Async & Await (ES8)

* It is an extention to promises itself.
* Async also returns a promise
* So it's basically syntatic sugar, does the same thing as promises but code looks much more readable.

// with promises [following do the same thing]

```
performTask(task1)
    .then(() => PerformTask(task2)
    .then(() => Perform Task(task3)
    .then(() => PerformTask(task4)
```

// async await

do some async task inside this function

```
async function playGame(){
    const task1 = await performTask(task1)
    await performTask(task2);
    await performTask(task3);
    await performTask(task4);
}
```

await keyword basically indicates we're waiting for something to give us a response.
Instead of chaining with .then() we're awaiting for every task to perform before next

Practical example

The fetch() methods that let's you make an API call, is actually a promise, so we can do the following

```
fetch('https://jsonoutput/users')
    .then(response => response.json())
    .then(console.log(response))
```

Because fetch is a promise we're able to do .then() on it

* Note :- response.json() , this json() to convert to json format is also a promise hence again a .then() after it.

→ Let's convert this to async await

```
asyn function fetchStudents(){
    const response = await fetch('https://.....')
    const data = await response.json()
    console.log(data);
}
```

Error handling in async await.

```
async function fetchStudents() {
    try {
        await fetch(url)
        :
        }
    catch (error) {
            console.log(error)
        }
}
```

→ Error handling with async await can be achieved using try catch block.
Put your fetch call (await) inside try and it any error occurs within try then it will be caught in catch block