

## Perkenalan Javascript dan Manipulasi DOM

JavaScript adalah bahasa pemrograman sisi klien, sehingga seluruh proses berjalan di sisi pengguna, bukan sisi server. JavaScript diperlukan dalam pengembangan situs web ketika kita membutuhkan interaksi dari pengguna. Padahal, jika sebuah website hanya menggunakan HTML dan CSS, maka hanya akan menampilkan konten statis.

Karena JavaScript diproses di sisi klien, itu bergantung pada kemampuan dan konfigurasi browser saat melakukan proses seperti kompilasi atau rendering di DOM. Bahkan pengguna dapat melarang JavaScript berjalan di browser mereka dengan menonaktifkan dukungan untuk JavaScript.

Document Object Model, atau DOM, adalah alat yang disediakan untuk pengembang yang memungkinkan mengubah gaya dan konten situs web. Dokumen CSS situs web dan HTML-nya mengatur tampilan situs. Ini karena browser memelihara Model Objek Dokumen yang menampung semua manipulasi elemen yang ditangani oleh JavaScript. DOM memungkinkan untuk menata situs web melalui objek induknya.

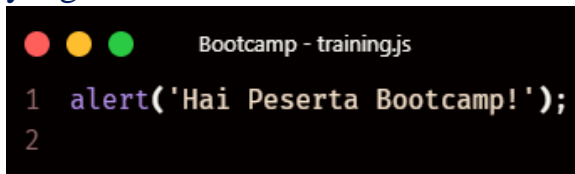
Untuk menghubungkan file HTML dengan javascript kita harus menuliskan tag script didalam tab body pada HTML.

```
<script src="js/script.js"></script>
```

Didalam javascript kita harus memahami beberapa hal dasar sebagai berikut :

### ❖ *Statement*

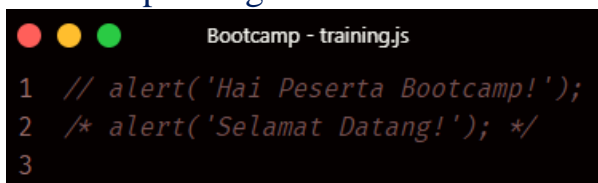
Statement adalah sebuah perintah yang bertujuan untuk memberitahu apa yang harus dilakukan oleh browser.

A screenshot of a code editor window titled "Bootcamp - training.js". It shows two lines of code: line 1 is `alert('Hai Peserta Bootcamp!');` and line 2 is empty.

```
1 alert('Hai Peserta Bootcamp!');
2
```

### ❖ *Comment*

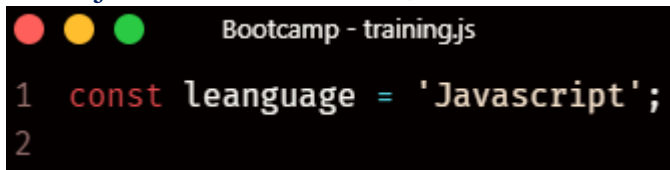
Komentar adalah sebuah baris yang tidak akan di eksekusi oleh interpreter/compiler. Fungsi dari komentar untuk menjelaskan maksud dari suatu baris kode. Dengan cara memberikan double slash (//) dan bisa juga baris di apit dengan slash dan hasterisk (/\* kode terkomentar \*/)

A screenshot of a code editor window titled "Bootcamp - training.js". It shows three lines of code: line 1 is a single-line comment `// alert('Hai Peserta Bootcamp!');`, line 2 is a multi-line comment `/* alert('Selamat Datang!'); */`, and line 3 is empty.

```
1 // alert('Hai Peserta Bootcamp!');
2 /* alert('Selamat Datang!'); */
3
```

### ❖ *Variable*

Variabel umumnya digunakan untuk menyimpan informasi atau nilai yang akan dikelola di dalam sebuah program. Penamaan variable harus jelas dan masuk akal, dan tidak boleh diawali dengan angka.

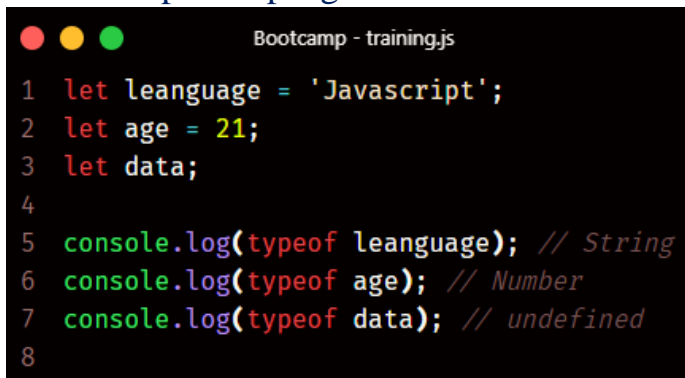


```
Bootcamp - training.js
1  const leanguage = 'Javascript';
2
```

Terdapat tiga macam keyword yang dapat digunakan untuk mendefinisikan sebuah variable, yaitu : var, let, dan const.

### ❖ *Tipe data*

Nilai yang ditetapkan dalam variable pasti memiliki tipe data. Tipe data merupakan pengklasifikasian data berdasarkan jenis data tersebut.



```
Bootcamp - training.js
1  let leanguage = 'Javascript';
2  let age = 21;
3  let data;
4
5  console.log(typeof leanguage); // String
6  console.log(typeof age); // Number
7  console.log(typeof data); // undefined
8
```

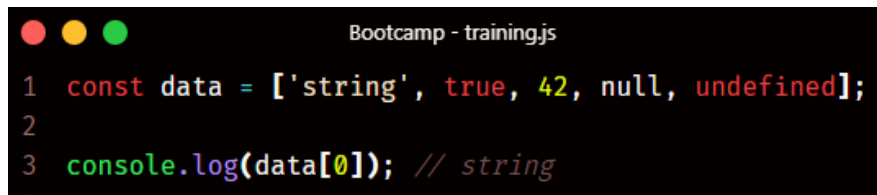
Didalam javascript terdapat beberapa tipe data diantara : undefined, null, number, object, nan, string, dan boolean.

### ❖ *Array & object*

Array dan object keduanya sama-sama dapat menyimpan lebih dari satu tipe data dasar yang akan digunakan untuk mengolah data.

Array :

Array merupakan tipe data yang dapat mengelompokkan lebih dari satu nilai dari tipe data lain dengan menempatkannya pada satu variabel. Setiap data yang ada di dalam array disebut dengan element, Nilai - nilai yang berada pada array disusun dan diakses secara indexing. Untuk mengakses nilai di dalam array kita gunakan tanda kurung siku [] yang di dalamnya berupa angka yang merupakan posisi nilai yang ingin diakses.



```

Bootcamp - training.js
1  const data = ['string', true, 42, null, undefined];
2
3  console.log(data[0]); // string

```

## Object :

Objek seperti array yang dapat menyimpan banyak tipe data yang berbeda. Objek diakses melalui pendekatan key-value bukan pengindeksan. Mengakses nilai dengan menggunakan key. Key juga dikenal sebagai properti. Untuk menetapkan objek pada variabel gunakan tanda kurung kurawal { } dalam menginisialisasinya. Kemudian di dalamnya kita tetapkan key: value.



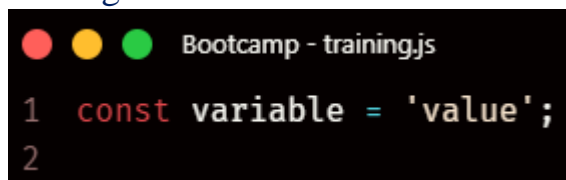
```

Bootcamp - training.js
1  const person = {
2    name: 'John',
3    age: 30,
4    city: 'New York',
5  };
6
7  console.log(person.name); // John
8  console.log(person['city']); // New York
9

```

### ❖ *Assignment operator*

Pada dasarnya operator ini adalah tanda sama dengan (=), di mana tanda ini digunakan untuk memberi atau menginisialisasi nilai pada variabel. Variabel yang akan diberikan nilai ditempatkan pada sebelah kiri dan nilainya ditempatkan pada sebelah kanan (nilai dapat berupa variabel lain atau nilai primitif, array, atau objek). Di antara keduanya terdapat operator assignment.



```

Bootcamp - training.js
1  const variable = 'value';
2

```

### ❖ *Comparison operator*

Operator perbandingan (comparison operators) digunakan untuk membandingkan suatu nilai dari masing-masing operan. Hasil dari operator perbandingan ini adalah boolean true atau false.

Operator	Penjelasan	Contoh	Hasil
==	Sama dengan	5 == 5	1 (true)
!=	Tidak sama dengan	5 != 5	0 (false)
>	Lebih besar	5 > 6	0 (false)
<	Lebih kecil	5 < 6	1 (true)
>=	Lebih besar atau sama dengan	5 >= 3	1 (true)
<=	Lebih kecil atau sama dengan	5 <= 5	1 (true)

Operator perbandingan biasa dipakai dalam proses pengambilan keputusan atau percabangan kode program. Sebagai contoh, jika angka pertama lebih besar dari kedua, maka jalankan perintah ini. Atau jika string password = '123456' maka berikan hak akses.

#### ❖ *If/else statement atau condition statement*

Ketika mengembangkan sebuah program tentu terdapat alur atau flow proses ketika kode dijalankan. Kita dapat mengontrol alur jalannya program ketika suatu kondisi terjadi. If/else statement dapat digambarkan seolah-olah kita memberikan pertanyaan benar atau salah pada JavaScript, lalu memberikan perintah sesuai output dari pertanyaan tersebut. Misalnya, terdapat variabel x dengan nilai 40, kemudian kita bertanya “Hai Js Apakah x lebih dari 60?” jika kondisi tersebut true, maka kita dapat memerintahkan JavaScript untuk menampilkan huruf C. Jika false, kita perintahkan JavaScript untuk menampilkan huruf D”.

```

Bootcamp - training.js
1  const x = 40;
2
3  if (x > 60) {
4    console.log('C');
5  } else {
6    console.log('D');
7  }
8  //Output C
9

```

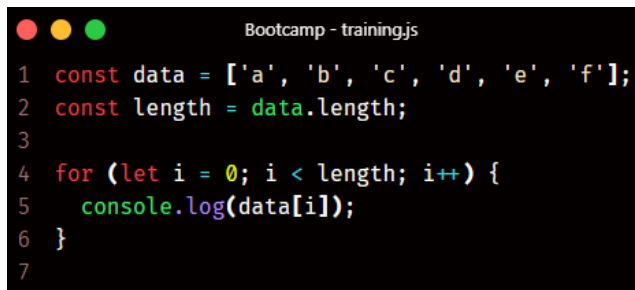
Selain dapat menggunakan statement kondisi if/else ada juga statement switch case untuk melakukan pengecekan kondisi.

## ❖ *Loop*

Dalam dunia programming, Looping adalah sebuah urutan perintah yang secara menerus diulang hingga suatu kondisi tercapai. Kondisi yang dimaksud disini dapat dalam bentuk yang berbeda-beda seperti hanya mendapatkan data atau merubah data, bisa juga memeriksa sebuah nilai apakah sudah mencapai jumlah yang ditentukan atau belum.

Didalam javascript terdapat beberapa cara untuk melakukan looping beberapa diantaranya :

- For loop
- For of loop
- For in loop
- While
- Do while

A screenshot of a code editor window titled "Bootcamp - training.js". The code is as follows:

```
1 const data = ['a', 'b', 'c', 'd', 'e', 'f'];
2 const length = data.length;
3
4 for (let i = 0; i < length; i++) {
5   console.log(data[i]);
6 }
7
```

## ❖ *Function*

Function merupakan potongan kecil dari program dan tidak akan di eksekusi sampai fungsi tersebut di panggil. Function dapat memiliki argument maupun tidak sama sekali. Jelasnya 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.

```

fe-umkmbp - training.js
1  function print(data) {
2      console.log(data);
3  }
4
5  const person = {
6      name: 'Zain',
7      age: 20,
8  };
9
10 print(person.name); // Zain
11 print(person['age']); // 20
12

```

### ❖ *Variable scope*

Scope dalam JavaScript adalah konsep yang digunakan untuk membatasi akses suatu variabel. Ada dua tipe scope yaitu lokal dan global. Variabel JavaScript menggunakan fungsi untuk mengelola cakupannya, jika variabel didefinisikan di luar fungsi, maka variabel akan bersifat global. Jika variabel didefinisikan di dalam fungsi, maka variabel bersifat lokal dan cakupannya hanya pada fungsi tersebut atau turunannya.

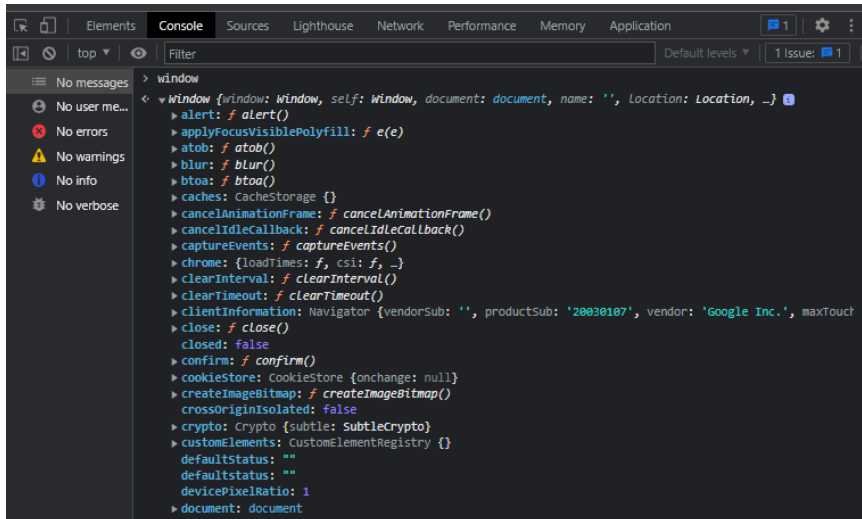
```

fe-umkmbp - training.js
1  let events = 'Bootcamp';
2
3  function training() {
4      console.log(events);
5      events = 'Wilayah 1';
6      console.log(events);
7      const foo = 'Bar';
8      console.log(foo);
9  }
10
11 training();
12 /* Output:
13 Bootcamp
14 Wilayah 1
15 Bar
16 */
17 console.log(foo); // Reference Error
18

```

### ❖ *Browser Object Model (BOM)*

Pada JavaScript, browser dikenal sebagai window object. Di dalam objek window itu sendiri terdapat banyak properties dan method yang bisa digunakan, salah satu yang sudah kita ketahui adalah alert(). Kita bisa melihat secara lengkap apa saja properti, method serta event yang ada pada objek window melalui console browser dengan mengetikkan window pada console.



Berikut beberapa properti dan method yang sering digunakan seperti:

Property/Method	Description
History	Sebagai navigasi (go back atau go forward) histori URL browser.
Location	Mendapatkan URL yang terdapat pada address bar browser.
Alert()	Menampilkan dialog alert dengan pesan dan tombol “ok”.
Close()	Menutup tab yang aktif.
Confirm()	Menampilkan dialog dengan pesan dan tombol “ok” dan “cancel”. Method ini akan mengembalikan nilai boolean sesuai response dari pengguna.
Prompt()	Menampilkan dialog dengan pesan dan teks input. Method ini akan mengembalikan nilai string sesuai response dari pengguna.

Dalam mengakses properti maupun method pada object window, tidak harus memanggil object window terlebih dahulu karena method ataupun property tersebut bersifat global.

#### ❖ *Manipulasi DOM :*

##### ○ *Get element*

*Cara mendapatkan element melalui DOM yaitu dengan menggunakan properti dari object window yaitu property **document**. Property ini merepresentasikan dokumen dari HTML*

##### ○ *Manipulation attribute*

Untuk mengubah nilai atribut pada suatu element kita dapat menggunakan method **setAttribute()**. Method ini membutuhkan dua buah argument string dari nama attribute dan valuenya.

- *Manipulation element*

Dengan javascript kita dapat mengubah tampilan atau konten yang ada didalam HTML. Konten pada element dapat di ubah ataupun di akses dengan property **innerHTML** (dalam bentuk HTML) atau **innerText** (dalam bentuk teks).

- *Membuat element baru*

Selain menggunakan **innerHTML** untuk menambahkan element, kita juga dapat menggunakan method **createElement()**. Namun method ini hanya akan membuat element saja sehingga belum akan tampil di browser kita. Untuk memunculkan element yang telah di buat dengan method ini kita perlu memasukkan element tersebut pada **body** element dengan menggunakan method **append()**.

- *Event*

Javascript dapat menerima event atau aksi yang terjadi pada element, hal ini disebut sebagai event handler. Event merupakan interaksi yang dilakukan oleh pengguna seperti click, mengarahkan cursor ke element, dan lain sebagainya

. Untuk menambahkan Event Handler pada suatu element kita dapat menggunakan **addEventListener()**. Method ini membutuhkan setidaknya dua buah argument. Argument pertama adalah tipe dari eventnya, dan yang kedua adalah fungsi yang akan dijalankan bila event yang kita definisikan terjadi.

Beberapa tipe event :

- *DOMContentLoaded*

Event **DOMContentLoaded** diaktifkan ketika dokumen HTML telah sepenuhnya diuraikan, dan semua skrip telah diunduh dan dijalankan. Tidak menunggu hal-hal lain seperti gambar, skrip asinkron selesai dimuat.

- *Submit*

Event submit terjadi ketika form di kirim/submit

- *Click*

Event click terjadi ketika user menclick element

- *Keyup*

Event keyup terjadi ketika user menekan dan melepaskan tombol pada keyboard

- *Keypress*



- *Event keypress terjadi ketika user menekan dan menahan tombol pada keyboard*

- *Custom event*

*Untuk membuat custom event kita dapat menuliskan baris kode*

```
const myalert = new Event('myalert');

document.addEventListener('myalert', () => {
  alert('ok');
});
```

- *dispatchEvent()*

*Function dispatchEvent() untuk trigger custom event yang telah dideklarasikan dan dipasangkan,*

```
document.dispatchEvent(new Event('myalert'));
```