

# Hongyang

生命不息，奋斗不止，万事起于忽微，量变引起质变

目录视图

个人资料



鸿洋\_

关注

发私信

访问：9689028次

积分：40449

等级：BLOG > 8

排名：第64名

原创：188篇

转载：0篇

译文：6篇

评论：11485条

我的微信公众号

点击直达推送文章汇总

长期为您推荐优秀博文、开源项目、视频等，进入还有好玩的等着你，欢迎扫一扫。



联系方式

新动态



给我写信

QQ群：

429757068

264950424

463081660

请勿重复加群，Thx

文章分类

【Android 5.x】 (11)

【公告】博客专栏旧貌换新颜

【hot】直播技术精选

主流编程语言图谱之二

## Android FoldingLayout 折叠布局 原理及实现（一）

标签：Android FoldingLayout Shader

2015-03-16 09:26 23535人阅读 评论

分类：【Android 自定义控件实战】（28） 【Android 精彩案例】（36）

版权声明：本文为博主原创文章，未经博主允许不得转载。

目录(?)

[+]

转载请标明出处：<http://blog.csdn.net/Imj623565791/article/details/44278417>，本文出自：【张鸿洋的博客】

### 1、概述

无意中翻到的FoldingLayout的介绍的[博客](#)，以及[github地址](#)。感觉很nice呀，于是花了点时间研究以及编写，本篇博文将对其进行原理分析，一步一步的实现我们的FoldingLayout，当然了，如果你能力过硬，可以直接下载github上的代码进行学习。

博客基本分为以下几个部分：

- 1、Matrix的setPolyToPoly使用
- 2、在图片上使用渐变和阴影
- 3、初步的FoldingLayout的实现，完成图片的折叠显示（可控制折叠次数、包含阴影的绘制）
- 4、引入手势，手指可以可以FoldingLayout的折叠
- 5、结合DrawerLayout实现折叠式侧滑
- 6、结合SlidingPanelLayout实现折叠式侧滑

ok，贴下部分的效果图：

- 【Android 精彩案例】 (37)
- 【Android 源码解析】 (29)
- 【Android 自定义控件实战】 (29)
- 【Android 自定义控件之起步】 (7)
- 【Android 快速开发】 (12)
- 【Android 原生开发游戏】 (3)
- 【Java 并发专题】 (15)
- 【android 进阶之路】 (65)
- 【Java 设计模式】 (10)
- 【Android 百度地图】 (4)
- 【html5 css3精彩案例】 (14)
- 【Android github 控件】 (10)
- 【Android 基础】 (16)
- 【Javascript】 (9)
- 【rabbitMQ 用法】 (5)
- 【Android微知识点】 (4)

友情链接

- 郭霖的博客
- 夏安明的博客
- 任玉刚的博客
- 元斌的博客
- 敬佩的孔老师
- foruok的订阅号程序视界
- OpenCV大神shiter
- 专为Android程序员的导航
- 泡在网上的日子

博客专栏

-  HTML5 & CSS3 实战  
文章：11篇  
阅读：135335
-  设计模式融入生活  
文章：10篇  
阅读：82523
-  Android 精彩案例  
文章：67篇  
阅读：4088814

阅读排行

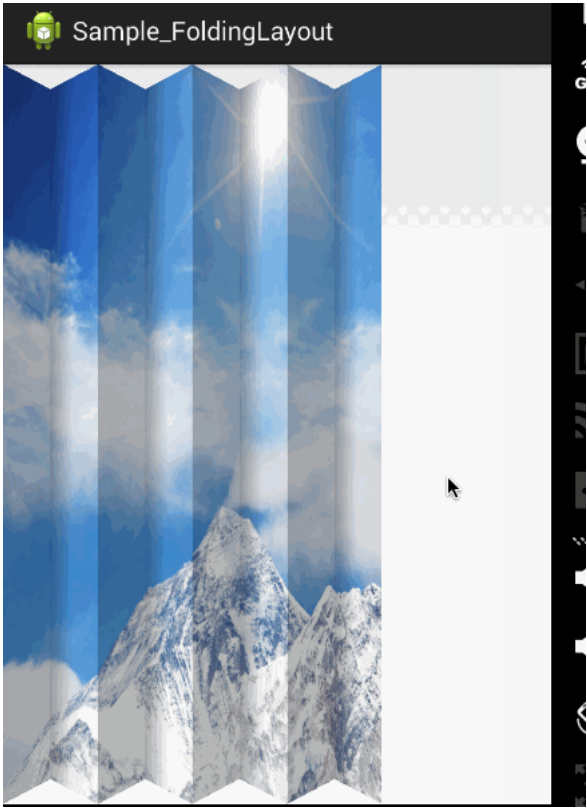
- Android Https相关完全解析 ... (1524068)
- Android Fragment 真正的完... (473032)
- Android RecyclerView 使用完... (353564)
- Android OkHttp完全解析 是... (268891)
- Android 自定义View (一) (197237)
- Android 属性动画 (Property... (190922)
- Android Fragment 真正的完... (184061)
- Android 屏幕适配方案 (141709)
- Android 手把手教您自定义Vi... (140712)
- Android 自定义RecyclerView... (125035)

最新评论

- Android 属性动画 (Property Animation...  
Super-B :好东西



改图对应上述3，妹子不错吧~

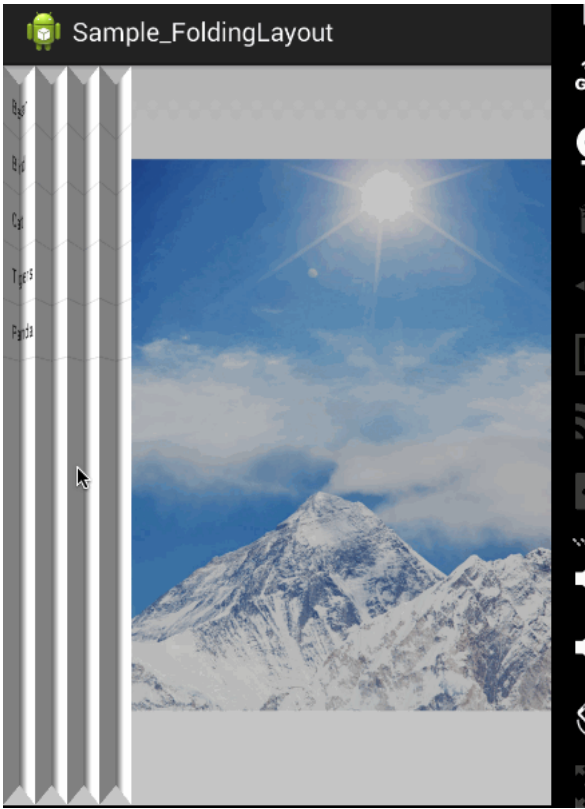


ok，对应上述4.

- FloatingActionButton 完全解析[Design ...  
Scott163425 : 大神,我学习你的博文很久了,收益很多。现在我遇到一个小问题:CollapsingToolbarLay...
- Android 屏幕适配方案  
nuttertools798 : @b87936260:请问value s-sw000dp适配方案是怎样的呢?哪里可以找到资料?
- Android 自定义View (二) 进阶  
Ruffian-痞子 : 楼主这个场景的开发中经常用到。但是我们一般使用textview的drawableViewLeft即可实现可惜...
- 如何利用github打造博客专属域名  
sunshine2050\_csdn : 为啥采用这种方法后,打开网站很慢
- Android NestedScrolling机制完全解析 ...  
im哆来咪 : @IT\_peng:嗯 判断一下第一次上移的时候增加父容器高度。
- Android 异步消息处理机制 让你深入理解...  
坏蛋lk : @liqianwei1230:可以参考慕课网的nate老师讲的http://www.imooc.co...
- Android 自定义View (三) 圆环交替 等待...  
demon\_die : @u013007459:大神发一个demo学习一下,谢谢!292374728@qq.com
- Android NestedScrolling机制完全解析 ...  
niaonao : 顶
- Android NestedScrolling机制完全解析 ...  
IT\_peng : @duolaimila:你的意思是动态增加父容器高度?

2015博客之星评选

统计



对应上述5。

ok,挑选了部分图,不然太占篇幅了。

那么接下来,我们就按照顺序往下学习了~~~

## 2、Matrix的setPolyToPoly使用

想要实现折叠,最重要的就是其核心的原理了,那么第一步我们要了解的就是,如何能把一张正常显示的图片,让它能其实精髓就在于Matrix的setPolyToPoly的方法。

```
[java]
01. public boolean setPolyToPoly(float[] src, int srcIndex, float[] dst, int dstIndex,int pointCount)
```

简单看一下该方法的参数,src代表变换前的坐标;dst代表变换后的坐标;从src到dst的变换,可以通过srcIndex和dst的,一般可能都设置位0。pointCount代表支持的转换坐标的点数,最多支持4个。

如果不明白没事,下面通过一个简单的例子,带大家了解:

```
[java]
01. package com.zhy.sample.folderlayout;
02.
03. import android.app.Activity;
04. import android.content.Context;
05. import android.graphics.Bitmap;
06. import android.graphics.BitmapFactory;
07. import android.graphics.Canvas;
08. import android.graphics.Matrix;
09. import android.os.Bundle;
10. import android.view.View;
11.
12. public class MatrixPolyToPolyActivity extends Activity
13. {
14.
15.     @Override
16.     protected void onCreate(Bundle savedInstanceState)
17.     {
18.         super.onCreate(savedInstanceState);
19.         setContentView(new PolyToPolyView(this));
20.     }
21.
22.     class PolyToPolyView extends View
23.     {
24.
25.         private Bitmap mBitmap;
26.         private Matrix mMatrix;
27.
28.         public PolyToPolyView(Context context)
```

```

29.         {
30.             super(context);
31.             mBitmap = BitmapFactory.decodeResource(getResources(),
32.                 R.drawable.tanyan);
33.             mMatrix = new Matrix();
34.             float[] src = { 0, 0, //
35.                 mBitmap.getWidth(), 0, //
36.                 mBitmap.getWidth(), mBitmap.getHeight(), //
37.                 0, mBitmap.getHeight() };
38.             float[] dst = { 0, 0, //
39.                 mBitmap.getWidth(), 100, //
40.                 mBitmap.getWidth(), mBitmap.getHeight() - 100, //
41.                 0, mBitmap.getHeight() };
42.             mMatrix.setPolyToPoly(src, 0, dst, 0, src.length >> 1);
43.         }
44.
45.         @Override
46.         protected void onDraw(Canvas canvas)
47.         {
48.             super.onDraw(canvas);
49.             canvas.drawBitmap(mBitmap, mMatrix, null);
50.         }
51.     }
52. }
53.
54. }

```

我们编写了一个PolyToPolyView作为我们的Activity的主视图。

在PolyToPolyView中，我们加载了一张图片，初始化我们的Matrix，注意src和dst两个数组，src就是正常情况下图片的四个点的坐标，dst就是倾斜后图片的四个点的坐标，我们只修改了右侧两个点的y坐标做了些许的修改。

大家可以在纸上稍微标一下src和dst的四个点的位置。

最后我们在onDraw的时候进行图像的绘制，效果为：



如果你已经在纸上稍微的画了dst的四个点，那么这个结果你一定不陌生。

可以看到我们通过matrix.setPolyToPoly实现了图片的倾斜，那么引入到折叠的情况，假设折叠两次，大家有思路么，继续往下看。

### 3、引入阴影

其实阴影应该在实现初步的折叠以后来说，这样演示其实比较方便，但是为了降低其理解的简单性，我们先把阴影抽取出来。假设我们现在要给上图加上阴影，希望的效果图是这样的：



可以看到我们左侧加入了一点阴影，怎么实现呢？

主要还是利用LinearGradient，我们从左到右添加一层从黑色到透明的渐变即可。

```
[java] C P
01. public class MatrixPolyToPolyWithShadowActivity extends Activity
02. {
03.
04.     @Override
05.     protected void onCreate(Bundle savedInstanceState)
06.     {
07.         super.onCreate(savedInstanceState);
08.         setContentView(new PolyToPolyView(this));
09.
10.     }
11.
12.     class PolyToPolyView extends View
13.     {
14.
15.         private Bitmap mBitmap;
16.         private Matrix mMatrix;
17.
18.         private Paint mShadowPaint;
19.         private Matrix mShadowGradientMatrix;
20.         private LinearGradient mShadowGradientShader;
21.
22.         public PolyToPolyView(Context context)
23.         {
24.             super(context);
25.             mBitmap = BitmapFactory.decodeResource(getResources(),
26.                 R.drawable.tanyan);
27.             mMatrix = new Matrix();
28.
29.             mShadowPaint = new Paint();
30.             mShadowPaint.setStyle(Style.FILL);
31.             mShadowGradientShader = new LinearGradient(0, 0, 0.5f, 0,
32.                 Color.BLACK, Color.TRANSPARENT, TileMode.CLAMP);
33.             mShadowPaint.setShader(mShadowGradientShader);
34.
35.             mShadowGradientMatrix = new Matrix();
36.             mShadowGradientMatrix.setScale(mBitmap.getWidth(), 1);
37.             mShadowGradientShader.setLocalMatrix(mShadowGradientMatrix);
38.             mShadowPaint.setAlpha((int) (0.9*255));
39.
40.         }
41.
42.         @Override
43.         protected void onDraw(Canvas canvas)
44.         {
45.             super.onDraw(canvas);
46.             canvas.save();
47.             float[] src = {...};
48.             float[] dst = {...};
49.             mMatrix.setPolyToPoly(src, 0, dst, 0, src.length >> 1);
50.
51.             canvas.concat(mMatrix);
52.             canvas.drawBitmap(mBitmap, 0, 0, null);
53.             //绘制阴影
```

```

54.         mShadowPaint);
55.         canvas.restore();
56.
57.     }
58.
59. }
60.
61. }

```

重点看mShadowPaint, mShadowGradientShader, mShadowGradientMatrix一个是画笔, 我们为画笔设置了一个Shader的参数为

```
new LinearGradient(0, 0, 0.5f, 0, Color.BLACK, Color.TRANSPARENT, TileMode.CLAMP);
```

起点(0, 0)、终点(0.5f, 0); 颜色从和BLACK到透明; 模式为CLAMP, 也就是拉伸最后一个像素。

这里你可能会问, 这才为0.5个像素的区域设置了渐变, 不对呀, 恩, 是的, 继续看接下来我们使用了

setLocalMatrix(mShadowGradientMatrix);, 而这个

mShadowGradientMatrix将和坐标扩大了mBitmap.getWidth()倍, 也就是说现在设置渐变的区域为(0.5f\*mBitmap的大小, 那么后半张图呢?

后半张应用CLAMP模式, 拉伸的透明。

关于Shader、setLocalMatrix等用法也可以参考: [Android BitmapShader 实战 实现圆形、圆角图片](#)

## 4、初步实现折叠

了解了原理以及阴影的绘制以后, 接下来要开始学习真正的去折叠了, 我们的目标效果为:



妹子折叠成了8份, 且阴影的范围为: 每个沉下去夹缝的左右两侧, 左侧黑色半透明遮盖, 右侧短距离的黑色到透明阴影。现在其实大家以及会将图片简单倾斜和添加阴影了, 那么唯一的难点就是怎么将一张图分成很多块, 我相信每块的折叠。其实我们可以通过绘制该图多次, 比如第一次绘制往下倾斜; 第二次绘制往上倾斜; 这样就和我们标题2的实现类似了, setPolyToPoly。

那么绘制多次, 每次显示肯定不是一整张图, 比如第一次, 我只想显示第一块, 所以我们还需要clipRect的配合, 说到这

```

[java]
01. package com.zhy.sample.folderlayout;
02.
03. import android.app.Activity;
04. import android.content.Context;
05. import android.graphics.Bitmap;
06. import android.graphics.BitmapFactory;
07. import android.graphics.Canvas;
08. import android.graphics.Color;
09. import android.graphics.LinearGradient;
10. import android.graphics.Matrix;
11. import android.graphics.Paint;

```



```
12. import android.graphics.Paint.Style;
13. import android.graphics.Shader.TileMode;
14. import android.os.Bundle;
15. import android.view.View;
16.
17. public class SimpleUseActivity extends Activity
18. {
19.
20.     @Override
21.     protected void onCreate(Bundle savedInstanceState)
22.     {
23.         super.onCreate(savedInstanceState);
24.         setContentView(new PolyToPolyView(this));
25.
26.     }
27.
28.     class PolyToPolyView extends View
29.     {
30.
31.         private static final int NUM_OF_POINT = 8;
32.         /**
33.          * 图片的折叠后的总宽度
34.          */
35.         private int mTranslateDis;
36.
37.         /**
38.          * 折叠后的总宽度与原图宽度的比例
39.          */
40.         private float mFactor = 0.8f;
41.         /**
42.          * 折叠块的个数
43.          */
44.         private int mNumOfFolds = 8;
45.
46.         private Matrix[] mMatrices = new Matrix[mNumOfFolds];
47.
48.         private Bitmap mBitmap;
49.
50.         /**
51.          * 绘制黑色透明区域
52.          */
53.         private Paint mSolidPaint;
54.
55.         /**
56.          * 绘制阴影
57.          */
58.         private Paint mShadowPaint;
59.         private Matrix mShadowGradientMatrix;
60.         private LinearGradient mShadowGradientShader;
61.
62.         /**
63.          * 原图每块的宽度
64.          */
65.         private int mFlodWidth;
66.         /**
67.          * 折叠时, 每块的宽度
68.          */
69.         private int mTranslateDisPerFlod;
70.
71.         public PolyToPolyView(Context context)
72.         {
73.             super(context);
74.             mBitmap = BitmapFactory.decodeResource(getResources(),
75.                 R.drawable.tanyan);
76.
77.             //折叠后的总宽度
78.             mTranslateDis = (int) (mBitmap.getWidth() * mFactor);
79.             //原图每块的宽度
80.             mFlodWidth = mBitmap.getWidth() / mNumOfFolds;
81.             //折叠时, 每块的宽度
82.             mTranslateDisPerFlod = mTranslateDis / mNumOfFolds;
83.
84.             //初始化matrix
85.             for (int i = 0; i < mNumOfFolds; i++)
86.             {
87.                 mMatrices[i] = new Matrix();
88.             }
89.
90.             mSolidPaint = new Paint();
91.             int alpha = (int) (255 * mFactor * 0.8f) ;
92.             mSolidPaint
93.                 .setColor(Color.argb((int) (alpha*0.8F), 0, 0, 0));
94.
95.             mShadowPaint = new Paint();
96.             mShadowPaint.setStyle(Style.FILL);
97.             mShadowGradientShader = new LinearGradient(0, 0, 0.5f, 0,
98.                 Color.BLACK, Color.TRANSPARENT, TileMode.CLAMP);
```



```

309.         mShadowPaint.setShader(mShadowGradientShader);
310.         mShadowGradientMatrix = new Matrix();
311.         mShadowGradientMatrix.setScale(mFlodWidth, 1);
312.         mShadowGradientShader.setLocalMatrix(mShadowGradientMatrix);
313.         mShadowPaint.setAlpha(alpha);
314.
315.         //纵轴减小的那个高度, 用勾股定理计算下
316.         int depth = (int) Math.sqrt(mFlodWidth * mFlodWidth
317.             - mTranslateDisPerFlod * mTranslateDisPerFlod)/2;
318.
319.         //转换点
320.         float[] src = new float[NUM_OF_POINT];
321.         float[] dst = new float[NUM_OF_POINT];
322.
323.         /**
324.          * 原图的每一块, 对应折叠后的每一块, 方向为左上、右上、右下、左下, 大家在纸上自己画下
325.          */
326.         for (int i = 0; i < mNumOfFolds; i++)
327.         {
328.             src[0] = i * mFlodWidth;
329.             src[1] = 0;
330.             src[2] = src[0] + mFlodWidth;
331.             src[3] = 0;
332.             src[4] = src[2];
333.             src[5] = mBitmap.getHeight();
334.             src[6] = src[0];
335.             src[7] = src[5];
336.
337.             boolean isEven = i % 2 == 0;
338.
339.             dst[0] = i * mTranslateDisPerFlod;
340.             dst[1] = isEven ? 0 : depth;
341.             dst[2] = dst[0] + mTranslateDisPerFlod;
342.             dst[3] = isEven ? depth : 0;
343.             dst[4] = dst[2];
344.             dst[5] = isEven ? mBitmap.getHeight() - depth : mBitmap
345.                 .getHeight();
346.             dst[6] = dst[0];
347.             dst[7] = isEven ? mBitmap.getHeight() : mBitmap.getHeight()
348.                 - depth;
349.
350.             //setPolyToPoly
351.             mMatrices[i].setPolyToPoly(src, 0, dst, 0, src.length >> 1);
352.         }
353.     }
354.
355.     @Override
356.     protected void onDraw(Canvas canvas)
357.     {
358.         super.onDraw(canvas);
359.         //绘制mNumOfFolds次
360.         for (int i = 0; i < mNumOfFolds; i++)
361.         {
362.             canvas.save();
363.             //将matrix应用到canvas
364.             canvas.concat(mMatrices[i]);
365.             //控制显示的大小
366.             canvas.clipRect(mFlodWidth * i, 0, mFlodWidth * i + mFlodWidth,
367.                 mBitmap.getHeight());
368.             //绘制图片
369.             canvas.drawBitmap(mBitmap, 0, 0, null);
370.             //移动绘制阴影
371.             canvas.translate(mFlodWidth * i, 0);
372.             if (i % 2 == 0)
373.             {
374.                 //绘制黑色遮盖
375.                 canvas.drawRect(0, 0, mFlodWidth, mBitmap.getHeight(),
376.                     mSolidPaint);
377.             }
378.             else
379.             {
380.                 //绘制阴影
381.                 canvas.drawRect(0, 0, mFlodWidth, mBitmap.getHeight(),
382.                     mShadowPaint);
383.             }
384.             canvas.restore();
385.         }
386.     }
387. }
388.
389. }
390.
391. }

```



简单讲解下，不去管绘制阴影的部分，其实折叠就是：

- 1、初始化转换点，这里注释说的很清楚，大家最好在纸上绘制下，标一下每个变量。
- 2、为matrix.setPolyToPoly
- 3、绘制时使用该matrix，且clipRect控制显示区域（这个区域也很简单，原图的第一块到最后一块），最好就是绘制bi阴影这里大家可以换个明亮点的图片去看看~~

好了，由于篇幅原因，剩下的内容将在下一篇继续完成，下一篇将展示如何将简单的图片的及引入手势、和DrawerLayout等结合应用到侧滑中去。

对于类似这种效果的，一定要拿出稿纸笔去画一画，否则很难弄明白。



源码：[下载](#)  
微信公众号请扫描（第一时间通知博客、视频等通知）：



  
儿童编程

  
软件工程师培训

  
软件工程师待遇

  
程序员

顶

42

踩

2

- [上一篇](#) [Android 自定义控件玩转字体变色 打造炫酷ViewPager指示器](#)
- [下一篇](#) [Android FoldingLayout 折叠布局 原理及实现（二）](#)

我的同类文章

【Android 自定义控件实战】（ 28 ）		【Android 精彩案例】（ 36 ）	
• <a href="#">Android 高清加载巨图方案 拒绝压缩图片</a>	2015-10-21 阅读 50394	• <a href="#">Android TagFlowLayout完全解析 一款...</a>	2015-09-
• <a href="#">ViewDragHelper实战 自己打造Drawer...</a>	2015-08-11 阅读 30739	• <a href="#">Android FoldingLayout 折叠布局 原理...</a>	2015-03-
<a href="#">更多文章</a>			

猜你在找

【Android APP开发】Android高级·· <a href="#">反编译Android应用</a> <a href="#">解析移动应用的身份认证，数据分··</a> <a href="#">Android高级界面控件难点精讲</a> <a href="#">Qt基础与Qt on Android入门</a>	<a href="#">Android FoldingLayout 折叠布局··</a> <a href="#">Android FoldingLayout 折叠布局··</a> <a href="#">Android FoldingLayout 折叠布局··</a> <a href="#">Android FoldingLayout 折叠布局··</a> <a href="#">Android FoldingLayout 折叠布局··</a>
--	--

[查看评论](#)



viclee108  
学习了



诗未冷  
一些方法不知道是做什么用的，能达到什么效果，不熟悉。



大西北的风  
不错，学到了，支持。



单车Luke  
怎么没有mianactivity????运行直接runtime异常



单车Luke  
mMatrix.setPolyToPoly(src, 0, dst, 0, src.length >> 1);  
>>1 是什么意思。？



vivian陈薇  
回复qq\_26642739：相当于除以2



passerby\_whu  
用绘制mNumOffFolds次全图加clipRect的方式感觉可以改为  
public void drawBitmap(Bitmap bitmap, Rect src, Rect dst, Paint paint)，只绘制所需要区域。以免损耗性能。



-琥珀川-  
可否说一下save()和restore作用



GcsSloop  
回复u010032372：快照(save)和回滚(restore)  
画布的操作是不可逆的，而且画布操作会影响后续的步骤，为此需要保存当前画布状态，当前操作完成后在回滚到操作之前的状态，更多可以参考Canvas之画布操作后面部分：[http://www.gcssloop.com/2015/02/Canvas\\_Convert/](http://www.gcssloop.com/2015/02/Canvas_Convert/)



sungerk  
纵向怎么折叠求指教？



xietansheng  
有部分块不能正常显示，结果发现 depth 值手动设置小一点（我设置成小于30）可以全部块正常显示，depth 的值设置超过某一个值，则有部分块不能正常显示图片（阴影可以正常显示）



飞天猪23  
为什么运行无法显示能，折叠部分只能显示一块，也就是第一次Canvans.clipRect(l,t,r,b);就显示 r-l 这一块。为什么啊



梦天2015  
回复u012386007：换不同机型测试，就会发现这没有问题



fangchao3652  
八块 直显示第一块



根号3的故事  
赞，很适合学习~~~



 Qiujuer

学习了~~ 向大神靠近~~

41楼 2016-09-05 10:00

 Jason\_stu mark

40楼 2016-09-05 09:59

 花-开-花-谢

亲，能不更讲讲增量更新啊，感觉挺酷挺实用的

39楼 2016-09-05 09:58

 陨落烟雨

撸的好！

 akira\_xiao

来晚了

 阿\_yang

赞

36楼 2016-09-05 09:54


 huxyz

请问之前的视频教程网址怎么找不到了？

35楼 2016-09-05 09:53


 vane0001

效果很酷炫，妹子像唐嫣

 yyforandroid


为什么折叠部分的效果只有一块啊。。。不应该一张图都显示吗

33楼 2016-09-05 09:51

 鸿洋\_

回复sinat\_24396671：就是一张图呀，折叠后肯定看起来的宽度变小了呀

Re: 2016-09-05 09:50

 yyforandroid

回复Imj623565791：不是，我的意思是只有8块折叠图中的一块。

Re: 2016-09-05 09:50

 dyamail

很赞

 Joshua\_zhanglz

很吊~非常吊~

 Say\_tomorrow

朋友，我想学习UI这块，有个大体的思路路线吗？谢谢了

 寻梦-finddreams

效果很好，当做学习可以，就是不够实用！

 nuanuan9

前排





姝然彩神

弘扬，留图不留种啊！！

27楼 20



鸿洋\_

回复qq315297923: <http://download.csdn.net/detail/lmj623565791/8505245> 种子交给你了。

Re: 20



AigeStudio

厉害！膜拜！19楼说得在理！



Cruzetang

回复aigestudio: 活捉



25楼 20



闭上眼睛看

yo~炫酷



志在愚乐

屌炸天

24楼 20



十七人华

太酷了。

23楼 20



xuqiang1990121

那个女的是不是唐嫣，我的女神呀

22楼 20



\_deadline

赞！

21楼 20



007ChengHao

Mark

20楼 20



frank909

鸿洋，这个讲得真不错。向爱哥那边靠齐了。

19楼 20



fenghebaiyang

Mark

18楼 20



qq\_21903005

前排

17楼 20



福临天下

厉害，膜拜！

16楼 20



linin29

纵轴减小的那个高度为什么是除以2，上下都减少了不应该是乘以2吗

15楼 20



eclipse\_xu

下一步准备引入曲线了是哇~~~

14楼 20

 JerryloveEmily

Re: 20

回复x359981514：这个可以有啊

 Buggersss

13楼 20

赞!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

 Hi-Daily

果然高产，占座看撸

 kirenty

刚更新的,大赞,马克!!!

 boosy2

先赞一个

dahuaaihuyue

9楼 20

高产

zxq9133

8楼 20

像我这种先大概浏览一遍之后再回帖的，肯定是没有沙发

 helloquartz

7楼 20

太快了作为都没了

 DongStudio


6楼 20

高产

 解忧杂货铺

5楼 20

挤挤。

 亓斌

4楼 20

二排

 JerryloveEmily

3楼 20

占座 开撸

 Yan-S

2楼 20

先顶

 small-dream

1楼 20

前排

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

\* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题

IE10

HTML

Cassandra

Compuware

Cloud Foundry

Hadoop

Eclipse

SDK

CloudStack

大数据

Redis

AWS

CRM

IIS

FTC

aptech

Scala

移动游戏

JavaScript

XML

coremail

Perl

Django

Java

数据库

LBS

OPhone

Tornado

Ruby

Android

Ubuntu

Unity

CouchBase

Hibernate

Solr

iOS

NFC

Splashtop

云计算

iOS6

Pure

Swift

WAP

UML

components

ThinkPHP

Solr

智能硬件

BI

Windows Mobile

Web App

An

Docker

HTML5

Rackspace

HBase

OpenStack

Spring

Web App

Solr

VPI

Apa

Rai

An

微信关注我的公众号



公司简介 | 招贤纳士 | 广告服务 | 银行汇款帐号 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

网站客服

杂志客服

微博客服

webmaster@csdn.net

400-600-2320

北京创新乐知信息技术有限公司 版权所有

江苏乐知网络技术有限公司

京 ICP 证 09002463 号 | Copyright © 1999-2014, CSDN.NET, All Rights Reserved

19



儿童编程



软件工程师待遇



软件工程师培训



程序员