Manejo de Errores

Introducción

- Supongamos que estamos programando una calculadora en JavaScript.
- Queremos implementar una función que retorne la suma de dos números.
- La función sería similar a la siguiente:

```
function suma(a, b) {
    return a + b;
}
```

Introducción

- Debido a que javascript los parametros no tiene tipado de datos, esta función puede recibir cualquier tipo de datos, y funcionará de forma distinta según los parámetros que reciba

- Nosotros queremos que únicamente se realice la suma si los parámetros son del tipo number.

Lanzar un error

```
function suma(a, b) {
    if (typeof a === "number" && typeof b === "number") {
        return a + b;
    } else {
        throw Error("Debe ingresar numeros!")
    }

}
```

- Si los tipos de datos no son numbers, arrojamos un error.
- **Throw** arroja un error, el cual podemos crear con el objeto error

Exception

- En la programación, a los errores en tiempo de ejecución, se les llama excepciones (o exception en inglés)

- Como programadores, podemos crear y arrojar errores, y también podemos capturar errores, para tratarlos, ya sea mostrando un mensaje de error, o detener una ejecución.

Tipos de Errores en JavaScript

Nombre	Descripción
Error()	Constructor genérico, que representa un error genérico
InternalError()	Representa un error interno en el motor de JavaScript.
RangeError()	Representa un error que ocurre cuando una variable numérica o parámetro está fuera de su rango válido.
ReferenceError()	Representa un error que ocurre cuando se quita la referencia a una referencia no válida.
SyntaxError()	Representa un error de Sintaxis en el código
TypeError()	Representa un error que ocurre cuando una variable o parámetro no es de un tipo válido.

Mejorando nuestro código:

- En este caso, nos conviene arrojar un TypeError()

```
function suma(a, b) {
    if (typeof a === "number" && typeof b === "number")
        return a + b;

    throw TypeError("Debe ingresar numeros!")
}
```

- El bloque else no es necesario, al haber un return.

Capturando errores
Bloque Try - Catch

- Try, Catch significa intentar, capturar
- Dentro del bloque try, colocaremos todo el código que puede arrojar un error
- Dentro del bloque catch capturaremos errores, en caso que se produzcan, y los trataremos
-
```
try {
    // codigo
} catch (error) {
    // manejo del error
}
```

Ejemplo de Try Catch

```
button.addEventListener("click", (evt)=> {
    try {
        let a = document.getElementById("num_a").value;
        let b = document.getElementById("num_b").value;
        let resultado = suma(a, b);
        divResultado.innerHTML = `Resultado: ${resultado} `

    } catch (error) {
        console.error(error);
        alert(error.message);
    }

})
```

Bloque Try, Catch, Finally

```
try {
    // codigo que puede arrojar errores
} catch (error) {
    // manejo del error
}  finally {
    // codigo a ejecutar luego (haya errores o no)
}
```