In JavaScript, the keywords "async" and "await" are used to work with asynchronous code and make it appear more synchronous and easier to read and write. The async/await" pattern was introduced in ECMAScript 2017 (ES8) and has since" .become a standard way of dealing with asynchronous operations in JavaScript

The "async" keyword is used to define an asynchronous function. An async function always returns a promise, which can be resolved with the function's return value or rejected with an error. Within an async function, you can use the "await" keyword to .pause the execution of the function until a promise is resolved or rejected

Here's an example that demonstrates the usage of "async" and "await" in JavaScript

```
function delay(ms){

;return new Promise(resolve => setTimeout(resolve, ms))

}

async function asyncFunction (){

;console.log("Start")


try  {

await delay(2000); // Pause execution for 2 seconds

console.log("After 2 seconds");


const result = await someAsyncOperation(); // Wait for an asynchronous operation to complete

console.log("Async operation result:", result   );

catch (error) {

;console.error("Error:", error)

  }


console.log("End")  ;

}
```

asyncFunction ();

,In this example, the function `asyncFunction()` is declared with the "async" keyword "indicating that it is an asynchronous function. Within the function, the "await keyword is used to pause the execution at specific points until the promises are .resolved or rejected

The `delay()` function returns a promise that resolves after a given number of milliseconds. By using "await" with `delay(2000)`, the execution of the function is .paused for 2 seconds

The `someAsyncOperation()` function represents an asynchronous operation that returns a promise. By using "await" with this function, the execution is paused until the promise is resolved. The result of the asynchronous operation is then assigned .to the `result` variable

Inside the async function, you can also use try-catch blocks to handle potential errors thrown by the awaited promises. If an error occurs during the execution of the .awaited promise, the catch block will be executed

When the async function is called (`asyncFunction()`), it returns a promise that can be further chained or awaited

:The output of the above code will be something like

Start

After 2 seconds

<Async operation result: <result

End