Android 笔记

简介

名称：Android 笔记
作者：zhaoqp

简介：Android 完全学习笔记
资料的来源包括书籍，网络，力求准确，全面，实用，经过我的部分修改和调整，一定会对你有帮助，由于个人能力有限，如有问题敬请批评，欢迎大家一起纠正错误，谢谢。

支持论坛：http：//immortal.5d6d.com （Android 学习园地，打造资料最全面的学习社区）
欢迎大家共同学习，新的信息和大量的资料请进**论坛下载**。
QQ：396196516

章节会陆续更新。。。。。
其他章节正在修正中，敬请关注。
大家如果有好的资料可以联系我，一起把这本笔记做好，谢谢。

第七章 计算几何学

这一章将介绍下开发中常用的几何公式，公式都是大家学过的，只是写成了方法。
大家如果喜欢也可以自己实现一些复杂的放到论坛上与大家分享。

点在直线上

直线方程两点式：$\frac{y-y_1}{y_2-y_1}=\frac{x-x_1}{x_2-x_1}$（$P_1(x_1,y_1)$、$P_2(x_2,y_2)$ $x_1\neq x_2$， $y_1\neq y_2$）.

```
点 A（x，y）,B(x1,y1),C(x2,y2)    点 A 在直线 BC 上吗?
public  boolean pointAtSLine
          (double x,double y,double x1,double y1,double x2,double y2){
       int result = ( x - x1 ) * ( y2 - y1 ) - ( y - y1 ) * ( x2 - x1 );
        if(result==0){
           //System.out.println("点在直线上");
           return true;
        }else{
           //System.out.println("点不在直线上");
           return false;
        }
    }
```

点在线段上

```
点 A（x，y）,B(x1,y1),C(x2,y2)    点 A 在线段 BC 上吗?
public static boolean pointAtELine
     (double x,double y,double x1,double y1,double x2,double y2){
       int result = ( x - x1 ) * ( y2 - y1 ) - ( y - y1 ) * ( x2 - x1 );
        if(result==0){
           //System.out.println("点（"+x+","+y+"）在直线上");
           if(x >= Math.min(x1, x2) && x <= Math.max(x1,x2)
               && y >= Math.min(y1, y2) && y <= Math.max(y1,y2)){
               //System.out.println("点（"+x+","+y+"）在线段上");
                return true;
           }else{
               System.out.println("点（"+x+","+y+"）不在线段上");
               return false;
           }
        }else{
           System.out.println("点（"+x+","+y+"）在不在直线上");
           return false;
        }
    }
```

两条直线相交

斜率公式：$k=\frac{y_2-y_1}{x_2-x_1}$，其中 $P_1(x_1,y_1)$、$P_2(x_2,y_2)$.

点 A（x1，y1）,B(x2,y2),C(x3,y3),D(x4,y4)　直线 AB 与直线 CD 相交吗?

```
public  boolean LineAtLine(double x1,double y1,double x2,
                double y2,double x3,double y3,double x4,double y4){
        double k1 = ( y2-y1 )/(x2-x1);
        double k2 = ( y4-y3 )/(x4-x3);
        if(k1==k2){
            //System.out.println("平行线");
            return false;
        }else{
          double x = ((x1*y2-y1*x2)*(x3-x4)-(x3*y4-y3*x4)*(x1-x2))
                        /((y2-y1)*(x3-x4)-(y4-y3)*(x1-x2));
          double y = ( x1*y2-y1*x2 - x*(y2-y1) ) / (x1-x2);
          //System.out.println("直线的交点("+x+","+y+")");
          return true;
        }
    }
```

两条线段相交

点 A（x1，y1）,B(x2,y2),C(x3,y3),D(x4,y4)　线段 AB 与线段 CD 相交吗?

```
public boolean ELineAtELine(double x1,double y1,double x2,
                    double y2,double x3,double y3,double x4,double y4){
        double k1 = ( y2-y1 )/(x2-x1);
        double k2 = ( y4-y3 )/(x4-x3);
        if(k1==k2){
            System.out.println("平行线");
            return false;
        }else{
          double x = ((x1*y2-y1*x2)*(x3-x4)-(x3*y4-y3*x4)*(x1-x2))
                        /((y2-y1)*(x3-x4)-(y4-y3)*(x1-x2));
          double y = ( x1*y2-y1*x2 - x*(y2-y1) ) / (x1-x2);
          System.out.println("直线的交点("+x+","+y+")");
          if(x >= Math.min(x1, x2) && x <= Math.max(x1,x2)
                  && y >= Math.min(y1, y2) && y <= Math.max(y1,y2)
                  && x >= Math.min(x3, x4) && x <= Math.max(x3,x4)
                  && y >= Math.min(y3, y4) && y <= Math.max(y3,y4) ){
                //System.out.println("交点（"+x+","+y+"）在线段上");
```

```
                return true;
            }else{
                System.out.println("交点（"+x+","+y+"）不在线段上");
                return false;
            }
        }
}
```

线段直线相交

```
点A（x1，y1），B(x2,y2),C(x3,y3),D(x4,y4)    线段AB与直线CD相交吗?
public boolean ELineAtELine(double x1,double y1,double x2,
                    double y2,double x3,double y3,double x4,double y4){
        double k1 = ( y2-y1 )/(x2-x1);
        double k2 = ( y4-y3 )/(x4-x3);
        if(k1==k2){
            System.out.println("平行线");
            return false;
        }else{
          double x = ((x1*y2-y1*x2)*(x3-x4)-(x3*y4-y3*x4)*(x1-x2))
                        /((y2-y1)*(x3-x4)-(y4-y3)*(x1-x2));
          double y = ( x1*y2-y1*x2 - x*(y2-y1) ) / (x1-x2);
          System.out.println("交点("+x+","+y+")");
          if(x >= Math.min(x1, x2) && x <= Math.max(x1,x2)
                  && y >= Math.min(y1, y2) && y <= Math.max(y1,y2)){
                //System.out.println("交点（"+x+","+y+"）在线段上");
                return true;
           }else{
                System.out.println("交点（"+x+","+y+"）不在线段上");
                return false;
          }
        }
}
```

矩形包含点

矩形的边都是与坐标系平行或垂直的。
只要判断该点的横坐标和纵坐标是否夹在矩形的左右边和上下边之间。

```
点A（x，y），B(x1,y1),C(x2,y2)    点A在以直线BC为对角线的矩形中吗?
public boolean pointAtRect(
         double x,double y,double x1,double y1,double x2,double y2){
         if(x >= Math.min(x1, x2) && x <= Math.max(x1,x2)
                 && y >= Math.min(y1, y2) && y <= Math.max(y1,y2)){
```

```
            //System.out.println("点（"+x+","+y+"）在矩形内上");
            return true;
        }else{
          //System.out.println("点（"+x+","+y+"）不在矩形内上");
        return false;
        }
    }
```

线段，折线，多边形在矩形中

因为矩形是个凸集，所以只要判断所有端点是否都在矩形中就可以了。

矩形在矩形中

只要对角线的两点都在另一个矩形中就可以了。
点 A(x1,y1),B(x2,y2)，C(x1,y1),D(x2,y2) 以直线 AB 为对角线的矩形在以直线 BC 为对角线的矩形中吗?

```
public boolean RectAtRect(double x1,double y1,double x2,double y2,
                    double x3,double y3,double x4,double y4){
        if(x1 >= Math.min(x3, x4) && x1 <= Math.max(x3,x4)
                && y1 >= Math.min(y3, y4) && y1 <= Math.max(y3,y4)
                && x2 >= Math.min(x3, x4) && x2 <= Math.max(x3,x4)
                && y2 >= Math.min(y3, y4) && y2 <= Math.max(y3,y4)){
            //System.out.println("矩形在矩形内");
            return true;
        }else{
            //System.out.println("矩形不在矩形内");
            return false;
        }
    }
```

圆在矩形中

圆心在矩形中且圆的半径小于等于圆心到矩形四边的距离的最小值。
圆心(x,y) 半径 r 矩形对角点 A（x1，y1），B(x2，y2)

```
public boolean circleAtRect(double x,double y,double r,double x1,
                    double y1,double x2,double y2){
        //圆心在矩形内
        if(x >= Math.min(x1, x2) && x <= Math.max(x1,x2)
                && y >= Math.min(y1, y2) && y <= Math.max(y1,y2)){
        //圆心到4条边的距离
```

```
    double l1= Math.abs(x-x1);
    double l2= Math.abs(y-y2);
    double l3= Math.abs(x-x2);
    double l4= Math.abs(y-y2);
    if(r<=l1 && r<=l2 && r<=l3 && r<=l4){
          //System.out.println("圆在矩形内");
          return true;
      }else{
          //System.out.println("圆不在矩形内");
          return false;
      }

      }else{
         //System.out.println("圆不在矩形内");
        return false;
      }
}
```