



11

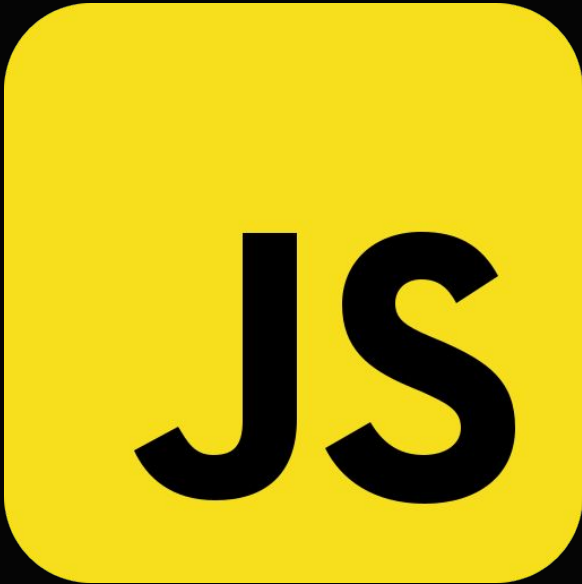
Promesas

Yael Ruiz Pucheta

Temas puntuales de la sección

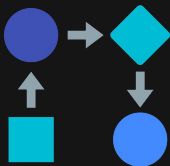


- Código Sincrono y Asincrono
- ¿Que es una promesa?
- Objeto Promise
 - Creación de una promesa
 - Usar una promesa
- Async / Await
- Manejo de errores Async / Await



JS

Código Sincrono y Asincrono



Sincrono: La ejecución de un solo proceso de manera simultanea es decir las tareas pueden ejecutarse secuencialmente.



Asíncrono: Concepto en el cual más de una cosa ocurre al mismo tiempo, o múltiples cosas relacionadas ocurren sin esperar a que la previa se haya completado.

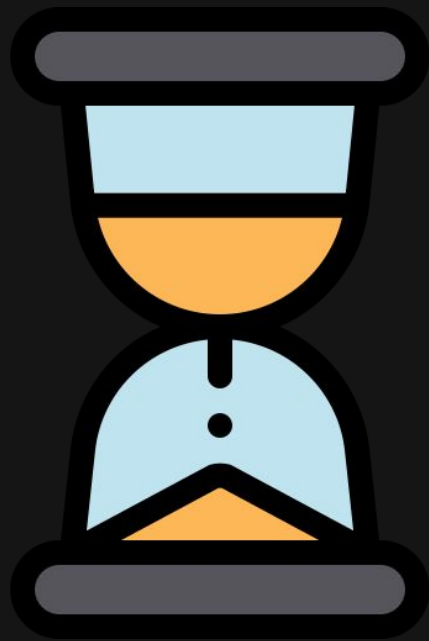


Promesas

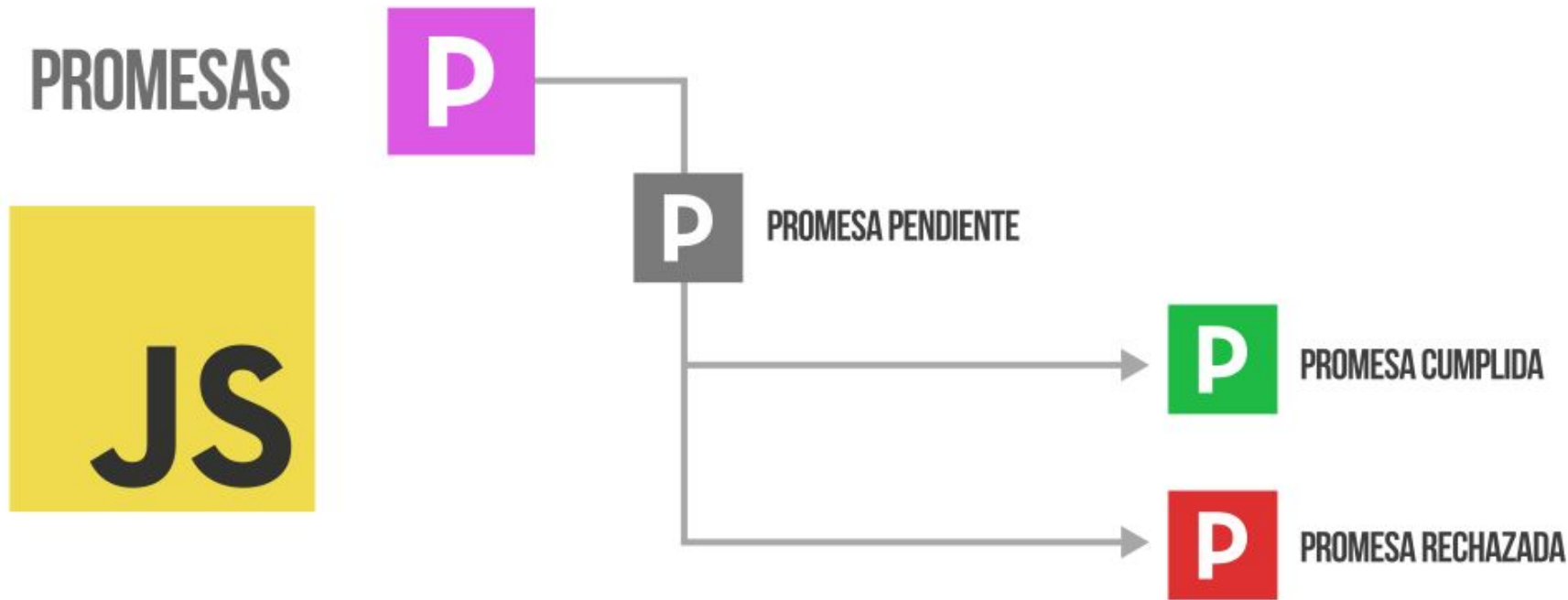
¿Qué es ?

Incorporadas en el E56, Una promesa es un objeto que representa un **valor** que **puede** que **esté disponible «ahora», en un «futuro»** o que **«nunca» lo esté.**

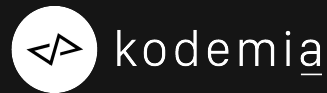
Esta característica permite crear código **asíncrono** que es **dependiente** a otro elemento como conexión a internet



Promesas



Promesas



El **objeto Promise** (Promesa) es usado para computaciones asíncronas. Una promesa representa un valor que puede estar disponible ahora, en el futuro, o nunca.

Definición de una promesa

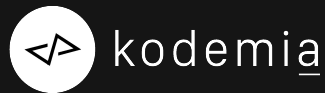
```
// Promesa(resolver, rechazar)

const encontrarCero = new Promise((resolve, reject) => {

  //contenido de la Promesa
  const valor = Math.random();

  if(valor == 0){
    // Si la respuesta es correcta se llama la resolución
    //resolve puede recibir datos simples y estructurados
    resolve(true);
  }else{
    //Si la respuesta es incorrecta se llama al rechazo
    //reject puede recibir datos simples y estructurados
    reject(false);
  }
});
```

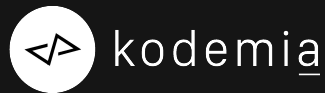
Promesas



Cada promesa tiene los siguientes métodos, que podremos utilizar para utilizarla:

Función	Definición
.then(resolve)	Ejecuta la función callback resolve cuando la promesa se cumple.
.catch(reject)	Ejecuta la función callback reject cuando la promesa se rechaza.
.finally(end)	Ejecuta la función callback end tanto si se cumple como si se rechaza.

Promesas

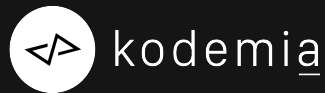


Uso o invocación de promesa



```
encontrarCero.then((valor) =>{  
    // Bloque de código si se resuelve la promesa satisfactoriamente  
  
    console.log(valor);  
}).catch((error)=>{  
    // Bloque de código si la promesa es rechazada  
  
    console.log(error);  
});
```

Promesas



Uso o invocación de promesa



```
encontrarCero.then((valor) =>{  
    // Bloque de código si se resuelve la promesa satisfactoriamente  
  
    console.log(valor);  
}).catch((error)=>{  
    // Bloque de código si la promesa es rechazada  
  
    console.log(error);  
});
```

Async / Await

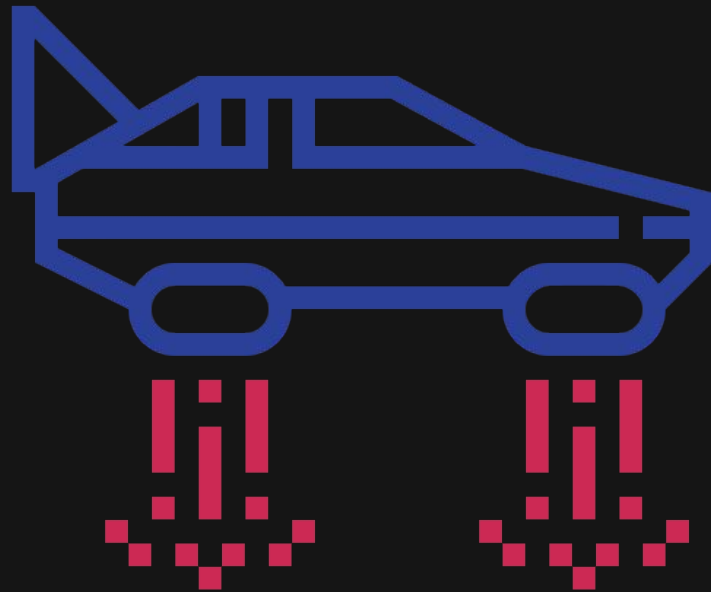
Async /Await

¿Qué es ?

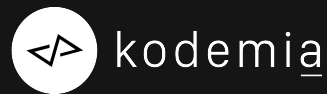
Es una forma alternativa de declarar una promesa como función asíncrona usando **async** y **await**.

Cuando se llama a una función async, esta devuelve un elemento Promise.

Una función async puede contener una expresión **await**, la cual **pausa la ejecución** de la función asíncrona y espera la resolución.



Async /Await

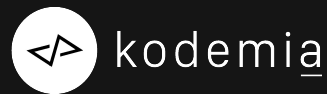


```

// Promesa (resolver, rechazar)
const encontrarCero = new Promise((resolve, reject) =>{

    //contenido de la Promesa
    const valor = Math.random();
    if (valor == 0) {
        // Si la respuesta es correcta se llama la resolución
        //resolve puede recibir datos simples y estructurados
        resolve(true);
    } else {
        //Si la respuesta es incorrecta se llama al rechazo
        //reject puede recibir datos simples y estructurados
        reject(false);
    }
});
```

Async / Await

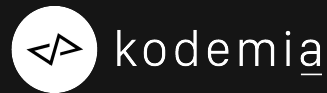


Opción 1



```
cont main = () =>{  
  encontrarCero.then((valor) =>{  
    // Bloque de código si se resuelve la promesa satisfactoriamente  
  
    console.log(valor);  
  }).catch((error)=>{  
    // Bloque de código si la promesa es rechazada  
  
    console.log(error);  
  });  
}
```

Async /Await



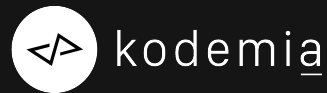
Opción 2



```
// Esto se vuelve una promesa
const main = async () =>{
  const result = await encontrarCero;
  console.log(result);
}
```

Manejo de errores Async / Await

Manejo de errores Async /Await



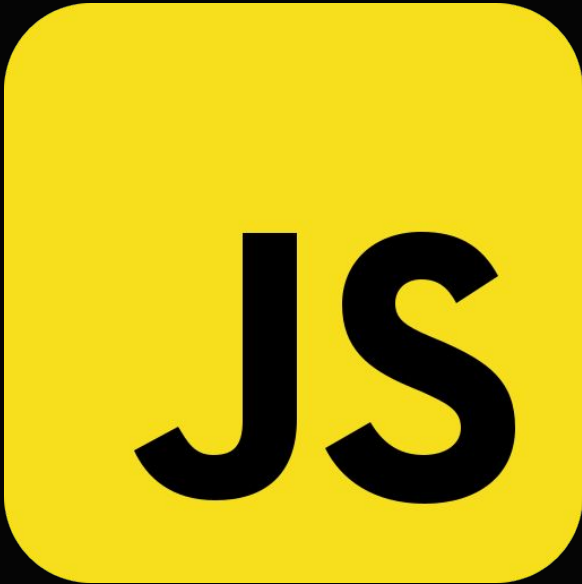
Cuando una función utiliza **await** para esperar la respuesta de las promesas, en algunas ocasiones el resultado puede ser un rechazo, para manejar el elemento antes mencionado se debe envolver la función en un bloque **try / catch**

```
const main = async () =>{
  try{
    // bloque de código si todo es correcto
    const result = await encontrarCero;
    console.log(result);
  }catch(error){
    // bloque de código en caso de presentar algún error
    console.log(error);
  }
}
```

Temas puntuales de la sección



- Código Sincrono y Asincrono
- ¿Que es una promesa?
- Objeto Promise
 - Creación de una promesa
 - Usar una promesa
- Async / Await
- Manejo de errores Async / Await



JS



11

Promesas

Yael Ruiz Pucheta