

DOM HTML

Dengan DOM, Javascript dapat mengakses semua elemen didalam dokumen HTML. DOM adalah interface yang bersifat netral terhadap platform dan bahasa yang membuat program dan script dapat mengakses secara dinamis dan mengupdate struktur, style, dan konten dokumen.

PENGERTIAN DOM

DOM atau *Document Object Model* adalah antarmuka yang tidak berpihak kepada platform dan bahasa tertentu yang membuat program dan script dapat mengakses dokumen HTML secara dinamis dan mengupdate struktur, konten, dan style dokumen.

DOM adalah standar dari W3C yang terdiri dari 3 bagian:

1. Core DOM, model standar untuk semua jenis dokumen
2. XML DOM, model standar untuk dokumen XML
3. HTML DOM, model standar untuk dokumen HTML

DOM HTML adalah model obyek standar dan antarmuka pemrograman untuk HTML. Model ini mendefinisikan:

- Elemen-elemen HTML sebagai obyek
- Properti seluruh elemen HTML
- Metode untuk mengakses seluruh elemen HTML
- Event untuk seluruh elemen HTML

Metode DOM HTML adalah tindakan yang dapat dilakukan terhadap elemen HTML dan **properti** adalah nilai dari elemen HTML yang dapat ditentukan atau diubah.

DOM HTML menggunakan bahasa pemrograman untuk mengakses obyek-obyeknya, biasanya Javascript. Semua elemen HTML diperlakukan sebagai obyek. Antarmuka pemrogramannya adalah metode dan properti dari setiap obyek.

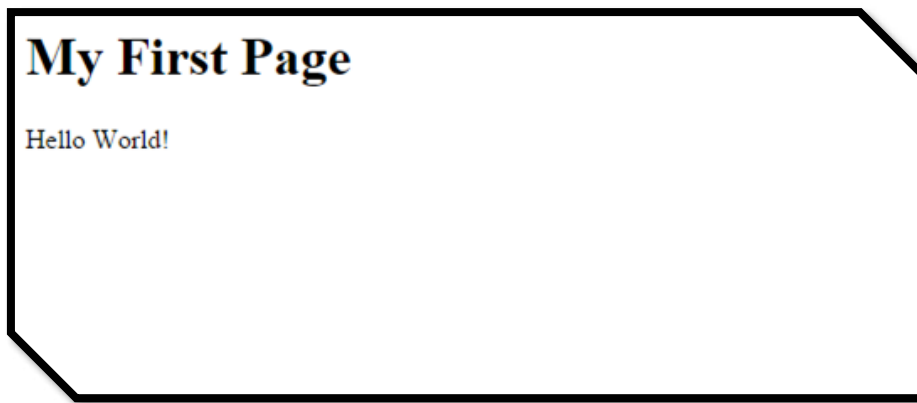
Contoh berikut akan mengubah konten dari elemen paragraf dengan id="demo" menggunakan properti innerHTML:

```
<html>
<body>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = "Hello World!";
</script>

</body>
</html>
```



Dalam contoh diatas, **getElementById** (perhatikan penulisannya) adalah sebuah metode dan innerHTML adalah **properti**.

Cara yang biasa digunakan untuk mengakses suatu elemen HTML adalah menggunakan atribut id dari elemen. Contoh diatas metode getElementById menggunakan id="demo" untuk menemukan elemen paragraf.

Sementara cara yang paling mudah untuk mendapatkan konten suatu elemen adalah menggunakan properti innerHTML. Properti ini dapat digunakan untuk memperoleh atau mengubah setiap elemen HTML termasuk <html> dan <body>.

OBJEK DOCUMENT

Dalam DOM HTML, terdapat suatu objek yang memiliki semua objek lainnya. Objek ini adalah document. Objek ini mewakili halaman web. Artinya, jika akan mengakses elemen-elemen HTML didalam halaman web, maka harus melewati objek document ini.

Berikutnya adalah daftar contoh penerapan objek document untuk mengakses dan memanipulasi HTML.

Mencari Elemen HTML

Metode	Penjelasan
<code>document.getElementById()</code>	mencari elemen HTML menggunakan elemen id
<code>document.getElementsByTagName()</code>	mencari elemen HTML menggunakan nama tag
<code>document.getElementsByClassName()</code>	mencari elemen HTML menggunakan nama kelas

Mengubah Elemen HTML

Metode	Penjelasan
<code>elemen.innerHTML=</code>	mengubah inner HTML elemen HTML
<code>elemen.atribut=</code>	mengubah atribut elemen HTML
<code>elemen.setAttribute(atribut, nilai)</code>	mengubah atribut elemen HTML
<code>elemen.style.property=</code>	mengubah style elemen HTML

Menambah dan Menghapus Elemen

Metode	Penjelasan
<code>document.createElement()</code>	membuat elemen HTML
<code>document.removeChild()</code>	menghapus elemen HTML
<code>document.appendChild()</code>	menambah sebuah elemen HTML
<code>document.replaceChild()</code>	mengganti elemen HTML
<code>document.write(teks)</code>	menulis teks ke layar

Untuk menambahkan event handler perintah yang digunakan:

```
document.getElementById(id).onclick=function(){kode}
```

Perintah diatas akan menambahkan event handler "kode" ke event onclick

Berikut ini adalah daftar objek yang telah didefinisikan sejak DOM HTML level 1 sampai level 3 yang masih digunakan.

Method	Description	DOM
document.anchors	Returns all <a> with a value in the name attribute	1
document.applets	Returns all <applet> elements (Deprecated in HTML5)	1
document.baseURI	Returns the absolute base URI of the document	3
document.body	Returns the <body> element	1
document.cookie	Returns the document's cookie	1
document.doctype	Returns the document's doctype	3
document.documentElement	Returns the <html> element	3
document.documentMode	Returns the mode used by the browser	3
document.documentURI	Returns the URI of the document	3
document.domain	Returns the domain name of the document server	1
document.domConfig	Returns the DOM configuration	3
document.embeds	Returns all <embed> elements	3
document.forms	Returns all <form> elements	1
document.head	Returns the <head> element	3
document.images	Returns all <image> elements	1
document.implementation	Returns the DOM implementation	3
document.inputEncoding	Returns the document's encoding (character set)	3
document.lastModified	Returns the date and time the document was updated	3
document.links	Returns all <area> and <a> elements value in href	1
document.readyState	Returns the (loading) status of the document	3
document.referrer	Returns the URI of the referrer (the linking document)	1
document.scripts	Returns all <script> elements	3
document.strictErrorChecking	Returns if error checking is enforced	3
document.title	Returns the <title> element	1
document.URL	Returns the complete URL of the document	1

MENCARI ELEMEN

Untuk mencari elemen HTML yang diinginkan, ada beberapa cara:

1. Mencari elemen HTML berdasarkan id
2. Mencari elemen HTML berdasarkan nama tag
3. Mencari elemen HTML berdasarkan nama kelas
4. Mencari elemen HTML berdasarkan kumpulan obyek HTML

MENCARI DENGAN ID

Adalah cara termudah, contoh ini akan mencari elemen HTML dengan **id="intro"** lalu mengambil konten dan menyimpannya dalam variabel **myElement**. Setelah itu menampilkan konten tersebut dalam elemen paragraf dengan **id="demo"**.

```
<!DOCTYPE html>
<html>
<body>

<p id="intro">Hello World!</p>
<p>This example demonstrates the <b>getElementById</b> method!</p>
<p id="demo"></p>

<script>
myElement = document.getElementById("intro");
document.getElementById("demo").innerHTML =
"The text from the intro paragraph is " + myElement.innerHTML;
</script>

</body>
</html>
```

Hello World!

This example demonstrates the `getElementById` method!

The text from the intro paragraph is Hello World!

MENCARI DENGAN NAMA TAG

Contoh dibawah ini akan mencari elemen dengan `id="main"` yang disimpan kedalam variabel `x` lalu mencari **seluruh elemen paragraf** dan disimpan kedalam variabel `y`. Setelah itu, isi dari elemen paragraf yang pertama akan ditampilkan dalam elemen paragraf dengan `id="demo"`.

```
<!DOCTYPE html>
<html>
<body>

<p>Hello World!</p>

<div id="main">
<p>The DOM is very useful.</p>
<p>This example demonstrates the <b>getElementsByTagName</b> method</p>
</div>
<p id="demo"></p>

<script>
var x = document.getElementById("main");
var y = x.getElementsByTagName("p");
document.getElementById("demo").innerHTML =
'The first paragraph inside "main" is ' + y[0].innerHTML;
</script>

</body>
</html>
```

Hello World!

The DOM is very useful.

This example demonstrates the `getElementsByTagName` method

The first paragraph inside "main" is The DOM is very useful.

MENCARI DENGAN NAMA KELAS

Untuk mencari elemen menggunakan nama kelas, berikut ini kodenya yang akan mencari seluruh kelas dengan nama **"intro"**.

```
document.getElementsByClassName("intro");
```

MENCARI BERDASAR KUMPULAN OBYEK

Dalam contoh berikut ini akan diambil nilai-nilai yang dimasukkan kedalam dua inputan form (fname dan lname). Pertama akan mencari elemen form (frm1) dahulu, setelah itu dengan menggunakan perulangan akan diambil satu persatu nilai yang dimasukkan kedalam masing-masing inputan.

```
<!DOCTYPE html>
<html>
<body>

<form id="frm1" action="form_action.asp">
First name: <input type="text" name="fname" value="Pemrograman"><br>
Last name: <input type="text" name="lname" value="Web"><br><br>
<input type="submit" value="Submit">
</form>

<p>Click "Try it" to display the value of each element in the form.</p>

<button onclick="myFunction()">Try it</button>

<p id="demo"></p>

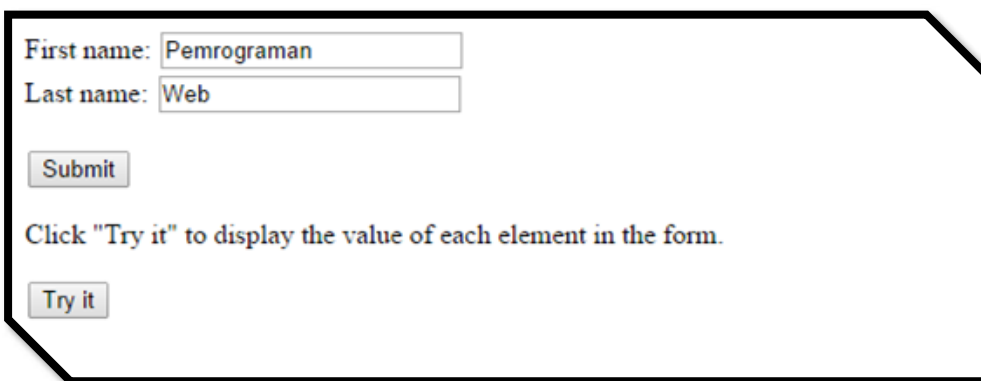
<script>
function myFunction() {
  var x = document.getElementById("frm1");
  var text = "";
```


Lanjutan kode...

```
var i;
for (i = 0; i < x.length ;i++) {
    text += x.elements[i].value + "<br>";
}
document.getElementById("demo").innerHTML = text;
}
</script>

</body>
</html>
```

Jika contoh diatas ditampilkan ke browser maka akan tampak:

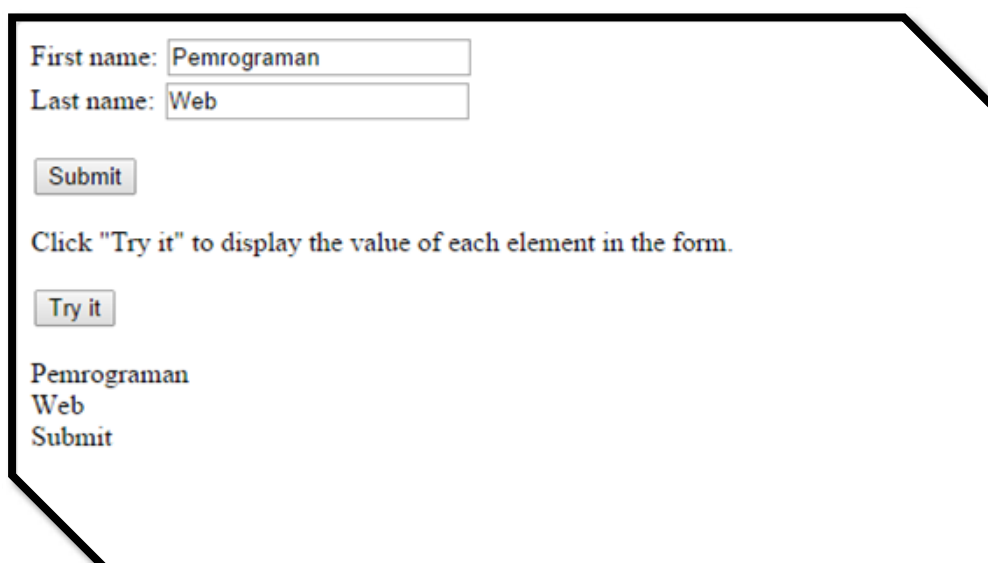


First name:

Last name:

Click "Try it" to display the value of each element in the form.

Jika kemudian tombol submit diklik maka hasilnya:



First name:

Last name:

Click "Try it" to display the value of each element in the form.

Pemrograman
Web
Submit

Selain obyek-obyek HTML diatas, Obyek-obyek dan kumpulan obyek HTML berikut ini juga dapat diakses.

ANCHORS

Code:

```
<!DOCTYPE html>
<html>
<body>

<a name="html">HTML Tutorial</a><br>
<a name="css">CSS Tutorial</a><br>
<a name="xml">XML Tutorial</a><br>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
"Number of anchors are: " + document.anchors.length;
</script>

</body>
</html>
```

Result:

```
HTML Tutorial
CSS Tutorial
XML Tutorial

Number of anchors are: 3
```

BODY

Laman di www.w3schools.com menyatakan:

```
<p id="demo"></p>
```

```
<script>alert(document.body.innerHTML);</script>
```

Oke

Edit the code and click "Submit" to see the result.

Code:

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
alert(document.body.innerHTML);
</script>

</body>
</html>
```

DOCUMENTELEMENT

Laman di www.w3schools.com menyatakan:

```
<head></head><body>
```

```
<p id="demo"></p>
```

```
<script>alert(document.documentElement.innerHTML);</script></body>
```

Oke

Edit the code and click "Submit" to see the result.

Code:

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
alert(document.documentElement.innerHTML);
</script>

</body>
</html>
```

EMBEDS

Code:

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
"Number of embeds: " + document.embeds.length;
</script>

</body>
</html>
```

Result:

Number of embeds: 0

FORMS

Code:

```
<!DOCTYPE html>
<html>
<body>

<form action="">
First name: <input type="text" name="fname" value="Donald">
<input type="submit" value="Submit">
</form>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
"Number of forms: " + document.forms.length;
</script>

</body>
</html>
```

Result:

First name:

Number of forms: 1

IMAGES

Code:

```
<!DOCTYPE html>
<html>
<body>

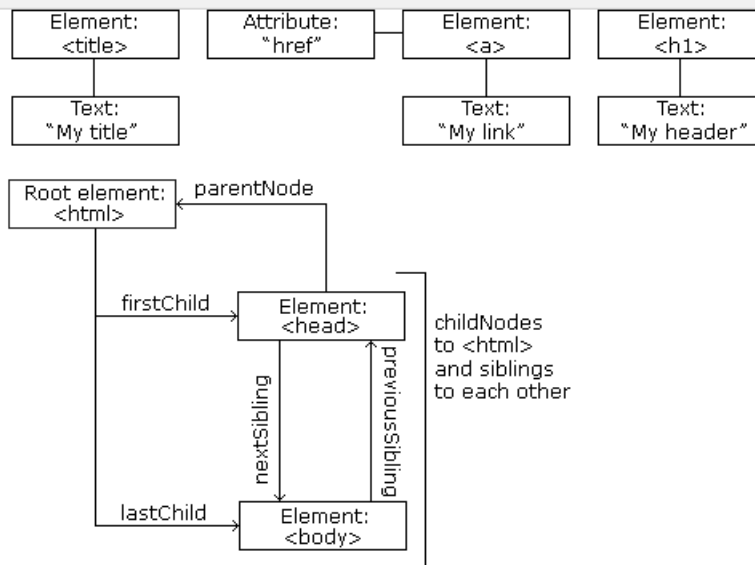



<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
"Number of images: " + document.images.length;
</script>

</body>
</html>
```

Result:



Number of images: 2

LINKS

Code:

```
<!DOCTYPE html>
<html>
<body>

<p>
<a href="/html/default.asp">HTML</a>
<br>
<a href="/css/default.asp">CSS</a>
</p>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
"Number of links: " + document.links.length;
</script>

</body>
</html>
```

Result:

[HTML](#)
[CSS](#)

Number of links: 2

SCRIPTS

Code:

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
"Number of scripts: " + document.scripts.length;
</script>

</body>
</html>
```

Result:

Number of scripts: 1

Code:

```
<!DOCTYPE html>
<html>
  <head>
    <title>W3Schools Demo</title>
  </head>
  <body>

    <p id="demo"></p>

    <script>
    document.getElementById("demo").innerHTML =
    "The title of this document is: " + document.title;
    </script>

  </body>
</html>
```

Result:

The title of this document is: W3Schools Demo

MENGUBAH HTML

MENGUBAH KONTEN ELEMEN

Cara untuk mengubah konten elemen HTML yang paling mudah adalah menggunakan properti **innerHTML**.

document.getElementById(id).innerHTML=HTML baru.

Contoh berikut mengubah konten dari elemen paragraf:

```
<!DOCTYPE html>
<html>
<body>

<p id="p1">Hello World!</p>

<script>
document.getElementById("p1").innerHTML = "New text!";
</script>

<p>The paragraph above was changed by a script.</p>

</body>
</html>
```

New text!
The paragraph above was changed by a script.

MENGUBAH NILAI ATRIBUT

Untuk mengubah nilai atribut elemen HTML, sintaksnya:

```
document.getElementById(id).attribute=nilai baru
```

Contoh berikut akan mengubah nilai dari atribut src elemen image:

```
<!DOCTYPE html>
<html>
<body>



<script>
document.getElementById("image").src = "landscape.jpg";
</script>

<p>The original image was smiley.gif, but the script changed it to landscape.jpg</p>

</body>
</html>
```



The original image was smiley.gif, but the script changed it to landscape.jpg

MENGUBAH CSS

Dalam DOM HTML, style dari elemen HTML juga dapat diubah. Sintaksnya:

document.getElementById(id).style.property=style baru

Contoh berikut akan mengubah style dari elemen paragraf:

```
<!DOCTYPE html>
<html>
<body>

<p id="p1">Hello World!</p>
<p id="p2">Hello World!</p>

<script>
document.getElementById("p2").style.color = "blue";
document.getElementById("p2").style.fontFamily = "Arial";
document.getElementById("p2").style.fontSize = "larger";
</script>

<p>The paragraph above was changed by a script.</p>

</body>
</html>
```

Hello World!

Hello World!

The paragraph above was changed by a script.

EVENTS

DOM HTML memungkinkan javascript untuk bereaksi terhadap event HTML. Sebuah skrip javascript dapat dieksekusi ketika suatu event terjadi, misalnya ketika seorang pengguna melakukan klik terhadap suatu elemen HTML.

Untuk mengeksekusi suatu kode ketika pengguna melakukan klik terhadap suatu elemen, tambahkan skrip javascript ke atribut event elemen HTML:

onclick=javascript

Event yang dapat terjadi:

- ✓ Ketika pengguna melakukan klik mouse
- ✓ Ketika sebuah halaman web telah dimuat (load)
- ✓ Ketika sebuah gambar telah dimuat
- ✓ Ketika mouse melintasi sebuah elemen
- ✓ Ketika form HTML disubmit
- ✓ Ketika pengguna menekan tombol keyboard

Pada contoh dibawah ini, konten elemen heading tingkat satu akan berubah ketika pengguna melakukan klik terhadapnya.

```
<!DOCTYPE html>
<html>
<body>
<h1 onclick="this.innerHTML='Oops!'">Click on this text!</h1>
</body>
</html>
```

ATAU

```
<!DOCTYPE html>
<html>
<body>
<h1 onclick="changeText(this)">Click on this text!</h1>
<script>
function changeText(id) {
  id.innerHTML = "Oops!";
}
</script>
</body>
</html>
```

Click on this text!

Ketika pengguna melakukan klik terhadap teks di layar, maka teks akan berubah:

Ooops!

MENUGASI EVENT DENGAN JAVASCRIPT

Event juga dapat ditugaskan dengan menggunakan skrip javascript. Contoh dibawah ini menugasi event click menggunakan `document.getElementById("myBtn").onclick=function(){displayDate()}`

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>

<p>Click "Try it" to execute the displayDate() function.</p>

<button id="myBtn">Try it</button>

<p id="demo"></p>

<script>
document.getElementById("myBtn").onclick = function(){displayDate()};

function displayDate() {
  document.getElementById("demo").innerHTML = Date();
}
</script>
```

Click "Try it" to execute the `displayDate()` function.

Try it

Click "Try it" to execute the `displayDate()` function.

Try it

Wed Sep 24 2014 13:19:23 GMT+0700 (SE Asia Standard Time)

EVENT ONLOAD DAN ONUNLOAD

Dua event ini akan dipicu ketika pengguna masuk atau keluar dari halaman web. Event onload dapat digunakan untuk memeriksa jenis dan versi browser pengguna yang kemudian berdasarkan informasi itu dapat menentukan versi halaman web yang tepat untuk ditampilkan kepada pengguna tersebut.

Contoh dibawah ini akan menjalankan sebuah skrip javascript yang akan memeriksa dukungan browser pengguna terhadap cookies.

```
<!DOCTYPE html>
<html>
<body onload="checkCookies()">

<p id="demo"></p>

<script>
function checkCookies() {
  var text = "";
  if (navigator.cookieEnabled == true) {
    text = "Cookies are enabled.";
  } else {
    text = "Cookies are not enabled.";
  }
  document.getElementById("demo").innerHTML = text;
}
</script>

</body>

</html>
```

Cookies are enabled.

EVENT ONCHANGE

Event ini biasanya digunakan bersama dengan validasi field inputan. Berikut contoh yang akan menunjukkan bagaimana fungsi javascript upperCase() akan dieksekusi ketika pengguna mengubah isi dari field inputan


```
<!DOCTYPE html>
<html>
<head>
<script>
function myFunction() {
  var x = document.getElementById("fname");
  x.value = x.value.toUpperCase();
}
</script>
</head>
<body>

Enter your name: <input type="text" id="fname" onchange="myFunction()">
<p>When you leave the input field, a function is triggered which transforms the input text to
upper case.</p>

</body>
</html>
```

Enter your name:

When you leave the input field, a function is triggered which transforms the input text to upper case.

EVENT ONMOUSEOVER DAN ONMOUSEOUT

Event onmouseover akan terjadi ketika mouse pengguna melintasi sebuah elemen HTML dan event onmouseout akan terjadi ketika mouse pengguna keluar dari sebuah elemen HTML. Contoh:

```
<!DOCTYPE html>
<html>
<body>

<div onmouseover="mOver(this)" onmouseout="mOut(this)"
style="background-color:#D94A38;width:120px;height:20px;padding:40px;">
Mouse Over Me</div>

<script>
function mOver(obj) {
  obj.innerHTML = "Thank You"
}

function mOut(obj) {
  obj.innerHTML = "Mouse Over Me"
}
</script>

</body>
</html>
```

Selama mouse melintasi elemen maka teks yang tampil adalah **Thank You**, tetapi jika mouse keluar dari area elemen maka teks akan berubah menjadi **Mouse Over Me**.



ONMOUSEDOWN, ONMOUSEUP DAN ONCLICK

Ketiga event ini bagian dari kejadian klik mouse. Event onmousedown akan dipicu ketika tombol mouse diklik. Lalu ketika tombol mouse dilepaskan maka event onmouseup akan terpicu. Akhirnya, ketika proses klik mouse ini selesai maka event onclick yang akan dipicu.

```
<!DOCTYPE html>
<html>
<body>

<div onmousedown="mDown(this)" onmouseup="mUp(this)"
style="background-color:#D94A38;width:90px;height:20px;padding:40px;">
Click Me</div>

<script>
function mDown(obj) {
  obj.style.backgroundColor = "#1ec5e5";
  obj.innerHTML = "Release Me";
}

function mUp(obj) {
  obj.style.backgroundColor="#D94A38";
  obj.innerHTML="Thank You";
}
</script>

</body>
</html>
```

Dalam contoh, event onmouse down akan menjalankan fungsi **mDown()** dan event onmouseup akan menjalankan fungsi **mUp()**. Silahkan kode diatas dicoba untuk mengetahui hasilnya.

EVENT LISTENER

METODE ADDEVENTLISTENER()

Eventlistener adalah suatu event yang diikatkan pada suatu elemen HTML. Untuk mengikatkan suatu event pada elemen HTML menggunakan metode `addEventListener()`. Contoh:

```
<!DOCTYPE html>
<html>
<body>

<p>This example uses the addEventListener() method to attach a click event to a button.</p>
<button id="myBtn">Try it</button>
<p id="demo"></p>

<script>
document.getElementById("myBtn").addEventListener("click", displayDate);
function displayDate() {
    document.getElementById("demo").innerHTML = Date();
}
</script>

</body>
</html>
```

Dalam contoh, tombol dengan id “myBtn” disematkan event handler click yang akan menjalankan fungsi `displayDate()` jika terpicu.

Metode ini tidak akan menindih event handler yang sudah ada pada elemen. Dengan demikian kita dapat mengikatkan banyak event handler pada sebuah elemen HTML. Bahkan kita dapat mengikat lebih dari satu event handler yang sama ke sebuah elemen HTML. Obyek yang dapat diikatkan dengan metode ini tidak hanya obyek elemen HTML saja, misalnya obyek window.

Ketika menggunakan metode ini lebih baik jika kode javascript dipisah dari kode HTML untuk memudahkan menambah event listener. Untuk menghapus event listener menggunakan metode `removeEventListener()`.

Sintaks penggunaan metode ini:

elemen.addEventListener(event, function, useCapture)

- Parameter pertama adalah jenis event yang disematkan
- Parameter kedua adalah fungsi yang akan dijalankan jika event terjadi.

- Parameter yang ketiga bersifat opsional berupa nilai boolean yang menentukan apakah mau menggunakan event bubbling atau event capturing.

Contoh-contoh lain:

```
<!DOCTYPE html>
<html>
<body>

<p>This example uses the addEventListener() method to attach a click event to a button.</p>
<button id="myBtn">Try it</button>

<script>
document.getElementById("myBtn").addEventListener("click", function(){
  alert("Hello World!");
});
</script>

</body>
</html>
```

```
<!DOCTYPE html>
<html>
<body>

<p>This example uses the addEventListener() method to execute a function when a user clicks
on a button.</p>

<button id="myBtn">Try it</button>

<script>
document.getElementById("myBtn").addEventListener("click", myFunction);

function myFunction() {
  alert ("Hello World!");
}
</script>

</body>
</html>
```

MENAMBAHKAN BANYAK EVENT HANDLER KE ELEMEN YANG SAMA

Contoh dibawah ini menambahkan event click, mouseover, dan mouseout pada tombol myBtn.

```
<!DOCTYPE html>
<html>
<body>

<p>This example uses the addEventListener() method to add many events on the same
button.</p>

<button id="myBtn">Try it</button>

<p id="demo"></p>

<script>
var x = document.getElementById("myBtn");
x.addEventListener("mouseover", myFunction);
x.addEventListener("click", mySecondFunction);
x.addEventListener("mouseout", myThirdFunction);

function myFunction() {
    document.getElementById("demo").innerHTML += "Moused over!<br>"
}

function mySecondFunction() {
    document.getElementById("demo").innerHTML += "Clicked!<br>"
}

function myThirdFunction() {
    document.getElementById("demo").innerHTML += "Moused out!<br>"
}
</script>

</body>
</html>
```

Result:

This example uses the `addEventListener()` method to add many events on the same button.

Try it

Moused over!
Moused out!
Moused over!
Moused out!
Moused over!
Clicked!
Clicked!
Clicked!
Moused out!

Contoh diatas akan menampilkan teks "Moused over" ketika mouse melintasi tombol try it, menampilkan teks "Moused out" jika mouse keluar dari area tombol dan teks "Clicked!" saat mouse diklik.

MENAMBAHKAN EVENT HANDLER KE OBYEK WINDOW

Metode `addEventListener` dapat digunakan untuk segala obyek, elemen HTML, dokumen HTML, obyek window, atau obyek-obyek lain yang mendukung event, seperti obyek `xmlHttpRequest`.

Contoh berikut akan menampilkan angka random jika jendela browser berubah ukurannya.

```

<!DOCTYPE html>
<html>
<body>

<p>This example uses the addEventListener() method on the window object.</p>

<p>Try resizing this browser window to trigger the "resize" event handler.</p>

<p id="demo"></p>

<script>
window.addEventListener("resize", function(){
    document.getElementById("demo").innerHTML = Math.random();
});
</script>

</body>
</html>

```

The screenshot shows a web browser window with the URL `www.w3schools.com/js/tryit.asp?filename=tryjs_addeventlistener_dom`. The page features a yellow and blue banner for 'Harga Murah Tiket Pesawat' (Cheap Flight Tickets) with a search form. Below the banner, there is a 'Submit' button and a 'w3schools.com' logo. The main content area is divided into two sections: 'Code' and 'Result'. The 'Code' section contains the HTML and JavaScript code from the previous block. The 'Result' section shows the output of the JavaScript code, which is a random number: `0.7018989322241396`.

MELEWATKAN PARAMETER

Untuk melewatkan parameter pada penggunaan Event Listener ini, gunakan **fungsi anonymus** yang memanggil fungsi tertentu yang memiliki parameter.

```
<!DOCTYPE html>
<html>
<body>

<p>This example demonstrates how to pass parameter values when using the
addEventListener() method.</p>

<p>Click the button to perform a calculation.</p>

<button id="myBtn">Try it</button>

<p id="demo"></p>

<script>
var p1 = 5;
var p2 = 7;

document.getElementById("myBtn").addEventListener("click", function() {
  myFunction(p1, p2);
  });

function myFunction(a, b) {
  var result = a * b;
  document.getElementById("demo").innerHTML = result;
}
</script>

</body>
</html>
```

Result:

This example demonstrates how to pass parameter values when using the `addEventListener()` method.

Click the button to perform a calculation.

Try it

35

Jika tombol **Try it** ditekan, maka akan tampil hasil perhitungannya.

EVENT BUBBLING DAN EVENT CAPTURING

Jika Anda memiliki sebuah elemen paragraf didalam elemen `<div>` yang masing-masing memiliki event handler click lalu pengguna melakukan klik terhadap elemen paragraf, event click dari elemen mana yang akan dieksekusi terlebih dahulu? Jawabannya tergantung dari jenis propagasi event yang digunakan metode `addEventListener()`.

Propagasi event adalah suatu cara untuk menentukan urutan kejadian suatu event. Ada dua jenis propagasi, yaitu bubbling dan capturing. Dalam jenis **bubbling**, event dari element yang paling dalamlah yang akan dieksekusi terlebih dahulu. Jadi urutannya adalah event click dari elemen paragraf yang akan dijalankan terlebih dahulu baru event elemen `<div>`. Sedangkan jenis **capturing** urutan kejadian event sebaliknya, event dari elemen yang terluar yang akan dijalankan terlebih dahulu.

Untuk menentukan jenis propagasi menggunakan parameter ketiga dari metode `addEventListener()`:

`addEventListener(event, function, useCapture)`

Defaultnya adalah false yang akan menggunakan propagasi bubbling.

Contoh berikut memperagakan penggunaan kedua jenis propagasi. Kombinasi elemen paragraf dan `<div>` yang pertama ditentukan berjenis propagasi bubbling sedang yang kedua berjenis capturing.

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  background-color: coral;
  border: 1px solid;
  padding: 50px;
}
</style>
</head>
<body>

<p>This example demonstrates the difference between bubbling and capturing when adding event
listeners.</p>

<div id="myDiv">
  <p id="myP">Click this paragraph, I am Bubbling.</p>
</div><br>

<div id="myDiv2">
  <p id="myP2">Click this paragraph, I am Capturing.</p>
</div>

<script>
document.getElementById("myP").addEventListener("click", function() {
  alert("You clicked the P element!");
}, false);

document.getElementById("myDiv").addEventListener("click", function() {
  alert("You clicked the DIV element!");
}, false);

document.getElementById("myP2").addEventListener("click", function() {
  alert("You clicked the P element!");
}, true);

document.getElementById("myDiv2").addEventListener("click", function() {
  alert("You clicked the DIV element!");
}, true);
</script>

</body>
</html>
```

Result:

This example demonstrates the difference between bubbling and capturing when adding event listeners.

Click this paragraph, I am Bubbling.

Click this paragraph, I am Capturing.

METODE REMOVEEVENTLISTENER()

Metode ini akan menghapus event handler yang telah disematkan dengan metode `addEventListener()`

Contoh:

Jika tombol **Try it** diklik, maka penampilan angka random secara acak akan berhenti

Result:

This div element has an `onmousemove` event handler that displays a random number every time you move your mouse inside this orange field.

Click the button to remove the DIV's event handler.

Try it

0.37126118200831115

```
<!DOCTYPE html>
<html>
<head>
<style>
#myDIV {
  background-color: coral;
  border: 1px solid;
  padding: 50px;
  color: white;
}
</style>
</head>
<body>

<div id="myDIV">This div element has an onmousemove event handler that displays a random
number every time you move your mouse inside this orange field.
  <p>Click the button to remove the DIV's event handler.</p>
  <button onclick="removeHandler()" id="myBtn">Try it</button>
</div>

<p id="demo"></p>

<script>
document.getElementById("myDIV").addEventListener("mousemove", myFunction);

function myFunction() {
  document.getElementById("demo").innerHTML = Math.random();
}

function removeHandler() {
  document.getElementById("myDIV").removeEventListener("mousemove", myFunction);
}
</script>

</body>
</html>
```

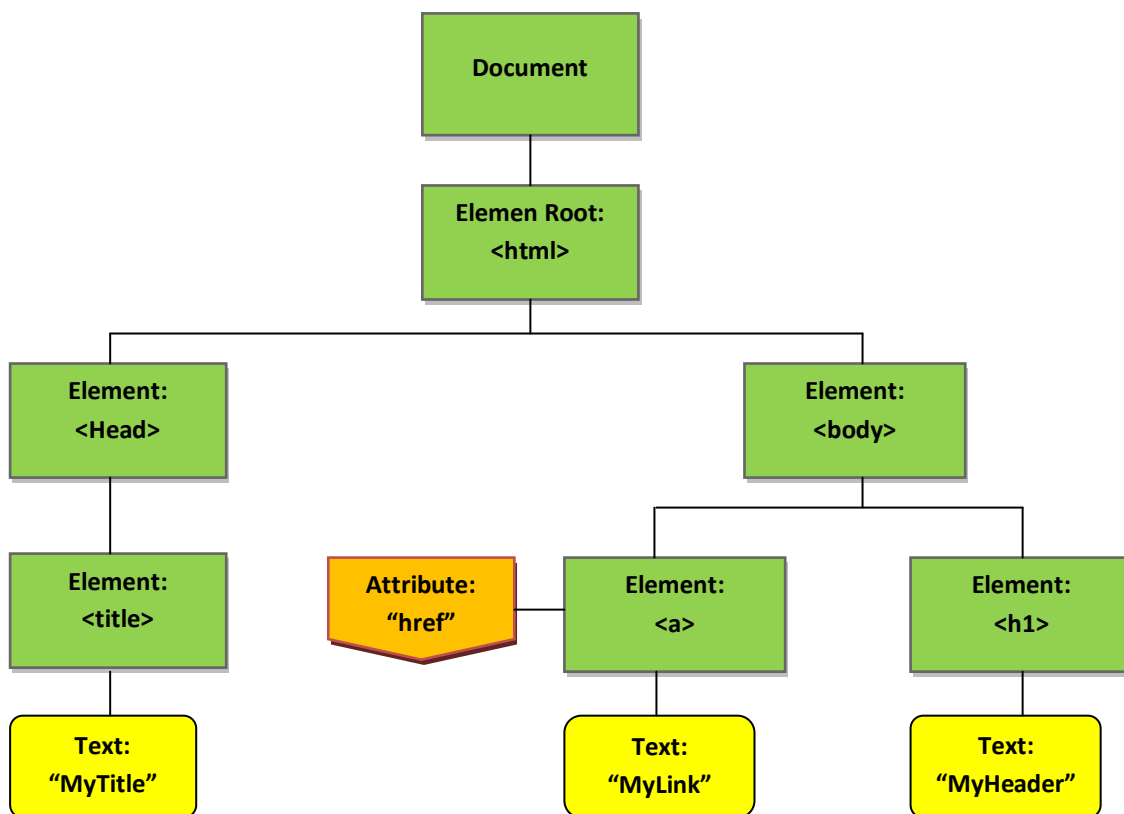
NAVIGASI DOM

Dengan menggunakan DOM HTML, kita dapat menyusuri pohon simpul (node tree) menggunakan hubungan diantara simpul-simpulnya.

SIMPUL DOM

Berdasarkan standar DOM HTML W3C, segala sesuatu didalam sebuah halaman web adalah simpul:

- ✓ Keseluruhan dokumen adalah simpul document
- ✓ Setiap elemen HTML adalah simpul element
- ✓ Teks didalam elemen HTML adalah simpul text
- ✓ Setiap atribut HTML adalah sebuah simpul atribut
- ✓ Semua komentar adalah simpul comment

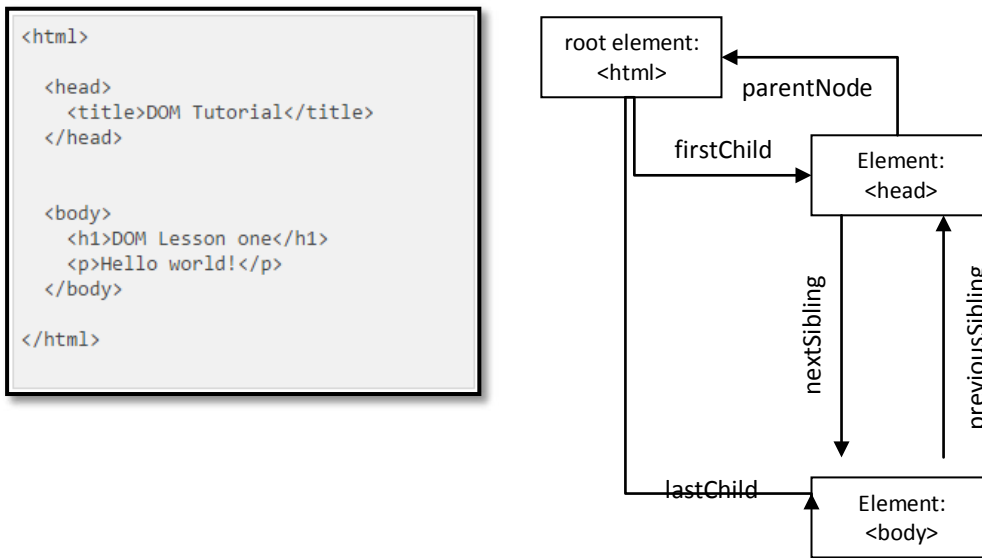


Gambar Pohon Simpul

Semua simpul didalam pohon simpul ini dapat diakses menggunakan javascript. Simpul baru dapat dibuat dan semua simpul dapat diubah atau dihapus.

Simpul-simpul dalam pohon simpul memiliki hubungan hirarki satu sama lain. Istilah orang tua (*parent*), anak (*child*), dan saudara (*sibling*) digunakan untuk menggambarkan hubungan.

- ✓ Dalam pohon simpul, simpul atas disebut akar (root) atau simpul akar (root node)
- ✓ Setiap simpul memiliki tepat satu simpul orangtua, kecuali simpul akar (yang tidak memiliki orang tua)
- ✓ Sebuah simpul dapat memiliki sejumlah simpul anak-anak
- ✓ Simpul saudara (kakak dan adik) memiliki simpul orang tua yang sama



Dari dokumen HTML diatas dapat dibaca bahwa:

- ✓ <html> adalah simpul akar, tidak berorang tua, orang tua dari <head> dan <body>
- ✓ <head> adalah anak pertama dari <html> dan mempunyai satu anak yaitu <title>
- ✓ <body> adalah anak terakhir dari <html> dan punya dua anak yaitu <h1> dan <p>
- ✓ <title> mempunyai satu anak simpul text yaitu "DOM Tutorial"
- ✓ <h1> mempunyai satu anak simpul text yaitu "DOM Lesson one"
- ✓ <p> mempunyai satu anak simpul text yaitu "Hello world!"
- ✓ <p> dan <h1> bersaudara

PENELUSURAN DIANTARA SIMPUL

Properti-properti simpul dibawah ini dapat digunakan untuk menyusuri simpul-simpul dengan javascript:

- parentNode
- childNodes[nodenumber]
- firstChild
- lastChild
- nextSibling
- previousSibling

PENTING

Kesalahan yang paling sering dilakukan adalah menganggap bahwa simpul elemen berisi teks. Hal ini tidak benar! Dalam contoh diatas simpul elemen “<title>DOM Tutorial</title>” tidak berisi teks tetapi berisi simpul text dengan nilai “DOM Tutorial”.

Nilai dari simpul text dapat diakses menggunakan properti **innerHTML** dari simpul text tersebut atau properti **nodeValue**.

PROPERTI CHILDNODES DAN NODEVALUE

Selain menggunakan properti innerHTML, kita juga dapat menggunakan properti childNodes dan nodeValue untuk mendapatkan konten suatu elemen. Contoh berikut mengumpulkan nilai simpul dari sebuah elemen <h1> dan menggandakannya kedalam sebuah elemen <p>

```
<!DOCTYPE html>
<html>
<body>

<h1 id="intro">My First Page</h1>

<p id="demo">Hello World!</p>

<script>
var myText = document.getElementById("intro").childNodes[0].nodeValue;
document.getElementById("demo").innerHTML = myText;
</script>

</body>
</html>
```

Result:

My First Page

My First Page

Dalam contoh diatas, `childNodes[0]` dapat diganti dengan properti `firstChild`

SIMPUL AKAR

Terkait simpul akar atau root nodes, ada dua properti khusus yang dapat mengakses seluruh dokumen:

- **`document.body`** - akan mengakses isi tubuh suatu dokumen HTML
- **`document.documentElement`** - dapat mengakses seluruh isi suatu dokumen HTML

Contoh dibawah ini akan menampilkan kotak alert yang berisi seluruh konten dari tubuh dokumen, yaitu yang terletak diantara tag `<body>` dan `</body>`

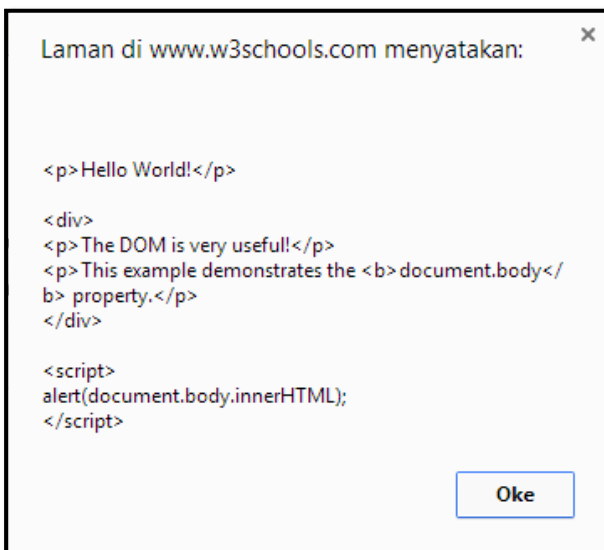
```
<!DOCTYPE html>
<html>
<body>

<p>Hello World!</p>

<div>
<p>The DOM is very usefull!</p>
<p>This example demonstrates the <b>document.body</b> property.</p>
</div>

<script>
alert(document.body.innerHTML);
</script>

</body>
</html>
```



Jika `document.body` diganti dengan `document.documentElement`, maka pada kotak alert akan berisi seluruh isi dokumen termasuk bagian kepala, `<head>...</head>`.

```
Laman di www.w3schools.com menyatakan:
<head> </head> <body>
<p>Hello World!</p>
<div>
<p>The DOM is very usefull!</p>
<p>This example demonstrates the
<b>document.documentElement</b> property.</p>
</div>
<script>
alert(document.documentElement.innerHTML);
</script> </body>
```

Oke

PROPERTI NODENAME

Properti ini menentukan nama sebuah simpul.

- ✓ Properti nodeName bersifat read-only
- ✓ nodeName dari simpul *element* sama dengan nama tag
- ✓ nodeName dari simpul *attribute* sama dengan nama atribut
- ✓ nodeName dari simpul *text* selalu #text
- ✓ nodeName dari simpul *document* selalu bernilai #document

PROPERTI NODEVALUE

Properti ini menentukan nilai dari sebuah simpul.

- ✓ nodeName dari simpul *element* bernilai *undefined*
- ✓ nodeName dari simpul *text* adalah teks itu sendiri
- ✓ nodeName dari simpul *attribute* adalah nilai atribut tersebut

PROPERTI NODETYPE

Properti nodeType mengembalikan jenis dari simpul dan bersifat read-only. Jenis simpul yang paling penting adalah:

Jenis Element	Jenis Simpul
Element	1
Attribute	2
Text	3
Comment	8
Document	9

SIMPUL ELEMENT

Penjelasan berikut ini akan membahas tentang simpul jenis *element*.

MEMBUAT ELEMEN BARU

Untuk menambah elemen HTML baru, kita harus membuat simpul *element* terlebih dahulu baru menambahkannya ke elemen yang ada.

```
<!DOCTYPE html>
<html>
<body>

<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
var para = document.createElement("p");
var node = document.createTextNode("This is new.");
para.appendChild(node);
var element = document.getElementById("div1");
element.appendChild(para);
</script>

</body>
</html>
```

Pada baris yang telah ditandai pada contoh diatas:

- baris pertama bermaksud membuat paragraf baru
- baris kedua membuat simpul *text* yang akan menjadi isi dari paragraf baru
- baris ketiga menambahkan simpul *text* ke paragraf baru
- baris keempat mengambil elemen div dengan id "div1"
- dan baris kelima menambahkan paragraf yang baru kedalam elemen div.

MEMBUAT ELEMEN BARU DENGAN INSERTBEFORE()

Metode `appendChild()` dalam contoh sebelumnya menambahkan elemen baru sebagai anak terakhir (last child). Jika tidak menginginkan demikian, maka gunakan metode `insertBefore()`.

```
<!DOCTYPE html>
<html>
<body>

<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
var para = document.createElement("p");
var node = document.createTextNode("This is new.");
para.appendChild(node);

var element = document.getElementById("div1");
var child = document.getElementById("p1");
element.insertBefore(para,child);
</script>

</body>
</html>
```

MENGHAPUS ELEMEN HTML YANG SUDAH ADA

Untuk menghapus suatu elemen, harus diketahui terlebih dahulu orang tuanya.

```
<!DOCTYPE html>
<html>
<body>
<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
var parent = document.getElementById("div1");
var child = document.getElementById("p1");
parent.removeChild(child);
</script>

</body>
</html>
```

Untuk menghapus suatu elemen, elemen orang tua dari elemen tersebut memang harus diketahui. Dengan demikian kodenya menjadi kurang praktis. Meskipun demikian untuk membuat kode lebih ringkas, kita dapat menggunakan menggunakan properti parentNode dari elemen yang akan dihapus untuk mengetahui elemen orang tuanya. Kodenya sebagai berikut:

```
var child = document.getElementById("p1");  
child.parentNode.removeChild(child);
```

MENGGANTI ELEMEN HTML

Untuk mengganti elemen HTML menggunakan metode replaceChild().

```
<!DOCTYPE html>  
<html>  
<body>  
  
<div id="div1">  
<p id="p1">This is a paragraph.</p>  
<p id="p2">This is another paragraph.</p>  
</div>  
  
<script>  
var parent = document.getElementById("div1");  
var child = document.getElementById("p1");  
var para = document.createElement("p");  
var node = document.createTextNode("This is new.");  
para.appendChild(node);  
parent.replaceChild(para,child);  
</script>  
  
</body>  
</html>
```

NODE LIST

Metode `getElementsByName` dalam kerjanya mengembalikan nilai berupa sebuah daftar simpul (node list). Node list merupakan sekumpulan simpul yang mirip array. Dengan demikian untuk mengakses item tertentu dari daftar tersebut menggunakan nomor indeksinya, seperti array.

Contoh:

```
var x = document.getElementsByTagName("p");  
y = x[1]
```

Contoh diatas mengumpulkan seluruh elemen paragraf dalam variabel x. Kemudian mengakses elemen paragraf yang kedua, yaitu menggunakan nomor indeks 1.

PANJANG DARI NODE LIST

Properti `length` dapat digunakan untuk mencari jumlah simpul didalam node list:

```
var myNodelist = document.getElementsByTagName("p");  
document.getElementById("demo").innerHTML = myNodelist.length;
```

Panjang node list akan bermanfaat ketika kita ingin memeriksa satu persatu simpul dalam node list:

```
<!DOCTYPE html>
<html>
<body>

<p>This is a p element</p>

<p>This is also a p element.</p>

<p>This is also a p element - Click the button to change the background color of all p
elements in this document.</p>

<button onclick="myFunction()">Try it</button>

<script>
function myFunction() {
  var myNodelist = document.getElementsByTagName("p");
  var i;
  for (i = 0; i < myNodelist.length; i++) {
    myNodelist[i].style.backgroundColor = "red";
  }
}
</script>

</body>
</html>
```