2024

chapter_7_part_2_task

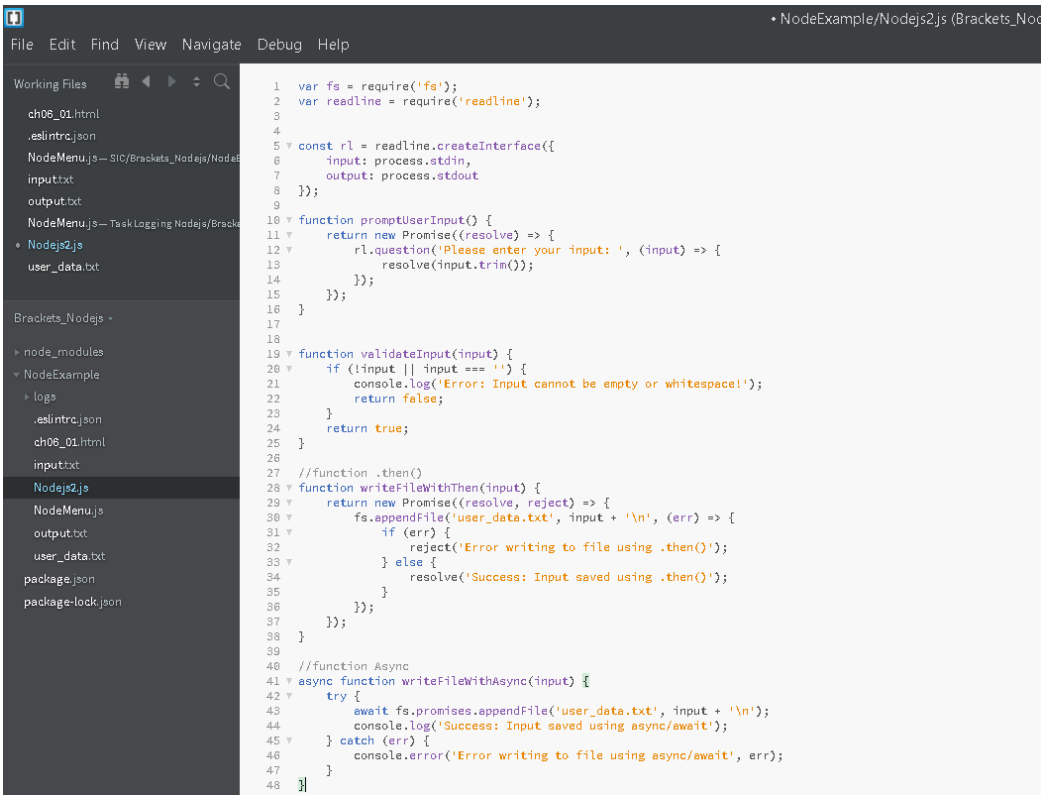
JOLIANA EMAD KAMAL NAGUIB 20P3292

SAMSUNG INNOVATION CAMPUS

TASK Node.js

Nodejs2.js

->Nodejs2.js script inside NodeExample folder ,Using the ‘readline’ module, this code will ask you to enter an input then the ‘fs.appendFile’ function will write that data into a file. It handles this in two ways which include using Promises with ‘.then()’ for chaining as well as ‘async/await ‘for a more synchronous style and it includes error handling to control invalid input or file writing failures.



Promises & Async Handling

➔ 1st Approach using. then()

➔ With then() the rest of the function will continue to execute, but JavaScript won't execute the then() callback until the promise settles (resolve, reject).

➔ Each call to then() creates another step in the Promise chain, and if there's an error at any point in the chain, the next catch() block will be triggered.

```
//function .then()
function writeFileWithThen(input) {
    return new Promise((resolve, reject) => {
        fs.appendFile('user/user_data.txt', input + '\n', (err) => {
            if (err) {
                reject('Error writing to file using .then()');
            } else {
                resolve('Success: Input saved using .then()');
            }
        });
    });
}
```

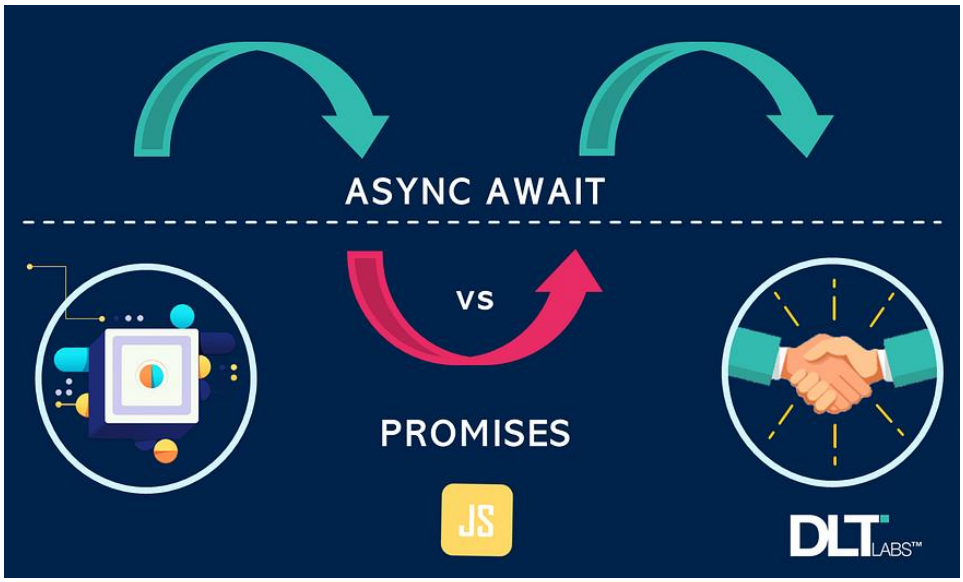
➔ 2nd Approach using async/await

➔ While With await, JavaScript will pause the function execution until the promise settles (resolve, reject). In short: asynchronous code behaves like synchronous.

```
//function Async
async function writeFileWithAsync(input) {
    try {
        await fs.promises.appendFile('user_data.txt', input + '\n');
        console.log('Success: Input saved using async/await');
    } catch (err) {
        console.error('Error writing to file using async/await', err);
    }
}
```

Which one is better?

From a performance point of view, await is just an internal version of. then() (doing basically the same thing). The reason to choose one over the other doesn't really have to do with performance, but has to do with desired coding style or coding convenience.



The difference between then() and async/await is that when a promise needs to get resolved in a function having then(), the function containing then() doesn't get suspended from the call stack. It remains in the call stack till the promise gets resolved.

Whereas, in the async/await, when the resolve () keyword is encountered, the async function is suspended from the call stack, and the thread remains inactive if the async function is the entry point of the event handler. Only when the promise resolves, the async function gets pushed again in the call stack.

To conclude: async/await is generally preferred in modern JavaScript because it’s easier to read and understand, especially when dealing with multiple asynchronous tasks. However, both methods are valid and accomplish the same task: handling asynchronous code with Promises

Node.js Terminal

- ➔ Use node Nodejs.js to run file
- ➔ 1st Choice using. then()
- ➔ 2nd Choice using async/await
- ➔ Input saved two times using async/await method and one time using .then() method
- ➔ Message Success indicating that the input was successfully written to the file.(Feedback for success)

```
Node.js command prompt - n

Your environment has been set up for using Node.js 22.8.0 (x64) and npm.

C:\Users\Lenovo>cd NodeExample
The system cannot find the path specified.

C:\Users\Lenovo>cd C:\Users\Lenovo\SUMMER .2024\SIC\Brackets_Nodejs\NodeExample

C:\Users\Lenovo\SUMMER .2024\SIC\Brackets_Nodejs\NodeExample>node Nodejs2.js
Please enter your input: Hello Juliana
Choose method for saving input (1 for .then, 2 for async/await): 2
Success: Input saved using async/await

C:\Users\Lenovo\SUMMER .2024\SIC\Brackets_Nodejs\NodeExample>node Nodejs2.js
Please enter your input: Hiiiii
Choose method for saving input (1 for .then, 2 for async/await): 1
Success: Input saved using .then()

C:\Users\Lenovo\SUMMER .2024\SIC\Brackets_Nodejs\NodeExample>node Nodejs2.js
Please enter your input: Hello AGAINNN
Choose method for saving input (1 for .then, 2 for async/await): 2
Success: Input saved using async/await
```

- ➔ Write invalid path to throw an error

```
//function .then()
function writeFileWithThen(input) {
    return new Promise((resolve, reject) => {
        fs.appendFile('user/user_data.txt', input + '\n', (err) => {
            if (err) {
                reject('Error writing to file using .then()');
            } else {
                resolve('Success: Input saved using .then()');
            }
        });
    });
}
```

➔ Message Failure indicating that error occurred during writing the input to the file. (Feedback for failure)

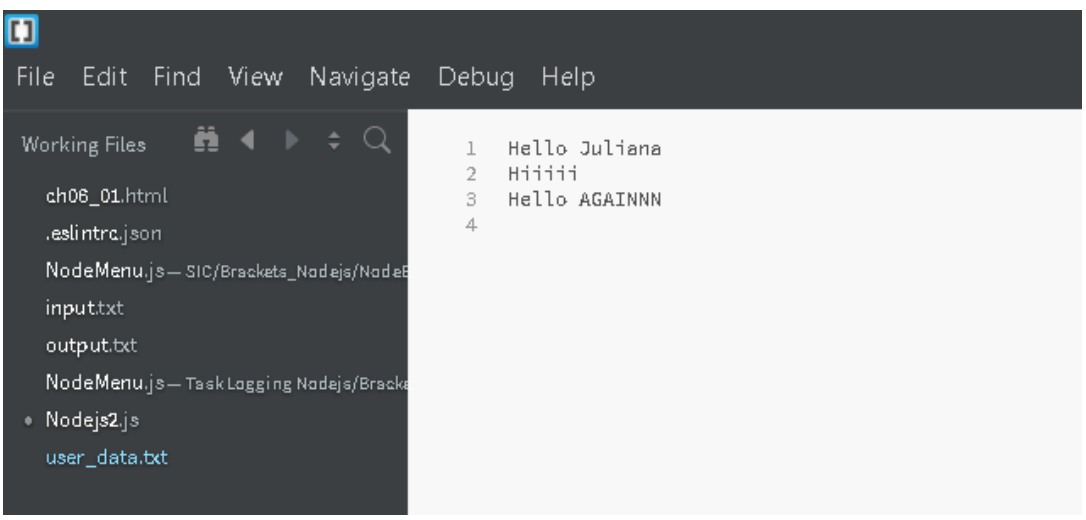
```
C:\Users\Lenovo\SUMMER .2024\SIC\Brackets_Nodejs\NodeExample>node Nodejs2.js
Please enter your input: JULI
Choose method for saving input (1 for .then, 2 for async/await): 1
Error writing to file using .then()

C:\Users\Lenovo\SUMMER .2024\SIC\Brackets_Nodejs\NodeExample>
```

File Output

➔ Output file named user_data.txt, should append each new input to the file.

```
        fs.appendFile('user_data.txt', input + '\n', (err) => {
            if (err) {
                reject('Error writing to file using .then()');
            } else {
                resolve('Success: Input saved using .then()');
            }
        });
    });
```



Error Handling:

```
function validateInput(input) {
    if (!input || input === '') {
        console.log('Error: Input cannot be empty or whitespace!');
        return false;
    }
    return true;
}
```

- ➔ Prints Error message if input empty or whitespace
- ➔ Ask user to re-enter valid data.

```
C:\Users\Lenovo\SUMMER .2024\SIC\Brackets_Nodejs\NodeExample>node Nodejs2.js
Please enter your input:
Error: Input cannot be empty or whitespace!
Please enter your input:
Error: Input cannot be empty or whitespace!
Please enter your input:
```

Main Flow:

```
async function main() {
    let validInput = false;
    let input = '';

    while (!validInput) {
        input = await promptUserInput();
        if (validateInput(input)) {
            validInput = true;
        }
    }
    rl.question('Choose method for saving input (1 for .then, 2 for async/await): ', async (choice) => {
        if (choice === '1') {
            writeFileWithThen(input)
                .then((message) => console.log(message))
                .catch((error) => console.error(error));
        } else if (choice === '2') {
            await writeFileWithAsync(input);
        } else {
            console.log('Invalid choice. Please restart and choose 1 or 2.');
        }
        rl.close();
    });
}
main();
```

References

https://stackify.com/node-js-error-handling/

https://beyondthecloud.dev/blog/then-vs-async-await-in-lwc