
IIC2026

Visualización de Información

— Hernán F. Valdivieso López —
(2023 - 1 / Clase 08)

Temas de la clase - Utilidades D3 II

1. Eventos.
2. Transiciones.
3. Data Join personalizado.

Eventos

Eventos

Anteriormente vimos que podemos conectar eventos del DOM con Javascript.

```
// Creamos un elemento con tag <h1>
```

```
const elemento = document.createElement("h1");
```

```
// Conectamos la ocurrencia de un evento en el elemento a una función
```

```
elemento.addEventListener(evento, funcion);
```

¿Qué evento puede ser?

- Click sobre el elemento ([click](#)).
- Pasar el mouse sobre el elemento ([mouseover](#)).
- Copiar el elemento ([copy](#)).
- Hacer scroll en el elemento ([scroll](#)).
- Muchos más: https://www.w3schools.com/jsref/dom_obj_event.asp

D3 también puede hacer lo mismo 🎉.

Eventos

🤔 ¿Cómo se hace en D3?

1. Generar una selección de elementos.
2. Utilizar el método `.on(evento, function)` para indicar que todos los elementos de dicha selección gatillen dicho evento.

Por ejemplo:

```
// Seleccionamos todos los rect
const elementos = d3.selectAll("rect");

// Cada vez que hagamos click, imprime uwu
elementos.on("click", () => console.log("uwu"));
```

Eventos

🤔 ¿Cómo se hace en D3?

```
elementos.on("click", (evento, data) => {  
    console.log(evento);  
    console.log(evento.currentTarget);  
    console.log(data);  
});
```

Los eventos en D3 le entregan 2 elementos a la función:

1. Un objeto evento. Aquí podemos ver que tipo de evento fue y acceder, por ejemplo, al elemento HTML que gatilló el evento.
2. El dato asociado al elemento HTML que gatilló el evento. Este solo existe si previamente se hizo un *data joins*, en otro caso será `undefined`.

Eventos

En la clase de hoy veremos 3 eventos:

1. `selection.on('click', ...)`: cuando haces *click* en un elemento de la selección.
2. `selection.on('mouseenter', ...)`: cuando el mouse pasa por encima de un elemento de la selección.
3. `selection.on('mouseleave', ...)`: cuando el mouse deja de estar por encima de un elemento de la selección.

Existe más eventos que veremos más adelante, tales como:

1. Arrastrar un elemento por la pantalla.
2. Doble click en un elemento.
3. Hacer zoom con el mouse.

Eventos

Vamos al código  

Eventos - desafío

1. Lee la [documentación oficial sobre gatillar eventos](#) con D3.
2. Implementar un pequeño HTML+JS donde hacer click en un rectángulo nos indique la posición **dentro del rectángulo** donde se hizo click.
3. Mejore el desafío anterior:
 - a. Defina una escala lineal cuyo dominio es -30 a 30 y el rango es de 0 al ancho del rectángulo.
 - b. Utilice el [método invert](#) de la escala lineal para obtener un número entre -30 y 30 a partir de la posición (eje X) del click.

Se recomienda descargar ambos archivos, abrir solo el HTML en el navegador para ver lo que sucede, crear el código por su cuenta y finalmente revisar `desafio.js`.

Transiciones

Transiciones

- Interfaz que entrega D3 para crear animaciones sobre elementos del DOM.
- Podemos crear diferentes tipos de animaciones:
 - Traslado de figuras.
 - Cambio de color.
 - Cambio de tamaño.

Transiciones

Formato

```
d3.select("rect").transition().duration(200).attr("x", 200).attr("fill", "red")
```

1. Definir una selección.
2. Agregar `transition()` para indicar que todos los cambios siguientes en la selección van a ser animados.
3. Agregar `duration()` para indicar cuantos milisegundos debe durar la animación.
4. Usar tantos `attr()` como uno desea para alternar los atributos de una forma animada.

Transiciones

Múltiples transiciones

```
d3.select("rect")  
  .transition().duration(200).attr("x", 200)  
  .transition().attr("fill", "red")  
  .transition().attr("y", 1000)
```

Podemos definir varias transiciones para que ocurra una después de otra.

Transiciones

Múltiples transiciones 🙄 🙄

```
const rects = d3.select("rect")  
rects.transition().duration(200).attr("x", 200).attr("fill", "red") // T1  
rects.transition().duration(200).attr("y", 300) // T2
```

- En este caso, la transición 2 (T2) se ejecutará y la 1 no.
- Esto ocurre porque T2 sobrescribe la transición actual.
- Todo se soluciona agregando nombres a las transiciones para indicar a D3 que son diferentes transiciones.

```
rects.transition("t1").duration(200).attr("x", 200).attr("fill", "red") // T1  
rects.transition("t2").duration(200).attr("y", 300) // T2
```

Transiciones

Cuidado con lo que retorna una transición 👁️ 👁️

```
const rects = d3.select("rect")  
rects.transition() != rects
```

- Cuando se aplica una transition, ya no se retorna la selección. Se retorna un objeto tipo transición.
- Cuando hagan *data join* asegurarse de no retornar una transición. Sino todo puede llegar a fallar 😭

Transiciones

Vamos al código 

Transiciones - extra

- Podemos retrasar las transiciones con el comando `delay()`.
- También podemos cambiar la velocidad de la transición o dar efectos a las transiciones. Todo esto con el comando `ease()`.
- Encontrarán ejemplos de estos comandos en: [Transitions | D3 in Depth](#)

Data joins personalizado

Data joins personalizado

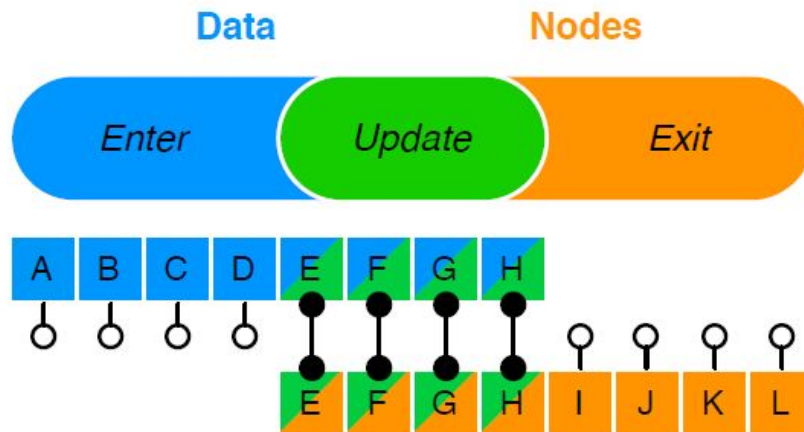
- Antes de profundizar en este tema, vamos a gatillar un típico problema en D3, lo vamos a solucionar y luego entenderemos por qué funcionó la solución.
- Para generar esta experiencia, vamos a implementar la eliminación de datos en la visualización cuando hacemos click en una barra.

Vamos al código 

Data joins personalizado

🤔 ¿Por qué al principio siempre se elimina la última barra aunque hacía click para eliminar otra barra?

- Esto se debe al funcionamiento **interno** del data joins.
- Recordemos que el data joins consiste en vincular datos a elementos del DOM.
- Este vínculo lo hace en el código mediante una llave. Si no le damos esa llave, usa el índice de los elementos y el índice de los datos como llave.



Data joins personalizado

Imaginemos los siguientes elementos y datos.

Elementos

<rect>

<rect>

<rect>

<rect>

<rect>

Datos

4

15

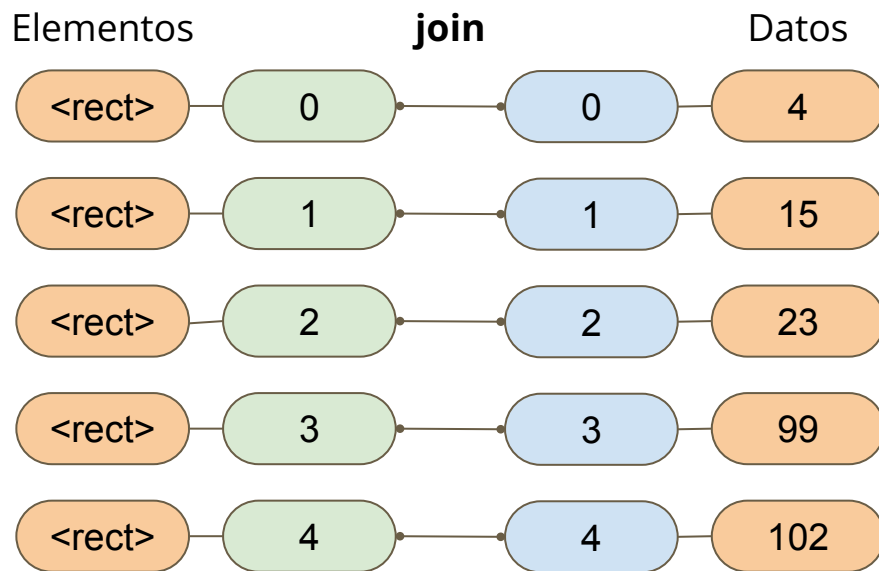
23

99

102

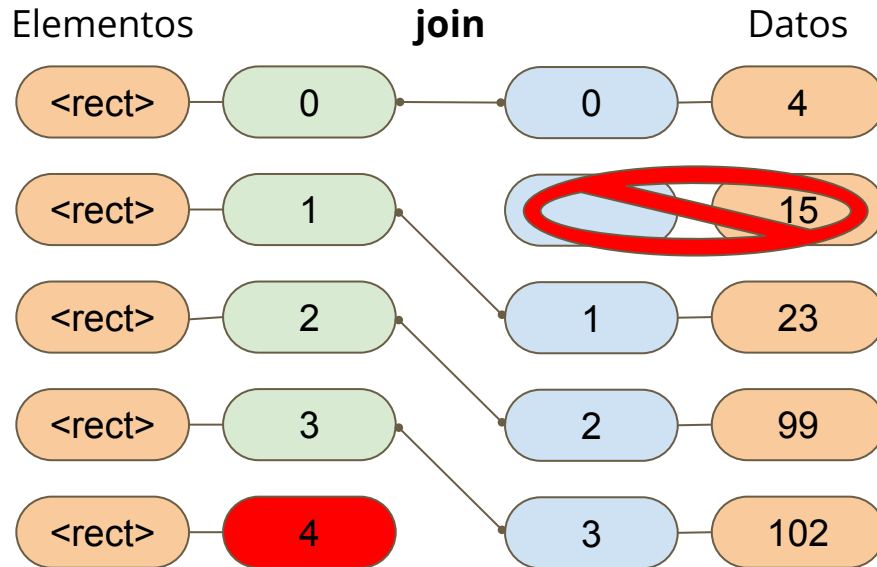
Data joins personalizado

Hacemos `.data(datos)` sin dar una llave. Por lo tanto, D3, **usa el índice de los elementos y el índice de los datos como llave.**



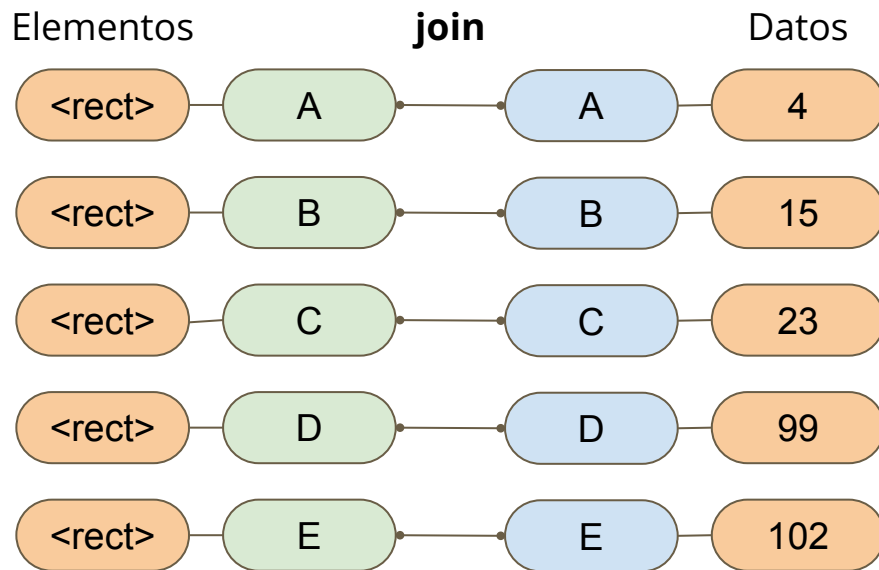
Data joins personalizado

Eliminamos el dato 15. Esto hace que el índice de los siguientes datos se actualice. Por lo tanto, el data join conecta cada rect con su nuevo dato y deja el último rect sin datos.



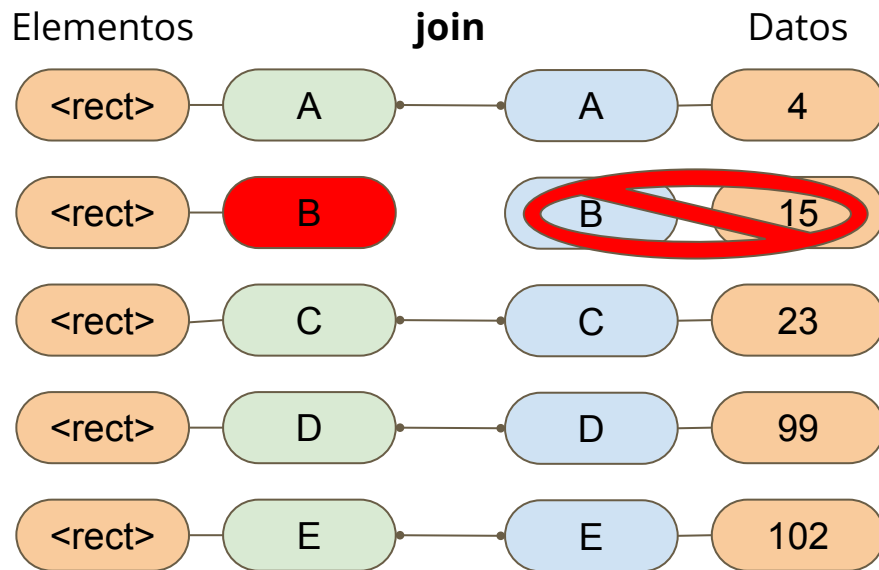
Data joins personalizado

Hacemos `.data(datos, d => d.letra)` pero ahora usamos una categoría única por dato. Por ejemplo, letras.




Data joins personalizado

Eliminamos el dato 15, pero la llave ya no es el índice, sino una letra. Los demás datos no cambian su llave. Con esto hacemos que el rect conectado al 15 sea eliminado.



Próximos eventos

Próxima clase

- Layout tabulares en D3.
- Traigan notebook 

Tarea 2

Se entrega el lunes 3 de abril. Luego comienzan los 3 días de entregas atrasadas.

Próximo miércoles

No hay ayudantía. Miércoles santo para visualización.

IIC2026

Visualización de Información

— Hernán F. Valdivieso López —
(2023 - 1 / Clase 08)
