Geometry with the HTML5 Canvas

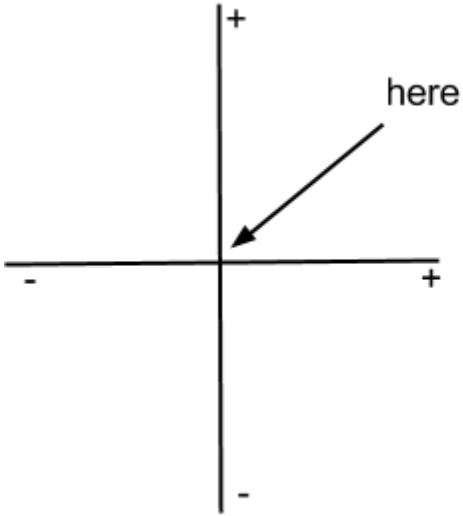
by Jim Town and Elaine Blomeyer

HTML5 Canvas Basics

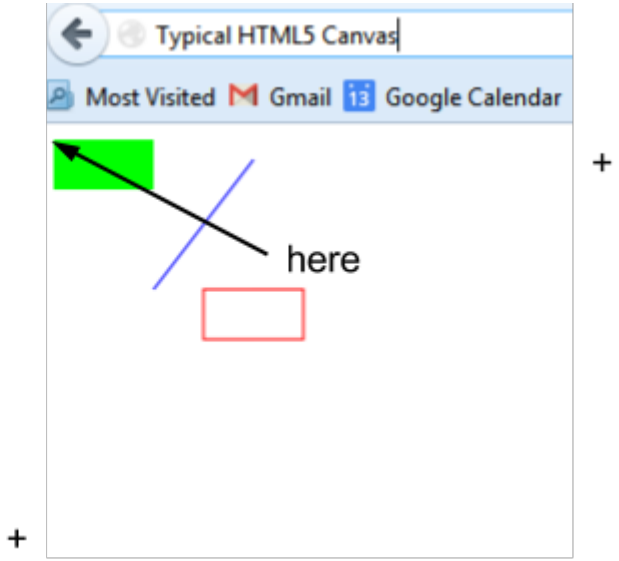
To start with go to http://www.missblomeyer.com/todos2014/
and right click on template to save file.

The origin

You might be familiar with Cartesian Coordinates where the origin (0,0) is



But on a typical HTML5 Canvas the origin (0,0) is



Note: there are no negatives on the HTML5 canvas

Caution: though x is the same (right=positive), y is opposite (down=positive)

Copy the template

1. Open a text editor (eg notepad++).
2. Open the template.html file you downloaded.
3. Save as basic_example.html

4. Fill in your template

The upper code you are writing creates the canvas and clears it.

The lower code you are writing draws two rectangles and a square.

```
<!DOCTYPE html>
<html>
<head>
    <title>Optical Illusion</title>
    <script>
        var ctx;
        var canvas;
        function init() {
            canvas = document.getElementById("mycan");
            ctx = canvas.getContext("2d"); // cxt short for context
            ctx.clearRect(0,0,canvas.width,canvas.height); //clear the canvas
            draw();
        }

        function draw() {
        //begin green rectangle
            ctx.fillStyle = '#00ff00'; //neon green
            ctx.fillRect(0,0,50,25); //make a solid rectangle
        //end green rectangle
        //begin red rectangle
            ctx.strokeStyle = '#ff0000';//super red
            ctx.strokeRect(75,75,50,25); //make an outlined rectangle
        //end red rectangle
        //begin blue square
            ctx.strokeStyle = '#000011'; //dark blue
            ctx.fillStyle='#0000ff';// light blue
            ctx.fillRect(50,40,20,20); //make a solid square
            ctx.strokeRect(50,40,20,20); //make an outlined square
        //end blue line
        }
    </script>
</head>
<body onload="init()">
    <canvas width="300" height="300" id="mycan">No Canvas Support</canvas>
</body>
</html>
```

A closer look: top section

```
function init() {
    canvas = document.getElementById("mycan");
    ctx = canvas.getContext("2d"); // cxt short for context
    ctx.clearRect(0,0,canvas.width,canvas.height); //clear the canvas
    draw();
}
```

Does your code match? Good, it is time to move on.

A closer look: bottom section

```
function draw() {
//begin green rectangle
    ctx.fillStyle = '#00ff00'; //neon green
    ctx.fillRect(0,0,50,25); //make a solid rectangle
//end green rectangle
//begin red rectangle
    ctx.strokeStyle = '#ff0000';//super red
    ctx.strokeRect(75,75,50,25); //make an outlined rectangle
//end red rectangle
//begin blue square
    ctx.strokeStyle = '#000099'; //dark blue
    ctx.lineWidth=2;//thicker stroke line
    ctx.fillStyle='#0000ff';// light blue
    ctx.fillRect(50,40,20,20); //make a solid square
    ctx.strokeRect(50,40,20,20); //make an outlined square
//end blue square
}
```

Check your code

4. Minimize your text editor then open your new file in a browser (eg Chrome).
5. Tada! You should see something like this:

Looking back:

What is the difference between fill and stroke?

Colors are stored as 6 digit hexadecimal ‘numbers’ in RGB (RedGreenBlue) format.
Common colors:
Black ‘#000000’ (no color)
White ‘#ffffff’ (all color)
Grey ‘#cccccc’ (in between)

What do you think yellow would be?

Try it out, were you right?

Teacher hat:

Why did we make you type in the code?

Where will your students struggle? How can you scaffold them to ensure success?

How much geometry have they learned so far?

Any questions so far?

Next, animation

Open up the template again and save as stepping_feet.html

Type the code into the template to match ours at the right.

What are the key differences between this animation code and the previous code?

```
<!DOCTYPE html>
<html>
<head>
<title>Optical Illusion - Stepping Feet</title>
<meta charset="UTF-8">
<script type="text/javascript">
    var canvas;
    var ctx;
    var x;
    function init() {
        canvas = document.getElementById("optical");
        ctx = canvas.getContext("2d");
        x = 0;
        setInterval(draw,10);
    }

    function draw() {
        ctx.clearRect(0,0,canvas.width,canvas.height);
        ctx.fillStyle="#FFFF00"; //yellow rectangle
        ctx.fillRect(x, 50, 50, 20);
        //put blue rectangle here
        //put bars here
        x++;
        if(x > canvas.width)
            x = 0;
    }

</script>
</head>
<body onload="init()">
<canvas id="optical" width="1000" height="600">No support for canvas</canvas>
</body>
</html>
```

A closer look:

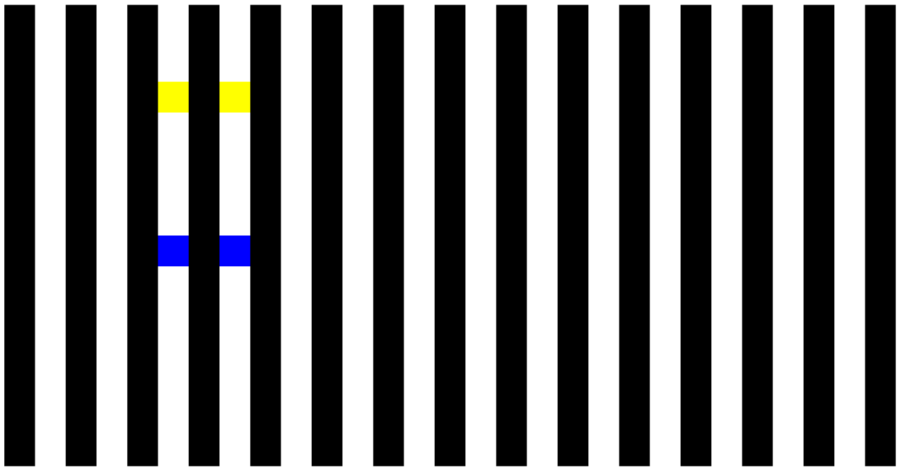
```
var canvas;
var ctx;
var x;
function init() {
    canvas = document.getElementById("optical");
    ctx = canvas.getContext("2d");
    x = 0;
    setInterval(draw,10);
}

function draw() {
    ctx.clearRect(0,0,canvas.width,canvas.height);
    ctx.fillStyle="#FFFF00"; //yellow rectangle
    ctx.fillRect(x, 50, 50, 20);
    //put blue rectangle here
    //put bars here
    x++;
    if(x > canvas.width)
        x = 0;
}
```

First challenge:

1. Add a second foot to the illusion.

2. Add the bars to the background.



Are the yellow and blue bars in sync?

Drawing lines:

1. Open the template and save as arrows.html
2. Type the code into the template to match ours at the right.
3. Looking at the drawing and the code, how is moveTo different from lineTo?

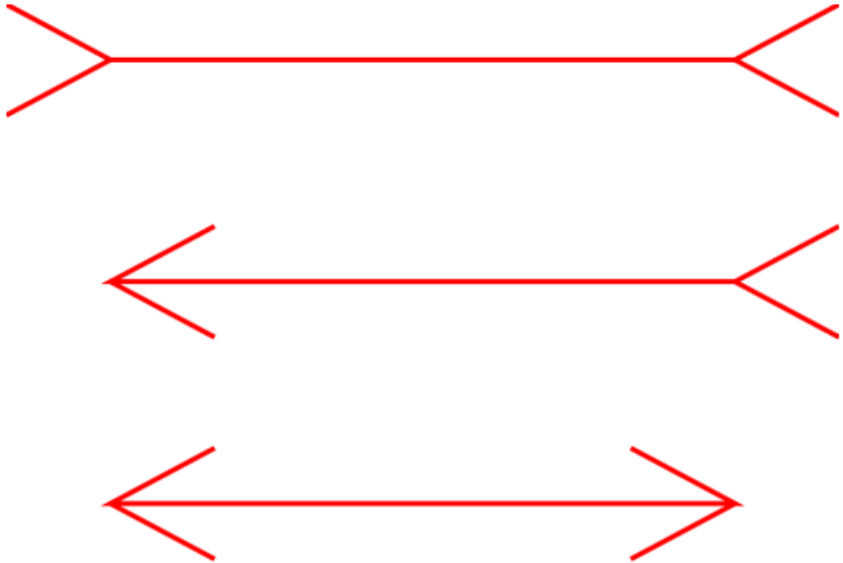
```
<!DOCTYPE html>
<html>
<head>
<title>Which line is the longest?</title>
<meta charset="UTF-8">
<script type="text/javascript">
    var canvas;
    var ctx;
    function init() {
        canvas = document.getElementById("optical");
        ctx = canvas.getContext("2d");
        ctx.clearRect(0,0,canvas.width,canvas.height);
        draw();
    }
    function draw() {
        ctx.beginPath();
    //put top arrow here
    //begin middle arrow
        ctx.moveTo(250,200);
        ctx.lineTo(200,250);
        ctx.lineTo(500,250);
        ctx.lineTo(550,200);
        ctx.moveTo(200,250);
        ctx.lineTo(250,300);
        ctx.moveTo(500,250);
        ctx.lineTo(550,300);
    //end middle arrow
    //put bottom arrow here
        ctx.stroke();
    }
</script>
</head>
<body onload="init()">
<canvas id="optical" width="700" height="500">No support for canvas</canvas>
</body>
</html>
```

A closer look:

```
var canvas;
var ctx;
function init() {
    canvas = document.getElementById("optical");
    ctx = canvas.getContext("2d");
    ctx.clearRect(0,0,canvas.width,canvas.height);
    draw();
}
function draw() {
    ctx.beginPath();
//put top arrow here
//begin middle arrow
    ctx.moveTo(250,200);
    ctx.lineTo(200,250);
    ctx.lineTo(500,250);
    ctx.lineTo(550,200);
    ctx.moveTo(200,250);
    ctx.lineTo(250,300);
    ctx.moveTo(500,250);
    ctx.lineTo(550,300);
//end middle arrow
//put bottom arrow here
    ctx.stroke();
}
```

Second Challenge

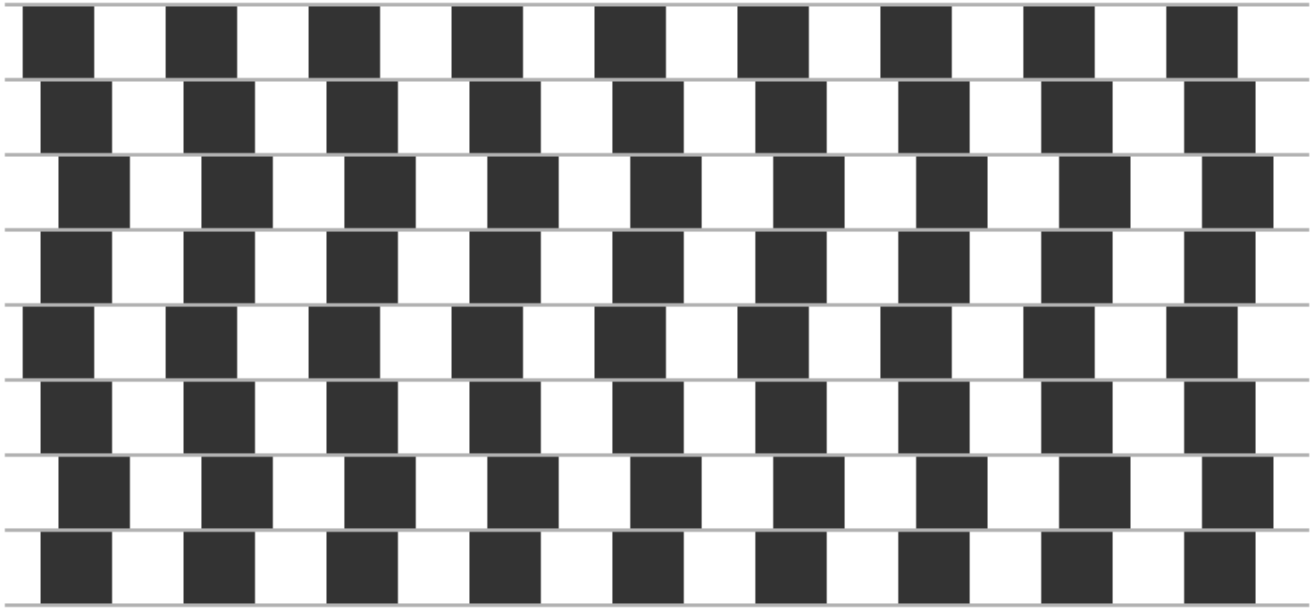
Add the missing lines to complete the illusion.



Which line is the longest, or are they equal length?

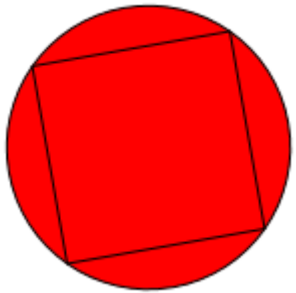
Bonus challenge

Draw this optical illusion using the HTML5 Canvas



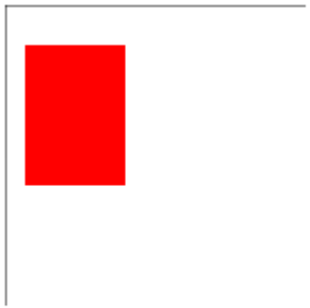
Are the lines parallel?

Translating and Rotating the Canvas

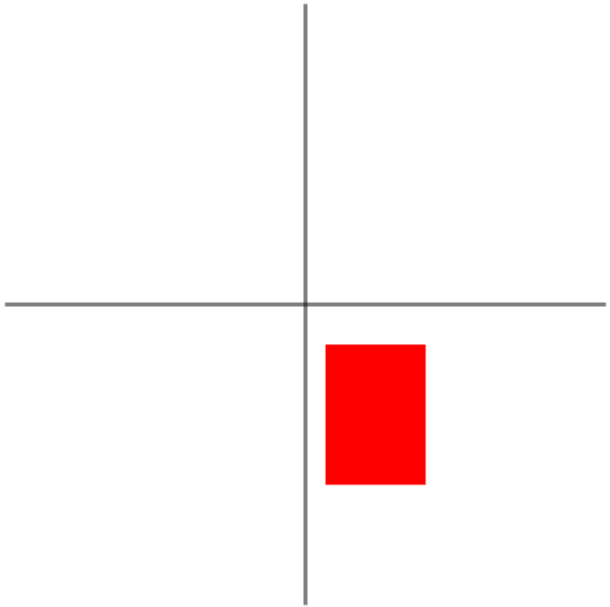


Translating canvas = moving origin

Regular canvas

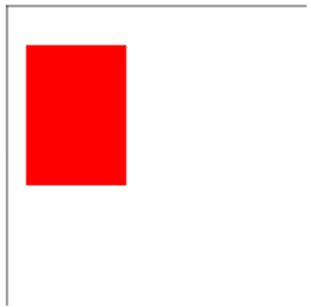


Translated

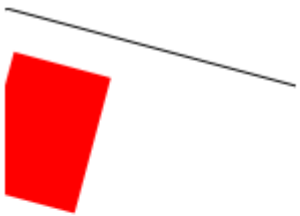


Rotating canvas = rotating axes

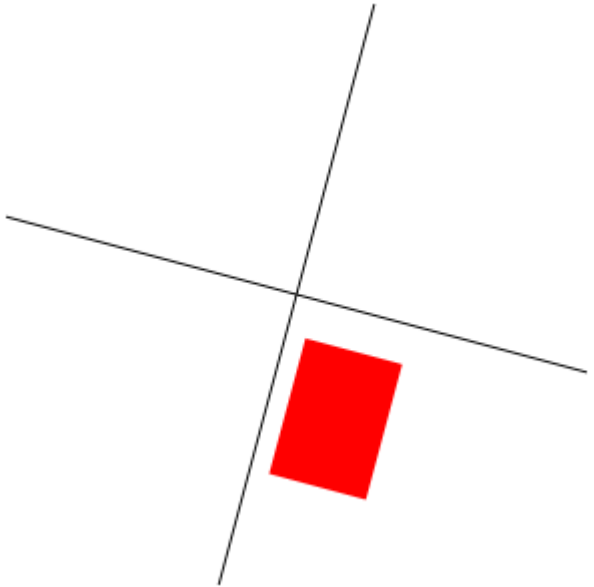
Regular canvas



Rotated 15º



Translated and rotated



Rotating square

```
<!DOCTYPE html>
<html>
<head>
<title>Rotating Square</title>
<meta charset="UTF-8">
<script type="text/javascript">
    var canvas;
    var ctx;

    function init() {
        canvas = document.getElementById("optical");
        ctx = canvas.getContext("2d");
        ctx.fillStyle =  "#FF0000";
        ctx.strokeStyle = "#000000";
        ctx.translate(canvas.width / 2, canvas.height / 2); //translates origin to middle
        ctx.save(); //saves current state of canvas
        setInterval(draw, 100);
    }
    function draw() {
        //ctx.restore(); //restores canvas to saved position (not needed here, but important)
        ctx.clearRect(-1 * canvas.width/2, -1 * canvas.height/2, canvas.width, canvas.height);
    //begin circle
        ctx.beginPath();
        ctx.arc(0,0,50*Math.sqrt(2),0,Math.PI*2, true);
        ctx.closePath();
        ctx.fill();
        ctx.stroke();
    //end circle
    //begin square
        ctx.fillRect(-50, -50, 100, 100);
        ctx.strokeRect(-50, -50, 100, 100);
    //end square
        ctx.rotate(Math.PI/36);  // rotates axes 5 degrees
    }
</script>
</head>
<body onload="init()">
<canvas id="optical" width="700" height="500">No support for canvas</canvas>
</body>
</html>
```

A closer look: init section

```
function init() {
    canvas = document.getElementById("optical");
    ctx = canvas.getContext("2d");
    ctx.fillStyle =  "#FF0000";
    ctx.strokeStyle = "#000000";
    //ctx.save(); //saves current state of canvas
    ctx.translate(canvas.width / 2, canvas.height / 2);
    setInterval(draw, 100);
}
```

A closer look: draw section

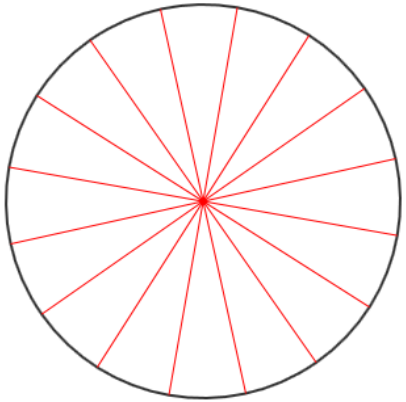
```
function draw() {
    //ctx.restore(); //restores canvas to saved position (not needed here, but important)
    ctx.clearRect(-1 * canvas.width/2, -1 * canvas.height/2, canvas.width, canvas.height);
//begin circle
    ctx.beginPath();
    ctx.arc(0,0,50*Math.sqrt(2),0,Math.PI*2, true);
    ctx.closePath();
    ctx.fill();
    ctx.stroke();
//end circle
//begin square
    ctx.fillRect(-50, -50, 100, 100);
    ctx.strokeRect(-50, -50, 100, 100);
//end square
    ctx.rotate(Math.PI/36);  // rotates axes 5 degrees
}
```

Challenge 3:

Choice:

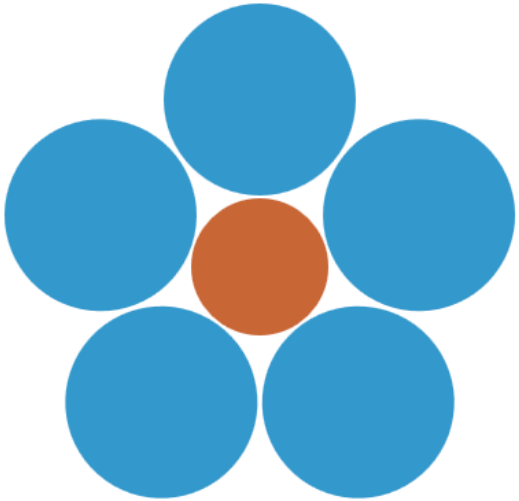
a) Create a rotating wagon wheel

b) Create the “Which circle is bigger” illusion.



Can you draw a rotating wheel?

Which brown circle is bigger?



Final Challenge

1. Read through the list of example tasks.
2. Choose one.
3. Find others (2 or 3 ideally) who chose the same task and complete it together.
4. Create a plan for teaching this lesson in your classroom.
5. Be ready to share your ideas with the group.

Example Geometry Canvas Tasks:

Beginning Tasks (part 1)
Geometry Task: Draw a rectangle: Draw a rectangle using *fillRect*, then draw the outline of the rectangle using the *beginPath, moveTo, lineTo, closePath, context.stroke* sequence.
Questions: What are the properties of rectangles? How can you draw a right angle in the canvas?

Geometry Task: Draw a house.
Questions: How did you figure out what size to make the shapes? How did you figure out the placement of each shape?

Geometry Task: Draw a face.
Questions: How did you figure out what size to make the shapes? How did you figure out the placement of each shape?

Illusion Task: “Which red line is longer?”
Questions: Does the black shape matter? Does it have to be a parallelogram? What happens if you make it a rectangle? trapezoid? Does the illusion still work?

Illusion Task: “White’s Illusion.”
Questions: What is the minimum distance the gray rectangles should be apart? is there a maximum?

Geometry Task: Circumcenter
Draw a scalene triangle and its circumscribed circle.
Questions: How did you figure out where the center of the circle would go? What about the radius?

Example Geometry Canvas Tasks:

Beginning Tasks (part 2)
Illusion Task: “Is the red shape a square?”
Question: How far apart should the radii of the concentric circles be?

Illusion Task: “Is the red shape a circle?”
Questions: What is the minimum degrees in between each spoke? Maximum?

Geometry Task: Animate a robot.
Question: What algebra concepts did you use when you made the robot move?

Geometry Task: Animate a rotating square.
Question: What is the radius of a circle that circumscribes the square?

Geometry Task: Rolling Wheel
Animate the rotating wagon wheel to move across the canvas.
Question: How far does it move each time as it rotates to mimic a real wheel?

Example Geometry Canvas Tasks:

Advanced Tasks

Geometry Task: Incenter
Draw a scalene triangle, then place a circle inside of it so that each side of the triangle is tangent to the circle at exactly one point.
Question: How did you figure out where the center of the circle would go? What about the radius?

Geometry Task: Scaling area
Draw a 1x2 rectangle, then animate it to double in size each cycle until it fills the whole 1000x2000 canvas.
Questions: How long will it take to fill the canvas? what if it triples every cycle? quadruples? n?

Illusion Task: “The Hermann Grid Illusion.”
Question: How did you find the center for each circle?

Illusion Task: “The Hering Illusion.”
Question: How did you figure out the slopes of the lines?

Illusion Task: “Lilac chaser.”
Question: How did you figure out how fast to make the animation? Would more circles work better? less?

Illusion Task: “Motion Induced Blindness.”
Questions: Is the six by six grid ideal for this illusion? What if it were eight by eight? three by three?

Reflection/discussion

How could you use this in your classroom?

What challenges will your students face?

What geometry are they learning?

How does this promote equity?

How will this help ELLs?

Bonus bonus challenge

What could you do if you wanted the HTML5 canvas to be a true cartesian coordinate system?