



Web Design Training Module

June 2011

For the latest information, please see bluejack.binus.ac.id



The entire risk of the use or the results from the use of this document remains with the user. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Binus University.

Binus University may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Binus University, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, email address, logo, person, place or event is intended or should be inferred.

© 2011 Binus University. All rights reserved.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

SYSTEM REQUIREMENT

Software yang digunakan adalah **Microsoft Expression Web 4**.


Berikut adalah persyaratan sistem untuk *Microsoft Expression Web 4*:

Windows

- Microsoft® Windows® XP with Service Pack 3, Windows Vista™, Windows 7 or Windows Server 2008 operating system
- PC with 1 GHz or faster processor
- 1 GB of RAM or more
- 2 GB or more of available hard-disk space
- .NET Framework 4.0
- Silverlight 4.0
- Support for Microsoft DirectX® 9.0 graphics with Windows Vista Display Driver Model (WDDM) Driver, 128 MB of graphics RAM or more, Pixel Shader 3.0 in hardware, 32-bits per pixel
- DVD compatible drive
- 1024 x 768 or higher resolution monitor with 24-bit color
- Internet functionality requires Internet access (additional fees may apply)
- Actual requirements and product functionality may vary based on your system configuration and operating system.
- Some product features require FireFox 3.0 or later, and Internet Explorer 8

Contents

SYSTEM REQUIREMENT	3
Windows	3
Web Design	5
Module 01 – HTML Basic.....	5
Web Design	9
Module 02 - HTML Intermediate	9
Web Design	15
Module 03 – HTML Advanced	15
Web Design	18
Module 04 – JavaScript Basic	18
Web Design	26
Module 05 – JavaScript Intermediate	26
Web Design	37
Module 06 – Using JavaScript on Form Validation	37
Web Design	40
Module 07 – VBScript Basic	40
Web Design	45
Module 08 - CSS	45
Web Design	74
Module 09 – Data Binding	74

Web Design	
<i>Module 01 – HTML Basic</i>	
Last Update 07/11/2011	Revision 00

1. Module Description

- Pada bagian ini akan dijelaskan mengenai pengenalan HTML dasar, seperti tag-tag HTML, Text Styling, Linking, Image dan Line.

2. Learning Outcomes

- Mahasiswa dapat memahami konsep dasar pembuatan Web dengan HTML.

3. Material

a. HTML Introduction

HTML (Hyper Text Markup Language) adalah sebuah bahasa pemrograman yang dibuat untuk membuat web page. Untuk membuat web page kita dapat menggunakan text editor biasa seperti notepad. Kita cukup mengetik tag – tag HTML di notepad tersebut dan menyimpan filenya dengan format html atau htm (namafile.htm). Tag – tag HTML selalu ditandai dengan tanda (< >) dan case-insensitive.

Tag biasanya muncul sepasang, ada pembuka dan penutup. Meskipun tidak semua tag membutuhkan tag penutup. Bentuknya adalah sebagai berikut : <tag> </tag> dimana <tag> merupakan pembuka dan </tag> merupakan penutup. Struktur penulisan tag yang tepat adalah sebagai berikut :

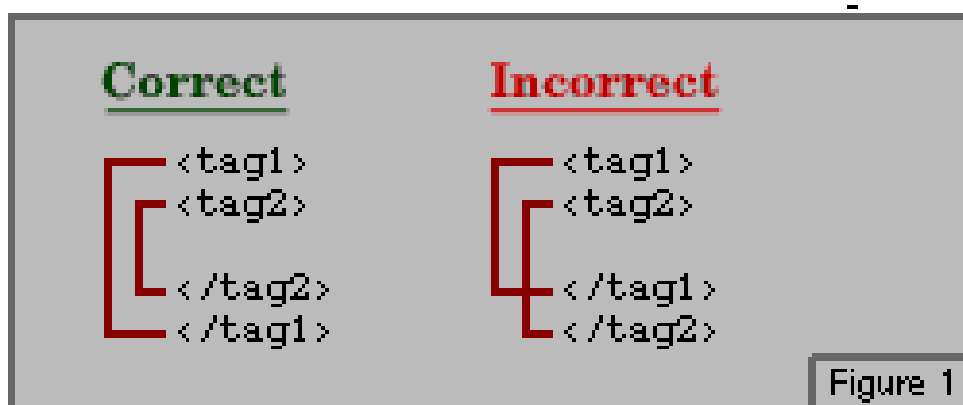


Figure 1

b. General Tag HTML, Text Styling, Linking, Image, Line

- **General Tag HTML :**

- ***HTML***

Tag pertama dan terakhir dalam sebuah dokumen web adalah tag HTML. Ini adalah tag yang memberi tahu web browser dimana awal dan akhir suatu halaman dokumen HTML.

- ***Head***

Teks yang ditulis di dalam tag <head> merupakan informasi header. Informasi header tidak untuk ditampilkan di dalam window browser.

- ***Title***

Tag ini diletakkan di dalam tag <head>. Teks yang ditulis di antara tag <title> merupakan judul dari dokumen dan akan muncul diatas title bar browser.

- ***Body***

Tag <body> diletakkan setelah tag <head>. Apa yang diletakkan di antara tag <body> akan dimunculkan di window browser, baik gambar maupun tulisan.

- ***Comment***

Untuk memberikan komentar di dalam dokumen HTML namun tidak ingin ditampilkan di window browser kita menggunakan tag comment, yaitu sbb : <!-- ini komentar -->.

- **Text Styling :**

- ***Headings***

Terdapat 6 tingkatan heading yaitu dari heading1 sampai heading6, ditulis dengan <h1> sampai <h6>. Secara otomatis baris baru akan ditambahkan sebelum dan sesudah heading.

<H1> Heading 1 </H1> Heading 1

<H2> Heading 2 </H2> Heading 2

<H3> Heading 2 </H2> Heading 3

<H4> Heading 4 </H4> Heading 4

<H5> Heading 5 </H5> Heading 5

<H6> Heading 6 </H6> Heading 6

- ***Paragraphs***

Paragraph dibuat dengan tag <p>. Secara otomatis baris baru akan ditambahkan sebelum dan sesudah paragraph.

```
<p> Paragrah 1 </p>
<p> Paragrah 2 </p>
<p> Paragrah 3 </p>
```

- **Linking**

HTML menggunakan tag anchor <a> untuk menciptakan link ke document lain. Sebuah anchor dapat menunjuk ke banyak sumber pada web seperti halaman HTML, gambar, sound file, movie, dll.

```
<a href="url" name="label" target="_blank">Ini untuk di
klik</a>
```

- Tag <a> digunakan untuk membuat anchor.
- Atribut href digunakan untuk memberitahu alamat tujuan link.
- Atribut name digunakan untuk membuat nama anchor, sehingga kita dapat membuat link langsung menuju alamat spesifik dari halaman.
- Atribut target digunakan untuk mendefinisikan dimana dokumen yang dituju akan dibuka. Untuk membuka dokumen di window browser baru digunakan : target="_blank".
- Tulisan diantara tag <a> akan ditampilkan sebagai hyperlink.

Contoh :

Untuk membuat link ke halaman google.com

```
<a href="http://google.com" name="link1"
target="_blank">www.yahoo.com</a>
```

- **Image**

Image didefinisikan dengan tag . Untuk menampilkan sebuah gambar di halaman, kita menggunakan atribut src. Atribut alt digunakan untuk mendefinisikan "alternate text" untuk sebuah image. Alternate text akan ditampilkan ketika browser tidak dapat memload gambar.

```

```

- **Line :**

- *Horizontal Lines*


Tag `<hr>` digunakan membuat garis pada halaman web. Tag `<hr>` ini tidak membutuhkan tag penutup.

```
<hr width=50% size=10 noshade>
```

- ***Line Breaks***

Tag `
` digunakan ketika kita ingin membuat baris baru namun tidak ingin memulai paragraph baru. Tag `
` ini tidak membutuhkan tag penutup.

```
<p>TPADB <br> Think Positive and <br> Do the Best </p>
```


Web Design	
<i>Module 02 - HTML Intermediate</i>	
Last Update 07/11/2011	Revision 00

- **Module Description**
 - Pada Bagian ini akan dijelaskan mengenai List, Table, Frameset, dan Meta.
- **Learning Outcomes**
 - Mahasiswa dapat memahami penggunaan List, Table, Frameset, dan Meta.
 - Mahasiswa dapat mengaplikasikan beberapa dasar – dasar komponen web dengan menggunakan *tag* HTML.
- **Material**
 - **List**

An ordered list:	An unordered list:
<ol style="list-style-type: none"> 1. The first list item 2. The second list item 3. The third list item 	<ul style="list-style-type: none"> • List item • List item • List item

HTML Unordered List :

Code:

```
<ul>
  <li>Coffee</li>
  <li>Milk</li>
</ul>
```

Result:

- Coffee
- Milk

HTML Ordered List :

Code:

```
<ol>
  <li>Coffee</li>
  <li>Milk</li>
</ol>
```

Result:

1. Coffee
2. Milk

HTML Definition List :

Code:

```
<dl>
  <dt>Coffee</dt>
  <dd>- black hot drink</dd>
  <dt>Milk</dt>
  <dd>- white cold drink</dd>
</dl>
```

Result:

```
Coffee
    - black hot drink
Milk
    - white cold drink
```

○ **Table**

Table didefinisikan dengan tag <table>. Table dibagi terdiri dari row <dengan tag <tr>) dan setiap row terdiri dari data cell (dengan tag <td>).

```
<table border="1">
  <tr>
    <td>satu</td>
    <td>dua</td>
  </tr>
  <tr>
    <td>tiga</td>
    <td>empat</td>
  </tr>
</table>
```

Tampilan pada browser :

satu	dua
tiga	empat

```
<table border="1">
  <tr >
    <td colspan = 2>satu</td>
    <td rowspan = 2>dua</td>
  </tr>
  <tr>
    <td>tiga</td>
    <td>empat</td>
  </tr>
</table>
```

Tampilan pada browser :

satu		dua
tiga	empat	

○ Meta

Beberapa engine pencarian akan menggunakan nama dan isi dari sebuah atribut yang disebut dengan meta. Berikut ini adalah element dari tag meta yang menggambarkan sesuatu dari sebuah website:

```
<meta name="description" content="Free Web tutorials on HTML, CSS, XML" />
```

Berikut ini adalah element dari tag meta yang mendefinisikan sesuatu dari sebuah website:

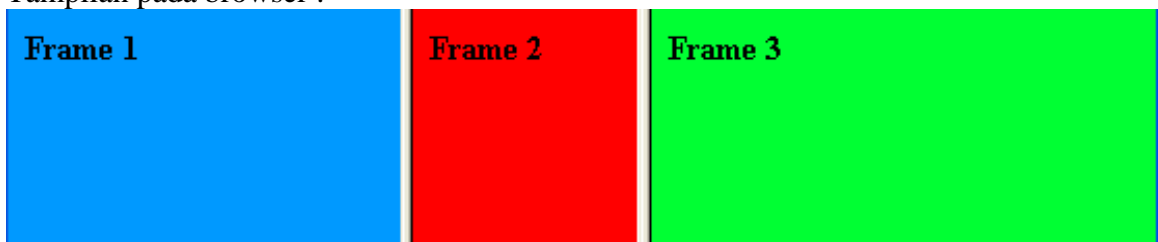
```
<meta name="keywords" content="HTML, CSS, XML" />
```

○ Frames

Dengan frames, kita dapat menampilkan lebih dari satu dokumen HTML dalam window browser yang sama. Tag <frameset> mendefinisikan bagaimana membagi window ke dalam frame-frame. Tag <frame> mendefinisikan dokumen HTML apa yang diletakkan di dalam setiap frame

```
<frameset cols="35%,20%,*">
<frame src="1.html">
<frame src="2.html">
<frame src="3.html">
</frameset>
```

Tampilan pada browser :




Ukuran kolom frameset dapat pula kita atur dalam pixels (cols="200,500") dan salah satu kolom dapat diatur menggunakan ukuran space yang tersisa (cols="25%,*").

Nama Tag	Atribut	Contoh	Keterangan
<title> ... </title>			Mencetak judul pada caption browser
<body> ... </body>	background	Background = "nama_file.tipe_file"	Memberikan gambar background pada halaman web
	bgcolor	bgcolor = "#xxxxxx"	Memberikan warna

			background pada halaman web
	leftmargin	leftmargin = "0"	Menentukan margin kiri dokumen
	topmargin	topmargin = "0"	Menentukan margin atas dokumen.
<h1..h6> ... </h1.../h6>			Mengubah ukuran teks
	align	align = "center"	Perataan heading
<p> ... </p>	align		Perataan paragraph
<hr>			Mencetak garis horizontal
	align		Mencetak garis horizontal di kanan, kiri dan tengah
	color	color = "#xxxxxx"	Menentukan warna garis
	noshade		Membuat garis tanpa efek 3D
	size	size = "5"	Menentukan tinggi garis
 			Mengganti baris
<a> ... 	href	href = "nama_file.tipe_file"	Menentukan tujuan dari suatu link
	name	name = "link"	Memberi nama anchor
 ... 			Membuat teks menjadi tebal
 ... 			Membuat teks menjadi tebal
<i> ... </i>			Membuat teks menjadi miring
 ... 			Membuat teks menjadi miring
<s> ... </s>			Membuat teks memiliki sebuah coretan
_{...}			Membuat teks menjadi subscript
^{...}			Membuat teks menjadi superscript
<pre> ... </pre>			Menampilkan sesuai dengan teks editor
 .. 	size	size = "+2"	Menentukan ukuran huruf
	color	color = "#xxxxxx"	Menentukan warna huruf
	face	face = "tipe_huruf"	Menentukan jenis huruf

<table> ... </table>			Membuat table
	border	border = "2"	Menentukan lebar garis
	align	align = "center"	Menentukan letak table
	width	width = "20"	Menentukan lebar table
	cellpadding	cellpadding = "3"	Mensetting jarak data dengan garis
	cellspacing	cellspacing = "5"	Mensetting jarak antar sel-sel table
	bgcolor	bgcolor = "#xxxxxx"	Memberikan warna table
	background	background = "nama_file.tipe_file"	Memberikan gambar background table
	bordercolor	bordercolor = "#xxxxxx"	Menentukan warna garis table
<tr> ... </tr>			Sebagai baris pada table
	valign	valign = "top"	Menentukan perataan vertical teks dalam sel table
	align	align = "center"	Menentukan perataan teks dalam table
	bgcolor	bgcolor = "#xxxxxx"	Menentukan warna latar belakang sel
	background	background = "nama_file.tipe_file"	Menentukan gambar latar belakang sel
<td> ... </td>			Sebagai kolom pada table
	valign	valign = "top"	Menentukan perataan vertical teks dalam sel table
	align	align = "center"	Menentukan perataan teks dalam table
	bgcolor	bgcolor = "#xxxxxx"	Menentukan warna latar belakang sel
	background	background = "nama_file.tipe_file"	Menentukan gambar latar belakang sel
	width	width = "20"	Menentukan lebar sel table
	height	height = "50"	Menentukan tinggi sel table
	colspan	colspan = "2"	Menentukan jumlah kolom yang digabung menjadi satu

	rowspan	rowspan = "2"	Menentukan jumlah baris yang digabung menjadi satu
 ... 	type(1, i, I, a, A)	type = "a"	Pengurutan data
 ... 	type(disc, circle, square)	type = "disc"	Pengurutan data
 ... 			Menyisipkan sebuah daftar
<!-- ... -->			Komentar
 ... 			Meletakkan gambar
	src	src = "nama_file.type_file"	Mengambil file gambar
	width	width = "50"	Menentukan lebar gambar
	height	height = "50"	Menentukan tinggi gambar
	alt	alt = "akan dimunculkan saat kursor didekatkan ke gambar"	Tooltip dari gambar

Web Design	
<i>Module 03 – HTML Advanced</i>	
Last Update 07/11/2011	Revision 00

- **Module Description**
 - Pada Bagian ini akan dijelaskan mengenai HTML Advanced yang meliputi Form beserta dengan Komponen Form.
- **Learning Outcomes**
 - Mahasiswa memahami cara menggunakan Form dan Komponen Form
 - Mahasiswa dapat memahami cara kerja Form dan mengenal object-object dalam Form.
- **Material**

- **Form and Form Component**

Form didefinisikan dengan tag `<form>`. Sebuah form adalah area yang mengandung element-element form. Element form adalah element yang memungkinkan user untuk memasukkan informasi, seperti (textfield, textarea, drop-down menu, radio button, checkbox, dll) dalam form.

Textfield : untuk meminta inputan berupa huruf, angka, dsb.

Radio button : untuk meminta inputan dengan memilih satu dari pilihan yang ada.

Checkbox : untuk meminta inputan dengan memilih satu atau lebih pilihan yang ada.

Textarea : untuk meminta inputan berupa huruf, angka, dsb dalam jumlah karakter yang tidak terbatas.

Drop-down : untuk meminta inputan dari list ke bawah yang dapat dipilih.

Button : untuk membuat tombol pada halaman form.

```

<form id="form1" name="form1" method="post" action="">

<input type = "text" name="nama" value="-isi nama disini-">
<br>

<input type = "password" name="pass" value="-isi password-">
<br>

<input type = "radio" name="gender" value="male"> Male

```

```

<input type = "radio" name="gender" value="female"> Female
<br>

<input type = "checkbox" name="hobby" value="reading"> reading
<br>
<input type = "checkbox" name="hobby" value="coding"> coding
<br>

<textarea name = "address">Kemanggisan Raya, Jakarta</textarea>
<br>

<select name = "animal">
    <option value="bear"> Bear </option>
    <option value="cat"> Cat </option>
    <option value="dog"> Dog </option>
</select>
<br>

<input type = "submit" name="btnRegister" value="Register">
<input type = "reset" name="btnReset" value="Reset">


</form>

```

Tampilan di browser :

Nama Tag	Atribut	Contoh	Keterangan
<form> ... </form>	action	Action = ”nama_file.type_file”	Menyatakan alamat tujuan server
	method	method = “post”	Menentukan bagaimana form

			dikirim
	name	Name = "nama_form"	Memberikan nama form
<input>	maxlength	maxlength = "25"	Menentukan jumlah maksimal karakter yang bisa dimasukkan
	name	name = "nama_input"	Memberi nama jenis input
	size	size = "25"	Menentukan ukuran jenis input
	src		Mengambil data atau gambar
	type	type = "button"	Menentukan jenis input
	value	value = "nilai_input"	Menentukan nilai input
<textarea> ... </textarea>	rows	rows = "10"	Menentukan tinggi textarea
	cols	cols = "20"	Menentukan lebar textrea
	name	name = "nama_textarea"	Memberi nama textarea
<select> ... </select>	size	size = "10"	Menentukan tinggi menu pilihan
	multiple		Membuat user dapat memilih lebih dari satu
	name	name = "nama_menu"	Memberi nama menu pilihan
<option> ... </option>			Membuat daftar pilihan
	value	value = "nilai_dari_item"	Memberi nilai setiap daftar

Web Design	
<i>Module 04 – JavaScript Basic</i>	
Last Update 07/11/2011	Revision 00

- **Module Description**

- Pada Bagian ini akan dijelaskan mengenai JavaScript Basic seperti input / output, variable, arithmetic operation, selection, dan module / function

- **Learning Outcomes**

- Mahasiswa dapat memahami input / output, variable, arithmetic operation, selection, dan module / function dengan JavaScript.
- Mahasiswa dapat memahami dasar komponen web dan menerapkannya menggunakan bahasa HTML serta penggunaan client-side dengan JavaScript.

- **Material**

- **Input / Output**

Javascript adalah bahasa scripting pada web. Javascript digunakan untuk menambah interaktivitas dengan halaman HTML. Javascript dapat ditulis dalam suatu dokumen html melalui salah satu cara dibawah ini :

- ❖ **Inline**

Dituliskan dalam tag body sehingga ketika browser dijalankan Javascript langsung dijalankan.

Contoh :

```
<a href='javascript : alert("Hello World")'> Klik me </a>
```

- ❖ **Embed**

Diletakkan diantara tag `<script> ... </script>` dalam sebuah dokumen HTML. Di dalam tag `<script>` kita gunakan attribute “type=” untuk mendefinisikan bahasa scripting yang digunakan. Jadi, `<script type="text/javascript">` dan `</script>` memberitahukan dimana javascript dimulai dan diakhiri.

Contoh :

```
<script type="text/javascript">
    Document.write("Hello World");
</script>
```

Javascript statement adalah perintah untuk memberitahu browser apa yang akan dikerjakan. Ini adalah perintah untuk menulis “Hello World” ke dalam halaman web.

```
document.write("Hello World");
```

Setiap akhir statement boleh menggunakan tanda titik dua (;) atau pun tidak. Setiap akhir baris akan diinterpretasikan sebagai akhir dari statement oleh browser. Sehingga penggunaan tanda (;) pada akhir statement tidak harus digunakan, namun memungkinkan kita untuk menulis multiple statement dalam satu baris.

JavaScript juga memiliki sebuah Comment dimana pada baris tersebut tidak akan dijalankan oleh browser. Comment untuk satu baris menggunakan //, dan komentar untuk banyak baris diawali dengan /* dan diakhiri dengan */.

```
<script type="text/javascript">
document.write("<h1>Heading 1</h1>"); // untuk membuat heading
/*
document.write("<p>Membuat paragraph</p>");
document.write("<p>dengan menggunakan javascript</p>");
*/
</script>
```

○ Variable

Variabel adalah tempat dimana kita menyimpan nilai-nilai atau informasi-informasi pada JavaScript. Variabel yang dideklarasikan dapat diisi dengan nilai apa saja.

Dalam JavaScript pendeklarasian sebuah variabel sifatnya opsional, artinya anda boleh mendeklarasikan atau tidak hal tersebut tidak menjadi masalah. Jika anda memberi nilai pada variabel, maka dalam JavaScript dianggap bahwa anda telah mendeklarasikan variabel tersebut.

Aturan penamaan variabel :

- Harus diawali dengan karakter (huruf atau baris bawah)
- Tidak boleh menggunakan spasi
- Huruf Kapital dan kecil memiliki arti yang berbeda
- Tidak boleh menggunakan kata-kata yang merupakan perintah dalam JavaScript.

Untuk mendeklarasikan variabel dalam javascript digunakan **var**.

Contoh:

```
var angka = 5;
var nama = "Bluejack";
```

Apabila kita mendeklarasikan ulang suatu variabel, nilai awal yang telah diberikan tidak akan hilang.

```
var x = 5;
var x;
```

Setelah eksekusi dari statement diatas, variabel x masih tetap memiliki nilai 5. Nilai dari variabel x tidak dihilangkan ketika kita mendeklarasikan ulang variabel tersebut.

- Tipe Data

Tidak seperti bahasa pemrograman lainnya, JavaScript tidak memiliki tipe data secara eksplisit. Hal ini dapat dilihat dari beberapa contoh variabel diatas. Anda mendeklarasikan variabel tapi tidak menentukan tipenya.

Meskipun JavaScript tidak memiliki tipe data secara eksplisit. JavaScript mempunyai tipe data implisit. Terdapat empat macam tipe data implisit yang dimiliki oleh JavaScript yaitu :

- Numerik, seperti : 0222532531, 1000, 45, 3.146789 dsb
- String, seperti : “Hallo”, “April”, “Jl. Setiabudi No 17A”, “Cece Kirani” dsb
- Boolean, bernilai true atau false
- Null, variabel yang tidak diinisialisasi

○ Arithmetic Operation

Jika $y = 5$ maka hasil dari Arithmetic Operation / operasi aritmatika sesuai dengan operator aritmatika adalah

Operator	Description	Example	Result
+	Addition	$x=y+2$	$x=7$
-	Subtraction	$x=y-2$	$x=3$
*	Multiplication	$x=y*2$	$x=10$
/	Division	$x=y/2$	$x=2.5$
%	Modulus (division remainder)	$x=y\%2$	$x=1$
++	Increment	$x=++y$	$x=6$
--	Decrement	$x=--y$	$x=4$

- Operator Penugasan
Jika $x = 10$ dan $y = 5$

Operator	Example	Same As	Result
=	$x=y$		$x=5$
+=	$x+=y$	$x=x+y$	$x=15$
-=	$x-=y$	$x=x-y$	$x=5$
=	$x=y$	$x=x*y$	$x=50$
/=	$x/=y$	$x=x/y$	$x=2$
%=	$x\%=y$	$x=x\%y$	$x=0$

- Operator Perbandingan
Jika $x = 5$

Operator	Description	Example
==	is equal to	$x==8$ is false
===	is exactly equal to (value and	$x===5$ is true

	type)	x==="5" is false
!=	is not equal	x!=8 is true
>	is greater than	x>8 is false
<	is less than	x<8 is true
>=	is greater than or equal to	x>=8 is false
<=	is less than or equal to	x<=8 is true

- Operators Logical
Jika $x = 6$ dan $y = 3$

Operator	Description	Example
&&	and	(x < 10 && y > 1) is true
	or	(x==5 y==5) is false
!	not	!(x==y) is true

Operator + dapat pula digunakan untuk menggabungkan variabel string atau text.

```
txt1 = "Enjoy and";
txt2 = "cross the limit";
txt3 = txt1 + " " + txt2;
```

Setelah eksekusi diatas maka nilai dari txt3 yaitu "Enjoy and cross the limit".

- **Selection**

If...else if...else Statement

Syntax :

```
if (condition1)
{
    code to be executed if condition1 is true
}
else if (condition2)
{
    code to be executed if condition2 is true
}
else
{
    code to be executed if condition1 and condition2 are not true
}
```

Contoh :

```
<script type="text/javascript">
var d = new Date()
var time = d.getHours()
if (time<10)
{
    document.write("<b>Good morning</b>");
}
else if (time>10 && time<16)
```

```
{
  document.write("<b>Good day</b>");
}
else
{
  document.write("<b>Hello World!</b>");
}
</script>
```

Switch Case Statement

Syntax :

```
switch(n)
{
  case 1:
    execute code block 1
    break;
  case 2:
    execute code block 2
    break;
  default:
    code to be executed if n is
    different from case 1 and 2
}
```

Contoh :

```
<script type="text/javascript">
  var d = new Date();
  theDay = d.getDay(); // mengambil hari ini. return berupa integer yaitu 1 untuk
senin, 2 untuk selasa, dan sebagainya.
```

```
switch (theDay){
  case 1:
    document.write("Monday");
    break;
  case 2:
    document.write("Tuesday");
    break;
  case 3:
    document.write("Wednesday");
    break;
  case 4:
    document.write("Thursday");
    break;
  case 5:
    document.write("Friday");
    break;
  case 6:
```

```

        document.write("Saturday");
        break;
    default:
        document.write("Holiday");
}

```

Contoh output jika hari ini ialah hari ke 5 yaitu:



Friday

○ Module / Function

Fungsi adalah subprogram yang memungkinkan kita untuk menjalankan sekelompok instruksi dengan satu pemanggilan sederhana nama fungsi tersebut dari satu atau beberapa bagian di dalam badan suatu program. Bentuk subprogram yang kita sebut fungsi ini sangat umum di pakai di banyak bahasa pemrograman (tentu saja dengan cara yang sedikit berbeda antara satu dengan lainnya). Di lain pihak suatu fungsi, juga bisa memanggil dirinya sendiri, ini kita sebut dengan fungsi rekursif (akan tetapi jangan lupa untuk meletakkan kondisi khusus supaya fungsi bisa berhenti, kalau tidak bisa membahayakan kelangsungan program secara global).

➤ Deklarasi fungsi

Sebelum digunakan, suatu fungsi harus di definisikan terlebih dahulu karena untuk memanggil fungsi tersebut dari dalam program, navigator harus mengenalinya terlebih dahulu, dalam artian mengenali namanya, argumennya, dan instruksi di dalamnya. Pendefinisian satu fungsi dinamakan deklarasi. Sintaks pendeklarasian suatu fungsi adalah sebagai berikut :

```

function Nama_dari_Fungsi(argumen1, argumen2, ...){
    daftar instruksi atau blok instruksi
}

```

Catatan :

- ❖ Kata kunci function diikuti nama dari fungsi tersebut.
- ❖ Penamaan dari fungsi mempunyai aturan yang sama dengan penamaan dari variabel
- Nama harus dimulai oleh huruf.

- Nama dari fungsi bisa berisi huruf, angka atau karakter _ dan &, karakter kosong atau spasi tidak diperbolehkan.
- Nama fungsi seperti juga nama variabel adalah case sensitive, jadi memperhatikan huruf besar dan huruf kecilnya.
- ❖ Argumen adalah optional, boleh ada dan boleh tidak ada, jika argumen tidak ada maka tanda kurung harus tetap ditampilkan.

➤ **Pemanggilan fungsi**

Untuk mengeksekusi satu fungsi, kita cukup memanggil nama dari fungsi tersebut yang diikuti dengan kurung buka, argumen(jika ada) dan di tutup dengan kurung tutup.

```
Nama_dari_Fungsi() ;
```

Catatan :

1. Titik koma memberikan tanda kepada navigator tentang akhir dari blok instruksi yang berbeda.
2. Jika anda mendefinisikan suatu argumen untuk suatu fungsi, maka pada saat pemanggilannya anda harus memanggil fungsi tersebut lengkap dengan argumentnya.

Untuk membuat suatu script tidak langsung dijalankan saat halaman di load, kita dapat meletakkan script tersebut ke dalam function, sehingga function tersebut dapat dieksekusi melalui suatu event atau melalui pemanggilan function tersebut.

Function dapat didefinisikan di dalam <head> maupun <body> dalam dokumen. Namun, untuk memastikan bahwa function tersebut telah diload oleh browser sebelum pemanggilannya, lebih baik bila diletakkan di dalam <head>.

```
<html>
<head>
  <script type="text/javascript">
    function displaymessage()
    {
      alert("Hello World!");
    }
  </script>
</head>

<body>
  <form>
    <input type="button" value="Click me!" onclick="displaymessage()" >
  </form>
</body>
</html>
```

Berikut adalah contoh function yang mengembalikan hasil dua angka yang diterima (a dan b). Untuk mengembalikan nilai sebuah function harus menggunakan statement return.

```
function add( a , b )
```




```
{  
    x = a + b;  
    return x;  
}
```

Ketika function diatas dipanggil, maka anda harus memberikan dua buah parameter

```
hasil = add( 2, 3 );
```

Function add tersebut akan mengembalikan nilai 5, dan disimpan dalam variable hasil.

Web Design	
<i>Module 05 – JavaScript Intermediate</i>	
Last Update 07/11/2011	Revision 00

- **Module Description**

- Pada Bagian ini akan dijelaskan mengenai JavaScript Intermediate yang meliputi Looping, Random, Array and Object (Math, String, Date, etc).

- **Learning Outcomes**

- Mahasiswa dapat memahami Looping, Random, Array and Object dalam JavaScript.

- **Material**

- **Looping**

Ketika kita coding, maka kita akan menemukan kasus dimana kita harus mengulang kumpulan syntax yang telah kita buat sebelumnya. Daripada kita mengetik ulang syntax tersebut berkali – kali, lebih baik kita menggunakan looping / perulangan. Loop akan menjalankan sekumpulan kode yang berurutan berulang – ulang sesuai dengan jumlah perulangan yang ingin kita lakukan, atau selama syarat / kondisi perulangan itu masih benar.

Ada tiga jenis looping dalam JavaScript, antara lain :

- **For** : Melakukan perulangan pada sekumpulan kode dengan jumlah perulangan tertentu. Perulangan For biasa digunakan ketika kita mengetahui berapa kali script tersebut seharusnya dijalankan.

Syntax :

```
for (var=startvalue;var<=endvalue;var=var+increment)
{
    code to be executed
}
```

- **While** : Melakukan perulangan pada sekumpulan kode ketika keadaan tertentu bernilai *true*.

Syntax :

```
while (var<=endvalue)
{
    code to be executed
}
```

Catatan : Tanda <= dapat diganti dengan operator pembandingan lainnya.

- **Do..While**

Do..while loop merupakan variasi dari while loop. Perulangan ini akan menjalankan dahulu kumpulan kode tersebut satu kali, dan kemudian akan mengulang selama suatu kondisi tertentu bernilai *true*.

Syntax :

```
do
{
    code to be executed
}while (var<=endvalue);
```

Catatan : Tanda <= dapat diganti dengan operator pembandingan lainnya.

○ **Random**

Untuk melakukan random di javascript, kita menggunakan method random() yang ada pada class Math. Method random() mengembalikan sebuah angka yang nilainya acak antara 0 sampai 1.

```
Math.random();
```

Untuk memperoleh hasil random berupa bilangan bulat, maka kita perlu melakukan pembulatan terhadap hasil dari random. Pembulatan dilakukan dengan menggunakan method yang ada di class Math, yaitu floor(), ceil(), atau round()

- Math.floor() melakukan pembulatan ke bawah
- Math.ceil() melakukan pembulatan ke atas
- Math.round() melakukan pembulatan menjadi angka yang paling dekat dengan bilangan tersebut. Contoh :

13.7 -> 14

13.1 -> 13

Syntax :

```
Math.round(Math.random()*10) → random dari 0 sampai 10
Math.floor(Math.random()*10) → random dari 0 sampai 9
Math.ceil(Math.random()*10) -> random dari 1 samapi 10
```

○ **Array and Object**

Array object digunakan untuk menyimpan nilai – nilai dalam sebuah variabel tunggal. Untuk mendefinisikan object array menggunakan keyword new.
Contoh pendefinisian :

```
var myArray = new Array();
```

Ada tiga cara untuk mendefinisikan array :

○ Cara Pertama

```
var myCars=new Array(); // regular array (dapat diberikan argument berupa  
int untuk  
myCars[0]="Saab";    // mengontrol ukuran array  
myCars[1]="Volvo";  
myCars[2]="BMW";
```

○ Cara Kedua

```
var myCars=new Array("Saab","Volvo","BMW"); // condensed array
```

○ Cara Ketiga

```
var myCars=["Saab","Volvo","BMW"]; // literal array
```

Cara mengakses array :

```
myCars[0]="Opel";var x;
```

Method yang ada di Class Array :

Method	Description
<u>concat()</u>	Menggabungkan dua atau lebih array, dan mengembalikan hasil penggabungan dari array – array tersebut.
<u>join()</u>	Menggabungkan semua elemen yang ada dalam array tersebut ke dalam String.
<u>pop()</u>	Menghilangkan elemen terakhir dalam sebuah array, dan mengembalikan elemen tersebut
<u>push()</u>	Menambah elemen baru ke bagian akhir dari array, dan mengembalikan panjang array yang baru
<u>reverse()</u>	Memutar balikan urutan elemen dalam array.
<u>shift()</u>	Menghilangkan element pertama dari sebuah array, dan mengembalikan elemen tersebut
<u>slice()</u>	Memilih suatu bagian dari array, dan mengembalikannya dalam bentuk array baru
<u>sort()</u>	Mengurutkan elemen yang ada dalam array
<u>splice()</u>	Menambah atau menghapus elemen dari sebuah array
<u>toString()</u>	Mengubah sebuah array menjadi sebuah String, mengembalikan hasilnya
<u>unshift()</u>	Menambahkan sebuah elemen pada awal array, dan mengembalikan panjang array yang baru
<u>valueOf()</u>	Mengembalikan nilai primitive dari sebuah array

- *Boolean object* : digunakan untuk mengkonversi nilai non-boolean menjadi nilai Boolean (true / false).

Cara membuat sebuah object Boolean :

```
var objBoolean = new Boolean();
```

Berikut adalah cara membuat objek Boolean yang bernilai false :

```
var myBoolean=new Boolean();  
var myBoolean=new Boolean(0);  
var myBoolean=new Boolean(null);  
var myBoolean=new Boolean("");  
var myBoolean=new Boolean(false);  
var myBoolean=new Boolean(NaN);
```

Berikut adalah cara membuat objek Boolean yang bernilai true :

```
var myBoolean=new Boolean(1);  
var myBoolean=new Boolean(true);  
var myBoolean=new Boolean("true");  
var myBoolean=new Boolean("false");  
var myBoolean=new Boolean("Richard");
```

Method dari Boolean object :

Method	Description
<u>toString()</u>	Mengubah nilai Boolean menjadi sebuah String, dan mengembalikan hasilnya
<u>valueOf()</u>	Mengembalikan nilai primitive dari objek boolean

- **Date Object**

Date Object dibuat ketika kita ingin menggunakan tanggal dan waktu. Ada empat cara untuk menginisialisasi sebuah tanggal :

```
new Date() // tanggal dan waktu sekarang  
new Date(milliseconds) //millidetik sejak 01/01/1970  
new Date(dateString)  
new Date(year, month, day, hours, minutes, seconds, milliseconds)
```

Sekali objek dari Date dibuat, maka kita dapat mengakses dan mengubah detik, menit, jam, tanggal, bulan, tahun dari objek tersebut. Waktu yang digunakan adalah waktu local atau waktu UTC.

Contoh inisialisasi sebuah objek dari kelas Date

```
today = new Date()  
d1 = new Date("October 13, 1975 11:13:00")  
d2 = new Date(79,5,24)  
d3 = new Date(79,5,24,11,33,0)
```

Method	Description
<u>getDate()</u>	Mengembalikan hari dari bulan (dari 1-31)
<u>getDay()</u>	Mengembalikan hari dari minggu (dari 0-6)
<u>getFullYear()</u>	Mengembalikan tahun (empat angka)
<u>getHours()</u>	Mengembalikan jam (dari 0-23)
<u>getMilliseconds()</u>	Mengembalikan milidetik (dari 0-999)
<u>getMinutes()</u>	Mengembalikan menit (dari 0-59)
<u>getMonth()</u>	Mengembalikan bulan (dari 0-11)
<u>getSeconds()</u>	Mengembalikan detik (dari 0-59)
<u>getTime()</u>	Mengembalikan jumlah milidetik sejak 1 Januari 2011
<u>getTimezoneOffset()</u>	Mengembalikan perbedaan waktu antara GMT dengan waktu local, dalam menit
<u>getUTCDate()</u>	Mengembalikan hari dari bulan, berdasarkan waktu universal (dari 1-31)
<u>getUTCDay()</u>	Mengembalikan hari dari minggu, berdasarkan waktu universal (from 0-6)
<u>getUTCFullYear()</u>	Mengembalikan tahun, berdasarkan waktu universal (empat angka)
<u>getUTCHours()</u>	Mengembalikan jam, berdasarkan waktu universal (dari 0-23)
<u>getUTCMilliseconds()</u>	Mengembalikan milidetik, berdasarkan waktu universal (dari 0-999)
<u>getUTCMinutes()</u>	Mengembalikan menit, berdasarkan waktu universal (dari 0-59)
<u>getUTCMonth()</u>	Mengembalikan bulan, berdasarkan waktu universal (dari 0-11)
<u>getUTCSeconds()</u>	Mengembalikan detik, berdasarkan waktu universal (dari 0-59)
<u>getYear()</u>	Deprecated. Gunakan method <u>getFullYear()</u> sebagai pengganti
<u>parse()</u>	Mengubah sebuah tanggal berupa string dan mengembalikan jumlah milidetik sejak 1 Januari 1970
<u>setDate()</u>	Mengatur tanggal dari bulan (dari 1-31)
<u>setFullYear()</u>	Mengatur tahun (empat angka)
<u>setHours()</u>	Mengatur jam (dari 0-23)
<u>setMilliseconds()</u>	Mengatur milidetik (dari 0-999)
<u>setMinutes()</u>	Mengatur menit (dari 0-59)
<u>setMonth()</u>	Mengatur bulan (dari 0-11)
<u>setSeconds()</u>	Mengatur detik (dari 0-59)
<u>setTime()</u>	Mengatur tanggal dan waktu dengan menambahkan atau mengurangi sejumlah milidetik ke/dari 1 Januari 2011
<u>setUTCDate()</u>	Mengatur hari dari bulan, berdasarkan waktu universal (dari 1-31)

<u>setUTCFullYear()</u>	Mengatur tahun, berdasarkan waktu universal (empat angka)
<u>setUTCHours()</u>	Mengatur jam, berdasarkan waktu universal (dari 0-23)
<u>setUTCMilliseconds()</u>	Mengatur milidetik, berdasarkan waktu universal (dari 0-999)
<u>setUTCMinutes()</u>	Mengatur menit, berdasarkan waktu universal (dari 0-59)
<u>setUTCMonth()</u>	Mengatur bulan, berdasarkan waktu universal (from 0-11)
<u>setUTCSeconds()</u>	Mengatur detik, berdasarkan waktu universal (from 0-59)
setYear()	Deprecated. Gunakan method setFullYear() sebagai pengganti
<u>toDateString()</u>	Mengubah bagian dari sebuah objek Date menjadi sebuah String yang dapat dibaca
toGMTString()	Deprecated. Gunakan method toUTCString() sebagai pengganti
<u>toLocaleDateString()</u>	Mengembalikan bagian dari objek tanggal sebagai sebuah String, menggunakan locale convention
<u>toLocaleTimeString()</u>	Mengembalikan bagian waktu dari sebuah objek date sebagai sebuah String, menggunakan locale convention
<u>toLocaleString()</u>	Mengubah sebuah objek Date menjadi sebuah String, menggunakan locale convention
<u>toString()</u>	Mengubah sebuah objek Date menjadi sebuah String
<u>toTimeString()</u>	Mengubah bagian waktu dari objek tanggal menjadi sebuah String
<u>toUTCString()</u>	Converts a Date object to a string, according to universal time
<u>UTC()</u>	Mengembalikan jumlah milidetik dari sebuah String tanggal sejak 1 Januari 1970 berdasarkan waktu universal
<u>valueOf()</u>	Mengembalikan nilai primitive dari objek Date

- **Math Objects**

Math Object mempermudah kita untuk melakukan tugas – tugas perhitungan. Objek dari Math memiliki beberapa konstanta dan method.

Contoh konstanta yang bisa direferensikan dalam javascript :

Math.E
 Math.PI
 Math.SQRT2
 Math.SQRT1_2
 Math.LN2
 Math.LN10
 Math.LOG2E
 Math.LOG10E

Contoh penggunaan method dari class Math :

```
document.write(Math.round(4.7));
```

Method	Description
<u>abs(x)</u>	Mengembalikan nilai absolut dari x
<u>acos(x)</u>	Mengembalikan nilai arccosine dari x, dalam radians
<u>asin(x)</u>	Mengembalikan arcsine dari x, dalam radians
<u>atan(x)</u>	Mengembalikan arctangent dari x sebagai nilai numeric antara -PI/2 dan PI/2 radians
<u>atan2(y,x)</u>	Mengembalikan arctangent dari hasil bagi argument tersebut
<u>ceil(x)</u>	Mengembalikan x, melakukan pembulatan ke atas
<u>cos(x)</u>	Mengembalikan nilai cosines dari x (x dalam radians)
<u>exp(x)</u>	Mengembalikan nilai dari E^x
<u>floor(x)</u>	Mengembalikan x, melakukan pembulatan ke bawah
<u>log(x)</u>	Mengembalikan logaritma (base E) dari x
<u>max(x,y,z,...,n)</u>	Mengembalikan nilai yang merupakan nilai tertinggi
<u>min(x,y,z,...,n)</u>	Mengembalikan nilai yang merupakan nilai terendah
<u>pow(x,y)</u>	Mengembalikan nilai dari x pangkat y
<u>random()</u>	Mengembalikan sebuah angka acak antara 0 sampai 1
<u>round(x)</u>	Mengembalikan nilai dari pembulatan terdekat
<u>sin(x)</u>	Mengembalikan sinus of x (x dalam radians)
<u>sqrt(x)</u>	Mengembalikan nilai akar dari x
<u>tan(x)</u>	Mengembalikan tangent dari sebuah sudut

- **Number Objects**

Number object adalah sebuah wrapper object untuk nilai numeric primitive.

Syntax :


```
var num = new Number(value);
```

Note : Jika parameter gagal dikonversi menjadi angka, maka akan mengembalikan nilai NaN (Not-a-Number)

Method	Description
<u>toExponential(x)</u>	Mengubah sebuah bilangan menjadi notasi eksponensial
<u>toFixed(x)</u>	Mengatur format angka menjadi x angka di belakang koma
<u>toPrecision(x)</u>	Mengatur panjang format angka menjadi x
<u>toString()</u>	Mengubah objek Number menjadi sebuah String
<u>valueOf()</u>	Mengembalikan nilai primitive dari objek Number

- **String Objects**

Object String digunakan untuk memanipulasi penyimpanan tulisan

Syntax :

```
var txt = new String(string);
atau cara yang lebih mudah :
var txt = string;
```

Method	Description
<u>charAt()</u>	Mengembalikan karakter pada indeks tertentu
<u>charCodeAt()</u>	Mengembalikan Unicode dari karakter pada index tertentu
<u>concat()</u>	Menggabungkan dua buah string atau lebih, dan mengembalikan sebuah string baru hasil gabungan string – string tersebut
<u>fromCharCode()</u>	Mengubah nilai Unicode menjadi karakter
<u>indexOf()</u>	Mengembalikan posisi dari posisi pertama dari penemuan suatu nilai dari sebuah String
<u>lastIndexOf()</u>	Mengembalikan posisi dari posisi terakhir dari penemuan suatu nilai dari sebuah String
<u>match()</u>	Mencari kesamaan antara sebuah regular expression dengan sebuah string, dan mengembalikan kesamaan
<u>replace()</u>	Mencari kesamaan antara sebuah substring (atau regular expression) dan sebuah string, dan menggantikan substring yang sama dengan substring yang baru
<u>search()</u>	Mencari kesamaan antara regular expression dan sebuah string, dan mengembalikan posisi tempat kesamaan itu ditemukan.
<u>slice()</u>	Mengambil suatu bagian dari string, dan mengembalikan string baru

<u>split()</u>	Membagi sebuah string menjadi sebuah array yang berisi substring.
<u>substr()</u>	Mengambil sejumlah karakter dari sebuah string, dimulai dari posisi awal dan ditentukan jumlah karakter yang akan diambil., dan mengembalikan string baru
<u>substring()</u>	Mengambil sejumlah karakter dari sebuah string, dengan menentukan posisi awal pengambilan karakter dan posisi akhir pengambilan karakter. Karakter yang berada pada posisi terakhir tidak diikutserakan dalam pengembalian string baru.
<u>toLowerCase()</u>	Mengubah sebuah string menjadi huruf kecil semua
<u>toUpperCase()</u>	Mengubah sebuah string menjadi huruf besar semua
<u>valueOf()</u>	Mengembalikan nilai primitive dari objek string

Wrapper Method	Description
<u>anchor()</u>	Membuat sebuah anchor
<u>big()</u>	Menampilkan string dalam ukuran yang besar
<u>blink()</u>	Membuat String yang ditampilkan menjadi <i>blinking</i>
<u>bold()</u>	Menampilkan String dalam keadaan <i>bold</i> (tebal)
<u>fixed()</u>	Menampilkan string menggunakan fixed-pitch font
<u>fontcolor()</u>	Menampilkan string dengan warna tertentu
<u>fontsize()</u>	Menampilkan font dengan ukuran tertentu
<u>italics()</u>	Menampilkan string dalam keadaan <i>italic</i>
<u>link()</u>	Menampilkan string sebagai hyperlink
<u>small()</u>	Menampilkan string dalam ukuran yang kecil
<u>strike()</u>	Menampilkan string dalam keadaan dicoret
<u>sub()</u>	Menampilkan string dalam bentuk subscript text
<u>sup()</u>	Menampilkan string dalam bentuk superscript text

- **RegExp Object**

RegExp singkatan dari regular expression, adalah sebuah objek yang menjelaskan pola dari deretan karakter. Ketika kita ingin mencari suatu kata/kalimat, kita dapat menggunakan suatu pola yang mendeskripsikan apa yang kita cari.

Syntax :

```
var txt=new RegExp(pattern,modifiers);
atau cara yang lebih mudah :
var txt=/pattern/modifiers;
```

Modifiers digunakan untuk melakukan pencarian secara case-insensitive dan pencarian secara global :

Modifier	Description
i	Melakukan pencocokan secara case-insensitive
g	Melakukan pencocokan secara global (Mencocokkan semua kemungkinan, tidak berhenti setelah pencocokan pertama)
m	Melakukan pencocokan multiline

Brackets digunakan untuk mencari sekumpulan karakter :


Expression	Description
[abc]	Mencari karakter yang ada dalam brackets
[^abc]	Mencari karakter yang tidak ada dalam brackets
[0-9]	Mencari angka dari 0 sampai 9
[A-Z]	Mencari karakter dari uppercase A sampai uppercase Z
[a-z]	Mencari karakter dari lowercase a sampai lowercase z
[A-z]	Mencari karakter dari uppercase A sampai lowercase z
(red blue green)	Mencari karakter dengan beberapa alternatif

Metacharacters adalah karakter yang memiliki arti khusus.

Metacharacter	Description
.	Mencari karakter apapun selain new line dan line terminator
\w	Mencari sebuah karakter kata
\W	Mencari sebuah karakter non-kata
\d	Mencari sebuah karakter angka
\D	Mencari sebuah karakter non-angka
\s	Mencari sebuah karakter whitespace
\S	Mencari sebuah karakter non-whitespace
\b	Mencari sebuah kata pada awal atau akhir kalimat
\B	Mencari sebuah kata yang letaknya bukan pada awal atau akhir kalimat
\0	Mencari karakter null
\n	Mencari karakter enter
\f	Mencari karakter form feed

<code>\r</code>	Mencari sebuah carriage mengembalikan karakter
<code>\t</code>	Mencari karakter tab
<code>\v</code>	Mencari karakter tab vertikal
<code>\xxx</code>	Mencari karkter yang sesuai dengan bilangan oktal xxxx
<code>\xdd</code>	Mencari karakter yang sesuai dengan bilangan hexadecimal dd
<code>\uxxxx</code>	Mencari karakter unicode yang sesuai dengan bilangan hexadecimal xxx

Method	Description
<code><u>compile()</u></code>	Mengcompile sebuah regex
<code><u>exec()</u></code>	Mencoba kecocokan sebuah string, mengembalikan kecocokan pertama
<code><u>test()</u></code>	Mencoba kecocokan sebuah string, mengembalikan true atau false

Web Design	
Module 06 – Using JavaScript on Form Validation	
Last Update 07/11/2011	Revision 00

- **Module Description**

- Pada Bagian ini akan dijelaskan mengenai Form Validation menggunakan JavaScript.

- **Learning Outcomes**

- Mahasiswa dapat memahami pembuatan validasi dalam Form

- **Material**

Dalam melakukan validasi terhadap form terdapat 2 tipe, yaitu melakukan validasi secara *client side* maupun secara *server side*. Validasi *server side* contohnya validasi dengan PHP, JSP, ASP, dll, sedangkan validasi *client side* dapat dilakukan dengan *javascript*, dll.

- **Form Validation**

Javascript adalah bahasa scripting pada web. Javascript digunakan untuk menambah interaktivitas dengan halaman HTML.

Javascript dapat ditulis dalam suatu dokumen html melalui salah satu cara dibawah ini :

- **Inline**

Dituliskan dalam tag body sehingga ketika browser dijalankan Javascript langsung dijalankan.

Contoh :

```
<a href='javascript : alert(“Hello World”)’> Klik me </a>
```

- **Embed**

Diletakkan diantara tag `<script> ... </script>` dalam sebuah dokumen HTML. Di dalam tag `<script>` kita gunakan attribute “type=” untuk mendefinisikan bahasa scripting yang digunakan. Jadi, `<script type="text/javascript">` dan `</script>` memberitahukan dimana javascript dimulai dan diakhiri.

Contoh :

```

<script>
    function validasi()
    {
        var user=document.getElementById("txtUser").value;
        var pass=document.getElementById("txtPass").value;

        if(user=="")
        {
            alert("isi username");
        }
        else if(pass=="")
        {
            alert("isi password");
        }
        else
        {
            alert("berhasil");
        }
    }
</script>

```


```

<script>
    function emailValidation()
    {
        var email=document.getElementById("txtEmail").value;
        var apos=email.indexOf("@");
        var dotpos=email.lastIndexOf(".");
        if(email=="")
        {
            alert("isi email");
            return false;
        }
        else if (apos<1 || dotpos-apos<2)
        {
            alert(alerttxt);
            return false;
        }
        else
        {
            return true;
        }
    }
</script>

```

Cara – cara untuk memeriksa karakter dengan *javascript*:

- `indexOf("@")` : mencari posisi karakter @ pada suatu string.
- `charAt(2)` : mencari karakter pada sebuah string yang menempati index ke-2.
- `Form.text.value` : mengambil isi dari input "text" yang dimaksud
- `Form.text.value.length` : mencari panjang dari sebuah string yang dimaksud
- `Return` : mengembalikan nilai
- Membuat tanggal : `var tanggal = new Date();`
 - a. Mengambil tanggal : `tanggal.getDate();`
 - b. Mengambil hari : `tanggal.getDay();`
 - c. Mengambil bulan : `tanggal.getMonth();`
 - d. Mengambil tahun : `tanggal.getFullYear();`
 - e. Mengambil jam : `tanggal.getHours();`
 - f. Mengambil menit : `tanggal.getMinutes();`
 - g. Mengambil detik : `tanggal.getSeconds();`
- `Window.setTimeout ("function", 1000)` : untuk merefresh suatu function setelah 1000 seconds.

Web Design	
<i>Module 07 – VBScript Basic</i>	
Last Update 07/11/2011	Revision 00

- **Module Description**
 - Pada Bagian ini akan dijelaskan mengenai VBScript Basic seperti Operator, Data Type, Control Structure, Function dan Array.
- **Learning Outcomes**
 - Mahasiswa dapat memahami Operator, Data Type, Control Structure, Function dan Array dalam VBScript.
 - Mahasiswa dapat memahami dasar komponen web dan menerapkannya menggunakan bahasa HTML serta penggunaan client-side dengan VBScript.
- **Material**

Microsoft Visual Basic Script atau yang biasa disebut dengan VBScript merupakan bahasa pemrograman dari keluarga Visual Basic yang membawa skrip pada lingkungan yang luas, mencakup Web Client scripting pada browser Internet Explorer dan Web Server scripting pada Microsoft Information Server.

Untuk memasukkan vbscript ke dalam halaman HTML, kita menggunakan tag <script>. Di dalam tag <script> kita gunakan attribute “type=” untuk mendefinisikan bahasa scripting yang digunakan.

```
<script type="text/vbscript">
.....
</script>
```

- **Operator**
- **Data Type**

VBScript hanya memiliki satu tipe data yaitu **variant**. Variant adalah jenis tipe data spesial yang dapat memiliki isi dari berbagai jenis informasi yang berbeda – beda. Secara sederhana, variant dapat berisi baik jenis informasi numerik maupun string. Subtype yang dimiliki oleh variant :

Subtype	Deskripsi
Empty	Variant yang belum

	diinisialisasikan
Null	Variant yang dengan sengaja memiliki data tidak sah
Boolean	Berisi true atau false
Byte	Berisi nilai integer (0 s/d 255)
Integer	Berisi nilai integer (-32,768 s/d 32,767)
Currency	-922,337,203,685,477.5808 s/d 922,337,203,685,477.5807
Long	Berisi nilai integer dengan nilai jangkauan -2,147,438,648 s/d 2,147,438,647
Single	Memiliki ketepatan tunggal (single-precision), floating point dengan nilai jangkauan -3.402823E38 s/d -1.401298E-45 untuk nilai negatif; 1.401298E-45 s/d 3.402823E38 untuk nilai positif.
Double	Memiliki lebih dari satu ketepatan (double-precision)
Date/Time	Memiliki nilai yang menampilkan tanggal antara 1 Januari 100 s/d 31 Desember 9999
String	Memiliki panjang string yang dapat mencapai 2 milyar karakter
Object	Memiliki value sebagai object
Error	Memiliki value yang didapat dari nomor error

○ Control Structure

1. VBScript Conditional Statements

a. *If ... Then ... Elseif ... Else*

```

if payment="Cash" then
    msgbox "You are going to pay cash!"
elseif payment="Visa" then
    msgbox "You are going to pay with visa."
elseif payment="AmEx" then
    msgbox "You are going to pay with American Express."
else
    msgbox "Unknown method of payment."
end If

```

b. *Select Case*

```

if payment="Cash" then
    msgbox "You are going to pay cash!"
elseif payment="Visa" then

```

```
msgbox "You are going to pay with visa."
elseif payment="AmEx" then
    msgbox "You are going to pay with American Express."
else
    msgbox "Unknown method of payment."
end If
```

2. VBScript Looping Statement

a. *For Each ... Next Loop*

```
For i=1 to 10
    some code
Next
```

Untuk dapat keluar dari sebuah statement For .. Next menggunakan keyword Exit For.

b. *Do ... Loop*

```
Do While i>10
    some code
Loop
```

```
Do
    some code
Loop While i>10
```

```
Do Until i=10
    some code
Loop
```

```
Do
    some code
Loop Until i=10
```

c. *Exit do ... Loop*

Untuk dapat keluar dari sebuah statement Do...Loop menggunakan keyword Exit Do.

```
Do Until i=10
    i=i-1
    If i<10 Then Exit Do
Loop
```

o **Function and Array**

❖ **Deklarasi Function**

Function atau fungsi di VBScript sering disebut dengan procedures. Terdapat dua macam procedures, yaitu sub procedure dan function procedure. Sub procedure adalah sekumpulan statement yang dimulai dan diakhiri dengan statement Sub dan End Sub. Sub procedure tidak mengembalikan nilai.

```
Sub mysub()  
    some statements  
End Sub
```

Or

```
Sub mysub(argument1,argument2)  
    some statements  
End Sub
```

Function procedure adalah sekumpulan statement yang dimulai dan diakhiri dengan statement Function dan End Function. Function procedure dapat mengembalikan nilai dengan cara memberi nilai pada nama function tersebut.

```
Function myfunction()  
    some statements  
    myfunction = some value  
End Function
```

Or

```
Function myfunction(argument1,argument2)  
    some statements  
    myfunction=some value  
End Function
```

❖ Pemanggilan Sub atau Function

Untuk memanggil Sub procedure dapat menggunakan statement Call, seperti berikut :

```
Call MyProc(argument)
```

Atau, dapat pula tanpa menggunakan statement Call, seperti berikut :

```
MyProc argument
```

Untuk memanggil Function procedure, dapat dengan cara seperti berikut :

```
name = findname()
```

Disini, anda memanggil function “findname” yang mengembalikan nilai yang akan disimpan dalam variable name.

Atau, dengan cara seperti ini :

```
msgbox "Your name is " & findname()
```

❖ Array

Untuk mendeklarasikan variable menggunakan statement dim, public atau private. Contohnya adalah sebagai berikut :

```
dim nama  
nama = "vanilla"
```

Kita juga dapat langsung mendeklarasikan variable dengan menuliskan nama nya, seperti berikut ini :


```
nama = "vanilla"
```

Cara diatas kurang baik untuk digunakan karena jika anda salah menuliskan nama variable maka secara otomatis variable baru akan dibuat sehingga mengakibatkan kesalahan saat program dijalankan. Untuk mencegah hal ini, anda dapat menggunakan statement Option Explicit. Statement ini mengharuskan setiap variable yang digunakan dideklarasikan dengan dim, public atau private. Statement Option Explicit diletakkan di paling atas script. Contohnya adalah sebagai berikut :

```
option explicit  
dim nama  
nama = "vanilla"
```

Berikut adalah contoh mendeklarasikan array dan memasukkan data ke setiap element array :

```
dim names(2)  
names(0) = "Vanilla"  
names(1) = "Strawberry"  
names(2) = "Chocolate"
```

Web Design	
<i>Module 08 – CSS</i>	
Last Update 07/11/2011	Revision 00

1. Module Description

- Pada Bagian ini akan dijelaskan mengenai CSS (Cascading Style Sheet).

2. Learning Outcomes

- Mahasiswa dapat memahami konsep dasar pembuatan style dengan CSS pada Web.
- Mahasiswa dapat mengaplikasikan CSS yang sudah dibuat pada Web.

3. Material

➤ CSS

❖ Aturan dan style sheet

CSS kependekan dari **Cascading Style Sheets**. Dengan CSS, halaman dokumen HTML dapat ditampilkan dengan style output yang berbeda. CSS mengandung aturan-aturan. Sebuah *aturan* adalah sebuah pernyataan mengenai aspek yang akan memberikan style pada sebuah elemen atau lebih. Sebuah *style sheet* adalah kumpulan dari aturan-aturan yang yang diterapkan pada sebuah dokumen HTML.

Bagian-bagian dari sebuah aturan

Sebuah aturan terdiri dari dua bagian:

- Selektor (Selector) - bagian sebelum kurung kurawal
- Deklarasi (Declaration) - bagian yang terdapat ditengah kurung kurawal

Selektor adalah penghubung antara dokumen HTML dan style. Selektor mendefinisikan elemen apa yang terkena dampak dari deklarasi tersebut. **Deklarasi** tersebut adalah bagian dari aturan yang menentukan apa efek yang akan diberikan.

Bagian-bagian dari sebuah deklarasi

Sebuah deklarasi terdiri dari dua bagian yang dipisahkan oleh titik dua:

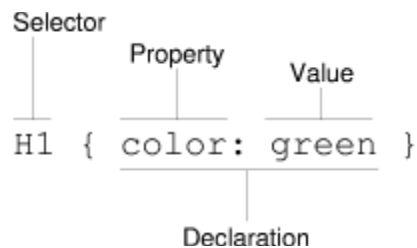
1. Properti - bagian yang ada sebelum titik dua
2. Nilai - bagian setelah titik dua

Properti adalah kualitas atau ciri-ciri yang dimiliki sesuatu hal.

Nilai adalah spesifikasi yang tepat dari properti.

Diagram dibawah menunjukkan semua komponen dari sebuah aturan. Tanda kurung kurawal ({ }) dan titik dua (:) memungkinkan browser membedakan antara selektor, properti, dan nilai.

selector {property: value}



❖ **Pengelompokan selektor dan aturan**

CSS mempunyai beberapa mekanisme untuk memperpendek style sheets dengan mengelompokkan selektor dan deklarasi. Sebagai contoh pertimbangkan tiga aturan ini:

```
H1 { font-weight: bold }
H2 { font-weight: bold }
H3 { font-weight: bold }
```

Seluruh tiga aturan ini mempunyai deklarasi yang sama, mereka menetapkan font menjadi bold. Karena semua deklarasi ini identik, kita dapat mengelompokkan selektor-selektor tadi dalam sebuah *daftar yang dipisahkan koma (comma-separated list)* dan hanya menuliskan deklarasinya satu kali, seperti ini:

```
H1, H2, H3 { font-style: bold }
```

Aturan ini akan menghasilkan hasil yang sama dengan tiga aturan diatas.

Sebuah selektor dapat memiliki lebih dari tiga deklarasi. Sebagai contoh, kita dapat menulis sebuah style sheet dengan dua aturan di bawah:

```
H1 { color: green }
H1 { text-align: center }
```

Dalam hal ini, kita menetapkan bahwa seluruh h1 menjadi hijau dan di posisikan ditengah kanvas. Tetapi kita dapat mendapatkan hasil yang sama lebih cepat dengan mengelompokkan deklarasi-deklarasi tadi pada selektor yang sama dalam sebuah *daftar yang dipisahkan dengan titik koma (semicolon-separated list)*, seperti ini:

```
H1 {
  color: green;
  text-align: center;
}
```

Seluruh deklarasi harus di taruh diantara kurung kurawal. Sebuah titiak koma memisahkan dan dapat juga muncul di akhir dari deklarasi terakhir.

❖ **Menampilkan informasi style pada dokumen**

Ada beberapa cara untuk menampilkan informasi style. Style dapat dispesifikasikan di dalam setiap element HTML, di dalam element <head> dari halaman HTML, atau dalam file CSS external.

Prioritas browser dalam menampilkan style dalam dokumen HTML, yaitu :

- **Browser default**
- **External style sheet**

- **Internal style sheet**
- **Inline style**

Jadi, inline style memiliki prioritas tertinggi, yang berarti akan mengoverride style yang dideklarasikan di dalam internal style sheet, di dalam external style sheet, atau browser default.

Browser default

- Warna teks dari link berubah menjadi biru tanpa ditentukan pada style sheet.
- Browser mengetahui bagaimana membentuk h1 tanpa harus ditentukan secara khusus.

External style sheet

Sebuah external stylesheet biasanya digunakan untuk banyak halaman. Dengan external stylesheet, anda dapat merubah tampilan seluruh website hanya dengan merubah sebuah isi file. Setiap halaman harus dihubungkan dengan style sheet menggunakan tag <link>. Tag <link> diletakkan di bagian <head> suatu halaman :

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css" />
</head>
```

Internal style sheet (di dalam tag <head>)

Sebuah internal style sheet digunakan oleh sebuah dokumen yang memiliki style tersendiri. Internal style sheet didefinisikan di bagian <head> menggunakan tag <style>, seperti berikut ini :

```
<head>
<style type="text/css">
hr {color: blue}
p {margin-left: 20px}
body {background-image: url("images/bear.jpg")}
</style>
</head>
```

Inline style (di dalam element HTML)

Untuk menggunakan inline style kita menggunakan attribute style dalam tag yang bersangkutan. Attribute style dapat mengandung property CSS. Contoh berikut menunjukkan bagaimana merubah warna dan left margin dari suatu paragraph.

```
<p style="color: blue; margin-left: 20px">
```

This is a paragraph

</p>

❖ CSS Comments

CSS juga memiliki bentuk komentarnya sendiri yang dapat digunakan dalam style sheet. Sebuah komentar CSS dimulai dengan "/*" dan diakhiri dengan "*/." Aturan CSS dalam sebuah komentar CSS comment tidak akan mempunyai pengaruh pada tampilan dokumen.

Contoh:

/* Ini komentar dalam CSS */

❖ Mengatur style link

Anda dapat mempergunakan CSS untuk mengatur style hypertext link, dengan style yang berbeda untuk link yang belum pernah anda akses, link yang pernah anda akses dan link yang akan kemudian anda akses serta link yang aktif. Anda bahkan dapat mengatur stylenya pada saat kursor mouse berada diatas daerah yang akan dilink. Hal ini dapat anda tuliskan dalam bentuk perintah berikut :

- Untuk style link yang belum diakses

:link {property: value}

Contoh:

```
:link {  
    Color : rgb(0, 0, 153)  
}
```

- Untuk style link yang telah diakses

:visited {property: value}

Contoh:

```
:visited {  
    Color : rgb(153, 0, 153)  
}
```

- Untuk style link ketika link diklik

:active {property: value}

Contoh:


```
:active {
    Color : rgb(255, 0, 102)
}
```

- Untuk style link ketika mouse diatasnya

```
:hover {property: value}
```

Contoh:

```
:hover {
    Color : rgb(0, 96, 255)
}
```

❖ CSS Properties

Berikut macam-macam property dalam CSS :

CSS Background Properties			
Property	Description	Possible Values	Examples
background-attachment	Declares the attachment of a background image (to scroll with the page content or be in a fixed position).	fixed scroll	div { background-attachment:fixed; } div { background-attachment:scroll; }
background-color	Declares the background color.	Valid color names, RGB values, hexadecimal notation.	div { background-color:green; } div { color:#00FF00; }
background-image	Declares the background image of an element.	URL values.	div { background-image:url(images/img.jpg); } body { background-image:url(img.jpg); }
background-position	Declares the position of a background image.	Lengths or percentages for the x and y positions, or one of the predefined values: top left top center top right center left center center center right	div { background-position:10px 50px; } div { background-position:bottom right; }

		bottom left bottom center bottom right	
background-repeat	Declares how and/or if a background image repeats.	repeat repeat-x repeat-y no-repeat	div { background-repeat:repeat-x; } div { background-repeat:no-repeat; }
background	Used as a shorthand property to set all the background properties at once.	Separate values by a space in the following order (those that are not defined will use inherited or default initial values): background-color background-image background-repeat background-attachment background-position	div { background:green url(image.jpg) no-repeat fixed center center; } div { background:url(image.jpg) fixed; }

CSS Border Properties			
Property	Description	Possible Values	Examples
border-top-color	Declares the color of the top border.	Valid color names, RGB values, hexadecimal notation, or the predefined value transparent.	div { border-top-color:green; } div { border-top-color:#00FF00; }
border-top-style	Declares the style of the top border.	none hidden dotted dashed solid double groove ridge inset outset	div { border-top-style:solid; } div { border-top-style:inset; }
border-top-width	Declares the width of the top border.	Lengths or the following predefined values: thin medium thick	div { border-top-width:2px; } div { border-top-width:thin; }
border-top	Used as a shorthand property to set all the border-top properties	Separate values by a space in the following order (those that are not defined will use	div { border-top:2px solid green; }

	at once.	inherited or default initial values): border-top-width border-top-style border-top-color	div { border-top:thick double #00FF00; }
border-right-color	Declares the color of the right border.	Valid color names, RGB values, hexadecimal notation, or the predefined value transparent.	div { border-right-color:green; } div { border-right-color:#00FF00; }
border-right-style	Declares the style of the right border.	none hidden dotted dashed solid double groove ridge inset outset	div { border-right-style:solid; } div { border-right-style:inset; }
border-right-width	Declares the width of the right border.	Lengths or the following predefined values: thin medium thick	div { border-right-width:2px; } div { border-right-width:thin; }
border-right	Used as a shorthand property to set all the border-right properties at once.	Separate values by a space in the following order (those that are not defined will use inherited or default initial values): border-right-width border-right-style border-right-color	div { border-right:2px solid green; } div { border-right:thick double #00FF00; }
border-bottom-color	Declares the color of the bottom border.	Valid color names, RGB values, hexadecimal notation, or the predefined value transparent.	div { border-bottom-color:green; } div { border-bottom-color:#00FF00; }
border-bottom-style	Declares the style of the bottom border.	none hidden dotted dashed solid double groove ridge inset outset	div { border-bottom-style:solid; } div { border-bottom-style:inset; }

border-bottom-width	Declares the width of the bottom border.	Lengths or the following predefined values: thin medium thick	div { border-bottom-width:2px; } div { border-bottom-width:thin; }
border-bottom	Used as a shorthand property to set all the border-bottom properties at once.	Separate values by a space in the following order (those that are not defined will use inherited or default initial values): border-bottom-width border-bottom-style border-bottom-color	div { border-bottom:2px solid green; } div { border-bottom:thick double #00FF00; }
border-left-color	Declares the color of the left border.	Valid color names, RGB values, hexadecimal notation, or the predefined value transparent.	div { border-left-color:green; } div { border-left-color:#00FF00; }
border-left-style	Declares the style of the left border.	none hidden dotted dashed solid double groove ridge inset outset	div { border-left-style:solid; } div { border-left-style:inset; }
border-left-width	Declares the width of the left border.	Lengths or the following predefined values: thin medium thick	div { border-left-width:2px; } div { border-left-width:thin; }
border-left	Used as a shorthand property to set all the border-left properties at once.	Separate values by a space in the following order (those that are not defined will use inherited or default initial values): border-left-width border-left-style border-left-color	div { border-left:2px solid green; } div { border-left:thick double #00FF00; }
border-color	Declares the border color of all four borders at once.	Valid color names, RGB values, hexadecimal notation, or the predefined value transparent. Separate the color for each border by a space, declaring the colors for the borders in	div { border-color:green red blue olive; } div { border-color:green; } div { border-color:green red; }

		<p>the following order: border-top-color border-right-color border-bottom-color border-left-color</p> <p>Undeclared values work as further shorthand notation. If only one color value is declared, all four borders will use that color. If two colors are declared, the top and bottom borders will use the first color while the right and left borders will use the second color. If three colors are declared, the top border will use the first color, the right and left borders will use the second color, and the bottom border will use the third color.</p>	<div> <div>div { border-color:green red blue; }</div> </div>
border-style	Declares the border style of all four borders at once.	<p>none hidden dotted dashed solid double groove ridge inset outset</p> <p>Undeclared values work as further shorthand notation. If only one style value is declared, all four borders will use that style. If two styles are declared, the top and bottom borders will use the first style while the right and left borders will use the second style. If three styles are declared, the top border will use the first style, the right and left borders will use the second style, and the bottom border will use the third style.</p>	<div> <div>div { border-style:solid dotted dashed double; }</div> <div>div { border-style:solid; }</div> <div>div { border-style:solid dotted; }</div> <div>div { border-style:solid dotted dashed; }</div> </div>
border-width	Declares the width of all four borders at	Lengths or the following predefined values:	<div> <div>div { border-width:1px 3px</div> </div>

	once.	thin medium thick Undeclared values work as further shorthand notation. If only one width value is declared, all four borders will use that width. If two widths are declared, the top and bottom borders will use the first width while the right and left borders will use the second width. If three widths are declared, the top border will use the first width, the right and left borders will use the second width, and the bottom border will use the third width.	5px 2px; } div { border-width:thin; } div { border-width:2px 4px; } div { border-width:2px 4px 5px; }
border	Used as a shorthand to declare the border properties when all four borders will have the same appearance.	Separate values by a space in the following order (those that are not defined will use inherited or default initial values): border-width border-style border-color	div { border:1px double green; } div { border:thin solid #00FF00; }

CSS Classification/Positioning Properties			
Property	Description	Possible Values	Examples
clear	Declares the side(s) of an element where no previous floating elements are allowed to be adjacent.	left right both none	div { clear:right; } div { clear:both; }
cursor	Declares the type of cursor to be displayed.	URL values, and the following predefined values: auto crosshair default pointer move e-resize	div { cursor:crosshair; } div { cursrsor:url(image.csr); } div { cursrsor:url(image.csr), pointer; }

		ne-resize nw-resize n-resize se-resize sw-resize s-resize w-resize text wait help	
display	Declares if/how the element displays.	none inline block list-item run-in compact marker table inline-table table-row-group table-header-group table-footer-group table-row table-column-group table-column table-cell table-caption	<div display:none;="" {="" }<br=""></div> div { display:inline; } div { display:marker; }
float	Declares whether a box should float to the left or right of other content, or whether it should not be floated at all.	left right none	<div float:left;="" {="" }<br=""></div> div { float:right; }
visibility	Declares the visibility of boxes generated by an element.	visible hidden collapse	<div visibility:visible;="" {="" }<br=""></div> div { visibility:hidden; }
top	Declares the distance that the top content edge of the element is offset below the top edge of its containing block. The position property of the element must also be set to a value other than static.	Lengths, percentages, and the predefined value auto.	<div top:15px;="" {="" }<br=""></div> div { top:2%; }
right	Declares the distance that the right content edge of the element is offset to the left of the right edge of its	Lengths, percentages, and the predefined value auto.	<div right:15px;="" {="" }<br=""></div> div { right:2%; }

	containing block. The position property of the element must also be set to a value other than static.		
bottom	Declares the distance that the bottom content edge of the element is offset above the bottom edge of its containing block. The position property of the element must also be set to a value other than static.	Lengths, percentages, and the predefined value auto.	div { bottom:15px; } div { bottom:2%; }
left	Declares the distance that the left content edge of the element is offset to the right of the left edge of its containing block. The position property of the element must also be set to a value other than static.	Lengths, percentages, and the predefined value auto.	div { left:15px; } div { left:2%; }
position	Declares the type of positioning of an element.	static relative absolute fixed	div { position:absolute; } div { position:relative; }
clip	Declares the shape of a clipped region when the value of the overflow property is set to a value other than visible.	Shapes, or the predefined value auto. In CSS 2, the only valid shape is a rectangle, using the following format to specify the offset lengths from each side of the box: rect(top, right, bottom, left)	div { clip:auto; } div { clip:rect(2px, 4px, 7px, 5px); }
overflow	Declares how content that overflows the element's box is handled.	visible hidden scroll auto	div { overflow:hidden; } div { overflow:scroll; }
vertical-align	Declares the vertical alignment of an inline-level element or a table cell.	Lengths, percentages, and the following predefined values: baseline sub super top	span { vertical-align:middle; } td { vertical-align:top; }

		text-top middle bottom text-bottom	
z-index	Declares the stack order of the element.	Integer values and the predefined value auto.	div { z-index:2; } div { z-index:auto; }

CSS Dimension Properties			
Property	Description	Possible Values	Examples
height	Declares the height of the element.	Lengths, percentages, and the predefined value auto.	div { height:200px; } div { height:50%; }
max-height	Declares the maximum height of the element.	Lengths, percentages, and the predefined value auto.	div { max-height:200px; } div { max-height:50%; }
min-height	Declares the minimum height of the element.	Lengths, percentages, and the predefined value auto.	div { min-height:200px; } div { min-height:50%; }
width	Declares the width of the element.	Lengths, percentages, and the predefined value auto.	div { width:500px; } div { width:75%; }
max-width	Declares the maximum width of the element.	Lengths, percentages, and the predefined value auto.	div { max-width:500px; } div { max-width:75%; }
min-width	Declares the minimum width of the element.	Lengths, percentages, and the predefined value auto.	div { min-width:500px; } div { min-width:75%; }

CSS Font Properties			
Property	Description	Possible Values	Examples
font-family	Declares the name of the font to be used. Previously set in HTML via the <i>face</i> attribute in a	Valid font family names or generic family names, i.e. <i>Arial</i> , <i>Verdana</i> , <i>sans-serif</i> , <i>"Times New Roman"</i> ,	div { font-family:Arial; } div { font-family:Arial,

	 tag.	<i>Times, serif</i> , etc. Font family names can be separated by a comma in the same declaration to allow additional and/or generic family names to be used if the preferred font is unable to be displayed.	Helvetica, sans-serif; }
font-size	Declares the size of the font. Previously set in HTML via the <i>size</i> attribute in a tag.	Lengths (number and unit type— i.e. <i>1em</i> , <i>12pt</i> , <i>10px</i> , <i>80%</i>) or one of the following predefined values: xx-small x-small small medium large x-large xx-large smaller larger	div { font-size:70%; } div { font-size:0.85em; } div { font-size:medium; }
font-size-adjust	Limited browser support: Was part of CSS 2, but not in CSS 2.1. This property may return in CSS 3. Declares the <i>aspect value</i> (font size divided by x-height).	Numeric value	div { font-size-adjust:0.54; } div { font-size-adjust:0.46; }
font-stretch	Limited browser support: Was part of CSS 2, but not in CSS 2.1. This property may return in CSS 3. Declares the stretch of the font face.	normal wider narrower ultra-condensed extra-condensed condensed semi-condensed semi-expanded expanded extra-expanded ultra-expanded	div { font-stretch:narrower; } div { font-stretch:ultra-expanded; }
font-style	Declares the font style.	normal italic oblique	div { font-style:italic; } div { font-style:oblique; }
font-variant	Declares the font variant.	normal small-caps	div { font-variant:normal; } div { font-

			variant:small-caps; }
font-weight	Declares the font weight (lightness or boldness)	normal bold bolder lighter 100 200 300 400 500 600 700 800 900	div { font-weight:bolder; } div { font-weight:200; }
font	Used as a shorthand property to declare all of the font properties at once (except font-size-adjust and font-stretch).	Separate values by a space in the following order (those that are not defined will use inherited or default initial values): font-style font-variant font-weight font-size line-height font-family	div { font:italic small-caps bold 1em 1.2em Arial } div { font:bold 0.8em Verdana }

CSS Generated Content Properties			
Property	Description	Possible Values	Examples
content	Generates content in the document in conjunction with the :before and :after pseudo-elements.	String values, URL values, and predefined value formats: counter(name) counter(name, list-style-type) counters(name, string) counters(name, string, list-style-type) attr(X) open-quote close-quote	div:before { content:"some text"; } div:after { content:url(page2.html); }

		no-open-quote no-close-quote	
counter-increment	Declares the counter increment for each instance of a selector.	Integers and the predefined value none.	
counter-reset	Declares the value the counter is set to on each instance of a selector.	Integers and the predefined value none.	
quotes	Declares the type of quotation marks to use for quotations and embedded quotations.	String values and the predefined value none.	

CSS List Properties			
Property	Description	Possible Values	Examples
list-style-type	Declares the type of list marker used.	disc circle square decimal decimal-leading-zero lower-roman upper-roman lower-alpha upper-alpha lower-greek lower-latin upper-latin hebrew armenian georgian cjk-ideographic hiragana katakana hiragana-iroha katakana-iroha	<pre>ol { list-style-type:upper-roman; } ul { list-style-type:square; }</pre>
list-style-position	Declares the position of the list marker.	inside outside	<pre>ol { list-style-position:inside; } ul { list-style-position:outside; }</pre>
list-style-image	Declares an image to be used as the list marker.	URL values.	<pre>ul { list-style-image:url(image.jpg); }</pre>
list-style	Shorthand property to declare three list properties at once.	Separate values by a space in the following order (those that are	<pre>ul { list-style:disc inside url(image.gif); } ol { list-style:upper-</pre>

		not defined will use inherited or default initial values): list-style-type list-style-position list-style-image	roman outside; }
marker-offset	Declares the marker offset for elements with a value of marker set for the display property.	Lengths and the predefined value auto.	li:before { display:marker; marker-offset:5px; }

CSS Margin Properties			
Property	Description	Possible Values	Examples
margin-top	Declares the top margin for the element.	Lengths, percentages, and the predefined value auto.	div { margin-top:5px; } div { margin-top:15%; }
margin-right	Declares the right margin for the element.	Lengths, percentages, and the predefined value auto.	div { margin-right:5px; } div { margin-right:15%; }
margin-bottom	Declares the bottom margin for the element.	Lengths, percentages, and the predefined value auto.	div { margin-bottom:5px; } div { margin-bottom:15%; }
margin-left	Declares the left margin for the element.	Lengths, percentages, and the predefined value auto.	div { margin-left:5px; } div { margin-left:15%; }
margin	Shorthand property used to declare all the margin properties at once.	Separate values by a space in the following order (those that are not defined will use inherited or default initial values): margin-top margin-right margin-bottom margin-left Undeclared values work as further shorthand notation. If only one length value is declared, all four margins will use that length. If two lengths are declared, the top and bottom	div { margin:5px 12px 4px 7px; } div { margin:5px; } div { margin:5px 10px; } div { margin:5px 7px 4px; }

		margins will use the first length while the right and left margins will use the second length. If three lengths are declared, the top margin will use the first length, the right and left margins will use the second length, and the bottom margin will use the third length.	
--	--	---	--

CSS Outline Properties			
Property	Description	Possible Values	Examples
outline-color	Declares the outline color.	Valid color names, RGB values, hexadecimal notation.	div { outline-color:green; } div { outline-color:#00FF00; }
outline-style	Declares the style of the outline.	none dotted dashed solid double groove ridge inset outset	div { outline-style:solid; } div { outline-style:inset; }
outline-width	Declares the width of the outline.	Lengths or the following predefined values: thin medium thick	div { outline-width:2px; } div { outline-width:thin; }
outline	Used as a shorthand property to set all the background properties at once.	Separate values by a space in the following order (those that are not defined will use inherited or default initial values): outline-color outline-style outline-width	div { outline:green solid 2px; } div { outline:#00FF00 double thick; }

CSS Padding Properties			
Property	Description	Possible Values	Examples
padding-top	Declares the top	Lengths, percentages, and the	div { padding-

	padding for the element.	predefined value auto.	top:5px; } div { padding-top:15%; }
padding-right	Declares the right padding for the element.	Lengths, percentages, and the predefined value auto.	div { padding-right:5px; } div { padding-right:15%; }
padding-bottom	Declares the bottom padding for the element.	Lengths, percentages, and the predefined value auto.	div { padding-bottom:5px; } div { padding-bottom:15%; }
padding-left	Declares the left padding for the element.	Lengths, percentages, and the predefined value auto.	div { padding-left:5px; } div { padding-left:15%; }
padding	Shorthand property used to declare all the margin properties at once.	<p>Separate values by a space in the following order (those that are not defined will use inherited or default initial values):</p> <p>padding-top padding-right padding-bottom padding-left</p> <p>Undeclared values work as further shorthand notation. If only one length value is declared, all four sides will use that length. If two lengths are declared, the top and bottom sides will use the first length while the right and left sides will use the second length. If three lengths are declared, the top side will use the first length, the right and left sides will use the second length, and the bottom side will use the third length.</p>	<div>div { padding:5px 12px 4px 7px; }</div> <div>div { padding:5px; }</div> <div>div { padding:5px 10px; }</div> <div>div { padding:5px 7px 4px; }</div>

Property	Description	Possible Values	Examples
marks	Declares the type of marks to display outside the page box.	crop cross	@page { marks:crop; }
orphans	Declares the minimum number of lines of a paragraph that must be left at the bottom of a page.	Integers	@page { orphans:2; }
page	Declares the type of page where an element should be displayed.	Identifiers	
page-break-after	Declares a page break.	auto always avoid left right	
page-break-before	Declares a page break.	auto always avoid left right	
page-break-inside	Declares a page break.	auto avoid	
size	Declares the size and orientation of a page box.	Lengths, and the following predefined values: auto landscape portrait	
widows	Declares the minimum number of lines of a paragraph that must be left at the top of a page.	Integers	@page { widows:2; }

CSS Table Properties			
Property	Description	Possible Values	Examples
border-collapse	Declares the way borders are displayed.	collapse separate	table { border-collapse:collapse; } table { border-collapse:separate; }
border-spacing	Declares the distance	Lengths for the horizontal	table { border-

	separating borders (if border-collapse is separate).	and vertical spacing, separated by a space. If one length is value is declared, that length is used for both the horizontal and vertical spacing. If two lengths are declared, the first one is used for horizontal spacing and the second one is used for vertical spacing.	spacing:5px; } table { border-spacing:5px 10px; }
caption-side	Declares where the table caption is displayed in relation to the table.	top bottom left right	caption { caption-side:top; } caption { caption-side:right; }
empty-cells	Declares the way empty cells are displayed (if border-collapse is separate).	show hide	table { empty-cells:show; } table { empty-cells:hide; }
table-layout	Declares the type of table layout.	auto fixed	table { table-layout:auto; } table { table-layout:fixed; }

CSS Text Properties			
Property	Description	Possible Values	Examples
color	Declares the color of the text.	Valid color names, RGB values, hexadecimal notation. The predefined color names are: aqua black blue fuchsia gray green lime maroon navy olive purple red silver teal	div { color:green; } div { color:rgb(0,255,0); } div { color:#00FF00; }

		white yellow	
direction	Declares the reading direction of the text.	ltr rtl ltr = left-to-right rtl = right-to-left	div { direction:ltr; } div { direction:rtl; }
line-height	Declares the distance between lines.	Numbers, percentages, lengths, and the predefined value of normal.	div { line-height:normal; } div { line-height:2em; } div { line-height:125%; }
letter-spacing	Declares the amount of space between text characters.	A length (in addition to the default space) or the predefined value of normal.	div { letter-spacing:normal; } div { letter-spacing:5px; } div { letter-spacing:-1px; }
text-align	Declares the horizontal alignment of inline content.	left right center justify If used on a set of table cells, this property can be given a string value to which the text of each row of the column will be aligned.	div { text-align:center; } div { text-align:right; } td { text-align:"."; }
text-decoration	Declares the text decoration.	none underline overline line-through blink	div { text-decoration:none; } div { text-decoration:underline; }
text-indent	Declares the indentation of the first line of text.	Lengths and percentages.	div { text-indent:12px; } div { text-indent:2%; }
text-shadow	Declares shadow effects on the text.	A list containing a color followed by numeric values (separated by spaces) that specify: The color for the shadow effect Horizontal distance to the right of the text Vertical distance below the text Blur radius	div { text-shadow:green 2px 2px 7px; } div { text-shadow:olive -3px -4px 5px; }

text-transform	Declares the capitalization effects on the letters in the text.	none capitalize uppercase lowercase	div { text-transform:uppercase; } div { text-transform:lowercase; }
unicode-bidi	Declares values relating to bidirectional text. May be used in conjunction with the the <i>direction</i> property.	normal embed bidi-override	div { unicode-bidi:embed; } div { unicode-bidi:bidi-override; }
white-space	Declares how white space is handled in an element.	normal pre nowrap	div { white-space:pre; } div { white-space:nowrap; }
word-spacing	Declares the space between words in the text.	A length (in addition to the default space) or the predefined value of normal.	div { word-spacing:normal; } div { word-spacing:1.5em; }

Other CSS Properties			
Property	Description	Possible Values	Examples
azimuth	Declares the angle that sound travels to the listener.	Angle values in degrees (deg), or one of the following predefined values: left-side far-left left center-left center center-right right far-right right-side behind leftwards rightwards	div { azimuth:90deg; } div { azimuth:behind; }
cue-after	Declares an audio cue to play after an element.	URL values and the predefined value none.	div { cue-after:url(sound.wav); } div { cue-after:none; }
cue-before	Declares an audio cue to play before an element.	URL values and the predefined value none.	div { cue-before:url(sound.wav); } div { cue-before:none; }

cue	Shorthand property to set both cue values at once.	URL values and the predefined value none. Separate the values by a space in the following order: cue-before cue-after If only one cue value is declared, it is used for both before and after.	div { cue:url(sound.wav) url(sound2.wav); } div { cue:url(sound.wav); }
elevation	Declares the elevation of a sound.	Angle values in degrees (deg), or one of the following predefined values: below level above higher lower	div { elevation:30deg; } div { elevation:higher; }
pause-after	Declares the amount of time to pause after an element.	Time in milliseconds (ms) or percentages.	div { pause-after:100ms; } div { pause-after:20%; }
pause-before	Declares the amount of time to pause before an element.	Time in milliseconds (ms) or percentages.	div { pause-before:100ms; } div { pause-before:20%; }
pause	Shorthand property to set both pause values at once.	Separate the values by a space in the following order: pause-before pause-after If only one pause value is declared, it is used for both before and after.	div { pause:200ms 100ms; } div { pause:100ms; }
pitch	Declares the average speaking pitch of a voice.	Frequencies in hertz (Hz) or the following predefined values: x-low low medium high x-high	div { pitch:120Hz; } div { pitch:high; }
pitch-range	Declares a change in the pitch range of a voice.	Number values between 0 and 100 (lower values indicate a flat voice while	div { pitch-range:50; } div { pitch-range:99; }

		higher values indicate an animated voice).	
play-during	Declares a background sound to be played while the current element is spoken.	URL value, followed by one or more of the following keywords, separated by spaces: mix repeat Alternatley, one of the following keywords: auto none	div { play-during:url(music.wav); } div { play-during:url(music.wav) repeat; } div { play-during:none; }
richness	Declares the richness of the voice in spoken text.	Numeric values between 0 and 100 (lower values have less richness and higher values have more richness).	div { richness:50; } div { richness:0; }
speak	Declares if/how text is spoken.	normal none spell-out	div { speak:none; } div { speak:spell-out; }
speak-header	Declares how often table header cells are spoken.	once always	th { speak-header:once; } th { speak-header:always; }
speak-numeral	Declares how numerals are spoken.	digits continuous	div { speak-numeral:digits; } div { speak-numeral:continuous; }
speak-punctuation	Declares how punctuation is spoken.	code none	div { speak-punctuation:code; } div { speak-punctuation:none; }
speech-rate	Declares the speech rate of spoken text.	A number indicating the number of words per minute, or one of the following predefined values: x-slow slow medium fast x-fast faster slower	div { speech-rate:50; } div { speech-rate:medium; }
stress	Declares the stress of the voice on spoken	Numeric values between 0 and 100	div { stress:50; } div { stress:0; }

	text.	(lower values have less stress and higher values have more stress).	
voice-family	Declares the voice family of spoken text.	Generic or specific voice family names.	
volume	Declares the median volume.	Numbers between 0 and 100, percentages, or one of the following predefined values: silent x-soft soft medium loud x-loud	div { volume:50; } div { volume:silent; }

➤ Event Handler

Kita dapat membuat halaman dokumen HTML dapat ditampilkan dengan style output yang berbeda dengan ***Event Handler***.

Event Handler adalah property yang sudah didefinisikan oleh JavaScript dari sebuah objek yang digunakan untuk menghandel sebuah event dalam halaman web.

Event adalah sesuatu yang terjadi ketika user melakukan aksi pada halaman tertentu seperti mengklik tombol mouse pada halaman web.

Event Handler dapat dilihat pada tabel di bawah ini:

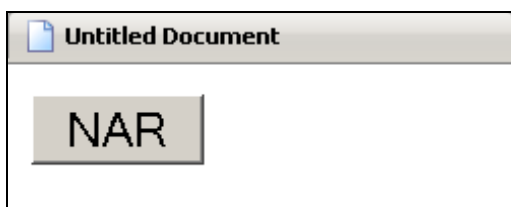
Event	Event Handler	Event Trigger
Abort	onabort	Sebuah gambar distop saat sebelum selesai loading.
Blur	onblur	User berpindah fokus dari satu elemen.
Change	onchange	User merubah isi dari sebuah elemen.
Click	onclick	User mengklik sebuah elemen
ContextMenu	oncontextmenu	User membuka context menu
Copy	oncopy	User menggunakan perintah copy pada bagian dari halaman web
Cut	oncut	User menggunakan perintah cut pada bagian dari halaman web
Dblclick	ondblclick	User mendobel klik mouse
Error	onerror	Browser user mendapatkan error JavaScript

Focus	onfocus	User memberikan fokus pada elemen
Keydown	onkeydown	User menekan key pada keyboard
Keypress	onkeypress	User menekan key pada keyboard lalu melepaskannya
Keyup	onkeyup	User melepaskan key pada keyboard
Load	onload	Halaman web selesai loading
Mousedown	onmousedown	User menekan tombol mouse
Mousemove	onmousemove	User memindahkan mouse
Mouseout	onmouseout	User memindahkan mouse dari sebuah elemen
Mouseover	onmouseover	User memindahkan mouse diatas elemen
Mouseup	onmouseup	User melepas tombol mouse
Paste	onpaste	User menggunakan perintah paste pada sebagian halaman web
Reset	onreset	User mereset form pada sebuah halaman web
Resize	onresize	Sebuah window di resize
Scroll	onscroll	User menscroll sebuah area yang bisa discroll
Select	onselect	User membuat seleksi
Submit	onsubmit	User mensubmit form dalam sebuah halaman web
Unload	onunload	User meninggalkan halaman current

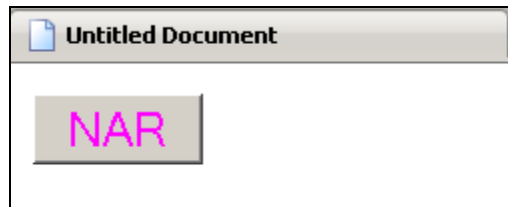
Contoh:

```
<input type="button" value="NAR" onclick="this.style.color='#FF00FF';"/>
```

Button sebelum di klik



Button setelah di klik



Keterangan:

Saat event click dijalankan maka event handler onclick yang berfungsi, sehingga dengan menuliskan `this.style.color='#FF00FF'`; akan mengubah warna tulisan button tersebut.

Perlu diingat bahwa, satu komponen dapat memiliki lebih dari satu event handler dan satu event handler dapat memiliki lebih dari satu aturan style.

Contoh:

```
<label  
onclick="this.style.color='#FF00FF';this.style.opacity=0.25;"  
onmouseover="this.style.cursor='wait';">  
    Assistant Recruitment 11-1  
</label>
```

Sebelum event click dan mouseover



Setelah event click dan mouseover



Keterangan:

Saat event click dijalankan maka event handler onclick yang berfungsi, yaitu mengubah warna tulisan dan opacity. Sedangkan saat kita memindahkan mouse diatas elemen (event mouseover) akan mengubah jenis kursor menjadi tipe wait.

Selain dengan memberikan style secara langsung, kita juga dapat mengganti jenis **class** CSSnya.

Contoh:

```
<html>  
<head>
```

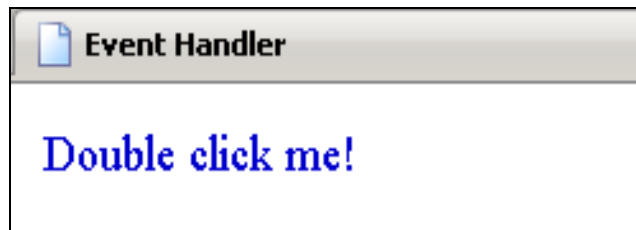


```

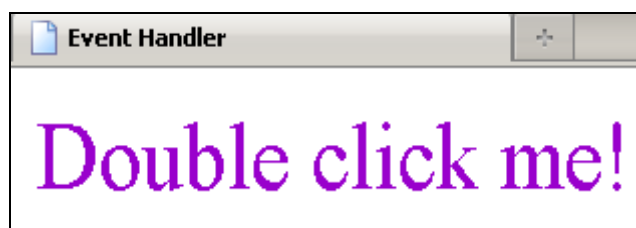
<title>Event Handler</title>
<style>
.before
{
    font-size:10px;
    color:#0000CC;
}
.after
{
    font-size:24px;
    color:#9900CC;
}
</style>
</head>
<body>
    <div class="before" ondblclick="this.className='after';">
        Double click me!
    </div>
</body>
</html>

```

Sebelum event double click




Setelah event double click



Keterangan:

Saat event dblclick dijalankan maka event handler ondblclick yang berfungsi, yaitu mengubah class dari div tersebut dari *before* menjadi *after*.

Web Design	
<i>Module 09 – Data Binding</i>	
Last Update 07/11/2011	Revision 00

- **Module Description**
 - Pada Bagian ini akan dijelaskan mengenai Data Binding dimana Data akan di-binding ke dalam bentuk Table. Data yang di-binding pun dapat di-sorting.
- **Learning Outcomes**
 - Mahasiswa dapat mengambil data dari source luar untuk ditampilkan ke dalam table pada Web.
 - Mahasiswa dapat melakukan sorting atau pengurutan data.
- **Material**

Data Binding adalah pengikatan *element HTML* pada data yang disediakan atau ditampung dalam **Data Source Object (DSO)**. **DSO** tersebut berbentuk *file* yang disimpan dalam *harddisk* komputer. Data – data yang berada pada **DSO** akan ditampilkan melalui *element HTML* yang nantinya akan dibaca oleh *browser*.

○ **Data Binding to Table**

Untuk melakukan **data binding**, diperlukan **tag object** serta parameter yang digunakan sebagai *config*-nya seperti di bawah ini :

```
<object id="[id dari object]"
  classid="CLSID:333C7BC4-460F-11D0-BC04-0080C7055A83">
  <param name="DataURL" value="[nama file]" />
  <param name="UseHeader" value="true" />
  <param name="TextQualifier" value="" />
  <param name="FieldDelim" value="[karakter pembatas field]" />
</object>
```

Tag **<param/>** digunakan untuk membuat paramater yang akan digunakan dalam pembuatan *object data binding*.

1. **DataURL**, untuk menentukan nama file yang ingin di *binding*.
2. **UseHeader**, bernilai **TRUE** jika pada binding menggunakan *header field* pada *file*.
3. **TextQualifier**, bernilai karakter pemisah untuk setiap text yang ingin didapat.
4. **FieldDelim**, bernilai karakter pemisah antar *field*.

Contoh :

Format data yang di *binding* :

```
@Cake Name@#@Size@#@Price@  
@Chiffon Cokelat@#@L@#@90000@  
@Chiffon Putih@#@S@#@90000@
```

Maka pembuatan *object data binding* sebagai berikut :

```
<object id="Data" classid="CLSID:333C7BC4-460F-11D0-BC04-  
0080C7055A83">  
  <param name="DataURL" value="data.txt" />  
  <param name="UseHeader" value="TRUE" />  
  <param name="TextQualifier" value="@" />  
  <param name="FieldDelim" value="#" />  
</object>
```

Kemudian dari data tersebut akan dibinding pada *tag table*. Sehingga kita perlu membentuk *tag table* yang atribut **datasrc** bernilai id dari object yang telah dibuat.

```
<table datasrc="[id dari object]">  
</table>
```

Untuk menampilkan data dengan tag **span** yang properti **DATAFLD** bernilai sama dengan nama field dari data yang ingin di tampilkan.

```
<span datafld="[nama field]"></span>
```

○ **Sorting Data**

Untuk membuat fasilitas *sorting* dengan menggunakan *javascript* seperti di bawah ini :

```
[nama object].Sort="[nama field]"; → untuk sorting secara ascending  
[nama object].Sort="[-nama field]"; → untuk sorting secara descending  
[nama object].Reset();
```

Untuk fasilitas *moving* (*first, prev, next, last*) dengan menggunakan *javascript* juga.

Membuat variabel dari **recordset** dari tag **object**.

Membuat validasi untuk *first, previous, next* dan *last*.