**Async/await**

Selain menggunakan *callback* dan *promise*, kita juga bisa menggunakan ***async/await*** untuk menggunakan *asynchronous* pada JavaScript. ***Async/await*** baru ada ketika *update* [ES8](https://en.wikipedia.org/wiki/ECMAScript#8th_Edition_%E2%80%93_ECMAScript_2017) JavaScript dan dibangun menggunakan *promise*. Jadi sebenarnya *async/await* dan *promise* itu sama saja, namun hanya berbeda dari *syntax* dan cara penggunaannya.

Ada 2 kata kunci yang memiliki pengertian sebagai berikut:

* async, mengubah *function synchronous* menjadi *asynchronous*.
* await, menunda eksekusi hingga proses *asynchronous* selesai.

Sebuah *async function* bisa tidak berisi *await* sama sekali atau lebih dari satu *await*. *Keyword await* hanya bisa digunakan didalam *async function*, jika digunakan di luar *async function* maka akan terjadi *error*.

**Async**

Berikut ini contoh penggunaan dari async :

// async menggunakan keyword function

async function tesAsyncAwait() {

return "Fulfilled";

}

console.log(tesAsyncAwait());

// async menggunakan arrow function

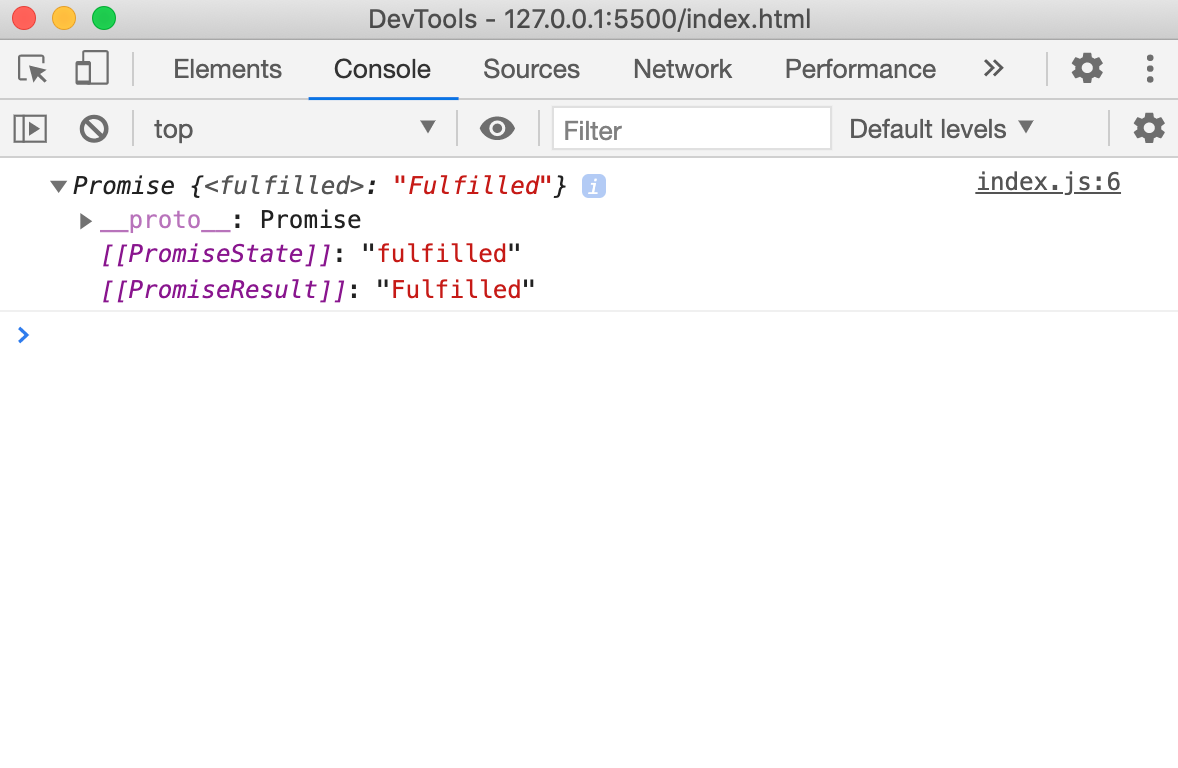
const tesAsyncAwait = async () => {

return "Fulfilled";

};

console.log(tesAsyncAwait());

Jika salah satu kode di atas dijalankan, maka akan terlihat tampilan seperti berikut ini:



**Await**

await hanya bisa digunakan dalam *async function* dan await adalah *keyword* dalam async yang digunakan untuk menunda hingga proses *asynchronous* selesai.

Berikut ini contoh penggunaan dari *async/await* :

async function tesAsyncAwait() {

await 'Fulfilled';

}

Kita juga bisa memberikan *error handling* pada *async/await*. Contoh lengkap penggunaan *async/await*:

// Definisikan dahulu promise yang ingin digunakan

let condition = true;

let tesAsyncAwait = async (condition) => {

if (condition) {

return "Condition is fulfilled!";

} else {

throw "Condition is rejected!";

}

};

// Membuat fungsi run menjadi asynchronous menggunakan async/await

const run = async (condition) => {

try {

const message = await tesAsyncAwait(condition);

console.log(message); // Output: Condition is fulfilled!

console.log("After condition is fulfilled"); // Output: After condition is fulfilled

} catch (error) {

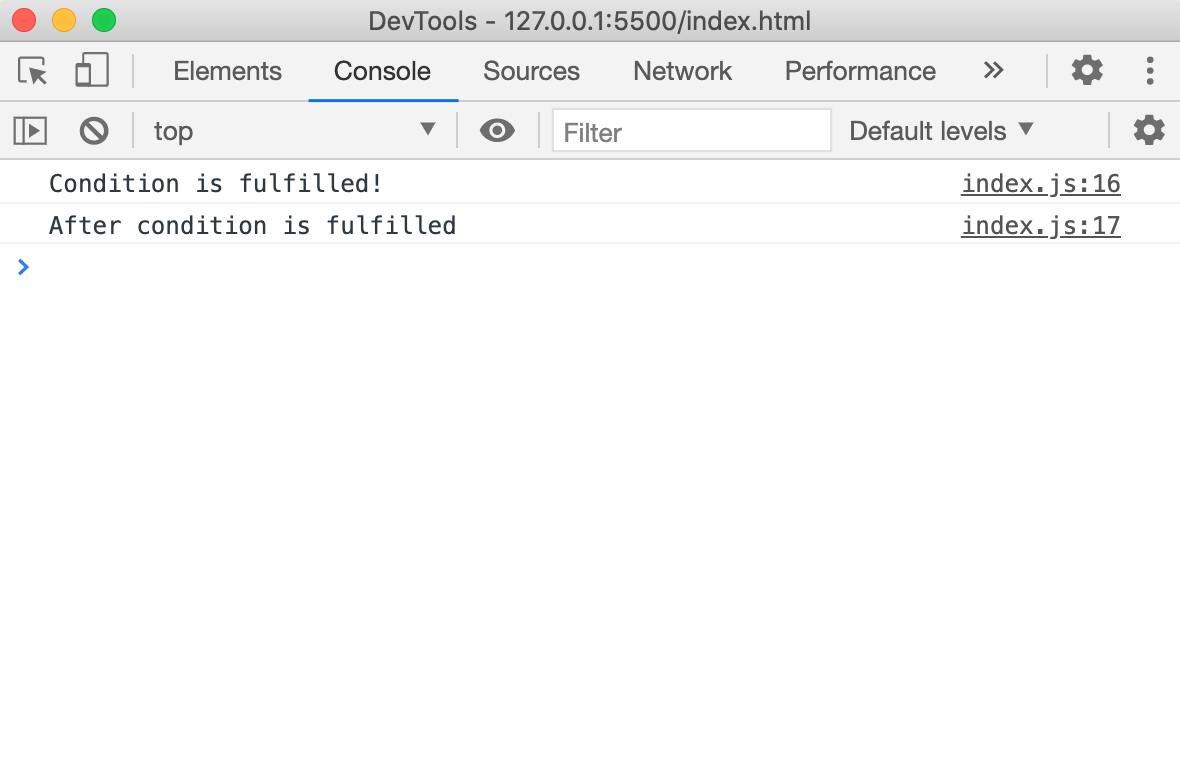
console.log(error);

}

};

run(true);

Dari kode di atas, kita dapat melihat bahwa run adalah sebuah fungsi *async* dan *await* dipanggil bersamaan dengan fungsi tesAsyncAwait(condition). await pada fungsi ini artinya, console.log pada message dan After condition is fulfilled tidak akan dijalankan (ditunda) hingga proses tesAsyncAwait(condition) selesai dijalankan.



Berikut ini contoh perbandingan jika program sebelumnya dibuat menggunakan promise :

let condition = true;

let tesPromise = new Promise((resolve, reject) => {

if (condition) {

resolve("Condition is fulfilled!");

} else {

reject(new Error("Condition is rejected!"));

}

});

tesPromise

.then(result =>{

console.log(result); // Condition is fulfilled!

console.log("After condition is fulfilled"); // After condition is fulfilled

})

.catch(error =>{

console.log(error);

})

Berikut hasil console.log() jika kode di atas dijalankan:

