Async Function

1. Async function syntax
2. Handle errors
3. Parallel and sequence APIs

1. Async function syntax

```
// Async function declaration
async function func1() {}
// Async function expression
const func2 = async function () {};
// Async arrow function
const func3 = async () => {};
// Async method definition in an object literal
const obj = { async say() {} };
```

await should be used in **async function**

```
// works
async function fetchData() {
    const data = await studentApi.getAll();
    console.log(data);
}
// error
function fetchData() {
    const data = await studentApi.getAll();
    console.log(data);
}
```

async function **always** return Promise

```
async function getNumber() {
    return 10;
}
getNumber(); // it returns a promise
```

```
async function getNumber() {
    return Promise.resolve(10);
}
getNumber(); // it returns a promise
```

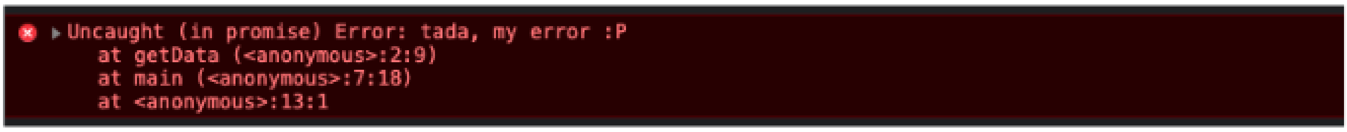
2. Handle errors

IMPORTANT: always handle errors when using Promise / calling APIs

```
async function getAllStudent() {
    try {
        const url = 'http://js-post-api.herokuapp.com/api/students?_page=1';
        const response = await fetch(url);
        const data = await response.json();
        console.log(data);
    } catch (error) {
        console.log(error);
    }
}
```

DON'T FORGET await keyword 🔥

```
async function getData() {
    // return Promise.reject(new Error('tada, my error :P'))
    throw new Error('tada, my error :P')
}
async function main() {
    try {
        const data = getData();
    } catch (error) {
        console.log(error);
    }
}
main();
```



```
Uncaught (in promise) Error: tada, my error :P
    at getData (<anonymous>:2:9)
    at main (<anonymous>:7:18)
    at <anonymous>:13:1
```

4. Parallel and sequence APIs

- Parallel: use Promise.all()
- Sequence APIs: use async/await

```
const promiseList = [
    getAllStudent,
    getAllCategories,
    getAllCities,
];
// Trigger API calls at the same time
Promise.all(promiseList)
    .then(([studentList, categoryList, cityList]) => {
        console.log(studentList, categoryList, cityList);
    })
    .catch(error => console.log(error));
```

> Source: https://exploringjs.com/impatient-js/ch_async-functions.html

Tham khảo

- https://exploringjs.com/es2016-es2017/ch_async-functions.html
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/async_function
- Top-level await keyword: https://github.com/tc39/proposals/blob/main/finished-proposals.md