27조

기존의 데이터 시각화 방법 비교 및 분석 보고서

# 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.

## 기존의 데이터 시각화 방법

### Javascript

### Google Charts

### Chart.js

### D3.js

### 그 외 기타 등.. (Chartist.js, Dc.js, Plotly.js, TechanJS, Cola.js, VivaGraph 등 여러 종류가 있다. )

# 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 svg태그를 이용하는 것과 canvas 태그를 이용하는 것으로 나뉜다.

|  |  |  |
| --- | --- | --- |
|  | svg | canvas |
| 장점 | 벡터 그래픽 포맷으로 해상도에 독립적.애니메이션에 대한 지원o자바스크립트로 svg dom api 를 이용한 모든 요소를 자유자재로 컨트롤.xml파일 포맷이므로 모든 웹 브라우저에 지원, 웹 어플리케이션 ui에 좀 더 좋은 솔루션으로 이용. | 높은 퍼포먼스의 2D영역으로 원하는 어떤것이든 드로잉 가능..png 또는 . jpg 파일로 저장 가능.게임이나 도형 처리, 이미지 편집과 같이 픽셀 수준의 처리에 용이함. |
| 단점 | 문서 복잡도가 증가할수록 (DOm이 많을수록) 느리다.게임을 만들때에는 부적합. | dom노드가 없고, 픽셀로만 컨트롤 가능.애니메이션에 대한 Api가 없어서 필요할때마다 이벤트를 업데이트하거나 타이머를 작동시켜야함.지원되지 않는 브라우저가 있음. |

## **Javascript만으로 구현**

## 데이터 시각화를 위한 여러 라이브러리가 등장하기전에는 순수 자바스크립트 만으로 여러 차트들을 구현해야했다. 하지만 단순한 막대그래프 하나를 만들기 위해서도 불필요하게 최소 몇백줄의 코드가 필요했기 때문에 차차 여러 라이브러리가 등장하기 시작했다.

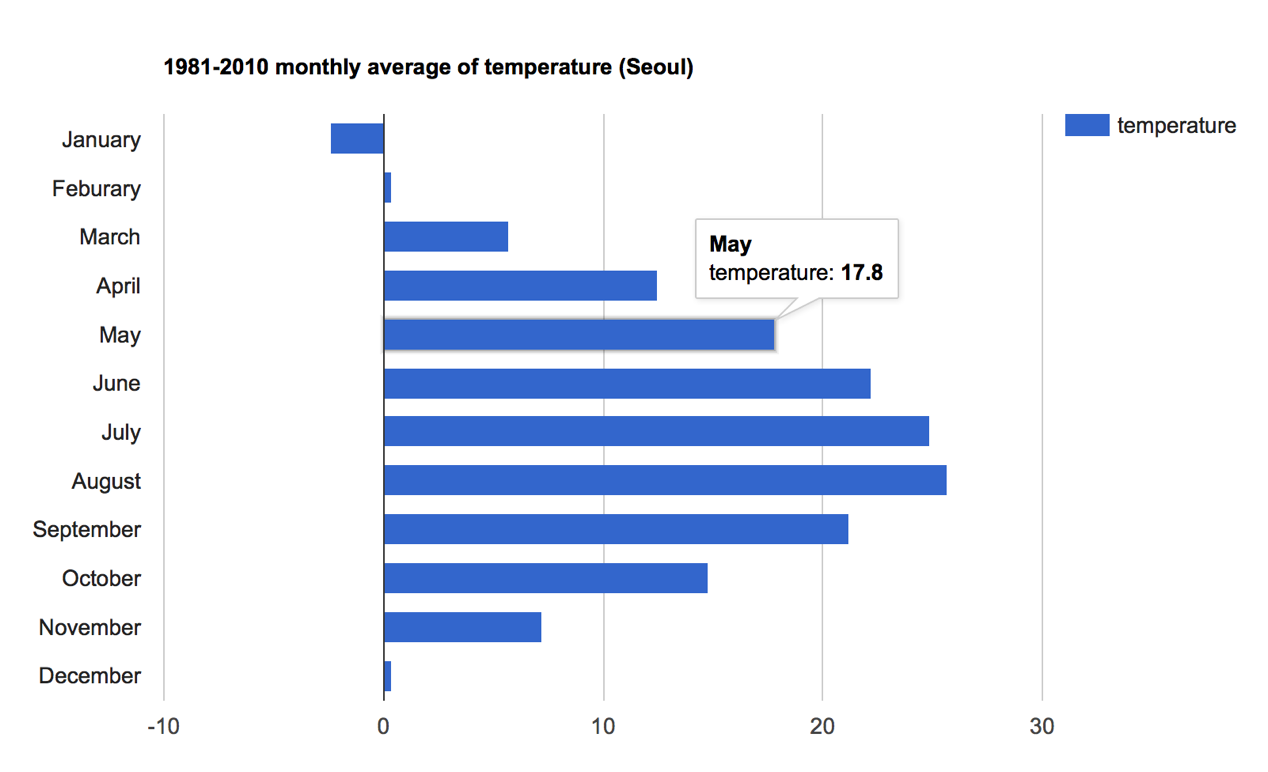
## **Google Charts**

## Google Charts 는 구글에서 제공하는 그래프 구현 서비스로 HTML5 / Svg 기반으로 차트가 구성되며, 별도의 플러그인 설치없이 사용가능하다.

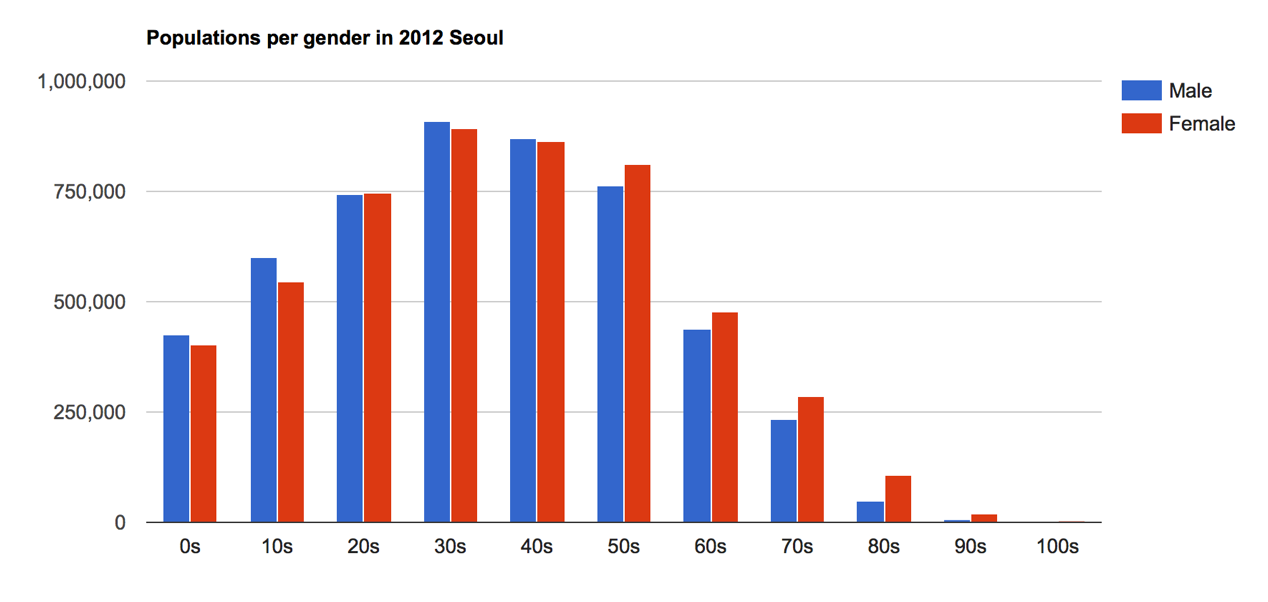
## Geo chart, Scatter chart, Column chart, Histogram, Pie chart, Tree map, Gauge, Timeline, Org chart, Bubble chart 등 다양한 종류의 차트로 구성되어있다.

## 크게 2가지의 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.

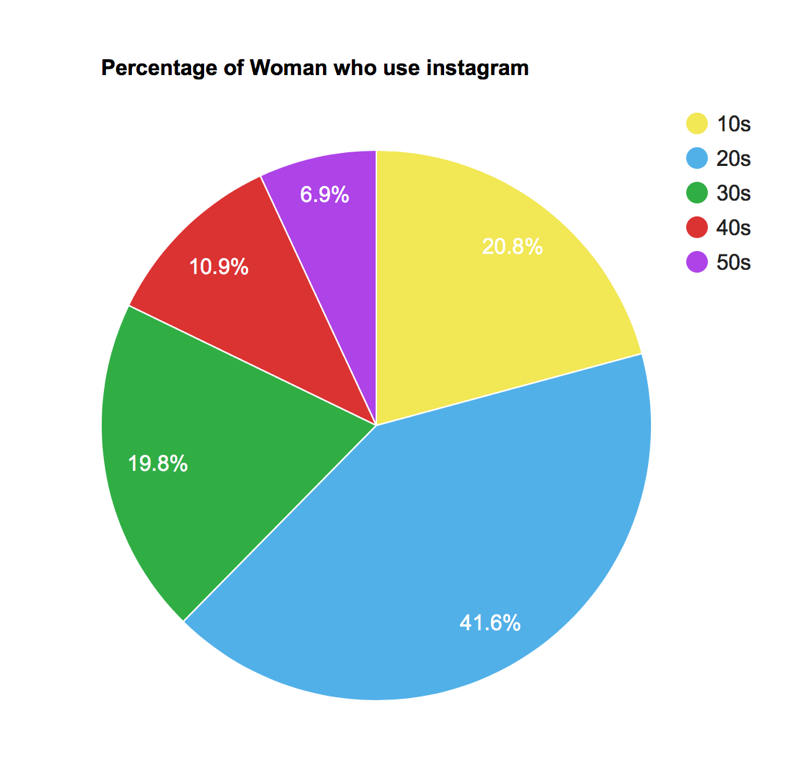
## 데이터 구성을 위하여 그래프 구축 시 테이블 형태의 자료구조를 지원한다. 데이터 테이블에 배열의 형태로 순차적으로 추가하거나, 데이터 테이블의 특정 행과 열에 값을 추가하는 방식 2가지 중 선택해 사용 할 수 있다.



(Google Charts를 이용한 Horizontal bar chart 1)



(Google Charts를 이용한 Grouped bar chart 1)



(Google Charts를 이용한 pie chart 1)

## 장점

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

## 다른 차트 라이브러리에 비해 종류가 다양하다.

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

## 단점

## 차트의 종류가 많다고는 하지만, 원하는 디자인의 차트를 구현하기 어렵다.

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

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

## **Chart.js**

## Chart.js 는 디자이너와 개발자들을 위해 쉽고 빠른 차트를 제공하는 HTML5 Canvas 방식의 오픈소스 라이브러리이다. (라이선스: MIT License)

## D3.js 는

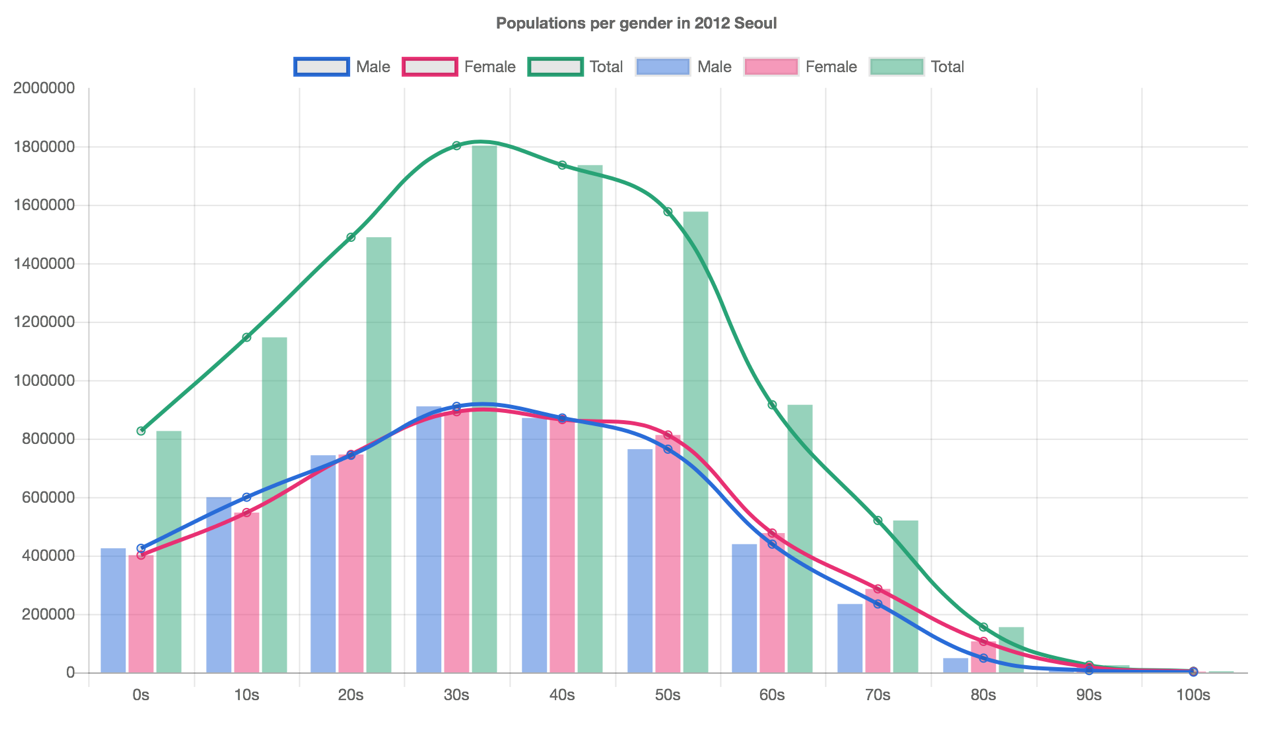
## Github 에서 chart.js 최신 버전을 다운받아 사용하거나, 스크립트 주소를 html 코드안에 넣어줌으로써 바로 사용가능하다.

## 다양한 데이터 시각화가 아닌 오로지 차트만을 위해 만들어진 라이브러리이다. Line chart, Bar chart, Doughnut & Pie chart, Polar Area chart, Bubble chart, Scatter chart, Area chart, Mixed chart 의 총 8가지 종류의 차트를 제공하며, type 값에 속성을 적용함으로써 사용할 수 있다.

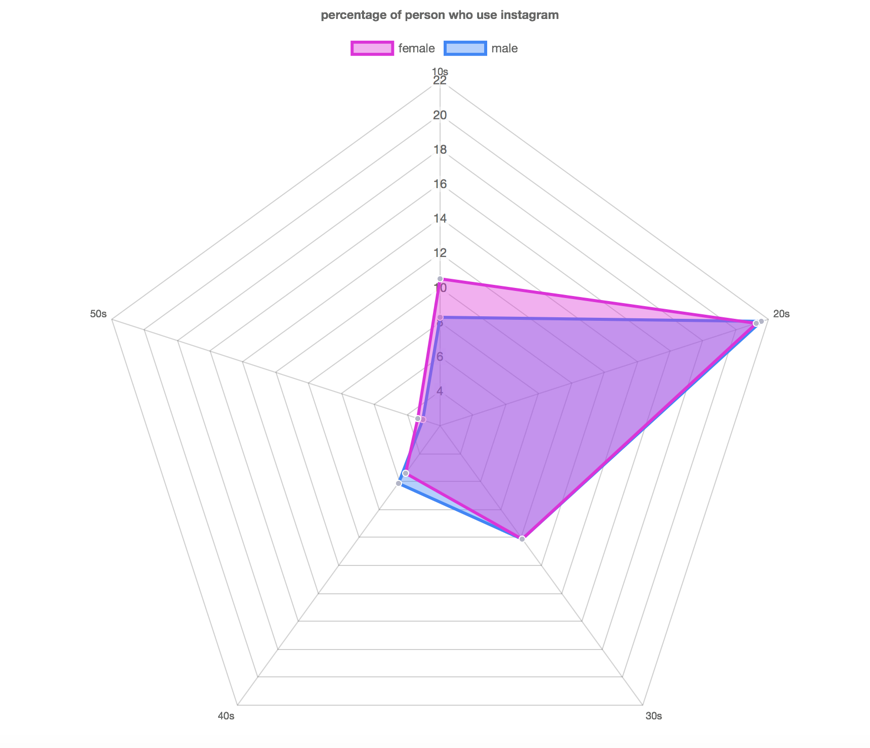
## 8가지 종류를 차트를 제공하지만, 딱 8개가 아닌 그 안에서 여러 종류의 차트들을 제공한다. 예를 들면 bar chart 같은 경우에는, 기본적인 차트부터 수평차트(horizontal chart), Stacked chart, Multi axis chart 등이 있고, line chart에는 가장 기본적인 차트, Stepped chart, Interpolation chart, Point style chart, 그리고 area chart 에는 Boundary chart, Dataset chart, Radar chart등이 존재한다.

## 

( Chart.js를 이용한 grouped bar chart 1)



(Chart.js를 이용한 Mixed chart 1)



(Chart.js를 이용한 radar chart 1)

## 장점

## 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.

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

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

## 다양한 색상 적용이 가능하다. Html5 색상코드를 그대로 사용가능하며, rgba 형태의 색상도 제공한다.

## 각 데이터가 의미하는게 무엇인지 title밑에 자동으로 명시해준다.

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

## 단점

## 차트의 종류가 너무 제한적이다.

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

## **D3.js**

## D3란 Data-Driven Documents 를 의미하는 단어로, 웹 브라우저 상에서 동적이고 인터렉티브한 시각화를 위한 자바스크립트 라이브러리이며, canvas 가 아니라 svg 를 사용한다.

## XML, CSV, JSON 등 다양한 데이터 형식을 취급한다.

## 

(D3.js를 이용한 pie chart 1)

## 장점

## 디자이너가 상상하는 거의 모든 것을 생성할 수 있다. 이 점에서 D3는 아주 강력하기 때문에 많은 사람들이 데이터 시각화로 d3.js 를 사용한다.

## jQuery, CSS에서의 셀렉터를 사용하여 DOM 요소들을 선택하고 변경하기 때문에, 직접적인 자바스크립트를 사용하는 것보다 쉽다.

## 단점

## 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.

## D3의 일부는 예전 브라우저가 지원되지 않으므로, 옛날 버전을 지원하기 위해서는 Sizzle 또는 Raphael등의 라이브러리를 이용한 추가적인 코딩이 필요할 수 있다.

## Google Charts나 Chart.js로 만든 차트와 같은 결과물을 D3.js 를 이용해 만들기 위해선 코드량이 최소 몇배 이상 늘어난다.

## 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 svg에서 빼줌으로써 차트의 높이를 바로 설정해야한다.

## 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.

## Google Charts나 Chart.js와 달리 결과로 나오는 차트에서 각 색상 별 차트가 의미하는게 무엇인지 설정해주지 않으면 차트 옆에 따로 표시되지 않는다.