

Re: Math.cos() problems

Source: <http://coding.derkeiler.com/Archive/Java/comp.lang.java.programmer/2007-12/msg01030.html>

- *From:* Daniel Pitts <newsgroup.spamfilter@xxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Thu, 06 Dec 2007 12:18:33 -0800
-

Brandon McCombs wrote:

Hello,

I'm trying to do something simple but Java turns it into a federal case. All I want to do is draw a couple circles on a Graphics2D canvas with one movable and attracted to the 2nd one which is stationary. Basically I'm just trying to get basic functionality working that simulates gravitational attraction between the spheres.

I had everything working concerning the movement calculations (force, acceleration, velocity) up to the point of performing the calculation that would generate each new pixel positions for the circle that I am allowing to move. I ran into problems because I had to determine the angle between the moving circle and the stationary one and use the angle in the equations $x = r\cos(\text{angle})$ and $y = r\sin(\text{angle})$ so I could determine how much the movable circle should move.

The problem is that I have no clue how Java is calculating the cosine and sine of the angle I calculate (even calculating the angle is a chore which may be confusing the problem even further). With respect to what screen location are the trig functions being evaluated? (0,0) on the drawing canvas? Does it matter? I thought it did especially since my angle is based on the location of one of my circles and it isn't at 0,0 on the canvas. Where am I going wrong? This is frustrating. If you need more info let me know.

thanks

Math.cos, Math.sin, and Math.atan2 deal with radians. Make sure that your code is using the appropriate units in the appropriate places.

It may help to create an Angle class that abstracts the unit from the user, and implements the specific functions you need. This is all part of avoiding primitive obsessions.

<http://virtualinfinity.net/wordpress/program-design/2007/10/28/primitive-obsession/>

For example, you might decide to use this approach:

```
class Angle {
private double radians;
```

Re: Math.cos() problems

```
public double sine() {
return Math.sin(getRadians());
}

public double cosine() {
return Math.cos(getRadians());
}

public void setFromVector(double x, double y) {
setRadians(Math.atan2(y, x));
}

public double getRadians() {
return radians;
}

public void setRadians(double radians) {
this.radians = radians;
}

public double getDegrees() {
return getRadians() / Math.PI * 180.0;
}

public void setDegrees(double degrees) {
setRadians(degrees / 180.0 * Math.PI);
}
}
```

This particular version is a mutable instance. I personally prefer making "radians" final and using factory methods in lieu of the set methods. I also prefer using a vector class (not java.util.Vector), to represent the x/y coordinates. vector could also include getAngle() (which would delegate to Angle.fromVector(getX(), getY())).

—
Daniel Pitts' Tech Blog: <<http://virtualinfinity.net/wordpress/>>