Node.js - Laboratorium 5

Obsługa błedów (`try..catch`)

Konstrukcja bloku `try..catch`:

```
try {
    // some code ...
} catch(error) {
    // catch error ...
} finally {
    // execute this code after try/catch block
}
```

`instanceof`

```
const someError = new SyntaxError('abc');
console.log(someError instanceof ReferenceError); // false
console.log(someError instanceof SyntaxError); // true
console.log(someError instanceof Error); // true
```

`Promise` z wykorzystaniem `async/await`

Konstrukcja bloku `async/await`

`async`

```
async function someFunction(someArg) {
    return ...; // ciało funkcji
}
```

lub

```
const someFunction = async (someArg) => {
    return ...; // ciało funkcji
}
```

`await`

```
const someVar = await <Promise>;
```

> przykład wykorzystania

```
const axios = require('axios');

// IIFE
(async function () {
    const response = await axios ('https://jsonplaceholder.typicode.com/users/1');
    console.log(response.data.name);
})();
```

Przydatne linki

Dodatkowa lektura uzupełniająca wiedzę odnośnie `error handling` oraz `async/await Promise`

- https://javascript.info/try-catch
- https://javascript.info/async-await

Zadania do wykonania na laboratorium

1. W katalogu `01` znajduje się plik `user.json`. Wykorzystując wiedzę z poprzednich zajęć, stwórzmy aplikację wczytującą naszego użytkownika z pliku i zamieńmy go na obiekt JS oraz wyświetlmy w konsoli jego imię.

Użyjmy w tym zadaniu funkcję `readFileSync` z wbudowanego modułu `fs` oraz funkcję `JSON.parse` do przeparsowania wczytanej zawartości do obiektu.

Zabezpieczmy naszą aplikację tak aby wyłapać błąd prasowania lub odczytu pliku i poinformujmy o tym użytkownika.

2. Stwórzmy aplikację która będzie posiadała funkcję dzielenia 2 liczb.

Jak wiadomo JS jest ciekawym językiem, który pozwala dzielić przez 0... efektem dzielenia przez zero jest wartość `Infinity`, np.:

```
const result = 2 / 0;
console.log(result); // => Infinity
```

Zabezpieczmy naszą aplikację tak aby funkcja dzielenia rzucała wyjątkiem w przypadtku gdy drugi parametr ma wartość 0 (np. `new Error('divide by 0')`).

Oczywiście stwórzmy przykładowe wywołanie naszej funkcji z blokiem `try..catch`.

3. Wykorzystując składnię `async` stwórzmy funkcję zwracającą nasz pierwszy `Promise` i wyświetlmy na ekranie `hello world!`.

```

```

```
// my func ... etc...

myFunc()
    .then(result => {
        console.log(result);
    });
```

4. Stwórzmy aplikację która będzie posiadała funkcję asynchroniczną dodawnia 2 liczb do siebie. Jeżeli wynik będzie liczbą parzystą powinniśmy wyrzucić błąd i poinformować użytkownika o tym fakcie.

```
// my func ... etc...

add(4, 5)
    .then( ... )
    .catch( ... );
```

5. Zmodyfikujmy nasze zadanie 4 tak aby zamiast `.then..catch` użyć `await`.
6. Wykorzystując wiedzę z poprzednich zajęć użyjmy zewnętrznej biblioteki `axios` i pobierzmy użytkownika dane wykorzystując składnię `async/await`.

> Endpoint do użytkownika: https://jsonplaceholder.typicode.com/users/2

7. Dodajmy do naszego zadania 6 obsługę błędów `try..catch`.
8. Wykorzystując składnię `async/await` zmodyfikujmy zadanie 7 tak aby pobrać kilku użytkowników w tej samej chwili wykorzystując `Promise.all()`. Wyświetlmy ich imiona w konsoli. (id użytkowników: 2,3,5,7,8,10).
9. Dodajmy do naszej aplikacji z zadania 7 pobieranie pogody dla naszego użytkownika (z odpowiedzi weźmy `main.temp` i wyświetlmy na ekranie). Zadanie analogiczne jak w poprzednim laboratorium z wykorzystaniem składni `async/await`.

> Endpoint do pogody: https://api.openweathermap.org/data/2.5/weather?appid=0ed761300a2725ca778c07831ae64d6e&lat={LAT}&lon={LNG}

10. Stwórzmy aplikację która pobierze informację o użytkowniku i statystykach jego postów i komentarzy.

- Z pobranego użytkownika wyświetlmy na ekranie nazwę użytkownika oraz email.
- Pobierzmy wszystkie posty użytkownika i wyświetlmy ich ilość w konsoli.
- Dodatkowo sprawdźmy aktywność szukanego użytkownika w komentarzach i wyświetlmy łączną ilość komentarzy w konsoli.

> Endpoint do użytkownika: https://jsonplaceholder.typicode.com/users/2

> Endpoint do postów: https://jsonplaceholder.typicode.com/posts?userId=2

> Endpoint do komentarzy: https://jsonplaceholder.typicode.com/comments?postId=11