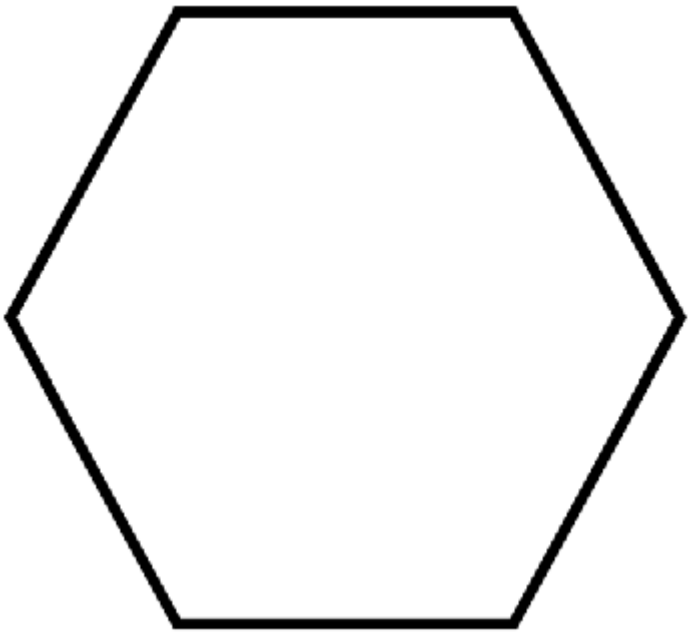
HexGrid.java

A program to illustrate object oriented programming

Evan Cummings

November 22, 2009



Goal:

Create a framework for a wargame board for later use in a two player strategy wargame which consists of a map designer.

Analysis:

Design a hexagon class that defines a hexagon object and draw an array of these hexagon objects on a panel. Include methods for filling the hexagons with a color of the user's choice by clicking on the the hexagon with the mouse.

Design:

Hexagon class - Includes all the math for drawing a hexagon. The constructor's formal parameters should include an x and y coordinate and the length of one of the hex's sides. Include all necessary accessor and mutator methods for class variables. Create a method for drawing the hexagon on the string by drawing lines from the points that were instantiated by the constructor. Include a method for filling the hexagon with color by drawing a series of smaller hexagons using recursion.

HexGridPanel class – JPanel child class that creates and draws any number of hexagon objects. Define class variables for the width and height of window, length of a hexagon side, number of hexagons per window by using the length of a hexagon side, and the current color. Constructor should load a two-dimensional array with new hexagon objects of specified side length. The paintComponent method should draw this array on the screen row by row, and moving each even row over by a set amount to make a honey comb pattern. Add a MouseListener class that waits for mouse clicks within the grid and then loads that grid coordinate into another two dimensional array of x and y coordinates. Create an if-statement in the paint component method that determines if there is something in the array of filled hexes and fills each hex with color. Add buttons for selecting a color.

HexGrid class – JFrame child class that holds the HexGridPanel class.

Implementation:

I First looked up the math to draw a hexagon and used that math to make the hexagon class. I built the framework for the HexGridPanel and HexGrid classes and after a bit of testing was able to display the array of hexagons. I added a MouseListener class and went about loading each hexagon that was clicked into an array of “filled” hexagons and painted them on the screen. I then wrote a method for filling the hexagon with color by drawing a series of successively smaller hexagons within the “filled” hexagon. I found that this did not work because I was casting floating points into integers and loosing precision which created holes in my hexes. I decided that it would be far simpler to just draw a hexagon with the drawPolygon method of the Graphics class with the points calculated in the constructor of the hexagon class. After the hexagon class drew the hexagons correctly I decided that the array of filled hexes was not really necessary and to instead change a color value of the hexagon object. Now instead of

Implementation (Continued):

loading an array of hexes and redrawing the hexagons on the screen, the mouse click defines the hexagon of the set color and repaints the screen.

Along the way I also decided to add the HexGridPanel class to another panel class called GamePanel, added the ability to drag the color across the screen, provided a color chooser dialog box to pick any color the user desires, and cleaned up the remaining bits of code that remained from previous builds.

All in all this was a very satisfying project, and now that it is done I see that I have a perfect framework for my next project. I learned many things along the way, but the most important thing I have learned is to draw up a UML diagram before going into the project. I realized part way through that I would have saved myself quite a bit of time if I had laid all my functions out in front of me and organized them in a logical manner. I remember quite well being told that it was a very important first step, but I guess learning by experience was necessary.

Appendix A: The Code

```
//Hexagon.java
//Evan Cummings
//11.10.9

import java.awt.Graphics;
import java.awt.Point;
import java.awt.Color;
import java.awt.Polygon;

public class Hexagon
{
	private int h, r, b, a, s, x, y;
	private Polygon p;
	private int[] xPts;
	private int[] yPts;
	private Color c = new Color(0, 150, 0);

	// Constructor.
	public Hexagon(int x, int y, int s)
	{
		this.s = s;
		this.x = x;
		this.y = y;

		h = (int)((float)Math.sin(Math.PI/6) * s);
		r = (int)((float)Math.cos(Math.PI/6) * s);
		b = s + 2 * h;
		a = 2 * r;

		int p1X, p2X, p3X, p4X, p5X, p6X;
		int p1Y, p2Y, p3Y, p4Y, p5Y, p6Y;

		p1X = x;
		p2X = x+r;
		p3X = x+2*r;
		p4X = x+2*r;
		p5X = x+r;
		p6X = x;

		p1Y = y+h;
		p2Y = y;
		p3Y = y+h;
```

Hexagon.java (Continued):

```
            p4Y = y+h+s;
            p5Y = y+2*h+s;
            p6Y = y+h+s;

            int []xPts = {p1X, p2X, p3X, p4X, p5X, p6X};
            int []yPts = {p1Y, p2Y, p3Y, p4Y, p5Y, p6Y};

            this.xPts = xPts;
            this.yPts = yPts;

            p = new Polygon(xPts, yPts, 6);
    }

    public int getB()
    {
            return b;
    }

    public int getA()
    {
            return a;
    }

    public int getH()
    {
            return h;
    }

    public int getR()
    {
            return r;
    }

    public int getS()
    {
            return s;
    }

    public int getX()
    {
            return x;
    }
```

Hexagon.java (Continued):

```
	public int getY()
	{
		return y;
	}

	public Color getColor()
	{
		return c;
	}

	public void setColor(Color c)
	{
		this.c = c;
	}

	// Draws hexagon polygon object.
	public void drawHex(Graphics page)
	{
		fillHex(page, c);
		page.setColor(Color.black);
		page.drawPolygon(p);
	}

	// Fills hexagon polygon object.
	public void fillHex(Graphics page, Color c)
	{
		this.c = c;
		page.setColor(c);
		page.fillPolygon(p);
	}

	public String toString()
	{
		String str = "";

		str += "Side: " + Integer.toString(s);
		str += "\nH: " + Integer.toString(h);
		str += "\nR: " + Integer.toString(r);
		str += "\nA: " + Integer.toString(b);
		str += "\nB: " + Integer.toString(a);

		return str;
	}
}
```

```
//HexGridPanel.java
//Evan Cummings
//11.10.9

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;

public class HexGridPanel extends JPanel
{
	// Screen and hex side dimensions.
	private final int WIDTH = 640;
	protected static final int HEIGHT = 480;
	private final int S = 20;

	//	Example hex object for calculations.
	private Hexagon hexEx = new Hexagon(0, 0, S);

	// Hex calculations.
	private int h = hexEx.getH();
	private int r = hexEx.getR();
	private int b = hexEx.getB();
	private int a = hexEx.getA();
	private int s = hexEx.getS();
	private int x = hexEx.getX();
	private int y = hexEx.getY();

	// Gradient of angled hex edge.
	float m = (float)h / r;

	// Number of hexagons on the screen.
	private int numRow = HEIGHT / (h + S);
	private int numCol = WIDTH / a;

	// 2D array of hex objects.
	private Hexagon [][] grid = new Hexagon [numRow][numCol];

	// point used for display of x, y coordinates.
	private Point p = null;

	// Dimension of reference grid.
	private int sectX, sectY;

	// Coordinates within reference grid.
	private int sectPxlX, sectPxlY;
```

HexGridPanel.java (Continued):

```
// Type of reference Box.
private char sectTyp = ' ';

// Final hex array coordinates.
private int arrayX, arrayY;

// Color to fill.
protected Color color = Color.cyan;

// Constructor
public HexGridPanel()
{
	setPreferredSize(new Dimension(WIDTH -8, HEIGHT +13));
	setBackground(Color.black);

	MyMouseListener mListener = new MyMouseListener();
	addMouseMotionListener (mListener);
	addMouseListener (mListener);

	System.out.println(hexEx);
	System.out.println("\nNumRows: " + numRow +
		"\nNumCols: " + numCol);

	// array loader
	for (int r = 0; r < numRow; r++)
	{
		for (int c = 0; c < numCol; c++)
		{
			int PxX = c * 2 * this.r + (r & 1) * this.r;
			int PxY = r * (h + s);
			grid[r][c] = new Hexagon(PxX, PxY, S);
		}
	}
}

// Draw each individual hexagon object
public void paintComponent(Graphics page)
{
	super.paintComponent(page);

	// draw hexes
	page.setColor(new Color(50,50,50));
	for (int r = 0; r < numRow; r++)
	{
```

HexGridPanel.java (Continued):

```
                        for (int c = 0; c < numCol; c++)
                        {
                                grid[r][c].drawHex(page);
                        }
                }

                // print out coordinates for testing
        /*      if (p != null)
    {
      page.setColor (Color.yellow);

        if (p.x > WIDTH-42)
                        {
        page.drawString (p.x + " " + p.y, p.x-42, p.y);
                                page.drawString (sectX + " " + sectY, p.x-42, p.y-10);
                                page.drawString (sectPxlX + " " + sectPxlY, p.x-42, p.y-20);
                                page.drawString (arrayX + " " + arrayY, p.x-42, p.y);
                                page.drawString (Character.toString(sectTyp), p.x-42, p.y-40);
                }

      else
                        {
        page.drawString (p.x + " " + p.y, p.x, p.y);
                                page.drawString (sectX + " " + sectY, p.x, p.y-10);
                                page.drawString (sectPxlX + " " + sectPxlY, p.x, p.y-20);
                                page.drawString (arrayX + " " + arrayY, p.x, p.y);
                                page.drawString (Character.toString(sectTyp), p.x, p.y-40);
                        }
                } */
        }

        // Gets current color.
        public Color getColor()
        {
                return color;
        }

        // Sets color.
        public void setColor(Color c)
        {
                color = c;
        }
```

HexGridPanel.java (Continued):

```
// Evaluate Hex Coordinates.
public Point getHexCoord(Point p)
{
        sectX = (int)p.getX() / (2 * r);
        sectY = (int)p.getY() / (h + s);

        sectPxlX = (int)p.getX() % (2 * r);
        sectPxlY = (int)p.getY() % (h + s);

        if ((sectY & 1) == 0)
                sectTyp = 'A';
        else
                sectTyp = 'B';

        if (sectTyp == 'A')
        {
                // middle
                arrayY = sectY;
                arrayX = sectX;
                // left Edge
                if (sectPxlY < (h - sectPxlX * m))
                {
                  arrayY = sectY - 1;
                  arrayX = sectX - 1;
                }
                // right Edge
                if (sectPxlY < (-h + sectPxlX * m))
                {
                  arrayY = sectY - 1;
                  arrayX = sectX;
                }
        }

        if (sectTyp == 'B')
        {
                // right side
                if (sectPxlX  >= r)
                {
                        if (sectPxlY < (2 * h - sectPxlX * m))
                        {
                                arrayY = sectY - 1;
                                arrayX = sectX;
                        }
```

HexGridPanel.java (Continued):

```
                        else
                        {
                                arrayY = sectY;
                                arrayX = sectX;
                        }
                }
                // left side
                if (sectPxlX < r)
                {
                        if (sectPxlY < (sectPxlX * m))
                        {
                                arrayY = sectY - 1;
                                arrayX = sectX;
                        }
                        else
                        {
                                arrayY = sectY;
                                arrayX = sectX - 1;
                        }
                }
        }

        Point temp = new Point(arrayX, arrayY);
        return temp;
}

// Mouse input events.
private class MyMouseListener implements MouseListener,

  MouseMotionListener
{
        // Set color variable for hex coordinate.
        public void mousePressed(MouseEvent e)
        {
                p = e.getPoint();
                p = getHexCoord(p);

                int hexX = (int)p.getX();
                int hexY = (int)p.getY();

                if (hexY < numRow && hexX < numCol &&
                        hexY >= 0 && hexX >= 0)
                        grid[hexY][hexX].setColor(color);
```

HexGridPanel.java (Continued):

```
                repaint();
            }

            public void mouseReleased(MouseEvent e) {}
            public void mouseEntered(MouseEvent e) {}
            public void mouseExited(MouseEvent e) {}
            public void mouseClicked(MouseEvent e) {}

            // Display hex coordinates under mouse cursor.
            public void mouseMoved(MouseEvent e)
            {
                p = e.getPoint();
                repaint();
            }

            // Set color variable for hex coordinate.
            public void mouseDragged(MouseEvent e)
            {
                p = e.getPoint();
                p = getHexCoord(p);

                int hexX = (int)p.getX();
                int hexY = (int)p.getY();

                if (hexY < numRow && hexX < numCol &&
                    hexY >= 0 && hexX >= 0)
                    grid[hexY][hexX].setColor(color);

                repaint();
            }
        }
}
```

```
// GamePanel.java
// Evan Cummings
// 11.22.9

import javax.swing.*;
import java.awt.Dimension;
import java.awt.Color;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class GamePanel extends JPanel
{
	// Panel for the grid and controls.
	private JPanel controlPanel, colorPanel;
	private HexGridPanel hexGridPanel;

	private JButton green, blue, brown, chooser;

	private JLabel title;

	private int h;

	// Constructor
	public GamePanel()
	{
		MyListener listener = new MyListener();
		Dimension dim = new Dimension(80, 20);
		h = HexGridPanel.HEIGHT;

		hexGridPanel = new HexGridPanel();
		controlPanel = new JPanel();

		colorPanel = new JPanel();
		colorPanel.setPreferredSize(new Dimension(300, 100));

		controlPanel.setPreferredSize(new Dimension(100, h));
		controlPanel.setBackground(Color.black);

		title = new JLabel("Pick your Color");
		title.setForeground(Color.gray);

		brown = new JButton("BROWN");
		green = new JButton("GREEN");
		blue = new JButton("BLUE");
		chooser = new JButton("PICK");
```

GamePanel.java (Continued):

```
		brown.setPreferredSize(dim);
		green.setPreferredSize(dim);
		blue.setPreferredSize(dim);
		chooser.setPreferredSize(dim);

		brown.addActionListener(listener);
		green.addActionListener(listener);
		blue.addActionListener(listener);
		chooser.addActionListener(listener);

		controlPanel.add(title);
		controlPanel.add(brown);
		controlPanel.add(green);
		controlPanel.add(blue);
		controlPanel.add(chooser);

		this.add(hexGridPanel);
		this.add(controlPanel);
	}

	// ButtonListener
	private class MyListener implements ActionListener
	{
		public void actionPerformed(ActionEvent e)
		{
			if (e.getSource() == chooser)
			{
				Color shade = JColorChooser.showDialog(new JFrame(),

		"Pick a Color!",

		hexGridPanel.getColor());
					hexGridPanel.setColor(shade);
			}

			if (e.getSource() == brown)
			{
				hexGridPanel.setColor(new Color(102,51,0));
			}

			if (e.getSource() == green)
			{
				hexGridPanel.setColor(new Color(0,75,0));
			}
```

GamePanel.java (Continued):

```
                        if (e.getSource() == blue)
                        {
                                hexGridPanel.setColor(Color.cyan);
                        }
                }
        }
}
```

//HexGrid.java
//Evan Cummings
//11.10.9

```
import javax.swing.JFrame;
import java.awt.Toolkit;
import java.awt.Dimension;

public class HexGrid extends JFrame
{
        public static void main(String[]args)
        {
                Toolkit toolkit =  Toolkit.getDefaultToolkit ();
    Dimension dim = toolkit.getScreenSize();

                JFrame frame = new JFrame("Hex Grid");
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.getContentPane().add(new GamePanel());
                frame.setLocation((dim.width-640)/2, (dim.height-480)/2);
                frame.setResizable(true);
                frame.setVisible(true);
                frame.pack();
        }
}
```

Appendix B: Sources

Butler, Mark. "Web Wargaming." Webwargaming. 1998
<http://www.webwargaming.org/mapgrid/default.htm>.

Jahn, Thomas. "Coordinates in Hexagon-Based Tile Maps." GameDev. 28 Feb. 2002
<http://www.gamedev.net/reference/articles/article1800.asp>.

Java. Computer software. Sun Microsystems, 2009.

jGRASP. Computer software. Auburn University, 2009.

Lewis, John, and Loftus, William. Java Software Solutions. 6th ed. Addison Wesley, 2008

Patel, Amit. "Amit's Thoughts on Grids." CS-Students Stanford University. 9 Jan. 2006
<http://www-cs-students.stanford.edu/~amitp/game-programming/grids/>.