巻末付録　技術系の数値計算に対する Java 超入門

１．はじめに

本章は通常の Java の解説ではない点を、あらかじめお断りしておく。従来、Java 関連の書籍では、"Hello World"と表示するところから始まって、クラスの概念、インスタンスの生成、継承、・・・等、オブジェクト指向のプログラミングの考え方が展開される場合が多い。Java を用いてソフトウェア（ごく簡単なものも含めて）を作成することを目的とするならば、このような教科書テキストは有用である。しかし、技術者・研究者で、ソフトウェア作成ではなく、単純に数値計算がしたいだけであるならば、上記のテキストの大部分は必要のない知識である。なぜならば、通常、技術者・研究者が計算において必要とする操作の大部分は、以下の３種類のみであるからである。

（１）手続き的な数値の計算
（２）結果の可視化（グラフや画像）
（３）数値データの読出しと保存

きちんとしたソフトウェア作成自体を目的としなければ、プログラムにおいて、ボタン等のコンポーネントの作成や過度の例外処理なども必要ない。単にソースプログラムが書けて、それをコンパイルし、上記３つの操作に対して、プログラムを実行し、結果さえ得られれば良い。プログラムに入力値が必要な場合には、極論すれば、ソースコードに毎回直接書き込み、その都度コンパイルして実行しても、ほとんどの場合、日常必要とする数値計算に支障は起きない。さらに実行プログラムも実行するたびにその都度強制終了するものであってもかまわない。

Java に限る話ではないが、基本的に通常の Windows アプリケーションを作成することを目的としてしまうと、イベントドリブン型の部分、すなわち、ボタンやツールーバー、クリック、ドラッグなどの、本来の数値計算部分とは全く無関係な部分を作りこまなくてはならなくなるために、Windows プログラミングは非常に複雑になる（コンポーネントウェアの概念にてプログラミングの手間はかなり削減されて来てはいるが、根本的に数値計算以外の部分が複雑である点にかわりはない）。もう一度言う。数値計算をしたいだけであるならば、ボタンやツールーバー、クリック、ドラッグなどのイベントドリブン型の部分は全く必要ないので、この部分の学習は時間のムダである（後々、本当に Windows ソフトウェアを作成したくなった時に改めて勉強すれば良い）。

そこで、本章では、上記の３つの操作に限定して、非常に簡単に Java アプリケーションを作成する手法を解説する。なお Java アプレットについては言及しない。Java アプレットはインターネット上で稼動するソフトウェアを簡単に作成できる利点があるが、セキュリティーの関係で、データのディスクへの保存に対する機能がない。これでは数値計算してもデータを保存できないことになるので、上記の（３）の操作が実現できず、技術者・研究者における日常の数値計算の目的からはふさわしくない。したがって、本章では、Java アプリケーションのみについて解説する。ちなみに、Java アプリケーションを Java アプレットに書き直すことは、それほど難しくはなく、また Java アプレットに関しては多くの入門書が出版されているので、興味のある読者はトライしてみることを進める。

２．Java 本体の入手先およびソースコードの実行方法

Java 本体のインストールについては、Sun マイクロシステムズのホームページ

http://java.sun.com/javase/downloads/index.jsp

等を参照していただきたい。なお近年 Java 本体のバージョンナンバーの呼び方、および Java 本体の基本分類が変更となったので、この点も注意しておくと良い。上記から Java 本体をダウンロードしてインストールした後、１つ忘れずにやっておくことは、環境変数 CLASSPATH の設定である（比較的古いバージョンの場合で、最新版は設定不要）。これについても、本稿では具体的な説明はしない。Google などの探索サイトにおいて、キーワード（Java　CLASSPATH）などで簡単に調べることができる（基本的にインターネットで簡単に調べることができる内容については、本稿では説明を省略する）。

次にソースコードの実行方法について説明する。ソースコードの名前を、"prog1.java"としよう。prog1.java の形式は通常のテキストファイルである（Ms-Windows の"メモ帳"などで書ける形式)。ソースコードをコンパイルするために、バッチファイルを作成しておくと良い。著者は、以下のようなバッチファイルを使用している。ファイル名は"vjc.bat"で（内容の形式はテキストファイル)、内容は、

```
set path_1=%path%
path c:¥j2sdk¥bin;c:¥j2sdk¥include;c:¥j2sdk¥lib;
javac -deprecation %1 %2 %3 %4 %5 %6 %7 %8 %9
path ;
path %path_1%
```

である(上記は Java 本体が C ドライブの c:¥j2sdk にインストールされている場合を想定している)。このバッチファイル vjc.bat とソースコード prog1.java は、C:¥tmp にあるものとする。コマンドプロンプト（通称 DOS 窓とも呼ばれる）を起動し、カレントディレクトリを C:¥tmp に移動して、

```
C:¥tmp>vjc prog1.java
```

と入力してリターンすることによって、prog1.java がコンパイルでき、エラーがなければクラスファイル prog1.class が生成される。エラーがある場合には、エラーメッセージとそのエラーが存在するソースコード上の行番号がコマンドプロンプト画面に示されるので、エラーメッセージに従いソースコードを修正すればよい。

プログラムの実行（得られた prog1.class の実行）には、次のバッチファイルを使用すると良い（ファイル名を"vj.bat"とする)。

```
set path_1=%path%
path c:¥j2sdk¥bin;c:¥j2sdk¥include;c:¥j2sdk¥lib;
java %1 %2 %3 %4 %5 %6 %7 %8 %9
path ;
path %path_1%
```

具体的には、

```
C:¥tmp>vj prog1
```

と入力（拡張子の class は必要ない）してリターンすることによって、プログラム（prog1.class）を実行することができる。

３．その他のプログラムの説明

組織形成を計算するプログラムは前章までの章末に記載されている。以下では、計算結果を画像ファイル化するなど、実際の研究に役に立つ代表的なその他のプログラム（以下の３つ）について説明する。

（１）plot_c.java

これは、計算結果の２次元イメージ（この場合は濃度場）を、ディスプレイ上に可視化（単に計算データを表示して見るだけ）するプログラムである。

```
//**** プログラム (plot_c.java) ************************************************
// このプログラムは、濃度場の時間発展の計算結果（２次元）を
// 単に表示するだけのプログラムである。

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

```
import java.io.*;

public class plot_c extends Frame{

    static int ND=80;                    //組織１辺の分割数
    static int nd=ND, ndm=ND-1;
    static int width, height, insetx, insety, xwidth, yheight;
    static double [][] ch=new double[ND][ND];  //組織内の濃度データ配列

    //グローバル変数化
    static double time1;    //計算時間（カウント数）
    Image buff;
    static Graphics bg, g;

    //*************** コンストラクタ  ***************************
    public plot_c(){
        xwidth=400; yheight=400; insetx=4; insety=30;
        width=xwidth+insetx*2;    height=yheight+insetx+insety;
        setSize(width, height);
        setBackground(Color.lightGray);
        setVisible(true);
        buff=this.createImage(width, height);
        bg=buff.getGraphics();
        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e){ System.exit(0); }
        });
    }

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

        plot_c prog=new plot_c();

        int    i, j, k, l;                    //整数
        double sumc;
        String s_data, s_time1;

        try{
            BufferedReader infile=new BufferedReader(new FileReader("test.dat"));
            try{
                while( (s_time1=infile.readLine())!=null ){
                    time1=new Double(s_time1).doubleValue();
                    for(i=0;i<=ndm;i++){
                        for(j=0;j<=ndm;j++){
                            s_data=infile.readLine();    ch[i][j]=new Double(s_data).doubleValue();
                        }
                    }
                    prog.repaint();
//--- 描画が速すぎるので、強制的に 1 秒スリープ  -----------------
                    try{ Thread.sleep(1000); } catch( InterruptedException e){ }
                }
            }
            catch(Exception e){System.out.println(e); System.exit(1);}
            finally{infile.close();}
        }
        catch(Exception e){System.out.println(e); System.exit(1);}
```

```
    }//main

// ***** update 関数のオーバーライド（再描画時における画面のチラツキ防止）*******
    public void update(Graphics g){paint(g);}

// ****  濃度場の描画  ***************************************************
    public void paint(Graphics g){
        bg.clearRect(0, 0, width, height);
        bg.setColor(Color.lightGray);

        int i, j, ii, jj;
        int icol;
        int ixmin=0, iymin=0, igx, igy, irad0;
        int ixmax, iymax;
        double c, x, xmax, xmin, y, ymax, ymin, rad0;

        xmin=0.0; xmax=1.0;    ymin=0.0; ymax=1.0;
        ixmax=xwidth;    iymax=yheight;

        System.out.println(time1);
        rad0=1.0/(double)nd/2.0;
        irad0=1+(int)( ((double)ixmax-(double)ixmin)/(xmax-xmin)*rad0 );

        for(i=0;i<=nd;i++){
            for(j=0;j<=nd;j++){
                x=1.0/(double)nd*(double)i+rad0;
                igx=(int)( ((double)ixmax-(double)ixmin)*(x-xmin)/(xmax-xmin)+(double)ixmin );
                y=1.0/(double)nd*(double)j+rad0;
                igy=(int)( ((double)iymax-(double)iymin)*(y-ymin)/(ymax-ymin)+(double)iymin );
                ii=i; jj=j;
                if(i==nd){ii=0;}    if(j==nd){jj=0;}
                //icol=(int)(255.0*ch[ii][jj]);
                icol=(int)(255.0*(1.0-ch[ii][jj]));
                if(icol>=255){icol=255;}    if(icol<=0){icol=0;}
                bg.setColor(new Color(icol,icol,icol));
                bg.fillRect(insetx+igx-irad0,insety+igy-irad0, irad0*2, irad0*2);
                //bg.setColor(Color.blue);
                //bg.setColor(new Color(0,0,icol));
            }
        }
        g.drawImage(buff, 0, 0, this);
    }

//************[濃度場データの保存]**********************************
    private void datsave(){
        int i,j;

        try{
            PrintWriter outfile= new PrintWriter(
                new BufferedWriter(new FileWriter("test.dat", true)) );

            outfile.println(time1);
            //outfile.println(time1+'¥n');
            for(i=0;i<=ndm;i++){
                for(j=0;j<=ndm;j++){
```

```
                outfile.println(ch[i][j]);   //濃度場の書き込み
            }
        }
        outfile.close();
    }
    catch(Exception e){System.out.println(e);System.exit(1);}
  }

//*************************************************************
}//plot_c

//**** プログラム終了 ************************************************************
```

（２）bplot_c.java

これは、計算結果（この場合は濃度場）をディスプレイ上で可視化し、かつプログラム中で指定した時間ステップの図をイメージファイルに変換してディスク内に保存するプログラムである。

```
//**** プログラム (bplot_c.java) ************************************************
//ディスプレイ上に表示される部分は上下左右が若干欠けるが、
//image 全体は欠けることなく保存される点に注意。

import javax.imageio.*;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.awt.image.*;

public class bplot_c extends Frame{

  static int ND=80;                    //組織１辺の分割数
  static int nd=ND, ndm=ND-1;
  static int width, height, insetx, insety, xwidth, yheight;
  static double [][] ch=new double[ND][ND];  //組織内の濃度データ配列

  //グローバル変数化
  static double time1;   //計算時間（カウント数）
  //Image buff;
  static BufferedImage buff;
  static Graphics bg, g;

  //*************** コンストラクタ ***************************
  public bplot_c(){
    xwidth=400; yheight=400; insetx=0; insety=0;
    //xwidth=400; yheight=400; insetx=4; insety=30;
    width=xwidth+insetx*2;    height=yheight+insetx+insety;
    setSize(width, height);
    setBackground(Color.lightGray);
    setVisible(true);
    //buff=this.createImage(width, height);
    buff=new BufferedImage(width, height, BufferedImage.TYPE_3BYTE_BGR);
    bg=buff.getGraphics();
    addWindowListener(new WindowAdapter(){
      public void windowClosing(WindowEvent e){ System.exit(0); }
    });
```

```
    }

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

        bplot_c prog=new bplot_c();

        int    i, j, k, l;                    //整数
        int    ii=1;                        //整数
        double sumc;
        String str_data, str_time1;
        String jpg_file0, jpg_file1, jpg_file2;
        int [] jpg_n=new int[100];
        File outfile;

//--- 各画像ファイル名の共通部分 ------------------------------------
        jpg_file1="Image_";   jpg_file0=jpg_file1;

//--- 読み出す画像ファイルのカウント数(自然数) ------------------------
//---  （この値が保存される画像ファイル名に追加される）  ----------------
        jpg_n[1]=0;
        jpg_n[2]=1000;
        jpg_n[3]=2000;
        jpg_n[4]=5000;
        jpg_n[5]=10000;
        //jpg_n[6]=30000;
        //jpg_n[7]=3000;
        //jpg_n[8]=3000;
        //jpg_n[9]=3000;
        //jpg_n[10]=3000;
        //jpg_n[11]=3000;
        //jpg_n[12]=3000;
        //jpg_n[13]=3000;
        //jpg_n[14]=3000;
        //jpg_n[15]=3000;
        //jpg_n[16]=3000;
        //jpg_n[17]=3000;
        //jpg_n[18]=3000;

//---------------------------------------------------------------------
        try{
            BufferedReader infile=new BufferedReader(new FileReader("test.dat"));
            try{
                while( (str_time1=infile.readLine())!=null ){
                    time1=new Double(str_time1).doubleValue();
                    for(i=0;i<=ndm;i++){
                        for(j=0;j<=ndm;j++){
                            str_data=infile.readLine();
                            ch[i][j]=new Double(str_data).doubleValue();
                        }
                    }
                    if((int)time1==jpg_n[ii]){
                        prog.update_draw(g);
                        jpg_file2=Integer.toString(jpg_n[ii]);   jpg_file1=jpg_file1+jpg_file2+".jpg";
                        outfile=new File(jpg_file1);   ImageIO.write(buff, "jpeg", outfile);
                        ii=ii+1;   jpg_file1=jpg_file0;
```

```
                }
            }
        }
        catch(Exception e){System.out.println(e); System.exit(1);}
        finally{infile.close();}
    }
    catch(Exception e){System.out.println(e); System.exit(1);}

}//main


// ********************************************************************
    public void update_draw( Graphics g ){ g=getGraphics(); draw(g); }

// ****  濃度場描画  ******************************************************
    public void draw(Graphics g){
        bg.clearRect(0, 0, width, height);
        bg.setColor(Color.lightGray);

        int i, j, ii, jj;
        int icol;
        int ixmin=0, iymin=0, igx, igy, irad0;
        int ixmax, iymax;
        double c, x, xmax, xmin, y, ymax, ymin, rad0;

        xmin=0.0; xmax=1.0;    ymin=0.0; ymax=1.0;
        ixmax=xwidth;    iymax=yheight;

        System.out.println(time1);
        rad0=1.0/(double)nd/2.0;
        irad0=1+(int)( ((double)ixmax-(double)ixmin)/(xmax-xmin)*rad0 );

        for(i=0;i<=nd;i++){
            for(j=0;j<=nd;j++){
                x=1.0/(double)nd*(double)i+rad0;
                igx=(int)( ((double)ixmax-(double)ixmin)*(x-xmin)/(xmax-xmin)+(double)ixmin );
                y=1.0/(double)nd*(double)j+rad0;
                igy=(int)( ((double)iymax-(double)iymin)*(y-ymin)/(ymax-ymin)+(double)iymin );
                ii=i; jj=j;
                if(i==nd){ii=0;}    if(j==nd){jj=0;}
                //icol=(int)(255.0*ch[ii][jj]);
                icol=(int)(255.0*(1.0-ch[ii][jj]));
                if(icol>=255){icol=255;}    if(icol<=0){icol=0;}
                bg.setColor(new Color(icol,icol,icol));
                bg.fillRect(insetx+igx-irad0,insety+igy-irad0, irad0*2, irad0*2);
                //bg.setColor(Color.blue);
                //bg.setColor(new Color(0,0,icol));
            }
        }
        g.drawImage(buff, 0, 0, this);
    }

//************************************************************
}// bplot_c
//****  プログラム終了  ************************************************************
```

（3）F_curve.java

これは、自由エネルギーの計算や自由エネルギーの共通接線の計算、曲線のグラフ表示、データの保存および読み出しなどを行うプログラムである。

```
//**** プログラム (F_curve.java) ********************************************
// このプログラムは、以下の一連の操作を行っている。
//    (1) 濃度 c の関数 G(c)を計算し、c と G(c)の離散データを配列に入力
//    (2) c と G(c)の配列を描画
//    (3) c と G(c)の配列をディスクに保存
//    (4) 保存された c と G(c)の配列を読み出し、別の配列に入力
//    (5) 上記の別の配列のデータを描画
// したがって、データの数値計算、グラフ化、保存、読出しの一連の操作が
// 含まれている。通常の技術系の計算では、以上が出来れば、ほとんどの作業が
// 可能となる。

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

public class F_curve extends Frame{

    static int ND=201;     //濃度の分割数+1
    static int ndm=ND-1; //濃度の分割数
    static int width;     // Window 全体の幅
    static int height;// Window 全体の高さ
    static int xwidth;   // 描画領域の幅
    static int yheight;   // 描画領域の高さ
    static int insetx;// Window の枠の幅（左右および下）
    static int insety;// Window の枠の幅（上）
    static double PI=3.141592;
    static double RR=8.3145;    //ガス定数
    static double [] ch=new double[ND];       //濃度
    static double [] Fh=new double[ND];       //自由エネルギー
    static double [] c2h=new double[ND]; //濃度
    static double [] F2h=new double[ND];//自由エネルギー
    static int color_flg=0; //(ch,Fh)を描くか、(c2h,F2h)を描くかを区別するフラグ
    static Graphics g;           //自由うエネルギー曲線画面のグラフィックスオブジェクト

//*************** コンストラクタ ***************************
    public F_curve(){
        xwidth=600; yheight=400; insetx=4; insety=30;
        width=xwidth+insetx*2;    height=yheight+insetx+insety;
        setSize(width, height);
        setBackground(Color.white);
        setVisible(true);
        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e){ System.exit(0); }
        });
    }

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

```
        F_curve prog=new F_curve();   //F_curve のインスタンス prog を生成

        int i;                    //整数
        double temp, c; //温度, 組成
        String s_temp;  //温度（文字列）

//---- 温度の入力 ------------------------------------------
        BufferedReader input=new BufferedReader(new InputStreamReader(System.in));
        s_temp="1000.0";
        try{ System.out.print("temp(1000.0)/K =    "); s_temp=input.readLine(); }
        catch(IOException e){ System.out.println("Exception : "+e); }
        temp=new Double(s_temp).doubleValue();

//--- 自由エネルギーの計算 ---------------------------------
        for(i=0;i<=ndm;i++){
            c=ch[i]=(double)i/(double)ndm;
            if(c==0.0){c=1.0e-6;}    if(c==1.0){c=1.0-1.0e-6;}
            Fh[i]=prog.G(c,temp);    //自由エネルギー
            //System.out.println(Fh[i]);
        }

//--- 自由エネルギーの描画 ---------------------------------
        prog.repaint();

//--- データの保存 ----------------------------------------
        prog.datsave();

//--- 描画が速すぎるので、強制的に２秒スリープ ----------------
        try{ Thread.sleep(2000); }          // 2 seconds
        catch( InterruptedException e){ }

//--- 上記で保存したデータを別の配列に読み込み ----------------
        prog.datin();

//--- 新しく読み込んだ配列の自由エネルギーを描画 --------------
        color_flg=1; prog.repaint();

//-------------------------------------------------------------
}//main

// ****[自由エネルギー関数]*********************************************
    double G(double c, double temp){
        double c1, c2;
        double L0, gc;

        if(c==0.0){c=1.0e-10;}    if(c==1.0){c=1.0-1.0e-10;}
        c1=1.0-c; c2=c;
        L0=2.5e+04;
        gc=L0*c1*c2+RR*temp*(c1*Math.log(c1)+c2*Math.log(c2));
        return(gc);
    }

// *****[自由エネルギーのグラフ描画]*****************************
    public void paint(Graphics g){
```

```
        g.clearRect(0, 0, width, height);

        int i, i1, i2;
        double xmax, xmin, dx, ymax, ymin, dy;
        double gx1, gy1, gx2, gy2;
        int ixmax, iymax, ixmin, iymin, igx1, igy1, igx2, igy2;
        int idx, idy, ir;

        xmin=0.0;       xmax=1.0;       dx=0.1;
        ymin=-1000.0; ymax=1000.0; dy=100.0;
        ixmin=0; iymin=0;    ixmax=xwidth; iymax=yheight;
        ir=4;

        idx=(int)(ixmax*(dx/(xmax-xmin))+0.5);
        idy=(int)(iymax*(dy/(ymax-ymin))+0.5);

        g.setColor(Color.white);    g.fillRect(insetx, insety, xwidth, yheight);

        g.setColor(Color.lightGray);
        g.drawRect(insetx, insety, xwidth, yheight);
        for(i=0;i<=ixmax;i+=idx){ g.drawLine(insetx+i, insety+iymin, insetx+i, insety+iymax); }
        for(i=0;i<=iymax;i+=idy){ g.drawLine(insetx+ixmin, insety+i, insetx+ixmax, insety+i); }

//-------------------------------------------------------------------------------------------
        if(color_flg==0){
            g.setColor(Color.red);
            for(i=0;i<ndm;i++){
                i1=i; i2=i+1;
                gx1=ch[i1];    gy1=Fh[i1];       gx2=ch[i2];    gy2=Fh[i2];

                igx1=(int)( ((double)ixmax-(double)ixmin)*(gx1-xmin)/(xmax-xmin)+(double)ixmin );
                igy1=(int)( (double)iymin+(double)iymax
                        -(((double)iymax-(double)iymin)/(ymax-ymin)*(gy1-ymin)+(double)iymin) );
                igx2=(int)( ((double)ixmax-(double)ixmin)*(gx2-xmin)/(xmax-xmin)+(double)ixmin );
                igy2=(int)( (double)iymin+(double)iymax
                        -(((double)iymax-(double)iymin)/(ymax-ymin)*(gy2-ymin)+(double)iymin) );
                g.drawLine(insetx+igx1, insety+igy1, insetx+igx2, insety+igy2);
            }
        }
        else{
            g.setColor(Color.blue);
            for(i=0;i<=ndm;i++){
                gx1=c2h[i];    gy1=F2h[i];
                igx1=(int)( ((double)ixmax-(double)ixmin)*(gx1-xmin)/(xmax-xmin)+(double)ixmin );
                igy1=(int)( (double)iymin+(double)iymax
                        -(((double)iymax-(double)iymin)/(ymax-ymin)*(gy1-ymin)+(double)iymin) );
                g.fillOval(insetx+igx1-ir, insety+igy1-ir, 2*ir, 2*ir);
            }
        }

    }

//************[濃度場データの保存]**********************************
    private void datsave(){
        int i;

        try{
```

```
            PrintWriter outfile= new PrintWriter(
                new BufferedWriter(new FileWriter("ini000.dat")) );//上書きの場合
                //new BufferedWriter(new FileWriter("ini000.dat", true)) );//追記の場合

            for(i=0;i<=ndm;i++){
                outfile.println(ch[i]);  //濃度場の書き込み
                outfile.println(Fh[i]);  //濃度場の書き込み
            }
            outfile.close();
        }
        catch(Exception e){System.out.println(e);System.exit(1);}
    }

//************[濃度場データの読込み]***********************************
    private void datin(){
        int i;
        double sumc;
        String s_data;

        try{
            BufferedReader infile=new BufferedReader(new FileReader("ini000.dat"));
            try{
                for(i=0;i<=ndm;i++){
                    s_data=infile.readLine();    c2h[i]=new Double(s_data).doubleValue();
                    s_data=infile.readLine();    F2h[i]=new Double(s_data).doubleValue();
                }
            }
            catch(Exception e){System.out.println(e); System.exit(1);}
            finally{infile.close();}
        }
        catch(Exception e){System.out.println(e); System.exit(1);}
    }

//*****************************************************************
}//F_curve
//**** プログラム終了 ***************************************************************
```

以上のソースコードを修正・改造することによって、計算結果をグラフ化、またイメージ化するプログラムを自由に作成することができる。

参考図書

(1) 宮坂雅輝,「エッセンシャル Java (2nd edition)」, ソフトバンククリエイティブ, (2003).
(2) 桑原恒夫,「3 日で解る Java―例題学習方式(第 2 版版)」, 共立出版, (2000).
(3) 山本芳人,「Java による図形処理入門」, 工学図書, (1998).
(4) 峯村吉泰,「Java で学ぶシミュレーションの基礎」, 森北出版, (2006).
(5) 峯村吉泰,「Java によるコンピュータグラフィックス―基礎からシミュレーションの可視化まで」, 森北出版, (2003).
(6) 峯村吉泰,「Java による流体・熱流動の数値シミュレーション」, 森北出版, (2001).
(7) 矢部 孝, 尾形陽一, 滝沢研二,「CIP 法と Java による CG シミュレーション」, 森北出版, (2007).