Resumen rápido

La sesión se centró en la introducción a las funciones en JavaScript, incluyendo su sintaxis, la importancia del orden de los argumentos y el concepto de valores de retorno. También se discutió el uso de funciones anónimas, el ámbito de las variables y la programación asíncrona utilizando callbacks, promesas y la sintaxis async/await. Por último, se proporcionaron ejercicios de programación asíncrona y promesas, y se discutieron problemas relacionados con la normalización de datos y la comparación de valores en objetos.

Siguientes pasos

Estudiantes: Completar los ejercicios de funciones y asincronía proporcionados por EBIS.

Estudiantes: Revisar el concepto de promesas y la sintaxis async/await para programación asíncrona.

Estudiantes: Practicar la conversión de callbacks a promesas en los ejercicios.

Estudiantes: Consultar con Héctor o el tutor asignado sobre la elección de API para el proyecto final del primer módulo.

EBIS: Actualizar el documento PDF de ejercicios con los símbolos matemáticos correctos (como pi).

EBIS: Solicitar la reorganización de las clases en el campus virtual para corregir el orden de los contenidos.

Estudiantes: Revisar el material de la clase anterior (clase 4) que se encuentra en la sección de la clase 5 del campus.

Estudiantes: Prepararse para aprender sobre el consumo de APIs en futuras clases.

EBIS: Proporcionar información sobre la función de normalización de texto para manejar acentos y caracteres especiales en futuros ejercicios.

Resumen

Introducción a Las Funciones en JavaScript.

La sesión se centra en la introducción a las funciones en JavaScript. EBIS explica que las funciones permiten reutilizar código y evitar repeticiones innecesarias. Se discute la sintaxis básica de las funciones, incluyendo el nombre, los parámetros y el bloque de código. EBIS demuestra cómo crear una función simple de suma y cómo llamarla con diferentes argumentos, destacando la flexibilidad y eficiencia que ofrecen las funciones en la programación.

JavaScript Function Argument Order

La discusión se centra en aspectos de programación en JavaScript. EBIS explica la importancia del orden de los argumentos en las funciones y cómo se pueden asignar valores por defecto a los parámetros. También se aborda la falta de tipado estricto en JavaScript y cómo TypeScript resuelve este problema. Finalmente, se introduce el concepto de valores de retorno en las funciones, utilizando como ejemplo una función para comprobar si un número es par.

Funciones Flecha en JavaScript

La discusión se centra en funciones en JavaScript, específicamente en el uso de la palabra clave "return" y funciones flecha. Ebis explica que todas las funciones devuelven algo por defecto, pero se puede usar "return" para especificar explícitamente qué devolver. Luego, introduce las funciones flecha como una sintaxis más concisa para funciones anónimas con retorno explícito. Ebis también aclara que en las funciones flecha, el retorno es implícito si se escribe directamente después de la flecha, pero si hay más código, se debe usar "return" explícitamente. Finalmente, se menciona que las funciones flecha son útiles para operaciones como forEach, pero no se pueden reutilizar al ser anónimas.

Funciones Anónimas Y Ámbito De Variables.

El instructor explica el concepto de funciones anónimas y el ámbito de las variables en JavaScript. Se discute cómo las funciones anónimas solo existen dentro del contexto donde se definen, como en un forEach. También se aborda la diferencia entre variables declaradas con var, que tienen alcance global, y aquellas declaradas con let o const, que tienen alcance de bloque. Se enfatiza la importancia de usar let y const para evitar problemas de alcance global no deseados en aplicaciones más complejas.

Programación Asíncrona en JavaScript.

El profesor explica el concepto de programación asíncrona en JavaScript utilizando callbacks. Demuestra cómo los callbacks permiten ejecutar código de forma secuencial, esperando a que una función termine antes de ejecutar la siguiente. Luego introduce el uso de setTimeout como ejemplo de operación asíncrona, mostrando cómo permite continuar la ejecución del código mientras se espera que se cumpla el temporizador. El profesor aclara la diferencia entre programación paralela y asíncrona, enfatizando que JavaScript es monohilo pero puede manejar tareas asíncronas.

Promesas en JavaScript

El instructor explica el concepto de promesas en JavaScript. Las promesas son objetos que representan un valor que puede estar disponible ahora, en el futuro o nunca. Tienen tres estados posibles: pendiente, cumplida (fulfilled) o rechazada (rejected). Se muestra la

sintaxis para crear y manejar promesas usando .then() para el caso de éxito y .catch() para los errores. Se enfatiza que las promesas permiten un mejor manejo de operaciones asíncronas comparado con los callbacks anidados, y que se pueden encadenar múltiples .then() para ejecutar código secuencialmente después de que se resuelva la promesa inicial.

Async/Await Para Promesas Encadenadas

EBIS explica los problemas de las promesas encadenadas y presenta la sintaxis async/await como una solución más intuitiva. Describe cómo usar async/await con funciones asíncronas y el bloque try/catch para manejar errores. Demuestra cómo esperar la resolución de promesas secuencialmente usando await, mientras que el código síncrono se ejecuta en paralelo. Aclara que las funciones llamadas con await deben devolver promesas para que funcione correctamente.

Ejercicios De Programación Asíncrona

La clase se centra en ejercicios de programación asíncrona y promesas en JavaScript. EBIS proporciona una serie de ejercicios que cubren funciones, manejo de eventos, cálculos matemáticos y conversión de callbacks a promesas. Se discute brevemente sobre el proyecto final del primer módulo, que implicará el consumo de una API sencilla. EBIS aclara que la función "ready" del navegador es en realidad una promesa que indica cuándo se ha cargado completamente el DOM.

Normalización De Datos Y Objetos.

En la reunión, Ignacio y EBIS discutieron problemas relacionados con la normalización de datos y la comparación de valores en objetos. Ignacio estaba teniendo dificultades para convertir el valor de una clave en minúsculas, lo que estaba causando errores. EBIS aclaró que Ignacio necesitaba convertir el valor del nombre de la persona en minúsculas, no la clave del nombre. También discutieron el problema de los caracteres con acento en el objeto del sistema solar, y EBIS sugirió usar una función de Javascript para localizar el texto para normalizar los datos. La reunión concluyó con EBIS ofreciendo su ayuda para cualquier otra pregunta o problema.