IIC2026
Visualización de Información

Hernán F. Valdivieso López
(2023 - 1 / Clase 04)

Temas de la clase - D3 y Data join 1

1. Introducción a D3.
2. Selecciones en D3.
    a. Comandos: `select` y `selectAll`.
    b. Comandos: `attr` y `style`.
    c. Method chaining.
    d. Comando: `append`.
3. Preámbulo data joins en D3.
    a. Crear elementos vinculados a datos (*enter*)

Introducción a D3.js

Introducción a D3.js

- Su intención es utilizar HTML, CSS y SVG para crear visualizaciones.
- Apareció en un momento en que esto no era común para herramientas de visualización.
- Escrita por Mike Bostock.

Enlaces de interés:

- Artículo (paper) donde se propone D3.
- Página oficial D3.

Introducción a D3.js

- **NO** es una librería de visualización de alto nivel

```
const grafico_de_barra = crear_grafico_barra(datos); // ❌
grafico_de_barra.graficar(); // ❌
```

- AnyChart es un ejemplo de librería de alto nivel.
- Es una herramienta para **crear visualizaciones desde 0**.
  - Definir cada elemento del SVG.
  - Personalizar los elementos del SVG según los datos.
  - Definir las interacciones posibles.
  - Agregar leyendas a mano.
  - entre otros...

Introducción a D3.js

Utilizaremos las versiones 7 o 6 en el curso.

👀 👀 con recursos y ejemplos escritos en la versión 5 o menor que encuentren.

- 🤦‍♂️ No se aceptará trabajos con versiones diferente a la 6 o 7.
- 👨‍💻 Es su deber adaptar dichos recursos a la versión 6 o 7.

Introducción a D3.js

Cargar D3 en un archivo HTML - Desde URL

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Ejemplo con D3</title>
    <script src="https://d3js.org/d3.v7.min.js"></script>
  </head>
  <body>
    <script src='programa_4.js' charset='utf-8'></script>
  </body>
</html>
```

Introducción a D3.js

Cargar D3 en un archivo HTML - De forma local

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Ejemplo con D3</title>
    <script src="./ruta/en/computador/d3.v7.min.js"></script>
  </head>
  <body>
    <script src='programa_4.js' charset='utf-8'></script>
  </body>
</html>
```

Introducción a D3.js

```
const datos = [150, 256, 130, 0, 11, 420, 235];

// Encontrar el body en el DOM y agregar un elemento SVG. Luego definir ancho y largo
const svg = d3.select("body").append("svg");
svg.attr("width", 50 + datos.length * 100).attr("height", 500);

// Agregar “rect” y personalizar según los datos que tenemos
svg
  .selectAll("rect")
  .data(datos)
  .join("rect")
  .attr("width", 50)
  .attr("fill", "orange")
  .attr("height", (d) => d)
  .attr("x", (_, index) => 50 + index * 100);
```

Selecciones en D3

Selecciones en D3

Objetos en D3 que corresponden a una colección de elementos HTML.

`d3.selectAll("p")` → Una colección de todos los elementos con tag `<p>`.

2 formas de seleccionar:

- `select` → Selecciona el primer elemento de la búsqueda.
- `selectAll` → Selecciona todos los elementos de la búsqueda.

Selecciones en D3

Formas de seleccionar con `select`

- `d3.select("p")` → Por tag del elemento HTML.
- `d3.select(".class")` → Por la clase del elemento HTML.
- `d3.select("#id")` → Por el ID del elemento HTML.

Se aplica de igual forma al `selectAll`.

- `d3.selectAll("p")` → Por tag del elemento HTML.
- `d3.selectAll(".class")` → Por la clase del elemento HTML.
- `d3.selectAll("#id")` → Por el ID del elemento HTML.

Selecciones en D3 - Ejemplos

```
<body>
    <h1>Soy un título</h1>
    <p>Yo soy un párrafo</p>
    <p>¡Yo tambien :D!</p>
    <p class="uwu">¡Yo igual, pero tambien tengo clase</p>
    <p class="uwu">jejej yo tambien tengo clase</p>
    <h1 id="owo">Pero yo soy un título y tengo un ID único :P </h1>
</body>
```

Selecciones en D3 - Ejemplos

```
<body>
    <h1>Soy un título</h1>
    <p>Yo soy un párrafo</p>
    <p>¡Yo tambien :D!</p>
    <p class="uwu">¡Yo igual, pero tambien tengo clase</p>
    <p class="uwu">jejej yo tambien tengo clase</p>
    <h1 id="owo">Pero yo soy un título y tengo un ID único :P </h1>
</body>
```

```
d3.select("h1")
```

Selecciones en D3 - Ejemplos

```
<body>
    <h1>Soy un título</h1>
    <p>Yo soy un párrafo</p>
    <p>¡Yo tambien :D!</p>
    <p class="uwu">¡Yo igual, pero tambien tengo clase</p>
    <p class="uwu">jejej yo tambien tengo clase</p>
    <h1 id="owo">Pero yo soy un título y tengo un ID único :P </h1>
</body>
```

```
d3.select(".uwu")
```

Selecciones en D3 - Ejemplos

```
<body>
    <h1>Soy un título</h1>
    <p>Yo soy un párrafo</p>
    <p>¡Yo tambien :D!</p>
    <p class="uwu">¡Yo igual, pero tambien tengo clase</p>
    <p class="uwu">jejej yo tambien tengo clase</p>
    <h1 id="owo">Pero yo soy un título y tengo un ID único :P </h1>
</body>
```

```
d3.select("#owo")
```

Selecciones en D3 - Ejemplos

```
<body>
    <h1>Soy un título</h1>
    <p>Yo soy un párrafo</p>
    <p>¡Yo tambien :D!</p>
    <p class="uwu">¡Yo igual, pero tambien tengo clase</p>
    <p class="uwu">jejej yo tambien tengo clase</p>
    <h1 id="owo">Pero yo soy un título y tengo un ID único :P </h1>
</body>
```

```
d3.selectAll("h1")
```

Selecciones en D3 - Ejemplos

```
<body>
    <h1>Soy un título</h1>
    <p>Yo soy un párrafo</p>
    <p>¡Yo tambien :D!</p>
    <p class="uwu">¡Yo igual, pero tambien tengo clase</p>
    <p class="uwu">jejej yo tambien tengo clase</p>
    <h1 id="owo">Pero yo soy un título y tengo un ID único :P </h1>
</body>
```

```
d3.selectAll(".uwu")
```

Selecciones en D3 - Ejemplos

```
<body>
    <h1>Soy un título</h1>
    <p>Yo soy un párrafo</p>
    <p>¡Yo tambien :D!</p>
    <p class="uwu">¡Yo igual, pero tambien tengo clase</p>
    <p class="uwu">jejej yo tambien tengo clase</p>
    <h1 id="owo">Pero yo soy un título y tengo un ID único :P </h1>
</body>
```

```
d3.selectAll("#owo")
```

Selecciones en D3 - Ejemplos

```
<body>
    <h1>Soy un título</h1>
    <p>Yo soy un párrafo</p>
    <p>¡Yo tambien :D!</p>
    <p class="uwu">¡Yo igual, pero tambien tengo clase</p>
    <p class="uwu">jejej yo tambien tengo clase</p>
    <h1 id="owo">Pero yo soy un título y tengo un ID único :P </h1>
</body>
```

```
d3.selectAll("p")
```

Selecciones en D3 - Ejemplos

```
<body>
    <div>
        <h1>Soy un título</h1>
        <h1>Soy otro título</h1>
    </div>
    <div>
        <h1>Soy un título</h1>
        <h1>Soy otro título</h1>
    </div>
    <p>Yo soy un párrafo</p>
    <h1>Yo no!!!</h1>
    <h1>Soy un título fuera de un contenedor</h1>
</body>
```

Quiero los H1 dentro de los divs

```
selection.selectAll("h1") ????
```

❌ Esto también incluirá los últimos h1

Solución:

```
const selection = d3.selectAll("div")
selection.selectAll("h1")
```

✅ Primero seleccionamos los div. Luego seleccionamos los h1 dentro de esos divs.

Selecciones en D3 - Attr y style

seleccion.attr(atributo, valor)

Define un atributo y su valor a cada elemento de la selección. Si ese atributo ya existe, sobreescribe su valor.

```
<body>
    <p>Mi párrafo es un anime: Full Metal Alchemists</p>
    <p class="uwu">Pero mi párrafo es un anime con clase: Gintama</p>
</body>
```

Antes

```
const seleccion = d3.selectAll("p");
seleccion.attr("class", "nani!!!");
```

```
<body>
    <p class="nani!!!">Mi párrafo es un anime: Full Metal Alchemists</p>
    <p class="nani!!!">Pero mi párrafo es un anime con clase: Gintama</p>
</body>
```

Después

Aprendizaje random: nani es una expresión en japonés que se puede entender como qué!!! (es muy ocupada en los anime y mangas).

Selecciones en D3 - Attr y style

seleccion.style(propiedad, valor)

Agrega una propiedad de CSS y su valor a cada elemento de la selección. Si esa propiedad del CSS ya existe, sobreescribe su valor.

```
<body>
    <p>Full Metal Alchemists</p>
    <p class="uwu" style="color: red;">Gintama</p>
</body>
```

Antes

```
const seleccion = d3.selectAll("p");
seleccion.style("color", "orange");
```

```
<body>
    <p style="color: orange;">Full Metal Alchemists</p>
    <p class="uwu" style="color:orange;">Gintama</p>
</body>
```

Después

Selecciones en D3 - Funciones en Attr y style

seleccion.style(propiedad, valor) - seleccion.attr(atributo, valor)

En ambos casos, el valor puede ser una constante (como en los ejemplos anteriores) o una función.

```
<body>
    <p>Full Metal Alchemists</p>
    <p class="uwu" style="color: red;">Gintama</p>
</body>
```

Antes

```
const seleccion = d3.selectAll("p");
seleccion.style("fill", () => "orange");
```

```
<body>
    <p style="color: orange;">Full Metal Alchemists</p>
    <p class="uwu" style="color:orange;">Gintama</p>
</body>
```

Después

Selecciones en D3 - Funciones en Attr y style

seleccion.style(propiedad, valor) - seleccion.attr(atributo, valor)

En ambos casos, el valor puede ser una constante (como en los ejemplos anteriores) o una función.

```
<svg>
    <rect></rect>
    <rect></rect>
</svg>
```

Antes

```
const seleccion = d3.selectAll("rect");
seleccion.attr("x", (_, i, _) => i*100); // Segundo argumento → índice en la selección
```

```
<svg>
    <rect x="0"></rect>
    <rect x="100"></rect>
</svg>
```

Después

Selecciones en D3 - Method chaining

Métodos como `attr` y `style` retornan la misma selección.

```
const selection = d3.selectAll("rect");
selection.attr("x", 10);
selection.attr("y", 50);
selection.style("fill", "orange");
```

Es equivalente a:

```
d3.selectAll("rect").attr("x", 10).attr("y", 50).style("fill", "orange");
```

De forma más ordenada:

```
d3.selectAll("rect")
    .attr("x", 10)
    .attr("y", 50)
    .style("fill", "orange");
```

Selecciones en D3 - Append

seleccion.append(tag)

Crea un nuevo elemento dentro de cada elemento de la selección sobre la cual actúa con el tag definido en el método.

```
<body>
    <div></div>
    <div></div>
</body>
```

Antes

```
<body>
    <div>
        <h1></h1>
    </div>
    <div>
        <h1></h1>
    </div>
</body>
```

Después

```
const selection = d3.selectAll("div")
selection.append("h1")
```

Selecciones en D3 - Append

seleccion.append(tag)

Crea un nuevo elemento dentro de cada elemento de la selección sobre la cual actúa con el tag definido en el método.

```
<body>
    <div></div>
    <div></div>
</body>
```

Antes

```
<body>
    <div>
        <h1>uwu</h1>
    </div>
    <div>
        <h1>uwu</h1>
    </div>
</body>
```

Después

```
const selection = d3.selectAll("div")
selection.append("h1").text("uwu");
```

👀 **selection.text(texto)**

Agrega un texto dentro del elemento HTML. Notar que este texto no tiene un tag asociado.

Selecciones en D3 - Append

seleccion.append(tag)

Crea un nuevo elemento dentro de cada elemento de la selección sobre la cual actúa con el tag definido en el método.

```
<body>
    <div></div>
    <div></div>
</body>
```

Antes

```
<body>
    <div>
        <h1>uwu</h1>
        <h1>uwucito</h1>
    </div>
    <div>
        <h1>uwu</h1>
        <h1>uwucito</h1>
    </div>
</body>
```

Después

```
const selection = d3.selectAll("div")
selection.append("h1").text("uwu");
selection.append("h1").text("uwucito");
```

Selecciones en D3 - Append

seleccion.append(tag)

Crea un nuevo elemento dentro de cada elemento de la selección sobre la cual actúa con el tag definido en el método.

```
<body>
    <div></div>
    <div></div>
</body>
```

Antes

```
<body>
    <div>
        <h1 style="color:orange;">uwu</h1>
        <h1>uwucito</h1>
    </div>
    <div>
        <h1 style="color:orange;">uwu</h1>
        <h1>uwucito</h1>
    </div>
</body>
```

Después

```
const selection = d3.selectAll("div")
selection.append("h1").style("color", "orange").text("uwu");
selection.append("h1").text("uwucito");
```

Selecciones en D3 - Append - ejemplo 1

Vamos al código 👨‍💻 👩‍💻

Selecciones en D3 - Resumen de comandos

- **`d3.select(selector)`**: Selecciona el primer elemento según el `selector` indicado.
- **`d3.selectAll(selector)`**: Selecciona todos los elementos según el `selector` indicado.
- **`seleccion.style(propiedad, valor)`**: Agrega una `propiedad` de CSS junto con su `valor` a cada elemento de la selección.
- **`seleccion.attr(atributo, valor)`**: Agrega un `atributo` y su `valor` a cada elemento de la selección.
- **`seleccion.text(texto)`**: Agrega un `texto` dentro de cada elemento de la selección.
- **`seleccion.append(tag)`**: Agrega un nuevo elemento HTML (o svg) con el `tag` indicado dentro de cada elemento de la selección.

Data joins

Parte 1

Data Joins

Vincular datos a elementos del SVG.

Podemos generar un vínculo de marcas y canales con datos mediante código.

```
const datos = [23, 45, 120, 64]

const datos = [“uwu”, “awa”, “owo”]

const datos = [
 {title: "Spy x Family", year: 2019},
 {title: "Kaguya-sama: Love Is War", year: 2019},
 {title: "Deaimon", year: 2016},
];
```

```
<svg>
    <rect></rect>
    <text></text>
    <circle></circle>
    <path></path>
</svg>
```

Data Joins - Comando data

selection.data(lista_datos)

Comando para vincular una selección de elementos a una lista de datos

```
const datos = [23, 45, 120, 64]
selection.data(datos)
```

Ejemplo:

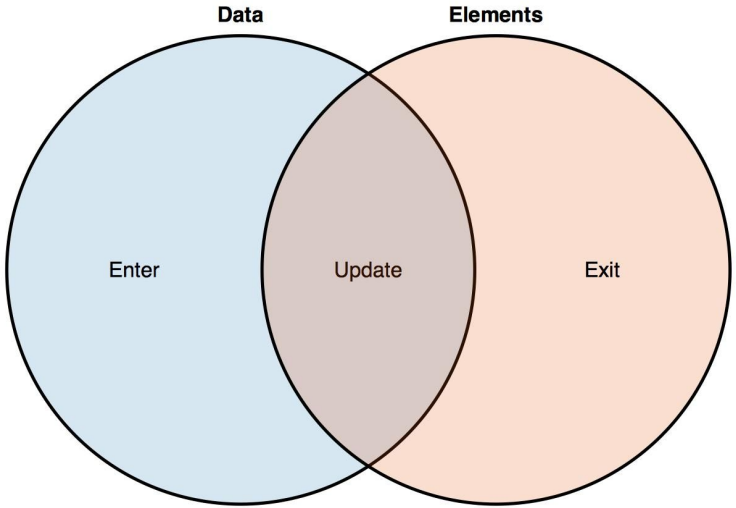
```
d3.selectAll("rect").data(datos)
```

Data Joins - Comando data

selection.data(lista_datos)

Cuando se ejecuta este comando, se generan **3 nuevas selecciones**:

- **Enter**: datos que no quedan asociados a ningún elemento.
- **Update**: datos asociados a algún elemento de la selección.
- **Exit**: elementos que no quedan asociados a ningún dato.

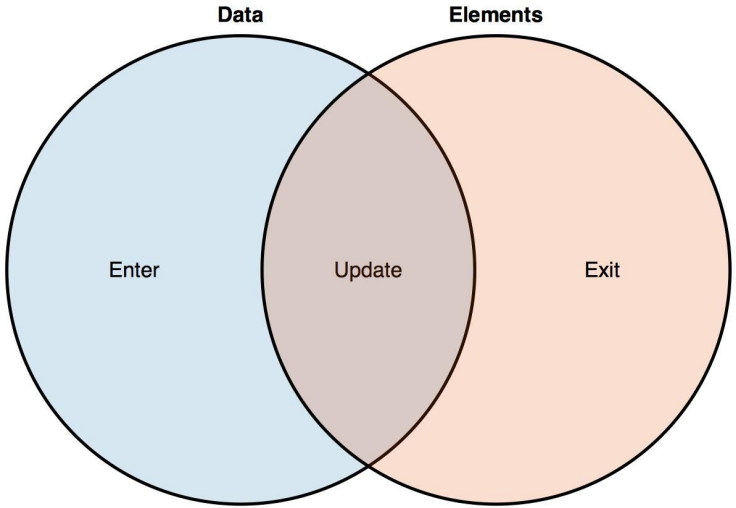


Data Joins - Comando data

selection.data(lista_datos)

Cuando se ejecuta este comando, se generan **3 nuevas selecciones**:

- **Enter**: datos que no quedan asociados a ningún elemento.
- **Update**: datos asociados a algún elemento de la selección.
- **Exit**: elementos que no quedan asociados a ningún dato.



Link recomendado para estudio: D3 Data Joins

Data Joins - Comando join

```
// Caso original
<svg id="vis" width="400" height="250">
</svg>
```

Agregar elementos vinculados a datos

```
const datos = [23, 45, 99, 64]
d3.select("#vis").selectAll("rect").data(datos)
```

¿Qué ocurre?

Visualmente nada. Por dentro, d3 entiende que hay 4 datos sin vincularse a nada

```
<svg id="vis" width="400" height="250">
 ?? <!-- 23 -->
 ?? <!-- 45 -->
 ?? <!-- 99 -->
 ?? <!-- 64 -->
</svg>
```

Data Joins - Comando join

```
// Caso original
<svg id="vis" width="400" height="250">
</svg>
```

Agregar elementos vinculados a datos

```
const datos = [23, 45, 99, 64]
d3.select("#vis").selectAll("rect").data(datos).join("rect");
```

¿Qué ocurre?

```
<svg id="vis" width="400" height="250">
 <rect></rect> <!-- 23 -->
 <rect></rect> <!-- 45 -->
 <rect></rect> <!-- 99 -->
 <rect></rect> <!-- 64 -->
</svg>
```

Data Joins - Comando join

```
// Caso original
<svg id="vis" width="400" height="250">
</svg>
```

Agregar elementos vinculados a datos

```
const datos = [23, 45, 99, 64]
d3.select("#vis").selectAll("rect").data(datos).join("rect").attr("height", 40);
```

¿Qué ocurre?

```
<svg id="vis" width="400" height="250">
 <rect height=”40”></rect> <!-- 23 -->
 <rect height=”40”></rect> <!-- 45 -->
 <rect height=”40”></rect> <!-- 99 -->
 <rect height=”40”></rect> <!-- 64 -->
</svg>
```

Data Joins - Comando join

```
// Caso original
<svg id="vis" width="400" height="250">
</svg>
```

Agregar elementos vinculados a datos

```
const datos = [23, 45, 99, 64]
d3.select("#vis").selectAll("rect").data(datos).join("rect").attr("height", 40)
  .attr("width", (dato, i, _) => dato);
```

¿Qué ocurre?

```
<svg id="vis" width="400" height="250">
 <rect height=”40” width=”23”></rect> <!-- 23 -->
 <rect height=”40” width=”45”></rect> <!-- 45 -->
 <rect height=”40” width=”99”></rect> <!-- 99 -->
 <rect height=”40” width=”64”></rect> <!-- 64 -->
</svg>
```

Data Joins - Comando join

```
// Caso original
<svg id="vis" width="400" height="250">
</svg>
```

Agregar elementos vinculados a datos

```
const datos = [23, 45, 99, 64]
d3.select("#vis").selectAll("rect").data(datos).join("rect").attr("height", 40)
  .attr("width", (dato, i, _) => dato)
  .attr("y", (d, i, _) => i * 50);
```

¿Qué ocurre?

```
<svg id="vis" width="400" height="250">
 <rect height=”40” width=”23” y=”0”></rect> <!-- 23 -->
 <rect height=”40” width=”45” y=”50”></rect> <!-- 45 -->
 <rect height=”40” width=”99” y=”100”></rect> <!-- 99 -->
 <rect height=”40” width=”64” y=”150”></rect> <!-- 64 -->
</svg>
```

Data Joins - Comando join - Ejemplo 2

Vamos al código 👨‍💻 👩‍💻

Data Joins - Comando join 👀

Agregar elementos vinculados a datos

1. Los datos no tienen que ser solo números. Puede ser un diccionario con múltiples datos.
2. Puede pasar que no exista un SVG y primero sea necesario agregarlo.

```
const datos = [
  { valor: 20, color: "red" }, { valor: 32, color: "magenta" },
  { valor: 12, color: "blue" }, { valor: 86, color: "orange" },
]

const SVG = d3.select("body").append("svg").attr("width", 400).attr("height", 400);

SVG.selectAll("rect").data(datos).join("rect").attr("height", 40)
  .attr("width", (d, i, _) => d.valor)
  .attr("fill", (d, i, _) => d.color)
```

Data Joins - Comando join 👀 - Ejemplo 3 y 4

Vamos al código 👨‍💻 👩‍💻

Data Joins - Comando Join - Spoiler

- 🤔 ¿Si ya existen elementos en el HTML cuando hago el join?
- 🤔 ¿Si quiero eliminar algún dato? ¿qué pasa con el elemento?
- 🤔 ¿Si quiero actualizar el valor de algún dato? ¿cómo actualizar visualmente el elemento?

Respuesta: utilizar todo el potencial del comando **join.** De forma interna no solo hace append, hace mucho más.

1. **Si hay datos pero no elementos. Crea un nuevo elemento para cada dato.**
2. Si ya existe algún elemento para vincular al dato, **no** creará un nuevo elemento.
3. Si hay algún elemento que **no quedó vinculado** a algún dato, **lo elimina**.

Data Joins - Comando Join - Spoiler

Respuesta: utilizar todo el potencial del comando **join**

También podemos definir 3 funciones que se encargarán de procesar cada conjunto del *data joins*.

```
d3.selectAll("rect").data(datos).join(
     (enter) => enter.append("rect"), // función 1: selección de datos sin elementos
     (update) => update,       // función 2: selección de elementos ya existentes
     (exit) => exit.remove(), // función 3: selección de elementos que perdieron su dato
);
```

En la próxima clase estudiaremos el caso de *enter*, *update* y *exit* con estas 3 funciones.

IIC2026
Visualización de Información

Hernán F. Valdivieso López
(2023 - 1 / Clase 04)