NYCDSA logo

Malcolm Hess

March 12, 2015

Contributed by Malcolm Hess

This project is based on my work with D3.js

Links

The result can be found here: http://jsfiddle.net/MalTheGreat/4dwe28bd/22/

You can find the full code here: https://github.com/Mal-Hess/DataScienceAcademyLogo

If you enjoyed this or would like to see more of my work feel free to check out my github account for more projects.
https://github.com/Mal-Hess/

About the project.

Goal

```
Recreate the NYC Data Science Academy logo in 3d.js.  Also use the 3d.js functions to create a unique trans
ition for loading the logo on the screen.
```

The logo

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

Process

I broke the task to create an animated logo into three main parts. The first part was to create multiple color gradients for all of the objects in the logo. This was made in the HTML side of the code. Then using javascript code I created the objects and aligned them into position. Afterwards I moved these objects to a hidden spot so I could animate them into view using javascript's transition effects.

Examining the HTML side, first I created the svg which will contain all of the objects that contain the logo. I also put into the html all of the color gradients for the objects. I chose to make the gradients in the html side because it was much easier than doing it on the javascript side. Each one has a unique id which will get called later.

To get the colors I used a color selector gagdet. A color selector program lets you click on a specific spot on a picture and

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

it returns the RGB color values of that pixel. There are a variety of color selectors out there to use, but I have no particular recommendation of which one to use. Each of the blue and yellow towers had their own color gradient. The color for antenna was a little unique, it was designed to transition from the base color to full black, this created the appearance that the tower fades to a point.

Below is some of the html code used to make the color gradients. The text container at the end was used for the words in the logo.

```
<svg width="50" height="50">
    <linearGradient id="greybase" x1="0%" y1="0%" x2="0%" y2="100%">
        <stop offset="0%" style="stop-color:rgb(235,235,236);stop-opacity:1" />
        <stop offset="100%" style="stop-color:rgb(100,100,100);stop-opacity:1" />
    </linearGradient>

    <linearGradient id="antenna" x1="0%" y1="0%" x2="0%" y2="100%">
        <stop offset="0%" style="stop-color:rgb(20,20,20);stop-opacity:1" />
        <stop offset="100%" style="stop-color:rgb(205,201,201);stop-opacity:1" />
    </linearGradient>

  <!-- yellow and blue boxes are numbered 1-4 from left to right-->
    <linearGradient id="yellow1" x1="0%" y1="0%" x2="0%" y2="100%">
        <stop offset="0%" style="stop-color:rgb(254,198,0);stop-opacity:1" />
        <stop offset="100%" style="stop-color:rgb(255,255,150);stop-opacity:1" />
    </linearGradient>

    <linearGradient id="blue4" x1="0%" y1="0%" x2="0%" y2="100%">
        <stop offset="0%" style="stop-color:rgb(0,126,255);stop-opacity:1" />
        <stop offset="100%" style="stop-color:rgb(0,180,255);stop-opacity:1" />
    </linearGradient>
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
    <text></text>
</svg>
```

All visible objects were made on the javascript side of the code. The objects were appended onto the svg container which was created in the HTML code.

To tackle on the challenge of making the logo both scalable and movable on the screen I first created x and y origin points and a radius. I set the size of all objects in relation to the radius. Each object's position is based off of the origin points and the radius. This way all of the points retain their relative position wherever the logo is set and the sizes remain proportional to the size of the main circle.

For example the variable the width_t is proporitonally sized to the radius. This variable will be the width for the four yellow and four blue towers.

```
var svgContainer = d3.select("body").append("svg:svg")
     .attr("width", 1000)
     .attr("height", 1000);

 var radius = new Number();
 var originx = new Number();
 var originy = new Number();

 //LOCATION OF ALL FIGURES BASED ON THESE ORIGIN POINTS
 //SIZE OF ALL FIGURES BASED ON RADIUS
 radius = 110;
 originx = 150;
 originy = 220;

 var width_t = new Number();
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
width_t = .121 * radius;

var circle = svgContainer.append("circle")
    .attr("cx", originx)
    .attr("cy", originy)
    .attr("r", radius);
```

To get the proportional locations and sizes in relation to the radius I used Data Thief. Data Thief let me set the middle of the image as an origin point for an x,y grid. It then created an x-axis that went directly to the right and finished at the end of the circle. The y-axis was the set to be the top most part of the circle. The lenght was set to 1000 units on both the x-axis and y-axis. Data Thief then showed me the euclidean co-ordinates of any point on the image.

Using the yellow1 bar (the bar on the far left) as an example, the upper-left corner of the bar was at co-ordinates (-847.5, -464.4). That means the bar is -.8475*radius units on the x-axis away from the center. Since for Javascript the y-axis is inverted, my yellow bar must be placed .4644* radius units added to the y-origin. This technique is how I established the location and sizes for all of the objects.

```
//Yellow Bars
var yellow1 = svgContainer.append("rect")
    .attr("id", "yellowtower1")
    .attr("x", originx + (-.8475 * radius))
    .attr("y", originy + (.4643 * radius))
    .attr("width", width_t)
    .attr("height", (.266 * radius))
    .attr("fill", 'url(#yellow1)');



//Blue Bars
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
var blue1 = svgContainer.append("rect")
    .attr("id", "bluetower1")
    .attr("x", originx + (.344 * radius) - width_t)
    .attr("y", originy + (-.1 * radius))
    .attr("width", width_t)
    .attr("height", (1.1 * radius))
    .attr("fill", 'url(#blue1)');
```

The middle tower consists of 6 seperate rectangles. I simplified the base by making it one giant rectangle and then overlaying three black lines on-top of it, these lines break the rectangle up into fourths. Since setting a gradient for a line is tricky I decided to make the antenna a very thin rectangle. It fades to black at the top which creates the illusion that it has a point at the top even though it is flat.

```
//Middle Grey Bars
 //Mid tower Base
 var greybase = svgContainer.append("rect").attr("id", "base1")
     .attr("x", originx + (-.1655 * radius))
     .attr("y", originy + (-.246 * radius))
     .attr("width", (.1655 * radius * 2))
     .attr("height", (1.28 * radius))
     .attr("fill", 'url(#greybase)')
     .append("base");


 //Grey bars to "split-up" Mid tower Base
 var line1 = svgContainer.append("line").attr("id", "bar1")
     .attr("x1", originx + (-.0905 * radius))
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
    .attr("y1", originy + (-.246 * radius))
    .attr("x2", originx + (-.0905 * radius))
    .attr("y2", originy + (-.246 * radius) + (1.28 * radius))
    .attr("stroke-width", 2)
    .attr("stroke", "black");

var line2 = svgContainer.append("line").attr("id", "bar2")
    .attr("x1", originx + (.0905 * radius))
    .attr("y1", originy + (-.246 * radius))
    .attr("x2", originx + (.0905 * radius))
    .attr("y2", originy + (-.246 * radius) + (1.28 * radius))
    .attr("stroke-width", 2)
    .attr("stroke", "black");

var line3 = svgContainer.append("line").attr("id", "bar3")
    .attr("x1", originx)
    .attr("y1", originy + (-.246 * radius))
    .attr("x2", originx)
    .attr("y2", originy + (-.246 * radius) + (1.28 * radius))
    .attr("stroke-width", 2)
    .attr("stroke", "black");

//Antenna
var line4 = svgContainer.append("rect").attr("id", "antenna")
    .attr("x", originx - ((.025 * 1 / 2 * radius * 2) / 3))
    .attr("y", originy + (-.767 * radius))
    .attr("width", ((.025 * radius * 2) / 3))
    .attr("height", (.196 * radius))
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
    .attr("fill", 'url(#antenna)');
```

The logo was almost complete. However I still needed to hide the parts of the towers that stuck out beyond the circle. To do that, I created a white ring and overlayed it around the edge of the black circle. I call objects like this white ring a hiding panel.

I made another white ring and another box which are hiding panels to ensure that the towers remain unseen before transitioning into position.

```
 //inner-white ring used to cover up parts of the bars
 var whitering = svgContainer.append("circle")
     .attr("cx", originx)
     .attr("cy", originy)
     .attr("r", radius * 1.06)
     .attr("fill", "transparent")
     .attr("stroke-width", radius / 8)
     .attr("stroke", "white");



//Cover-up white box placed below the main black circle, this is needed to hide the bars when before they a
re transitioned upward into position.
var whitebox = svgContainer.append("rect")
    .attr("x", originx - radius)
     .attr("y", originy + radius)
     .attr("width", (radius *2))
     .attr("height", (radius * 1.5))
     .attr("fill", 'white');

//another white ring used to cover up some parts between the inner black circle and the white box where the
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
towers became visible.
var whitering = svgContainer.append("circle")
     .attr("cx", originx)
     .attr("cy", originy)
     .attr("r", radius * 1.17)
     .attr("fill", "transparent")
     .attr("stroke-width", radius / 4)
     .attr("stroke", "white");



 //out-black ring
 var blackring = svgContainer.append("circle")
     .attr("cx", originx)
     .attr("cy", originy)
     .attr("r", radius * 1.08)
     .attr("fill", "transparent")
     .attr("stroke-width", radius / 17)
     .attr("stroke", "black");
```

The following code created text for New York Data Science Academy. I made it as two seperate text boxes with different font sizes to best copy the actual logo. I got the size and position of the text through trial and error. The font size is also proportionial to the radius so it scale with the entire logo.

```
var text = svgContainer.append("text")
 .attr("y", originy - radius - radius * .5)
     .attr("x", originx - radius * 1.15)
     .text("NYC DATA SCIENCE")
     .attr("font-family", "sans-serif")
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
    .attr("font-size", radius * .24);



 var text2 = svgContainer.append("text")
    .attr("y", originy - radius - radius * .25)
    .attr("x", originx - radius * .55)
    .text("ACADEMY")
    .style("font-weight", "bold")
    .attr("font-family", "sans-serif")
    .attr("font-size", radius * .24);
```

The main logo was complte. I moved all of the bars below the center black circle and I set the opacity of the middle tower to 0% which made it invisible.

```
//Following code creates the transition where the bars raise up
//from the bottom of the circle, the middle tower fades in.

//Make the middle towers "invisible" by setting opacity to zero.
d3.selectAll("#base1")
    .style("opacity", 0);
 d3.selectAll("#base2")
    .style("opacity", 0);
 d3.selectAll("#base3")
    .style("opacity", 0);
 d3.selectAll("#base4")
    .style("opacity", 0);
 d3.selectAll("#base5")
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
    .style("opacity", 0);
 d3.selectAll("#antenna")
    .style("opacity", 0);
 d3.selectAll("text")
    .style("opacity", 0);


//Setting side towers height to below the circle
 d3.select("#bluetower4")
    .attr("y", originy + radius);
 d3.select("#yellowtower1")
    .attr("y", originy + radius);
 d3.select("#bluetower3")
    .attr("y", originy + radius);
 d3.selectAll("#yellowtower2")
    .attr("y", originy + radius);
 d3.select("#bluetower2")
    .attr("y", originy + radius);
 d3.selectAll("#yellowtower3")
    .attr("y", originy + radius);
 d3.select("#bluetower1")
    .attr("y", originy + radius);
 d3.selectAll("#yellowtower4")
    .attr("y", originy + radius);
```

Lastly, the transition commands brought the logo to life. The transitions raised the bars to their original starting heights and the middle tower faded into existance. Getting the timing to feel smooth and clean took significant trail and error. After playing around with delay times I found a timing that gave the entire image a good flow.

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
//Transitions to raise the side bars up to the correct height for each.

//Outer towers raise first
 d3.select("#bluetower4").transition()
     .duration(1000)
     .attr("y", originy + (.4643 * radius))
     .ease("linear");

 d3.selectAll("#yellowtower1").transition()
     .duration(1000)
     .attr("y", originy + (.4643 * radius))
     .ease("linear");

//second set of towers
 d3.select("#bluetower3").transition()
     .delay(500)
     .duration(1000)
     .attr("y", originy + (.2571 * radius))
  .ease("linear");

 d3.selectAll("#yellowtower2").transition()
     .delay(500)
      .duration(1000)
     .attr("y", originy + (.2571 * radius))
  .ease("linear");

//third set of towers
 d3.select("#bluetower2").transition()
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
      .delay(800)
      .duration(1000)
      .attr("y", originy + (.07857 * radius))
   .ease("linear");

 d3.selectAll("#yellowtower3").transition()
      .delay(800)
      .duration(1000)
      .attr("y", originy + (.07857 * radius))
 .ease("linear");

//inner-most towers
 d3.select("#bluetower1").transition()
      .delay(1000)
      .duration(1000)
      .attr("y", originy + (-.1 * radius))
      .ease("linear");

 d3.selectAll("#yellowtower4").transition()
      .delay(1000)
      .duration(1000)
      .attr("y", originy + (-.1 * radius))
      .ease("linear");


 //Fade-in middle tower starting with the base
 d3.selectAll("#base1").transition()
      .delay(2000)
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
      .duration(2000)
      .style("opacity", 100);

  d3.selectAll("#base2").transition()
      .delay(2300)
      .duration(2000)
      .style("opacity", 100);

  d3.selectAll("#base3").transition()
      .delay(2600)
      .duration(2000)
      .style("opacity", 100);
  d3.selectAll("#base4").transition()
     .delay(2800)
     .duration(2000)
      .style("opacity", 100);
  d3.selectAll("#base5").transition()
      .delay(3000)
      .duration(2000)
      .style("opacity", 100);
  d3.selectAll("#antenna").transition()
      .delay(3200)
      .duration(2000)
      .style("opacity", 100);

//Fade in text of "NYC Data Science Academy"
  d3.selectAll("text").transition()
      .delay(3800)
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
    .duration(2000)
    .style("opacity", 100);
```

PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com