



DISKUSIKAN MATERI

LAPORKAN MATERI

Daftar Modul

Masukkan kata kunci

Persetujuan Hak Cipta ✓

Modul 1: Introduction Course (Gratis) ✓

Prasyarat Kelas (Gratis) ✓

Apa yang Akan Kita Pelajari? (Gratis) ✓

Tools Requirement ✓

Modul 2: ECMAScript 6 (Gratis) ✓

Running Snippet Code (Gratis) ✓

Persiapan Project Latihan (Gratis) ✓

This Keyword

Perbedaan karakteristik dari arrow function dan regular function selanjutnya ada pada penggunaan keyword **this**. Penjelasan dari **this** sendiri menyusul di materi class. Namun kita akan bahas sedikit mengenai ini untuk menggambarkan perbedaan ketika **this** digunakan oleh arrow function dan regular function.

Jika sebuah regular function dipanggil dengan menggunakan keyword **new**. Maka nilainya akan menjadi objek, contohnya:

```
1. function People(name, age, hobby) {
2.   this.name = name;
3.   this.age = age;
4.   this.hobby = hobby;
5. }
6.
7. const programmer = new People("John", 18, ["Coding", "Read book", "Ping-pong"]);
8.
9. console.log(programmer.name);
10. console.log(programmer.age);
11. console.log(programmer.hobby);
12.
13. /* output:
14. John
15. 18
16. [ 'Coding', 'Read book', 'Ping-pong' ]
17. */
```

Objek yang dibuat menggunakan function dengan keyword **new**, sama halnya seperti kita membuat objek seperti menggunakan objek literals **{}**.

```
1. const programmer = {
2.   name: "John",
3.   age: 18,
4.   hobby: ["Coding", "Read book", "Ping-Pong"]
5. }
6.
7. console.log(programmer.name);
8. console.log(programmer.age);
9. console.log(programmer.hobby);
10.
11. /* output:
12. John
13. 18
14. [ 'Coding', 'Read book', 'Ping-pong' ]
15. */
```

Pada objek, **this** keyword mengembalikan nilai objeknya sendiri. **this** dapat digunakan untuk mengelola properti pada objeknya. Namun jika fungsi dipanggil tanpa menggunakan keyword **new**, **this** akan memiliki nilai global object (Window jika di browser).

Sedangkan fungsi yang dibuat dengan menggunakan gaya arrow tidak akan pernah memiliki nilai **this**, yang artinya kita tidak pernah bisa membuat objek menggunakan arrow function. Jika kita menggunakan **this** pada arrow function maka nilai **this** tersebut merupakan nilai objek di mana arrow function itu berada.

Perhatikan kedua contoh kode berikut:

Regular Function Arrow Function

```
1. function People(name, age, hobby) {
2.   this.name = name;
3.   this.age = age;
4.   this.hobby = hobby;
5. }
6.
7.
8. // menambahkan introMyself ke People
9. People.prototype.introMyself = function () {
10.  // this -> People
11.  setTimeout(function() {
12.    // this -> ??
13.    console.log(`Hello! Nama saya ${this.name}, umur saya ${this.age}.`);
14.    console.log(`Hobby saya adalah ${this.hobby}`);
15.  }, 300);
16. }
17.
18.
19. const programmer = new People("John", 18, ["Coding", "Read book", "Ping-pong"]);
20. programmer.introMyself();
21.
22.
23. /* output:
```

Fungsi yang dituliskan di dalam `setTimeout()` dipanggil tanpa `new`. Itu berarti nilai dari `this` jika digunakan di dalam fungsi tersebut adalah *global object*. Itulah mengapa output akan menghasilkan nilai `undefined` ketika properti `name`, `age`, dan `hobby` dipanggil.

Berbeda ketika kita menuliskan arrow function di dalam `setTimeout()`, nilai `this` memiliki nilai objek sesuai dengan konteksnya (`People`). Arrow function akan sangat berguna untuk kasus seperti ini.

[← KEMBALI KE MATERI SEBELUMNYA](#)[LANJUTKAN KE MATERI BERIKUTNYA →](#)

image
click bola
dengan mouse

image
click bola
dengan mouse

PERUSAHAAN

[Tentang Kami](#)[Blog](#)[Berita Terbaru](#)

PROGRAM

[Academy](#)[Challenge](#)[Event](#)[Job](#)[Rewards](#)

SUPPORT

[Bantuan](#)[FAQ](#)[Hubungi Kami](#)