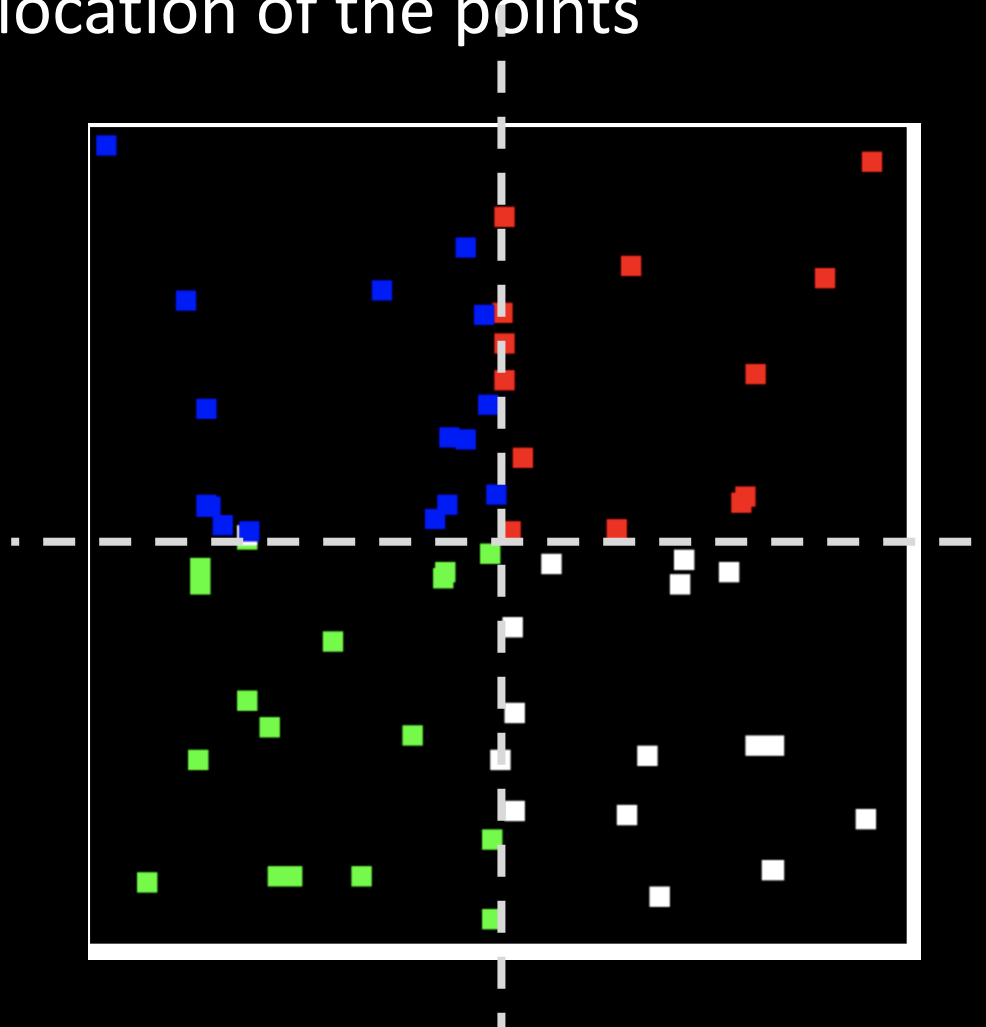
Lab 1

What You will Learn

- Some javascript basic syntax
- Array
- Mouse event
- Convert mouse event position from browser space to WebGL drawing canvas space
- Pass data to shaders

What We Want

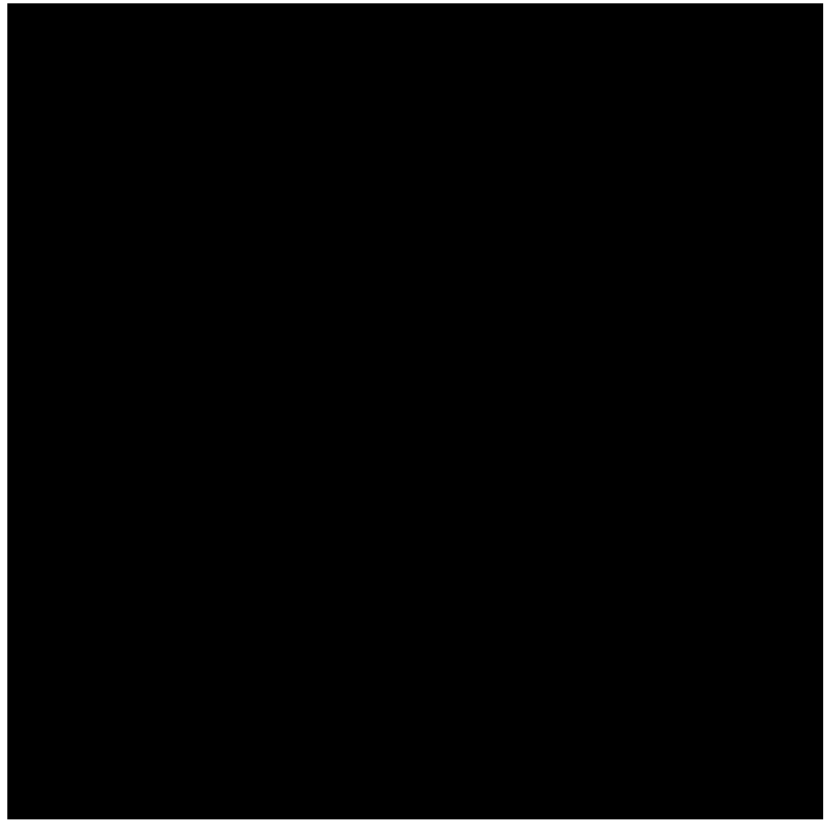
- Virtually divide the canvas into 4 quadrants
- When user clicks on the canvas, add one more point
- The color of the point is determined by the location of the points
  - Top right quadrant: red
  - Top left quadrant: blue
  - Bottom left quadrant : green
  - Bottom right quadrant :white



What You Should Do Step by Step

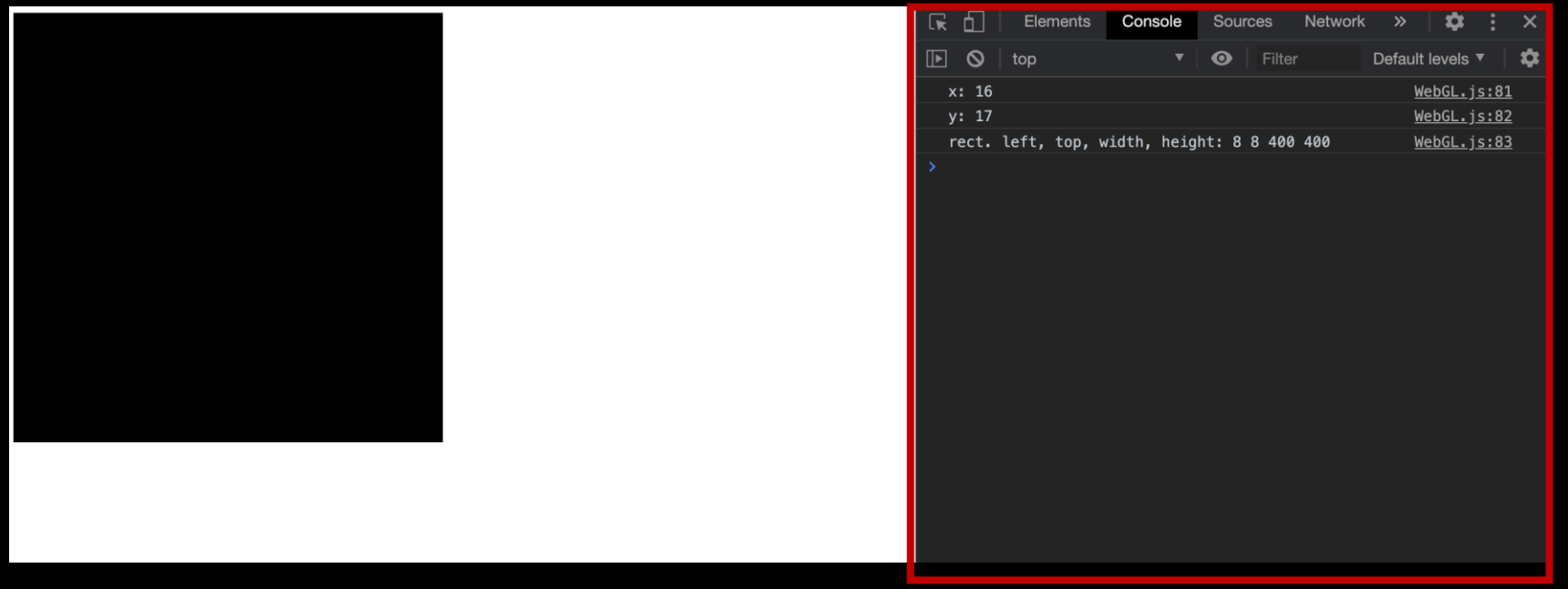
1. Download and Run

- Download the zip file which contains “index.html” and “WebGL.js” from Moodle (unzip the zip file)
- Drag “index.html” to a browser (e.g. Chrome)
  - You suppose to see a black canvas



2. Coordinate System of Mouse Event

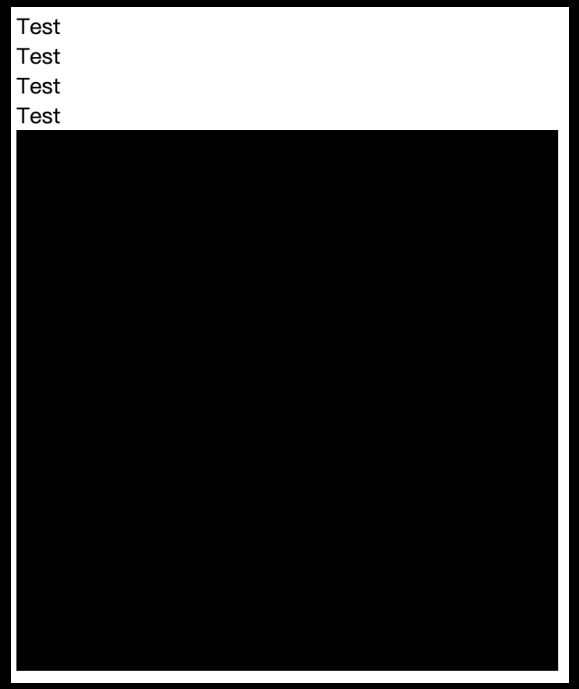
- Read main() function in WebGL.js
- Read the function “click()” in WebGL.js
- Open the developer tool



2. Coordinate System of Mouse Event

- Click at the left-top corner of the canvas and check what text is printed in the developer tool
- Change the position of the canvas by add multiple ”test
” in index.html
- Click at the left-top corner of the black “canvas” again and check what text is printed in the developer tool

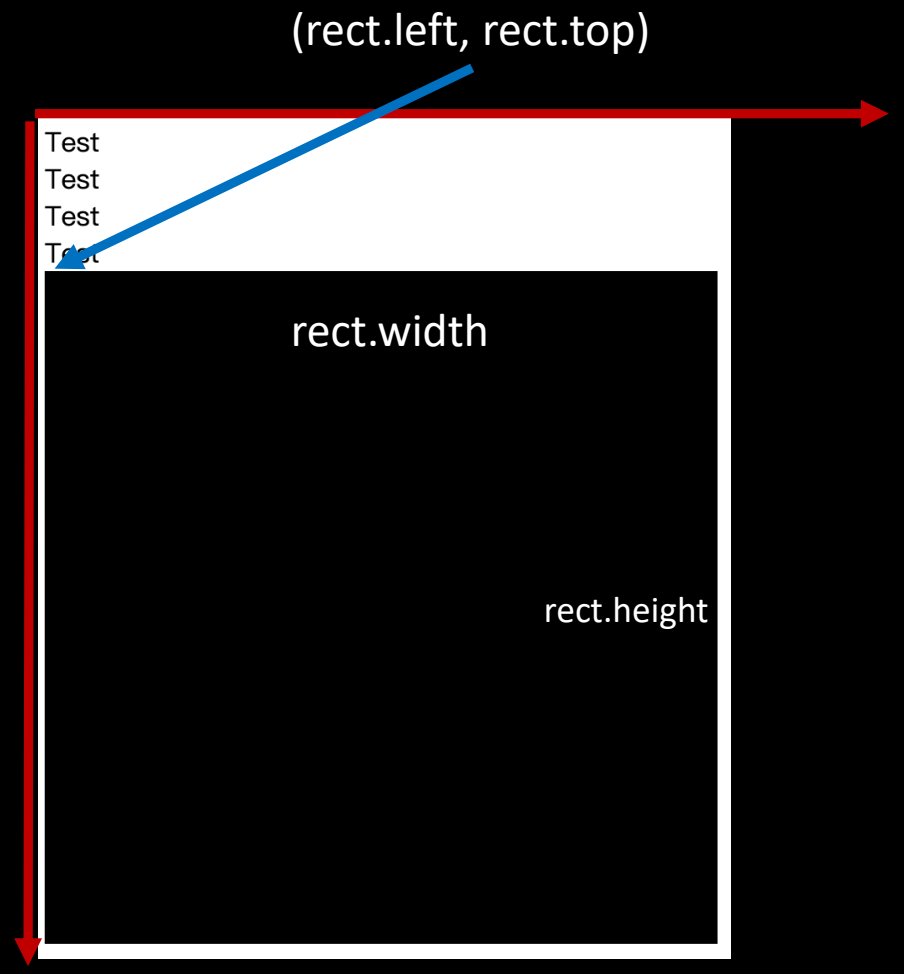
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>WebPage Title </title>
    </head>

    <body onload="main()">
        Test<br>
        Test<br>
        Test<br>
        Test<br>
        <canvas id="webgl" width = "400" height = "400">
            Please use a browser that support "canvas"
        </canvas>
        <script src="WebGL.js"></script>
    </body>
</html>
```



2. Coordinate System of Mouse Event

- ev.clickX and ev.clickY return the coordinate according to the ”red” coordinate system
- ev.target.getBoundingClientRect() returns information of the canvas (because you click on it)

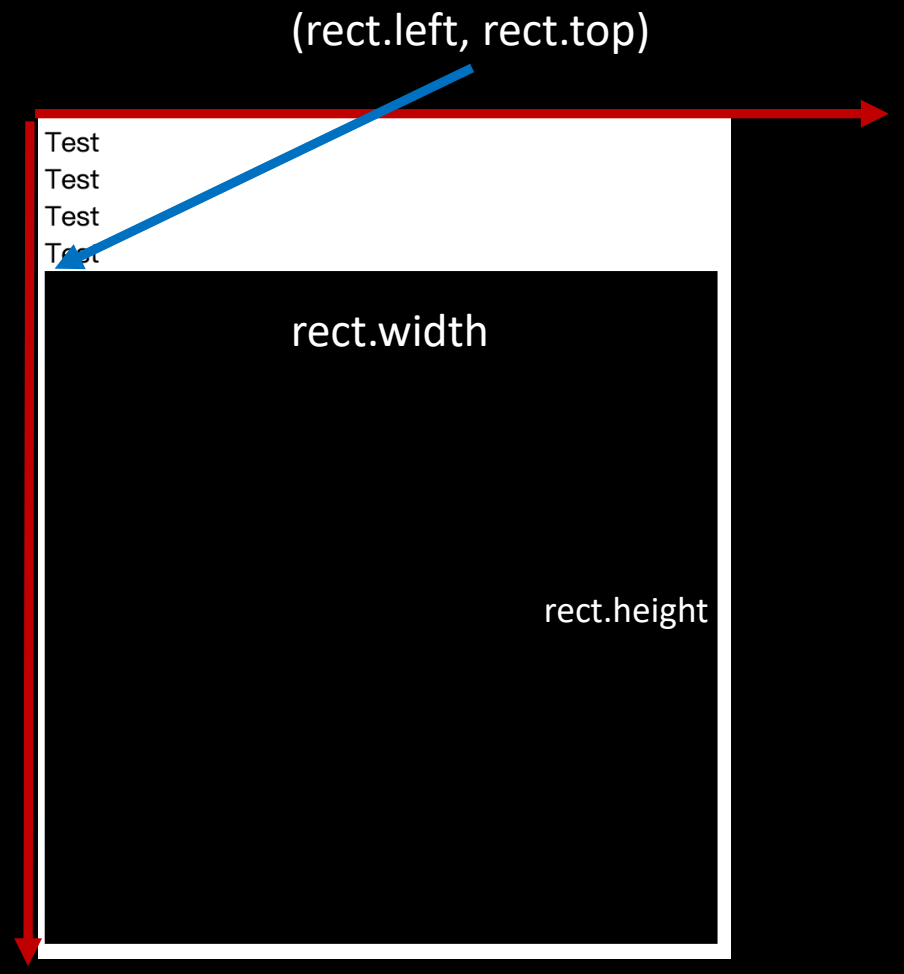
- Check here for more information
  - https://developer.mozilla.org/en-US/docs/Web/API/Element/getBoundingClientRect



3. TODO-1

- When we draw in WebGL
  - The span of x-axis is -1 to +1
  - The span of y-axis is -1 to +1

- Complete TODO-1 for this conversion



4. TODO-2

- Fill out “???”
- Calculate color for each point

5. TODO-3

- Read the for loop
- Fill out “???” to pass data to shader for drawing

What You Should Do for “Submission”

Submission Instruction

- Create a folder
  - Put the html and js files in the folder
  - Zip the folder
  - Rename the zip file to your student ID
    - For example, if your student ID is “40312345s”, rename the zip file to “40312345s.zip”
  - Submit the renamed zip file to Moodle

- Make sure
  - you put all files in the folder to zip
  - You submit the zip file with correct name
- You won’t get any point if
  - the submitted file does not follow the naming rule,
  - TA cannot run your code,
  - or cannot unzip your zip file.