

async/await

JS

ASYNC AWAITS



codewithgauri



SYNC: SYNCHRONOUS

Imagine there is a new ATM machine setup in a marketplace.

Initially, there were 10-15 people to use it.



The machine can serve only one person at a time which initially feels like nothing to its customer but later as the number of users will increase the problem will increase simultaneously



The is synchronous execution. It is similar to a compiler executing the code one at a time.

PROBLEM:

During the initial days, everything will work fine as the number of customers will be low.



But as the number of customers **will increase the problem will be visible** .



ASYNC: ASYNCHRONOUS

One fine day another bank urged to the same marketplace and **set up another ATM** near the old one.



This is similar to the **asynchronous process**. The customer will be diverted to the 2nd ATM machine, reducing the line on ATM 1.



The above-mentioned synchronous process is also known as the **blocking process**.

Which unnecessarily blocks further processes
That's why running file uploading processes should be asynchronous.



ASYNC FUNCTION

An async function is a function declared with the `async` keyword, and the `await` keyword is permitted within it. The `async` and `await` keywords enable asynchronous, promise-based behavior to be written in a cleaner style, avoiding the need to explicitly configure promise chains.



```
function resolveAfter2Seconds() {  
  return new Promise(resolve => {  
    setTimeout(() => {  
      resolve('resolved');  
    }, 2000);  
  });  
}  
  
async function asyncCall() {  
  console.log('calling');  
  const result = await resolveAfter2Seconds();  
  console.log(result);  
  // expected output: "resolved"  
}  
  
asyncCall();
```



SYNTAX:

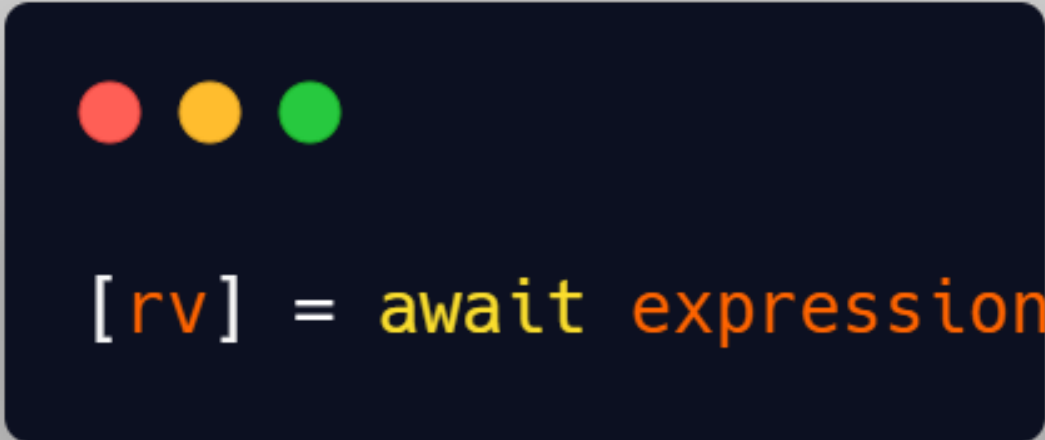
```
async function name(param0) {  
    statements  
}  
async function name(param0, param1) {  
    statements  
}  
async function name(param0, param1, /* ... */ paramN) {  
    statements  
}
```



Awaits:

The `await` operator is used to wait for a **Promise**. It can only be used inside an `async` function within regular JavaScript code; however, it can be used on its own with **JavaScript modules**.

Syntax:



```
[rv] = await expression
```

expression

A Promise or any value to wait for.

rv

Returns the fulfilled value of the promise, or the value itself if it's not a Promise.



And for amazing stuff you can follow me



Gaurav Pandey

LinkedIn :Gaurav Pandey

Twitter : @gauravcode

References:

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/await>

Rajan Verma on Instagram: "ASYNC calls are the core of nodejs environment. The node processes are by default asynchronous and that's why faster. But there are many..."