Javascript

Javascript

- JavaScript adalah bahasa pemrograman paling populer di dunia.
- JavaScript adalah bahasa pemrograman Web.
- JavaScript mudah dipelajari

Why Study JavaScript?

- JavaScript adalah salah satu dari 3 bahasa yang harus dipelajari semua pengembang web:
  1. HTML untuk mendefinisikan konten halaman web
  2. CSS untuk menentukan tata letak halaman web
  3. JavaScript untuk memprogram perilaku halaman web

JavaScript Dapat Mengubah Konten HTML

- Contoh : getElementById()

```
<!DOCTYPE html>
<html>
<body>
<h2>What Can JavaScript Do?</h2>
<p id="demo">JavaScript can change HTML content.</p>
<button type="button"
onclick='document.getElementById("demo").innerHTML =
"Hello JavaScript!"'>Click Me!</button>
</body>
</html>
```

JavaScript Dapat Mengubah Nilai Atribut HTML

- Contoh: JavaScript mengubah nilai atribut src (sumber) dari tag <img>:

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

<h2>What Can JavaScript Do?</h2>

<p>JavaScript can change HTML attribute values.</p>

<p>In this case JavaScript changes the value of the src (source) attribute of an
image.</p>

<button onclick="document.getElementById('myImage').src='pic_bulbon.gif'">Turn on the
light</button>

<img id="myImage" src="pic_bulboff.gif" style="width:100px">

<button onclick="document.getElementById('myImage').src='pic_bulboff.gif'">Turn off the
light</button>

</body>
</html>
```

JavaScript Dapat Mengubah StyleHTML (CSS)

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

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can change the style of an HTML element.</p>

<button type="button"
onclick="document.getElementById('demo').style.fontSize='35px'">Click Me!</button>

</body>
</html>
```

Letak JavaScript dalam HTML

- Dalam HTML, kode JavaScript disisipkan di antara tag <script> dan </script>.
- JavaScript lama menggunakan atribut type:

  <script type="text/javascript">
- Contoh :

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

<h2>JavaScript in Body</h2>

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

<script>
document.getElementById("demo").innerHTML = "My First JavaScript";
</script>

</body>
</html>
```

JavaScript Functions and Events

- JavaScript Functions adalah blok kode JavaScript, yang dapat dijalankan saat "dipanggil".
- Misalnya, JavaScript Functions dapat dipanggil ketika suatu event terjadi, contoh ketika pengguna mengklik tombol.

JavaScript in <head> or <body>

- Javascript dapat ditempatkan/ditulis di <body>, atau di bagian <head> halaman HTML, atau di keduanya.

  Contoh yang ditempatkan di <head>:

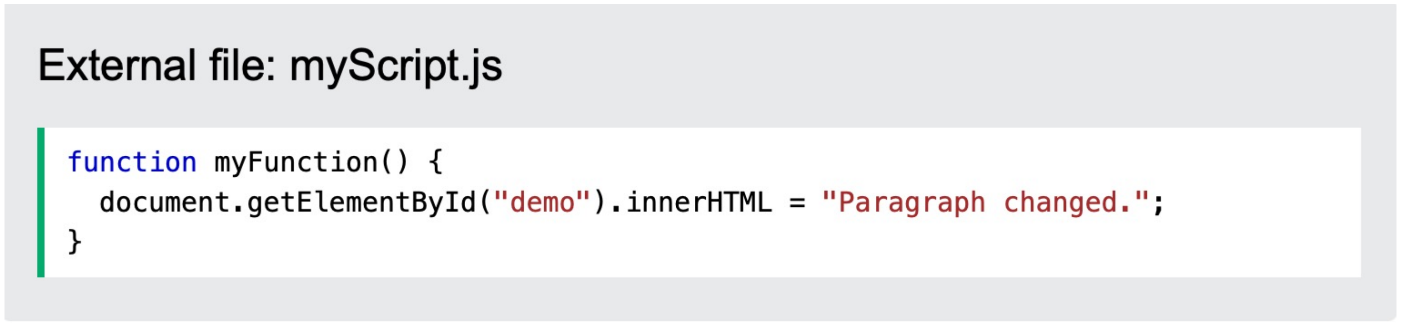
```
<!DOCTYPE html>
<html>
<head>
<script>
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>
</head>
<body><h2>Demo JavaScript in Head</h2>

<p id="demo">A Paragraph</p>
<button type="button" onclick="myFunction()">Try it</button>
</body>
</html>
```

Contoh yang ditempatkan di <body>:

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

  <h2>Demo JavaScript in Body</h2>

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

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

  <script>
  function myFunction() {
    document.getElementById("demo").innerHTML = "Paragraph changed.";
  }
  </script>

  </body>
  </html>
  ```

External JavaScript

- JavaScript juga dapat ditempatkan di file eksternal
- Script eksternal sangat praktis ketika kode tersebut digunakan di banyak halaman web yang berbeda.
- File JavaScript memiliki ekstensi file .js.



External file: myScript.js

```
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
```

- Untuk menggunakan skrip eksternal, letakkan nama file Script di atribut src (source) dari tag <script> contoh:

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

Keuntungan menggunakan External JS

- Memisahkan HTML dan kode
- Membuat HTML dan JavaScript lebih mudah dibaca dan dipelihara
- File JavaScript yang di-cache dapat mempercepat Page Load

External References

- Dengan Full URL (alamat web lengkap)

```
<script src="https://www.w3schools.com/js/myScript.js"></script>
```

- Dengan file path(seperti /js/)

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

- Tanpa path apapun

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

JavaScript Output

- JavaScript dapat "menampilkan" data dengan cara yang berbeda:
  - Writing into an HTML element, using innerHTML.
  - Writing into the HTML output using document.write().
  - Writing into an alert box, using window.alert().
  - Writing into the browser console, using console.log().
- Lihat Contoh : https://www.w3schools.com/js/js_output.asp

JavaScript Statements

- JavaScript Statement Terdiri dari : Values, Operators, Expressions, Keywords, and Comments.

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

<h2>JavaScript Statements</h2>

<p>In HTML, JavaScript statements are executed by the browser.</p>

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

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

</body>
</html>
```

Semicolon

Titik koma memisahkan JavaScript statements

```
let a, b, c;  // Declare 3 variables
a = 5;        // Assign the value 5 to a
b = 6;        // Assign the value 6 to b
c = a + b;    // Assign the sum of a and b to c


a = 5; b = 6; c = a + b;
```

White Space

- JavaScript mengabaikan banyak spasi.
- Kita dapat menambahkan spasi ke script agar lebih mudah dibaca.

```
let person =         "Hege";
let person= "Hege";
```

JavaScript Line Length and Line Breaks

- Untuk keterbacaan terbaik, pemrogram sering kali ingin menghindari baris kode yang lebih panjang dari 80 karakter.
- Jika pernyataan JavaScript tidak muat pada satu baris, tempat terbaik untuk memecahnya adalah setelah operator

Contoh :

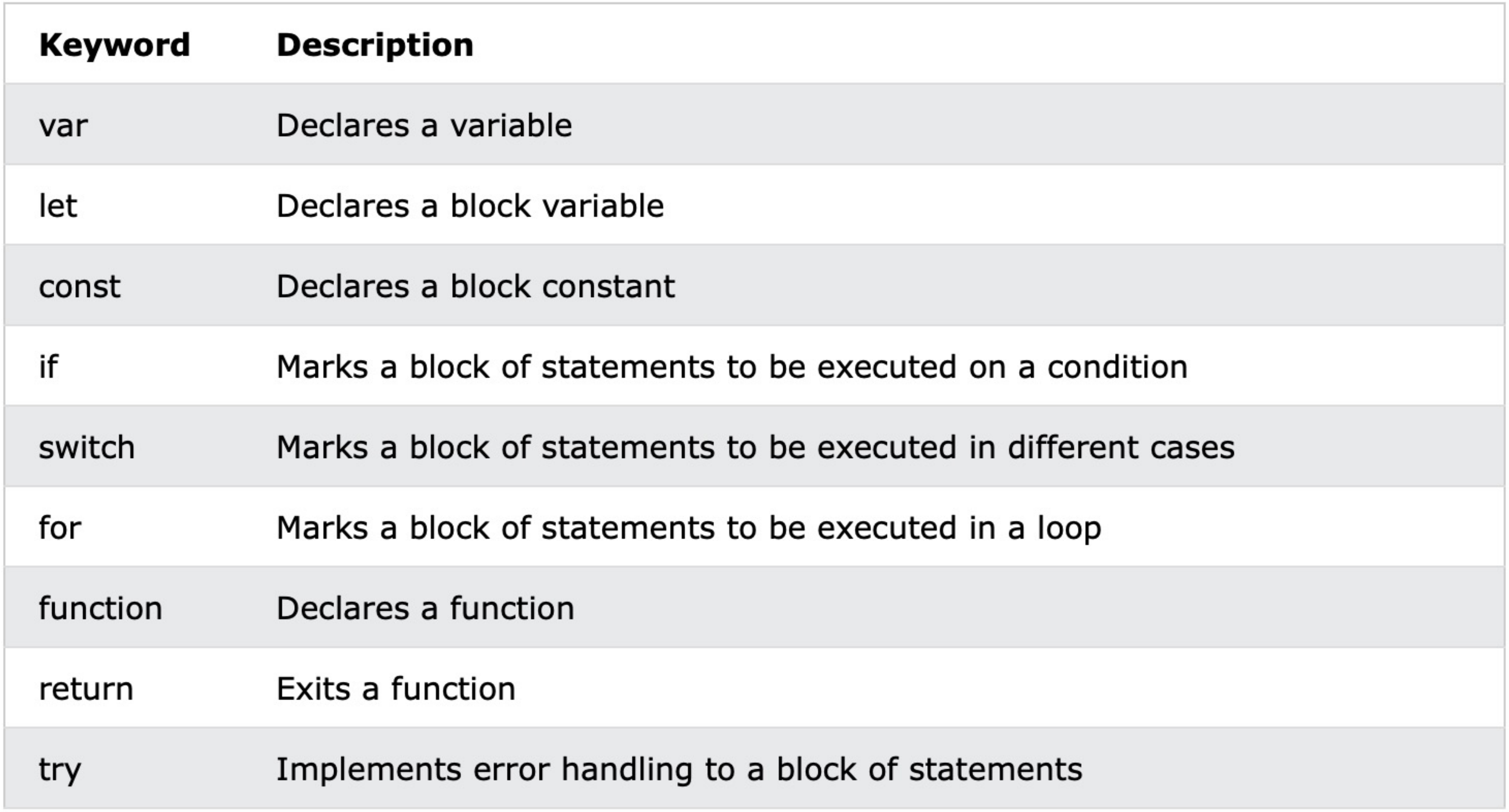
```
document.getElementById("demo").innerHTML =
"Hello Dolly!";
```

JavaScript Code Blocks

- JavaScript Statement dapat dikelompokkan bersama dalam blok kode, di dalam tanda kurung kurawal {...}.
- Tujuan dari blok kode adalah untuk mendefinisikan pernyataan yang akan dieksekusi bersama.
  - Contoh :

```
function myFunction() {
  document.getElementById("demo1").innerHTML = "Hello Dolly!";
  document.getElementById("demo2").innerHTML = "How are you?";
}
```

Beberapa Keyword



Keyword	Description
var	Declares a variable
let	Declares a block variable
const	Declares a block constant
if	Marks a block of statements to be executed on a condition
switch	Marks a block of statements to be executed in different cases
for	Marks a block of statements to be executed in a loop
function	Declares a function
return	Exits a function
try	Implements error handling to a block of statements

JavaScript Syntax

```
// How to create variables:
var x;
let y;

// How to use variables:
x = 5;
y = 6;
let z = x + y;
```

JavaScript Values

- The JavaScript syntax defines two types of values:
  - Fixed values
  - Variable values
- Fixed values are called **Literals**.

```
document.getElementById("demo").innerHTML = 10.50;

document.getElementById("demo").innerHTML = 'John Doe';
```

- Variable values are called **Variables**.

```
let x;
x = 6;
document.getElementById("demo").innerHTML = x;
```

JavaScript Operators

- JavaScript uses **arithmetic operators** (+ - * /) to **compute** values:

```
(5 + 6) * 10
```

- JavaScript uses an **assignment operator** (=) to **assign** values to variables:

```
let x, y;
x = 5;
y = 6;
```

JavaScript Comments

- Not all JavaScript statements are "executed".
- Code after double slashes // or between /* and */ is treated as a **comment**.
- Comments are ignored, and will not be executed:

```
let x = 5;   // I will be executed

// x = 6;   I will NOT be executed
```

JavaScript Identifiers / Names

- Identifiers are JavaScript names.
- Identifiers are used to name variables and keywords, and functions.
- The rules for legal names are the same in most programming languages.
- A JavaScript name must begin with:
  - A letter (A-Z or a-z)
  - A dollar sign ($)
  - Or an underscore (_)
- Subsequent characters may be letters, digits, underscores, or dollar signs.

- Note
  - Numbers are not allowed as the first character in names.
  - This way JavaScript can easily distinguish identifiers from numbers.

JavaScript is Case Sensitive

- All JavaScript identifiers are **case sensitive**.
- The variables lastName and lastname, are two different variables:

```
let lastname, lastName;
lastName = "Doe";
lastname = "Peterson";
```

JavaScript and Camel Case

- Historically, programmers have used different ways of joining multiple words into one variable name:
- **Hyphens** are **not allowed** in JavaScripts. **:**

  first-name, last-name, master-card, inter-city.
- **Underscore:**

  first_name, last_name, master_card, inter_city.
- **Upper Camel Case (Pascal Case):**

  FirstName, LastName, MasterCard, InterCity.
- **Lower Camel Case:**
- JavaScript programmers tend to use camel case that starts with a lowercase letter:

  firstName, lastName, masterCard, interCity.

JavaScript Variables

- 4 Ways to Declare a JavaScript Variable:
  - Using var

    ```
    var x = 5;
    var y = 6;
    var z = x + y;
    ```

  - Using let

    ```
    let x = 5;
    let y = 6;
    let z = x + y;
    ```

  - Using const

    ```
    const price1 = 5;
    const price2 = 6;
    let total = price1 + price2;
    ```

    If you want a general rule: always declare variables with const.

    If you think the value of the variable can change, use let.

    In this example, price1, price2, and total, are variables:

  - Using nothing

    ```
    x = 5;
    y = 6;
    z = x + y;
    ```

When to Use JavaScript var?

- Always declare JavaScript variables with var, let, or const.
- The var keyword is used in all JavaScript code from 1995 to 2015.
- The let and const keywords were added to JavaScript in 2015.
- If you want your code to run in older browser, you must use var.

JavaScript Let

- The let keyword was introduced in ES6 (2015) (ECMAScript 2015/ECMAScript 6).
- Variables defined with let cannot be Redeclared.

```
let x = "John Doe";
let x = 0;
// SyntaxError: 'x' has already been declared
```

  With var you can:

```
var x = "John Doe";
var x = 0;
```

- Variables defined with let must be Declared before use.

Variables defined with let have Block Scope.

- Before ES6 (2015), JavaScript had only **Global Scope** and **Function Scope**.
- ES6 introduced two important new JavaScript keywords: let and const.
- These two keywords provide **Block Scope** in JavaScript.
- Variables declared inside a { } block cannot be accessed from outside the block:

```
{
  let x = 2;
}
// x can NOT be used here
```

Variables declared with the var keyword can NOT have block scope.

```
{
  var x = 2;
}
// x CAN be used here
```

Redeclaring Variables

- Redeclaring a variable using the var keyword can impose problems.
- Redeclaring a variable inside a block will also redeclare the variable outside the block:

```
var x = 10;
// Here x is 10

{
var x = 2;
// Here x is 2
}
// Here x is 2
```

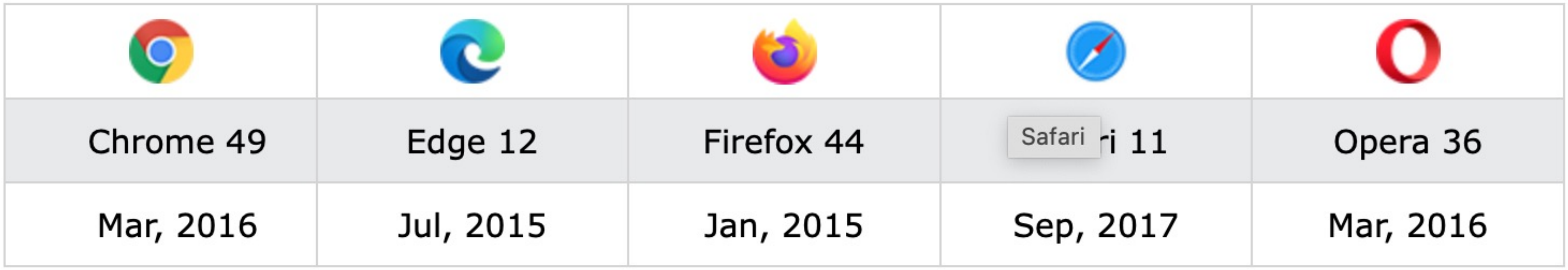
- Redeclaring a variable using the let keyword can solve this problem.

```
let x = 10;
// Here x is 10
{
let x = 2;
// Here x is 2
}

// Here x is 10
```

- Redeclaring a variable inside a block will not redeclare the variable outside the block:

Browser Support

- The let keyword is not fully supported in Internet Explorer 11 or earlier.
- The following table defines the first browser versions with full support for the let keyword:



Chrome 49	Edge 12	Firefox 44	Safari 11	Opera 36
Mar, 2016	Jul, 2015	Jan, 2015	Sep, 2017	Mar, 2016

Redeclaring

- Redeclaring a JavaScript variable with var is allowed anywhere in a program:

```
var x = 2;
// Now x is 2

var x = 3;
// Now x is 3
```

- With let, redeclaring a variable in the same block is NOT allowed:

```
var x = 2;   // Allowed
let x = 3;   // Not allowed

{
  let x = 2;   // Allowed
  let x = 3;   // Not allowed
}

{
  let x = 2;   // Allowed
  var x = 3;   // Not allowed
}
```

- Redeclaring a variable with let, in another block, IS allowed:

```
let x = 2;   // Allowed

{
    let x = 3;   // Allowed
}

{
  let x = 4;    // Allowed
}
```

Let Hoisting

- Variables defined with var are **hoisted** to the top and can be initialized at any time.
- Meaning: You can use the variable before it is declared:

```
carName = "Volvo";
var carName;
```

- Using a let variable before it is declared will result in a ReferenceError

```
carName = "Saab";
let carName = "Volvo";
```

JavaScript Const

- The const keyword was introduced in ES6 (2015).
- Variables defined with const cannot be Redeclared.

```
const PI = 3.141592653589793;
PI = 3.14;      // This will give an error
PI = PI + 10;   // This will also give an error
```

- Variables defined with const cannot be Reassigned.

```
const PI = 3.14159265359;  //Correct
```

```
const PI;
PI = 3.14159265359;
// Incorrect
```

- Variables defined with const have Block Scope.
  - Declaring a variable with const is similar to let when it comes to **Block Scope**.
  - The x declared in the block, in this example, is not the same as the x declared outside the block:

When to use JavaScript const?

- Use const when you declare:
  - A new Array

```
// You can create a constant array:
const cars = ["Saab", "Volvo", "BMW"];

// You can change an element:
cars[0] = "Toyota";

// You can add an element:
cars.push("Audi");
```

  - A new Object

```
// You can create a const object:
const car = {type:"Fiat", model:"500", color:"white"};

// You can change a property:
car.color = "red";

// You can add a property:
car.owner = "Johnson";
```

  - A new Function

```
function Car(make, model, year) {
  this.make = make;
  this.model = model;
  this.year = year;
}
const car1 = new Car('Eagle', 'Talon TSi', 1993);
```

  - A new RegExp

```
const re = new RegExp('ab+c', 'i');
```

JavaScript Data Types

```
let length = 16;                          // Number
let lastName = "Johnson";                 // String
let x = {firstName:"John", lastName:"Doe"};    // Object

let x = 16 + 4 + "Volvo"; //Result 20Volvo
let x = "Volvo" + 16 + 4; //Result Volvo164

let carName1 = "Volvo XC60";   // Using double quotes
let carName2 = 'Volvo XC60';   // Using single quotes

let x1 = 34.00;     // Written with decimals
let x2 = 34;        // Written without decimals

let y = 123e5;      // 12300000
let z = 123e-5;     // 0.00123

let x = 5;
let y = 5;
let z = 6;
(x == y)       // Returns true
(x == z)       // Returns false

const cars = ["Saab", "Volvo", "BMW"]; //Arrays

const person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};//Object
```

JavaScript Objects

- const car = {type:"Fiat", model:"500", color:"white"};
- const person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
- access object properties in two ways:
    - person.lastName;
    - person["lastName"];

-
  ```
  const person = {
    firstName: "John",
    lastName : "Doe",
    id       : 5566,
    fullName : function() {
      return this.firstName + " " + this.lastName;
    }
  };
  ```

Pembuatan fungsi dan cara pemanggilannya

```
<HTML>
<HEAD>
<TITLE>Contoh Program Javascript</TITLE>
</HEAD>
<SCRIPT>

function pesan(){
alert ("memanggil javascript lewat body onload")
}

</SCRIPT>
<BODY onload=pesan()>
</BODY>
</HTML>
```

Dasar Pemrograman Java Script

```
<HTML>
<HEAD>
<TITLE>Contoh Program Javascript</TITLE>
</HEAD>
<SCRIPT>
function test (val1,val2)
{
document.write("<br>"+"Perkalian : val1*val2 "+"<br>")
document.write(val1*val2)
document.write("<br>"+"Pembagian : val1/val2 "+"<br>")
document.write(val1/val2)
document.write("<br>"+"Penjumlahan : val1+val2 "+"<br>")
document.write(val1+val2)
document.write("<br>"+"Pengurangan : val1-val2 "+"<br>")
document.write(val1-val2)
document.write("<br>"+"Modulus : val1%val2 "+"<br>")
document.write(val1%val2)
}
</SCRIPT>
<BODY>
<input type="button" name="button1" value="arithmetic"
onclick=test(9,4)>
</BODY>
</HTML>
```

Operasi Relational

```
<HTML>
<HEAD>
<TITLE>Contoh Program Javascript</TITLE>
</HEAD>
<SCRIPT>
function test () {
val1=window.prompt("Nilai I :")
val2=window.prompt("Nilai II :")
document.write("<br>"+"val1==val2"+"<br>")
document.write(val1==val2)
document.write("<br>"+"val1!=val2"+"<br>")
document.write(val1!=val2)
document.write("<br>"+"val1&gtval2"+"<br>")
document.write(val1>val2)
document.write("<br>"+"val1&ltval2"+"<br>")
document.write(val1<val2) }
</SCRIPT>
<BODY>
<input type="button" name="button1" value="relational"
onclick=test()>
</BODY>
</HTML>
```

Seleksi kondisi (if..else)

```
<HTML>
<HEAD>
<TITLE>Contoh if-else</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE = "JavaScript">
<!--
var nilai = prompt("Nilai (0-100): ", 0);
var hasil = "";
if (nilai >= 60)
hasil = "Lulus";
else
hasil = "Tidak Lulus";
document.write("Hasil: " + hasil);
//-->
</SCRIPT>
</BODY>
</HTML>
```

Switch Case

```
<HTML>
<HEAD>
<TITLE>Contoh Program Javascript</TITLE>
</HEAD>
<SCRIPT language="Javascript">
function test ()
{
val1=window.prompt("Input Nilai (1-5):")
switch (val1)
{
case "1" :
document.write("bilangan satu")
break
case "2" :
document.write("bilangan dua")
break
case "3" :
document.write("bilangan tiga")
break
case "4" :
document.write("bilangan empat")
break
case "5" :
document.write("bilangan lima")
break
default :
document.write("bilangan lainnya")
}
}
</SCRIPT>
<BODY>
<input type="button" name="button1"
value="switch"
onclick=test()>
</BODY>
</HTML>
```

Pemakaian looping < for >

```
<HTML>
<HEAD>
<TITLE>Contoh Program Javascript</TITLE>
</HEAD>
<BODY>
<SCRIPT language="Javascript">
<!--
for (x=0;x<=10;x++)
document.write(x+"<br>")
// -->
</SCRIPT>
</BODY>
</HTML>
```

Pemakaian looping < do..while >

```
<HTML>
<HEAD>
<TITLE>Contoh Program Javascript</TITLE>
</HEAD>
<BODY>
<SCRIPT language="Javascript">
<!--
var x=0
do{
document.write(x+"<br>")
x++;
}
while (x<=10)
// -->
</SCRIPT>
</BODY>
</HTML>
```

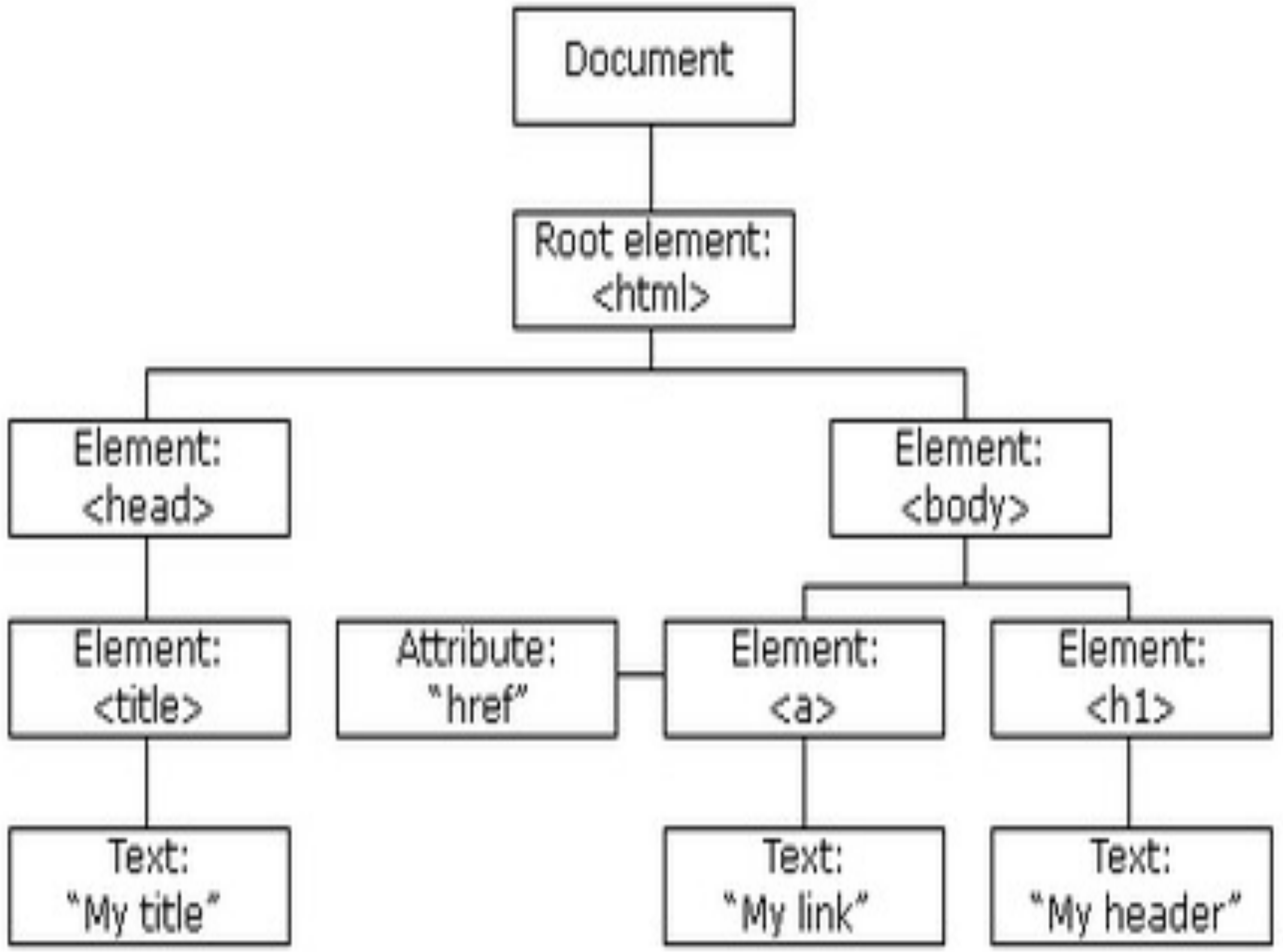
Pemakaian looping < while >

```
<HTML>
<HEAD>
<TITLE>Contoh Program Javascript</TITLE>
</HEAD>
<BODY>
<SCRIPT language="Javascript">
<!--
var x=0
while (x<=10){
document.write(x+"<br>")
x++;
}
// -->
</SCRIPT>
</BODY>
</HTML>
```

JavaScript HTML DOM

- Ketika sebuah halaman web diload, browser menciptakan halaman Document Object Model(DOM)

The HTML DOM Tree



Dengan DOM, JavaScript dapatmembuat HTML yang dinamis:

- JavaScript dapat mengubah semua elemen HTML
- JavaScript dapat mengubah semua atribut HTML
- JavaScript dapat mengubah semua CSS style pada halaman HTML
- JavaScript dapat bereaksi terhadap semua kejadian di halaman HTML

Menemukan Elemen HTML

- Ada beberapa cara:
  - Menemukan elemen HTML dengan id (elements by id)
  - Menemukan elemen HTML dengan nama tag (elements by tag name)
  - Menemukan elemen HTML dengan nama kelas (elements by class name)

Contoh

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

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

<script>
x=document.getElementById("intro");
document.write("<p>The text from the intro paragraph: " +
x.innerHTML + "</p>");
</script>

</body>
</html>
```

Merubah Content HTML

```
<!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>

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

</body>
</html>
```

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

<h1 id="header">Old Header</h1>

<script>
var element=document.getElementById("header");
element.innerHTML="New Header";
</script>

<p>"Old Header" was changed to "New Header"</p>

</body>
</html>
```

Merubah Atribut HTML

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

<img id="image" src="smiley.gif" width="160" height="120">

<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>
```

Changing HTML Style

```
<html>
<body>

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

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

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

</body>
</html>
```

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

<h1 id="id1">My Heading 1</h1>
<button type="button"
onclick="document.getElementById('id1').style
.color='red'">
Click Me!</button>

</body>
</html>
```

JavaScript HTML DOM Events

```
<!DOCTYPE html>
<html>
<body>
<h1
onclick="this.innerHTML='Ooops!'">C
lick on this text!</h1>
</body>
</html>
```

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

HTML Event Attributes

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
<p>Click the button to execute the <em>displayDate()</em> function.</p>
<button id="myBtn">Try it</button>
<script>
document.getElementById("myBtn").onclick=function(){displayDate()};
function displayDate()
{
document.getElementById("demo").innerHTML=Date();
}
</script>
<p id="demo"></p>
</body>
</html>
```

```
<!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>
```

Contoh Form Input

```
<html>
<head> </head>
<SCRIPT language="Javascript">
function test () {
var val1=document.kirim.T1.value
if (val1%2==0)
document.kirim.T2.value="bilangan genap"
else
document.kirim.T2.value="bilangan ganjil"
}
</SCRIPT>
<body>
<form method="POST" name="kirim">
<p>BIL <input type="text" name="T1" size="20">
MERUPAKAN BIL <input type="text" name="T2" size="20">
</p>
<p><input type="button" value="TEBAK" name="B1" onclick=test()>
</p>
</form>
</body>
</html>
```

Form Button

```
<HTML>
<HEAD>
<TITLE>Objek document</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE = "JavaScript">
<!--
function ubahWarnaLB(warna) {
document.bgColor = warna;
}
function ubahWarnaLD(warna) {
document.fgColor = warna;
}
//-->
</SCRIPT>
<H1>TES</H1>
<FORM>
<INPUT TYPE = "BUTTON"
VALUE = "Latar Belakang Hijau"
onClick = "ubahWarnaLB('GREEN')">
<INPUT TYPE = "BUTTON"
VALUE = "Latar Belakang Putih"
onClick = "ubahWarnaLB('WHITE')">
<INPUT TYPE = "BUTTON"
VALUE = "Teks Kuning"
onClick = "ubahWarnaLD('YELLOW')">
<INPUT TYPE = "BUTTON"
VALUE = "Teks Biru"
onClick = "ubahWarnaLD('BLUE')">
</FORM>
<SCRIPT LANGUAGE = "JavaScript">
<!--
document.write("Dimodifikasi terakhir
pada " +
document.lastModified);
//-->
</SCRIPT>
</BODY>
</HTML>
```

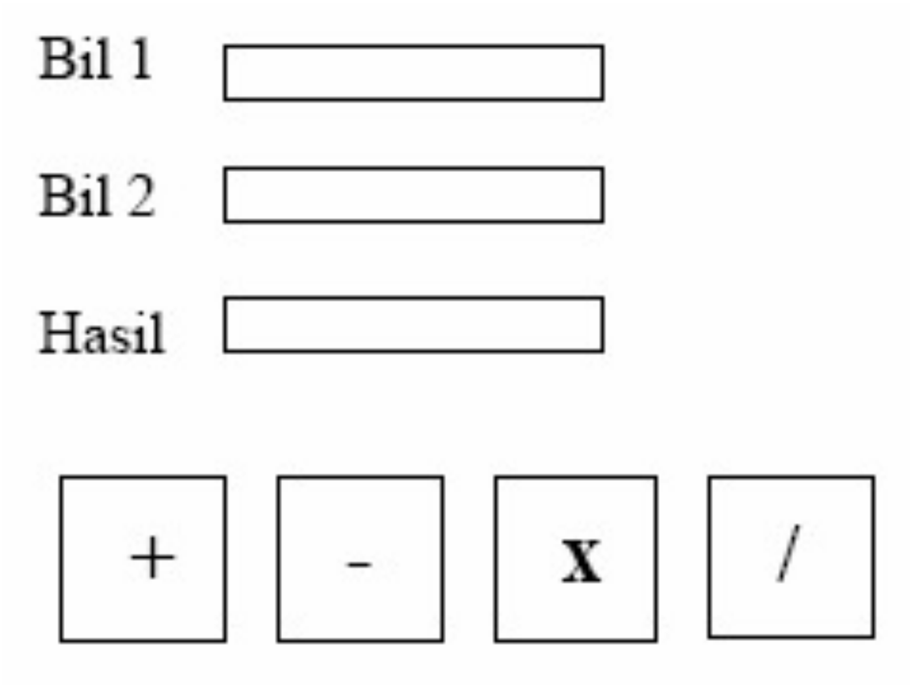
Tugas

Buat halaman html untuk mengkonversi nilai angka menjadi nilai huruf dengan menggunakan javascript

Konversi :

- 0-40 =E
- 41-55=D
- 56-60=C
- 61-65=BC
- 66-70=B
- 71-80=AB
- 81-100=A

- Buat halaman html untuk menampilkan aplikasi program kalkulator sederhana dengan menggunakan javascript.



- Bil 1 dan Bil 2 merupakan text box, dapat diisi angka, bila tombol + atau – atau x atau / ditekan, maka akan keluar bilangan pada text box hasil, dimana bilangan ini merupakan operasi arithmetic sesuai dengan tombol yang ditekan

Contoh dan referensi

- http://www.w3schools.com/js/