Computing Project 1 Project 1 Report

Spray tool

My first addition was the spray tool, this tool will place a random arrangement of dots within a certain area surrounding your mouse and will continue to do so until you release the mouse button. This mimics the unpredictable spray of a spray can and is a fitting addition to our application. To begin implementing this feature, I create two separate for loops and will loop through each of the 2 dimensional spaces under our zone of coverage. It then will use the spread ratio to determine how often to place a point within those sections.

This will keep repeating while the mouse is pressed until it is released. I then implemented sliders for the density and area of effect. To do this I used html DOM manipulation and various user input elements then manipulated them within the tool when called upon. By saving the html objects into javascript variables, we have effectively given full access to DOM manipulation from the backend, I then used these user inputted values to determine the shape and spread.

I initially tried to draw several line objects from origin to current position. This however would be a much more complicated feature than anticipated as I would first have to calculate the position of each pixel when sprayed initially, then I would have to connect that pixel to its corresponding pixel the next loop. This means that each pixel would have to be its own object with properties that are updated and translated at each loop which is excessive and would cause undue performance issues.

Rectangle tool

The rectangle tool was a particular headache to sort out as I had to fully understand the Updatepixels and Loadpixels functions for p5. From my experiments; update pixels deletes the previous changes to the canvas and updates to the next drawloop canvas state. a canvas state will only be preserved if LoadPixels is called, it uploads the changes, then creates a new state for update pixels to work with. This means that Updatepixels effects all drawings from when load pixels is called, until the current frame where this line of code is called, then Loadpixels takes the current frame and makes that the new start for update pixels. Loadpixels and update pixels let you specify what is temporary and what is permanent basically and this how I created the rectangle tool.

By temporarily drawing a rectangle from the origin to the mouse each frame the mouse is held down, by never actually using load pixels, we can get this effect. Then when the mouse is released, we call load pixels and permanently apply wherever we were holding our rectangle object.

Toolbox

The hardest part of this project was figuring out the utility of the toolbox.js file and how to use its member functions and properties effectively, as well as understand its prewritten functions. The SelectTool() function within the toolbox is truly what allowed me to add my rectangle tool to the project. The tool must have a PopulateOptionsFunction() which is then called upon by the toolbox to create the sliders and optional colours etc.

These are all instantiated within sketch when called upon, using the “this.” keyword creates an attribute in the object, thus making it globally accessible to the rest of the object. Remember that “this .” is not live use and thus visibility is only constructed when its parent function is instantiated. This allows for your functions and objects to become modular and create different instances of the same function.

Think of the “this.” Keyword simply as a placeholder for whatever the overall function gets called upon as.

Structure

Structuring code is important for a number of reasons – namely; readability, reusability and control flow. I have placed my global variables at the top for each tool.js page and have moved functions to the bottom of the page for better readability. This is assisted by my notes to allow both myself and peers to understand and improve on previous work.

By separating each tool from the toolbox, the toolbox from everything else and having them in separate files allows for the separation of concerns. This concept predicates that using separate pages and environments for each separate feature allows for firstly; an easier visual representation of the aspects within our project, making it easier to understand and read for a fellow coder to interpret. Secondly, it makes trouble shooting much easier as each file is separated by its overall function, this then means that when an error does in fact occur, I can easily find what has caused the issue as they are not interconnected and do not have critical interdependencies that would break my program when altered.

Evaluation

To evaluate any project, I suppose you must look at the brief and compare your work with each objective. In that respect, application stability is good and there are some very tiny bugs, but stability is high. Usability is also logical and intuitive using icons and prompts etc. I perhaps could have added some more modifications and additions but overall, I am happy with how my work has come out and I believe I have effectively met the requirements of our initial brief and my app is an effective and enjoyable drawing app that needs only a few additions and tidying up before being fully user ready.

Progress Log

18/07	19/07	20/07	21/07	22/07	23/07	24/07
Started project	Working on rectangle tool	Trying to understand the toolbox	Finally added the rectangle tool	Added Spray tool	Added spread and density sliders	Finished project