Capture error and data in async-await without try-catch

Adarsh Apr 30

#javascript #webdev **#backend** **#frontend**

One of the things that took the Javascript community by storm was the introduction of async-await. It was simple and looked a lot better than the then-catch of Promises and also more readable and debuggable than the callback hell. But one thing that bothered me was the usage of try-catch. At first I thought it isn't a problem but as fate would have it I was working on chained API calls and the problem cropped up where each API call has specific error message that had to be printed. I realized soon that I was creating a try-catch hell.

Let's consider this Promise that resolves or rejects after 2 seconds based on a parameter `rejectPromise`

```
// api.js

const fetchData = async (duration, rejectPromise) => (
  new Promise((resolve, reject) => {
    setTimeout(() => {
      if (rejectPromise) {
        reject({
          error: 'Error Encountered',
          status: 'error'
        })
      }
      resolve({
        version: 1,
        hello: 'world',
```

```
module.exports = {
  fetchData,
};
```

So my typical usage of async-await is gonna be like this.

```
const { fetchData } = require('./api');

const callApi = async () => {
  try {
    const value = await fetchData(2000, false);
    console.info(value);
  } catch (error) {
    console.error(error);
  }
}

callApi();

/*
 OUTPUT:
 { version: 1, hello: 'world' } (rejectPromise=false)

 { error: 'Error Encountered', status: 'error' } (rejectPromise=true)

 */
```

As you can see when the `rejectPromise` parameter is `false` the await resolves to `{ version: 1, hello: 'world' }` and when it's `true` it rejects the promise and catch is called and the error is `{ error: 'Error Encountered', status: 'error' }`.

will leverage the promise functions then-catch to make the process simpler. Let's write a wrapper that does this.

```
// wrapper.js

const wrapper = promise => (
  promise
    .then(data => ({ data, error: null }))
    .catch(error => ({ error, data: null }))
);

module.exports = wrapper;
```

We can see that the wrapper takes a promise as an input and returns the resolved/rejected values through then-catch. So let's go and modify the original code we wrote in try-catch to utilize the wrapper.

```
const { fetchData } = require('./api');
const wrapper = require('./wrapper');

const callApi = async () => {
  const { error, data } = await wrapper(fetchData(2000, false));
  if (!error) {
    console.info(data);
    return;
  }
  console.error(error);
}

callApi();

/*
 OUTPUT:
 { version: 1, hello: 'world' } (rejectPromise=false)
```

```
*/
```

Voila the same output but the this way makes it better to understand the code.

Adarsh + FOLLOW

A Software Engineer with the mind of a curious kid who likes to explore and deep dive into frameworks, libraries and computers in general.

aadi2203 sadarshannaiynar

rhymes

Apr 30

Nice!

FYI: You might still need try..catch though, the wrapper catches asynchronous errors but if inside `callApi` anything you call issues a "standard" error that exception will bubble up to the user.

```
  const { error, data } = await wrapper(fetchData(2000, false));
  callAFunctionThatDividesByZero()
  if (!error) {
    console.info(data);
    return;
  }
  console.error(error);
}
```

2 REPLY

Adarsh

Apr 30

Thanks! :) And Yeah on regular exceptions I would need a catch but I was focusing on the API part only! :)

2 REPLY

rhymes

Apr 30

I like it, it reminds me of how Go does error handling :D

2 THREAD

Adarsh

Apr 30

I still haven't worked with Go yet planning to start soon.

1 REPLY

Yury Matuhin

May 22

Already existed github.com/scopsy/await-to-js and my version github.com/ymatuhin/flatry (I just like you didn't know about await-to-js).

2 REPLY

Adarsh

May 25

Didn't know about that. Thanks for sharing :)

1 REPLY

code of conduct - report abuse

Classic DEV Post from Apr 20

What is git rebase?


Jim Borden

A simple explanation of the git rebase feature. It's something you should definitely use!

118 7

READ POST SAVE FOR LATER

Classic DEV Post from Dec 12 '17

What I Learned From Not Planning A Web App (from start to finish)


jen chan

I got an art commission to make a stylized website that included a question-and-answer form sent to a columnist's email. "Simple enough", I thought. I was wrong.

72 13

READ POST SAVE FOR LATER

Another Post You Might Like

How I Doubled my Salary in 5 Months and Got an Amazing Job


Sam Williams

It certainly wasn't easy but I managed to go from a low paid junior dev to a we...

159 13

READ POST SAVE FOR LATER

An Overview of JavaScript Functions

HowToCodejs - May 31

How I learned to Stop Looping and Love the Iterator

Kushan Joshi - May 30

Render Props and Higher Order Components

K 🥸 - May 30

How to make your Javascript code run faster

Nahuel Scotti - May 31