



Front End Development 1

Sesi 18



JavaScript⁺ Object

Point of Discussion

- Konsep Object
- Object vs Array
- Cara buat Object
- Cara akses Object
- Manipulasi Object



Konsep Object

Apa itu “Orang”?

- nama : “Hanna”
- warna rambut : “multi-color”
- panjang rambut : “armpit length”
- berkacamata : true
- jenis pakaian : “bohemian”
- umur : 33
- gender : “wanita”
- tinggi badan : 172
- jumlah gigi : 31

Javascript Object

Kumpulan tidak berurut yang merangkai beberapa property dan property memiliki nama/key dan value (key-value pairs).



Object vs Array

Object

```
1 let orang = {  
2   nama: "Hanna",  
3   "warna rambut": "multi-color",  
4   berkacamata: true,  
5   umur: 33  
6 }
```

Array

```
1 let orang = [  
2   ["nama", "Hanna"],  
3   ["warna rambut", "multi-color"],  
4   ["berkacamata", true],  
5   ["umur", 33]  
6 ]
```



Kapan kita pakai Object?

1. Key - Value pair
2. tidak harus urut
3. membuat suatu objek nyata





Kapan kita pakai Array?

1. Membuat list
2. Item di listurut
3. membuat suatu antrian/ urutan/ tabel (multi-array)



Cara buat Object

{ key: value, key2: value2 }

```
1 let orang = {  
2     nama: "Hanna",  
3     "warna rambut": "multi-color",  
4     berkacamata: true,  
5     umur: 33  
6 }
```

Tiga cara akses Value

```
1  let orang = {  
2      nama: "Hanna",  
3      "warna rambut": "multi-color",  
4      berkacamata: true,  
5      umur: 33  
6  }  
7  
8  let somekey = "berkacamata"  
9  orang[somekey]  
10 orang["berkacamata"]  
11 orang.berkacamata
```

Perhatikan tiga cara berbeda (line 8-11) untuk mendapatkan value yang sama dari object orang

Tiga cara akses Value

```
let somekey = "berkacamata"  
orang[somekey]
```

```
orang["berkacamata"]
```

```
orang.berkacamata
```

1. dengan kurung kotak diisi variable yang isinya string
2. dengan kurung kotak diisi string
3. dengan titik dan langsung nama key nya



Tiga cara akses Value

```
1 let orang = {  
2     nama: "Hanna",  
3     "warna rambut": "multi-color",  
4     berkacamata: true,  
5     umur: 33  
6 }  
7  
8 let somekey = "berkacamata"  
9 orang[somekey]  
10 orang["berkacamata"]  
11 orang.berkacamata
```

value yang didapat sama, object orang, key “berkacamata”, value nya

true



Create/ Add Value milik Key

Untuk menambahkan key-value baru kedalam sebuah object, ada beberapa cara:

1. `object.key = value`
2. `object["key"] = value`
3. `object[varBerisiStringKey] = value`



Manipulasi object

Add

diberikan object
orang

```
let orang = {nama: "Hanna"}
```

coba tambahkan
detail berkacamata
(add key-value)

```
orang["berkacamata"] = true
```

object nya akan
berubah menjadi

```
{nama: "Hanna", berkacamata: true}
```



Edit Value milik Key

Untuk edit value baru sebuah object, ada beberapa cara:

1. `object.key = valueBaru`
2. `object["key"] = valueBaru`
3. `object[varBerisiStringKey] = valueBaru`

Manipulasi object

Edit

masih di object
yang sama

coba ganti detail
namanya
(edit key-value)

object nya akan
berubah menjadi

```
{nama: "Hanna", berkacamata: true}
```

```
orang["nama"] = "Hanna Montana"
```

```
{nama: "Hanna Montana", berkacamata: true}
```



Delete Value milik Key

Untuk delete value baru sebuah object, ada beberapa cara:

1. `delete` object.key
2. `delete` object["key"]
3. `delete` object[varBerisiStringKey]



Manipulasi object

delete

masih di object
yang sama

```
{nama: "Hanna Montana", berkacamata: true}
```

coba delete detail
berkacamata
(delete key-value)

```
delete orang["berkacamata"]
```

object nya akan
berubah menjadi

```
{nama: "Hanna Montana"}
```





Javascript⁺ Function

Apa itu Function / Fungsi

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.

Cara Membuat Fungsi

1. Menggunakan cara biasa

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

2. Menggunakan ekspresi;

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

3. Menggunakan tanda panah (=>)

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

Cara Memanggil/Eksekusi Fungsi

Cara Panggil Fungsi =>

```
namaFungsi();
```

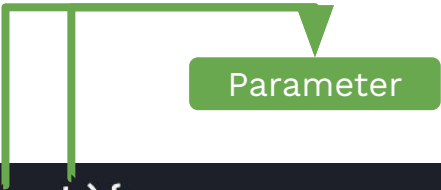
```
// membuat fungsi
function sayHello(){
    console.log("Hello World!");
}

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



Fungsi dengan Parameter

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



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

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

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



Fungsi yang 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
```



Study Case: Modular Function

Buatlah sebuah program untuk membuat enkripsi password yang di-input user agar tidak bisa dimengerti oleh orang lain dengan aturan sebagai berikut:

1. Hilangkan semua spasi yang ada di dalam input
2. Reverse input
3. Ganti huruf vokal menjadi satu huruf setelahnya (A menjadi B, I menjadi H dan seterusnya)

Bisa saja kita membuat satu function untuk menyelesaikan semuanya, tapi akan lebih sulit untuk di-debug (mencari kesalahan logika/code).

Best practice-nya, kamu bisa buat 3 function untuk masalah di atas. Jadi setiap step masalah akan diselesaikan dengan 1 function.



Study Case: Modular Function

```
1 function removeSpaces (text) {  
2   // Code to remove spaces from text  
3 }  
4  
5 function reverseText (text) {  
6   // Code to reverse the text  
7 }  
8  
9 function updateVowels (text) {  
10  // Code to update vowels  
11 }  
12  
13 var password = 'dimitri w';  
14 var noSpaces = removeSpaces(password);  
15 var reversed = reverseText(noSpaces);  
16 var encryptedPassword = updateVowels(reversed);  
17  
18 console.log(encryptedPassword);
```



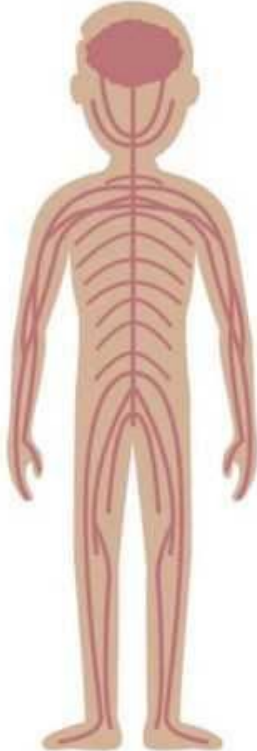
The background is a solid dark blue. It features several decorative geometric elements: a light blue triangle in the upper left, a dark blue circle in the upper right, a light blue rectangle in the top right corner, a white circle in the lower left, and a white arc in the bottom right corner.

JQuery DOM

HTML

JS

CSS



Review Website

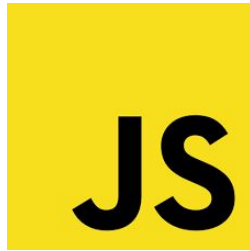
Struktur website

- **HTML**
Struktur kerangka informasi halaman website
- **CSS**
Style dekorasi untuk memperindah website
- **Javascript**
Fungsi penunjang untuk memberikan aksi dan interaksi pada halaman website

DOM

Apa itu DOM?

Document Object Model atau lebih sering disebut dengan DOM adalah suatu pemrograman interface pada dokumen HTML. Dimana DOM sendiri akan mendefinisikan struktur dari setiap HTML Element menjadi Object yang nantinya dapat di akses dan dimanipulasi menggunakan Javascript.



DOM Tolong ganti warna tulisan "Hacktiv8" jadi Orange dong

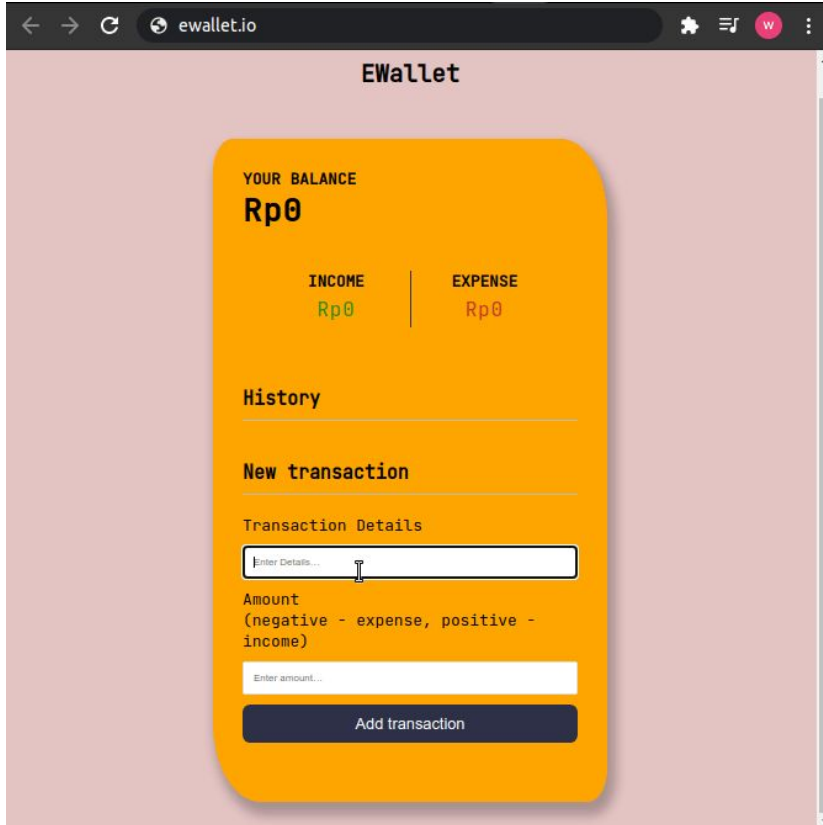


DOM Interface



File HTML

Manfaat DOM di HTML



- Membuat Website menjadi lebih interaktif
- Mengakses element dalam suatu HTML
- Manipulasi attribute,element, event dalam HTML



Akses DOM

Sekarang, kita akan mencoba mengakses elemen dom dari html dibawah ini:

```
i 1 <html>
2   <head>
3     <title>Contoh Webpage Standard</title>
4   </head>
5   <body>
6     <div id="page-title">Sample Page Title</div>
7     <h1>Test Sample Heading</h1>
8     <div class="page-box">Page Box 1</div>
9     <div class="page-box">Page Box 2</div>
10    <div class="page-box">Page Box 3</div>
11    <script src="js-simple-dom-script.js"></script>
12  </body>
13 </html>
```

Akses DOM

Pertama, dimulai dengan mengakses elemen dengan nama id, nama class dan nama tag.

```
1 <html>
2   <head>
3     <title>Contoh Webpage Standard</title>
4   </head>
5   <body>
6     <div id="page-title">Sample Page Title</div>
7     <h1>Test Sample Heading</h1>
8     <div class="page-box">Page Box 1</div>
9     <div class="page-box">Page Box 2</div>
10    <div class="page-box">Page Box 3</div>
11    <script src="js-simple-dom-script.js"></script>
12  </body>
13 </html>
```

```
1 var pageTitleElement =
  document.getElementById("page-title");

2 // Menyeleksi DOM berdasarkan Id element dan
  menampungnya ke dalam variabel. Isinya merupakan object
  HTML element

3

4 var pageBoxElements =
  document.getElementsByClassName("page-box");

5 // Menyeleksi DOM berdasarkan nama class element dan
  menampungnya ke dalam variabel. Isinya merupakan array
  dari object HTML element, walau <h1> hanya ada 1.

6

7 var pageHeadings =
  document.getElementsByTagName("h1");

8 // Menyeleksi DOM berdasarkan tag <h1> dan
  menampungnya ke dalam variabel. Isinya merupakan array
  dari object HTML element
```



Mengakses isi Elemen

Selanjutnya, kita akan melihat isi dari elemen yang sudah diakses, dengan menggunakan sintaks `innerHTML`.

```
1 var pageTitleElement =  
document.getElementById("page-title");  
  
2 // Menyeleksi DOM berdasarkan Id element dan  
menampungnya ke dalam variabel. Isinya merupakan object  
HTML element  
  
3  
4 var pageBoxElements =  
document.getElementsByClassName("page-box");  
  
5 // Menyeleksi DOM berdasarkan nama class element dan  
menampungnya ke dalam variabel. Isinya merupakan array  
dari object HTML element, walau <h1> hanya ada 1.  
  
6  
7 var pageHeadings =  
document.getElementsByTagName("h1");  
  
8 // Menyeleksi DOM berdasarkan tag <h1> dan  
menampungnya ke dalam variabel. Isinya merupakan array  
dari object HTML element
```

```
9  
10 var pageTitleElementsContent =  
pageTitleElement.innerHTML;  
  
11 console.log('isi <div id="page-title"> :' +  
pageTitleElementsContent);  
  
12 // isi <div id="page-title"> adalah Sample Page Title  
  
13  
14 var pageBoxElementsContent =  
pageBoxElements.innerHTML;  
  
15 console.log('isi <div class="page-box"> :' +  
pageBoxElementsContent);  
  
16 // isi <div class="page-box"> adalah undefined!
```

Karena isi adalah
Array, tidak bisa
langsung
menggunakan
innerHTML



Mengakses isi Elemen

Bila tipe isi elemen adalah array, maka kita perlu melakukan looping untuk melihat isinya. Contoh dibawah adalah melihat isi dari `pageBoxElements`.

```
14 for(var i = 0; i < pageBoxElements.length; i++) {  
  
15   var currentPageBoxElement      =  
   pageBoxElements[i];  
  
16   var currentPageBoxElementContent =  
   currentPageBoxElement.innerHTML;  
  
17  
  
18   // Mengambil isi elemen pageBoxElements yang kedua,  
   yaitu index ke 1  
  
19   var secondPageBoxElement        =  
   pageBoxElements[1];  
  
20   var secondpageBoxElementContent =  
   secondPageBoxElement.innerHTML;  
  
21  
  
22   // Mengambil isi elemen pageBoxElements yang ketiga,  
   yaitu index ke 2  
  
23   var thirdPageBoxElement         = pageBoxElements[2];  
  
24   var thirdpageBoxElementContent =  
   thirdPageBoxElement.innerHTML;
```

```
25  
  
26 // Menampilkan isi elemen dengan console.log  
  
27 console.log('isi <div class="page-box"> yang  
   pertama:' + firstpageBoxElementContent);  
  
28 console.log('isi <div class="page-box"> yang kedua:'  
   + secondpageBoxElementContent);  
  
29 console.log('isi <div class="page-box"> yang ketiga:'  
   + thirdpageBoxElementContent);
```

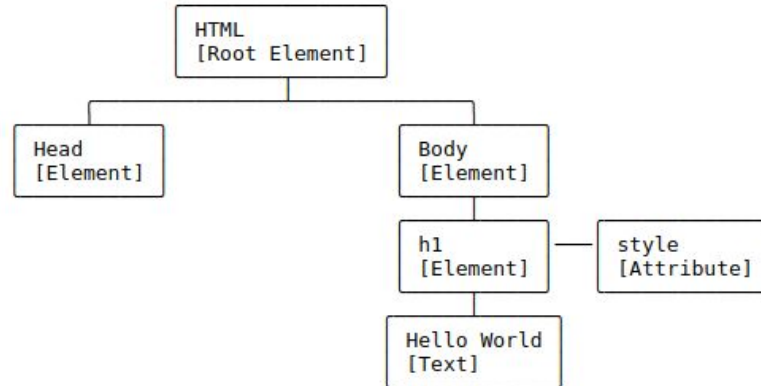


DOM Traversing

Ketika website kita telah selesai di load, maka DOM akan dimuat dan semua element yang ada di dalam html kita akan menjadi sebuah object, dimana objek-objek tersebut memiliki sekumpulan fungsi dan attribute yang dapat kita akses sehingga mempunyai full control terhadap element HTMLnya.

Berikut bagaimana DOM akan mendefinisikan setiap element dalam javascript akan dijadikan suatu hierarki Object yang nantinya dapat di akses atau bahkan bisa dimanipulasi untuk tujuan tertentu

```
<html>
  <head></head>
  <body>
    <h1 style="font-weight: bold;">
      Hello World
    </h1>
  </body>
</html>
```



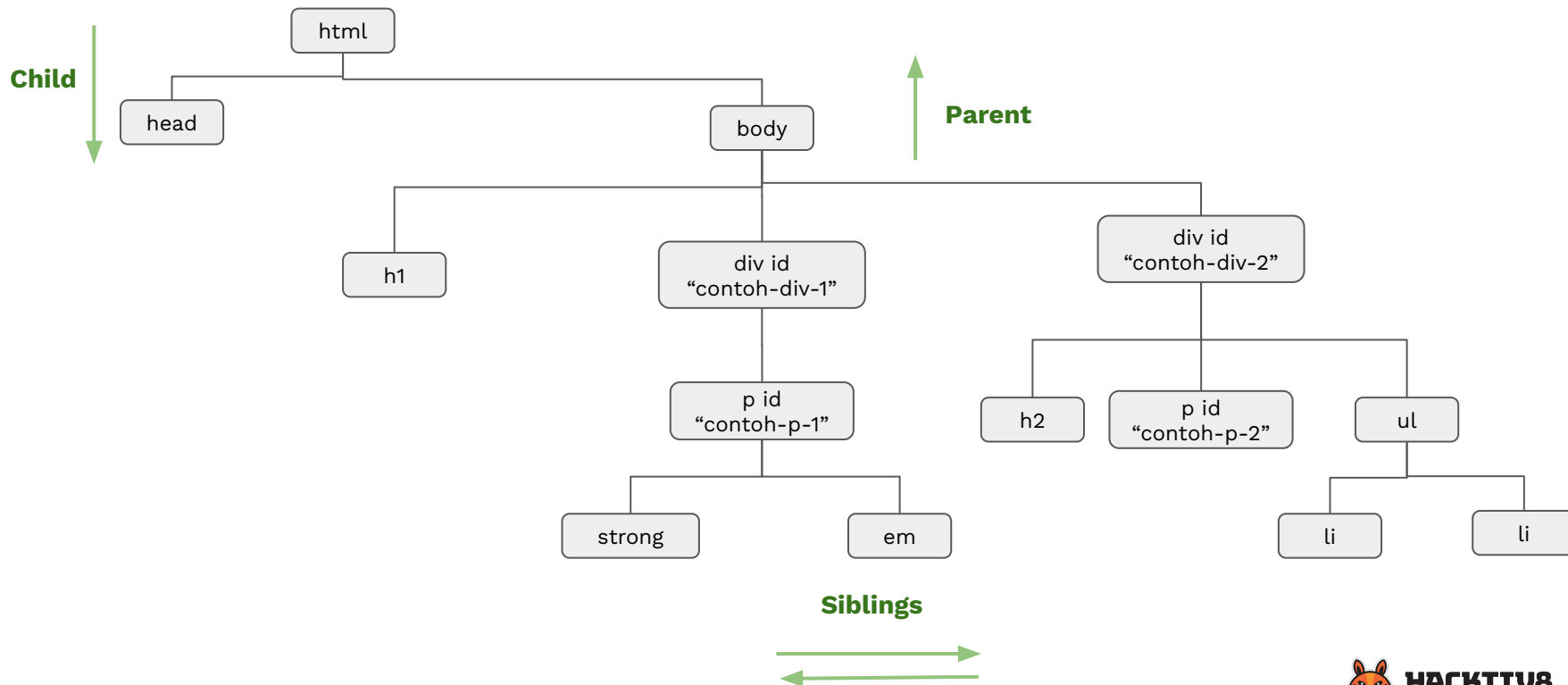
DOM Traversing

Sekarang, mari kita lihat hirarki dari html disamping.

```
<html>
  <head></head>
  <body>
    <h1></h1>
    <div id="contoh-div-1">
      <p id="contoh-p-1">
        <strong></strong>
        <em></em>
      </p>
    </div>
    <div id="contoh-div-2">
      <h2></h2>
      <p id="contoh-p-2"></p>
      <ul>
15      <li></li>
16      <li></li>
17      </ul>
18    </div>
19    <script src="dom-transverse-1-intro.js"></script>
20    <script
src="dom-transverse-2-siblings.js"></script>
21    <script
src="dom-transverse-3-chaining-selectors.js"></script>
22  </body>
23 </html>
```



DOM Traversing



DOM Traversing

html : merupakan parent paling atas

head : merupakan child dari html
body : merupakan child dari html, sibling dari head
h1 : merupakan child dari body

div id="contoh-div-1" : merupakan child dari body, sibling dari <h1>

p id="contoh-p-1" : merupakan child dari div id="contoh-div-1"

strong : merupakan child dari p id="contoh-p-1"

em : merupakan child dari p id="contoh-p-1", sibling dari strong

div id="contoh-div-2" : merupakan child dari body, sibling dari h1 dan div id="contoh-div-1"

h2 : merupakan child dari div id="contoh-div-2"
p id="contoh-p-2" : merupakan child dari

div id="contoh-div-2", sibling dari h2

ul : merupakan child dari div id="contoh-div-2", sibling dari h2 dan p id="contoh-p-2"

li : merupakan child dari ul



Seleksi DOM hubungan Parent - Child

```
7 // Menseleksi element <body>
8 var body = document.body;
9
10 // Mendapatkan element children dari <body>
11 var bodyChilds = body.children;
12
13 // Menampilkan DOM yang menjadi child dari <body>
    dalam bentuk array
14 console.log(bodyChilds); // h1, div
    id="contoh-div-1", div id="contoh-div-2", scripts js
15
16 // Menseleksi element <div id="contoh-div-1">
17 var contohDiv1 =
    document.getElementById('contoh-div-1');
18
19 // Mendapatkan element children dari <div
    id="contoh-div-1"> dalam bentuk array
20 var contohDiv1Childs = contohDiv1.children;
```

```
21
22 // Mendapatkan element children dari <div
    id="contoh-div-1"> dalam bentuk array
23 var contohDiv1Childs = contohDiv1.children;
24 console.log(contohDiv1FirstChild); // <p
    id="contoh-p-1">...</p>
25
26// Note: Walaupun children mungkin hanya 1 element,
    tetap tertampung dalam array!
```



Seleksi DOM hubungan Siblings

```
1 /*
2 =====
3 Menseleksi DOM berdasarkan hubungan Sibling
4 =====
5 */
6
7 // Menseleksi element <div id="contoh-div-1">
8 var contohDiv1 =
  document.getElementById('contoh-div-1');
9
10 // Mendapatkan sibling setelah <div
  id="contoh-div-1">
11 var contohDiv1NextSibling =
  contohDiv1.nextElementSibling;
```

```
12
13 console.log(contohDiv1NextSibling); // <div
  id="contoh-div-2">...</div>
14
15 // Mendapatkan sibling sebelum <div
  id="contoh-div-1">
16 var contohDiv1PrevSibling =
  contohDiv1.previousElementSibling;
17
18 console.log(contohDiv1PrevSibling); // <h1></h1>
```



Apa itu JQuery

jQuery adalah sebuah library JavaScript yang cepat, ringan, dan memiliki banyak fitur yang bermanfaat.

jQuery memudahkan kita untuk melakukan penjelajahan memanipulasi DOM, mengatur efek dan animasi, dan hal-hal keren lainnya, dan tentunya berfungsi di browser manapun.

Cara Menggunakan JQuery

```
<html>
  <head>
    <title>The jQuery Example</title>
  </head>

  <body>
    <div>
      <p class = "myclass">This is a paragraph.</p>
    </div>

    <script type = "text/javascript" src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js"></script>
  </body>
</html>
```

Agar dapat menggunakan fungsi-fungsi dalam JQuery, cantumkan cdn JQuery dalam tag `<script>` seperti contoh diatas.

CDN JQuery bisa didapatkan di: <https://code.jquery.com/>



Jquery - Selector

Untuk mengakses elemen dalam jquery menggunakan selector **\$()**

```
<html>
<head>
  <title>The jQuery Example</title>
</head>

<body>
  <div>
    <p class = "myclass">This is a paragraph.</p>
    <p id = "myid">This is second paragraph.</p>
    <p>This is third paragraph.</p>
  </div>

  <script type = "text/javascript"
    src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
  </script>

  <script type = "text/javascript" language = "javascript">
    $(document).ready(function() {
      $("p").css("background-color", "yellow");
    });
  </script>
</body>
</html>
```

1. Cara akses elemen dengan nama tag

Cara akses: **\$("nama_tag_elemen")**



Jquery - Selector

```
<html>
<head>
  <title>The jQuery Example</title>
</head>

<body>
  <div>
    <p class = "myclass">This is a paragraph.</p>
    <p id = "myid">This is second paragraph.</p>
    <p>This is third paragraph.</p>
  </div>

  <script type = "text/javascript"
    src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
  </script>

  <script type = "text/javascript" language = "javascript">
    $(document).ready(function() {
      $("#myid").css("background-color", "yellow");
    });
  </script>
</body>
</html>
```

2. Akses elemen dengan nama id.

Cara akses: `$("#nama_id_elemen")`

3. Akses elemen dengan nama class.

Cara akses: `$(".nama_class_elemen")`

```
<html>
<head>
  <title>The jQuery Example</title>
</head>

<body>
  <div>
    <p class = "myclass">This is a paragraph.</p>
    <p id = "myid">This is second paragraph.</p>
    <p>This is third paragraph.</p>
  </div>

  <script type = "text/javascript"
    src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
  </script>

  <script type = "text/javascript" language = "javascript">
    $(document).ready(function() {
      $(".myclass").css("background-color", "yellow");
    });
  </script>
</body>
</html>
```



Jquery Events

Event di HTML adalah "sesuatu" yang terjadi pada elemen HTML. Ketika kita menggunakan JavaScript di sebuah halaman HTML, JavaScript dapat "bereaksi" dan "merespon" terhadap event-event ini.

Event di HTML banyak jenisnya, namun yang sering digunakan adalah sebagai berikut:

- **click**, ketika user mengklik sebuah element HTML
- **mouseover**, ketika pointer mouse user masuk ke dalam area element HTML
- **mouseleave**, ketika pointer mouse user keluar dari area element HTML
- **keypress**, ketika user memencet tombol di keyboard
- **focus**, ketika user masuk ke dalam sebuah input (form)
- **blur**, ketika user keluar dari sebuah input (form)
- **dblclick**, ketika user melakukan double click sebuah element HTML



Jquery - Event Bubbling / Propagation

Di JavaScript, sebuah event untuk sebuah elemen, akan disampaikan juga kepada elemen yang mengandungnya atau element parent. Lalu dari parent ke parent-nya parent. Begitu seterusnya, hingga akhirnya sampai ke yang paling atas, yaitu document. Perilaku ini biasanya disebut sebagai event bubbling atau event propagation.

Hal ini, akan menyebabkan *overhead* apabila tidak dikontrol

```
1 // Attach a delegated event handler
2 $("#list").on("click", "a", function(event) {
3     event.preventDefault();
4     console.log($(this).text());
5 });
```

`event.preventDefault()`

Digunakan untuk
menghentikan event bubbling

