# texture与sprite的区别

Texture :保存在GPU缓冲中的一张纹理。

TextureRegion： 定义了一块矩形区域的纹理。主要作用是从一块纹理中择取需要使用的矩形区域。

Sprite：Holds the geometry, color, and texture information for drawing 2D。这个类是继承于TextureRegion，是对TextureRegion的功能的扩展，相比增加了旋转，动画等效果。

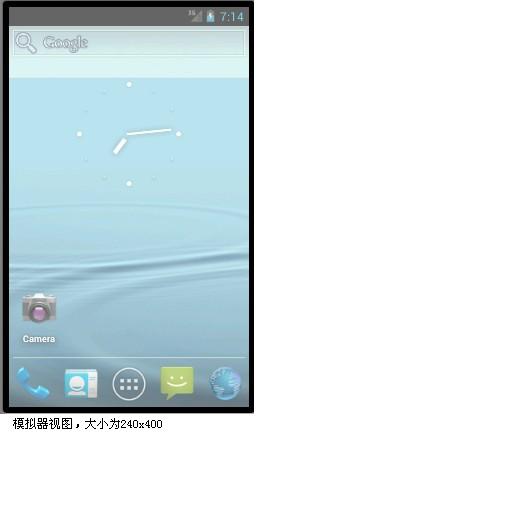
SpriteBatch：A SpriteBatch is used to draw 2D rectangles that reference a texture (region). 主要处理对纹理的绘画，绘画前的优化等。

从源码中可以知道：

Sprite 可以使用 TextureRegion构造，TextureRegion可以使用Texture构造。所以SpriteBatch在底层最终操作的还是Texture。他们的关系就是这些了。

上代码比较他们的区别：

环境：我的模拟器分辨率是240X400，例子中使用到的图片是512X512尺寸的。注意纹理图片必须是2的N次幂大小。





测试图片 大小512x512

使用纹理texture绘画：

1 private Texture texture;//定义一张纹理

2 private SpriteBatch spriteBatch;//定义一个纹理绘画器

3 private TextureRegion textureRegion;//定义一个纹理区域

4

5 public MogulListener() {

6

7 }

8 @Override

9 public void create() {

10 texture =new Texture(Gdx.files.internal("wallpaper.jpg"));

11 spriteBatch =new SpriteBatch();

12 }

13 @Override

14 public void render() {

15 Gdx.gl.glClear(GL10.GL\_COLOR\_BUFFER\_BIT);

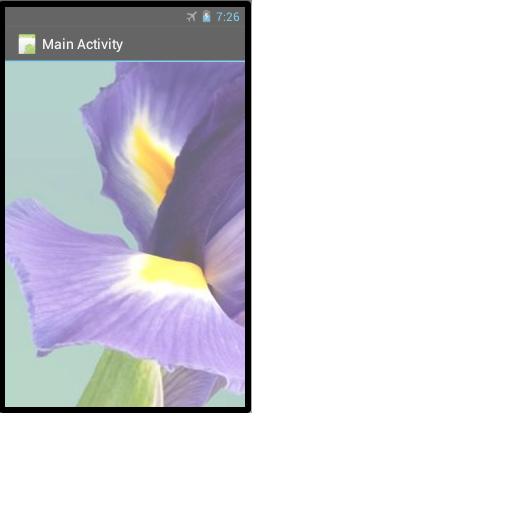
16 spriteBatch.begin();

17 spriteBatch.draw(texture, 0,0);//根据屏幕大小，只能画出从左下角开始大小为240x400的图片。

18 spriteBatch.end();

19

20 }



由于纹理不能设置可见区域的坐标和大小，所以看到的图片如下，还有一部分没有画出来：

使用纹理区域绘图：可以定义区域在纹理中的相对坐标和大小。下面画出图片的右上角部分。

复制代码

1 @Override

2 public void create() {

3 texture =new Texture(Gdx.files.internal("wallpaper.jpg"));

4 textureRegion=new TextureRegion(texture,272,112,240,400);

5 spriteBatch =new SpriteBatch();

6 }

7 @Override

8 public void render() {

9 Gdx.gl.glClear(GL10.GL\_COLOR\_BUFFER\_BIT);

10 spriteBatch.begin();

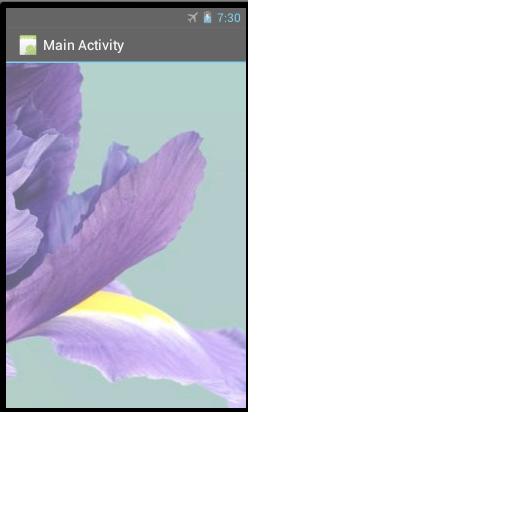
11 spriteBatch.draw(textureRegion, 0,0);

12 spriteBatch.end();

13

14 }

效果图：



使用sprite绘图。由于Sprite继承于TextureRegion，所以Sprite拥有TextureRegion所有属性。Sprite还实现了一些自己特有的属性，比如旋转动画等。

复制代码

1 private Texture texture;

2 private SpriteBatch spriteBatch;

3 private TextureRegion textureRegion;

4 private Sprite sprite;

5 public MogulListener() {

6

7 }

8 @Override

9 public void create() {

10 texture =new Texture(Gdx.files.internal("wallpaper.jpg"));

11 sprite=new Sprite(texture, 292, 20,220, 380); //画右下角图片

12 sprite.setPosition(20, 20); //从屏幕坐标20,20开始绘画

13 sprite.setRotation(45); //旋转45度

14 spriteBatch =new SpriteBatch();

15 }

16 @Override

17 public void render() {

18 Gdx.gl.glClear(GL10.GL\_COLOR\_BUFFER\_BIT);

19 spriteBatch.begin();

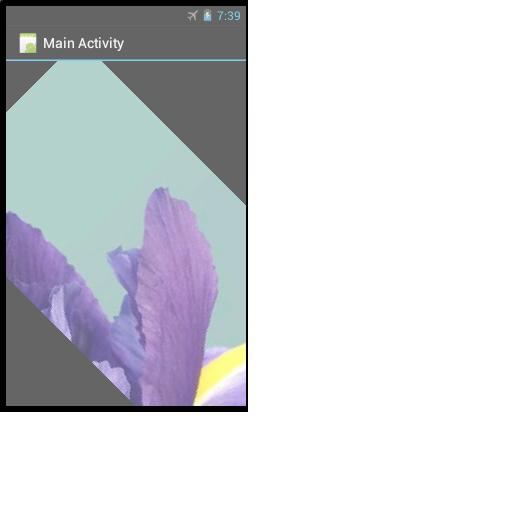
20 sprite.draw(spriteBatch);

21 spriteBatch.end();

22

23 }

效果图：



这里偷懒了，没有将状态栏和应用的label去掉，影响效果显示。

---------------------

原文：https://blog.csdn.net/huaweitman/article/details/11477493