Vincular Nodos con D3

Progreso del proyecto

Actualmente podemos diferenciar 3 etapas del proyecto en donde cada una se distingue por el enfoque de desarrollo \*, para comprender esto repasaremos un poco las bibliotecas utilizadas:

D3:

Actualmente se está implementando la Versión 5 en el proyecto.

En internet encontramos proyectos utilizando Version 3, Version 4, Version 5, Version 6 de D3.

La versión 3, al ser la mas antigua, tenía formas muy programáticas/algorítmicas de implementar los gráficos, la versión 4 es la más utilizada, sin embargo, entre la versión 4 y la versión 5 al ser más actual se eligió esta última. Hay pocos ejemplos con la versión 6, y poca documentación. La mayoría de la documentación utiliza la versión 4.

Por lo tanto por cantidad de documentación, fácil implementación, la intención de no usar lo más anticuado y más ejemplos, se optó por la versión 5.

¿Cómo trabaja D3?

Básicamente, selecciona un elemento del documento HTML, y a partir de este, ingresamos objetos SVG, que según los datos brindados (pueden ser internos o externos) arma un gráfico.

Los objetos SVG del gráfico pueden detectar eventos de mouse, o tiempo.

¿Cuáles son los aspectos más problemáticos de D3?

Es que como dijimos anteriormente, selecciona un elemento del documento HTML, por ende, hasta que no sea creado el documento HTML, los gráficos no pueden generarse.

Otro punto para tener en cuenta es que, una vez creado un gráfico, si nosotros queremos agregar nueva información al gráfico, D3 nos ofrece una forma de actualizar el gráfico (o en todo caso se podría ejecutar constantemente, requiriendo mucho proceso). Pero esta actualización debe ser disparada por un evento, es decir, no se disparará automáticamente cuando agregamos nueva información.

Vue3js:

Actualmente estamos implementando Vue3js Composition API.

En un principio se utilizó la denominada Options API, que consiste en utilizar los ciclos de vida de los componentes, mas el estado interno (data), las propiedades (props), etc.

¿Cúal fue el problema con Options API?

Al recibir información constantemente el objeto de gráficos ya sea por un evento de mouse, o una actualización de la información entrante, el código para resolver esto se tornaba complejo y extenso.

Utilizar la actualización del gráfico cada vez que se recibe información no era una opción ya que no estaría aplicando los conceptos de reactividad que requiere la aplicación, en otras palabras, no estaría aprovechando los beneficios de Vue3.

¿Qué utilizamos para resolverlo?

Utilizamos la llamada Composition API, que es otra forma de programar con Vue3 y está diseñada para estos casos de reactividad.

El gráfico de D3 es un componente que requiere mucha reactividad porque recibe mucha información para ser procesada.

Entonces hubo muchos cambios en el código tanto de Vue3js como la implementación de D3 ya que al delegar la actualización a Vue3js, se separa la lógica de manipulación de la información para Vue3 y la lógica de gráficos para D3.

(\*) La primera implementación fue: graficar elementos vinculados con otros elementos desde información recibida y cargada localmente

La segunda implementación fue vincular nodos a través de enlaces desde información recibida y cargada localmente

La tercera implementación fue vincular nodos a través de enlaces con información recibida y observada, cuando cambia la información cambia el gráfico, no se carga localmente entonces el gráfico es instantáneo.