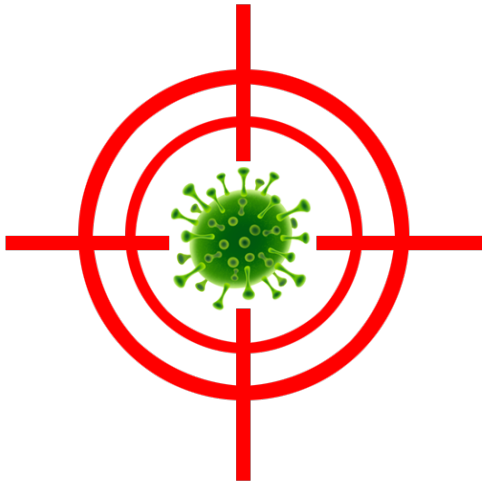


Async Error Handling

in Javascript

by @AllieBrosh

Jorge Marín

April 15th, 2020 – AllTheTalks Online

https://jorgemarin.me
@chipironcin

Hidden slide

Welcome to AllTheTalks Online and thanks for coming to my talk 🤩

UTC **Spain (UTC+2)**	**09:35** **11:35**
San Francisco (UTC-8)	02:35
London (UTC+1)	10:35
Moscow (UTC+3)	12:35
New Delhi (UTC+5:30)	15:05
Shanghai (UTC+8)	17:35
Singapore (UTC+8)	17:35
Sydney (UTC+10)	19:35

breathe *no, srsly, breathe*

My name is Jorge, and I am an engineer 👋🏻 🇪🇸

Async Error Handling

in Javascript

Jorge Marín
April 15th, 2020 – AllTheTalks Online

https://jorgemarin.me
@chipironcin

Poll time - Raise your hands if you…

Poll time - Raise your hands if you…

Use the comments section!

Follow me on Twitter!

Like and subscribe!

(after the talk)

https://jorgemarin.me
@chipironcin

Today I am going to talk a bit about

- TryCatch blocks
- Asynchronous programming
- Error stack traces
- Node 12 🎉 (April 23rd 2019)

Error handling in Javascript

```
function goToTheBeach(location) {
  if(location === "Spain") return false
  return true
}

if(goToTheBeach("Antartica"))
  console.log("Success");
else
  console.log("Failure");
```

Asynchronous programming

Asynchronous programming is a design pattern which ensures the non-blocking code execution.

JavaScript is asynchronous in nature and so is Node.

Asynchronous code executes without having any dependency and no order. This improves the system efficiency and throughput.

Promises (example)

```
function goToTheBeach(location) {
  if(location === "Spain") return false
  return new Promise(resolve => setTimeout(resolve, 1000));
}

(async () => {
  goToTheBeach("Antartica").then(() => console.log("Success"));
})();
```

Promises (example)

```
function goToTheBeach(location) {
  if(location === "Spain") return false
  return new Promise(resolve => setTimeout(resolve, 1000));
}

(async () => {
  goToTheBeach("Antartica")
    .then(() => console.log("Success"))
    .catch((err) => console.log("Failure", err));

  console.log("Turning on the radio in the meantime")
})();
```

Promi

```
function goToTheBeach(location) {
  if(location === "Spain") return false
  return new Promise(resolve => setTimeout(resolve, 1000));
}

function makeBocataDeTortilla() {
  return new Promise(resolve => setTimeout(resolve, 1000));
}

(async () => {
  goToTheBeach("Antartica")
    .then(() => {
      console.log("Success");
      makeBocataDeTortilla()
        .then(() => console.log("Bocata is ready"))
        .catch((err) => console.log("Failed to make bocata", err));
    })
    .catch((err) => console.log("Failure", err));

  console.log("Turning on the radio in the meantime")
})();
```

Promise

```
function goToTheBeach(location) {
  if(location === "Spain") return false
  return new Promise(resolve => setTimeout(resolve, 1000));
}

function makeBocataDeTortilla() {
  return new Promise(resolve => setTimeout(resolve, 1000));
}

(async () => {
  goToTheBeach("Antartica")
    .then(() => console.log("Success"))
    .then(() => makeBocataDeTortilla())
    .then(() => console.log("Bocata is ready"))
    .catch((err) => console.log("Failure", err));

  console.log("Turning on the radio in the meantime")
})();
```

Async/Await (example)

```
function goToTheBeach(location) {
  if(location === "Spain") return false
  return new Promise(resolve => setTimeout(resolve, 1000));
}

(async () => {
  await goToTheBeach("Antartica");
})();
```

Error handling in Javascript: Try Catch

```
try {
  Block of code to try
}
catch(err) {
  Block of code to handle errors
}
```

```
function goToTheBeach() {
 if(location === "Spain") throw new Error("Please stay at home");
}

try {
  goToTheBeach();
}
catch(err) {
  console.log(err);
}
```

https://jorgemarin.me
@chipironcin

Async/Await (example)

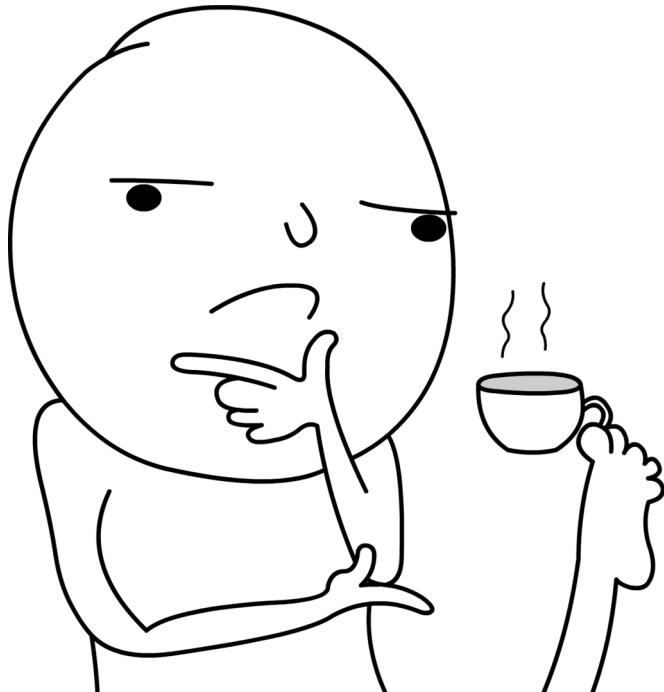
```
function goToTheBeach(location) {
  if(location === "Spain") return false
  return new Promise(resolve => setTimeout(resolve, 1000));
}

function makeBocataDeTortilla() {
  return new Promise(resolve => setTimeout(resolve, 1000));
}

(async () => {
  try {
    await goToTheBeach("Antartica");
    console.log("Success");
    await makeBocataDeTortilla();
    console.log("Bocata is ready");
  } catch(err) {
    console.log("Failure", err);
  }
})();
```

```javascript
function goToTheBeach(location) {
  if(location === "Spain") return false
  return new Promise(resolve => setTimeout(resolve, 1000));
}

function makeBocataDeTortilla() {
  return new Promise(resolve => setTimeout(resolve, 1000));
}

(async () => {
  try {
    await goToTheBeach("Antartica");
    console.log("Success");
  } catch(err) {
    console.log("Failure", err);
    process.exit(1);
  }

  try {
    await makeBocataDeTortilla();
    console.log("Bocata is ready");
  } catch(err) {
    console.log("Failed to make bocata", err);
  }
})();
```

https://jorgemarin.me
@chipironcin

Error handling in Javascript:
~~Try Catch~~ The Golang Way

```
data, err := db.Query("SELECT ...")
if err != nil { return err }
```



```javascript
async function goToTheBeach(location) {
  let res;
  if(location === "Spain") return [new Error("Forbidden")];

  try {
    res = await new Promise(resolve => setTimeout(resolve, 1000));
    return [null, res];
  } catch (err) {
    return [err]
  }
}

async function makeBocataDeTortilla() {
  try {
    res = await new Promise(resolve => setTimeout(resolve, 1000));
    return [null, res];
  } catch (err) {
    return [err]
  }
}

(async () => {
  let [err, res] = await goToTheBeach("Antartica");

  if(err) {
    console.log("Failure");
    process.exit(1);
  }

  console.log("Success");
  [err, res] = await makeBocataDeTortilla();

  if(err) {
    console.log("Failed to make bocata");
  }
  console.log("Bocata is ready");
})();
```

https://jorgemarin.me
@chipironcin

Error handling in Javascript: ~~Try Catch~~ The Golang Way

```javascript
// to.js
export default function to(promise) {
    return promise.then(data => {
        return [null, data];
    })
    .catch(err => [err]);
}
```

https://blog.grossman.io/how-to-write-async-await-without-try-catch-blocks-in-javascript/

https://jorgemarin.me
@chipironcin

```javascript
function to(promise) {
   return promise.then(data => {
      return [null, data];
   })
   .catch(err => [err]);
}
// ---

function goToTheBeach(location) {
  let res;
  if(location === "Spain") return Promise.reject();
  return new Promise(resolve => setTimeout(resolve, 1000));
}

function makeBocataDeTortilla() {
  return new Promise(resolve => setTimeout(resolve, 1000));
}

(async () => {
  let [err, res] = await to(goToTheBeach("Antartica"));

  if(err) {
    console.log("Failure");
    process.exit(1);
  }

  console.log("Success");
  [err, res] = await to(makeBocataDeTortilla());

  if(err) {
    console.log("Failed to make bocata");
  }
  console.log("Bocata is ready");
})();
```

https://jorgemarin.me
@chipironcin

```
}
// ---

function goToTheBeach(location) {
  let res;
  if(location === "Spain") return Promise.reject();
  return new Promise(resolve => setTimeout(resolve, 1000));
}

function makeBocataDeTortilla() {
  return new Promise(resolve => setTimeout(resolve, 1000));
}
```

```
function goToTheBeach(location) {
  if(location === "Spain") return false
  return new Promise(resolve => setTimeout(resolve, 1000));
}

function makeBocataDeTortilla() {
  return new Promise(resolve => setTimeout(resolve, 1000));
}
```

```
(async () => {
  let [err, res] = await to(goToTheBeach("Antartica"));

  if(err) {
    console.log("Failure");
    process.exit(1);
  }

  console.log("Success");
  [err, res] = await to(makeBocataDeTortilla());

  if(err) {
    console.log("Failed to make bocata");
  }
  console.log("Bocata is ready");
})();
```

```
(async () => {
  try {
    await goToTheBeach("Antartica");
    console.log("Success");
  } catch(err) {
    console.log("Failure", err);
    process.exit(1);
  }

  try {
    await makeBocataDeTortilla();
    console.log("Bocata is ready");
  } catch(err) {
    console.log("Failed to make bocata", err);
  }
})();
```

Error handling in Javascript: ~~Try Catch~~ The Golang Way

- *Dima Grossman (@dimagrossman)*
- https://blog.grossman.io/how-to-write-async-await-without-try-catch-blocks-in-javascript/

- *"This post is just a different way of looking on async/await error handling. It should not be used as a goto for every async/await function you write and in a lot cases having a single catch at the top will do just fine. Sometimes we don't want to expose the error object of the implementation of the model and want instead to provide a custom error object […]"*

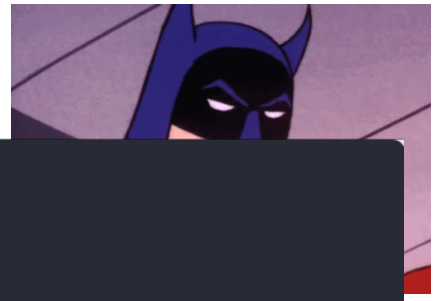
Async (errors)

```
async function goToTheBeach(location) {
  if(location === "Spain") return Promise.reject();
  await new Promise(resolve => setTimeout(resolve, 1000));
  await makeBocataDeTortilla();
}

(async () => {
  await goToTheBeach("Antartica");
})();
```

Async (stack trace)

```
$ node badStackTrace.js
(node:1) UnhandledPromiseRejectionWarning: ReferenceError: makeBocataDeTortilla is not defined
    at goToTheBeach (/usr/src/app/badStackTrace.js:4:3)
```

Async (better stack trace)



```javascript
async function goToTheBeach(location) {
  const startStack = new Error().stack;

  if(location === "Spain") return Promise.reject();
  await new Promise(resolve => setTimeout(resolve, 1000));

  try {
    await makeBocataDeTortilla();
  } catch(err) {
    err.stack = err.stack + "\n" + startStack.substring(startStack.indexOf("\n") + 1);
    throw err;
  }
}

(async () => {
  await goToTheBeach("Antartica");
})();
```

Async (better stack trace)

```
$ node betterStackTrace.js
(node:9843) UnhandledPromiseRejectionWarning: ReferenceError: makeBocataDeTortilla is not defined
    at goToTheBeach (/Users/jorgemarin/betterStackTrace.js:8:5)
    at <anonymous>
    at goToTheBeach (/Users/jorgemarin/betterStackTrace.js:2:22)
    at /Users/jorgemarin/betterStackTrace.js:16:9
    at Object.<anonymous> (/Users/jorgemarin/betterStackTrace.js:17:3)
    at Module._compile (module.js:652:30)
    at Object.Module._extensions..js (module.js:663:10)
    at Module.load (module.js:565:32)
    at tryModuleLoad (module.js:505:12)
    at Function.Module._load (module.js:497:3)
    at Function.Module.runMain (module.js:693:10)
    at startup (bootstrap_node.js:188:16)
```

Async (with NodeJS 12, now LTS)

```
$ docker run -it --rm -v "$PWD":/usr/src/app -w /usr/src/app node:10-alpine node badStackTrace.js
(node:1) UnhandledPromiseRejectionWarning: ReferenceError: makeBocataDeTortilla is not defined
    at goToTheBeach (/usr/src/app/badStackTrace.js:4:3)
```

```
$ docker run -it --rm -v "$PWD":/usr/src/app -w /usr/src/app node:12-alpine node badStackTrace.js
(node:1) UnhandledPromiseRejectionWarning: ReferenceError: makeBocataDeTortilla is not defined
    at goToTheBeach (/usr/src/app/badStackTrace.js:4:3)
--> at async /usr/src/app/badStackTrace.js:8:3
```

Async (with NodeJS 12, now LTS)

```
async function wait_1(x) {
 await wait_2(x)
}

async function wait_2(x) {
 await wait_3(x);
}

async function wait_3(x) {
 await x;

 throw new Error("Oh boi")
}

wait_1(1).catch(e => console.log(e.stack));
```

https://blog.risingstack.com/node-js-12-new-features/

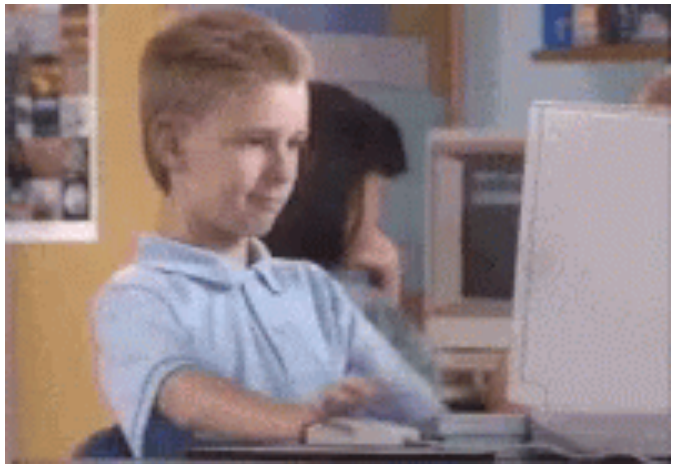
Async (with NodeJS 12, now LTS)

```
➜  ~/work/node-12  node index.js
Error: Oh boi
    at wait_3 (/Users/matehuszarik/work/node-12/index.js:21:9)
    at <anonymous>
    at process._tickCallback (internal/process/next_tick.js:188:7)
    at Function.Module.runMain (module.js:695:11)
    at startup (bootstrap_node.js:188:16)
    at bootstrap_node.js:609:3
```

```
➜  ~/work/node-12  node index.js
Error: Oh boi
    at wait_3 (/Users/matehuszarik/work/node-12/index.js:21:9)
    at process.runNextTicks [as _tickCallback] (internal/process/task_queues.js:54:5)
    at Function.Module.runMain (internal/modules/cjs/loader.js:828:11)
    at internal/main/run_main_module.js:17:11
    at async wait_2 (/Users/matehuszarik/work/node-12/index.js:15:3)
    at async wait_1 (/Users/matehuszarik/work/node-12/index.js:11:3)
```

https://blog.risingstack.com/node-js-12-new-features/

Async (with NodeJS 12, now LTS)



Recap

- Beautiful alternative way for error handling in Javascript
- Use NodeJS 12 to get better error traces from async functions

- Be happy
- Be kind
- Stay safe

Resources

https://medium.com/front-end-weekly/error-handling-in-node-javascript-suck-unless-you-know-this-2018-aa0a14cfdd9d

https://dev.to/oieduardorabelo/javascript-handling-errors-like-go-3efk

https://blog.grossman.io/how-to-write-async-await-without-try-catch-blocks-in-javascript/

https://blog.risingstack.com/node-js-12-new-features/

Use the comments section!

Follow me on Twitter!

Like and subscribe!

https://jorgemarin.me
@chipironcin

That’s all folks

https://jorgemarin.me
@chipironcin