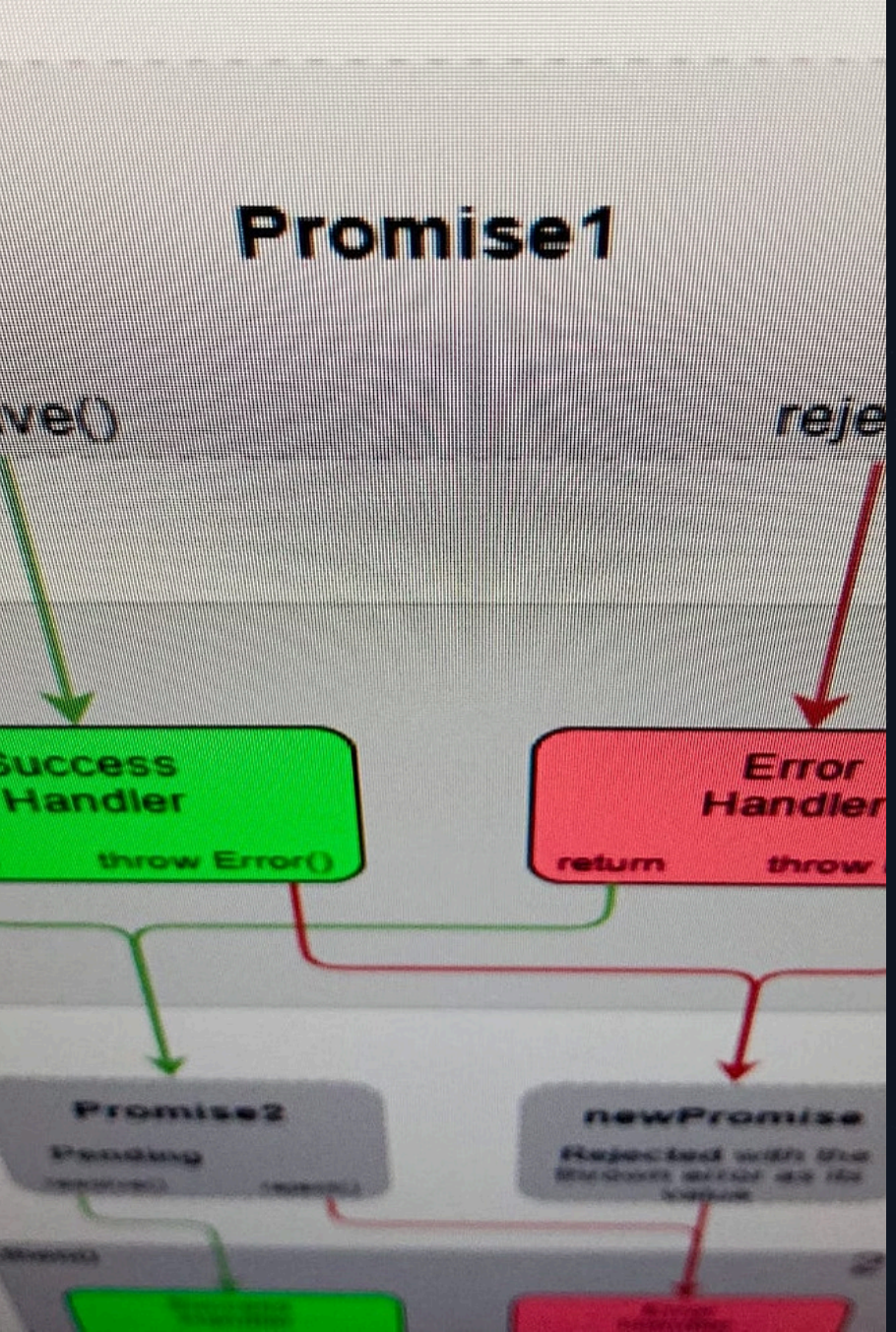
Async/Await в JavaScript

Сучасний підхід до асинхронного програмування, який робить код читабельним та зрозумілим



Ключове слово async

Що таке async?

Ключове слово async використовується для оголошення асинхронних функцій. Це спрощує роботу з Promise та робить асинхронний код більш читабельним, особливо при роботі з великою кількістю операцій та ланцюжками викликів.

Асинхронні функції **завжди повертають Promise**, навіть якщо ви повертаєте звичайне значення. Це одна з ключових особливостей async функцій.

Приклад використання

```
async function hello() {
  return 'Hello'
}
```

Виклик функції повертає Promise:

```
hello().then(console.log)
// Виведе: Hello
```

💡 Значення автоматично перетворюється на Promise

Ключове слово await

Призупинення виконання

Await призупиняє виконання функції до завершення асинхронної операції

Отримання результату

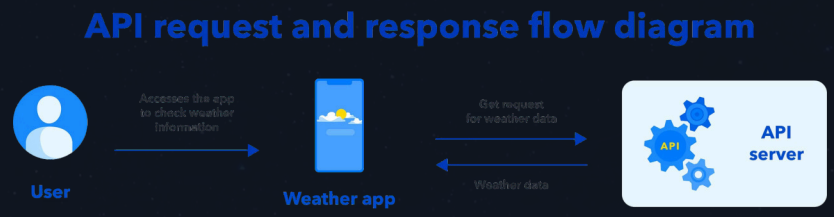
Після завершення Promise повертається результат і виконується наступний код

Тільки в async функціях

Await можна використовувати лише всередині функцій, оголошених з async

Практичний приклад

```
async function fetchData() {
  const response = await fetch(
    'https://swapi.dev/api/people/1/'
  );
  const data = await response.json();
  return data;
}
```



У цьому прикладі `await` використовується двічі: спочатку для очікування завершення HTTP-запиту, потім для аналізу JSON-відповіді. Код виглядає синхронним, але виконується асинхронно.

Обробка помилок

Обробка помилок в async/await здійснюється за допомогою знайомої конструкції **try...catch**. Це дозволяє ловити помилки та обробляти їх у блоці catch, як у синхронному коді.

01

Спроба виконання

Код виконується в блоці try

02

Перехоплення помилок

Помилки ловляться в блоці catch

03

Обробка та реакція

Виконується логіка обробки помилки

Приклад обробки помилок

```
async function fetchData() {
 try {
 const response = await fetch(
 'https://api.example.com/data'
 );
 if (!response.ok) {
 throw new Error('Request failed');
 }
 const data = await response.json();
 return data;
 } catch (error) {
 console.error('An error occurred:', error);
 throw error;
 }
}
```