EDUCACIÓN IT

Bootcamp de Full Stack

Bienvenidos a la clase N°28

- Función setTimeout
- Función setInterval
- Asincronismo
- Promise
- Proyecto Integrador F1 (i1)

EDUCACIÓN IT

JavaScript

Función setTimeout

setTimeout es una función que ejecuta una función o fragmento de código una sola vez después de transcurrido un determinado período de tiempo (en milisegundos).

Usos comunes:

- Mostrar un mensaje o alerta luego de unos segundos.
- Simular una carga o animación que ocurre después de un tiempo.
- Esperar antes de redirigir a otro sitio.
- Retrasar la ejecución de una función.

Sintaxis:

```
const timeOut = setTimeout( ( ) => {
   console.log("!Hola Mundo¡");
}, 2000);

clearTimeout(timeOut);
```

EDUCACIÓN IT

JavaScript

Función setInterval

setInterval es una función de JavaScript que ejecuta una función o fragmento de código de forma repetitiva cada cierto intervalo de tiempo (en milisegundos), hasta que se detenga manualmente.

Usos comunes:

- Crear relojes en tiempo real o contadores.
- Actualizar una interfaz cada cierto tiempo.
- Animaciones que se repiten.

Sintaxis:

```
const interval = setInterval( ( ) => {
  console.log("¡Hola Mundo¡");
}, 2000);

clearInterval(interval);
```

EDUCACIÓN IT

JavaScript

Asincronismo

setInterval es una función de JavaScript que ejecuta una función o fragmento de código de forma repetitiva cada cierto intervalo de tiempo (en milisegundos), hasta que se detenga manualmente.

Usos comunes:

- Crear relojes en tiempo real o contadores.
- Actualizar una interfaz cada cierto tiempo.
- Animaciones que se repiten.

Sintaxis:

```
const interval = setInterval( ( ) => {
   console.log("¡Hola Mundo¡");
}, 2000);

clearInterval(interval);
```

EDUCACIÓN IT

JavaScript

Asincronismo

El **asincronismo** es un mecanismo que permite ejecutar ciertas tareas sin detener la ejecución del resto del programa. Es decir, el programa puede seguir ejecutando otras instrucciones mientras una tarea se está resolviendo.

En otras palabras, JavaScript funciona en un solo hilo, lo que significa que solo puede realizar una tarea a la vez. Si una operación tarda mucho y no se maneja de forma asincrónica, el programa se bloqueará hasta que finalice esa operación.

Usos comunes:

- Ejecutar una acción luego de un tiempo determinado (setTimeout).
- Repetir una acción en intervalos regulares (setInterval).
- Esperar la respuesta de un servidor o base de datos.
- Procesar archivos o recursos externos sin bloquear la interfaz del usuario.

EDUCACIÓN IT

JavaScript

Promise

Una **Promise** (promesa) es un objeto que representa el resultado futuro de una operación asincrónica. Se utiliza para manejar tareas asincrónicas, como pedir datos a un servidor, leer archivos o esperar un temporizador, sin bloquear el resto del código.

Estados de una Promise:

- pending — La promesa está en curso, todavía no terminó.
- fulfilled — La operación terminó exitosamente.
- rejected — La operación falló.

Sintaxis:

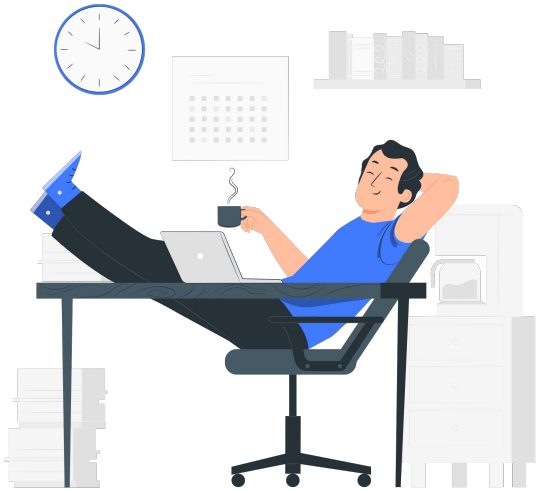
```
const promise = new Promise( (resolve, reject) => {
  if ( isNaN(number) ) {
    reject("Retorna de error");
  }
  resolve("Retorna de éxito");
});
```

```
promise
  .then( (result) => console.log("Se resolvió") )
  .catch( (error) => console.error("Falló") )
  .finally( ( ) => console.log("Se ejecuta siempre al finalizar") );
```

EDUCACIÓN IT

BREAK

Descansemos 10 minutos



EDUCACIÓN IT

Proyecto Integrador F1

Iteración N°1

Para llevar a cabo el proyecto integrador F1, accede al documento de la consigna a través del siguiente enlace:

https://docs.google.com/document/d/181methzV3GUL-YCBNp-qE94qE8-eEiA9Q169gyzUvSs/edit?usp=sharing

EDUCACIÓN IT

CIERRE DE CLASE

Continuaremos en la próxima clase