Programming Hero

Promise

Promise

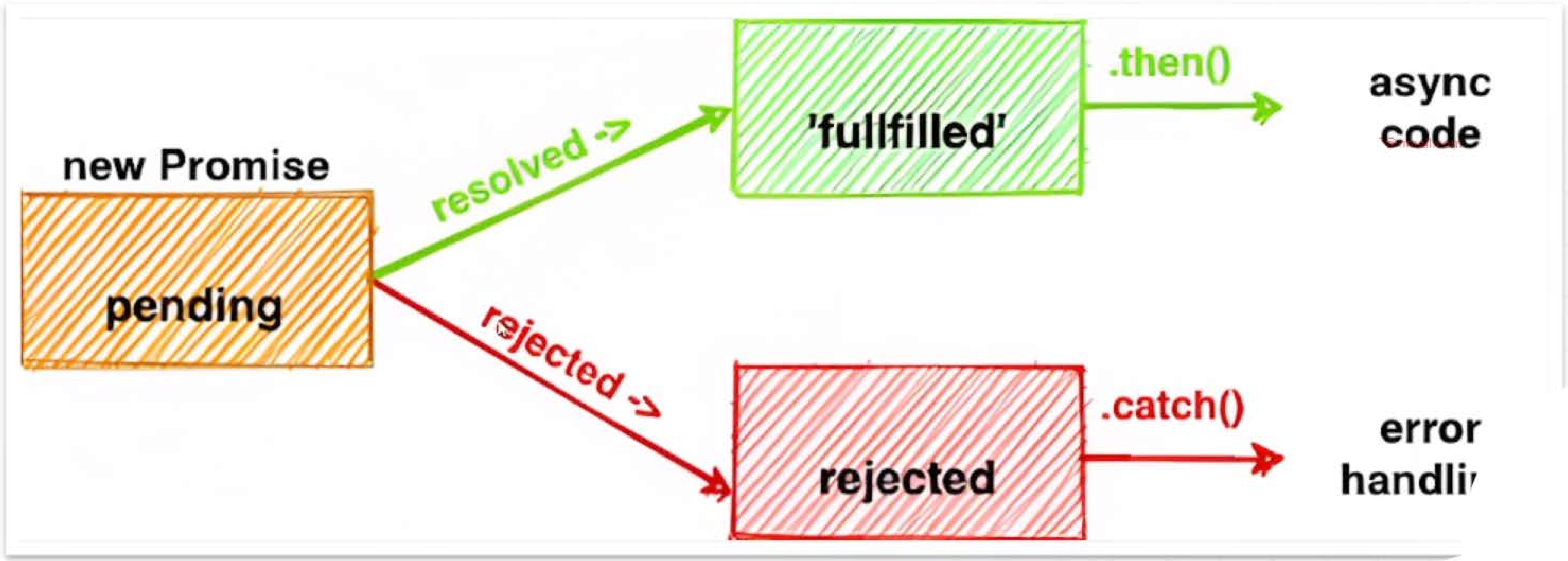
The Promise object represents the eventual completion (or failure) of an asynchronous operation and its resulting value. A Promise is in one of these states:

1 **Pending**
initial state, neither fulfilled nor rejected.

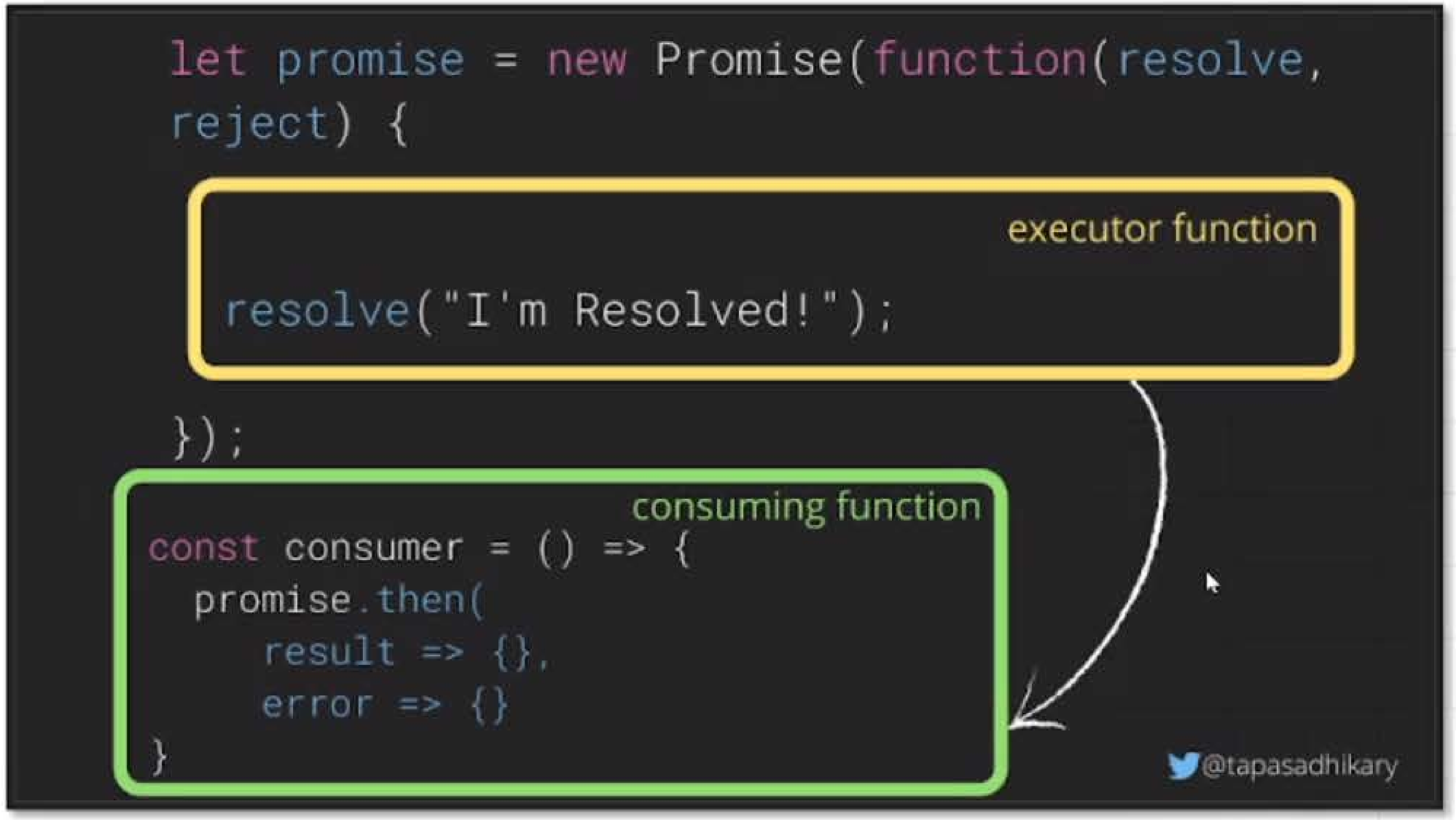
2 **Resolve**
the operation was completed successfully.

3 **Reject**
the operation failed.

Promise Structure



Examples

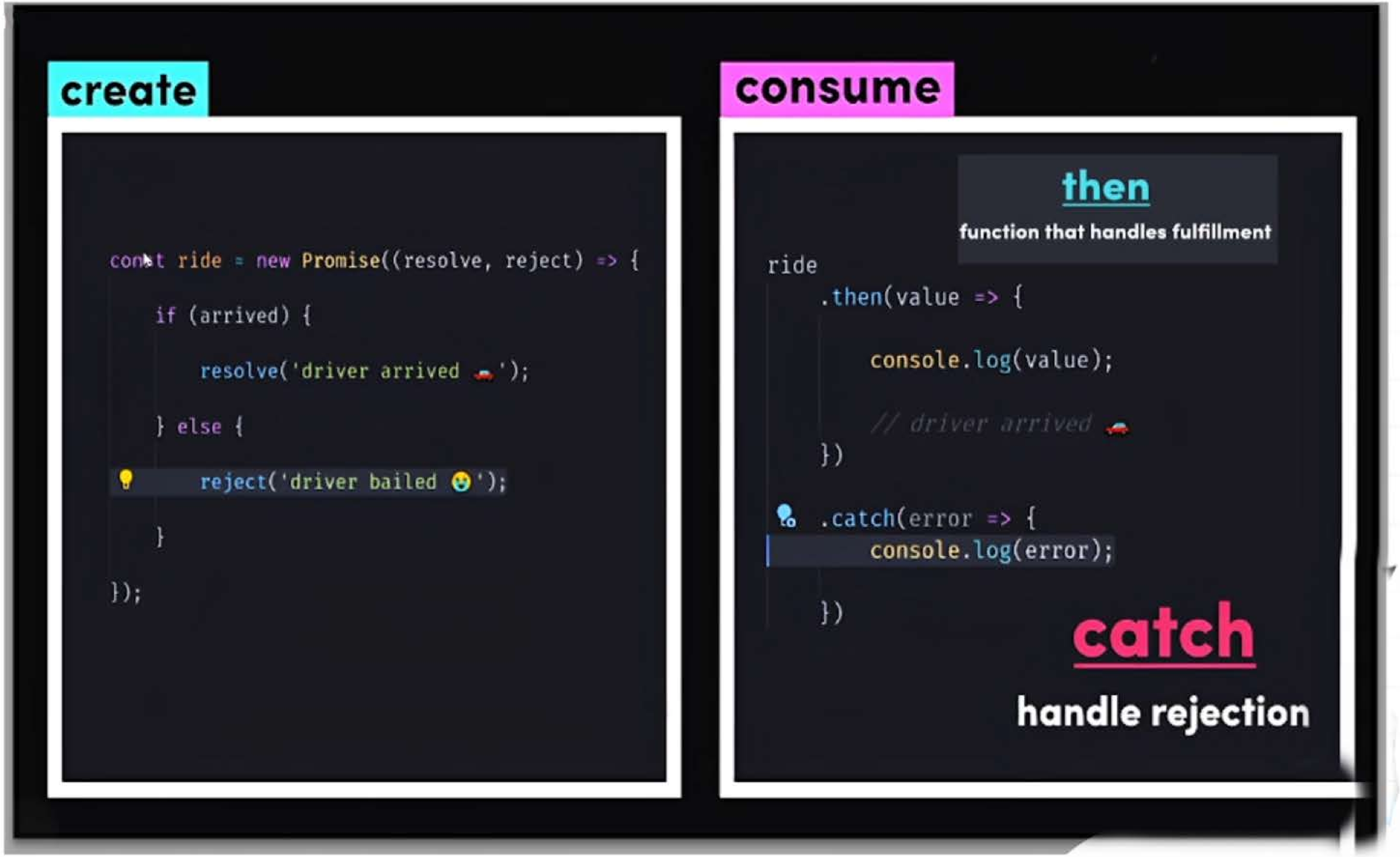


```
let promise = new Promise(function(resolve,
reject) {

  resolve("I'm Resolved!");

});
const consumer = () => {
  promise.then(
     result => {},
     error => {}
}
```

Create & Consume



create

```
const ride = new Promise((resolve, reject) => {
    if (arrived) {
        resolve('driver arrived 🚗');
    } else {
        reject('driver bailed 😱');
    }
});
```

consume

then

function that handles fulfillment

```
ride
    .then(value => {
        console.log(value);
        // driver arrived 🚗
    })
    .catch(error => {
        console.log(error);
    })
```

catch

handle rejection

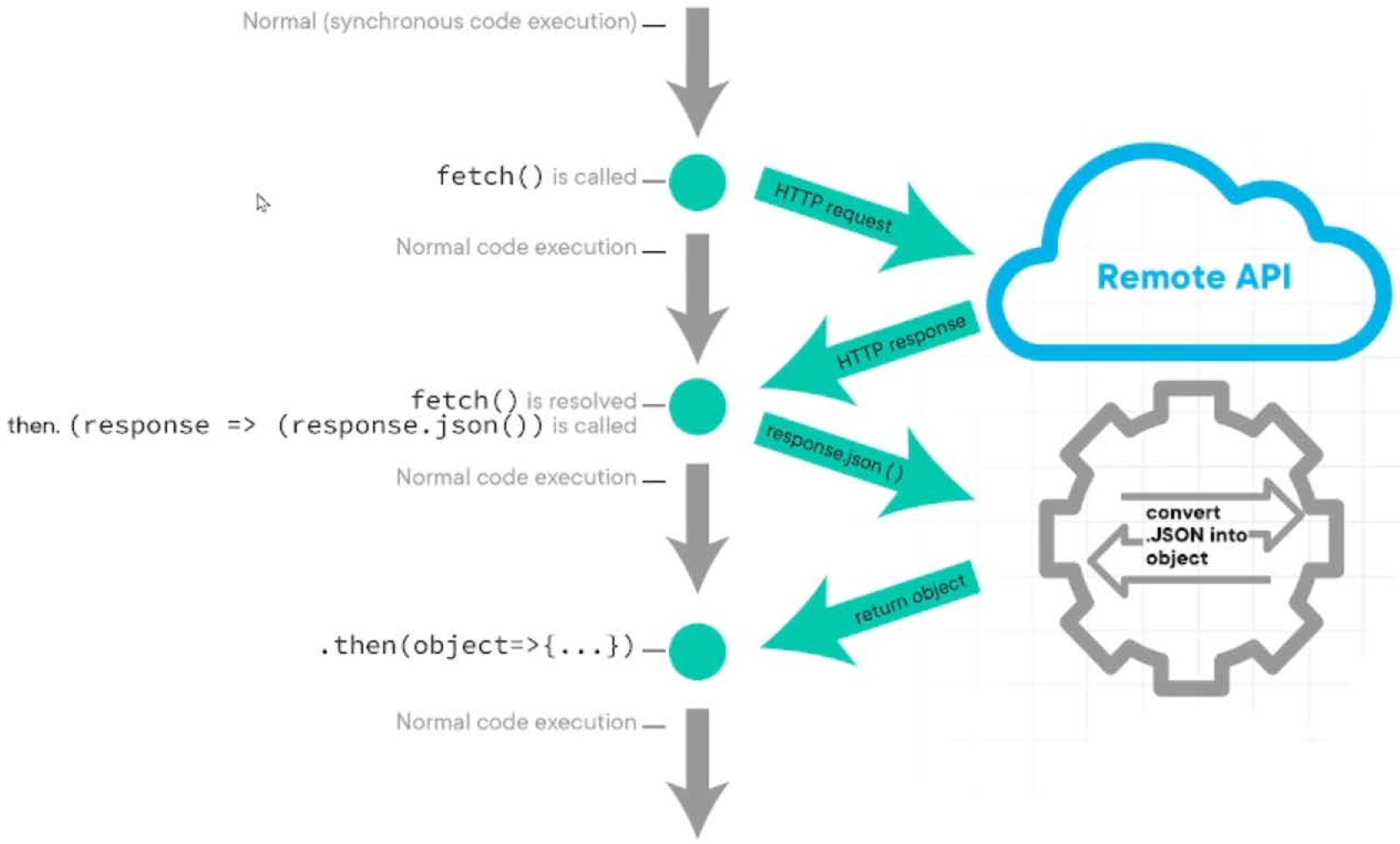
Fetch API

01 The fetch() method starts the process of fetching a resource from a server

02 The fetch() method returns a Promise that resolves to a Response object.

03 A fetch() method only rejects when a network error is encountered.

Fetch API


Normal (synchronous code execution)
fetch() is called
Normal code execution
fetch() is resolved
then. (response => (response.json()) is called
Normal code execution
.then(object=>{...})
Normal code execution
HTTP request
HTTP response
response.json ()
return object
Remote API
convert .JSON into object

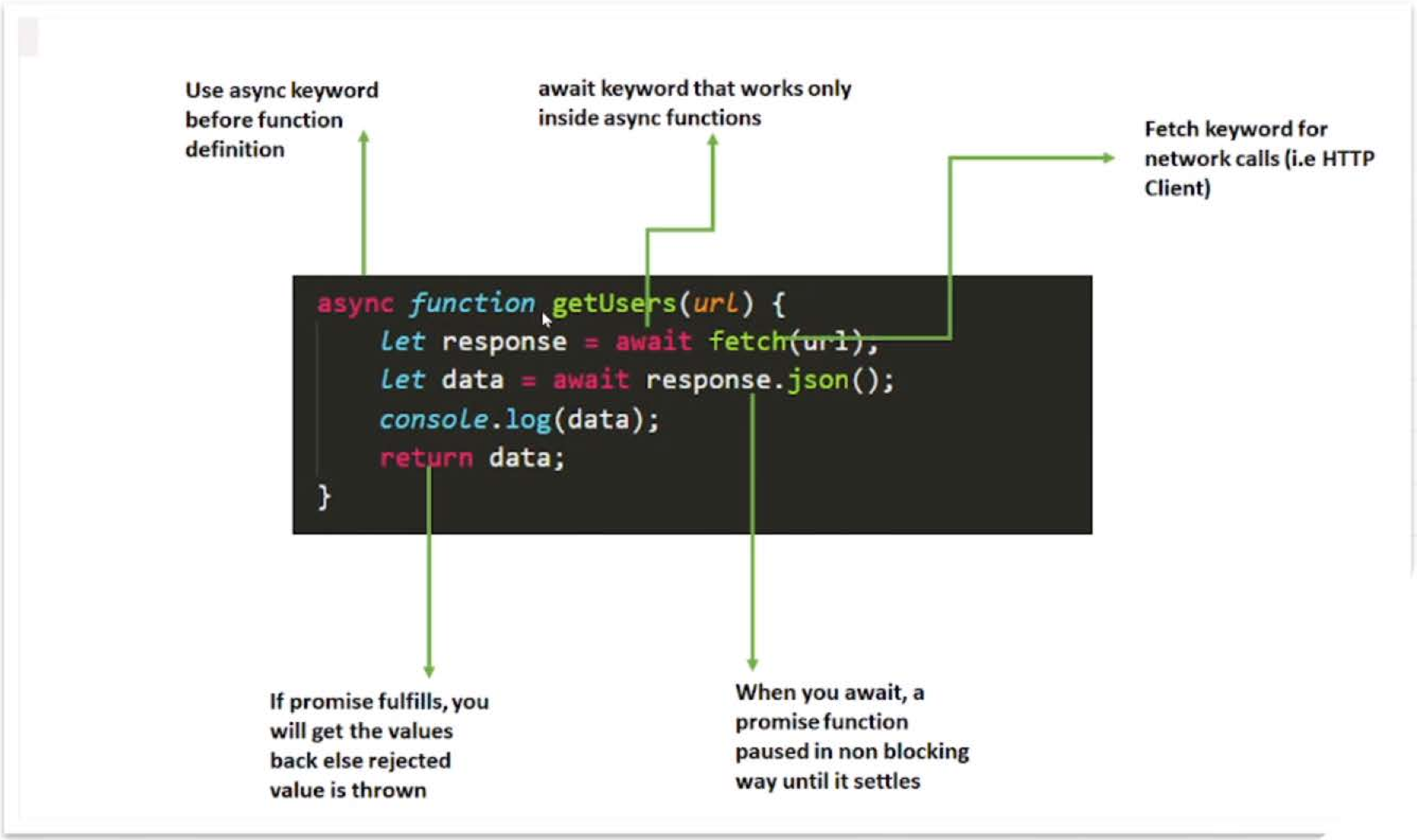
Programming Hero

Async/Await

Async/Await

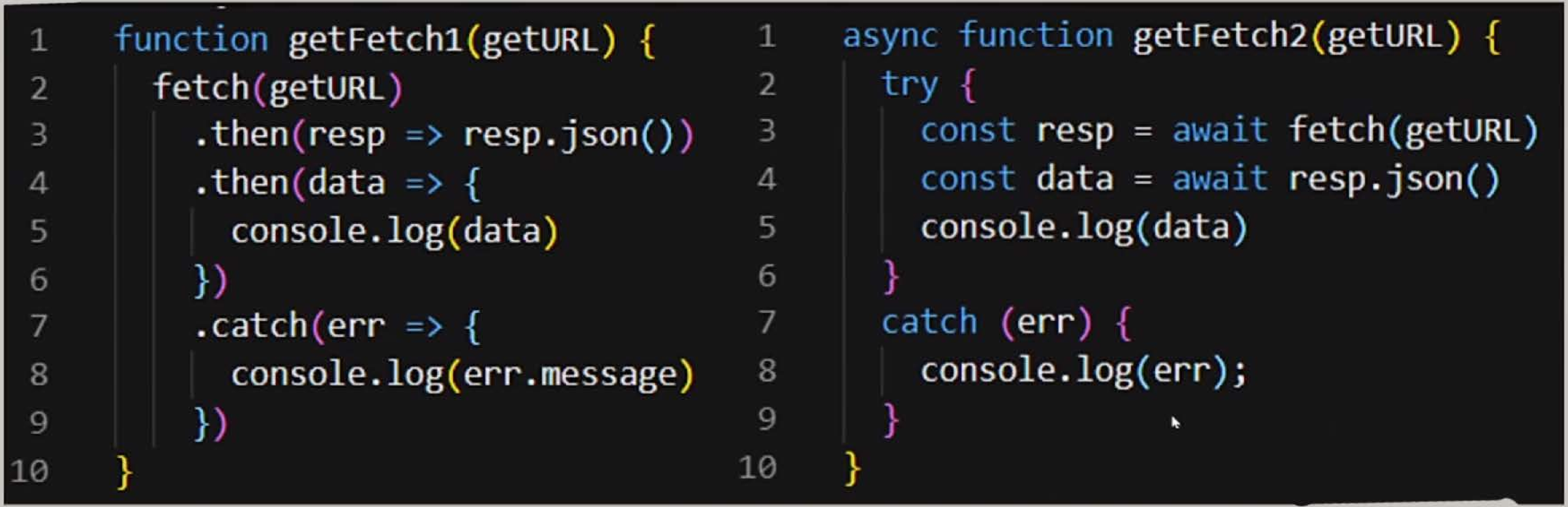
- "async/await" is a special syntax to work with promises in a more comfortable way
- It's surprisingly easy to understand and use.
- The await keyword can only be used inside an async function.
- The await keyword makes the function pause the execution and
- wait for a resolved promise before it continues

Async/Await


Use async keyword before function definition
await keyword that works only inside async functions
Fetch keyword for network calls (i.e HTTP Client)
If promise fulfills, you will get the values back else rejected value is thrown
When you await, a promise function paused in non blocking way until it settles

```
async function getUsers(url) {
    let response = await fetch(url);
    let data = await response.json();
    console.log(data);
    return data;
}
```

Fetch vs Async/Await



```
function getFetch1(getURL) {
  fetch(getURL)
    .then(resp => resp.json())
    .then(data => {
      console.log(data)
    })
    .catch(err => {
      console.log(err.message)
    })
}
```

```
async function getFetch2(getURL) {
  try {
    const resp = await fetch(getURL)
    const data = await resp.json()
    console.log(data)
  }
  catch (err) {
    console.log(err);
  }
}
```

Programming Hero

We will learn more about JS in our next videos