2022

MODUL 4

PENGEMBANGAN APLIKASI PLATFORM KHUSUS

DAVID SETIADI

UNSAP | SUMEDANG 2022

MODUL 4

FUNGSI DI JAVASCRIPT

Fungsi adalah sub-program yang bisa digunakan kembali baik di dalam program itu sendiri, maupun di program yang lain.

Dalam pemrograman, fungsi sering digunakan untuk membungkus program menjadi bagian-bagian kecil. Logika program yang ada di dalam fungsi dapat kita gunakan kembali dengan memanggilnya. Sehingga tidak perlu menulis ulang.

Fungsi di dalam Javascript adalah sebuah objek. Karena memiliki properti dan juga method.

Ada 4 cara yang bisa kita lakukan untuk membuat fungsi di Javascript:

1. Menggunakan cara biasa;
2. Menggunakan ekspresi;
3. Menggunakan tanda panah (`=>`);
4. dan menggunakan Constructor.

Mari kita coba semuanya…

1. Membuat Fungsi dengan Cara Biasa

Cara ini paling sering digunakan, terutama buat yang baru belajar Javascript.

```
function namaFungsi(){
    console.log("Hello World!");
}
```

2. Membuat Fungsi dengan Ekspresi

Cara membuatnya :

```
var namaFungsi = function(){
    console.log("Hello World!");
}
```

Kita menggunakan variabel, lalu diisi dengan fungsi. Fungsi ini sebenarnya adalah fungsi anonim (anonymous function) atau fungsi tanpa nama.

3. Membuat Fungsi dengan Tanda Panah

Cara ini sering digunakan di kode Javascript masa kini, karena lebih sederhana. Akan tetapi sulit dipahami bagi pemula. Fungsi ini mulai muncul pada standar ES6.

Contoh :

```
var namaFungsi = () => {
    console.log("Hello World!");
}

// atau seperti ini (jika isi fungsi hanya satu baris):
var namaFungsi = () => console.log("Hello World!");
```

Sebenarnya hampir sama dengan yang menggunakan ekspresi. Bedanya, kita menggunakan tanda panah (=>) sebagai ganti function. Pembuatan fungsi dengan cara ini disebut arrow function.

4. Membuat Fungsi dengan Kostruktor

Cara ini sebenarnya tidak direkomendasikan oleh Developer Mozilla, karena terlihat kurang bagus. Soalnya body fungsinya dibuat dalam bentuk string yang dapat mempengaruhi kinerja engine javascript.

Contoh :

```
var namaFungsi = new Function('console.log("Hello World!");');
```

Untuk yang masih pemula, direkomendasikan gunakan cara yang pertama dulu. Nanti kalau sudah terbiasa baru coba gunakan cara ke-2 dan ke-3.

Cara Memanggil/Eksekusi Fungsi

Kita bisa memanggil fungsi di dalam kode Javascript dengan menuliskan nama fungsinya seperti ini:

```
namaFungsi();
```

COntoh :

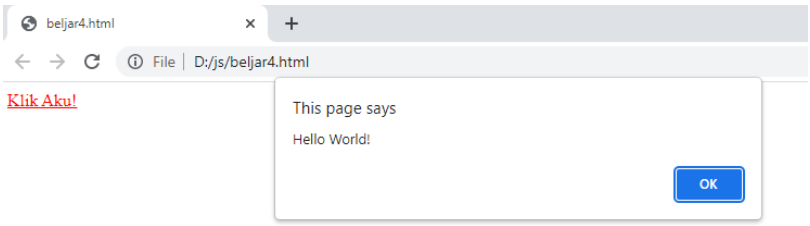
```
// membuat fungsi
function sayHello(){
    console.log("Hello World!");
}

// memanggil fungsi
sayHello() // maka akan menghasilkan -> Hello World!
```

Selain dengan cara di atas, kita juga bisa memanggil fungsi melalui atribut event pada HTML.

Contoh :

```
<!DOCTYPE html>
<html>
<head>
    <script>
    // membuat fungsi
    var sayHello = () => alert("Hello World!");
    </script>
</head>
<body>
    <!-- Memanggil fungsi saat link diklik -->
    <a href="#" onclick="sayHello()">Klik Aku!</a>
</body>
</html>
```

Hasilnya :



Fungsi dengan Parameter

Parameter adalah variabel yang menyimpan nilai untuk diproses di dalam fungsi.

Contoh :

```
function kali(a, b){
    hasilKali = a * b;
    console.log("Hasil kali a*b = " + hasilKali);
}
```

Pada contoh di atas, a dan b adalah sebuah parameter.

Lalu cara memanggil fungsi yang memiliki parameter adalah seperti ini:

```
kali(3, 2); // -> Hasil kali a*b = 6
```

Kita memberikan 3 untuk parameter a dan 2 untuk parameter b.

Fungsi yang Mengembalikan Nilai

Agar hasil pengolahan nilai di dalam fungsi dapat digunakan untuk proses berikutnya, maka fungsi harus mengembalikan nilai.

Pengembalian nilai pada fungsi menggunakan kata kunci return kemudian diikuti dengan nilai atau variabel yang akan dikembalikan. Contoh:

```
function bagi(a,b){
    hasilBagi = a / b;
    return hasilBagi;
}

// memanggil fungsi
var nilai1 = 20;
var nilai2 = 5;
var hasilPembagian = bagi(nilai1, nilai2);

console.log(hasilPembagian); //-> 4
```

Contoh Program Javascript dengan Fungsi

Setelah kita paham dasar-dasar pembuatan fungsi dan jenis-jenisnya, sekarang mari kita coba membuat program sederhana.

Program ini berisi CRUD (Crete, Read, Update, Delete) data barang yang tersimpan dalam sebuah array.

Silahkan buat dua file baru **fungsi.js** dan **index.html**

File index.html adalah file yang menampilkan halaman web. Sedangkan file fungsi.js adalah programnya.

Berikut ini isi file index.html:

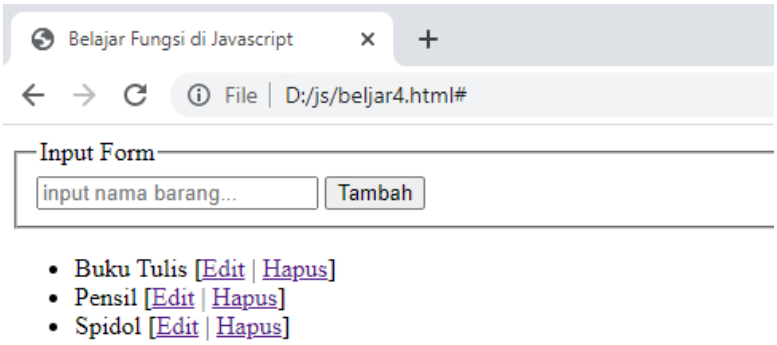
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Belajar Fungsi di Javascript</title>
</head>
<body>

    <fieldset>
        <legend>Input Form</legend>
        <input type="text" name="barang" placeholder="input nama barang..."
        <input type="button" onclick="addBarang()" value="Tambah" />
    </fieldset>

    <div>
        <ul id="list-barang">
        </ul>
    </div>

    <script src="fungsi.js"></script>
</body>
</html>
```

Berikutnya kita akan buat kode di file fungsi.js. Silahkan gunakan gaya pembuatan fungsi yang kamu sukai. Pada contoh ini, kita akan menggunakan cara yang pertama. Karena lebih mudah.

Berikut ini isi file fungsi.js:

```js
var dataBarang = [
    "Buku Tulis",
    "Pensil",
    "Spidol"
];
function showBarang(){
    var listBarang = document.getElementById("list-barang");
    // clear list barang
    listBarang.innerHTML = "";
    // cetak semua barang
    for(let i = 0; i < dataBarang.length; i++){
        var btnEdit = "<a href='#' onclick='editBarang("+i+")'>Edit</a>";
        var btnHapus = "<a href='#' onclick='deleteBarang("+i+")'>Hapus</a>";
        listBarang.innerHTML += "<li>" + dataBarang[i] + " ["+btnEdit + " | "+ btnHapus +"]</li>";
    }
}
function addBarang(){
    var input = document.querySelector("input[name=barang]");
    dataBarang.push(input.value);
    showBarang();
}
function editBarang(id){
    var newBarang = prompt("Nama baru", dataBarang[id]);
    dataBarang[id] = newBarang;
    showBarang();
}
function deleteBarang(id){
    dataBarang.splice(id, 1);
    showBarang();
}
showBarang();
```

Hasilnya :



OBJEK DI JAVASCRIPT

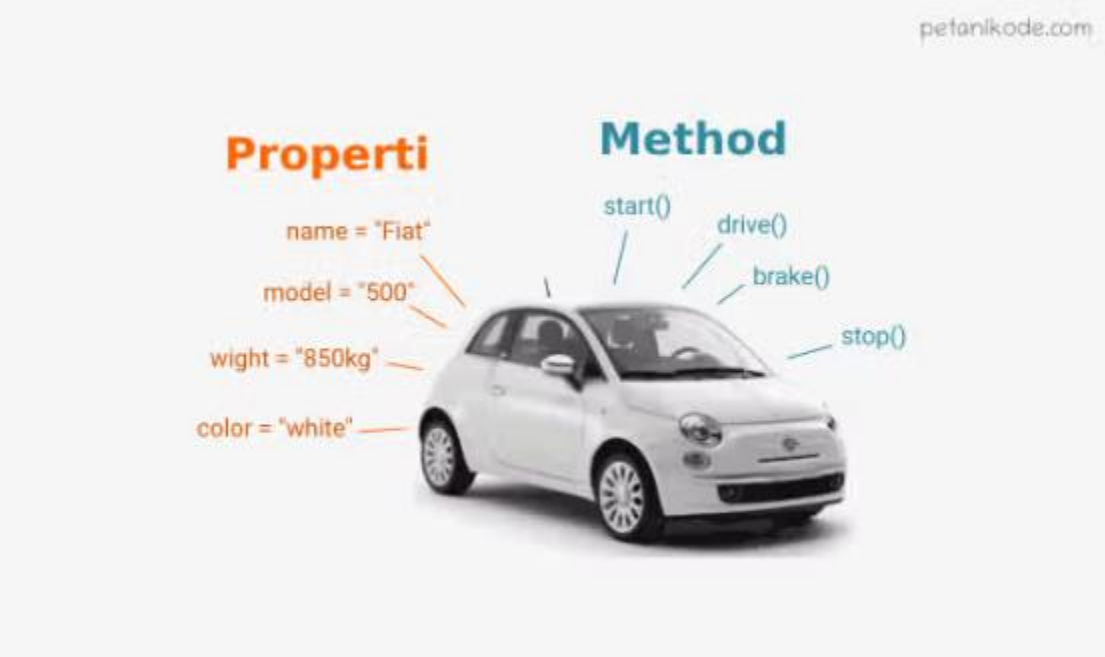
Dalam kehidupan nyata, kita sering menjumpai objek. Objek adalah segala sesuatu yang ada di dunia ini.

Entah itu benda mati ataupun makhluk hidup. Semuanya objek. Objek-objek ini dapat kita modelkan di dalam pemrograman.

Biasanya menggunakan paradigma OOP (Object Oriented Programming) atau pemrograman beorientasikan objek. Nah paradigma OOP ini merupakan sebuah teknik atau cara di dalam pemrograman dimana segala sesuatu di pandang sebagai objek.

Objek-objek ini dapat saling berinteraksi sehingga membentuk sebuah program. Objek sebenarnya adalah sebuah variabel yang menyimpan nilai (properti) dan fungsi (method).

Contoh objek mobil:



Bagaimana cara kita memodelkan mobil ini di dalam kode program?

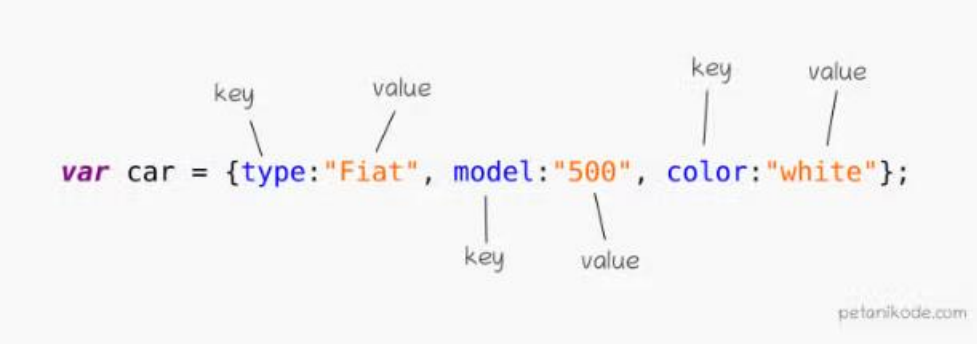
Bisa saja seperti ini:

```
var car = "Mobil Fiat";
```

Tapi variabel car hanya akan menyimpan nama mobil saja. Karena itu, kita harus menggunakan objek.

Objek pada javascript, dapat dibuat dengan tanda kurung kurawal dengan isi berupa key dan value.

Contoh:



Kode di atas bisa juga ditulis seperti ini:

```
var car = {
    type:"Fiat",
    model:"500",
    color:"white"
};
```

Apa Perbedaan Properti dan Method?

Pada contoh di atas, kita baru hanya membuat properti saja.

Properti adalah ciri khas dari objek (variabel). Sedangkan method adalah perilaku dari objek (fungsi).

Lalu bagaimana cara membuat metohd di dalam objek?

Method dapat dibuat dengan cara mengisi nilai (value) dengan sebuah fungsi.

Contoh:

```
var car = {
    // properti
    type: "Fiat",
    model: "500",
    color: "white",

    // method
    start: function(){
        console.log("ini method start");
    },
    drive: function(){
        console.log("ini method drive");
    },
    brake: function(){
        console.log("ini method brake");
    },
    stop: function(){
        console.log("ini method stop");
    }

};
```

Cara Mengakses Properti dan Method Objek

Kita sudah tahu cara membuat objek…

Lalu pertanyaanya:

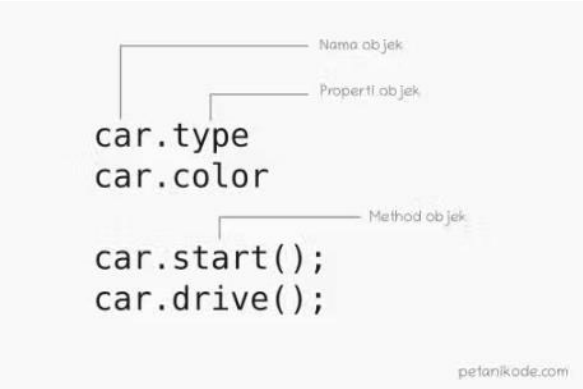
Bagaimana cara mengakses properti dan method dari objek?

Caranya menggunakan tanda titik atau dot (.), lalu diikuti dengan nama properti atau method.

Contoh:

```
console.log(car.type);
console.log(car.color);

car.start();
car.drive();
```

perhatikan car.type, car.color, car.start(), dan car.drive()!



Untuk mengakses properti, kita cukup gunakan nama objek.properti. Sedangkan untuk method, kita harus menggunakan tanda kurung. Ini menyatakan kalau kita ingin mengeksekusi fungsi.

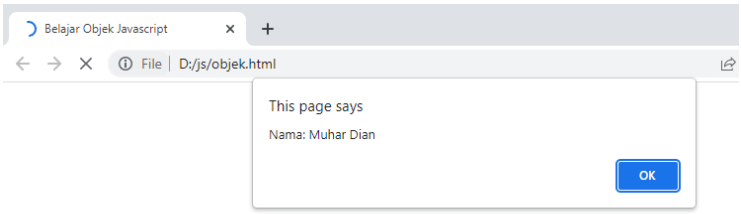
Menggunakan Keyword this

Kata kunci this digunakan untuk mengakses properti dan method dari dalam method (objek).

Contoh:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Belajar Objek Javascript</title>

    <script>
        var person = {
            firstName: "Muhar",
            lastName: "Dian",
            showName: function(){
                alert(`Nama: ${this.firstName} ${this.lastName}`);
            }
        };

        person.showName();
    </script>
</head>
<body>
```

Hasilnya :



Kata kunci this pada contoh di atas akan mengacu pada objek person.

Membuat Class Objek dengan this

Pada pemrograman berorientasikan objek, kita biasanya membuat objek dengan membuat instance dari class. Akan tetapi pada contoh-contoh di atas, kita membuat objeknya secara langsung. Lalu bagaimana kalau kita inign membuat objek yang lain dengan properti yang sama.

Bisa saja dibuat dua seperti ini:

```
var person = {
    firstName: "Muhar",
    lastName: "Dian",
    showName: function(){
        alert(`Nama: ${this.firstName} ${this.lastName}`);
    }
};

var person2 = {
    firstName: "Petani",
    lastName: "Kode",
    showName: function(){
        alert(`Nama: ${this.firstName} ${this.lastName}`);
    }
};
```

Ini tentu tidak efektif, jika kita ingin membuat banyak objek. Karena kita harus menulis ulang kode yang sama. Solusinya kita bisa gunakan class. Oh iya, saya ingin kasih tahu:

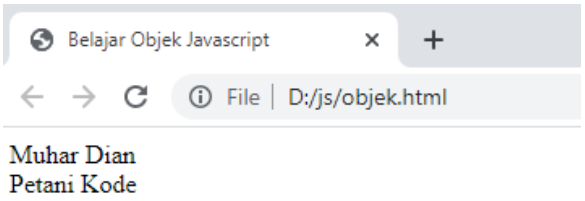
Pada Javascript versi ES5, class belum ada. Fitur ini baru ditambahkan pada Javascript versi ES6.

Pada ES5, kita bisa membuat class dengan fungsi. Lalu di dalamnya menggunakan kata kunci this.

Contoh:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Belajar Objek Javascript</title>

    <script>
        function Person(firstName, lastName){
            // properti
            this.firstName = firstName;
            this.lastName = lastName;

            // method
            this.fullName = function(){
                return `${this.firstName} ${this.lastName}`
            }
            this.showName = function(){
                document.write(this.fullName());
            }
        }

        var person1 = new Person("Muhar", "Dian");
        var person2 = new Person("Petani", "Kode");

        person1.showName();
        document.write("<br>");
        person2.showName();
    </script>
</head>
<body>
```

Hasilnya :



Perhatikanlah contoh di atas!

Kita membuat objek baru dengan kata kunci new, lalu diberikan nilai parameter firstName dan lastName.

```
var person1 = new Person("Muhar", "Dian");
```

Jadi, berapapun objek yang ingin kita buat cukup gunakan kata kunci new saja.

Referensi :

- Petanikode.com
- Dicoding.com
- W3schools.com