Java Programming

— Graphics: Colors · Graphs —

Waseda University

Graphics

Today's topic

- Change colors of graphics
- Draw graphs by looping the drawLine method

Model

- Copy & paste the model on your editor
- Save and run the model as it is (without any editing)
- The program is successfully running if a red line and two blue lines appear on a frame.

Model (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);
    }
  }
}
```

Set colors in Java

Set colors

- For a variable g that is an element of Graphics class, we change the color of graphics by using the setColor method, i.e.,

  g.setColor(Color);

- Color is defined by the Color class in Java, e.g., Color.red， Color.blue, Color.green， Color.white， Color.pink

Set colors in Java

Initial setting

```java
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);

}
```

Set any colors in Java

Set any colors

- Set any colors by using the RGB color model.
- The RGB comes from the initials of the three additive primary colors, i.e.,
  - 🟥 Red
  - 🟩 Green
  - 🟦 Blue

  We can set any colors by adjusting the intensity of these three colors (0〜255).

Set any colors in Java

Set colors by using RGB

For the variable g of Graphics class,

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

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

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

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

Example

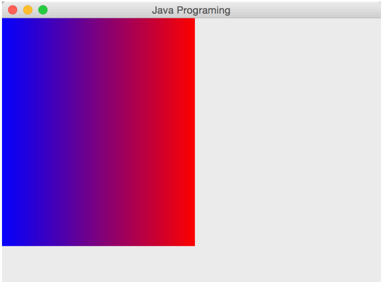
Imagine the result of the following program:

```
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);
  }
}
```

Example

Imagine the result of the following program:

```
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);
  }
}
```



Drawing graphs

The aim is to draw a graph of the function $y = f(x)$.

Line graphs

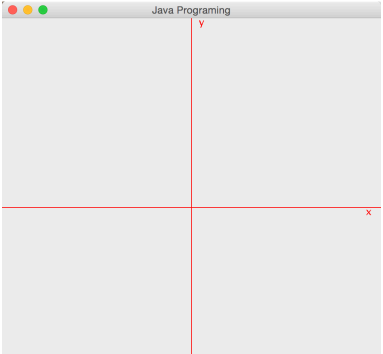
```
public void paintComponent(Graphics g) {
       // write a program here that draw the graph
}
```

- There is no method that draws a curved line in Java.
- Alternatively, we loop drawing small lines that approximate the original curve.

Drawing graphs

Drawing axes

```
g.setColor(new Color(255, 0, 0)); // Set red color
g.drawLine(0,250,500,250); // Draw x axis
g.drawLine(250,0,250,500); // Draw y axis
g.drawString("x",480,260); // Set a label "x"
g.drawString("y",260,10);// Set a label "y"
```



Drawing graphs

To draw a graph (algorithm)

- Set a division number and set a step size s
- To draw a function $y = f(x)$, we loop for
    - drawing a line from $(x, f(x))$ to $(x + s, f(x + s))$,
    - drawing a line from $(x + s, f(x + s))$ to $(x + 2s, f(x + 2s))$,
    - drawing a line from $(x + 2s, f(x + 2s))$ to $(x + 3s, f(x + 3s))$...

Basic algorithm for drawing graphs

Let's consider a flow of the drawing program:

```
s = 2/10.0; // Set a step size
for (x=-1;x<1;x=x+s) {
  // x1=x;
  // y1=x1*x1;
  // x2=(x+s);
  // y2=x2*x2;
  // Draw a line (x1,y1) to (x2,y2);
}
```

To draw the graph on the frame, we need transformation of coordinates, because the above algorithm is based on Cartesian coordinates.

Change coordinates

- We transform Cartesian coordinates in $(-1,1)\times(-1,1)$ into the coordinate system on computers: 500×500

  Cartesian coordinates correspond to...
- $(0,0)$ corresponds to $(250,250)$,
- $(1,1)$ corresponds to $(500,0)$,
- $(-1,-1)$ corresponds to $(0,500)$, respectively.

- Consider the transformation that maps the coordinates in $(-1,1)\times(-1,1)$ into the coordinates on the pixels 500×500.
- (x,y) of Cartesian coordinates corresponds to (x',y') on the screen.
- $x' =?????$
- $y' =?????$ (Describe yourself)

Main part of drawing a graph

```
g.setColor(Color.black);
s = 2/10.0;
for (x=-1;x<1;x=x+s) {
  x1=x;
  y1=x*x;
  x2=(x+s);
  y2=(x+s)*(x+s); // Cartesian coordinates.
  px1=(int)(250*x1+250); // Maps x1 to px1
  py1=(int)(250-250*y1); // Maps y1 to py1
  px2=(int)(250*x2+250);
  py2=(int)(250-250*y2);
  g.drawLine(px1,py1,px2,py2);
}
```

Discussions

Problem of this program

- No generality with respect to the function $f(x)$
- The area of coordinates is fixed.

Room for improvement

- Change the size of the frame if more large graphics are necessary.
- Then we must change all coordinates in the program.
- This wastes your time writing a code. Furthermore, the specified number should not be used in your programming for any language.

Discussions

Problem of this program

- No generality with respect to the function $f(x)$
- The area of coordinates is fixed.

Room for improvement

- Change the size of the frame if more large graphics are necessary.
- Then we must change all coordinates in the program.
- This wastes your time writing a code. Furthermore, the specified number should not be used in your programming for any language.

Improvement: getSize of frame

Dimension class

- Use Dimension class for getting the size of your frame

```
Dimension d; // Declare "d"
d=getSize(); // get the size of tne frame
```

- "d.width" contains the width of the frame.
- "d.height" contains the height of the frame.

For more details, see
http://docs.oracle.com/javase/1.5.0/docs/api/java/awt/Dimension.html

Example

Get the size of the screen and draw axes:

```
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);
```