How to move an object into a target in Java ME

This article explains how to move a projectile towards an enemy in Java ME. The article requires an understanding of Java ME basics and simple threading, and also trigonometry (pythagorean theorum)

Introduction

Have you ever played a shoot'em up space shooter game and wonder how the enemy always shoot at you (at an angle)? This article explains exactly how to code this. It requires an understanding of Java ME basics and simple threading, and also trigonometry (pythagorean theorem). The code is self-explanatory (commented).

Code

```
import java.util.Random;
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Graphics;
import javax.microedition.midlet.MIDlet;
import javax.microedition.midlet.MIDletStateChangeException;


 /* A demonstration of moving a bullet ("O") into a target ("X").
 *  In game programming, this is very useful.
 *  For simplicity we use drawString instead of Sprites and only this class.
 *  @author George Roberto Peres.  reflexus@ig.com.br (JavaMan)
 */

public class TestAng extends MIDlet implements Runnable {
  private GameScreen myCanvas;
  private boolean running;

  public TestAng() {
    // Construct a canvas.
    myCanvas = new GameScreen();
    running = true;
    Thread t = new Thread(this);
    t.start();
  }

  public void run() {
    while (running) {
      myCanvas.repaint();
      myCanvas.move();
      try { Thread.sleep(1); }
      catch (InterruptedException e) { }
    }
  }

  public void startApp() throws MIDletStateChangeException {
    Display.getDisplay(this).setCurrent(myCanvas);
  }

  public void pauseApp() {
    //
  }
```


(C) Copyright Nokia 2014. All rights reserved.

```
  public void destroyApp(boolean destroy) throws MIDletStateChangeException {
    running = false;
    System.gc();
    notifyDestroyed();
  }

  // inner class = the canvas
  private class GameScreen extends Canvas {

    private int width; // canvas width
    private int height; // canvas height
    // "O" position (for example, enemy projectile)
    private double x;
    private double y;
    //
    private double ix;  // increment x axis
    private double iy;  // increment y axis
    //
    private int angle;  // angle in degrees
    // target = "X" position (for example, player position)
    private int targetX;
    private int targetY;
    //
    private static final double VEL = 2;   // velocity
    // pseudo-random seed
    private final Random RAND = new Random();
    //
    private final int GRAPHICS_TOP_LEFT = Graphics.TOP | Graphics.LEFT;   // anchor


    public GameScreen() {
      setFullScreenMode(true);
      // getWidth() & getHeight() after setFullScreenMode(boolean)
      width = getWidth();
      height = getHeight();
      // set "O" position centered on the screen
      x = width/2;
      y = height/2;
      setIncrement();
      //for (int i=0; i<360; i++) System.out.println(i+" atan =
"+Math.toDegrees(atan(i)));
    }

    // set increment ix & iy based on target position
    private void setIncrement() {
      double catX = Math.abs(targetX - x);  // cateto x axis
      double catY = Math.abs(targetY - y);  // cateto y axis
      angle = getAngle((int)catX, (int)catY);
      // distance between "O" and "X". Can be used, for example to detect the player’s
      // presence within a certain range; if the player is within that range, the enemy
      // will fire only if the player is far enough
      double hipotenusa = Math.sqrt(catX*catX + catY*catY);
      // set increment. inverted iy to match screen coordinates
      if (targetX > x) ix = (catX / hipotenusa)*VEL; else ix = -(catX / hipotenusa)*VEL;
      if (targetY > y) iy = (catY / hipotenusa)*VEL; else iy = -(catY / hipotenusa)*VEL;
    }
```


(C) Copyright Nokia 2014. All rights reserved.

```
    private int randomNumber(int max) {
      return Math.abs(RAND.nextInt() % max);   // abs value from -(max-1) to max-1
    }


    // return the angle in degrees
    private int getAngle(float catX, float catY) {
      int ang;
      if (catX==0 || catY==0) ang=0; else ang=(int)(Math.toDegrees(atan(catY/catX)));
      //System.out.println("catX="+catX+"    catY="+catY);
      //System.out.println("ang="+ang+"   x="+x+"   y="+y+"   targetX="+targetX+"
targetY="+targetY);
      // adjust for which quadrant we are, based on O and X coordinates
      // left side.
      if (targetX < x) {
        if (targetY < y) return 180 - ang;  //2
        return 180 + ang;  //3
      }
      // right side.
      else
      {
        if (targetY < y) { if (targetX == x) return 90; else return ang; }  //1
        if (targetX == x) return 270; else return 360 - ang;  //4
      }
    }


    // return arc tangent of a (in radians). (c) Sun MicroSystems, Inc.
    private float atan(float a) {
      //formula:
      //atan(x) = x/(1+ 0.28*x^2) (|x|<=1)
      //atan(x) = pi/2 - x/(x^2 + 0.28) (|x|>=1)
      if (Math.abs(a) <= 1.0f) {
        return (a / (1 + 0.28f * (a * a)));
      }
      else
      {
        float retval = (((float) Math.PI) / 2.0f) - (a / ((a * a) + 0.28f));
        if (a < -1.0f) {
          return (retval - (float) Math.PI);
        }
        else
        {
          //if a > 1.0f
          return retval;
        }
      }
    }


    // implements the javax.microedition.lcdui.Canvas method
    public void paint(Graphics g) {
      // Clear the screen.
      g.setColor(0);
```


(C) Copyright Nokia 2014. All rights reserved.

```
      g.fillRect(0, 0, width, height);
      // bullet
      g.setColor(255,255,255);
      g.drawString("O", (int)x, (int)y, GRAPHICS_TOP_LEFT);
      // target
      g.setColor(120,196,112);
      g.drawString("X", targetX, targetY, GRAPHICS_TOP_LEFT);
      // coordinates and angle
      g.setColor(240,148,80);
      g.drawString("x:"+(int)x+" y:"+(int)y+" angle:"+angle, 0, 0, GRAPHICS_TOP_LEFT);
    }

    /**
     * Moves the "O" based on the target "X" (targetX, targetY).
     */
    public void move() {
      x += ix;  // Add the x component of the movement
      y += iy;  // Add the y component of the movement
      // Check "O" position and wrap around to the center of the canvas,
      // set new "X" random position and its increment
      if (x < 0 || x > width || y < 0 || y > height) {
        x = width/2;
        y = height/2;
        targetX = randomNumber(230);
        targetY = randomNumber(295)+10;
        setIncrement();
      }
    }
  }
}
```


(C) Copyright Nokia 2014. All rights reserved.