Lab 2: SVG

Bahl, Vikram edited this page 5 hours ago · 9 revisions

Learning Objectives

After completing this lab you will be able to:

- Understand how to do the following with Scalable Vector Graphics (SVG)
  - Create an SVG element
  - Edit the attributes of SVG elements
  - Position SVG elements
  - Create SVG groups
- Learn how to style SVG elements with CSS
- Add SVG elements to the DOM

Prerequisites

- Update your *starter code repository* using one of the following methods:
  i. From the GitHub Desktop app click `Sync` on the top right
  ii. Open a command line prompt. Navigate to the repository directory, for example `cd ~\Development\CS4460-Spring2018\Labs` and run command `git pull` .
- You have **read all of Chapter 3** in D3 - Interactive Data Visualization for the Web by Scott Murray (the SVG section)
- You have read Understanding SVG Coordinate Systems and Transformations (Parts 1 + 2) by S. Soueidan

Additional Reading

- HongKiat SVG Blog Series
- MDN SVG Tutorial
- Illustrated Guide to SVG "path" element
- w3 SVG Primer
- Intro to Web Technologies (SVG Section) by A. Lex of U. of Utah
- Nadieh Bremer's blog on Data Art - great showcase of the possibilities with SVG.

Activity 0 - Drawing a Slope Graph

For today's lab exercises we will be using a dataset covering the historical performance of World Cup Soccer teams. **For now we are only going to deal with 4 countries**, but we'll add all **77 countries for the second exercise**.

For your reference here is a sample of the overall data table (the csv file for this data table is located in the 2nd exercise files):

Pages 4

Lab 0: HTML & CSS
Lab 1: Javascript 101
Lab 2: SVG
Lab 3: Intro to D3
Lab 4: D3 Chart Types & Scales
Lab 5: D3 Selections & Grouping
Lab 6: D3 Enter, Update & Exit
Lab 7: Interaction & Transition 1
Lab 8: Interaction & Transition 2
Lab 9: D3 Layouts
Lab 10: D3 Maps

Clone this wiki locally
https://github.gatech.edu

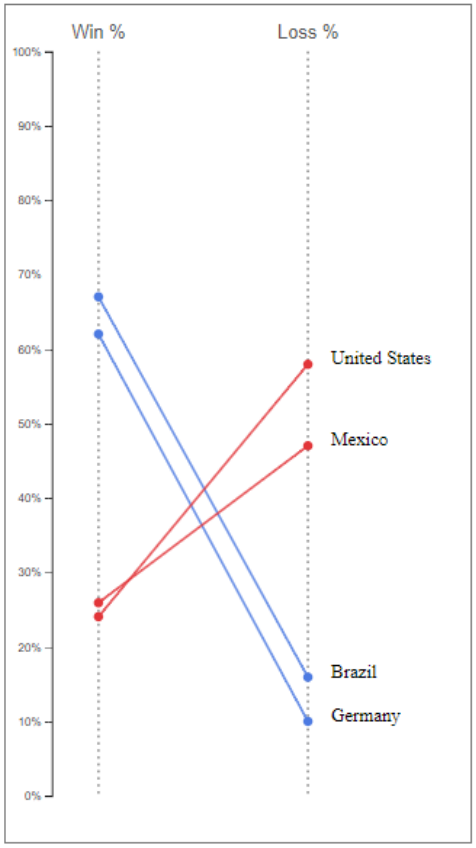
Country	Played	Wins	Draws	Losses	Goals_For	Goals_Against	Points	Best_finish
Brazil	104	70	17	17	221	102	227	Winner
Germany	106	66	20	20	224	121	218	Winner
Italy	83	45	21	17	128	77	156	Winner
Argentina	77	42	14	21	131	84	140	Winner
Spain	59	29	12	18	92	66	99	Winner
England	62	26	20	16	79	56	98	Winner
France	59	28	12	19	106	71	96	Winner
Netherlands	50	27	12	11	86	48	93	Runner-up
Uruguay	51	20	12	19	80	71	72	Winner
Sweden	46	16	13	17	74	69	61	Runner-up
Russia	40	17	8	15	66	47	59	Fourth Place
Serbia	43	17	8	18	64	59	59	Fourth Place
Mexico	53	14	14	25	57	92	56	Quarter-finals
Belgium	41	14	9	18	52	66	51	Fourth Place
Poland	31	15	5	11	44	40	50	Third Place

In this exercise we will create a slope graph. Slope graphs are useful for comparing the correlation between two quantitative data variables, especially when it makes sense to normalize the variables.

For this exercise we will only be using the following *4 countries*. We want to compare the number of wins for each team to their number of losses. To normalize this data, and to support better comparison on teams that have played a lot of games vs. teams that haven't played many, we will **compare the win and loss percentages.**

Country	Total Played	Win %	Win y-value	Loss %	Loss y-value
Brazil	104	67%	251	16%	578
Germany	106	62%	283	10%	616
United States	33	24%	526	58%	309
Mexico	53	25%	514	47%	379

The goal of this exercise is to create the following slope graph:



John Doe

You will be directly editing the HTML file in `02_lab\01_slope_graph\index.html`. Notice that we have already added axes for your slope graph in the SVG element. You will add new SVG elements to the bottom of the `<svg>` (before the closing tag `</svg>`. Also, we have already computed a "scaled value" for each country's win and loss percentages. You will use these `y-values` when making your graph.

1. Create a Slope Graph

To create a slope graph you will need to make the following SVG elements for each country:

- 1 line element - with the start point @ (x=80, y= `Win y-value`) and the end point @ (x=260, y= `Loss y-value`)
- 2 circle elements - each centered on the start and end points of the above line
- 1 text element - the content should be the country name, and positioned to the right of the end point

Check to make sure your slope graph looks like the above example.

2. Style a Slope Graph

Feel free to edit the linked CSS file at `02_lab\01_slope_graph\style.css` to style your chart. Or you can add styling directly in-line within your SVG elements (e.g. `<circle style="fill: red;"/>`

Next, style the lines and circles for each Country. Color the team's in the following two categories:

- blue = a higher `Win %` than `Loss %`
- red = a higher `Loss %` than `Win %`

You will need to use the `fill` and `stroke` SVG style properties to add color to your circles and lines.

3. Submission

Change the text of `<p>` tag right before the closing of on index.html file located in `02_lab\01_slope_graph\index.html` with your name.

Take a screenshot of the slope graph from the activity. (Note that this is required for you to receive the grade for this activity!)**

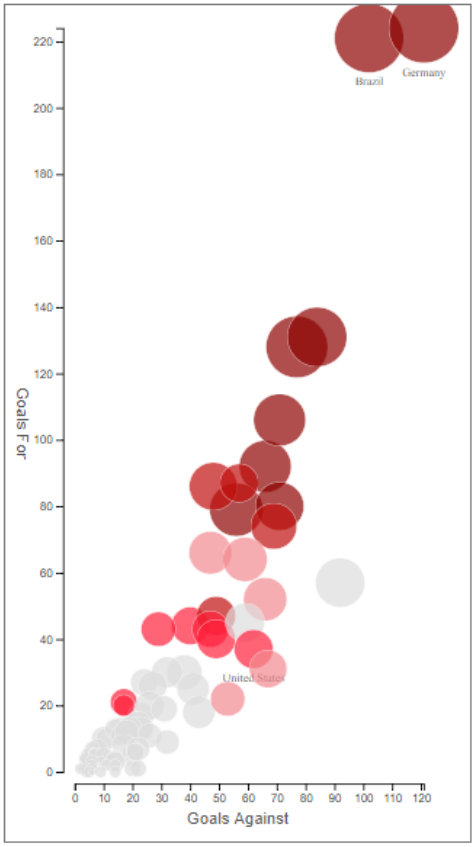
> **Reload Page** Remember to do a hard reload of your page after changing JS or CSS files. This can be done in Chrome with `cmd+shift+R` or `ctrl+shift+R`

Activity 1 - Create a Bubble Chart

We will be using the full World Cup Soccer dataset now to create a bubble chart. A bubble chart is basically a scatterplot, however the **area** of the circles represents a quantitative value (sometimes ordinal can be used). As mentioned in the previous exercise, the data looks this:

Country	Played	Wins	Draws	Losses	Goals_For	Goals_Against	Points	Best_finish
Brazil	104	70	17	17	221	102	227	Winner
Germany	106	66	20	20	224	121	218	Winner
Italy	83	45	21	17	128	77	156	Winner
Argentina	77	42	14	21	131	84	140	Winner
Spain	59	29	12	18	92	66	99	Winner
England	62	26	20	16	79	56	98	Winner
France	59	28	12	19	106	71	96	Winner
Netherlands	50	27	12	11	86	48	93	Runner-up
Uruguay	51	20	12	19	80	71	72	Winner
Sweden	46	16	13	17	74	69	61	Runner-up
Russia	40	17	8	15	66	47	59	Fourth Place
Serbia	43	17	8	18	64	59	59	Fourth Place
Mexico	53	14	14	25	57	92	56	Quarter-finals
Belgium	41	14	9	18	52	66	51	Fourth Place
Poland	31	15	5	11	44	40	50	Third Place

Again, the goal of this exercise is to create the following bubble chart of world cup performances for each country:



John Doe

In this exercise we will introduce you to the `d3.append()`, `d3.style()`, and `d3.attr()` functions as a preview for next week's d3 lab. The following code adds a new circle element to the svg and changes its visual properties:

```
var svg = d3.select('svg');
var circle = svg.append('circle');
circle.attr('r', 10);
circle.style('fill', 'white');
```

What is going on here? The `svg` variable that I created is a d3-selection. Selections are references to DOM elements. In this case the `svg` variable refers to the `<svg>` element in the DOM. Then I call `append('circle')` on `svg` to add a new `<circle>` element inside of the `<svg>` (you can add any DOM element using `append()`). The `append` function returns a d3-selection of the DOM element we just added, in this case we now have a reference to the `<circle>` element.

Then we can use the `attr()` and `style()` methods on our `circle` d3-selection to change the properties of the shape.

Now we are ready to create our bubble chart. Start by serving the `02_lab\` directory with `python -m SimpleHTTPServer 8080` (or `python -m http.server 8080` in Python v3). You should see a blank chart at `localhost:8080/02_bubble_chart/`. Next, opening the directory `02_lab\02_bubble_chart` in your code editor. You will see a familiar code structure `index.html` with an `<svg>` element already defined, `bubble_chart.js` is where you will add code to create each circle for the bubble chart. Also notice that our dataset `worldcup.csv` is in our directory. **The starter code will load the dataset for you,** but it is helpful to inspect the rows and columns of the dataset.

In this exercise you will need to edit the function `createCountryBubble(svg, goalsAgainst, goalsFor, gamesPlayed, country)`. This function will be called by the starter code. The function

will be called each time for a country, and will pass the inputs for making a bubble for that specific country. Your job is to append a `circle` element to the `svg`, and then change the attributes and style of the returned `circle` variable (a d3-selection).

> Since we have not covered axes and scales yet in the lab or during class, **we have already scaled the `goalsAgainst, goalsFor, gamesPlayed` attributes for you**. Notice, that we used a *square root* scale for the radius of a circle since we are encoding the area of each circle, not the circumference.

1. Add a circle for each country

In `createCountryBubble()` append a new circle element to the `svg` element. Be sure to instantiate a variable for the returned circle selection.

You will not see the circles on the canvas, because they do not have a radius yet. You can however inspect the DOM, and you should see 77 `<circle>` elements in the `<svg>`.

2. Set circle radius

Still in `createCountryBubble()`, use the `circle` d3-selection returned by `append()` to change the radius attribute of the circle to the `gamesPlayed` input value.

Now you will see all of the circles bunched at the top-left of the canvas. This is because their default center is at (0,0). We will need to position all of the circles next.

3. Set circle position

Use the same `circle` d3-selection returned by `append()` to change the center position of the circle. You will set the x-position of the circle to the `goalsAgainst` value and the y-position to `goalsFor` value.

Now you should see completed bubble chart.

4. Style the circles

Notice that all of the circles are black (the default fill color), and that we cannot see a lot of the bubbles. You should use either the style-sheet `style.css` or add in-line styling to the circle with `.style(style-attribute, value);`

Challenge #1

Color the circles based on the country's best finish. You can do this in a number of different ways, I would recommend using in-line styling, and use the following color map (hint colorMap["Winner"] will return the "#8C0200" RGB code, and so on):

```
var colorMap = {
        "Winner": "#8C0200",
        "Runner-up": "#C00F0D",
        "Third Place": "#FF1F3A",
        "Fourth Place": "#F2898F",
        "Quarter-finals": "#DDDDDD",
        "First Round": "#DDDDDD",
        "Second Round": "#DDDDDD"
};
```

Challenge #2

Give labels to *United States, Germany and Brazil*. You will need to use your JavaScript skills to append a `text` element to the svg *if* the country's `country` attribute is one of the above values.

Note: use the `.text("content")` method on a d3-selection to add text content.

5. Submission

Change the text of `<p>` tag right before the closing of on index.html file located in `02_lab\02_bubble_chart\index.html` with your name.

Take a screenshot of the slope graph from the activity. (Note that this is required for you to receive the grade for this activity!)**

In total, you need to submit 2 screenshots (Activity 0 - Slope Graph and Activity 1 - Bubble chart) via T-Square. Remember to submit these 2 screenshots before 10:05 AM of lab day.

> **Reload Page** Remember to do a hard reload of your page after changing JS or CSS files. This can be done in Chrome with `cmd+shift+R` or `ctrl+shift+R`

A final note on presentation attributes:

As you might have noticed by comparing the HTML examples with the SVG ones, some appearance aspects are controlled by HTML attributes; others are controlled by CSS properties. This is a perennial source of confusion, and unfortunately there's no good way around it. To add to the confusion, a subset of SVG attributes can also be specified via CSS: these are the "presentation attributes".

It's worth remembering this because CSS declarations for these attributes will override inline attribute definitions in the DOM. This is in turn inconsistent with the rule for the style attribute itself, which overrides CSS definitions (on behalf of whoever designed this standard: I am sorry).

This lab is based on the following material:

- Intro to Web Technologies (SVG Section) by Alex Lex of U. of Utah
- Understanding SVG Coordinate Systems and Transformations (Parts 1 + 2) by S. Soueidan
- Hanspeter Pfister's CS171 Lab Material (Harvard)
- D3 - Interactive Data Visualization for the Web by Scott Murray

Home