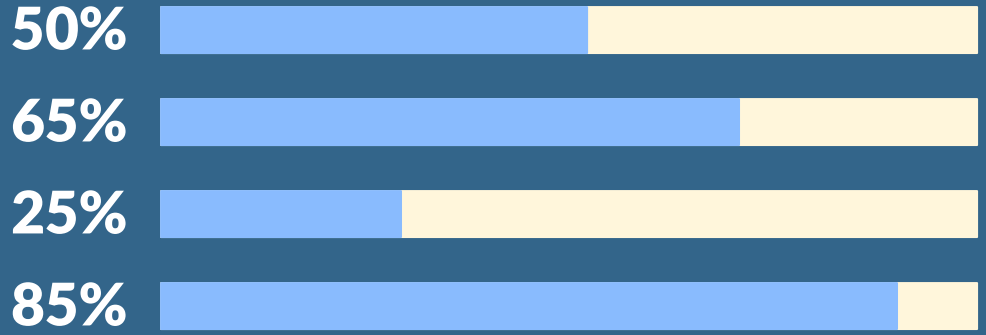


Data Visualization

using JavaScript

Eva Peso Adán
Raimon Mejías Hernández

10/4/2023

About us

Raimon Mejías Hernández

alu0101390161@ull.edu.es

raimon.mejias.35@ull.edu.es

Eva Peso Adán

alu0101398037@ull.edu.es

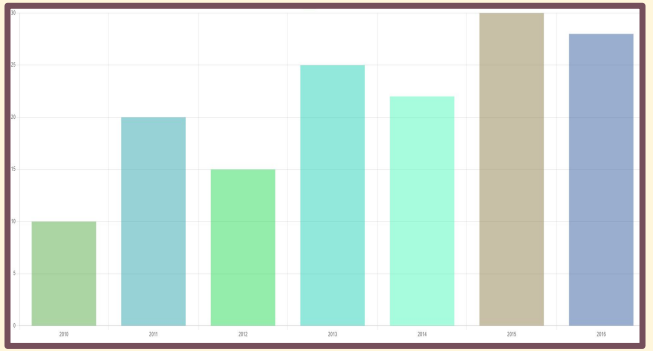
peso.adan.05@ull.edu.es

Content

What is Data Visualization?

Data visualization is a means to represent data and information in some form of graphical format.

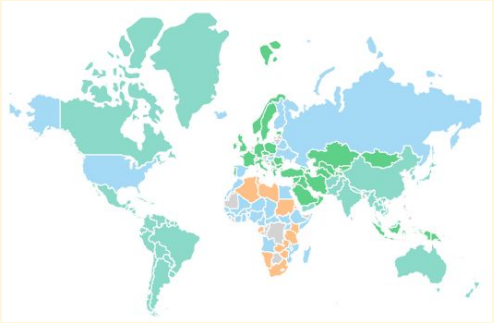
It is a particularly efficient way of communicating when the data or information is numerous.



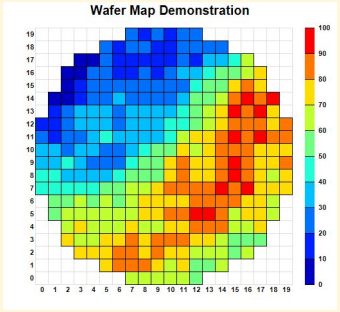
This chart was made using Chart.js

What is Data Visualization?

More information



There are not only charts, but also infographics and plots



It can be considered as a mapping between the original data and graphic elements.

It has to meet two conditions:

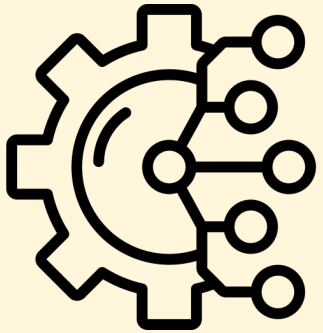
1-) The elements vary according to the data.

2-) It has to be understandable by humans.

What is Data Visualization?

But, why is so important?

It is increasingly applied as a critical component in:



Data analysis is an indispensable part of all applied research and problem solving in industry.

What is Data Visualization?

Summarizing

"... Allow users to see, explore, and understand large amounts of information at once. Information visualization focused on the creation of approaches for conveying abstract information in intuitive ways."

- Jmas J. Thomas & Kristin A. Cook -

"Main goal of data visualization is to communicate information clearly and effectively... It doesn't mean that data visualization needs to look boring to be functional or extremely sophisticated to look beautiful..."

- Vitaly Friedman -

Data Visualization libraries

Data visualization in JavaScript

JavaScript offers many libraries for visualizing data, creating charts and graphs, made in 2D or 3D, with animations, dynamic updates and many more.

Chart.js

Cube

Taucharts

D3.js

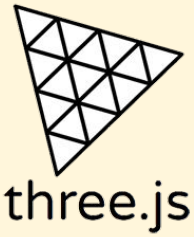
Echarts

Data Visualization libraries

There are many (too many, for real)

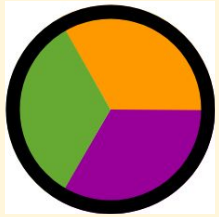



Chartkick



three.js


Raphaël




Recharts

VICTORY

AnyChart

Data Visualization libraries

Which charting library to use?

To choose a charting library you have to ask yourself some questions:

1-) What type of chart you will use?

2-) How fast do you want it to be?

3-) How is the learning curve?

4-) Does it have to be open-source?

5-) Do you really need a charting Library?

6-) Which frameworks does the library support?

Data Visualization libraries

Choosing Chart.js

"For a JS developer, the ability to visualize data is just as valuable as making interactive Web pages. Especially since the two often go in pairs" - Jakub Majorek -

Getting Data from JSON files

Before we start making our charts

A very simple way to read a JSON file is to put the data inside a variable with Javascript, and then import said variable:

```
import { VARIABLE } from '<PATH>.js';
```

Example

Getting Data from JSON files

Before we start making our charts

Another way is to use the following:

```
import * as <IDENTIFIER> from '<PATH>.json';
```

And then including in tsconfig.json the following line:

```
"resolveJsonModule": true
```

Getting Data from JSON files

Translating from .csv to .json

There are many CSV-to-JSON translators in the Internet.
The one used here is:

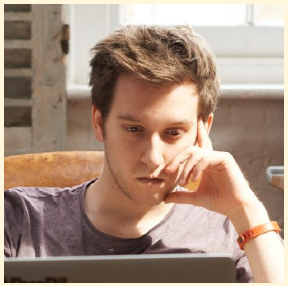
Turnkey CSV importer

Basics of Chart.js

What is Chart.js?

Chart.js

- Charting library for JavaScript
- Created and released in 2013
- Easy learning curve
- Works directly with canvas
- Pretty fast
- 7 officially supported Chart types



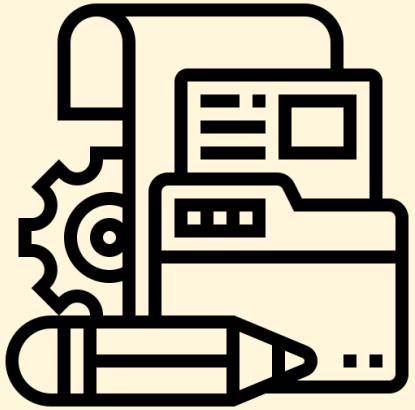
Created by
Nick Downie

Basics of Chart.js

Why use Chart.js?

Very popular library

Open Source



Active community
Good documentation

Basics of Chart.js

Installation

To use Chart.js in your TypeScript project you have two simple options:

1-) Add the following line in the head of the html file that will show the charts

```
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
```

2-) Install the library using npm

Example

```
npm install chart.js
```

```
npm install @types/chart.js
```

Basics of Chart.js

Installation

Remember!

If you install the library using only npm, you have to write the following lines in your script to use chart.js:

```
import { Chart, registerables } from '<PATH>/node_modules/chart.js/dist/chart.js';
Chart.register(...registerables);
```

Examples

Basics of Chart.js

Chart construction

Every chart created with Chart.js has the same syntax:

```
new Chart(canvasOrContext, chartConfiguration)
```

Where:

```
canvasOrContext: HTMLCanvasElement | CanvasRenderingContext2D
```

```
chartConfiguration: ChartConfiguration
```

Basics of Chart.js

Chart construction 2

Detecting when the canvas size changes is not done directly from the canvas element.

Chart.js use the parent element of the Canvas to do so.

```
<div class="chart-container">
    <canvas id="chart"></canvas>
</div>
```

Basics of Chart.js

Chart construction 3

The style of the canvas can be placed in the same div class mark but it is a good practice to put it in a .css file.

```
.chart-container {
  position: relative;
  height:720px;
  width:100%; # px, % or v(h|w)
}
```

Basics of Chart.js

Chart configuration

The configuration is used to change how the chart behaves.

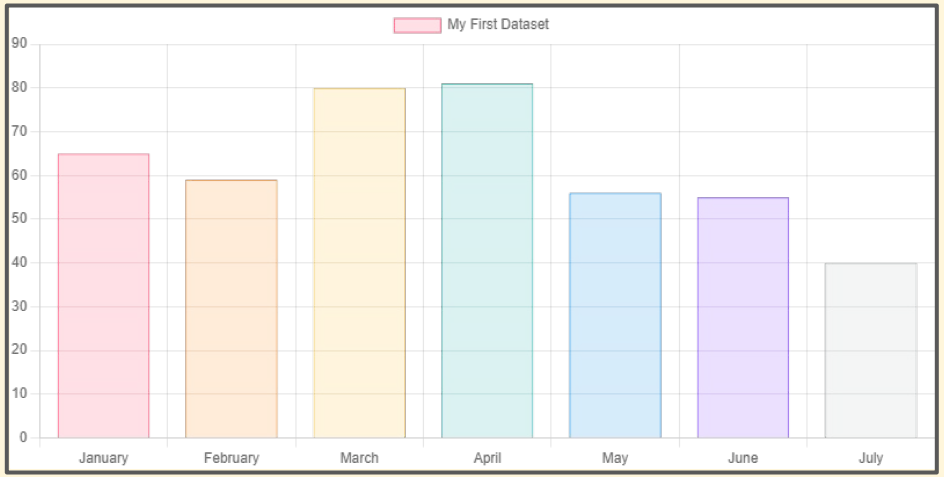
```
const chartConfiguration = {
  type: '<chartType>',
  data: {},  # Object containing the information about to be displayed
  options: {},  # Object containing the colors, title, legend, etc.
  plugins: []  # User created objects that extend the functionality
}
```

Basics of Chart.js

Bar Chart

A bar chart provides a way of showing data values represented as vertical bars.

It is used to show the trend of the data, and to compare them.



Bar Chart example

Basics of Chart.js

Bar Chart Configuration - Data

```
const barData = {
   labels: string[]
      <names that are going to show in the Y axis>,
   datasets: ChartData[],
};
```

Basics of Chart.js

Bar Chart Configuration - Data

```
const dataSet: ChartData<optional> = {
    label: string <name that will show in the legend>,
    data: Object[]|number[]|string[],
    backgroundColor: Color[],
    borderColor: Color[],
    borderWidth: number,
    barThickness: number
};
```

Basics of Chart.js

Bar Chart Configuration - options

```
const barOptions: ChartOptions = {
    indexAxis: ('y'|'x'),
    scales: {
        (y|x): {
            stacked: boolean, #allows stack Bar charts
            beginAtZero: boolean,
        # the axis start at 0 and not at the min value
};
```

Basics of Chart.js

Bar Chart Creation

```
const configuration: ChartConfiguration = {
   type: 'bar',
   data: barData,
   options:barOptions
};
```

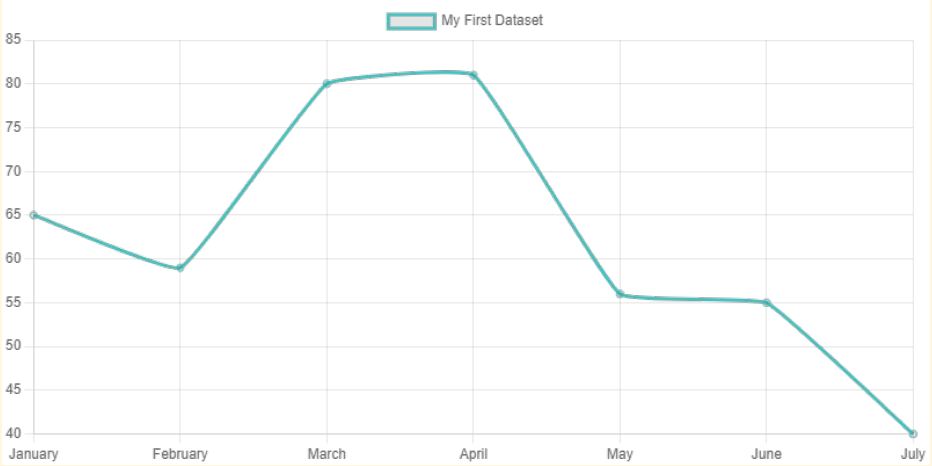
```
new Chart(<Canvas>, configuration);
```

Basics of Chart.js

Line Chart

A line chart is a way of plotting data points on a line.

It is used to show the trend of the data, and to compare them.



Line Chart example

Basics of Chart.js

Line Chart Configuration - Data

```
const lineData = {
   labels: string[]
      <names that are going to show in the Y axis>,
   datasets: ChartData[],
};
```

Basics of Chart.js

Line Chart Configuration - Data

```
const dataSet: ChartData<optional> = {
    label: string <name that will show in the legend>,
    data: object|object[]|number[]|string[],
    borderColor: Color, # line color
    borderWidth: number, # thickness of the line
    tension: [0-1] # can be more, but is discouraged
    fill: boolean # fill the area under the line
    backgroundColor: Color, # area under the line color
};
```

Basics of Chart.js

Line Chart Configuration - options

```
const lineOptions: ChartOptions = {
    indexAxis: ('y'|'x'),
    # change the orientation of the chart
    scales: {
        (y|x): {
            stacked: boolean, # allows stack Line charts
            beginAtZero: boolean,
        # the axis start at 0 and not at the min value
};
```

Basics of Chart.js

Line Chart Creation

```
const configuration: ChartConfiguration = {
   type: 'line',
   data: lineData,
   options: lineOptions
};
```

```
new Chart(<Canvas>, configuration);
```

Basics of Chart.js

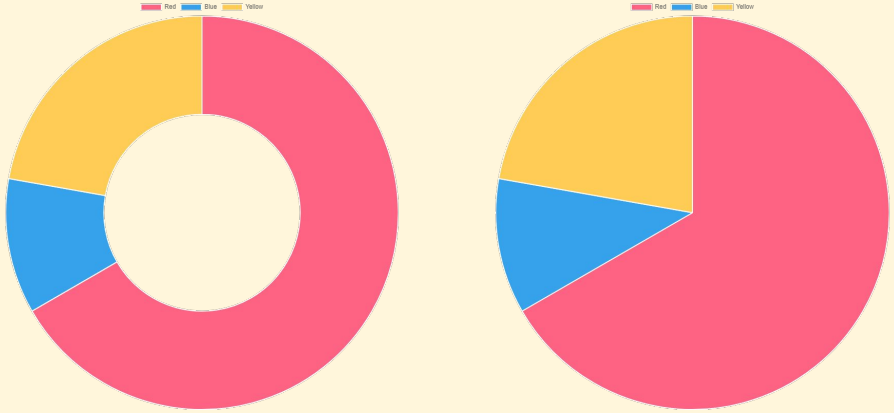
Doughnut and Pie Charts

Are probably the most commonly used charts.

They are divided into segments.

The arc of each segment shows the proportional value of each piece of data.

They are the same type of chart



Pie Charts example

Basics of Chart.js

Pie Chart Configuration - Data

```
const pieData = {
   labels: string[]
      <Names that will be show above the pie chart>,
   datasets: ChartData[],
};
```

Basics of Chart.js

Pie Chart Configuration - Data

```
const dataSet: ChartData<optional> = {
    label: string <Name that will show in the legend>,
    data: number[], # Only accept numbers
    borderColor: Color,
    borderWidth: number, # Thickness of the border
    backgroundColor: Color[], # Color of each segment
    hoverOffset: number # Segment animation
};
```

Basics of Chart.js

Pie Chart Configuration - Options

```
const pieOptions: ChartOptions = {
    animation:
        animateScale: boolean
    # enable a special animation of scaling the pie
};
```

Basics of Chart.js

Pie Chart Creation

```
const configuration: ChartConfiguration = {
   type: ('doughnut'|'pie'),
   data: pieData,
   options: pieOptions
};
```

```
new Chart(<Canvas>, configuration);
```

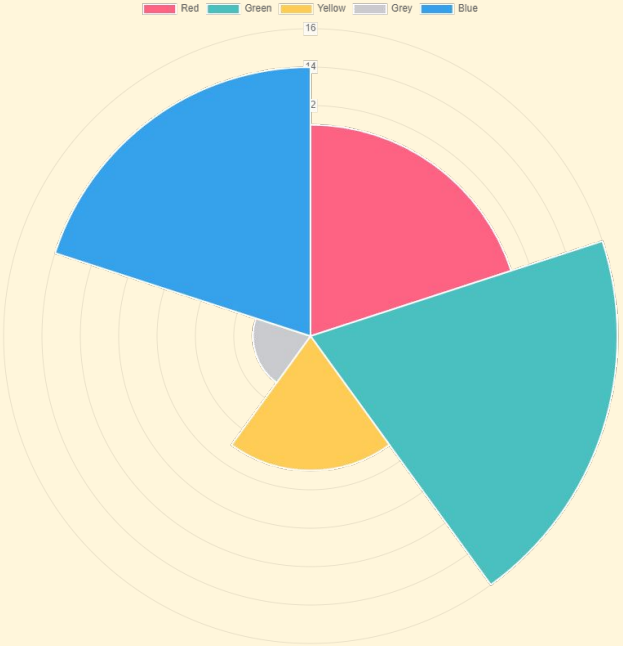
Basics of Chart.js

Polar Area Charts

Polar area charts are similar to pie charts.

Each segment has the same angle but the radius of the segment differs depending on the value.

They are built in the same way as pie charts.



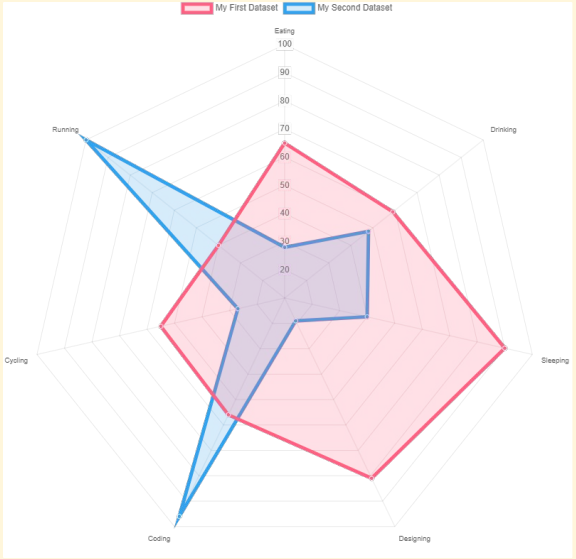
Polar Area example

Basics of Chart.js

Radar Chart

A radar chart is a way of showing multiple data points and the variation between them.

They are often useful for comparing the points of two or more different data sets.



Radar example

Basics of Chart.js

Radar Chart Configuration - Data

```
const radarData = {
   labels: string[]
   <Names that will be show above the radar chart>,
   datasets: ChartData[],
};
```

Basics of Chart.js

Radar Chart Configuration - Data

```
const dataSet: ChartData<optional> = {
    label: string <name that will show in the legend>,
    data: number[],
    borderColor: Color, # line color
    borderWidth: number, # thickness of the line
    fill: boolean, # fill the area under the line
    tension: [0-1], # can be more, but is discouraged
    backgroundColor: Color, # area under the line color
    pointBackgroundColor: Color,
    pointBorderColor: Color,
};
```

Basics of Chart.js

Radar Chart Configuration - Options

```
const radarOptions: ChartOptions = {
    # Here you can set options for all datasets in
    the chart, this can be done with all types of
    charts
};
```

Basics of Chart.js

Pie Chart Creation

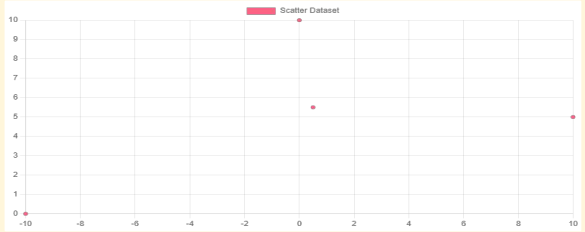
```
const configuration: ChartConfiguration = {
   type: 'radar',
   data: radarData,
   options: radarOptions
};
```

```
new Chart(<Canvas>, configuration);
```

Basics of Chart.js

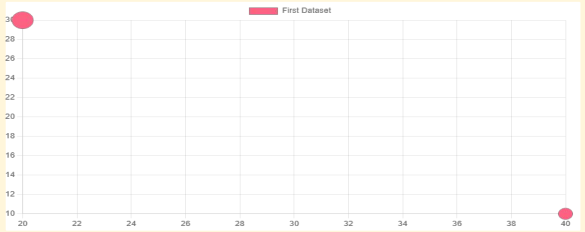
Bubble and Scatter Charts

Scatter charts are based on basic line charts with the x axis changed to a linear axis.



Scatter example

A bubble chart is used to display three dimensions of data at the same time. The third dimension is the size of the bubble



Bubble example

Basics of Chart.js

Bubble Chart Configuration - Data

```
const bubbleData = {
   labels: string[]
   <Names that will be show above the bubblechart>,
   datasets: ChartData[],
};
```

Basics of Chart.js

Bubble Chart Configuration - Data

```
const dataSet: ChartData<optional> = {
    label: string <name that will show in the legend>,
    data: { x: number, y: number, [r:number]},
    # the r attribute is only in bubble chart
    backgroundColor: Color, # color inside the circle
    borderColor: Color, # the circle border color
    pointStyle: string, # change the shape of the circle
};
```

Basics of Chart.js

Bubble Chart Creation

```
const configuration: ChartConfiguration = {
   type: ('bubble'|'scatter'),
   data: bubbleData,
   options: bubbleOptions
};
```

```
new Chart(<Canvas>, configuration);
```

Basics of Chart.js

Mixed Charts

It is possible to create mixed charts that are a combination of two or more different chart types.

Mixed Charts Example

Basics of Chart.js

And much more...

there are many topics about Chart.js that are not covered in this presentation. This was only the basic.

There are more things like:

- **Different types of Axes (Cartesian, Radial, Labeling).**
- **OnClick and onHover Events.**
- **Animations.**
- **Configuration of Title and Legends.**
- **Creating your own Charts.**
- **etc.**

Example

Questions?

Our Contacts

Raimon Mejías Hernández

alu0101390161@ull.edu.es

raimon.mejias.35@ull.edu.es

Eva Peso Adán

alu0101398037@ull.edu.es

peso.adan.05@ull.edu.es

References

Data and Information Visualization - Wikipedia

Top 15 JS Data Visualization libraries- Software Testing Help

19 Best JS Data Visualization libraries - Monterail

Data Visualization tools - Awesome cube

Chart.js - API documentation