Cápsula 2: JavaScript II

Hola, bienvenidxs a una cápsula del curso Visualización de Información. En esta les presentaré la segunda parte de la introducción a JavaScript. Revisaré más aspectos del lenguaje y veremos JavaScript en ejecución.

Para iterar en JavaScript, existen las declaraciones convencionales de “while” y “for”. La sentencia de “while” funciona como esperaríamos, que repite el código dentro de las llaves mientras la condición dentro de los paréntesis se cumpla.

Las declaraciones de “for” básicas tienen la siguiente forma. En la definición con paréntesis define cómo se realiza la iteración, en base a una variable que comienza con cierto valor, la iteración funcionará mientras la condición en la segunda posición se cumpla, y al final de cada iteración hará una modificación sobre la variable, en este caso aumenta su valor en 1. Encontrarás esta forma de definir “for” en muchos lenguajes, en base a un paréntesis con tres partes.

De todas formas, desde ES6 se introducen otras formas de iterar sobre estructuras como arreglos y objetos, como los indicados en pantalla, que utilizan palabras claves especiales para distintos tipos de datos.

Por otro lado, como en muchos lenguajes es posible declarar y crear funciones, que permite definir secciones de código que se ejecutarán cuando la función sea llamada. Es posible entregar argumentos a dichas definiciones, que alteran su ejecución.

Desde ES6 hay múltiples formas de definir funciones, como las mostradas en pantalla. En este ejemplo las tres funciones son equivalentes en resultado, pero es importante considerar que cada forma de definir funciones tienen detalles distintos de funcionamiento y en contextos más complicados pueden variar mucho.

Entre ellas las funciones de flecha son interesantes a considerar, ya que en algunos casos permiten definiciones de una sola línea. Los argumentos están a la izquierda de la flecha, y el valor retornado por la función a la derecha. Si es necesario más de una línea para una función de flecha, entonces es posible usar las llaves para abrir una sección de código.

Ahora, nos falta ver programas de JavaScript en funcionamiento. Hoy en día, hay formas de ejecutar código JavaScript directamente en una computadora, similar a como se haría con otros lenguajes como Python. Pero originalmente la intención de JavaScript era, y sigue siendo en parte, ejecutarse en el navegador, y así es como lo utilizaremos en el curso, ya que buscamos utilizarlo para crear visualizaciones de información en el documento.

Entonces, para ejecutar un programa de JavaScript, una forma es asociándolo a un documento HTML. Luego, al abrir tal documento en un navegador, los programas asociados

se ejecutarán automáticamente, y podremos ver sus efectos y resultados en una pestaña del inspector de elementos llamada consola.

Toma el siguiente ejemplo de programa. Este define un arreglo que comienza vacío. Luego se ejecuta una iteración que agrega números pares al arreglo. Y finalmente otra iteración recorre los valores dentro del arreglo y llama a la función console.log. Esta función es propia de JavaScript y permite imprimir en consola los argumentos entregados, similar al print de Python.

Si escribimos este programa en un archivo de extensión “js”, podemos importarlo desde un documento HTML mediante la etiqueta “script”, especificando la ruta al archivo a importar. La ubicación de esta etiqueta en el documento es importante ya que define en qué orden se cargan los elementos del documento.

En general es buena idea ubicarlos dentro de “head”, pero ahora por simplicidad le ubicamos al final del body para asegurarnos que exista su contenido al ejecutarse el programa. Luego, al abrir el documento y abriendo la consola dentro del inspector de elementos, podemos ver los resultados de nuestro programa.

Aquí en pantalla puedes ver el programa abierto en mi editor de código favorito, y el documento HTML que importa este programa para ser ejecutado. Luego si es que abro este documento en un navegador, podemos ver que el documento está vacío. Esto es correcto, porque el documento HTML estaba vacío, y nuestro programa no imprimía en el documento mismo, sino en la consola del inspector de elementos.

Si abro la consola, puedo ver el resultado del programa. Por medio de esta consola puedes encontrar los errores de ejecución de tu programa, y también es buena para que revises los valores de distintas cosas en tu programa. Puedes acceder a definiciones dentro de ella incluso.

Nuevamente quiero mencionar que probablemente faltarán detalles a considerar en JavaScript en estas cápsulas, pero mientras avancemos durante el semestre, introduciremos nuevos detalles a medida que sea necesario. De todas formas, si tienes dudas de como hacer ciertas cosas en JavaScript, te invito a investigar por tu cuenta en la web, como usando los recursos de MDN web doc, o preguntándonos simplemente.

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!