

Ayudantía 6

| 2022 - 2 | IIC2026 Visualización de Información |

| [Vicente Paiva](mailto:vpaivag@uc.cl) | vpaivag@uc.cl |

Temas de la ayudantía



Selecciones en D3



Ejercicio con selecciones



Data Join



Ejemplo Data Join

¿Qué son las selecciones?

Selecciones

- Objetos en D3 que corresponden a colección de elementos HTML.
- Permiten seleccionar cosas.
- Objetos en D3 que corresponden a colección de elementos HTML.



```
d3.select(elemento);  
d3.selectAll(elemento);
```

d3.select();

Retorna una selección con **el primer elemento** del documento que coincide con el argumento indicado.



```
const seleccion = d3.select('p'); // Por tag de html  
const seleccion = d3.select('.importante'); // Por clase  
const seleccion = d3.select('#principal'); // Por id
```

d3.selectAll();

Retorna una selección con **todos los elementos** del documento que coincidan con el argumento indicado.



```
const seleccion = d3.selectAll('p'); // Por tag de html  
const seleccion = d3.selectAll('.importante'); // Por clase  
const seleccion = d3.selectAll('#principal'); // Por id
```

Las selecciones permiten manejar el DOM

→ `seleccion.attr();`

→ `seleccion.style();`

→ `seleccion.append();`

→ `seleccion.text();`

seleccion.attr();

Permite agregar (o cambiar si ya existe) un **atributo** de un elemento HTML, y darle un valor determinado. Retorna el mismo objeto de la selección.

‘valor’ puede ser una función!



```
seleccion.attr('atributo', 'valor');  
seleccion.attr('class', 'my_seleccion');
```


seleccion.style();

Define (o cambia si ya existe) una **propiedad de CSS** sobre el elemento determinado y le asigna un valor. Retorna el mismo objeto de la selección.

‘valor’ puede ser una función!



```
seleccion.style('atributo', 'valor');  
seleccion.style('color', 'tomato');
```

seleccion.append();

Permite **crear un nuevo elemento HTML** dentro de cada elemento de la selección (lo agrega al DOM). Retorna una selección del elemento agregado.



```
seleccion.append( 'elemento' );
```

seleccion.append();



```
<body>
  <ul> </ul>
  <ul> </ul>
</body>
```



```
d3.selectAll('ul').append('li');
```



```
<body>
  <ul>
    <li></li>
  </ul>
  <ul>
    <li></li>
  </ul>
</body>
```

seleccion.text();

Permite **agregar o editar texto** de un elemento. Retorna el elemento seleccionado



```
seleccion.text( 'texto' );
```

seleccion.text();



```
<body>
  <ul> </ul>
  <ul> </ul>
</body>
```



```
d3.selectAll('ul').append('li').text('Soy un li');
```



```
<body>
  <ul>
    <li> Soy un li </li>
  </ul>
  <ul>
    <li> Soy un li </li>
  </ul>
</body>
```

Chaining

→ `seleccion.attr();`

→ `seleccion.style();`

→ `seleccion.append();`

→ `seleccion.text();`

Todas retornan una selección. Lo que nos permite ir **encadenando** métodos a selecciones de forma continua.

Chaining



// Sin Chaining

```
const myTitulo = d3.select('body');  
myTitulo.append('h1');  
myTitulo.text('Soy el titulo');  
myTitulo.attr('class', 'titulo');  
myTitulo.style('color', 'red');
```

// Chaining!

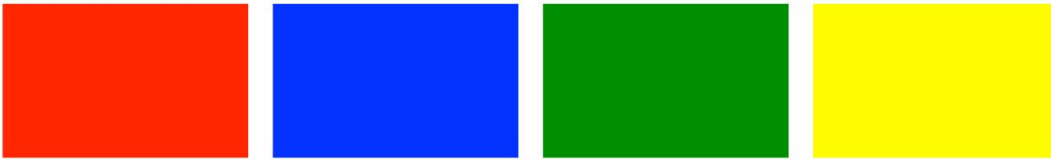
```
const myTitulo = d3.select('body').append('h1').text('Soy el titulo').attr('class', 'titulo').style('color', 'red');  
  
const myTitulo = d3.select('body')  
  .append('h1')  
  .text('Soy el titulo')  
  .attr('class', 'titulo')  
  .style('color', 'red');
```

Ejercicio selecciones

InfoVis es el mejor ramo del DCC

[Aca esta el link a d3.js](#)

- Soy un elemento de lista solitario :/.
- Hola Mundo desde un
- Hello World from a
- Ciao Mondo da un



A solid blue vertical bar is positioned on the left side of the slide.

Data Join!

Data Join

Forma de vincular elementos con datos, generamos vínculo de marcas y canales a través de código.

→ `data()` ;

→ `enter()` ;

→ `update()` ;

→ `exit()` ;

→ `join()` ;

seleccion.data();

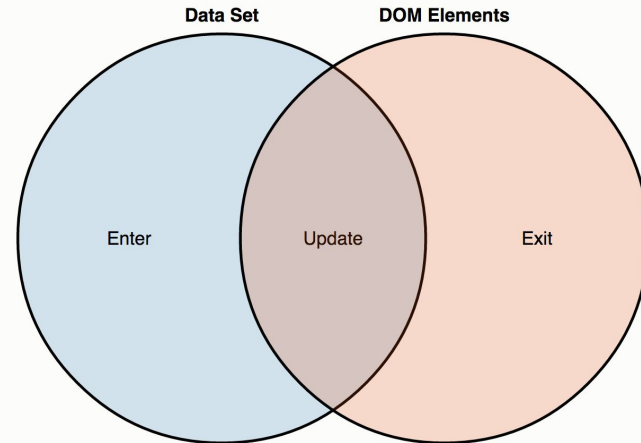
Permite vincular una selección de elementos a una lista de datos. Retorna una selección con los elementos que se asociaron a datos (**retorna update**).

Genera 3 selecciones:

→ enter(); → Datos que quedan sin asociarse a elementos.

→ update(); → Datos asociados a elementos.

→ exit(); → Elementos no asociados a datos.



```
const datos = [5, 10, 3, 12]
d3.select('#svg')
  .selectAll('rect')
  .data(datos);
```

seleccion.data();

Caso 1: Cantidad de elementos es igual a cantidad de datos

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

```
const datos = [5, 10, 3, 12]
const update = d3.select('#svg')
  .selectAll('rect')
  .data(datos)
  .attr('height', '10px')
  .attr('width', (d) => d)
  .attr('y', (d, i) => i * 20);
```

```
<svg>
  <rect width='5' height='10', y='0'></rect>
  <rect width='10' height='10', y='20'></rect>
  <rect width='3' height='10', y='40'></rect>
  <rect width='12' height='10', y='60'></rect>
</svg>
```

seleccion.data();

Caso 2: Cantidad de elementos es mayor a cantidad de datos

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

```
const datos = [5, 10]
const update = d3.select('#svg')
  .selectAll('rect')
  .data(datos)
  .attr('height', '10px')
  .attr('width', (d) => d)
  .attr('y', (d, i) => i * 20);

update.exit().remove();
```

```
<svg>
  <rect width='5' height='10', y='0'></rect>
  <rect width='10' height='10', y='20'></rect>
</svg>
```

seleccion.data();

Caso 3: Cantidad de elementos es menor a cantidad de datos

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

```
const datos = [5, 10, 3, 12, 20]
const update = d3.select('#svg')
  .selectAll('rect')
  .data(datos)
  .attr('height', '10px')
  .attr('width', (d) => d)
  .attr('y', (d, i) => i * 20);
```

```
update.enter().append('rect')
  .attr('height', '10px')
  .attr('width', (d) => d)
  .attr('y', (d, i) => i * 20);
```

```
<svg>
  <rect width='5' height='10', y='0'></rect>
  <rect width='10' height='10', y='20'></rect>
  <rect width='3' height='10', y='40'></rect>
  <rect width='12' height='10', y='60'></rect>
  <rect width='20' height='10', y='80'></rect>
</svg>
```

seleccion.data(); → **merge()**

Caso 3: Cantidad de elementos es menor a cantidad de datos

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

```
const datos = [5, 10, 3, 12, 20]
const update = d3.select('#svg')
  .selectAll('rect')
  .data(datos);


const nuevosElementos = update.enter().append('rect');

update.merge(nuevosElementos)
  .attr('height', '10px')
  .attr('width', (d) => d)
  .attr('y', (d, i) => i * 20);
```

```
<svg>
  <rect width='5' height='10', y='0'></rect>
  <rect width='10' height='10', y='20'></rect>
  <rect width='3' height='10', y='40'></rect>
  <rect width='12' height='10', y='60'></rect>
  <rect width='20' height='10', y='80'></rect>
</svg>
```


join();

Comando que automáticamente genera un **remove()** en el **exit**, un **append()** en **enter**. Retorna la unión de enter y update.



```
const datos = [5, 10, 3, 12, 20]
const update = d3.select('#svg')
  .selectAll('rect')
  .data(datos)
  .join('rect')
  .attr('height', '10px')
  .attr('width', (d) => d)
  .attr('y', (d, i) => i * 20);
```

join();

Este comando permite especificar en sí mismo lo que ocurre con enter, update y exit.



```
const datos = [5, 10, 3, 12, 20]
const update = d3.select('#svg')
  .selectAll('rect')
  .data(datos)
  .join(
    (enter) => {
      enter.append('rect')
        .attr('fill', 'red') // New data in red
    },
    (update) => {
      update
        .attr('fill', 'blue')
        .attr('width', (d) => { return d })
        .attr('y', (d, i) => { return i * 20 });
    },
    (exit) => exit.remove()
  )
  .attr('height', '10px');
```

Ejemplo práctico join()

Para aprender mas sobre data-join y selecciones revisar
<https://github.com/d3/d3-selection>

Para aprender mas sobre enter, update y exit en join()
<https://www.d3indepth.com/enterexit/>