

Problems rotating BufferedImage

CAREERS 2.0
by stackoverflow



+



Have projects on Google Code?
Import them easily to your profile

I have some problems with rotating images in Java using the AffineTransform class.

I have the following method for creating a rotated (90 degrees) copy of an image:

```
private BufferedImage createRotatedCopy(BufferedImage img, Rotation rotation) {
    int w = img.getWidth();
    int h = img.getHeight();

    BufferedImage rot = new BufferedImage(h, w, BufferedImage.TYPE_INT_RGB);

    double theta;
    switch (rotation) {
        case CLOCKWISE:
            theta = Math.PI / 2;
            break;
        case COUNTERCLOCKWISE:
            theta = -Math.PI / 2;
            break;
        default:
            throw new AssertionError();
    }

    AffineTransform xform = AffineTransform.getRotateInstance(theta, w / 2, h / 2);
    Graphics2D g = (Graphics2D) rot.createGraphics();
    g.drawImage(img, xform, null);
    g.dispose();

    return rot;
}
```

Rotation is a simple enum with the values NONE, CLOCKWISE and COUNTERCLOCKWISE.

The symptoms of my problems are displayed here:

http://perp.se/so/rotate_problems.html

So, the rotation works OK, but the resulting images aren't anchored to the correct coordinates (or how one should put it). And since I don't really know what the heck I'm doing in the first place (my linear algebra is weak), I don't know how to solve this on my own.

I've tried with some random fiddling with the AffineTransform instance, but it hasn't helped me (of course). I've tried googling (and searching SO), but all examples I've seen basically use the same approach as I do... which doesn't work for me.

Thankful for advice.

[java](#) [image](#) [rotation](#) [affinetransform](#)

asked Feb 13 '10 at 10:07



perp

867 7 15

88% accept rate

1 Equivalent question for .NET: stackoverflow.com/questions/2225363/c-rotate-bitmap-90-degrees – finnw Feb 13 '10 at 10:46

feedback

4 Answers

If you must express the transform as a single rotation, the anchor point depends on the direction of rotation:
Either $(w/2, w/2)$ or $(h/2, h/2)$.

But it's probably simpler to express as `translate; rotate; translate`, e.g.

```
AffineTransform xform = new AffineTransform();
xform.translate(0.5*h, 0.5*w);
xform.rotate(theta);
xform.translate(-0.5*w, -0.5*h);
```

Also consider using `getQuadrantRotateInstance` instead of `getRotateInstance`.

edited Feb 13 '10 at 16:18

answered Feb 13 '10 at 10:45



finnw

16.2k 1 42 88

I get what you're saying, and I have tried that too. The problem persists, however. They're still being drawn "outside" the target area, albeit on the "other side" horizontally, so to speak. I could supply some more screenshots if it's unclear what I'm getting at. – [perp](#) Feb 13 '10 at 12:35

@perp, you're right, fixed. I've tested the new version and it works. – [finnw](#) Feb 13 '10 at 16:19

Works like a charm! I think I have a better understanding of what's going on now, too. Thanks! – [perp](#) Feb 13 '10 at 16:43

feedback



Since you only need 90 degree rotation you can avoid using the AffineTransform stuff:

```
public BufferedImage rotate90DX(BufferedImage bi) {
    int width = bi.getWidth();
    int height = bi.getHeight();
    BufferedImage biFlip = new BufferedImage(height, width, bi.getType());
    for(int i=0; i<width; i++)
        for(int j=0; j<height; j++)
            biFlip.setRGB(height-1-j, width-1-i, bi.getRGB(i, j));
    return biFlip;
}
```

This also avoids cutting off edges of rectangular images.

From: <http://snippets.dzone.com/posts/show/2936>

answered Jul 7 '11 at 12:37



David Tinker

2,123 10 27

- 3 Just a heads up, forcing a direct access to the RGB pixel values takes the image out of the accelerated pipeline in Java2D. This approach may give you the result you want, but it is much slower. – [Riyad Kalla](#) Aug 3 '11 at 1:46

Aha. Didn't know that. Thanks – [David Tinker](#) Aug 3 '11 at 8:08

feedback

You could try an alternative approach and create an Icon from the image and then use a [Rotated Icon](#).

Or you can try this old code I found in the Sun forums:

```
import java.awt.*;
import java.awt.geom.*;
import java.awt.image.*;
import java.io.*;
import java.net.*;
import javax.imageio.*;
import javax.swing.*;

public class RotateImage {
    public static void main(String[] args) throws IOException {
        URL url = new URL("https://blogs.oracle.com/jag/resource/JagHeadshot-small.jpg");
        BufferedImage original = ImageIO.read(url);
        GraphicsConfiguration gc = getDefaultConfiguration();
        BufferedImage rotated1 = tilt(original, -Math.PI/2, gc);
    }
}
```

```

        BufferedImage rotated2 = tilt(original, +Math.PI/4, gc);
        BufferedImage rotated3 = tilt(original, Math.PI, gc);
        display(original, rotated1, rotated2, rotated3);
    }


    public static BufferedImage tilt(BufferedImage image, double angle, GraphicsConfigu
        double sin = Math.abs(Math.sin(angle)), cos = Math.abs(Math.cos(angle));
        int w = image.getWidth(), h = image.getHeight();
        int neww = (int)Math.floor(w*cos+h*sin), newh = (int)Math.floor(h*cos+w*sin);
        int transparency = image.getColorModel().getTransparency();
        BufferedImage result = gc.createCompatibleImage(neww, newh, transparency);
        Graphics2D g = result.createGraphics();
        g.translate((neww-w)/2, (newh-h)/2);
        g.rotate(angle, w/2, h/2);
        g.drawRenderedImage(image, null);
        return result;
    }

    public static GraphicsConfiguration getDefaultConfiguration() {
        GraphicsEnvironment ge = GraphicsEnvironment.getLocalGraphicsEnvironment();
        GraphicsDevice gd = ge.getDefaultScreenDevice();
        return gd.getDefaultConfiguration();
    }

    public static void display(BufferedImage im1, BufferedImage im2, BufferedImage im3,
        JPanel cp = new JPanel(new GridLayout(2,2));
        addImage(cp, im1, "original");
        addImage(cp, im2, "rotate -PI/2");
        addImage(cp, im3, "rotate +PI/4");
        addImage(cp, im4, "rotate PI");

        JFrame f = new JFrame("RotateImage");
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        f.setContentPane(cp);
    }

```

edited Sep 9 at 17:05
 Luke Woodward
 11.9k 2 20 35

answered Feb 13 '10 at 16:12
 camickr
 60.8k 4 18 47

feedback

I don't know if this might be your issue.

```
AffineTransform xform = AffineTransform.getRotateInstance(theta, w / 2, h / 2);
```


Why not try?

```
AffineTransform xform = AffineTransform.getRotateInstance(theta);
```

OR

```
g.transform(AffineTransform.getRotateInstance(theta));
g.drawImage(img, 0, 0, w/2, h/2, null, null);
```

edited Feb 13 '10 at 16:06

answered Feb 13 '10 at 10:12
 The Elite Gentleman
 24.2k 3 29 61

I have tried it. :-) – perp Feb 13 '10 at 12:31

feedback

Not the answer you're looking for? Browse other questions tagged [java](#) [image](#)

[rotation](#) [affinetransform](#) or [ask your own question](#).