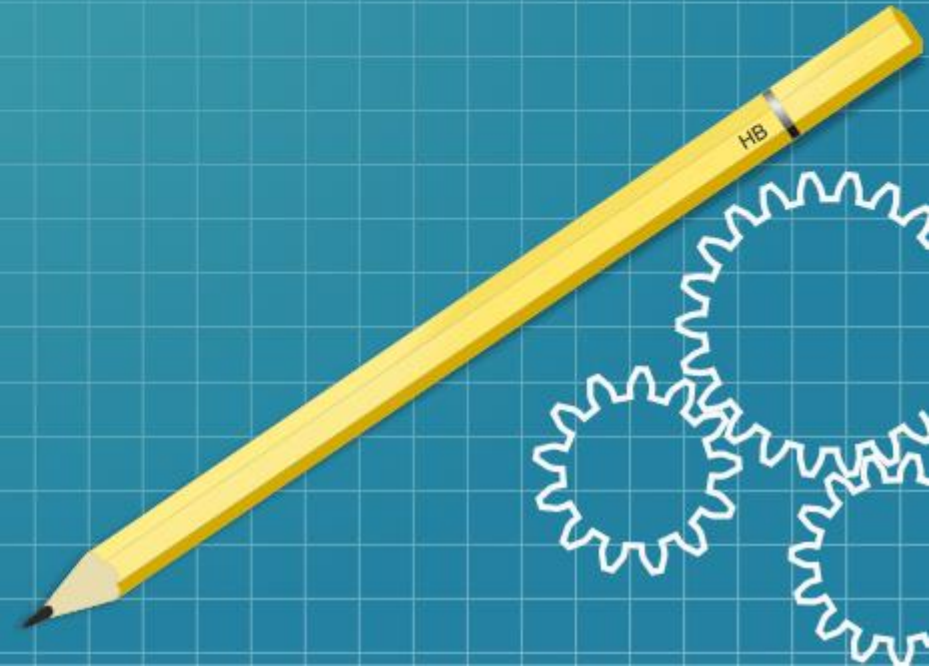




Asynchronous





Asynchronous adalah hasil eksekusi atau output tidak selalu berdasarkan urutan kode, tetapi berdasarkan waktu proses. Eksekusi dengan asynchronous tidak akan membloking atau menunggu suatu perintah sampai selesai. Daripada menunggu, asynchronous akan mengeksekusi perintah selanjutnya.

cara kerja **asynchronous** adalah berdasarkan waktu proses. Jika ada salah satu eksekusi membutuhkan proses yang agak lama, maka sembari menunggu proses tersebut javascript mengeksekusi perintah selanjutnya.



Async/await adalah fitur yang hadir sejak ES2017. Fitur ini mempermudah kita dalam menangani proses asynchronous. Async/Await merupakan sebuah syntax khusus yang digunakan untuk menangani Promise agar penulisan code lebih efisien dan rapih.

```
const getAllUser = async ()=> {  
  const data = await getUser()  
  console.log(data)  
}
```

penggunaan Async Await




- Async/Await adalah salah satu cara untuk mengatasi masalah asynchronous pada Javascript selain menggunakan callback dan promise.
- Pada implementasi Async/Await, kita menggunakan kata kunci async sebelum fungsi. Await sendiri hanya bisa digunakan pada fungsi yang menggunakan async.
- Untuk menggunakan Async/Await, kembalian dari suatu fungsi harus merupakan suatu Promise. Async/Await tidak dapat berdiri tanpa adanya Promise.
- Tidak seperti Promise, dengan Async/Await maka suatu baris kode dapat tersusun rapi mirip dengan kode


Error Handling Async/Await



Untuk handle error Async/Await kita dapat menggunakan try catch di dalam function yang kita buat, sehingga jika terjadi error kita dapat menangkap errornya dalam block catch, berikut contohnya.




```
const getAllUser = async ()=> {  
  try {  
    const result = await getUser()  
    console.log(result)  
  } catch (error) {  
    console.log(error)  
  }  
}
```



```
function tanyaKabar(name) {  
    console.log('Apa kabar,', name);  
}
```

```
function katakanHallo(name) {  
    setImmediate(function () {  
        console.log('Hallo,', name);  
    });  
}
```

Pemanfaatan async sangat berguna pada saat melakukan operasi-operasi yang tidak harus menunggu proses yang lainnya. Misalnya, memanipulasi DOM ketika melakukan proses ajax.

Agar bisa memahami, kita perlu mengetahui cara handle kode async tersebut dengan cara callback ataupun promises.



Synchronous



setiap perintah harus menunggu perintah sebelumnya selesai. Proses seperti ini disebut 'blocking'.

```
function tanyaKabar(name) {  
    console.log('Apa kabar,', name);  
}
```

```
function katakanHalo(name) {  
    console.log('Halo,', name);  
}
```

Selesai, Jum'at - 09/09/2022

