TEXT CLASS REVIEW

TEMAS A TRATAR EN EL CUE:

- Funciones async.
- Expresiones await.
- Promesas.

En el CUE anterior aprendimos con éxito sobre la naturaleza de los procesos asincrónicos, y cómo se relacionan con el uso de Promesas.

Para seguir reforzando nuestro conocimiento acerca de los elementos más avanzados de JavaScript, consideraremos conceptos que están relacionados con las “Promesas”, éstos son las funciones `async/await`, y las funciones anónimas autoejecutables (también llamadas "Expresión de función invocada inmediatamente" o por IIFE, su abreviación en inglés).

¿QUÉ SON LAS FUNCIONES ASYNC?

Una función declarada con la palabra clave async, hace que cualquier contenido devuelva o retorne una Promesa, y además, permite el uso de la palabra clave `await` dentro de ellas. En realidad, las palabras clave `async` y `await` habilitan funciones asincrónicas que **siempre** devuelven promesas.

Una ventaja de usar funciones `async/await` es su sintaxis sencilla, que, en contraste con las Promesas, evita la necesidad de configurar explícitamente las cadenas. Esta simplificación de sintaxis, a menudo se denomina *"azúcar sintáctico"*, y aplicando esta terminología al tema que se está considerando, podemos decir que las "funciones `async/await`" son Promesas con "*azúcar sintáctico*".

¿CÓMO FUNCIONAN LAS FUNCIONES ASYNC?

Éstas pueden ejecutar muchos procesos diferentes de forma asíncrona, al igual que las Promesas, pero a diferencia de ellas, las funciones `async` pueden contener cero o más expresiones `await`.

Las expresiones `await` hacen que las funciones `async` (que son asincrónicas) se comporten como si fueran *síncronas*, al suspender temporalmente la ejecución hasta que la promesa (en este caso, la

expresión `await`) devuelta se cumpla o se rechace. El valor resuelto de la expresión `async` se trata como el valor de retorno de la promesa.

APLICACIÓN DE LAS FUNCIONES ASYNC/AWAIT

El propósito y uso de las funciones `async/await` en la vida real, consiste en simplificar la sintaxis necesaria para consumir API basadas en Promesas (lo cual haremos más adelante en este curso). Se le otorga este nombre porque, como establecimos al inicio: las funciones `async` están estrechamente ligadas con las Promesas, dado que siempre devuelven una. Si el valor de retorno de una función asíncrona no es explícitamente una Promesa, estará implícitamente envuelto. Afortunadamente, estamos bien capacitados para manejar todo lo relacionado con esta temática, lo que significa que aprender sobre las funciones `async/await` será muy fácil.

PROMESAS

Otro componente notorio de la nueva revisión es el uso de Promesas, o en inglés `Promises`. Las Promesas en JavaScript nos permiten realizar acciones asíncronas, de una manera muy similar a las acciones sincrónicas, ¿en qué sentido?: con el uso de éstas, las acciones asíncronas pueden manejar los errores y los resultados correctos de alguna acción. Esto siempre ha sido una característica sólo de las acciones síncronas, pero ahora ha cambiado. A continuación, podrá ver un esquema de lo que involucra una Promesa:

De manera muy sencilla, las Promesas ejecutan una acción, y al retornar el resultado utilizando `callbacks` será un valor correcto o, incorrecto. Mediante una serie de ejemplos analizaremos en profundidad cada uno de estos elementos nuevos de la revisión ES6.