
IIC2026

Visualización de Información

— Hernán F. Valdivieso López —
(2022 - 2 / Clase 11)

Antes de empezar... Revisión de contenidos (RC)

1. Hoy no tendremos ninguna actividad, pero no dejen que se acumulen las que ya tenemos:
 - a. Bonus daltonismo.
 - b. Cuestionario de abstracción de tareas.
 - c. Inventar pregunta de Verdadero/Falso.

Temas de la clase - Utilidades D3 I

1. Cargar datos de un archivo o enlace.
2. Uso de escalas.
3. Agregar ejes a la visualización.

Modalidad de esta clase

Híbrido entre teoría y programación.

1. Para cada tema se explicará lo que deseamos hacer junto con algunos de los comandos más importantes.
2. Luego se pasará a la etapa de programación donde haremos uso de dichos comandos.
3. Inspirándose en [Art Attack](#), se subió el código resultante al que “idealmente” llegaremos al final de la clase. A diferencia del código que haremos en clase, estos archivos además incluyen comentarios de casi cada línea del código.

**Cargar datos de un
archivo o enlace**

Cargar datos de un archivo o enlace

Hasta el momento hacíamos

```
// Creamos una lista inicial de datos y llamamos a joinDeDatos
```

```
const datos = [10, 20, 30, 40];
```

```
joinDeDatos(datos);
```

- Si necesitamos 100 datos, ¿escribimos todos estos en el código? ¿Y si son 1 millón? ¿Y si los datos están en la nube? 🤔
- D3 provee de funciones para cargar archivos locales o desde un enlace.

Cargar datos de un archivo o enlace - CSV y JSON

CSV (*Comma-Separated Values*): Archivo de texto en el cual los caracteres están separados por comas, haciendo una especie de tabla en filas y columnas. La extensión de sus archivos es `.csv`.

```
id,anime,rating
1,overlord,4.7
2,sakura card captor,4.8
3,detective conan,4.7
```

JSON (*JavaScript Object Notation*): Archivo de texto que registra los datos con el formato de un objeto de Javascript.

```
[
  {id: 1, anime: "overlord", rating: 4.7},
  {id: 2, anime: "sakura card captor", rating: 4.8},
  {id: 3, anime: "detective conan", rating: 4.7}
]
```

Cargar datos de un archivo o enlace - CSV y JSON

`d3.csv(url_o_path, funcion_de_parseo)`

- `url_o_path`: string con la ruta o link al archivo.
- `funcion_de_parseo`: función que se ejecuta por cada fila archivo y se encarga de procesar dicha fila. Este parámetro es **opcional**.

Ejemplo de un archivo llamado `datos.csv`.

```
id,anime,rating
1,overlord,4.7
2,sakura card captor,4.8
3,detective conan,4.7
```

```
d3.csv("datos.csv")
```

```
d3.csv("datos.csv", d=> {anime: d.anime, rating: +d.rating})
```


Cargar datos de un archivo o enlace - CSV y JSON

`d3.json(url_o_path)`

- `url_o_path`: string con la ruta o link al archivo.

Esta función ya carga los datos con el tipo que vienen en el `.json`.

Ejemplo de un archivo llamado `datos.json`.

```
[  
  {id: 1, anime: "overlord", rating: 4.7},  
  {id: 2, anime: "sakura card captor", rating: 4.8},  
  {id: 3, anime: "detective conan", rating: 4.7}  
]
```

```
d3.json("datos.json")
```

Cargar datos de un archivo o enlace - CSV y JSON

return de `d3.csv()` y `d3.json()`

- Ambas funciones retornan una **promesa**. ¿Qué es una promesa?
 - Es un objeto en JS que representa la terminación o el fracaso de una operación **asíncrona**.
 - En palabras sencillas: D3 hace que Javascript lea de forma paralela el archivo o enlace. Cuando la promesa termina, tendremos los datos o un objeto tipo error indicando que falló.
- Usamos `.then(funcion)` y `.catch(funcion)` para acceder al resultado de la promesa.

// Estamos usando function arrows en este caso.

```
d3.csv("datos.csv").then(datos => {joinDeDatos(datos)})  
.catch(error => {...})
```

Cargar datos de un archivo o enlace - CSV y JSON

Vamos al código  

Cargar datos de un archivo o enlace - extra

- D3 también provee de funciones para leer otros tipos de archivos.
- Enlace recomendado: [Reading in Data - Learn JS Data](#)

Escalas

Escalas

Son funciones en Javascript que:

- Reciben un dato (usualmente un número, fecha o categoría)
- Retornan un valor (por ejemplo una coordenada, un color, un número, etc.)

Permiten mapear un dato de cierto dominio a un nuevo dominio. Por ejemplo, mapear números entre 50 y 100 a números entre 0 y 12.

En esta clase veremos 2 tipos de escalas:

- `d3.scaleLinear` para datos numéricos.
- `d3.scaleBand` para datos categóricos.

Escalas - scaleLinear

```
const myScale = d3.scaleLinear().domain([0, 10]).range([0, 600]);  
  
myScale(0);      // returns 0  
myScale(0.1);    // returns 6  
myScale(2);      // returns 120  
myScale(3);      // returns 180  
...  
myScale(10);     // returns 600
```

- Si vemos scaleLinear como una función $f(x)$
- `domain` recibe una lista de números que representan el rango de los x .
- `range` recibe una lista de números que representan el rango de los $f(x)$.

Escalas - scaleBand

```
let bandScale = d3.scaleBand().domain(['Mon', 'Tue', 'Wed', 'Thu', 'Fri'])  
    .rangeRound([0, 200]);
```

```
bandScale('Mon'); // returns 0  
bandScale('Tue'); // returns 40  
bandScale('Fri'); // returns 160  
bandScale('uwu'); // returns undefined
```

- domain recibe una lista de datos. Estos representan todos los posibles valores de x.
- rangeRound recibe una lista de números que representan el rango de valores de f(x). Además, rangeRound redondea el valor de salida. Si usamos sólo range(), pueden salir valores decimales.

Escalas

Vamos al código  

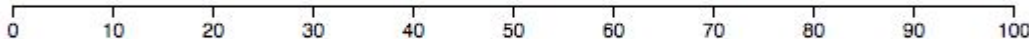
Escalas - extra

- D3 incluye muchas más escalas y parámetros para personalizar dichas escalas.
- Enlaces recomendados:
 - [Scale functions | D3 in Depth](#)
 - [d3.scaleBand](#)
 - [d3.scaleLinear](#)

Ejes

Ejes

D3 provee de funciones para dibujar automáticamente los ejes en una visualización.

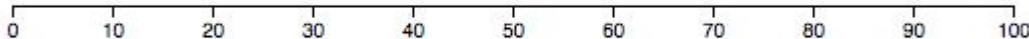


Solo se requiere:

1. Un elemento SVG para contener al eje. Por ejemplo, un contenedor (`<g>`).
2. Una escala de D3 con su dominio y rango ya definidos.
3. Asegurarnos que exista espacio para el eje.

Ejes

D3 provee de funciones para dibujar automáticamente los ejes en una visualización.



Solo se requiere:

1. Un elemento SVG para contener al eje. Por ejemplo, **un contenedor (<g>)**.
2. Una escala de D3 con su dominio y rango ya definidos.
3. **Asegurarnos que exista espacio para el eje.**

Contenedores <g>

<g> </g> es un contenedor de SVG para agrupar elementos.

Visualmente no es nada, pero nos permite ahorrar código.

```
<svg width="200" height="200">  
  <g stroke="green" fill="white" stroke-width="5">  
    <circle cx="25" cy="25" r="15" />  
    <circle cx="40" cy="25" r="15" />  
    <circle cx="55" cy="25" r="15" />  
    <circle cx="70" cy="25" r="15" />  
  </g>  
</svg>
```

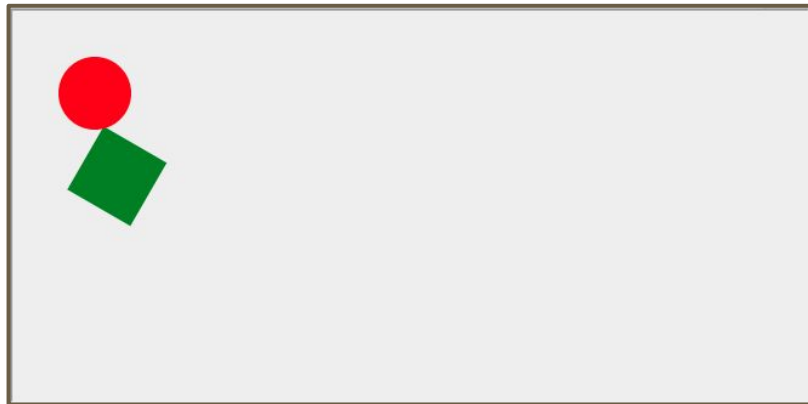


Contenedores <g>

<g> </g> es un contenedor de SVG para agrupar elementos.

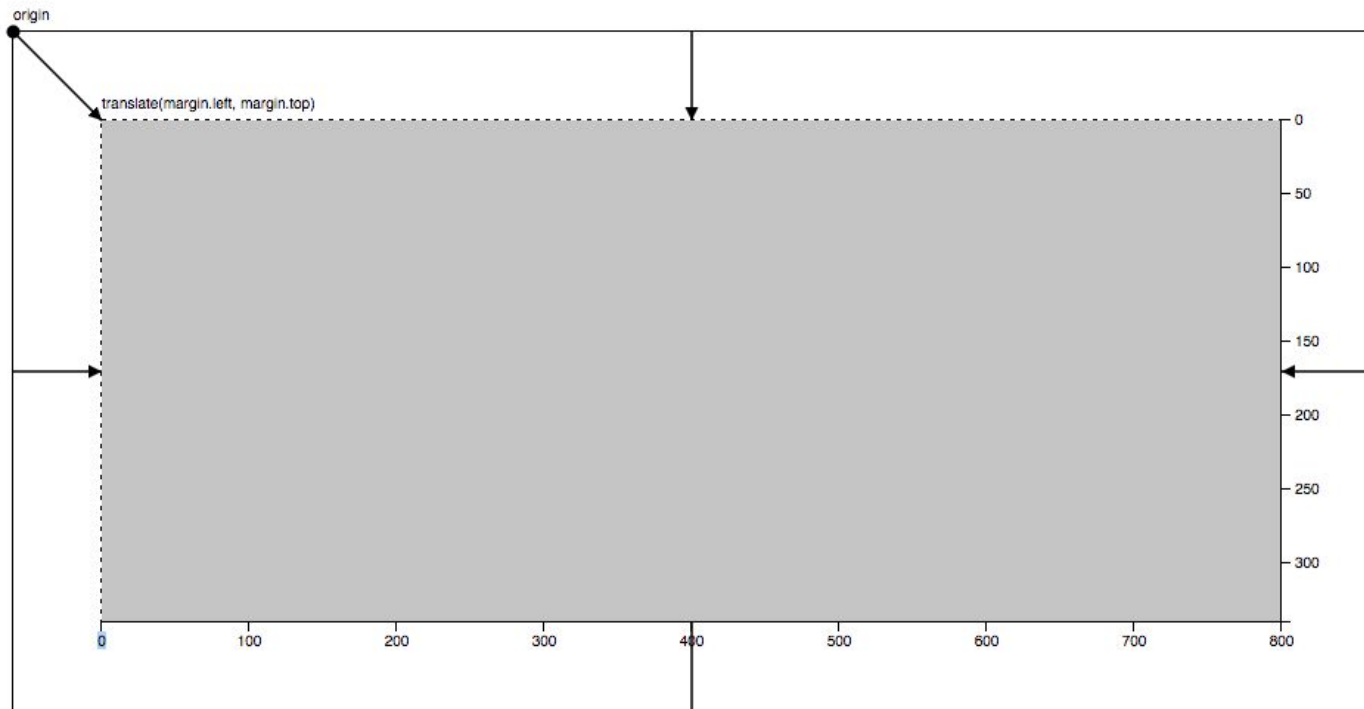
Visualmente no es nada, pero nos permite ahorrar código.

```
<svg width="300" height="300">  
  <g transform="translate(60,60) rotate(30)">  
    <rect x="20" y="20" width="60" height="60" fill="green">  
    </rect>  
    <circle cx="0" cy="0" r="30" fill="red" />  
  </g>  
</svg>
```



Márgenes

Es muy común definir márgenes en el SVG para que la visualización no esté tocando la orilla y así tener espacio para incluir los ejes.

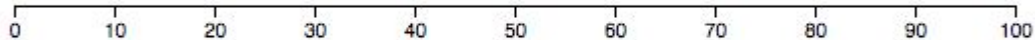


Ejes

```
<svg width="600" height="100">  
  <g transform="translate(20, 50)"></g>  
</svg>
```

```
let scale = d3.scaleLinear()  
  .domain([0, 100])  
  .range([0, 500]);  
  
let axis = d3.axisBottom(scale);  
d3.select('g').call(axis);
```

1. Definimos una escala.
2. Definimos un objeto eje con dicha escala.
3. Seleccionamos un `<g>` del SVG y hacemos `call(axis)` para pedirle al objeto eje que llene el elemento `<g>` con lo necesario para dibujar el eje.



Vamos al código  

Ejes - extra

- Se pueden ocupar diferentes tipos de escalas para definir un eje.
 - [Drawing axis in d3.js](#)
- También se pueden personalizar los ejes
 - [axis.ticks / D3 / Observable](#)

Próximos eventos

- Hoy en la noche compartiré la autoevaluación de la T01. Tendrán 1 semana para responder.

Próxima clase

- Veremos visualización de datos tabulares (teoría).
- **Actividad RC:** Tendremos Quizizz (similar a kahoot) al inicio de la clase. Aprox 25 preguntas y se pedirá un porcentaje de ellas correctas para obtener el puntaje RC.
 - Al inicio de la clase publicaré por Discord el link al Quizizz para que puedan participar de forma remota. Queda a su responsabilidad conectarse a tiempo o llegar a la clase.
 - En la T02 incluiré un bonus para recuperar puntaje RC, por si alguno/a no logró participar en esta actividad o no consiguió el porcentaje solicitado.

IIC2026

Visualización de Información

— Hernán F. Valdivieso López —
(2022 - 2 / Clase 11)
