27 조

기존의 데이터 시각화 방법 비교 및 분석 보고서

1. 개요

현재 데이터 시각화를 위한 여러가지 방법이 존재하지만, 각각의 방법마다 한계가 있다. 따라서 이를 보완하기 위한 데이터 시각화용 자바스크립트 프레임워크(IRMI.js)개발에 도움이 될 수 있도록, 기존에 있는 여러 데이터 시각화 방법들을 이용하여 여러 차트들을 만들어보고, 그 경험을 통해 비교, 분석을 위한 보고서를 작성한다.

A. 기존의 데이터 시각화 방법

i. JavaScript
ii. Google Charts
iii. Chart.js
iv. D3.js
v. 그 외 기타 등.. (Chartist.js, Dc.js, Plotly.js, TechanJS, Cola.js, VivaGraph 등 여러 종류가 있다.)

2. 기존의 데이터 시각화 방법 비교, 분석

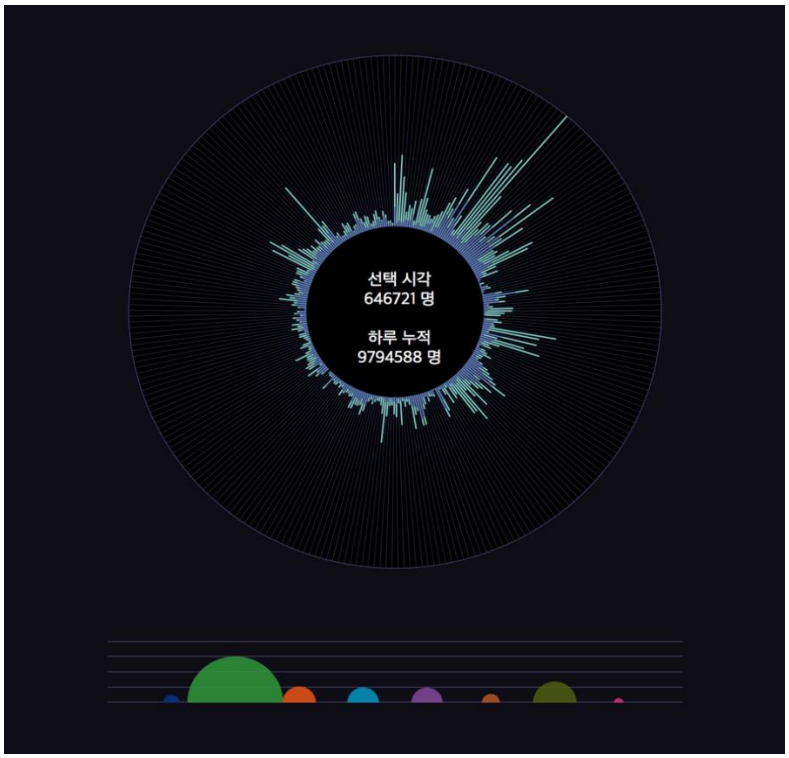
- 기본적으로 데이터 시각화를 하는 과정에서는 데이터를 차트로 표현하는 경우가 대부분이다. 순수 자바스크립트만 사용하여 차트를 구현하려면 코드가 굉장히 길어지기 때문에, 이를 도와주기 위한 여러 라이브러리가 존재한다. 기본적으로 자바스크립트 라이브러리는 SVG 태그를 이용하는 것과 CANVAS 태그를 이용하는 것으로 나뉜다.

	SVG	CANVAS
장점	1) 벡터 그래픽 포맷으로 해상도에 독립적. (그리기 프로그램과 같다고 볼 수 있다.) 2) 높은 수준의 애니메이션 지원 O 3) 자바스크립트로 SVG DOM API 를 이용한 모든 요소를 자유자재로 컨트롤. 4) XML 파일 포맷이므로 모든 웹 브라우저에 지원, 웹 어플리케이션 UI 에 좀 더 좋은 솔루션으로 이용.	1) 픽셀 수준의 처리에 용이함. (페인트 프로그램과 같다고 볼 수 있다.) 2) .PNG 또는 . JPG 파일로 저장 가능. 3) 게임이나 도형 처리, 이미지 편집과 같이 작업을 할 때 유리.
단점	1) 문서 복잡도가 증가할수록 (DOM 이 많을수록) 느리다. 2) 게임을 만들때에는 부적합.	1) DOM 노드가 없고, 픽셀로만 컨트롤 가능. 2) 애니메이션에 대한 API 가 없어서 필요할때마다 이벤트를 업데이트하거나 타이머를 작동시켜야함. 3) 지원되지 않는 브라우저가 있음.

A. JavaScript 만으로 구현

- 자바스크립트는 웹을 위한 객체 기반의 스크립트 프로그래밍 언어이다. 주로 웹 페이지에 기능을 더해 HTML 웹 페이지를 동적이고 살아있게 만드는 기능을 한다. HTML 요소와 콘텐츠를 추가하고 제거할 수 있으며 여러 이벤트를 다룬다.
- 데이터 시각화를 위한 여러 라이브러리가 등장하기전에는 순수 자바스크립트 만으로 여러 차트들을 구현해야 했다. 하지만 단순한 그래프 하나를 만들기 위해서도 불필요하게 많은 양의 코드가 필요했기 때문에 차차 여러 라이브러리가 등장하기 시작했다.
- 자바스크립트는 그래프나 차트를 위해 만들어진 언어가 아니라 웹 페이지 자체의 동적인 효과나 이벤트 등을 위해 만들어진 언어이기 때문에, 데이터 시각화를 하는데 있어 가장 널리 사용되는 차트를 만든다 할지라도, 차트로 인식을 하는게 아니라 하나의 도형으로 인식을 한다. 따라서 막대그래프의 선 하나하나를 하나의

도형으로 보고 생성해야하며, 아래의 그림과 같이 원형으로 만들고 싶을때는 각 막대그래프에 일정한 각도의 회전을 주어 원형이 되는 형식으로 생성해야한다.



(JavaScript 를 이용한 데이터 시각화물 구현)

✓ 장점

1) 원하는 디자인에 매우 가깝게 구현할 수 있다.
2) 정해진 차트만 사용해야 한다는 제약이 없다.
3) 동적으로 움직이는 디자인을 만들어 낼 수 있다.

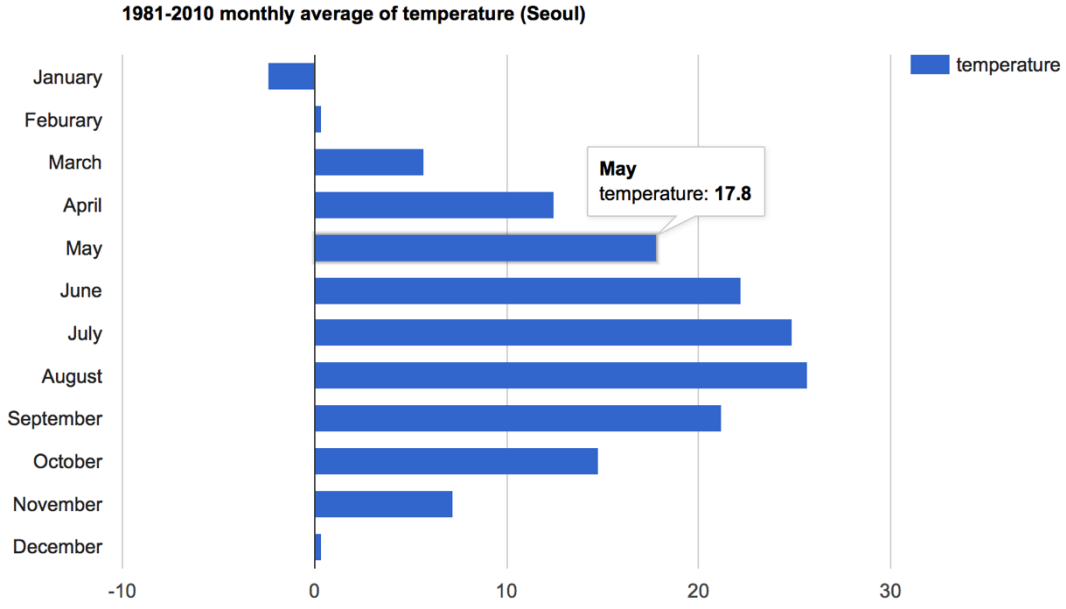
✓ 단점

1) 기본으로 제공되는 차트가 없어서 모든 그래프를 직접 구현해야 한다.
2) 자바스크립트 언어를 잘 알지 못하면 막대 그래프 하나 구현하기도 힘들다.

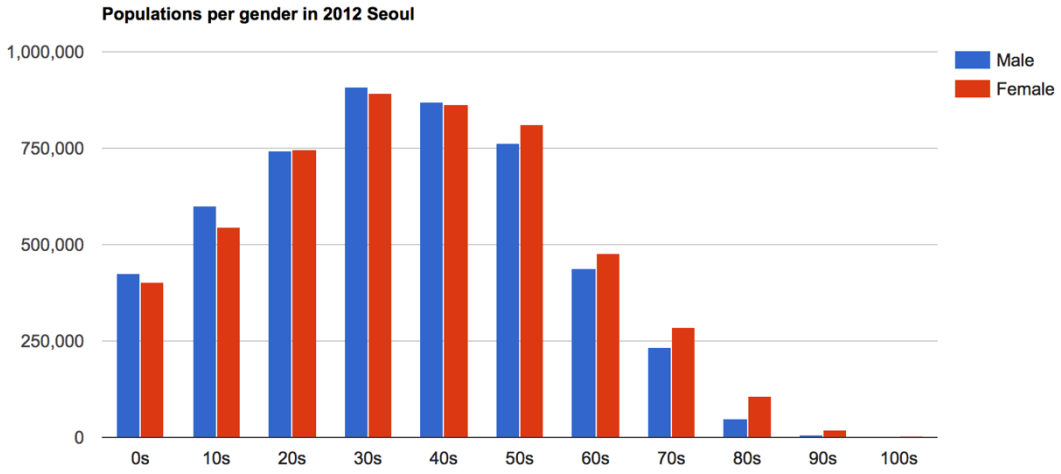
3) 간단한 시각화물을 구현해도 코드가 매우 길고 복잡해진다.

B. Google Charts

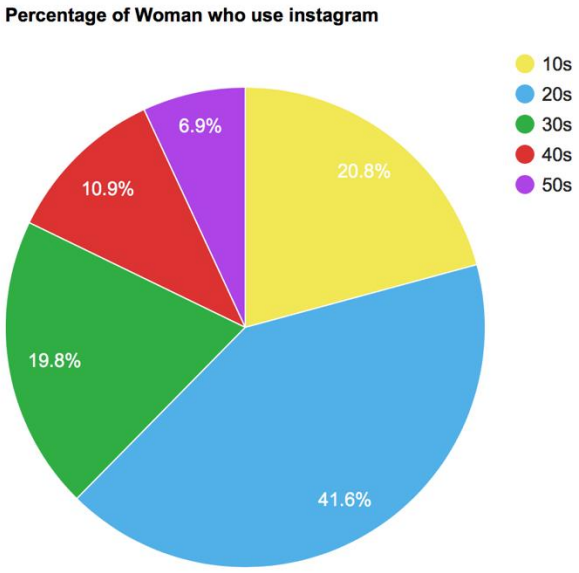
- Google Charts 는 구글에서 제공하는 그래프 구현 서비스로 HTML5 / SVG 기반으로 차트가 구성되며, 별도의 플러그인 설치없이 사용 가능하다.
- Geo chart, Scatter chart, Column chart, Histogram, Pie chart, Tree map, Gauge, Timeline, Org chart, Bubble chart 등 다양한 종류의 차트로 구성되어있다.
- 크게 2 가지의 구조로 나뉘는데, 처음으로 라이브러리를 load 한 뒤, 차트를 생성하는 함수를 정의해준다. 차트 생성을 위해서는 차트로 나타내고자 하는 데이터를 세팅해주고, title, 색상, 각 축의 label 등을 설정하는 옵션을 세팅해준다.
- 데이터 구성을 위하여 그래프 구축 시 테이블 형태의 자료구조를 지원한다. 데이터 테이블에 배열의 형태로 순차적으로 추가하거나, 데이터 테이블의 특정 행과 열에 값을 추가하는 방식 2 가지 중 선택해 사용 할 수 있다.



(Google Charts 를 이용한 Horizontal bar chart 1)



(Google Charts 를 이용한 Grouped bar chart 1)



(Google Charts 를 이용한 pie chart 1)

✓ 장점

1) 차트를 하나하나 직접 구현하는 것이 아니라, 차트의 이름만 지정해주면 해당 차트가 자동으로 구현된다.

2) 다른 차트 라이브러리에 비해 종류가 다양하다.

3) 데이터를 받아오는 부분, 색상 등 여러 옵션을 지정하는 부분, 차트를 불러오는 부분의 구분이 명확해서 처음 접하는 사람도 쉽게 사용할 수 있다.

✓ 단점

1) 차트의 종류가 많다고는 하지만, 원하는 디자인의 차트를 구현하기 어렵다.

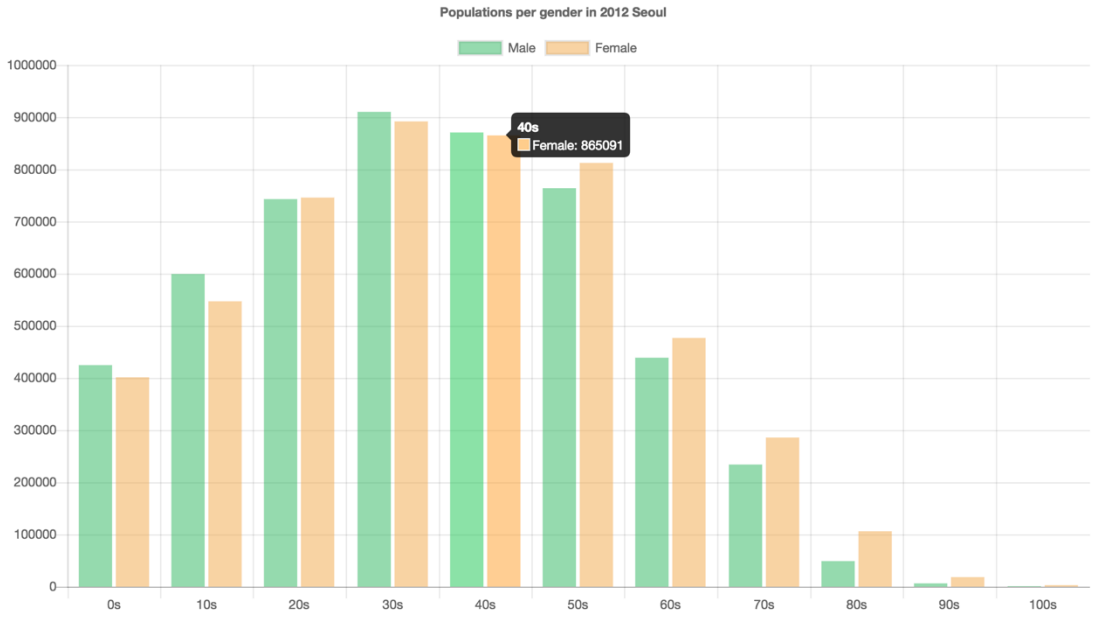
2) Horizontal chart(수평으로 된 bar 차트) 같은 경우에는 각 그래프의 색상을 지정해줄 수 없다.

3) 색상 변경을 하는 경우 rgb 형태의 색상은 사용가능하지만, 투명도가 포함된 rgba 형태의 색상은 인식하지 못한다. (색상이 굉장히 단조로움)

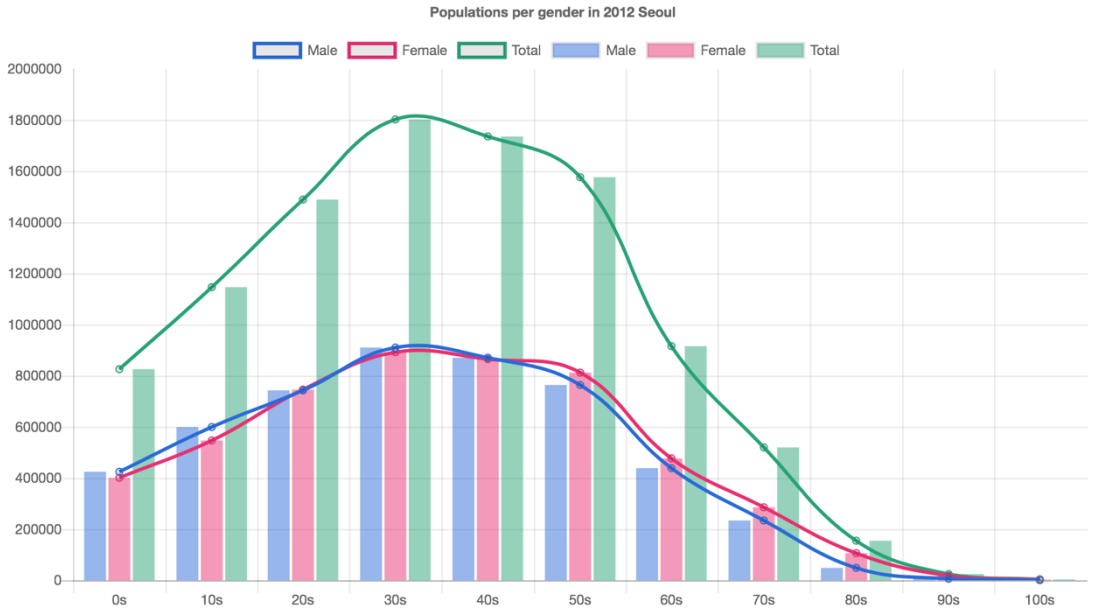
C. Chart.js

- Chart.js 는 디자이너와 개발자들을 위해 쉽고 빠른 차트를 제공하는 HTML5 Canvas 방식의 오픈소스 라이브러리이다. (라이선스: MIT License)
- Chart.js 는 svg 를 사용하지 않고, 대신에 canvas 태그를 사용한다.
- GitHub 에서 chart.js 최신 버전을 다운받아 사용하거나, 스크립트 주소를 html 헤더안에 삽입해주면 바로 사용 가능하다.
- 데이터 시각화에는 여러 영역이 있지만 데이터가 가장 많이 활용되는 부분이 차트이기 때문에, 그것을 위해 만들어진 라이브러리이다. Line chart, Bar chart, Doughnut & Pie chart, Polar Area chart, Bubble chart, Scatter chart, Area chart, Mixed chart 의 총 8 가지 종류의 차트를 제공하며, type 값에 속성을 적용함으로써 사용할 수 있다.
- 8 가지 종류를 차트를 제공하지만, 딱 8 개가 아닌 그 안에서 여러 종류의 차트들을 제공한다. 예를 들면 bar chart 같은 경우에는, 기본적인 차트부터 수평차트(horizontal chart), Stacked chart, Multi axis chart 등이 있고, line chart 에는 가장 기본적인 차트, Stepped chart, Interpolation chart, Point style chart, 그리고 area chart 에는 Boundary chart, Dataset chart, Radar chart 등이 존재한다.
- type 란에 line, pie 등의 차트 종류를 선언해주면 자동으로 그에 해당하는 차트를 생성한다.
- Option 에는 색상 지정 등 여러 기능이 있는데, 이 역시 배열의 형태를 허용하며 각 옵션은 같은 인덱스 값을 가진 데이터셋의 배열에 대응된다. (color 옵션의 첫번째 요소 > data[0]의 색상에 해당)

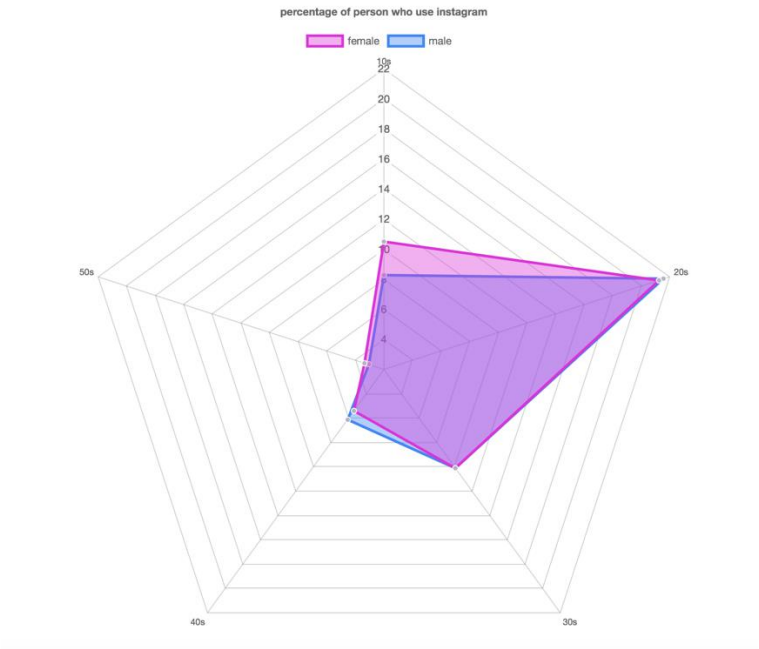
- 간단한 브라우저 이벤트도 지원이 되는데, mousemove, onClick, onHover 등의 이벤트가 가능하다. 기본적으로 차트 위에 마우스를 올리면 해당 차트가 어떤 데이터 값을 나타내는지, 해당 데이터 값은 얼마인지를 보여준다.



(Chart.js 를 이용한 grouped bar chart 1)



(Chart.js 를 이용한 Mixed chart 1)



(Chart.js 를 이용한 radar chart 1)

✓ 장점

1) 보통 예제 코드를 실행해보고 간단하게 바꾸어 보려면, 오픈소스의 메뉴얼이나 레퍼런스를 봐야하지만, chart.js 경우 따로 보지않아도 이 코드가 무슨 기능을 하는지 알기 쉽게 되어있어 초보자도 쉽게 접근할 수 있다.

2) type 에 무슨 차트를 사용할 건지만 명시해준다면, 차트를 직접 그리지 않아도 자동으로 차트가 생성된다.

3) 축은 입력된 데이터의 값에 맞춰서 자동으로 그려지며, 각 데이터의 라벨 또한 지정만 해주면 데이터 바로 밑에 자동으로 생성된다.

4) 다양한 색상 적용이 가능하다. Html5 색상코드를 그대로 사용가능하며, rgba 형태의 색상도 제공한다. 디자인적인 측면에서 다른 라이브러리에 비하여 색감이 예쁘게 나오는 편이다.

5) 각 데이터가 의미하는게 무엇인지 title 밑에 자동으로 명시해준다.

6) 따로 자바스크립트 이벤트 코드를 넣어주지 않아도, 마우스를 각 컨텐츠에 올리면 데이터 값을 표시해준다.

✓ 단점

1) 차트의 종류가 너무 제한적이다.

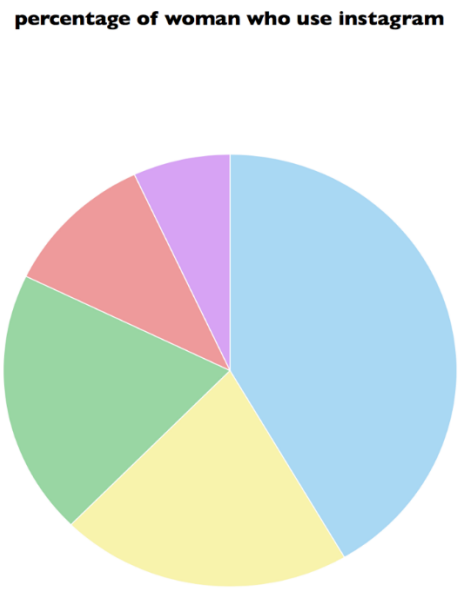
2) 만들고자 하는 종류의 차트가 존재한다해도 원하는 디자인은 천차만별인데 색상과 크기 변경외에는 디자인을 변경 할 수 없다.

D. D3.js

- D3 란 Data-Driven Documents (데이터 기반 문서) 를 의미하는 단어로, 데이터에 기반을 둔 문서를 다룰 수 있는, 웹 브라우저 상에서 동적이고 인터렉티브한 시각화를 위한 자바스크립트 라이브러리이며, HTML, CSS, SVG 를 사용하여 데이터를 시각적인 결과물로 나타낸다.
- 자바스크립트 변수에 직접 데이터를 할당하여 사용할 수도 있지만, 데이터의 양이 방대할 때에는 XML, CSV, JSON 등 다양한 데이터 형식을 취급한다.
- jQuery 사용법과 매우 유사하며, 돔(DOM)을 제어할 수 있다.

```
- d3.select("body").selectAll("div")
```

- 처음 d3.js 를 시작할 때, 위와 같이 html 파일에 선언된 body 문서요소를 선택하고 그 내부의 모든 div 를 선택하는 코드를 많이 볼 수 있는데, 실제 html 파일을 보면 body 만 선언해 주었을 뿐, div 는 어디에도 선언되지 않았다. 이것은 가상의 문서요소에 'data(dataSet)' 을 이용하여 우리가 정의하는 데이터를 바인딩 하고, 'enter()' 함수로 dataSet 배열의 각 요소를 순화하며 가상의 문서요소 div 를 만든 후 마지막으로 'append('div')' 함수를 통해 가상의 문서요소를 body 요소의 하위로 추가를 해주면 데이터를 기반으로 DOM 이 만들어진다. (화면상에서는 아무런 변화가 없지만 개발자도구로 보면 문서 요소가 추가된 것을 확인할 수 있다.)



(D3.js 를 이용한 pie chart 1)

-

✓ 장점

1) 디자이너가 상상하는 거의 모든 것을 생성할 수 있다. 이 점에서 D3 는 아주 강력하기 때문에 많은 사람들이 데이터 시각화로 d3.js 를 사용한다.

2) jQuery, CSS 에서의 셀렉터를 사용하여 DOM 요소들을 선택하고 변경하기 때문에, 직접적인 자바스크립트를 사용하는 것보다 쉽다.

✓ 단점

1) 완전히 처음부터 학습을 해야하기 때문에 다른 사람들이 만든 코드를 수정하는 것이 아니라면, 코딩과 디자인 기술 모두가 필요하다. 따라서 학습하는데에 오랜 시간이 소요되며, 디자이너가 원하는대로 동적이고 화려한 시각화물을 만들어내기 위해선 전문가 수준의 숙련도가 필요하다.

2) D3 의 일부는 예전 브라우저가 지원되지 않으므로, 옛날 버전을 지원하기 위해서는 Sizzle 또는 Raphael 등의 라이브러리를 이용한 추가적인 코딩이 필요할 수 있다.

3) Google Charts 나 Chart.js 로 만든 차트와 같은 결과물을 D3.js 를 이용해 만들기 위해선 코드량이 최소 몇배 이상 늘어난다.

4) 모든 것 하나하나를 직접 설정해주어야 한다. 예를 들면, 모든 svg 가 그러하듯 원점인 왼쪽 위를 기준으로 아래로 갈수록 좌표가 커지기 때문에 그냥 차트를 그리면 모든 차트가 반대로 생기게 된다. 따라서 데이터값을 svg 에서 빼줌으로써 차트의 높이를 바로 설정해야한다.

5) Google Charts 나 Chart.js 는 데이터 값에 따라 축이 자동으로 생성되지만, d3.js 는 축의 단위, 크기, 위치 모두를 직접 설정해주어야 한다. 각 데이터의 label 또한 위치, 크기 등을 모두 설정해주어야 한다.

6) Google Charts 나 Chart.js 와 달리 결과로 나오는 차트에서 각 색상 별 차트가 의미하는게 무엇인지 설정해주지 않으면 차트 옆에 따로 표시되지 않는다.

7) 막대 그래프를 예로 들었을 때, 음수로 된 데이터 값을 받아오면 다른 차트 라이브러리들은 자동으로 음수 처리를 하여 반대 방향의 그래프를 만들어내지만, d3.js 는 음수 데이터 값을 나타내지 못한다. (transform 같은 다른 별도의 작업이 필요.)

8) 데이터 값 만큼을 표현해 줄 때, 화면의 크기에 맞춰 자동으로 커지지 않는다. 예를 들어, 데이터 값이 최대가 300 인 만큼을 표시했을 때 꽉 차는 크기의 화면에, 데이터 값이 최대가 30 인 만큼을 표현하려 해도 한눈에 알아볼 수 있게 30 만큼이 표시 되는게 아니라 정말 데이터 값에 비례해서 표시되기 때문에 일일이 축을 바꿔주고 크기 또한 다른 옵션을 주어야 한다.

3. 각 라이브러리 비교 표

	Google Charts	Chart.js	D3.js
SVG	o		o
Canvas		o	
IE 8 지원	o	o	
무료 이용 가능	o	o	o
기본 제공 차트	o	o	
자유도	낮음	낮음	매우 높음
난이도	낮음	낮음	원하는 디자인을 구현하려면 전문가 수준의 숙련도
색상	투명도 지원 x	모든 형태 지원	모든 형태 지원