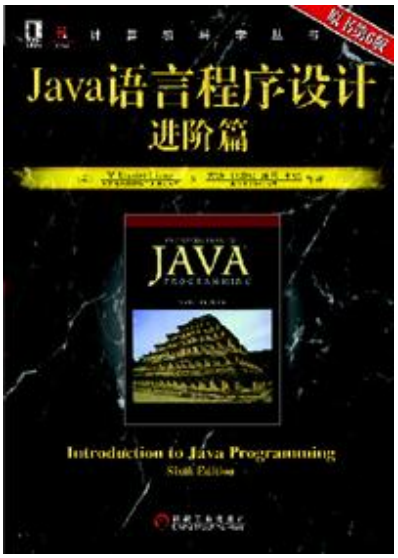


&

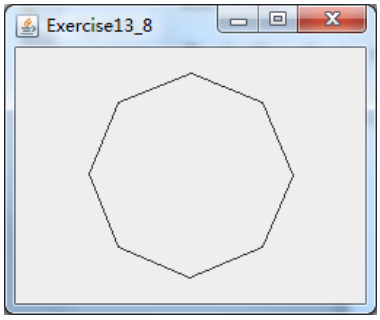


[编程练习题答案]Java 语言程序设计（基础篇）（进阶篇）第 13、14、15、24、25、26、28、29、30、31 章部分习题答案，奇数题和部分双数题是我自己做的，在此和大家分享(^_^～

Chapter 13	Exercise 13.2 / 13.8 / 13.11

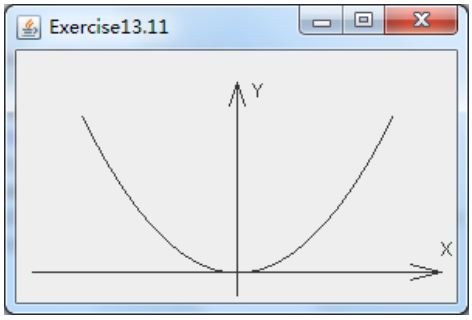

```
import javax.swing.*;
import java.awt.*;

public class Exercise13_2 extends JFrame {
  // Create two buttons
  private OvalButton jbtOk = new OvalButton("OK");
  private OvalButton jbtCancel = new OvalButton("Cancel");

  /** Default constructor */
  public Exercise13_2() {
    // Set the window title
    setTitle("Exercise13_2");

    // Set FlowLayout manager to arrange the components
```

```
    // inside the frame
    getContentPane().setLayout(new FlowLayout());

    // Add buttons to the frame
    getContentPane().add(jbtOk);
    getContentPane().add(jbtCancel);
  }

  /** Main method */
  public static void main(String[] args) {
    Exercise13_2 frame = new Exercise13_2();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.pack();
    frame.setVisible(true);
  }
}

class OvalButton extends JButton {
  public OvalButton() {
  }

  public OvalButton(String text) {
    super(text);
  }

  protected void paintComponent(Graphics g) {
    // Draw an oval
    super.paintComponent(g);
    g.drawOval(5, 5, getWidth() - 10, getHeight() - 10);
  }

  /** Override get method for preferredSize */
  public Dimension getPreferredSize() {
    return new Dimension(100, 50);
  }

  /** Override get method for minimumSize */
  public Dimension getMinimumSize() {
    return new Dimension(100, 50);
  }
}
```




```
import java.awt.*;
import javax.swing.*;

public class Exercise13_8 extends JFrame {
  public static void main(String[] args) {
    JFrame frame = new Exercise13_8();
    frame.setSize(300, 300);
    frame.setTitle("Exercise13_8");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
  }

  public Exercise13_8() {

    getContentPane().add(new OctagonPanel());
  }
}

class OctagonPanel extends JPanel {
  protected void paintComponent(Graphics g) {
    super.paintComponent(g);

    int xCenter = getWidth() / 2;
    int yCenter = getHeight() / 2;
    int radius =
      (int) (Math.min(getWidth(), getHeight()) * 0.4);

    // Create a Polygon object
    Polygon polygon = new Polygon();

    // Add points to the polygon
    polygon.addPoint(xCenter + radius, yCenter);
    polygon.addPoint((int)(xCenter + radius * Math.cos(2 * Math.PI / 8)),
      (int)(yCenter - radius * Math.sin(2 * Math.PI / 8)));
    polygon.addPoint((int)(xCenter + radius * Math.cos(2 * 2 * Math.PI
```

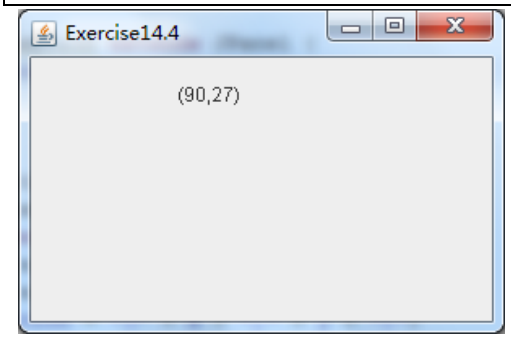
```
/ 8)),
      (int)(yCenter - radius * Math.sin(2 * 2 * Math.PI / 8)));
    polygon.addPoint((int)(xCenter + radius * Math.cos(3 * 2 * Math.PI
/ 8)),
      (int)(yCenter - radius * Math.sin(3 * 2 * Math.PI / 8)));
    polygon.addPoint((int)(xCenter + radius * Math.cos(4 * 2 * Math.PI
/ 8)),
      (int)(yCenter - radius * Math.sin(4 * 2 * Math.PI / 8)));
    polygon.addPoint((int)(xCenter + radius * Math.cos(5 * 2 * Math.PI
/ 8)),
      (int)(yCenter - radius * Math.sin(5 * 2 * Math.PI / 8)));
    polygon.addPoint((int)(xCenter + radius * Math.cos(6 * 2 * Math.PI
/ 8)),
      (int)(yCenter - radius * Math.sin(6 * 2 * Math.PI / 8)));
    polygon.addPoint((int)(xCenter + radius * Math.cos(7 * 2 * Math.PI
/ 8)),
      (int)(yCenter - radius * Math.sin(7 * 2 * Math.PI / 8)));

    // Draw the polygon
    g.drawPolygon(polygon);
  }
}
```




```
import javax.swing.*;
import java.awt.*;

public class Exercise13_11 extends JFrame {
    public Exercise13_11(){
        add(new SquareFunction());
    }

    /**
     * @param args
     */
    public static void main(String[] args) {
```

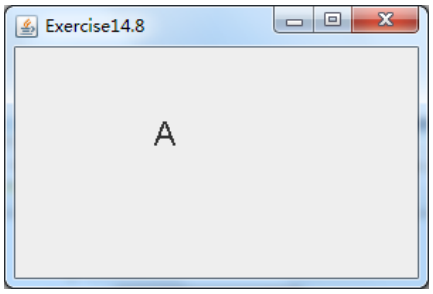
```
        // TODO 自动生成方法存根
        Exercise13_11 frame = new Exercise13_11();
        frame.setTitle("Exercise13.11");
        frame.setSize(300,200);
        frame.setLocationRelativeTo(null);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);

    }

}

class SquareFunction extends JPanel {
    public SquareFunction(){

    }

    protected void paintComponent(Graphics g){
        super.paintComponent(g);

        g.drawLine(10, this.getHeight()-20, this.getWidth()-10,
this.getHeight()-20);
        g.drawLine(this.getWidth()-30, this.getHeight()-15,
this.getWidth()-10, this.getHeight()-20);
        g.drawLine(this.getWidth()-30, this.getHeight()-25,
this.getWidth()-10, this.getHeight()-20);
        g.drawString("X", this.getWidth()-10, this.getHeight()-30);

        g.drawLine(this.getWidth()/2, 20, this.getWidth()/2,
this.getHeight()-5);
        g.drawLine(this.getWidth()/2, 20, this.getWidth()/2-5, 35);
        g.drawLine(this.getWidth()/2, 20, this.getWidth()/2+5, 35);
        g.drawString("Y", this.getWidth()/2 + 10, 30);

        Polygon p = new Polygon();
        double scaleFactor = 0.01;
        for (int x=-100; x<=100; x++){
            p.addPoint(x+this.getWidth()/2,
this.getHeight()-20-(int)(scaleFactor*x*x));
        }
        g.drawPolyline(p.xpoints, p.ypoints, p.npoints);
    }
}
```

Chapter 14	Exercise 14.4 / 14.8 / 14.9




```
package chapter14;

import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

public class Exercise14_4 extends JFrame {
    public Exercise14_4(){
        MousePosition p = new MousePosition();
        add(p);
    }

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO 自动生成方法存根
        Exercise14_4 frame = new Exercise14_4();
        frame.setTitle("Exercise14.4");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);

    }

    static class MousePosition extends JPanel {
        private String position = "";
        private int x;
        private int y;

        public MousePosition(){
            addMouseListener(new MouseAdapter(){
                public void mousePressed(MouseEvent e){
```

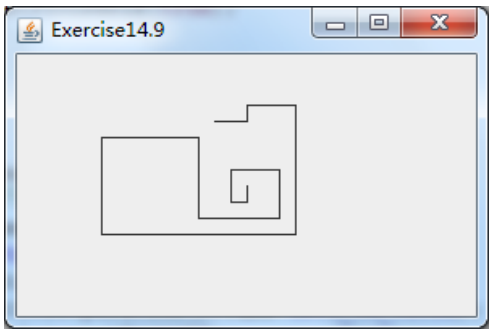
```
                x = e.getX();
                y = e.getY();
                position = "(" + x + "," + y + ")";
                repaint();
            }

            public void mouseReleased(MouseEvent e){
                x = e.getX();
                y = e.getY();
                position = "";
                repaint();
            }

            public void mouseClicked(MouseEvent e){
                x = e.getX();
                y = e.getY();
                position = "(" + x + "," + y + ")";
                repaint();
            }
        });
    }

    protected void paintComponent(Graphics g){
        super.paintComponent(g);

        g.drawString(position, x, y);
    }
}

}
```




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

public class Exercise14_8 extends JFrame {
```

```
private CharacterPanel characterPansel = new CharacterPanel();

public Exercise14_8(){
    add(characterPansel);

    characterPansel.setFocusable(true);
}

/**
 * @param args
 */
public static void main(String[] args) {
    // TODO 自动生成方法存根
    JFrame frame = new Exercise14_8();
    frame.setTitle("Exercise14.8");
    frame.setSize(300, 200);
    frame.setLocationRelativeTo(null);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);

}

static class CharacterPanel extends JPanel{
    private int x = 5;
    private int y = 10;
    private char character;

    public CharacterPanel(){
        addKeyListener(new KeyAdapter(){
            public void keyTyped(KeyEvent e){
                character = e.getKeyChar();
                    repaint();
                    }
        });

        addMouseListener(new MouseAdapter(){
            public void mouseClicked(MouseEvent e){
                x = e.getX();
                y = e.getY();

                repaint();
            }
        });
        }
```

```
        protected void paintComponent(Graphics g){
            super.paintComponent(g);
            //g.setFont(new Font("TimesRoman", Font.PLAIN, 24));
            g.drawString("" + character, x, y);
        }
    }
}
```

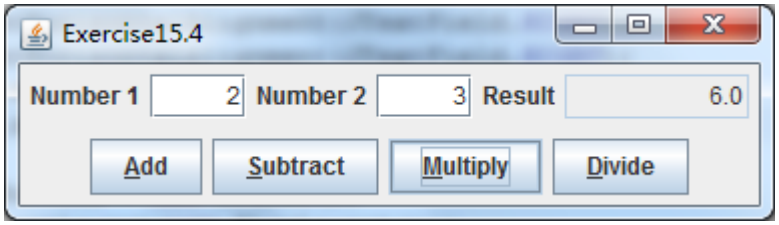



```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Exercise14_9 extends JFrame{
    private DrawLinesPanel drawLinesPanel = new DrawLinesPanel();

    public Exercise14_9(){
        add(drawLinesPanel);

        drawLinesPanel.setFocusable(true);
    }

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO 自动生成方法存根
        Exercise14_9 frame = new Exercise14_9();
        frame.setTitle("Exercise14.9");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);

    }
```

```
static class DrawLinesPanel extends JPanel{
    private int x, y;
    private Polygon polygon = new Polygon();

    public DrawLinesPanel(){
        polygon.addPoint(x, y);

        addKeyListener(new KeyAdapter(){
            public void keyPressed(KeyEvent e){
                switch(e.getKeyCode()){
                case KeyEvent.VK_UP:
                    y -= 10;
                    polygon.addPoint(x, y);
                    break;

                case KeyEvent.VK_DOWN:
                    y += 10;
                    polygon.addPoint(x, y);
                    break;

                case KeyEvent.VK_LEFT:
                    x -= 10;
                    polygon.addPoint(x, y);
                    break;

                case KeyEvent.VK_RIGHT:
                    x += 10;
                    polygon.addPoint(x, y);
                    break;

                }

                repaint();
            }
        });

    }

    protected void paintComponent(Graphics g){
        super.paintComponent(g);

        Polygon p = new Polygon();
        int xCenter = getWidth()/2;
```

```
            int yCenter = getHeight()/2;

            for(int i=0; i<polygon.npoints; i++){
                p.addPoint(polygon.xpoints[i] +
xCenter,polygon.ypoints[i] + yCenter);

            }

            g.drawPolyline(p.xpoints, p.ypoints, p.npoints);
        }
    }

}
```

Chapter 15	Exercise 15.4 / 15.6 / 15.7

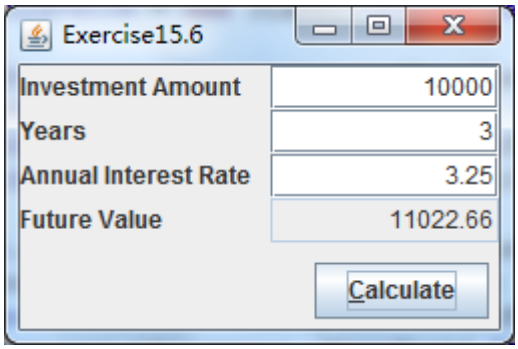



```
import java.awt.*;
import java.awt.event.*;

import javax.swing.*;

public class Exercise15_4 extends JFrame {
    protected JButton jbtAdd, jbtSubtract, jbtMultiply, jbtDivide;
    protected JTextField jtfNum1, jtfNum2, jtfResult;

    protected double x, y, result;

    public Exercise15_4(){
        setLayout(new BorderLayout());

        JPanel jpIO = new JPanel();
        jpIO.setLayout(new FlowLayout());

        add(jpIO, BorderLayout.CENTER);

        jpIO.add(new JLabel("Number 1"));
        jpIO.add(jtfNum1 = new JTextField(4));
```

```
jpIO.add(new JLabel("Number 2"));
jpIO.add(jtfNum2 = new JTextField(4));
jpIO.add(new JLabel("Result"));
jpIO.add(jtfResult = new JTextField(8));

jtfNum1.setHorizontalAlignment(JTextField.RIGHT);
jtfNum2.setHorizontalAlignment(JTextField.RIGHT);
jtfResult.setHorizontalAlignment(JTextField.RIGHT);
jtfResult.setEditable(false);

JPanel jpButtons = new JPanel();
jpButtons.setLayout(new FlowLayout());

add(jpButtons, BorderLayout.SOUTH);

jpButtons.add(jbtAdd = new JButton("Add"));
jpButtons.add(jbtSubtract = new JButton("Subtract"));
jpButtons.add(jbtMultiply = new JButton("Multiply"));
jpButtons.add(jbtDivide = new JButton("Divide"));

ButtonGroup group = new ButtonGroup();
group.add(jbtAdd);
group.add(jbtSubtract);
group.add(jbtMultiply);
group.add(jbtDivide);

jbtAdd.setMnemonic('A');
jbtSubtract.setMnemonic('S');
jbtMultiply.setMnemonic('M');
jbtDivide.setMnemonic('D');

jbtAdd.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e){
        x = Double.parseDouble(jtfNum1.getText());
        y = Double.parseDouble(jtfNum2.getText());
        result = add(x, y);
        jtfResult.setText("" + (float)result);

    }
});
jbtSubtract.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e){
        x = Double.parseDouble(jtfNum1.getText());
        y = Double.parseDouble(jtfNum2.getText());
```

```
            result = subtract(x, y);
            jtfResult.setText("" + (float)result);

        }
    });
    jbtMultiply.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e){
            x = Double.parseDouble(jtfNum1.getText());
            y = Double.parseDouble(jtfNum2.getText());
            result = multiply(x, y);
            jtfResult.setText("" + (float)result);

        }
    });
    jbtDivide.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e){
            x = Double.parseDouble(jtfNum1.getText());
            y = Double.parseDouble(jtfNum2.getText());
            result = divide(x, y);
            jtfResult.setText("" + (float)result);

        }
    });

}

public double add(double x, double y){
    return x + y;
}

public double subtract(double x, double y){
    return x - y;
}

public double multiply(double x, double y){
    return x * y;
}

public double divide(double x, double y){
    return x / y;
}

/**
 * @param args
```

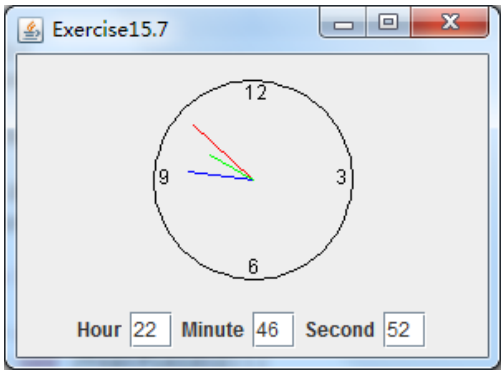
```
     */
    public static void main(String[] args) {
        // TODO 自动生成方法存根
        Exercise15_4 frame = new Exercise15_4();
        frame.pack();
        frame.setLocationRelativeTo(null);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setTitle("Exercise15.4");
        frame.setVisible(true);

    }

}
```




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

public class Exercise15_6 extends JFrame {
    private double futureValue, investmentAmount, annualInterestRate,
years;
    protected JLabel jlbFutureValue, jlbInvestmentAmount,
jlbAnnualInterestRate, jlbYears;
    protected JTextField jtfFutureValue, jtfInvestmentAmount,
jtfAnnualInterestRate, jtfYears;
    protected JButton jbtCalculate;


    public Exercise15_6(){
        setLayout(new BorderLayout(10, 5));

        JPanel  jpLabels= new JPanel();
        JPanel  jpTextFields= new JPanel();
        JPanel  jpButton = new JPanel();
```

```
        add(jpLabels, BorderLayout.WEST);
        add(jpTextFields, BorderLayout.CENTER);
        add(jpButton, BorderLayout.SOUTH);

        jpLabels.setLayout(new GridLayout(4, 1));
        jpTextFields.setLayout(new GridLayout(4, 1));
        jpButton.setLayout(new FlowLayout(FlowLayout.RIGHT));

        jpLabels.add(jlbInvestmentAmount = new JLabel("Investment
Amount"));
        jpLabels.add(jlbYears = new JLabel("Years"));
        jpLabels.add(jlbAnnualInterestRate = new JLabel("Annual Interest
Rate"));
        jpLabels.add(jlbFutureValue = new JLabel("Future Value"));

        jpTextFields.add(jtfInvestmentAmount = new JTextField(10));
        jpTextFields.add(jtfYears = new JTextField(10));
        jpTextFields.add(jtfAnnualInterestRate = new JTextField(10));
        jpTextFields.add(jtfFutureValue = new JTextField(10));

        jtfInvestmentAmount.setHorizontalAlignment(JTextField.RIGHT);
        jtfYears.setHorizontalAlignment(JTextField.RIGHT);

    jtfAnnualInterestRate.setHorizontalAlignment(JTextField.RIGHT);
        jtfFutureValue.setHorizontalAlignment(JTextField.RIGHT);
        jtfFutureValue.setEditable(false);

        jpButton.add(jbtCalculate = new JButton("Calculate"));
        jbtCalculate.setMnemonic('C');

        jbtCalculate.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e){
                investmentAmount =
Double.parseDouble(jtfInvestmentAmount.getText());
                annualInterestRate =
Double.parseDouble(jtfAnnualInterestRate.getText());
                years =  Double.parseDouble(jtfYears.getText());

                futureValue = investmentAmount * (Math.pow((1 +
annualInterestRate / 12 / 100), (years * 12)));

                jtfFutureValue.setText("" + (double)(int)(futureValue *
100) / 100);
```

```
            }
        });

    }

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO 自动生成方法存根
        Exercise15_6 frame = new Exercise15_6();
        frame.setTitle("Exercise15.6");
        frame.pack();
        frame.setLocationRelativeTo(null);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);

    }

}
```

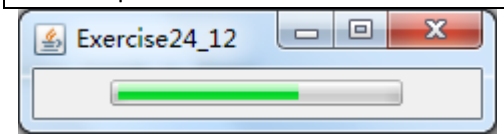



```
import java.awt.*;
import java.awt.event.*;

import javax.swing.*;

public class Exercise15_7 extends JFrame {
    protected StillClock stillClock= new StillClock();
    private JLabel jlbHour, jlbMinute, jlbSecond;
    protected JTextField jtfHour, jtfMinute, jtfSecond;
    private int hour, minute, second;
```

```
public Exercise15_7(){
    setLayout(new BorderLayout());

    add(stillClock, BorderLayout.CENTER);

    JPanel jp = new JPanel();
    add(jp, BorderLayout.SOUTH);

    jp.setLayout(new FlowLayout());

    jp.add(jlbHour = new JLabel("Hour"));
    jp.add(jtfHour = new JTextField(2));
    jp.add(jlbMinute = new JLabel("Minute"));
    jp.add(jtfMinute = new JTextField(2));
    jp.add(jlbSecond = new JLabel("Second"));
    jp.add(jtfSecond = new JTextField(2));

    jtfHour.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e){
            hour = Integer.parseInt(jtfHour.getText());
            stillClock.setHour(hour);
        }
    });
    jtfMinute.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e){
            minute = Integer.parseInt(jtfMinute.getText());
            stillClock.setMinute(minute);
        }
    });
    jtfSecond.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e){
            second = Integer.parseInt(jtfSecond.getText());
            stillClock.setSecond(second);
        }
    });

}

/**
 * @param args
 */
public static void main(String[] args) {
    // TODO 自动生成方法存根
    Exercise15_7 frame = new Exercise15_7();
```

```
        frame.setTitle("Exercise15.7");
        frame.setSize(300, 220);
        frame.setLocationRelativeTo(null);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);

    }

}
```

Chapter 24	Exercise 24.12

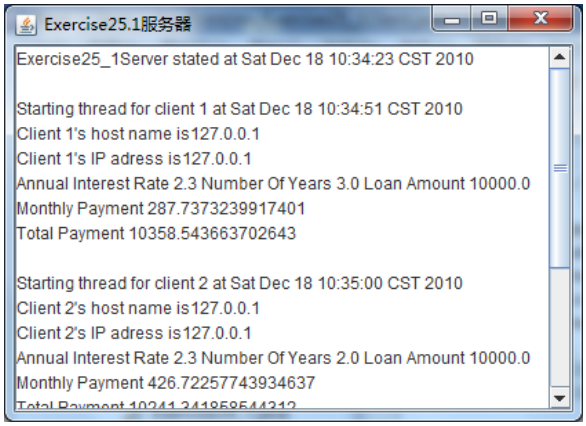


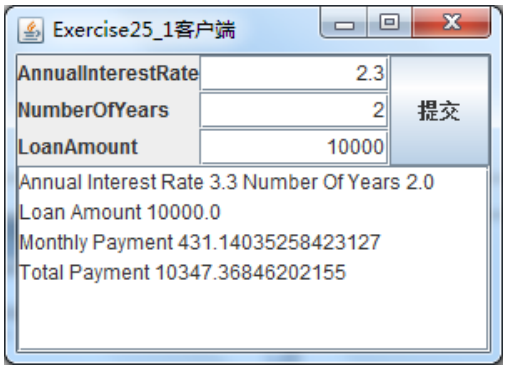

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class Exercise24_12 extends JApplet implements Runnable {
  boolean isStandalone = false;
  JProgressBar jProgressBar1 = new JProgressBar();
  FlowLayout flowLayout1 = new FlowLayout();

  Thread thread;

  /**Construct the applet*/
  public Exercise24_12() {
  }

  /**Initialize the applet*/
  public void init() {
    try {
      jbInit();
    }
    catch(Exception e) {
      e.printStackTrace();
    }
  }

  //Component initialization
  private void jbInit() throws Exception {
    this.setSize(new Dimension(400,300));
```

```
    this.getContentPane().setLayout(flowLayout1);
    this.getContentPane().add(jProgressBar1, null);

    thread = new Thread(this);
    thread.start();
  }

  public void run() {
    while (true) {
      try {
        Thread.sleep(500);
      }
      catch (InterruptedException ex) { }

      jProgressBar1.setValue((int)(jProgressBar1.getMaximum()*
        Math.random()));
    }
  }

  /**Main method*/
  public static void main(String[] args) {
    Exercise24_12 applet = new Exercise24_12();
    applet.isStandalone = true;
    JFrame frame = new JFrame();
    frame.setTitle("Exercise24_12");
    frame.getContentPane().add(applet, BorderLayout.CENTER);
    applet.init();
    applet.start();
    frame.pack();
    Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
    frame.setLocation((d.width - frame.getSize().width) / 2, (d.height
- frame.getSize().height) / 2);
    frame.setVisible(true);
  }

  // static initializer for setting look & feel
  static {
    try {

UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());

//UIManager.setLookAndFeel(UIManager.getCrossPlatformLookAndFeelClass
Name());
    }
```

```
    catch (Exception e) {}
  }
}
```

Chapter 25	Exercise 25.1 / 25.10






```
import java.io.*;
import java.net.*;
import java.util.*;
import java.awt.*;
import javax.swing.*;

public class Exercise25_1Server extends JFrame{
    private JTextArea jta = new JTextArea();
    /*
    private ObjectOutputStream toClient;
    private ObjectInputStream fromClient;
    */
    public Exercise25_1Server(){
        setLayout(new BorderLayout());
        add(new JScrollPane(jta), BorderLayout.CENTER);

        jta.setEditable(false);
        jta.setLineWrap(true);
        jta.setWrapStyleWord(true);

        setTitle("Exercise25.1服务器");
        setSize(420, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
```

```
        try{
            ServerSocket serverSocket = new ServerSocket(8000);
            jta.append("Exercise25_1Server stated at " + new Date() +
"\n");

            int client = 0;

            while(true){
                //监听连接，等待新的连接请求
                Socket socket = serverSocket.accept();

                client++;

                jta.append("\nStarting thread for client " + client + " at
" + new Date() + "\n");
                //获取客户域名及IP地址
                InetAddress inetAddress = socket.getInetAddress();
                jta.append("Client " + client + "'s host name is" +
inetAddress.getHostName() + "\n");
                jta.append("Client " + client + "'s IP adress is" +
inetAddress.getHostAddress() + "\n");

                HandleAClient task = new HandleAClient(socket);

                new Thread(task).start();

            }

        }catch(IOException ex){
            System.err.println(ex);
        }

    }

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        new Exercise25_1Server();
    }

    /**内部类
```

```
     * 定义线程类来处理新的连接*/
    class HandleAClient implements Runnable {
        private Socket socket;

        public HandleAClient(Socket socket){
            this.socket = socket;
        }
        @Override
        public void run() {
            try{
                ObjectInputStream inputFromClient = new
ObjectInputStream(socket.getInputStream());
                DataOutputStream outputToClient = new
DataOutputStream(socket.getOutputStream());

                while(true){
                    Loan loan = (Loan)(inputFromClient.readObject());

                    double monthlyPayment = loan.getMonthlyPayment();
                    double totalPayment = loan.getTotalPayment();

                    String information = "Annual Interest Rate " +
loan.getAnnualInterestRate()
                    + " Number Of Years " + loan.getNumberOfYears()
                    + " Loan Amount " + loan.getLoanAmount() + "\n"
                    + "Monthly Payment " + monthlyPayment + "\n"
                    + "Total Payment " + totalPayment + "\n";

                    jta.append(information);

                    outputToClient.writeUTF(information);
                }

            } catch(ClassNotFoundException ex){
                ex.printStackTrace();
            } catch(IOException ex){
                System.err.println(ex);
            }
        }

    }

}
```

```
import java.io.*;
import java.net.*;
import java.util.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.*;

public class Exercise25_1Client extends JFrame{
    private JLabel jlbAnnualInterestRate, jlbNumberOfYears,
jlbLoanAmount;
    private JTextField jtfAnnualInterestRate, jtfNumberOfYears,
jtfLoanAmount;
    private JButton jbtSubmit = new JButton("提交");
    private JTextArea jta = new JTextArea();

    //private ObjectOutputStream toServer;
    //private ObjectInputStream fromServer;

    public Exercise25_1Client(){
        setLayout(new BorderLayout());

        JPanel jpTop = new JPanel(new BorderLayout());
        JPanel jp1 = new JPanel(new GridLayout(3, 1));
        JPanel jp2 = new JPanel(new GridLayout(3, 1));

        jpTop.add(jp1, BorderLayout.WEST);
       jp1.add(jlbAnnualInterestRate = new JLabel("AnnualInterestRate",
JLabel.LEFT));
        jp1.add(jlbNumberOfYears = new JLabel("NumberOfYears",
JLabel.LEFT));
        jp1.add(jlbLoanAmount = new JLabel("LoanAmount", JLabel.LEFT));

        jpTop.add(jp2, BorderLayout.CENTER);
        jp2.add(jtfAnnualInterestRate = new JTextField(10));
        jp2.add(jtfNumberOfYears = new JTextField(10));
        jp2.add(jtfLoanAmount = new JTextField(10));

        jpTop.add(jbtSubmit, BorderLayout.EAST);

        add(new JScrollPane(jta), BorderLayout.CENTER);
        add(jpTop, BorderLayout.NORTH);
```

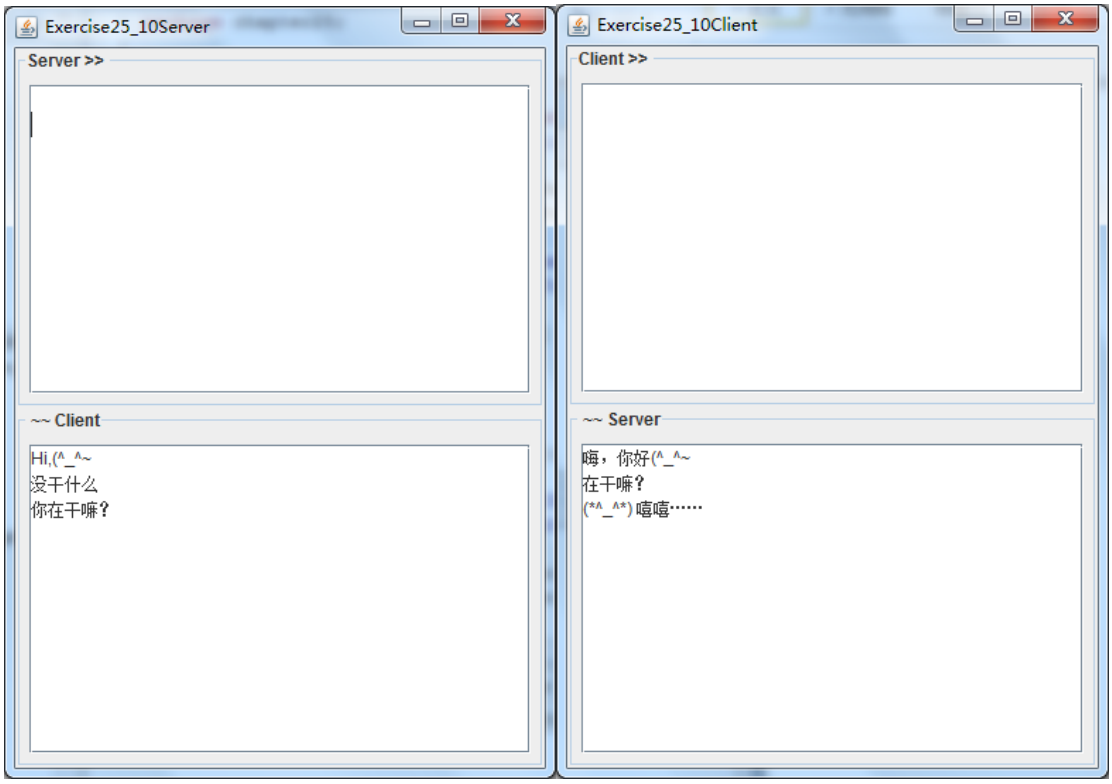
```
        jta.setRows(6);
        jta.setEditable(false);
        jta.setLineWrap(true);
        jta.setWrapStyleWord(true);


    jtfAnnualInterestRate.setHorizontalAlignment(JTextField.RIGHT);
        jtfNumberOfYears.setHorizontalAlignment(JTextField.RIGHT);
        jtfLoanAmount.setHorizontalAlignment(JTextField.RIGHT);
        /*至此用户界面设置完成*/

        setTitle("Exercise25_1客户端");
        pack();
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setVisible(true);

        jbtSubmit.addActionListener(new SubmitListener());

    }

    private class SubmitListener implements ActionListener{
        public void actionPerformed(ActionEvent e){
            try{
                Socket socket = new Socket("localhost", 8000);

                ObjectOutputStream toServer = new
ObjectOutputStream(socket.getOutputStream());

                DataInputStream fromServer = new
DataInputStream(socket.getInputStream());

                double annualInterestRate =
Double.parseDouble(jtfAnnualInterestRate.getText().trim());
                double numberOfYears =
Double.parseDouble(jtfNumberOfYears.getText().trim());
                double loanAmount =
Double.parseDouble(jtfLoanAmount.getText().trim());

                Loan l = new Loan(annualInterestRate, numberOfYears,
loanAmount);
                toServer.writeObject(l);
                toServer.flush();
```

```
                String information = fromServer.readUTF();

                jta.setText(information);

            } catch(IOException ex){
                System.err.println(ex + "\n");
            }
        }
    }

    public static void main(String[] args){
        new Exercise25_1Client();
    }

}
```

Loan类

```
public class Loan implements java.io.Serializable {//这里必须实现
Serializable接口
    private double annualInterestRate, numberOfYears, loanAmount;

    public Loan(){
        this(0, 0, 0);
    }

    public Loan(double annualInterestRate, double numberOfYears, double
loanAmount){
        this.annualInterestRate = annualInterestRate;
        this.numberOfYears = numberOfYears;
        this.loanAmount = loanAmount;
    }

    public double getMonthlyPayment(){
        double monthlyInterestRate = annualInterestRate / 1200;
        double monthlyPayment = loanAmount * monthlyInterestRate /
        (1 - (Math.pow(1 / (1 + monthlyInterestRate), numberOfYears
*12)));

        return monthlyPayment;
    }

    public double getTotalPayment(){
        double totalPayment = getMonthlyPayment() * 12 * numberOfYears;
```

```
        return totalPayment;
    }

    public double getAnnualInterestRate(){
        return annualInterestRate;
    }

    public double getNumberOfYears(){
        return numberOfYears;
    }

    public double getLoanAmount(){
        return loanAmount;
    }

}
```



服务器端：

```
import java.io.*;
import java.net.*;
import java.awt.event.*;
import java.awt.*;
import javax.swing.*;
```

```
import javax.swing.border.*;

public class Exercise25_10Server extends JFrame{
    private JTextArea serverArea, clientArea;

    private JPanel jpServer = new JPanel();
    private JPanel jpClient = new JPanel();

    private DataInputStream inputFromClient;
    private DataOutputStream outputToClient;


    public static void main(String[] args){
        new Exercise25_10Server();
    }

    public Exercise25_10Server(){
        setLayout(new BorderLayout());

        add(jpServer, BorderLayout.NORTH);
        add(jpClient, BorderLayout.CENTER);

        jpServer.add(new JScrollPane(serverArea = new JTextArea(12,
32)));
        jpClient.add(new JScrollPane(clientArea = new JTextArea(12,
32)));

        jpServer.setBorder(new TitledBorder("Server >> "));
        jpClient.setBorder(new TitledBorder(" ~~ Client"));

        clientArea.setEditable(false);

        setTitle("Exercise25_10Server");
        setLocation(200, 100);
        pack();
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);


        serverArea.addKeyListener(new KeyAdapter(){
            public void keyPressed(KeyEvent e){
                if(e.getKeyCode() == KeyEvent.VK_DOWN){
                    String wordsToServer = serverArea.getText();
                    serverArea.setText(wordsToServer + "\n");
```

```
                }

                if(e.getKeyCode() == KeyEvent.VK_ENTER){
                    String wordsToClient = serverArea.getText();
                    serverArea.setText(null);

                    try{
                        outputToClient.writeUTF(wordsToClient);
                        outputToClient.flush();
                    }
                    catch(IOException ex){
                        System.err.println(ex);
                    }
                }

            }
        });

        try{
            ServerSocket serverSocket = new ServerSocket(8000);

            Socket socket = serverSocket.accept();//监听连接

            inputFromClient = new
DataInputStream(socket.getInputStream());
            outputToClient = new
DataOutputStream(socket.getOutputStream());

            while(true){

                String wordsFormClient = inputFromClient.readUTF();

                clientArea.append(wordsFormClient);

            }


        }
        catch(IOException ex){
            System.err.println(ex);
        }

    }
```

```
}
```

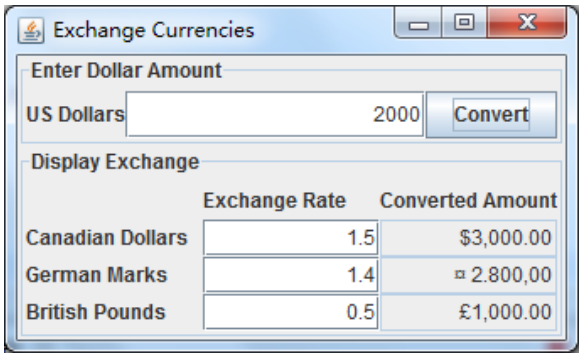
客户端：

```
import java.io.*;
import java.net.*;
import java.awt.event.*;
import java.awt.*;
import javax.swing.*;
import javax.swing.border.TitledBorder;

public class Exercise25_10Client  extends JFrame{
    private JTextArea serverArea, clientArea;

    private JPanel jpServer = new JPanel();
    private JPanel jpClient = new JPanel();

    private DataInputStream inputFromServer;
    private DataOutputStream outputToServer;

    public static void main(String[] args){
        new Exercise25_10Client();
    }

    public Exercise25_10Client(){
        setLayout(new BorderLayout());

        add(jpClient, BorderLayout.CENTER);
        add(jpServer, BorderLayout.SOUTH);

        jpClient.add(new JScrollPane(clientArea = new JTextArea(12,
32)));
        jpServer.add(new JScrollPane(serverArea = new JTextArea(12,
32)));

        jpClient.setBorder(new TitledBorder("Client >> "));
        jpServer.setBorder(new TitledBorder(" ~~ Server"));



        serverArea.setEditable(false);

        setTitle("Exercise25_10Client");
        setLocation(600, 100);
```

```
        pack();
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);


        clientArea.addKeyListener(new KeyAdapter(){
            public void keyPressed(KeyEvent e){
                if(e.getKeyCode() == KeyEvent.VK_DOWN){
                    String wordsToServer = clientArea.getText();
                    clientArea.setText(wordsToServer + "\n");

                    }

                if(e.getKeyCode() == KeyEvent.VK_ENTER){
                    String wordsToServer = clientArea.getText();
                    clientArea.setText(null);

                    try{
                        outputToServer.writeUTF(wordsToServer);
                        outputToServer.flush();
                    }
                    catch(IOException ex){
                        System.err.println(ex);
                    }

                }
            }
        });

        try{
            Socket socket = new Socket("localhost", 8000);//请求连接

            inputFromServer = new
DataInputStream(socket.getInputStream());
            outputToServer = new
DataOutputStream(socket.getOutputStream());

            while(true){

                String wordsFormServer = inputFromServer.readUTF();

                serverArea.append(wordsFormServer);
            }
        }
```

```
        catch(IOException ex){
            System.err.println(ex);
        }


    }

}
```

Chapter 26	Exercise 26.6



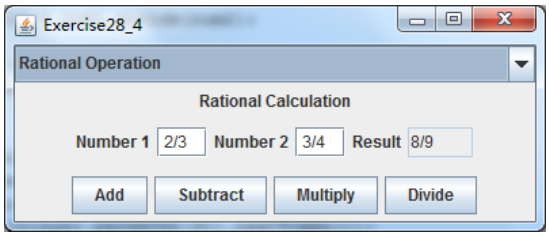

```
// Exercise26_6.java: Convert currency
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;
import java.text.*;
import java.util.*;

public class Exercise26_6 extends JApplet implements ActionListener {
  // Text fields for US dollars, Canadian dollars, German marks
  // and British pounds
  JTextField jtfUSDollars = new JTextField();
  JTextField jtfCanadianDollars = new JTextField();
  JTextField jtfGermanMarks = new JTextField();
  JTextField jtfBritishPounds = new JTextField();

  // Text fields for excahnge rates
  JTextField jtfCanadianDollarsRate = new JTextField();
  JTextField jtfGermanMarksRate = new JTextField();
  JTextField jtfBritishPoundsRate = new JTextField();

  // Button to convert currencies
  JButton jbtConvert = new JButton("Convert");
```

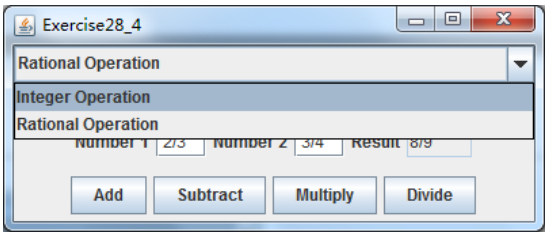
```
  // Number formater
  NumberFormat nfCanada =
NumberFormat.getCurrencyInstance(Locale.CANADA);
  NumberFormat nfGerman =
NumberFormat.getCurrencyInstance(Locale.GERMAN);
  NumberFormat nfUK = NumberFormat.getCurrencyInstance(Locale.UK);


  public void init() {
    // Panel p1 to hold the text field and button for US dollars
    JPanel p1 = new JPanel();
    p1.setLayout(new BorderLayout());
    p1.add(new JLabel("US Dollars"), BorderLayout.WEST);
    p1.add(jtfUSDollars, BorderLayout.CENTER);
    p1.add(jbtConvert, BorderLayout.EAST);
    p1.setBorder(new TitledBorder("Enter Dollar Amount"));

    // Panel p2 to hold the text field and button for US dollars
    JPanel p2 = new JPanel();
    p2.setLayout(new GridLayout(4, 3));
    p2.add(new JLabel());
    p2.add(new JLabel("Exchange Rate"));
    p2.add(new JLabel("Converted Amount"));
    p2.add(new JLabel("Canadian Dollars"));
    p2.add(jtfCanadianDollarsRate);
    p2.add(jtfCanadianDollars);
    p2.add(new JLabel("German Marks"));
    p2.add(jtfGermanMarksRate);
    p2.add(jtfGermanMarks);
    p2.add(new JLabel("British Pounds"));
    p2.add(jtfBritishPoundsRate);
    p2.add(jtfBritishPounds);
    p2.setBorder(new TitledBorder("Display Exchange"));

    // Set the text fields properties
    jtfCanadianDollars.setEditable(false);
    jtfGermanMarks.setEditable(false);
    jtfBritishPounds.setEditable(false);
    jtfUSDollars.setHorizontalAlignment(JTextField.RIGHT);
    jtfCanadianDollars.setHorizontalAlignment(JTextField.RIGHT);
    jtfGermanMarks.setHorizontalAlignment(JTextField.RIGHT);
    jtfBritishPounds.setHorizontalAlignment(JTextField.RIGHT);
    jtfCanadianDollarsRate.setHorizontalAlignment(JTextField.RIGHT);
    jtfGermanMarksRate.setHorizontalAlignment(JTextField.RIGHT);
```

```
    jtfBritishPoundsRate.setHorizontalAlignment(JTextField.RIGHT);

    // Add panels to the frame
    this.getContentPane().add(p1, BorderLayout.NORTH);
    this.getContentPane().add(p2, BorderLayout.CENTER);

    // Register listener
    jbtConvert.addActionListener(this);
  }

  // Handle ActionEvent
  public void actionPerformed(ActionEvent e) {
    if (e.getSource() == jbtConvert) {
      double USDollars = new
Double(jtfUSDollars.getText().trim()).doubleValue();
      double rateCanada = new
Double(jtfCanadianDollarsRate.getText().trim()).doubleValue();
      double rateGermany = new
Double(jtfGermanMarksRate.getText().trim()).doubleValue();
      double rateBritan = new
Double(jtfBritishPoundsRate.getText().trim()).doubleValue();


jtfCanadianDollars.setText(nfCanada.format(USDollars*rateCanada));
      jtfGermanMarks.setText(nfGerman.format(USDollars*rateGermany));
      jtfBritishPounds.setText(nfUK.format(USDollars*rateBritan));
    }
  }

  // Main method
  public static void main(String[] args) {
    // Create a frame
    JFrame frame = new JFrame("Exchange Currencies");

    // Create an instance of the applet
    Exercise26_6 applet = new Exercise26_6();

    // Add the applet instance to the frame
    frame.getContentPane().add(applet, BorderLayout.CENTER);

    // Invoke init() and start()
    applet.init();
    applet.start();
```

```
    // Display the frame
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setLocationRelativeTo(null);
    frame.pack();
    frame.setVisible(true);
  }
}
```

Chapter 28	Exercise 28.4 / 28.6 / 28.7






```
// Exercise28_4.java: Use CardLayout
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;

public class Exercise28_4 extends JApplet implements ActionListener {
  private CardLayout queue = new CardLayout();
  private JPanel cardPanel = new JPanel();
  private JMenuItem jmiInt, jmiRat, jmiClose;
  private JComboBox jcboIntRational = new JComboBox(new Object[]{
    "Integer Operation", "Rational Operation"});

    public Exercise28_4() {

    // Create JMenuBar jmb
    JMenuBar jmb = new JMenuBar();

    //add a menu "Operation" in jmb
    JMenu operationMenu = new JMenu("Operation", false);
    jmb.add(operationMenu);

    //add a menu "Exit" in jmb
    JMenu exitMenu = new JMenu("Exit", true);
    jmb.add(exitMenu);

    //add JMenuItems
    operationMenu.add(jmiInt = new JMenuItem("Integer"));
```

```
    operationMenu.add(jmiRat = new JMenuItem("Rational"));
    exitMenu.add(jmiClose = new JMenuItem("Close"));

    //create intPanel for integer arithmetic
    JPanel intPanel = new IntPanel();

    //create rationalPanel for rational arithmetic
    JPanel rationalPanel = new RationalPanel();

    cardPanel.setLayout(queue);
    cardPanel.add(intPanel, "Integer");
    cardPanel.add(rationalPanel, "Rational");

    //set FlowLayout in the frame
    getContentPane().setLayout(new BorderLayout());
    getContentPane().add(jcboIntRational, BorderLayout.NORTH);
    getContentPane().add(cardPanel, BorderLayout.CENTER);

    jcboIntRational.addActionListener(this);
  }

  //handling menu selection
  public void actionPerformed(ActionEvent e) {
    if (jcboIntRational.getSelectedItem().equals("Integer Operation"))
      queue.first(cardPanel);
    else if (jcboIntRational.getSelectedItem().equals("Rational
Operation"))
      queue.last(cardPanel);
  }

  public static void main(String[] args) {
    Exercise28_4 applet = new Exercise28_4();
    JFrame frame = new JFrame();
    //EXIT_ON_CLOSE == 3
    frame.setDefaultCloseOperation(3);
    frame.setTitle("Exercise28_4");
    frame.getContentPane().add(applet, BorderLayout.CENTER);
    applet.init();
    applet.start();
    frame.setSize(400,320);
    Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
    frame.setLocation((d.width - frame.getSize().width) / 2,
      (d.height - frame.getSize().height) / 2);
    frame.setVisible(true);
```

```
  }
}

class IntPanel extends CalculationPanel {
  IntPanel() {
    super("Integer Calculation");
  }

  void add() {
    int result = getNum1() + getNum2();
    //set result in JTextField tf3
    tfResult.setText(String.valueOf(result));
  }

  void subtract() {
    int result = getNum1() - getNum2();
    //set result in JTextField tf3
    tfResult.setText(String.valueOf(result));
  }

  void multiply() {
    int result = getNum1() * getNum2();
    //set result in JTextField tfResult
    tfResult.setText(String.valueOf(result));
  }

  void divide() {
    int result = getNum1() / getNum2();
    //set result in JTextField tfResult
    tfResult.setText(String.valueOf(result));
  }

  private int getNum1() {
    //use trim() to trim extraneous space in the text field
    int num1 = Integer.parseInt(tfNum1.getText().trim());
    return num1;
  }

  private int getNum2() {
    //use trim() to trim extraneous space in the text field
    int num2 = Integer.parseInt(tfNum2.getText().trim());
    return num2;
  }
}
```

```
class RationalPanel extends CalculationPanel {
  RationalPanel() {
    super("Rational Calculation");
  }

  void add() {
    Rational num1 = getNum1();
    Rational num2 = getNum2();
    Rational result = num1.add(num2);

    //set result in JTextField tfResult
    tfResult.setText(result.toString());
  }

  void subtract() {
    Rational num1 = getNum1();
    Rational num2 = getNum2();
    Rational result = num1.subtract(num2);

    //set result in JTextField tfResult
    tfResult.setText(result.toString());
  }

  void multiply() {
    Rational num1 = getNum1();
    Rational num2 = getNum2();
    Rational result = num1.multiply(num2);

    //set result in JTextField tfResult
    tfResult.setText(result.toString());
  }

  void divide() {
    Rational num1 = getNum1();
    Rational num2 = getNum2();
    Rational result = num1.divide(num2);

    //set result in JTextField tfResult
    tfResult.setText(result.toString());
  }

  Rational getNum1() {
    StringTokenizer st1 = new
```

```
      StringTokenizer(tfNum1.getText().trim(), "/");
    int numer1 = Integer.parseInt(st1.nextToken());
    int denom1 = Integer.parseInt(st1.nextToken());
    return new Rational(numer1,denom1);
  }

  Rational getNum2() {
    StringTokenizer st2 = new
      StringTokenizer(tfNum2.getText().trim(), "/");
    int numer2 = Integer.parseInt(st2.nextToken());
    int denom2 = Integer.parseInt(st2.nextToken());
    return new Rational(numer2,denom2);
  }
}

/*design a generic calculation user interface for int and
  rational arithmetic*/
abstract class CalculationPanel extends JPanel
  implements ActionListener {
  private JPanel p0 = new JPanel();
  private JPanel p1 = new JPanel();
  private JPanel p2 = new JPanel();
  JTextField tfNum1, tfNum2, tfResult;
  private JButton jbtAdd, jbtSub, jbtMul, jbtDiv;

  public CalculationPanel(String title) {
    p0.add(new JLabel(title));

    //add labels and text fields
    p1.setLayout(new FlowLayout());
    p1.add(new JLabel("Number 1"));
    p1.add(tfNum1 = new JTextField(" ", 3));
    p1.add(new JLabel("Number 2"));
    p1.add(tfNum2 = new JTextField(" ", 3));
    p1.add(new JLabel("Result"));
    p1.add(tfResult = new JTextField(" ", 4));
    tfResult.setEditable(false);

    //set FlowLayout for p2
    JPanel p2 = new JPanel();
    p2.setLayout(new FlowLayout());
    p2.add(jbtAdd = new JButton("Add"));
    p2.add(jbtSub = new JButton("Subtract"));
    p2.add(jbtMul = new JButton("Multiply"));
```

```
    p2.add(jbtDiv = new JButton("Divide"));

    //add panels into CalculationPanel
    setLayout(new BorderLayout());
    add("North",p0);
    add("Center",p1);
    add("South",p2);

    //register listener for source objects
    jbtAdd.addActionListener(this);
    jbtSub.addActionListener(this);
    jbtMul.addActionListener(this);
    jbtDiv.addActionListener(this);
  }

  public void actionPerformed(ActionEvent e) {
    String actionCommand = e.getActionCommand();
    if (e.getSource() instanceof JButton) {
      if ("Add".equals(actionCommand))
        add();
      else if ("Subtract".equals(actionCommand))
        subtract();
      else if ("Multiply".equals(actionCommand))
        multiply();
      else if ("Divide".equals(actionCommand))
        divide();
    }
  }

  abstract void add();

  abstract void subtract();

  abstract void multiply();

  abstract void divide();
}
```

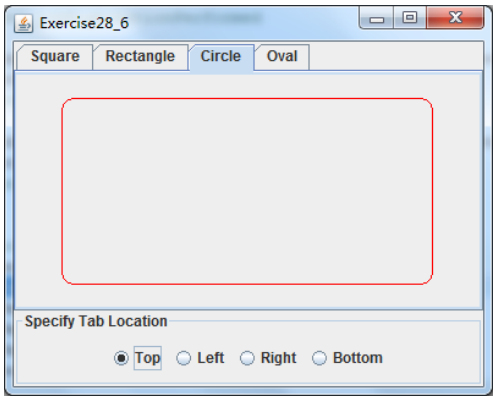
Rational 类

```
// Rational.java: Define a rational number and its associated
// operations such as add, subtract, multiply, and divide
public class Rational extends Number implements Comparable {
  // Data fields for numerator and denominator
  private long numerator = 0;
```

```
private long denominator = 1;

/** Default constructor */
public Rational() {
  this(0, 1);
}

/** Construct a rational with specified numerator and denominator */
public Rational(long numerator, long denominator) {
  long gcd = gcd(numerator, denominator);
  this.numerator = ((denominator > 0) ? 1 : -1) * numerator / gcd;
  this.denominator = Math.abs(denominator) / gcd;
}

/** Find GCD of two numbers */
private long gcd(long n, long d) {
  long t1 = Math.abs(n);
  long t2 = Math.abs(d);
  long remainder = t1 % t2;

  while (remainder != 0) {
    t1 = t2;
    t2 = remainder;
    remainder = t1%t2;
  }

  return t2;
}

/** Return numerator */
public long getNumerator() {
  return numerator;
}

/** Return denominator */
public long getDenominator() {
  return denominator;
}

/** Add a rational number to this rational */
public Rational add(Rational secondRational) {
  long n = numerator * secondRational.getDenominator() +
    denominator * secondRational.getNumerator();
  long d = denominator * secondRational.getDenominator();
```

```
  return new Rational(n, d);
}

/** Subtract a rational number from this rational */
public Rational subtract(Rational secondRational) {
  long n = numerator * secondRational.getDenominator()
    - denominator * secondRational.getNumerator();
  long d = denominator * secondRational.getDenominator();
  return new Rational(n, d);
}

/** Multiply a rational number to this rational */
public Rational multiply(Rational secondRational) {
  long n = numerator * secondRational.getNumerator();
  long d = denominator * secondRational.getDenominator();
  return new Rational(n, d);
}

/** Divide a rational number from this rational */
public Rational divide(Rational secondRational) {
  long n = numerator * secondRational.getDenominator();
  long d = denominator * secondRational.numerator;
  return new Rational(n, d);
}

/** Override the toString() method */
public String toString() {
  if (denominator == 1)
    return numerator + "";
  else
    return numerator + "/" + denominator;
}

/** Override the equals method in the Object class */
public boolean equals(Object parm1) {
  if ((this.subtract((Rational)(parm1))).getNumerator() == 0)
    return true;
  else
    return false;
}

/** Override the hashCode method in the Object class */
public int hashCode() {
  return new Double(this.doubleValue()).hashCode();
```

```
  }

  /** Override the abstract intValue method in java.lang.Number */
  public int intValue() {
    return (int)doubleValue();
  }

  /** Override the abstract floatValue method in java.lang.Number */
  public float floatValue() {
    return (float)doubleValue();
  }

  /** Override the doubleValue method in java.lang.Number */
  public double doubleValue() {
    return numerator * 1.0 / denominator;
  }

  /** Override the abstract longValue method in java.lang.Number */
  public long longValue() {
    return (long)doubleValue();
  }

  /** Override the compareTo method in java.lang.Comparable */
  public int compareTo(Object o) {
    if ((this.subtract((Rational)o)).getNumerator() > 0)
      return 1;
    else if ((this.subtract((Rational)o)).getNumerator() < 0)
      return -1;
    else
      return 0;
  }
}
```




```
import javax.swing.*;
```

```
import java.awt.*;

public class Exercise28_6 extends javax.swing.JApplet {
  /** Creates new form Exercise28_6 */
  public Exercise28_6() {
    buttonGroup1 = new javax.swing.ButtonGroup();
    jPanel1 = new javax.swing.JPanel();
    jrbTop = new javax.swing.JRadioButton();
    jrbLeft = new javax.swing.JRadioButton();
    jrbRight = new javax.swing.JRadioButton();
    jrbBottom = new javax.swing.JRadioButton();
    jtpFigures = new javax.swing.JTabbedPane();
    figurePanel1 = new FigurePanel();
    figurePanel2 = new FigurePanel();
    figurePanel3 = new FigurePanel();
    figurePanel4 = new FigurePanel();

    jPanel1.setBorder(new javax.swing.border.TitledBorder("Specify Tab
Location"));
    jrbTop.setText("Top");
    buttonGroup1.add(jrbTop);
    jrbTop.addActionListener(new java.awt.event.ActionListener() {
      public void actionPerformed(java.awt.event.ActionEvent evt) {
        jrbTopActionPerformed(evt);
      }
    });

    jPanel1.add(jrbTop);

    jrbLeft.setText("Left");
    buttonGroup1.add(jrbLeft);
    jrbLeft.addActionListener(new java.awt.event.ActionListener() {
      public void actionPerformed(java.awt.event.ActionEvent evt) {
        jrbLeftActionPerformed(evt);
      }
    });

    jPanel1.add(jrbLeft);

    jrbRight.setText("Right");
    buttonGroup1.add(jrbRight);
    jrbRight.addActionListener(new java.awt.event.ActionListener() {
      public void actionPerformed(java.awt.event.ActionEvent evt) {
        jrbRightActionPerformed(evt);
```

```
      }
    });

    jPanel1.add(jrbRight);

    jrbBottom.setText("Bottom");
    buttonGroup1.add(jrbBottom);
    jrbBottom.addActionListener(new java.awt.event.ActionListener() {
      public void actionPerformed(java.awt.event.ActionEvent evt) {
        jrbBottomActionPerformed(evt);
      }
    });

    jPanel1.add(jrbBottom);

    getContentPane().add(jPanel1, java.awt.BorderLayout.SOUTH);

    jtpFigures.addTab("Square", figurePanel1);

    figurePanel2.setType(2);
    jtpFigures.addTab("Rectangle", figurePanel2);

    figurePanel3.setType(3);
    jtpFigures.addTab("Circle", figurePanel3);

    figurePanel4.setType(4);
    jtpFigures.addTab("Oval", figurePanel4);

    getContentPane().add(jtpFigures, java.awt.BorderLayout.CENTER);

  }

  private void jrbBottomActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jrbBottomActionPerformed
    jtpFigures.setTabPlacement(JTabbedPane.BOTTOM);
  }//GEN-LAST:event_jrbBottomActionPerformed

  private void jrbRightActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jrbRightActionPerformed
    jtpFigures.setTabPlacement(JTabbedPane.RIGHT);
  }//GEN-LAST:event_jrbRightActionPerformed

  private void jrbLeftActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jrbLeftActionPerformed
```

```
    jtpFigures.setTabPlacement(JTabbedPane.LEFT);
  }//GEN-LAST:event_jrbLeftActionPerformed

  private void jrbTopActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jrbTopActionPerformed
    jtpFigures.setTabPlacement(JTabbedPane.TOP);
  }//GEN-LAST:event_jrbTopActionPerformed


  // Variables declaration - do not modify//GEN-BEGIN:variables
  private javax.swing.JPanel jPanel1;
  private FigurePanel figurePanel4;
  private FigurePanel figurePanel3;
  private FigurePanel figurePanel2;
  private FigurePanel figurePanel1;
  private javax.swing.JRadioButton jrbBottom;
  private javax.swing.ButtonGroup buttonGroup1;
  private javax.swing.JRadioButton jrbRight;
  private javax.swing.JRadioButton jrbTop;
  private javax.swing.JTabbedPane jtpFigures;
  private javax.swing.JRadioButton jrbLeft;
  // End of variables declaration//GEN-END:variables

  public static void main(String[] args) {
    Exercise28_6 applet = new Exercise28_6();
    JFrame frame = new JFrame();
    //EXIT_ON_CLOSE == 3
    frame.setDefaultCloseOperation(3);
    frame.setTitle("Exercise28_6");
    frame.getContentPane().add(applet, BorderLayout.CENTER);
    applet.init();
    applet.start();
    frame.setSize(400,320);
    Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
    frame.setLocation((d.width - frame.getSize().width) / 2,
      (d.height - frame.getSize().height) / 2);
    frame.setVisible(true);
  }
}
```

FigurePanel 类

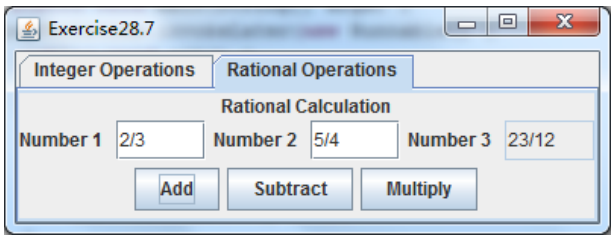
```
import java.awt.*;
import javax.swing.JPanel;
```

```
public class FigurePanel extends JPanel {
  // Define constants
  public static final int LINE = 1;
  public static final int RECTANGLE = 2;
  public static final int ROUND_RECTANGLE = 3;
  public static final int OVAL = 4;
  public static final int ARC = 5;
  public static final int POLYGON = 6;

  private int type = 1;
  private boolean filled;

  /** Construct a default FigurePanel */
  public FigurePanel() {
  }

  /** Construct a FigurePanel with the specified type */
  public FigurePanel(int type) {
    this.type = type;
  }

  /** Construct a FigurePanel with the specified type and filled */
  public FigurePanel(int type, boolean filled) {
    this.type = type;
    this.filled = filled;
  }

  /** Draw a figure on the panel */
  public void paintComponent(Graphics g) {
    super.paintComponent(g);

    // Get the appropriate size for the figure
    int width = getSize().width;
    int height = getSize().height;

    switch (type) {
      case LINE: // Display two cross lines
        g.setColor(Color.BLACK);
        g.drawLine(10, 10, width - 10, height - 10);
        g.drawLine(width - 10, 10, 10, height - 10);
        break;
      case RECTANGLE: // Display a rectangle
        g.setColor(Color.BLUE);
        if (filled)
```

```
      g.fillRect((int)(0.1 * width), (int)(0.1 * height),
        (int)(0.8 * width), (int)(0.8 * height));
    else
      g.drawRect((int)(0.1 * width), (int)(0.1 * height),
        (int)(0.8 * width), (int)(0.8 * height));
    break;
  case ROUND_RECTANGLE: // Display a round-cornered rectangle
    g.setColor(Color.RED);
    if (filled)
      g.fillRoundRect((int)(0.1 * width), (int)(0.1 * height),
        (int)(0.8 * width), (int)(0.8 * height), 20, 20);
    else
      g.drawRoundRect((int)(0.1 * width), (int)(0.1 * height),
        (int)(0.8 * width), (int)(0.8 * height), 20, 20);
    break;
  case OVAL: // Display an oval
    g.setColor(Color.BLACK);
    if (filled)
      g.fillOval((int)(0.1 * width), (int)(0.1 * height),
        (int)(0.8 * width), (int)(0.8 * height));
    else
      g.drawOval((int)(0.1 * width), (int)(0.1 * height),
        (int)(0.8 * width), (int)(0.8 * height));
    break;
  case ARC: // Display an arc
    g.setColor(Color.BLACK);
    if (filled) {
      int xCenter = getWidth() / 2;
      int yCenter = getHeight() / 2;
      int radius =
        (int)(Math.min(getWidth(), getHeight()) * 0.4);

      int x = xCenter - radius;
      int y = yCenter - radius;

      g.fillArc(x, y, 2 * radius, 2 * radius, 0, 30);
      g.fillArc(x, y, 2 * radius, 2 * radius, 90, 30);
      g.fillArc(x, y, 2 * radius, 2 * radius, 180, 30);
      g.fillArc(x, y, 2 * radius, 2 * radius, 270, 30);
    }
    else {
      int xCenter = getWidth() / 2;
      int yCenter = getHeight() / 2;
      int radius =
```

```
        (int)(Math.min(getWidth(), getHeight()) * 0.4);

      int x = xCenter - radius;
      int y = yCenter - radius;

      g.drawArc(x, y, 2 * radius, 2 * radius, 0, 30);
      g.drawArc(x, y, 2 * radius, 2 * radius, 90, 30);
      g.drawArc(x, y, 2 * radius, 2 * radius, 180, 30);
      g.drawArc(x, y, 2 * radius, 2 * radius, 270, 30);
    };
    break;
  case POLYGON: // Display a polygon
    g.setColor(Color.BLACK);
    int xCenter = getWidth() / 2;
    int yCenter = getHeight() / 2;
    int radius =
      (int)(Math.min(getWidth(), getHeight()) * 0.4);

    // Create a Polygon object
    Polygon polygon = new Polygon();

    // Add points to the polygon
    polygon.addPoint(xCenter + radius, yCenter);
    polygon.addPoint((int)(xCenter + radius *
      Math.cos(2 * Math.PI / 6)), (int)(yCenter - radius *
      Math.sin(2 * Math.PI / 6)));
    polygon.addPoint((int)(xCenter + radius *
      Math.cos(2 * 2 * Math.PI / 6)), (int)(yCenter - radius *
      Math.sin(2 * 2 * Math.PI / 6)));
    polygon.addPoint((int)(xCenter + radius *
      Math.cos(3 * 2 * Math.PI / 6)), (int)(yCenter - radius *
      Math.sin(3 * 2 * Math.PI / 6)));
    polygon.addPoint((int)(xCenter + radius *
      Math.cos(4 * 2 * Math.PI / 6)), (int)(yCenter - radius *
      Math.sin(4 * 2 * Math.PI / 6)));
    polygon.addPoint((int)(xCenter + radius *
      Math.cos(5 * 2 * Math.PI / 6)), (int)(yCenter - radius *
      Math.sin(5 * 2 * Math.PI / 6)));

    // Draw the polygon
    if (filled)
      g.fillPolygon(polygon);
    else
      g.drawPolygon(polygon);
```

```
    }
  }

  /** Set a new figure type */
  public void setType(int type) {
    this.type = type;
    repaint();
  }

  /** Return figure type */
  public int getType() {
    return type;
  }

  /** Set a new filled property */
  public void setFilled(boolean filled) {
    this.filled = filled;
    repaint();
  }

  /** Check if the figure is filled */
  public boolean isFilled() {
    return filled;
  }

  /** Specify preferred size */
  public Dimension getPreferredSize() {
    return new Dimension(80, 80);
  }
}
```




```
import java.awt.BorderLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JPanel;
```

```
import javax.swing.JTabbedPane;
import javax.swing.JTextField;
import javax.swing.SwingConstants;

import javax.swing.WindowConstants;
import javax.swing.SwingUtilities;

public class Exercise28_7 extends javax.swing.JFrame {
    private JTabbedPane jTabbedPane;
    private JPanel jpIntegerOperations;
    private JPanel jpIntegerNumbers;
    private JLabel jlbIntNum1;
    private JTextField jtfIntNum2;
    private JButton jbtAddInt;
    private JButton jbtMultiplyRat;
    private JButton jbtSubtractRat;
    private JButton jbtAddRat;
    private JPanel jpRatButtons;
    private JTextField jtfRatNum3;
    private JLabel jlbRatNum3;
    private JTextField jtfRatNum2;
    private JLabel jlbRatNum2;
    private JTextField jtfRatNum1;
    private JLabel jlRatNum1;
    private JPanel jpRationalNumbers;
    private JLabel jlbRationalCalculation;
    private JButton jbtMultiply;
    private JButton jbtSubtractInt;
    private JPanel jpIntButtons;
    private JTextField jtfIntNum3;
    private JLabel jlbIntNum3;
    private JLabel jlbIntNum2;
    private JTextField jtfIntNum1;
    private JLabel jlbIntegerCalculation;
    private JPanel jpRationalOperations;

    private int intNum1, intNum2, intNum3;
    private Rational ratNum1, ratNum2, ratNum3;

    /**
    * Auto-generated main method to display this JFrame
    */
    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
```

```
            public void run() {
                Exercise28_7 inst = new Exercise28_7();
                inst.setLocationRelativeTo(null);
                inst.setVisible(true);

                inst.setTitle("Exercise28.7");
                inst.setSize(400, 150);
            }
        });
    }

    public Exercise28_7() {
        super();
        initGUI();
    }

    private void initGUI() {
        try {
            BorderLayout thisLayout = new BorderLayout();

    setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
            getContentPane().setLayout(thisLayout);
            {
                jTabbedPane = new JTabbedPane();
                getContentPane().add(jTabbedPane, BorderLayout.CENTER);
                {
                    jpIntegerOperations = new JPanel();
                    BorderLayout jpIntegerOperationsLayout = new
BorderLayout();

    jpIntegerOperations.setLayout(jpIntegerOperationsLayout);
                    jTabbedPane.addTab("Integer Operations", null,
jpIntegerOperations, null);
                    {
                        jlbIntegerCalculation = new JLabel();
                        jpIntegerOperations.add(jlbIntegerCalculation,
BorderLayout.NORTH);
                        jlbIntegerCalculation.setText("Integer
Calculation");

    jlbIntegerCalculation.setHorizontalAlignment(SwingConstants.CENTE
R);
                    }
                    {
```

```
                    jpIntButtons = new JPanel();
                    jpIntegerOperations.add(jpIntButtons,
BorderLayout.SOUTH);
                    {
                        jbtAddInt = new JButton();
                        jpIntButtons.add(jbtAddInt);
                        jbtAddInt.setText("Add");
                        jbtAddInt.addActionListener(new
ActionListener() {
                            public void actionPerformed(ActionEvent
evt) {
                                intNum1 =
Integer.parseInt(jtfIntNum1.getText().trim());
                                intNum2 =
Integer.parseInt(jtfIntNum2.getText().trim());
                                intNum3 = intNum1 + intNum2;

                                jtfIntNum3.setText("" + intNum3);
                            }
                        });
                    }
                    {
                        jbtSubtractInt = new JButton();
                        jpIntButtons.add(jbtSubtractInt);
                        jbtSubtractInt.setText("Subtract");
                        jbtSubtractInt.addActionListener(new
ActionListener() {
                            public void actionPerformed(ActionEvent
evt) {
                                intNum1 =
Integer.parseInt(jtfIntNum1.getText().trim());
                                intNum2 =
Integer.parseInt(jtfIntNum2.getText().trim());
                                intNum3 = intNum1 - intNum2;

                                jtfIntNum3.setText("" + intNum3);
                            }
                        });
                    }
                    {
                        jbtMultiply = new JButton();
                        jpIntButtons.add(jbtMultiply);
                        jbtMultiply.setText("Multiply");
                        jbtMultiply.addActionListener(new
```

```
ActionListener() {
                            public void actionPerformed(ActionEvent
evt) {
                                intNum1 =
Integer.parseInt(jtfIntNum1.getText().trim());
                                intNum2 =
Integer.parseInt(jtfIntNum2.getText().trim());
                                intNum3 = intNum1 * intNum2;

                                jtfIntNum3.setText("" + intNum3);
                            }
                        });
                    }
                }
                {
                    jpIntegerNumbers = new JPanel();
                    GridLayout jpIntegerNumbersLayout = new
GridLayout(1, 6);
                    jpIntegerNumbersLayout.setHgap(5);
                    jpIntegerNumbersLayout.setVgap(5);
                    jpIntegerNumbersLayout.setColumns(6);
                    jpIntegerOperations.add(jpIntegerNumbers,
BorderLayout.CENTER);

    jpIntegerNumbers.setLayout(jpIntegerNumbersLayout);
                    {
                        jlbIntNum1 = new JLabel();
                        jpIntegerNumbers.add(jlbIntNum1);
                        jlbIntNum1.setText("Number 1");
                    }
                    {
                        jtfIntNum1 = new JTextField();
                        jpIntegerNumbers.add(jtfIntNum1);
                    }
                    {
                        jlbIntNum2 = new JLabel();
                        jpIntegerNumbers.add(jlbIntNum2);
                        jlbIntNum2.setText("Number 2");
                    }
                    {
                        jtfIntNum2 = new JTextField();
                        jpIntegerNumbers.add(jtfIntNum2);
                    }
                    {
```

```
                        jlbIntNum3 = new JLabel();
                        jpIntegerNumbers.add(jlbIntNum3);
                        jlbIntNum3.setText("Number 3");
                    }
                    {
                        jtfIntNum3 = new JTextField();
                        jpIntegerNumbers.add(jtfIntNum3);
                        jtfIntNum3.setEditable(false);
                    }
                }
            }
            {
                jpRationalOperations = new JPanel();
                BorderLayout jpRationalOperationsLayout = new
BorderLayout();

    jpRationalOperations.setLayout(jpRationalOperationsLayout);
                jTabbedPane.addTab("Rational Operations", null,
jpRationalOperations, null);
                {
                    jlbRationalCalculation = new JLabel();
                    jpRationalOperations.add(jlbRationalCalculation,
BorderLayout.NORTH);
                    jlbRationalCalculation.setText("Rational
Calculation");

    jlbRationalCalculation.setHorizontalAlignment(SwingConstants.CENT
ER);
                }
                {
                    jpRationalNumbers = new JPanel();
                    GridLayout jpRationalNumbersLayout = new
GridLayout(1, 6);
                    jpRationalNumbersLayout.setHgap(5);
                    jpRationalNumbersLayout.setVgap(5);
                    jpRationalNumbersLayout.setColumns(6);
                    jpRationalOperations.add(jpRationalNumbers,
BorderLayout.CENTER);

    jpRationalNumbers.setLayout(jpRationalNumbersLayout);
                    {
                        jlRatNum1 = new JLabel();
                        jpRationalNumbers.add(jlRatNum1);
                        jlRatNum1.setText("Number 1");
```

```
                        }
                        {
                            jtfRatNum1 = new JTextField();
                            jpRationalNumbers.add(jtfRatNum1);
                        }
                        {
                            jlbRatNum2 = new JLabel();
                            jpRationalNumbers.add(jlbRatNum2);
                            jlbRatNum2.setText("Number 2");
                        }
                        {
                            jtfRatNum2 = new JTextField();
                            jpRationalNumbers.add(jtfRatNum2);
                        }
                        {
                            jlbRatNum3 = new JLabel();
                            jpRationalNumbers.add(jlbRatNum3);
                            jlbRatNum3.setText("Number 3");
                        }
                        {
                            jtfRatNum3 = new JTextField();
                            jpRationalNumbers.add(jtfRatNum3);
                            jtfRatNum3.setEditable(false);
                        }
                    }
                    {
                        jpRatButtons = new JPanel();
                        jpRationalOperations.add(jpRatButtons,
BorderLayout.SOUTH);
                        {
                            jbtAddRat = new JButton();
                            jpRatButtons.add(jbtAddRat);
                            jbtAddRat.setText("Add");
                            jbtAddRat.addActionListener(new
ActionListener() {
                                public void actionPerformed(ActionEvent
evt) {
                                    ratNum1 = new
Rational(jtfRatNum1.getText().trim());
                                    ratNum2 = new
Rational(jtfRatNum2.getText().trim());
                                    ratNum3 = Rational.add(ratNum1,
ratNum2);
```

```
    jtfRatNum3.setText(ratNum3.getString());
                            }
                        });
                    }
                    {
                        jbtSubtractRat = new JButton();
                        jpRatButtons.add(jbtSubtractRat);
                        jbtSubtractRat.setText("Subtract");
                        jbtSubtractRat.addActionListener(new
ActionListener() {
                            public void actionPerformed(ActionEvent
evt) {
                                ratNum1 = new
Rational(jtfRatNum1.getText().trim());
                                ratNum2 = new
Rational(jtfRatNum2.getText().trim());
                                ratNum3 = Rational.subtract(ratNum1,
ratNum2);


    jtfRatNum3.setText(ratNum3.getString());
                            }
                        });
                    }
                    {
                        jbtMultiplyRat = new JButton();
                        jpRatButtons.add(jbtMultiplyRat);
                        jbtMultiplyRat.setText("Multiply");
                        jbtMultiplyRat.addActionListener(new
ActionListener() {
                            public void actionPerformed(ActionEvent
evt) {
                                ratNum1 = new
Rational(jtfRatNum1.getText().trim());
                                ratNum2 = new
Rational(jtfRatNum2.getText().trim());
                                ratNum3 = Rational.multiply(ratNum1,
ratNum2);


    jtfRatNum3.setText(ratNum3.getString());
                            }
                        });
```

```
                    }
                }
            }
        }
        pack();
        setSize(500, 400);
    } catch (Exception e) {
        //add your error handling code here
        e.printStackTrace();
    }
}

}
```

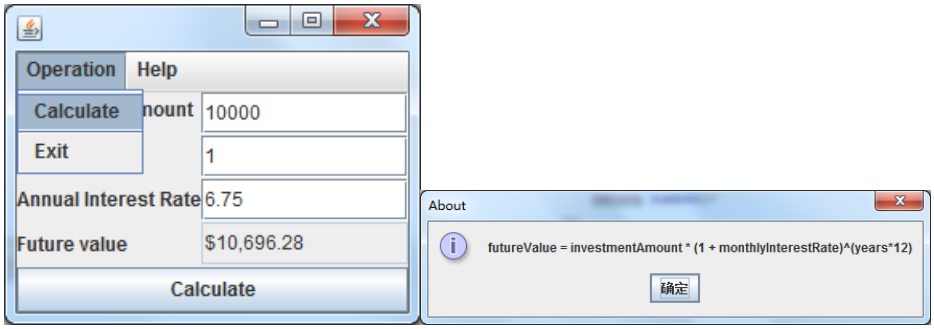
Rational 类

```
public class Rational {
    private String s;
    private double number;
    private int n1, n2;

    public Rational(){
        this(null);
    }

    public Rational(String s){
        this.s = s;
        number = getValue();
        n1 = getN1();
        n2 = getN2();
    }

    public static Rational add(Rational r1, Rational r2){
        Rational r;
        String rS = "" + (r1.n1 * r2.n2 + r2.n1 * r1.n2) + "/" + (r1.n2
* r2.n2);
        r = new Rational(rS);
        return r;
    }

    public static Rational subtract(Rational r1, Rational r2){
        Rational r;
        String rS = "" + (r1.n1 * r2.n2 - r2.n1 * r1.n2) + "/" + (r1.n2
* r2.n2);
        r = new Rational(rS);
```

```
    return r;
}

public static Rational multiply(Rational r1, Rational r2){
    Rational r;
    String rS = "" + (r1.n1 * r2.n1) + "/" + (r1.n2 * r2.n2);
    r = new Rational(rS);
    return r;
}

public String getString(){
    int commonDivisor = 1, k = 1;
    while(k <= n1 && k <= n2){
        if(n1 % k == 0 && n2 % k == 0)
            commonDivisor = k;
        k++;
    }

    n1 = n1 / commonDivisor;
    n2 = n2 / commonDivisor;

    s = "" + n1 + "/" + n2;

    return s;
}

private int getN1(){
    int lineIndex = s.length();
    for(int i = 0; i < s.length(); i++){
        if(s.charAt(i) == '/'){
            lineIndex = i;
            break;
        }
    }

    String s1 = s.substring(0, lineIndex);

    int n1 = Integer.parseInt(s1);

    return n1;
}
private int getN2(){
    int lineIndex = s.length();
    for(int i = 0; i < s.length(); i++){
```

```
            if(s.charAt(i) == '/')
                lineIndex = i;
        }

        String s2 = s.substring(lineIndex+1);

        int n2 = Integer.parseInt(s2);

        return n2;
    }

    private double getValue() {
        number = getN1() / getN2();

        return number;
    }


}
```

Chapter 29	Exercise 29.1 / 29.2 / 29.6




```
import java.awt.BorderLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.text.NumberFormat;
import java.util.Locale;

import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
```

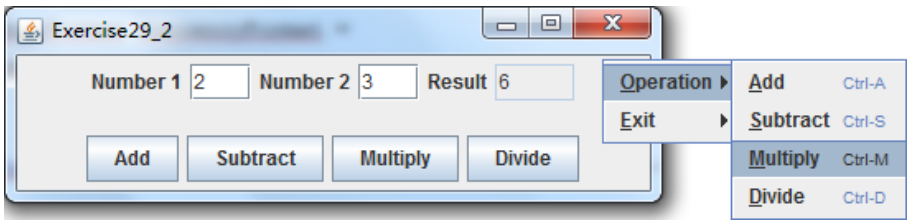
```
import javax.swing.JPanel;
import javax.swing.JSeparator;
import javax.swing.JTextField;

import javax.swing.WindowConstants;
import javax.swing.SwingUtilities;

public class Exercise29_1 extends javax.swing.JFrame {
    private JMenuBar jMenuBar1;
    private JPanel jPanel1;
    private JLabel jLabel3;
    private JLabel jLabel4;
    private JButton jbtCalculate;
    private JTextField jtfFutureValue;
    private JTextField jtfAnnualInterestRate;
    private JTextField jtfYears;
    private JTextField jtfInvestmentAmount;
    private JPanel jPanel2;
    private JLabel jLabel2;
    private JLabel jLabel1;
    private JMenuItem jMenuItemAbout;
    private JMenuItem jMenuItemExit;
    private JSeparator jSeparator1;
    private JMenuItem jMenuItemCalculate;
    private JMenu jmHelp;
    private JMenu jmOperation;

    /**
    * Auto-generated main method to display this JFrame
    */
    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                Exercise29_1 inst = new Exercise29_1();
                inst.setLocationRelativeTo(null);
                inst.setVisible(true);

                inst.setSize(260, 200);
            }
        });
    }

    public Exercise29_1() {
        super();
```

```
        initGUI();
    }

    private void initGUI() {
        try {
            BorderLayout thisLayout = new BorderLayout();

    setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
            getContentPane().setLayout(thisLayout);
            {
                jPanel1 = new JPanel();
                GridLayout jPanel1Layout = new GridLayout(4, 1);
                jPanel1Layout.setHgap(5);
                jPanel1Layout.setVgap(5);
                jPanel1Layout.setColumns(1);
                jPanel1Layout.setRows(4);
                getContentPane().add(jPanel1, BorderLayout.WEST);
                jPanel1.setLayout(jPanel1Layout);
                {
                    jLabel1 = new JLabel();
                    jPanel1.add(jLabel1);
                    jLabel1.setText("Investment Amount");
                }
                {
                    jLabel2 = new JLabel();
                    jPanel1.add(jLabel2);
                    jLabel2.setText("Years");
                }
                {
                    jLabel3 = new JLabel();
                    jPanel1.add(jLabel3);
                    jLabel3.setText("Annual Interest Rate");
                }
                {
                    jLabel4 = new JLabel();
                    jPanel1.add(jLabel4);
                    jLabel4.setText("Future value");
                }
            }
            {
                jPanel2 = new JPanel();
                GridLayout jPanel2Layout = new GridLayout(4, 1);
                jPanel2Layout.setHgap(5);
                jPanel2Layout.setVgap(1);
```

```
                jPanel2Layout.setColumns(1);
                jPanel2Layout.setRows(4);
                getContentPane().add(jPanel2, BorderLayout.CENTER);
                jPanel2.setLayout(jPanel2Layout);
                {
                    jtfInvestmentAmount = new JTextField();
                    jPanel2.add(jtfInvestmentAmount);
                }
                {
                    jtfYears = new JTextField();
                    jPanel2.add(jtfYears);
                }
                {
                    jtfAnnualInterestRate = new JTextField();
                    jPanel2.add(jtfAnnualInterestRate);
                }
                {
                    jtfFutureValue = new JTextField();
                    jPanel2.add(jtfFutureValue);
                    jtfFutureValue.setEditable(false);
                }
            }
            {
                jbtCalculate = new JButton();
                getContentPane().add(jbtCalculate, BorderLayout.SOUTH);
                jbtCalculate.setText("Calculate");
                jbtCalculate.addActionListener(new
CalculateListener());
            }
            {
                jMenuBar1 = new JMenuBar();
                setJMenuBar(jMenuBar1);
                {
                    jmOperation = new JMenu();
                    jMenuBar1.add(jmOperation);
                    jmOperation.setText("Operation");
                    {
                        jMenuItemCalculate = new JMenuItem();
                        jmOperation.add(jMenuItemCalculate);
                        jMenuItemCalculate.setText("Calculate");
                        jMenuItemCalculate.addActionListener(new
CalculateListener());
                    }
                    {
```

```
                    jSeparator1 = new JSeparator();
                    jmOperation.add(jSeparator1);
                }
                {
                    jMenuItemExit = new JMenuItem();
                    jmOperation.add(jMenuItemExit);
                    jMenuItemExit.setText("Exit");
                    jMenuItemExit.addActionListener(new
ActionListener() {
                        public void actionPerformed(ActionEvent evt) {
                            setVisible(false);
                        }
                    });
                }
            }
            {
                jmHelp = new JMenu();
                jMenuBar1.add(jmHelp);
                jmHelp.setText("Help");
                {
                    jMenuItemAbout = new JMenuItem();
                    jmHelp.add(jMenuItemAbout);
                    jMenuItemAbout.setText("About");
                    jMenuItemAbout.addActionListener(new
ActionListener() {
                        public void actionPerformed(ActionEvent evt) {
                            JOptionPane.showMessageDialog(null,
                                    "futureValue = investmentAmount * (1
+ monthlyInterestRate)^(years*12)",
                                    "About",
JOptionPane.INFORMATION_MESSAGE);
                        }
                    });
                }
            }
        }
        pack();
        setSize(500, 400);
    } catch (Exception e) {
        //add your error handling code here
        e.printStackTrace();
    }
}
```

```
    class CalculateListener implements ActionListener {

        @Override
        public void actionPerformed(ActionEvent arg0) {
            double investmentAmount, years, annualInterestRate,
futureValue;
            investmentAmount =
Double.parseDouble(jtfInvestmentAmount.getText().trim());
            years = Double.parseDouble(jtfYears.getText().trim());
            annualInterestRate =
Double.parseDouble(jtfAnnualInterestRate.getText().trim());
            futureValue = investmentAmount * Math.pow(1 +
annualInterestRate / 1200, years * 12);

            Locale local = Locale.US;
            NumberFormat currencyFormat =
NumberFormat.getCurrencyInstance(local);


    jtfFutureValue.setText(currencyFormat.format(futureValue));
        }

    }

}
```



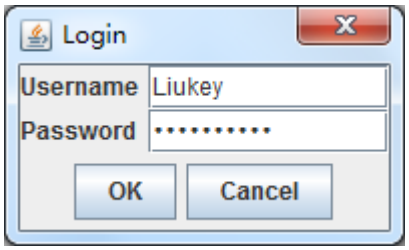

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Exercise29_2 extends JApplet implements ActionListener {
  // Text fields for Number 1, Number 2, and Result
  private JTextField jtfNum1, jtfNum2, jtfResult;

  // Buttons "Add", "Subtract", "Multiply" and "Divide"
  private JButton jbtAdd, jbtSub, jbtMul, jbtDiv;

  // Menu items "Add", "Subtract", "Multiply","Divide" and "Close"
```

```
private JMenuItem jmiAdd, jmiSub, jmiMul, jmiDiv, jmiClose;

private JPopupMenu jPopupMenu1 = new JPopupMenu();

public Exercise29_2() {
  // Add menu "Operation" to menu bar
  JMenu operationMenu = new JMenu("Operation");
  operationMenu.setMnemonic('O');
  jPopupMenu1.add(operationMenu);

  // Add menu "Exit" in menu bar
  JMenu exitMenu = new JMenu("Exit");
  exitMenu.setMnemonic('E');
  jPopupMenu1.add(exitMenu);

  // Add menu items with mnemonics to menu "Operation"
  operationMenu.add(jmiAdd= new JMenuItem("Add", 'A'));
  operationMenu.add(jmiSub = new JMenuItem("Subtract", 'S'));
  operationMenu.add(jmiMul = new JMenuItem("Multiply", 'M'));
  operationMenu.add(jmiDiv = new JMenuItem("Divide", 'D'));
  exitMenu.add(jmiClose = new JMenuItem("Close", 'C'));

  // Set keyboard accelerators
  jmiAdd.setAccelerator(
    KeyStroke.getKeyStroke(KeyEvent.VK_A, ActionEvent.CTRL_MASK));
  jmiSub.setAccelerator(
    KeyStroke.getKeyStroke(KeyEvent.VK_S, ActionEvent.CTRL_MASK));
  jmiMul.setAccelerator(
    KeyStroke.getKeyStroke(KeyEvent.VK_M, ActionEvent.CTRL_MASK));
  jmiDiv.setAccelerator(
    KeyStroke.getKeyStroke(KeyEvent.VK_D, ActionEvent.CTRL_MASK));

  // Panel p1 to hold text fields and labels
  JPanel p1 = new JPanel();
  p1.setLayout(new FlowLayout());
  p1.add(new JLabel("Number 1"));
  p1.add(jtfNum1 = new JTextField(3));
  p1.add(new JLabel("Number 2"));
  p1.add(jtfNum2 = new JTextField(3));
  p1.add(new JLabel("Result"));
  p1.add(jtfResult = new JTextField(4));
  jtfResult.setEditable(false);

  // Panel p2 to hold buttons
```

```
    JPanel p2 = new JPanel();
    p2.setLayout(new FlowLayout());
    p2.add(jbtAdd = new JButton("Add"));
    p2.add(jbtSub = new JButton("Subtract"));
    p2.add(jbtMul = new JButton("Multiply"));
    p2.add(jbtDiv = new JButton("Divide"));

    getContentPane().add(p1, BorderLayout.NORTH);
    getContentPane().add(p2, BorderLayout.SOUTH);

    // Register listeners
    jbtAdd.addActionListener(this);
    jbtSub.addActionListener(this);
    jbtMul.addActionListener(this);
    jbtDiv.addActionListener(this);
    jmiAdd.addActionListener(this);
    jmiSub.addActionListener(this);
    jmiMul.addActionListener(this);
    jmiDiv.addActionListener(this);
    jmiClose.addActionListener(this);

    p1.addMouseListener(new MouseAdapter() {
      public void mousePressed(MouseEvent e) { // For Motif
        showPopup(e);
      }

      public void mouseReleased(MouseEvent e) { // For Windows
        showPopup(e);
      }
    });
  }

  /** Display popup menu when triggered */
  private void showPopup(java.awt.event.MouseEvent evt) {
    if (evt.isPopupTrigger())
      jPopupMenu1.show(evt.getComponent(), evt.getX(), evt.getY());
  }

  /** Handle ActionEvent from buttons and menu items */
  public void actionPerformed(ActionEvent e) {
    String actionCommand = e.getActionCommand();

    // Handle button events
    if (e.getSource() instanceof JButton) {
```

```
    if ("Add".equals(actionCommand))
      calculate('+');
    else if ("Subtract".equals(actionCommand))
      calculate('-');
    else if ("Multiply".equals(actionCommand))
      calculate('*');
    else if ("Divide".equals(actionCommand))
      calculate('/');
  }
  else if (e.getSource() instanceof JMenuItem) {
    // Handle menu item events
    if ("Add".equals(actionCommand))
      calculate('+');
    else if ("Subtract".equals(actionCommand))
      calculate('-');
    else if ("Multiply".equals(actionCommand))
      calculate('*');
    else if ("Divide".equals(actionCommand))
      calculate('/');
    else if ("Close".equals(actionCommand))
      System.exit(0);
  }
}

/** Calculate and show the result in jtfResult */
private void calculate(char operator) {
  // Obtain Number 1 and Number 2
  int num1 = (Integer.parseInt(jtfNum1.getText().trim()));
  int num2 = (Integer.parseInt(jtfNum2.getText().trim()));
  int result = 0;

  // Perform selected operation
  switch (operator) {
    case '+': result = num1 + num2;
              break;
    case '-': result = num1 - num2;
              break;
    case '*': result = num1 * num2;
              break;
    case '/': result = num1 / num2;
  }

  // Set result in jtfResult
  jtfResult.setText(String.valueOf(result));
```

```
  }

  public static void main(String[] args) {
    Exercise29_2 applet = new Exercise29_2();
    JFrame frame = new JFrame();
    //EXIT_ON_CLOSE == 3
    frame.setDefaultCloseOperation(3);
    frame.setTitle("Exercise29_2");
    frame.getContentPane().add(applet, BorderLayout.CENTER);
    applet.init();
    applet.start();
    frame.setSize(400,320);
    Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
    frame.setLocation((d.width - frame.getSize().width) / 2,
      (d.height - frame.getSize().height) / 2);
    frame.setVisible(true);
  }
}
```

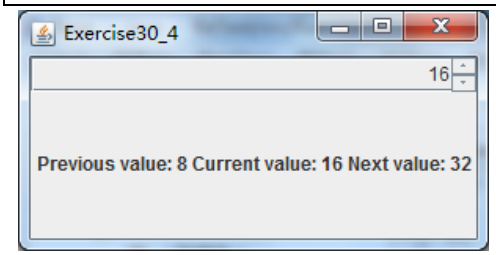



```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class Exercise29_6 {
  public static void main(String[] args) {
    new UserInfoDialog().setVisible(true);
  }
}

class UserInfoDialog extends JDialog {
  private JTextField jtfUsername = new JTextField(20);
  private JPasswordField jtfPassword = new JPasswordField(20);
  private JButton jbtOK = new JButton("OK");
  private JButton jbtCancel = new JButton("Cancel");
  private Login login = new Login();
  class Login {
    String username;
```

```
  String password;
}

public UserInfoDialog() {
  this(null, true);
}

public UserInfoDialog(java.awt.Frame parent, boolean modal) {
  super(parent, modal);
  setSize(200, 120);
  setTitle("Login");

  // Group two labels
  JPanel jpLabels = new JPanel(new GridLayout(2, 1));
  jpLabels.add(new JLabel("Username"));
  jpLabels.add(new JLabel("Password"));

  // Group two text fields
  JPanel jpTextFields = new JPanel(new GridLayout(2, 1));
  jpTextFields.add(jtfUsername);
  jpTextFields.add(jtfPassword);

  // Group jpLabels and jpTextFields
  JPanel panel = new JPanel(new BorderLayout(5, 2));
  panel.add(jpLabels, BorderLayout.WEST);
  panel.add(jpTextFields, BorderLayout.CENTER);

  JPanel jpButtons = new JPanel();
  jpButtons.add(jbtOK);
  jpButtons.add(jbtCancel);

  getContentPane().add(jpButtons, BorderLayout.SOUTH);
  getContentPane().add(panel, BorderLayout.CENTER);

  jbtOK.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
      login.username = jtfUsername.getText().trim();
      login.password = new String(jtfPassword.getPassword());
      setVisible(false);
    }
  });

  jbtCancel.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
```

```
        login = null;
        setVisible(false);
      }
    });
  }

}
```

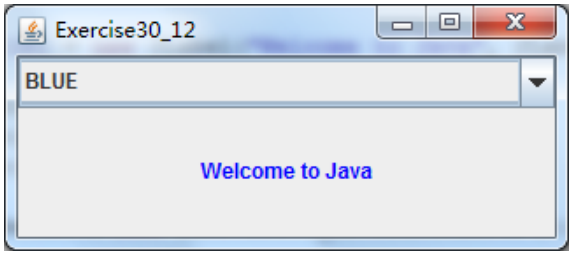
Chapter 30	Exercise 30.4 / 30.12




```
import javax.swing.*;
import javax.swing.event.*;
import java.awt.BorderLayout;

public class Exercise30_4 extends JApplet {
  // Create a JSpinner
  private JSpinner spinner =
    new JSpinner(new SpinnerPowerNumberModel());

  // Create a JLabel
  private JLabel label = new JLabel("", JLabel.CENTER);

  public Exercise30_4() {
    // Add spinner and label to the UI
    getContentPane().add(spinner, BorderLayout.NORTH);
    getContentPane().add(label, BorderLayout.CENTER);

    // Register and create a listener
    spinner.addChangeListener(new ChangeListener() {
      public void stateChanged(javax.swing.event.ChangeEvent e) {
        label.setText("Previous value: " + spinner.getPreviousValue()
          + " Current value: " + spinner.getValue()
          + " Next value: " + spinner.getNextValue());
      }
    });
  }

  public static void main(String[] args) {
```

```
    javax.swing.JFrame frame = new javax.swing.JFrame(
        "Exercise30_4");

    Exercise30_4 applet = new Exercise30_4();

    // Add the applet instance to the frame
    frame.getContentPane().add(applet, java.awt.BorderLayout.CENTER);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    // Invoke init and start
    applet.init();
    applet.start();

    // Display the frame
    frame.setSize(300, 300);
    frame.setVisible(true);
  }
}

class SpinnerPowerNumberModel extends AbstractSpinnerModel {
  private int value = 2;

  public Object getPreviousValue() {
    if (value / 2 == 0)
      return null;

    return new Integer(value / 2);
  }

  public Object getNextValue() {
    return new Integer(value * 2);
  }

  public Object getValue() {
    return new Integer(value);
  }

  public void setValue(Object value) {
    if ((value == null) || !(value instanceof Number)) {
      throw new IllegalArgumentException("illegal value");
    }
    if (!new Integer(this.value).equals(value)) {
      this.value = ((Number)value).intValue();
      fireStateChanged();
```

```
    }
  }
}
```




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

public class Exercise30_12 extends JApplet {
  private JComboBox jcboColor = new JComboBox(new String[] {
    "BLACK", "BLUE", "CYAN", "DARK_GRAY", "GRAY", "GREEN",
    "LIGHT_GRAY", "MAGENTA", "ORANGE", "PINK", "RED", "WHITE",
    "YELLOW"});
  private JLabel jlbl = new JLabel("Welcome to Java", JLabel.CENTER);

  public Exercise30_12() {
    getContentPane().add(jcboColor, BorderLayout.NORTH);
    getContentPane().add(jlbl, BorderLayout.CENTER);

    jcboColor.setRenderer(new Exercise30_12CellRenderer());

    jcboColor.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jcboColor.getSelectedItem().equals("BLACK"))
          jlbl.setForeground(Color.black);
        else if (jcboColor.getSelectedItem().equals("BLUE"))
          jlbl.setForeground(Color.blue);
        else if (jcboColor.getSelectedItem().equals("CYAN"))
          jlbl.setForeground(Color.cyan);
        else if (jcboColor.getSelectedItem().equals("DARK_GRAY"))
          jlbl.setForeground(Color.darkGray);
        else if (jcboColor.getSelectedItem().equals("GRAY"))
          jlbl.setForeground(Color.gray);
        else if (jcboColor.getSelectedItem().equals("GREEN"))
          jlbl.setForeground(Color.green);
        else if (jcboColor.getSelectedItem().equals("LIGHT_GRAY"))
          jlbl.setForeground(Color.lightGray);
        else if (jcboColor.getSelectedItem().equals("MAGENTA"))
```

```
        jlbl.setForeground(Color.magenta);
      else if (jcboColor.getSelectedItem().equals("ORANGE"))
        jlbl.setForeground(Color.orange);
      else if (jcboColor.getSelectedItem().equals("PINK"))
        jlbl.setForeground(Color.pink);
      else if (jcboColor.getSelectedItem().equals("RED"))
        jlbl.setForeground(Color.red);
      else if (jcboColor.getSelectedItem().equals("WHITE"))
        jlbl.setForeground(Color.white);
      else if (jcboColor.getSelectedItem().equals("YELLOW"))
        jlbl.setForeground(Color.yellow);
    }
  });
 }

 public static void main(String[] args) {
  Exercise30_12 applet = new Exercise30_12();
  JFrame frame = new JFrame();
  //EXIT_ON_CLOSE == 3
  frame.setDefaultCloseOperation(3);
  frame.setTitle("Exercise30_12");
  frame.getContentPane().add(applet, BorderLayout.CENTER);
  applet.init();
  applet.start();
  frame.setSize(400, 320);
  Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
  frame.setLocation((d.width - frame.getSize().width) / 2,
                    (d.height - frame.getSize().height) / 2);
  frame.setVisible(true);
 }
}
```

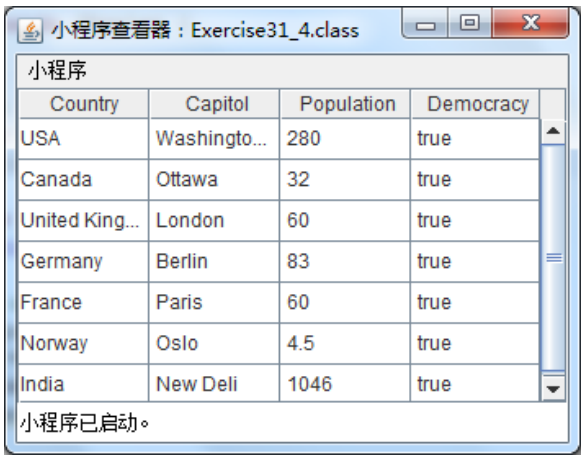
Exercise30_12CellRenderer 类

```
import java.awt.*;
import javax.swing.*;
import javax.swing.border.*;

public class Exercise30_12CellRenderer implements ListCellRenderer {
 JLabel jlbl = new JLabel(" ", JLabel.LEFT);
 Border lineBorder =
  BorderFactory.createLineBorder(Color.black, 1);
 Border emptyBorder =
  BorderFactory.createEmptyBorder(2, 2, 2, 2);
```

```
public Exercise30_12CellRenderer() {
}

public Component getListCellRendererComponent
  (JList list, Object value, int index, boolean isSelected,
   boolean cellHasFocus) {
  //TODO: implement this javax.swing.ListCellRenderer method;
  String colorString = (String)value;
  jlbl.setText(colorString);

  if (colorString.equals("BLACK"))
    jlbl.setForeground(Color.black);
  else if (colorString.equals("BLUE"))
    jlbl.setForeground(Color.blue);
  else if (colorString.equals("CYAN"))
    jlbl.setForeground(Color.cyan);
  else if (colorString.equals("DARK_GRAY"))
    jlbl.setForeground(Color.darkGray);
  else if (colorString.equals("GRAY"))
    jlbl.setForeground(Color.gray);
  else if (colorString.equals("GREEN"))
    jlbl.setForeground(Color.green);
  else if (colorString.equals("LIGHT_GRAY"))
    jlbl.setForeground(Color.lightGray);
  else if (colorString.equals("MAGENTA"))
    jlbl.setForeground(Color.magenta);
  else if (colorString.equals("ORANGE"))
    jlbl.setForeground(Color.orange);
  else if (colorString.equals("PINK"))
    jlbl.setForeground(Color.pink);
  else if (colorString.equals("RED"))
    jlbl.setForeground(Color.red);
  else if (colorString.equals("WHITE"))
    jlbl.setForeground(Color.white);
  else if (colorString.equals("YELLOW"))
    jlbl.setForeground(Color.yellow);

  if (isSelected) {
    jlbl.setBorder(lineBorder);
    if (colorString.equals("BLACK"))
      jlbl.setForeground(Color.black);
    else if (colorString.equals("BLUE"))
      jlbl.setForeground(Color.blue);
    else if (colorString.equals("CYAN"))
```

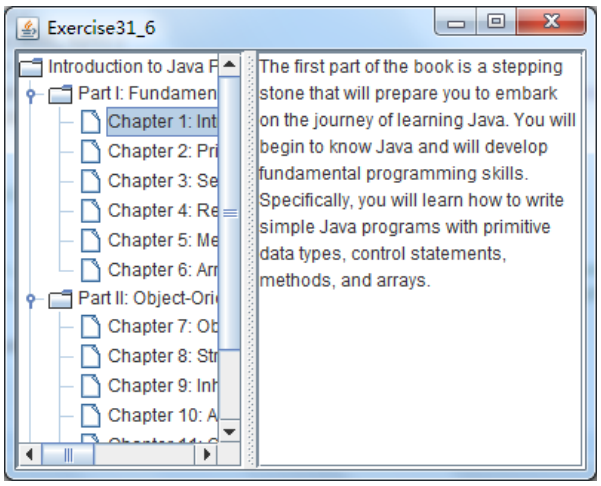
```
        jlbl.setForeground(Color.cyan);
      else if (colorString.equals("DARK_GRAY"))
        jlbl.setForeground(Color.darkGray);
      else if (colorString.equals("GRAY"))
        jlbl.setForeground(Color.gray);
      else if (colorString.equals("GREEN"))
        jlbl.setForeground(Color.green);
      else if (colorString.equals("LIGHT_GRAY"))
        jlbl.setForeground(Color.lightGray);
      else if (colorString.equals("MAGENTA"))
        jlbl.setForeground(Color.magenta);
      else if (colorString.equals("ORANGE"))
        jlbl.setForeground(Color.orange);
      else if (colorString.equals("PINK"))
        jlbl.setForeground(Color.pink);
      else if (colorString.equals("RED"))
        jlbl.setForeground(Color.red);
      else if (colorString.equals("WHITE"))
        jlbl.setForeground(Color.white);
      else if (colorString.equals("YELLOW"))
        jlbl.setForeground(Color.yellow);
    }
    else {
      jlbl.setBorder(emptyBorder);
    }

    if (cellHasFocus)
      jlbl.setBorder(lineBorder);
    else
      jlbl.setBorder(emptyBorder);

    return jlbl;
  }
}
```

Chapter 31	Exercise 31.4 / 31.6




```
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import javax.swing.*;
import java.util.*;

public class Exercise31_4 extends JApplet {
  //Construct the applet
  public Exercise31_4() {
    // Load the data from the file to the vectors
    Vector columnHeader = new Vector();
    Vector rows = new Vector();
    try {
      BufferedReader in = new BufferedReader(
        new FileReader("Exercise31_4Table.txt"));

      // Read the first line as header
      String line = in.readLine();
      StringTokenizer tokens = new StringTokenizer(line, ",\n\r");
      while (tokens.hasMoreTokens())
        columnHeader.addElement(tokens.nextToken());

      while ((line = in.readLine()) != null) {
        Vector row = new Vector();
        tokens = new StringTokenizer(line, ",\n\r");
        while (tokens.hasMoreTokens())
          row.addElement(tokens.nextToken());
        rows.addElement(row);
      }
    }
    catch (Exception ex) {
      ex.printStackTrace();
    }
```

```
    JTable jTable1 = new JTable(rows, columnHeader);
    jTable1.setRowHeight(25);
    getContentPane().add(new JScrollPane(jTable1),
BorderLayout.CENTER);
  }

  //Main method
  public static void main(String[] args) {
    Exercise31_4 applet = new Exercise31_4();
    JFrame frame = new JFrame();
    //EXIT_ON_CLOSE == 3
    frame.setDefaultCloseOperation(3);
    frame.setTitle("Exercise31_4");
    frame.getContentPane().add(applet, BorderLayout.CENTER);
    applet.init();
    applet.start();
    frame.setSize(400, 320);
    Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
    frame.setLocation((d.width - frame.getSize().width) / 2,
      (d.height - frame.getSize().height) / 2);
    frame.setVisible(true);
  }
}
```




```
import javax.swing.*;
import javax.swing.tree.*;
import javax.swing.event.*;
import java.awt.*;

public class Exercise31_6 extends javax.swing.JApplet {
  // Declare a tree model
```

```
  DefaultTreeModel treeModel;

  /** Creates new form Exercise31_6 */
  public Exercise31_6() {
    // Create a tree model and set the tree model in jTree1
    treeModel = createTreeModel();

    initComponents();
  }

  /** This method is called from within the constructor to
   * initialize the form.
   * WARNING: Do NOT modify this code. The content of this method is
   * always regenerated by the Form Editor.
   */
  private void initComponents() {//GEN-BEGIN:initComponents
    jSplitPane1 = new javax.swing.JSplitPane();
    jScrollPane1 = new javax.swing.JScrollPane();
    jTree1 = new javax.swing.JTree();
    jScrollPane2 = new javax.swing.JScrollPane();
    jta = new javax.swing.JTextArea();

    jTree1.setModel(treeModel);
    jTree1.addTreeSelectionListener(new
javax.swing.event.TreeSelectionListener() {
      public void valueChanged(javax.swing.event.TreeSelectionEvent evt)
{
        jTree1ValueChanged(evt);
      }
    });

    jScrollPane1.setViewportView(jTree1);

    jSplitPane1.setLeftComponent(jScrollPane1);

    jta.setLineWrap(true);
    jta.setWrapStyleWord(true);
    jScrollPane2.setViewportView(jta);

    jSplitPane1.setRightComponent(jScrollPane2);

    getContentPane().add(jSplitPane1, java.awt.BorderLayout.CENTER);

  }//GEN-END:initComponents
```

```
  private void jTree1ValueChanged(javax.swing.event.TreeSelectionEvent
evt) {//GEN-FIRST:event_jTree1ValueChanged
    TreePath path = evt.getNewLeadSelectionPath();
    TreeNode treeNode = (TreeNode)path.getLastPathComponent();
    System.out.println("The selected node is " + treeNode.toString());

    if (treeNode.toString().equals
      ("Chapter 1: Introduction to Computers, Programs, and Java")) {
      jta.setText("The first part of the book is a stepping stone that will
prepare you to embark on the journey of learning Java. You will begin to
know Java and will develop fundamental programming skills. Specifically,
you will learn how to write simple Java programs with primitive data types,
control statements, methods, and arrays.");
    }
  }//GEN-LAST:event_jTree1ValueChanged

  private DefaultTreeModel createTreeModel() {
    DefaultMutableTreeNode root = new DefaultMutableTreeNode
      ("Introduction to Java Programming");

    DefaultMutableTreeNode parent = new DefaultMutableTreeNode
      ("Part I: Fundamentals of Programming");
    root.add(parent);

    parent.add(new DefaultMutableTreeNode
      ("Chapter 1: Introduction to Computers, Programs, and Java"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 2: Primitive Data Types and Operations"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 3: Selection Statements"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 4: Repetition Statements"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 5: Methods"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 6: Arrays"));

    parent = new DefaultMutableTreeNode
      ("Part II: Object-Oriented Programming");
    root.add(parent);

    parent.add(new DefaultMutableTreeNode
      ("Chapter 7: Objects and Classes "));
```

```
    parent.add(new DefaultMutableTreeNode
      ("Chapter 8: Strings and Text I/O"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 9: Inheritance and Polymorphism"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 10: Abstract Classes and Interfaces"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 11: OO Analysis and Design"));

    parent = new DefaultMutableTreeNode
      ("Part III: GUI Programming");
    root.add(parent);

    parent.add(new DefaultMutableTreeNode
      ("Chapter 12: GUI Basics"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 13: Graphics"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 14: Event-Driven Programming"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 15: Creating User Interfaces"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 16: Applets"));

    parent = new DefaultMutableTreeNode
      ("Part IV: Exception Handling and Simple IO");
    root.add(parent);

    parent.add(new DefaultMutableTreeNode
      ("Chapter 17: Exceptions and Assertions"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 18: Binary IO"));

    parent = new DefaultMutableTreeNode
      ("Part V: Data Structures and Collections Framework");
    root.add(parent);

    parent.add(new DefaultMutableTreeNode
      ("Chapter 17: Data Structure Implementations"));
    parent.add(new DefaultMutableTreeNode
      ("Chapter 18: Collections Framework"));

    return new DefaultTreeModel(root);
  }
```

```
  // Variables declaration - do not modify//GEN-BEGIN:variables
  private javax.swing.JScrollPane jScrollPane2;
  private javax.swing.JScrollPane jScrollPane1;
  private javax.swing.JSplitPane jSplitPane1;
  private javax.swing.JTextArea jta;
  private javax.swing.JTree jTree1;
  // End of variables declaration//GEN-END:variables

  //Main method
  public static void main(String[] args) {
    Exercise31_6 applet = new Exercise31_6();
    JFrame frame = new JFrame();
    //EXIT_ON_CLOSE == 3
    frame.setDefaultCloseOperation(3);
    frame.setTitle("Exercise31_6");
    frame.getContentPane().add(applet, BorderLayout.CENTER);
    applet.init();
    applet.start();
    frame.setSize(400,320);
    Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
    frame.setLocation((d.width - frame.getSize().width) / 2, (d.height
- frame.getSize().height) / 2);
    frame.setVisible(true);
  }

  //static initializer for setting look & feel
  static {
    try {

//UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName())
;

//UIManager.setLookAndFeel(UIManager.getCrossPlatformLookAndFeelClass
Name());
    }
    catch(Exception e) {
    }
  }
}
```