HTML 5

Canvas tag – reference <http://www.w3schools.com/tags/ref_canvas.asp>

The Canvas tag in html5 contains so many methods that will allow someone to draw on it through a JavaScript implementation, some of the most vital methods being:

fillStyle – sets or returns the color, gradient, or pattern used to fill the drawing

StrokeStyle – sets or returns the color, gradient, or pattern used for strokes

lineWidth – Sets por return the width of the line

fill – fills the current drawing or path

stroke – Draws the path that has been defined

beginPath – begins a path or resets the current path

closePath –creates a path from the current point back to the starting point, closes path

moveTo – moves the current path to the specified path without creating a line

lineTo – adds a new point nad creates a line from that point to the last specified point

quadraticCurveTo – creates a quadratic Bezier curve (curved line)

strokeText – draws text onto canvas

drawImage – draw an image to canvas

The canvas tool can have a transparent background, potentially allowing the possibility of layers

Layers

Layers can be approached in one of two ways

Create several canvas tags on top of each other and maintaining them

Use a single canvas tag and add layers onto it

Layers can be added to the canvas through a found library, allowing ease of layer creation, a potential flaw to the current implementation of this is that the original canvas does not seem to be able to considered as a layer, therefore not allowing it’s order to be changed.

A history tool could possibly be made by saving the canvas and adding the version to a collection of canvases, allowing the potential of an undo/redo feature, along with a history tool to show the user a set amounts of past image states

History Tool

A history tool saves one of two states

Save state – saves the state of the data into a collection, the current state would be reverted to a previous state when a history tool is used

Generate State – in a generate state, the action taken would be saved into a collection, an undo would require a logical reversal of the action, such as removing the tools effect on the data

Utilizing save states requires memory and compression, but allows a rather easy way to undo and redo actions as all that is required is changing the current state to a previous state. The problem lies in the amount of states saved, too many and the list would require a lot of memory to store so many states. Saving each state could require some time to complete, along with compressing the state as to not overload the history tool collection with too much data

Utilizing generate states does not require much memory to store each action. Undoing an action logically as to remove the action completely is the difficult part. The action must be reversed thoroughly to ensure a complete undo, this can be difficult to achieve and more difficult actions

Paint Bucket tool might be achievable by iterating through every pixel on the canvas from where the mouse click took place, looking for a boundary (any colored pixel) proceeding to color the area with the specified color, very inefficient, but a possibility NEEDS TESTING

Javascript

By using the current methods provided with the canvas tag, implementing mouse event listeners with JavaScript to grab current mouse positions allows the possibility to create a drawing

Have not thought of a way to utilize JavaScript in any other way other than reading mouse events and performing some logic