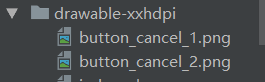
1. 实现步骤：

1、在res文件

### 1.1、在drawable-xxhdpi添加图片



### 1.2、在draw.xml定义ImageView控件



代码：

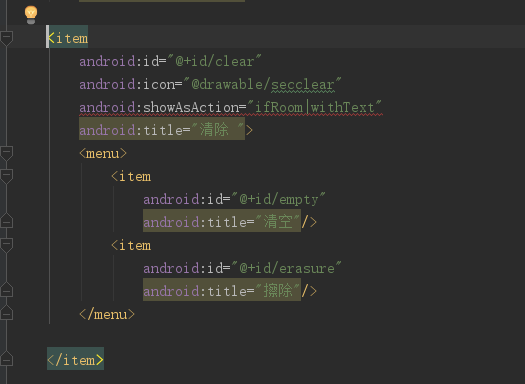
<ImageView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/erasure\_cancel"  
 android:layout\_centerHorizontal="true"  
 android:layout\_alignParentBottom="true"/>

### 1.3、在option\_menu.xml中修改清除功能菜单代码

将原：



更改为：



android:showAsAction="ifRoom|withText"这个属性，ifRoom的意思就是说只要ActionBar上有空间，就把该Item显示出来，否则就坠在后面。

代码：

<item  
 android:id="@+id/clear"  
 android:icon="@drawable/secclear"  
 android:showAsAction="ifRoom|withText"  
 android:title="清除 ">  
 <menu>  
 <item  
 android:id="@+id/empty"  
 android:title="清空"/>  
 <item  
 android:id="@+id/erasure"  
 android:title="擦除"/>  
 </menu>  
  
</item>

2、在DrawView()文件

### 2.1、定义全局变量



### 2.2、在initDraw()函数中初始话橡皮属性



代码：

mEraser\_Paint = new Paint();  
mEraser\_Paint.setAlpha(0); //设置透明度  
  
mEraser\_Paint.setAntiAlias(true);  
mEraser\_Paint.setDither(true);  
mEraser\_Paint.setStyle(Paint.Style.*STROKE*);  
mEraser\_Paint.setStrokeJoin(Paint.Join.*ROUND*); //用圆形画笔  
mEraser\_Paint.setStrokeWidth(30);  
  
Eraser\_path = new Path();

### 2.3、新建clear()功能函数

代码：

public void clear(float x,float y,int Action) { //UI擦除实现  
  
 switch (Action) {  
 case MotionEvent.*ACTION\_DOWN*:  
 Eraser\_path.reset();  
 Eraser\_path.moveTo(x, y); //不会进行绘制，只用于移动移动画笔。  
 canvas.drawPath(Eraser\_path,mEraser\_Paint); //绘制路径  
 preX = x;  
 preY = y;  
 invalidate(); //数据刷新  
 break;  
 case MotionEvent.*ACTION\_MOVE*:  
 float dx = Math.*abs*(x - preX);  
 float dy = Math.*abs*(y - preY);  
 if (dx >= 4 || dy >= 4) { // 判断是否在允许的范围内  
 Eraser\_path.quadTo(preX, preY, (x + preX) / 2, (y + preY) / 2); //用于绘制圆滑曲线，即贝塞尔曲线。  
 preX = x;  
 preY = y;  
 canvas.drawPath(Eraser\_path,mEraser\_Paint);  
 }  
 invalidate();  
 break;  
 case MotionEvent.*ACTION\_UP*:  
 Eraser\_path.lineTo(preX,preY); //用于进行直线绘制  
 canvas.drawPath(Eraser\_path,mEraser\_Paint);  
 invalidate();  
 break;  
 }  
  
  
}

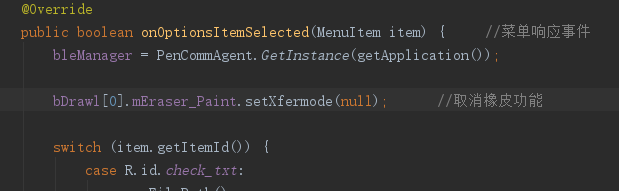
## 3、在OidActivity()文件

### 3.1、定义全局变量

代码：

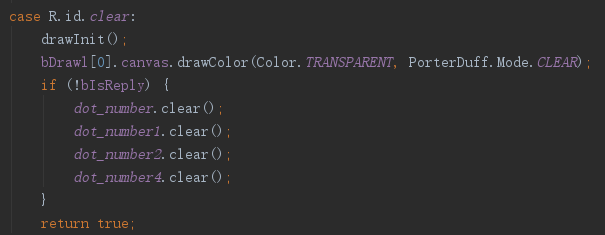
public boolean isErasure = false; //擦除功能开启  
private List<Float> erasure\_x = new ArrayList<Float>(); //擦除笔迹x坐标  
private List<Float> erasure\_y = new ArrayList<Float>(); //擦除笔迹y坐标  
private ImageView erasure\_iv; //擦除关闭按钮控件

### 3.2、修改onOptionsItemSelected()菜单响应事件



代码：

bDrawl[0].mEraser\_Paint.setXfermode(null); //取消橡皮功能

将

替换为



代码：

case R.id.*empty*:  
  
 show\_Normal\_Dialog(1);  
 return true;  
case R.id.*erasure*: //擦除  
 drawInit();  
 bDrawl[0].mEraser\_Paint.setXfermode(new PorterDuffXfermode(PorterDuff.Mode.*DST\_IN*)); //设置橡皮  
 erasure\_x.clear();  
 erasure\_y.clear();  
 isErasure = true;  
 erasure\_iv.setImageDrawable(getResources().getDrawable(R.drawable.*button\_cancel\_1*)); //设置删除按钮图片  
 return true;

### 3.3、在onTouchEvent()中添加触发擦除功能

代码：

if (isErasure){  
// 在SelectDeviceActivity中进行了设置影响了View大小,y坐标需进行相反处理  
// SelectDeviceActivity进行删除后，仍然存在，原因暂时未知

int actionBarHeight = getActionBar().getHeight();  
 int statusHeight = getStatusBarHeight();  
  
 float x = event.getX();  
 float y = event.getY() - (actionBarHeight + statusHeight + 10);  
  
 erasure\_x.add(x);  
 erasure\_y.add(y);  
  
 bDrawl[0].clear(x,y,event.getAction()); //UI擦除效果实现  
 }

### 3.4、在onCreate()注册擦除取消点击监听

代码：

erasure\_iv = (ImageView)findViewById(R.id.*erasure\_cancel*);  
erasure\_iv.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 erasure\_iv.setImageDrawable(getResources().getDrawable(R.drawable.*button\_cancel\_2*));  
 bDrawl[0].mEraser\_Paint.setXfermode(null);  
 new Thread(){  
 public void run(){  
 SystemClock.*sleep*(300);  
 runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 erasure\_iv.setImageDrawable(null);  
// Log.i("ceshi", "Erasure\_dots\_go" );

show\_Normal\_Dialog(2);  
  
 isErasure = false;  
 }  
 });  
 }  
 }.start();  
 }  
});

### 3.5、添加show\_Normal\_Dialog()对话框确认函数



代码：

private void show\_Normal\_Dialog(int type)  
{  
 final AlertDialog.Builder normal\_dialog = new AlertDialog.Builder(OidActivity.this);  
 if (type == 1) {  
 normal\_dialog.setMessage("删除本页所有内容");  
 normal\_dialog.setPositiveButton("删除",  
 new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialogInterface, int i) {  
 drawInit();  
  
 bDrawl[0].canvas.drawColor(Color.*TRANSPARENT*, PorterDuff.Mode.*CLEAR*); //铺透明幕布  
 if (!bIsReply) {  
 *dot\_number*.removeAll(gCurPageID); //删除页内点数据  
 *dot\_number1*.removeAll(gCurPageID);  
 *dot\_number2*.removeAll(gCurPageID);  
 *dot\_number4*.removeAll(gCurPageID);  
  
 }  
 }  
 });  
 normal\_dialog.setNegativeButton("取消",  
 new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialogInterface, int i) {

}  
 });  
  
 }  
  
 else if (type == 2)  
 {  
 normal\_dialog.setMessage("保存擦除修改内容");  
 normal\_dialog.setPositiveButton("保存",  
 new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialogInterface, int i) {  
 Erasure\_dots(gCurBookID,gCurPageID);   
 drawInit();  
 bDrawl[0].canvas.drawColor(Color.*TRANSPARENT*, PorterDuff.Mode.*CLEAR*);  
 DrawExistingStroke(gCurBookID, gCurPageID);  
 }  
 });  
 normal\_dialog.setNegativeButton("取消",  
 new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialogInterface, int i) {  
 drawInit();  
 bDrawl[0].canvas.drawColor(Color.*TRANSPARENT*, PorterDuff.Mode.*CLEAR*);  
 DrawExistingStroke(gCurBookID, gCurPageID);  
 }  
 });

}  
 normal\_dialog.show();  
  
}

### 3.6、添加Erasure\_dots()点清除函数

代码：

public void Erasure\_dots(int BookID, int PageID) { //点清除  
// Log.i("ceshi", "Erasure\_dots\_in" );  
 if (BookID == 100) {  
 *dot\_number4* = *dot\_number*;  
 } else if (BookID == 0) {  
 *dot\_number4* = *dot\_number1*;  
 } else if (BookID == 1) {  
 *dot\_number4* = *dot\_number2*;  
 }  
  
 if (*dot\_number4*.isEmpty()) {  
 return;  
 }  
  
  
 int erasure\_dots\_size = erasure\_x.size();  
  
 Set<Integer> keys = *dot\_number4*.keySet();  
  
 ArrayListMultimap<Integer, Dots> dot\_number5 = ArrayListMultimap.*create*();  
 List<Integer> erasure\_matching\_id = new ArrayList<Integer>(); //存储橡皮和笔迹坐标匹配的id  
  
 List<Dots> dots = *dot\_number4*.get(PageID);  
 int length=0;  
// Log.i("ceshi", "Erasure\_dots\_ok" );  
 float distance\_x;  
 float distance\_y;  
 float distance;  
 boolean isMatching=false;  
 for (int key : keys) {  
 if (key == PageID) {  
  
// dot\_number4.removeAll(gCurPageID);  
 length = dots.size();  
 for (int i=0;i<erasure\_dots\_size;i++){  
// Log.i("ceshi", "erasure——zuobiao:"+erasure\_x.get(i)+"，"+ erasure\_y.get(i));  
 for (int j=0;j<length;j++){  
 distance\_x = dots.get(j).pointX - erasure\_x.get(i);  
 distance\_y = dots.get(j).pointY - erasure\_y.get(i);  
 if (distance\_x<0)  
 distance\_x=-distance\_x;  
 if (distance\_y<0)  
 distance\_y=-distance\_y;  
// Log.i("ceshi", "distance\_zuobiao:"+distance\_x+"，"+distance\_y);

//进行点擦除时，要修改所匹配在线段的起、止点的状态  
 if (distance\_x<=15 && distance\_y<=15 && !erasure\_matching\_id.contains(j)) //橡皮的像素半径为15  
 {  
  
 distance = distance\_x\*distance\_x + distance\_y\*distance\_y;  
 if (distance <= 225) {  
// Log.i("ceshi", "Erasure\_dots\_true-1" );  
 erasure\_matching\_id.add(j);  
// Log.i("ceshi", "Erasure\_dots\_true-2" );  
 if (!isMatching) {  
  
 if ((j-1) >= 0) {  
 dots.get(j-1).ntype=2;  
// Log.i("ceshi", "START:"+dots.get(j-1).ntype);  
 }  
 isMatching=true;  
  
 }  
 }  
  
 else if (isMatching && dots.get(j).ntype!=2) {  
 dots.get(j).ntype=0;  
 isMatching=false;  
  
 }  
  
 }  
  
 else if (isMatching && dots.get(j).ntype!=2 && !erasure\_matching\_id.contains(j)) {  
 dots.get(j).ntype=0;  
 isMatching=false;  
  
 }  
 }  
 }  
  
 }  
 else{  
  
 dot\_number5.putAll(key,*dot\_number4*.get(key)); //将第key页的点数据全部存入dot\_number5  
 }  
 }  
  
 for (int i=0;i<length;i++){  
  
 if (!erasure\_matching\_id.contains(i)) {  
 dot\_number5.put(PageID,dots.get(i)); //将当前页内没有匹配点数据存入dot\_number5  
 }  
  
 }  
  
// dot\_number.removeAll(PageID);  
 if (BookID == 100) {  
 *dot\_number* = dot\_number5;  
 } else if (BookID == 0) {  
 *dot\_number1* = dot\_number5;  
 } else if (BookID == 1) {  
 *dot\_number2* = dot\_number5;  
 }  
 }

### 3.7、修改ProcessDots()中的条件

代码：

// 回放和擦除模式，不接受点  
if (bIsReply&&(!isErasure)) {  
 return;  
}

二、参考资料：

UI擦除实现：

https://blog.csdn.net/FDGFGFDGFD/article/details/80413588

三、改进建议：

擦除功能在单向直线滑动时，有概率出现失灵。擦除点坐标和笔迹点坐标在x轴坐标的相对距离逼近最小值时，y轴坐标的相对距离会出现50~100的误差，导致擦除失败。猜测需要在api层或硬件驱动层着手解决。