

Chapter 5. AWTを使った描画

5.1. AWTパッケージ

■AWTパッケージとグラフィックキャンパスクラス

setColor, drawStringといったメソッドは、前にも紹介しましたが、これは、JavaのAWT (Abstract Window Toolkit) と呼ばれるクラスライブラリ (パッケージ) の中のグラフィックス (Graphics) クラスの中に定義されているものです。このクラスには、いろいろな描画を行なうためのメソッドが他にも用意されています。ここで、それらのうちの幾つかを紹介します。

まず、このAWTのクラスを使うときには、次のように、import文を使って、クラスを用いると指定を予めしておく必要があります。

```
import java.awt.*; // これからAWTのクラスを使う準備をせよ
```

この指定の「*」の部分は、すべてのクラスを使うという意味です。例えば、Graphicsクラスしか使用しないのであれば、次のように書くこともできますが、どのクラスを利用するか予め見当がつくことが少ないので、たいていは「*」で指定することの方が多いようです。

```
import java.awt.Graphics; // AWTパッケージのGraphicsクラスを使う準備をせよ
```

■図形を描くメソッド

図形を描くメソッドを紹介していきます。これらのメソッドは、paintメソッドでパラメータとして受け取ったグラフィックスオブジェクトを描画領域の対象としていますので注意してください。例では、gcという変数を用いて、このグラフィックスオブジェクトのことを指し示すこととします。説明では、ある点のx座標とy座標の組を、(x, y)という形で表現します。座標は、ドット単位 (整数あるいは整数の式) で記述します。

★線を描くメソッド

▼drawLine(始点のx座標 , 始点のy座標 , 終点のx座標 , 終点のy座標);
→始点から終点まで線を引きます。

始点と終点は、どちらをどちらにしても同じです。ですから、次の2つのメソッド呼出しは共に同じ直接を引くことになります。

```
gc.drawLine( 20, 30, 200, 180 );  
gc.drawLine( 200, 180, 20, 30 );
```

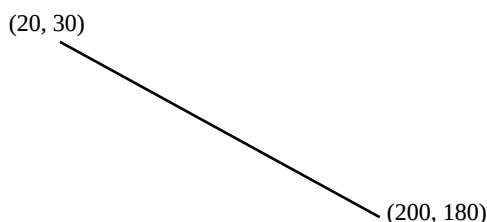


図5-1 線の描画例

★四角形を描くメソッド

▼drawRect(始点のx座標 , 始点のy座標 , 幅 , 高さ);
→始点から、幅・高さで指定された四角形の枠を描きます。

始点は、左上の角のことです。drawLineとdrawRectでは、3番目と4番目のパラメータの意味が違うことに注意してください。例えば、以下の2つのメソッド呼出しでは右下の端が違う位置になります。

```
gc.drawLine( 10, 10, 100, 100 );    // (10, 10) → (100, 100)
gc.drawRect( 10, 10, 100, 100 );    // (10, 10) → (110, 110)
```

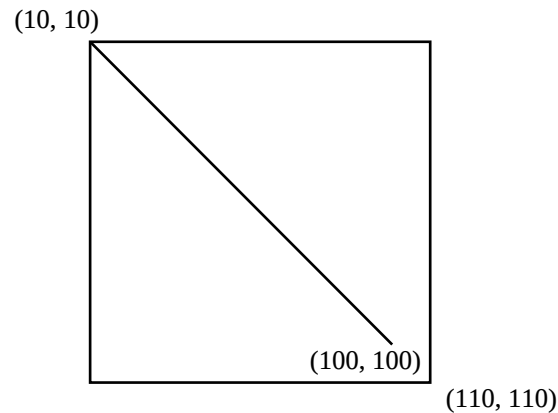


図5-2 四角形と線の描画の差

★楕円や正円を描くメソッド

▼drawOval(始点のx座標 , 始点のy座標 , 外接する四角形の幅 , 外接する四角形の高さ);
→楕円や正円を描きます

円から見て外接する四角形の幅と高さが等しければ、正円になります。例えば、次のメソッド呼出しでは、幅80ドット、高さ50ドットの四角形に内接する円を描きます。これが、例えば、幅80ドット、高さ80ドットの場合には、正円になります。

```
gc.drawOval( 10, 10, 80, 50 );
```

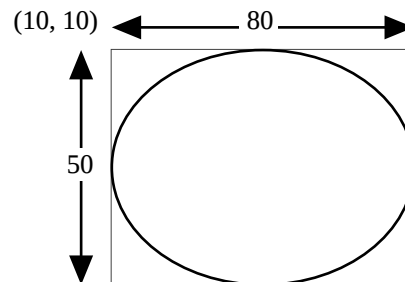


図5-3 楕円の描画例

★弧を描くメソッド

▼drawArc(始点のx座標 , 始点のy座標 , 外接四角形の幅 , 外接四角形の高さ , 開始する角度 , 角度差);
→弧を描きます。

開始角度や角度差は、通常角度指定 (degree) で行ないます。整数あるいは整数の式で記述します。-360° ~ 360° の範囲で指定します。極座標と同じように反時計回りになっています。開始角度の0° は、右の水平方向になっています。

```
gc.drawArc( 20, 20, 70, 60, 135, 90 );
```

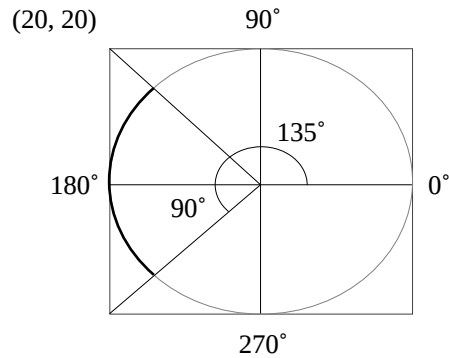


図5-4 弧の角度と描画例

★丸みを帯びた四角形を描画するメソッド

▼drawRoundRect(始点のx座標 , 始点のy座標 , 幅 , 高さ , 丸みの幅 , 丸みの高さ);
→始点から、幅・高さで指定された丸みを持った四角形を描く。

丸みの幅と高さも整数式で指定します。もし、幅や高さに対して、丸みの幅や高さがその半分の大きさになってしまうと、俵型を表示することになるでしょう。

```
gc.drawRoundRect( 100, 100, 50, 40, 10, 10 );
```

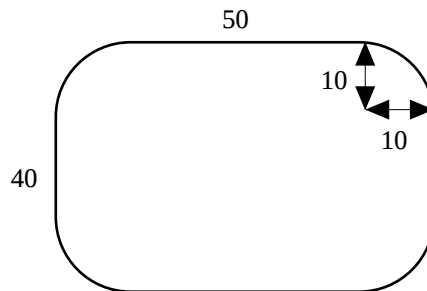


図5-5 丸みを帯びた四角形の描画例

★3次元的な四角形を描画するメソッド

▼draw3DRect(始点のx座標 , 始点のy座標 , 幅 , 高さ , 凹凸の指定);
→3次元的な四角形を描画します。

凹凸の指定は、論理値で行います。**true**だと、飛び出して見えます。**false**だと窪んで見えます。ただし、色を灰色系（Color.grayなど）にしないとそれっぽく見えないので注意が必要です。

```
gc.setColor( Color.gray );
gc.draw3DRect( 10, 10, 100, 100, true );
gc.draw3DRect( 50, 50, 20, 20, false );
```

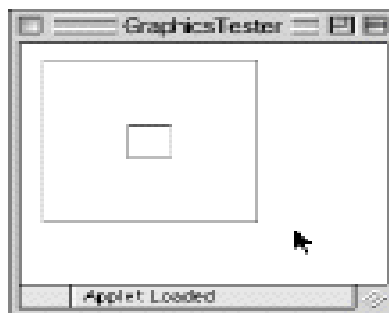


図5-6 3次元的な四角形の描画例

■塗りつぶしで図形を描くメソッド

枠を書くメソッドに対応して、その内部を塗りつぶすメソッドも次のようなものが用意されています。

```
fillRect( 始点のx座標 , 始点のy座標 , 幅 , 高さ );  
fillOval( 始点のx座標 , 始点のy座標 , 外接四角形の幅 , 外接四角形の高さ );  
fillArc( 始点のx座標 , 始点のy座標 , 幅 , 高さ , 開始角度 , 角度差 );  
fillRoundRect( 始点のx座標 , 始点のy座標 , 幅 , 高さ , 丸みの幅 , 丸みの高さ );  
fill3DRect( 始点のx座標 , 始点のy座標 , 幅 , 高さ , 凹凸の指定 );
```

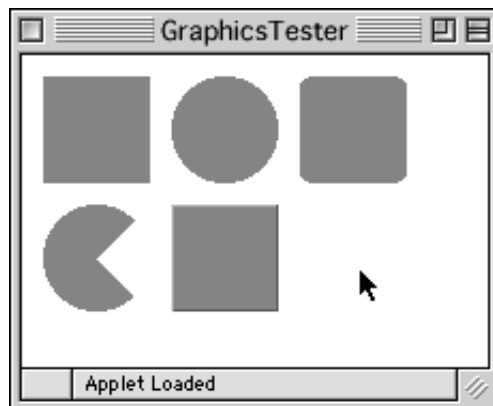


図5-7 塗りつぶしの描画例

■彩色と描画

描画や塗りつぶしのときの色を変えるためには、予め色を設定しておく必要があります。setColorメソッドで色を設定した場合は、その後の描画はすべてその色で行なわれます。別の色にしたければ、またsetColorメソッドを使って色を設定する必要があります。なお、アプレットが起動されたときの最初の描画色は黒色に設定されています。

```
import java.awt.*;  
import java.applet.*;  
  
public class ColoredDrawing extends Applet {  
    public void paint( Graphics gc ) {  
        gc.setColor( Color.red );           // 以降は赤色で描画される  
        gc.drawLine( 0, 0, 100, 100 );  
        gc.drawRect( 100, 0, 100, 100 );  
        gc.setColor( Color.green );         // 以降は緑色で描画される  
        gc.drawLine( 250, 0, 350, 100 );  
        gc.drawRect( 350, 0, 100, 100 );  
    }  
}
```

■コピーエリアなど

描画以外にもいろいろなメソッドが用意されています。

▼clearRect(始点のx座標 , 始点のy座標 , 幅 , 高さ);
→四角形の領域を背景色で塗りつぶします。

▼copyArea(始点のx座標 , 始点のy座標 , 幅 , 高さ , 横の移動量 , 縦の移動量);
→四角形の領域に書かれている内容を縦横に移動した後コピーして表示します。

copyAreaは、移動した後の四角形の領域が元の四角形の領域と重なるときは、移動後の方が優先されますので、注意してください。なお、移動量はマイナスの値も取ることができます。以下のサンプルプログラムを実行してみたり、移動量のパラメータの値を変えたりして、確かめてみてください。

```

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

public class GraphicsTester extends Applet {
    public void paint( Graphics gc ){
        gc.fillRect( 10, 10, 80, 80 );
        gc.setColor( Color.red );
        gc.fillOval( 50, 50, 80, 80 );
        gc.copyArea( 10, 10, 120, 120, 150, 20 ); // 横方向150, 縦方向20の位置にコピー
    }
}

```

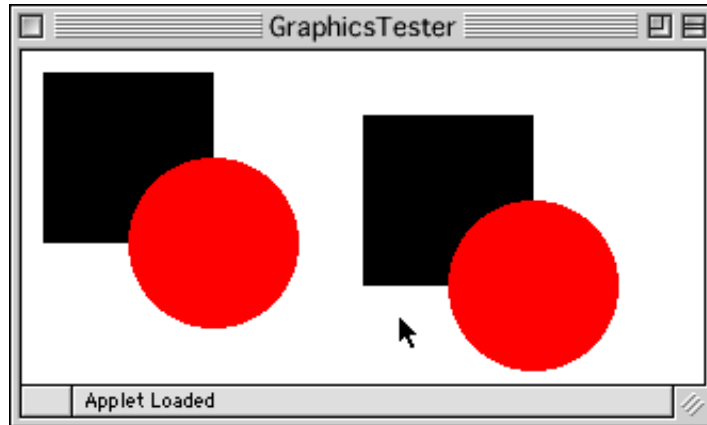


図5-8 コピーエリアの実行例

5.2. 様々な図を描くアプレットの例

★例題1 対数グラフの方眼紙を描くアプレット

対数グラフとは、 $y = e^x$ みたいなべき乗を基本にした関係があるときに、その関係を記すために利用する蔵うフのことです。ここでは、基数を2としています。変数 x の値の変化に注目してください。縦方向と、横方向の両方を対数グラフを描けるようにしています。例えば、縦方向は通常のグラフのための方眼紙にしたい場合はどうするのか考えてみなさい。

```

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

public class Exponential extends Applet {
    public void paint( Graphics gc ){
        int x;
        x = 1;
        gc.drawLine( x, 1, x, 128 ); gc.drawLine( 1, x, 128, x );
        x = x * 2;
        gc.drawLine( x, 1, x, 128 ); gc.drawLine( 1, x, 128, x );
        x = x * 2;
        gc.drawLine( x, 1, x, 128 ); gc.drawLine( 1, x, 128, x );
        x = x * 2;
        gc.drawLine( x, 1, x, 128 ); gc.drawLine( 1, x, 128, x );
        x = x * 2;
        gc.drawLine( x, 1, x, 128 ); gc.drawLine( 1, x, 128, x );
        x = x * 2;
        gc.drawLine( x, 1, x, 128 ); gc.drawLine( 1, x, 128, x );
        x = x * 2;
        gc.drawLine( x, 1, x, 128 ); gc.drawLine( 1, x, 128, x );
        x = x * 2;
        gc.drawLine( x, 1, x, 128 ); gc.drawLine( 1, x, 128, x );
        x = x * 2;
        gc.drawLine( x, 1, x, 128 ); gc.drawLine( 1, x, 128, x );
    }
}

```

}

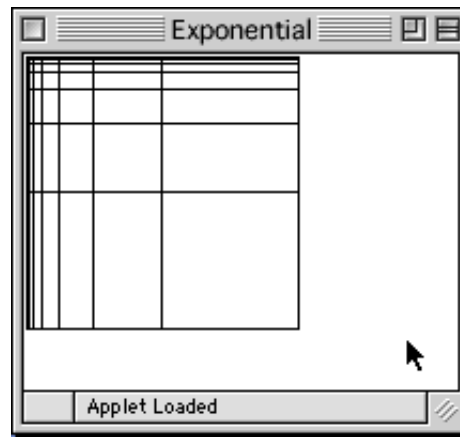


図5-9 対数アプレットの描画例

★例題2 四角形を描くアプレット

枠の中に枠を描いていきます。始点の座標と幅と高さが、変数 x を元に、どのように変化しているのか考えてみてください。余裕があれば、drawRectを、drawRoundRectに変えてみてください。その場合は、丸みの指定も忘れずに。

```
import java.awt.*;  
import java.applet.*;  
  
public class Pyramid extends Applet {  
    public void paint( Graphics gc ) {  
        int x;  
        x = 10;  
        gc.drawRect( x, x, 180 -x*2, 180 -x*2 );  
        x = x + 10;  
        gc.drawRect( x, x, 180 -x*2, 180 -x*2 );  
        x = x + 10;  
        gc.drawRect( x, x, 180 -x*2, 180 -x*2 );  
        x = x + 10;  
        gc.drawRect( x, x, 180 -x*2, 180 -x*2 );  
        x = x + 10;  
        gc.drawRect( x, x, 180 -x*2, 180 -x*2 );  
        x = x + 10;  
        gc.drawRect( x, x, 180 -x*2, 180 -x*2 );  
        x = x + 10;  
        gc.drawRect( x, x, 180 -x*2, 180 -x*2 );  
    }  
}
```

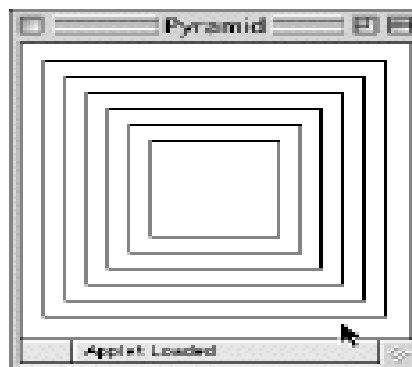


図5-10 ピラミッドアプレットの描画例

★例題3 円や弧を描くサンプル

何故、drawArcでは、始点の座標を変数 x が持つ値の半分にしているのでしょうか？例題2の始点の位置の変

化と、幅と高さの変化の関係を元に考えてみなさい。また、開始角度と角度差も、半分の関係にあります。いろいろ、パラメータの式を変えてみて、どのように描画されるのか考察してみてください。

```
import java.awt.*;
import java.applet.*;

public class ArcCircle extends Applet {
    public void paint(Graphics gc ){
        int x;
        x = 10;
        gc.drawOval( 10, 10, x, x );
        gc.drawArc( 150-x/2, 150-x/2, 100+x, 100+x, -x/4, x/2 );
        x = x + 30;
        gc.drawOval( 10, 10, x, x );
        gc.drawArc( 150-x/2, 150-x/2, 100+x, 100+x, -x/4, x/2 );
        x = x + 30;
        gc.drawOval( 10, 10, x, x );
        gc.drawArc( 150-x/2, 150-x/2, 100+x, 100+x, -x/4, x/2 );
        x = x + 30;
        gc.drawOval( 10, 10, x, x );
        gc.drawArc( 150-x/2, 150-x/2, 100+x, 100+x, -x/4, x/2 );
    }
}
```

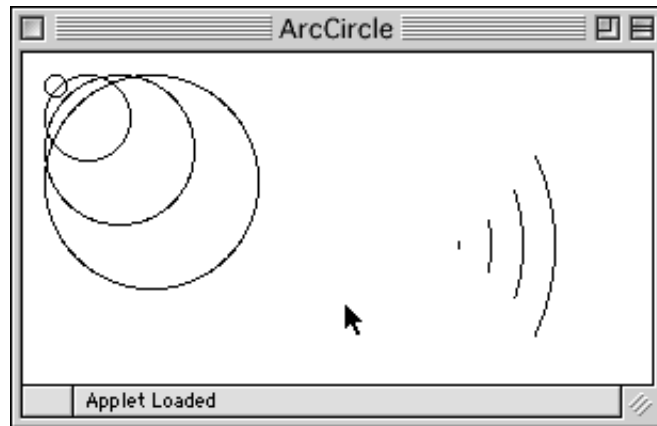


図5-11 円弧アプレットの描画例

5.3. まとめと課題

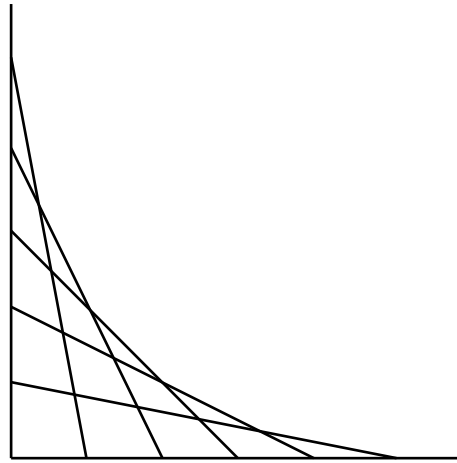
■まとめ

- ★AWTを使う準備
プログラムの先頭で、**import java.awt.*;**をいれておく。
- ★枠線を表示するメソッドには次のようなものがある。
drawLine, drawRect, drawOval, drawArc, drawRoundRect, draw3DRect
- ★中を塗りつぶして表示するメソッドには次のようなものがある。
fillRect, fillOval, fillArc, fillRoundRect, fill3DRect
- ★彩色と描画
別の色で表示したいときは、描画する前に必ず色を変えておく。

■課題

5-1.
変数を1つまたは2つ使って、次のような図を書くアプレットを作成しなさい。高さ・幅は表示する位置は各自で決めて構いません。好みに応じて、色を変えても構いません。あるいは、線の数をもっと緻密に増やしても構いません。クラス名は、Russianとしなさい（1970年の万国博覧会のソ連館がこれに似ていたことから）。ただし、クラス名は別の名前にしても構いません。

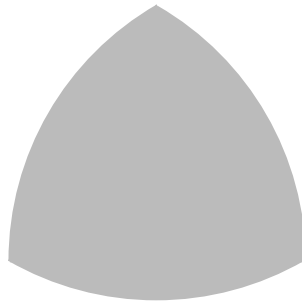
ヒント：例えば終点のx座標は20ずつ増えて行くようにし、始点のy座標は20ずつ増えて行くようにすればいいのではないのでしょうか？



問題の図

5-2.

弧を塗りつぶして描くメソッドを用いて、次のような図を描くようなアプレットを作成しなさい。クラス名は、Tripletとしなさい。

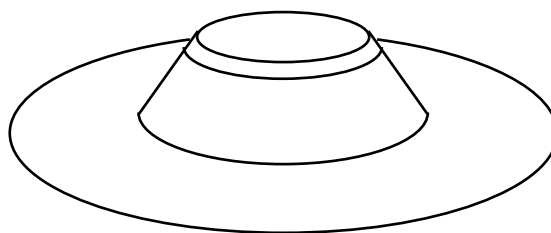


問題の図

5-3.

弧を描くメソッドと、直線を描くメソッド、楕円を描くメソッドを用いて、次のような図を描くようなアプレットを作成しなさい。クラス名は、Pudding（プリン）としなさい。

ヒント：弧の開始角度や、角度差を微妙に調整してみてください。



問題の図

出題の権限を持っている方へ：大変そうと思ったら、止めても構いません。