

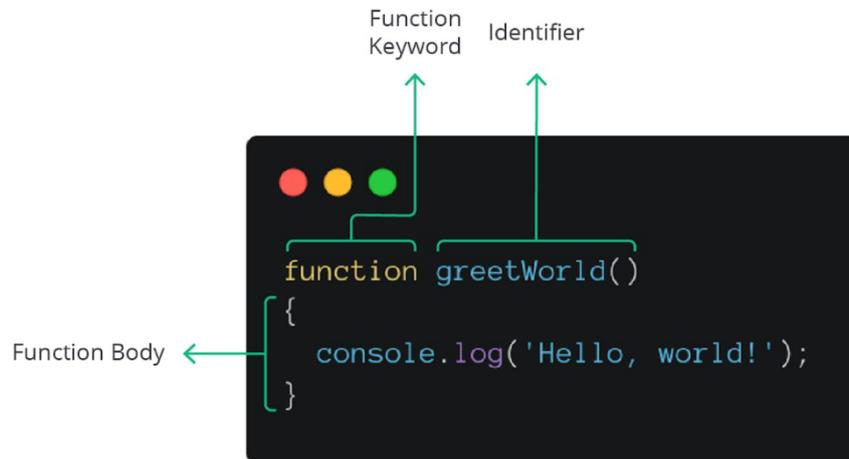


JavaScript

A Game Changer of
Complex Applications

Function

Fungsi adalah blok kode atau prosedur yang dirancang untuk melakukan tugas tertentu. Fungsi menerima input (yang disebut parameter atau argumen), memprosesnya, dan mungkin menghasilkan output.



Keyword “function” harus selalu disertakan ketika mendeklarasikan function. Berikutnya, keyword tersebut diikuti dengan identifier. Ini mirip seperti variabel, kita dapat memberi nama sesuai dengan kebutuhan. Setelah itu, kita tuliskan *parentheses* untuk mendefinisikan parameter-parameter yang diperlukan dan function body. Apa itu parameter? Sebelum itu, kami jelaskan function body dahulu.

Function body berisi sekumpulan statement atau perintah yang akan dieksekusi oleh interpreter. Ia dibungkus dengan tanda kurung kurawal buka-tutup (*curly braces*). Di sinilah kita dapat mengelompokkan kode-kode untuk melakukan penyederhanaan dan dapat dipanggil kapan pun developer butuhkan. Dengan kata lain, kelompok kode ini adalah tugas yang perlu dieksekusi dan diselesaikan oleh function ketika dipanggil.

Ada kalanya tugas function memerlukan input data agar dapat diproses. Ini mirip seperti function `console.log` ketika menampilkan data ke console karena ada data input yang diberikan. Nah, input data itulah yang disebut dengan parameter.

The screenshot shows a code editor with a file named 'function1.js'. The code is: `function salam() { console.log('Haloo, selamat pagi!'); } salam(); //Pemanggilan fungsi`. The bottom panel shows the 'DEBUG CONSOLE' with the command `/usr/local/bin/node ./function1.js` and the output `Haloo, selamat pagi!`.

Parameter & Argumen

Berikut adalah parameter dan argumen untuk function.

```
function2.js ×
function2.js > ...
Codeium: Refactor | Explain | Generate JSDoc | X
1 function konversiCelciusKeFahrenheit(temperatur) {
2   const fahrenheit = 9 / 5 * temperatur + 32;
3
4   console.log('Hasil Konversi :', fahrenheit);
5 }
6
7 konversiCelciusKeFahrenheit(100);

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL ... Filter (e.g. text, !exclude, \escape)
/usr/local/bin/node ./function2.js
Hasil Konversi : 212
```

Argumen dapat bernilai undefined (tidak beri nilai apa pun) dalam *parentheses* saat function dipanggil.

```
function3.js ×
function3.js > ...
Codeium: Refactor | Explain | Generate JSDoc | X
1 function konversiCelciusKeFahrenheit(temperatur) {
2   const fahrenheit = 9 / 5 * temperatur + 32;
3
4   console.log('Hasil Konversi :', fahrenheit);
5 }
6
7 konversiCelciusKeFahrenheit();

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL ... Filter (e.g. text, !exclude, \escape)
/usr/local/bin/node ./function3.js
Hasil Konversi : NaN
```

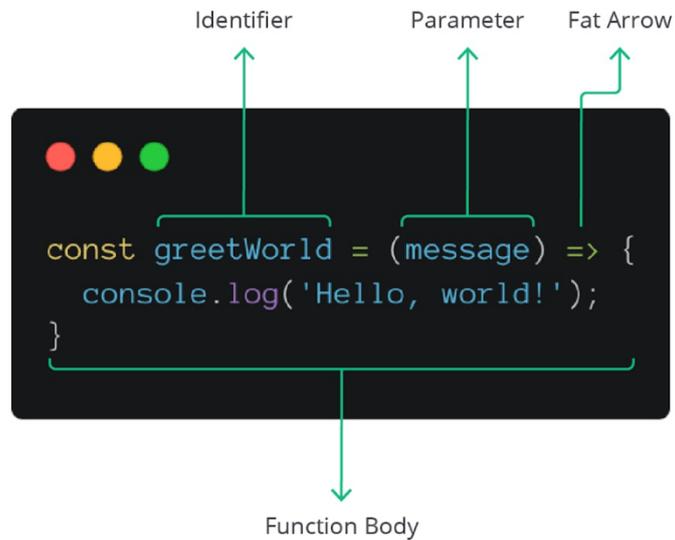
Apabila terdapat parameter berlebih maka tidak akan error, hanya sebatas parameter tersebut tidak digunakan.

```
function4.js ×
function4.js > ...
Codeium: Refactor | Explain | Generate JSDoc | X
1 function konversiCelciusKeFahrenheit(temperatur, suhu) { // Parameter suhu tidak dipakai
2   const fahrenheit = 9 / 5 * temperatur + 32;
3
4   console.log('Hasil Konversi :', fahrenheit);
5 }
6
7 konversiCelciusKeFahrenheit(200);

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL ... Filter (e.g. text, !exclude, \escape)
/usr/local/bin/node ./function4.js
Hasil Konversi : 392
```

Arrow Function

Ada alternatif sintaksis lain dalam JavaScript yang bisa mengubah cara kita membuat function. Sintaksis tersebut bernama arrow function.



Contoh :

```
function5.js x
function5.js > ...
Codeium: Refactor | Explain | Generate JSDoc | X
1 const konversiCelciusKeFahrenheit = (temperatur) => {
2   const fahrenheit = 9 / 5 * temperatur + 32;
3
4   console.log('Hasil Konversi :', fahrenheit);
5 }
6
7 konversiCelciusKeFahrenheit(30);

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL ... Filter (e.g. text, lexclude, \escape)
/usr/local/bin/node ./function5.js
Hasil Konversi : 86
```