Java プログラミング入門

— グラフィックの描画：色・グラフ —

早稲田大学

雛形（Hina02.java）

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class Hina02 extends JFrame{
  public Hina02(){
    setSize(500,500);
    setTitle("Java Programing");
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    MyJPanel myJPanel= new MyJPanel();
    Container c = getContentPane();
    c.add(myJPanel);
    setVisible(true);
  }
  public static void main(String[] args){
    new Hina02();
  }
  public class MyJPanel extends JPanel{
    public MyJPanel(){
    }
    public void paintComponent(Graphics g){
      g.setColor(Color.red);
      g.drawLine(100,100,200,200);
      g.setColor(Color.blue);
      g.drawLine(100,200,200,300);
      g.drawLine(100,300,200,400);
    }
  }
}
```

Java での色の設定

色の設定

- Graphics 型を g とし，色を設定するときは

  g.setColor(色);

  として色を設定する
- 色は Color クラスで設定されている
  Color.red，Color.blue, Color.green，Color.white，Color.pink など

Javaでの色の設定（実験）

初期設定

```
public void paintComponent(Graphics g) {

   g.setColor(Color.red);
   g.drawLine(100,100,200,200);

   g.setColor(Color.blue);
   g.drawLine(100,200,200,300);
   g.drawLine(100,300,200,400);

}
```

Java での色の自由な設定

色の設定

- もっと自由に色を設定したいという人のために，三原色を組み合わせて色を設定できる
- 三原色とは RGB，すなわち
  - Red
  - Green
  - Blue

  のことです．これらの強度を 0〜255 までの整数で与えることで色の設定が可能

Javaでの色の自由な設定

色の設定

Graphics型をgとすると

```
g.setColor(new Color(255,255,255)); //白色
g.fillRect(100,100,100,100);

g.setColor(new Color(0,0,0)); //黒色
g.fillRect(200,100,100,100);

g.setColor(new Color(128,128,128)); //灰色
g.fillRect(300,100,100,100);

g.setColor(new Color(255,255,0)); //黄色
g.fillRect(400,100,100,100);
```

例題

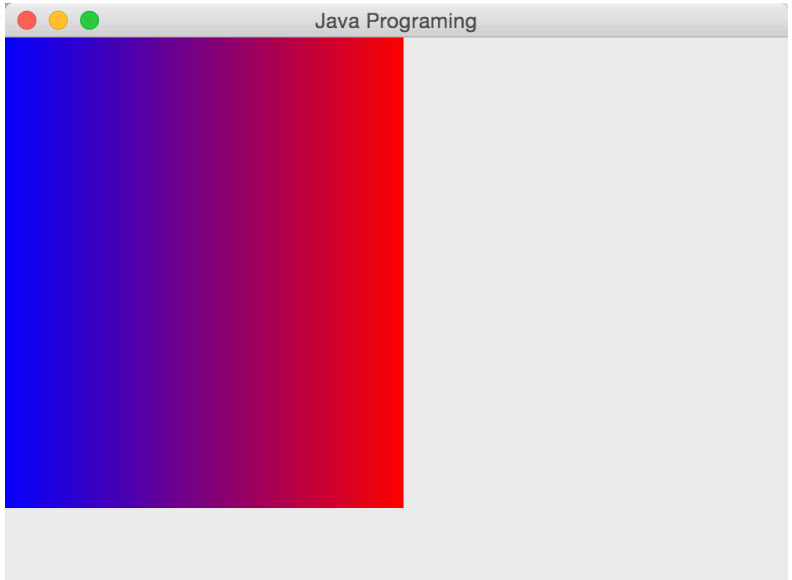
次のプログラムの実行結果を想像せよ

```
public void paintComponent(Graphics g) {
   int i;
   for (i=0;i<255;i++) {
   g.setColor(new Color(i,0,255-i));
   g.drawLine(i,0,i,300);
   }
}
```

例題

次のプログラムの実行結果を想像せよ

```
public void paintComponent(Graphics g) {
   int i;
   for (i=0;i<255;i++) {
   g.setColor(new Color(i,0,255-i));
   g.drawLine(i,0,i,300);
   }
}
```



グラフの描写

関数 $y = f(x)$ のグラフを描く．

折れ線グラフ

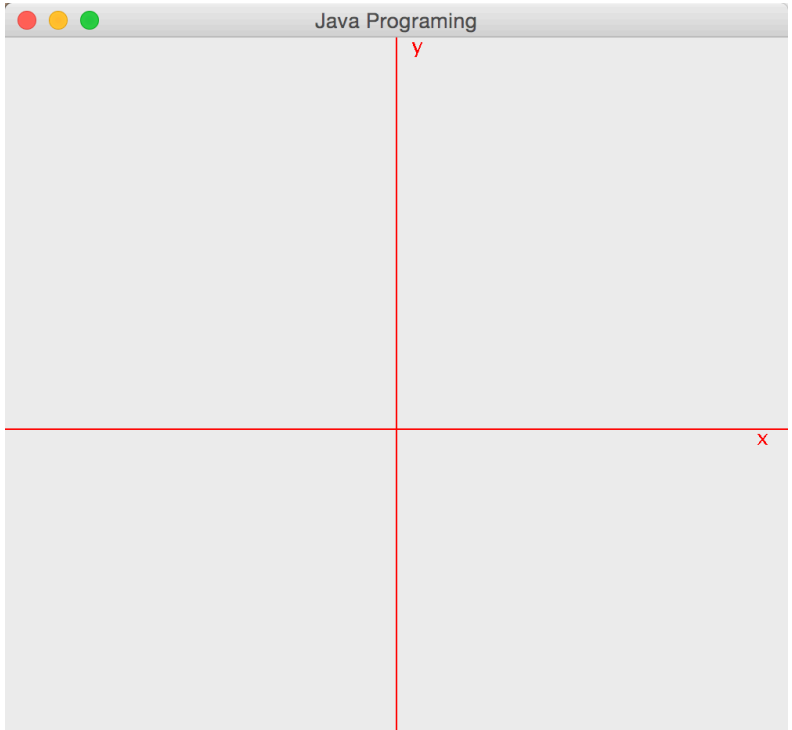
```
public void paintComponent(Graphics g) {
        //ここに作業を行うプログラムを書く
}
```

- 関数を描くとき，曲線を描くようなメソッドは用意されていない
- 細かい折れ線を書いていき，擬似的に曲線を描くことになる

グラフの描写

軸の描写

```
g.setColor(new Color(255, 0, 0));
g.drawLine(0,250,500,250);
g.drawLine(250,0,250,500);
g.drawString("x",480,260);
g.drawString("y",260,10);
```



グラフの描写

グラフを描くために

- 分割数を考え，刻み幅を指定：s
- 関数 $y = f(x)$ を描画するために
  - $(x, f(x))$ から $(x + s, f(x + s))$ まで，
  - $(x + s, f(x + s))$ から $(x + 2s, f(x + 2s))$ まで，
  - $(x + 2s, f(x + 2s))$ から $(x + 3s, f(x + 3s))$ まで，

  を繰り返す．

基本のアルゴリズム

描画プログラムの流れを考えよう！(イメージ)

```
incx=2/10.0;
for (x=-1;x<1;x=x+incx) {
  // x1=x;
  // y1=x1*x1;
  // x2=(x+incx);
  // y2=x2*x2;
  // (x1,y1) から (x2,y2) に線を引く;
}
```

これを Java で記述するためには，座標の変換が必要．

座標の変換

- 数学上の座標 $(-1,1)\times(-1,1)$ とコンピュータの画面 500×500 との対応関係を考える

数学上の座標では

- $(0,0)$ は画面の $(250,250)$ に対応
- $(1,1)$ は画面の $(500,0)$ に対応
- $(-1,-1)$ は画面の $(0,500)$ に対応

- 数学上の座標 $(-1,1)\times(-1,1)$ とコンピュータの画面 500×500 との対応関係を考える
- 数学上の座標 (x,y) は画面の (x',y') に対応するとすると
- $x'=?????$
- $y'=?????$（各自で考える）

描画プログラムのメイン部分

```
g.setColor(Color.black);
incx=2/10.0;
for (x=-1;x<1;x=x+incx) {
  x1=x;
  y1=x*x;
  x2=(x+incx);
  y2=(x+incx)*(x+incx); //座標の変換
  px1=(int)(250*x1+250);
  py1=(int)(250-250*y1);
  px2=(int)(250*x2+250);
  py2=(int)(250-250*y2);
  g.drawLine(px1,py1,px2,py2);
}
```

考察

プログラムの課題点

- 関数に対する一般性の無さ
- 使用できる座標の範囲が固定されている
- 計算の無駄が多い（各自で考えてみよう）

一般性を持たせるためには

- もっと大きな画面にグラフを描かせたいとすると，画面のサイズを変更する必要あり
- その際，全ての座標を変えなければならない
- これは面倒であり，プログラムではなるべく固定の数値を使用するべきではない

考察

プログラムの課題点

- 関数に対する一般性の無さ
- 使用できる座標の範囲が固定されている
- 計算の無駄が多い（各自で考えてみよう）

一般性を持たせるためには

- もっと大きな画面にグラフを描かせたいとすると，画面のサイズを変更する必要あり
- その際，全ての座標を変えなければならない
- これは面倒であり，プログラムではなるべく固定の数値を使用するべきではない

画面のサイズを取得

Dimension クラス

- Dimension クラスを利用する

```
Dimension d; //宣言
d=getSize(); //画面のサイズを得る
```

- d.width には横の長さが格納されている
- d.height には縦の長さが格納されている

参考：

http://docs.oracle.com/javase/1.5.0/docs/api/java/awt/Dimension.html

取得した画面に軸を描画

```
g.setColor(new Color(255, 0, 0));
g.drawLine(0,d.height/2,d.width,d.height/2);
g.drawLine(d.width/2,0,d.width/2,d.height);
g.drawString("x",d.width-20,d.height/2+10);
g.drawString("y",d.width/2+20,10);
```