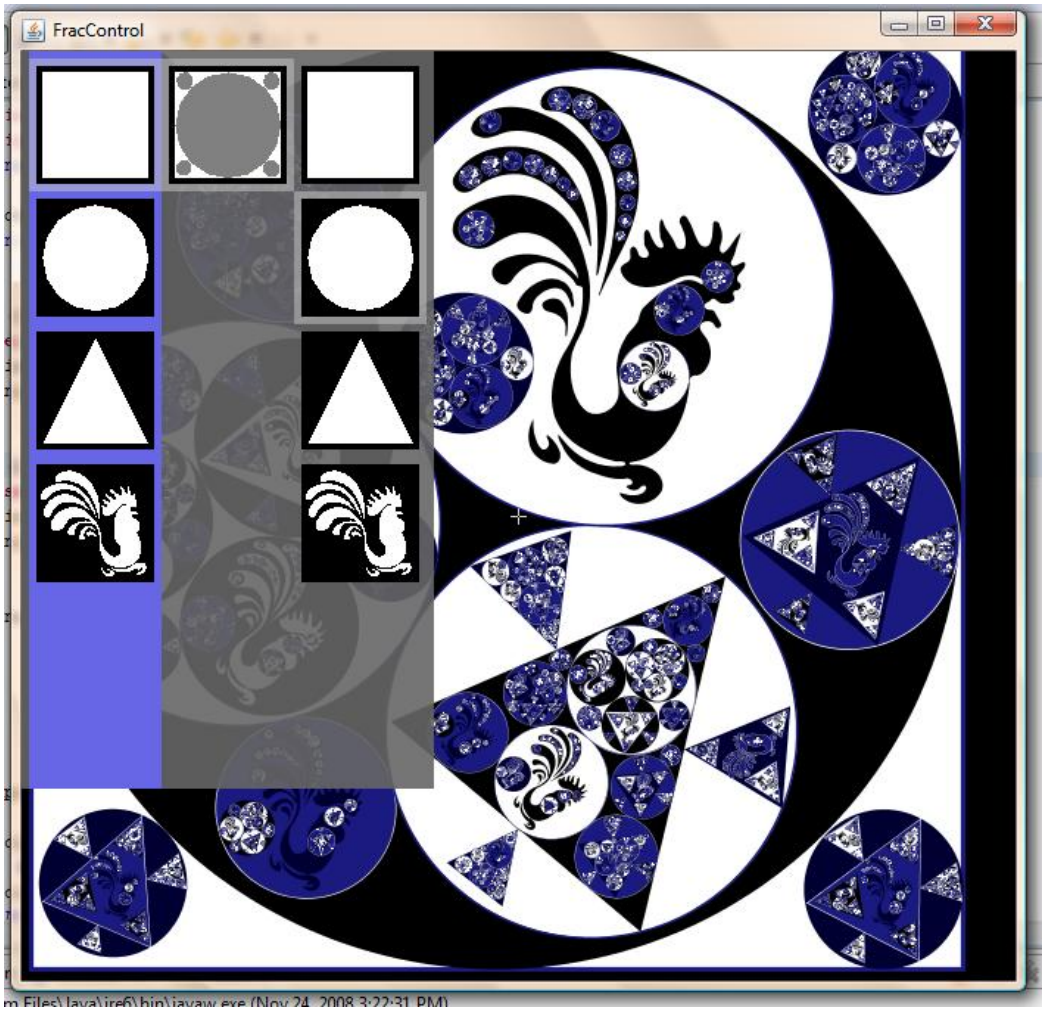
FracControl

An Easy Way To Build Fractals that Can Use an Xbox Controller

Michael Hewner
David Colvin



Code Overview

OverallFrame

The OverallFrame class contains the main method and is the entry point for the project. The central part of the project is the FractalComponent which is the content pane of the OverallFrame. The OverallFrame also adds DesignTemplates to the DesignTemplateLibrary and loads custom templates from the Scalable Vector Graphics (SVG) repository. It also creates the MouseControl and GameControl classes that will handle input. Finally it creates an instance of RuleMenu and adds it to the pallet layer of the Frame.

RuleMenu
The RuleMenu provides the user with a graphical menu for selecting the shapes to edit and draw with. The first column is the designTemplate (shape class) that is being edited. The middle column is all the designs created thus far of the selected designTemplate (the user starts out with a blank design of each designTemplate) and the selected one is the one being edited. The third column is the designTemplate that is selected to be drawn onto the design (creating a subDesign). In other words the first column is the canvas shape, the middle is the current canvas, and the last is the stencil to be drawn on the canvas. Pressing "I" on the keyboard or "X" on the controller creates a new design (or canvas). The Animation class handles the animation of hiding and showing the RuleMenu.

ArtistState
The ArtistState stores information on the state of the application, including the menu selections and the design being edited. The ArtistState also manages changes to the view such as zooming and panning. The ArtistState keeps track of subdesign previews before adding them to the design when the user places them. The ArtistState also implements the save function by serializing all of its state into a file. The state includes the view-state, designs, random seed, and menu selections.

MouseControl and GameControl
The MouseControl class is a mouse and keyboard listener which accepts all key presses and calls methods in the ArtistState class to perform actions from changing the menu to creating shapes. The mouseListener interface is used to differentiate between dragging, clicking and double-clicking. The GameControl class is a change listener for the Xbox 360 controller and also has timer driven polling to read in values from the analog sticks and triggers. The JXInput library was used to get change events from the controller. The controller calls the same methods in the ArtistState to edit designs and make menu selections as the MouseController.

Design, DesignBounds, and DesignTemplate,
These set of similarly named keep tracks of the fractal data. When users add sub-shapes to a larger shape they are modifying a Design. Each Design has an "outer" shape (square, circle, triangle, or chicken) - this is represented by the DesignTemplate. Each design also has several scaled, translated, or rotated non-overalapping inner shapes – these inner shapes are represented by DesignBounds.

The "Design" classes implement various aspects of designing new shapes, storing them, and drawing them. The DesignBounds class has methods for most of the preview tasks; the preview object is an instance of DesignBounds. The DesignTemplate class contains methods for computing the size of an object and drawing shapes. The Design class has methods for fitting new shapes in-between other shapses, adding and deleting subdesigns (the elements of a design), and performing decisions about color. The DesignTemplateLibrary stores all the Templates and Designs.

FractalPainter and DrawTask

The FractalPainter does the actual work of drawing the fractal. The FractalPainter has a painting thread that takes DrawTasks off a queue as it draws. When it draws one design the subdesigns of that design are added to the queue as DrawTasks. DrawTasks are only added if they are above a certain size threshold, which is scaled based on zoom level. The thread runs until all DrawTasks have been consumed. The FractalPainter also implements much of the coloring scheme for the fractals.

Instructions

The user can create a shape using the mouse or the game controller anywhere inside of the selected design. Using the left-mouse button the user selects the center point for the shape then drags to size and rotate the shape. While continuing to hold the left mouse button the user can also hold shift to move the entire shape (and center point) around the design. Using the game controller the user can move the crosshair with the right analog stick and place the center point by pressing "A". Then the shape can be sized and rotated using the left analog stick. The entire shape (and center point) can still be moved using the right analog stick. The user presses "A" again to place the shape. When not drawing a shape, the right stick still pans the view. The view can be zoomed using the "X" and "C" keys on the keyboard or the triggers on the controller.

Function	M&K	Contoller
New Shape	Hold Left Mouse Button	A
Place Shape	Release Left Mouse Button	A
Delete Shape	Double-click shape	B
Scale and Rotate Shape	Drag Mouse	Left Analog Stick
Move Shape	Shift-Drag Mouse	Right Analog Stick
Zoom In	z	Right Trigger
Zoom Out	x	Left Trigger
Pan Right	l	Right Analog Stick
Pan Left	h	Right Analog Stick
Pan Up	k	Right Analog Stick
Pan Down	j	Right Analog Stick
Zoom and Pan Original	v	Right Front Buton
Toggle Menu	q	Y
Menu Up	a	Directional Pad
Menu Down	z	Directional Pad
Menu Right	p	Directional Pad
Menu Left	o	Directional Pad
Save Work	U	Back Button
Load Saved Work	Y	Start Button
New Design	I	X
New Seed	N	Left Front Button

When the user places a designTemplate on to a design, a "random" design is picked to fill that template from all the designs that have already been created for that particular designTemplate. In other words when creating a new canvas (the middle column) of a particular canvas shape, the stencil that the user places is filled in with canvases of that stencil shape class. For example if the user is creating a circle design and places another circle inside it, then that circle can be filled with the very design they are creating.

Once the user has built several designs then the design can become complicated with multiple self referential shapes inside each other. Every design has at most one user specified level of shapes inside it, everything below that level is filled in automatically. The user can press the "LeftButton" or "N" on the keyboard to generate a new random seed that is used for the fill-in decisions. The coloring is based on the relative size of a shape compared to other shapes in its design and the color of its parent. For example, in a new design there are 2 circles; the larger will be black and the smaller dark blue; the background is always white. Once placed if that design is the larger shape in its parent design, and its parent design is white then it will have a black background and its inner shapes will become white and light blue. This color decision algorithm guarantees contrast between layers and gives variety to the design, using complementary colors.

Building & External Libraries

Building

This project was built using Eclipse, but it should be pretty easy to compile and run with a regular java environment. I compiled it on the command line using Java 1.6.0 in unix:

```
javac -cp batik-awt-util.jar:batik-
dom.jar:jxinput.jar:svgSalamander.jar:. OverallFrame.java

java -cp batik-awt-util.jar:batik-
dom.jar:jxinput.jar:svgSalamander.jar:. OverallFrame
```

In windows:

```
javac -cp "*" -sourcepath C:\temp\FracControl OverallFrame.java

java -cp C:\temp\FracControl;"*" OverallFrame
```

External Libraries

Our code makes use of the following libraries:

JXInput http://www.hardcode.de/jxinput/	Provides events for Xbox controller
Batik http://xmlgraphics.apache.org/batik/	Provides Polygon2D object that forms triangle shape
svgSalamander https://svgsalamander.dev.java.net/	Provides code to convert svg paths into Java Shape objects, which we use to get the rooster shape