



[Beranda](#) / [Academy](#) / [Belajar Dasar Pemrograman Web](#) / Function

! Pembaruan! Modul ini dibuat pada tanggal 6 December 2019. Pembaruan terakhir adalah: **Konversi potongan kode menjadi kode interaktif..**

[Lihat riwayat »](#)

Function

Tanpa kita sadari sebenarnya kita sudah menggunakan sebuah fungsi pada contoh kode yang ada sebelumnya. `console.log()` (lebih tepatnya pada `log()`) merupakan sebuah *function* yang berfungsi untuk menampilkan data pada console browser. Tapi sebenarnya apa itu *function*? Bagaimana ia bisa bekerja?

Function atau fungsi merupakan potongan kecil kode yang tidak akan dieksekusi sebelum dipanggil. Contoh lain fungsi JavaScript yang sudah ada pada browser adalah `alert()`. Ia bertujuan untuk menampilkan pesan dalam bentuk *pop up dialog*. Fungsi tersebut sudah ada pada browser dan akan muncul hanya ketika kita menggunakan/memanggilnya dengan `alert()`.

Atau dalam arti lain, kita dapat berfikir bahwa *function* merupakan sebuah variabel yang berisi *block* logika, dan *block* logika tersebut akan dieksekusi ketika variabelnya dipanggil.

Semua fungsi memiliki struktur yang sama. Nama fungsi selalu diikuti dengan tanda kurung (*parentheses*) tanpa spasi, lalu terdapat sepasang kurung kurawal yang berisi logika dari fungsi tersebut.

Terkadang di dalam tanda kurung kita membutuhkan sebuah informasi tambahan yang disebut dengan arguments. Argument merupakan data yang digunakan pada fungsi yang dapat mempengaruhi perilaku dari fungsinya tersebut. Contoh, fungsi `alert()` dapat menerima argument string yang digunakan untuk menampilkan teks pada dialog.

Berikut merupakan ilustrasi dari struktur fungsi.

	Nama fungsi	Argument		Tidak semua fungsi membutuhkan argument
Kode yang akan dieksekusi		┌		
	<code>multiply(a, b) {</code>	<code>return a * b;</code>	<code>multiply() {</code>	<code>return 5 * 5;</code>
	<code>}</code>	<code>}</code>	<code>}</code>	

Terdapat dua tipe fungsi pada JavaScript, yakni *native function* dan *custom function*.

Native function merupakan fungsi yang sudah terdapat pada JavaScript atau Browser sehingga kita tidak perlu membuat hanya tinggal menggunakan saja. Contohnya `alert()`, `confirm()`, `Date()`, `parseInt()` dll. Sebenarnya terdapat ratusan *native function* yang tersedia.

Custom function merupakan fungsi yang kita buat sendiri, tentu *custom function* dibuat menyesuaikan kebutuhan. Untuk membuat sebuah *custom function*, kita perlu menuliskan *keyword function* dilanjutkan dengan menuliskan seluruh logika fungsinya.





```
2.     console.log( "Good Morning: " );
3. }
```

Kemudian kita dapat memanggil fungsinya tersebut dengan menggunakan `greeting()`.

main.js +

```
1 function greeting() {
2     console.log("Good Morning!")
3 }
4
5 greeting();
6
7 /* output
8 Good Morning!
9 */
```

INPUT

RESET

JALANKAN

Output:

Tetapi jika sebuah fungsi hanya menjalankan baris kode secara sama dirasa kurang fungsional bukan? Kita dapat membuat fungsi tersebut untuk menerima argumen dan memanfaatkan argumen untuk mengubah perilaku dari fungsinya.

Untuk menambahkan *argument* pada fungsi, tambahkan variabel di dalam tanda kurung fungsi namun variabel tersebut tidak memerlukan keyword `var`, `let`, ataupun `const`. Kita juga bisa menambahkan lebih dari satu argumen dengan memberikan tanda koma antar variabel argumennya. Contohnya fungsi `greeting` akan kita ubah dengan menambahkan argument, sehingga akan nampak seperti ini:

```
1. function greeting(name, language) {
2.     if(language === "English") {
3.         console.log("Good Morning " + name + "!");
4.     } else if (language === "French") {
5.         console.log("Bonjour " + name + "!");
6.     } else {
7.         console.log("Selamat Pagi " + name + "!");
8.     }
9. }
```

Sehingga dalam memanggilnya pun kita perlu mengirimkan dua buah nilai pada fungsinya seperti berikut:

main.js +

```
1 function greeting(name, language) {
2     if(language === "English") {
3         console.log("Good Morning " + name + "!");
4     } else if (language === "French") {
5         console.log("Bonjour " + name + "!");
6     } else {
```



DIBANTU



```
10
11 greeting("Harry", "French");
12
13 /* output
```

☐ INPUT

RESET

JALANKAN

Output:

Satu hal lagi, *function* dapat mengembalikan sebuah nilai. Hal ini benar-benar sangat berguna dan membuat kita lebih mudah. Dengan nilai kembalian, kita dapat membuat *function* yang berfungsi untuk melakukan perhitungan matematika dan hasilnya dapat langsung kita masukkan ke dalam sebuah variabel. Contohnya seperti ini:

main.js +

```
1 function multiply(a, b) {
2   return a * b;
3 }
4
5 let result = multiply(10, 2)
6 console.log(result)
7
8 /* output
9 20
10 */
```

☐ INPUT

RESET

JALANKAN

Output:

Untuk membuat nilai kembalian dari fungsi gunakan *keyword* **return** diikuti dengan nilai yang akan dikembalikan. Nilai kembalian tidak hanya number, bisa saja berupa string, boolean, objek ataupun array. Seperti inilah fungsi **greeting()** jika kita ubah dengan menetapkan dengan nilai kembalian string:

main.js +

```
1 function greeting(name, language) {
2   if(language === "English") {
3     return "Good Morning " + name + "!";
4   } else if (language === "French") {
5     return "Bonjour " + name + "!";
6   } else {
7     return "Selamat Pagi " + name + "!";
8   }
9 }
10
11 let greetingMessage = greeting("Harry", "French");
12 console.log(greetingMessage);
13
```

☐ INPUT

RESET

JALANKAN





Yang perlu kita perhatikan lagi, ketika statement return tereksekusi, maka fungsi akan langsung terhenti dan mengembalikan nilai.

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

Dicoding Space
Jl. Batik Kumeli No.50, Sukaluyu,
Kec. Cibeunying Kaler, Kota Bandung
Jawa Barat 40123



Penghargaan



Decode Ideas Discover Potential

[➤ Tentang Kami](#)[Blog](#)[Reward](#)[Showcase](#)[Hubungi Kami](#)[FAQ](#)

© Copyright Dicoding Indonesia 2021

[Terms](#) • [Privacy](#)

