

Funciones, scope y closure.



Funciones Anónimas



Definición

- Las funciones anónimas son aquellas que no tienen nombre.
- Se suelen usar como argumentos para otras funciones o asignadas a variables.

```
const sumar = function(a, b) {  
    return a + b;  
};  
console.log(sumar(2, 3)); // 5
```

Ventajas

- Son útiles para tareas rápidas y temporales.
- Menos código para escribir.

Funciones de Flecha



Definición

- Introducidas en ES6, tienen una sintaxis más corta.
- No tienen su propio this.

Características

- Más concisas que las funciones normales.
- No tienen this propio, lo que evita problemas con el contexto.

```
const restar = (a, b) => a - b;  
console.log(restar(5, 2)); // 3
```

```
// Función normal  
function multiplicar(a, b) {  
    return a * b;  
}
```

```
// Función de flecha  
const multiplicar = (a, b) => a * b;
```

Scope (alcance)

Scope Global

- Variables accesibles en cualquier parte del código.
- Definidas fuera de funciones o bloques.

```
let nombre = "Juan";
function saludar() {
  console.log("Hola, " + nombre);
}
saludar(); // Hola, Juan
```

Scope Local

- Variables accesibles solo dentro de la función donde se declaran.

```
function saludar() {
  let nombre = "Ana";
  console.log("Hola, " + nombre);
}
saludar(); // Hola, Ana
console.log(nombre); // Error
```

Closures

Definición

- Una closure es una función que recuerda el ámbito en el que se creó.

¿Por Qué Importan?

Permiten mantener el estado de variables en funciones anidadas.

Útiles para funciones que deben recordar información entre llamadas.

```
function saludar(nombre) {  
  const saludo = "Hola, " +  
 nombre;  
  
  return function() {  
    console.log(saludo);  
  };  
}
```

```
const saludoJuan =  
saludar("Juan");  
saludoJuan(); // Hola, Juan
```



```
function crearPersona(nombre, edad) {  
    return {  
        obtenerNombre: function() {  
            return nombre;  
        },  
        obtenerEdad: function() {  
            return edad;  
        },  
        cambiarNombre: function(nuevoNombre) {  
            nombre = nuevoNombre;  
        }  
    };  
}
```

```
const persona = crearPersona("Ana", 30);  
console.log(persona.obtenerNombre()); // Ana  
persona.cambiarNombre("Maria");  
console.log(persona.obtenerNombre()); // Maria
```



Diferencia entre let, var y const en JavaScript

VAR

En cualquier parte
del código.

CONST

Solo lectura.

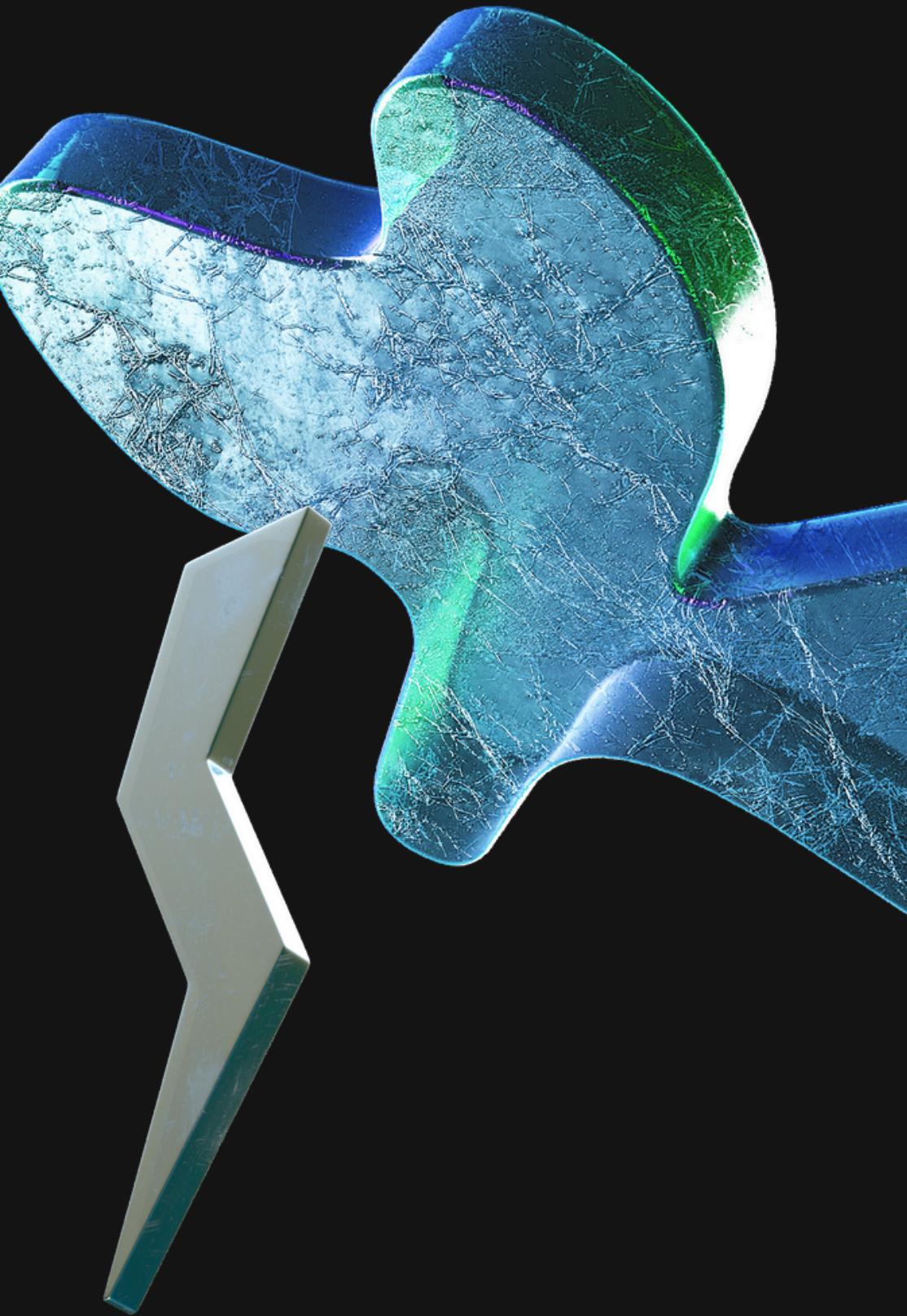
LET

Dentro de un
scope
determinado

LINK

<https://codesandbox.io/s/lucid-lalande-vanilla>

Alguna duda....



Preguntas:

¿Qué es una función anónima en JavaScript?

- a) Una función sin nombre
- b) Una función que se llama automáticamente
- c) Una función que no puede aceptar parámetros
- d) Una función que se ejecuta en el navegador

Preguntas:

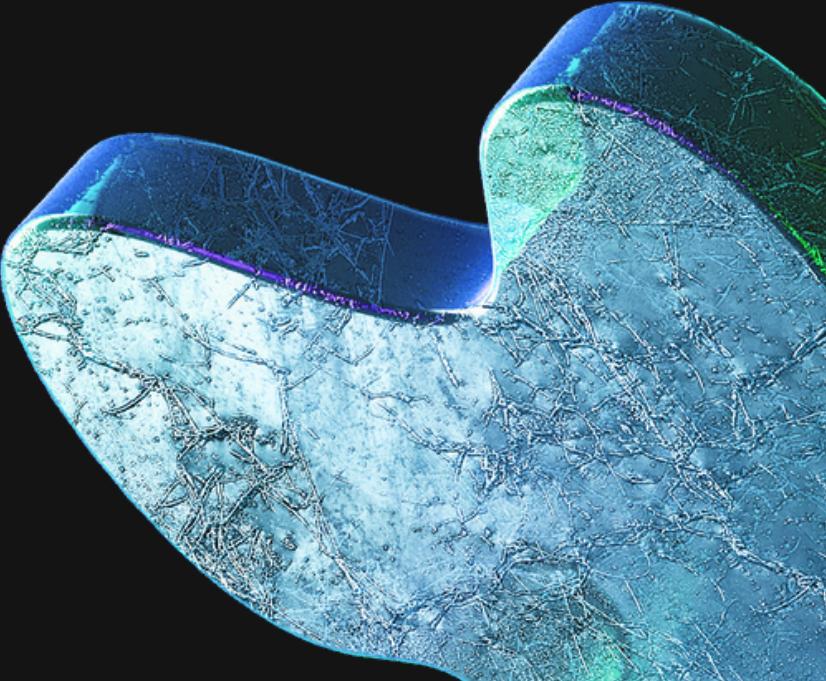
¿Cuál es la sintaxis correcta para definir una función anónima?

- a) let suma(a, b) = { return a + b; };
- b) const suma = function(a, b) { return a + b; };
- c) function(a, b) { return a + b; }
- d) anon function(a, b) { return a + b; }



Preguntas:

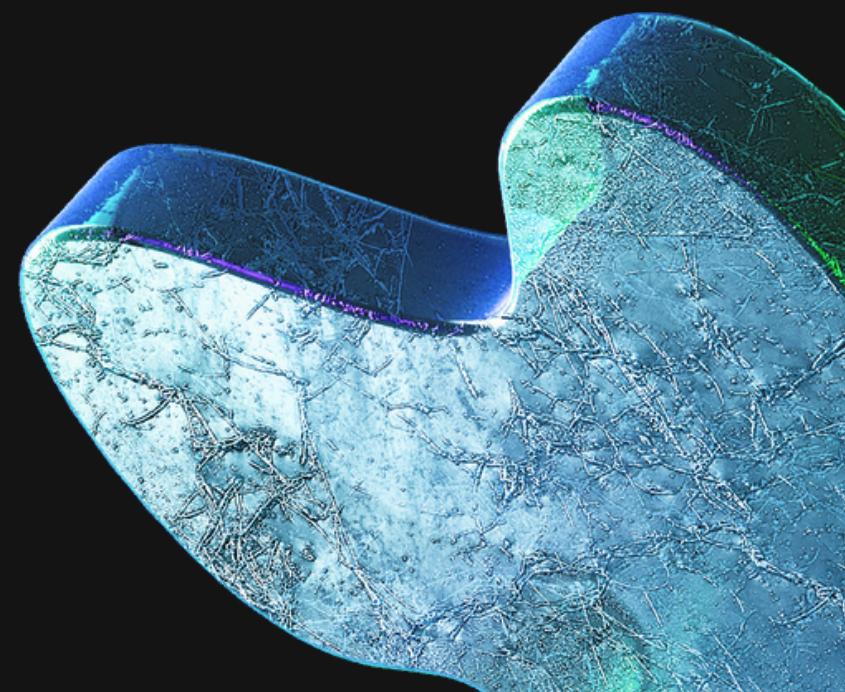
¿Cuál es la sintaxis correcta para definir una función anónima?

- a) let suma(a, b) = { return a + b; };
 - b) const suma = function(a, b) { return a + b; };
 - c) function(a, b) { return a + b; }
 - d) anon function(a, b) { return a + b; }
- 



Preguntas:

¿Cuándo es más común utilizar funciones anónimas?

- a) Cuando se necesita una función de una sola línea
 - b) Como argumentos para otras funciones
 - c) Cuando se desea que la función sea global
 - d) Para definir métodos en un objeto
- 

Preguntas:

¿Cuál es la sintaxis correcta para definir una función de flecha que suma dos números?

- a) let suma = (a, b) => { return a + b; };
- b) let suma = function(a, b) => { return a + b; };
- c) let suma => (a, b) { return a + b; };
- d) let suma = (a, b) => a + b;

Preguntas:

¿Qué es el scope global en JavaScript?

- a) El ámbito dentro de una función
- b) El ámbito dentro de un bloque
- c) El ámbito accesible desde cualquier parte del código
- d) El ámbito solo dentro de un método

Preguntas:

¿Cuál es la salida del siguiente código?

```
let x = 10;  
function imprimirX() {  
    let x = 20;  
    console.log(x);  
}  
imprimirX();  
console.log(x);
```

- a) 20, 20
- b) 10, 10
- c) 20, 10
- d) 10, 20

Preguntas:

¿Qué es un closure en JavaScript?

- a) Una función que se ejecuta automáticamente
- b) Una función que tiene acceso a su propio ámbito léxico
- c) Una función que no puede aceptar parámetros
- d) Una función que se ejecuta solo una vez

Preguntas:

¿Cuál es la salida del siguiente código?

```
function crearSaludo(nombre) {  
    return function() {  
        console.log("Hola, " + nombre);  
    };  
}  
const saludoJuan = crearSaludo("Juan");  
saludoJuan();
```

- a) Hola, undefined
- b) Hola, Juan
- c) Error de sintaxis
- d) Hola, Hola, Juan



Reto del dia de hoy

Explorando la API de Rick and Morty

Objetivo

Crear una aplicación web que consuma la API de Rick and Morty para mostrar información sobre personajes, episodios y ubicaciones. Este proyecto ayudará a practicar los conceptos de funciones anónimas, funciones de flecha, scope, closures y el uso de var, let y const en JavaScript.

Funciones y Conceptos a Practicar

Funciones Anónimas

Funciones de Flecha

Scope (Global, Local, y de Bloque)

Closures

Diferencias entre var, let, y const



Requerimientos

Función para consumir la API

Función para cargar personajes

Función para cargar episodios

Función para cargar ubicaciones

Asignación de eventos a botones

Crear una función que encapsule el estado: listas de personajes, episodios y ubicaciones(Uso de Closures y Scope)

Fin de la presentación.

