JAVASCRIPT ASYNC/AWAIT

1) async is a keyword that is used before a function to indicate that the function will return a promise.

```
async function example() {
    return "Hello World";
}
```

2) await is used before an asynchronous expression, such as a promise, to pause execution of the function until the promise is resolved.

```
async function example() {
    const result = await promise;
    console.log(result); // "Hello World"
}
```

3) await can only be used within an async function, if you try to use it outside of one, you will get a syntax error

```
function example() {
    const result = await promise; //SyntaxError
    console.log(result);
}
```

4) By using await you can make synchronous-like code out of asynchronous code.

```
async function example() {
    const data = await fetch('https://jsonplaceholder.typicode.com/todos/1');
    const jsonData = await data.json();
    console.log(jsonData);
}
```

4) If you have multiple async operations and you want to wait for all of them to complete you can use Promise.all()

```
async function example() {
```

JAVASCRIPT ASYNC/AWAIT

```
    const [data1, data2] = await Promise.all([
        fetch('https://jsonplaceholder.typicode.com/todos/1'),
        fetch('https://jsonplaceholder.typicode.com/todos/2'),
    ]);
    const jsonData1 = await data1.json();
    const jsonData2 = await data2.json();
    console.log(jsonData1, jsonData2);
}
```

6) await also can be used with try and catch block to handle errors in an async flow

```
async function example() {
    try {
        const data = await fetch('https://jsonplaceholder.typicode.com/todos/1');
        const jsonData = await data.json();
        console.log(jsonData);
    } catch (error) {
        console.error(error);
    }
}
```

LAB 1

Create a movie search application that allows users to search for movies by title and display a list of results. The application should retrieve the data from the OMDb API (http://www.omdbapi.com/) The application should be built using async and await functions.

Some of the required features are:

• Display a form for users to enter a movie title

• Use async and await functions to make a GET request to the OMDb API to retrieve the search results for the entered movie title

• Display the movie title, poster, year of release and IMDb rating of each result in a list on the page

• Handle errors that may occur during the API call (e.g. movie not found, invalid API key)

• Provide loading indication while waiting for the API response

JAVASCRIPT ASYNC/AWAIT

- Bonus: Allow users to filter the results by year of release

- Bonus: Store the previous successful search for the same title.

Some of the technical requirements:

- Use of async and await functions to handle the API call.

- Use of fetch or axios (or similar) to make the GET request.

- Use of event listeners and DOM manipulation to handle the form submission and update the display.

- Use of error handling mechanisms like try-catch to handle errors.

Note that the OMDb API requires an API key, which you can get by signing up to the site.

GETTING HELP

We always want to ensure that your questions are answered. There are a number of ways to get in touch.

1) We operate a lively Discord server. Join us at https://discord.gg/tgxX2fCrv5 and you can ask your question on Discord. Mark and our team of instructional assistants monitor this Discord and answer questions ASAP.

2) This certification program is offered on several platforms. Most platforms have a Q & A section where you can post questions. We monitor these and respond as quickly as we can.

3) You may use our question hotline email at questions@dollardesignschool.com.

Remember to always to include which section your question is from and send any code you're working on!

JAVASCRIPT ASYNC/AWAIT