Contents

[**Pengenalan Struktur Data** 1](#_Toc107415309)

**Pengenalan Struktur Data**

Sebelumnya kita sempat mempelajari tentang variabel dan tipe data yang ada di JavaScript. Tipe data yang kita bahas sebelumnya termasuk dalam kategori tipe data primitif yang hanya bisa menyimpan satu nilai.

Pada modul ini kita akan membahas tentang tipe data JavaScript yang dapat menyimpan struktur data yang lebih kompleks. Beberapa jenis struktur data yang akan kita pelajari, antara lain:

* Object
* Array
* Map
* Set

## Object

Kali ini kita akan berkenalan dengan tipe data object. Sebuah tipe data yang sangat berguna dalam pengembangan aplikasi dengan JavaScript. Object mampu menyimpan nilai dari beragam tipe data dan membentuk data yang lebih kompleks.

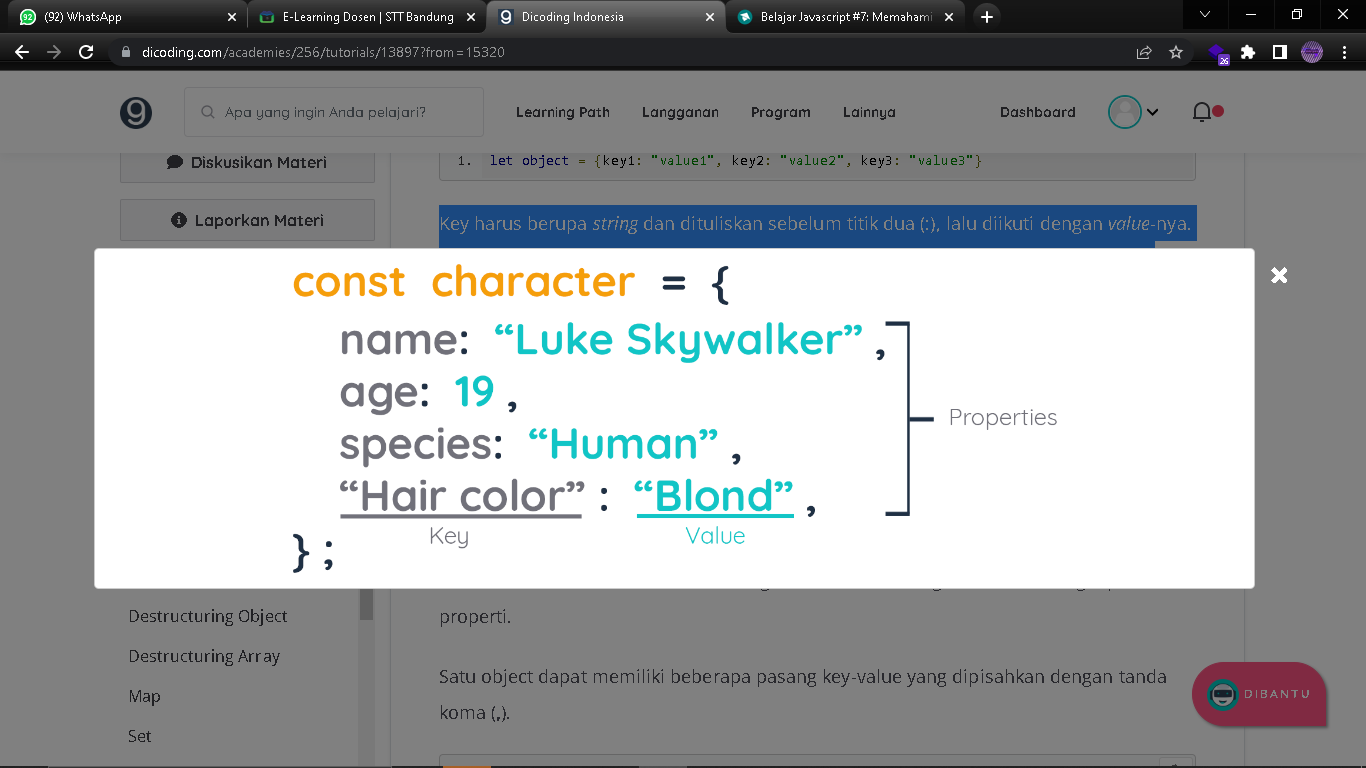
Untuk menetapkan objek pada sebuah variabel kita gunakan tanda kurung kurawal {}.

1. const user = {};

Object berisi pasangan key dan value yang juga dikenal dengan property. Key berperan mirip seperti nama variabel yang menyimpan sebuah nilai. Sementara, value berisi nilai dengan tipe data apa pun termasuk objek lain. Key dan value di dalam object dituliskan seperti berikut:

1. let object = {key1: "value1", key2: "value2", key3: "value3"}

Key harus berupa string dan dituliskan sebelum titik dua (:), lalu diikuti dengan value-nya. Meskipun key merupakan string, kita tidak perlu menuliskan tanda petik kecuali ada karakter khusus seperti spasi.



Tanda koma di akhir properti bersifat opsional. Namun, jika tanda koma tersebut ditulis akan lebih memudahkan ketika kita ingin memindah, mengubah, atau menghapus properti.

Satu object dapat memiliki beberapa pasang key-value yang dipisahkan dengan tanda koma (**,**).

1. const user = {firstName: "Luke", lastName: "Skywalker", age: 19, isJedi: true};

Dalam menuliskan objek, baris baru tidaklah penting dan tidak akan berpengaruh apa pun. Sehingga lebih baik setiap kita menetapkan key-value buatlah baris baru untuk memisahkan antar nilainya. Hal ini akan memudahkan kita dalam membaca dan memahami struktur data dari sebuah object.

1. const user = {
2. firstName: "Luke",
3. lastName: "Skywalker",
4. age: 19,
5. isJedi: true,
6. };

Kemudian untuk mengakses nilai dari properti object, kita dapat memanggil nama object lalu tanda titik dan diikuti nama propertinya. Contoh:

const user = {

firstName: "Luke",

lastName: "Skywalker",

age: 19,

isJedi: true,

};

console.log(`Halo, nama saya ${user.firstName} ${user.lastName}`);

console.log(`Umur saya ${user.age} tahun`);

Selain dot operator, kita juga bisa mengakses properti dari object menggunakan bracket atau tanda kurung siku.

1. user[“home world”];

Untuk mengakses key yang memiliki spasi atau karakter khusus lainnya maka kita perlu menggunakan bracket seperti di atas.

const user = {

firstName: "Luke",

lastName: "Skywalker",

age: 19,

isJedi: true,

"home world": "Tattooine"

};

console.log(`Halo, nama saya ${user.firstName} ${user.lastName}`);

console.log(`Umur saya ${user.age} tahun`);

console.log(`Saya berasal dari ${user["home world"]}`);

/\* output

Halo, nama saya Luke Skywalker

Umur saya 19 tahun

Saya berasal dari Tattooine

Setelah mempelajari bagaimana membuat object dan menampilkan property di dalamnya, selanjutnya kita akan memodifikasi sebuah object. Untuk mengubah nilai properti di dalam object kita gunakan assignment operator (=).

const spaceship = {

name: "Millenium Falcon",

manufacturer: "Corellian Engineering Corporation",

maxSpeed: 1200,

color: "Light gray"

};

spaceship.color = "Glossy red";

spaceship["maxSpeed"] = 1300;

console.log(spaceship);

/\* output

{

name: 'Millenium Falcon',

manufacturer: 'Corellian Engineering Corporation',

maxSpeed: 1300,

color: 'Glossy red'

}

\*/

Tunggu dulu. Object spaceship dideklarasikan sebagai const, tetapi kenapa kita bisa mengubah nilainya?

Yang perlu diperhatikan adalah mengubah nilai berbeda dengan menginisialisasi ulang nilai. Ketika membuat sebuah object, kita tidak terikat dengan properti di dalamnya sehingga kita masih bisa memodifikasi nilainya. Berbeda jika kita menginisialisasi ulang variabel dari object.

1. const spaceship = {
2. name: "Millenium Falcon",
3. manufacturer: "Corellian Engineering Corporation",
4. maxSpeed: 1200,
5. color: "Light gray"
6. };
8. spaceship = { name: "New Millenium Falcon" }; // Error

Ketika kita mengubah object menggunakan assignment operator dan property/key-nya sudah ada, maka nilai di dalamnya akan tergantikan dengan nilai yang baru. Sedangkan, jika property dengan nama key yang ditentukan tidak ditemukan, maka property baru akan ditambahkan ke object.

const spaceship = {

name: "Millenium Falcon",

manufacturer: "Corellian Engineering Corporation",

maxSpeed: 1200,

color: "Light gray"

};

spaceship.color = "Glossy red";

spaceship["maxSpeed"] = 1300;

spaceship.class = "Light freighter";

console.log(spaceship);

/\* output

{

name: 'Millenium Falcon',

manufacturer: 'Corellian Engineering Corporation',

maxSpeed: 1300,

color: 'Glossy red',

class: 'Light freighter'

}

\*/

Kita juga dapat menghapus property pada object menggunakan keyword delete seperti berikut:

const spaceship = {

name: "Millenium Falcon",

manufacturer: "Corellian Engineering Corporation",

maxSpeed: 1200,

color: "Light gray"

};

spaceship.color = "Glossy red";

spaceship["maxSpeed"] = 1300;

delete spaceship.manufacturer;

console.log(spaceship);

/\* output

{ name: 'Millenium Falcon', maxSpeed: 1300, color: 'Glossy red' }

\*/

## Array

Array merupakan tipe data yang dapat mengelompokkan lebih dari satu nilai dan menempatkannya dalam satu variabel. Contoh:

let myArray = ["Cokelat", 42.5, 22, true, "Programming"];

console.log(myArray);

/\* output:

[ 'Cokelat', 42.5, 22, true, 'Programming' ]

\*/

Perbedaan array dengan object adalah data pada array disusun secara berurutan dan diakses menggunakan index. Untuk mengakses nilai di dalam array, kita gunakan tanda kurung siku [] yang di dalamnya berisi angka yang merupakan posisi nilai yang ingin diakses.

1. console.log(myArray[1]);

Jika Anda mengira bahwa konsol akan menampilkan teks ‘Cokelat’, maka tebakan Anda kurang tepat. Dalam sebuah array, indeks dimulai dari 0, sehingga ketika kita mengakses data pada myArray yang berada pada indeks ke-1 artinya data tersebut merupakan data pada posisi ke-2. Jadi nilai yang akan ditampilkan pada konsol adalah 42.5.

let myArray = ["Cokelat", 42.5, 22, true, "Programming"];

console.log(myArray[1]);

/\* output:

42.5

\*/

Lalu, apa yang akan terjadi jika kita berusaha mengakses index di luar ukuran array-nya? Jika kita mengakses nilai array lebih dari index-nya, maka hasilnya akan undefined. Index terakhir array selalu jumlah nilai array - 1.

let myArray = ["Cokelat", 42.5, 22, true, "Programming"];

console.log(myArray[0]);

console.log(myArray[1]);

console.log(myArray[2]);

console.log(myArray[3]);

console.log(myArray[4]);

console.log(myArray[5]);

console.log("Panjang nilai myArray adalah " + myArray.length + ".");

/\* output:

Cokelat

42.5

22

true

Programming

undefined

Panjang nilai myArray adalah 5.

\*/

Sejauh ini kita baru belajar menginisialisasi dan mengakses elemen dari sebuah array. Pastinya Anda bertanya, “Bagaimana kita memanipulasi data pada array tersebut?”

Nah, untuk menambahkan data ke dalam array, kita bisa menggunakan metode push(). Fungsi push ini akan menambahkan data di akhir array.

const myArray = ["Cokelat", 42.5, 22, true, "Programming"];

myArray.push('JavaScript');

console.log(myArray);

/\* output

[ 'Cokelat', 42.5, 22, true, 'Programming', 'JavaScript' ]

\*/

Sedangkan untuk mengeluarkan data atau elemen terakhir dari array, kita bisa gunakan metode pop().

const myArray = ["Cokelat", 42.5, 22, true, "Programming"];

myArray.pop();

console.log(myArray);

/\* output

[ Cokelat, 42.5, 22, true ]

\*/

Metode lain yang bisa kita gunakan untuk memanipulasi data pada array adalah shift() dan unshift(). Metode shift() digunakan untuk mengeluarkan elemen pertama dari array, sementara unshift() digunakan untuk menambahkan elemen di awal array.

const myArray = ["Cokelat", 42.5, 22, true, "Programming"];

myArray.shift();

myArray.unshift("Apple");

console.log(myArray);

/\* output

[ 'Apple', 42.5, 22, true, 'Programming' ]

\*/

Lalu bagaimana jika kita ingin menghapus data dari array? Sama seperti object, kita bisa menggunakan keyword delete.

const myArray = ["Cokelat", 42.5, 22, true, "Programming"];

delete myArray[1];

console.log(myArray);

/\* output

[ 'Cokelat', <1 empty item>, 22, true, 'Programming' ]

\*/

Namun, perhatikan di sini bahwa keyword delete hanya menghapus data pada index yang ditentukan lalu membiarkan posisi tersebut kosong. Untuk menghapus elemen, gunakan metode splice() seperti ini:

const myArray = ["Cokelat", 42.5, 22, true, "Programming"];

myArray.splice(2, 1); // Menghapus dari index 2 sebanyak 1 elemen

console.log(myArray);

/\* output

[ 'Cokelat', 42.5, true, 'Programming' ]

\*/

metode splice() bisa juga untuk menambahkan array baru seperti ini:

const fruits = ["Banana", "Orange", "Apple", "Mango"];

// At position 2, add 2 elements:

fruits.splice(2, 1, "Lemon", "Kiwi");

console.log(fruits);

//Output :

Banana,Orange,Lemon,Kiwi,Mango