Cápsula 1: Eventos en D3

Hola, bienvenidxs a una cápsula del curso Visualización de Información. En esta hablaré sobre eventos en D3, que sentará las bases para muchos tipos de interacción que revisaremos más adelante.

Comenzaremos con una adaptación del programa con el que contábamos en la última cápsula, que organiza mejor los contenedores de ejes y barras. Construiremos sobre este ejemplo y agregaremos gatillantes de algún evento que ocurra en elementos del documento.

Las selecciones de D3 nos proveen el método “on”, que actúa de forma muy similar al método “addEventListener” de JavaScript. Estás reciben un tipo o tipos de evento, y una función que se gatilla cuando ocurran eventos especificados sobre algunos de los elementos de la selección sobre la cual se actúa.

Asignaremos algo simple, una función que imprime “evento” cuando un clic ocurre sobre una de las barras de nuestro gráfico. Si lo probamos, podemos ver en consola que efectivamente se imprimen los mensajes al cliquear sobre barras. Nota que varios navegadores resumen el mismo mensaje impreso mediante un número al costado, y vemos que el número aumenta.

Ahora, la función gatillante puede hacer más que eso. Estas reciben dos argumentos por defecto: primero un objeto de evento que describe el evento ocurrido, de igual forma que con “addEventListener” de JavaScript; y segundo el dato actual asociado al elemento que se presionó. Ese segundo argumento lo provee D3, y nos da acceso directo al dato en caso de necesitarlo.

Usemos estos argumentos para algo simple. Imprimimos el dato asociado a la barra sobre la cual se haga clic, e imprimimos la barra gatillante, que se puede obtener mediante la propiedad “currentTarget” del objeto evento. Si lo probamos, podemos ver que efectivamente se imprimen ambos objetos, y de forma correspondiente a la barra cliqueada.

Pero podemos ir más allá con esto. Como tenemos acceso a todas las definiciones exteriores al gatillante desde ahí, podemos afectar lo que necesitemos del documento. Para verlo, agregaré un párrafo después del elemento SVG. Y desde el gatillante de clic, alteraré su contenido de texto para que incluya los detalles del dato asociado.

Si lo probamos, obtenemos una forma simple de interacción donde un usuario puede ver el detalle de lo que observa en la visualización. Si bien el ejemplo es simple, pueden extender esta idea a lo que se les ocurra. Por ejemplo, implementar un cuadro de texto flotante que aparezca sobre el elemento cliqueado.

Además, hay muchos otros tipos de eventos. Por ejemplo, el hacer clic sobre la barra para ver su detalle puede considerarse ya mucho trabajo para un usuario. Podemos reemplazar el

evento de forma que se muestre el detalle solo si el cursor pasa sobre la barra. Para eso existe el evento “mouseenter”, que se gatilla cuando el cursor entra inicialmente al espacio del elemento.

Si reemplazamos “click” por “mouseenter” como evento en nuestra definición, veremos que ahora no es necesario cliquear para ver el detalle, solo basta con pasar el cursor.

Ahora, lo raro es que el detalle queda ahí a pesar de que el cursor no esté sobre la barra. Para reiniciar el contenido podemos agregar el evento “mouseleave”, que es la contraparte de “mouseenter”. Si lo probamos, ahora contamos con detalle solo moviendo el mouse.

Finalmente, esta interfaz de D3 para agregar comportamiento no es única para elementos SVG, también permite hacerlo sobre el resto del DOM. Ahora crearé un botón en el documento, que utilizaremos para agregar nuevos datos al arreglo interno del programa, y por lo tanto también nuevas barras a la visualización.

El gatillante asociado lo agregaré una vez cargado el estado inicial del *dataset*, así no permite agregar datos antes de obtener los datos iniciales. Modifiqué un poco el código inicial para facilitar la posibilidad de un conjunto de datos que cambia en el tiempo.

Esta declaración indica que sobre cada clic en el botón, se agrega al conjunto de datos un nuevo dato y se volverá a llamar a la función “joinDeDatos”, con los datos actualizados. Aquí se hace uso de una función definida arriba para crear un dato aleatorio para nuestro ejemplo.

Si lo probamos, vemos que aparece una nueva barra cada vez que se apreta el botón, y la visualización recalcula todas las posiciones. Esto último es gracias al trabajo dedicado en las cápsulas pasadas, donde adaptamos escalas y ejes en base a los valores de los datos. Si agregamos más y más datos, podemos ver cada vez más barras correspondientemente.

Ahora, con eventos nos encontramos en la situación donde la versión de D3 importa. La definición de funciones gatillantes para eventos cambió para la versión 6 de D3, y esa es la que revisamos en esta cápsula recién. Es probable que encuentres ejemplos de uso de eventos en versiones anteriores que varían levemente o usen métodos que ahora se consideran anticuados, así que tengan ojo con eso.

Una forma de poner atención a eso es refiriéndose a la documentación oficial. En la versión 6 de D3 se dejó el uso de un módulo “d3-event”, y ahora se simplificó la interacción usando eventos de JavaScript directamente. En el submódulo de “d3-selection”, en la descripción del método “on” menciona como usar cualquier evento de DOM.

A su vez menciona algunas utilidades para tratar con particularidades de eventos. Por ejemplo, el uso de coordenadas de eventos de mouse. Investiga por tu cuenta cómo obtener las coordenadas del mouse en un evento y escribe código que haga algo con ellas.

Con eso termina el contenido de esta cápsula. Recuerda que si tienes preguntas, puedes dejarlas en los comentarios del video para responderlas en la sesión en vivo de esta temática. ¡Chao!