import java.awt.Canvas;

import java.awt.Color;

import java.awt.Dimension;

import java.awt.Graphics;

import java.awt.image.BufferStrategy;

import java.awt.image.BufferedImage;

import javax.swing.JFrame;

public class Game extends Canvas implements Runnable{

private static final long serialVersionUID = 1L;

public static JFrame frames;

private Thread thread;

private boolean isRunning = true;

private final int WIDTH = 240;

private final int HEIGHT = 160;

private final int SCALE = 3;

private BufferedImage image;

public Game() {

setPreferredSize(new Dimension(WIDTH\*SCALE,HEIGHT\*SCALE));

initFrame();

image = new BufferedImage(WIDTH,HEIGHT,BufferedImage.TYPE\_INT\_RGB);

}

public void initFrame() {

frames= new JFrame("Zelda Clone");

frames.add(this);

frames.setResizable(false);

frames.pack();

frames.setLocationRelativeTo(null);

frames.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frames.setVisible(true);

}

public synchronized void start() {

thread = new Thread(this);

isRunning = true;

thread.start();

}

public synchronized void stop() {

isRunning = false;

try {

thread.join();

} catch (InterruptedException e) {

e.printStackTrace();

}

}

public static void main(String [] args) {

Game game = new Game();

game.start();

}

public void tick() {

}

public void render() {

BufferStrategy bs = this.getBufferStrategy();

if(bs == null) {

this.createBufferStrategy(3);

return;

}

Graphics g = image.getGraphics();

g.setColor(new Color(0,0,0));

g.fillRect(0, 0, WIDTH,HEIGHT);

/\*\*/

g.dispose();

g = bs.getDrawGraphics();

g.drawImage(image,0,0,WIDTH\*SCALE,HEIGHT\*SCALE,null);

bs.show();

}

public void run() {

long lastTime = System.nanoTime();

double amontOfTicks = 60.0;

double ns = 1000000000 / amontOfTicks;

double delta = 0;

int frames = 0;

double timer = System.currentTimeMillis();

while(isRunning) {

long now = System.nanoTime();

delta+= (now - lastTime) / ns;

lastTime = now;

if(delta >= 1) {

tick();

render();

frames++;

delta--;

}

if(System.currentTimeMillis() - timer >= 1000) {

System.out.println("FPS: "+ frames);

frames = 0;

timer+= 1000;

}

}

stop();

}

}