School of Computer Science University of Lincoln 2018/19

CMP1130M Web Authoring

Lab (8)

Brief

The aim of this workshop is to get familiar with the JavaScript <canvas>, and using mouse events to track curser positions relative to the canvas and draw on the context(2d). The point of this workshop is to get you up to speed with event listeners. There are heavily commented support files which does not give all the code but gets you to consider the logic and points you in the right direction.

By the end of this workshop, you should be able to:

1. Add a remote Javascript and CSS files.
2. Add multiple mouse event listeners to call a function.
3. Return event information from the the function(e) to capture mouse co-ordinates
4. Convert co-ordinates from global to local, relative to the canvas element.
5. Use moveTo and lineTo to draw to the context 2d
6. Draw and image to the context 2d
7. Use html5 input to manipulate colour and line width.

Hints have been added as place holder for required scripts. Its up to you to add the logic, functions and listeners.

Task 1

1. Review the Lecture slides.
2. Download sample.
3. Please look up event bubbling, event target and propagation. (http://javascript.info/tutorial/bubbling-and-capturing, http://eloquentjavascript.net/14_event.html)

Task 2

1. Create a blank HTML document and link an external Java script and CSS file to that document.
2. Add a canvas element with an id=”canvas”
3. Console.log a “ready” string using a **window.onload** function**.**
4. Make sure your body style has no padding or margins
5. When the page is loaded make sure the canvas is full screen

```
function Init() {
  canvas = document.getElementById("canvas");
```

```
  canvas.width = window.innerWidth;
  canvas.height = window.innerHeight;
}
```

Task 3

Add a “click: event listener on the canvas element and

1. Log the click is working
2. Log the event data
3. Get the x and y position of the mouse curser on click
   note: e.pageX, e.pageY, this.offsetLeft, this.offsetTop
   // note the inline Anonymous function so we we can get the event data

```
canvas.addEventListener("click", function(e) {
var x = e.pageX - this.offsetLeft;
var y = ?;
console.log('x= ', x, 'y= ', y);
});
```

4. Add “mousedown”, “mousemove”, “mouseup”, “mouseleave” listeners
   Make sure they are all working a console.logging data before moving on. Below shuod get you started,

```
canvas.addEventListener("mouseleave", (
        function(e) {
         mousePressed = false;
         })
);
```

Task 4

1. Create a draw function that passes x, y and boolean parameters
2. Call that function in your mouse events

```
//draw function the third argument is  a boolean
Draw(x mouse position, y mouse position, isdown);
```

Don’t expect any results yet, we ate just setting up and checking things are doing what they should be via the console. We still need to add drawing to the canvas, which comes later. First we need to add html elements to interact with.

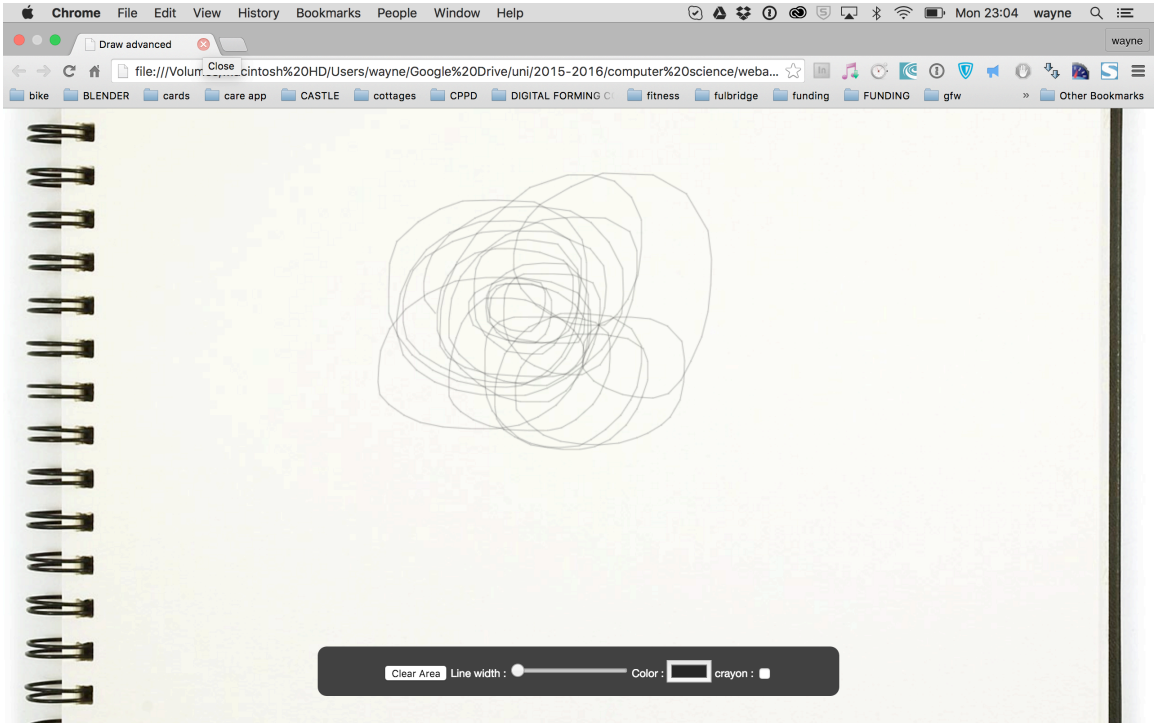
Task 4

1. Create a containing and give it a class name so we can position it later
2. Add 1 input fields for size (look up range input) give it an id of line_size
3. Add 1 input fields for colour (look up color input) give it an id of line_colour
4. Add 1 check box give it an id of texture

5. Add “change” event listeners and use the console.log if the checkbox is checked or not. (in the code provided)
6. texture.addEventListener("change", function(e) {
   // Is the checkbox checked
   });

7. Style the div to float on top of the page - aligned centre, bottom. (Make it neat and tidy)
8. Add background image proportionally to the page as a background

Task 5



Draw to the canvas on mouse press and mouse move. Enable a button that clears the canvas using the draw function you created earlier.
Note: Only draw when the mouse is down. (Pass a Boolean parameter in your function)

1. Pass x, y and a Boolean through to your draw function from the mouse events created earlier.
2. Add a condition to check if the checkbox is checked so we can determine whether we draw a line or an image. Image info is next.
3. If it is not checked draw a line to the x and y coordinates passed through the function
4. Look up the line to and move to methods for canvas other you will need are:
   .beginPath();
   .strokeStyle
   .fillStyle
   .lineWidth

```
.lineJoin ="round";
.moveTo();
.lineTo(x, y);
.closePath();
.stroke();
```

Task 6

Us the check box to draw a texture instead of a line in your GUI. Look up image.loaded to make sure the image is loaded before being used. Add a Boolean to use an image or draw a stroke on thr canvas.

```
var crayonTextureImage = new Image();
crayonTextureImage.src = "crayon-texture.png";

my_context.globalAlpha = 0.8; // No IE support
my_context.drawImage(crayonTextureImage, x, y, 10, 10);
```

Further work 1

Add an eraser and utilise blend modes to improve the look?, i.e. multiply.

Can you make the line look 3d, or bevelled.

Further work 2

Create an interface that allows multiple background textures with multiple types of drawing effects i.e. pen, marker, spray paint etc.

Further work 3

Using media queries and touch events make the drawing board is functional with touch devices.
How can you use JS to dynamically switch between touch events and mouse events? Do you even have to?