**Basic 2D Drawing**

**JavaScript & HTML 5**

Tuesday September 9, 2014.

---

**Why?**

- Key Concepts
  - Drawing Contexts
  - Paths / Strokes / Fills
  - Colors
  - 2D Transformations
  - Events
- Immediate Gratification and Hopefully Fun.
Start Small – One Little Line

FYI – This canvas is 300 by 300 pixels in size.

HTML / CSS / JavaScript

- Web programming very different
  - Get used to a blend of 3 languages
- HTML – never went away, start here.
- CSS – how it looks and what it is called
  - Often for ‘naming’ JavaScript elements.
- JavaScript
  - Powerful interactive client side language
  - Binds functionality to (HTML) objects
Example 01 HTML

```html
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>CS 410: Drawing Canvas Example 01</title>
  <link href="canvas01.css" rel="stylesheet" type="text/css"/>
  <script type="text/javascript" src="canvas01.js"></script>
  <script type="text/javascript">
    window.onload = init;
  </script>
</head>
<body>
  Simple example of drawing a line.
  <canvas id="dboard" width="300" height="300"></canvas>
</body>
</html>
```

Example 01 CSS

- In CSS there are selectors:
  - Class – multiple instances per page
  - ID – unique per page
    - Often used to bind object to name in JavaScript

```css
CHARSET UTF-8;

canvas#dboard {
  padding: 1px;
  border: 1px solid #000000;
}
```
Example 01 JavaScript

- Function `init` is called on window load
- In turn calls a drawing function

```
1: function drawIt() {
2:     ctx.clearRect(0, 0, dboard.width, dboard.height);
3:     ctx.beginPath();
4:     ctx.moveTo(25, 50);
5:     ctx.lineTo(225, 100);
6:     ctx.stroke();
7: }

8: function init() {
9:     dboard = document.getElementById("dboard");
10:    ctx = dboard.getContext("2d");
11:    drawIt();
12: }
```

Getting the Drawing Context

Observe the interplay between CSS and JavaScript

1. Get a document element by CSS ID name
2. Get the drawing context from that object

Note:
- `document` is a high level DOM object (DOM?)
- By construction we know "dboard" is a canvas object
Many excellent sources of documentation are available on the web.

The programming model of manipulating a complex object with state, the graphics context, is common throughout the world of 2D and 3D graphics.

The Canvas provides an excellent, and comparatively simple example.

What is a path?

Path as a Sequence of Lines

The 2D rendering context for the drawing surface of a <canvas> element. To get this object, call `getCanvas()` on <canvas>. Supplying "2D" as the argument:

```javascript
var canvas = document.getElementById('tutorial');
var ctx = canvas.getContext('2d');
```

Once you have the 2D rendering context for a canvas, you can draw within it. For example:

```javascript
ctx.fillStyle = "rgb(255,0,0);"
ctx.fillRect(10, 10, 25, 25);
```

See the Canvas tutorial for more information.

Method overview

- `arc(x, y, radius, startAngle, endAngle)`
- `arcTo(x1, y1, x2, y2)`
- `beginPath()`
- `closePath()`

What is a path?

Path as a Sequence of Lines
Close and Fill the Polygon

```javascript
function drawIt() {
  ctx.clearRect(0, 0, dboard.width, dboard.height);
  ctx.fillStyle = "LightGreen";
  ctx.strokeStyle = "DarkGreen";
  ctx.beginPath();
  ctx.moveTo(25, 50);
  ctx.lineTo(225, 100);
  ctx.lineTo(200, 200);
  ctx.lineTo(50, 250);
  ctx.closePath();
  ctx.fill();
  ctx.stroke();
}
```

Beginnings of a Viewport

```javascript
function drawIt() {
  ctx.fillStyle = "Yellow";
  ctx.strokeStyle = "DarkRed";
  ctx.beginPath();
  ctx.moveTo(0.1, 0.1);
  ctx.lineTo(0.8, 0.1);
  ctx.lineTo(0.7, 0.6);
  ctx.lineTo(0.2, 0.6);
  ctx.closePath();
  ctx.fill();
  ctx.stroke();
}
```

```javascript
function init() {
  dboard = document.getElementById("dboard");
  ctx = dboard.getContext("2d");
  w = dboard.width;
  h = dboard.height;
  ctx.lineWidth = 0.01;
  ctx.stroke();
  drowIt();
}
```
Standard XY Coordinates

Transformation Chains
The End