Cápsula 2: Transiciones en D3

Hola, bienvenidxs a una cápsula del curso Visualización de Información. En esta hablaré sobre las transiciones de D3, que son una interfaz que entrega la librería para crear animaciones sobre elementos del DOM.

Ya contamos con una visualización bastante buena. Tiene interacción básica y se ajusta a tamaños fijos y a los datos que se le entregan. Pero cuando hay cambios, estos son inmediatos y bruscos. Si estos cambios pudieran animarse, mejoraría bastante la experiencia de usuario. ¡Aprenderemos a hacerlo en esta cápsula!

Para comenzar, decidí hacerlo simple, así que comenté momentáneamente el código de la visualización y agregué un rectángulo en nuestro SVG, como podrás ver en pantalla.

Comenzaremos animando uno de sus atributos: el ancho. Para hacerlo, podemos llamar al método de selecciones “transition”. Esta sigue la forma de encadenamiento de métodos de las selecciones, pero no retorna un objeto selección, si no que retorna un objeto transición. Este tiene un comportamiento distinto a las selecciones, y se usan específicamente para animar situaciones.

Si le encadenamos el método “attr” para modificar su ancho a un valor de “300”, básicamente estamos declarando que se aplique una transición sobre la selección rectángulo, de tal forma que se modifique el ancho al valor 300. Si probamos el código actualizando, veremos una animación bastante rápida, pero que efectivamente cambia su ancho.

Podemos alterar cuánto se demora en hacer la animación, llamando al método “duration” sobre la transición y entregando el tiempo en milisegundos. Si actualizamos, ahora apreciamos mejor cómo se genera la animación en el tiempo.

Nota que si bien el objeto transición se crea en el llamado a “transition”, esta no comienza a ejecutarse de inmediato, por lo que podemos adaptarla, como mediante “duration” y especificar qué ocurrirá en la transición. Es posible alterar múltiples atributos en la misma transición, de forma que compartan el cómo y la duración de la animación.

Alteremos el color de relleno del rectángulo. Automáticamente D3 detecta el tipo de atributo a alterar, genera pasos intermedios desde el valor inicial del atributo al valor objetivo especificado en la transición, y los modifica en el tiempo. Si ejecutamos, podemos ver que logra alargar y cambiar el color.

También es posible encadenar transiciones. Si llamamos a “transition” sobre un objeto de transición, se crea otro objeto de transición que comenzará después de la anterior. Podemos entonces volver a los valores originales. Si lo probamos, vemos que funciona. Nota que la

duración de la segunda transición fue la misma que la anterior, y es porque automáticamente comparte ese tipo de características al encadenar transiciones.

Ahora, imaginemos que queremos también animar el aumento del alto del rectángulo, pero deseo que sea más rápido, que solo demore un segundo. Como se comporta distinto a estas transiciones, podemos crear un objeto de transición aparte que haga esto, así.

Si intentamos ejecutar, vemos que solo cambió el alto del rectángulo. Esto es porque las transiciones pueden sobreescribir e interrumpir otras. Como aquí definimos dos transiciones sobre los mismos elementos, la segunda reemplaza la primera. Este tipo de comportamiento está bien, ya que debería ser posible interrumpir transiciones en ejecución.

En este caso no es lo que esperamos, por lo que hay que arreglarlo. Al crear transiciones se puede especificar un nombre de transición. Al darles nombres, quedan registradas como distintas definiciones y si no comparten nombre, entonces no se reemplazan o interrumpen, y funcionan en conjunto. Si lo probamos sobre el ejemplo, ahora si funciona.

Ahora que tenemos la idea general más clara, volvamos a nuestro programa de barras.

Vamos a animar la aparición de las barras, de forma que crezcan desde la base del eje hasta su altura. Para eso, podemos ir a la definición de atributos de barras, luego del join de datos (que revisamos hace unas cápsulas). Podemos separar de las definiciones a aquella definición que deseamos animar.

Aquellas que no se apliquen sobre una transición, se asignan de forma inmediata, pero las que vayan sobre la transición, se animarán. Podemos probar animando solamente la altura. Al ejecutar, vemos que funciona… pero no como queríamos.

Y claro, tiene sentido lo que ocurrió. Lo que pasa es que animamos solamente la altura, pero si queremos que se alineen las barras en el eje inferior, la altura de las barras también debe cambiar al mismo tiempo. Podemos intentar moverla de lugar en las definiciones e incluirla en la transición. Si lo probamos… todavía sigue raro.

Si se fijan, las barras se mueven desde arriba hacia abajo, lo cual hace sentido con la definición de la transición. El problema, es que en realidad esperaríamos que todas las barras comiencen abajo, y mientras su largo aumenta, su posición vertical disminuye.

Para arreglar eso, solo falta definir el valor inicial de este atributo para la transición. Por defecto comienza en 0, pero nos gustaría que comience en la posición del eje. Es cosa de colocar antes de la transición el valor con el comenzará para “y”. Así, la transición hará la interpolación entre ese valor hasta el calculado por la escala.

Al probar este cambio, ahora si comienza como esperamos. Pero, si agrego un nuevo elemento al gráfico, algo extraño pasa con las barras anteriores. Esto ocurre por una razón similar, se está aplicando la misma transición sobre barras que ya fueron creadas y que en la

segunda transición comienzan con altura mayor a cero. Debemos fijar que la altura también comience con un valor fijo explícito.

Al hacer ese cambio, ahora la animación funciona incluso agregando nuevas barras. Otro arreglo que podríamos pensar es que las barras antiguas no deberían porque volver a cambiar su altura, ya que estas ya están en pantalla.

Esto ocurre porque hasta el momento aplicamos las transiciones sobre ambas selecciones en común: *enter* y *update*, las barras nuevas y las existentes. Las transiciones de altura solo las queremos para las entrantes, así que es momento de que diferenciemos el comportamiento entre grupos del data join.

Con el código en pantalla, se genera justo esto. Ahora dejo las llamadas solo necesarias para la transición *enter* dentro del join, y luego las llamadas que son comunes entre *update* y *enter*. Esta forma de escribir el *data join* era propia del método “join”, donde especificamos comportamiento extra mediante funciones.

Nota que agregué una llamada a un método “selection” al final. Esta retorna la selección sobre la cual actúa un objeto transición, y debemos llamarlo para que la función en general retorne una selección y no un objeto transición. Si lo probamos, ¡vemos que funciona!

Finalmente, podemos agregar más transiciones, como las posiciones horizontales de las barras y también animar las actualizaciones de los ejes. Lo bueno es que esto último está bien solucionado por parte de D3, y es tan fácil como agregar un par de líneas a nuestro código. Si lo hacemos, vemos este resultado. Analiza el código final por tu cuenta, a ver si entiendes lo que pasó.

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!