gki2Line.java

```
package gobalkrishnan_v_18_06_1995.graphics;

import gobalkrishnan_v_18_06_1995.color.gkiColor;

public class gki2Line {
@Override
   public String toString() {
      return  sp + ":" + ep ;
   }

public ArrayList<gkiColor>  color=new ArrayList<>();
public ArrayList<gki2Point> point=new ArrayList<>();

double sx,sy,ex,ey;
gkiColor sc,ec;
gki2Point sp,ep;

public void sx(double sx){this.sx=sx;sp.x=sx;process();}
public void sy(double sy){this.sy=sy;sp.y=sy;process();}
public void ex(double ex){this.ex=ex;ep.x=ex;process();}
public void ey(double ey){this.ey=ey;ep.y=ey;process();}
public void sc(gkiColor sc){this.sc=sc;process();}
public void ec(gkiColor ec){this.ec=ec;process();}
public void sp(gki2Point sp){
   this.sp=sp;
   sx=sp.x;
   sy=sp.y;
   process();
}
public void ep(gki2Point ep){
   this.ep=ep;
  ex=ep.x;
```

```
  ey=ep.y;
  process();
}
public void set(gki2Point sp,gkiColor sc,gki2Point ep,gkiColor ec){

   sp(sp);
   ep(ep);
   sc(sc);
   ec(ec);
}
gkiColor col;
public void process(){
   color.removeAll(color);
   point.removeAll(point);
   double xdiff=ex-sx;
   double ydiff=ey-sy;
   if(xdiff==0 && ydiff==0){
      point.add(new gki2Point(sp));
      color.add(sc);
   }
   if(fabs(xdiff)>fabs(ydiff)){
      double xmin,xmax;
      if(sx<ex){
         xmin=sx;
         xmax=ex;
      }else{
         xmin=ex;
         xmax=sx;
      }
      double slope=ydiff/(double)xdiff;
      for(double x=xmin;x<=xmax;x++){
         double y=(double)(sy+((x-sx)*slope));
```

```
        try{
        double ratio = (x-sx)/(double)xdiff;
        int alpha=(int)(ec.alpha*ratio + sc.alpha*(1-ratio));
        int red=(int)(ec.red*ratio+sc.red*(1-ratio));
        int green=(int)(ec.green*ratio+sc.green*(1-ratio));
        int blue=(int)(ec.blue*ratio+sc.blue*(1-ratio));
        if(alpha<0){alpha=0;}
        if(alpha>255){alpha=255;}
        if(red<0){red=0;}
        if(red>255){red=255;}
        if(green<0){green=0;}
        if(green>255){green=255;}
        if(blue<0){blue=0;}
        if(blue>255){blue=255;}
        col=new gkiColor(alpha, red, green, blue);
        color.add(col);
        }catch(NullPointerException e){

        }
        gki2Point p=new gki2Point(x, y);
        p.color(col);
        point.add(p);
    }
}else{
    double ymin,ymax;
    if(sy<ey){
        ymin=sy;
        ymax=ey;
    }else{
        ymin=ey;
        ymax=sy;
    }
```

```
        double slope=xdiff/(double)ydiff;
        for(double y=ymin;y<=ymax;y++){
           double x=sx+((y-sy)*slope);
           try{
           double ratio=(y-sy)/(double)ydiff;
           int alpha= (int) (ec.alpha*ratio+sc.alpha*(1-ratio));
           int red=(int)(ec.red*ratio+sc.red*(1-ratio));
           int green=(int)(ec.green*ratio+sc.green*(1-ratio));
           int blue=(int)(ec.blue*ratio+sc.blue*(1-ratio));
           if(alpha<0){alpha=0;}
           if(alpha>255){alpha=255;}
           if(red<0){red=0;}
           if(red>255){red=255;}
           if(green<0){green=0;}
           if(green>255){green=255;}
           if(blue<0){blue=0;}
           if(blue>255){blue=255;}
           col=new gkiColor(alpha, red, green, blue);
           color.add(col);
           }catch(NullPointerException e){

           }

           gki2Point p=new gki2Point(x, y);
           p.color(col);

           point.add(p);
        }
     }
}
public void reverse(){
   gki2Point[] li=new gki2Point[point.size()];
```

```
    int a=0;
    for(int i=point.size()-1;i>=0;i--){
        li[a++]=point.get(i);
    }
    point.removeAll(point);
    for(int i=0;i<li.length;i++){
        point.add(li[i]);
    }

}
private double fabs(double g){
    if(g<0){
        g*=-1;
    }
    return g;
}


}
```