一、相关概念

1、Drawable就是一个可画的对象，其可能是一张位图（BitmapDrawable），也可能是一个图形（ShapeDrawable），还有可能是一个图层（LayerDrawable），我们根据画图的需求，创建相应的可画对象

2、Canvas画布，绘图的目的区域，用于绘图

3、Bitmap位图，用于图的处理

4、Matrix矩阵

二、Bitmap

**1、从资源中获取Bitmap**

**Java代码  [收藏代码](javascript:void())**

1. Resources res = getResources();
2. Bitmap bmp = BitmapFactory.decodeResource(res, R.drawable.icon);

**2、Bitmap → byte[]**

**Java代码  [收藏代码](javascript:void())**

1. **public** **byte**[] Bitmap2Bytes(Bitmap bm) {
2. ByteArrayOutputStream baos = **new** ByteArrayOutputStream();
3. bm.compress(Bitmap.CompressFormat.PNG, 100, baos);
4. **return** baos.toByteArray();
5. }

**3、byte[] → Bitmap**

**Java代码  [收藏代码](javascript:void())**

1. **public** Bitmap Bytes2Bimap(**byte**[] b) {
2. **if** (b.length != 0) {
3. **return** BitmapFactory.decodeByteArray(b, 0, b.length);
4. } **else** {
5. **return** **null**;
6. }
7. }

**4、Bitmap缩放**

**Java代码  [收藏代码](javascript:void())**

1. **public** **static** Bitmap zoomBitmap(Bitmap bitmap, **int** width, **int** height) {
2. **int** w = bitmap.getWidth();
3. **int** h = bitmap.getHeight();
4. Matrix matrix = **new** Matrix();
5. **float** scaleWidth = ((**float**) width / w);
6. **float** scaleHeight = ((**float**) height / h);
7. matrix.postScale(scaleWidth, scaleHeight);
8. Bitmap newbmp = Bitmap.createBitmap(bitmap, 0, 0, w, h, matrix, **true**);
9. **return** newbmp;
10. }

**5、将Drawable转化为Bitmap**

**Java代码  [收藏代码](javascript:void())**

1. **public** **static** Bitmap drawableToBitmap(Drawable drawable) {
2. // 取 drawable 的长宽
3. **int** w = drawable.getIntrinsicWidth();
4. **int** h = drawable.getIntrinsicHeight();
6. // 取 drawable 的颜色格式
7. Bitmap.Config config = drawable.getOpacity() != PixelFormat.OPAQUE ? Bitmap.Config.ARGB\_8888
8. : Bitmap.Config.RGB\_565;
9. // 建立对应 bitmap
10. Bitmap bitmap = Bitmap.createBitmap(w, h, config);
11. // 建立对应 bitmap 的画布
12. Canvas canvas = **new** Canvas(bitmap);
13. drawable.setBounds(0, 0, w, h);
14. // 把 drawable 内容画到画布中
15. drawable.draw(canvas);
16. **return** bitmap;
17. }

**6、获得圆角图片**

**Java代码  [收藏代码](javascript:void())**

1. **public** **static** Bitmap getRoundedCornerBitmap(Bitmap bitmap, **float** roundPx) {
2. **int** w = bitmap.getWidth();
3. **int** h = bitmap.getHeight();
4. Bitmap output = Bitmap.createBitmap(w, h, Config.ARGB\_8888);
5. Canvas canvas = **new** Canvas(output);
6. **final** **int** color = 0xff424242;
7. **final** Paint paint = **new** Paint();
8. **final** Rect rect = **new** Rect(0, 0, w, h);
9. **final** RectF rectF = **new** RectF(rect);
10. paint.setAntiAlias(**true**);
11. canvas.drawARGB(0, 0, 0, 0);
12. paint.setColor(color);
13. canvas.drawRoundRect(rectF, roundPx, roundPx, paint);
14. paint.setXfermode(**new** PorterDuffXfermode(Mode.SRC\_IN));
15. canvas.drawBitmap(bitmap, rect, rect, paint);
17. **return** output;
18. }

**7、获得带倒影的图片**

**Java代码  [收藏代码](javascript:void())**

1. **public** **static** Bitmap createReflectionImageWithOrigin(Bitmap bitmap) {
2. **final** **int** reflectionGap = 4;
3. **int** w = bitmap.getWidth();
4. **int** h = bitmap.getHeight();
6. Matrix matrix = **new** Matrix();
7. matrix.preScale(1, -1);
9. Bitmap reflectionImage = Bitmap.createBitmap(bitmap, 0, h / 2, w,
10. h / 2, matrix, **false**);
12. Bitmap bitmapWithReflection = Bitmap.createBitmap(w, (h + h / 2),
13. Config.ARGB\_8888);
15. Canvas canvas = **new** Canvas(bitmapWithReflection);
16. canvas.drawBitmap(bitmap, 0, 0, **null**);
17. Paint deafalutPaint = **new** Paint();
18. canvas.drawRect(0, h, w, h + reflectionGap, deafalutPaint);
20. canvas.drawBitmap(reflectionImage, 0, h + reflectionGap, **null**);
22. Paint paint = **new** Paint();
23. LinearGradient shader = **new** LinearGradient(0, bitmap.getHeight(), 0,
24. bitmapWithReflection.getHeight() + reflectionGap, 0x70ffffff,
25. 0x00ffffff, TileMode.CLAMP);
26. paint.setShader(shader);
27. // Set the Transfer mode to be porter duff and destination in
28. paint.setXfermode(**new** PorterDuffXfermode(Mode.DST\_IN));
29. // Draw a rectangle using the paint with our linear gradient
30. canvas.drawRect(0, h, w, bitmapWithReflection.getHeight()
31. + reflectionGap, paint);
33. **return** bitmapWithReflection;
34. }

 三、Drawable

**1、Bitmap转换成Drawable**

**Java代码  [收藏代码](javascript:void())**

1. Bitmap bm=xxx; //xxx根据你的情况获取
2. BitmapDrawable bd= **new** BitmapDrawable(getResource(), bm);
3. 因为BtimapDrawable是Drawable的子类，最终直接使用bd对象即可。

**2、Drawable缩放**

**Java代码  [收藏代码](javascript:void())**

1. **public** **static** Drawable zoomDrawable(Drawable drawable, **int** w, **int** h) {
2. **int** width = drawable.getIntrinsicWidth();
3. **int** height = drawable.getIntrinsicHeight();
4. // drawable转换成bitmap
5. Bitmap oldbmp = drawableToBitmap(drawable);
6. // 创建操作图片用的Matrix对象
7. Matrix matrix = **new** Matrix();
8. // 计算缩放比例
9. **float** sx = ((**float**) w / width);
10. **float** sy = ((**float**) h / height);
11. // 设置缩放比例
12. matrix.postScale(sx, sy);
13. // 建立新的bitmap，其内容是对原bitmap的缩放后的图
14. Bitmap newbmp = Bitmap.createBitmap(oldbmp, 0, 0, width, height,
15. matrix, **true**);
16. **return** **new** BitmapDrawable(newbmp);
17. }